The University of Plymouth

A Collision Avoidance System for Autonomous Underwater Vehicles

Chiew Seon Tan

A Thesis Submitted in Partial Fulfilment for the Degree
of
Doctor of Philosophy

in the
Faculty of Technology

School of Engineering

In Colloboration with J&S Marine Ltd

January 2006

To my beloved parents and my girlfriend...

Abstract

The work in this thesis is concerned with the development of a novel and practical collision avoidance system for autonomous underwater vehicles (AUVs). Synergistically, advanced stochastic motion planning methods, dynamics quantisation approaches, multivariable tracking controller designs, sonar data processing and workspace representation, are combined to enhance significantly the survivability of modern AUVs.

The recent proliferation of autonomous AUV deployments for various missions such as seafloor surveying, scientific data gathering and mine hunting has demanded a substantial increase in vehicle autonomy. One matching requirement of such missions is to allow all the AUV to navigate safely in a dynamic and unstructured environment. Therefore, it is vital that a robust and effective collision avoidance system should be forthcoming in order to preserve the structural integrity of the vehicle whilst simultaneously increasing its autonomy.

This thesis not only provides a holistic framework but also an arsenal of computational techniques in the design of a collision avoidance system for AUVs. The design of an obstacle avoidance system is first addressed. The core paradigm is the application of the Rapidly-exploring Random Tree (RRT) algorithm and the newly developed version for use as a motion planning tool. Later, this technique is merged with the Manoeuvre Automaton (MA) representation to address the inherent disadvantages of the RRT. A novel multi-node version which can also address time varying final state is suggested. Clearly, the reference trajectory generated by the aforementioned embedded planner must be tracked. Hence, the feasibility of employing the linear quadratic regulator (LQG) and the nonlinear kinematic based state-dependent Ricatti equation (SDRE) controller as trajectory trackers are explored.

The obstacle detection module, which comprises of sonar processing and workspace representation submodules, is developed and tested on actual sonar data acquired in a sea-trial via a prototype forward looking sonar (AT500). The sonar processing techniques applied are fundamentally derived from the image processing perspective. Likewise, a novel occupancy grid using nonlinear function is proposed for the workspace representation of the AUV. Results are presented that demonstrate the ability of an AUV to navigate a complex environment.

To the author's knowledge, it is the first time the above newly developed methodologies have been applied to an AUV collision avoidance system, and, therefore, it is considered that the work constitutes a contribution of knowledge in this area of work.

Acknowledgements

Honestly, to state that my PhD life was full of enjoyment would be an obvious lie. Just as life, there have been ups and downs throughout the whole journey. Nevertheless, I am always grateful to those people that have accompanied me through the hard times. Without them, the journey would definitely be tougher. It is true that threading through my PhD journey, I have been touched by people from all walks of life. In addition, I am not ruling out any divine intervention.

From the formative stages of this thesis, to the final draft, I owe an immense debt of gratitude to my Director of Studies, Professor Robert Sutton, who provided me with the opportunity to pursue my research in this field. His sound advice and generosity have been much appreciated while his support and guidance have been crucial in my research. Secondly, gratitude goes to Dr. John Chudley whose diplomatic and charming personality never fail to motivate me.

Thanks goes to the University of Plymouth staff: Dr. Wasif Naeem, Dr. Dedy Loebis, Tao Xu, and Chirag Rathi for the many intellectual discussions, Chen Jun Tjhai for his C programming skills, Mike Sloman, the trouble-shooter, Dr. John Summerscales for his understanding and Frank Knot. Additionally, Dr. Emillio Frazzoli from the UCLA for his enlightening explanations pertaining to the "Manoeuvre Automaton" approach and Dr. Tena R. Isoba from Heriot-Watt University for sharing his expertise regarding underwater sonar data processing. Special thanks to Peter Robinson, Clive Williams and Chris Agar from J&S Marine Ltd. Not forgetting, that the author is partially sponsored by the IMarEST via a Stanley Gray award, an Overseas Research Scholarship (ORS), and the University of Plymouth scholarship.

I owe my greatest debt to my parents, Liang Tong Tan and Lily Teh, who have not only brought me into this world, but nurtured me and made me what I am today. I would take this opportunity to express my deepest gratitude to my girlfriend, Sook Yee Choo, who without her love and encouragement and moral support, I would not have finished the study. Lastly, I shall never forget the support from my sister, Tsu May Tan and brother, Jun Xiang Tan.

Declaration

I hereby declare that the thesis submitted in partial fulfilment for the requirements for the degree of Doctor of Philosophy that

- the thesis comprises only of my original work.
- due acknowledgment has been made in the text to all other material used,
- during the candidature, I have not been registered for any other award at any other institution,
- this study was financed with the aid of studentship from The University of Plymouth and the Institute of Marine Engineering, Science and Technology (IMarEST), and Overseas Research Scholarship (ORS).
- relevant scientific seminars and conferences were regularly attended at which work was often presented; external institutions were visited for consultation purposes and several papers prepared for publication,
- the thesis is below 60 000 words in length, exclusive of tables, bibliographies and appendices.

Chiew Seon Tan
January 2006

Table of Contents

List of Figures

List of Tables

Nomenclature

δ_r	rudder deflection
η	vector of position and Euler angles in earth-fixed frame, $[x\ y\ z\ \phi\ \theta\ \psi]^T$
$\mathbf{C}$	Matrix of Coriolis and centripetal terms including added mass
$\mathbf{C}_{RB}$	Matrix of Coriolis and centripetal forces acting on the rigid body
$\mathbf{D}$	Hydrodynamic damping matrix
$\mathbf{M}$	Inertia matrix including added mass
$\mathbf{M}_{RB}$	Rigid body inertia matrix
ν	Body-fixed linear and angular velocity vector,$[u\ v\ w\ p\ q\ r]^T$
ϕ	Heave angle
ψ	Pitch angle
τ	Control inputs vector
τ	Propulsion forces from thrusters and control surfaces
τ_E	Environmental forces such as ocean currents, waves and wind
τ_H	The radiation induced forces and moments which includes added inertia, hydrodynamic damping and restoring forces
τ_{RB}	Generalised external forces and moments vector acting in the body-fixed coordinate system, $[X\ Y\ Z\ K\ M\ N]^T$
θ	Roll angle
g	Gravitational and buoyant generalised forces vector

I_{zz} moment of inertia at z axis

J Transformation matrix from body-fixed coordinate to inertia frame

K Moment acting on x-axis wrt body fixed frame

kD-Tree k Dimensional Tree

M Moment acting on y-axis wrt body fixed frame

N Moment acting on z-axis wrt body fixed frame

N_r coefficient of sway moment from yaw

N_δ rudder input coefficient for yaw

$N_{\dot{r}}$ coefficient for added mass moment of inertia in yaw

$N_{\dot{v}}$ coefficient for added mass moment of inertia in sway

N_v coefficient of sway moment from side slip

p Roll rate

q Pitch rate

r Yaw rate, also known as heading rate

$SE(2)$ Special Euclidian group of rigid body displacements in two dimensions

$SE(3)$ Special Euclidian group of rigid body displacements in three dimensions

$SO(2)$ Special orthogonal group of rigid body spherical displacements in two dimensions

$SO(3)$ Special orthogonal group of rigid body spherical displacements in three dimensions

u Surge

$u_{c/E}$ x-axis current velocity component with respect to the Earth fixed reference frame

$u_{c/O}$ x-axis current velocity component with respect to the vessel body-fixed reference frame

v Sway

V_c Underwater current velocity

$v_{c/E}$ y-axis current velocity component with respect to the Earth fixed reference frame

$v_{c/O}$ y-axis current velocity component with respect to the vessel body-fixed reference frame

w Heave

X Force acting on x-axis wrt body fixed frame

x Displacement in x-axis wrt inertia frame

X_{prop} Coefficient for propulsion thrust

Y Force acting on y-axis wrt body fixed frame

y Displacement in y-axis wrt inertia frame

Y_δ rudder input coefficient for sway

$Y_{\dot{v}}$ coefficient for added mass in sway

Y_r coefficient for sway force induced by yaw

Y_v coefficient of sway force induced by side slip

Z Force acting on z-axis wrt body fixed frame

z Displacement in z-axis wrt inertia frame

AHRS Altitude and Heading Reference System

ANN Approximate Nearest Neighbour

ARE Algebraic Riccati Equation

AUV Autonomous Underwater Vehicle

CFD Computational Fluid Dynamics

CVF Collision Violation Frequency

DOF Degree of Freedom

DP Dynamic Programming

EGNOS European Geostationary Navigation Overlay Service

ENU East-North-Up coordinate convention

FSM Finite State Machine

GA Genetic Algorithm

GPS Global Positioning System

IMU Inertia Measurement Unit

IP Integer Programming

LBL Long Baseline

LCG Linear Congruential Generator

LP Linear Programming

LQG Linear Quadratic Gaussian

LQR Linear Quadratic Regulator

MA Manoeuvre Automaton

MILP Mixed Integer Linear Programming

MRE Matrix Riccati Equation

NED North-East-Down coordinate convention

NN Nearest Neighbour

ROV Remote Operate Vehicle

RRT Rapidly-exploring Random Tree

SA Simulated Annealing

SBL Short Baseline

SDRE State Dependent Riccati Equation

SLAM Simultaneous Localisation and Mapping

SQP Sequential Quadratic Programming

TCAS Traffic Alert/Collision Avoidance System

UUV Unmanned Underwater Vehicle

WAAS Wide Area Augmentation System

Chapter 1

Introduction

1.1 Motivation

It is surprising to know that approximately 80% of the Earth bound organisms are in the ocean and the oceans constitute 99% of the living space on the planet. Thereby, supporting the fact that oceans play a vital role in sustaining the Earth's ecology (MarineBio 1998). Even with current technology advancement, the surfaces of Mars, Venus, and the Moon are much better mapped than Earths's ocean floors (Smith 2004). One obvious reason is that the exploration of this environment is extremely difficult to perform, however it is still desirable for the advancement of economic, political, scientific and military purposes.

Consequently, over the last few decade there has been an exponential growth in the applications of unmanned underwater vehicles (UUVs), particularly in the field of science, the offshore industry and the military. Cost reductions and the mitigation of the risk of human life have become the impetus for UUV exploitation. UUVs can be used for sea bottom exploration, repairing, surveying, policing exclusive economic zones, mine-hunting, seabed mapping and scientific data, and intelligence gathering. In this context, the term 'unmanned underwater vehicle' is considered as a generic expression to describe both an autonomous underwater vehicle (AUV) and a remotely operated vehicle (ROV). ROVs are human operated via an umbilical cable and are highly manoeuvrable underwater vehicle. However, this does severely limit its oper-

ating range. Unlike the ROV, the AUV without the restraint of an umbilical, is a free swimming vehicle of higher autonomy, capable of performing missions that require longer operating ranges without human intervention. For clarity of exposition, the term AUV will be used for the remainder of the thesis because of its more challenging and rigid requirements. Nonetheless, the ideas discussed are still applicable to a wide range of vehicles. It is worth reviewing the recent trends in AUV applications, to better appreciate their contributions.

The offshore and scientific communities, who are especially sensitive to financial constraints, were quick to seize the opportunity in exploiting the potential of AUVs. Some of the commercial AUVs for offshore survey are *Hugin* (Norway) (Vestgård 1999), *Aqua Explorer 1000*(Japan) (Kato *et al.* 1993) and *Theseus*(Canada) (Butler and Hertog 1993). These has been reinforced by the recent placing of orders to purchase AUVs by Fugro-Geos Ltd, C&C Technologies and Racal Survey Ltd (Anon 1999). In the case of the scientific community, *AUTOSUB* (Griffiths *et al.* 1999) (Fig 1.1) and *Theseus* AUV (Ferguson *et al.* 1999) (Fig 1.2) have demonstrated their ability to navigate under polar ice caps while the *Autonomous Benthic Explorer* (ABE) (Yoerger *et al.* 2000) (Fig 1.3) has performed a fine-scale sea floor survey in a rugged deep-ocean terrain. All of these have been achieved at significant financial cost saving. These impressive achievements further strengthen the belief that AUV applications will continue to escalate as the realisation of the importance of ocean resources unfolds.

Recently, the military has shifted its focus from blue-water to brown-water warfare. This was instigated by the increase propensity for littoral water operations and the attendant focus on amphibious power projection (Foxwell 2000). Technically, the littoral zone is a subdivision of the benthic province that lies between the high and low tide marks and can be considered as an extension of the shoreline to 600 ft (183 m) out into the water. The importance of accessing the littoral zone is critical if a successful amphibious launch is to be achieved. The littoral zone is an intricate area to navigate by default, with unpredictable nature's effects such as bioluminescence[1], internal waves, coastal currents, changing beach profile, reefs and artificial objects. This is made increasingly difficult by the proliferic deployment of cheap un-

[1]Refers to the light producing ability of certain surface organisms. Any provocation of the organisms will cause them to emit light. Thus to maintain stealthiness, AUVs must take extra precautions when travelling on or near the surface.

Figure 1.1: Retrieval process of the *AUTOSUB* (courtesy of National Oceanography Centre)

derwater mines by defending countries which have the effect of retarding or halting any military advancement. Consequently, this has prompted a search for the most effective countermeasure that culminated with the employment of AUV technology. Currently, the US employs the highly portable and cost effective *Remus* AUV (Fig 1.4) for mine hunting as illustrated in the recent 2003 Iraq conflict (Jordan 2003). Moreover, the design of a more advanced, stealth AUV codename *Manta* is already in progress (Lisiewicz and French 1999) and the Long Term Mine Reconnaissance System (LMRS) is also due in service next year. Elsewhere, the UK is using the highly modular and reconfigurable *Marlin* (Tong 2000) that is also submarine-launch capable and a smaller, more agile, *Gambit* (Morrison *et al.* 2003) for mine counter measure mission.

From the aforementioned, the impression may have been given that AUVs are going to become the panacea for a number of subsea activities. This is certainly not the case as AUVs still suffer from numerous inherent technical difficulties specifically from power, sensing, communication, and reliability limitations. The increase exploitation of AUVs as mentioned above has also demanded a more robust and autonomous capability. One of the areas that needs particular addressing is collision avoidance, which is required to maintain the structural integrity of the AUV in a very hostile

Figure 1.2: *Theseus* operating under ice (courtesy of ISE Research Ltd)

environment. Clearly, once a collision occurs and assuming that the AUV structure is breached, then it is inevitable that a catastrophic failure will follow. Thus an AUV collision in the ocean is intolerable for two main reasons; the recovery process can be arduous and the replacement process in terms of cost and time can be prohibitive. ***In this thesis, collision avoidance is defined as the ability of a vehicle to detect and avoid colliding with both static and dynamic obstacles, while still attempting to accomplish the current mission objective.*** The processes involved also encompass obstacle detection, digital map building (workspace representation), motion planning and reflexive obstacle avoidance. To date AUV collision avoidance schemes have been somewhat *ad hoc* whereby only a simple reflexive module is added depending on the customer requirements. The problem with this approach is that the resulting AUV system is not fully integrated with its various subsystems which leads to suboptimal behaviour in terms of manoeuvrability and capability to navigate in a dynamic, unknown and unstructured terrain. Moreover, under operational conditions, this is exacerbated still further by the poor sensor performance and highly complex nonlinear dynamic nature of an AUV. Previous AUVs were designed as efficient swimmers with very limited manoeuvrability, but some recent AUVs are

Figure 1.3: *Autonomous Benthic Explorer* (ABE) being deployed (courtesy of WHOI, `www.whoi.edu/home`)

equipped with auxiliary vector thrusters[2] to achieve more sophisticated manoeuvres such as hovering and pure sway movement. As a consequence, a new collision avoidance system that is capable of addressing all these factors is required.

Perhaps, before delving deeper into the technical content, it is worth reviewing briefly the history of collisions, both at sea and in the air. Since man's early foray into the oceans, mid-sea collision has been a frequent occurrence. The poor navigation technology at that time was partly to be blamed for this situation. Various regulations, or 'Rules of the Road', were enacted in the hope to mitigate the occurrence of such incidents. The proliferic deployment of radar, a significant technology, in commercial vessels at the end of World War II was hailed as the solution to this centuries old problem. Yet, much to the surprise of the maritime community even with this innovative technology, regrettably failure to prevent collisions from happening occurred as exemplified by the *Andrea Doria/Stockholm* (Moscow 1959, Cahill 1997) disaster in 1956. It was painfully clear subsequently, that most of the collision causes were not technology related but mostly due to gross human errors, particularly the incorrect application of the technology and collision avoidance regulations. More recently, such unfortunate events have also been extended to the air domain as well, as confirmed by

[2]One example is the novel flexible foil propulsor called *Nektor* (Hobson *et al.* 1999). These flexible thrusters are capable of maximising AUV agility by transforming it into a holonomic vehicle.

Figure 1.4: *Remus* being launched from a boat which clearly demonstrates its portability and versatility (courtesy from Hydroid Inc)

the mid-air collision between a *DHL Boeing 757* and the Bashkirian Airlines *Tupolev 154* near the Swiss/Germany border in July 2002 (BBC News 2002, Harro and Fabian 2002).

Undeniably, every collision either at the sea or in the air has serious environmental, economical, and human life implications. To make matters worse, the current forecast indicates an escalating trend in the number of vessels, aircraft and their operations in the near future. This is set to increase the probability of collision as a result. Fortunately, this predicament has not been taken lightly by the Civil Aviation Authority (CAA) who just published the CAP 722 report (Civil Aviation Authority (CAA) 2002), outlining some of the regulations pertaining to the legal and safety operations of unmanned aerial vehicles (UAVs). NASA, being slightly more pragmatic, is employing an UAV named *Proteus* (Aerotech News and Review 2003) as a testbed for state-of-the-art collision avoidance technology. They envisage that *Proteus* will be able to fly reliably and autonomously in national civil airspace within two years.

Although there exists some pioneering efforts in establishing certain public laws concerning AUV operations (SUT 2000*a*, SUT 2000*b*, SUT 2000*c*), what is urgently

lacking is an authority who can implement these instituted rules. One reason for the absence of interest in enforcing these rules is probably due to insufficient risk justification, especially the risk to human life. Unlike an UAV, which shares the same civil airspace with commercial aircraft, an AUV normally conducts its mission under the water where the chance of encountering another AUV or submarine is extremely unlikely at the moment.

In spite of this, the current scenario is about to change as there is a sudden surge of interest in the field of multi-agent underwater robots. By working cooperatively and via mutual information sharing, these AUVs will be able to complete missions such as oceanographic sampling (Chappell *et al.* 1997) and mine hunting with substantial reduction in both operational time and cost. This, as a result, necessitates a set of proper 'Rules of the Road' to safely and successfully conduct a multi-AUV mission.

In the forthcoming text several notions such as configuration space, holonomic system, nonholonomic system and underactuated system, will be used pervasively. Therefore it is felt that an explanation of these notions is in order.

Configuration space (C-space) is a fundamental tool introduced in the late seventies to address the basic motion planning problem (Lozano-Perez and Wesley 1979). C-space is a set of all possible configurations of a robot or, to be more precise, a vehicle in this case. The dimensionality of a C-space is equivalent to the number of degrees of freedom (independent parameters) of the vehicle. For instance, Fig 1.5 shows a 4-wheel vehicle constrained to plane movement. One can describe the vehicle configuration using 3 variables; (x, y, ψ), two translations and one rotation, concluding that this is a 3 dimensional C-space. Unlike a workspace, in a C-space the vehicle shape is 'patched' to the obstacles. Subsequently, the vehicle can be represented as a point which has the effect of simplifying the path planning process. In general, the high dimensionality of the C-space of nontrivial devices is perceived as the principle reason behind the complexities of a motion planning problem (Hwang and Ahuja 1992).

A system is considered to be *holonomic* or *fully actuated* when it has the same number of independent inputs as the configuration variables. Elsewhere, a nonholonomic system arises when the system has less control inputs than its configuration variables. These are generally characterised by nonintegrable constraint equations involving the time derivatives of the system configuration variables. As stated in the configuration

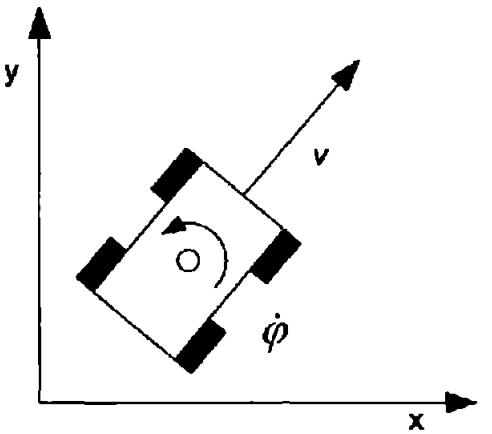


Figure 1.5: A diagram showing a 4-wheel vehicle configuration variables and control inputs

space section, the 4-wheel vehicle has only two independent control inputs (v, speed and, $\dot{\psi}$, angular velocity) contrasting to three configuration variables, which results in a nonholonomic system. A first order nonholonomic relation can normally be written in the form of a time-invariant ordinary differential equation (ODE),

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{1.1}$$

that deals with only nonintegrable velocity. Where $\mathbf{x}$ is the state vector, and $\mathbf{u}$ is the input vector. On the other hand, the second order nonholonomic relation, can be written as,

$$\ddot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t)) \tag{1.2}$$

which deals with nonintegrable acceleration. This type of problem is also known as a kinodynamics problem (Amato and Wu 1996, LaValle and Kuffner 1999) and is frequently found in *underactuated* systems such as surface vessels, spacecraft, manipulators and underwater vehicles. Suffice to say that controlling and motion planning for these systems is significantly more challenging than for the holonomic cases. A thorough review of nonholonomic controls and planning can be found in Laumond (1998).

1.2 Aim and Objectives of the Research

The overall aim of this research is to design and develop a collision avoidance system for AUVs. This encompasses the obstacle detection and the obstacle avoidance subsystems. Both simulations and implementation are investigated. As for the proof of concept, implementations are sought wherever possible, however, due to the unavailability of certain critical sensors, hardware and facilities, this is not always feasible. The simulations must be conducted as realistic as possible using accurate nonlinear dynamics of AUVs. It should be stressed that to formulate a meaningful and realistic thesis research project, there should be limitations on the scope of the research to make the problem manageable. Thus in keeping this research focus, the reflexive techniques, navigation and multiple target tracking topics have been omitted. The objectives of this research are provided as follows:

a. Critically review the current collision avoidance techniques, with a special emphasis on UUV applications.

b. Analyse, adopt and enhance a potential motion planning scheme for the AUV implementation.

c. Design a novel motion planner which can generate system dynamic compliant and near-optimal trajectories whilst remaining computationally attractive.

d. Develop a robust, multivariable trajectory tracker for underactuated vehicles such as AUVs.

e. Critically assess the suitability of various trajectory tracking controllers.

f. Evaluate the proposed obstacle avoidance technique's performance in simulations for various scenarios.

g. Employ suitable techniques and the AT500 sonar to develop a functional obstacle detection system.

h. Conduct a sea trial to check the viability of the proposed sonar data processing and workspace representation schemes.

Objective (a) is detailed in Chapter 2. Likewise, Objectives (b) to (e) are discussed within Chapters 4 to 6. These objectives pertain to the design of the obstacle avoidance module of an AUV. Specifically, the work within Chapter 5 details the results concerning objective (c). Chapter 6 documents work satisfying the objectives (d) to (f) whilst objectives (g) and (h) are accomplished in Appendix A. The last two objectives detail the development of obstacle detection techniques with a particular emphasis on the AT500 sonar application.

1.3 Thesis Overview

Accordingly, the thesis is structured as follows, Chapter 2 elaborates upon the related research and technologies employed for the collision avoidance purpose, particular for those that are being applied in AUVs. Due to the complexity of the topic, the survey has been divided into two distinct parts, the obstacle detection and the obstacle avoidance. Subsequently, Section 2.1 surveys control architecture and Section 2.2 deals with system architecture. The detection system, both software and hardware, are explored in Section 2.3. The remaining section focuses on a plethora of motion planning and reflexive techniques. This detailed technology survey has culminated to the publication of two papers (Tan *et al.* 2004*b*, Tan *et al.* 2004*c*).

Chapter 3 elaborates regarding the system and disturbance modelling. Clearly, in order to run a realistic simulation, an accurate mathematical description of both the plant and the disturbances must be available. The rigid body dynamics of an underwater vehicle are briefly explained before being presented with two models of AUVs, the *Remus* and the *AUTOSUB*. The *Remus* model is employed only for RRT simulation in Chapter 4. Owing to the absence of proper surge dynamics, the more realistic and complex model of *AUTOSUB* is preferred and adopted for the rest of the simulation runs. The modelling of both exogenous and endogenous disturbances which correspond to underwater current and sensor noise are also being treated herein.

With the knowledge gained through the aforementioned survey, it is decided that this research will focus on the RRT algorithm. Its intriguing properties coupled with its strong potential for practical implementation rendered it a worthy topic of research. As a motion planning algorithm, the RRT algorithm is used to find the feasible control

inputs that can take the vehicle from the initial state to the final state. Hence, Chapter 4 is reserved to detail the RRT mechanism, properties, algorithm, and its implementation. An enhanced version of the RRT algorithm is proposed and applied to a 3-DOF, nonlinear, *Remus* AUV model, where the results are also compared and discussed.

Chapter 5 combines the RRT algorithm with Manoeuvre Automaton (MA) approach, to ameliorate the former algorithm performance. In essence, MA transforms the continuous time nonlinear AUV model into a hybrid model. Effectively, this lowers the computational requirement and provides higher level of abstraction in solving the motion planning problem. Since, RRT solutions are inherently suboptimal, one exploits the linear programming optimisation algorithm to obtain a near-optimal trajectory instead. Also, for the case of performance betterment, the pseudo-random generator is replaced by a quasi-random generator. A novel multi-node version of RRT which can also cater for the case of time varying target is proposed.

Nonetheless, in practice, applying only the control inputs found by the trajectory planner, RRT in this case, to the vehicle will not be sufficient to guarantee that it will arrive at the desired final state. This is true since even with very small internal and external disturbances, the vehicle will diverge from the predefined trajectory. Therefore, it is crucial that a trajectory tracking controller be designed to address this issue. Chapter 6 investigates and compares two candidate multivariable tracking controllers, one is the popular Linear Quadratic Regulator (LQR), another a less well-known but still a highly effective, nonlinear controller known as State-Dependent Riccati Equation (SDRE) controller. Simulation results are also presented and discussed. A few remarkable features of the proposed SDRE controller are: it is AUV dynamics independent, considerably robust, and very simple to tune. Indeed, a highly pragmatic solution to AUV tracking control problems.

Conclusions, achievements to date and future work are condensed in the Chapter 7.

Appendix A documents an additional and original work carried out on the topic of sonar data processing and workspace representation. These two submodules, including the multiple-target tracking submodule constitute a modern, generic, AUV obstacle detection module. All the data presented in this chapter were acquired using an AT500 sonar (Robinson *et al.* 2003) from J&S Marine Ltd, during a sea-trial at

Coxside, Plymouth. This prototype forward looking sonar was designed especially for AUV's obstacle avoidance purposes. The raw data supplied by the sonar is processed within the context of image processing theory. The extracted information is later transformed into an efficient workspace representation structure known as the occupancy grid. Simulations demonstrate how an AUV navigates the environment. Additionally, Appendix B provides the author's published work, and Appendix C outlines the AT500 specifications.

1.4 Contributions of the Thesis

The major contributions of this work are seen as:

- Providing an up-to-date, comprehensive review of the current collision avoidance techniques, with special attention to UUVs. As a sign of keen maritime community interest, a section of this review has been republished in *Oceanology Today* (Tan *et al.* 2005*a*).
- A true 6-DOF nonlinear SIMULINK model of the *AUTOSUB* model was developed. The original model provided by QinetiQ Ltd lacks surge dynamics.
- An enhanced RRT algorithm, based on the 'reconnection' concept was devised. The algorithm optimization is based on a prescribed cost function, in this example, the shortest distance criterion is employed.
- A novel multi-node version of RRT+MA which can also cater for the case of time varying target is proposed. To author's knowledge, this is the first study of an AUV implementation of this particular technique.
- Performed a detailed study between the LQR and SDRE controller, with the objective to select a potential candidate as the tracking controller. It was discovered that the SDRE controller performance is substantially superior compared to LQR in this case and is suggested for future AUV applications. It is shown that it provides a flexible and yet simpler alternative to other underlying multivariable controllers.

- Sonar data processing and workspace representation techniques have been conducted by employing the prototype AT500 sonar. This was achieved with collaboration with J&S Marine Ltd. Here, the emphasis is on practicability of the techniques.

Chapter 2

Related Research and Literature Survey

A survey of the current state-of-the-art algorithms and methodologies has been pursued with the aim of identifying the most suitable technology to address the AUV collision avoidance problem considered in this thesis. For clarity of exposition, the structure of this chapter commences with an generic AUV control architecture, Section 2.1, and immediately followed with the system architecture in Section 2.2. As suggested by the section titles, control architecture and system architecture are inexorably linked. Control architecture pertains to a framework which manages the sensorial and actuator system in order to enable an AUV to undertake a user-specified mission. On the other hand, a system architecture defines the interconnection map of vital modules to allow the proper functioning of an AUV.

Section 2.3 elaborates upon the detection system, both software and hardware are discussed. A detection system functions as the 'eye' of an AUV, an essential module for a collision avoidance system. This is then accompanied by Section 2.4, 'Rules of the Road', Section 2.5 on motion planning techniques and Section 2.6 on reflexive techniques. Although, motion planning and reflexive avoidance techniques share the same objective, which is to avoid any collisions from occurring, compared to the reflexive avoidance techniques, the motion planning techniques are more comprehensive and utilised more information related to the environment and obstacles. This allows the motion planning techniques to function in an complex environment without being

trapped in a local minima. Conversely, reflexive avoidance techniques are *ad hoc* in nature, and able to function with minimal environmental information and computational resources while ensuing fast reaction. Generally speaking, the majority of the techniques are also adopted by a few land and air research vehicles. However, in this case emphasis shall be given to those techniques that are potentially useful to AUVs. Finally, concluding remarks are provided in Section 2.7.

2.1 AUV Control Architecture

A control architecture is a framework which manages the sensorial and actuator system in order to enable an AUV to undertake a user-specified mission. This is a major topic of research and different approaches to AUV control architectures are discussed in the literature (Ridao *et al.* 1999, Valavanis *et al.* 1997, Caccia *et al.* 1995*b*). This section intends to elaborate on three major types of control architecture.

2.1.1 Deliberative architecture

This architecture is also known as a top down, structured, symbolic, goal-driven, model-based, hierarchical or sense-plan-act approach. Deliberative architecture always maintains internal representations of its surroundings and this allows it to make reasoning, prediction and inferencing concerning the environment. The information flow direction is depicted in Fig 2.1(a). This scheme represents a well-defined tightly coupled structure thus simplifying the process of designing, debugging and evaluating the system. However, the amount of information flow from sensors to the centralised computing resources can be significant. Exacerbating the situation is the synchronisation difficulty of workspace representations and the environment. Owing to the computationally intensive nature of the architecture, there is a tendency to exhibit unresponsive or erratic behaviours in unpredicted situations. This architecture is employed in the *EAVE* (Blidberg *et al.* 1990) and the *OTTER* (Rock *et al.* 1995).

2.1.2 Reactive architecture

Also known as a bottom up, sensor-driven, layered, forward inferencing, subsumptive, heterachical, behavioural and reflexive or sense-react approach. The theory of reactive architecture was initiated by Arbib (1981) and implemented by Brooks (1986). It is based on a parallel structure where each individual sensor is used to sense the environment, providing its own perception and activating its own behaviour, refer to Fig 2.1(b). A global behaviour is produced by coordinating the parallel execution of individual behaviour. Its performance is excellent particularly in unforeseen situations. Furthermore this scheme is known for its flexible and modular nature. However, its propensity to demonstrate elusive behaviour when subjected to conflicting sensor information is a major concern. Also, its nondeterministic nature does not lend itself to a straight-forward performance evaluation. Lastly, its deficiency in global mapping and in relation to workspace objects often results in simplistic behaviours which tend to get trapped in certain cases. This architecture is employed in the *Sea Squirt* (Bellingham *et al.* 1990) and the *Twin Burger* AUV (Fujii and Ura 1996).

2.1.3 Hybrid architecture

In the search for a superior architecture than the two previously discussed, the hybrid architecture was born through the amalgamation of both the above architectures. Generally, it is decomposed into three task specific layers: deliberative, reactive and execution layer, refer to Fig 2.1(c). In military parlance, it is called the strategic, tactical and execution layer. Abstraction and real-time responsiveness varies correspondingly at each level. The deliberative layer is in charge of high level planning (non time critical) while the reactive layer is responsible for real-time issues. The execution layer acts as supervisor to facilitate inter-layer interactions. Due to its apparent advantages, most recent AUVs have employed a variant of this architecture. The *Garbi* (Ridao *et al.* 2001), the *SAUVIM* (Yuh and Choi 1999) and the *Phoenix* AUV (Healey *et al.* 1995) are examples that exploit this architecture.

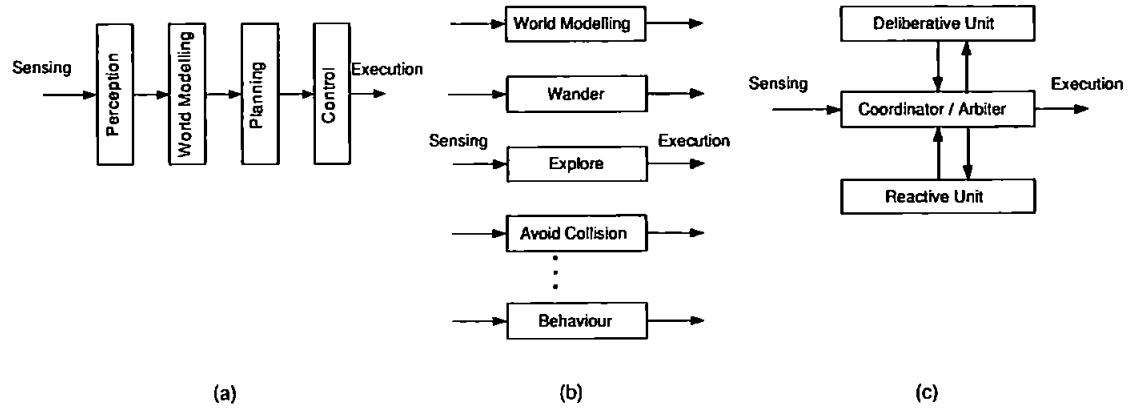


Figure 2.1: Types of control architecture (a) Generic deliberative control architecture (b) Generic reactive control architecture (c) Generic hybrid architecture

2.2 Collision Avoidance System Architecture

As stated previously, pure deliberative and reactive architecture do not function adequately for a collision avoidance task. As hybrid control architecture provides an ideal platform for integrating the functionality of the individual submodules, it is not surprising that it is applied in the majority of the proposed obstacle avoidance architectures (Hyland 1989, Arinaga *et al.* 1996, Antonelli *et al.* 2001, Moitie and Seube 2000, Lane and Trucco 2000). Before proceeding to an in depth discussion pertaining to the individual submodules, it would be more enlightening to provide a simple description of the collision avoidance process to better elucidate the utility of each submodule. A typical collision avoidance task can be considered like this. First and foremost, a target must be acquired by a **forward looking sonar**. Classification of static or dynamic targets are then performed. Depending on the types of object, their information will then be fused with AUV navigation data such as velocity, depth and altitude in order to represent the object in a **digital map**. From the digital map, a **motion planning** technique is employed to steer the AUV safely to its predefined goal or subgoal. Motion planning is computationally expensive and not very suitable for tackling unexpected objects. Therefore, the **reflexive obstacle avoidance** submodule is employed to provide the AUV with a time critical, *in situ* response to an unexpected object. Once the obstacle has been successfully avoided, the AUV should resume its preplanned mission. The bolded phases denote critical processes in collision avoidance. These processes are highly dependent, particularly the one in the lowest of the process chain such as motion planning. For this reason,

a method of designing an efficient, optimal and practical collision avoidance system requires the perfect integration of these processes. On the whole, a collision avoidance system can be decomposed into two principal functional modules; the obstacle detection module and the obstacle avoidance module, where both of them comprise of further submodules.

Obstacle detection module

1. Forward looking sonar
2. Sonar data processing submodule
3. Navigation submodule
4. Map builder (Workspace representation submodule)

Obstacle avoidance module

1. Motion planner and waypoint generator
2. Trajectory tracker. (Autopilot and actuator controller)
3. Reflexive submodule

A detail discussion of aspects of obstacle avoidance is given in the Section ??. Fig 2.2 illustrates the interconnection of the submodules of a generic collision avoidance system. The arbiter is used to coordinate the activation and inhibition of various submodules.

Forward Looking Sonar

Forward looking sonar is frequently used for AUV obstacle detection. Recently, the advent of digital signal processing (DSP) technology has increased the popular usage of cost effective, high resolution, electronic beamformed sonar for obstacle detection

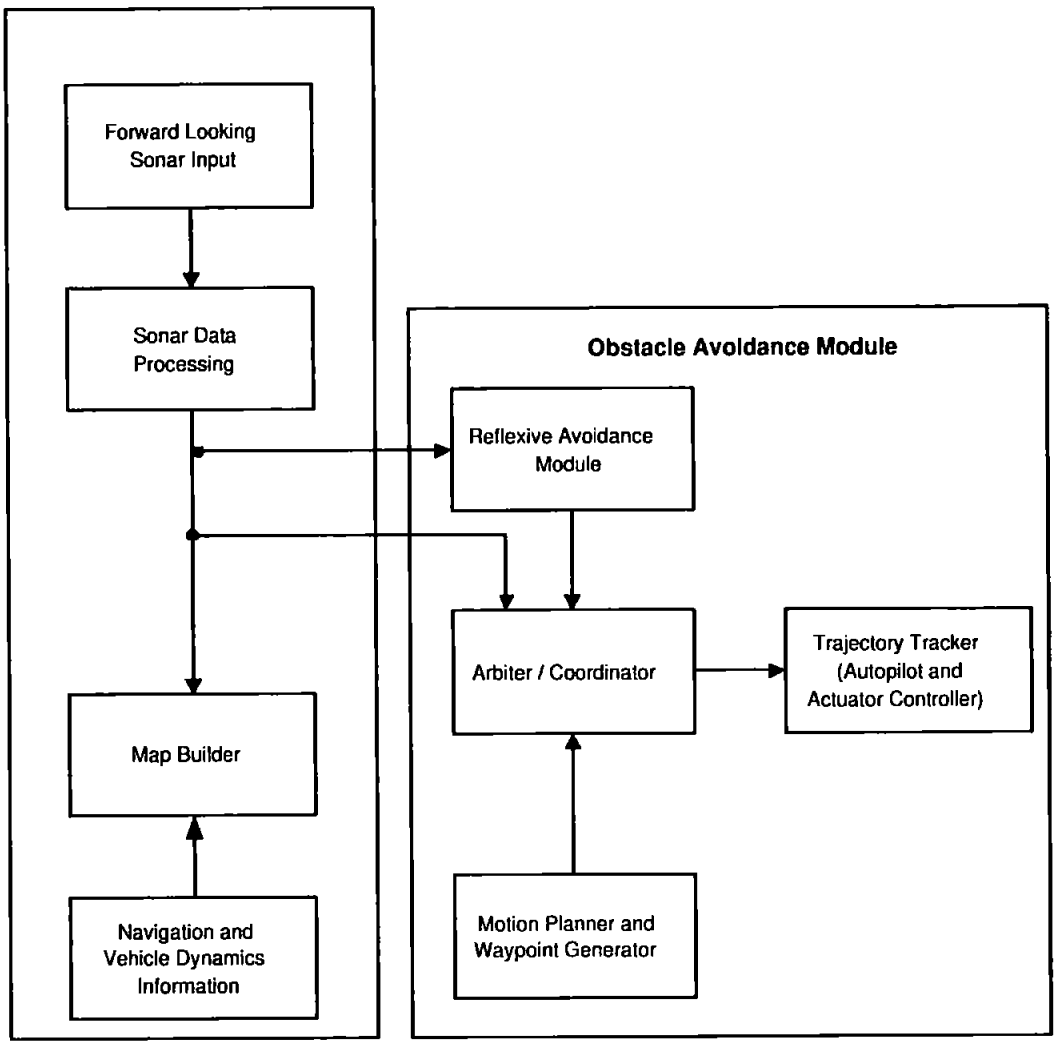


Figure 2.2: Generic collision avoidance architecture

purposes. Besides providing adequate bearing and range resolution, its rapid scanning rate (frame rate) also permits temporal information extraction, which is vital for motion planning in a dynamic environment.

2.2.1 Sonar Data processing submodule

Although, acoustic sensor performance is unprecedented in underwater applications, obtaining high quality and reliable sonar data is still problematic. Reverberation, reflection, refraction and scattering tend to corrupt the data and cause frequent false alarms hence subsequent processing of the data is required. This submodule is re-

sponsible for object discrimination, verification and tracking.

2.2.2 Navigation submodule

A navigation submodule typically comprises of an inertial measurement unit, digital compass, depth sensor, altimeter, and a GPS unit (when surfaced). When submerged, the AUV is deprived of any global frame of reference and dead reckoning is the only viable method for localisation.

2.2.3 Map builder

Deprived of any global frame of reference, it is critical to have an online map which is incrementally developed to assist an AUV in navigating the unknown terrain. A digital map is also required for localisation and motion planning processes. Clearly, there are numerous methods of representing the AUV environment; three well known methods are cell decomposition, geometrical representation and topology representation.

2.2.4 Motion planner and waypoint generator

A motion planner is used to assist an AUV in navigating through an unstructured and unknown environment via the generation of a time-parameterised path, whilst simultaneously taking into account several factors such as AUV safety, kinematics, dynamics and energy constraints. It is true that motion planners are intimately related to guidance techniques (Lin 1991, Tan *et al.* 2003). Both attempt to furnish the AUV with corresponding continuous configuration variables, or states as to guide the AUV to the designated destination. The only subtle but significant dissimilarity here is that the guidance techniques assume that the environment is obstacles free. However, sometimes the term guidance tends to be abused and used to pertain to motion planning.

2.2.5 Trajectory tracker

A trajectory tracker should not be confused with the term autopilot, a commonly used controller despite that their functions are inexorably linked. In essence, trajectory tracker primary responsibility is to ensure that the AUV output follows the desired input. The actuators to be controlled can be a rudder, hydroplane or motor. This is not a trivial assignment when one needs to take into consideration the effect of the vehicle dynamics, modelling uncertainty, sensor noise and external disturbances. Presently, this area is a major area for research. On the contrary, the autopilot does not track a trajectory but only require to maintain the vessel heading given a reference one. This form of controller is significantly simpler to design, owing to the less degree of freedom. Examples of autopilots implemented in *Hammerhead* AUV can be found in Naeem (2002) and Naeem *et al.* (2003).

2.2.6 Reflexive submodule

A reflexive submodule function is similar to a backup system in the unfortunate event of motion planner failure. The failure can either be a system malfunction or a failure to meet the predefined time constraint which is a more common occurrence than the former. Unlike the motion planning submodule, this submodule is highly capable of responding to unforeseen circumstances. This submodule is activated when an object intersects a predefined virtual boundary (Hyland 1989, Zanoli and Affaitati 1999).

2.3 Obstacle Detection Module

As previously outlined, an obstacle avoidance module comprises of a forward looking sonar, a navigation submodule and a map builder. The navigation submodule will not be reviewed since this has been provided by Loebis *et al.* (2002). The primary function of an obstacle detection module is to detect, discriminate and represent the object information into a digital map for disposal by the obstacle avoidance module.

2.3.1 Forward looking sonar

In the underwater domain, radiowaves and vision suffer from inherent limitations. Radiowaves are virtually useless underwater due to its high attenuation while vision effectiveness is restricted to a range of a few meters, and highly dependent on the turbidity of the water. This is caused by the scattering effect of light by suspended matter. Obviously, one method is to employ a higher intensity light source to offset the light attenuation, but this only results in a massive power drain.

Unlike radiowaves and optical energy, sound transmission is the single most effective means of directing energy transfer over long distances in sea-water. Consequently, an acoustic sensor in the form of sonar, is largely employed underwater. There are numerous sonar types such as bathymetric sonar, side scan sonar, tow-array sonar, etc, which are all applications specific. One type that is commonly employed for obstacle detection is the forward looking sonar. The main purpose of a forward looking sonar is to provide spatial information such as the range, bearing and size of an object via some processes of signal processing and data fusing.

A forward looking sonar is required to detect objects at the longest range possible in order to allow for further information processing before an avoidance manoeuvre can be initiated. However, at moderate ranges of several hundred meters, sonar paths can be distorted significantly because of continuous refraction from sound speed variation caused by changes in water temperature, salinity, and pressure. To aggravate the situation, sonar range is also highly frequency-dependent, thus for long range detection, a low frequency sonar is required. Nonetheless, low frequency results in poor acoustic resolution. In the case of shallow water (200 m or less) and when an AUV is cruising near to the sea bed (pipe tracking or terrain following), this issue is exacerbated by the combined effect of boundary reverberation noise, multi-path returns and bottom clutter (Nussbaum *et al.* 1996). Increasing the acoustic resolution on the other hand, can significantly enhance an AUV ability to perform boundary reverberation discrimination while obtaining a more precise bearing on echo returns. Besides, high acoustic resolution is also critical for the purpose of map building and optimal path generation (Henriksen 1994). For this reason, there is a constant trade off between operating range and acoustic resolution, proper selection should be based on the AUV mission. Some of the desired qualities of an AUV sonar are listed below:

- Low power consumption
- High resolutions with adequate detection range (depending on the AUV manoeuvrability)
- High scanning rate
- Low cost
- Adaptive thresholding/clustering logic (Optional)
- Embedded static and dynamic objects tracker (Optional)

2.3.2 Types of forward looking sonar

A detail review of different types of forward looking sonars can be found in (Loggins 2001). Lately, the advent of DSP technology has culminated in the development of high performance, low cost electronic beamform sonar. In principle, beam-forming (Veen and Buckley 1988, Curtis and Ward 1980) is a process of listening to or transmitting energy (sound in this case) from an array at selected angles. The core concept is to sum the incoming signal such that those that are coming from a given direction are added coherently resulting in maximum magnitude response, while those signals arriving from other directions are attenuated as a result of the self-destructive interference effect. The two main approaches in beamforming are the time domain and the frequency based methods. Typically, a number of fixed directional receiver beams are formed simultaneously to cover the ensonified region in order to obtain better directional resolution while maximising the scanning rate (Nussbaum *et al.* 1996). One obvious advantage of this sonar is its high scanning rate (frame rate), rendering it less susceptible to platform motion disturbance. Besides, the high scanning rate can be exploited for temporal and spatial information (Lane *et al.* 1998). The lack of mechanical moving parts also increases the reliability of the sonar.

There are several types of sonar that are based on slightly different operational principles. Neglecting this distinction, one can as a whole, categorise them into a 1-D, 2-D and 3-D sonar. For clarification, the most primitive form of sonar is a 1-D sonar such as an echo-sounder or depth-sounder, which is capable of only providing range (altitude) information. Two examples of 2-D sonar, with different configuration are

shown in Fig 2.3(a) and (b). The Fig 2.3(a) sonar configuration is commonly employed by most commercial AUV forward looking sonars. Here, range and bearing information is acquired but not depth. This makes it suitable for AUVs performing mid-sea surveying and mine-searching missions, where the environment is uncluttered or sparse. However, discrimination of object depth can be difficult, example of graphical output from this type of sonar is depicted in Fig 2.4. Taking into account of the worst case scenario and assuming that the obstacle is on the same plane as the AUV, this certainly restricts the AUV to perform only planar evasive manoeuvres. A planar manoeuvre might not be the most effective in certain circumstances, since a better approach is may be to allow the AUV to climb or dive over the obstacle if possible. Another advantage of multibeam sonar is its capability to ensonified the illustrated region (Fig 2.3(a)) simultaneously in a single ping while the pencil beam sonar is limited to scanning the entire sector incrementally.

Fig 2.3(b), shows an alternative sonar configuration which is similar to the former but with the transducers being rotated through 90°. This configuration is commonly employed by a surface vessel for collision avoidance purposes. It provides depth and range information at the expense of bearing information. This mode tends to put more emphasis on discriminating objects in the vertical direction (terrain) than the horizontal (suspended object). This configuration is also beneficial for an AUV that is performing terrain hugging manoeuvres such as in a pipeline tracking and seafloor surveying missions as it needs to estimate the terrain gradient in advance to facilitate a successful manoeuvre.

The most advance forward looking sonar is of the 3-D type, (Fig 2.3(c)). This is

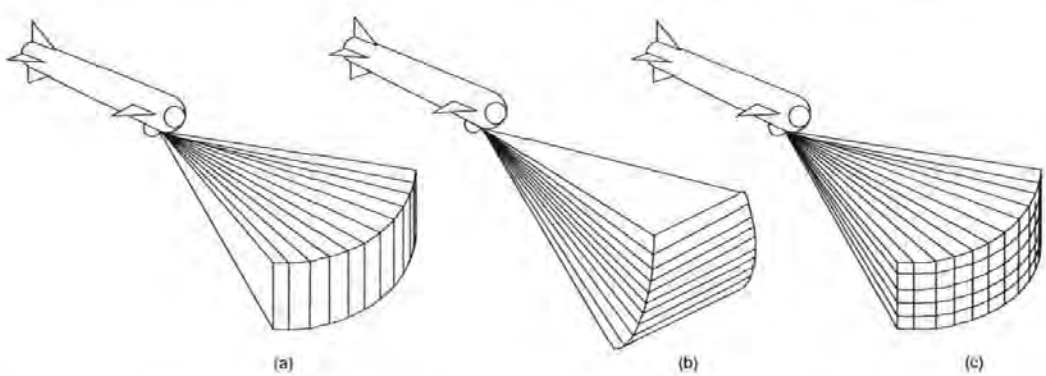


Figure 2.3: Multibeam sonar type: (a) Horizontal scanning 2-D sonar (b) Vertical scanning 2-D sonar (C) Horizontal and vertical scanning 3-D sonar

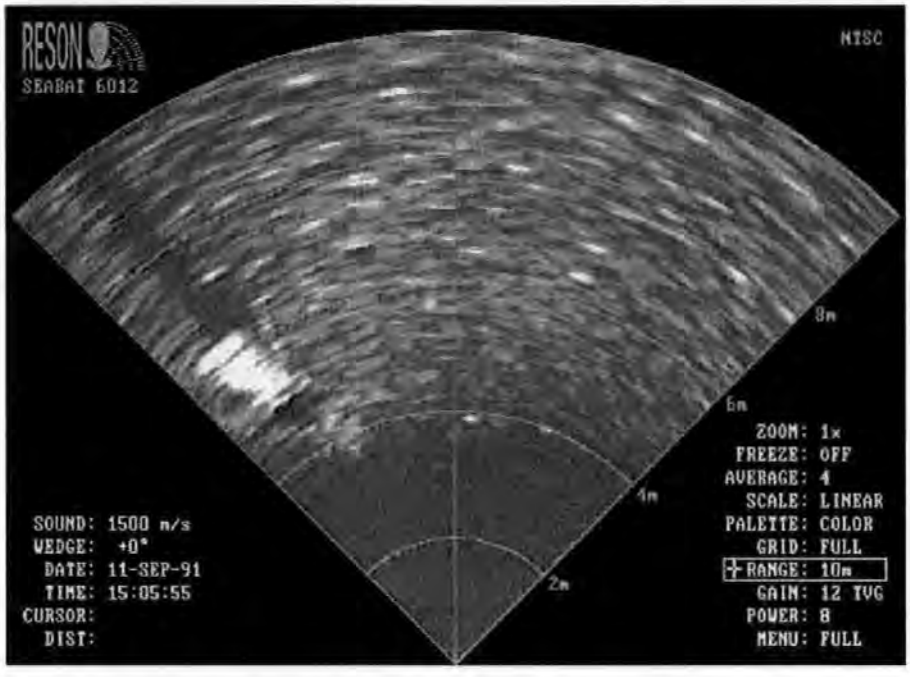


Figure 2.4: Reson 6012 multibeam 2D sonar output. Note the target on the left side, the rest are noises caused by back-scattering of the seabed (courtesy of Reson Ltd).

achieved using an array of transducers. It possesses the capability to discern an object in a spatial domain by providing range, bearing and depth information simultaneously. An example of graphical output of the sonar is shown in Fig.2.5 This type of sonar is needed to exploit fully the true capability of some motion planning algorithms in generating optimal, 3-D trajectory. Nevertheless, the cost of this type of the sonar can be prohibitive. To circumvent this issue, one method is to utilise the Fig 2.3(a) configuration, but making the vertical beam width thinner and steerable in the vertical direction, either mechanically or electronically. Assuming a sufficient fast steering (sweeping) rate is obtainable, then one can consider it as a pseudo 3-D sonar.

2.3.3 Sonar data processing

Owing to the operating principle and operating condition of a sonar, the raw data obtained tends to be corrupted with noise. This imposes further processing stages to obtain better representation of the environment. Petillot *et al.* (1998) reported that sonar data processing can be classified into four distinctive processes :

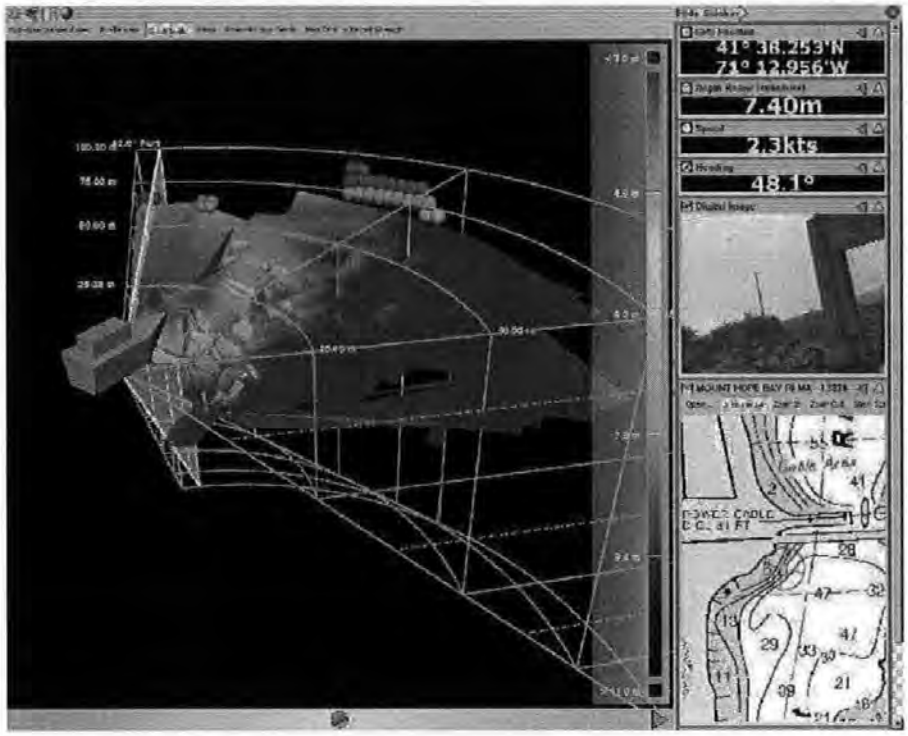


Figure 2.5: FarSounder 3-D sonar output which depicts a 3-D model of the shore. The small windows (right hand side) are maps used for navigation purposes (courtesy of FarSounder Inc).

- Filtering and Segmentation
- Feature Extraction
- Tracking
- Map Building (Workspace Representation)

Filtering and segmentation

The first step in the sonar data processing typically entails the elimination of noise and backscatter of sonar images caused by the scattering and reverberation effect. This can be achieved by applying a simple Gaussian, median or mean filter to the image. A median filter is the most effective in elimination of backscatter noise, however the Gaussian filter is sometimes used due to its lower computational requirement (Petillot *et al.* 1998).

Segmentation, a process of regioned pixel extraction, is applied to enhance object background discrimination in order to increase the robustness and accuracy of the tracking process. The most popular and simple is the thresholding technique also known as binarisation. In principle, thresholding is a process of defining a limit so that any colour above the limit will be converted into black while those below the limit will be converted into white. It is effective when the intensity levels of the objects fall squarely outside the range of levels in the background. A more sophisticated version, called the adaptive threshold technique, uses a switching function-integration to provide improved results (Lane *et al.* 1989). The use of a unsupervised hierarchical Markov random field (MRF) model together with contextual information has also been reported (Mignotte *et al.* 2000). Simulated annealing has also been attempted, but the algorithm was applied to segmentation of synthetic aperture radar images (Stewart *et al.* 2000) and not sonar. Both the algorithms are very computationally intensive, making real-time implementation very difficult.

Segmentation processes can be very costly in terms of computational requirement, as such some authors advocate using selective, multirate/multidepth filtering and data compressing techniques. In the selective approach, the static and dynamic part of the image is discriminated using a frequency domain method (one dimensional Fast Fourier Transform (FFT)) or a time domain method (moving average) (Dai *et al.* 1995). Once the dynamic object is detected, it will be tracked and segmented only at the particular region of interest. For a static object, only new objects need to be segmented. In contrast, the multirate/multidepth technique tries to redistribute the computational load by sampling the area at various rates depending on the degree of their importance (Henriksen 1994). Clearly, those regions adjacent to the AUV are more critical and deserve a higher sampling rate. Instead, Zanoli and Affaitati (1999) attempted to compress the sonar data before filtering, significantly reducing the processing requirement.

Feature extraction

Feature extraction is a process that is intimately linked with object classification (Lane *et al.* 1989, Lane *et al.* 1988). In the case of image processing, feature extraction entails accurate measurement of object features. Ideally, the feature selected should be invariant under various circumstances while extracting maximum informa-

tion regarding the object. Such features can be object size such as area, perimeter, surface and centre of mass which can be easily obtained by counting pixels of the object, or more complicated parameters such as moments, mean, variance, and median used to describe statistical distributions.

Tracking

In this context, tracking is a process where object attributes such as position velocity and estimation confidence level are estimated and recorded. In video processing, one tries to correlate a predetermined feature with subsequent frame features and noting their difference. Tracking is typically a forward-looking process, requiring a computer to anticipate the object position and velocity ahead of time. The accuracy of the predicted target attributes play a critical part in characterising its behaviour. Hence, making it indispensable for the motion planning process. Lane *et al.* (1996) applied an optical flow with an associative searching trees technique while Moran *et al.* (1993) advocated using a multiple hypothesis for object tracking. Multiple hypothesis is effective in cases where multi-modal representation is required, such as in the presence of background clutter, self-occlusions and complex dynamics. However, both these pixel based schemes are very computationally expensive thus precluding their application in time-critical applications.

Alternatively, the classical Kalman filter (Kalman 1960), has been applied with success in sonar tracking systems (Williams *et al.* 1990, Henriksen 1994, Ruiz *et al.* 1999, Trucco *et al.* 2000). To simplify the analysis and lighten the computational requirement, Williams *et al.* (1990) employed two different Kalman filters for tracking dynamic and static objects while Henriksen (1994) preferred using separate Kalman filters, a total of five, to track the corresponding states. However, this is not without problems. One inherent limitation of the Kalman filter, due to its derivation, is the assumptions of a linear model, Gaussian white noise, and uni-modal representation. Although, the extended Kalman filter can be adopted to address the non-linear model case, extreme precaution must be taken to avoid divergence issues (Bizup 2003). In addition, if certain discrepancies exist between the process description of the ideal case and practical case then its effectiveness can be greatly affected. These issues will be reserved for future research.

2.3.4 Map building (Workspace representation)

Knowledge representation is one of the key elements that determines the capabilities and performance of machine intelligence. This is particularly true for map building or workspace representation processes, which can be defined as a process of generating models that represents the vehicle environment via sensor measurements. The generated model or digital map, other than containing metric information, can also be embedded with supplementary user defined information to better characterise the environment. This information is vital for motion planning, obstacle avoidance and localisation processes.

The condensed information is more suitable for high-level symbolic manipulation and model inference. Therefore, the aggregate of discarded information and sensor induced errors, can be considered as noise, and is detrimental to the overall system performance. In the case of an AUV, the sensor drift in a dead-reckoning scheme tends to degrade the map reliability after a certain time period. Whilst increasing the model-fidelity will definitely enhance the system performance, but at the expense of memory and computational requirements.

For this purpose, map building can be considered as a trade-off between model-fidelity, memory requirement, robustness, computational efficiency, implementation simplicity and expansibility (Dudek and Jenkin 2000). There are three fundamental schemes in workspace representation; metric based spatial decomposition, geometry representation and non-metric based topological representation.

Spatial decomposition

Spatial decomposition is a scheme of representing space via a discrete sampling process; division of space into non-overlapping cells. Either only the free space is taken into account, or only objects are mapped and free space is found by implication. There are various variants of the spatial decomposition method.

The most conceptually simple and yet prevalent scheme in the field of mobile robotics is where the environment space is partitioned into uniform, non-overlapping grids or

cells in a spatial lattice (Fig 2.6(a)). Each cell can be allocated with user defined attributes such as confidence of obstacle presence, terrain geometry and safety factor. This scheme conventionally employs probabilistic sensor interpretation models to update the cell value (Movarec and Elfes 1985). Due to its popularity, it is known by different names, such as evidence grids, probability grids, certainty grids and occupancy grids. Hyland (1989) and Allison *et al.* (1989) have implemented this scheme in their AUV simulations. One overriding constraint concerning this approach is the high memory requirement. One must understand that the number of cells employed to approximate a model are finite, hence decreasing the cell size will definitely improve the model fidelity but at the expense of escalating the cells quantity. This problem is intensified for cases of higher dimensional space. As a result, various researchers have resorted to dual resolution maps; each map using different resolution. Ridao *et al.* (2000) describe using a high resolution map to record sonar echoes for the *SAUVIM* AUV. Only echoes that have not expired after a preset time interval are recorded into a coarser map, and used for path planning. Fundamentally, the high resolution map is functioning like a low pass filter to eliminate false alarms. Similarly, Moitie and Seube (2000) employed a low resolution map for global path planning and a more detailed local map when executing local motion planning.

To circumvent the inherent memory and computational inefficiency of a uniform cell map, a type of multi-resolution algorithm has been proposed (Kambhampati and Davis 1986). It is known as a quadtree and octree in their 2-D and 3-D forms respectively (Fig 2.6(b)). A quadtree is fundamentally a recursive data structure with a hierarchical representation property. It tries to exploit the occupancy of adjacent cells by clustering them much like a data compressing algorithm. It adaptively subdivides into smaller cells in order to improve the modelling accuracy while the minimum cell size determines the depth of the tree and the accuracy of the mapping. Fig 2.7 shows how it is represented in the form of a tree to facilitate quick searching. Its efficiency is much superior than the former method particularly for environments that are sparsely populated with objects (Yahja *et al.* 1988). However, its performance suffers significantly for the case of a dynamic object due to constant tree structure changes.

Another efficient workspace representation scheme (Naylor 1993), highly popular in the computer graphic domain, is the binary spatial partitioning (BSP) scheme. A BSP tree is a hierarchical representation structure, that exploits the recursive subdivision

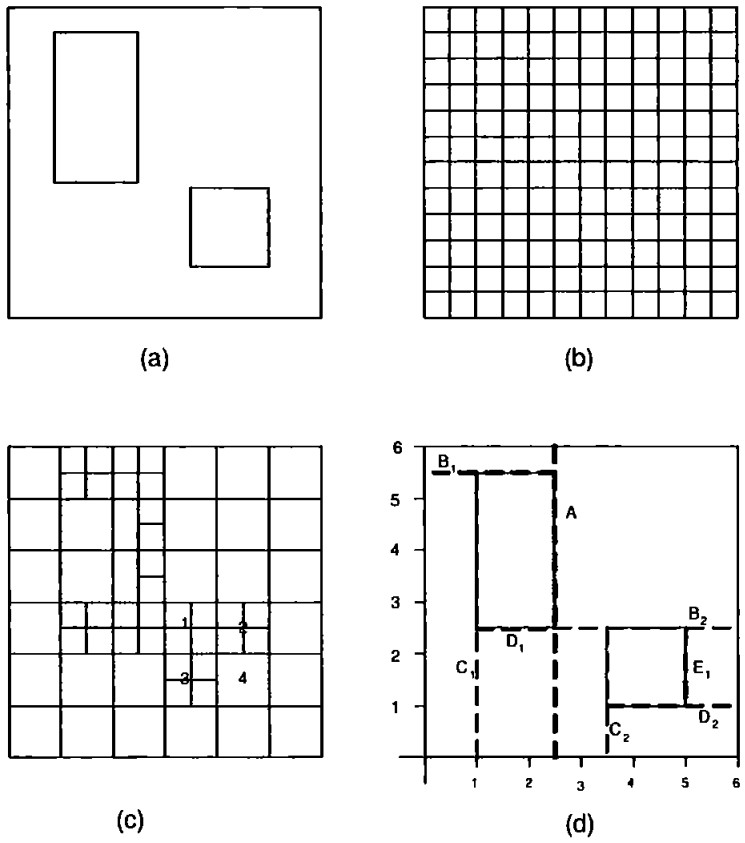


Figure 2.6: Diagram showing: (a) Workspace (b) Uniformed grid (c) Quadtree representation and (d) Binary spatial partitioning

by hyperplanes (Fig 2.6(d)). Since there is no restraint on the types of hyperplane used, exact polyhedra and polygon representations are possible. All of this information is then compactly encoded in the form of a binary tree structure, as shown in (Fig 2.7(b)), ready for subsequent implementation of path finding algorithms. Unlike the quadtree, the BSP tree structure is preserved by affine and perspective transformations, which result in its capability to incorporate dynamic objects without resorting to changing the tree structure. This scheme has been exploited by Arinaga *et al.* (1996) for the *Umihico* AUV workspace representation.

Geometric map

Probably the oldest of the workspace representation method is the geometric map. As suggested by the name, the geometric map tends to use geometric primitives such as points, lines, polygons, polyhedrals and polynomial functions to characterise the

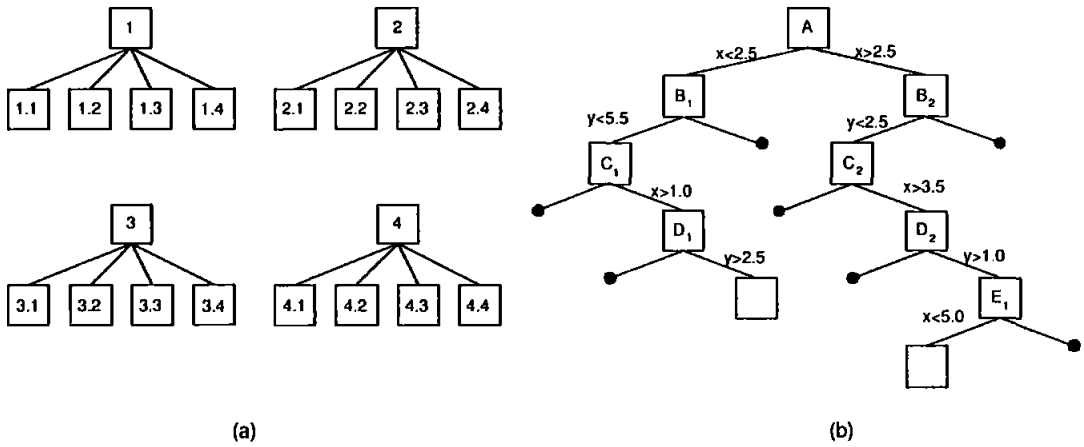


Figure 2.7: A tree representation of (a) Quadtree (b) Binary spatial partitioning

environment. One of the compelling advantages of a geometric map, assuming application of appropriate modelling primitives, is its capability to model complex objects with a very low memory requirement. This concise mathematical representation also facilitates a rapid and accurate collision checking process (Lin *et al.* 1996).

The simplest primitives such as point and line are rarely used in isolation but as a preliminary form of model inference. Leal (2003) employed points in what he referred to as the Sampled Environment Map (SEM) scheme. Unlike a uniform grids representation, here the environment is divided into discrete point locations. Then a decision theoretic scheme is used to adapt a geometrical model from the sampled environment distribution. Brutzman *et al.* (1992) employed this scheme to trace incrementally the obstacle contour by aggregating piecewise, linear lines into polygons. Caccia *et al.* (1995*a*) and Moran *et al.* (1993) developed modules to process and classify sonar data into corresponding geometrical features. The systems, however, are constrained to function only in a partially man-made environment since it is more geometrically distinctive compared to the non-homogeneous features found in nature. Of all the polygons, the circle or sphere have particularly interesting attributes such as simple formulation, orientation invariance, convex shape and ease of manipulations, thus explaining its popularity. This method of representation is employed by Fox *et al.* (2000), Garcia (1997) and Wang and Lane (1997). Others prefer to approximate the obstacles as polygons, particularly convex types (Mckendrick 1989, Liu *et al.* 2000). Convex attributes are vital in simplifying the implementation of various motion planning algorithms while fostering faster convergence.

Unsatisfied with the limitation of simple polygonal representation, Lane *et al.* (1998) resorted to using a constructive solid geometry(CSG) method, a technology extensively used in the CAD industry. The CSG method allows explicit representation of objects using simple primitives such as spheres and cuboids via boolean operators: subtraction, intersection and union. One key attribute is the lack of ambiguity between the inner and outer part of the object. To ease the implementation of the optimisation algorithm, Wang and Lane (1997) restricted themselves to using only sphere and ellipsoid primitives. Alternatively, one can try to approximate the seabead surface using a surface modelling technique (Subramaniam and Bahl 1995).

Notwithstanding the above advantages, one of the obvious shortcomings of the geometry map is its difficulty in making inference from noisy measured sensor information which has a great impact on its reliability. Furthermore, a stochastic model can rarely be described in a simple parameterised, geometric manner. Attempts to do so have achieved limited success. Other problems are also encountered such as lack of stability and lack of expressive power to model the object (Dudek and Jenkin 2000). Lack of stability is due to parameters that are sensitive to variation, causing additional erratic model shape changes. On the other hand, lack of expressive power is caused by using oversimplified geometric models which severely restrict their approximating capability.

Topological map

Topology is concerned particularly with the global connectivity of an object by considering how the object is connected locally. A topology map represents the environment as graphs, where nodes correspond to distinct places (landmarks), and arcs represent adjacency or orientation. Fig 2.8 illustrates a hypothetical workspace with landmarks and (b) the topology representation of the workspace. The orientation regions (OR) are used for localisation. In a sense, this information can be considered as a qualitative type. The key to topological representation is its compactness and high immunity to noise since it is less dependent on metric information. This compact representation also facilitates high-level symbolic reasoning for map-building, navigation, planning, and communication.

Since this scheme is not especially susceptible to noise, one obvious application of this

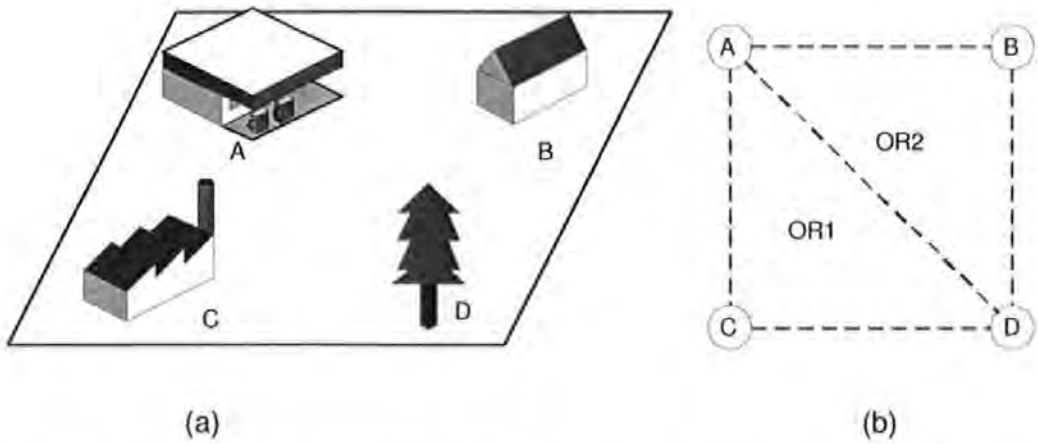


Figure 2.8: (a) Landmarks in workspace (b) Topology representation of the workspace

scheme would be on the problem of simultaneous localisation and mapping (SLAM) (Tomatis *et al.* 2001) to assist an AUV to navigate uncharted waters. In spite of this, its effectiveness is currently being outperformed by a metric approach using an extended Kalman filter (EKF). This scheme is also known as stochastic mapping (Rikoski *et al.* 2002, Carpenter 1998).

In reality, the topology mapping scheme performance is far from outstanding, hence explaining its unpopularity. One particular downfall is its excessive dependence on the presence of landmarks. A landmark can be defined as an individual or a cluster of objects that have distinctive feature relationships. Other desirable criteria are ease of identification, uniqueness and repeatability. Zimmer (2000) advocated embedding local metrical map patches with a globally consistent topological map. Then again, the results tend to be biased as it was simulated using a land based robot in an indoor environment where proper landmarks can easily be identified. These landmarks are particularly difficult to find in a non-homogeneous and 'noisy' environment where an AUV operates, thus severely limiting its usage.

Hybrid representation

Owing to their apparent advantages and disadvantages of each method of representation, some researchers resorted to using hybrid representations such as a combination of metric based and topological paradigms (Simhon and Dudek 1998, Tomatis *et al.* 2001, Zimmer 2000). Hino (1989) utilised an uniform grid scheme for preliminary

terrain representation. A data reduction scheme is introduced to convert the previous map into a contour like map, with significant memory saving. Nevertheless, the transformed contour map lacks flexibility for further modification. Zanoli and Affaitati (1999) adopted spatial decomposition techniques for local map building, but all AUV obstacle avoidance tasks are conducted using a geometry model of the environment. To conclude, hybrid representation provides the user with better flexibility, simplicity and robustness that is difficult to achieve using an individual type representation, however extra precautions are required in synchronising and maintaining the data integrity in between different representations.

2.4 Rules of the Road Relevant to an AUV

The 'Rules of the Road' for AUVs pertain to a set of protocols or regulations applied to assist in tackling a collision predicament. Ironically, both marine vehicles and aircraft employ very similar regulations. The idea of incorporating these rules into an automatic collision avoidance system is not entirely new and has been essayed by various researchers (Pietzykowski 2002, Tran *et al.* 1997). Even so, their implementations are restricted to surface vessels. These selection of guidelines, as presented below, are derived from the International Regulations for Preventing Collisions at Sea (Brown 1983, Cockcroft and Lameijer 2001).

- **Rule 2** *Responsibility*, requires that "due regard shall be given to all dangers of navigation and collision". This rule allows an AUV to depart from all the rules as necessary to avoid the immediate danger of collision.
- **Rule 4** *Look-out*, requires that "every vessel shall at all times maintain a proper lookout by all available means appropriate in the prevailing circumstances so as to make a full appraisal of the situation and of the possible risk of collision." This is the primary task of the obstacle detection unit, where the primary lookout sensor employed is the sonar. There is even a suggestion that future AUVs shall be equipped with a system similar to the identify friend or foe (IFF) unit commonly used in military aircraft.
- **Rule 6** *Safe Speed*, requires that "every vessel shall at all times proceed at a safe speed so that she can take proper and effective action to avoid collision

and be stopped within a distance appropriate to the prevailing circumstances and conditions." The speed of an AUV will be determined by these factors: the detectability, traffic density, manoeuvrability of the vessel with special reference to stopping distance and turning ability, the state of the sea, current, and proximity of navigational hazards. Slow speed, however, can affect the manoeuvrability of AUVs.

- **Rule 7** *Risk of Collision*, states that "every vessel shall use all available means to determine if risk of collision exists; if there is any doubt, assume that it does exist."

- **Rule 8** *Action to Avoid Collision*, states that "changes in course and speed shall be large enough so as to be readily apparent to the other vessels. If necessary to avoid collision or allow more time to assess the situation, a vessel shall slacken her speed or take all way off by stopping or reversing her propulsion. A vessel which is required not to impede the passage of another vessel shall take early and substantial action to allow sufficient sea room for the passage of the other vessel." Stopping and reversing the propulsion can, however, be problematic for a majority of AUVs which are underactuated and not neutrally buoyant, for example the loss of rudder effectiveness in low speed can induce higher collision risk instead.

- **Rule 14** *Head-On Situation*, states that "vessels which are approaching head-on shall alter course to starboard (right-hand-side) so each will pass port (left-hand-side) to port."

- **Rule 15** *Crossing Situation*, states that "when two vessels are crossing so as to involve risk of collision, the vessel which has the other vessel on her starboard side shall keep out of the way, and shall, if the circumstances of the case admit, avoid crossing ahead of the other vessel."

Note that for rules 8, 14 and 15, the general right-of-way rule states that the least manoeuvrable vessel has the right-of-way. For the case of a surface vessel, it is apparent that these manoeuvres occur in the planar domain. Whilst AUVs operate in a 3-D domain, and for the moment, their avoidance manoeuvres are still limited to only planar motion owing to the restriction imposed by the conventional 2-D obstacle avoidance sonar. The airline industry is currently employing the Traffic Alert/Collision Avoidance System (TCAS). The concept is to create a virtual bubble around

the aircraft and alerting the pilot if there is any incursion to the protected zone around the aircraft. The most primitive system, TCAS I only alerts the pilot on incoming threats and is referred as a tactical advisory (TA) system. TCAS II incorporates further feature enhancement to actually propose resolution advise (RA) in order to synchronise the vertical avoidance manoeuvre of both aircraft (Abdul-Baki *et al.* 1999). This is attained by transmitting and receiving of interrogating signals, using a transponder, with the nearby aircraft. The latest, TCAS III, provides the pilot with a horizontal manoeuvre resolution advisory capability. The airline TCAS implicates the importance of a system or regulations that can propose complementary manoeuvres as such that a collision can be avoided. Hence to be truly effective, a consensus of these rules need to be implemented in all AUVs.

2.5 Motion Planning Techniques

Both motion planning and path planning can be defined as a problem of the form: Given a configuration space, find a continuous sequence of configurations that leads from a start to a goal configuration while respecting certain constraints. However, the distinction is that, motion planning tends to denote the generation of time parameterised solutions (trajectories) while, on the other hand, path planning neglects the time parameter. Simply stated, path planning does not take into consideration the vehicle dynamics. Both these terms will be used alternately depending on their suitability in a different context.

Owing to the inclusion of differential constraints, motion planning can also be considered as a search in a state space for a control input that can bring a system from an initial state to a goal state. Employing this perspective, one can directly associate a motion planning problem to a control engineering problem. Indeed, this promotes better problem assimilation and understanding. There is no dearth of literature regarding the theory of motion planning (Latombe 1991, Fujimura 1991, Laumond *et al.* 1999, LaValle 2005). Thus, only a limited number of motion planning techniques that are associated with AUVs will be surveyed. Broadly speaking, motion planning approaches can be classified into three fundamental categories: Cell decomposition, roadmap, and potential fields.

2.5.1 Cell-decomposition

One of the most popular motion planning schemes is the cell-decomposition. It is strictly related to the spatial decomposition scheme for workspace presentation. The fundamental idea is to represent the adjacent relation between the free cells with efficient structures such as a connectivity tree or a graph. They are then searched from the start to the goal state to find a sequence of states (path), that connects both the start and the goal state together. Various search algorithms that are based on dynamic programming exist for performing this routine. A few of the prevalent ones are; breadth-first search, depth-first search, best-first search, A^*, single source shortest distance algorithm (Dijkstra's algorithm) (Dijkstra 1959) and unlimited variants.

The breadth-first search entails searching the neighbourhood cells, and expanding the list as it goes. While the depth-first search keeps probing in one path until an end is met, before trying the alternatives. Both search algorithms are exhaustive (complete), which means ultimately, all free space will be searched for solutions. For cases where multiple solutions exist, and optimality (shortest distance) is not a concern, the depth-first search tends to have a lower memory requirement while providing a quicker answer. However, a depth-first search can be deceived into searching long list of cells, or states, even when the goal may be very near. The Dijkstra's algorithm shares some resemblances with a breadth-first search, but unlike a breadth-first search, all the cells are encoded with distance from the goal which assists it in finding the shortest path. This search algorithm was applied in complement with a binary spatial partitioning scheme for the global path planning of the *Umihico* AUV (Arinaga *et al.* 1996).

Nonetheless, in most circumstances, searching the entire free space can be too computationally demanding. Unsatisfied with the performance of the former systematic search algorithms, some heuristically enhanced versions have been devised. Heuristic information is normally encoded in an evaluation function (cost function). The distance to the Euclidean path (line-of-sight) from start to goal state is chosen as in the case of a best-first search. This scheme is efficient and fast when a proper evaluation function is provided, but for cases when this cannot be found, then its performance degrades significantly. The best-first search tends to provide suboptimal solutions since it neglects the cost of the solution path. The A^* (Hart *et al.* 1968) is a combination of the best-first search and the breadth-first search, which attempts to find a solution that minimises the total length of the solution path. The A^* method

takes into account both the distance from the cell in question to the finish, and also the total distance taken from the start to the current cell. The evaluation function can be written as below:

$$f(node) = g(node) + h(node) \tag{2.1}$$

Where $f(node)$ is the total cost, which is the evaluation function, $g(node)$ is the path cost to the current cell and $h(node)$ is an estimate of the remaining cost to the goal state. A^* is guaranteed to find the shortest path if the $h(node)$ does not overestimate the cost to the solution.

Hyland (1989) incorporated a three dimensional A^* path planner with a reflexive obstacle avoidance module in his AUV simulation. The entire path is replanned by the path planner every time the vehicle completes a flat turn manoeuvre. Also, Hyland (1990) provided a detail comparison between the breadth-first and the A^* search method for an AUV obstacle avoidance task. However, the results were inconclusive, as neither the A^* nor the breadth-first search showed any significant advantages in this case. Others like, Allison *et al.* (1989) proposed a sensor-based exploration approach where a 3-valued occupancy grid is coupled with the A^* algorithm evaluation function that is biased to search the unexplored region. It must be noted that these algorithms mentioned above, do not function optimally for cases when the environment is dynamic, partially known or unknown. The D^*, also known as Dynamic A^* (Stentz 1994), has been developed to address these issues. Owing to the nature of the problem, a substantial difference in performance can be obtained if one sets the initial search point as the start or the goal. Hence, some authors (Arai *et al.* 1998) prefer to use a bidirectional motion planning approach. Arinaga *et al.* (1996) adopted this method for local path planning of the *Umihico* AUV. Their method involved moving the real AUV forward at the start point and a virtual AUV backward at the goal point simultaneously. Upon meeting, the real AUV is assigned to track the sequence of configurations created by the virtual AUV. Their method does necessitate a reflexive module for obstacle avoidance.

One of the apparent limitation of these search algorithms is the unrealistic computational requirement as the number of cells increase, a phenomenon known as the 'curse of dimensionality' or 'combinatorial explosion'. This might be caused by the increase of configuration space dimensionality or the scene complexity. For an heuristically

enhanced algorithm like the A^*, its performance is highly dependable on the selected evaluation function, which can be difficult to define for complicated problems.

2.5.2 Potential field

The potential field method utilises a very interesting approach. In essence, an artificial potential field is defined to emulate the space structure surrounding the vehicle (Krogh and Thorpe 1986). It consists of representing the goal with an attractive field and the obstacles with a repulsive field, as shown in Fig 2.9. A new field emerges through the interaction of both the former fields. Eventually, the vehicle is required to just follow the local gradient of the new field to reach the goal.

The mathematical equations pertaining to the potential field method can be found below. The equations below are used to generate the simulation results as illustrated in Fig 2.9. The related mathematical definitions are listed as follow (Khatib 1986): The field of artificial forces $\vec{F}(q)$ in configuration space, C is produced by a differentiable function $U : C_{free} \rightarrow R$, with:

$$\vec{F}(q) = -\vec{\nabla}U(q) \tag{2.2}$$

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{2.3}$$

where $\vec{\nabla}$ denotes the gradient operator, U_{att} is the attractive potential associated with the goal configuration, q_{goal} and U_{rep} is the repulsive potential associated with the C-obstacle region.

The attraction field can be formulated as below:

$$U_{att}(q) = \frac{1}{2}\xi {\rho_{goal}}^2(q) \tag{2.4}$$

The attraction force can be formulated as below:

$$\vec{F}_{att}(q) = -\xi(q - q_{goal}) \tag{2.5}$$

where ξ is a positive scaling factor, ρ is the Euclidean distance function, q is the current configuration and q_{goal} is the goal configuration.

The repulsion field can be formulated as below:

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)^2 & \text{if } \rho(q) \leq \rho_0 \\ 0 & \text{if } \rho(q) > \rho_0 \end{cases} \tag{2.6}$$

The repulsion force can be formulated as below:

$$\vec{F}_{rep}(q) = \begin{cases} \eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(q)}\vec{\nabla}\rho(q) & \text{if } \rho(q) \leq \rho_0 \\ 0 & \text{if } \rho(q) > \rho_0 \end{cases} \tag{2.7}$$

where η is a positive scaling factor, $\rho(q)$ is the distance to the obstacle and ρ_0 is the distance of influence.

Fig 2.9(a) shows a simulated workspace representation of the vehicle. Using Equation 2.4, an attractive field for the corresponding goal is simulated in Fig 2.9(b). Notice that the goal is the global minima, which is true for an ideal case. Using Equation 2.6, Fig 2.9(c) shows the repulsion field exerted by the obstacles. Ultimately, Fig 2.9(d) illustrated the combined repulsive and attractive potential field as stated in Equation 2.3. One major advantage of this method is its low computational requirement which makes it very suitable for real-time implementation.

Yoerger *et al.* (2000) applied a potential field local planner in the *Benthic Explorer* for a fine-scale rugged sea-floor surveying mission. The implementation is restricted to using an asymmetric potential field to alter the vehicle's forward and vertical speed. One problem which is inherent to the potential field method is its tendency to get trapped in a local minima. For this reason, it is normally used only as a local path planner, and in most implementations, it is combined with another global path planner that will be invoked when trapped. Its performance is also strictly linked to suitable definitions of heuristic potential functions, and this is not easily found when confronted with natural obstacles and differential constraints. Warren (1990) proposed a hybrid method that involves two major stages. The first stage generates a preliminary straight path from current to goal configuration. Then in the second stage, a method of path relaxation is introduced, the path is iteratively modified under the influence of the adjacent potential field in order to produce a feasible path. Warren argued that by considering the problem in such a global approach, the tendency

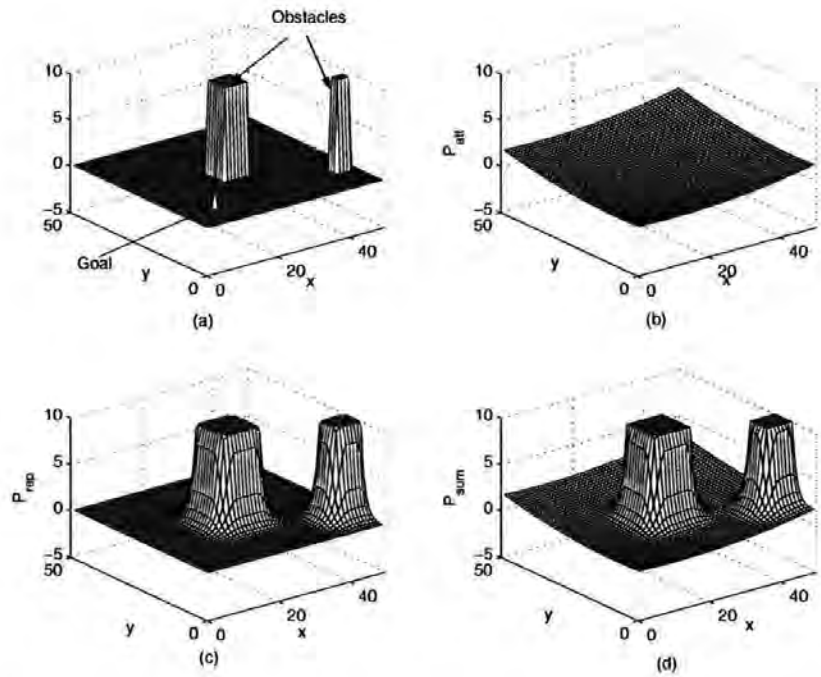


Figure 2.9: Potential Field Simulation (a) Workspace (b) Attraction Field (c) Repulsive Field (d) Combined Field

of local minimum entrapment is significantly reduced. Instead of the conventional gradient descent method, Lane and Trucco (2000) reported using a preliminary tree search technique, to be specific, the best-first search to find the global minima. Local minima traps are avoided using the backtracking feature of this algorithm.

Some researchers encourage the use of a minimum-free function that is based on a harmonic potential field. The problem formulation is analogous to solving a fluid flow problem such as fluid moving from the source location to the goal (Li and Bui 1998, Louste and Liegeois 2000). Nevertheless, this technique is not practical for a high dimensional problem as it is too costly in terms of computation demand.

2.5.3 Bug algorithm

The bug algorithm, also known as tangent bug or edge follower is a particularly simple and yet remarkable path planning scheme. The main principle behind it is to trace the obstacle boundary and this is continued until the obstacle no longer blocks the

desired path (Kamon *et al.* 1995)(Fig 2.10).

Fundamentally, the algorithm constitutes of only two modes: (1) moving to the goal and (2) circumnavigating the obstacle. Note that this algorithm does not need any *a priori* information regarding its environment. Furthermore, it is guaranteed to find a solution if it exists. This makes the bug algorithm suitable for dealing with unknown environments. Bennet and Leonard (2000) implemented the algorithm in the *Phoenix* AUV. A forward looking sonar is used to detect the obstacle boundary, then it is approximated by aggregating piecewise linear lines before applying the bug algorithm. Alternatively, Cornforth and Croff (2000) applied a wall-following algorithm in the *Autolycus* with the help of a side-facing sonar. Unfortunately, their current results were unsatisfactory but they anticipated further improvement can be realised by empirically tuning the controller gain. They envisaged using the *Autolycus* in environment-sensitive navigation. Better still, Laubach and Burdick (1999) devised a more memory efficient approach that utilises only obstacle boundary endpoints. Their concept, however, is exemplified in a planetary exploration rover and not an AUV. This algorithm, although simplistic in concept, is extremely difficult to be implemented in practice. Firstly, the influence of sensor drift in a dead reckoning system tends to limit its effectiveness. In addition, this algorithm also assumes that the vehicle is holonomic and operating in a static environment which is not entirely true for the case of an AUV.

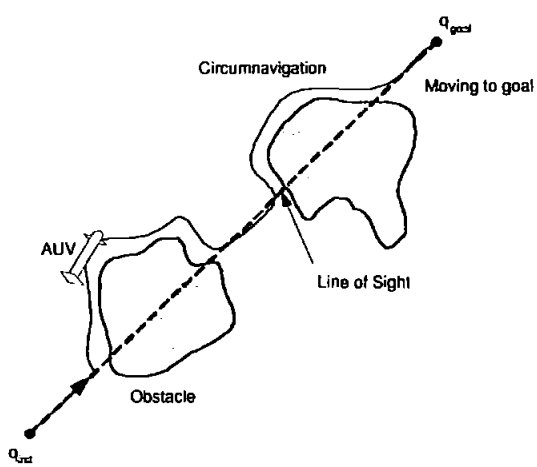


Figure 2.10: An AUV employing the bug algorithm to navigate the environment.

2.5.4 Evolutionary computation (EC)

Evolutionary computation encompasses several types of heuristic and stochastic optimisation schemes that are fundamentally based on the concept of natural selection. Some of the proposed schemes are the evolutionary algorithm (EA), genetic algorithm (GA), evolutionary programming, evolutionary strategy and artificial life. The EC has shown significant capability in solving complicated, highly constrained, large scale optimisation problems that have discontinuities on the response surface. Unlike the potential field method, EC is known to be highly resistant to becoming trapped in local minima. Another exceptional attribute of EC is its ability to offer solutions whenever it is interrupted.

Schultz (1991) proposed using the GA for on-line collision avoidance and local navigation of an AUV. Promising simulation results of an AUV successfully navigating through both a static and dynamic minefield are presented. Similarly, Fogel and Fogel (1990) simulated 2-D optimal routing of multiple AUVs using a EA. Their simulations, although confined to only two dimensional routings, still managed to demonstrate intriguing results. The AUV exhibited very intelligent behaviour by trying to avoid the detection region and, if that was not possible, the AUV proceeded at slower speeds to remain stealthy and speed up when it was a distance away from the detection site. Multiple AUVs cases are also addressed. They argued that sophisticated genetic operators such as crossover tends to disrupt the link between parent and offspring as coding structures become large. Sugihara (1998) suggested a local GA 3-D path planner that is capable of functioning in a partially known environment for the *SAUVIM* AUV. He employed a method of discretisation, where the 2-D maps are partitioned into cells, and each corresponding cell is then encoded with a binary string as a sequence of pairs of direction and distance. Then, the three 2-D, sequences of connected cells (paths), one in each respective plane, xy-plane, xz-plane and yz-plane, are merged via projection, into a single 3-D path.

There have been several attempts to hybridise evolutionary computation with other algorithms. Dozier *et al.* (1998) combined fuzzy inference along with tournament selection to select the best candidate paths based on several criteria. They claimed that the methodology does not only provide significant performance enhancement, but also obviates the need of explicit multi-objective evaluation function development. Instead, Vadakkepat *et al.* (2000) endeavoured to merge a potential field planning

method with evolutionary programming to derive an optimal potential field function. The resulting algorithm is not only capable of solving the local minima problem but also handling dynamic obstacles.

One glaring weakness inherent in all evolutionary computation algorithms is their high computational requirement. Although, some authors purported to solve it using a distributed processing approach, this is not applicable in the case of an AUV, due to the energy limitation of the batteries. Furthermore, their long convergence time, makes them unsuitable for a highly dynamic environment. Other weaknesses include difficulty in finding the exact global optimum and their performances are highly dependable on how the problem is structured and encoded.

2.5.5 Visibility graph

The visibility graph is a subset of the roadmap approach to path planning. The operation of this scheme can be depicted as following: initially, all the vertices of all polygonal obstacles including the start and goal, that are in a line-of-sight with respect to each other are connected as shown in Fig 2.11.

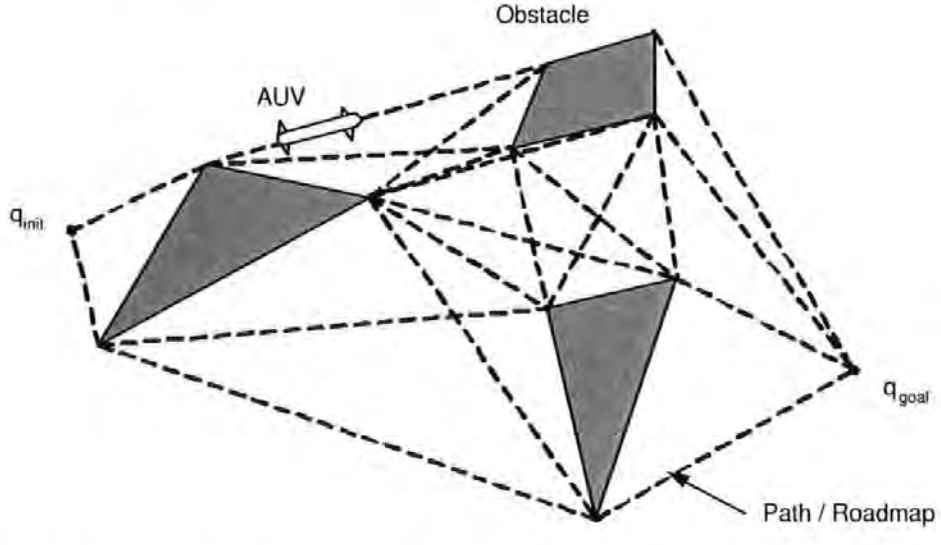


Figure 2.11: An AUV employing the visibility graph technique to navigate the environment.

This process is simplified if one limits the environment with only convex obstacles. Then, by representing the vertices as nodes, and the path as an edge, any tree search

algorithms, as elaborated in the cell decomposition section, can be used to find the shortest path. Unfortunately, one obvious disadvantage of using a visibility graph is the assumption that all of the obstacles are known. Furthermore, a visibility graph has the tendency to generate paths that are very close to the obstacles edges. One simple solution is to 'patch' the obstacles in order to take into account of the vehicle geometry. However, this is not a trivial process if the vehicle considered is underactuated. Mckendrick (1989) applied a visibility graph method in an unknown 2-D environment with convex polygonal obstacles. To be realistic, the AUV was simulated with a limited sensor range. An exploration phase is then required for information acquisition. The simulation demonstrated that the path is highly non-optimal, taking long detours and as such a simple bug algorithm easily surpasses its performance.

2.5.6 Probabilistic roadmap planner (PRM)

This is still a relatively new approach to path planning, where the construction of the roadmap is done probabilistically instead of deterministically. The main concept is to generate a number of nodes (vertices) randomly, eliminate those nodes in the obstacles, and then, connect all the adjacent nodes with straight lines. The primary reason that only adjacent nodes are connected is to avoid saturating the configuration space with too many paths. Later, the resulting roadmap is searched from the start point to the goal point for the shortest path (Fig 2.12).

Unlike other motion planning methods, its randomised nature tends to make its performance less susceptible to the effect of configuration space dimension (Overmars 2002). However, it does compromise solution optimality for enhanced robustness. Consequently, this method generally produces suboptimal solutions. PRM is also notoriously known for its long running times and difficulty in finding a path in a configuration space that has a small passage. Therefore, some heuristically enhanced PRMs such as visibility PRM (Simeon *et al.* 2000), lazy PRM (Bohlin and Kavraki 1998), obstacle based PRM (Amato *et al.* 1998) and Gaussian sampling PRM (Boor *et al.* 1999) have been recommended to improve the generic PRM performance. These methods mostly differ from their sampling strategies. Fox *et al.* (2000) and Garcia (1997) suggested an enhanced PRM that shares some similarity with the lazy PRM.

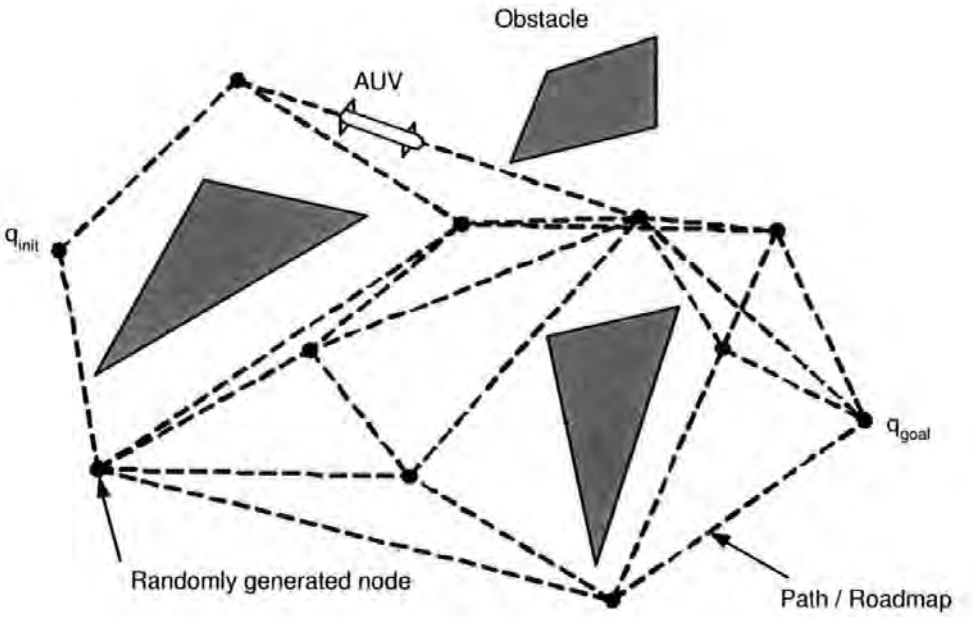


Figure 2.12: An AUV using the probabilistic roadmap method to navigate the environment.

They envisaged using it as a generic path planner for an AUV. Their method relies on generating a line-of-sight path from the initial point to the goal point. When an obstacle intersection occurs, a few points are generated at the obstacles vicinity to reroute the path. This process is then repeated until a solution is found or a time-limit is reached. Their scheme assumes a static environment, thus a reactive planner is required for avoiding a dynamic obstacle.

2.5.7 Rapidly-exploring Random Tree (RRT)

The majority of path planning algorithms, as mentioned above, only take into account the algebraic constraints (caused by obstacle) and not differential constraints (caused by kinematics and dynamics). The paths generated are typically rigid with abrupt directional changes, making it not viable for vehicle actuator operations due to saturation. This becomes an impetus for a search for a scheme that can address both the path planning and control issues simultaneously. Several potential benefits arise from the integration of the path planning and control process. Assuming no disturbance, the paths generated are guaranteed to be reproducible in practice. A linked optimisation for both path distance and control effort can also be achieved.

Cases of differential constraints can easily be incorporated, thus extending its application into fault-tolerant and reconfigurable systems for an AUV (Perrault and Nahon 1999, Sutton *et al.* 2001).

Most systematic search algorithms do not function well in high-dimensional space. This has prompted the introduction of the RRT (LaValle 1998, LaValle and Kuffner 2000), which can be considered as an incremental form of PRM and is designed to search efficiently non-convex high-dimensional spaces. It possesses a few fascinating properties: (a) It is biased to the freespace and exploits a probabilistic search method, (b) It has also been proven to be probabilistically complete (Cheng and LaValle 2002), (c) The simplistic nature of the algorithm facilitates performance analysis and lastly, (d) It allows one to take into account both algebraic and differential constraints simultaneously, which is vital for motion planning. A detail explanation of RRT operation from an algorithmic viewpoint can be found in Section 4.3.

Cheng *et al.* (2000) applied the RRT to optimise the trajectories of autonomous automobiles and spacecraft. The simulations show the viability of the method. Toussaint (2000) tried to combine motion planning using the RRT with nonlinear control employing the H^{∞} technique for an underactuated vehicle. He utilised an H^{∞} filter for improving the planned motion of the vehicle and also addressed multiple vehicles planning problems. But the simulations are limited to planar motion. The RRT has also been applied to solve nonlinear control problems in hybrid systems. Frazzoli *et al.* (2002) provided some realistic simulations of unmanned helicopter motion planning that employed the RRT. Unlike other path planning algorithms, he mentioned that the RRT is capable of exploiting fully the manoeuvrability of the helicopter.

Nonetheless, the RRT is not without problems. Firstly, as a novel algorithm, its capability is still not well characterised. Furthermore, its performance is highly sensitive with respect to the chosen metric. An incorrect metric will substantially deteriorate its performance. Cheng and LaValle (2001) described a technique to render the RRT less sensitive to the metric effect. In all of the experiments and simulations above, a known environment is assumed, and this is unrealistic for AUVs. Hence, supporting the fact that there is still room for further improvement.

2.6 Reflexive Avoidance Techniques

Reflexive avoidance techniques provide an AUV with a failsafe mechanism in the case of motion planner failure. The failure can either be a system malfunction or a failure to meet the predefined time constraint, which is a more common occurrence than the former. Technically, these techniques are fundamentally based on the reactive control approach (Brooks 1986), where the information from the sensors is sent directly to the actuator without passing through the high-level modules. This makes them amazingly fast and capable of handling dynamic environments especially in cases where *in situ* response is needed. Unfortunately they suffer from the identical problems that can be found in reactive control systems. Some of the problems are, highly non-optimal action, non-deterministic performance and lastly they are prone to get trapped in a canyon like environment.

2.6.1 Neural network

The neural network approach has been intensively researched over the last 20 years, and it still ongoing. The core of the neural network concept is based on a mathematical emulation of simplified human brain mechanisms. One obvious benefit of the neural network, is its intrinsic ability to model a very complex, multi-input-multi-output, and strongly coupled nonlinear system such as an AUV. It is also highly renowned for its 'generalisation' capability. Unlike other algorithms that are programmed, a neural network [1] is trained by exposing it to related input/output data. Properly trained, the neural network can provide an elegant solution to a very challenging problem. DeMuth and Springsteen (1990) proposed a neural network based obstacle avoidance controller for an AUV. They used two neural networks, one for a static object and another for a dynamic object classification. Then, a Boolean combiner is employed to reconcile the appropriate manoeuvre to be taken. Paradoxically, their neural network controllers were not trained but the weights were heuristically determined. Clearly, this is only applicable for a very simple case. In trying to exploit the adaptive nature (training ability) of a neural network, Sayyaadi *et al.* (2000) applied a stochastic real value reinforcement learning method to the collision avoidance controller of the *Twin*

[1] The majority of neural networks such as the popular multi-layer perceptron and radial basis networks do require supervised training.

Burger. They divided the obstacle avoidance mission into a targeting behaviour and an avoiding behaviour. Nonetheless, no tangible results were given concerning the avoiding behaviour performance as the research is still at a premature stage.

Interestingly, one tends to find more neural network applications in low-level controller design than at the higher level (Ishii *et al.* 1995, Wettergreen *et al.* 1999). This could be caused by the black box characteristic of neural network, which precludes vital information extraction that can be crucial for problem understanding. Furthermore, high-level controller typically exerts more influence over the entire system performance where small error tends to amplify quickly. This in turn demands a transparent system for analysis purpose which is not offered by such a network. The training processes can also be very time consuming, depending much on the 'suitableness' of the selected training data. To foster rapid convergence, another data pre-processing stage is also found to be compulsory.

2.6.2 Virtual force field (VFF)

The virtual force field method tries to simulate an artificial force field which can be two or three-dimensional, surrounding the vehicle. Thus any contact with neighbourhood objects will cause deformation to the force field. The aim here is to minimise these deformations by locally modifying the control vector. Its computational efficiency and fast reaction make it a valuable technique in a dynamic environment. Recalling the fact that the TCAS employed by the aviation industry also utilises a form of virtual bubble which is a variant of VFF. Zapata and Lepinay (1996) addressed the collision avoidance and bottom following problems of an AUV using a VFF scheme. Their results, however, are confined to only computer simulations.

VFF shares some similarity with the potential field method. Both trying to simulate the interaction of an artificial field between objects and the vehicle. However, VFF only considers the forces in a limited neighbourhood domain thus it is highly susceptible to being trapped in local minima, worst still it can also lead to unstable behaviour of the vehicle when surrounded with obstacles or travelling in a narrow passage. To solve this problem, a high level planner is usually introduced.

2.6.3 Vector field histogram (VFH)

The vector field histogram (Borenstein and Koren 1991) was developed to solve some of the problems of the VFF algorithm concerning detail spatial distribution information loss. The VFH is a data reduction process algorithm that can be decomposed into three distinct phases. The first phase entails representing the vehicle workspace as a two dimensional grid. The second phase involves constructing the vehicle surrounding into local polar histogram form where each sector represents obstacle density. The last phase involves a selection of the sector of the lowest obstacle density and alignment of vehicle heading to the selected sector. Its performance in most circumstances is better than the VFF (Koren and Borenstein 1991).

Williams *et al.* (1990) advocated a three-dimensional collision avoidance controller that has the intrinsic functionality of VFH. They exploited a merit function that defines a field that takes into account obstacle bearing and distance as well as the vehicles own heading, depth and the goal direction. For the three dimensional case, a data reduction process is performed to transform the presentation into an image showing different obstacle density. Consequently, the vehicle is just required to align its heading vector to the lowest obstacle density area. Antonelli *et al.* (2001) sought to integrate the VFF and geometrical approach proposed by Hyland (1989), and implemented it in the *RAIS* AUV. This approach takes into account the polar information instead of only the Cartesian, making the algorithm very similar to VFH. They also employed a high level path planner to detect the high risk area that can trap an AUV.

The VFH tends to take into account of the workspace Cartesian information, hence it is less affected by the local minima entrapment problem as suffered by the VFF method. However, this does not mean that it is impervious to local minima entrapment. Furthermore, by discarding all explicit distance information and representing them implicitly, the vehicle is sometimes deceived into assuming that there is no clear sectors when it is surrounded with only distanced obstacles.

2.6.4 Fuzzy logic

Fuzzy theory was introduced by Zadeh (1965) as an alternative technique for tackling complicated problems that are difficult to solve using conventional differential equation based approaches. In essence, fuzzy logic is a rule based, multi-value logic inference system that attempts to take into account the uncertainty and imprecision of the real world. Its intrinsic operational principle bears substantial resemblance with human cognition. In fact, the fuzzy logic ability to quantified abstract expert knowledge has made it a choice in solving complicated systems. Consequently, the control decisions of an expert can be formulated into an algorithm to control the desired plant. One example of fuzzy rule base in a collision avoidance context is :
IF *Target Direction* is *Left* AND *Target Range* is *Very Near* THEN *Heading* is *Hard Right.*
Clearly, the rule above is self-explanatory which facilitates problem understanding. As such, a set of rules can be promptly constructed without resorting to complex mathematical techniques. Its transparent nature and excellent immunity to both noise and error also contributes to its popularity.

Shinjo and Graeme (1995) suggested a collision avoidance system that is based on a combination of sensor-based navigation and fuzzy logic control. The fuzzy logic inference system provides a mapping framework to transform the acquired object information such as range, position, and size, into the respective control commands for heading and vertical movement of the vehicle. A short-term memory is also used to store successive obstacle avoidance processes with the objective to reduce abrupt changes or chattering of the control command outputs. This is achieved via reducing the degree of membership of the last executed fuzzy conclusion in order to reduce its dominance. The analysis of the algorithm was performed on a full six-degrees of freedom *Ocean Voyager* AUV simulator. Instead of encoding the problem directly as in former approach, Vasudevan *et al.* (1995) attempted a hybrid reasoning scheme by aggregating fuzzy rule sets and case-based reasoning to function as a high-level dynamic path selector. In what was called Reasoning from Fuzzy Indexed Cases Scheme (REFIC), it fundamentally exploits the *a priori* information such as the prestored cases to assist in determining a promising vehicle heading and also in selectively activating a subset of navigational behaviours. The simulated example proved to be very robust in navigating in the presence of noisy sensor data and cluttered obstacle distributions.

Liu *et al.* (1999) tried to tackle AUV navigation in an unknown environment by creating a virtual boundary and incorporating some heuristic rules via fuzzy logic. The vehicle emergence behaviour turns out to be very similar to the bug following algorithm. However, the effects of local minima and sea current were neglected in the simulation. Ridao *et al.* (2001) applied a collision avoidance controller in the *Garbi* AUV using a combination of VFF and fuzzy logic behavioural encoding techniques. The implementation is however limited to the horizontal plane. The vehicle is surrounded with several circular force fields of varying radii and each particular region is then mapped into the corresponding behaviours such as goto, spin, avoid, keep depth and avoid bottom. A simulation study has verified that the vehicle is able to exhibit 'intelligent' manoeuvres such as circumnavigating and escaping from a canyon like trap.

Although, many praises can be made regarding a fuzzy logic system, there are also an equivalent amount of criticisms. Owing to the heuristic method which the fuzzy logic paradigm is fundamentally based upon, a multitude of incoherent and diverse viewpoints exist regarding the types and fuzzy operators used. The lack of a solid framework also tends to make fuzzy logic appear to be an *ad hoc* approach to finding a solution. Although, a fuzzy system is renowned for its transparency property, it is virtually mathematically intractable and can complicate analysis.

2.7 Concluding Remarks

This chapter presented several a myriad of techniques for the design of a collision avoidance system for AUVs. An overview of AUV control architectures and individual collision avoidance system submodules have been presented. The obstacle detection module can be divided into four distinctive subunits; a forward looking sonar, a signal processing submodule, a map builder and a navigation submodule. On the other hand, the obstacle avoidance module comprises of the reflexive, motion planning and trajectory tracking subunits. Again, to reiterate, it should be stressed that to formulate a meaningful and realistic thesis research project, there should be limitations on the scope of the research. For this reason, the reflexive, navigation and multi-target tracking submodules have been omitted.

Techniques employed in the ocean, land and aerospace domains were explored. Their advantages and disadvantage were outlined. It has been shown that the collision avoidance system plays the vital role in bringing autonomy to the whole system.

In regards to the motion planning context, the majority of the motion planning methodologies mentioned are, in fact, considered to be path planning methodologies. They do not take into account the dynamics of an AUV. Systematic search techniques, although, extremely popular in the robotics community. Unfortunately, they do not function well in higher-dimensional search space which is found so commonly in most practical systems since they suffered from the state-explosion effect. The above discussion hinted that the recently developed probabilistic based algorithms, with their strong immunity to state-explosion effect, could provide an interesting topic of research. This resulted in the formulation of enhanced RRT algorithm in Chapter 4. In Chapter 5, the following algorithm is further improved by combining it with the technique that is based on system dynamic quantisation.

Since the motion planning algorithm is, in essence, a feedforward planner, any perturbations to the system, inevitable in practice, will aggravate the system performance. This strongly suggests that a robust trajectory tracker is needed. Therefore, it provoked an investigation into the feasibility of using a linear quadratic regulator or a nonlinear state dependent Riccati controller as a trajectory controller. Details can be found in Chapter 6.

The one major problem with the contemporary forward looking sonars is that they are not specifically designed for the AUV obstacle avoidance purpose. Most of these sonars employed by AUVs are 2-D mechanical scanning variants commonly employed by surface vessels. Numerous 3-D sonars exist but their costs are prohibitive. As such, the development of a novel, cost-effective, energy-efficient obstacle avoidance sonar should be forthcoming in order to address this critical issue. This study will employs the AT500 sonar (Robinson *et al.* 2003) from J&S Marine, that is especially designed to cater for this task. The sonar data processing and workspace representation submodules will be developed to cater especially for this sonar, nonetheless, the techniques must remain sufficiently flexible to be transferrable to other forward looking sonars. These techniques are detailed in Appendix A.

In conclusion, it is anticipated that with the fusion of these different methodologies,

a robust and computationally efficient collision avoidance system can be realised. In designing such a system, one must consistently bear in mind that an effective collision avoidance system derives its success through the synergistic interaction of submodules, and not because of a particular submodule functionality. Similarly, the application of these methodologies could also offer potential technological advances in the field of AUV collision avoidance while simultaneously benefiting the marine industry. The next chapter will delve upon the mathematical modelling of the AUV and environmental effect for computer simulation and performance evaluation.

Chapter 3

System Modelling

This chapter describes the mathematical models employed in the computer simulation studies herein in order to acquire deeper insights and to evaluate the suitability of the proposed algorithms and controllers. Computer simulation studies allow one to investigate various 'what if' scenarios applied to the model without the need of a physical model. This makes it highly attractive in terms of time and monetary cost aspects. More importantly, some of the tested scenarios can be considered to be highly dangerous and risky to human lives if proved in real time with hardware.

To evaluate properly the system performance, a holistic approach must be sought. One not only requires a plant model that can replicate the system dynamic behaviour as close as possible but also the possibility of evaluating the impact of disturbances to the plant. The disturbances might be induced by the environment and sensor noise. Hence, this chapter is partitioned into two sections, AUV modelling and disturbances modelling. The latter section includes sea currents disturbance and sensor noise disturbance modelling.

Notwithstanding the above, a note of caution here is made as regardless of how promising the simulation results appear to be, one should never underestimate the significance of a physical plant test.

3.1 Mathematical Modelling of an AUV

Underwater vehicles inherently possess very complex dynamics such as nonlinearities, cross coupling, numerous degrees of freedom and underactuated behaviour. These factors arise owing to the effect of the body-medium (vessel-water) interactions and partly due to the physical design of the vessel.

There exists an abundance of literature pertaining to the mathematical modelling of underwater vehicles. As a matter of fact, and not coincidental, most of the AUVs are very similar in terms of shapes and actuation configurations such that it allows the generalised underwater vehicle model to be exploited. Regretfully, to utilise the generalised model, it is necessary to estimate first the body and hydrodynamic coefficients. The evaluation of these coefficients of which there can be more than 100, is a nontrivial process (Healey and Lienard 1993). Certain coefficients, such as the mass coefficients can be obtained simply via a direct measuring process. Others require extensive experiments to be conducted in a test tank with a full-scale physical model of the vehicle equipped with relevant sensors (Prestero 2001). Lately, the use of computational fluid dynamics (CFD) analysis for coefficients estimation has also become increasingly popular (Sayer and Fraser 1998). As a last resort, an estimated guess of the coefficient value can be derived from a library of generic shapes, or coefficient scaling from another AUV (Ahmad and Sutton 2003).

In essence, AUVs are vehicles that operate in a 3-D physical space. This implies that an AUV needs a total of six configuration variables to describe fully its configuration, hence the term six degrees of freedom (DOF). Three of the variables are for linear displacements and another three for angular displacements. In order to describe completely the AUV dynamics, six more variables corresponding to the linear velocity and the angular velocity for each dimension are also needed.

Briefly, the generalised six degree of freedom rigid body equations of motion in vectorial form given by Fossen (1994), can be written as

$$\mathbf{M}_{RB}\dot{\nu} + \mathbf{C}_{RB}(\nu)\nu = \tau_{RB} \tag{3.1}$$

Here $\nu = [u\ v\ w\ p\ q\ r]^T$ is the body-fixed linear and angular velocity vector and $\tau_{RB} = [X\ Y\ Z\ K\ M\ N]^T$ is a generalised vector of external forces and moments

acting on the vehicle in body-fixed coordinate system. Please refer to Fig 3.1 for a clearer presentation of the related variables mentioned. Similarly, this thesis, if not mentioned, employs the North-East-Down (NED) Earth-fixed coordinate system. With reference to Fig 3.2, the NED coordinate convention is popular amongst the aerospace, marine, and navigation communities (Fig 3.2(a)). On the other hand, the robotics community and mathematicians prefer the ENU coordinate convention (Fig 3.2(b)). If one adopts the NED Earth-fixed coordinate convention, one should accord to use 'marine' body-fixed coordinate (Fig 3.2(c)) for the rigid body of interest to avoid confusion.

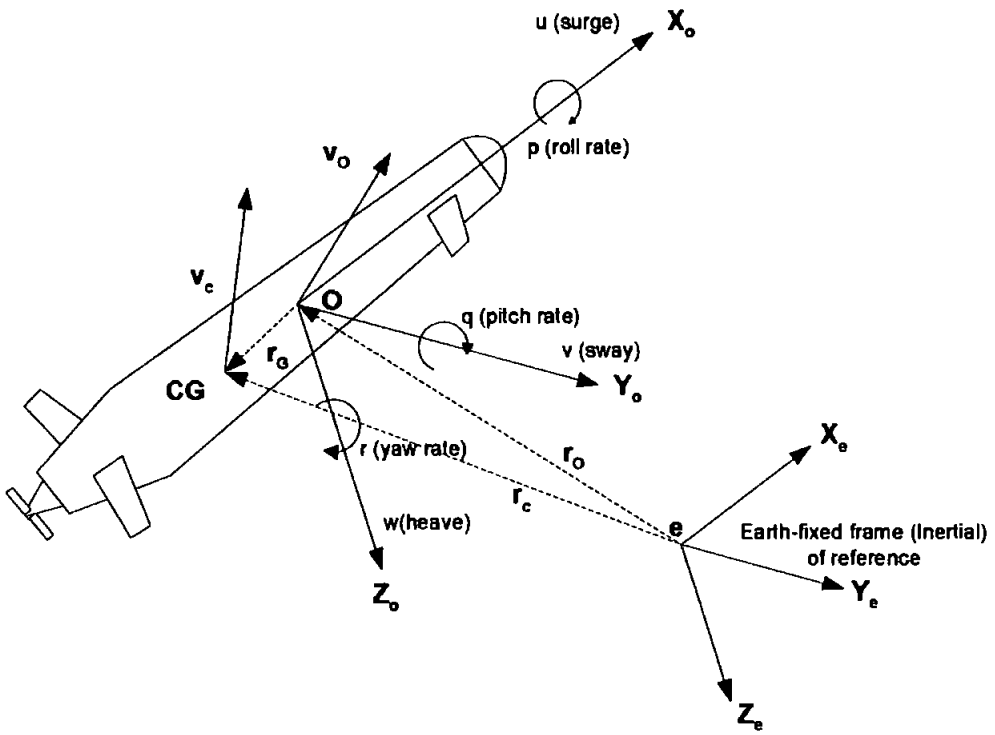


Figure 3.1: The inertial, Earth-fixed non-rotating reference frame $X_eY_eZ_e$ and the body-fixed rotating reference frame $X_OY_OZ_O$. The NED coordinate convention is employed here.

The assumption here is that the hydrodynamic forces and moments on a rigid body can be linearly superposed. The parameterisation of the rigid-body inertia matrix $\mathbf{M}_{RB}$ is unique and it satisfies:

$$\mathbf{M}_{RB} = \mathbf{M}_{RB}^T > 0; \quad \dot{\mathbf{M}}_{RB} = 0 \tag{3.2}$$

Referring to Equation 3.2, the elements in $\mathbf{M}_{RB}$, are mass and moment of inertia

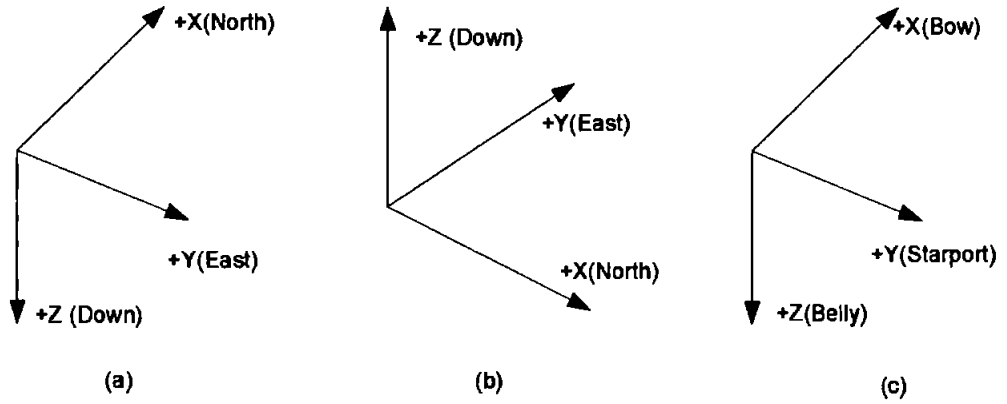


Figure 3.2: (a) North-East-Down (NED) Earth-fixed coordinate (b) East-North-Up (ENU) Earth-fixed coordinate (c) Aerospace and marine body-fixed coordinate

that correspond to the individual axis. These variables cannot have negative values. Assuming that the mass and the moment of inertia of the vehicle are constant through time, which is true for most AUVs without buoyancy depth control, as a corollary the time derivative of the inertia matrix should be $\mathbf{0}$.

The matrix $\mathbf{C}_{RB}$ corresponds to the Coriolis and centripetal forces and moments that can be parameterised to be a skew symmetric matrix i.e.

$$\mathbf{C}_{RB}(\nu) = -\mathbf{C}_{RB}^{T}(\nu)$$

The Coriolis and centripetal forces and moments, are virtual forces that arise because of the formulation of the dynamic equations with reference to the AUV body which is a non-Newtonian frame. These terms only come into effect when the vessel is executing some kind of rotational motion and they contribute nothing when the vessel is in straight line motion. Likewise, τ_{RB} can be expressed as

$$\tau_{RB} = \tau_H + \tau_E + \tau \tag{3.3}$$

where τ_H is the radiation induced forces and moments which includes added inertia, hydrodynamic damping and restoring forces. τ_E describes the environmental forces such as ocean currents, waves and wind. Obviously, in this context, assuming that the AUV is travelling at least 30 metres below the ocean, then only ocean currents are applicable to the AUV. Finally, τ is the propulsion forces from thrusters and control surfaces. The derivation of these forces and moments is not included, please refer to

Fossen (1994) for further details. Alternatively, Equation 3.1 can be put into a more insightful form as

$$\mathbf{M}\dot{\nu} + \mathbf{C}(\nu)\nu + \mathbf{D}(\nu)\nu + g(\eta) = \tau_E + \tau \tag{3.4}$$

where

$$\mathbf{M} \triangleq \mathbf{M}_{RB} + \mathbf{M}_A; \quad \mathbf{C} \triangleq \mathbf{C}_{RB}(\nu) + \mathbf{C}_A(\nu) \tag{3.5}$$

Refer to the nomenclature in the beginning of the thesis for more detail of the individual notation used. Here, the suffix A denotes the added mass version of the particular matrix. Added mass is caused by the mass of water moving with the AUV. $\mathbf{D}(\nu)$ is the damping matrix. $g(\eta)$ is the gravitational and buoyant forces vector and $\eta = [x\ y\ z\ \phi\ \theta\ \psi]^T$ is the vector of position and Euler angles in earth-fixed frame of reference. Conceptually, the above formula is analogous to a simple mass-spring-damper system. The spring behaviour is contributed by the $g(\eta)$ while the $\mathbf{M}$ and $\mathbf{D}(\nu)$ represent the mass and damper effects, respectively.

The matrix $\mathbf{M}$ is needed for explaining the kinetic energy of the total ambient vessel-water system, which is larger than the AUV rigid-body kinetic energy. The kinetic energy is contributed by the fluid motion as the fluid moves aside and close behind the AUV. The $\mathbf{D}(\nu)$ can be attributed to the effects of potential damping, skin friction, vortex shedding and wave drift damping. These forces make the system dissipative, ensuring that the system states are bounded for bounded inputs. The $g(\eta)$ term cannot be neglected if the AUV has a low metacentric height. Those AUVs that have low hydrostatic restoring compared to the inertia forces have the propensity to start rolling and pitching when the actuators are utilised.

Equation 3.1 can also be expanded to yield

$$\begin{aligned}
m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pg - \dot{r} + z_G(pr + \dot{q}))] &= X \\
m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p} + x_G(qp + \dot{r}))] &= Y \\
m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q} + y_G(rq + \dot{p}))] &= Z \\
I_{xx}\dot{p} + (I_{zz} - I_{yy})qr - (\dot{r} + pq)I_{xz} + (r^2 - q^2)I_{yz} + (pr - \dot{q})I_{xy} & \\
+m[y_G(\dot{w} - uq + vp) - z_G((\dot{v}) - wp + ur)] &= K \\
I_{yy}\dot{q} + (I_{xx} - I_{zz})qr - (\dot{p} + qr)I_{xy} + (p^2 - r^2)I_{zx} + (qp - \dot{r})I_{yz} & \\
+m[z_G(\dot{u} - vr + wq) - x_G((\dot{w}) - uq + vp)] &= M \\
I_{zz}\dot{r} + (I_{yy} - I_{xx})rp - (\dot{q} + rp)I_{yz} + (q^2 - p^2)I_{xy} + (rq - \dot{p})I_{zx} & \\
+m[x_G(\dot{v} - wp + ur) - y_G((\dot{u}) - vr + wq)] &= N
\end{aligned} \tag{3.6}$$

The first three equations represent the translational motion while the last three equations represent the rotational motion. Equation 3.6, also known as the general rigid-body equations of motion can be simplified in two ways. The first method is to take the origin of the body fixed reference frame to coincide with the centre of gravity, another way is to choose the origin such that the inertia tensor is diagonal. Referring to the former method, it implies that $r_G = [0\ 0\ 0]^T$ and $I_C = diag\{I_{xxc}\ I_{yyc}\ I_{zzc}\}$. However, the disadvantage with this approach is that the new coordinate system will differ from the longitudinal, lateral and normal symmetry axes of the vehicle. Subsequently, in practice, the second method is preferred. Diagonalisation of the body-fixed inertial tensor is achieved by applying the Parallel Axes Theorem. Both the *Remus* and *AUTOSUB* AUV model, as will be mentioned later, adopt this approach.

It must be noted here that velocity vector ν in the body coordinate frame cannot be directly integrated to obtain the position coordinates in earth-fixed reference frame, rather, they are related by the transformation matrix $J(\eta)$ which is given by:

$$J(\eta) = \begin{bmatrix} J_1(\eta_2) & \mathbf{O}_{3\times3} \\ \mathbf{O}_{3\times3} & J_2(\eta_2) \end{bmatrix} \tag{3.7}$$

where

$$J_1(\eta_1) = \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{3.8}$$

$$J_2(\eta_2) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi - s\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix} \tag{3.9}$$

and

$$\eta_2 = [\phi\ \theta\ \psi]^T$$

In the above transformations, $s\cdot = \sin(\cdot)$, $c\cdot = \cos(\cdot)$, $t\cdot = \tan(\cdot)$ and $\mathbf{O}_{3\times3}$ is a null matrix. This gives the vector $\dot{\eta}$

$$\dot{\eta} = J(\eta)\nu \tag{3.10}$$

which can be integrated to get the position coordinates in earth-fixed frame of reference. Equation 3.10 is also known as the kinematic equation. In fact, the set of all displacements or the set of all such matrices in 3.10 with the composition rule of standard multiplication operation between matrices, is called $SE(3)$, the special

Euclidean group of rigid body displacements in three dimensions. Together with Equation 3.6, they can be used to describe compactly the behaviour of the six DOF AUV. Notice that Equation 3.8 is a matrix that details spherical displacements, and is a subgroup of $SE(3)$. It is also called the special orthogonal group in three dimensions, or simply $SO(3)$. It is clear that one can also derive other subgroups from $SE(3)$.

Before proceeding further, the rudder deflection convention should be clarified. The convention adopted here is in accordance to the one employed by the marine community. A positive rudder deflection, defined to have the same sense as the yaw angle, causes a negative yaw perturbation, and a very small positive sway perturbation as exemplified by Figure 3.3.

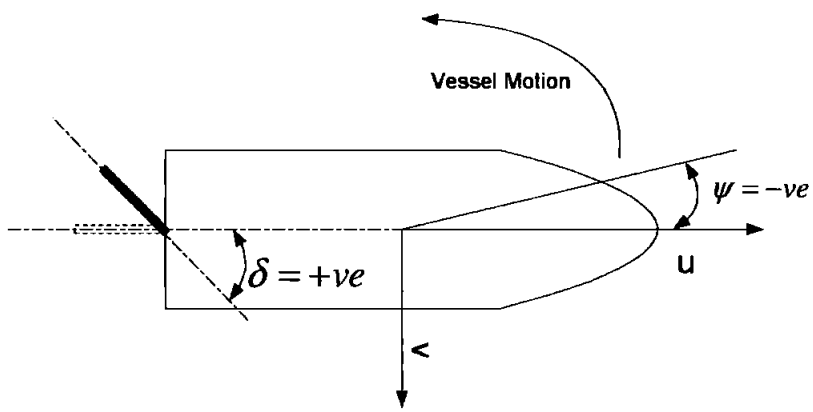


Figure 3.3: Rudder deflection convention

3.2 Three DOF AUV Model (Planar motion)

As aforementioned, a complete description of the dynamics behaviour of an AUV necessitates a set of six DOF kinematic and dynamic equations. Fortunately, due to certain practical constraints which will be detailed later, and the ease of implementation, the dynamics can be decoupled so that only a three DOF dynamic model is required.

The majority of the AUV missions such as mine hunting, seabed surveying, pipeline tracking, scientific data collecting entail diving to a certain depth and commencing to

meander within some predesignated area whilst maintaining a constant depth. Some specific missions do however require the AUV to hug the terrain whilst preserving the distance from the seabed. In the latter missions, a planar motion collision avoidance manoeuvre is executed when a preset terrain gradient or obstacle size threshold is triggered.

One crucial contributing factor in supporting the interest in the AUV planar motion is due to the hardware limitation. The 3-D forward looking sonar is still in its infancy. It is excessively expensive, bulky and has high power consumption, thus limiting wide adoption in AUVs. This explains the fact why there is such ubiquitous usage of 2-D forward looking sonars in AUVs. Typically, a forward looking sonar has the configuration as depicted in Fig 2.3(a) to increase the horizontal detection envelope. In this specific configuration the discrimination of object depth is very limited thus making 3-D manoeuvres unsafe to be performed.

Another substantial factor is related to the computational efficiency consideration. The search space in 2-D is clearly smaller than in 3-D case. The majority of motion planning algorithms time complexity are of $\mathcal{O}(a^d)$ type, where a is a real value constant and d is the number of states. Reducing the state dimension drastically lowers the computational requirement which propagates to smaller, cheaper hardware and lower energy consumption, a set of highly desirable features for AUVs.

Herein, the kinematic formulation is presented first since it is generic for all types of planar motion vehicles. From Equation 3.7 and taking $\phi = 0$, $\theta = 0$ while neglecting the depth, z dimension, the kinematic transformation matrix from body to Earth-fixed coordinate becomes

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ r \end{bmatrix} \tag{3.11}$$

There is no singularity problem for this trivial case but this effect will need to be considered if the Euler angle formulation is expressed and operates in $SE(3)$. By combining both Equation 3.11 and the reduced state Equation 3.6, one yields a longitude-latitude-yaw model where the state vector becomes $[u\ v\ r\ x\ y\ \psi]^T$.

Two types of AUV models are employed in this thesis.

- *Remus*, a small, agile AUV, latitude-yaw only model (no surge dynamics)
- *AUTOSUB*, a large AUV with nonlinear dynamics at different speeds

3.2.1 *Remus* AUV model

Remus AUV was developed by Alt and Grassle (1992) at the Oceanographic Systems Laboratory at the Woods Hole Oceanographic Institution. *Remus* is a low cost, modular vehicle. The nominal dimensions of the vehicle are 1.4 m in length and 0.3 m in diameter. Its missions range from autonomous docking, long-range oceanographic survey, and shallow-water mine reconnaissance (Alt *et al.* 1994). *Remus* currently uses the classical field-tuned PID controller. Although simple, nevertheless, it functions very well. Controller retuning is necessary when the payload or length are altered to cater for different missions. The versatility of *Remus* was proven in the 2003 Iraq conflict (Jordan 2003) where it was deployed primarily for mine-hunting missions.

A six DOF *Remus* model is provided by Prestero (2001). However, in conducting the experiment, the speed was kept constant at 1.54 m/s via the use of a speed controller. Consequently, this limits the model to be only true around this regime. The latitude-yaw only model, assuming the speed is fixed, is adopted for the Chapter 4 simulation study. This model was employed by Fodrea and Healey (2003). Note that the pitch and roll effects are neglected.

In matrix form, the linear three DOF dynamics equations of the vehicle can be defined as

$$\begin{bmatrix} m - Y_{\dot{v}} & 0 & 0 \\ 0 & I_{zz} - N_{\dot{r}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_v & Y_r - mU_o & 0 \\ N_v & N_r & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} Y_\delta \\ N_\delta \\ 0 \end{bmatrix} \delta_r(t) \tag{3.12}$$

Remus has a maximum rudder deflection of $\pm 13.6°$ and a rudder rate limit of $18°/s$. Embedding these two components into the model resulted in a nonlinear system. The related coefficients are provided by Prestero (2001).

3.2.2 *AUTOSUB* AUV model

The *AUTOSUB* AUV is the brainchild of the National Oceanography Centre. In contrast to the small size *Remus*, *AUTOSUB* is a very large torpedo shaped vehicle. Dimensionally, the vehicle is 7 m long, and approximately 1 m in diameter and has a nominal displacement of 3600 kgs. It is used for conducting under ice and deep sea surveying. Its normal cruising speeds are from 1 m/s to 2.2 m/s, with a top speed of 5 m/s. The detailed six DOF, nonlinear model of the AUV model was supplied by QinetiQ Ltd, formerly DERA (Marshfield 1992).

This model is more complex and realistic compared to the *Remus* model. Owing to its large size, the *AUTOSUB* is equipped with numerous actuators to increase the control authority. This indirectly accentuates the already severe cross-coupling effect inherent in the AUV. It is also equipped with two x-axial thrusters, which welcome the use of differential thrust control strategy at low speed.

In most underwater vehicles travelling at speeds in excess of approximately 0.5 m/s control surfaces are employed in preference to thrusters. The hydrodynamic forces acting upon the rudders and hydroplanes of the vehicle at such speeds provide much greater manoeuvring potential than the use of auxiliary thruster mechanisms, and are thus a more efficient means of controlling the vehicle motion (Cowling 1996). Additionally, it is a well known phenomenon that low speed control using rudders and hydroplanes is subject to reversal effects. Referring to the *AUTOSUB* AUV, the more appropriate means of achieving yaw control is via the use of locked upper and lower canard rudders. Employing these actuators in this manner leads to a cancellation of the rolling moment normally produced by the use of an individual rudder or differential main x-axial thruster strategy (Cowling and Corfield 1995). The stern upper and lower rudders are relegated to sway control, hence neglected in this thesis. The upper and lower canard rudders are used in locked formation throughout the remainder of the simulation studies. Furthermore, the cross-coupling effects in roll and pitch are neglected, assuming that they have been stabilised by some low-level controllers.

Unfortunately, the surge dynamic is not modelled in the six DOF model provided by QinetiQ Ltd. It was assumed that a constant speed can be maintained with the built in speed controller. To add realism to the simulation, the surge dynamics were

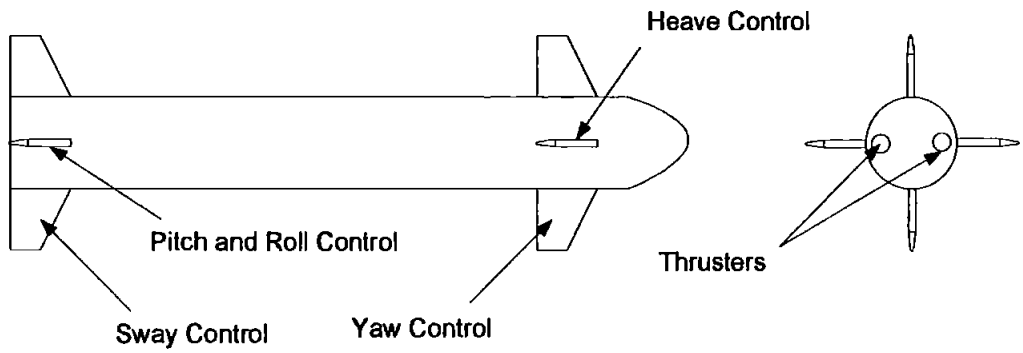


Figure 3.4: The complete control authority of *AUTOSUB* AUV

added to the *AUTOSUB* model. Therefore, an estimate of the value of the propulsion coefficient, X_{prop}, is required.

Neglecting the interactions from sway, heave, roll, pitch and yaw suggests that the speed equation is given by Fossen (1994) as

$$(m - X_{\dot{u}})\dot{u} = X_{|u|u}|u|u + (1-t)T + X_{ext} \tag{3.13}$$

Here it is assumed that quadratic damping is the dominating dissipative effect. Furthermore n represents the propeller revolution, u is the surge velocity, X_{ext} is external disturbances due to waves and current and t is the thrust deduction number t. Thrust equation can be expressed as

$$T = T_{|n|n} + T_{|n|V_a}|n|V_a \tag{3.14}$$

where T is the developed thrust and V_a is the advance speed at the propeller (speed of the water going into the propeller). A common method is to design an inner loop PI control to regulate the desired revolution, where revolution can be measured using a tachometer, or encoder. In the remaining study, the dynamics of the thruster are neglected assuming that it is much faster compared to the surge dynamics of the vehicle. This is true for a large AUV like *AUTOSUB*. Clearly, the forward and backward thrust will be non-symmetrical in practice but is again neglected in the simulation.

For simplicity, it is assumed that $T_{|n|va} = 0$ (affine system). Introducing ,the notation

$X_{|n|n} = (1-t)T_{|n|n}$, finally yields:

$$(m - X_{\dot{u}})\dot{u} = X_{|n|n}|u|u + X_{|n|n}|n|n + X_{ext} \tag{3.15}$$

Equation 3.15 is clearly nonlinear because of both the u and n quadratic term. A control allocation unit can be employed to 'remap' the inputs such that it is affine again with reference to the model.

Accordingly, the nonlinear three DOF *AUTOSUB* dynamic model can be expressed as

$$\begin{aligned}
&(m - X_{\dot{u}})\dot{u} = X_{u|u|}|u|u + (mv + X_{vr} + X_{r|r|} + mX_g)r + X_{prop}n^2 \\
&(m - Y_{\dot{v}})\dot{v} + (mX_g - Y_{\dot{r}})\dot{r} = (Y_{v|v|}|v| + Y_{uv}u)v + (Y_{r|r|} - mu + Y_{ur}u)r + u^2Y_{\delta r}\delta r \\
&(mX_g - N_{\dot{v}})\dot{v} + (I_{zz} - N_{\dot{r}})\dot{r} = N_{uv}v + (N_{ur}u + mX_gu)r + u^2N_{\delta r}\delta r
\end{aligned} \tag{3.16}$$

A SIMULINK model of Equation 3.16 is illustrated in Fig 3.5. The *AUTOSUB* has a rudder deflection limit of $\pm 25.2°$ and a slew rate of $9.9°/s$. The maximal thrust available for this set of actuators is ± 450 N. In the remaining study, the *AUTOSUB* cruising speed is taken to be 2.0 m/s when the motor is running at 140 rpm. X_{prop} is taken as 0.006810 after consulting the dynamics of the NDRE-AUV (Jalving and Størkersen 1994) and NPS AUV II (Healey and Lienard 1993).

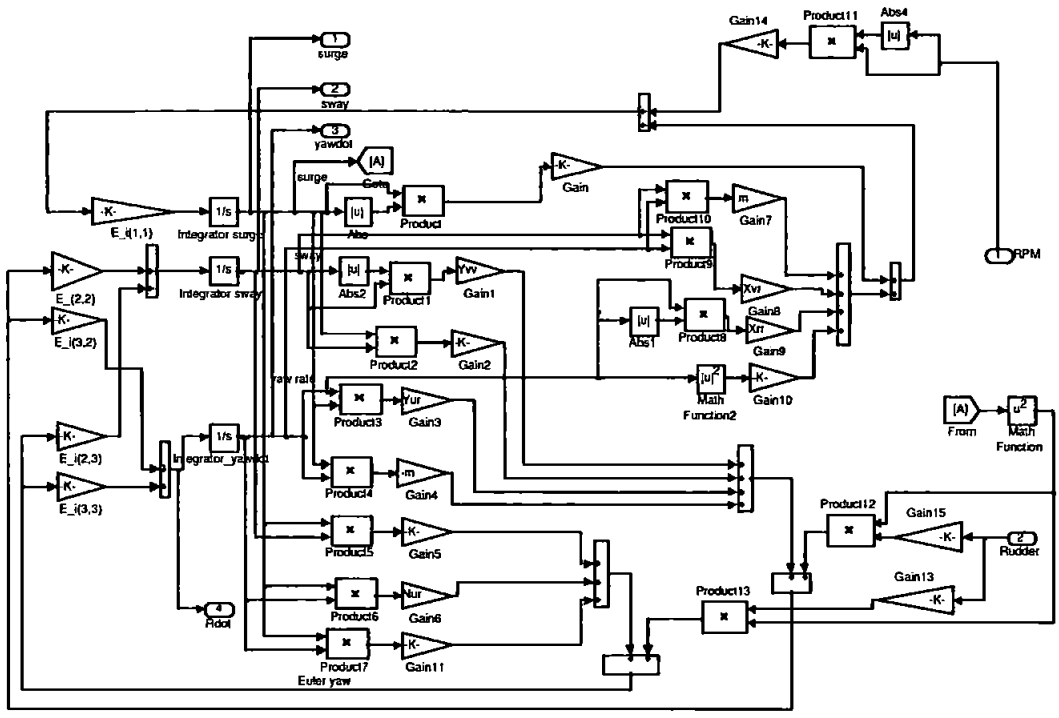


Figure 3.5: A nonlinear SIMULINK model of *AUTOSUB* AUV

The following parameters describe the AUV model used herein:

$W = 35316\ N$	$B = 35316\ N$	$L = 7.0\ m$	$m = 3600\ kg$
$\rho = 1025.2\ kg/m^3$	$I_{xx} = 0\ kgms^2$	$I_{xy} = 0\ kgms^2$	$I_{yy} = 0\ kgms^2$
$I_{zz} = 8304\ kgms^2$	$I_{xz} = 0\ kgms^2$	$I_{yz} = 0\ kgms^2$	$g = 9.81\ m/s^2$
$x_G = 0.34\ m$	$y_G = 0\ m$	$z_G = 0.02\ m$	$X_{prop} = 0.006810$

Table 3.1: Important parameters of *AUTOSUB* AUV. The rest of the hydrodynamic coefficients (not shown) are property of QinetiQ Ltd.

An open-loop *AUTOSUB* response of a saturated step input on the locked canard rudders is shown in Fig 3.6 and Fig 3.7. Fig 3.6 shows the path the AUV is initiating, which is not a perfect circle. True circle turning manoeuvre requires the AUV to hold the surge and sway velocity constant.

Fig 3.7 illustrates clearly the cross-coupling effect of the longitude-latitude-yaw *AUTOSUB* model. Notice that as the vessel commences the turning manoeuvre, the

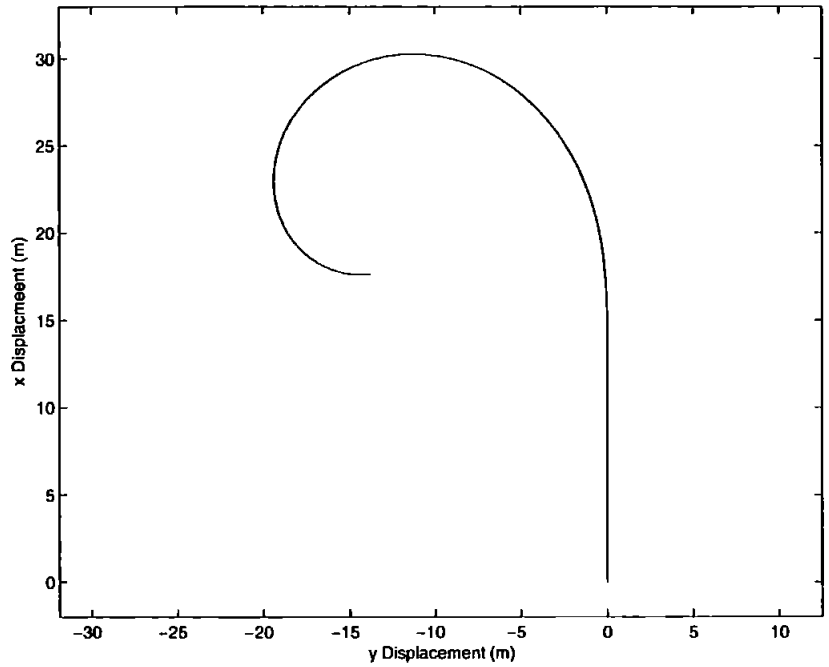


Figure 3.6: The x-y trajectory of the AUV when subjected to a saturated step input on the locked canard rudders in the open loop.

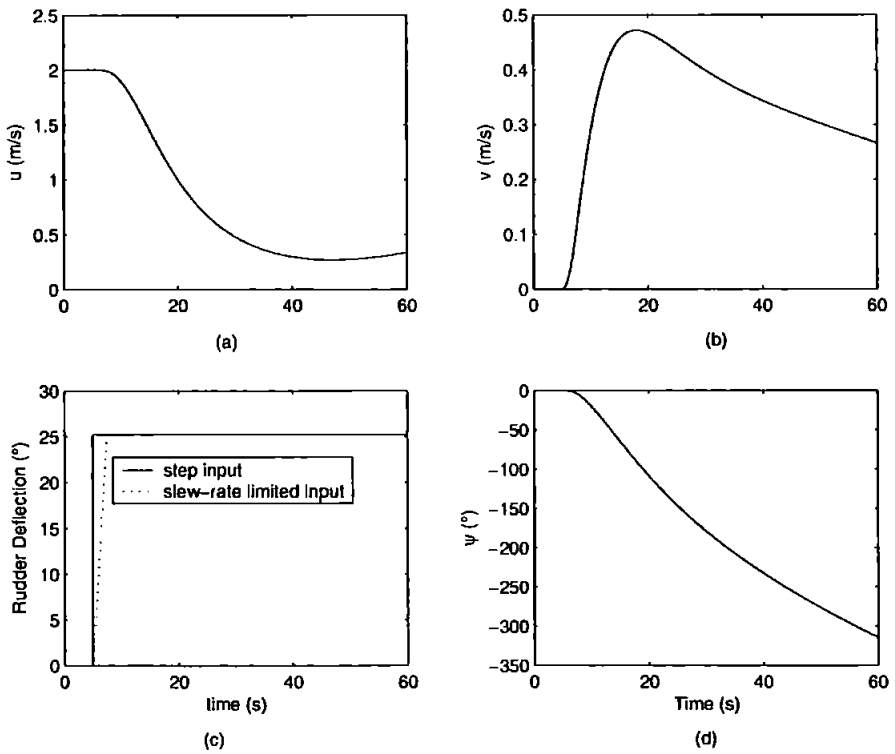


Figure 3.7: The cross-coupled motion of the AUV when subjected to a saturated step input on the locked canard rudders in the open loop.

surge velocity (Fig 3.7(a)) drops drastically due to the higher induced drag. The surge velocity converges gradually until the system thrust and drag are in a state of equilibrium. Fig 3.7(b) depicts that the sway velocity increases rapidly before it settles down. This behaviour is typical of non-holonomic vehicles which always exhibit a constant sway velocity when making a turning manoeuvres. The reason is the vehicle x-axis, body-fixed-frame of reference is not tangent to the arc the vehicle is executing. Note that the step input injected into the canard rudders as shown in Fig 3.7(c) will not directly effect the system because of the limitation imposed by the slew-rate.

3.3 Disturbances

A disturbance can be defined as a form of an undesirable effect injected into a plant. It has the propensity to perturb the normal behaviour of the plant, or to be precise, to deviate the plant from the expected response. It is an undeniable fact that disturbances are to be found in practice. One of the criteria in assessing a good a controller performance is with regards to its disturbances attenuation capability. This type of controller is termed as a regulator.

In the course of acquiring a better understanding of the subject, researchers have categorised disturbances into various types with regards to their frequency content such as high (HF) or low frequency (LF), the disturbance source either if it is endogenous or exogenous, and the effect to the plant model, additive or multiplicative, and lastly if the disturbances are zero-mean type. Two types of disturbances that are of interest here are the one induced by the environment and the one embedded in sensor measurements.

3.3.1 Environmental disturbances

For surface vessel, environmental disturbances include wind, waves and ocean currents. The ocean currents flow in complex patterns affected by wind, the water salinity and heat content, bottom topography, and the Earth's rotation. In the context of an AUV which is travelling below 30 m, one can safely ignore the sea surface effects

such as waves and winds. For most marine control applications, the ocean currents effect can be approximated as additive to the plant dynamics. The ocean currents are also considered to be of the low frequency type and are non zero-mean in nature. The AUV will suffer from drifting effect if the current effect is not taken into account. Elimination of this phenomenon can be achieved by introducing an integrating controller. Or better still, if an accurate mathematical model of the system exists, then currents and waves can be explicitly estimated using an observer (Torsetnes 2004).

The velocity magnitude of the ocean currents vary from 0 m/s to 2.5m/s depending on the depth and region considered. The fastest current is the Gulf Stream which tends to move at above 1.5 m/s with peak velocity reaching 2 m/s at the surface (Coble *et al.* 1996, Gross 1990). Gross (1982) reported that the speeds of deep currents vary from 0.02 - 0.25 m/s in deep water. Other currents such as the one in the narrowest point of the Florida Straits which has water masses in a cross section approximately 70 km wide and 200 m deep, move forward at a speed of more than 1 m/s (Gaskell 1973).

According to (Fossen 1994), ocean current velocity can be simulated by using a first order *Gauss-Markov Process*. For instance $V_c(t)$ can be described by the following differential equation:

$$\dot{V}_c(t) + \mu_o V_c(t) = w(t) \tag{3.17}$$

where $w(t)$ is a zero mean Gaussian white noise sequence and $\mu_0 \geq 0$ is a constant. Typically, it is possible to use $\mu_0 \geq 0$ which simply corresponds to a *random walk*. Clearly, the process must be bounded such that $V_{min} \leq V_c \leq V_{max}$ to preserve the fidelity of the ocean currents simulation. The associated pseudocode can be found in Algorithm 3.1.

Algorithm 3.1 (CURRENTS). The following algorithm simulates a current with bounded velocities.

Require: $V_{cmin}, V_{cmax}, T, \Delta T$
1: $V_c \longleftarrow \frac{(V_{cmin}+V_{cmax})}{2}$ {initialise the V_c}
2: **for** k from 0 to T **do**
3: $V_c(k+1) \longleftarrow V_c(k) + \Delta T \dot{V}_c(k)$ {Euler Integration}
4: **if** $V_c(k+1) > V_{cmax}$ or $V_c(k+1) < V_{cmin}$ **then**
5: $V_c(k+1) \longleftarrow V_c(k) - \Delta T \dot{V}_c(k)$
6: **end if**
7: **end for**
8: **return** V_c

Algorithm 3.1 has been translated into MATLAB code for simulation purpose. A sample run was executed to obtain the result as illustrated in Fig 3.8. The current velocity magnitude was bounded to be in the range of 1 m/s to 1.5 m/s. The Euler integration algorithm uses a 0.1 s time step with the variance of the Gaussian random generator function being set to 0.01 m/s. Pertaining to Fig 3.8 (a) one can observe the unfiltered current, the filtered current is shown in Fig 3.8 (b). The filtering process was achieved by employing a simple moving average filter with 10 coefficients where each one has a value of 0.1. The aim is to eliminate the HF noise hence making it much more realistic.

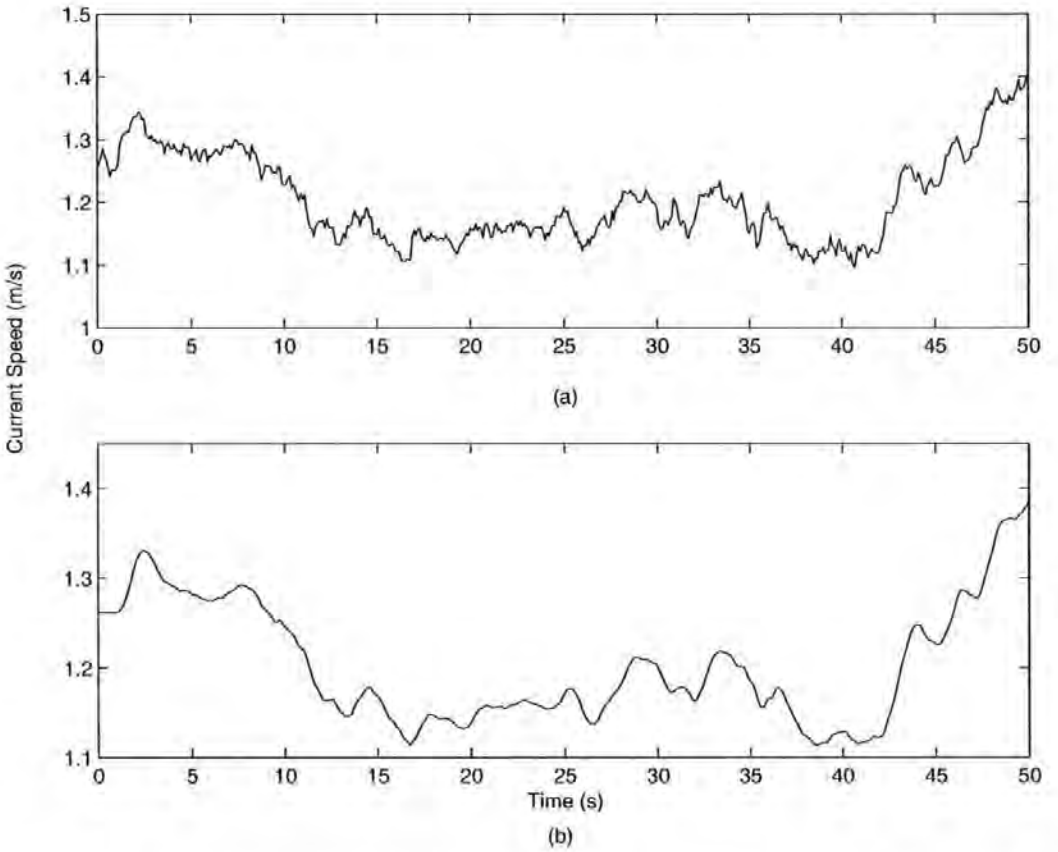


Figure 3.8: A simulated ocean current bounded between 1.0 m/s to 1.5 m/s for a time span of 50 s.

The 2-D simulated current model can be described in terms of the current velocity V_c and direction of the current β as

$$u_{c/O} = V_c \cos(\beta - \psi) \tag{3.18}$$

$$v_{c/O} = V_c sin(\beta - \psi) \tag{3.19}$$

where $u_{c/O}$ and $v_{c/O}$ are current velocity components with respect to the body fixed frame of the vessel. The pertinent variables are illustrated in Fig 3.9. On the other hand, $u_{c/E}$ and $v_{c/E}$ are the current velocity components with respect to the Earth fixed reference frame. The $u_{c/O}$ and $v_{c/O}$ can be directly added to the vessel velocity components for current effect simulation usage.

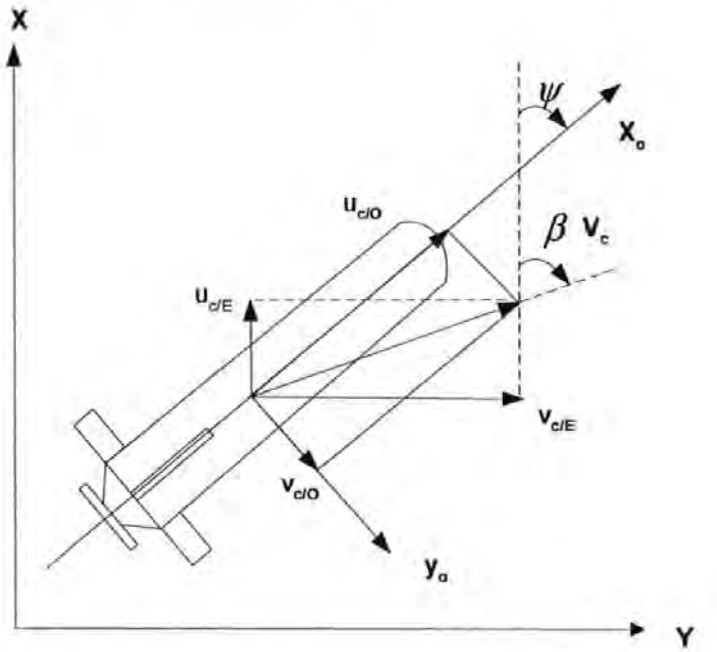


Figure 3.9: Variables for transforming the ocean current velocity in Earth- fixed frame such that it is coincided with the vehicle body-fixed frame of reference.

3.3.2 Sensor noise modelling

AUVs are equipped with an assortment of dedicated navigational sensors to accomplish missions. Sensors such as Doppler velocity sonar, inertia measurement unit (IMU), sonar, TCM and GPS have their own unique type of noise signature.

In principle, all sensors data are corrupted by HF zero-mean noise which is affecting almost every data sample. When this noisy data is utilised without proper filtering as a feedback to a controller, it can induce instability and chattering effect leading to actuator wear and tear. Consequently this not only can shorten the life span of the actuators but also renders the system not energy efficient. One common solution

is to treat the data by passing over a low pass filter before it is conveyed to the controller. Various filters can be adopted for this purpose, such as the Butterworth, Finite Impulse Response (FIR) to the more sophisticated Kalman filters. Care must be taken since the incorporation of the filters into the system also introduces lag which degrades the gain margin and phase margin of the system, thus rendering the closed-loop system more susceptible to instability. This thesis assumes that such filtering is taken care of, hence HF noise influence is neglected.

Certain sensors such as the Inertia Measurement Unit (IMU) tends to suffer from LF non-zero mean disturbances. To elaborate, IMU is critical in providing the important measurements to locate the vehicle in a 3-D space. This is attained through integration and transformation of the acquired linear acceleration and angular rate measurements. Unfortunately, the integration process accumulates the errors which become unbounded in a very short finite time interval.

A simple and yet insightful experiment was conducted to assess the drifting behaviour of a low cost IMU. The IMU employed is of a low cost miniature, strapped down, Altitude and Heading Reference System (AHRS) type, model MT9-B from the company Xsens. In the following experiment, the unit was left in a stationary position before the reference frame is reset. Local magnetic field distortion calibration was initiated through the software. The sampling rate was set at 40 Hz. Fig 3.10 illustrates the acceleration data of each axis acquired from the MT9-B for a time duration of 60 s. Notice that both x-axis and y-axis reveal zero-mean data but there is a mean of approximately 9.72 m/s^2 for the z-axis due to the gravity effect. Using the Euler integration technique, one may obtain the velocity and displacement quantities as depicted in Fig 3.11. It is apparent from the Fig 3.11(a) that the velocity data suffer from a drifting effect which is rather pronounced in the y-axis. One can also discern the sinusoidal distortion of the data which is suspected to be induced by the minute fluctuation of the angular rate measurements. Alternately, the Fig 3.11(b) shows the 3-D plot. The displacement data (Fig 3.11(c) and Fig 3.11(d)) do not exhibit the periodical behaviour but demonstrate severe drifting instead. A displacement drift of approximately 15 m in 60 s will render the data unusable for displacement estimation in practice. Furthermore, one can extrapolate from this experiment that the severity of drifting effects will accentuate when the vehicle is in motion.

The above experiment demonstrated that a periodic global reference frame reset is

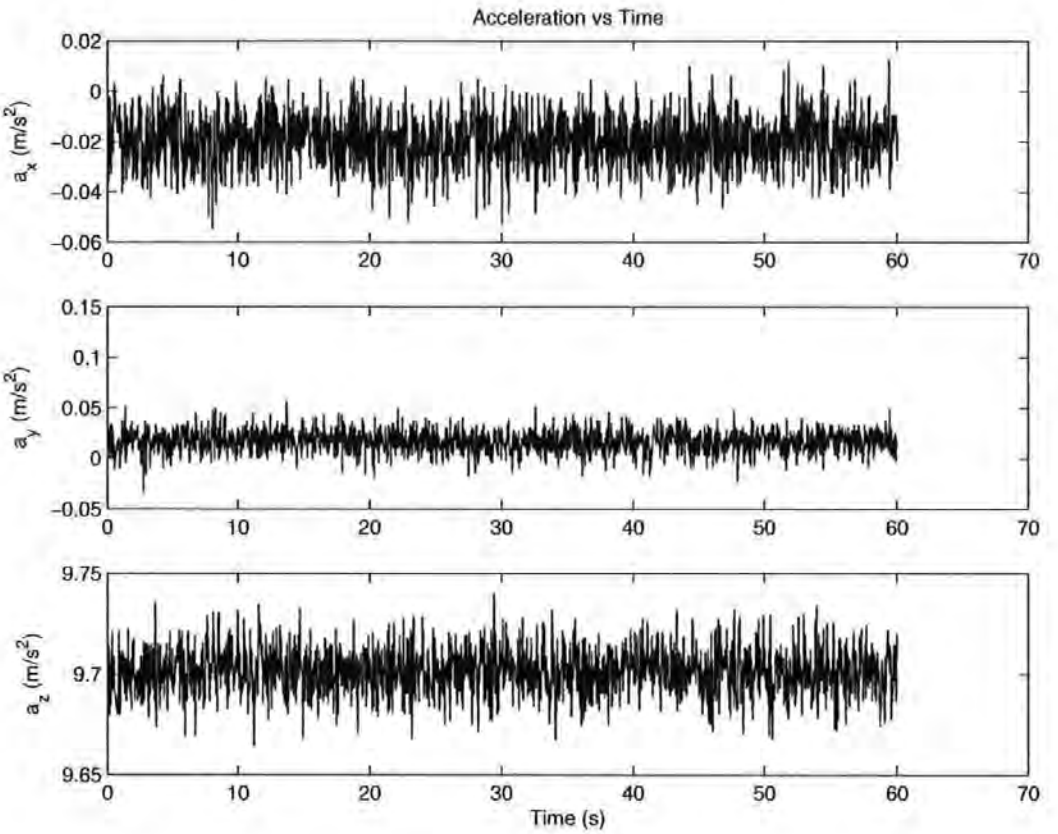


Figure 3.10: Acceleration measurements from MTB-9 IMU

essential if the positioning data is to be remain accurate. In practice, another sensor capable of providing position measurements with reference to a global frame must be applied to reset or to recalibrate the IMU measurements intermittently. The IMU and the positioning sensor functions are complementary, IMU provides accurate high frequency data crucial for feedback system, while the positioning sensor recalibrates, resets the IMU to mitigate the drifting effect.

A plethora of positioning sensors exist. These sensors can be a GPS, Long Baseline (LBL)/Short baseline (SBL) navigation systems and lastly a simultaneous localisation and mapping (SLAM) unit. The former is the cheapest option whist remaining very accurate especially if one is employing the latest Wide Area Augmentation System feature known as WAAS in the USA. Essentially, it is a system of satellites and ground stations that provide GPS signal corrections. An accuracy of better than three meters 95% of the time is attainable by this system. The European counterpart of this system is known as the European Geostationary Navigation Overlay Service (EGNOS). Even so, one apparent disadvantage is that the positioning data from the

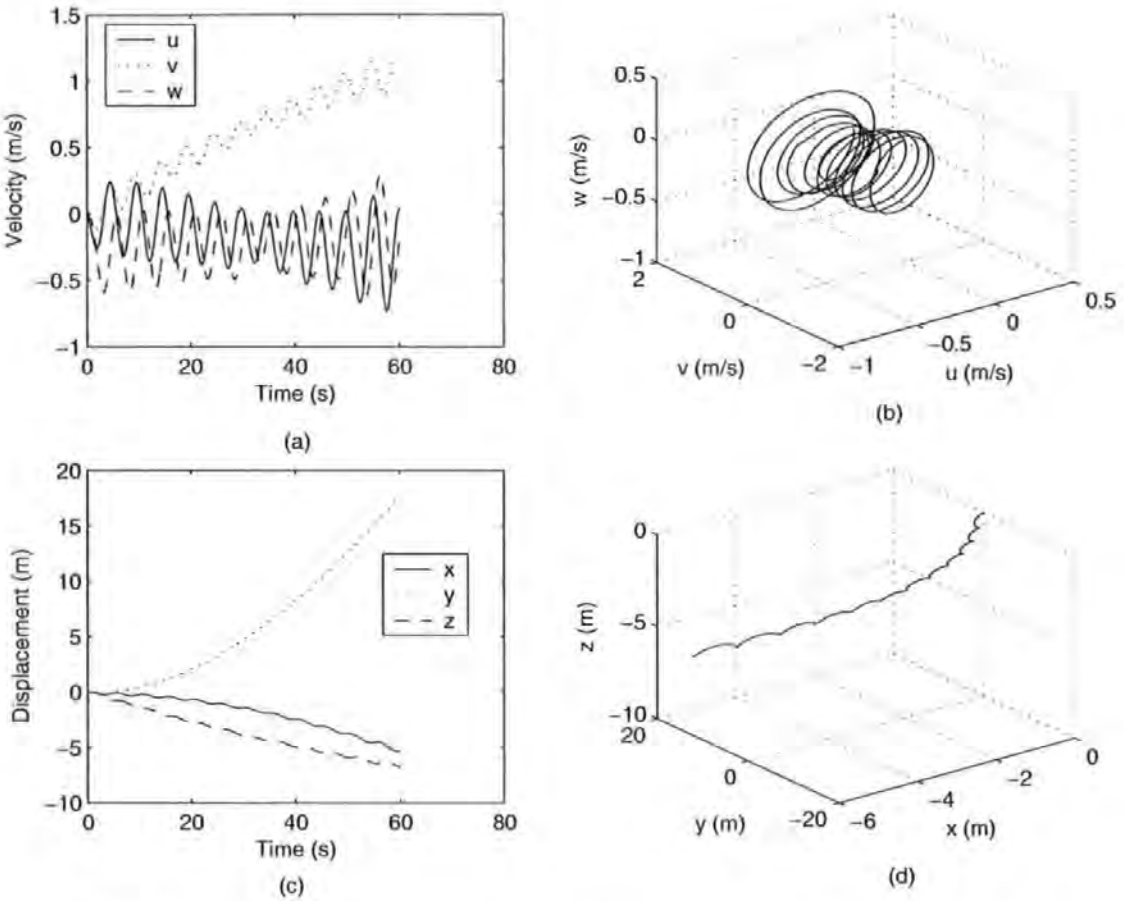


Figure 3.11: Velocity and displacement measurements from MTB-9 IMU

GPS is not accessible when submerged. The LBL and SBL are very accurate and yet precise, and have been frequently used for AUV's surveying task but the financial cost is prohibitive for the majority of missions. The final option is rather promising, the SLAM technology allows full AUV autonomy to be exercised but requires a high resolution forward looking sonar and a dedicated processing unit, which can be expensive (Carpenter 1998, Tomatis *et al.* 2001, Rikoski *et al.* 2002). Nonetheless, SLAM technology is still in a very premature stage in terms of commercial exploitation.

Assuming that the AUV is armed with a SLAM technology, the access of positional measurements may be inhibited for a period of time due to the failure to localise any reliable landmarks. Once the SLAM unit recalibrates itself, a large deviation, particularly with regards to position measurement, will be observed initially. Similar phenomenon can be observed from AUVs that calibrate the IMU using GPS on the surface. This effect will be simulated in this thesis to check the region of attraction

of the candidate controllers.

3.4 Conclusions

Details with regards to modelling of AUVs and disturbances to be used for the remaining of the thesis have been presented. The mathematical modelling of the AUVs were discussed based on a generalised model. Two specific AUVs mathematical models, the three DOF *Remus* and the *AUTOSUB* were outlined, followed by the appropriate assumptions.

It is critical to simulate the disturbances that will be endured by the AUV in order to assess its performance in real-life. Herein, two forms of disturbances, ocean currents and low frequency bias sensor noise were emphasised. Experiments were conducted to demonstrate the characteristics of the latter type of disturbance. With this understanding in mind, this completed the system modelling chapter and one shall commence with a novel motion planning technique in the next.

Chapter 4

The Rapid Exploring Random Tree

This chapter presents a novel motion planning technique based on the rapid-exploring random tree (RRT) algorithm (LaValle 1998), which is then applied to an AUV model in order to assess its viability. The enhanced version of the aforementioned algorithm is proposed to ameliorate the optimality of the returned solution. Furthermore, insights acquired from the study pertaining to the RRT behaviour, are thoroughly discussed. Some solutions to eliminate or failing that, extenuate the negative effects are proposed and addressed. As a consequence of this in depth analysis, a paper Tan *et al.* (2004*a*) was presented in an international conference.

4.1 Background

This chapter shall commence by investigating the prevailing issues suffered by most classical motion planning and path planning algorithms. It has been shown in Section 2.5 that the majority of the proposed techniques (Hyland 1990, Fogel and Fogel 1990, Arinaga *et al.* 1996, Arai *et al.* 1998, Sugihara 1998, Fox *et al.* 2000) do not explicitly take into account the dynamics of the vehicle. Secondary smoothing methods such as spline interpolation, are often employed to manipulate the path into conforming the vehicular dynamics. Without this interpolation process, one cannot ascertain that the paths are executable in practice. Frequently, rather conservative constraints are imposed on the derivatives of the flight path in order to avoid violating the low-

level feedback controller operating regime. To simplify further the process, the AUV body geometry is neglected by shrinking it into a point via the application of the configuration space concept. This assumption is valid if the vehicle considered is operating in a sparse environment. This trend is fast changing as AUVs are now being deployed in littoral waters, an environment densely populated by obstacles. The dynamic and unpredictable elements of littoral waters render it crucial for an AUV to exploit its dynamics to navigate.

Section 2.5 articulates that the approximate cell-decomposition methods such as A^*, dynamic programming and breadth-first search are highly susceptible to the curse of dimensionality. Therefore, it is reasonable for one to concentrate on randomised algorithms (Branicky *et al.* 2002). These algorithms do not have the completeness and optimality properties of the previous algorithms. However, their robustness to state explosion effects tends to make them preferable in practical and real-time applications. One interesting randomised algorithm is the RRT (LaValle 1998, LaValle and Kuffner 2000), which can be considered as an incremental form of Probabilistic Road Map (PRM) method and is designed to search efficiently nonconvex high-dimensional spaces. It possesses a few fascinating properties as outlined below:

1. It is biased to the unexplored space via a probabilistic search method. The free space bias property is vividly depicted in Fig 4.1. Both Fig 4.1(a) and Fig 4.1(b) indicate an identical tree but in different representations to assist exposition. From left to right, Fig 4.1(a) shows the gradual growth of the tree starting from q_{init} extending to the free space (largest Voronoi regions). Alternatively, Fig 4.1(b) highlights the nodes of the tree and the relevant space partitions correspondingly. This form of representation is termed as a Voronoi diagram. Fundamentally, the Voronoi diagram displays the locus of all points that are no nearer to one point than another.

2. Although the RRT algorithm is nondeterministic, it has been mathematically proven to be probabilistically complete (LaValle and Kuffner 2000). This implies that given a sufficient amount of time, the algorithm will find a solution to any configuration problem if such a solution exists. This property is highly appealing in an algorithmic perspective, however, its usefulness in real-time implementation is questionable, as the run-time required to discover a solution can be intolerable. Nonetheless, The RRT algorithm run time is usually much

faster than the dynamics of AUVs.

3. The simplistic nature of the algorithm also facilitates performance tuning, since only the metric and bias parameters need to be tuned. The RRT performance is actually rather sensitive to a prescribed metric as elaborated in Subsection 4.4.4.

4. It is capable of accommodating both algebraic (global) and differential constraints simultaneously, a vital feature for solving motion planning problems. In addition, algebraic constraints are induced by the static and dynamic obstacles in the environment whereas differential constraints are inherent in any dynamical systems. It must be noted that the terms algebraic and global constraints will be used interchangeably in this thesis. Strictly speaking, concurrent solving of both constraints is also realisable via classical cell decomposition algorithms. Even so, their susceptance to state explosion effects precludes their applications in time critical and high-dimensional problems.

4.2 Problem Description

Before looking into the details of the RRT algorithm, perhaps it is helpful to elaborate upon a general motion planning problem. Herein, the class of problems considered in this study can be formulated in terms of 9 components:

1. X, **state Space:** An n-dimensional closed and bounded manifold, $X \subset \mathbb{R}^n$. A set of continuous variables of $\mathbb{R}^n$, on which the dynamics of the system occur.

2. $\mathbf{x}_{init}, X_{goal}$, **boundary conditions:** $\mathbf{x}_{init} \in X_{free}$ is the initial state vector and $X_{goal} \subset X_{free}$ is a set of goal state vectors.

3. D , **collision detector:** A function, $D : X \rightarrow true, false$, that determines if the global constraints are satisfied for a given state. In an algorithmic perspective; If $D(x) = true$, hence it denotes that the state X satisfies the global constraints.

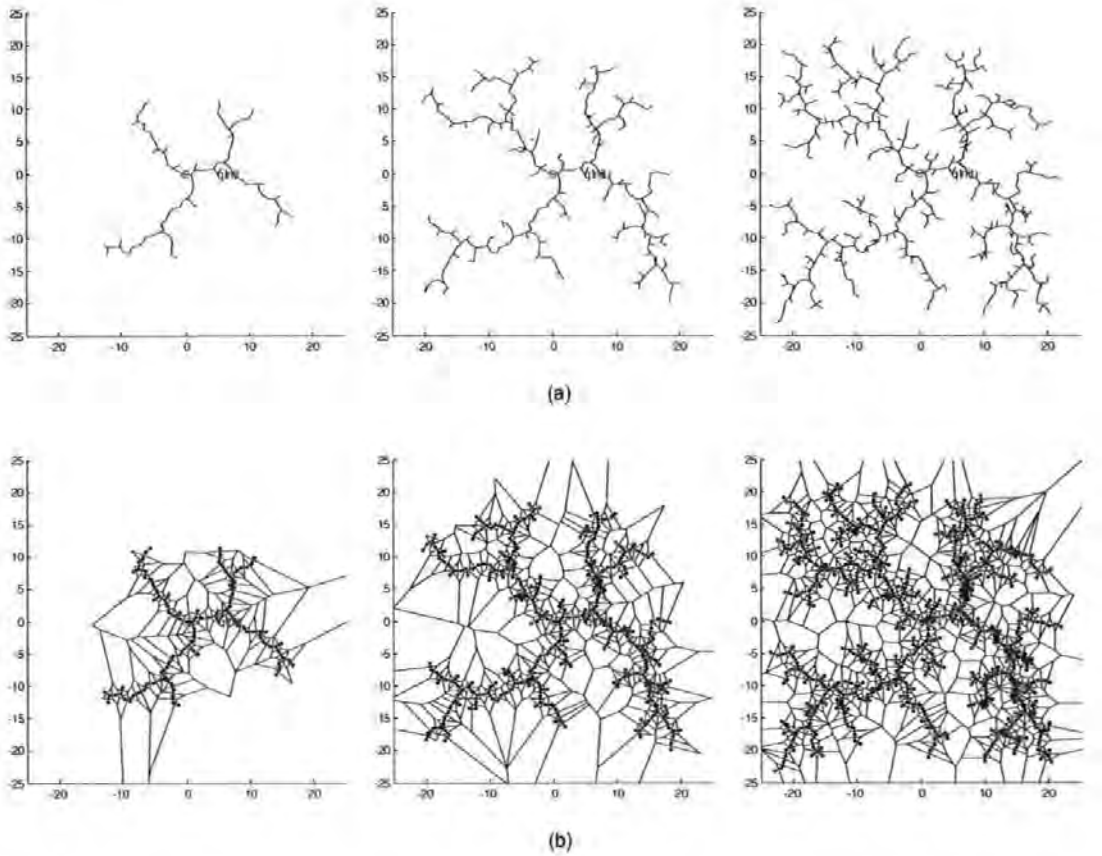


Figure 4.1: RRT propagation (a)(from left to right)An RRT growing from the initial state, q_{init}, symbolised by a circle, to the unexplored regions. (b)The nodes and Voronoi representation of a gradually expanding tree.

4. U , **finite input set:** A set, $U \subset \mathbb{R}^m$, which specifies the complete set of controls; U is independent of the state. Discretisation process is needed for continuous input. The $u_i \in [u_{imin}, u_{imax}], i = 1, ..., m$ denotes an input in U that has m inputs.

5. ρ , **metric function:** A function $\rho : X \times X \rightarrow [0, \infty)$. A weighted Euclidean metric is commonly employed.

6. Δt , **RRT time increment:** Time increment used by the algorithm to extend the RRT. Do not confuse this with δt.

7. λ , **goal tolerance:** $\lambda \in \mathbb{R}^+$, which denotes a threshold or tolerance such as $|\mathbf{x}_{goal} - \mathbf{x}| < \lambda$ so that a solution is reached.

8. f, **dynamic equation:** An equation expressed as a set of first order differential equations, $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$. It characterises the evolution of the state

and represents the differential constraints of a system. The equation can be nonlinear and time varying.

9. δt , **dynamic equation time increment:** Time increment used by the numerical integration routines (Euler, Runge-Kutta, Predictor etc) to solve the dynamic equation. The Euler integration formula is given by $\mathbf{x}((t) + \triangle t) = \mathbf{x(t)} + \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))\triangle t$.

In essence, the kinodynamic planning problem is to find a trajectory $\pi : [t_0, t_f] \rightarrow X_{free}$ from an initial state $\mathbf{x}_{init}$ to a goal state $\mathbf{x}_{goal} \in X_{goal}$ or goal region within the tolerance λ. A trajectory is defined as a time parameterised continuous path that satisfies both the algebraic and differential constraints. However, it can also be formulated in a way as a problem to find an input function $u : [t_0, t_f] \rightarrow U$ that results in a collision free trajectory connecting both $\mathbf{x}_{init}$ and $\mathbf{x}_{goal}$. As with most physical systems, input saturation and rate limit will also need to be taken into account. In some cases, it is also appropriate to select a path that optimises certain cost functions, such as the time to reach $\mathbf{x}_{goal}$ or the control effort which corresponds to the energy consumption of the system. Due to its randomised nature, the generated path will be suboptimal.

4.3 RRT Operation

To better appreciate the RRT, one needs to understand the basic operation of the algorithm: Starting from an initial state, $\mathbf{x}_{init}$, a tree is grown through a process of adding edge and vertex in each time step (iterations) (Algorithm 4.1). The following step is to call a function to extend the tree edge (Algorithm 4.2). This function constitutes several important subroutines.

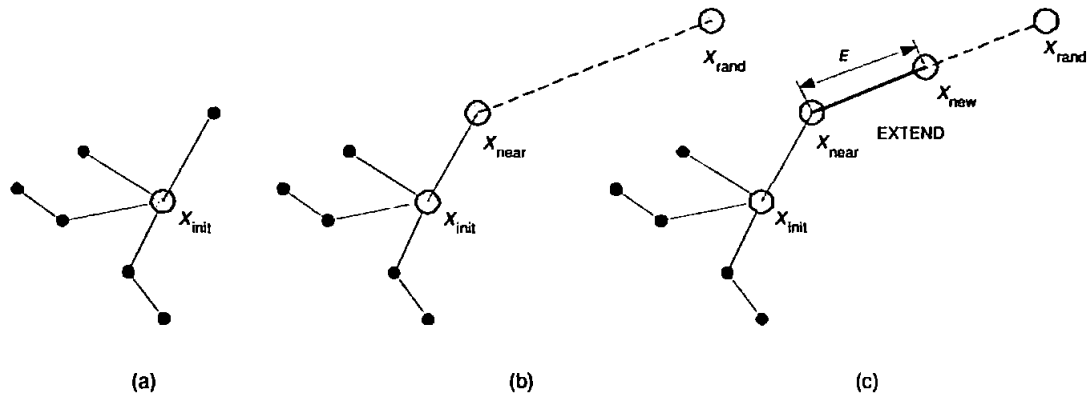


Figure 4.2: RRT Operations (a) Shows a simplified representation of a partial RRT with its initial state, $\mathbf{x}_{init}$ (b) Shows the inclusion of random state, $\mathbf{x}_{rand}$ and selection of the nearest state, $\mathbf{x}_{near}$ (c) Shows the addition of new state, $\mathbf{x}_{new}$ and its connection to near state, $\mathbf{x}_{near}$.

Algorithm 4.1 (BUILD_RRT). The following algorithm constructs an RRT, T, with K nodes

Require: $\mathbf{x}_{init}$

1: **call** T.init($\mathbf{x}_{init}$) {initialise tree, T}
2: **for** $k = 1$ to K **do**
3: $\mathbf{x}_{rand} \longleftarrow$ RANDOM_STATE(β) {extend T, see Algorithm 4.3.}
4: **call** EXTEND_RRT($T, \mathbf{x}_{rand}$) {extend T, see Algorithm 4.2.}
{If $\mathbf{x}$ is within the goal tolerance.}
5: **if** $\mathbf{x}_{new} \in \lambda$ **then**
6: **break**
7: **end if**
8: **end for**
9: **return** T

Algorithm 4.2 (EXTEND_RRT). The following algorithm extends a tree, T, towards $\mathbf{x}$ by taking a fixed step from the closest node in T towards $\mathbf{x}$.

Require: $T, \mathbf{x}_{rand}$

1: $\mathbf{x}_{near} \longleftarrow$ NEAREST_NEIGHBOUR($\mathbf{x}, T$) {Find the nearest node in T to $\mathbf{x}$, see Algorithm 4.4.}
2: $\mathbf{u}_{best}, success, \mathbf{x}_{new} \longleftarrow$ SELECT_INPUT($T, \mathbf{x}_{near}, \mathbf{x}_{rand}$) {Finding the 'best' control input,see Algorithm 4.5.}
{Checking for global constraints (collisions).}
3: **if** $D(\mathbf{x}_{new})$= TRUE **then**
4: **call** ADD_VERTEX($T, \mathbf{x}_{new}$)
5: **call** ADD_EDGE($T, \mathbf{x}_{new}$)
6: **end if**
7: **return** T

Assuming that one has a partial RRT as shown in Fig 4.2(a), the subsequent step is to introduce a random state into the configuration space, $\mathbf{x}_{rand}$ (Fig 4.2(b)) (Algorithm 4.3). The nearest-neighbour function (Algorithm 4.4) determines the 'nearest' state to the random state, $\mathbf{x}_{rand}$ to be extended. Here, the term 'nearest' is typically defined by a metric.

Algorithm 4.3 (RANDOM_STATE). This algorithm adopts an approach where the probability is slightly bias to returning the goal state.

Require: β

1: $\gamma \longleftarrow$ RANDOM[0, 1] {RANDOM is a standard uniform random number generator function.}
2: **if** $\gamma \geq \beta$ **then**
3: **return** RANDOM_STATE()
4: **else**
5: **return** $\mathbf{x}_{goal}$
6: **end if**

Algorithm 4.4 (NEAREST_NEIGHBOUR). This algorithm uses a naive method to check all x in T in order to find $\mathbf{x}_{near}$ which is nearest to $\mathbf{x}_{rand}$.

Require: $T, \mathbf{x}_{rand}$

1: $d_{min} \longleftarrow \infty$
2: **for all** x in T **do**
3: $d \longleftarrow \rho(\mathbf{x}, \mathbf{x}_{rand})$ {Find the nearest node in T to x.}
4: **if** $d < d_{min}$ **then**
5: $d_{min} \longleftarrow d$
6: $\mathbf{x}_{near} \longleftarrow x$
7: **end if**
8: **end for**
9: **return** $\mathbf{x}_{near}$

As shown in Fig 4.2(c), a new state, $\mathbf{x}_{new}$, that is E-distance away from $\mathbf{x}_{near}$ is then computed. E is a Minkowsky distance metric. The computation is required to find a suitable input $\mathbf{u}_{new}$, that is applied for a time increment, $\triangle t$ so that it can bring $\mathbf{x}_{near}$ to $\mathbf{x}_{new}$ (Algorithm 4.5). In essence, the computation is achieved by using a suitable numerical integration function. Before one proceeds to adding $\mathbf{u}_{new}$ to the tree, it is necessary to check if $\mathbf{x}_{new}$ satisfies the global constraints or reaches the goal state (Algorithm 4.2). The process is then repeated until either the maximum number of iterations or the goal is reached.

Algorithm 4.5 (SELECT_INPUT). The following algorithm evaluate, U of $\mathbf{x}_{near}$ and select the optimal control, $\mathbf{u}_{best}$ with reference to the metric ρ, that can grow $\mathbf{x}_{near}$ 'closest' to $\mathbf{x}_{rand}$.

Require: $T, \mathbf{x}_{near}, \mathbf{x}_{rand}$
1: $d_{min} \longleftarrow \infty$)
2: $success \longleftarrow false$
{Test all inputs of $\mathbf{x}_{near}$.}
3: **for all** u in U **do**
4: $\mathbf{x}_{new} \longleftarrow$ NEW_CONFIGURATION($T, \mathbf{x}_{near}, \mathbf{u}, \Delta t$) {Integrate the equation using Euler or Runge-Kutta method.}
5: **if** D($\mathbf{x}_{new}$)= TRUE **then**
6: $d \longleftarrow \rho(\mathbf{x}_{new}, \mathbf{x}_{rand})$
7: **if** $d < d_{min}$ **then**
8: $d_{min} \longleftarrow d$
9: $success \longleftarrow true$
10: $\mathbf{u}_{best} \longleftarrow \mathbf{u}$
11: **end if**
12: **end if**
13: **end for**
14: **return** $\mathbf{u}_{best}, success, \mathbf{x}_{new}$

4.4 RRT Performance Enhancement

4.4.1 Bias

RRT performance can be significantly improved by the introduction of certain biasing techniques. One such technique is to employ a Gaussian distribution function such that the expected value is located at the goal state as approached by Kim and Ostrowski (2003). Likewise, one can use a function to return either the goal state or a random state depending on a preset bias parameter as implemented in this study. The accompanied pseudocode is given in Algorithm 4.3. Basically, the bias parameter, β, with value ranges from zero to one is tuned to improve its searching performance. A low value near to zero will emphasise the free-space exploration, random search characteristic. Conversely, a high value near to one, transforms the RRT into a form of 'greedy' algorithm, a search directed to the goal direction. Understandably, a compromise between these two characteristics must be struck, and deducing from numerous experiments, a value from 0.05 to 0.2 has shown to provide adequate performance for most problems.

4.4.2 Quasi-random

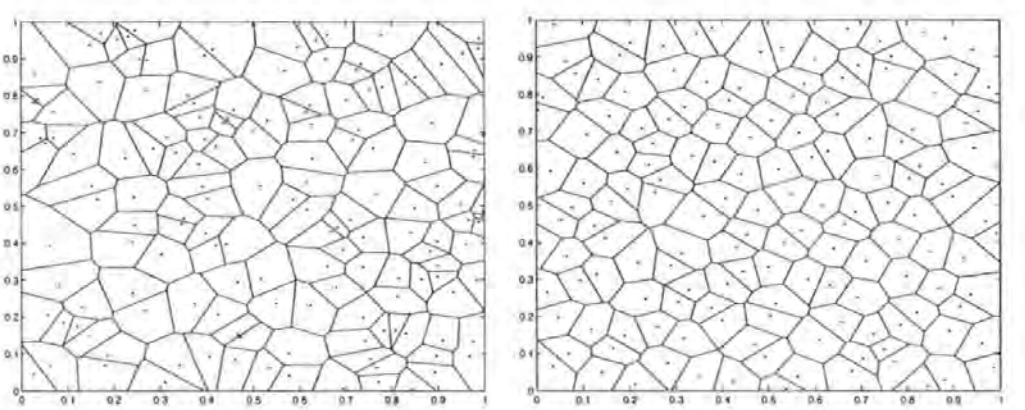
Low discrepancy sequences (quasi-random) such as the Hammersley (Hammersley 1960) and the Halton sequences (Halton 1960) have been argued to be more efficient than the pseudo random sequence (Branicky *et al.* 2002). Theoretically, the former sequence possesses certain desirable properties such as low discrepancy or improved uniformity over the sampling space. The generation of an element of a one-dimensional Halton sequence within the interval $[0, 1]$ is calculated using Equation 4.1 and Equation 4.2. Different prime numbers starting from the smallest, are used for the multi-dimensional sampling case.

$$x_i = \sum_{k=0}^{\infty} n_{k,i} p^{-k-1} \tag{4.1}$$

with $i > 0$,$p = 2$ and $n_{k,i}$ determined by the following equation:

$$i = \sum_{k=0}^{\infty} n_{k,i} p^k; \quad 0 \leq n_{k,i} \leq p; \; n_{k,i} \in \mathbb{N} \tag{4.2}$$

Fig 4.3(a) and Fig 4.3(b) show the Voronoi representation of 100 points generated by the pseudo-random and quasi-random generators, respectively. Notice that the pseudo-random points exhibit pronounced clustering behaviour, whereas the quasi-random points are more uniformly distributed.



(a) (b)

Figure 4.3: Voronoi diagram interpretation of 150 samples of (a)Pseudo-random and (b)Quasi-random point

Interestingly, the Hammersley sequence has been proven to possess better dispersion properties in a technical measure such as the discrepancy, but suffers from one glaring issue which precludes its wide spread adoption. Unlike the Halton sequence, the quantity of generated points must be known *a priori* before commencing the computation. For instance, if one has computed a Hammersley sequence of length 100, and would like to compute a Hammersley sequence of length 200, one must discard the current values and restart the whole process again. By contrast, if one computes 100 points of a Halton sequence, and then 100 more, and the result will be identical as computing the first 200 points of the Halton sequence in a single calculation.

Experiments were conducted to inspect the performance difference between the quasi-random generator based on the Halton sequence and the pseudo-random sequence when applied in RRT. The pseudo-random sequence is provided by a built-in C function, purported to be a linear congruential generator (LCG) (Knuth 1997). Despite the fact that the quasi-random sequence merit has been proven for the PRM method (Branicky *et al.* 2002), the results obtained for the RRT case are nonconclusive. In fact, the results are in concordance to that reported by Levine (2004). Therefore, the remaining simulations shall adopt the pseudo-random number generator.

4.4.3 Computational bottlenecks

Perusing through the algorithm sequence, one will notice that the two significant bottlenecks of the RRT algorithm are the nearest-neighbour (NN) subroutine and the collision detection subroutines.

Nearest-neighbour

The naive nearest-neighbour (NN) version which requires all of the nodes to be scanned, is the most computationally intensive. In this basic NN implementation, a search takes a time complexity of $\mathcal{O}(dn)$, where d is the configuration space dimension and n is the number of nodes in the tree. Clearly, the performance of this linear version algorithm degrades substantially as the tree expands. A more efficient data structure, implemented in a form of a template by Andrews (2001) is recommended

for faster query. It is claimed that the algorithm can query the nearest neighbour in $\mathcal{O}(\lg n \lg d))$ time, where lg denotes $\log_2$. To cater for real-time motion planning in fast mobile robots, Bruce and Veloso (2002) adapted the kD-tree, a form of data structure for fast searching, into the RRT algorithm. By embedding some additional novel features, they then christened the new algorithm as the execution extended RRT (ERRT).

Since exact query result from the NN search is not compulsory for the proper functioning of the RRT, this robust characteristic can be exploited by substituting the corresponding search with the approximate nearest neighbour (ANN) algorithm (Arya *et al.* 1998) or the approximate kD-Tree (Greenspan and Yurick 2003) that yields better computational efficiency. The ANN algorithm is expected to run in $\mathcal{O}(a(d, e) \log\ n)$ time, where $a(d, \epsilon) \leq d[1 + 6d/\epsilon]^d$, and $\epsilon \in \mathbb{R}$. The queried node will be bounded within a distance of $(1 + \epsilon)$ from the actual nearest neighbour. The following simulations, however, only employ the naive version of the NN search, since the primary objective herein is to examine the viability of RRT to solve the AUV motion planning problem. Algorithm efficiency issues shall be relegated to future research thereof.

Collision detection

Collision detection in this context pertains to a subroutine for detecting if two or more models are intersecting. The collection of these subroutines also known as an engine is predominantly used in games and CAD softwares. Regretfully, the computational demand frequently escalates as the dimension and complexity of the obstacle model are increased. Additionally, nonconvex obstacle models also compound the complexity of the calculations. For cases like this, high quality collision detection libraries are necessary to minimise the computational demands (Lin 1999). The following simulations avert the above mentioned problems by replicating the obstacles as rectangles and circles only, both are nonconvex polygons.

4.4.4 Metric sensitivity

It has been mentioned before that a suitable metric plays an important function in the operation of RRT. To add, in fact the intrinsic operation of RRT is fundamentally based upon Voronoi region bias (free space), which in effect depends predominantly on the embedded metric used. A metric, sometimes also known as a norm, is a simple and yet elegant mathematical method for assessing the 'closeness' between two elements. It is analogous to the concept of distance.

An appropriate metric, nevertheless, is indeed problem specific. The ideal metric is the optimal cost-to-go, which is the cost to move from one state to another state under the optimal trajectory. Interestingly, having knowledge of a perfect metric implies having knowledge of an optimal solution, which is the solution to the motion planning problem itself. From the discussion above, it can be inferred that a useful metric will be the one that can closely approximates the perfect metric. One near ideal metric is the quadratic performance index. To be precise, it is the ideal metric if the system dynamics are linear and there exists no global constraints. The metric ρ in the form of cost function or performance index is expressed with respect to $\mathbf{u}$ as

$$\rho^* = min\left(\phi[\mathbf{x}(t_f), t_f] + \int_{t_0}^{t_f} \xi[\mathbf{x}(t), \mathbf{u}(t), t]dt\right) \tag{4.3}$$

while satisfying the differential constraint,

$$\mathbf{f}[[\mathbf{x}(t), \mathbf{u}(t), t] - \dot{\mathbf{x}}(t)] = 0$$

where ρ is a function of $(\mathbf{x}_{near}, \mathbf{x}_{rand}, \mathbf{u})$. Note that Equation 4.3 assumes an obstacle-free environment, hence no global constraints are being imposed. It has been discovered that RRT performance tends to degrade as ρ and ρ^* diverge (LaValle and Kuffner 1999). In essence, the problem outlined above is an optimal control problem. The solution, frequently solved using numerical methods, can be time consuming since it entails solving a two-point boundary-value problem. The cost might be the distance travelled, system response performance, energy consumed, time elapsed during the execution of the trajectory, or any combination.

Therefore, the most popular metric utilised for most applications is the weighted

Euclidean distance. It is a compromised solution between solution optimality and computation issues. The metric in a two dimensional case can be expressed as $\sqrt{\alpha_1 x_1^2 + \alpha_2 x_2^2}$. The two constant coefficients α_1 and α_2 are weighting coefficients. This metric functions adequately in most problems and can be easily generalised to any number of dimensions. In fact, the Euclidean distance is a special case of the Minkowski L_m distance metric. For any integer $m \geq 1$, the L_m distance between points $p = (p_1, p_2, \ldots, p_d))$ and $q = (q_1, q_2, \ldots, q_3)$ in $\mathbb{R}^d$ is defined as $\sqrt[m]{\sum_{i=1}^{d} |p_i - q_i|^m}$. In the limiting case where $m = \infty$, this is equivalent to $\max_{1 \leq i \leq d} |p_i - q_i|$. The L_1, L_2 and L_∞ metrics are known as the Manhattan, Euclidean, and max metrics, respectively. Levine (2004) reported that a combination of Euclidean and Manhattan metrics function very well for a hybrid system. Then again, understanding each of these metrics potential is still a much researched subject.

For holonomic path planning problems in RRT based planners, the Euclidean metric usually produces excellent results. However, in an environment with global constraints, Euclidean metric can yield incorrect information. With reference to Fig 4.4, one can observe how incorrect the Euclidean metric is in accessing the true cost-to-go in an obstacles filled environment. Node $\mathbf{x}_3$ is considered closer to $\mathbf{x}_1$ than $\mathbf{x}_2$ to $\mathbf{x}_1$. Understandably, this is an error as if one refers to the correct cost-to-go, $\mathbf{x}_2$ is in reality nearer to $\mathbf{x}_1$.

Further aggravating the situation is the problem involving differential constraints and control saturations. Here again, the Euclidean metric has the propensity to provide misleading information, which consequently degrades the RRT performance. Fig 4.5 illustrates how node $\mathbf{x}_2$ is considered to be nearer to the $\mathbf{x}_{rand}$ (distance B), hence being selected for expansion in the direction of $\mathbf{x}_{rand}$. However, the differential constraints inherent in the AUV's dynamics impose limitation to the AUV's turning radius. The AUV is then coerced into executing a turning manoeuvres to arrive at $\mathbf{x}_{rand}$, but unable to do so completely. If node $\mathbf{x}_1$ is selected instead, in spite of having a longer distance to $\mathbf{x}_{rand}$, the AUV will arrive at node $\mathbf{x}_{rand}$ not only in a shorter time duration but via a less complex path. Also, depending on the structure of the AUV's dynamics, incorrect tuning of the weighting coefficients of the Euclidean metric can introduce unwanted bias into the RRT, diverging the search from the terminal state.

Figure 4.6 is used to better elucidate the above issue. It portrays an RRT simulation run using a simple kinematic car. The car model is given by Laumond (1998). The

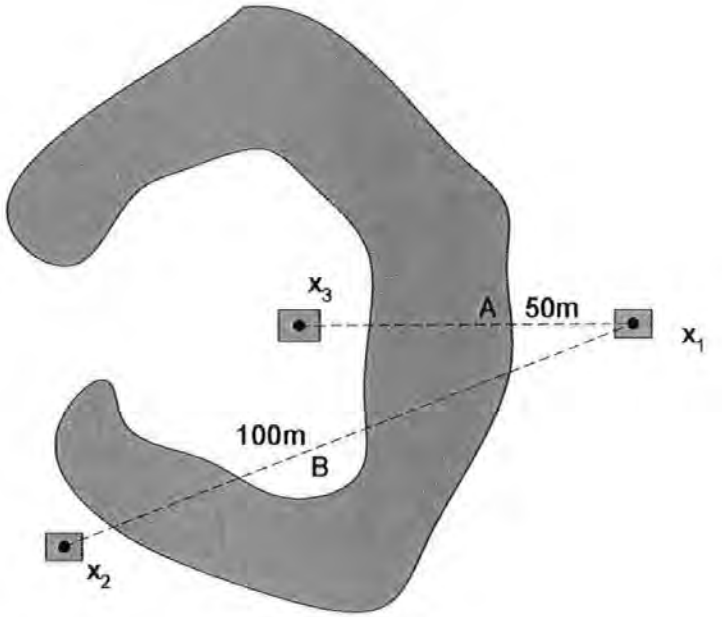


Figure 4.4: Euclidean metric giving misleading information with reference to the cost-to-go metric.

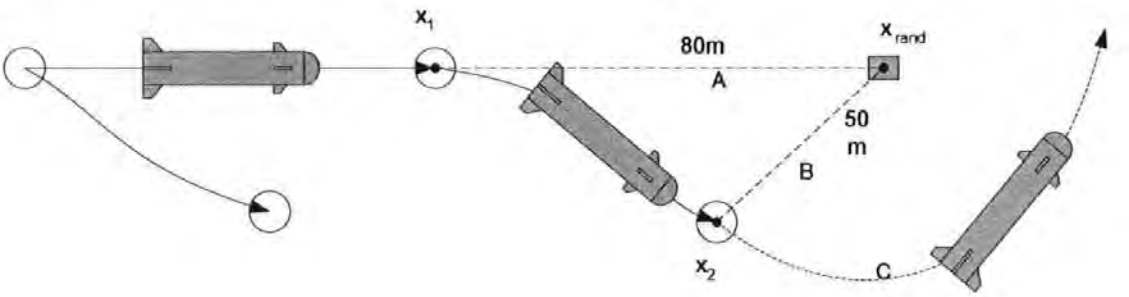


Figure 4.5: Euclidean metric differential constraints problem

heading control authority is via the front wheel and the steering angle is bounded to $\pm 15°$, acting as saturation constraints. The speed is held constant. Notice the numerous arc-shaped paths branching from the main tree, a clear evidence of the above mentioned problem.

Bruce and Veloso (2002) implemented a technique that is worth mentioning, they modified the distance metric to include not only the distance from the nearest state to the target state, but also from the root of the tree, multiplied by some gain values. A higher value of the gain value results in shorter paths from the root to the target

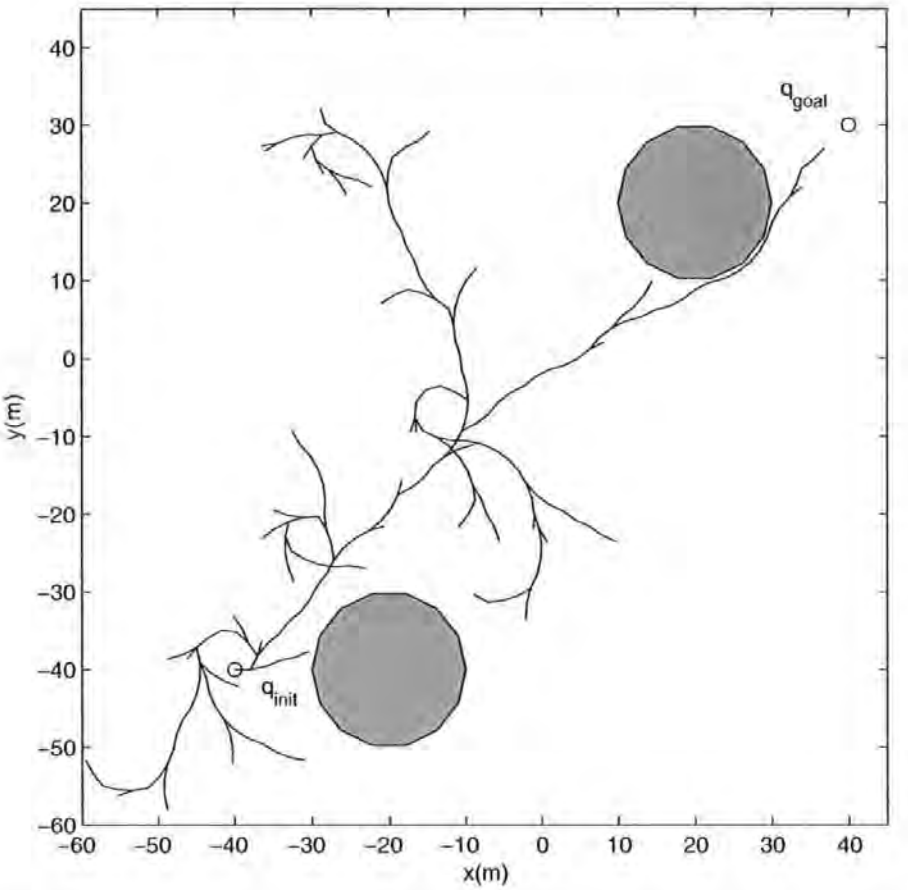


Figure 4.6: An RRT simulation based on a simple kinematic car model.

state, but also decreases the amount of exploration of the state space, biasing it near to the initial state .

One can now conclude that designing a good metric is a difficult problem and requires much knowledge in the problem domain. In addressing this issue, Cheng and LaValle (2001) have devised several methods to alleviate the dependency on the metric by applying information collected during the exploration, rendering the RRT less metric-sensitive and more robust. Briefly, the first method is to store the status of the 'expanded' inputs and the second one is to collect the collision violation frequency information as the RRT explores. Two of their proposed methods are also incorporated into the following simulations.

Exploration Information

The crux of the concept here is to acquire exploration information of each node to avoid node duplication and collision checking of the particular used state. Each node status concerning its input is recorded as either expanded or not expanded (Algorihtm 4.6). The expanded input will be excluded from the search space.

Constraints Violation Frequency (CVF)

The second method is to extract the environment information pertaining to the obstacles by recording the state collision tendency. This information is kept in the form of constraint violation frequency (CVF). The objective here is to avoid expanding the state in the region where a collision is bound to happen, hence biasing the search to the free space. Figure 4.7 is provided to assist the following exposition. When a child node of a state, $\mathbf{x}_d$ in this case, has collided (violated), a value of $\frac{1}{m}$ will be added to it. The CVF of the parent state, $\mathbf{x}_c$ will then be increased by $\frac{1}{m^2}$, this process propagates through the whole tree until it meets the initial state $\mathbf{x}^k$ with the CVF of $\frac{1}{m^k}$. The CVF is a monotonic increasing quantity, it starts from zero, but when a collision occurs, it is recorded and added to the existing CVF. A CVF of zero denotes that the all the child states are free to extend, diametrically, a CVF of one means no expansion is possible. Hence states with less CVF will be given more priority to expand since they are more likely to evade the obstacles. A more detail explanation is given by Cheng and LaValle (2001).

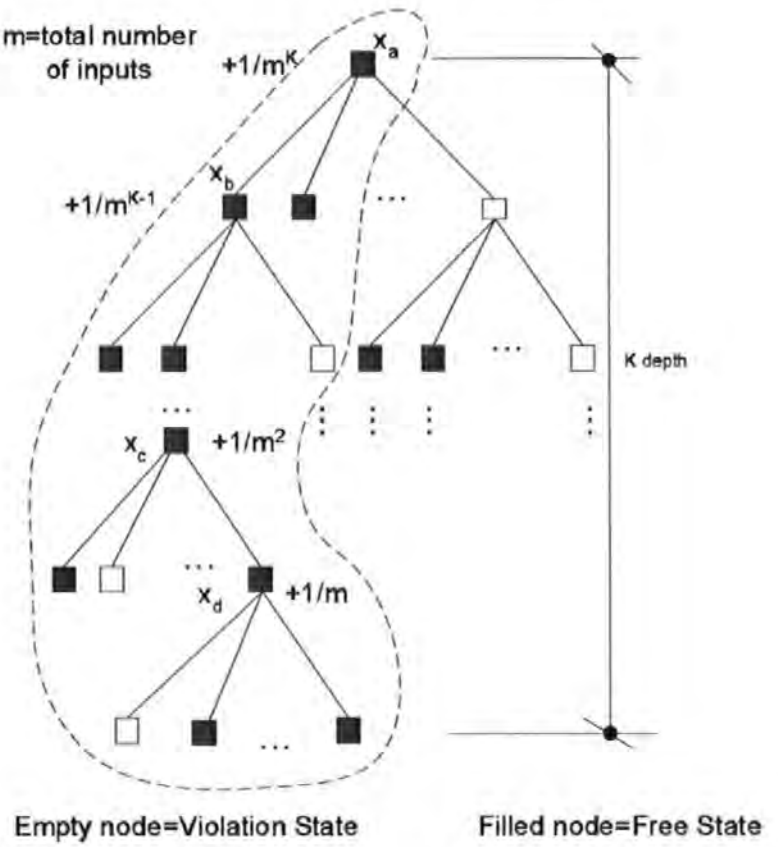


Figure 4.7: Constraint violation frequency

The calculation of the CVF can be computationally taxing if the tree is complex, or has a large input set and long input sequences. For each expansion, it is required to trace back all the parents until the origin node. It is decided after many experiments that ten backtracking depth (stages) are sufficient for practical implementation. The corresponding pseudocode is given in Algorithm 4.7. The corresponding CVF updating code is outlined in Algorithm 4.8.

4.4.5 Multiple trees, subconnection and tree prunning

Several researchers also advocate using multiple trees and tree pruning techniques to ameliorate the RRT performance (Li and Shie 2003). Indeed, the multiple trees RRT version does provide a fast solution but its effectiveness is limited to problems with algebraic constraints and holonomic systems. This is caused by the difficulty of connecting the trees without any gap in the presence of differential constraints.

Algorithm 4.6 (NEAREST_NEIGHBOUR2). This is an enhanced version of Algorithm 4.4. Here the expanded and CVF information are utilised.

Require: $\mathbf{x}_{rand}, T$
1: $d_{min} \longleftarrow \infty$
2: $d_{min'} \longleftarrow \infty$
3: **for all** $\mathbf{x}$ in T **do**
4: **if** $\exists \mathbf{u}$ of $\mathbf{x}$ are not EXPANDED **then**
5: $d \longleftarrow \rho(\mathbf{x}, \mathbf{x}_{rand})$
6: **if** $d < d_{min'}$ **then**
7: $d_{min'} \longleftarrow d$
8: $\mathbf{x}_{near'} \longleftarrow \mathbf{x}$
9: **end if**
10: $r \longleftarrow$ RANDOM$[0, 1]$
11: **if** $r > \sigma(x)$ **then**
12: **if** $d < d_{min}$ **then**
13: $d_{min} \longleftarrow d$
14: $\mathbf{x}_{near} \longleftarrow \mathbf{x}$
15: **end if**
16: **end if**
17: **end if**
18: **end for**
19: **if** $d_{min} \neq \infty$ **then**
20: **return** $\mathbf{x}_{near}$
21: **else**
22: **return** $\mathbf{x}_{near'}$
23: **end if**

Algorithm 4.7 (SELECT_INPUT2). This algorithm records the status of the inputs to avoid state duplications.

Require: $T, \mathbf{x}_{near}, \mathbf{x}_{rand}$
$d_{min} \longleftarrow \infty$;
$success \longleftarrow false$
for all u in **U do**
  if $\exists$**u** of U are not EXPANDED **then**
    $\mathbf{x}_{new} \longleftarrow$ NEW_CONFIGURATION$(T, \mathbf{x}_{near}, \mathbf{u}, \Delta t.)$
    if D$(\mathbf{x}_{new})$= TRUE **then**
      $d \longleftarrow \rho(\mathbf{x}_{new}, \mathbf{x}_{rand})$
      if $d < d_{min}$ **then**
        $d_{min} \longleftarrow d$
        $success \longleftarrow true$
        $\mathbf{u}_{best} \longleftarrow \mathbf{u}$
      end if
    else
      mark **u** as EXPANDED
      call UPDATE_TREECVF$(T, \mathbf{x}_{near})$
    end if
  end if
end for
mark $\mathbf{u}_{best}$ as EXPANDED
return $\mathbf{u}_{best}, success, \mathbf{x}_{new}$

Algorithm 4.8 (UPDATE_TREECVF). The following algorithm updates the CVF value of the tree using a recursive backtracking method.

Require: $T, \mathbf{x}_{near}$
$depth \longleftarrow 2$
$R \longleftarrow \frac{1}{m}$
$\sigma(\mathbf{x}_{near}) \longleftarrow \sigma(\mathbf{x}_{near}) + R$ {$\sigma(x)$ is a variable that records the CVF of x.}
$R \longleftarrow \frac{1}{m^2}$
$\mathbf{x}_1 \longleftarrow \mathbf{x}_{near}$
{Recursive trace back only 10 edges deep}
while $\mathbf{x}_1 \neq \mathbf{x}_{init}$ and $depth \leq 10$ **do**
  $\mathbf{x}_2 \longleftarrow parent(\mathbf{x}_1)$
  $\sigma(\mathbf{x}_2) \longleftarrow \sigma(\mathbf{x}_2) + R$
  $depth \longleftarrow depth + 1$
  $R \longleftarrow \frac{1}{m^{depth}}$
  $\mathbf{x}_1 \longleftarrow \mathbf{x}_2$
end while

Another interesting characteristic of the single tree RRT is its tendency to grow a few major branches at the initial state thus making connection with the terminal state very problematic. In obviating this problem, one method is to introduce another start tree, with a different time increment and metric when it is at the proximity of the goal states, thus improving the probability of connection (Kim and Ostrowski 2003). Bruce and Veloso (2002) devised a technique that utilises waypoint cache, a collection of feasible states from previous runs, for efficient replanning. Here the tree is bias using a parameter not only to the goal and free space, but also to the previous states. In essence, this technique attempts to reuse the information gathered in the previous run for a faster trajectory search.

4.4.6 Hybrid planner

Recalling from Section 4.4.4, the RRT tends to degrade as the ρ and ρ^* diverge. One promising technique initiated by Frazzoli *et al.* (2002) is to combine an optimal planner with the RRT. The optimal planner which exploits the precomputed trajectory primitives is used to plan an obstacle free path and the RRT attempts to reroute the path if there are obstacles. Detail simulation studies using a nonlinear dynamic model of a small helicopter has been presented. This technique is indeed very promising and is adopted herein. A more detail exposition of this novel method is presented in Chapter 5. Other researchers prefer to merge RRT with collocation and nonlinear programming (Karatas and Bullo 2001). The trajectories obtained via simulation studies show substantial improvement compared to the individual methods. Alternatively, Toussaint (2000) combined motion planning using the RRT with nonlinear control employing the H^∞ technique for an underactuated vehicle. Not only did he utilise a H^∞ filter for improving the planned motion of the vehicle but he also address multiple vehicles planning problems. His simulations, however are limited to only planar motions.

4.5 Reconnection

This study proposes a process termed as reconnection where the algorithm is initially executed to obtain a feasible trajectory which is then trimmed at a certain point and reexecuted again. This method exploits two inherent properties of RRT: (1) Its propensity to grow a few major branches from the initial point where these major branches are potential suboptimal trajectories. (2) Reconnecting the RRT entails recycling some of the residual branches thus achieving certain computational advantages compared to initiating a new tree. Certainly, two components need to be addressed: (1) the location of the trimming point and (2) the number of reruns required.

Strictly speaking, this method is a variation of tree pruning methods recommended by a number of researchers (LaValle and Kuffner 1999, Levine 2004). The novelty of the concept here is, instead of using this technique to assist in finding the goal, it is employed for optimising the trajectory with regards to a prescribed performance index.

In principle, a simple explanation of the algorithm flow is as follows: Once the first feasible trajectory is found, it is backtracked to the initial point. The trimming point is selected from 0.4 to 0.7 of the trajectory length. A value of 1.0 is equivalent to starting a new run since the whole core branch is trimmed. A too high value will risk destroying important branches and a too low value will not provide substantial improvement as the RRT will attempt to just reconnect the trimmed branch. Several experiments conducted by the author have indicated that two to three runs are sufficient to obtain a reasonable suboptimal trajectory. Additional runs will apparently deliver better solutions, but the improvement obtained is marginal, hence this effort is not being pursued to reduce any extra computing effort. Algorithm 4.9 is given below.

Algorithm 4.9 (TRIM). The following algorithm constructs an RRT, T, with K nodes

Require: $T, \mathbf{x}, \mathbf{x}_{init}$

1: $traj \longleftarrow$ BACKTRACK$(T, \mathbf{x})$ {Backtrack from $\mathbf{x}$ to $\mathbf{x}_{init}$, T}

2: $T \longleftarrow$ TRIM$(0.5 * traj)$

3: **return** T

Figure 4.5 reveals the consequence of applying the reconnection algorithm. Notice the first feasible path (grey colour) has been found and trimmed. The trimming coefficient is set to 0.5 in this case. The intrinsic RRT property of selecting the nearest node resulted in the extension of the longer untrimmed trajectory (dark colour). Both the trajectories are compared to select the shortest amongst the two. One can infer from the previous sentence that the current optimisation is in terms of distance, however, one can also optimise for minimum time or control effort. Amendments to the RRT algorithm are then incorporated to produce Algorithms 4.10 and 4.11.

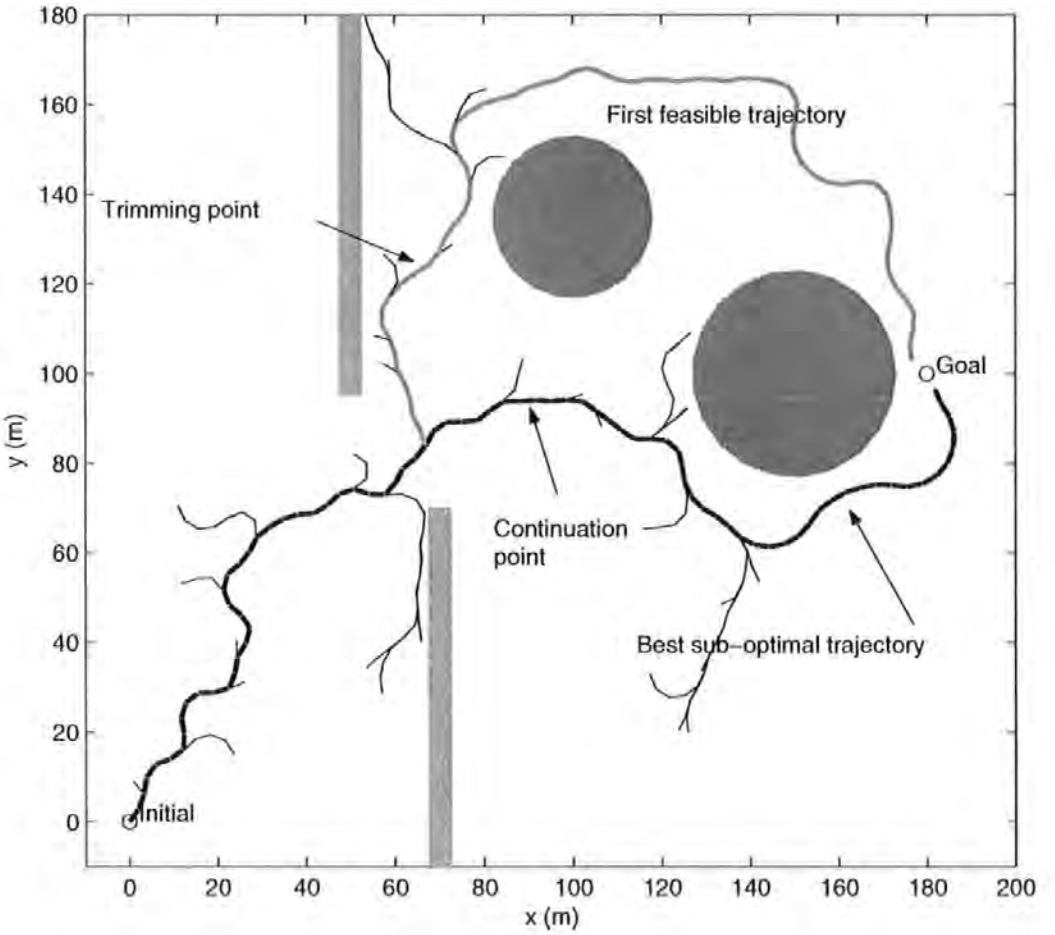


Figure 4.8: Motion planning with RRT in an environment populated with static obstacles. The reconnection process is also being depicted.

Algorithm 4.10 (BUILD_RRT2). The following algorithm constructs an RRT, T, with K nodes

Require: $\mathbf{x}_{init}, K, M$

$m \longleftarrow 0$ {initialise m, a global variable}
call T.init($\mathbf{x}_{init}$) {initialise tree, T.}
for $k = 1$ to K **do**
  $\mathbf{x}_{rand} \longleftarrow$ RANDOM_STATE(β) {extend T, see Algorithm 4.3.}
  call EXTEND_RRT2($T, \mathbf{x}_{rand}$) {extend T, see Algorithm 4.11.}
  $m \longleftarrow m + 1$
  {If $\mathbf{x}$ is within the goal tolerance.}
  if $\mathbf{x}_{new} \in \lambda$ **then**
    $T2 \longleftarrow$ TRIM($T, \mathbf{x}, \mathbf{x}_{init}$)
    $T2 \longleftarrow$ BUILD_RRT2($\mathbf{x}, K$)
    if $T2 > T$ **then**
      $T_{best} \longleftarrow T$
    end if
  end if
  {if maximum iteration limit is reached.}
  if $m = M$ is TRUE **then**
    break
  end if
end for
return T

Algorithm 4.11 (EXTEND_RRT2). The following algorithm extends a tree, T, towards x by taking a fixed step from the closest node in T towards x.

Require: $T, \mathbf{x}_{rand}$

$\mathbf{x}_{near} \longleftarrow$ NEAREST_NEIGHBOUR($\mathbf{x}, T$) {Find the nearest node in T to x, see Algorithm 4.6.}
$\mathbf{u}_{best}, success, \mathbf{x}_{new} \longleftarrow$ SELECT_INPUT($T, \mathbf{x}_{near}, \mathbf{x}_{rand}$) {Finding the 'best' control input,see Algorithm 4.7.}
{Checking for global constraints (collisions)}
if $D(\mathbf{x}_{new})$= TRUE **then**
  call ADD_VERTEX($T, \mathbf{x}_{new}$)
  call ADD_EDGE($T, \mathbf{x}_{new}$)
end if
return T

4.6 Simulation Results and Discussion

For these simulations, the sampling space was set to 200 m $\times$ 200m in dimension, slightly bigger than the environment, as illustrated in Fig 4.5. Here, one assumes an ideal case where *a priori* information of the environment has been catered and there is no external disturbance imposed by the environment. Admittedly, these two assumptions are not entirely true as far as practicability is concerned. The effect of an external disturbance will be addressed in Chapter 6. The *Remus* AUV model, as delineated in Section 3.2.1, was employed. The case of incremental sensing where only partial environment information is relegated to future work.

The convention here is to take the heading angle to start from the x axis (inertial) and positive when turn counter clockwise. The algorithms were all written in C on a 2.1 GHz Pentium IV machine, with 512 MB of RAM and running Windows XP. The algorithms were implemented in Matlab initially, before being ported to C language. Significant performance improvement in terms of running time, approximating 10 times the speed gain was observed after the language migration. Nonetheless, the codes were programmed without performance optimisation as a priority, clearly considerable speed increment can be expected if this is pursued.

The subsequent simulations were run with 2000 maximum nodes and 4000 iterations, terminating when either criterion is reached or if a solution is found. The AUV was assumed to be cruising at 1.5 m/s. The AUV configuration variables were set to [0 0 1], and the goal state to [150 100 κ] according to the following format $[x(m)\ y(m)\ \psi(rad)]$. Whereas, κ denotes a variable (unconstrained).

The Euclidean distance metric was employed for all the cases.

During preliminary simulation studies, all three configuration variables, x, y and ψ were used in the Euclidean metric but additional studies indicated adverse RRT behaviour. Upon further inspection, it was discovered that the RRT is indeed very sensitive to the weighting of the Euclidean metric, namely some states have been supplying contradictory information. For example, in an underactuated vehicle case, such as when an AUV is travelling in a straight path to a target, the configuration variables are in fact coupled. The AUV needs to align its heading and body such that

it coincides with the line of sight to the destination. More predictable behaviour was observed when the ψ variable is omitted. This is acceptable since the priority here is collision avoidance. However, if the final heading of the AUV is deemed important, then the heading state should be incorporated into the RRT algorithm.

The goal tolerance was defined as a $5m$ radius. This accuracy can easily be achieved via a modern GPS equipped with a WAAS or EGNOS when the AUV surfaced. Alternative one can utilise the SLAM technology instead. The time increment and dynamic equation time were set to $3s$ and $0.1s$ respectively. The Runge-Kutta method was used to propagate the dynamic equation. The rudder deflection input was discretised into 7 elements of input. Instead of assuming a constant input for Δt, the input was linearly interpolated as it propagates through the state equations. This method allows one to employ a larger time increment while easily taking into account the input rate constraint. All the plots shown are considered to be of an enhanced RRT unless stated otherwise.

Two forms of the RRT algorithms, the generic algorithm (LaValle 1998) and the enhanced algorithm, as given in Algorithm 4.1 and Algorithm 4.10 respectively, will be thoroughly compared. In order to verify the algorithm performances several testing scenarios must be appropriately designed. One shall rank the RRTs performance mainly using the four criteria as listed below,

1. The time where the first feasible trajectory is given. (Critical in real time applications.)
2. The quality of the trajectory reference to minimum distance travelled. (Reconnection)
3. The time required in providing a 'better' solution. (Applicable only to the enhanced RRT.)
4. The frequency of failure. (Unable to find a solution giving the require time or number of node constraints.)

4.6.1 Static environment

A sample of a feasible trajectory (bold) found by the enhanced RRT algorithm is depicted in Fig 4.9. Figure 4.10(a) shows its associated x-y-time plot, while (b) and (c) presents two identical CVF plot in different perspective for better visualisation.

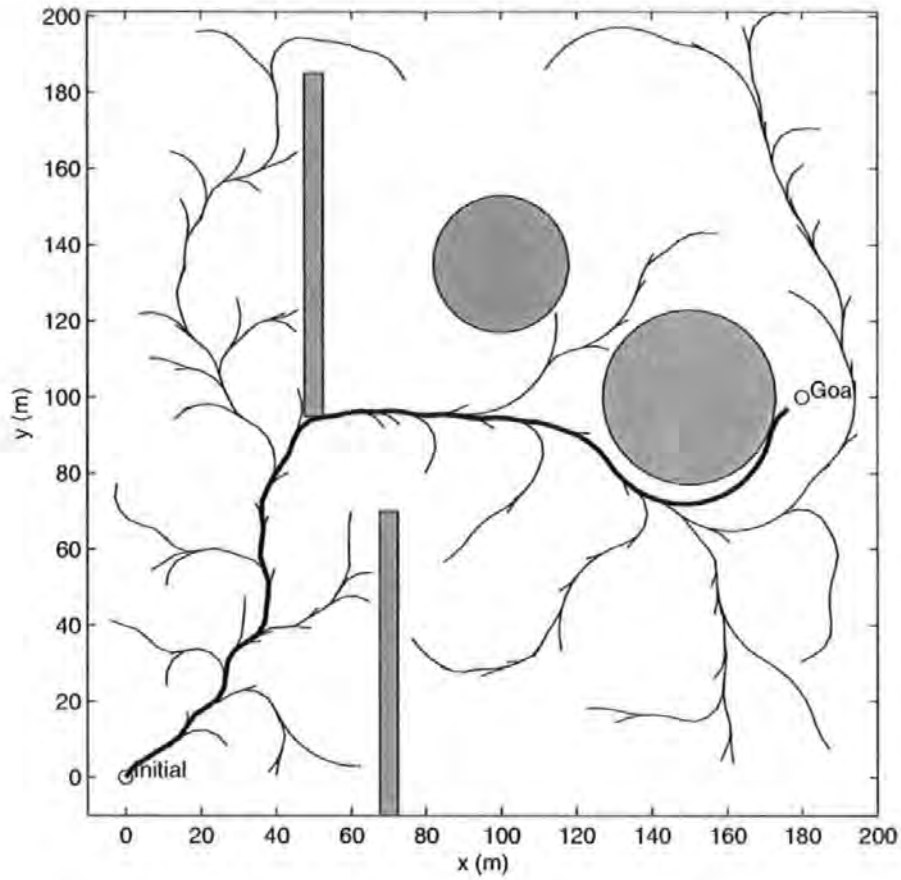


Figure 4.9: A feasible trajectory found in a static environment

The trimmed trajectory is not shown to avoid cluttering the view. Notice that the higher magnitude of CVF where the RRT collides with the obstacles. The rudder input history is depicted in Fig 4.10 (d). The chattering behaviour is apparent, and is not conducive for actuator life-span. Since the RRT is a randomised algorithm, a different run will yield a different result, therefore four unique feasible trajectories are provided in Fig 4.11 for comparison.

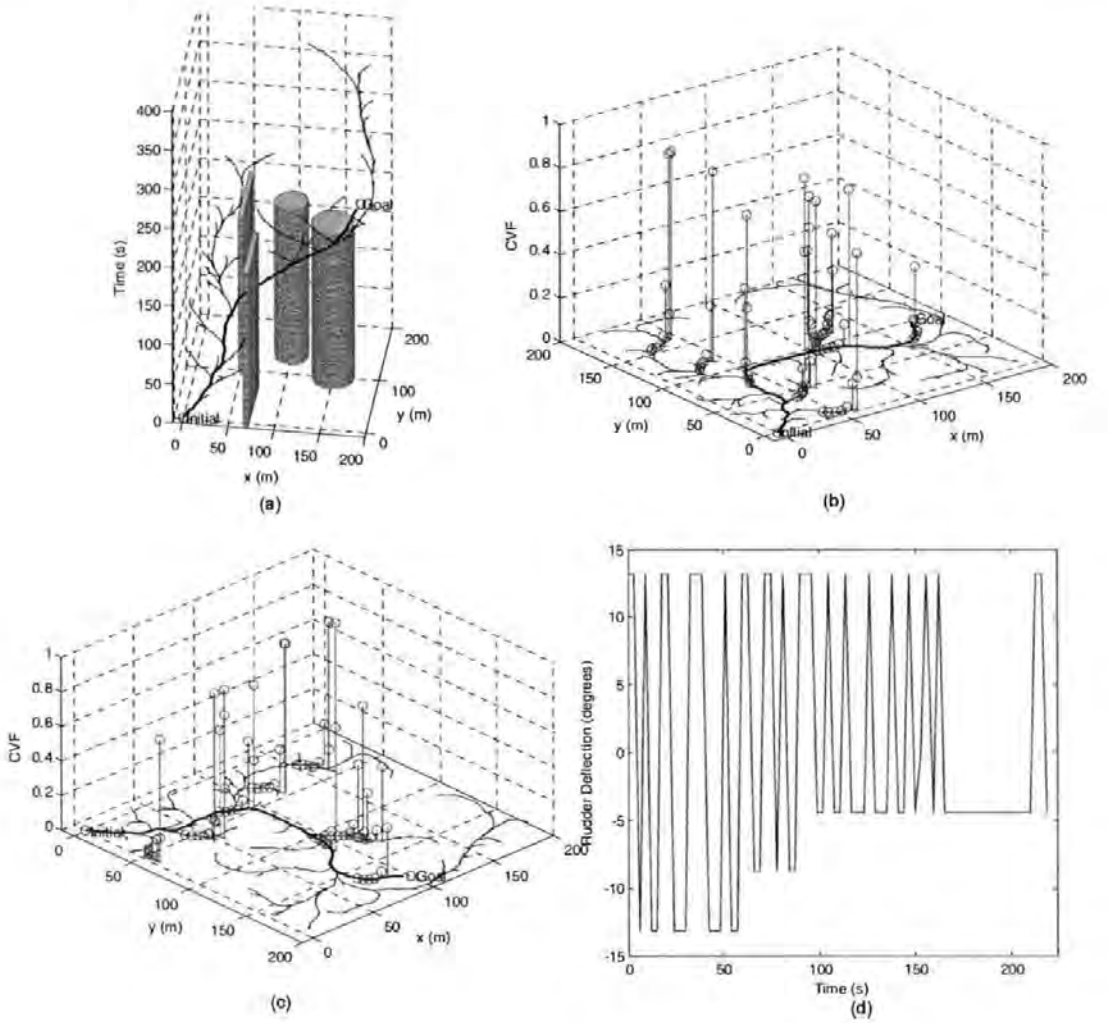


Figure 4.10: Static environment case (a)x-y-time plot (b)CVF plot (c)CVF plot (d) Rudder input history

Figure 4.12(a) shows a histogram plot of 100 simulation samples comparing both the enhanced RRT and a generic RRT performance. It is clear from the histogram that the enhanced algorithm returns the solution in shorter time, particularly for the first feasible trajectory. In Fig 4.12(a), it is observed that the enhanced algorithm out performs the generic RRT in terms of time response while returning the best suboptimal trajectory, shortest distance in this case (Fig 4.12(b)). The two means were then compared using a t-test, it is significance at the 0.01 level alpha whilst the 99% confidence interval for the true difference in means is [36.7 60.5]. However, one must be aware that trajectory selected remains suboptimal, future extension of the algorithm is likely to take this factor into consideration. A detail descriptive statistics

comparison of both the algorithms can be found in Table 4.1. The table indicates that the generic RRT has a higher failure rate. The failures are caused by the program reaching the predefined maximum nodes number or maximum iterations. Once again, the enhanced RRT performance is superior in comparison to the generic version. Median assessment provides a more unbias comparison for the skewed distributions, as evidenced by the histograms.

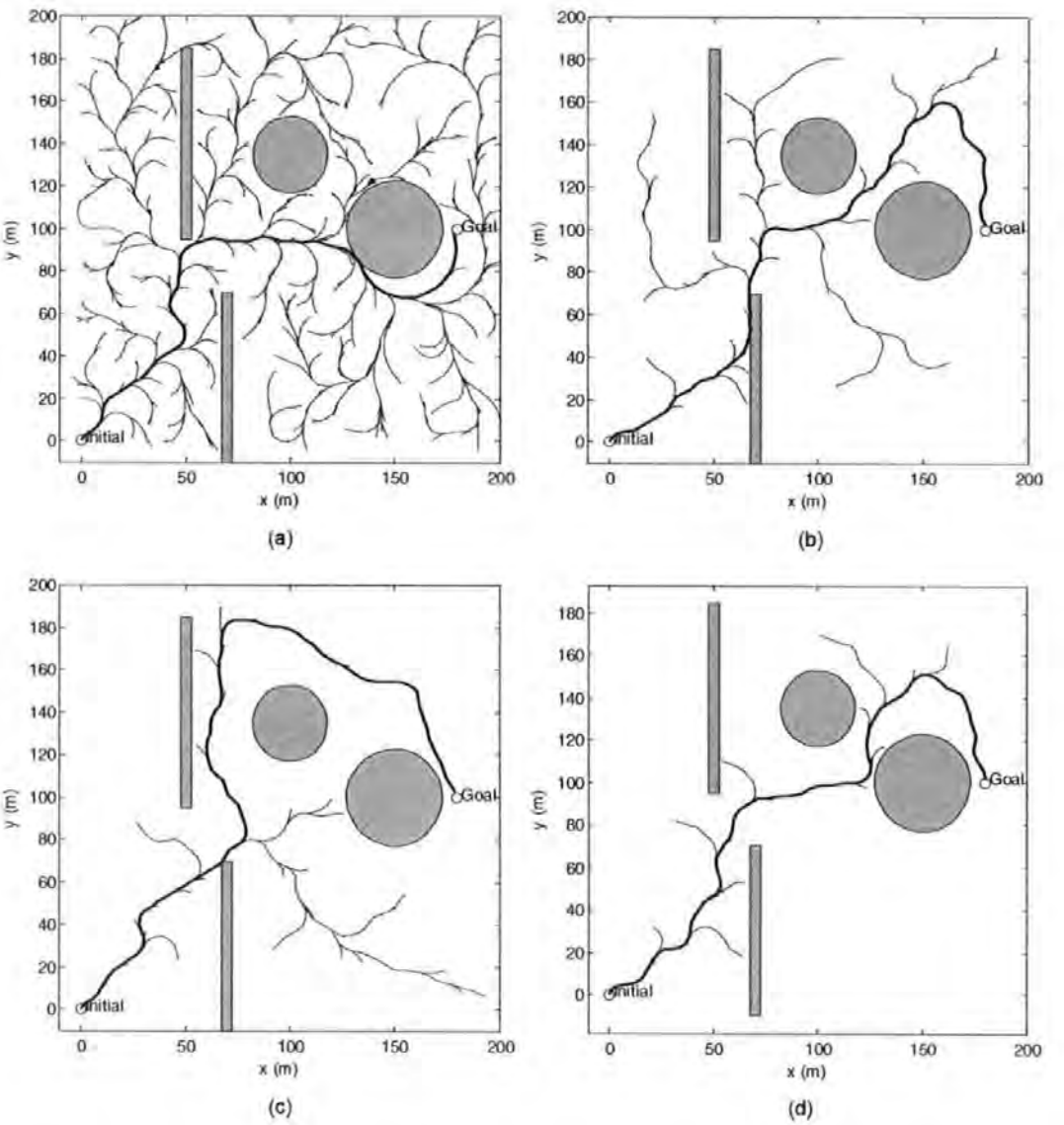
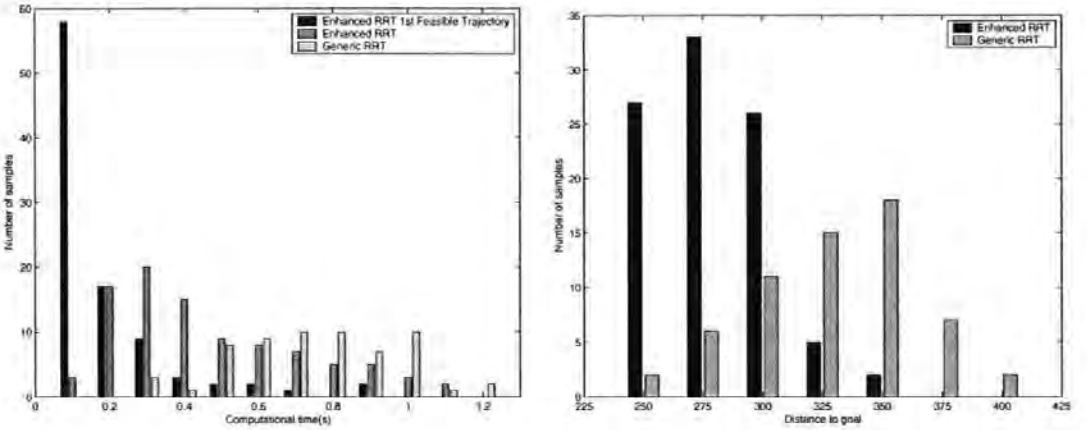


Figure 4.11: Four trajectories in different runs (static environment)



(a) Histogram plot of computational time (b) Histogram plot of total distance

Figure 4.12: Histogram results of 100 samples run (static environment)

Parameters	Enhanced RRT	Generic RRT (LaValle 1998)
Number of nodes used(mean)	870.5	1254.5
Number of nodes used(std)	397.5	515.5
Computational Time(mean, s)	0.46	0.74
Computational Time(std, s)	0.25	0.21
Computational Time (median,s)	0.32	0.86
Total distance(mean, m)	278.7	327.3
Total distance(std, m)	23.8	33.0
Total distance (median, m)	269.3	361.9
Percentage of failures	6%	39%

Table 4.1: Descriptive statistics collected from 100 samples run (static environment)

4.6.2 Dynamic environment

Figure 4.13 shows a more interesting environment where both the dynamic and static obstacles are present. The dynamic obstacles were assumed to have constant velocity. If required, the uncertainty of their position as time progresses can be replicated by expanding the obstacles size through time. Figure 4.14 indicates a feasible trajectory (bold) found by the RRT. Again, to assist in visualising the dynamic effect, it is plotted with respect to time in the z-axis (Fig 4.15 (a) and (b)), in different perspective. Figure 4.15(c) illustrates a plot of the CVF magnitude. Notice an increase in the CVF value of the corresponding node when it collides with an obstacle. This information allows RRT to behave in a more 'intelligent' way by avoiding an extension near to the collision area.

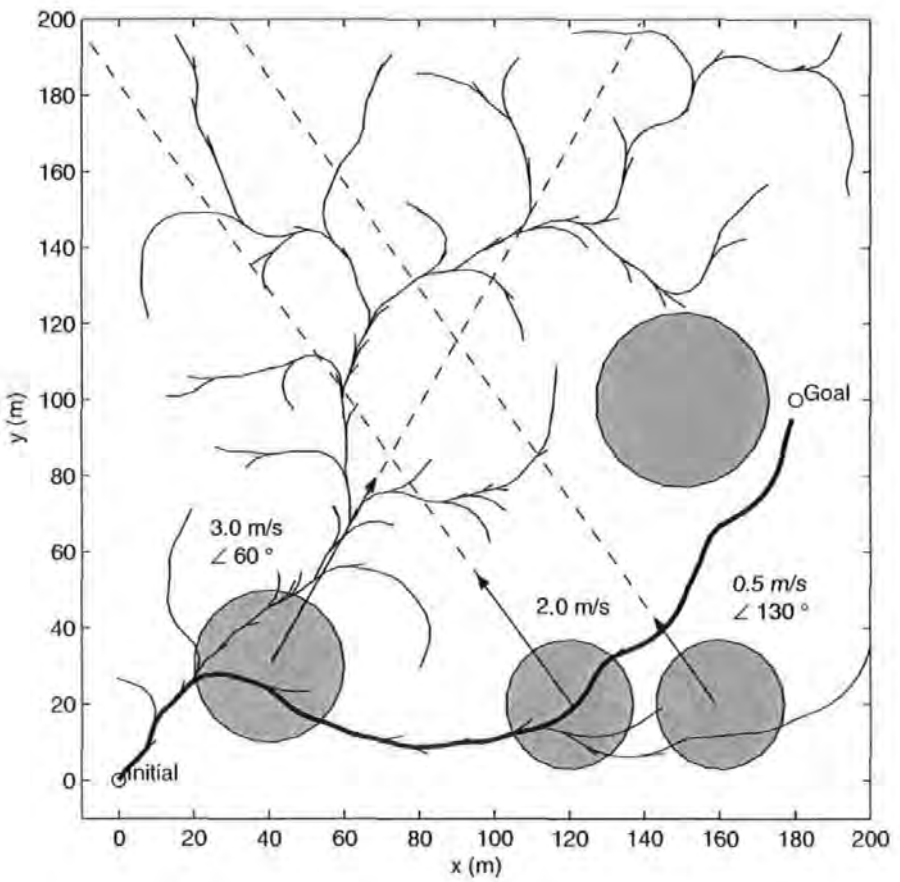


Figure 4.13: An environment with static and dynamic obstacles with their related parameters

The rudder input history reveals an average input that is biased to a negative rudder deflection starting at the mid-point of the trajectory till the terminal state (Fig. 4.15(c)). Notice, the rudder deflection tends to spend most of the time at approximately 8°. This outcome is indeed, in conformity with the shape of the feasible trajectory. Examination of the rudder input history illustrates a bang-bang control effect, this phenomenon is caused by a large RRT time increment and partly attributed by the nature of RRT which attempts to find the best input that can bring the current state nearer to the goal. Lastly, four unique feasible trajectories plot of different run are presented in Fig 4.16.

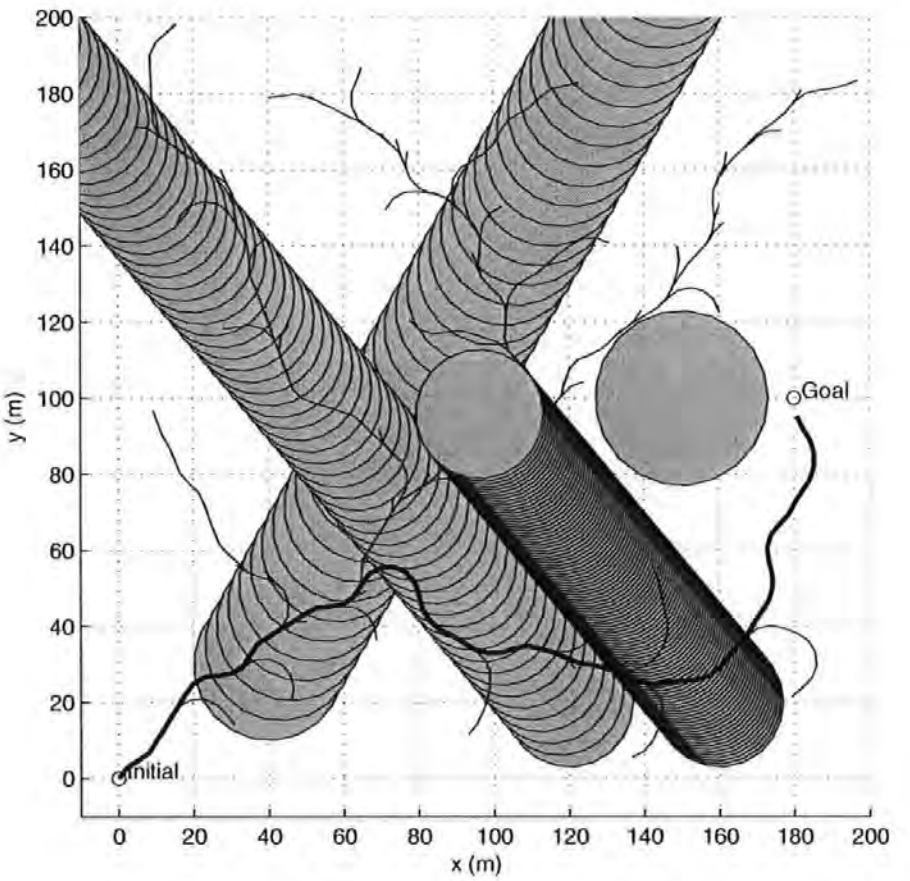


Figure 4.14: Feasible trajectory found in an environment with static and dynamic obstacles

Figure 4.17(a) shows a histogram plot of 100 simulation samples comparing both the enhanced RRT and the generic RRT performance. As in the static case, the enhanced algorithm still returns the solution in a shorter time, specifically for the first feasible trajectory. But for the overall time, both of them are more or less

equally matched. Nonetheless, as illustrated by Fig 4.17(b), the enhanced algorithm still outperforms the generic RRT in terms of minimum distance but the improvement is not as significant as the case of static environment.

A detail descriptive statistics comparison of both the algorithms can be found in Table 4.2. The table reports that both the RRT algorithms have near identical failure rate. It appears that this specific scenario is trivial for the RRT algorithms as noted from the number of nodes, time consumed and the failure rate. Once again, the enhanced RRT performance is better in comparison to the generic version but not by any substantial margin.

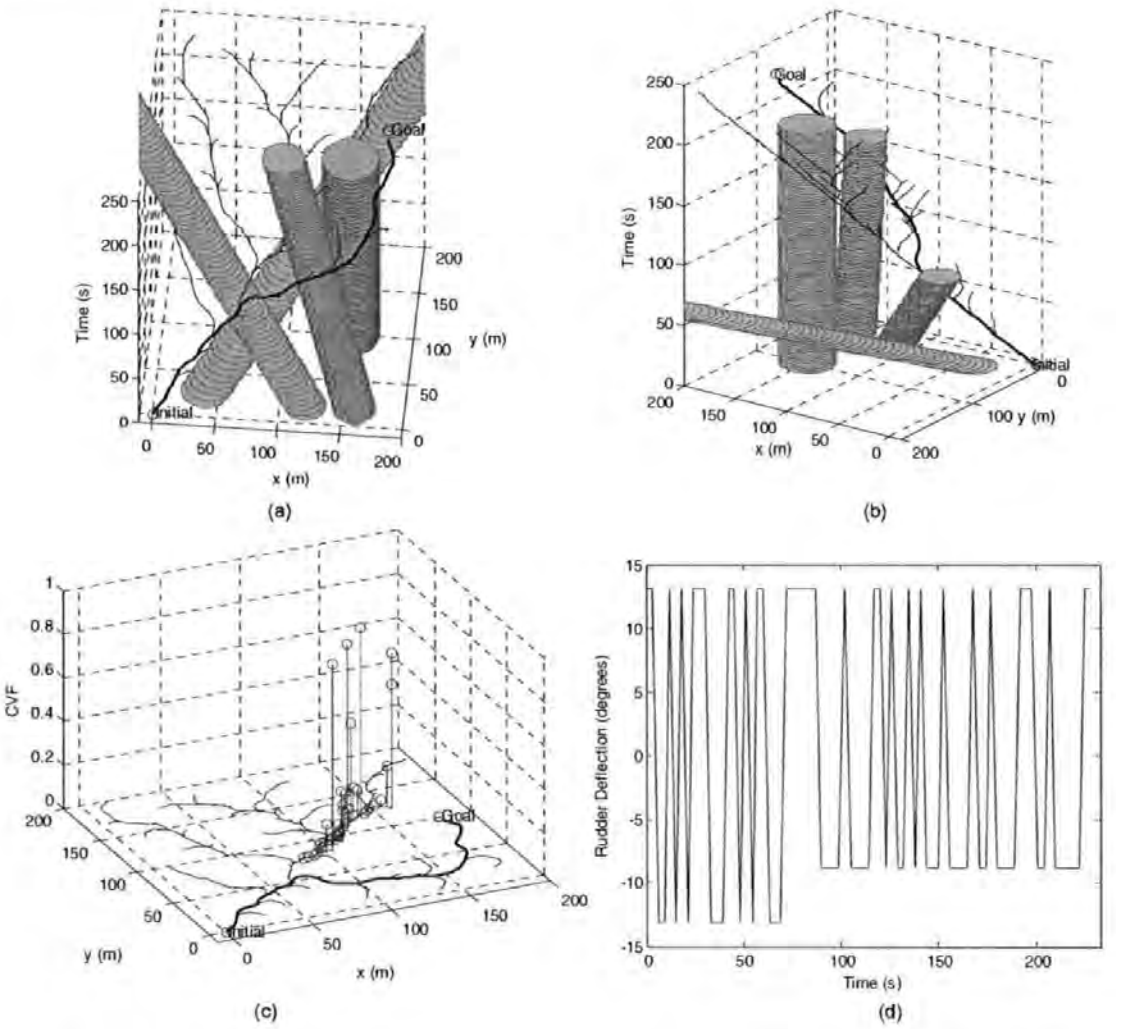


Figure 4.15: (a)x-y-time plot (b) x-y-time plot (c) CVF plot superimposed with trajectory (d) Rudder input history

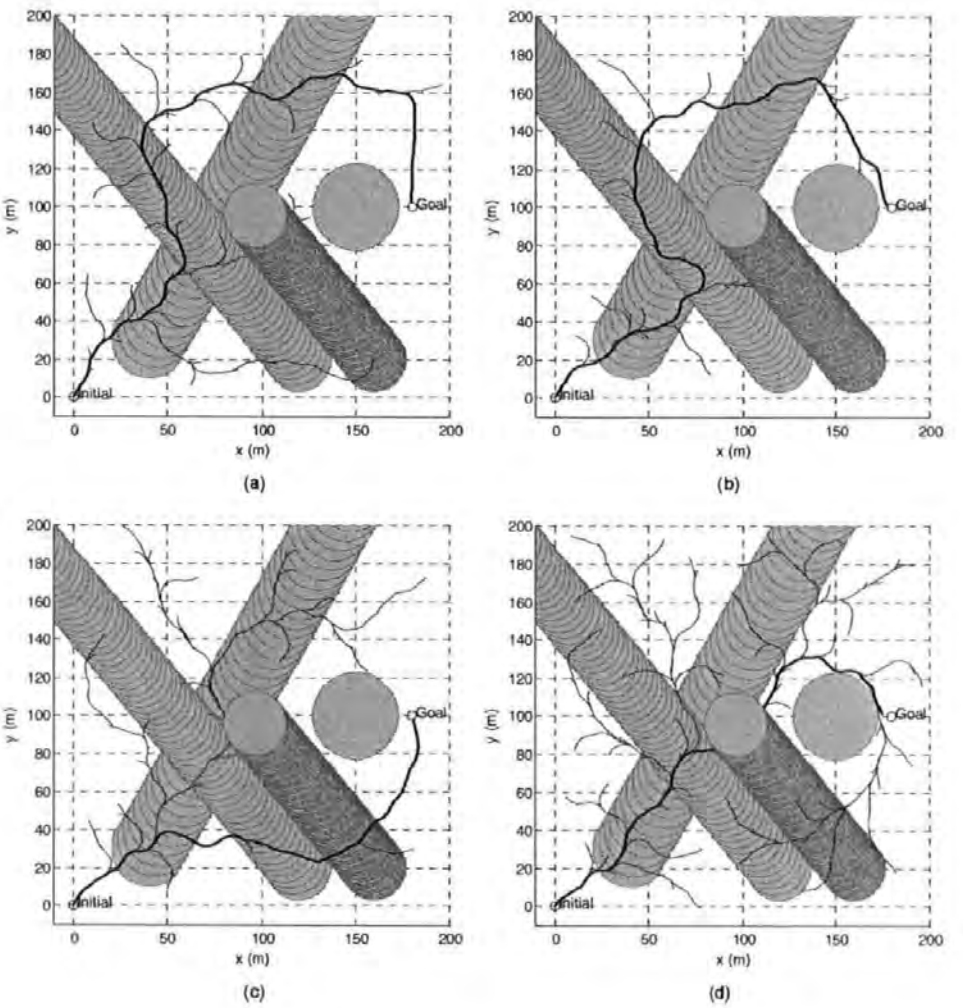
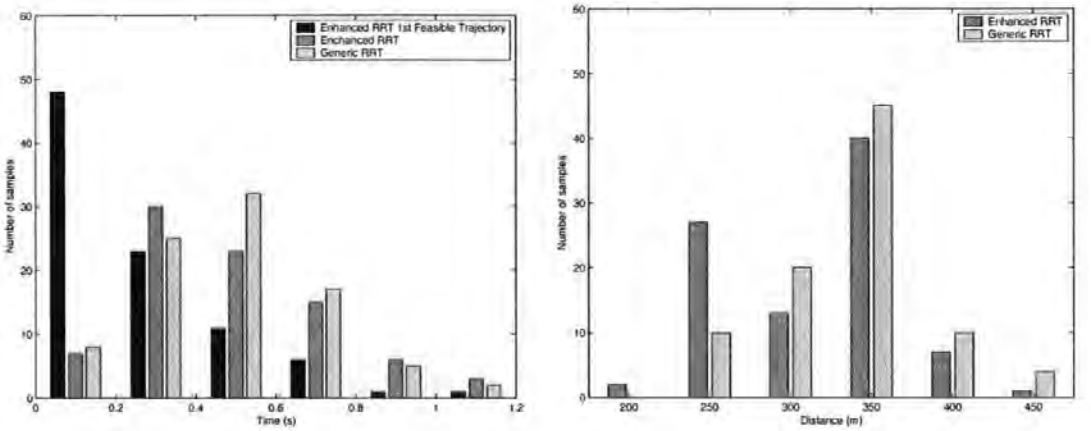


Figure 4.16: Four feasible trajectories in different runs



(a) Histogram plot of computational time (b) Histogram plot of total distance

Figure 4.17: Histogram results of 100 samples run (dynamic environment)

Parameters	Enhanced RRT	Generic RRT (LaValle 1998)
Number of nodes used(mean)	781.1	877.5
Number of nodes used(std)	284.2	232.3
Computational Time(mean, s)	0.49	0.47
Computational Time(std, s)	0.21	0.22
Computational Time (median,s)	0.46	0.51
Total distance(mean, m)	283.4	301.3
Total distance(std, m)	52.2	45.3
Total distance (median, m)	336.4	370.8
Percentage of failures	10%	11%

Table 4.2: Descriptive statistics collected from 100 samples run (dynamic environment)

4.6.3 Fault tolerant

Recently, there has been a sudden increase of interest in developing systems having very high fault tolerant capacity. One of a subcomponent of these systems is the fault tolerant control (FTC) system (Perrault and Nahon 1999, Sutton *et al.* 2001, Tortora 2002). These systems commonly have high reliability and are important in terms of operational cost and mission safety. The control system is designed such that, in the event of an actuator failure, the remaining ones can be re-tasked to compensate and the mission can be completed. This fault tolerant design, demonstrated on the *Theseus* and *ARC* AUVs greatly enhances the overall reliability of the system. Figure 4.18 reveals a simulation run of a case where the rudder is partially jammed probably due to seaweed and discarded fishing net entanglement or ice built-up.

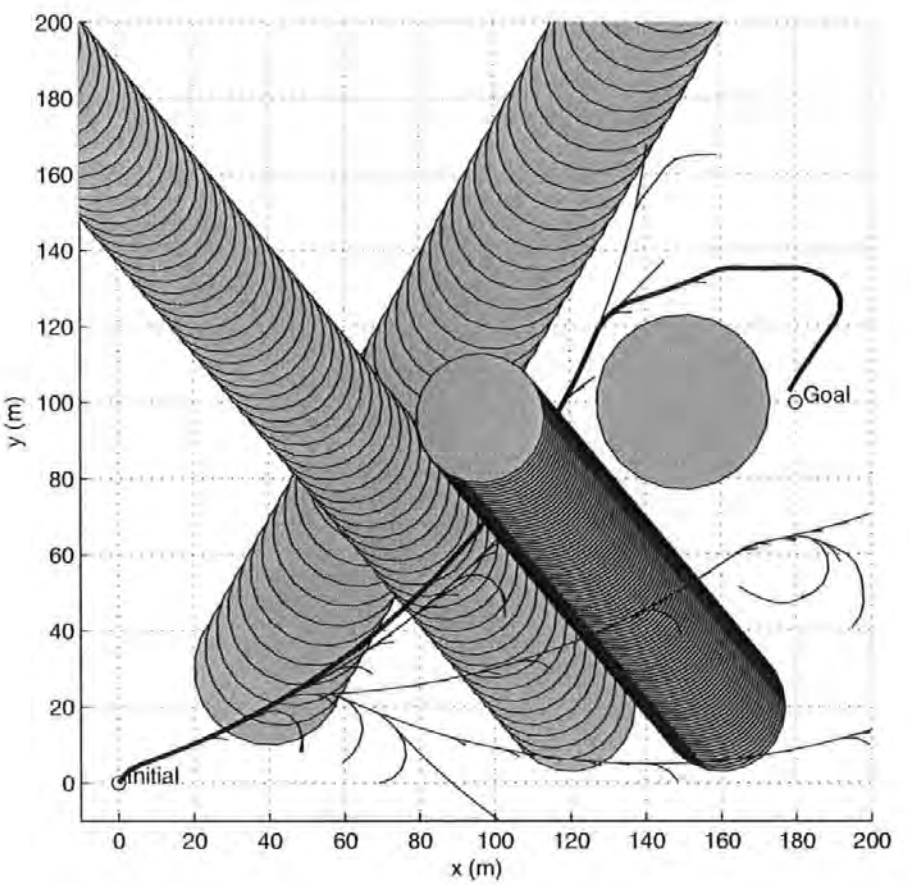


Figure 4.18: Plot showing a feasible trajectory (bold) for a crippled AUV

Instead of the normal rudder deflection range of $\pm 13.6°$, it is constrained to operate from $-0.5°$ to $+13.6°$. The effect can be deduced from the shape of the trajectories. The enhanced RRT algorithm found the goal in 3930 iterations and 1922 nodes in approximately $1.2s$. This clearly demonstrates RRT as a promising subunit of a FTC system. There is no doubt that the aforementioned system will increase the survival rate of AUVs operating in a harsh and unpredictable environment.

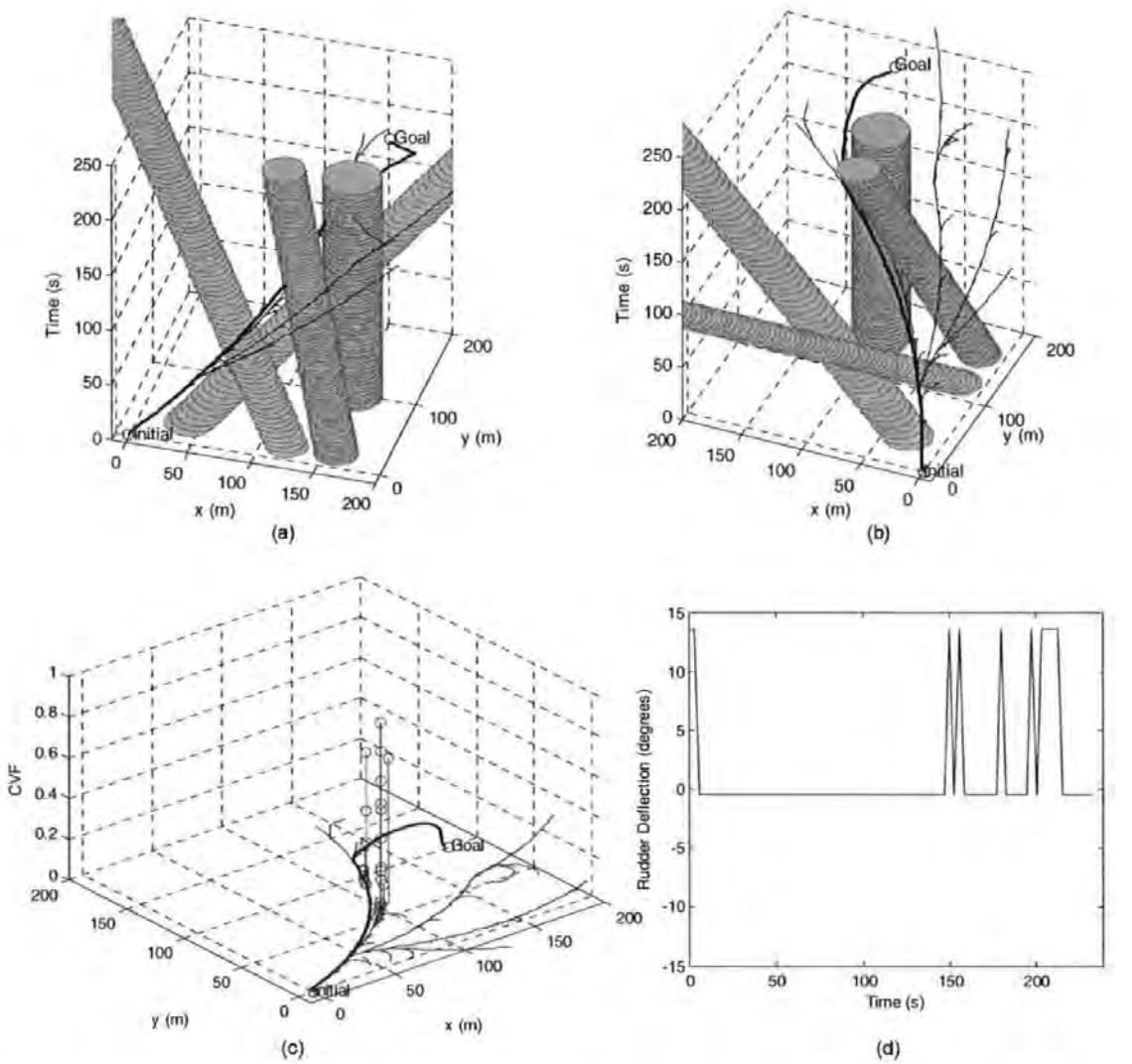


Figure 4.19: Case of a crippled AUV (a) x-y-time plot (b) x-y-time plot (c) CVF plot (d) Rudder input history plot

Figure 4.19 (a) and (b) depict the x-y-time plot from an alternate angle. Figure 4.19(c) shows the CVF plots. Examination of the rudder history (Fig 4.19(d)) shows

that the rudder deflection is now being restricted in a limited regime. Actuator saturation is detected and it can lead to complications in controller design and implementation. This can be avoided by prescribing a virtual bound (within the actuator saturation regime) in the RRT algorithm. This is required in practice since some control authorities must always be reserved in order to allow a proper functioning of the trajectory tracking controller.

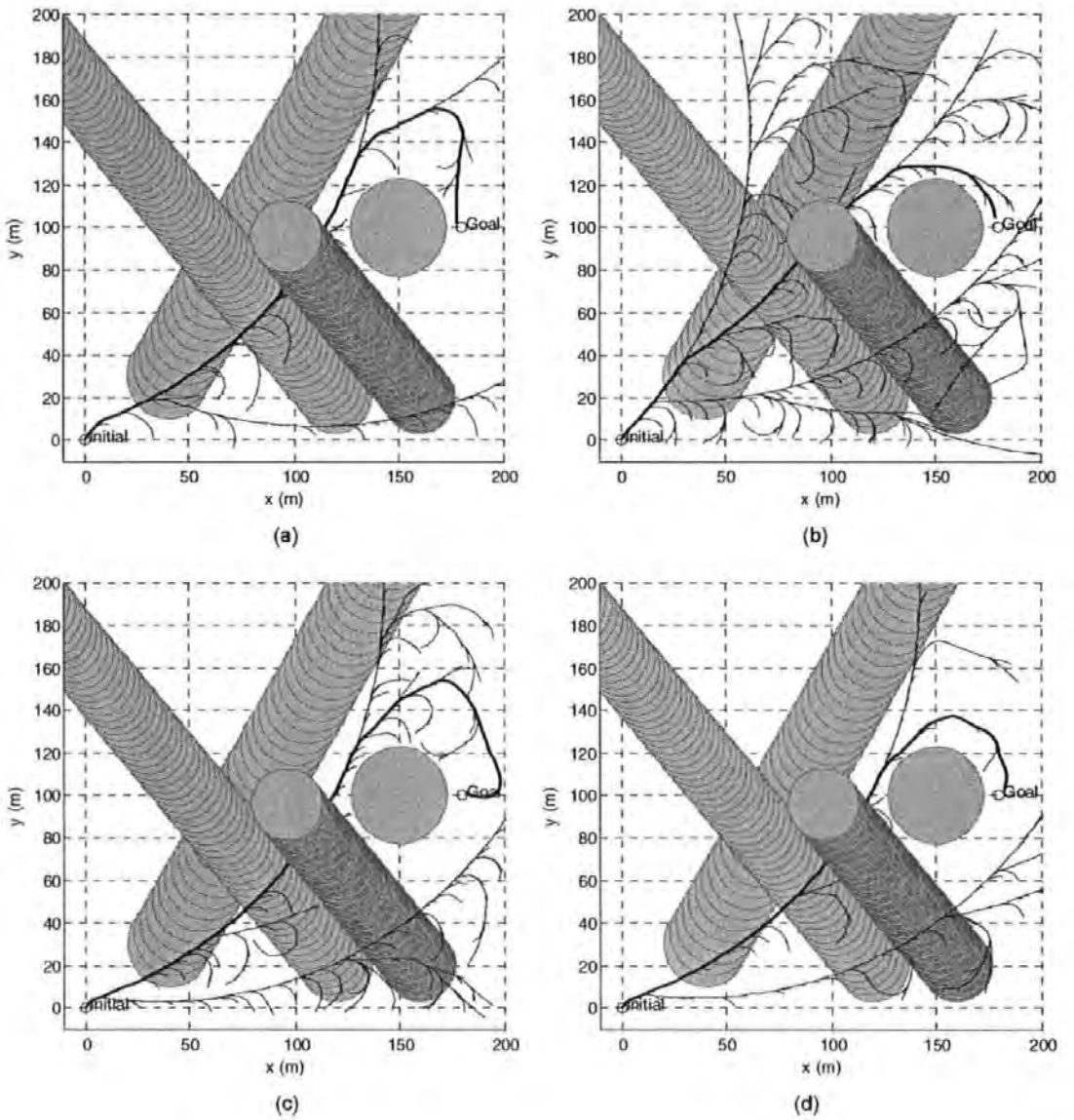


Figure 4.20: Four feasible trajectories in different runs

4.6.4 Disadvantages

The above appraisal has furnished plenty of in depth information pertaining to the RRT behaviour. Despite that the RRT based algorithm is endowed with various appealing characteristics, nonetheless a few of its critical deficiencies should not be overlooked.

One apparent drawback of RRT is the requisite for a comprehensive environment information, which is frequently not achievable in practice. Sonar equipped AUVs can only perform incremental sensing of the environment, gradually establishing and constructing the surrounding information. Bruce and Veloso (2002), however, have argued that the fast computational response time of the RRT allows one to rerun the program once a new target is confirmed or when an obstacle intersects the prescribed safety threshold of the AUV. Future investigation upon this issue is warranted.

Also, the RRT performance degrades drastically in a trapped environment such as those that have a small orifice or inlet. This symptom has been acknowledged by LaValle and Kuffner (1999). Figure 4.21 shows a trapped naive RRT tree.

Conversely, Fig 4.22 depicts a successful trajectory found using the enhanced RRT algorithm. Nonetheless, this particular simulation utilises 11071 iterations and 3091 nodes. A failure percentage of 78% is not encouraging even though it is much better than the generic RRT version which attained only 91%. There is little doubt that the accumulation of inputs and CVF information assist in making the enhanced RRT behaves more intelligently. Figure 4.23(b) and (c) which show the CVF plot superimposed with the trajectories in different perspective. A cluster of high magnitude CVF can be detected, in fact the increase in the CVF magnitude when the nodes collide with the obstacles is self evident, hence delivering crucial local information for the RRT success. The corresponding input history pertaining to the feasible trajectory found is plotted in Fig 4.23(d).

Another less obvious failing of the RRT, is its dependency on a system dynamic model, to be precise, an AUV dynamic model in this context. Exacerbating this problem is, the majority of commercial AUVs do not have an existing mathematical description of their dynamics. As remarked in Chapter 3 the process of extracting the mathematical model is rather time consuming and financially unattractive short

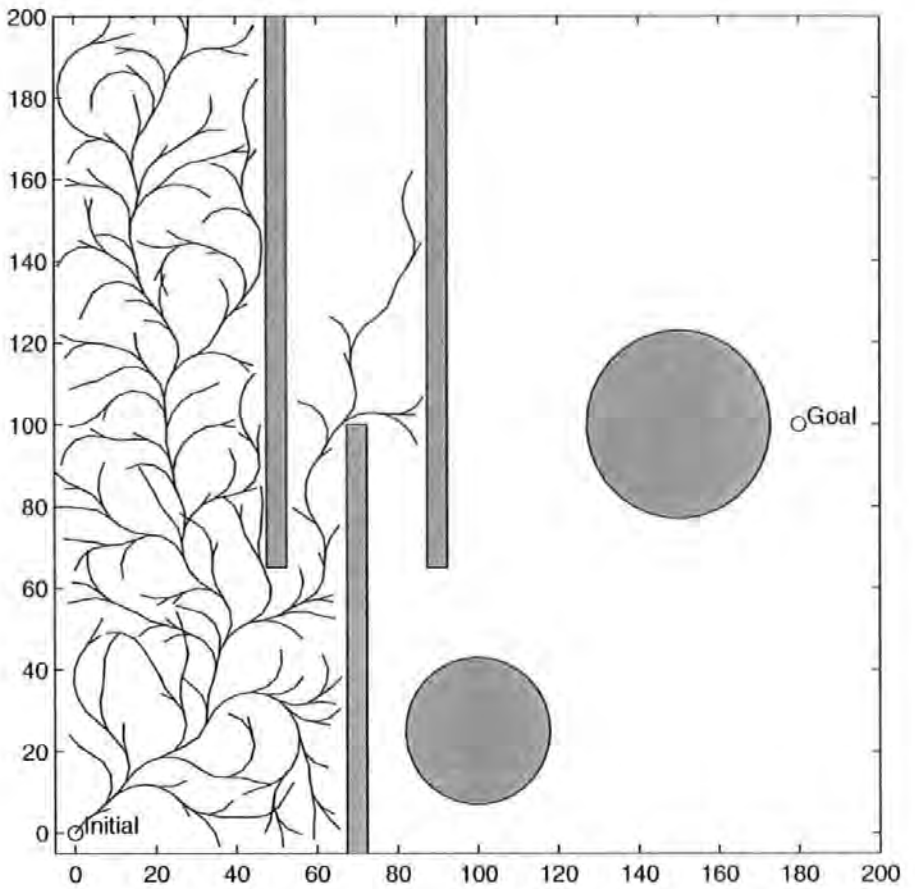


Figure 4.21: A trapped tree in a challenging environment (Naive RRT Algorithm)

term wise, hence explaining the apparent lack of industry enthusiasm in this aspect. Understandably, this model dependency issue must be addressed if one expects the algorithm to be useful for AUVs. Because of this reason, Chapter 5 is devoted to an interesting and yet pragmatic technique as a solution to this problem.

Apart from the above issues, all of the rudder input histories exhibit severe chattering behaviour, as evidently indicated in Fig 4.10(d) and Fig 4.15(d). Admittedly, the effect has been visually exaggerated by the time scale at x-axis of the plot. Even so, one cannot disregard the existence of this negative phenomenon as it is an inherent characteristic of all randomised algorithms. This chattering effect has a propensity to reduce the actuator lifespan and increase the energy consumption of AUVs. Mitigation of this effect can be attained by decreasing the time increment of the RRT algorithm. Smoothing of the control inputs with a low pass filter might be an alternative solution. Nevertheless, this will not only alter the trajectory and the terminal state, but defeat the RRT paradigm itself, which is to find a feasible trajectory for

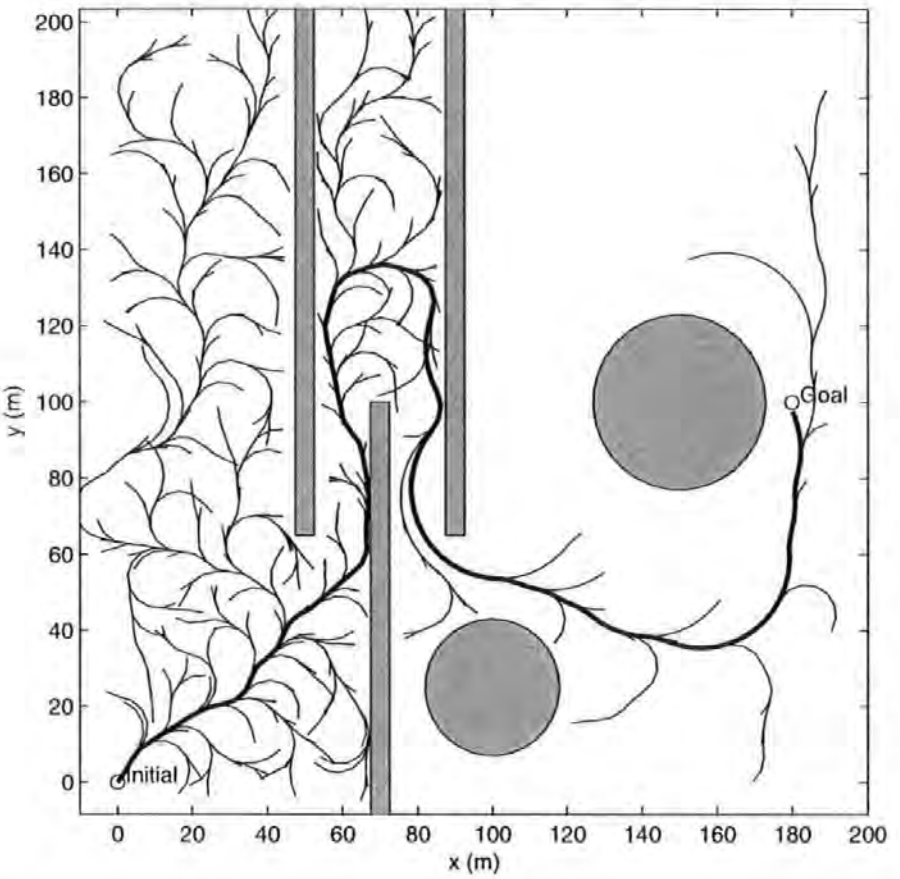


Figure 4.22: Successful trajectory found in a challenging environment (Enhanced) RRT Algorithm

a dynamic system without any auxiliary processes. Again, Chapter 5 will deal with this issue.

4.7 Summary

The objective of this study has been to compare and evaluate the feasibility of employing RRT algorithms in AUV motion planning problems. The chapter also introduced an improved version of motion planner for AUVs. The modified algorithm has been shown to be capable of generating feasible trajectories, satisfying both the algebraic and differential constraints. Its very short computational time makes it an ideal algorithm for real-time applications. The novel reconnection method, proposed in this chapter has been demonstrated to provide shorter trajectories. The RRT drawbacks

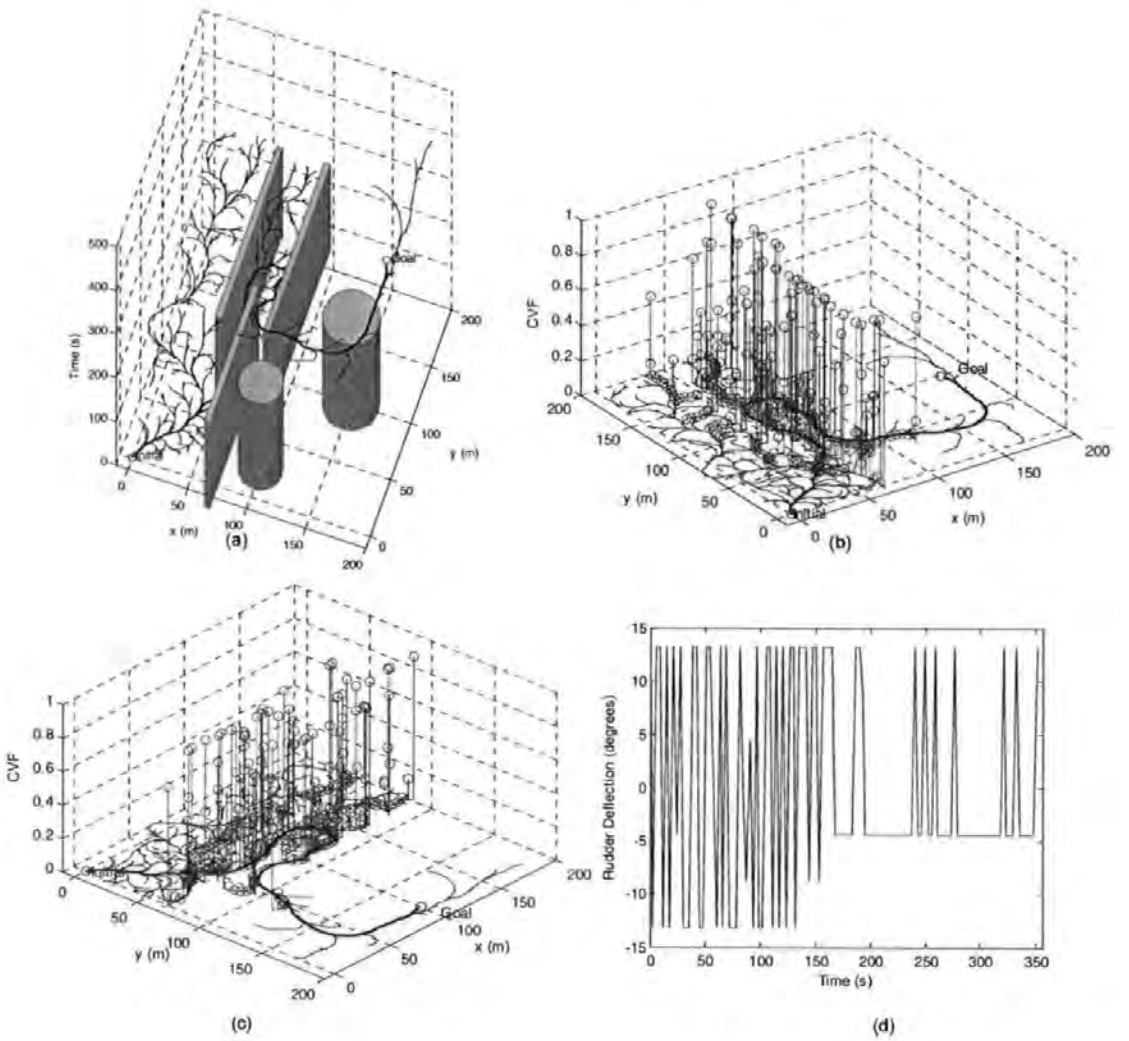


Figure 4.23: (a) x-y-time plot (b) CVF plot (c) CVF plot (d) Rudder input history

were delineated and discussed. Certain solutions to these problems are provided in the Chapter 5. It must be remembered that the algorithm is inherently a feedforward controller, a robust low-level feedback controller is needed to track the prescribed trajectory when subjected to external disturbance, as in practice. The designing of the tracking controller is nontrivial and is reserved for Chapter 6.

Chapter 5

The Manoeuvre Automaton and The RRT

Chapter 4 discussed and demonstrated the inherent ability of RRTs when applied to the motion planning problem of an AUV. It was noted that the preceding technique, even though highly appealing and promising, still elicits several drawbacks which are not conducive for practical implementation. In the light of this complication, this chapter focuses on a novel formulation, known as the Manoeuvre-Automaton (MA), a method based fundamentally on system dynamic quantisation. The MA when fused with the RRT is not only capable of addressing a few of the inherent deficiencies of RRT but greatly extends the algorithm functionality and versatility. This algorithm is then extended to the multiple-nested node version and also to cater for the case of a varying terminal state (Tan *et al.* 2005*b*).

5.1 Background

Here, one seeks a technique that can encode a set of finite behaviours of a dynamic system into a formal language or compact transcription for solving complex problems and of particular interest is the motion planning problem. This process is also termed as system behaviour quantisation or discretisation. Through this process, one derives a computational-efficient algorithm in the form of an embedded planner for

the generation of feasible trajectories, satisfying the required boundary conditions and differential constraints on the states and control inputs. Notice that here, one excludes the global constraints that are induced by surrounding obstacles, rather this will be tackled using the RRT paradigm. As an added benefit, it will be highly beneficial in practice if the feasible trajectory can be optimised based on meaningful measures such as time, distance and control effort.

Through the judicious application of the aforementioned technique, a distinct separation between the low-level and high-level control of a system is ensured. In other words, it introduces a form of abstraction to facilitate problem solving of high-level tasks. The ensuant high-level commands will automatically comply to the inherent behaviour of the system, and are guaranteed to be executable by the system. In addition, this permits the processor to allocate its resources to the more vital and challenging tasks, tasks that define the 'intelligence' of the system, as opposed to the low-level controls.

Such a form of a system behaviour quantisation as a whole reduces the complexity of the control task, but bounds to restrict the admissible responses of the system or limits the vehicle performance envelope when compared to the original one. This being a sacrifice that one must pay for adopting this method. Therefore, special care must be taken in conducting the quantisation process such that it captures the predominant behaviour of the system. This will ensure that when applied in practice, the difference between the quantised and the original is negligible.

In fact, the aforesaid technique is not entirely new and can be traced from the research conducted by Dubins (1957), and Reeds and Shepp (1990). They have shown that the minimum length paths for a kinematic car are comprised solely of straight line segments and tangential circular arcs of minimum radius, which implies that the problem can be recast as an optimisation problem. Alternatively, if one considers each of the paths can be symbolically represented, then a feasible path, which consists of a myriad of line segments can then be described by a series of symbols with the associated syntax, hence the term language. Unfortunately, for the case above the path curvature is related to the front wheels, and the car must stop at the path interconnection to reorient its front wheels thus rendering it unviable for practical usage. This research has been extended by Fraichard and Scheuer (2004) to obviate the requirement of intermittent stops at the interconnection of the line segments and

arcs. Their solution is, however, restricted to the domain of only car-like vehicles.

Behavioural robotics is a branch of robotics that does not use an internal model of the environment. It was instigated by Brooks (1986) and has received considerable attention even today (Arkin 1998). The gist of the concept is to develop a robot that would react to external stimuli, mimicking the behaviour of insects. Later, it was discovered that the exhibited behaviours are rather temporal, unpredictable, difficult to analyse and aimless. The situation is being further aggravated by the fact that the software are machine-specific and not reusable. This prompted the development of a language known as the 'Motion Description Language' (MDL) (Brockett 2000) (Manikonda *et al.* 1995), which was later extended to the 'Motion Description Language extended' (Hristu *et al.* 2000), that can provide a formal basis for programming behaviours and at the same time permit the incorporation of kinematic and dynamic models of robots in the form of differential equations. Indeed, the language provides a hierarchal approach to solve complex motion planning problems.

Deriving from the knowledge and experience gained from these previous studies, Frazzoli *et al.* (1999) introduced a method of state quantisation in the design of control systems, known as the 'Manoeuvre Automaton'. Instead of quantising time, the state or the control input values, the proposed technique is based on quantisation of the system's dynamics. Their approach is to select a finite number of state and control trajectories, termed as motion primitives, and concatenate them to generate feasible trajectories. From another perspective, it transforms a high dimensional, complex, nonlinear system into a hybrid system, which is more amenable in terms of computational and communication requirements. Their approach is capable of exploiting the symmetries properties found in most human 'engineered' vehicles. A detailed exposition of this technique is given in Section 5.2.

In a similar vein, purportedly in an independent study, Saimek and Li (2004) applied an almost identical method as the one proposed by Frazzoli *et al.* (1999) to the motion planning and control of an aquatic vehicle. The optimised motion plans are regulated by a controller that consists of a cascade of LQR, input-output feedback linearisation and sliding mode control. The novelty of their implementation is with respect to the use of time-scalable motion primitives. Experimental results which pertain only to the speed changing capabilities of the vehicle were presented, since turning behaviours have not been designed.

5.2 The Manoeuvre Automaton

The Manoeuvre Automaton, a form of finite state machine, was devised by Frazzoli *et al.* (1999) as a unified framework for formalising the control of high dimensional, nonlinear systems with symmetries. The principal idea here is to generate a complete trajectory via sequential combination of the copies of motion primitives from a library set. The main assumption behind the proposed method is that the vehicle dynamic equation must be time-invariant, and has a form that remains unchanged or in mathematical terms, invariant after the action of a certain class of transformation (group action) with respect to the states. Indeed, the later property is explicitly linked to the existence of symmetries. An avid reader is directed to the book authored by Bullo and Lewis (2004) for a more detailed exposition of symmetries in mechanical systems. Similarly, Frazzoli *et al.* (2004) advocated a mathematical rigourous approach to this subject with a special emphasis on its utility in the MA.

The MA method relies primarily on two distinct types of motion primitives known as *trim trajectories* and *manoeuvres*. Before delving into the details, it should be clear that herein, one is interested only on the planar motion of the vehicle. Subsequently, this resulted in the group $SE(2)$ acting on the configuration variables of the vehicle. The reasons behind these restriction have been stressed in Section 3.2. The generation of longer motion primitives from shorter ones can be accomplished through sequential combination or concatenation of the individual elements. Based on this assertion, a feasible trajectory is just a collection of repeatable motion primitives in a proper order.

5.2.1 Trim trajectories

Trim trajectories, also known as the relative equilibria for Lagrangian systems, corresponds to the steady state trajectories of a system, a vehicle in this context, where the velocities in the body-axes of the vehicle and the inputs are constants. Interestingly, trim trajectories are an intrinsic characteristic of human engineered vehicles, and include trivially all equilibrium points of a system. Since, each equilibrium point can be considered as a trim trajectory, it becomes the simplest form of a motion primitive.

Most vehicles are designed to have this property in mind. Imagine that when driving a car in a flat plane, once the vehicle speed is constant, very few control inputs are required from the driver to preserve the condition. The algebraic sum of the forces such as drag, friction and thrust acting on the vehicle is zero, hence the term equilibrium. Extra inputs are merely required to counter the effect of external perturbations. This condition, however, is nullified at the moment of driving up a hill, where the gradient varies. The gravity component acting on the vehicle destroys the symmetry in the pitch direction. Nonetheless, if the gradient is constant then a trim trajectory can always once again be found. Briefly speaking, trim trajectories are the composition of a constant rotation, in two dimensional space and screw motion (helix with a constant sideslip angle) in three dimensional space, where the group $SE(2)$ and $SE(2) \times (\mathbb{R}, +)$ acting on the configuration variables respectively for each case, assuming that gravity acts in the direction of z.

Frazzoli *et al.* (2004) also defined a trim primitive as a strongly repeatable motion primitive such that all of its non-trivial prefixes and suffixes are also strongly repeatable. As mentioned above, the control input must be constant, and the state flow is time-invariant. Trim primitives can be parameterised using a non-negative scalar τ, the coasting time which determines the duration to spend in executing a trim primitive. In other words, the system flows along the corresponding left-invariant vector field.

Mathematically, trim trajectories can be expressed in the Lie algebra form. The Lie algebra elements $\xi \in SE(2)$ are represented as matrices in $\mathbb{R}^{3\times3}$ of the form:

$$\xi = \begin{bmatrix} 0 & -\dot{\psi} & v_x \\ \dot{\psi} & 0 & v_y \\ 0 & 0 & 0 \end{bmatrix} \tag{5.1}$$

where v is the body fixed velocities with relative to the x and y axis respectively and $\dot{\psi}$ is the angular velocity.

In this case, the group exponential coincides with the matrix exponential and for a special case of $\omega = 0$, one yields a simple equation to describe the configuration

change after τ length of time in trim trajectory.

$$\exp^{\xi\tau} = \begin{bmatrix} 1 & 0 & v_x\tau \\ 0 & 1 & v_y\tau \\ 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

and if $w \neq 0$,

$$\exp(\xi\tau) = \begin{bmatrix} \cos(\omega\tau) & -\sin(\omega\tau) & c_x + r\cos(\omega\tau + \theta_0) \\ \cos(\omega\tau) & -\sin(\omega\tau) & c_x + r\cos(\omega\tau + \theta_0) \\ 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

where $r = \sqrt{v_x^2 + v_y^2}/\omega$ is the (signed) radius of curvature, and $(c_x, c_y) = (-r\cos(\theta_0), -r\sin(\theta_0))$ is the centre of rotation, with $\tan\theta_0 = -v_x/v_y$. For the planar motion case, the group $SE(2)$ acts on the configuration variables of the vehicle.

5.2.2 Manoeuvres

Manoeuvres are defined as a type of non-trivial motion primitive which can be sequentially combined at either end with the trim trajectories. Note that the initial and final conditions of the system are always assumed to lie on the trim trajectories. They are repeatable and can be sequentially combined. However, unlike a trim trajectories, their combination are confined by certain prescribed rules.

Manoeuvres are in fact a superset of trim trajectories or conversely, trim trajectories are manoeuvres that possess one unique property with their velocities and inputs being kept constant. As remarked, this excuses the slight abuse of the terms motion primitives and 'manoeuvres'. Both are frequently used interchangeably and their definition will be clear in the context. A manoeuvre is comprised of a complex connection of several motion primitives. They do not suffer from the restriction of constant input and constant velocity, implying that their state flows are not time-invariant. This allows manoeuvres to exhibit very complex behaviours in contrast to trim trajectories.

One invariant characteristic of manoeuvres is their group displacement. A planar underwater vehicle moving on a horizontal plane in a isotropic medium is invariant

with respect to rigid-body motions on the plane, $SE(2)$. The group $SE(2)$ can be identified within the space of a 3×3 matrix of the form:

$$g = \begin{bmatrix} \cos\psi & -\sin\psi & x \\ \sin\psi & \cos\psi & y \\ 0 & 0 & 1 \end{bmatrix} \tag{5.4}$$

This condition is violated if there exists constant currents, hence invalidating the isotropic assumption. The reason is that the symmetry about the vertical axis has been broken, rendering the system to be invariant to translations only. This can be extended to a three dimensional case when the medium is homogeneous, the system is then invariant with respect to $SE(2) \times (\mathbb{R}, +)$, assuming that gravity acts in the direction of z.

5.2.3 Hybrid formulation

In another perspective one can also recast the MA transcription into a hybrid form as given by Schouwenaars *et al.* (2003). The hybrid formulation provides a more elegant way of describing behaviour of the system. Similar to a differential or difference equation, the MA transcription describes a dynamic system, differing only in that it has hybrid elements in both its control inputs (τ, p), and state vector $(\mathbf{x}, q)$. MA evolves in a so-called 'dense time' by either continuous flows or discrete transitions. Consequently, at each particular moment, the system is constrained to be either in a trim condition q or performing a manoeuvre p. MA can be pictorially depicted as a direct graph $\mathrm{MA}(q, p)$ as shown in Fig 5.1, where q are vertices (trim trajectories), and p are edges (manoeuvre). Instead, the system behaviour can also be explicitly formulated as below.

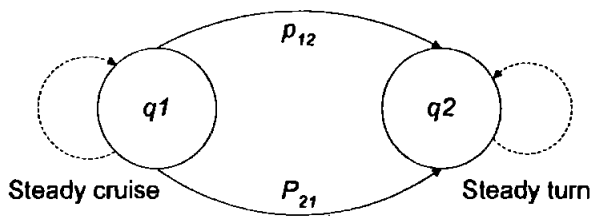


Figure 5.1: A simplified MA representation

- An MA system H starting at state vector $(\mathbf{x}_i, q_i)$ in trim trajectories, evolve according to $f_q(\cdot)$ as determined by the length of the τ_k, which can be infinite. Where $f_q(\cdot)$ is the governing differential equation at the specific discrete state q_k. The hybrid state then evolves as:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_k \dot{\mathbf{x}}_q \tag{5.5}$$

$$q_{k+1} = q_k \tag{5.6}$$

$$t_{k+1} = t_k + \tau_k \tag{5.7}$$

  where $\dot{\mathbf{x}}_q$ is the time rate of change of the vehicle's continuous state variables and k is the 'stage' number.

- In the case of performing a manoeuvre p, the vehicle leaves the trim trajectory q_1 for a finite length of time before settling to the trim trajectory q_2. Mathematically, the manoeuvre is initiated by the control action p, which is discrete, and is described by a fixed duration Δt_p and displacement $\Delta \mathbf{x}_p$ in the continuous state space, as illustrated in Fig 5.2 for a $SE(2)$ case. In reality, the control history of the continuous state-space system is implicitly encoded in the control action p. As such, when manoeuvering, the hybrid state evolves as:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}_p \tag{5.8}$$

$$q_{k+1} = q_2 \tag{5.9}$$

$$t_{k+1} = t_k + \Delta t_p \tag{5.10}$$

Although, the hybrid control input at instant k can be described by a vector $(\tau, p)_k$, however only one input, either τ or p can be active at any moment.

By having the AUV continuous behaviour encoded as a discrete state q, its configuration can be described by an element of the Lie group G of rigid motions in $\mathbb{R}^2$ or $\mathbb{R}^3$, called $SE(2)$ or $SE(3)$, respectively.

5.2.4 Motion plan

The sequential combination of motion must be performed in order, to be more precise, it must abide to certain rules. The rules are synonymous to the grammar of

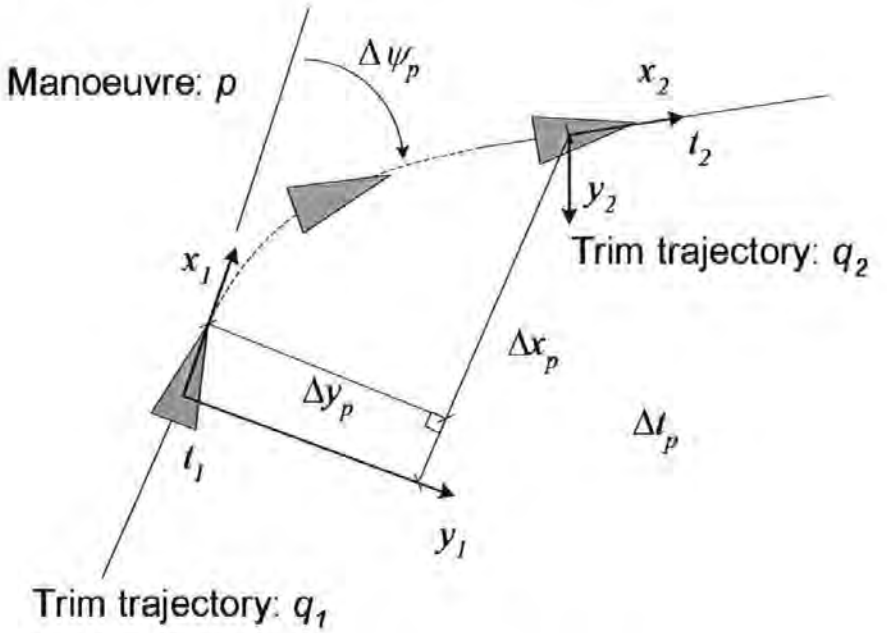


Figure 5.2: Displacement of configuration variables and its time duration for manoeuvre p

a language, and are commonly encoded via a finite state machine (FSM), which is called the MA.

As mentioned before, an MA can be conveniently depicted as a directed graph in which vertices represent states (trim primitives) and edges represent manoeuvres. Figure 5.3 shows a simplified MA of a hypothetical AUV, operating in a three dimensional, homogenous and isotropic medium. All the vertices are trim manoeuvres such as *cruise*, *turn left*, *turn right*, *climb*, *dive* and *stop*. Strictly speaking, *stop* cannot be considered as a trim trajectory since most AUVs suffer from degrading control authority at low speed and reaching zero control authority at stationary. The edges are the manoeuvres that must be performed to arrive at particular trim trajectories. The initial and final conditions for the motion planning problem are such that they can be represented as trim primitives. The trim primitives dictate the allowable set of manoeuvres that can be performed at a particular moment.

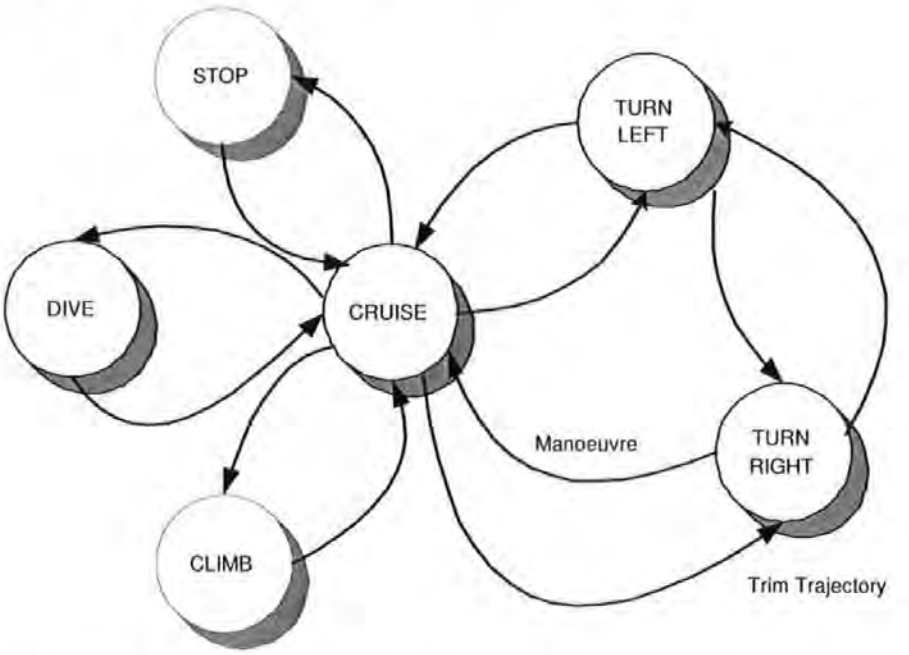


Figure 5.3: MA representation of the dynamics of a hypothetical AUV

The language generated by the MA is coined as the manoeuvre sequence. The beauty of the language is that it allows one to express mathematically the motion plan in a compact and precise manner. One can describe the configuration change resulted of a manoeuvre p using Equation 5.4. Thus, through the application of the MA representation, one can now express the system configuration by concatenating these motion primitives. The equation can be compactly expressed as below:

$$g_f = g_0 \left[\prod_{k=1}^{N} e^{(\xi_k, \tau_k)} p_k \right] e^{\xi_{k+1} \tau_{k+1}} \tag{5.11}$$

Where g_0 and g_f are the initial and final configuration. $e^{(\xi_k, \tau_k)}$ and p_k represent the transformation of applying the k-th trim trajectory and the k-th manoeuvre, respectively. One appealing characteristic of this transcription for the purpose of motion planning is the ability to recover in closed form the complete state of the system at any time during the execution of a motion plan.

As shown in Equation 5.11, this allows one to concentrate on the configuration variables, which are the most important states in motion planning, instead of the velocity states. It behaves like a kinematic map of a robot and produces paths that auto-

matically comply to the system dynamics. Furthermore, there is no approximation involved in the transformation from a nonlinear continuous system into a hybrid system via the MA approach. Nonetheless, it does restrict the admissible dynamics of the original system.

5.2.5 Optimisation

The MA transcription is highly suitable for an optimisation process. This is true, for certain cost function that shares the symmetry properties of the system, such as minimum time, minimum length (distance) or minimum control effort. Nonlinear optimisation and randomised techniques are possible candidates but their high computational demand impedes their use in this context. These techniques include sequential quadratic programming (SQR), genetic algorithm (GA) and simulated annealing (SA). Reformulation of the above problem into Ricatti equation is not possible either, as the dynamic equations are not continuous but hybrid instead. Similarly, pure gradient based optimisation fails because of the discontinuity in the hybrid equations. One should not forget the fact that the main idea of recasting the dynamic equations from continuous form to hybrid form is to render it amenable for computation. Given this situation, dynamic programming (DP), linear programming (LP), and mixed integer linear programming (MILP) have the most potential.

Dynamic Programming

The DP technique was invented by Bellman (1957) to solve the optimal control problems. The gist of the concept here is to store the cost-to-go map of the optimisation problem that is performed *a priori* and then exploit a look-up subroutine to find the subsequent optimal states in real-time. Interestingly, it can be applied to both linear and nonlinear optimisation problem alike without any alteration. Frazzoli (2001) has adopted this technique for solving the MA problems. It was later extended by Schouwenaars *et al.* (2003) to the design of a more robust system. They used the standard deviation of each manoeuvre to quantify the uncertainties.

Regretfully, the DP suffers from the notorious state explosion effect, and the gen-

eration of the cost-to-go map is rather time consuming, taking a few hours to days depending on the state dimensions of the problem in hand. This technique is not conducive for the implementation of fault tolerant systems. As in the unlikely event of any system's actuator malfunction when conducting mission, it is imminent that the system dynamics will change, and the cost-to-go must be recalculated. The time required to do this can be intolerable. To circumvent this problem, neuro-dynamic programming has been suggested by Bertsekas and Tsitsiklis (1996) as a promising substitute.

Linear Programming

Indeed, for the unique case when $\dot{\psi} = 0$, such that when all the trim trajectories are translations, the cost is linear relative to the coasting variables τ, thus one can employ the linear programming method instead (Frazzoli 2002*a*). The configuration variables change in trim trajectories can be described using Equation 5.2, without the transcendental functions as in Equation 5.3. For the specific case of a minimum time cost functional, one can formulate it as below:

$$\min_{p_k,\tau_k} \sum_{k=1}^{N} (\Delta t_{p(k)} + \tau_k) \tag{5.12}$$

such that Equation 5.11 is satisfied and $\tau \geq 0$. An extension to the minimum length and minimum control effort cases is trivial. Unlike the aforementioned DP, in LP the optimisation process operates in real-time. It does not require a lookup-subroutine to find the subsequent state based on an cost-to-go map performed *a priori*. This ensures that any dynamic alteration of the system can be accommodated by the optimisation algorithm, assuming that the appropriate motion primitives are provided.

To elaborate, a linear program is a problem of minimising, or maximising depending on the problem formulation, a linear function over a convex polyhedron. The feasible region is a convex polyhedron because both the objective function and the constraints are linear. Moreover, the optimal solution is always found at the boundary point of the feasible region. This optimisation problem can be expressed in a standard form

as follows:

$$\text{minimise } \mathbf{cx} \tag{5.13}$$

$$\text{subject to } \mathbf{Ax} = \mathbf{b} \tag{5.14}$$

$$\mathbf{x} \geq 0 \tag{5.15}$$

where $\mathbf{x}$ is the vector of variables to be solved for, A is a matrix of known coefficients, and $\mathbf{c}$ and $\mathbf{b}$ are vectors of known coefficients. Equation 5.13 is the objective function and the Equation 5.14 and Equation 5.15 are the constraints. In certain cases, where A has more columns than rows, the constraints will be under-determined, and this provides significant latitude in the choice of $\mathbf{x}$ with which to minimise the objective function. Clearly, all these entities must be in consistent dimensions for the program to function.

Two families of solution techniques, the simplex and the interior point, are frequently employed today. The simplex method is very efficient and functions well for most practical problems. It solves LP problems by constructing an admissible solution at a vertex of the polyhedron, and then progressively visits the vertices that possess improving values of the objective function, via the edges of the polyhedron. Nonetheless, it has a poor worst-case behaviour for certain problems which require exponential number of steps with reference to the problem size to obtain the solution. The interior point method, on the other hand, can move through the interior of the feasible region rendering it impervious to the worst-case behaviour but not as efficient.

LP solvers are widely used in industry and can be considered as a well established field. For this reason, an abundance of high-quality software libraries, both free and commercial versions, are available. In view of its benefits, this method is adopted into the proposed algorithm.

Mixed Integer Linear Programming

It is worth mentioning that there are unique cases where the problems encountered require some or all of the unknown variables to be integers. These problems are known as integer programming (IP) and mixed integer programming (MILP). In contrast to LP, these form of problems are much more difficult to solve. Of the two, MILP

is particularly intriguing and is well suited to solve problems cast in hybrid form, because the non-convexity and logic of the problem can be explicitly encoded using integers. It has been demonstrated recently by Richards *et al.* (2003) that MILP has significant application potential in real-time control and motion planning problems. In addition, Schouwenaars *et al.* (2005) implemented MILP in a guidance system for an unmanned aerial vehicle. Both of the studies attempt to fuse the receding horizon technique with MILP to solve the motion planning and guidance problems. Nonetheless, this promising technique will be reserved for future research.

5.3 System Quantisation

In this section, the AUV model and techniques for synthesising the motion primitives are presented. This is done, in order to convert the continuous system model into the MA representation. The AUV model employed is the *AUTOSUB* AUV. Figure 5.4 shows the MA representation of the *AUTOSUB* dynamics. Both 2 m/s and 5 m/s of cruising speeds are illustrated. However, the following simulations were limited to only one speed regime, selected to be 2 m/s to avoid the state explosion effect, to facilitate analysis, and partly due to the research time constraint imposed. The selection of states and manoeuvres are arbitrary and quite system dependent. Essentially, one needs to extract the predominant dynamics of the system whilst maintaining a sufficiently low numbers of motion primitives in order not to overload the computational requirements.

Once the MA representation of the vehicle under consideration has been dictated, the subsequent task will be to compile a repertoire of motion primitives that can exploit the vehicle operational envelope. A motion primitive can be acquired merely by applying an arbitrary piecewise-continuous control law to the vehicle in question, starting from arbitrary initial conditions for a finite time interval and storing the ensuing state and control trajectory. This can be attained by integrating the system state equation or running an experiment on a physical system. The extraction of motion primitives, especially the trim trajectories, is alleviated if the vehicle is velocity augmented. The next section explains this process.

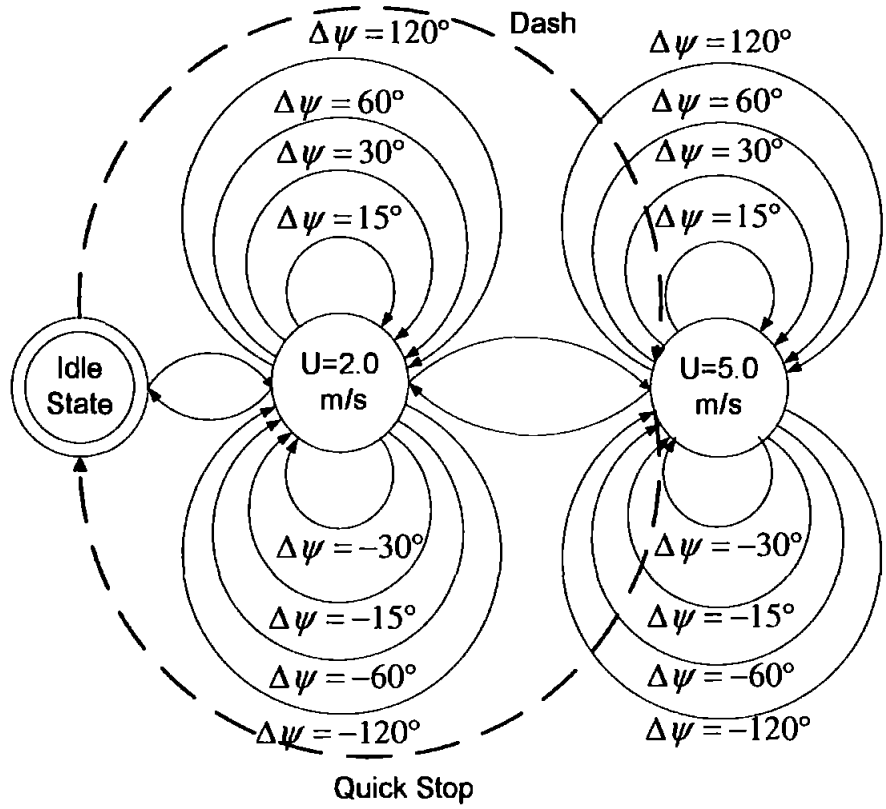


Figure 5.4: *AUTOSUB* dynamics in the MA representation

5.4 Velocity Augmentation

For the purpose of generating trim trajectories, a velocity augmentation loop must first be designed onto the AUV. To put differently, the system requires a set of inner loop controllers to be designed and incorporated such that the velocities, both linear and angular, can be directly controlled. This technique of control is not entirely new, and has been ubiquitously employed by the aerospace community for solving guidance problems (Corban *et al.* 2003). A diagrammatic representation pertaining to the overall system control structure is depicted in Fig 5.5. Notice that the velocity augmentation is connected to the AUV in the innermost loop while the outer loop is relegated to the tracking controller, which will be expounded in Chapter 6. The centre loop is occupied by the manoeuvre generator. The broken lines indicate that its usage is temporary and is removed during normal operating mode.

One positive implication that ensued from a stable, velocity augmented system, is the ease for manual control. Understandably, it is more intuitive for the system to be

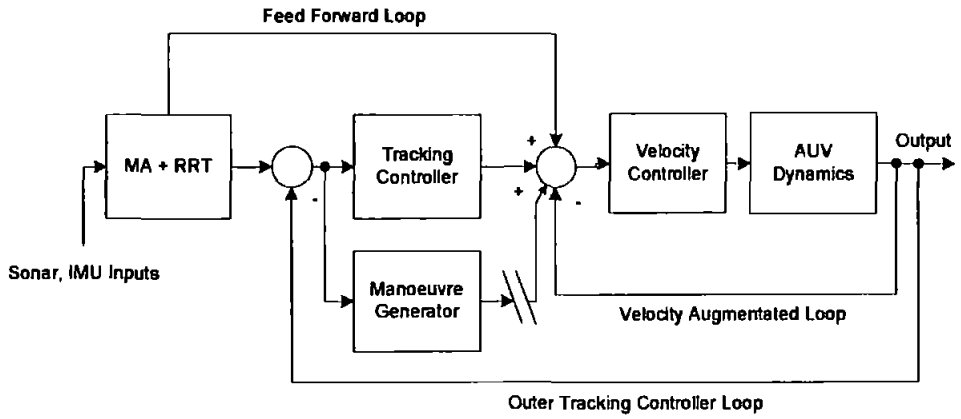


Figure 5.5: Overall Controller Structure

controlled by human operator since one can easily relate to the concept of speed. This permits the recording of complicated and aggressive manoeuvres via direct human inputs, obviating the need for time consuming task of designing a control law through controller synthesis.

In the subsequent simulations, the *AUTOSUB* model as set forth in Section 3.2.2 was discretised using the zero-order hold and the sampling frequency set to 10 Hz. The sampling rate was selected such that it is sufficiently fast to capture the dynamics of the AUV but also remain low enough not to overload the processing requirements. Two controllers were designed, one for the surge and another for the yaw rate. A proportional-integral (PI) controller was selected for this case. The integrator eliminates offset caused by the non-zero mean external disturbances acting on the vehicle.

To avoid confusion with the tracking time constant notation frequently employed by the industry, the following standard expression for the controller parameters is adopted.

$$u(k) = K_p e(k) + K_i \sum_{\alpha=(k-1)}^{k} e(\alpha)\Delta T \tag{5.16}$$

Where ΔT is the sampling time, k is the index, e is the error: the difference of the reference and measurement values, K_p is the proportional parameter, K_i is the integral parameter and u is the controller output.

The introduction of an integrator, however, incited a negative effect known as the integrator windup. This phenomenon is induced when the actuator output saturates.

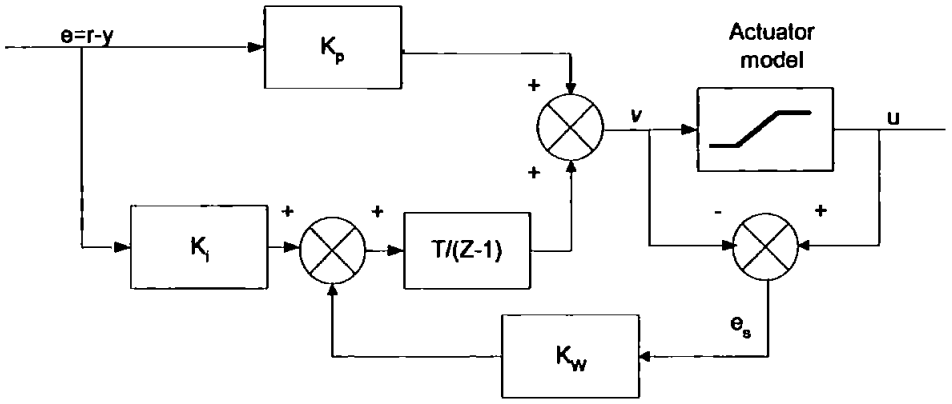


Figure 5.6: The PI controller with back-tracking anti-windup unit

At that exact moment, the error is integrated continuously, resulting in a very large integral term. Once it happens, a long period of large opposite sign error is needed to neutralise the effect and reverting the system back to normal. Thus, large transients are to be expected for a system equipped with integrator action and when the actuator saturates.

The windup phenomenon can also be explained by the nonlinearities that existed in all actuators. When the actuator saturates, reaches its limits, the feedback loop is broken and no longer applies. The system becomes open loop because the actuator will remain at its limit independent of the process output. The integral action, if presence, will integrate the error unceasingly.

Fortunately, a few techniques exist to address the integrator windup issue. This dissertation focuses on the back-tracking or back-calculation anti-windup scheme. A block diagram of the PI controller and back-tracking anti-windup unit is depicted in Figure 5.6. The primary idea is to recompute the integral term so that its new value gives an output at the saturation limit when the output saturates. It is advantageous not to reset the integrator instantaneously but dynamically with a coefficient K_w. Instant reset, can eliminate beneficial integral action and create a longer settling time. One can obtain a starting point for K_w using the following formula, $K_w = 2(K_i/K_p)$ (Astrom and Hagglund 1995). The actual formula was given in the tracking time constant format but has been recast to this form for consistency.

5.4.1 Surge controller design

The surge controller gains, K_p and K_i were empirically determined. The AUV propulsion dynamics is nonlinear due to the added mass and the thrust exerted by the propeller, which are proportional to the square of its rotation speed. The gains of $K_p = 5$ and $K_i = 10$ have been found to function favourably in a regime of 3 m/s to 1 m/s. It is imperative that the effect of integral wind up is studied as it frequently occurs when the thruster saturates, as subjected to constant velocity sea currents. Figure 5.7 shows a test current of magnitude 1 m/s, at direction 0° with reference to Earth fixed frame, being injected to the AUV to assist in selecting the appropriate anti-windup coefficient, K_w.

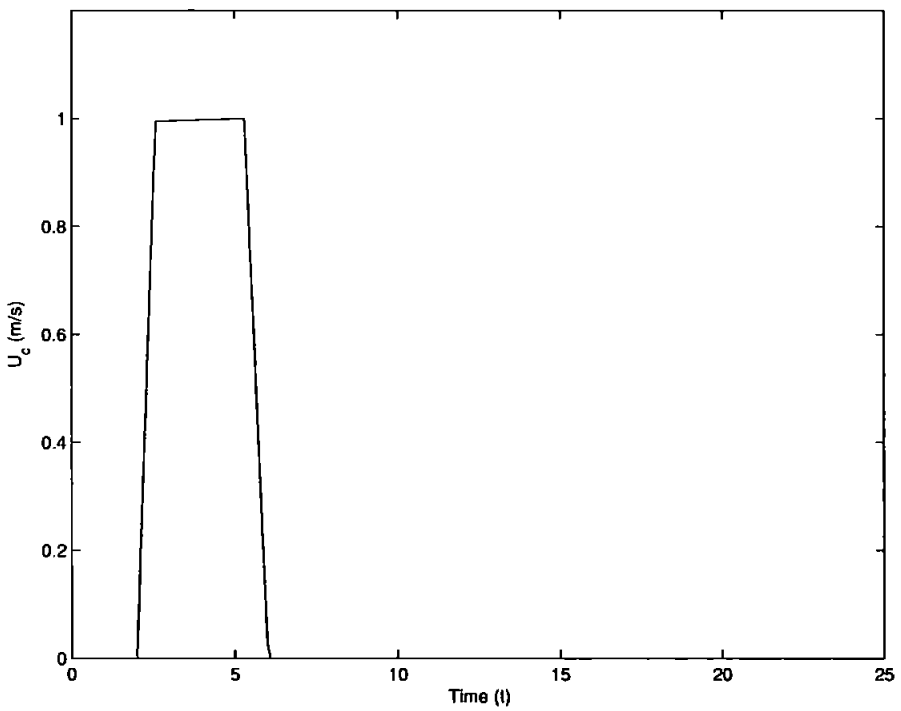


Figure 5.7: Current velocity and time plot

Figure 5.8 reveals the control laws with the corresponding K_w elicit by the PI controller. A constant value of 140 rpm is required to maintain the AUV at a cruising speed of 2 m/s. The initial downward pointing triangular pattern in Fig 5.8 is induced by slew-rate limits of the motor. Actuator rate saturation is less detrimental compared to the output saturation. Nonetheless, it can still initiate the windup phenomenon.

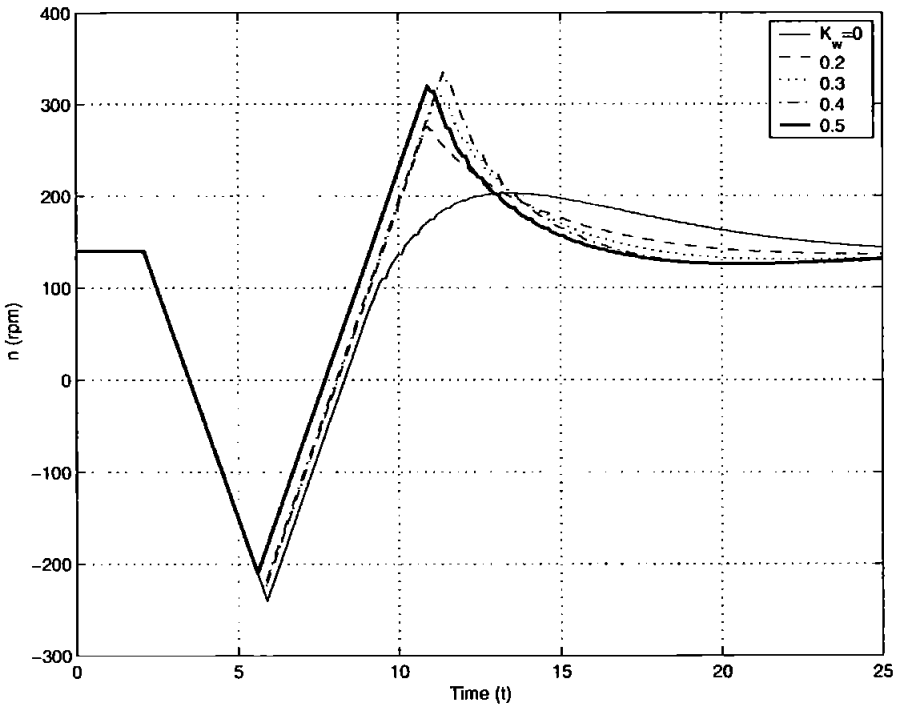


Figure 5.8: Controller output response with different K_w

The effects of different values of anti-windup coefficient and their ensuant responses are vividly depicted in Fig 5.9. One can deduce from the figure that the higher the value of anti-windup coefficient, the shorter the rise time but the longer the settling time. Understandingly, a high value of the anti-windup coefficient tends to mitigate the effect of error integration, resulting in a shorter rise time. On the other hand, a low value shows a damped behaviour because of the winding and unwinding effects of the integral term. A value of 0.3 was chosen as it provides a compromise solution between the two behaviours.

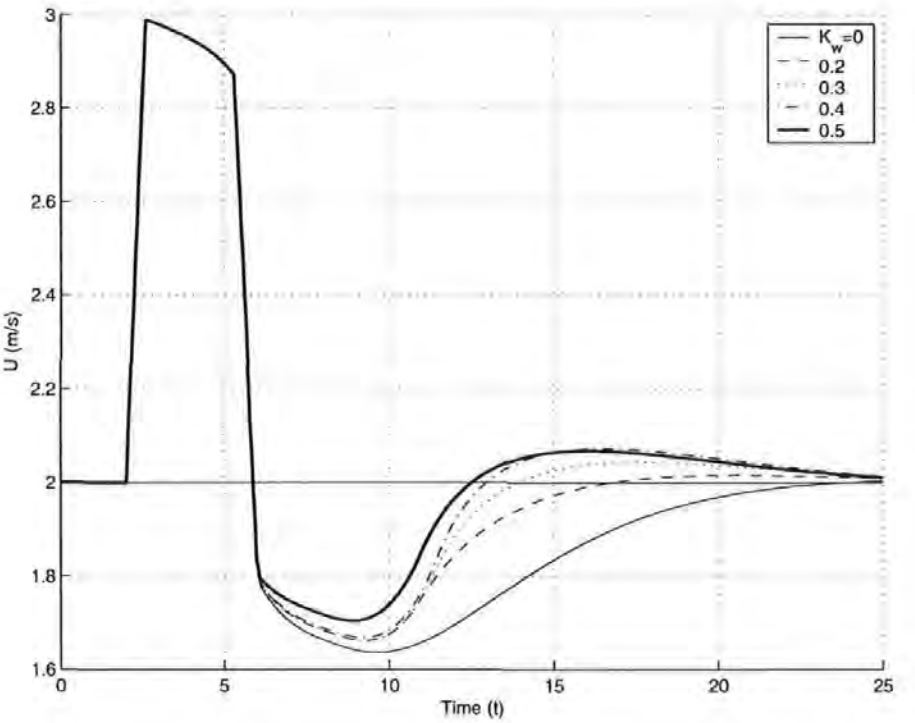


Figure 5.9: Surge Response with different K_w

5.4.2 Yaw rate controller design

Prior to designing the yaw rate controller, the yaw rate dynamics must be extracted from the nonlinear, coupled *AUTOSUB* model. The dynamics in S transform is given by

$$\frac{0.0189(s+0.282)}{(s+0.156)(s+0.667)} \tag{5.17}$$

The SISO yaw dynamics with a zero-order hold at 10 Hz, can be expressed in Z transform form as provided below.

$$\frac{0.0183(z-0.972)}{(z-0.985)(z-0.936)} \tag{5.18}$$

An open-loop Bode plot 5.10 reveals the similarity between the discretised model at 0.1 s sampling time and the continuous model at low frequency. The phase response for the 0, 1 s sampling time, discretised version drop off quickly after 1 rad/s, but that is beyond the predominant dynamics of the system. Here one is interested between 0 rad/s to 0.25 rad/s, the maximum yaw rate of the vehicle. Again, referring to

the figure, the system response changes drastically if the sampling time is changed to 0.2 s.

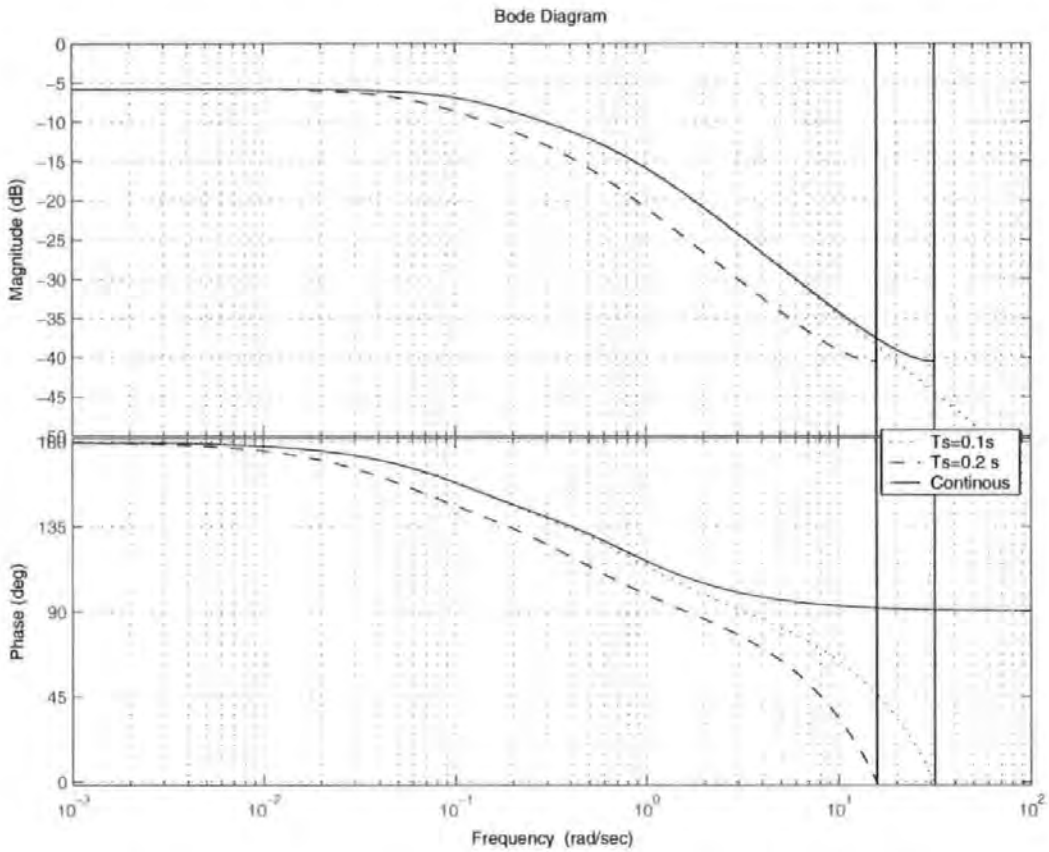


Figure 5.10: Open loop Bode plot of yaw rate dynamics

Figure 5.11 was used to assist in the tuning operation of the PI controller. According to the rule of thumb, the recommended gain margin should be within 6 dB to 10 dB and a phase margin in the range of 45° and to 60° are considered to have adequate stability and robustness properties in practice. The notion of gain margin and phase margin are inexorably linked to relative stability. The higher the gain margin, the more potential the system gain can be increased before instability occurs. Whereas, the higher the phase margin, the more impervious the system to time delay effect, which can cause system instability. Nonetheless, an excessively high gain margin and phase margin frequently indicates a sluggish or an underperforming system. A satisfactory value of K_p, K_i, K_w were found to be 5, 2, and 0.8 in the given order. The gain margin and phase margin in accordance to the Bode plot are 26.7 db and 92.9°, respectively. These higher margins are desirable in order to accommodate for

the effect of system nonlinearities.

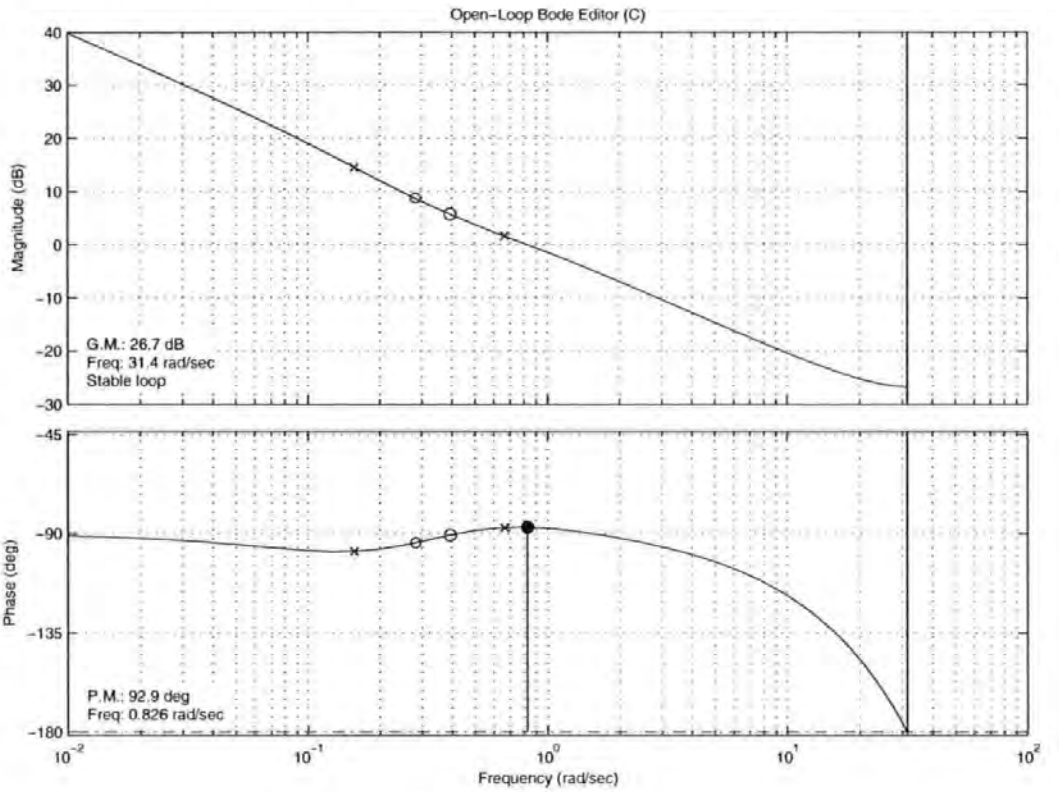


Figure 5.11: Open loop Bode plot of the yaw controlled system

A Nyquist plot is shown in Fig 5.12, its locus does not cross over the minus one value in the real axis, establishing the fact that the system has attained closed-loop stability. This method of analysing stability is based on Nyquist Stability Criterion. Nyquist plot conveys similar information as the Bode plot but in a different manner. It also provides a more powerful method for analysing and quantifying the robustness properties of a system compared to the Bode plot. Such as vector margin, a parameter to quantify robustness to combined gain scaling and phase shift perturbation, can only be revealed using a Nyquist plot (Ozbay 1999).

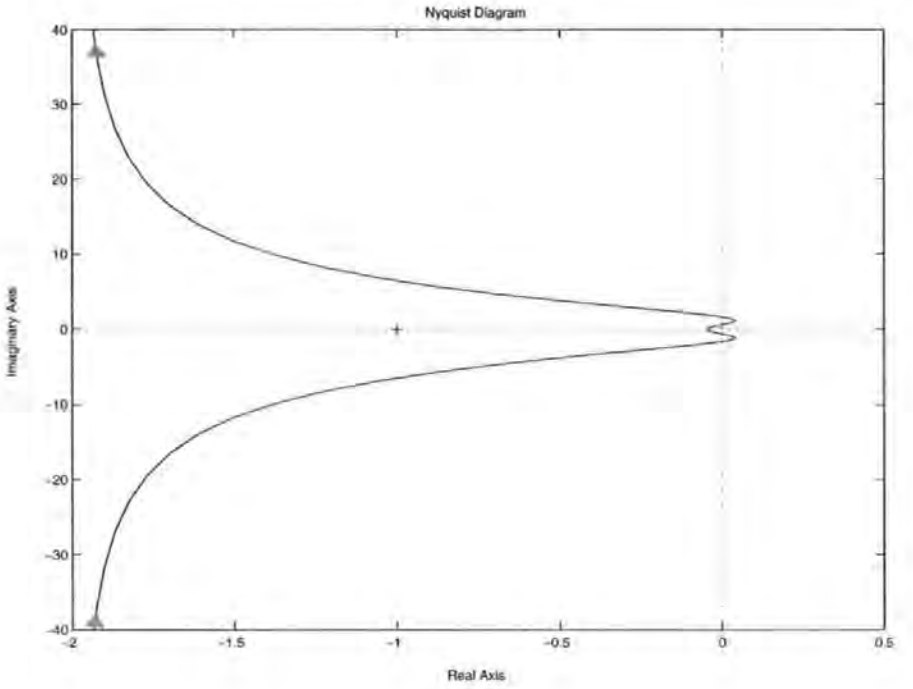


Figure 5.12: Nyquist plot of the system

The unit step response and its control effort of the yaw rate controller using the linear equation is shown in Fig 5.13. The response will be marginally different with the one obtained from the nonlinear model. To obtain the maximum yaw rate physically achievable by the *AUTOSUB* cruising at 2 m/s, a step input of 17.0°/s was used. This value is arbitrary, as long as it is higher than then the maximum yaw rate attainable. Figure 5.14 shows the heading rate response. One can conclude that the maximum yaw rate is 13.8°/s. Any other input values exceeding this value will not be satisfied by the system. The yaw rate ceiling can be attributed to the limited rudder moment and induced drag when executing a turn. Clearly, it is not a good idea to push it over the saturation limit.

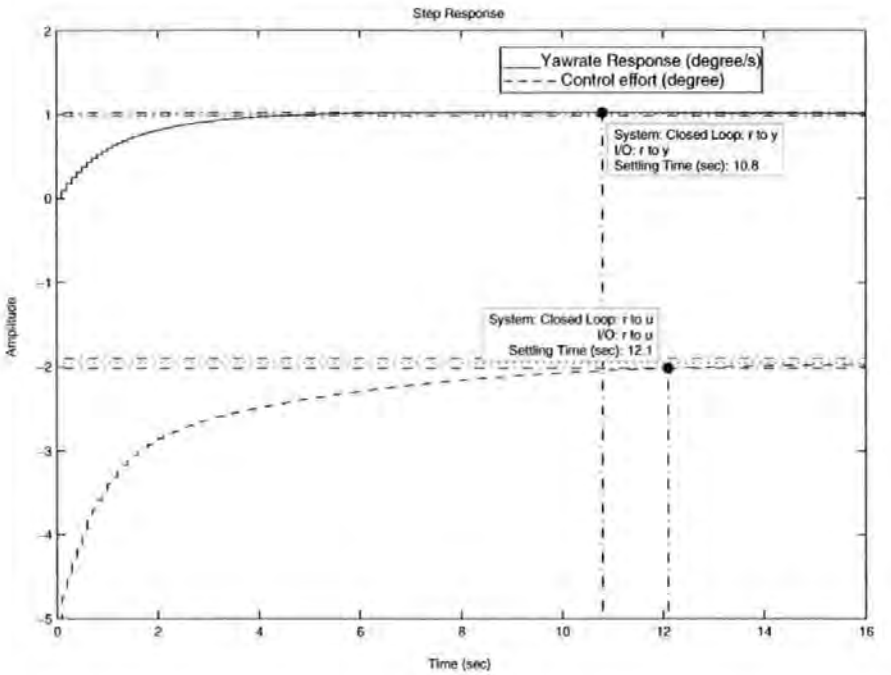


Figure 5.13: Step response of with yaw rate controller

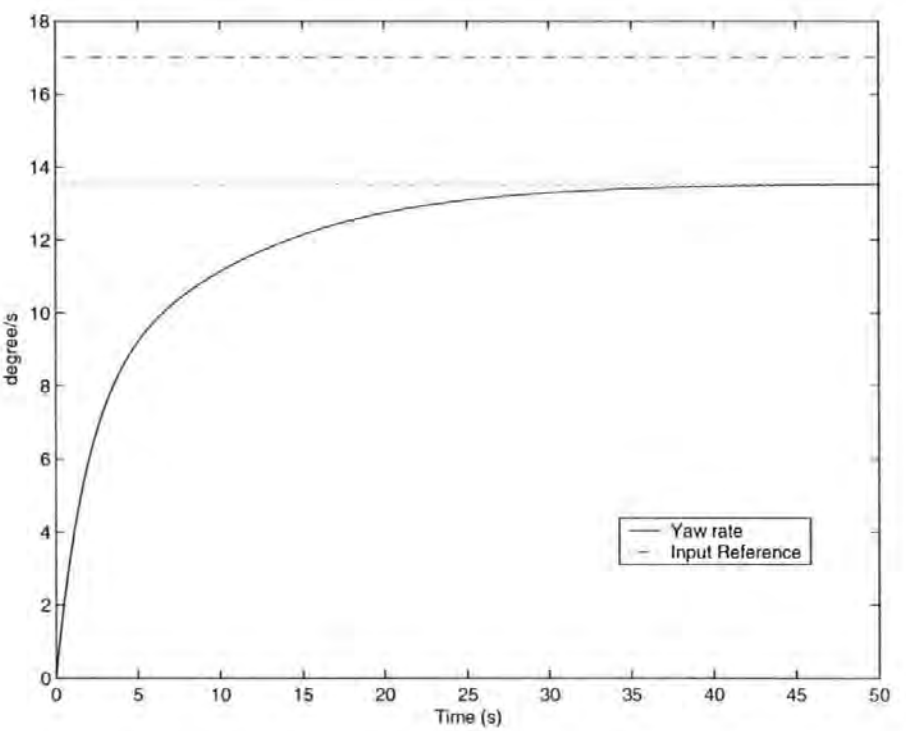


Figure 5.14: Maximum turning rate of the yaw controlled AUV

5.5 Motion primitives generation

The motion primitives or manoeuvres should encompass the crucial performance envelope of the AUV. As aforementioned, their generation can be attained via an human operator or controller inputs. The latter method was selected for the following study.

The AUV's rudder elicits an integrator term in the yaw dynamics of the AUV. The behaviour is discernable when the AUV is injected with a step input, and as expected the AUV will execute a turning manoeuvre continuously, an evidence of an in-built integrator. For this reason, a proportional-derivative (PD) autopilot was designed so that one can extract the manoeuvres by inputting step inputs. It was then discovered that the derivative term, even with small coefficient, introduces excessive damping behaviour to the heading response. Proportional gain was increased to counter the sluggishness induced by the incorporation of the derivative term. Hitherto, this increases the rudder deflection rate until it violates the slew rate limits. This phenomenon is undesirable, furthermore, small amount of control authority must be reserved for the proper functioning of the tracking controller. Later, it was felt that adequate performance can be attained by using only a proportional controller. Clearly, a more advanced controller such as the LQR can be used as a substitute to extract a better heading response from the AUV.

The rudder rate limit was prescribed to $\pm 6.8°/s$ compared to the actual physical one of $\pm 12.8°/s$ prior to generating the manoeuvres. Likewise, the rudder saturation was defined to be $\pm 20°$ instead of $\pm 25.2°$ as in the original model. This process will impose virtual bounds to the generated control law, in the meantime relegating the remnant control authority for tracking the prescribed trajectory. This controller is labelled as the manoeuvre generator in Fig 5.5. This controller is used solely for manoeuvres generations and its function is inhibited during a normal AUV operating mode.

Referring to Fig 5.4 again, it shows clearly the *AUTOSUB*'s MA that constitutes a manoeuvre set of 15°, 30°, 60°, 120° and the opposite direction ones. A tabulation of the corresponding proportional gains and manoeuvres are given in Table 5.1. The suffix of p indicates the heading degree.

p_{index}	K_p
p_{15}	0.25
p_{30}	0.30
p_{60}	0.30
p_{120}	0.30

Table 5.1: Heading controller proportional gain, $q = 2\ m/s$

Table 5.1 lists all the right turn manoeuvres with their associated execution time duration and displacements. The x, y and ψ presented are all rounded to the nearest integer, although internally, the computation is performed using the accurate floating point variables to avoid truncation error. In practical implementation, the x and y displacement variables accuracy should be further relaxed as it is impossible to obtain within $\pm 3m$ accuracy without the help of GPS, LBL, or SBL.

Referring to Table 5.2 again, notice that for some manoeuvres, the headings have been clipped before reaching the desired set-points, this is aimed to reduce the manoeuvre time. Exactness is not compulsory for the proper functioning of the system as the primary objective is to capture only the primary behaviours of the vehicle. A pictorial representation pertaining to the effect of executing the associated manoeuvres is given in Fig 5.15. Similarly, Fig 5.16 and Fig 5.17 shows the heading response and the rudder reflection of each manoeuvres, respectively.

It is true that the heading responses are asymptotic, and an infinite amount of time is required to reach the set-point. But in practice, the responses are clipped when they reached a predefined range, like 2% or 5% within the set-point. Frequently, the variance of the measurement data are used to delineate the clipping range. This clipping process will create a discontinuity at the interconnection of the motion primitives. Referring to Fig 5.17, notice that the rudder inputs do not reach zero at the end as it should in theory. This issue shall be addressed by the tracking controller.

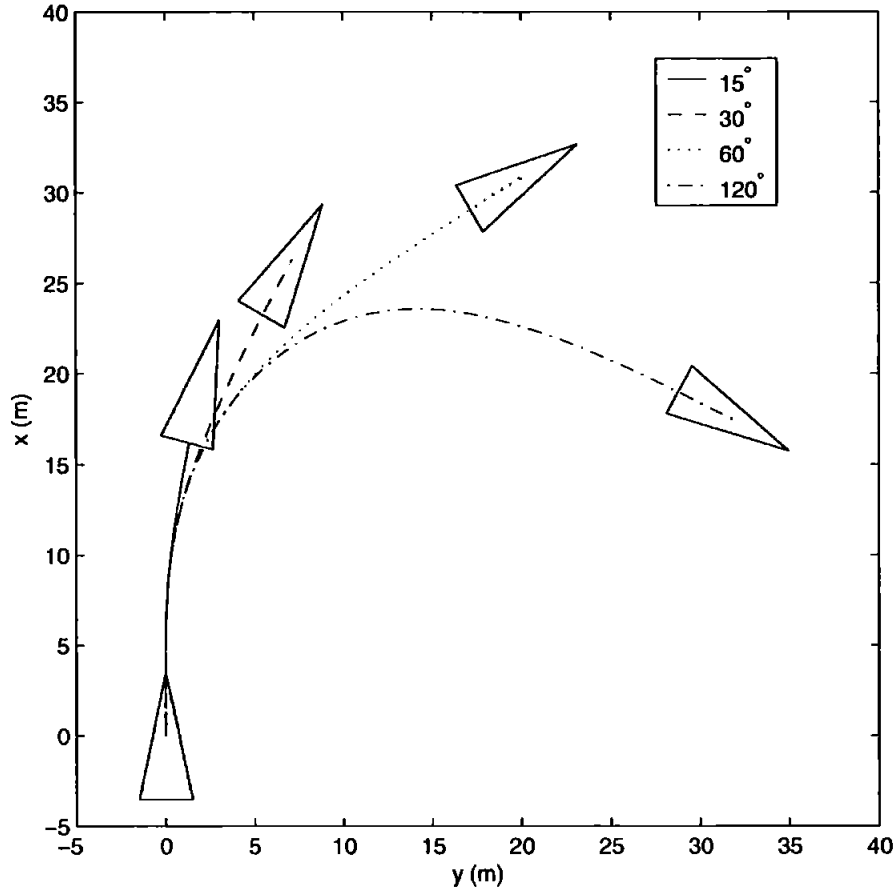


Figure 5.15: x-y displacement plot of the associated manoeuvres

p_{index}	$\Delta T(s)$	$\Delta x(m)$	$\Delta y(m)$	$\Delta\psi(\circ)$
p_{15}	9.9	196	2	15
p_{30}	13.9	26	7	30
p_{60}	19.9	31	20	59
p_{120}	24.9	17	32	119

Table 5.2: Manoeuvre library, $q = 2\ m/s$

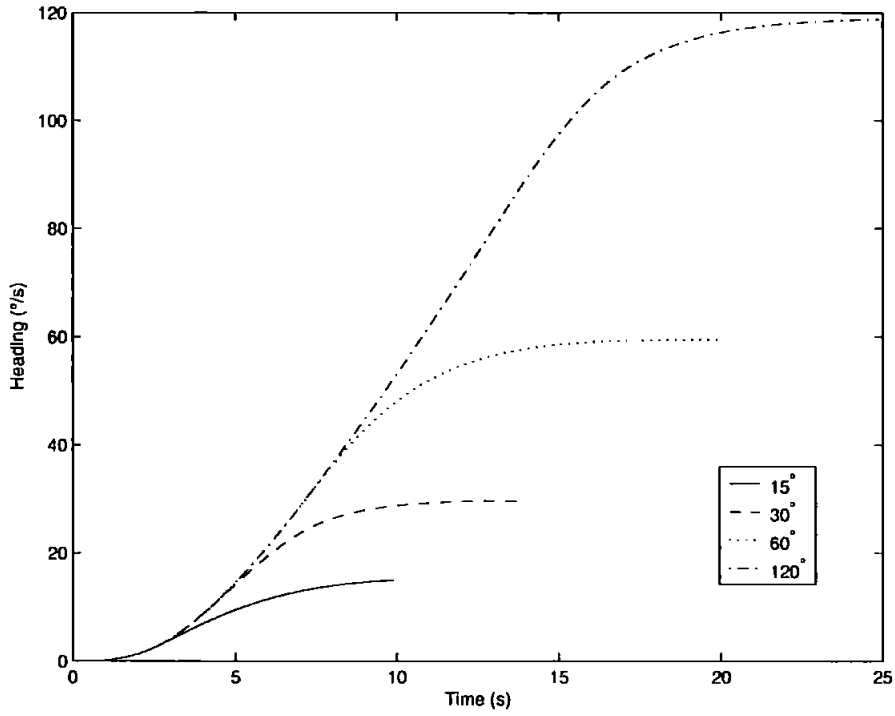


Figure 5.16: Heading response plot for the associated manoeuvres

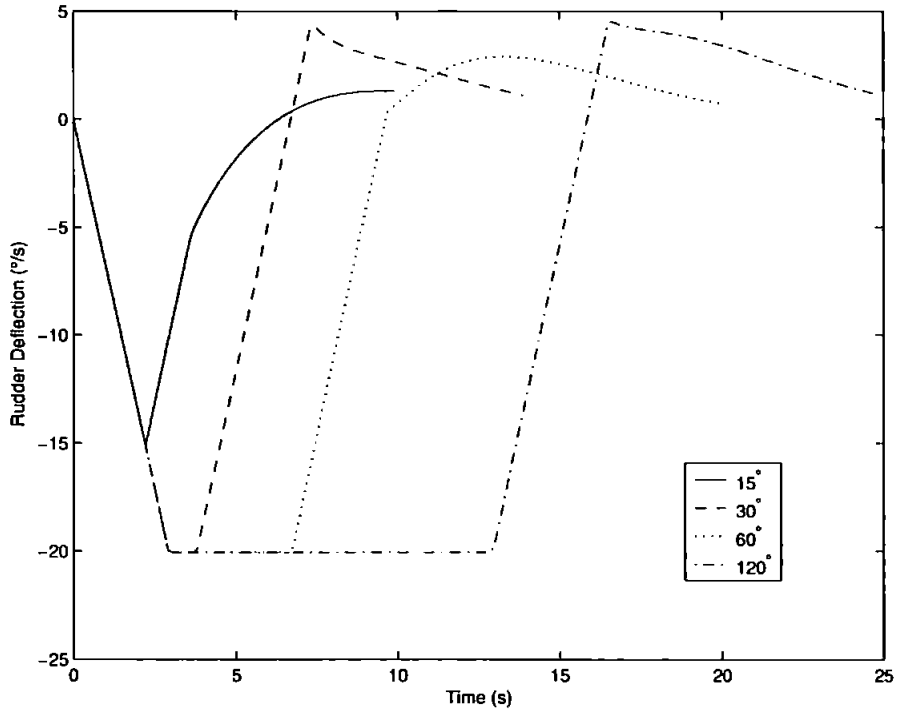
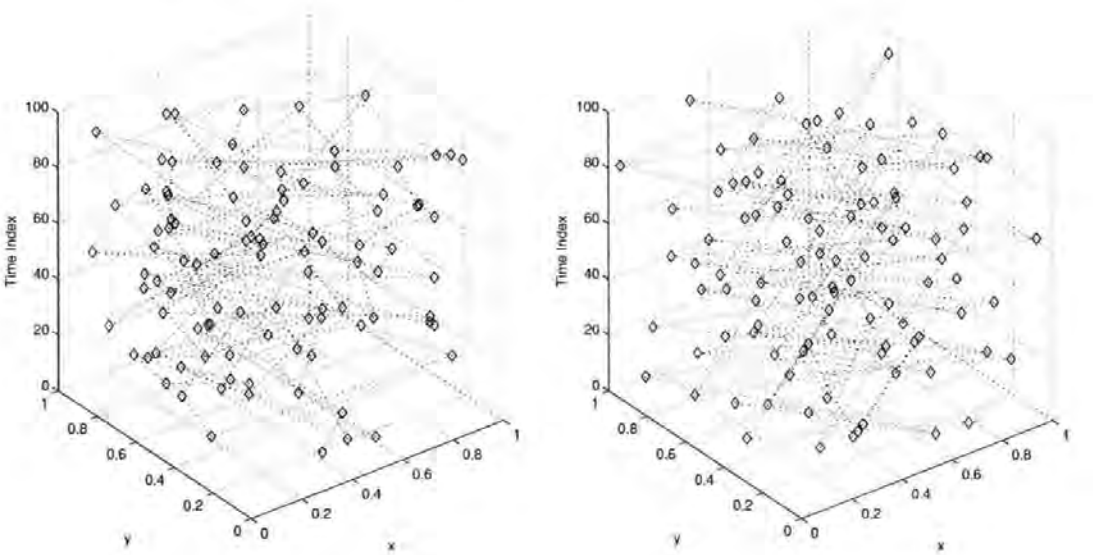


Figure 5.17: Rudder deflection history for the associated manoeuvres

5.5.1 Quasi-random sequence

Herein, the quasi-random (sub-random) generator based on the Halton sequence (Halton 1960) is utilised instead of the pseudo-random generator. Theoretically, the former generator possess certain desirable properties such as low discrepancy or improved uniformity over the sampling space. Different prime numbers are used for the multi-dimensional sampling case. Strictly speaking, Halton sequence is deterministic hence one way of generating a different motion plan for the same environment is to change the seeds for each individual simulation.

The notion of uniformity is made precise with the definition of discrepancy. The discrepancy of a sequence is low if the number of points falling in to a set B is close to the number one would expect from the measure of B. It is imperative to understand this intriguing property and its effect to the MA+RRT algorithm. Figure 5.18(a) and 5.18(b) show two types of random points contained in $[0, 1]$ sampling space being plotted against time. The dotted line which connects the two dots in subsequent time order, is used to highlight the discrepancy attribute. Notice the quasi-random version (Fig 5.18(a)) has a more uniform lines, eliciting the zigzag pattern. Conversely, the pseudo-random version (Fig 5.18(b)) indicates a chaotic pattern instead.



(a) (b)

Figure 5.18: Time plot of 100 sample points (a)Pseudo-random (b)Quasi-random

A plot of the distance between the subsequent random points revealed another insight into this behaviour. Figure 5.19 divulged that the quasi-random points have a approximated minimum distance of 0.4, the mean distance being 0.7. Whereas the pseudo-random can have points that are very near together. In fact, the distance spans from near zero to one, implying that it has high variance. Note that these distances were extracted from the random points in subsequent time order, more significant clustering behaviour will be detected for the pseudo-random points if one neglects the time dimension as shown in Fig 4.3(a).

Initial experiments with different scenarios conducted on the MA+RRT algorithm demonstrate that an improvement of 11% success rate can be attained if one employs quasi-random instead of pseudo-random sequence. It is believed that the effect of different random number sequence is more pronounced in this study, partly because of the algorithm's lower consumption of points. Which means, that higher quality points have substantial effect to the algorithm outputs. Nonetheless, these enticing results are premature to say the least, and more exhaustive studies are warranted. The quasi-random sequence is adopted for the remaining of the simulation studies thereof.

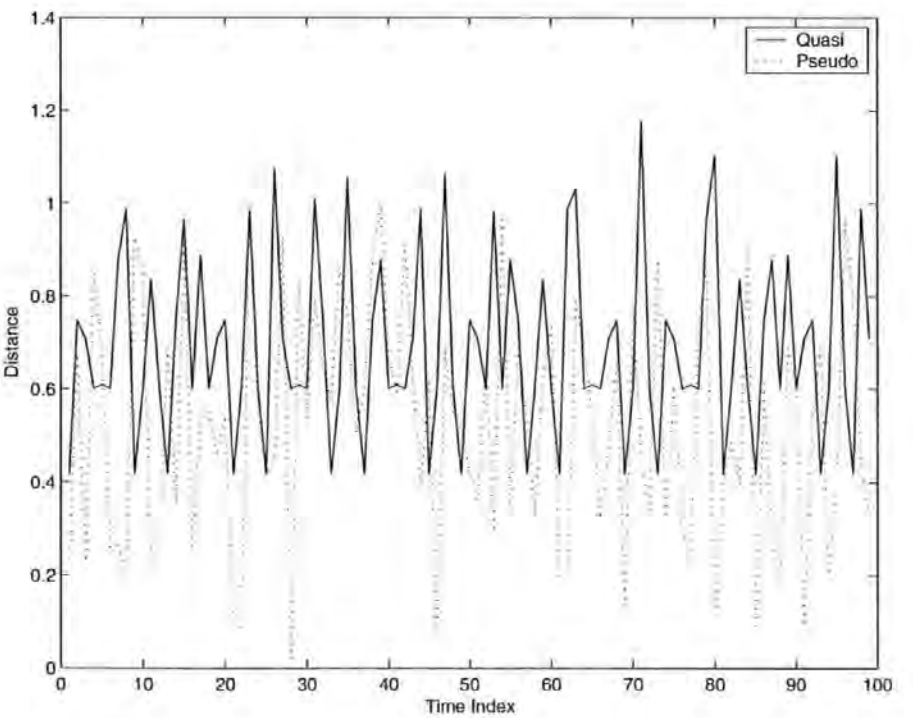


Figure 5.19: Distance plot of subsequent random points

5.5.2 Motion planning algorithm

Frazzoli (2002*b*) advocated enhancing the RRT algorithm by fusing it with the MA to solve motion planning problems with obstacles. The algorithm assumes that one has an embedded planner, that can plan an optimal trajectory in an obstacle free environment between two arbitrary states.

However, the approach in employed in this thesis is that of multiple nested nodes. Since every state in a trim trajectory can be considered as a starting point of a manoeuvre. This algorithm generates child nodes at every connection point between a trim trajectory and manoeuvre. This method improves the RRT branching capability by increasing the probability of finding a solution. The number of child nodes to generate are arbitrary, but in this algorithm, one child node is generated at the mid-point of each trim trajectory. Too many child nodes will saturate and slow down the computation. Additionally, the case for a time varying final state is also addressed with this newly developed algorithm. The forthcoming algorithm is an aggregation of the MA method and the RRT algorithm as detailed in Chapter 4. A brief explanation of this enhanced algorithm with reference to Fig 5.20 is outlined below:

1. Check to see if a direct connection to the goal from the initial states based on the minimum time criterion is possible. If this is attained then the algorithm terminates.

2. Failing that, generate a subgoal, $R1$ using the quasi-random generator and attempt to connect to it using the embedded planner, again based on the minimum time objective function.

3. If there is no collision, then generate an edge with new vertices at all interconnecting points of trim trajectories and manoeuvres. Explicit connection to the goal is attempted from all the new vertices (Greedy algorithm). If this is successful, the algorithm terminates.

4. If failed, generate another random subgoal, $R2$. Sort the shortest time trajectories from all vertices to $R2$ in an ascending order and attempt to connect to $R2$. Apply this to only the first few near-optimal trajectories to avoid vertices saturation. In this algorithm, only the first 3 near-optimal trajectories are stored and tested.

5. The whole process is repeated from 1 to 4 until a feasible trajectory to the final state is found, the maximum vertices size or the time limit is reached. Figure 5.20 shows that vertex, $nc1$ has connected successfully to the final state.

6. For the time-varying final state case, the 'false-position' optimisation procedure is applied when trying to connect to the changing final state.

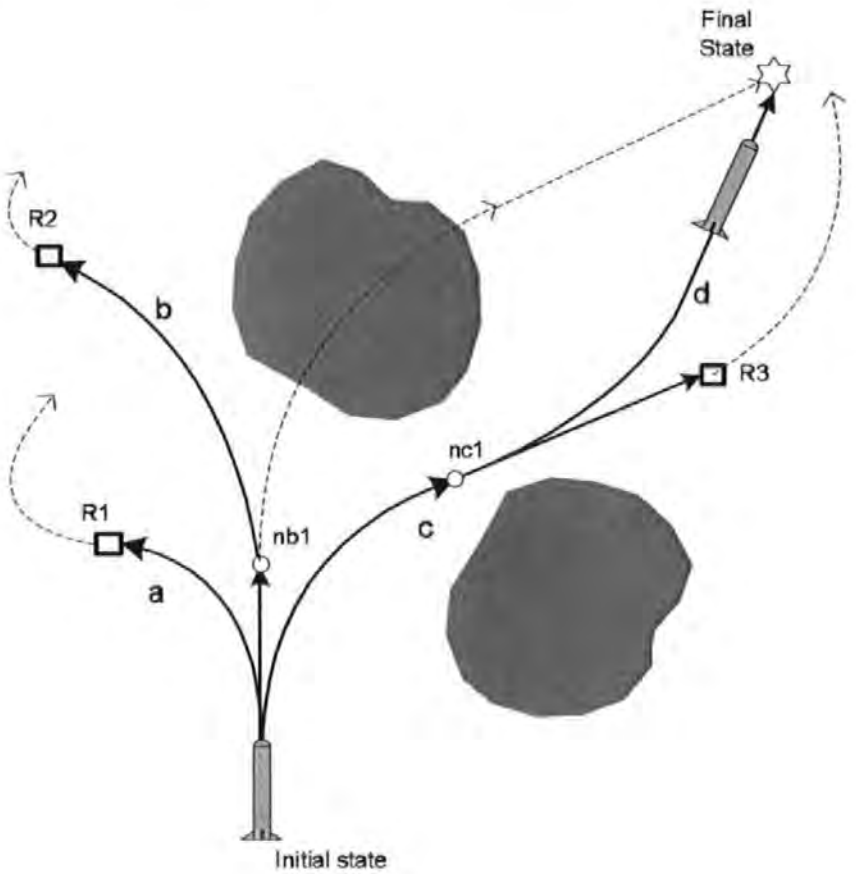


Figure 5.20: MA+RRT Algorithm operation

In practice, it is crucial to define a state-space tube or manifold such that the AUV will always be travelling in or near the centre of the tube at all times. Once the AUV state variables exceed the tube, the generation of a new trajectory is necessary. As a safety measure, a similar tube must also be specified to detect the environmental obstacles that intersect the AUV path. The specification of the manifold is related to the tracking capability of the controller, the predictability of the environment and the magnitude of external disturbances.

5.5.3 Error mitigation

A few researchers have expressed their concern regarding the prescribed error generated by RRT algorithm. Due to the discretised nature of the inputs in the original RRT algorithm, when the input history is applied, there will exist some errors in the final states. Hence, Kim and Ostrowski (2003) attempt to circumvent the problem by introducing a subconnection process. Similarly, Cervern *et al.* (2004) introduce an error mitigation scheme to reduce the error caused by the concatenation effects. Both of these techniques rely on 'integrating' the dynamic model with the input history to obtain the final state. However, in practice, unless RRT is applied in a disturbance free environment, the external disturbance effect should be of more a concern. One disadvantage, in this approach is the requirement of an accurate dynamic model of the system. This might not be true in practice, due to model complexity, or nonexistence of a mathematical model. This error can be considered as a form of disturbance, hence a tracking controller (outer loop) is required to assist in tracking the nominal trajectory. Frazzoli (2001) designed an invariant tracker which capable of preserving the open-loop symmetries, in the closed-loop mode. The design of a tracking controller is non-trivial, due to the multi-input-multi output (MIMO) and highly coupled dynamics of the AUV and as such it will be addressed in Chapter 6.

5.5.4 Time-varying final state

There is an increasing interest in the use of AUVs as force multipliers for a submarine in support of maritime expeditionary operations. Aggravating the problem is the limited battery technology of an AUV, which does not provide adequate servicing range. Likewise, the high bandwidth data transmission requirement for most surveillance tasks makes it compulsory for an AUV to upload its data intermittently. What is more, with the recent, advanced sensors equipped AUVs, it is not deemed economically variable to make them disposable. Thus, the concept for an AUV docking with a station/submarine for recharging, downloading data or even servicing purposes is an attractive proposition.

An AUV retrieval manoeuvre can be partitioned into two distinct phases, which are *interception* and *docking*. The retrieval manoeuvre problem had been addressed by

(Tan *et al.* 2003). In the paper, the interception phase employs the popular proportional navigational guidance (PNG) algorithm, whereas the docking phase utilises fuzzy logic, which is activated when the AUV reaches a prescribed circumference of acceptance of the target.

In the previous simulations, environmental obstacles were not included thus it would be interesting to substitute the previous PNG guidance system with the MA+RRT algorithm instead. The novel algorithm must however be extended to cater for the case of a time-varying terminal state. The problem of addressing a time-varying final condition using RRT was first pursued by Cervern *et al.* (2004). Their approach was based on embedding the time variable into the system state vector. Evidently, the immediate ensuant effect is the increase of state vector dimension.

A simpler solution proposed here is to adopt an iterative subroutine popularly known as the *false-position* method. Fundamentally, assuming that the target is moving at a constant velocity, the concept is to use a predict-correct process to converge within a tolerance of the final state. The procedure is simple, first the initial target position is employed and a trajectory search is executed. If that is successful, the time consumed obtained from the trajectory found is reinserted into the equation to anticipate the future position of target. The process is iterative, and terminates when predefined tolerance of the final state is achieved. Nonetheless, there is no guarantee of convergence, thus an upper bound to the iteration count is needed to terminate the loop as a contingency.

5.6 Simulation Results and Discussion

The sampling space was set to $300m \times 300m$ in dimension for the remaining simulations. Again, one assumes an ideal case where *a priori* information of the environment has been catered and there is no external disturbance imposed by the environment. The *AUTOSUB* AUV model was employed. Here, owing to the limited time span of this research, the case of incremental sensing where only partial environment information is acquired will not being accommodated.

All the conventions herein, follow that outlined in Chapter 3. The remanding simulations were hosted in MATLAB 6.5 /SIMULINK environment, on a 2.1 GHz Pentium IV machine, with 512 MB of RAM and running Windows XP. This dissertation employs the GNU Linear Programming Kit ver. 4.4 (GLPK)[1]. To facilitate communication with MATLAB, a MEX-interface known as GLPKMEX[2] provided by Girogetti (2004) was used.

The simulations were run with 200 maximum nodes and 300 iterations, terminating when either criterion is reached or if a solution is found. The AUV initial configuration states was set to [1 1 0.1] (angle in radian), and the goal state to [170 139.4 κ] where κ denotes a variable (unconstrained). The minimum time criterion was used throughout. Goal tolerance was defined to be $7m$ radius. ψ variable is omitted to increase the probability of finding a feasible trajectory. This is acceptable since the priority here is collision avoidance. However, if the final heading of the AUV is deemed important, then the heading state should be incorporated into the algorithm. It is recommended that inequality constraints be employed on the ψ variable, with the aim to introduce some 'slackness' to ease in finding a solution.

[1]Obtained from http://www.gnu.org /software/ glpk/ glpk.html
[2]Obtained from http://www.dii.unisi.it/ giorgetti/downloads.html

5.6.1 Simulation: static environment

A sample of feasible trajectory (bold) found by the enhanced RRT algorithm is depicted in Fig 5.21. The thinner lines are candidate trajectories. The solid triangle depicts the AUV, enlarged twice for reason of clarity. The squares denote random sampling points whereas the asterisks represent nodes. It should be kept in mind that, although, the time optimal criterion was adopted in the algorithm, a time optimal trajectory can only be found if there are no obstacles present. Clearly, the trajectory found, as in this case is not an optimal one but near optimal. Nevertheless, a few runs can always be conducted to select the 'best' trajectory. Figure 5.22 (a) and (b) show the associated displacements-time plots, in three dimensions and different perspective for better visualisation. The time needed to execute the trajectory being 208.9 s.

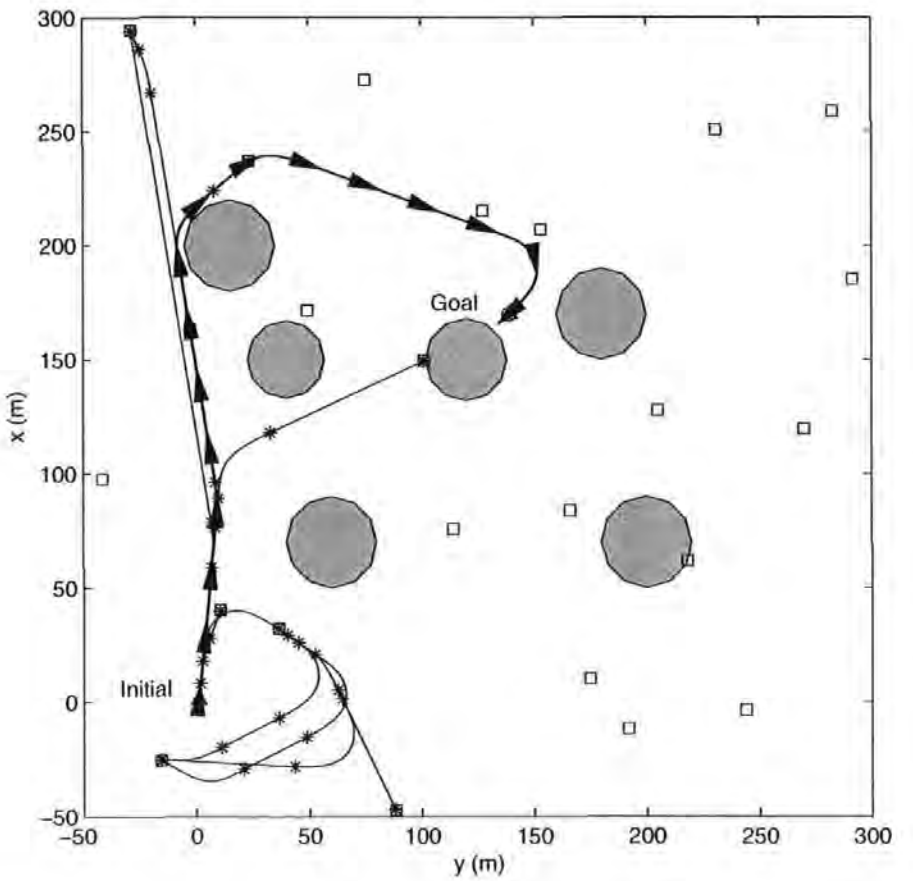


Figure 5.21: A feasible trajectory *(bold)* found in an environment with static obstacles

The AUV inputs history are depicted in Fig 5.23. In Fig 5.23 (a) one can observe that *rpm* increases when the AUV is performing a turn. The sudden drop in speed can be attributed to the increasing lateral drag thus explaining the gain in *rpm*. A notable increase in sway velocity is detected during the turning manoeuvre as caused by the side-slip effect which frequently encountered by an underactuated vehicle (Fig 5.23 (b)).

Unlike the previous RRT algorithm detailed in Chapter 4, there is no sign of chattering behaviour in the rudder history (Fig 5.23 (c)). Careful analysis of the rudder history indicates clipped values at around 20°. This is not to be confused with the actuator saturation but a virtual bound that is intentionally imposed with the aim to reserve sufficient control authority for trajectory tracking later. Sudden jumps or step discontinuities were detected in the propeller and rudder inputs, these small discrepancies can be attributed to the effect of clipping the manoeuvres, and are not critical in practice. The yaw rate is smooth and desirable for control (Fig 5.23 (d)).

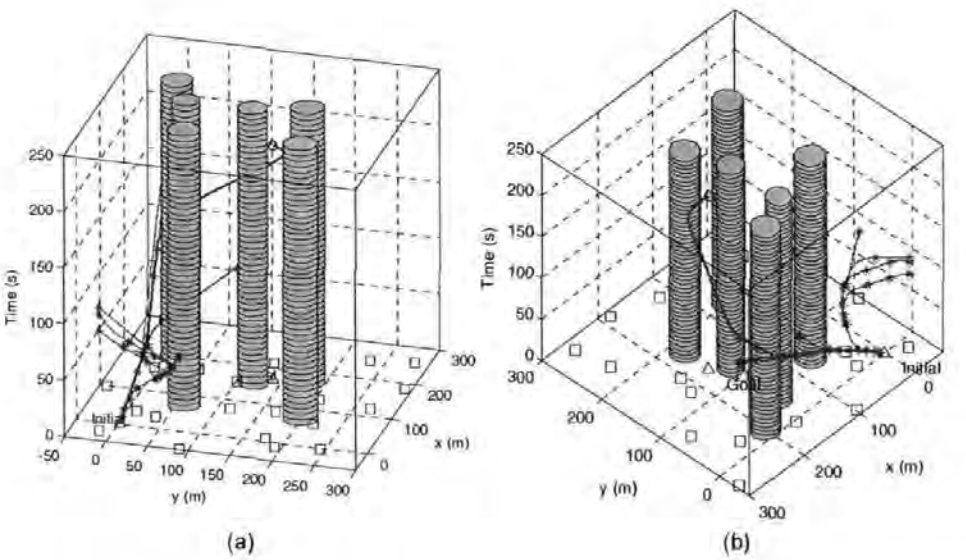


Figure 5.22: x-y-time plots for a feasible trajectory *(bold)* found in an environment with static obstacles

The above trajectory can be expressed in symbolic form using Equation 5.11 as follow:

$$g_f = g_0 \times q_1 \times p_{-15} \times q_2 \times p_{60} \times q_3 \times p_{60} \times q_4 \times p_{120} \tag{5.19}$$

Where g_0 and g_f is the initial and final configuration. q and p are trim trajectories and manoeuvres. The trim trajectories have the following coasting times, $\tau_{q1} = 38.1\ s$, $\tau_{q2} = 47.8\ s$, $\tau_{q3} = 5.1\ s$ and $\tau_{q4} = 43.0\ s$. All of the variables are encoded in a square 3×3 matrix as in Equation 5.4. The matrix multiplication must be operated from left to the right element wise. This form of compact expression is highly suitable for the application of cooperative robotics owing to its low bandwidth requirements. Since this is a randomise algorithm, different run will yield a different result, therefore four unique feasible trajectories are provided in Fig 5.24 for comparison. The four trajectories are shorter distance wise than the one discussed.

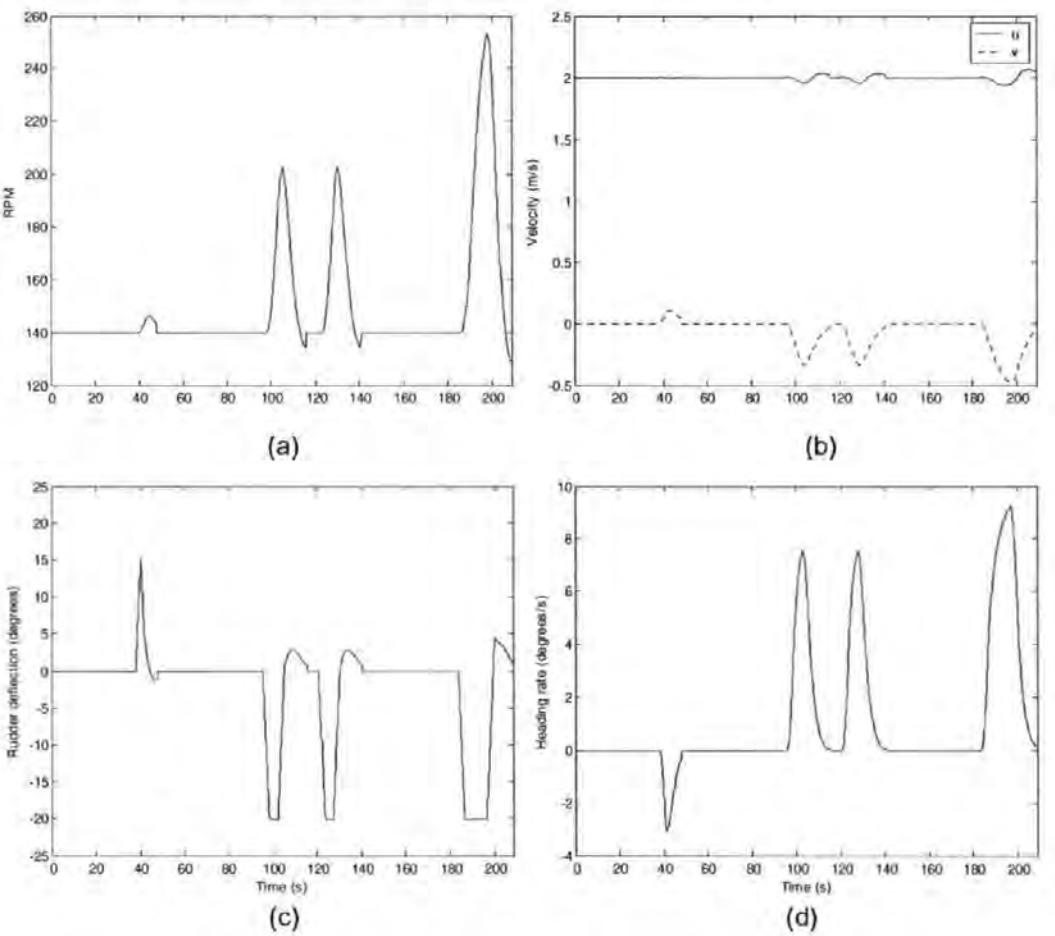


Figure 5.23: Inputs and states history (environment with static obstacles)(a)*rpm*-time plot (b)Velocities-time plot (c)Rudder deflection-time plot (d) yaw rate-time plot

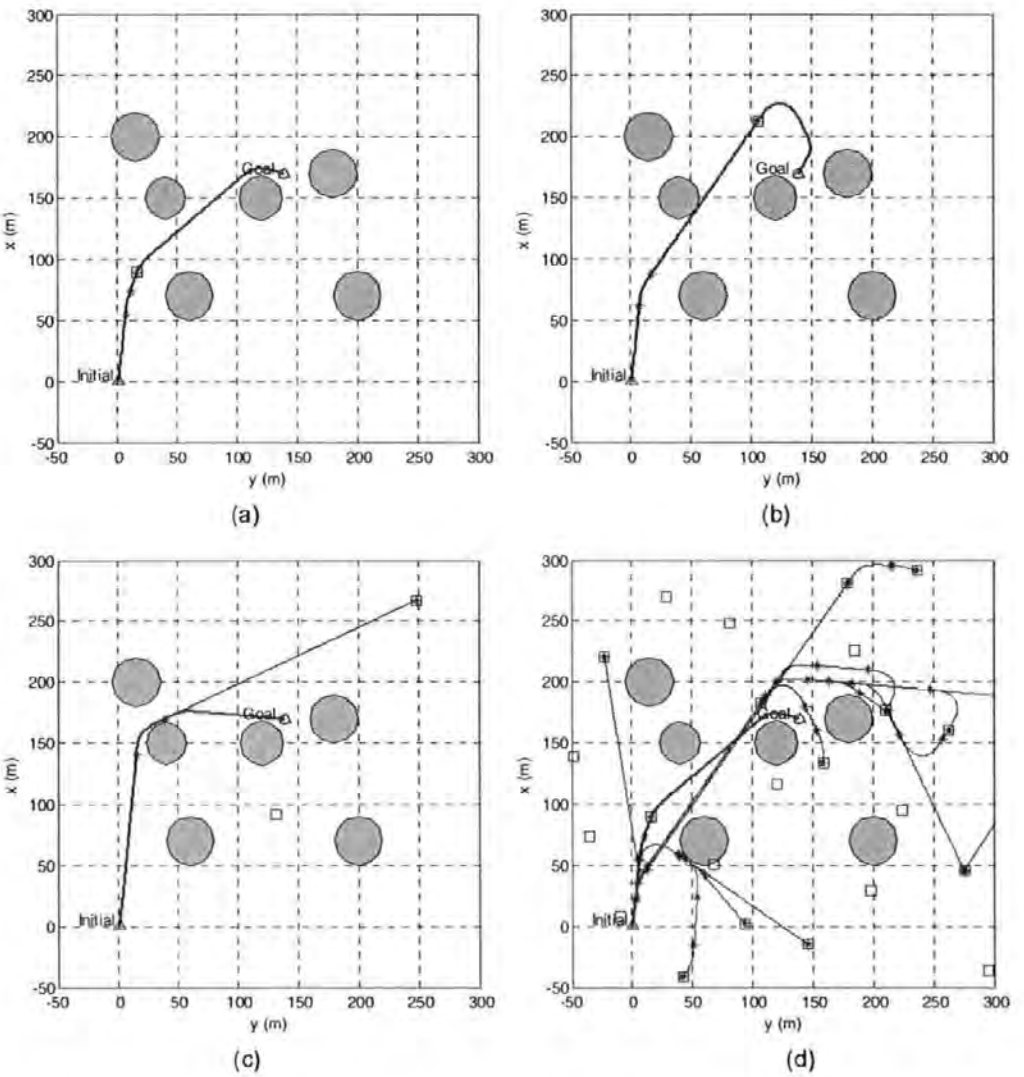


Figure 5.24: Four sample of feasible trajectories *(bold)* found in an environment with static obstacles

5.6.2 Simulation: dynamic environment

A dynamic environment is frequently encountered when there are a few AUVs operating collectively. Indeed, this is a much interesting scenario. The algorithm extension to accommodate dynamic obstacles is straightforward as long as the position of the dynamic obstacles are known *a priori* or can be predicted. Herein, the dynamic obstacles are assumed to have constant velocities. If required, the uncertainty of their position as time progresses can be replicated by expanding the obstacles size through time. Figure 5.25 indicates a feasible trajectory (bold) found by the RRT. Again, to assist in visualising the dynamic effect, it is plotted with respect to time in the z-axis (Fig 5.26 (a) and (b), in different perspective.

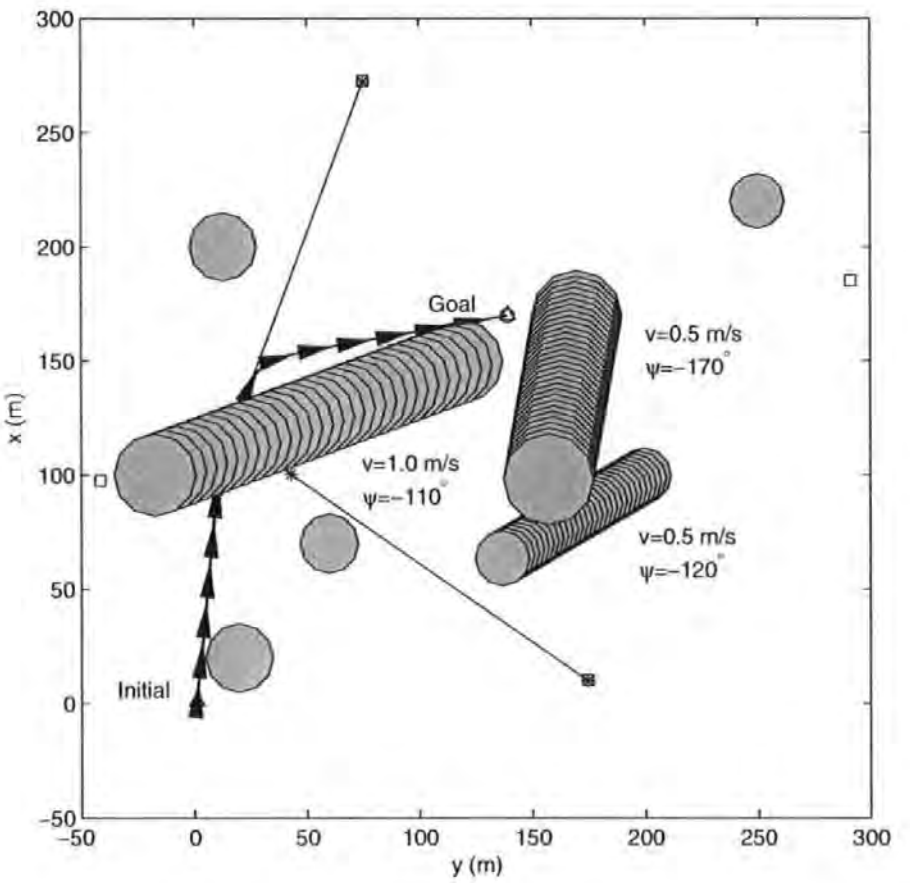


Figure 5.25: A feasible trajectory *(bold)* found in an environment with static and dynamic obstacles

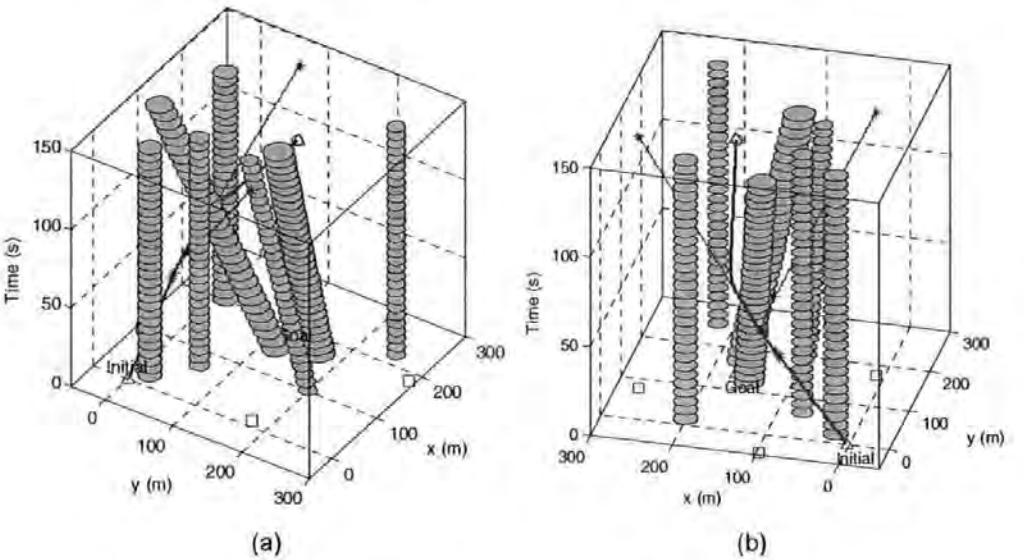


Figure 5.26: x-y-time plots for a feasible trajectory *(bold)* found in an environment with static and dynamic obstacles

The analysis of the inputs history is trivial and very similar to the simulation in static environment version. Figure 5.27(d) shows that two turning manoeuvres, one 15° right turn followed by a more aggressive 60° right turn are required to accomplish this trajectory. This outcome is indeed, in conformity with the shape of the feasible trajectory. The trajectory requires 131.5 s to be completed.

Again, in accordance to the same notation as exposed in Subsection 5.6.1, the MA sequence can be written as follow:

$$g_f = g_0 \times q_1 \times p_{15} \times q_2 \times p_{60} \times q_3 \tag{5.20}$$

The trim trajectory has the following coasting times, $\tau_{q1} = 45.6\ s$, $\tau_{q2} = 11.9\ s$ and $\tau_{q3} = 44.0\ s$. Four other feasible trajectories are provided in Fig 5.28 for comparison.

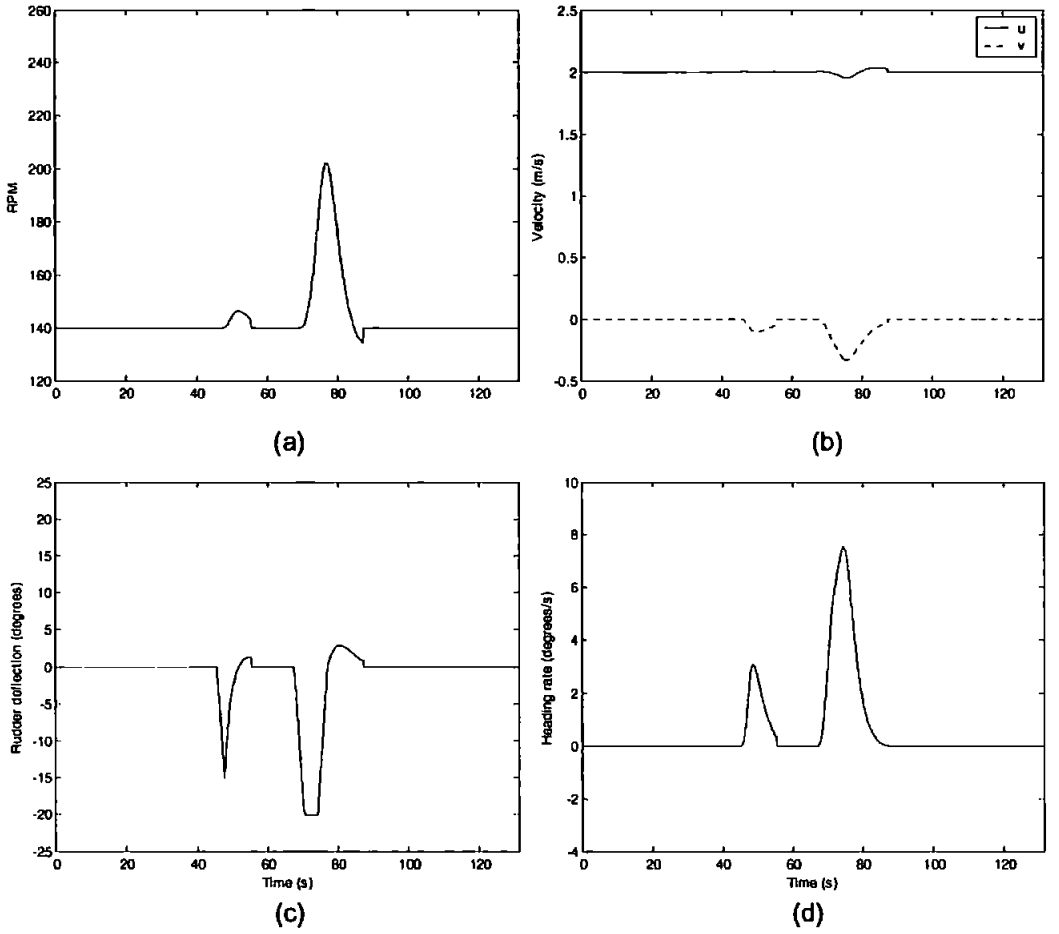


Figure 5.27: Inputs and states history (environment with static and dynamic obstacles) (a)rpm-time plot (b)Velocities-time plot (c)Rudder deflection-time plot (d)Yaw rate-time plot

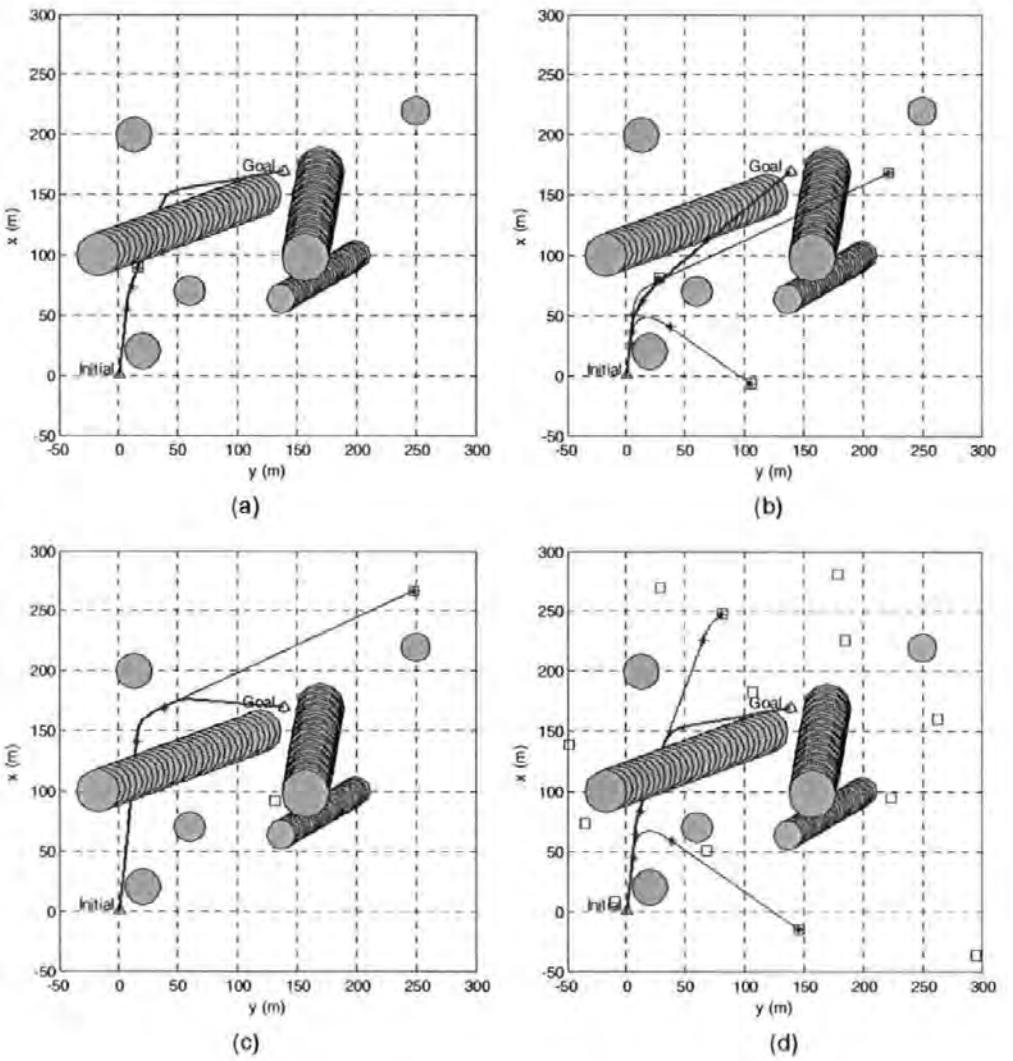


Figure 5.28: Four sample of feasible-trajectories *(bold)* found in an environment with static and dynamic obstacles

5.6.3 Simulation: time-varying final state

The AUV and initial goal positions are the same as in previous simulations, but the goal is now currently moving at 1 m/s in the direction of 2.2 rad from x axis clockwise, simulating a moving submarine. It is true that a vessel that depends on rudder actuation, has to sustain a minimum speed in order not to lose its control effectiveness, which explains the reason behind the moving target. Rudder are devices which develop large lift forces due to an angle of attack with respect to the oncoming fluid. The rudder stalls when the angle of attack reaches a critical value and thereafter develops much less lift.

A sample trajectory is depicted in Fig 5.29. The plot shows that the target is intercepted at position [64 283]. The triangle denotes the moving target. Figure 5.30 illustrates the x-y-time plots.

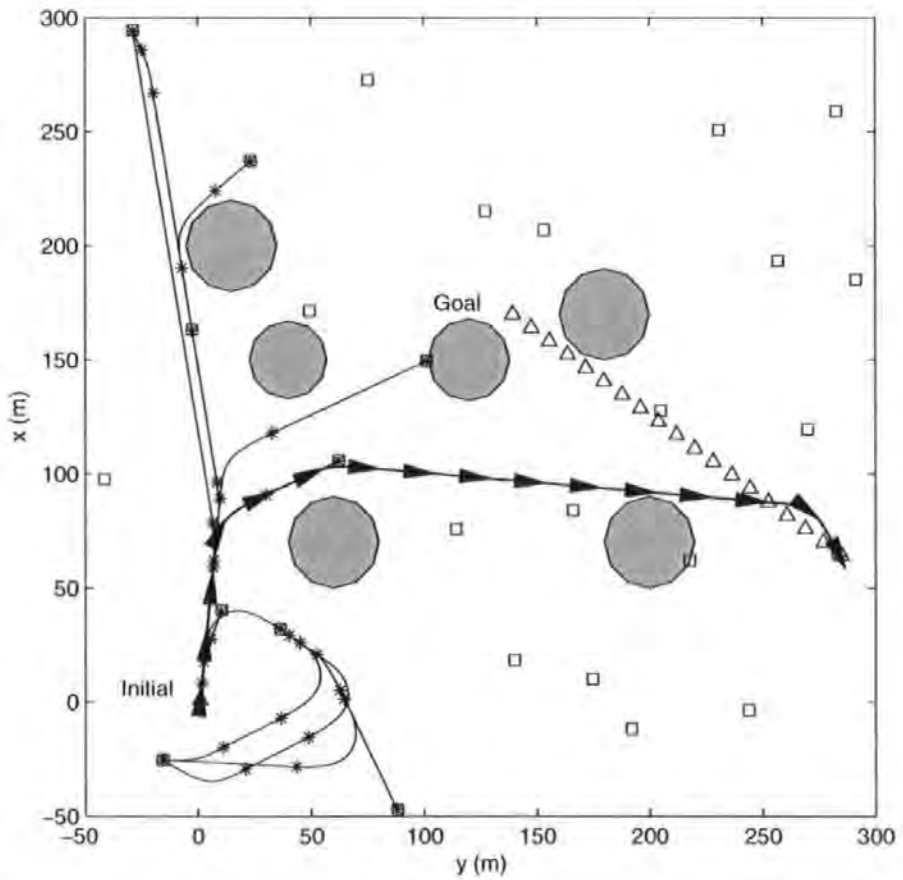


Figure 5.29: A feasible trajectory *(bold)* found in an environment with time-varying final state

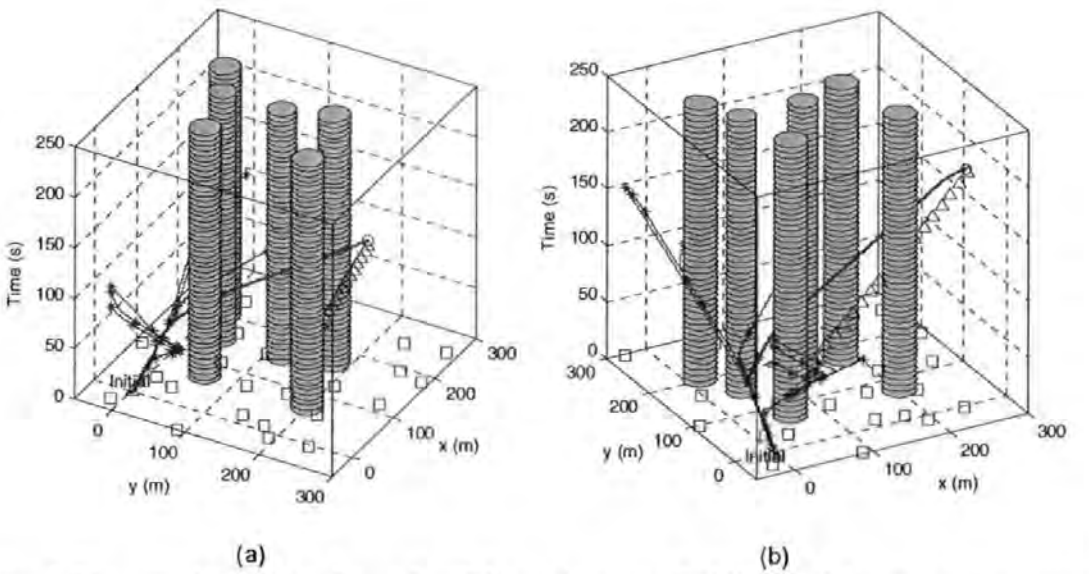


Figure 5.30: x-y-time plots for a feasible trajectory *(bold)* found in an environment with time-varying final state

Figure 5.31 reveals the inputs and states history of the AUV. The MA sequence is given as:

$$g_f = g_0 \times q_1 \times p_{60} \times q_2 \times p_{30} \times q_3 \times p_{60} \tag{5.21}$$

The trim trajectory discussed has the following coasting times, $\tau_{q1} = 30.9\ s$, $\tau_{q2} = 8.7\ s$ and $\tau_{q3} = 91.1\ s$. One can deduce from the above MA transcription that three trim trajectories and three manoeuvres are required to complete the course. The final motion primitive is an aggressive 60° manoeuvre that is not desirable for docking purposes. The ideal manoeuvre will be in a tail-chase fashion. This problem can be solved by constraining the final heading state of the AUV. The trajectory takes 185.1 s to be completed. One can notice that the final manoeuvre is clipped abruptly before it has finished executing. Nonetheless, it is already near the end-point of the manoeuvre and within the tolerance of the final position state (circle of acceptance).

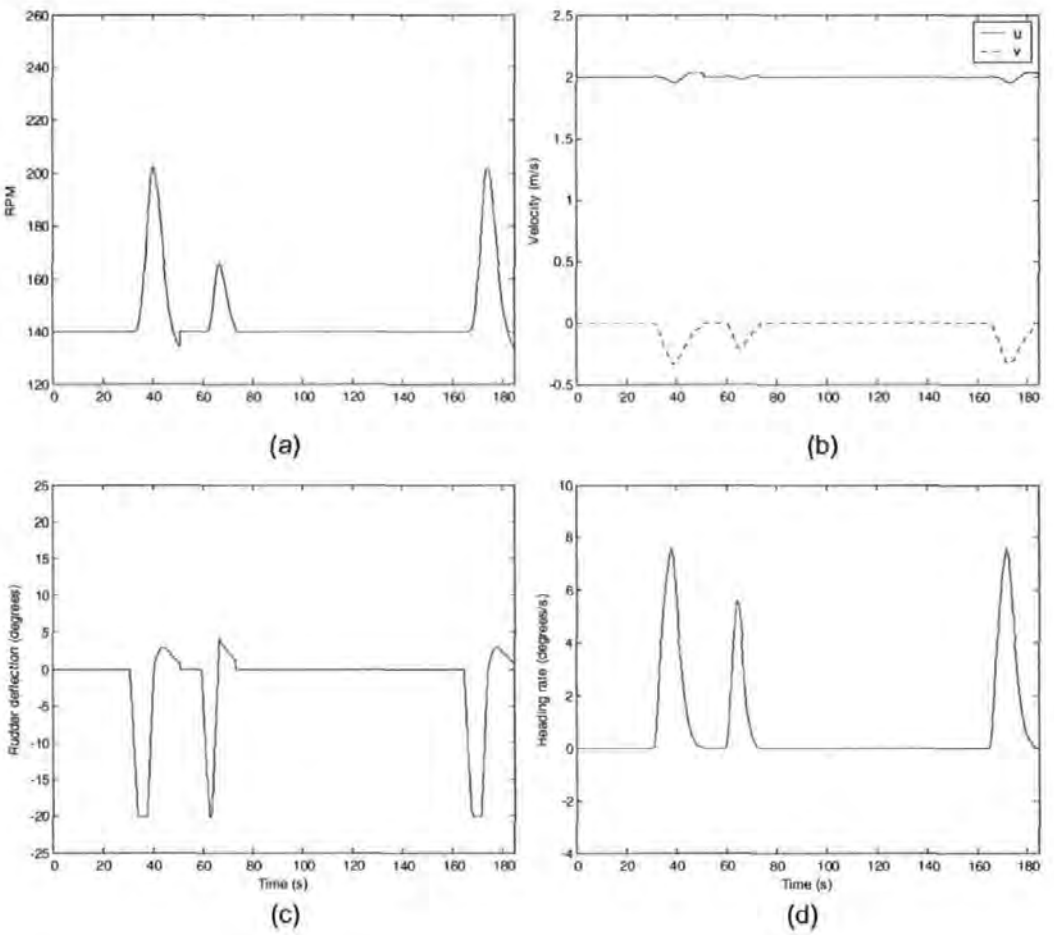


Figure 5.31: Inputs and states history (environment with time-varying final state) (a)rpm-time plot (b)velocities-time plot (c)rudder deflection-time plot (d)yaw rate-time plot

5.6.4 Performance statistics comparison

Given the probabilistic nature of the algorithm, a sample of 30 simulations are run to compile the statistics (Table 5.3) for each scenario. The appraisal is conducted based fundamentally on three criteria as listed below:

1. Computational time needed to provide a solution
2. The quality of the trajectory with reference to the minimum time criterion.
3. The frequency of failure. (Unable to find a solution giving the require time or number of node constraints.)

The simulations were run with 200 maximum nodes and 300 iterations, terminating when either criterion is reached or if a solution is found. Additionally, a maximum time of 4 s was also added to limit the run-time of the program.

Table 5.3: Statistics from 30 sample runs

Statistics	Static	Dynamic	Time-varying
Computational time(mean, s)	1.22	0.68	2.39
Computational time(std, s)	0.91	0.54	0.76
Computational time (median, s)	1.35	0.53	2.48
Trajectory time (mean, s)	159.18	127.25	251.19
Trajectory time (std, s)	37.07	9.80	60.99
Trajectory time (median,s)	148.20	125.25	242.00
Percentage of failures	21%	15%	64%

The above collected descriptive statistics of the algorithm run-time convey some important information regarding the performance of the MA+RRT algorithm for each scenario. Comparing their median statistics, it is observed that the solution time are within two seconds except the time-varying final state case. Unlike the RRT case, the mean and median are actually very close and both can be used as estimators. A significant improvement in run-time performance is to be expected if it is ported into C language.

In this case, the dynamic environment scenario is deemed to be easiest for the algorithm, as evidenced by the low failure rate and low computation time median value. But when the obstacle adjacent to the AUV was altered, to travel in the direction of North-West, the algorithm failed completely. Upon careful analysis, it was concluded that the dynamics of the AUV, the constant 2 m/s cruising speed, is the primary constraint. One simple solution in obviating this problem is to expand the sampling space, hence allowing the AUV to execute a full-circle turn in order to allow the obstructing object to pass before progressing to the goal. A more complicated MA that encompasses a near full performance envelope of the AUV will increase the chances of finding a feasible trajectory.

Again referring to Table 5.3, it reveals that the failure rate for the varying-final state case is the highest, as to be expected. This issue is partly attributed by the 'false-position' algorithm. One interesting anomaly arose during this experiment, as in some cases, the AUV will engage the target head-on instead of the more favourable tail-chase fashion. This is not a desirable trajectory for docking purposes. This phenomenon is to be anticipated with hindsight since the final state heading for the AUV is not explicitly taken in account in the algorithm settings. It can be prevented by wrapping the heading angle and constraining the final state heading in the linear program.

Deducing from several trajectory plots as presented above, one can conclude that the feasible trajectories acquired from the MA+RRT method are smooth in contrast to the one from the RRT algorithm, which is rather erratic. In essence, a smooth predictable trajectory is appealing in several aspects, not only is it beneficial to the actuator lifespan and energy cost but also indirectly functioning as a sub-technology that underpins an embryonic field, known as cooperative robotics. The idea here is to employ multiple AUVs, working collectively to complete a mission in the most efficient manner. In cooperative robotics, effective communication is of paramount importance due to the synchronisation requirements. A communication protocol in the form of a compact symbolic language facilitates robot synchronisation. Considering this, if an AUV motion, is comprised of only a few predictable time-profiles, this can be expressed succinctly using a compact symbolic language, thus lowering the communication bandwidth requirement as a result. A highly desirable trait derived from the adoption of MA representation.

5.7 Concluding Remarks

The primary objective of this chapter is to verify the feasibility of employing MA representation and RRT algorithm to an AUV to solve motion planning problem. Its very short computational time makes it an ideal algorithm for real-time applications. Additionally, simpler algorithm for solving time-varying final state has been proposed. Notwithstanding that, the algorithm is also able to addressed the chattering inputs problem that plagues the RRT algorithm. As far as the author's knowledge is concerned, this is the first attempt in employing the MA and RRT in an AUV.

As the algorithm is inherently a feedforward controller, a robust low-level feedback controller is needed to track the prescribed trajectory when subjected to external disturbance. Chapter 6 is reserved for this interesting topic. In addressing the case of navigating in an unknown environment, a sensor-based motion planning method is needed.

Chapter 6

The Trajectory Trackers

So far in this thesis, the development of the trajectory planner has been described in detail. Chapter 4 concentrated on the application of the RRT algorithm whilst Chapter 5 expanded the topic further by incorporating it with the MA representation. Both of the chapters are substantiated with extensive simulation studies. This chapter, on the other hand, deals with the development of trajectory tracking controllers for an AUV. Together with the trajectory planner, they complete the obstacle avoidance module of an AUV.

As aforementioned, the trajectory generated by the RRT+MA algorithm when supplied to the controller with the corresponding states and inputs is fundamentally a feedforward control law. Understandably, a feedforward system does not function well in the presence of internal and external perturbations. Ocean currents and sensor noise are prime examples of the common perturbations encountered in practice. A small deviation from the prescribed trajectory, if not corrected, will propagate and diverge the system from the desired states. Exacerbating the circumstances, a large trajectory deviation will increase the propensity of collision with the environmental obstacles, which could result in a catastrophic consequence if that happens.

Therefore, deducing from this exposition, it is imperative that the deviation from the reference trajectory be minimised at all time. In other words, one requires to find a control law such that when applied to the AUV, it can achieve asymptotic stabilisation on the reference trajectory. This issue can be resolved via the application

of a trajectory tracker hence leading to the main theme of this chapter. Herein, two variants of state-feedback trajectory trackers, the popular full-state feedback linear quadratic regulator (LQR) and the more advanced nonlinear state dependent Riccati equation (SDRE) controller are examined to evaluate their viability for trajectory tracking purposes in an AUV. Their individual advantages and disadvantages are highlighted and discussed.

It is true that the selection of a particular controller for an AUV is related to several factors. Some of them are

- Robustness to modelling errors (plant parameter variations)
- Disturbance handling characteristics
- Set point tracking and trajectory following
- Stability characteristics
- Application to linear and nonlinear plants
- Simplicity in implementation
- The requirement for a system model

These important factors must be borne in mind as one proceeds to evaluate more appropriate controller for the task in hand.

6.1 Preliminaries

An AUV, similar to the majority of ocean vehicles, is inherently an underactuated system. Briefly, an underactuated vehicle is also considered to be a second order nonholonomic system. This happens when the vehicle has less independent actuators than state variables to be tracked. Mathematically speaking, it is defined that the dimension of the space spanned by the control vector is less than the dimension of the configuration space. Furthermore, the system states are highly coupled and

the non-minimum phase behaviour of an AUV renders controller designs non-trivial (Toussaint 2000).

In reducing the problem complexity of dealing with numerous states, an established technique but very popular amongst the industrial community is to redefine the output space from a three degrees of freedom variable, assuming a planar motion, to a reduced two degrees of freedom variables. This is easily achieved through the use of a guidance control law where the x, y, ψ configuration variables are map into r, θ, the range and line-of-sight angle respectively. The resulting controllers are called course-keeping controllers (CKC), also commonly referred to as autopilots (Fossen 1994).

The crux of the technique is to employ a simple PID controller to generate a control law and assign it to the AUV rudder in order to maintain the desired reference course, which is the course between AUV's current position and the prescribed waypoint. The heading reference commands are frequently catered by an independent guidance system, the line-of-sight scheme remains the most popular (Lin 1991). Static set-points or waypoints for short, are utilised as intermediate 'milestones' or subgoals to assist the AUV in reaching the actual destination. Unfortunately, owing to its dependency upon static set-points, the AUV is susceptible to constant environmental disturbances such as winds, waves and currents, causing the AUV to drift from the ideal course. For this reason, frequent course correction is compulsory to ensure that the AUV arrives at the desired destination. One must also be aware that the separation of guidance and autopilot functions may not yield stability.

Owing to the immense popularity of the aforementioned technique, there are no dearth of articles reporting the implementation and supremacy of various types of autopilot. These autopilots are based on the classical, modern (Naeem *et al.* 2003) to soft-computing (Craven 1999, Zirrili *et al.* 2000) control theories. Hybrid approaches have also been reported to be very successful indeed (Kwiesielewicz *et al.* 2001, Akkizidis *et al.* 2003, Naeem *et al.* 2004).

Recalling from the above, the waypoint guidance scheme, in principle, focuses primarily to ensure that the AUV arrives at the predefined destinations (waypoints) without much consideration with regards to the trajectory undertaken. Obviously, this is not acceptable in a collision avoidance context, as the ability of an AUV to

arrive at a destination and to conform to a predefined trajectory are equally essential in order to prevent a collision and to preserve the structural integrity of the AUV. Therefore, a trajectory tracker is deemed mandatory for the simultaneous attainment of the above objectives. Comparing to autopilots, the design of trajectory trackers are more challenging partly to the multivariable nature and underactuated behaviour of the system. Moreover, the vehicle also exhibits complex nonlinear, hydrodynamic effects that must necessarily be taken into account during the controller design phase. These vehicles also exhibit sway velocity that generate non-zero angles of sideslip.

6.1.1 Types of motion control

Before proceeding to the detail design of the trajectory tracker, it would be more edifying to understand the nature of motion control problem one will be dealing with. As a matter of fact, the problem of motion control can be predominantly partitioned into three distinct classes, in accordance to the difficulty they impose and the way they are solved (Encarnacao and Pascoal 2002).

1. Point stabilisation, where the objective is to stabilise a vehicle at a given point with a desired orientation (static setpoint).
2. Trajectory tracking, where the vehicle is expected to track a time parameterised reference trajectory.
3. Path following, where the vehicle is required to converge to and follow a prescribed path, neglecting any temporal constraints.

The *point stabilisation* problem remains the most challenging amongst the three mentioned, especially when the vehicle being investigated has nonintegradable constraints. In fact, according to the celebrated Brockett's necessary condition for stability, there is no continuous differentiable, static state-feedback control law that can asymptotically stabilise an underactuated system to the equilibrium (a trajectory that degenerates into a single point) (Brockett 1983). In circumventing this difficulty, researchers have offered two techniques: smooth time-varying control laws (Murray and Sastry 1990, Samson 1991) and discontinues feedback laws (Canudis de Witt and Sørdalen 1991, Zhang and Hirschorn 1997).

Aggravating this situation is that an AUV is always required to function in the presence of unknown ocean currents. Interestingly, if the desired orientation does not coincide with the direction of the current, conventional control laws will yield one of two possible behaviours: 1) the vehicle will diverge from the intended target position, 2) the vehicle controller will keep the vehicle moving around a neighbourhood of the desired position, attempting to move persistently to the given point, and consequently inducing an oscillatory behaviour. Unsurprisingly, such behaviours can be anticipated since most AUVs are non-minimum phase systems which possess unstable zero dynamics.

Aguiar and Pascoal (2002) addressed the problem of dynamic positioning an AUV in horizontal plane in the presence of unknown, constant ocean currents by dropping the specification on the final desired orientation and using this extra degree of freedom to force the orientation of the vehicle such that it is aligned with the direction of the current. This 'weather-vane' like property is very well-known and has also been exploited by Pettersen and Fossen (2002). They developed a time-varying feedback control law including integral action that can exponentially stabilise both the positions and the orientation of the ship. The integral action is required to eliminate the oscillatory stationary errors.

The *trajectory tracking* problem entails the design of feedback control laws that can force the vehicle to reach and track a time parameterised inertial trajectory (a curve in the state-space with an associated timing law). This technique is very important in an environment with dynamic obstacles. Here, the temporal specifications must be strictly complied to avoid a collision. The majority of tracking controller schemes for underactuated marine vehicles employ the classical approaches such as local linearisation and decoupling of the multivariable model to steer as many degrees of freedom as the available control inputs (Repoulias and Papadopoulos 2005). This is frequently achieved by linearisation of the vehicle's error dynamics about trimming trajectories before proceeding to applying a state-feedback controller (Walsh *et al.* 1994, Divelbiss and Wen 1997, Toussaint 2000). Again, similar to most linearised models, the validity of these solutions is only limited about a small neighbourhood around the selected operating points.

A nonlinear Lyapunov-based technique has been shown to be a rather promising approach (Silvestre *et al.* 1995). Frazzoli (2001), on the other hand, implemented

a robust nonlinear controller based on backstepping approach on an autonomous helicopter. Logic-based switching control was proposed by Aguiar and Hespanha (2004). Notwithstanding the modern control theory approach, Vukic *et al.* (2001) preferred a soft-computing paradigm and they employed a neurocontroller instead. Wettergreen *et al.* (1999) applied an evolving neurocontroller based on Q-learning for a visual servoing task in the *Kambara* AUV. Strictly speaking, this is not a trajectory tracker *per se*, but a path tracker. In addition, Jiang *et al.* (2005) advocated the use of a predictive fuzzy logic controller. Nonetheless, the short-supply of rule-based fuzzy logic techniques being applied to this problem can be attributed to the multivariable nature of the problem which tends to cause state-explosion of the fuzzy rules.

The *path following* problem requires the vehicle to follow a path without complying to any temporal specifications. One clear advantage derived from this approach, is that the vehicle forward speed need not be controlled as rigorously when following the path since it is sufficient to act on the vehicle orientation to drive it to the path, therefore ensuring a smoother convergence to a path. As a consequence, the control signals are less likely to be pushed to saturation when compared to the performance obtained with trajectory tracking controllers (Hindman and Hauser 1992). Path following systems for underactuated vehicles especially of the marine type can be found in (Do *et al.* 2002, Aguiar and Hespanha 2004, Aguiar *et al.* 2005).

Unfortunately, by abrogating the temporal specifications, the solutions become less versatile and cannot be applied in an environment populated by dynamic obstacles. In fact, successful collision avoidance cannot be guaranteed in this context. Distinctly, this also precludes its usage in multiple-AUV rendezvous and cooperative missions, missions where the manoeuvre synchronisation of the AUVs are critical. For these reasons, the path following problem is considered to be less interesting, which explains the lack of attention it is receiving.

6.2 The Proposed Approach

Again, recalling from above that the Brockett's condition imposes a certain amount of difficulty to the point stabilisation problem. Therefore, it will be prudent if, instead of concentrating on a point stabilisation problem, one diverts the attention to stabil-

isation about a prescribed reference trajectory as adopted in this chapter. Note that the trajectory in this context must not degenerate to a point, implying that the surge velocity of the AUV must be nonzero. The problem is also made more complicated by the very fact that the underactuated vehicles such as the AUVs. As a consequence, this rules out any attempt to design a feedback only controller that would rely on its kinematic equations.

Nonetheless, this shortcoming can be addressed by the introduction of a feedforward unit. In fact, the use of a dedicated feedforward unit in the trajectory tracking case is not entirely new. Nieuwstadt (1997) has shown that substantial performance improvement can be derived from a system equipped with a feedforward unit compared to a feedback only controller. A feedforward unit is also beneficial to systems that suffer from large transport delays, in essence it evokes a form of anticipatory behaviour. One can conclude that feedback control is reactive but feedforward control is proactive. In spite of the advantages, feedforward control can only respond to known kinds of disturbances, but cannot do much with novel disturbances. As a matter of fact, great care must be taken in the design of such a unit, since it has the propensity to destabilise a system. Consequently, feedforward units are rarely employed alone and must be combined with feedback units. Merging the two units creates a two degrees of freedom controller as depicted in Fig 6.1.

Contextual speaking, the reference trajectory is consisted of the desired inertial position and the corresponding velocities. The reference trajectory must not only be consistent with the dynamics of the vehicle but also freed from environmental obstacles. A trajectory satisfying these two conditions is termed as a feasible trajectory. Clearly, the location of the obstacles must also be known *a priori* to a certain extend for that to be achieved.

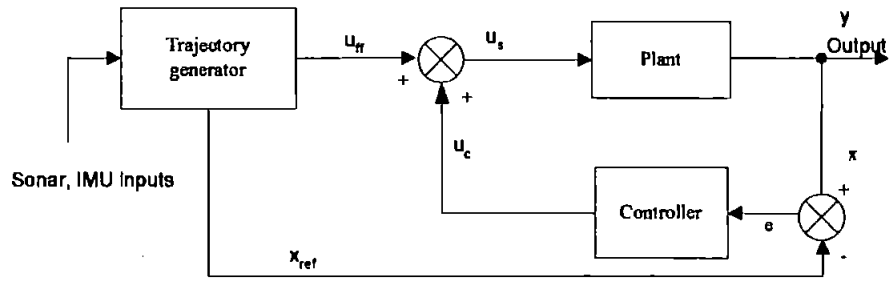


Figure 6.1: Controller structure

With that in mind, the trajectory tracking technique proposed can be defined as such: **Given that a feasible trajectory for an underactuated system is generated by an open-loop planner, one can then compute the linearisation of the system about this nominal trajectory.** For this particular case, the open-loop planner pertained is no other than the novel MA+RRT algorithm developed in Chapter 5. Moreover, if the linear time-varying system acquired through the linearisation process is completely controllable in a certain sense, one can then define a linear time-varying feedback law that forces the tracking errors to a neighbourhood of zero that can be reduced arbitrarily. To describe another way, it locally stabilised the system about this nominal trajectory. The control laws proposed herein were derived by solving the celebrated Riccati equation, in accordance with the spirit of optimal control. Such a controller, if properly designed should also be able to reject many types of disturbance including noise in the sensors, initial condition errors, and errors injected along the trajectory.

Even so, the closed-loop dynamics of the AUV can yield instability if it diverges too far from the neighbourhood of the reference trajectory since the errors introduced by the linearisation process becomes too large to be neglected. Consequently, it is crucial to define a state-space tube or manifold as mentioned in Subsection 5.5.2 be defined.

6.3 Motivation of Using Optimal Control

Amongst all the high performance, multivariable control methodologies, the optimal control framework brings forth the best balance between successful practical implementation, optimal system performance and mathematical tractability. In fact, the optimal control paradigm encompasses the LQR, model predictive control (MPC), H-infinity control and so forth. It is not surprising that it has been a preferred choice amongst controls academicians and engineers for tackling multivariable, high performance system control.

In the optimal control paradigm, the problem is cast into a minimisation or maximisation problem for which an objective function is defined, and the control law found must also satisfy the dynamics equation of the system. In most of the cases, the

objective function or performance index is a function of the design parameters or states to optimise. The use of these physical related quantities facilitates the tuning process. According to Burl (1998), a general requirement for the selection of a suitable objective function is (1) it should accurately reflect the designer's concept of good performance and (2) the control moves should be computed with a reasonable amount of effort. The latter requirement can actually be relaxed due to the recent availability of abundance and inexpensive computing power.

The traditional pole placement (pole assignment) approach works by placing the poles at designer's chosen location to attain some design specifications such as overshoot, rise time, settling time or bandwidth. However, a critical disadvantage of using this technique is that the pole locations must be worked out in advance. Moreover, the controller obtained by this method is not always optimal and a trial and error procedure is adopted until the system performance coincides with the desired specification. This difficulty is amplified if the system is of a multivariable type.

Optimal control theory suggests the poles at points should be placed such that the resulting controller is optimal in some sense. Furthermore, the knowledge of the pole locations prior to the design is not needed. The designer is needed to only decide what merit to use and achieve the design specifications by tuning the weighting coefficients contained in the objective function. The best pole locations to attain the desired response is then relegated to the algorithm.

Mathematically, optimal control system design can be defined as such: given the constraints $\mathbf{U}$ on control functions $\mathbf{u(t)}$ that form the set of admissible controls $\mathbf{u}(t) \in \mathbf{U}$ for all $t \in [t_0, t_f]$ and the constraints $\mathbf{X}$ on the state trajectories $\mathbf{x}(t)$ that form the set of admissible trajectories $\mathbf{x}(t) \in \mathbf{X}$ for all $t \in [t_0, t_f]$, the optimal control problem is to find an admissible control function $\mathbf{u}(t)$ that forces the continuous-time system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \tag{6.1}$$

or discrete-time control system

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}, \mathbf{u}(k)) \tag{6.2}$$

to follow an admissible trajectory $\mathbf{x}(t)$ while minimising the performance objective of

$$J = S(\mathbf{x}(t_f), t_f) + \int_{t=t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t)dt \tag{6.3}$$

whereas in discrete-time performance criterion is

$$J = S(\mathbf{x}(N)) + \sum_{k=0}^{N-1} L(\mathbf{x}(k), \mathbf{u}(k)) \tag{6.4}$$

If the solution to the optimal control problem can be found in the continuous form of

$$\mathbf{u}(t) = \mathbf{u}(\mathbf{x}(t), t) \tag{6.5}$$

or in discrete form

$$\mathbf{u}(k) = \mathbf{u}(\mathbf{x}(k)) \tag{6.6}$$

then if the control is said to exist in the closed-loop form, is refereed to as the optimal control law.

6.4 Linear Quadratic Regulator Design

The crux of the LQR is to find a linear stabilising feedback control law for the plant in question. It is an optimal controller which is derived on the basis of a linear model of a plant and a quadratic cost function to be optimised, hence the term 'linear quadratic'. Fig 6.2 depicts a simplified block diagram of an LQR, where $\mathbf{x}$ is the state vector and $\mathbf{u}$ is the input vector.

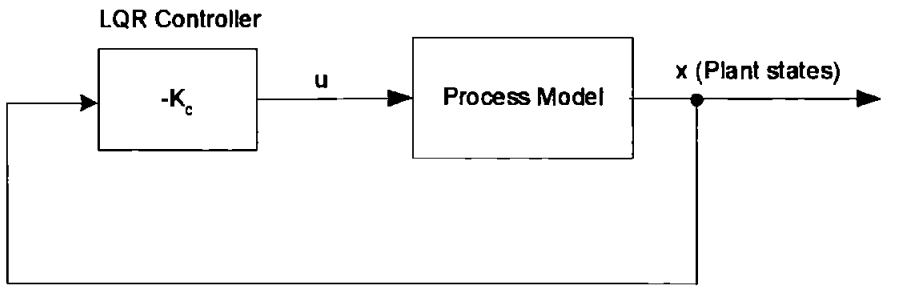


Figure 6.2: A block diagram of a generic LQR

Plainly, a linearised time-varying plant can written in a state-space form as

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \tag{6.7}$$

and quadratic cost functions,

$$S = \mathbf{x}^T(t_f)\mathbf{M}\mathbf{x}(t_f) \tag{6.8}$$

and

$$L = \mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}\mathbf{u}(t) \tag{6.9}$$

Matrices $\mathbf{M}$, $\mathbf{Q}$, and $\mathbf{R}$ must be square; $\mathbf{M}$ and $\mathbf{Q}$ must have a length equal to the number of states; and $\mathbf{R}$ must corresponds in dimension to the number of inputs. Additionally, to ensure that the solution is unique and finite, matrices $\mathbf{M}$ and $\mathbf{Q}$ must be positive semidefinite, whereby matrix $\mathbf{R}$ must be positive definite.

A definite requirement of the LQR controller is that the plant must be pointwise controllable and that the matrix $\mathbf{A}$ is nonsingular. When this is true then there exists a constant feedback gain matrix $\mathbf{K}_c$ that allows the eigenvalues of the closed-loop system to be assigned arbitrarily. This is mathematically stated by forming a controllability matrix $\mathbf{S}$ in terms of the matrices $\mathbf{A}$ and $\mathbf{B}$ given by

$$\mathbf{S} = \begin{bmatrix} \mathbf{B} & \mathbf{AB} & \mathbf{A}^2\mathbf{B} & \dots & \mathbf{A}^{n-1}\mathbf{B} \end{bmatrix} \tag{6.10}$$

Then the system is said to be *pointwise controllable* if the matrix $\mathbf{S}$ has rank n. Nonetheless, in practice, this requirement can be too restrictive and commonly relaxed, a less demanding requirement called *stabilisable* is frequently preferred. A system is stabilisable if the states that are not controllable are stable. These states are explicitly ignored and a new state transfer matrix is defined.

It is now required to evaluate the contents of $\mathbf{K}_c$ or in other words, the state feedback gain, such that a performance index is minimised whilst subjected to satisfying Equation 6.7. It should be observed that the state vector sequence $\mathbf{x}(t)$ and the input sequence $\mathbf{u}(t)$ are not independent variables that can be arbitrarily chosen to minimise J. Nonetheless, prior to finding $\mathbf{K}_c$, one has to proceed to find a symmetric positive semi definite matrix $\mathbf{P}$ which is the solution to the matrix riccati equation. It is worth mentioning that, there exists two families of the problem that called for different approaches. They can be classified as the finite planning horizon problem

and the infinite planning horizon problem.

For the specific case of finite planning horizon, $t \leq \infty$, linear-quadratic optimal control problems are solved by finding the symmetric positive semi definite P matrix that satisfies the matrix Riccati equation (MRE):

$$\dot{\mathbf{P}}(t) + \mathbf{A}(t)^T\mathbf{P}(t) + \mathbf{Q}(t)x + \mathbf{P}(t)\mathbf{A}(t) - \mathbf{P}(t)\mathbf{B}\mathbf{R}(t)^{-1}\mathbf{B}^T\mathbf{P}(t) + \mathbf{M} = 0, \quad \mathbf{P}(t_f) = \mathbf{M} \tag{6.11}$$

Conversely, for the infinite horizon problem, where $t = \infty$, one must satisfy the algebraic Riccati equation (ARE):

$$\mathbf{PA} + \mathbf{A}^T\mathbf{P} + \mathbf{Q} - \mathbf{PBR}^{-1}\mathbf{B}^T\mathbf{P} = 0, \tag{6.12}$$

Algebraic Riccati Equation is actually a unique case of the more general MRE where $\dot{\mathbf{P}} = 0$ and $\mathbf{M} = 0$.

Once $\mathbf{P}(t)$ is found, it can be substituted into equation below to obtain the state feedback gain matrix as given by

$$\mathbf{K_c} = [\mathbf{R} + \mathbf{B}^T\mathbf{PB}]^{-1}\mathbf{B}^T\mathbf{PA} \tag{6.13}$$

Then with K_c, upon substitution into Equation 6.14 provides the required control effort. The LQR output can be defined as

$$\mathbf{u}(k) = -\mathbf{K}_c\mathbf{x}(k) \tag{6.14}$$

For set point tracking or trajectory following cases, a feedforward unit must be added to eliminate the offset error. The LQR also assumes the availability of full state-feedback and the states measured are not corrupted by noise. In practice, this is addressed by introducing a Kalman filter to reconstruct the unavailable or noisy states. The resulting controller is termed as Linear Quadratic Gaussian (LQG) controller since it is optimal for processes contaminated with Gaussian noise. Augmenting a Kalman filter for state estimation seriously degrades the excellent stability margins available for an LQR controller.

6.4.1 Selection of weighting matrices

Control systems are often designed to specifications that involve the settling time, damping ratio and bandwidth constraint. They are also subjected to constraints on the maximum output error, state error and the maximum control input. Violating these constraints will induce actuator saturation, which also renders the linearised system nonlinear and invalidating the optimal control law. To achieve the specifications, one employs a trial and error selection of the weighting matrices $\mathbf{P}$, $\mathbf{Q}$ and $\mathbf{R}$ in the objective function defined in Equations 6.8 and 6.9.

The size of the weighting matrices is altered to yield a desired settling time or some other performance criterion. Through this tuning process, a tradeoff between speed of response and control effort can be accomplished. In the case of 'expensive' control, when $\mathbf{R} \gg \mathbf{Q}$, assuming that both the matrices have been normalised, the cost function is dominated by the control effort $\mathbf{u}$, and so the controller minimises the control action itself, ensuing an energy efficient, albeit sluggish response. Conversely, in the case of 'inexpensive' control, when $\mathbf{R} \ll \mathbf{Q}$, the cost function is dominated by the state errors, and there is no penalty for using large inputs, hence affording a more agile response. The control effort magnitude is actually limited by actuator saturation and its slew rate, therefore an excessively responsive closed-loop system has the inclination to cause actuator saturation. Similarly, the terminal state's weighting coefficient $\mathbf{M}$, determines how close to the desired final state the system will be at t_f.

A good starting point for trial and error selection of the state weighting matrix is to set the various state contributions approximately equal as given by the Bryson's rule (Bryson and Ho 1975). In principle, the rule scales the variables that appear in matrices $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{M}$ such that the maximum acceptable value for each term is unity. In other words, it 'normalises' the variables such that each of them has equal contribution. This is particularly important when there is a large discrepancy between the range of the variables such as in the case of different units. The different elements in of $\mathbf{u}$ and $\mathbf{x}$ make the values for these variables numerically very different from each other.

In the Bryson's rule, the diagonal $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{M}$ are selected with the following normali-

sation;

$$\begin{aligned} \mathbf{Q}_{ii} &= \frac{1}{\text{maximum acceptable value of } \mathbf{y}_i^2} & i \in 1, 2, \ldots \ell, \\ \mathbf{R}_{jj} &= \frac{1}{\text{maximum acceptable value of } \mathbf{u}_i^2} & j \in 1, 2, \ldots \ell, \\ \mathbf{M}_{kk} &= \frac{1}{\text{maximum acceptable value of } \mathbf{y}(t_f)_i^2} & k \in 1, 2, \ldots \ell, \end{aligned} \tag{6.15}$$

Although Bryson's rule usually gives a satisfactory results, often it is just the starting point to a trial and error iterative design procedure aimed at obtaining desirable properties for the closed-loop system.

6.4.2 State error model and linearisation

LQR is a form of regulator, and its intrinsic nature is to drive all the states, assuming controllable, to zero. When all the states reach zero then the controller output will be equated to zero. Thence before pursuing with the linearisation process, one needs to recast the formulation into state error form. To put mathematically, the state error is expressed as below,

$$\mathbf{x}_{error} = \mathbf{x}_{measured} - \mathbf{x}_{ref} \tag{6.16}$$

For all the LQR problems, one can safely assume that the $\mathbf{x}$ is actually $\mathbf{x}_{error}$. Given the fact that one is actually dealing with a tracking problem, one must also introduce a feedforward unit to ensure proper set point tracking. It can be concluded that when all the states reach zero, this implies that the system is exactly at the desired trajectory. A block diagram pertaining to this controller is given in Fig 6.3.

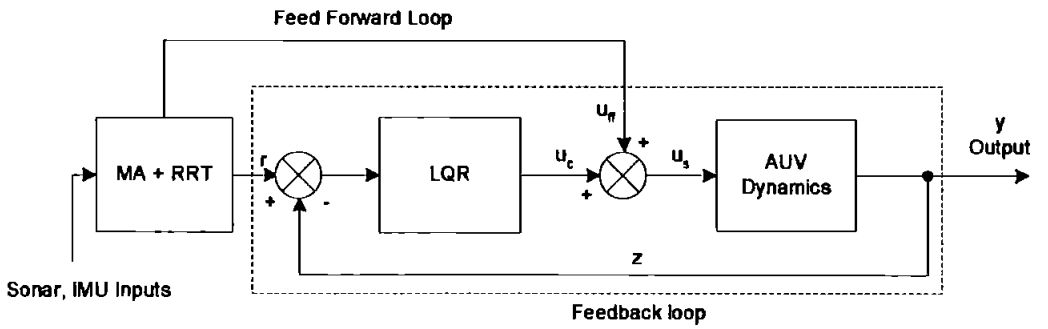


Figure 6.3: The LQR with a feedforward unit in an AUV

Referring to Fig 6.3, notice that the net control input supplied to the vehicle is the

sum of the controller and the feedforward unit outputs as expressed below,

$$\mathbf{u}_s = \mathbf{u}_c + \mathbf{u}_{ff} \tag{6.17}$$

It is evident from Fig 6.3 that the tracking controller and the feedforward unit are operating independently. Indeed, as will be explained later, this deficiency of 'information interaction' has a propensity to induce instability.

In the LQR design, a linearised model is necessary. The parameters $X_r|r|, mX_gY_{v|v|}$, and $Y_{r|r|}$ in Equation 3.16 are neglected since their quadratic products are too small. For the sake of mathematical clarity, it is frequently preferable to convert the nonlinear model into a standard form for underactuated vehicles before transforming the equations into state error equations. The standard form describing an underactuated vehicle executing planar motions can be expressed as (Toussaint 2000):

$$\dot{u} = m_u vr - d_u u + au_1 + w_1 \tag{6.18}$$

$$\dot{v} = m_v ur - d_v v + bu_2 + w_2 \tag{6.19}$$

$$\dot{r} = m_r uv - d_r r + cu_2 + w_3 \tag{6.20}$$

$$\dot{x} = u\cos(\psi) - v\sin(\psi) \tag{6.21}$$

$$\dot{y} = u\sin(\psi) + v\cos(\psi) \tag{6.22}$$

$$\dot{\psi} = r \tag{6.23}$$

Again, deducing from the above equation, the control inputs are not only coupled to the vehicle dynamics but also not affine since it is dependent on the square of the surge velocity. One must also set the operating point to linearise the equations. Herein, the surge velocity is set at 2 m/s, corresponding to the normal cruising speed of the *AUTOSUB*. The variables a, b and c, are constants. The variables w_* are used to model external disturbances acting of the vehicles but they are assumed to be zero for this case, as the LQR assumes no noise corruption. These terms would be required for H_∞ control.

Accordingly, one defines the current states using error states and desired states (ref-

erence state) as,

$$\begin{bmatrix} u \\ v \\ x \\ y \\ r \\ \psi \end{bmatrix} = \begin{bmatrix} u_e \\ v_e \\ x_e \\ y_e \\ r_e \\ \psi_e \end{bmatrix} + \begin{bmatrix} u_d \\ v_d \\ x_d \\ y_d \\ r_d \\ \psi_d \end{bmatrix} \tag{6.24}$$

Referring to Equation 3.16, where m is the inertia matrix with the associated states $[u\ v\ r\ \psi]^T$, one can define a new matrix below;

$$m^{-1} = \begin{bmatrix} m_{11} & 0 & 0 & 0 \\ 0 & m_{22} & m_{23} & 0 \\ 0 & m_{32} & m_{33} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{6.25}$$

Using a Taylor's series, the linearised model of *AUTOSUB*'s dynamic equations can be reformatted as below;

$$\begin{bmatrix} \dot{u_e} \\ \dot{v_e} \\ \dot{x_e} \\ \dot{y_e} \\ \dot{r_e} \\ \dot{\psi_e} \end{bmatrix} = \begin{bmatrix} 2m_a u_d & m_a r_d & 0 & 0 & m_a v_d & 0 \\ m_b v_d + m_c r_d & m_b u_d & 0 & 0 & m_c u_d & 0 \\ \cos(\psi_d) & -\sin(\psi_d) & 0 & 0 & 0 & 0 \\ \sin(\psi_d) & \cos\psi_d & 0 & 0 & 0 & 0 \\ m_d v_d + m_e r_d & m_d u_d & 0 & 0 & m_e u_d & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_e \\ v_e \\ x_e \\ y_e \\ r_e \\ \psi_e \end{bmatrix} + \begin{bmatrix} a & 0 \\ 0 & b \\ 0 & 0 \\ 0 & 0 \\ 0 & c \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_{1e} \\ u_{2e} \end{bmatrix} \tag{6.26}$$

where $m(.)$ are defined as follows;

$$\begin{aligned} m_a &\equiv m_{11} \\ m_b &\equiv m_{22} Y_{uv} + m_{23} N_{uv} \\ m_c &\equiv m_{22}(Y_{ur} - m) + m_{23}(N_{ur} + m X_g) \\ m_d &\equiv m_{32} Y_{ur} + m_{33} N_{ur} \\ m_e &\equiv m_{32}(Y_{ur} - m) + m_{33}(N_{ur} - m X_g) \\ a &\equiv 140 X_{prop} \\ b &\equiv 8 Y_{\delta r} \\ c &\equiv 8 N_{\delta r} \end{aligned} \tag{6.27}$$

Equation 6.26 contains a time-varying state transfer matrix $\mathbf{A}(t)$. Different trajecto-

ries will have different values due to the time-varying nature of variables u_d, v_d and ψ_d. This is essentially a finite horizon problem since $t_f \neq \infty$. $\mathbf{A}(t)$, however, will revert into a time-invariant matrix if the trajectories are restricted to only rectilinear motions.

6.5 Solving the Matrix Riccati Equation

Several numerical methods exists in solving the MRE, since analytical methods are rather limited. Darling (1997) proposed a technique where the MRE of matrix $n \times n$ size is transformed into a system of $n(n+1)/2$ linear second order ordinary differential equations, also known in differential geometry as the Jacobi equation.

As aforementioned, the problem one is dealing here is of a finite horizon variant. Herein, the approach recommended by Lewis (1986) is adopted. The optimal feedback control gains are computed backward in time according to the following recursive equations;

$$\mathbf{K}(k) = [\mathbf{B}^T\mathbf{P}(k)\mathbf{B} + \mathbf{R}(k)]^{-1}\mathbf{B}^T\mathbf{P}(k+1)\mathbf{A}(k) \tag{6.28}$$

$$\mathbf{P}(k) = \mathbf{A}^T(k)\mathbf{P}(k+1)[\mathbf{A}(k) - \mathbf{B}(k)\mathbf{K}(k)] + \mathbf{Q}(k) \tag{6.29}$$

Where $\mathbf{P}(N) \geq 0$, is the end configuration weighting matrix, $\mathbf{Q}(k) \geq 0$ is the configuration weighting matrix, and $\mathbf{R}(k) > 0$ is the control weighting matrix. The Riccati equation is always solved in reverse time order. The $\mathbf{P}(N) = \mathbf{M}$, $\mathbf{Q}(k)$, and $\mathbf{R}(k)$ matrices are diagonal and can be chosen to be time-invariant for best performance. A detailed exposition pertaining the tuning of these weights is delineated in Subsection 6.5.1.

One must also bear in mind that for every sampling interval of Equation 6.29, one requires a new state transfer matrix $\mathbf{A}(k)$ due to its time-varying nature. On the other hand, the input matrix $\mathbf{B}(k)$ is time-invariant for this specific case. Furthermore, the original Equation 6.26 is in continuous form. This suggests that a transformation from a continuous-time system to discrete-time approximation is necessary. A plethora of algorithms such as the Euler's method, the zero-order hold approximation, the bilinear transformation, the impulse invariant approximation can be accessed but two of the

more popular approximation methods are Euler's method and the zero-order hold approximation.

The zero order hold method was chosen in this context, as it provides a mean of generating discrete-time state equation for systems that are time-varying and non-linear. Whereas, the zero-order hold approximation is more suitable for linear, time invariant systems. In the Euler's method, it is critical that the sampling time be sufficiently short that the finite-time derivative is an accurate approximation of the true derivative (Burl 1998).

A discrete-time state equation can be obtained from the continuous state equation using Euler's method of approximating the derivative:

$$\frac{\mathbf{x}(kT+T)-\mathbf{x}(kT)}{T} \approx \dot{\mathbf{x}}(kT) \tag{6.30}$$

Hence, in computational form, a difference equation for the state can be explicitly expressed as

$$\mathbf{x}(kT+T) = \mathbf{x}(kT) + T(\mathbf{A}x(kT) + \mathbf{B}u(kT)) \tag{6.31}$$

or

$$\mathbf{x}(k+1) = \mathbf{x}(k) + T(\mathbf{A}x(k) + \mathbf{B}u(k)) \tag{6.32}$$

6.5.1 Tuning of the LQR weighting matrices

The tuning of the parameters **M**, **Q**, and **R** is nontrivial. The following weighting matrices had been chosen experimentally and employed in the remaining simulations.

$$\mathbf{Q} = diag([40\ 40\ 0.1\ 0.1\ 50\ 3000]) \tag{6.33}$$

$$\mathbf{R} = diag([62.5\ 1000]) \tag{6.34}$$

$$\mathbf{M} = diag([400\ 1000\ 4\ 4\ 100000\ 1000]) \tag{6.35}$$

Notice that there is a large range difference between the elements of the matrix, this is caused by the variety of units employed. The system response is very sensitive to the element values. Again, the weighting matrices above are time-invariant, exploitation

of time-varying version is expected to improve the controller performance but will definitely make the tuning process even more challenging, owing to the increase degree of freedom. Several factors contributed to the tuning problem above: 1) There are too many tuning parameters, 2) The coupling between the states and 3) The states x and y are based on Earth-fixed frame. These states change according to the particular trajectory, which implies that the controller is trajectory-dependent, making retuning a compulsory process for each different trajectory. Clearly, there is a great need for future research in this particular area. Once the gain matrix, $\mathbf{K}$ is found, the net control effort is obtained by a simple substitution into Equation 6.14.

6.6 State Dependent Riccati Equation (SDRE) Control

There are a myriad of nonlinear control techniques that are applicable to complex systems. Amongst the more attractive ones are the state feedback linearisation, adaptive control, model predictive control (receding horizon control), sliding mode, recursive backstepping, neural network, fuzzy logic, SDRE and so forth.

An excellent review of SDRE techniques can be found in Cloutier (1997). The very idea of using state-feedback Riccati equation based linear control methods for nonlinear systems was originally proposed by Pearson (1962) and later Cloutier *et al.* (1996) attempted to revitalised the interest. Coupled with the rapid advancement of microprocessor technology, which guarantees the availability of low cost and abundance computing power, this has made SDRE control a promising candidate for the control of nonlinear systems.

For instance, SDRE based designs have been used in advanced guidance law development for high performance aircraft and missiles (Cloutier *et al.* 1996, Wise and Sedwick 1997, Menon *et al.* 2004). Likewise, Parrish and Ridgely (1997) employed it for altitude control of a satellite. From a practical viewpoint, results of a real-time experiment using a two-link underactuated robot, which is a highly nonlinear fourth order system was also presented by Erdem (2001). In addition, SDRE control with nonlinear feedforward compensation for a small unmanned helicopter has been at-

tempted by Bogdanov *et al.* (2003). The helicopter successfully conducted various aggressive preprogrammed acrobatic manoeuvres.

Mathematically, the SDRE controller shares a great deal of similarity with the LQG controller, then again, its formulation and operation bear striking resemblance to the MPC. Unlike the LQR with infinite horizon case, instead of using a set of fixed weighting matrices, both the parameters $\mathbf{Q}(\mathbf{x})$ and $\mathbf{R}(\mathbf{x})$ can vary as a function of states. In other words, the state transfer matrix $\mathbf{A}(\mathbf{x})$ and $\mathbf{B}(\mathbf{x})$ can be defined such that they are dependent on the states *per se*. The ability to vary the $\mathbf{Q}$ and $\mathbf{R}$ 'on-the-fly', allows one the flexibly to expand the performance envelope of the plant. There is also a possibility to impose hard bounds on the control, control rate, and control acceleration to avoid actuator saturation (Cloutier *et al.* 1996, Mracek and Cloutier 1998). These two features render the SDRE controller a nonlinear controller.

A note worthy aspect of SDRE control which is similar to the MPC is that it runs the algebraic Riccati equation (ARE) solver at every sampled time, implying online optimisation. The SDRE control is able to accommodate for large state errors issue by implicitly assimilating them into the $\mathbf{A}(\mathbf{x})$ and $\mathbf{B}(\mathbf{x})$ matrices thus ensuring the system is able to react favourably against large external disturbance effects. Moreover, the SDRE controller does not not cancel beneficial nonlinearities. In principle, SDRE control possesses respectable robustness characteristic borrowed from the LQR. Better still, SDRE H_∞ control formulation has also been proposed by Cloutier *et al.* (1996). Despite of the aforesaid characteristics, it has been shown that the SDRE regulator is locally asymptotically stable and suboptimal (Banks and Manha 1992, Cloutier *et al.* 1996). Owing to the nonlinear characteristic of SDRE control, global asymptotic stability cannot be guaranteed. This is still an open issue which warrants further research.

On the aspect of optimality or lack thereof, SDRE control obtained is not usually the optimal one that minimises the performance index. The reason for this is because of the non-uniqueness of the parametrisation of $\mathbf{f}(\mathbf{x}) = \mathbf{A}(\mathbf{x})\mathbf{x}$ (Erdem 2001). Although, it is possible for the SDRE control to match the optimal one if a 'suitable' $\mathbf{A}(\mathbf{x})$ is chosen, but the task of finding it can arduous. Giving the apparent benefits of the design flexibility of SDRE control, one can welcome it as a tradeoff to optimality.

6.7 SDRE Formulation

The formulation of SDRE is actually quite straightforward and applicable to a wide range of nonlinear dynamic systems. The problem considered herein, is the infinite-horizon regulation of general autonomous nonlinear system affine in the input. Assuming a nonlinear system that can be expressed as such:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u} \tag{6.36}$$

and a performance index

$$J = \int_0^\infty (\mathbf{x}^T\mathbf{Q}(\mathbf{x})\mathbf{x} + \mathbf{u}^T\mathbf{R}(\mathbf{x})\mathbf{u})dt \tag{6.37}$$

which allows for trading-off state error $\mathbf{x}$ versus control input $\mathbf{u}$, via the weighting matrices $\mathbf{Q}(\mathbf{x}) \geq 0$, $\mathbf{R}(\mathbf{x}) > 0 \forall \mathbf{x}$, respectively, where $\mathbf{x} \in \mathbf{R}^n$, $\mathbf{u} \in \mathbf{R}^m$,$\mathbf{f}(\mathbf{x})$, $\mathbf{g}(\mathbf{x})$, $\mathbf{Q}(\mathbf{x})$ and $\mathbf{R}(\mathbf{x}) \in C^k$, with $k \geq 1$. Assuming that $f(0) = 0$ and $g(\mathbf{x}) \neq 0 \forall \mathbf{x}$, it is desired to find a feedback control law $\mathbf{u}(\mathbf{x})$ which will regulate the system to the origin for all $\mathbf{x}$. It is quite obvious here that, there is a striking similarity between the SDRE and LQR formulation.

First and foremost, the nonlinear system equation, Equation 6.36 must be converted into the 'linear extended' form of

$$\dot{\mathbf{x}} = \mathbf{A}(\mathbf{x})\mathbf{x} + \mathbf{B}(\mathbf{x})\mathbf{u} \tag{6.38}$$

where $\mathbf{f}(\mathbf{x}) = \mathbf{A}(\mathbf{x})\mathbf{x}$, ($\mathbf{A}(\mathbf{x})$ can be non-unique) and $\mathbf{g}(\mathbf{x}) = \mathbf{B}(\mathbf{x})$. The former parametrisation is possible if and only if $\mathbf{f}(0) = 0$ and $\mathbf{f}(\mathbf{x})$ is continuously differentiable. An extended linearisation technique is defined as one in which the nonlinear system equations are factored in a linear like form $\mathbf{A}(\mathbf{x})\mathbf{x} + \mathbf{B}(\mathbf{x})\mathbf{u}$, and linear control techniques are used to obtain a closed-loop system which has a pointwise Hurwitz coefficient matrix (Erdem 2001).

Again, under the condition that the pair ($\mathbf{A}(\mathbf{x})$, $\mathbf{B}(\mathbf{x})$) is pointwise stabilisable, the nonlinear state feedback control law can be constructed as

$$\mathbf{u}(\mathbf{x}) = -\mathbf{K}(\mathbf{x})\mathbf{x} = -\mathbf{R}^{-1}\mathbf{B}^T(\mathbf{x})\mathbf{P}(\mathbf{x})\mathbf{x} \tag{6.39}$$

where $\mathbf{P}(\mathbf{x}) > 0$ is obtained by solving the state-dependent Algebraic Riccati equation

$$\mathbf{A}^T(\mathbf{x})\mathbf{P}(\mathbf{x}) + \mathbf{P}(\mathbf{x})\mathbf{A}(\mathbf{x}) + \mathbf{Q}(\mathbf{x}) - \mathbf{P}(\mathbf{x})\mathbf{B}(\mathbf{x})\mathbf{R}^{-1}(\mathbf{x})\mathbf{B}^T(\mathbf{x})\mathbf{P}(\mathbf{x}) = 0 \qquad (6.40)$$

pointwise at each state $\mathbf{x}$. Nonetheless, finding a proper pointwise stabilisabe $(\mathbf{A}(\mathbf{x}), \mathbf{B}(\mathbf{x}))$ pair, can be nontrivial for higher order systems.

In implementing SDRE control, the most desirable option is to solve the state dependent Riccati equation analytically. This is only possible for lower order systems, with special structure. In generally, however, this is not possible and numerical methods must be adopted, which is rather straightforward with numerical tools such as MATLAB. The computation can be done online or off-line. The off-line approach has lower computational cost and is preferable in risk-avert situations, the solutions are thoroughly analyse to confirm that there are valid before implementation.

In contrast, the online version is computationally more demanding. It is desirable to compute the feedback control in real-time by solving the SDRE at a relatively high sampling rate, compared to the dynamics of the system. Real-time control becomes important when the disturbances or trajectories are not known, or changing from time to time, such as in the case of AUV obstacle avoidance in an unstructured environment. It is imperative that the SDRE must be solved at each time step, as a matter of fact, a sampling interval less than a critical value could lead to instability. The current available computation power is more than adequate to sustain SDRE control for systems which have very fast dynamics as demonstrated by Bogdanov *et al.* (2003).

6.8 Kinematic Based SDRE Controller Formulation

Instead of explicitly dealing with the dynamic model of the AUV in designing the SDRE controller, one uses a kinematics model instead, as advocated by Ren and Beard (2004). One of the main reasons behind this, is that, in practice and for most cases, one does not have the corresponding model dynamics of the vehicle. This prevents model-based controllers from being implemented. A noteworthy aspect of

this implication is that by omitting the need for a dynamic model of the system, one can extend its application to a more encompassing types of vehicles.

Furthermore, most AUVs and UAVs have built-in controllers to stabilise or maintain the velocities. The majority of these controllers are of the PID variety. It will be shown later that by employing this approach, one can actually build a high-level tracking controller on top (outer-loop) of the others. In other words, the SDRE controller will act in supervisory mode while leaving the low-level (velocity augmented) controller intact. In normal operating mode, the high-level controller can be inhibited so that the low-level controllers and the guidance system can function in a conventional matter, however when the need arises, the high-level controller can be activated to tackle complex tasks. A pictorial representation of the unified control system can be found in Fig 6.4.

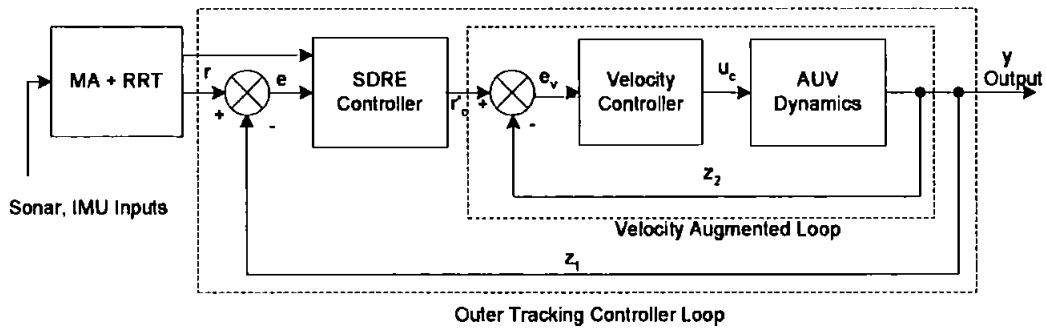


Figure 6.4: Block diagram of the SDRE controller and feedforward unit of the AUV

Note that unlike the LQR system (Fig 6.3), the feedforward unit is not directly fed into the AUV actuator, instead it is 'pre-processed' by the SDRE controller before passing to the AUV actuator. This form of interaction renders the system more robust and stable.

Herein, one commences the formulation of the controller with the assumption that the AUV is equipped with the appropriate velocity controllers as shown in Fig 6.4. These controllers are required for the MA representation. Letting (x, y), ψ and u denote the inertial position, heading angle and velocity of the AUV respectively. The subscripts c and r denote commanding and reference variables. The velocity augmented AUV's

kinematic equations of motion are adequately modelled by

$$\begin{aligned} \dot{x} &= u_c \cos(\psi) \\ \dot{y} &= u_c \sin(\psi) \\ \dot{\psi} &= w_c \end{aligned} \tag{6.41}$$

where w_c and u_c, are the commanded heading rate and velocity to the velocity controllers.

Additionally, the dynamics of the AUV impose the following input constraints

$$\mathcal{U}_1 = \{v_c, w_c | 0 \leq v_{min} \leq v_c \leq v_{max}, -w_{max} \leq w_c \leq w_{max}\} \tag{6.42}$$

The desired trajectory $(x_r, y_r, \psi_r, v_r, w_r)$ generated by the trajectory generator also satisfies Equation 6.41 under the constraints that v_r and w_r are piecewise continuous and satisfy the constraints

$$\begin{aligned} v_{min} + \epsilon_v \leq v_r \leq v_{max} - \epsilon_v \\ w_{min} + \epsilon_w \leq w_r \leq w_{max} - \epsilon_w \end{aligned} \tag{6.43}$$

where ϵ_v and ϵ_w are positive control parameters. In fact, the inclusion of ϵ_* in the constraints is to guarantee that there is sufficient control authority to track the trajectory. In another perspective, as the ϵ_* approach zero, the feasible control set vanishes. This also explains the reason for incurring a virtual bound on the AUV rudder deflection and slew rate in the MA representation, to reserve some control authority for tracking purposes. These input constraints assume cases where no external disturbances acting on the AUV.

One then transforms the tracking errors from the inertial frame to the AUV frame and it can be expressed as

$$\begin{bmatrix} x_e \\ y_e \\ \psi_e \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_y - y \\ \psi_r - \psi \end{bmatrix} \tag{6.44}$$

Accordingly, via algebraic substitution with Equation 6.41, Equation 6.44 and differ-

entiation. The tracking-error model can be represented as

$$\begin{aligned} \dot{x}_e &= w_c y_e - u_c + u_r \cos(\psi_e) \\ \dot{y}_e &= -w_c u_e + u_r \sin(\psi_e) \\ \dot{\psi}_e &= w_r - w_c \end{aligned} \tag{6.45}$$

Following, Equation 6.45 can be simplified as

$$\begin{aligned} \dot{x}_0 &= u_0 \\ \dot{x}_1 &= (w_r - u_0)x_2 + u_r \sin(x_0) \\ \dot{x}_2 &= -(w_r - u_0)x_1 + u_1 \end{aligned} \tag{6.46}$$

where

$$(x_0, x_1, x_2) \equiv (\phi_e, y_e, -x_e) \tag{6.47}$$

and $u_0 \equiv w_r - w^c$ and $u_1 \equiv v^c - v_r \cos(x_0)$. The input constraints under the transformation become

$$\mathcal{U}_2 = \{u_0, u_1 | \underline{w} \leq u_0 \leq \overline{w}, \underline{v} \leq u_1 \leq \overline{v}\} \tag{6.48}$$

where $\underline{w} \equiv w_r - w_{max}, \overline{w} \equiv w_r + w_{max}, \underline{v} \equiv u_{min} - u_r \cos(x_0)$, and $\overline{u} \equiv u_{max} - u_r \cos(x_0)$ are time-varying. These transformed constraints are actually with reference to the medium of the AUV is travelling. Equation 6.44 and 6.47 are invertible transformations, which means that $(x_e, y_e, \psi_e) = (0, 0, 0)$ is equivalent to $(x_e, y_e, \psi_e) = (0, 0, 0)$, or in other words $(x_r, y_r, \psi_r) = (x, y, \psi)$. Therefore, the original tracking control objective is converted to a stabilisation objective. The goal here is to find feasible control inputs u_0 and u_1 to stabilise x_0, x_1, and x_2.

To this end, the system Equation 6.46 can be rewritten as

$$\dot{\mathbf{x}} = \mathbf{A}(t, \mathbf{x})\mathbf{x} + \mathbf{B}(\mathbf{x})\mathbf{u} \tag{6.49}$$

where $\mathbf{x} = [x_0, x_1, x_2]^T$, $\mathbf{u} = [u_0, u_1]^T$,

$$\mathbf{A}(t, \mathbf{x}) = \begin{bmatrix} 0 & 0 & 0 \\ v_r(t)\frac{sin(x_0)}{x_0} & 0 & w_r(t) \\ 0 & w_r(t) & 0 \end{bmatrix} \tag{6.50}$$

and

$$\mathbf{B}(\mathbf{x}) = \begin{bmatrix} 1 & 0 \\ -x_2 & 0 \\ x_1 & 1 \end{bmatrix} \tag{6.51}$$

The pointwise controllability matrix is given by $\mathbf{S}(t,\mathbf{x}) = [\mathbf{B}(x)\ \mathbf{A}(t,\mathbf{x})\ \mathbf{B}(x)\ \mathbf{A}(t,\mathbf{x})^2\mathbf{B}(\mathbf{x})]$. It can be verified that $\mathbf{S}(t,x)$ has full rank when $w_r(t) \neq k\pi, k \in \mathbb{Z} \setminus 0$. As a result, $(\mathbf{A}(t,\mathbf{x}), \mathbf{B}(x))$ is pointwise stabilisable as long as $x_0 \neq k\pi, k \in \mathbb{Z} \setminus 0$. The control objective is to find feasible control inputs v_c and w_c such that $|x_r - x| + |y_r - y| + |\psi_r - \phi| \longrightarrow 0$ as $t \longrightarrow \infty$.

Defining a saturation function as

$$\text{sat}(\alpha, \beta, \gamma) = \begin{cases} \beta, & \alpha < \beta \\ \alpha, & \beta \leq \alpha \leq \gamma \\ \gamma, & \alpha > \gamma \end{cases} \tag{6.52}$$

Since the AUV may be subjected to non-zero mean ocean currents, the control $\mathbf{u}_{SDRE} = [u_a, u_b]^T$ may not satisfy the input constraints. The actual control will be saturated to satisfy the constraints shown in Equation 6.48 according to the following simple projection (Ren and Beard 2004):

$$\begin{aligned} u_0 &= \text{sat}(u_a, \underline{w}, \overline{w}) \\ u_1 &= \text{sat}(u_b, \underline{u}, \overline{u}) \end{aligned} \tag{6.53}$$

To recap, the algorithm of a SDRE control formulation can be summarised as below:

1. Cast the dynamic equation into an 'extended linearised' form.

$$\dot{\mathbf{x}} = \mathbf{A}(\mathbf{x})\mathbf{x} + \mathbf{B}(\mathbf{x})\mathbf{u} \tag{6.54}$$

2. Tune the $\mathbf{Q}(\mathbf{x})$ and $\mathbf{R}(\mathbf{x})$ in the performance index

$$J = \int_0^\infty (\mathbf{x}^T\mathbf{Q}(\mathbf{x})\mathbf{x} + \mathbf{u}^T\mathbf{R}(\mathbf{x})\mathbf{u})dt \tag{6.55}$$

3. Solve the ARE at every time step. This can be achieved in real-time or prior to the controller operation via fast and efficient ARE solvers that are available commercially or freely.

6.9 Simulations and Discussion

For the simulations, the trajectory as highlighted in Subsection 5.6.1 was selected as a candidate trajectory, also, the *AUTOSUB* AUV model was employed. Again, all the conventions herein, follow the one outlined in Chapter 3. The simulations were hosted in MATLAB 6.5 /SIMULINK environment, on a 2.1 GHz Pentium IV machine, with 512 MB of RAM and running Windows XP.

The reference states are the prescribed trajectory consisting of linear and angular displacements and velocities. Accordingly, the actuator outputs, which correspond to the reference trajectory is also included in feedforward system implementation.

The AUV was assumed to be equipped with an IMU, which is capable of supplying velocities and displacements for three-axis, both rectilinear and angular. Conversely, a Doppler velocity sensor and SLAM unit using sonar, could be employed to mitigate the positioning drifting effect. Precise absolute angular displacement can be acquired from a tilt-compensated compass such as the TCM2. Together, these sensors, through a data fusion process, are capable of catering with full-state measurements, a necessary requirement for proper functioning of the controller. However, in reality, the measurements will mainly be corrupted by high frequency noise. In this study, one assumes that the high frequency noise issue has been dealt with.

Proper evaluation parameters must be dictated to assess the veracity of the controllers performance. The appraisal will be based on the following criteria.

1. Convergence (asymptotically to the tracking trajectory)
2. Robustness to large initial errors
3. Robustness in terms of external disturbances
4. Resilience to induce unstable (oscillatory) responses
5. Minimal control effort

Before proceeding into the simulation studies, it is interesting to observe the LQR gains obtained based on the aforementioned trajectory (Fig 6.5). Several of the gains

overlapped giving the impression of insufficient number of gains. The LQR gains were attained by solving the Riccati equation off-line. This explains its lack of robustness in the presence of external disturbances since large disturbances normally violated the model linearity assumption. Different types of trajectories will yield different gains.

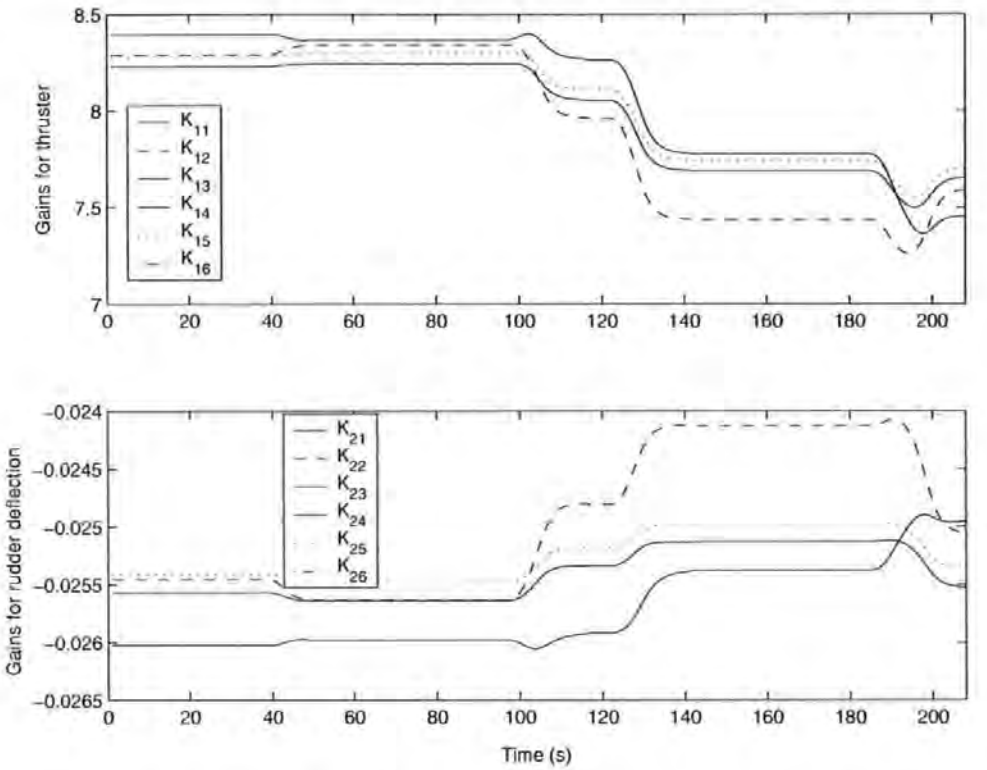


Figure 6.5: Time-varying LQR gains

Whereby, the $\mathbf{Q}(\mathbf{x})$ and the $\mathbf{R}(x)$ of the SDRE controller, and the input constraints as delineated in Equation 6.42 are depicted in Table 6.1. The SDRE controller parameter values were acquired experimentally.

Parameter	Value
$\mathbf{Q}(\mathbf{x})$	$diag[0.01,\ 1,\ 1]$
$\mathbf{R}(x)$	$diag[80000,\ 8000]/\sqrt{5x_0^2 + x_1^2 + x_2^2 + 0.1}$
u_{min}	$1.5\ m/s$
u_{max}	$3\ m/s$
w_{min}	$-0.173\ rad/s$
w_{max}	$0.173\ rad/s$

Table 6.1: Kinematic based SDRE parameters and their corresponding values

6.9.1 Controller performance study (initial errors only)

Fig 6.6 depicts an inertial displacement of the AUV from a simulation run. The initial configuration variables were set to $[-2, \ -2, \ 0.1]$, denoting $[x(m), \ y(m), \ \psi(rad)]$ respectively. The start point of the reference trajectory is set to [1, 1, 0.1]. Notice that employing only the feedforward control law without any tracking controller will result in a trajectory that deviates marginally from the reference trajectory (Fig 6.7(c). Other than the initial condition effect, the clipping of inputs in the MA representation also contributes to the deviation.

Referring to Fig 6.7(a), initially, the LQR controller attempts to track the trajectory, it overshoots and converges again, but later the controller succumbs to instability and diverges from the intended trajectory (Fig 6.7(c)). It is believed that the instability might be caused by the violation of the linear model, instability in the solution eigenvalues or a combination of both. The SDRE controller, on the other hand, shows very good convergence and tracking of the trajectory for the entire length, the slight discrepancy between the end condition is still within an acceptable tolerance in practice (Fig 6.7(a), (b) and (c)). Figure 6.7 (d) shows a $x-y-$ time plot of the trajectories. Attention should be given that in the SDRE controller, the reference trajectory states are supplied to the controller and explicitly processed. On the contrary, the LQR and the feedforward system operate independently without any data association, hence explaining its inferior tracking performance.

Figure 6.8 portrays the distance error relative to the reference trajectory. Indeed, the SDRE controller elicits gradual convergence with a bound of less than 2 m after 20 s, whilst the LQR shows a slight oscillating response which is discernable from the ripples before becoming unstable. There is a sudden jump of distance error near the end for the SDRE controller. This might be caused by the very aggressive turning manoeuvre. Figure 6.9 reveals the heading error of both the controllers. LQR seems to behave rather erratically but the SDRE controller manages to keep the error less than ± 5 ° for the entire length of the trajectory.

The surge velocity time plot of the AUV equipped with the corresponding controllers can be seen in Fig 6.10. Notice that both the controllers exhibit a ramping in the velocity at beginning part of the manoeuvre to reduce the initial displacement errors. At first, the LQR reacts with a very high velocity increase before slowing down

and repeating these processes, a clear sign of excessive gains and instability. The surge command elicits from the SDRE controller, shows very smooth tracking. The difference between the dynamic and kinematic models resulted in slight discrepancies between the commanded and output responses.

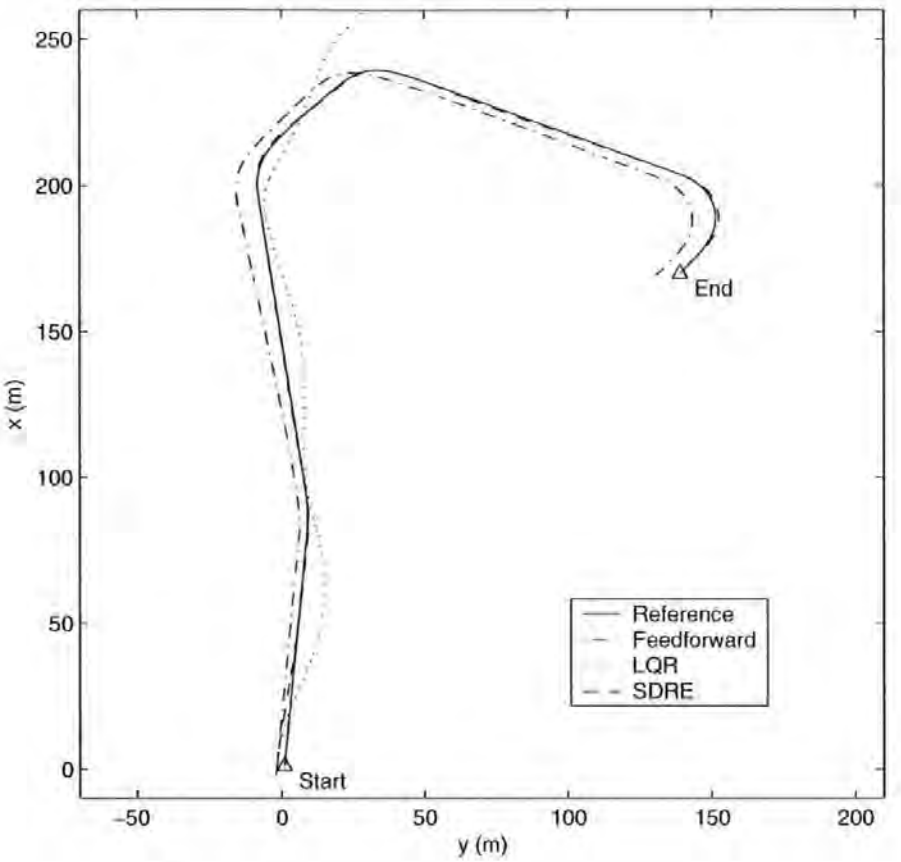


Figure 6.6: x-y position of trajectory plot

The heading rate of the SDRE controller, as shown in Fig 6.11 reveals the tracking of the reference very well. On the contrary, the rather poor LQR performance is perhaps, induced by the very high and low surge velocities. The crossed coupling between the surge velocity and yaw dynamics tends to amplify this negative phenomenon. Similarly, the feedforward system which lacks interaction with the LQR controller also contributes to this effect.

Figure 6.12 shows that the propeller speed response of the LQR case oscillates aggressively even in the regime of near constant reference, a clear sign of instability. The rudder responses, Fig (6.13) shows a very drastic pulse like inputs at the com-

mencement of the manoeuvre for both the LQR and SDRE controller. The SDRE deflection gradually follows the reference whereas the LQR oscillates and saturated at some points.

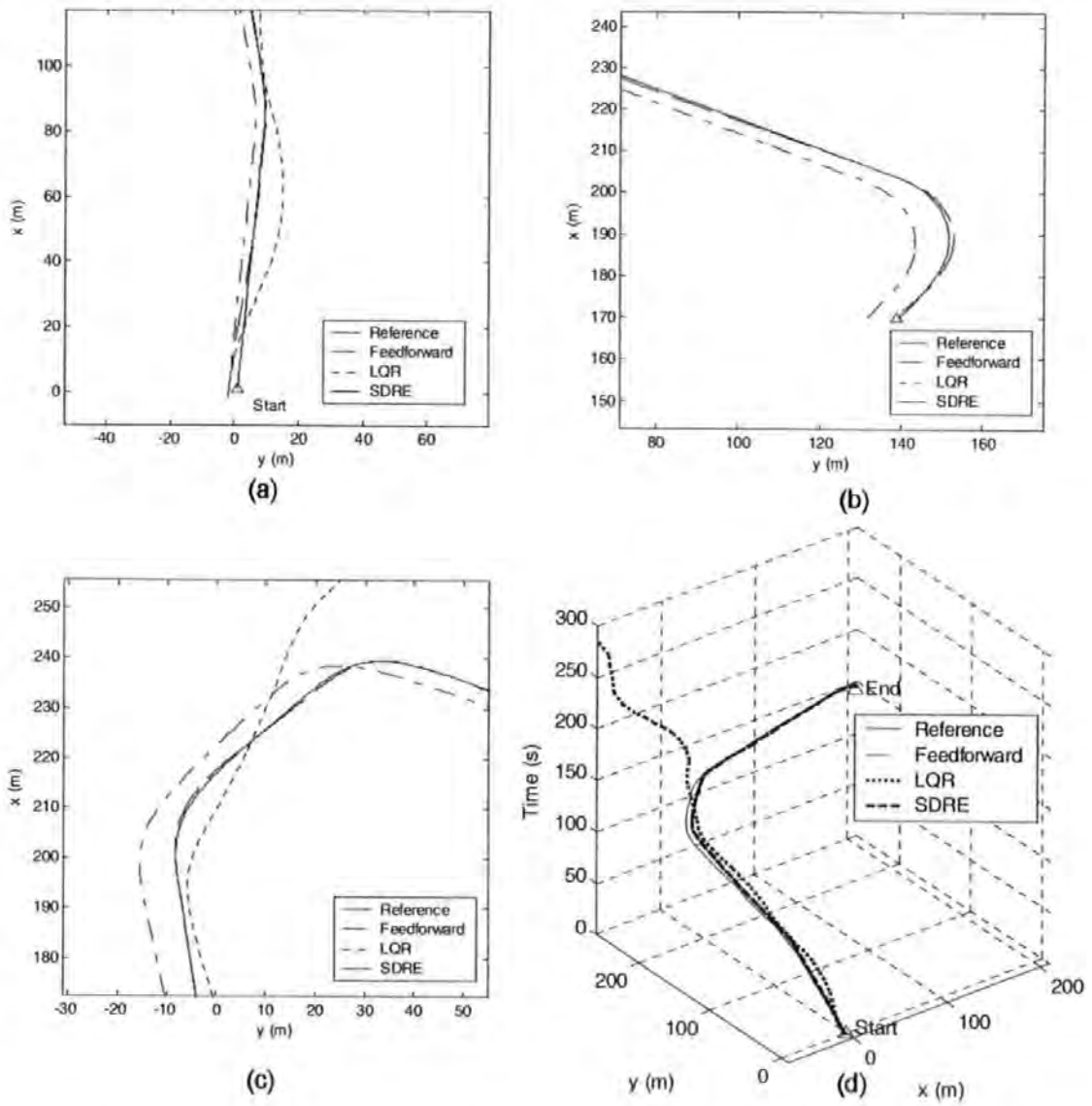


Figure 6.7: a) x-y starting point, b) x-y end point c) Mid trajectory d) x-y-time trajectory plot

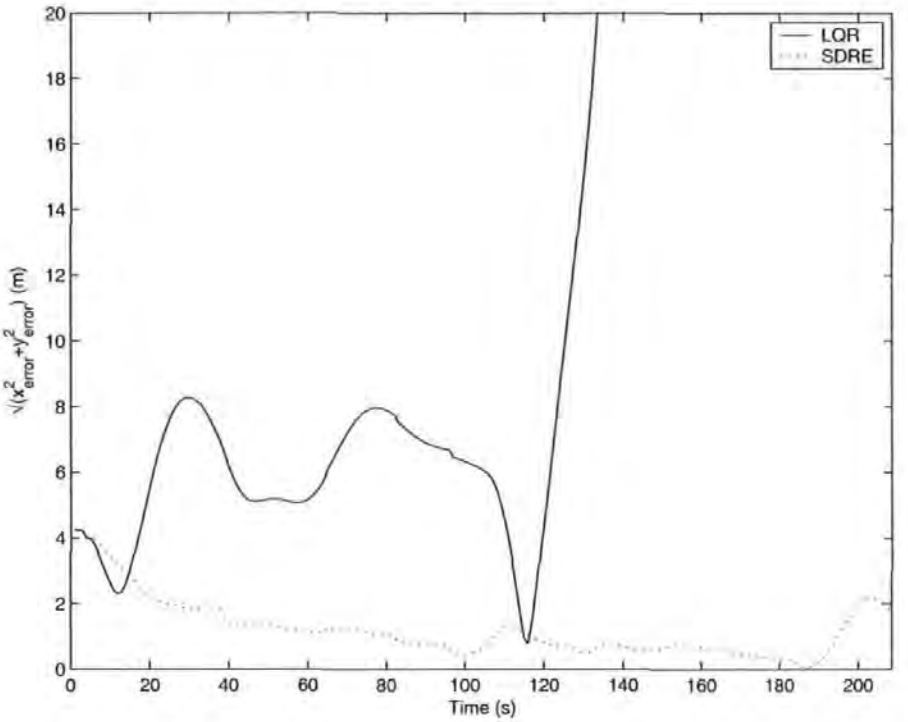


Figure 6.8: Distance error responses

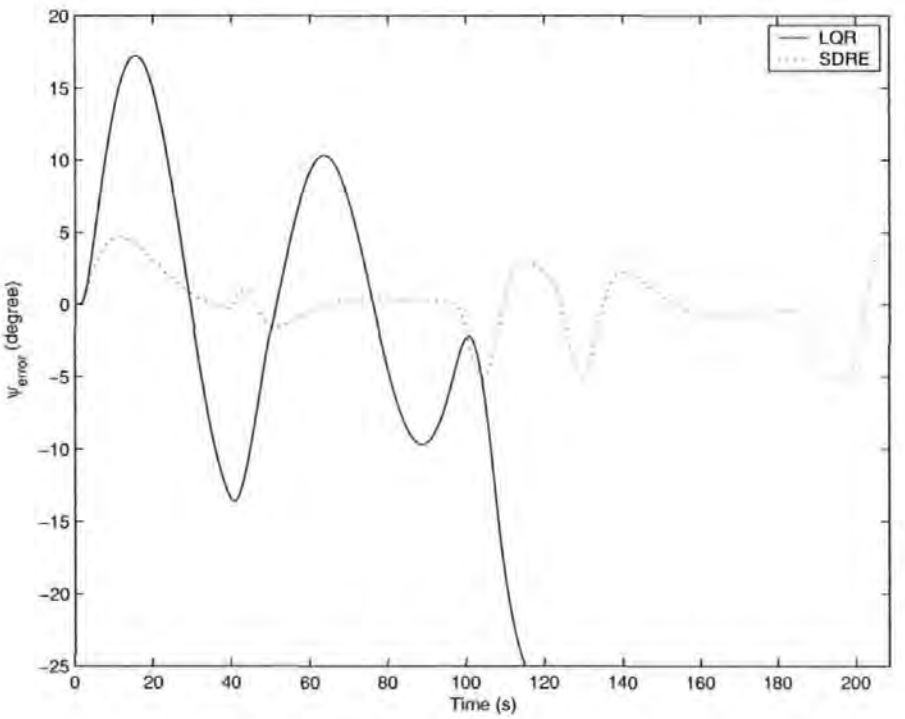


Figure 6.9: Heading error responses

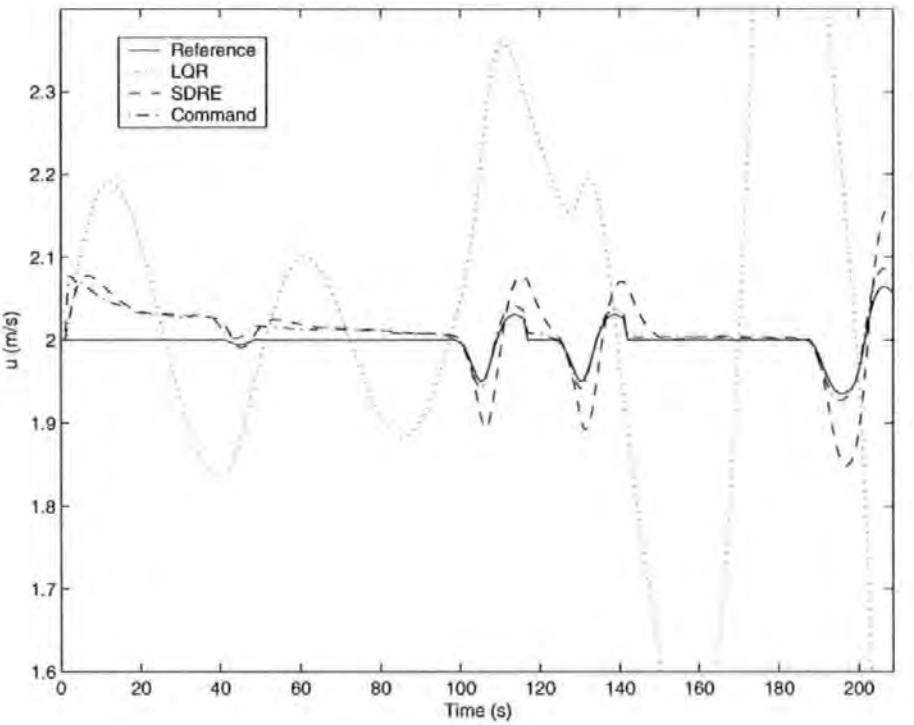


Figure 6.10: Surge responses

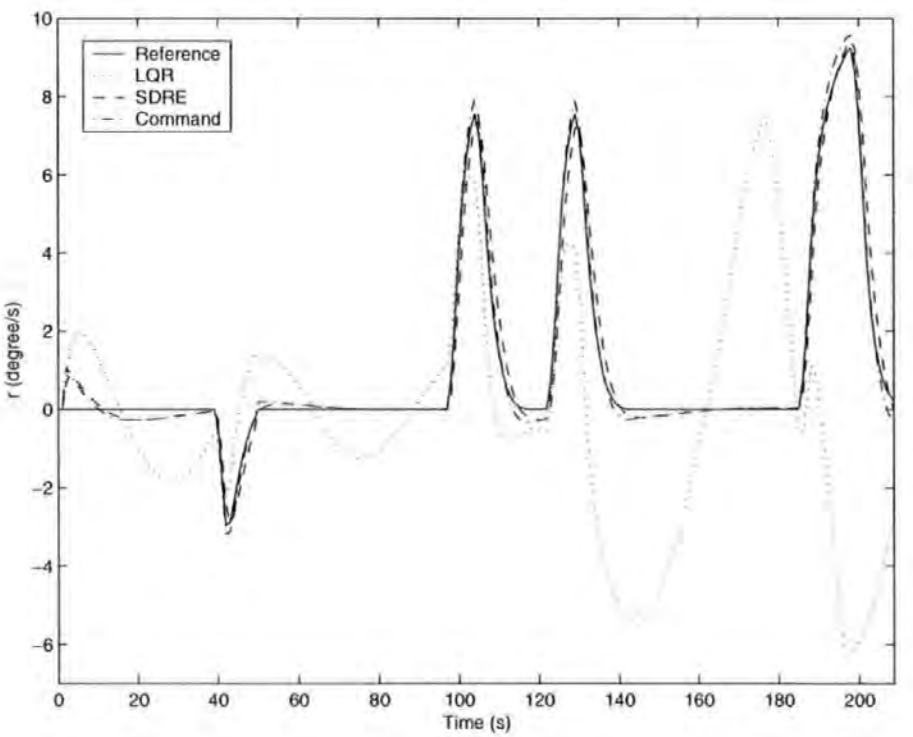


Figure 6.11: Heading rate responses

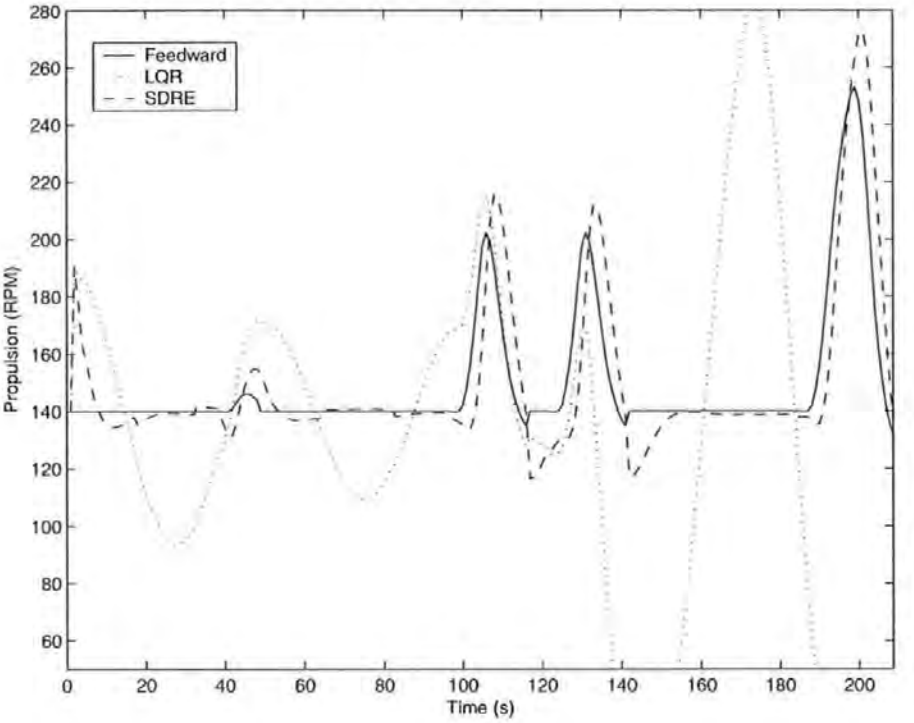


Figure 6.12: Propeller speed responses

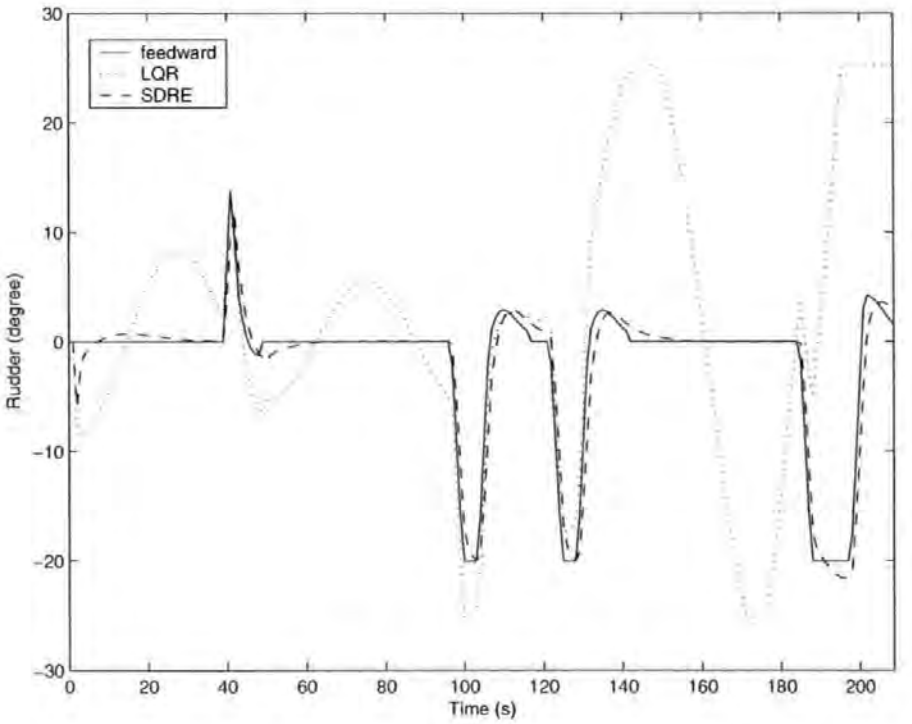


Figure 6.13: Rudder deflection responses

6.9.2 Controller performance study (initial errors and constant current)

In this particular simulation study, a more challenging scenario is envisaged, herein the initial configuration variables of the AUV and reference states are still considered the same as the former but with the introduction of nonzero mean ocean current of magnitude 0.42 m/s average and acting at 60° to the AUV with reference to the NED world coordinate system. The current time plot history is shown in Fig 6.14. Referring to Subsection 3.3.1, it can be seen that this is a realistic representative of underwater current encountered by an AUV.

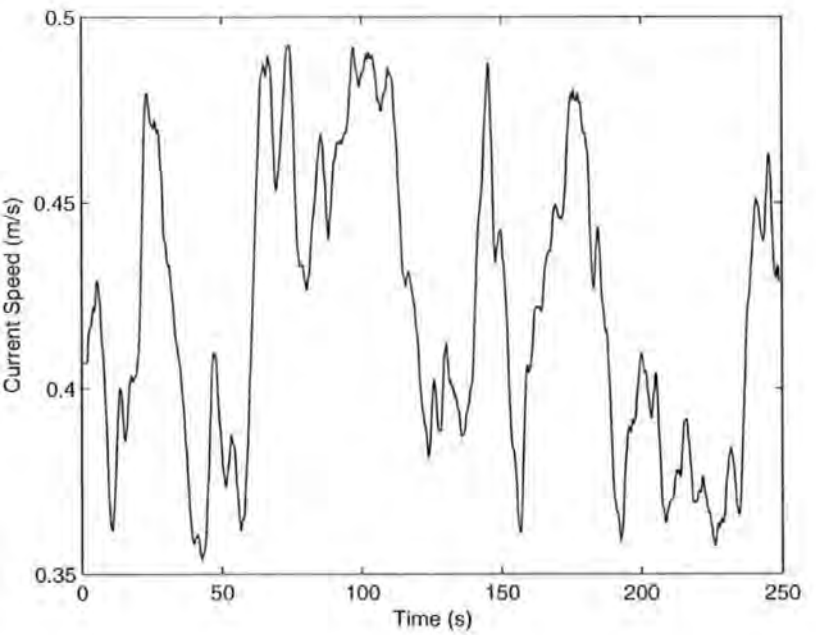


Figure 6.14: A plot of nonzero mean current time history

This time, due to the external disturbance, employing only the feedforward control law without any tracking controller results in a very different trajectory, as exemplified in Fig 6.15. Worse still, the final states are shifted far from the goal (Fig 6.16(b)). Again, referring to Fig 6.15, the LQR seems to have failed miserably in the tracking, diverging from the trajectory after the first turning (Fig 6.16(a)). From Fig 6.15 and 6.16, it is indisputable that the SDRE performance is rather impressive, with only a marginal offset from the prescribed trajectory. Figure 6.16(d) shows a $x-y-$time

plot of the trajectories.

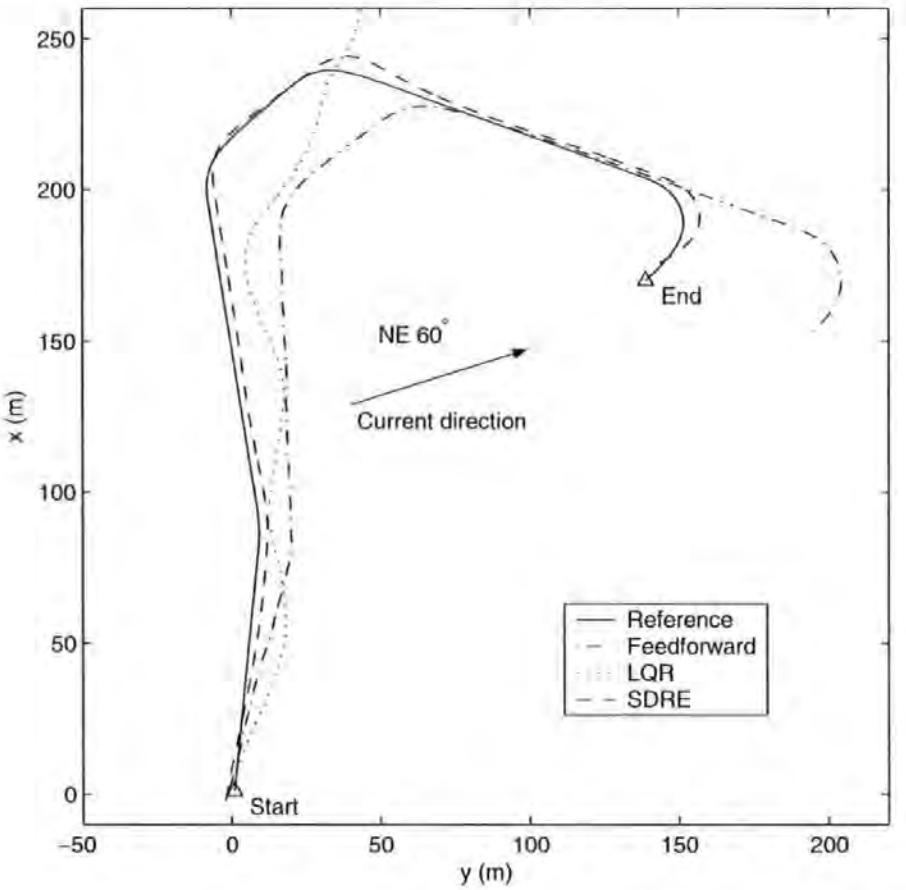


Figure 6.15: x-y position of trajectory plot

Figure 6.17 depicts the distance error relative to the reference trajectory. The SDRE controller settles at an approximately 3 m distance offset. A slight increase in distance error can be detected when the AUV is conducting aggressive turnings. The LQR distance error converges and diverges. It must be remembered that a state-feedback controller will not introduce an integrator into the plant, hence it is not able to eliminate steady state errors.

Integrating action must be explicitly introduced to increase the plant type. In essence, one can consider the state-feedback controller as a linear combination of proportional controllers. If the plant is inherently type one or above, then an integrator is not required. Fortunately, for most autonomous underwater vehicles, heading and depth dynamics are considered to be type 1 model. This is caused by the effect of rudder or hydroplane, where a fixed deflection will generate a constant force that attempts

to rotate the AUV. Nonetheless, the AUV is still susceptible to longitudinal and latitudinal offsets when acted by nonzero mean currents.

An alternative method of mitigating this offset is to employ an ocean current or waves observer (Torsetnes 2004). The effect of the current on the trajectory can then be explicitly taken into account in the optimisation process. Most of the time, these nonzero mean currents can be exploited such that it assist in overall energy saving of the AUV. Nonetheless, this technique will not function well when the currents are unpredictable.

One can observe that there is a large surge velocity deviation for the LQR case in Fig 6.19. The unstable surge dynamics induces an erratic behaviour in the yaw dynamics as illustrated in Fig 6.18 due to the cross-coupling effect. The SDRE controller, on the other hand, manages to keep the error in bound. Notice that the commanded velocity is frequently lower than the measured, this is caused by the relatively small velocity increase due to the current components acting on the AUV. The commanded velocity is actually relative to the water medium. Again, Fig 6.21 and Fig 6.22 depict the unstable propeller speed and rudder outputs.

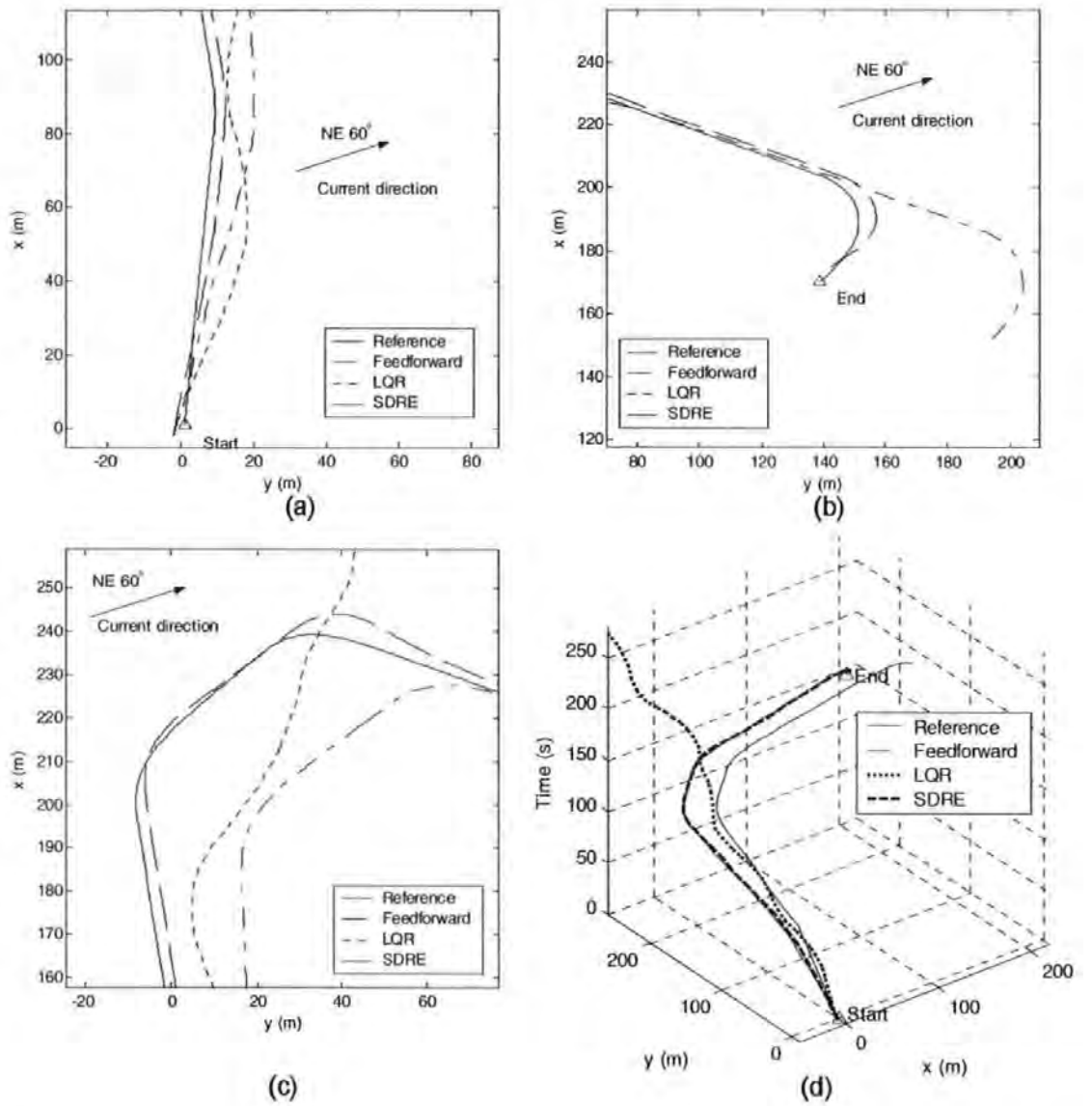


Figure 6.16: a) x-y starting point, b) x-y end point c) mid trajectory d) x-y-time trajectory plot

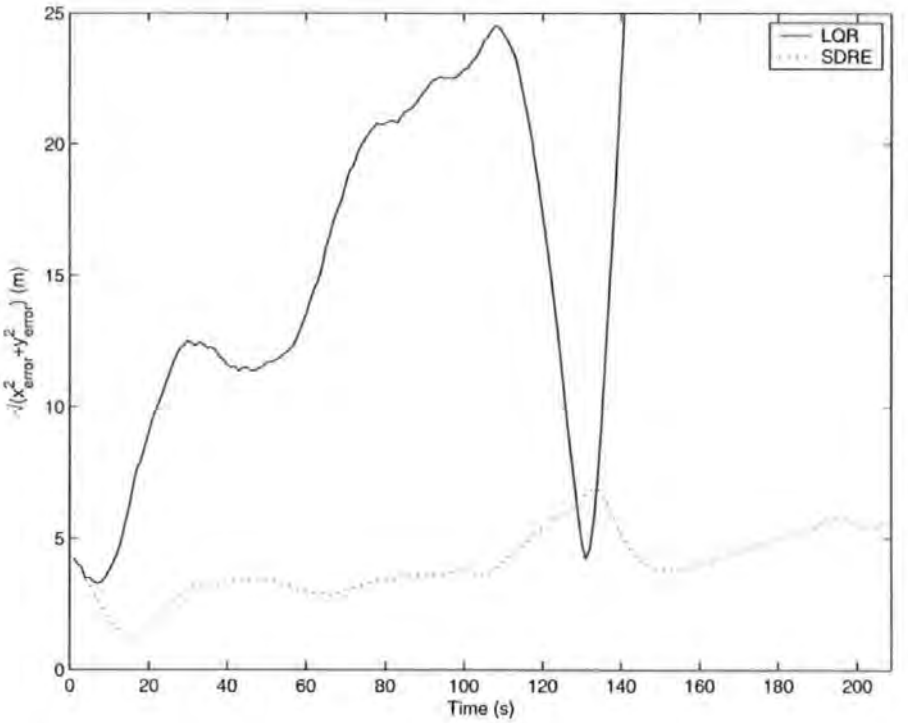


Figure 6.17: Distance error responses

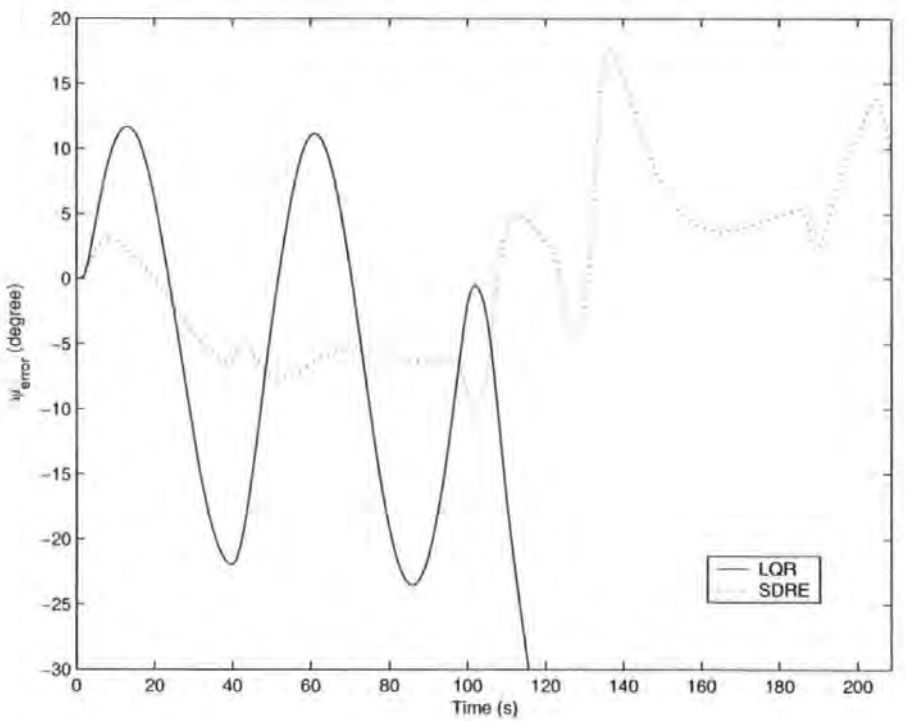


Figure 6.18: Heading error responses

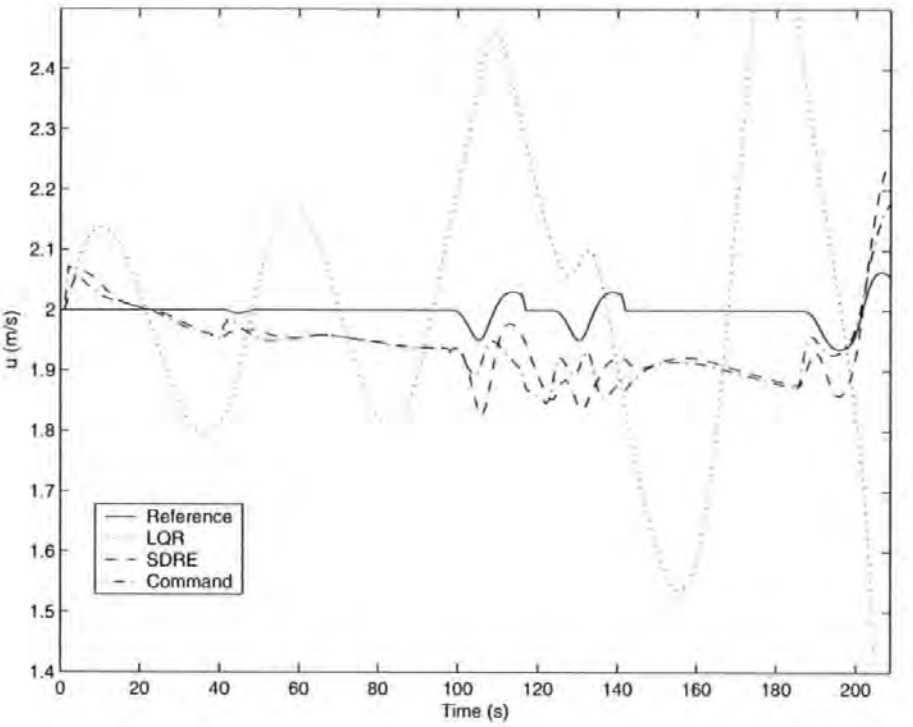


Figure 6.19: Surge responses

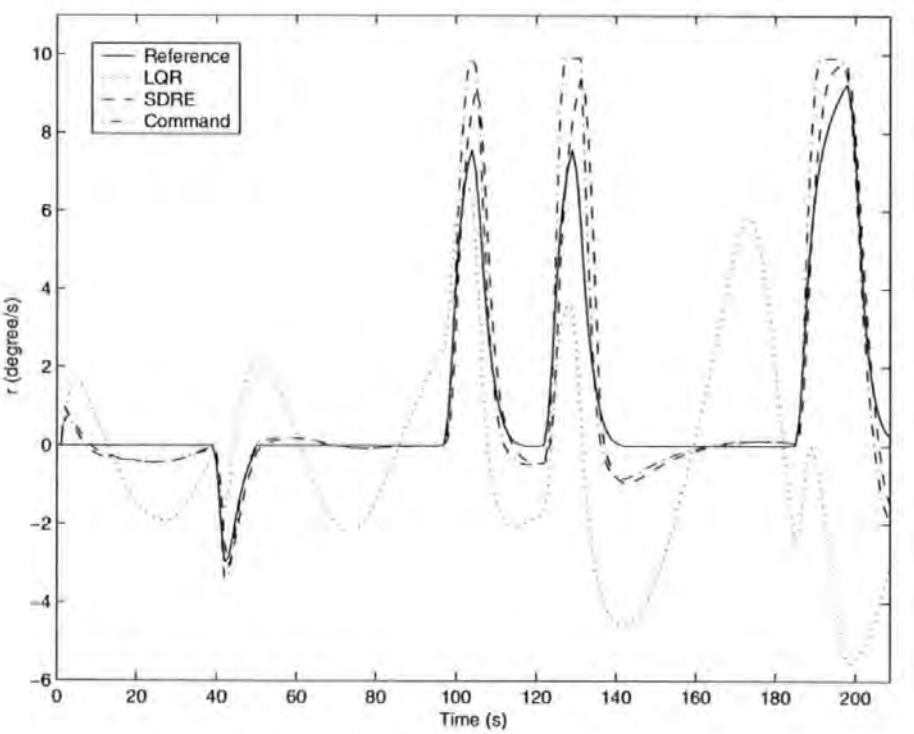


Figure 6.20: Heading rate responses

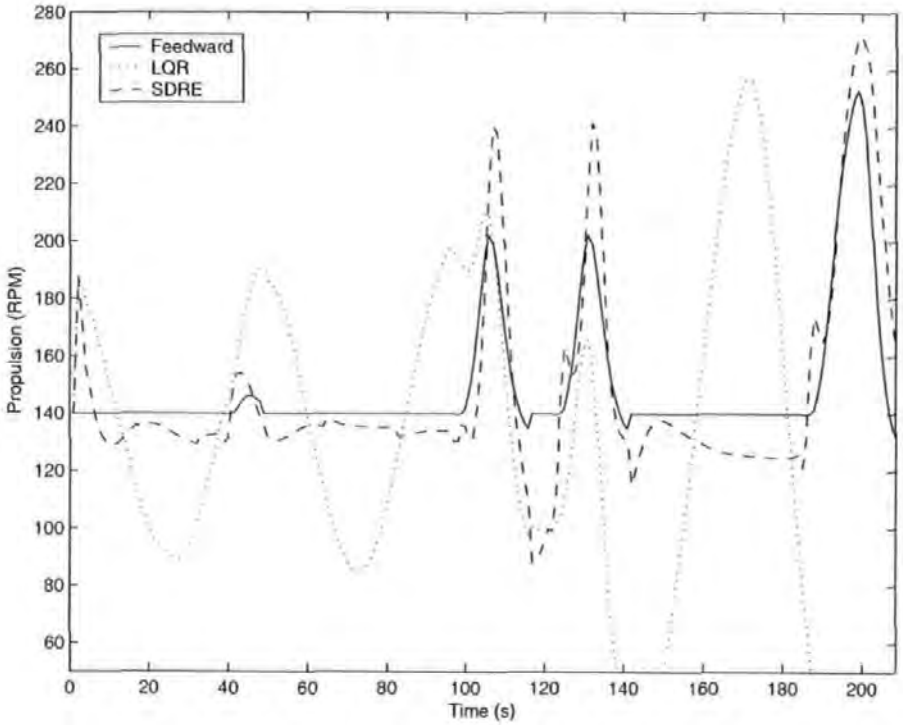


Figure 6.21: Propeller speed responses

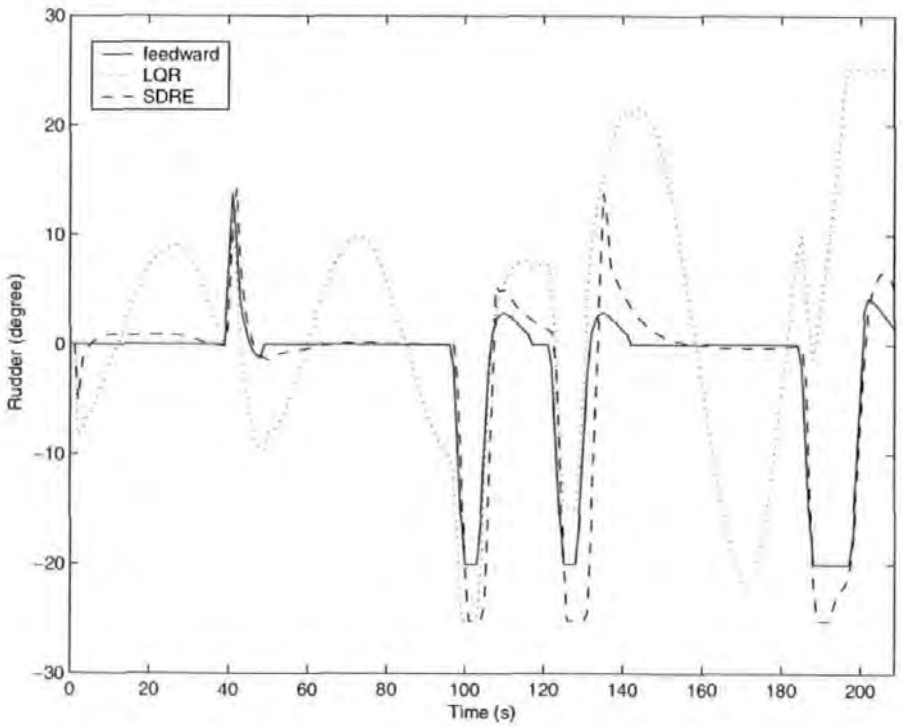


Figure 6.22: Rudder deflection responses

6.9.3 SDRE Controller performance study (various initial errors)

As a result of the rather impressive performance using the SDRE approach which was highlighted by the previous simulations, this controller deserves further inspection. Therefore, it is prudent if one concentrates on the performance evaluation of the SDRE controller whilst neglecting the LQR due to its rather disappointing tracking performance. Subsequently, five scenarios where each has different initial conditions are proposed. All of them include the identical nonzero mean current disturbance employed in the former simulations. Their initial configuration variables with specific alphabet denoting each particular case are given as follows:

- $a \equiv [-15, \ -20, \ -0.2]$
- $b \equiv [-40, \ 5, \ 0]$
- $c \equiv [-10, \ 20, \ 0.4]$
- $d \equiv [2, \ 15, \ 0.3]$
- $e \equiv [15, \ -3, \ 0.1]$

The initial conditions were deliberately selected to demonstrate the capabilities of the controller. Cases a, c and d were chosen such that the initial headings of the AUV diverged from the reference trajectory (Fig 6.23). In contrast, case b has an excessively large x-y displacement error. Finally, case e was selected such that the initial configuration variables significantly precede the desired trajectory. These cases are depicted in Fig 6.23 and 6.24(a). In Fig 6.24 (b), one observes that there is a slight offset from the of the tracks and endpoint, just within an acceptable tolerance. Slight overshoots are discernible from Figure 6.24(c), especially when the AUV is executing turning manoeuvres, but the external disturbances also exaggerated this effect. The displacement-time plot is shown in Fig 6.24(d).

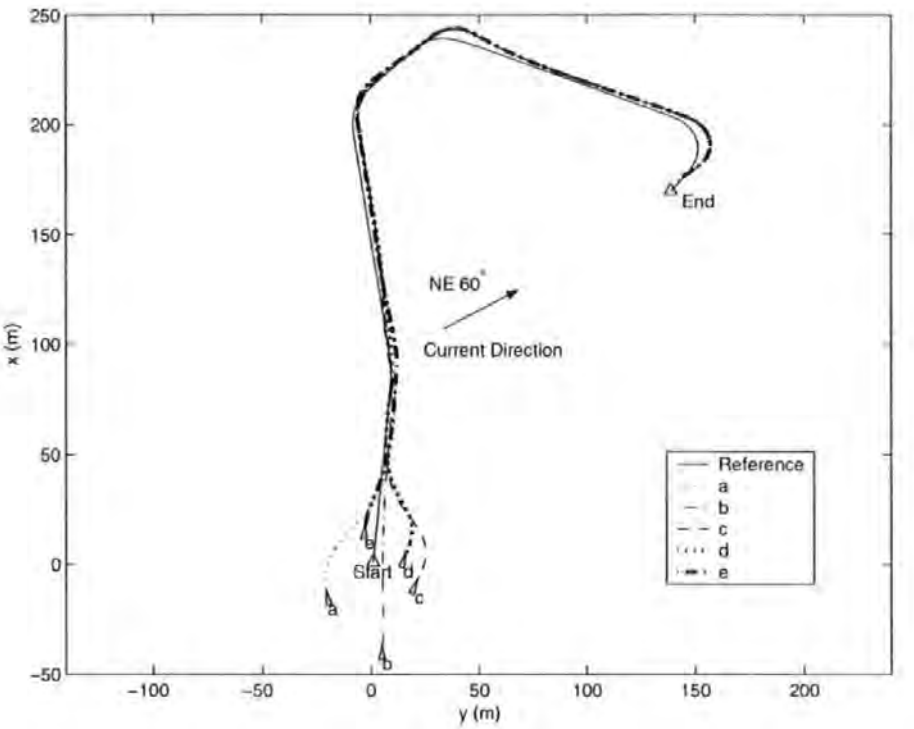


Figure 6.23: x-y position of trajectory plot

Evidently, all the cases show proper convergence and tracking of the prescribed trajectory as supported by Fig 6.25 and Fig 6.26. The convergence to below $10m$ error is less than 20 s for case e and d whereas case a and c, attained the specification in approximately 30 s. In spite of the very large initial displacement errors, case b takes only 40 s to achieve a similar convergence. Large heading errors for all cases except case b, were recorded in the beginning of the tracking manoeuvres. The heading error transients are displayed in Fig 6.26. All of the heading errors converged to within $\pm 10^{\circ}$ in less than 30 s.

Figure 6.27 shows the surge history of different cases. Attention should be given to cases e and d, where the AUV initial positions are such that they precede the prescribed trajectory. Notice that the controller attempts to slow down the vehicle such that the trajectory will catch up with the AUV. In the contrary, for the rest of the cases that have their positions behind the trajectory require some catching up, which leads to an initial increase in surge velocity. There is an evidence of steady state error in the surge velocity because of the ocean current effect. Similarly, Fig 6.28 shows that the heading rate responses follow the reference adequately after the

preliminary transients. There is no sign of saturation in the heading rate responses.

The propeller speed responses (Fig 6.29) highlighted one case of upper saturation in the early part of the manoeuvre. On the other hand, the rudder deflection responses (Fig 6.30 shows a few cases of saturation at 110 s, 130 s, and 190 s. Here one sees that reserve control authority is being expanded to track the trajectory. The nonlinear characteristic of the SDRE controller is more robust to saturation effects.

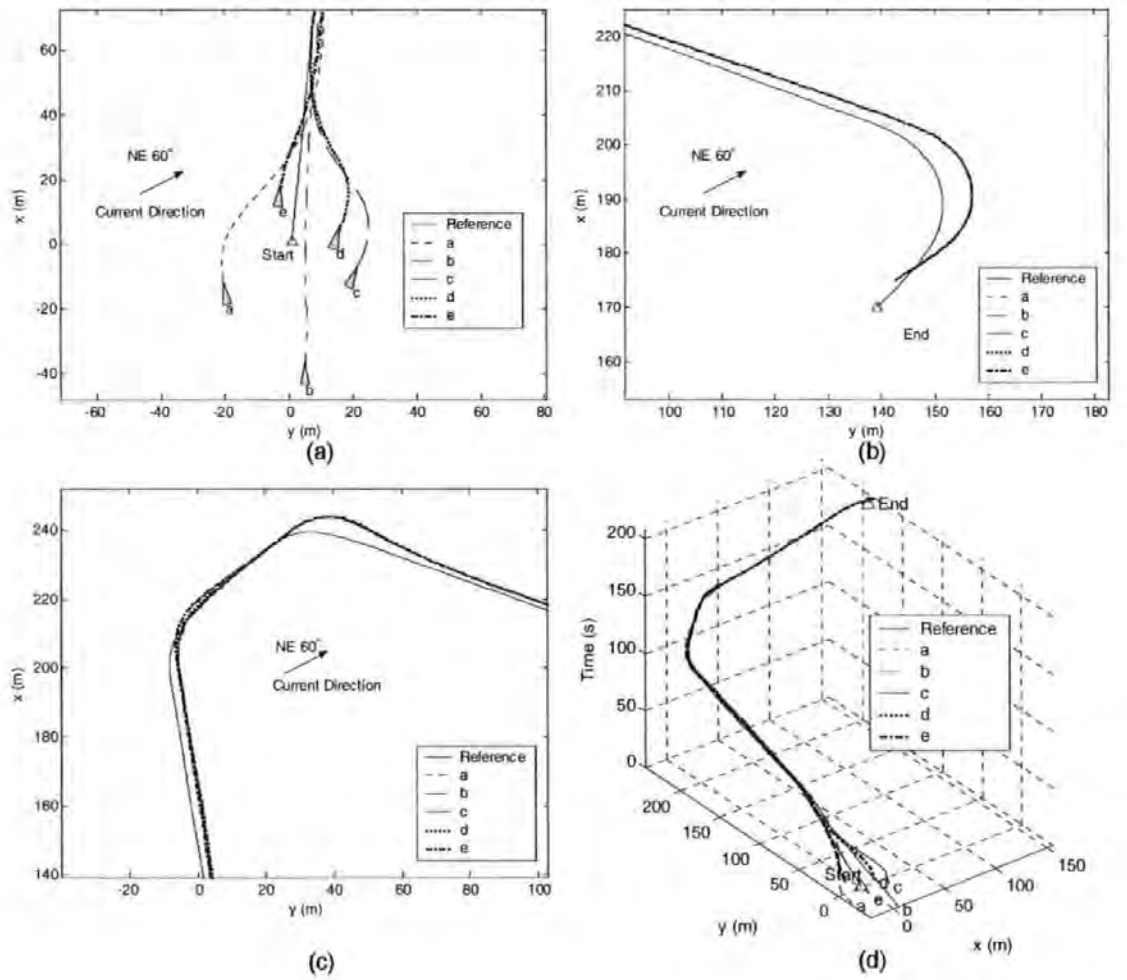


Figure 6.24: a) x-y starting point, b) x-y end point c) mid trajectory d) x-y-time trajectory plot

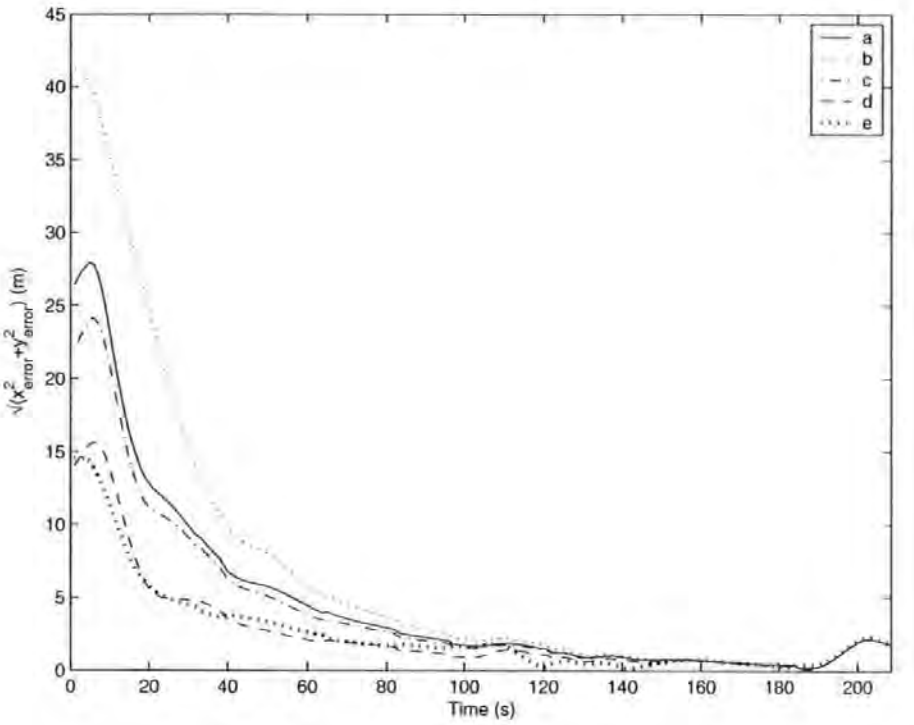


Figure 6.25: Distance error responses

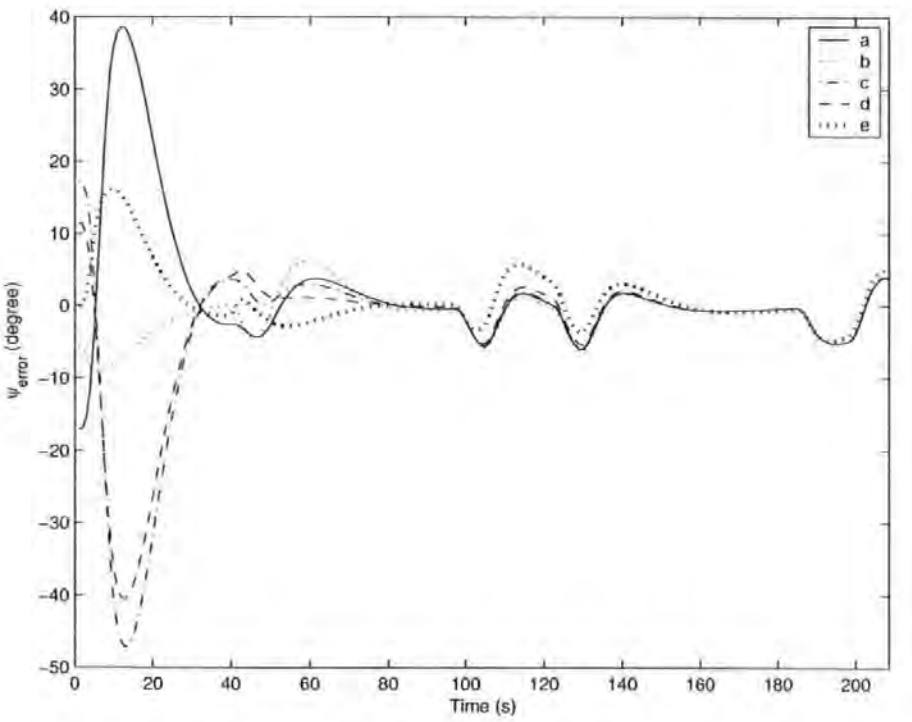


Figure 6.26: Heading error responses

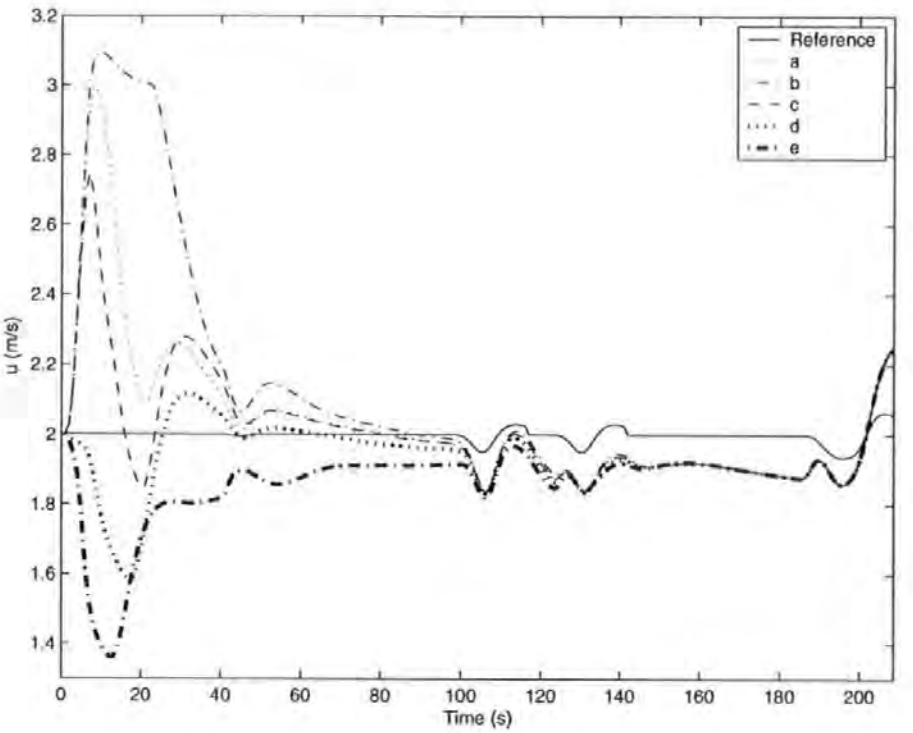


Figure 6.27: Surge responses

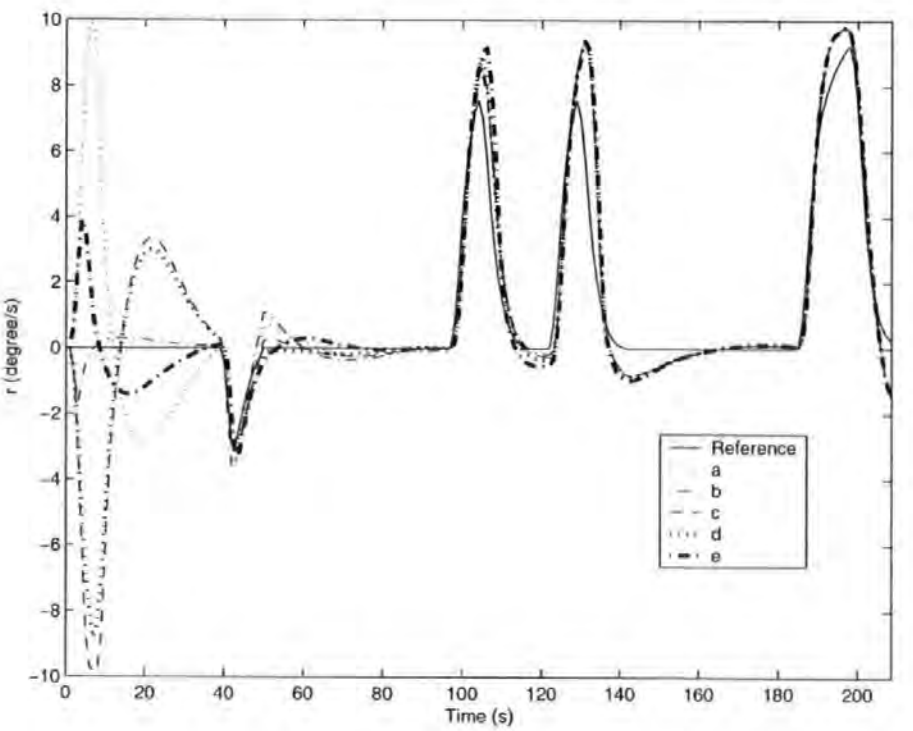


Figure 6.28: Heading rate responses

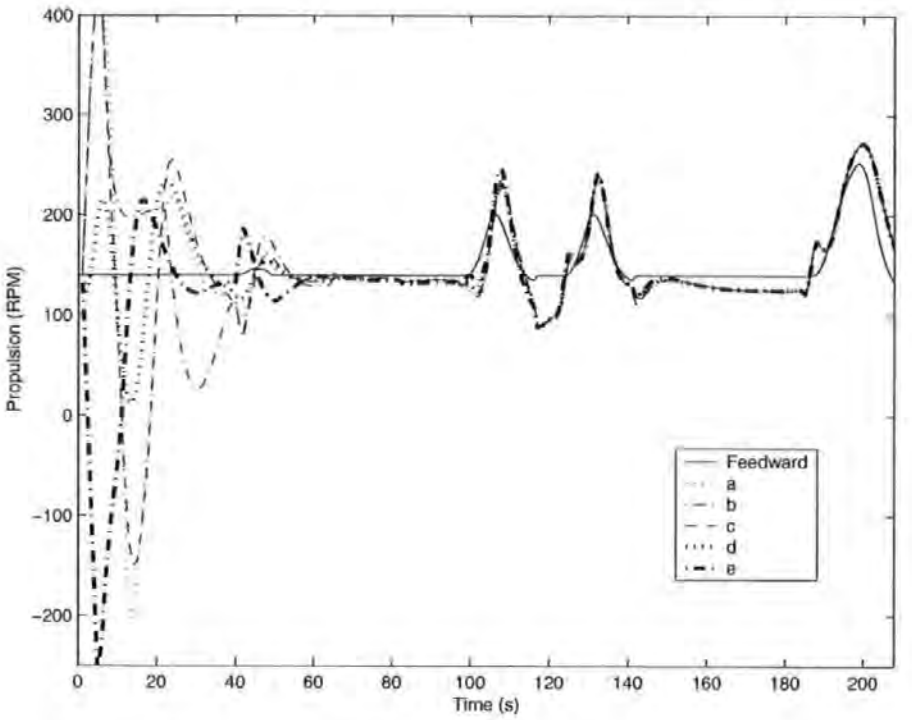


Figure 6.29: Propeller speed responses

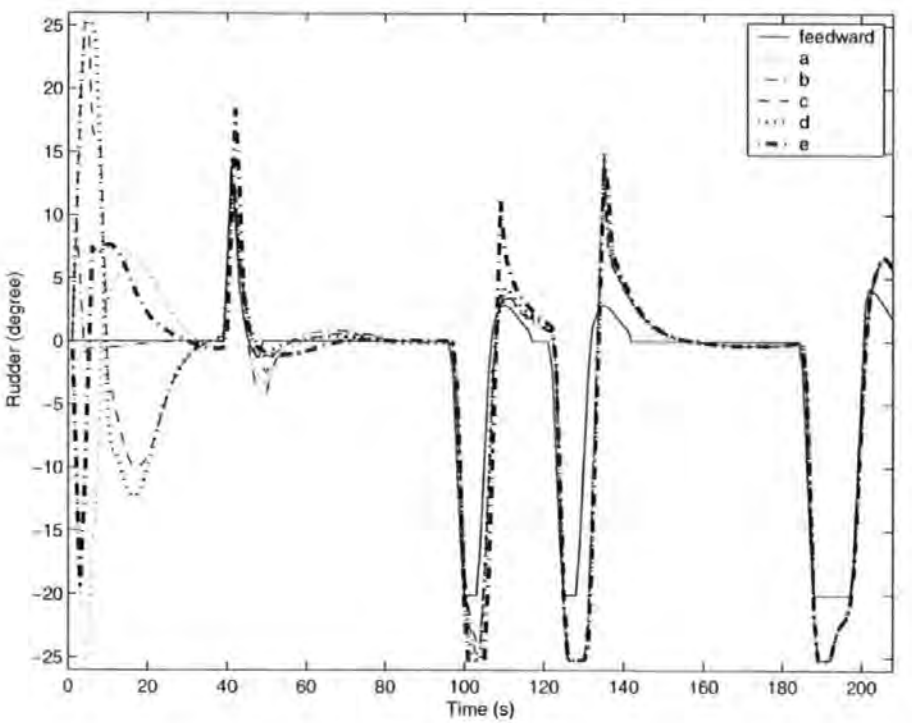


Figure 6.30: Rudder deflection responses)

6.9.4 SDRE Controllers performance comparisons (different weighting matrices)

It is true that different weighting matrices will have a significant impact on the behaviour of the SDRE controller. An incorrectly tuned controller will not function properly, worse still, it can induce instability to the feedback system. Indeed, there is great flexibility in tuning the parameters $\mathbf{Q}(\mathbf{x})$ and $\mathbf{R}(\mathbf{x})$ in SDRE control. As a benefit, the kinematic based SDRE controller contains only a few tuning parameters, which ensure simple tuning.

In these simulations, the initial conditions, current and trajectory are the same as the one outlined in Subsection 6.9.1. Additionally, four different form of weighting matrices were used and their performance examined. Herein, the $\mathbf{Q}(\mathbf{x})$ is considered a constant matrix $diag[0.01\ 1\ 1]$ while $\mathbf{R}(\mathbf{x})$ is made variable, and defined as:

- $a \equiv R \equiv diag[80000\ 8000]/\sqrt{5x_0^2 + x_1^2 + x_2^2 + 0.1}$
- $b \equiv R \equiv diag[800\ 800]/\sqrt{5x_0^2 + x_1^2 + x_2^2 + 0.1}$
- $c \equiv R \equiv diag[80000\ 8000]$
- $d \equiv R \equiv diag[80000\ 8000]/\sqrt{5|x_0| + |x_1| + |x_2| + 0.1}$

The $\mathbf{R}(\mathbf{x})$ is defined such that the weighting matrix is divided by the state norm. In other words, the bigger the state norm, the smaller the $\mathbf{R}(\mathbf{x})$, leading to a more responsive system behaviour at the expense of large control efforts. Conversely, with small state errors, yields loose regulation, with small control effort, rendering the system more sluggish but energy efficient. Notice that there is an additional constant in the state norm, this term is essential to prevent the $\mathbf{R}(\mathbf{x})$ becoming ∞ where all the states converge to zero.

a uses a state dependent $\mathbf{R}(\mathbf{x})$ matrix, with quadratic state norm. One can expect the controller to react very quickly to large initial errors, but slowly to the small ones. b is a case of inexpensive control, uses a very low values of $\mathbf{R}(\mathbf{x})$, A very agile closed-loop system is expected for this case, if it does not destabilise. c utilises only a fixed $R(x)$ matrix, it is used for performance comparisons. d uses an $\mathbf{R}(\mathbf{x})$ similar to

a but with state Manhattan state norm. This norm grows slower than the quadratic norm.

From Fig 6.31, one can deduce the responses of the AUV by looking at the shape of the resultant trajectories. Each trajectory corresponds to a particular case as denoted by the legend. Generally, all of the responses were acceptable except b. In case b, the AUV performs a very forceful turn to reduce the deviation at the very start of the manoeuvre (Fig 6.32(a)). Nonetheless, later, during other turning manoeuvres, it exhibits a tendency to be unstable as evidenced by the sinusoidal shape trajectory (Fig 6.32(b) and (c)). This is the case of to much emphasises on performance. In a different perspective, the displacement-time plot is provided in Fig 6.32(d) for visualisation.

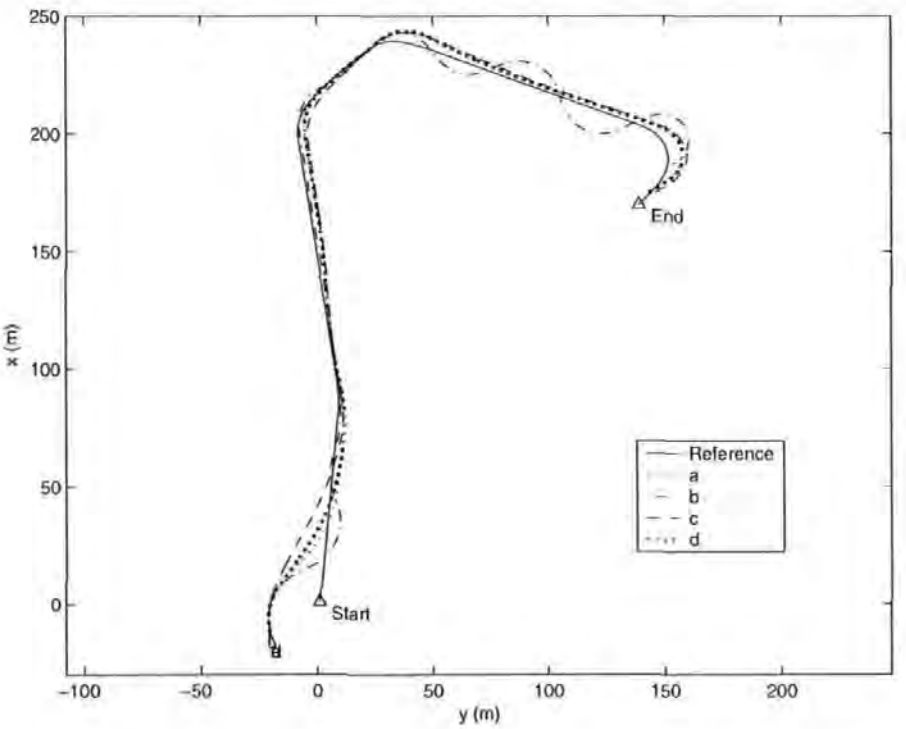


Figure 6.31: x-y position of trajectory plot

Figure 6.33 indicates that the distance error of b converges the fastest but unfortunately becomes unstable later. a and d have very close convergence rate at first until 110 s, after that d distance error starts increasing and matching c distance error. a distance error is superior to the others after 110 s. In the heading error aspect, as displayed in Fig 6.34, case b is clearly unstable. On the other hand, case b and c suffer some large errors ($\pm 15°$) whereas a shows a better than ($\pm 10°$) error.

From Fig 6.35, one can deduce that b the surge dynamics has become unstable after 120 s. a displays a rather aggressive propeller speed increase discernable from the saw tooth response in the beginning of the response (Fig 6.37). It, however, settles down quickly, after 20 s. There is no doubt that the quadratic norm contributes to this characteristic. The rest of the cases, show a more desirable, energy friendly response. It would be an interesting idea to tune the weights such that the AUV will have a mixture of a and c behaviours. This, however, will be reserved for future research. The heading rate, Fig 6.36 and the rudder response (Fig 6.38) convey that b is oscillating erratically. The rest of the cases, manages to track the references adequately. These examples illustrates how the design flexibility offered by SDRE control can be use to improve the classical state versus input trade-off that is encountered with other control methods.

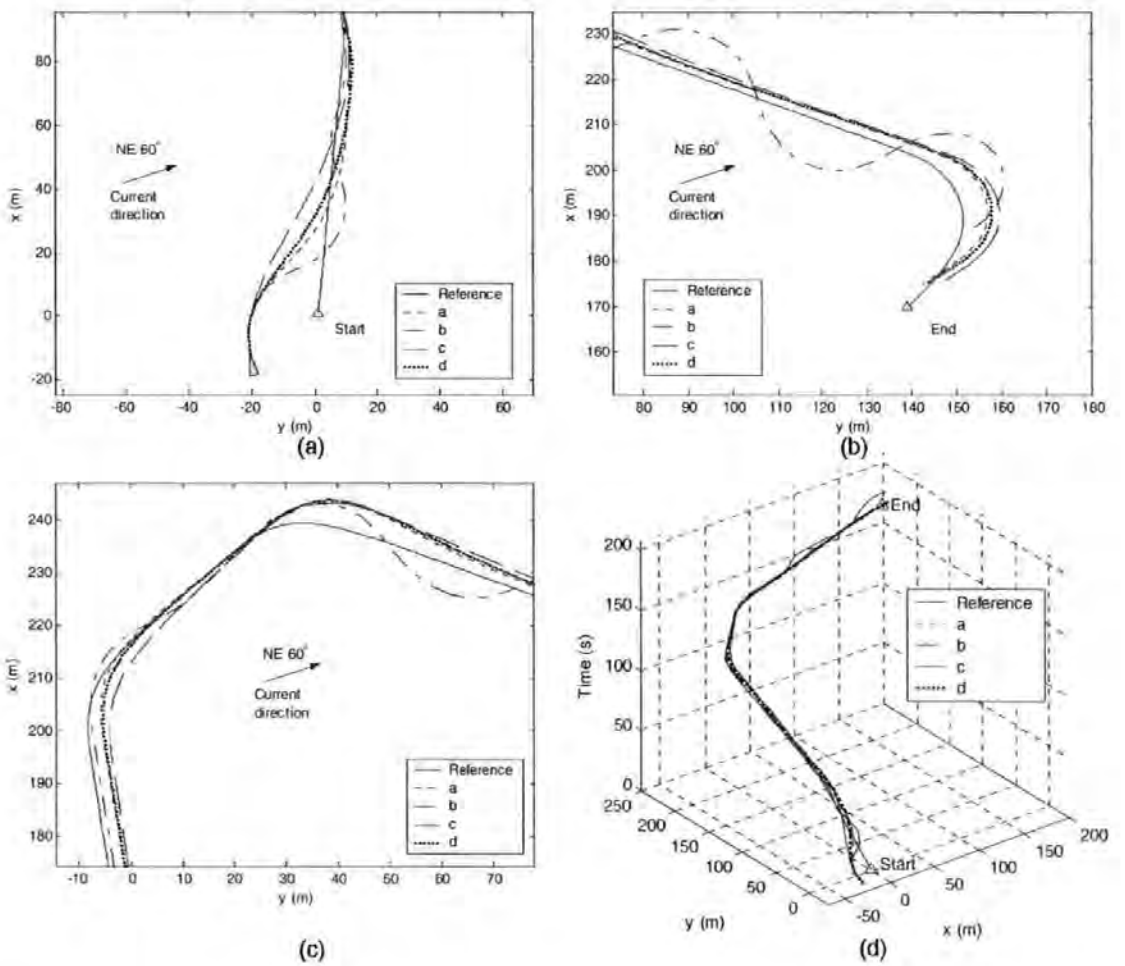


Figure 6.32: a) x-y starting point, b) x-y end point c) mid trajectory d) x-y-time trajectory plot

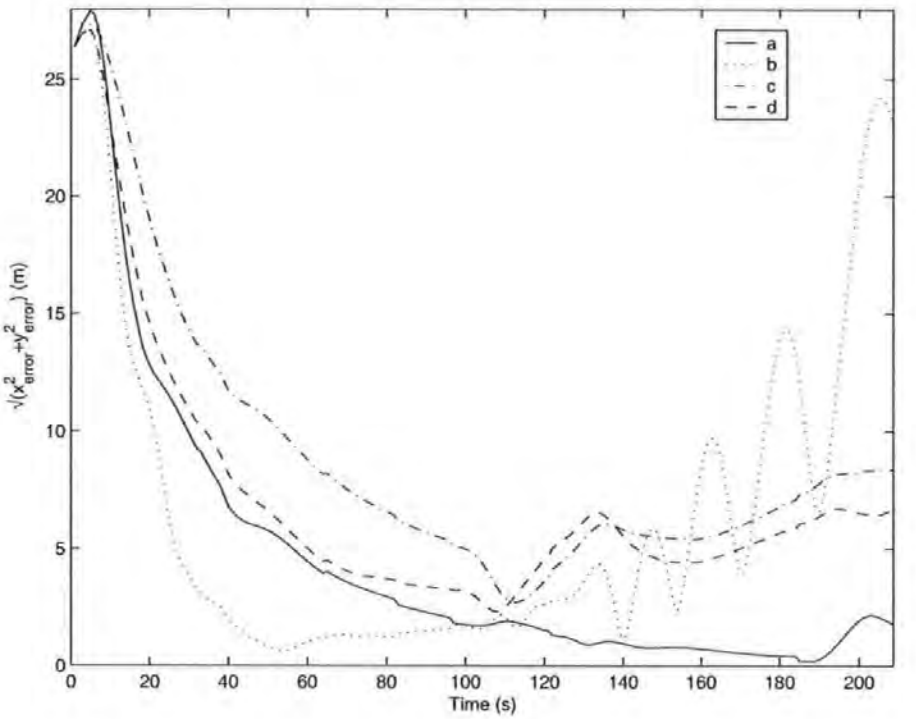


Figure 6.33: Distance error responses

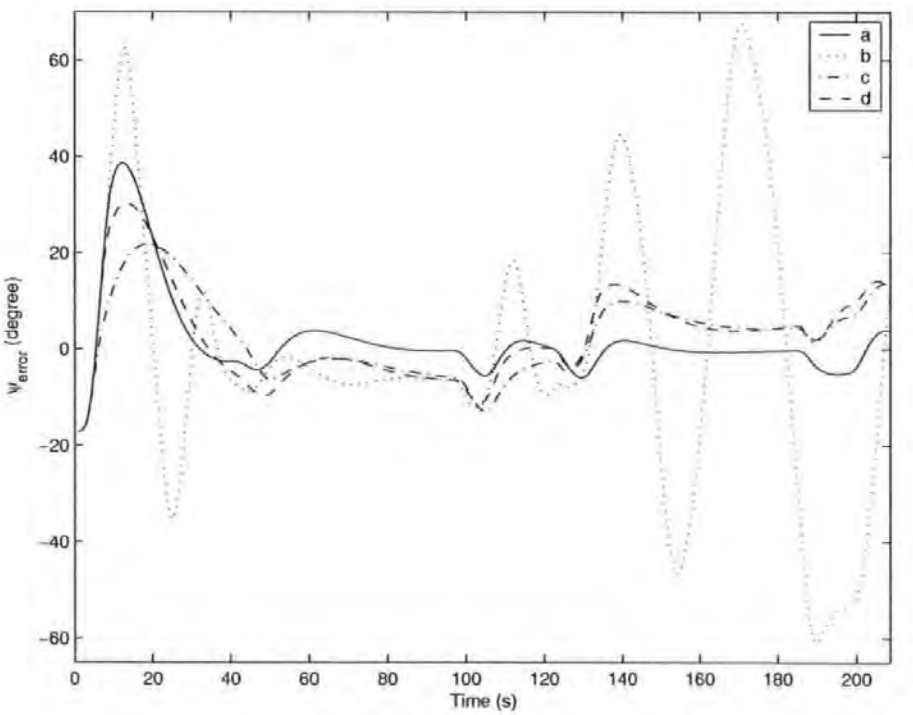


Figure 6.34: Heading error responses

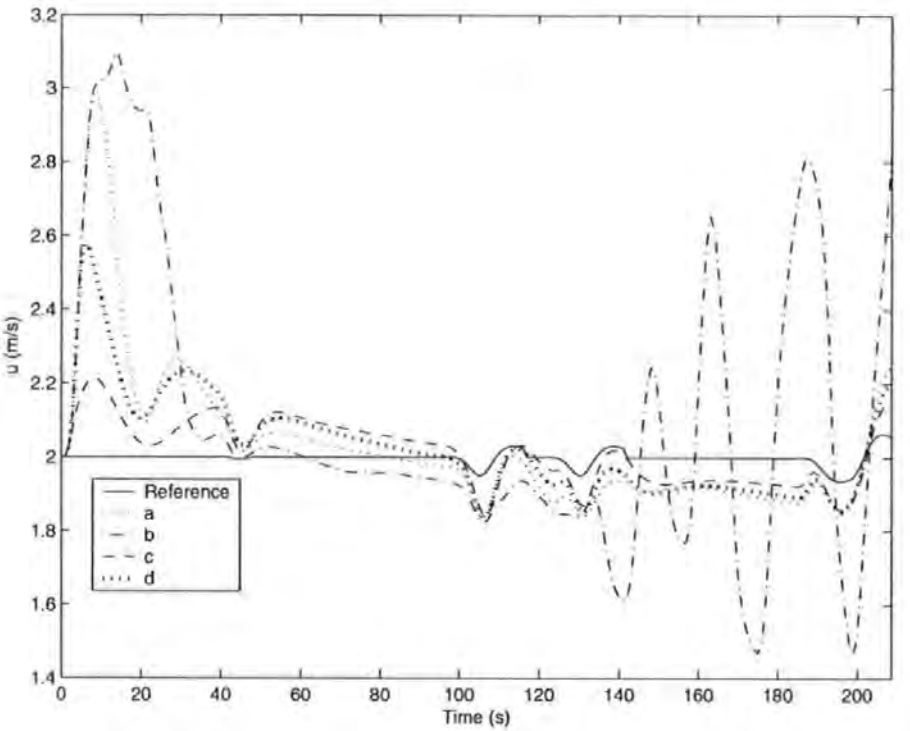


Figure 6.35: Surge responses

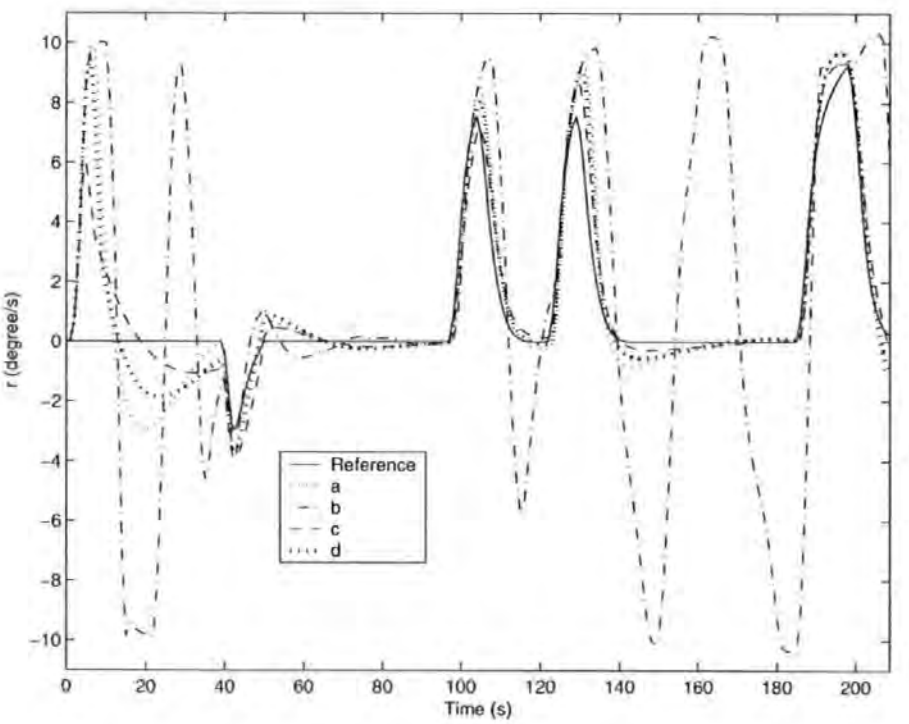


Figure 6.36: Heading rate responses

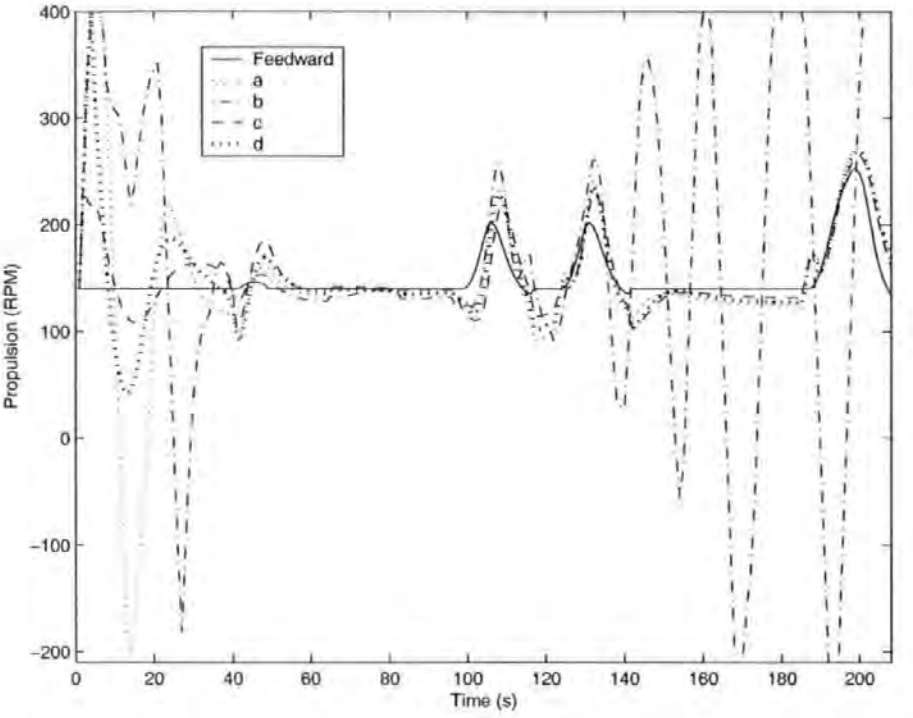


Figure 6.37: Propeller speed responses

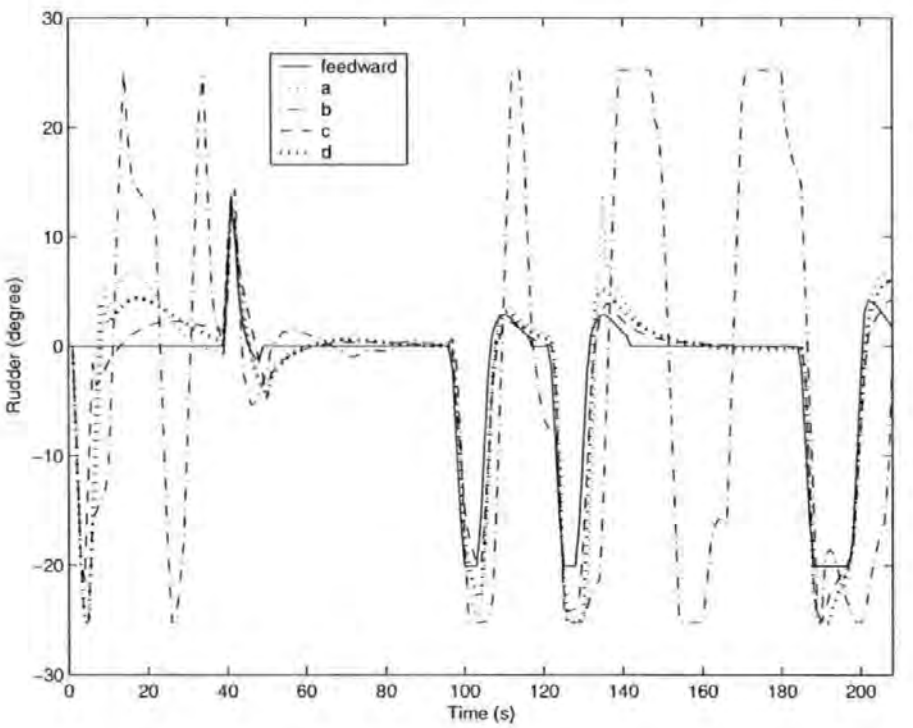


Figure 6.38: Rudder deflection responses

Table 6.2 shows the values proportional to the energy expended which corresponds to the individual case. The data were extracted by integrating the actuator output time history. In this context, the outputs are the propeller rotation and the rudder deflection. The squared value of the aforementioned outputs are directly proportional to the force they imparted, hence when integrated in time, they become proportional to the energy expended. One can observe that the energy consumption of case *a* is not the lowest amongst all except for its rudder actuation, but it is still a reasonable trade off for the performance achieved. Case *b* shows the worst energy consumption in confirmation to its oscillating trajectory. Case *c* reveals a very commendable performance even without the states dependent norm. On the other hand, case *d* exhibits a low propulsion output energy consumption but a rather high rudder output energy consumption. Its tracking performance is not comparable to the case *a*.

Case	RPM ($\times 10^6$)	Rudder
a	5.70	3.92
b	10.24	16.78
c	5.11	4.09
d	5.32	4.96

Table 6.2: Values proportional to the energy expended

6.10 Concluding Remarks

This chapter completed the design of an obstacle avoidance system for an AUV. To the extent of the author's knowledge, this is the first endeavour to combine the MA+RRT planner with the SDRE controller, let alone catering especially for AUVs. With regards to the trajectory trackers performance: the LQR and SDRE controller have been analysed in detail and accompanied with simulation results. In this case, the LQR displayed a very disappointing performance which could be caused by improper tuning. The SDRE controller, on the other hand, exhibited quick convergence of the tracking errors, smooth transient response, adequate control effort, and robustness, even in the case of large initial errors and current disturbances. One merit, pertinent to the kinematic based SDRE controller that should not be underestimated is its nondependence on system dynamics. This extends its implementation to the majority of the AUVs.

Chapter 7

Conclusions and Future Work

7.1 Concluding Remarks

This thesis focussed on the investigation of a novel motion planning algorithm, system dynamic quantisation, trajectory tracking, sonar processing and workspace representation methods as a solution to the AUV collision avoidance problem. It presented a repertoire of techniques which can be exploited as a whole to develop a functional collision avoidance system for AUVs. Both the obstacle detection and obstacle avoidance frameworks presented in this work are based on a computational approach. This work is the first known use of a partially hybrid technique in the AUV motion planning and is thus considered as a major contribution in relation to this field.

The feasibility of applying the RRT algorithm as a solution to the AUV motion planning problem has been intensively studied. Although the algorithm is endowed with several appealing properties such as robustness to the state explosion effect and ability to solve kinodynamic problems, it also suffers from a few drawbacks that degrade its practical utility. For instance, the randomise nature of the algorithm renders the solutions highly suboptimal and contains unnecessary micro random manoeuvres. Further aggravating the condition, its performance is highly sensitive to the metric definition and it also requires a system dynamic model to function properly.

Instigated by the inherent RRT issues, the Manoeuvre Automaton (MA) represen-

tation was introduced to address some of the deficiencies. This unique framework builds on the inherent characteristics of the vehicle dynamics to achieve a high level of computational efficiency, compromising between performance degradation and computational tractability. In quantisation the model, care must be taken not to over simplified or over complicated the model. The former will constraint the admissible responses of the original system, resulting in an excessively rigid dynamics characteristic. In contrast, the latter creates a state explosion effect, causing the model to be too computational intensive, making the quantisation process a counter productive one. Through the synergistic combination of the MA representation and the RRT algorithm, an enhanced performance version has been devised. This novel algorithm was extended to a multi-node version and for accommodating the time-varying final state problem. Simulation studies showed that the novel MA+RRT algorithm provides an interesting solution to the AUV motion planning problem.

Two controllers have been investigated as a potential candidate trajectory tracker. It was deduced that the complexity of tuning the $\mathbf{Q}(t)$, $\mathbf{R}(t)$ and the $\mathbf{M}(t)$ matrices of the LQR, induced by its multivariable and severe cross-coupling characteristic, limits its potential application. Additionally, there is a propensity for the LQR to be unstable when it violates the linearised regime. This phenomenon is accentuated when it is employed in a two degrees of freedom controller architecture. In contrast, the nonlinear state dependent Riccati equation (SDRE) controller displayed impressive performance for the tracking problem. Its characteristic to assimilate the state error into the $\mathbf{R}(\mathbf{x})$ and $\mathbf{Q}(\mathbf{x})$ matrices allows it to exhibit a more robust response. Likewise, the state dependent tuning parameters also significantly expand the performance latitude of the controller as evidenced by the quadratic state error function employed in the thesis. Finally, another noteworthy feature of the kinematic based SDRE controller, is its system dynamics independent characteristic, hence allowing a more widespread adoption.

The obstacle detection unit of a particular data processing was derived from the image processing methodology. Both the sonar processing and workspace representation were developed and the veracity of the techniques was tested in a sea trial using a dedicated forward looking sonar. The sonar, being the 'eye' of an AUV, is essential for a collision avoidance task. This thesis utilised the AT500 sonar for obstacle detection. It was discovered in the experiment that the sonar has a maximum range of only 40 m. The short detection range limits its usability to only small agile AUVs. Larger

AUVs such as the *AUTOSUB* and *Theseus* require a longer range sonar, due to their restricted minimum turning radius. Nonetheless, at a very appealing market price of approximately 5000 pounds a unit, the AT500 is expected to pose a very attractive and niche solution for cost conscious customers. The occupancy grid exploited in the workspace representation facilitates data maintenance and implementation. Instead of a linear function, a quadratic function was used instead to 'maintain' and filter the perceived targets in the workspace representation. Experiment results showed adequate false object elimination performance. These techniques remain sufficiently flexible for porting to other forward looking sonars with only a minor tuning requirement.

It is important to note that although the computational techniques set forth in this study were developed for AUVs, but it is not restrictive and can effortlessly be applied to other autonomous vehicles either on the land and aerospace domains. Therefore, the algorithms are valuable as generic practical tools for all types of collision avoidance task. It is felt that major contributions to knowledge are forthcoming from these techniques.

7.2 Recommendations for Future Research

Several different directions for future research have been highlighted through the completion of the work within this thesis. The following points provide a summary of these areas and are not considered to be exhaustive:

7.2.1 Optimal planning in constant currents

It is a common perception to consider a disturbance as a form of nuisance to a system. This, however is not entirely true. Disturbances such as non-zero mean currents, can be exploited to provide a beneficial effect to the AUV trajectory. The MA+RRT algorithm should be extended to accommodate the effect of nonzero-mean currents. By estimating the current and accommodating for its effect in the resulted trajectory during the optimisation process, one can extract a trajectory that is more efficient by

exploiting the intrinsic energy of the current. Simplistically speaking, one can design a path such as the current will attempt to 'push' the AUV to the goal instead of going against it, expending precious energy as a consequence.

7.2.2 Sensor-based motion planning

Sensor based motion planning is necessary for a system that functions in an unknown and unstructured environment. The unavailability of *a priori* information renders the AUV to depend on the sensor information. Sensor such as the sonar is limited by the sensor noise, environment and its specification such as resolution, frame-rate and number of beams. It is known that comprehensive information regarding an environment cannot be acquired easily, hence incremental sensing, gradual accumulation of evidence from sensors is needed for decision making. Furthermore, the optimal decision from the limited information is not the same from the optimal decision from having complete information.

7.2.3 Reflexive module

So far in this thesis the reflexive module has been neglected. Instead, of the conventional collision avoidance reflexive behaviour, which are *ad hoc* by nature, it would be an interesting proposition, if one is able to evolve the AUV's behaviour in accordance to the 'Rules of the Road', as remarked in Section 2.4.

7.2.4 Fault tolerant control or reconfigurable system

It is a common knowledge that AUVs are usually employed in hazardous and harsh terrains such as polar regions and littoral waters. This tends to increase the risk of failure, particularly of mechanical components as jammed control surfaces. This incident will certainly alter the dynamic behaviour of the AUV. It was mentioned previously that the proposed MA+RRT algorithm exploited LP to perform on-line optimisation of the trajectory. This feature allows one to accommodate easily for any

dynamics changes of the AUV in question, assuming the manoeuvres are explicitly designed for this. Consequently, this will make the AUV highly resilient faults, in the same time improving its survivability rate.

7.2.5 Multi-agent robotics

Higher mission requirements either in terms of time taken, larger operational envelope and cost saving, have become the impetus behind the technology of multi-agent underwater robots. Clearly, for a successful mission involving several AUVs, communication plays a vital role in the synchronisation of their activities. This leads to a dedicated communication protocol and also a definition of manoeuvres with unique properties that can be easily utilised for multiple AUV manoeuvres. The extension of the MA+RRT algorithm for this task will be crucial for the advancement of this field.

7.2.6 Multiple-target tracking system

The multiple-target tracking system is essential in a dynamic environment. It allows an AUV to estimate the target parameters such as velocity, such that proper preemptive action can be taken. In a static environment, tracking of multiple static targets is advantages for the implementation of the simultaneous localisation and mapping scheme. Indeed, this subject poses several challenging and yet appealing issues that warrant further research.

7.2.7 Workspace representation

This thesis uses the occupancy grid method, a variant of the spatial decomposition scheme for the workspace representation. On the other hand, most of the simulations are done using a geometry map. This however, will not invalidate the results, since workspace representation was designed to be 'isolated', a form of software abstraction, from the trajectory planner, through the use of a suitable 'collision detection'

function. The geometry map is well suited for small and dynamic target tracking while the occupancy grid is more efficient for modelling static, unstructured, partially observable objects. In light of this issue, it will be interesting to produce a hybrid workspace representation, exploiting the merits of the two different schemes.

References

Abdul-Baki, B., J. Baldwin and M. Rudel (1999). Independent Validation and Verification of the TCAS II Collision Avoidance Subsystem. *IEEE, AIAA Annual Digital Avionics Systems Conferance.* Available from: `http://www.rannoch.com/PDF/TCAS_DASC_paper.pdf`.

Aerotech News and Review (2003). Flight Demonstrations Evaluate UAV Collision-avoidance Technology. *Online Journal of Aerospace and Defence Industry News.* Available from: `http://www.aerotechnews.com/starc/2003/040703/UAV_collision.html`.

Aguiar, A. and A. Pascoal (2002). Dynamic Positioning and Way-point Tracking of Underactuated AUVs in the Presence of Ocean Currents. 41^{st} *IEEE Conf. on Decision and Control.*

Aguiar, A. and J. Hespanha (2004). Logic-based Switching Control for Trajectory-tracking and Path-following of Underactuated Autonomous Vehicles with Parametric Modeling Uncertainty. *Proc. Of ACC'04 American Control Conference* **4**, 3004–3010.

Aguiar, A., J. Hespanha and P. Kokotovic (2005). Path-following for Non-minimum Phase Systems Removes Performance Limitations. *IEEE Trans. on Automatic Control* **50**(2), 234–239.

Ahmad, S. M. and R. Sutton (2003). Dynamic Modelling of a Remotely Operated Vehicle. *Proc.* 1^{st} *IFAC Workshop on Guidance and Control of Underwater Vehicles (GCUV2003)* pp. 47–52.

Akkizidis, I. S., G. Roberts, P. Ridao and J. Batlle (2003). Designing a Fuzzy-like PD Controller for an Underwater Robot. *Control Engineering Practice* **11**(4), 471–480.

Allison, J. L., D. Watson and T. Cook (1989). Intelligent Waypoint Transiting In Complex AUV Environment. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 6^{th} pp. 246–257.

Alt, C. and J Grassle (1992). LEO-15: An Unmanned Long Term Environmental Observatory. *In Proc. MTS/IEEE Oceans 1992* pp. 12–37.

Alt, C., B. Allen, T. Austin and R. Stokey (1994). Remote Environmental Monitoring Units. *In Proc. MTS/IEEE Oceans 1994* pp. 13–37.

Amato, N. and Y. Wu (1996). A Randomized Roadmap Method for Path and Manipulation Planning. *IEEE Int. Conference on Robotics and Automation* pp. 113–120.

Amato, N., O. Bayazit, L. Dale, C. Jones and D. Vallejo (1998). OBPRM: An Obstacle-Based PRM for 3-D Workspaces. *Workshop on Algorithmic Foundations on Robotics (WAFR).* Available from: `http://citeseer.nj.nec.com/amato98prm.html`.

Andrews, L. (2001). A Data Structure and Algorithm for the Nearest Point. *IEEE Transactions on Software Engineering* **19**(11), 40–49.

Anon (1999). AUV Orders Show Industry Commitment. *Int. Ocean Systems Design.*

Antonelli, G., S. Chiaverini, R. Finotello and E. Morgavi (2001). Real-Time Path Planning and Obstacle Avoidance For An Autonomous Underwater Vehicle. *IEEE Int. Conference on Robotics and Automation* **1**, 78–83.

Arai, H., K. Tanie and N. Shiroma (1998). Time-scalling Control of An Underactuated Manipulator. *IEEE Int. Conference on Robotics and Automation* pp. 525–536.

Arbib, M. (1981). Perceptual Structures and Distributed Motor Control. *Handbook of Physiology The Nervous System II* pp. 1449–1456.

Arinaga, S., S. Nakajima, H. Okabe, A. Ono and Y. Kanayama (1996). Motion Planning Method for an AUV. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 477–484.

Arkin, R. C. (1998). *Behaviour-based Robotics: Intelligent Robotics and Autonomous Agents.* The MIT Press.

Arya, S., D. Mount, N. Netanyahu, R. Silverman and A. Y. Wu (1998). An Optimal Algorithm for Approximate Nearest Neighbour Searching Fixed Dimensions. *Journal of ACM* **45**(6), 891–923.

Astrom, K. J. and T. Hagglund (1995). *PID COntrollers.* 2 ed.. Instrument Society of America. North Carolina, US.

Banks, S. and K. Manha (1992). Optimal Control and Stabilisation for Nonlinear Systems. *IMA Journal of Mathematical Control and Information* **9**, 179–196.

Bar-Sharlom, Y. and X. R. Li (1995). *Multitarget/Multisensor Tracking: Principles and techniques.* YBS.

BBC News (2002). Relatives visit jet crash site. *BBC News.* Available from: `http://news.bbc.co.uk/1/hi/world/europe/2092010.stm`.

Bellingham, J. G., T. Consi and R. Beaton (1990). Keeping Layered Control Simple. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 3–8.

Bellman, R. E. (1957). *Dynamic Programming.* Princeton University Press. Princeton, N.J.

Bennet, A. and J. Leonard (2000). Behavior-Based Approach To Adaptive Feature Detection and Following With Autonomous Underwater Vehicles. *IEEE Journal of Oceanic Engineering* **25**(2), 213–226.

Bertsekas, D. P. and J. N. Tsitsiklis (1996). *Neural-Dynamic Programming: Optimisation and Neural Computation Series,3.* Athena Scientific. ISBN 1 8865 2910 8.

Bizup, D. F. (2003). The Over-Extended Kalman Filter–Don't Use it!. *Report SIE-030001* pp. 1–23. Available from: `http://www.sys.virginia.edu/techreps/2003/sie-030001.pdf`.

Blackman, S. S. (1986). *Multiple-Target Tracking with Radar Applications.* Artech Hounse Inc. MA.

Blackman, S. S. and R. Popoli (1999). *Design and Analysis of Modern Tracking Systems.* Artech House. MA.

Blidberg, D., S. Chappel, J. Jaibert, R. Turner, G. Sedor and P. Eaton (1990). The EAVE AUV Program at the Marine Systems Engineering Laboratory. *Proc. Of the IARP* 1^{st} *Workshop on: Mobile Robots for Subsea Environments* pp. 33–42.

Bogdanov, A., M. Carlsson, G. Harvey and J. Hunt (2003). State-Dependent Riccati Equation Control of a Small Unmanned Helicopter. *Proc. AIAA Guidance, Navigation and Control Conference* pp. 1–11.

Bohlin, R. and L. E. Kavraki (1998). Path Planning using Lazy (PRM). *IEEE Int. Conference on Robotics and Automation* pp. 521–528. Available from: `http:/citeseer.nj.nec.com/bohlin00path.html`.

Boor, V., M. H. Overmars and A. Stappen (1999). The Gaussian Sampling Strategy for Probabilistic Roadmap Planners. *IEEE Int. Conference on Robotics and Automation* pp. 1018–1023.

Borenstein, J. and Y. Koren (1991). The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots. *IEEE Journal on Robotics and Automation* **7**(3), 278–288.

Branicky, M. S., S. M. LaValle, K. Olson and L. Yang (2002). Deterministic vs. Probabilistic Roadmaps. *IEEE Transactions on Robotics and Automation, Submitted* pp. 1–13.

Brockett, R. (1983). Asymptotic Stability and Feedback Stabilisation. *Differential Geometric Control Theory* pp. 181–191.

Brockett, R. (2000). Formal Languages for Motion Description and Map Making. *Proc. of Symposia in Applied Mathematics* **41**, 181–193.

Brooks, R. (1986). A Robust Layered Control System for a Mobile Robot. *IEEE Transactions on Robotics and Automation* pp. 14–23.

Brown, H. (1983). *Brown's Rule of the Road Manual.* 7^{nd} ed.. Brown, Son & Ferguson,ltd. Glasgow, UK. ISBN 0 8517 4463 X.

Bruce, J. and M. Veloso (2002). Real-Time Randomised Path Planning for Robot Navigation. *IEEE Int. Conference on Intelligent Robots and Systems.*

Brutzman, D., M. Compton and Y. Kanayama (1992). Autonomous Sonar Classification using Expert Systems. *IEEE Oceans Conference Record* pp. 554–559.

Bryson, A. and Y. C. Ho (1975). *Applied Optimal Control.* Hemisphere. Washington, US.

Bullo, F. and A. D. Lewis (2004). *Geometric Control of Mechanical Systems: Modelling, Analysis, and Design for Simple Mechanical Control Systems.* Springer-Verlag. New York-Heidelberg-Berlin.

Burl, J. (1998). *Linear Optimal Control: H2 and H-Infinity Methods.* Prentice Hall. ISBN 0 2018 0868 4.

Butler, B. and V. Hertog (1993). Theseus: A Cable-laying AUV. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 8^{th} pp. 1–6.

Caccia, M., R. Bono and G. Veruggio (1995*a*). 2-D Acoustic World Reconstruction. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 9^{th} pp. 299–305.

Caccia, N., P. Virgili and G. Veruggio (1995*b*). Architectures for AUVs: A Comparative Study of Some Experiences. *AUVS'95 Int. Symposium and Exhibition* pp. 164–178.

Cahill, R. (1997). *Collisions and Their Causes.* 2^{nd} ed.. Nautical Books. US. ISBN 0 9630 0187 6.

Canudis de Witt, C. and O. J. Sørdalen (1991). Exponential Stabilisaiton of Mobile Wheeled Robots with Nonholonomic Constraints. *IEEE Conf. on Decision and Control* pp. 692–697.

Carpenter, R. N. (1998). Concurrent Mapping and Localization with FLS. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 113–148.

Cervern, W. T., F. Bullo and V. L. Coverstone (2004). Vehicle Motion Planning with Time-Varying Constraints. *Journal of Guidance, Control, and Dynamics* **27**, 506–509.

Chappell, S., R. Turner and E. Turner (1997). Coorperative Behavior in an Autonomous Oceanographic Sampling Network: MAUV Project Update. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology* pp. 375–384.

Cheng, P. and S. M. LaValle (2001). Reducing Metric Sensitivity in Randomised Trajectory Design. *IEEE Int. Conference on Intelligent Robots and Systems* pp. 43–48.

Cheng, P. and S. M. LaValle (2002). Resolution Complete Rapidly-exploring Random Trees. *IEEE Int. Conference on Robotics and Automation* pp. 267–272.

Cheng, P., Z. Shen and S. M. LaValle (2000). RRT-based Trajectory Design for Autonomous Automobiles and Spacecraft. *Control Sciences* **11**(3–4), 167–194.

Chevalier, F. (2002). *Principles of Radar and Sonar Signal Processing.* Artech House. Boston and London. ISBN 1 58053 338 8.

Civil Aviation Authority (CAA) (2002). CAP 722: Unmanned Aerial Vehicle Operations in UK Airspace-Guidance. *Directorate of Airspace Policy.* Available from: `http://caa.co.uk/docs/33/CAP722.pdf`.

Cloutier, J. R. (1997). State-dependent Riccati Equation Techniques. *Proc. of the American Control Conference* pp. 932–936.

Cloutier, J. R., C. D'Souza and C. Mracek (1996). Nonlinear Regulation and Nonlinear H_∞ Control via The State-dependent Riccati Equation Technique. *IFAC World Congress.*

Coble, C. R., E. G. Murray and D. R. Rice (1996). *Earth Science.* 3 ed.. Prentice Hall. Englewood Cliffs, NJ.

Cockcroft, A. and J. Lameijer (2001). *A guide to the Collision Avoidance Rules.* 5^{nd} ed.. Butterworth-Heinemann. Oxford, UK. ISBN 0 7506 2690 9.

Corban, J. E., A. Calise, J. Prasad, G. Heynen, B. Koenig and J. Hur (2003). Flight Evaluation of An Adaptive Velocity Command System for Unmandded Helicopters. *AIAA Guidance, Navigation, and Control Conference* (AIAA-2003-5594), 1–11.

Cornforth, W. and K. Croff (2000). Developement of an Environment-Sensitive Navigation System for the AUV Autolycus. *Marine Technology* **37**(4), 238–245.

Cowling, D. (1996). Full Range Autopilot Design for an Autonomous Underwater Vehicle. 13^{th} *IFAC Triennial World Congress* **Q**, 339–344.

Cowling, D. and S. Corfield (1995). Control Functions for Autonomous Underwater Vehicle On-board Command and Control Systems. *IEE Computing and Control Division Colloquium on Control and Guidance of Remotely Operated Vehicles* (124), 3/1–3/8.

Craven, P. J. (1999). *Intelligent Control Strategies for an Autonoumous Underwater Vehicle, PhD Thesis.* University of Plymouth. UK.

Curtis, T. E. and R. J. Ward (1980). Digital Beamforming for Sonar. *IEE Proc. Communication, Radar and Signal Processing.*

Dai, D., M. Chantler, D. Lane and N. Williams (1995). A Spatial-Temporal Approach for Segmentation of Moving and Static Objects in Sector Scan Sonar Image Sequences. *IEE* 5^{th}*, Int. Conference on Image Processing and Its Applications.*

Darling, R. (1997). Converting Matrix Riccati Equations to Second-order Linear ODE. *Society for Industrial and Applied Mathematics (SIAM) Journal* **39**(3), 508–510.

DeMuth, G. and S. Springsteen (1990). Obstacle Avoidance using Neural Networks. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 213–215.

Dijkstra, E. W. (1959). A Note on Two Problems in Connection. *Numerical Mathematics* **1**, 269–271.

Divelbiss, A. and J. T. Wen (1997). Trajectory Tracking Control of a Car-Trailer System. *IEEE Trans. on Control Systems Technology* **5**(3), 1063–278.

Do, K., Z. Jiang and J. Pan (2002). Robust Adaptive Path Following of Underactuated Ships. *Proc.* 41^{st} *IEEE Conf. on Decision and Control* pp. 3243–3248.

Dozier, G., S. McCullough, A. Homaifar, E. Tunstel and L. Moore (1998). Multiobjective Evolutionary Path Planning via Fuzzy Tournament Selection. *IEEE Int. Conference on Evolutionary Computation (ICEC'98)* pp. 684–689. Available from: `http://robotics.jpl.nasa.gov/people/tunstel/papers/icec98.pdf`.

Drumbheller, D. and H. Lew (2001). A Parameter-Insensitive Flase Alarm Rate Detection Processor. *DSTO-TR-1153 Technical Report (Department of Defence)* pp. 1–18.

Dubins, L. (1957). On Curves of Minimal Length with A Constraint on Average Curvature and with Prescribed Initial and Terminal positions and Tangents. *American Journal of Mathematics* **79**, 497–516.

Dudek, G. and M. Jenkin (2000). *Computational Principles of Mobile Robotics.* Cambridge University Press. Cambridge, UK. ISBN 0 5215 6876 5.

Encarnacao, P. and A. Pascoal (2002). Combined Trajectory Tracking and Path Following for Marine Craft. *Proc.* 15^{th} *IFAC World Congress.*

Erdem, E. (2001). *Analysis and Real-Time Implementation of State-Dependent Riccati Equation Controlled Systems, PhD Thesis.* University of Illinois. Urbana-Champaign.

Ferguson, J., A. Pope, B. Butler and R. Verrall (1999). Theseus AUV–Two Record Breaking Missions. *Sea Technology Magazine* pp. 65–70.

Fodrea, L. and A. Healey (2003). Obstacle Avoidance Control for The Remus Autonomous Underwater Vehicle. *Proceedings of the* 1^{st} *IFAC Workshop on Guidance and Control of Underwater Vehicles.*

Fogel, D. and L. Fogel (1990). Optimal Routing of Multiple Autonomous Underwater Vehicles Through Evolutionary Programming. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV).*

Fossen, T. I. (1994). *Guidance and Control of Ocean Vehicles.* John Wiley & Sons, Inc.

Fox, R., A. Garcia, Jr. and M. Nelson (2000). A Generic Path Planning Strategy for Autonomous Vehicles. *The University of Texas -Pan American, Department of Computer Science Technical Report CS-00-25.* Available from `http://citeseer.nj.nec.com/340664.html`.

Foxwell, D. (2000). Unmanned Vehicles, Submarine-launched UUVs Swims Ahead. *Warship Technology* pp. 6–7. May.

Fraichard, Th. and A. Scheuer (2004). From Reeds and Shepp's to Continous-Curvature Paths. *IEEE Trans. on Robotics* **20**(6), 1025–1035. Available from urlhttp://emotion.inrialpes.fr/bibemotion/2004/FS04.

Frazzoli, E. (2001). *Robust Hybrid Control for Autonomous Vehicle Motion Planning, PhD Thesis.* Massachusetts Institute of Technology, Cambridge, MA.

Frazzoli, E. (2002*a*). Manoeuvre-Based Motion Planning And Coodination for Single and Multiple UAV's. *Proc. of AIAA/IEEE Digital Avionics Systems Conference* **2**, 8D31–8D312.

Frazzoli, E. (2002*b*). Quasi-Random Algorithms for Real-Time Spacecraft Motion Planning and Coordination. *Acta Astronautica* **53**, 485–495.

Frazzoli, E., M. A. Dahleh and E. Feron (1999). Hybrid Control Architecture for Aggressive Maneuvering of Autonomous Helicopters. *Pro. of The IEEE Conference on Decision and Control* **3**, 2471–2476.

Frazzoli, E., M. A. Dahleh and E. Feron (2002). Real-Time Motion Planning for Agile Autonomous Vehicles. *Journal of Guidance, Control, and Dynamics* **25**(1), 116–129.

Frazzoli, E., M. Dahleh and E. Feron (2004). Manoeuvre-based Motion Planning for Nonlinear Systems with Symmetries. *IEEE Transactions on Robotics and Automation.*

Fujii, T. and T. Ura (1996). Development of an Autonomous Underwater Robot 'Twin-Burger' for Testing Intelligent Behaviors in Realistic Environments. *Autonomous Robots* **3**, 285–296.

Fujimura, K. (1991). *Motion Planning in Dynamic Environments.* Springer-Verlag. Tokyo.

Garcia, A. (1997). Overview of Path Planning Algorithms for Autonomous Vehicles. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 11^{th} pp. 426–431.

Gaskell, T. F. (1973). *The Gulf Stream.* John Day Company. New York.

Giardina, C. and E. Dougherty (1987). *Morphological Methods in Image and Signal Processing.* Prentice Hall. New Jersey. ISBN 0 13 601295 7 025.

Girogetti, Nicolo' (2004). GLPKMEX : GLPK-MATLAB MEX Interface. *Dipartimento di Ingegneria dell'Informazione.* Available from `http://www.dii.unisi.it/~giorgetti/downloads.html`.

Greenspan, M. and M. Yurick (2003). Approximate K-D Tree Search for Efficient ICP. *3D Digital Imaging and Modeling (3DIM 2003)* pp. 442–448.

Griffiths, G., P. Stevenson, A. Webb, N. Millard, S. McPhail, M. Pebody and J. Perrett (1999). Open Ocean Operation Experience with the AUTOSUB-1 AUV. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 11^{th} pp. 1–12.

Gross, M. G. (1982). *Oceanography: A View of the Earth.* Prentice Hall. New York.

Gross, M. G. (1990). *Oceanography.* 6 ed.. Merrill. Columbus.

Halton, J. H. (1960). On the Efficiency of Certain Quasi-random Sequences of Points in Evaluating Multi-dimensional Integrals. *Numerical Mathematics* **2**, 84–90.

Hammersley, J. M. (1960). Monte Carlo Methods for Solving Multivariable Problems. *Proceedings of the New York Academy of Science* **86**, 844–874.

Harro, R. and L. Fabian (2002). Accident Description. *Aviation Safety Network.* Available from: `http://aviation-safety.net/database/2002/020701-0.htm`.

Hart, P., J. Nilsson and B. Raphael (1968). A Formal Basis for The Heuristic Determination of Minimum Cost Paths. *Transactions on Systems, Man and Cybernetics* pp. 100–107.

Healey, A. and D. Lienard (1993). Multivariable Sliding Mode Control for Autonomous Diving and Sterring of Unmanned Underwater Vehicles. *IEEE Journal of Oceanic Engineering* **18**(3), 327–339.

Healey, A., D. Marco, R. McGhee, D. Brutzman and R. Cristi (1995). Evaluation of The NPS PHOENIX Autonomous Underwater Vehicle Hybrid Control System. *Proc. American Control Conference* pp. 477–484.

Henriksen, L. (1994). Real-Time Underwater Object Detection Based on an Electrical Scanned High-resolution Sonar. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 99–104.

Hindman, R. and J. Hauser (1992). Manoeuvre Modified Trajectory Tracking. *Proc. MTNS'96, Int. Symposium on the Mathematical Theory of Networks and Systems.*

Hino, J. H. (1989). Intelligent Planning for a Search and Surveillance Unmanned Underwater Vehicle. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 6^{th} pp. 236–245.

Hobson, B., M. Murray and C. Pell (1999). Pilotfish: Maximizing Agility in An Unmanned Underwater Vehicle. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 11^{th}.

Hristu, D., P. Krishnaprasad, S. Andersson, F. Zhang, P. Sodre and L. D'Anna (2000). The MDLe Engine: a Software Tool for Hybrid Motion Control. *ISR Technical Report TR2000-54* pp. 1–10. Available from: `http://techreports.isr.umd.edy/reports/2000/TR_2000-54.pdf`.

Hwang, Y. K. and N. Ahuja (1992). Gross Motion Planning - A Survey. *ACM Computing Surveys* **24**(3), 219–291.

Hyland, J. C. (1989). Optimal Obstacle Avoidance Path Planning For Autonomous Underwater Vehicles. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology* (6), 226–278.

Hyland, J. C. (1990). A Comparison of Two Obstacle Avoidance Path Planning For Autonomous Underwater Vehicles. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV).*

Ishii, K., T. Fujii and T. Ura (1995). An On-line Adaptation Method in Neural Network-based Control System for AUV's. *IEEE Journal of Oceanic Engineering.*

Jalving, B. and N. Størkersen (1994). The Control System of an Autonomous Underwater Vehicle. *Proc. of the Third IEEE Conference on Control Applications* **2**, 851–856.

Jiang, X., Y. Motai and X. Zhu (2005). Predictive Fuzzy Logic Controller for Trajectory Tracking. *IEEE Workshop on Computing in Industrial Applications* pp. 1–4.

Jordan, K. (2003). Remus AUV Plays Key Role in Iraq War. *Underwater Magazine.*

Kalman, R. E. (1960). A New Aproach to Linear Filtering and Prediction Problems. *Transactions of the ASME– Journal of Basic Engineering* **82**(Series D), 35–45.

Kambhampati, S. and L. Davis (1986). Multiresolution Path Planning for Mobile Robots. *IEEE Journal on Robotics and Automation* **2**, 135–145.

Kamon, I., E. Rimon and E. Rivlin (1995). A New Range-Sensor Based Globally Convergent Navigation Algorithm for Movile Robots. *CIS-Center of Intelligent Systems 9517, Computer Science DeptTechnion, Israel.*

Karatas, T. and F. Bullo (2001). Randomized Searches and Nonlinear Programming in Trajectory Planning. *Pro. of The IEEE Conference on Decision and Control* **5**, 5032–5037.

Kato, N., Y. ito, J. Kojima, K. Asakawa and Y. Shirasaki (1993). Guidance and Control of Autonomous Underwater Vehicle AQUA EXPLORER 1000 for Inspection of Underwater Cables. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 8^{th} pp. 195–211.

Khatib, O. (1986). Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *Int. Journal of Robotics Research* **5**(1), 90–98.

Kim, J. and J. P. Ostrowski (2003). Motion Planning of Aerial Robot using Rapidly-exploring Random Trees with Dynamic Constraints. *IEEE Int. Conference on Robotics and Automation* p. 343.

Knuth, D. E. (1997). *The Art of Computer Programming: Semi-numerical Algorithms.* Vol. 2. 3 ed.. Addison-Wesley. Reading, Mass., US.

Kolawole, M. (2002). *Radar Systems, Peak Detection and Tracking.* Newness. Oxford,UK.

Koren, Y. and J. Borenstein (1991). Potential Field Methods and Tehir Inherent limitations for Mobile Robot Navigation. *IEEE Int. Conference on Robotics and Automation* pp. 1398–1404.

Krogh, B. and C. R. Thorpe (1986). Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles. *IEEE Int. Conference on Robotics and Automation* pp. 1664–1669.

Kwiesielewicz, M., W. Piotrowski and R. Sutton (2001). Predictive versus Fuzzy Control of Autonomous Underwater Vehicle. *Proc. IEEE International Conference on Methods and Models in Automation and Robotics* pp. 609–612.

Lane, D. and E. Trucco (2000). Embedded Sonar and Video Processing for AUV Application. *Proc. Annual Offshore Technology Conference* **2**, 599–607.

Lane, D., M. Chantler and D. Dai (1998). Robust Tracking of Multiple Objects in Sector Scan Sonar Image Sequences using Optical Flow Motion Estimation. *IEEE Journal of Oceanic Engineering* **23**(1), 31–46.

Lane, D., M. Chantler, D. Dai and N. Williams (1996). Motion Estimation and Tracking of Multiple Objects in Sector Scan Sonar using Optical Flow. *IEE Colloquia on Autonomous Underwater Vechicles and Their Systems – Recent Developments and Future Prospects* pp. 1–11.

Lane, D., M. Chantler, E. Roberston and A. McFadzean (1988). Distributed Problem Solving Arhitecture for Sonar Image Interpretation. *Underwater Technology* **14**(3), 12–18.

Lane, D., M. Chantler, E. Roberston and A. McFadzean (1989). A Pyramidal Architecture for Knowledge-Based Sonar Image Intepretation. *IEE Colloquium on Transputers for Image Processing Applications.*

Latombe, J-C. (1991). *Robot Motion Planning.* Kluwer Academic Publishers. United State of America. ISBN 0 7923 9129 2.

Laubach, S. and J. Burdick (1999). An Autonomous Sensor-Based Path-Planner for Planetary Microrovers. *IEEE Int. Conference on Robotics and Automation* pp. 1–8. Available from: `http://citeseer.nj.nec.com/laubach99autonomous.html`.

Laumond, J-P. (1998). *Robot Motion Planning and Control.* Springer. ISBN 3 5407 6219 1.

Laumond, J. P., S. Sekhavat, F. Lamiraux, C. Samson, M. H. Overmars, A. Bellaiche, F. Jean, A. Oriolo and P. Svestka (1999). *Robot Motion Planning and Control (LAAS report 97438).* Internet. Laboratoire d'Analyse et d'Architecture des Systmes. Available from: `http://www.laas.fr/~jpl/book-toc.html`.

LaValle, S. M. (1998). Rapidly-exploring Random Trees: A New Tool for Path Planning. *TR 98-11 computer Science Dept., Iowa State University.* Available from: `http://janowiec.cs.iastate.edu/papers/rrt.ps`.

LaValle, S. M. (2005). *Planning Algorithms.* Internet. University of Illinois. Available from: `http://msl.cs.uiuc.edu/planning/`.

LaValle, S. M. and J. J. Kuffner (1999). Randomised kinodynamic planning. *IEEE Int. Conference on Robotics and Automation* pp. 473–479.

LaValle, S. M. and J. J. Kuffner (2000). Rapidly-exploring Random Trees: Progress and Prospects. *In Workshop on the Algorithmic Foundations of Robotics* pp. 293–308.

Leal, J. (2003). *Stochastic Environment Representation, PhD Thesis.* University of Sydney. Australia.

Levine, J. A. (2004). *Sampling-based Planning for Hybrid System, Master Thesis.* Case Western Reserve University.

Lewis, F. L. (1986). *Optimal Control.* Wiley. New York.

Li, G. and T. Bui (1998). Robot Path Planning using Fluid Model. *Journal of Intelligent and Robotic Systems* **21**, 29–50.

Li, T-Y. and Y-C. Shie (2003). An Incremental Learning Approach to Motion Planning with Roadmap Management. *IEEE Int. Conference on Robotics and Automation* pp. 3411–3416.

Lin, C-F. (1991). *Modern Navigation Guidance, and Control Processing.* Prentice Hall. New Jersey, USA.

Lin, M. C. (1999). Fast and Accurate Collision Detection for Virtual Environments. *Proc. IEEE Scientific Visual. Conf., 1999.* Available from: `www.citeseer.ist.psu.edu/article/lin99fast.html`.

Lin, M. C., D. Manocha, J. Cohen and S. Gottschalk (1996). Collision Detection: Algorithms and Applications). *In Proc. of The Algorithms for Robotics Motion and Manipulation* pp. 129–142.

Lisiewicz, J. S. and D. W. French (1999). The Naval Undersea Warfare Centre's Unmmaned Undersea Vehicle Initiative. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 11^{th} pp. 86–93.

Liu, C., M. Ang, Jr., K. Hariharan and L. Yong (2000). Virtual Obstacle Concept for Local Minimum Recovery in Potential Based Navigation. *IEEE Int. Conference on Robotics and Automation* **2**, 983–988.

Liu, X., S. Zhang, Y. Li and Y. Shang (1999). A New Method for AUV's Collision Avoidance. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 11^{th} pp. 587–591.

Loebis, D., R. Sutton and J. Chudley (2002). Review of Multisensor Data Fusion Techniques and Their Application to Autonomous Underwater Vehicle Navigation. *Journal of Marine Engineering and Technology* **1**(A1), 3–14.

Loggins, C. D. (2001). A Comparison of Forward-Looking Sonar Design Alternatives. *IEEE Oceans Conference Record* **3**, 1536–1545.

Louste, C. and A. Liegeois (2000). Near Optimal Robust Path Plannign for Mobile Robots. *Journal of Intelligent and Robotic Systems: Theory and Applications* **27**(1–2), 99–112.

Lozano-Perez, T. and M. Wesley (1979). An Algorithm for Planning Collision-free Paths among Polyhedral Obstacles. *Communicaitons of the ACM* **22**(10), 560–570.

Manikonda, V., P. Krinhnaprasad and J. Hendler (1995). A Motion Description Language and a Hybrid Architecture for Motion Planning with Nonholonomic Robots. *In Proc. IEEE Int. Conference on Robotics and Automation* **2**, 2021–2028.

MarineBio (1998). Ocean Facts. *www.marinebio.org.* Available from: `http:www.marinebio.org/MarineBio/Facts/`.

Marshfield, W. (1992). Submarine Data Set for use in Autopilot Research. *Technical Memorandum DERA/MAR TM.*

Mckendrick, J. D. (1989). Autonomous Knowledge-based Navigation in an Unknown Two-dimensional Environment with Convex Polygon Obstacles. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology* pp. 258–265.

Menon, P., G. Sweriduk and S. Vaddi (2004). Nonlinear Discrete-time Design Methods for Missile Flight Control Systems. *Proc. AIAA Guidance, Navigation and Control Conference* pp. 1–16.

Mignotte, M., C. Collet, P. Pérez and P. Bouthemy (2000). Sonar Image Segmentation using an Unsupervised Hierarchical MRF Model. *IEEE Transactions on Image Processing* **9**(7), 1216–1231.

Moitie, R. and N. Seube (2000). Guidance Algorithms For UUVs Obstacle Avoidance Systems. *IEEE Oceans Conference Record* **3**, 1853–1860.

Moran, B. A., J. Leonard and C. Chryssostomidis (1993). Geometric Shape from Sonar Ranging. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology* pp. 370–383.

Morrison, N., B. Evans, T. James and K. Allen (2003). Gambit MCM AUV: Overview and System Performance. *OCEANS 2003 Marine Technology and Ocean Science Conference* pp. 1723–1729. Sept.

Moscow, A. (1959). *Collision Course, the Adrea Doria and the Stockholm.* Putum and Sons. New York.

Movarec, H. P. and A. Elfes (1985). High Resolution Maps from Wide Angle Sonar. *IEEE Int. Conference on Robotics and Automation* pp. 116–121.

Mracek, C. and J. Cloutier (1998). Control Designs for the Nonlinear Benchmark Problem via the State-Dependent Riccati Equation Method. *Int. Journal of Robust and Nonlinear Control* 8(4–5), 401–433.

Murray, R. and S. Sastry (1990). Steering Nonholonomic Systems Using Sinusoids. *IEEE Conf. on Decision and Control* pp. 2097–2101.

Naeem, W. (2002). Model Predictive Control of an Autonomous Underwater Vehicle. *In Proceedings of UKACC 2002 Postgraduate Symposium* pp. 19–23.

Naeem, W., R. Sutton and S. Ahmad (2003). LQG/LTR Control of an Autonomous Underwater Vehicle using a Hybrid Guidance Law. *Proceedings of the* 1^{st} *IFAC Workshop on Guidance and Control of Underwater Vehicles* pp. 35–40.

Naeem, W., R. Sutton and S. Ahmad (2004). Model Predictive Control of An Autonomous Underwater Vehicle With A Fuzzy Objective Function Optimised Using A GA. *Proc. of Control Applications in Marine Systems (CAMS'04)* pp. 433–438.

Naylor, B. F. (1993). Constructing Good Partitioning Trees. *Graphics Interface* pp. 181–191.

Newman, P. and H. Durran-Whyte (1998). Using Sonar in Terrain-Aided Underwater Navigation. *IEEE Int. Conference on Robotics and Automation* pp. 440–445.

Nieuwstadt, M. J. (1997). *Trajectory Generation for Nonlinear Control Systems, PhD Thesis.* California Institute of Technology. Pasadena, California.

Nussbaum, F., G. Stevens and J. Kelly (1996). Sensors for a Forward-Looking High Resolution AUV Sonar. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 141–145.

Overmars, M. H. (2002). Recent Developments in Motion Planning. *Int. Conference on Computational Science (3)* pp. 3–13. Available from: `http://www-lns.tf.uni-kiel.de/ict/publikationen_hoeher.html`.

Ozbay, H. (1999). *Introduction to Feedback Control Theory.* CRC Press LLC. ISBN 0 8493 1867 x.

Parrish, D. K. and D. Ridgely (1997). Altitide Control of a Satellite Using The SDRE Method. *Proc. of the American Control Conference* **2**, 942–946.

Pearson, J. (1962). Approximation Methods in Optimal Control. *Journal of Electronics and Control.*

Perrault, D. and M. Nahon (1999). Fault-Tolerant Control of an Autonomous Underwater Vehicle. *Oceanic Engineering International* **3**(2), 85–94.

Petillot, Y., I. T. Ruiz and D. Lane (1998). Underwater Vehicle Path Planning using a Muti-beam Forward Looking Sonar. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 1194–1199.

Petrou, M. and P. Bosdogianni (1999). *Image Processing: The Fundamentals.* Wiley. England. ISBN 0 471 99883-4.

Pettersen, K. and T. Fossen (2002). Underactuated Dynamic Positioning of a Ship - Experimental Results. *IEEE Transactions on Control Systems Technology* **8**(5), 856–863.

Pettilot, Y., I. Ruiz and D. Lane (2001). Underwater Vehicle Obstacle Avoidance and Path Planning using a Multi-Beam Forward Looking Sonar. *IEEE Journal of Oceanic Engineering* **61**(3), 268–278.

Pietzykowski, Z. (2002). The Analyis of a Ship Fuzzy Domain in a Restricted Area. *IFAC Con. Control Applications in Marine Systems* pp. 1–6.

Prestero, T. (2001). *Verification of a Six-Degree of Freedom Simulation Model for the REMUS Autonomous Underwater Vehicle, Master Thesis.* MIT. Massachusetts, USA.

Reeds, L. and R. Shepp (1990). Optimal Paths for A Car That Goes Both Forwards and Backwards. *Pacific Journal of Mathematics.*

Ren, W. and R. W. Beard (2004). Constrained Nonlinear Tracking Control for Small Fixed-wing Unmanned Air Vehicles. *American Control Conference* pp. 4663–4668.

Repoulias, F. and E. Papadopoulos (2005). Trajectory Planning and Tracking Control of Underactuated AUVs. *IEEE International Conference on Robotics and Automation* pp. 1622–1627.

Richards, A., Y. Kuwata, and J.P. How (2003). Experimental Demonstrations of Real-time MILP Control. *Proc. of the AIAA Guidance, Navigation, and Control Conference* (AIAA-2003-5802), 1–11.

Ridao, P., J. Batlle and M. Carreras (2001). O^2CA^2 A New Object Oriented Control Architecture for Autonomy. The Reactive Layer. *Control Engineering in Practise Journal.*

Ridao, P., J. Batlle, J. Amat and G. Roberts (1999). Recent Trends in Control Architectures for Autonomous Underwater Vehicles. *Int.Journal of System Science* **30**(9), 1033–1056.

Ridao, P., J. Yuh, J. Batlle and K. Sugihara (2000). On AUV Control Architecture. *IEEE Int. Conference on Intelligent Robots and Systems* **2**, 885–860.

Rikoski, R. J., J. Leonard and P. M. Newman (2002). Stochastic Mapping Frameworks. *IEEE Int. Conference on Robotics and Automation* **1**, 426–433.

Robinson, P., C. S. Tan, C. Morris and R. Sutton (2003). Low Power Intelligent Sonars for Autonomous Underwater Vehicle Navigation and Collision Avoidance. *In Proceedings of Underwater Defence Technology (UDT) Europe.*

Rock, S. M., H. H. Wang and M. J. Lee (1995). Task-Directed Precision Control of the MBARI / Stanford OTTER AUV. *Proc. of the Int. Program Development in Undersea Robotics & Intelligent Control (URIC): A Joint US/Portugal Workshop* pp. 131–138.

Ruiz, I., Y. Petillot, D. Lane and J. Bell (1999). Tracking Object in Underwater Multibeam Sonar Images. *IEE Colloquium on Motion Analysis and Tracking* pp. 1–7.

Saimek, S. and P. Y. Li (2004). Motion Planning and Control of a Swimming Machine. *International Journal of Robotics Research* **23**(1), 27–53.

Samson, C. (1991). Feedback Stabilisation of a Nonholonomic Car-like Mobile Robot. *IEEE Conf. on Decision and Control.*

Sayer, T. Sarker P. and S. Fraser (1998). A Study of Autonomous Underwater Vehicle Hull Forms Using Computational Fluid Dynamics. *International Journal for Numerical Methods in Fluids* **25**(11), 1301–1313.

Sayyaadi, H., T. Ura and F. Teruo (2000). Collison Avoidance Controller for AUV Systems using Stochistic Real Value Reinforcement Learning Method. pp. 218–224. Available from: `http://citeseer.nj.nec.com/sayyaadi01auvs.html`.

Schouwenaars, T., B. Mettler, E. Feron and P. Jonathan (2003). Robust Motion Planning Using a Manoeuvre Automaton with Built-In Uncertainties. *Proc. of the American Control Conference* **3**, 2211–2216.

Schouwenaars, T., M. Valenti, E.Feron, and J.P. How (2005). Implementation and Flight Test Results of MILP-based UAV Guidance. *IEEE Aerospcace Conference* (1396), 1–13.

Schultz, A. (1991). Using a Genetic Algorithm to Learn Strategies for Collision Avoidance and Local Navigation. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology* pp. 213–215.

Shinjo, N. and J. Graeme (1995). Sensor-Based Fuzzy Obstacle Avoidance System for Autonomous Underwater Vehicles. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 9^{th} pp. 271–298.

Silvestre, C., P. Oliveira, A. Pascoal, D. Fryxell and I. Kaminer (1995). Design and Implementation of a Trajectory Tracking Controller for an Autonomous Underwater Vehicle. *Proc. of the American Control Conference* **5**, 2975–2979.

Simeon, T., J. Laumond and C. Nissoux (2000). Visibility-based probabilistic Roadmaps for Motion Planning. *Advanced Robotics Journal* **14**(6), 371–386.

Simhon, S. and G. Dudek (1998). A Global Topological Map Formed by Local Metric Maps. *IEEE Int. Conference on Intelligent Robots and Systems* **3**, 1708–1714.

Smith, W. (2004). Introduction to This Special Issue on Bathymetric from Space. *Oceanography: Special Issue-Bathymetric from Space* **17**(1), 6–7.

Stentz, A. (1994). Optimal and Efficient Path Planning for Partially-Known Environments. *IEEE Int. Conference on Robotics and Automation* (4), 3310–3317.

Stewart, D., D. Blacknell, A. Blake, R. Cook and C. Oliver (2000). An Optimal Approach to SAR Image Segmentation and Classification. *DERA* **9**(7), 1216–1231.

Subramaniam, L. V. and R. Bahl (1995). Segmentation and Surface Fitting of Sonar Images for 3-D Visualization. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 9^{th} pp. 290–298.

Sugihara, K. (1998). GA-based On-line Path Planning for SAUVIM. *Int. Conference on Industrial Engineering & Applications of Artificial Intelligence & Expert Systems* **2**, 329–338. Available from: `http://citeseer.nj.nec.com/sugihara98gabased.html`.

SUT (2000*a*). *The Operation of AUVs: Recommended Code of Practice.* Vol. 1. Society of Underwater Technology (SUT). UK. ISBN 0 9069 4038 9.

SUT (2000*b*). *The Operation of AUVs: Report On the Law.* Vol. 2. Society of Underwater Technology (SUT). UK. ISBN 0 90694 037 0.

SUT (2000*c*). *The Operation of AUVs: The Law Governing AUV Operations-Questions and Answers.* Vol. 3. Society of Underwater Technology (SUT). UK. ISBN 0 90694 037 0.

Sutton, R., A. Pearson and A. Tiano (2001). A Fuzzy Fault Tolerant Control Scheme for an Autonomous Underwater Vehicle. *IEEE Int. Conference on Methods and Models in Automation and Robotics* pp. 595-600.

Tan, C. S., R. Sutton and J. Chudley (2004*a*). An Incremental Stochastic Motion Planning Technique for Autonomous Underwater Vehicles. *IFAC Conference on Control Applications in Marine Systems (CAMS 2004), Italy* pp. 483–488.

Tan, C. S., R. Sutton and J. Chudley (2004*b*). Collision Avoidance Systems for Autonomous Underwater Vehicles, Part A: Obstacle Detection. *Journal of Marine Science and Environment, IMarEST* (C2), 39–50.

Tan, C. S., R. Sutton and J. Chudley (2004*c*). Collision Avoidance Systems for Autonomous Underwater Vehicles, Part B: Obstacle Avoidance. *Journal of Marine Science and Environment, IMarEST* (C2), 51–62.

Tan, C. S., R. Sutton and J. Chudley (2005*a*). CRASH, BANG, AVOIDANCE: Collision Avoidance Systems. *Oceanology today* (5), 22–23.

Tan, C. S., R. Sutton and J. Chudley (2005*b*). Quasi-Random Manoeuvre-based Motion Planning Algorithm for Autonomous Underwater Vehicles. *IFAC* 16^{th} *World Congress, Prague* pp. 483–488.

Tan, C. S., R. Sutton, J. Chudley and S. Ahmad (2003). Autonomous Underwater Vehicle Retrieval Manoeuvre Using Artificial intelligence Strategy. *Proc.* 1^{st} *IFAC Workshop on Guidance and Control of Underwater Vehicles (GCUV 2003)* pp. 143–148.

Tena Ruiz, I. (2001). *Enhanced Concurrent Mapping and Localisation using Forward-looking Sonar, PhD Thesis.* Heriot-Watt University. UK.

Tomatis, N., I. Nourbakhsh and R. Siegwart (2001). Simultaneous Localisation and Map Building: A Global Topological Model with Local Metric Maps. *IEEE Int. Conference on Intelligent Robots and Systems* pp. 421–426.

Tong, A. (2000). Marlin, The UK Military UUV Programme: A Programme Overview. *Proc. of Int. UUV Symposium' 2000* pp. 31–38.

Toomay, J. C. (1989). *Radar Principles for the Non-Specialist.* Scitech Publishing. Mendham. ISBN 0 442 20719 0.

Torsetnes, G. (2004). *Nonlinear Control and Observer Design for Dynamic Positioning using Contraction Theory, Master Thesis.* Norwegian University of Science and Technology. Norway.

Tortora, G. (2002). Fault-Tolerant Control and Intelligent Instrumentation. *Computing and Control Engineering Journal* **13**(5), 259–262.

Toussaint, G. J. (2000). *Motion Planning for Nonlinear Underactuated Vehicles using* $H_{Infinity}$ *Techniques, PhD Thesis.* University of Illinois. Urbana-Champaign.

Tran, T., C. Harris and P. Wilson (1997). A Vessel Management Expert System. *Proc Instn. Mechanical Engineers : Engineering for the Maritime Environment* **216**(Part M:J), 161–177.

Trucco, E., I. Ruiz Y. Pettilot, K. Plakas and D. Lane (2000). Feature Tracking in Video and Sonar Subsea Sequences with Applications. *Computer Vision and Image Understanding* **79**, 92–122. Available from: `http://www.idealibrary.com`.

Vadakkepat, P., K. C. Tan and M-L. Wang (2000). Evolutionary Artificial Potential Fields and Their Application in Real Time Robot Path Planning. *Congress of Evolutionary Computation (CEC'2000)* pp. 256–263.

Valavanis, P., D. Gracanin, M. Matijasevic, R. Kolluru and G. Demetriou (1997). Control Architecture for Autonomous Underwater Vehicles. *IEEE Control System Magazine* pp. 48–64.

Vasudevan, C., S. Smith and K. Ganesan (1995). Collision Avoidance by Dynamic Path Selection and Fuzzy Behaviour Control. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology* pp. 262–270.

Veen, B. V. and K. Buckley (1988). Beamforming: A Versatile Approach to Spatial Filtering. *IEEE ASSP Magazine* **5**, 4–24.

Vestgård, K. (1999). Seabed Surveying with the HUGIN UUV. *IEEE Int. Symposium on Unmanned, Untethered, Submersible Technology,* 11^{th} pp. 63–74.

Vukic, Z., G. Borovic and G. Beale (2001). Trajectory Following Neuro Controller for Submarine Maneuvering In Horizonal And Vertical Plane. 9^{th} *Mediterranean Conference on Control and Automation.*

Wahl, E., P. Eichel, D. Ghiglia and P. Thompson (1996). *Spotlight-Mode Synthetic Aperture Radar: A Signal Processing Approach.* Springer.

Walsh, G., D. Tilbury, S. Sastry, R. Murray and J. Laumond (1994). Stabilisation of Trajectories for Systems with Nonholonomic Constraints. *IEEE Trans. Automatic Control* **39**(1), 216–222.

Wang, Y. and D. M. Lane (1997). Subsea Vehicle Path Planning using Nonlinear Programming and Constructive Solid Geometry. *IEE Proc. Control Theory and Applications* **144**(2), 143–152.

Warren, C. W. (1990). A Technique for Autonomous Underwater Vehicle Route Planning. *IEEE Journal of Oceanic Engineering* **15**(3), 199–204.

Wettergreen, D., C. Gaskett and A. Zelinsky (1999). Autonomous Guidance and Control for an Underwater Robotic Vehicle. *Int. Conference on Field and Service Robotics.*

Williams, G. N., G. E. Lagace and A. Woodfin (1990). A Collision Avoidance Controller For Autonomous Underwater Vehicles. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV).*

Wise, K. and J. Sedwick (1997). Nonlinear Control of Agile Missiles using State-Dependent Riccati Equations. *Proc. of the American Control Conference* pp. 379–380.

Yahja, A., A. Stentz, S. Singh and B. L. Brumitt (1988). Framed-Quadtree Path Planning for Mobile Robots Operating in Sparse Environments. *IEEE Journal on Robotics and Automation* pp. 650–655.

Yoerger, D., A. M. Bradley, B. Walden and M. Cormier (2000). Fine-Scale Seafloor Survey in Rugged Deep-Ocean Terrain with an Autonomous Robot. *IEEE Int. Conference on Robotics and Automation* pp. 34–55.

Yuh, J. and S. K. Choi (1999). Semi-Autonomous Underwater Vehicle for Intervention Missions. *Sea Technology Magazine* **40**(10), 31–40.

Zadeh, L. A. (1965). Fuzzy Sets. *Informat. Control* **8**, 338–353.

Zanoli, S. M. and F. Affaitati (1999). Obstacle Avoidance with Scanning Sonar for Bottom Navigation. *IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 537–545.

Zapata, R. and P. Lepinay (1996). Collision Avoidance and Bottom Following of A Torpedo-like AUV. *IEEE Oceans Conference Record* **2**, 571–575.

Zhang, M. and R. M. Hirschorn (1997). Discontinuous Feedback Stabilisation of Nonholonomic Wheeled Mobile Robots. *Dynamics and Control* **7**(2), 155–169.

Zimmer, U. R. (2000). Embedding Local Metrical Map Patches in a Globally Consistent Topological Map. *Proc. Of Underwater Technologies.*

Zirrili, A., G. Roberts, A. Tiano and R. Sutton (2000). Adaptive Steering of a Containership based on Neural Networks. *International Journal of Adaptive Control and Signal Processing* pp. 849–873.

Appendix A

Obstacle Detection

Chapters 4 to Chapter 6 have concentrated on the development of a novel, yet functional obstacle avoidance module. The attention is diverted to the design of an obstacle detection module. The function of this specific module should not be underestimated, without it, a complete collision avoidance system cannot be developed.

Before embarking on a detailed exposition of the module, it would be informative to have an overview perspective of the critical submodules that constitute an advanced obstacle detection system for an AUV. The exposition here is more specific compared to Section 2.3. Referring to the block diagram as depicted in Fig A.1, it is apparent that the three critical elements are the sonar data processing, multiple-target tracking (MTT) and workspace representation submodules. All of these submodules are fed with measurements from various sensors. Although, in the older systems, these modules tend to function independently, recently, the more advanced versions are shown to possess complex intermodule-interaction where information is exchanged from different submodules such that the optimal detection criterion is maintained in spite of changing environmental parameters. In other words, it is a form of adaptive system.

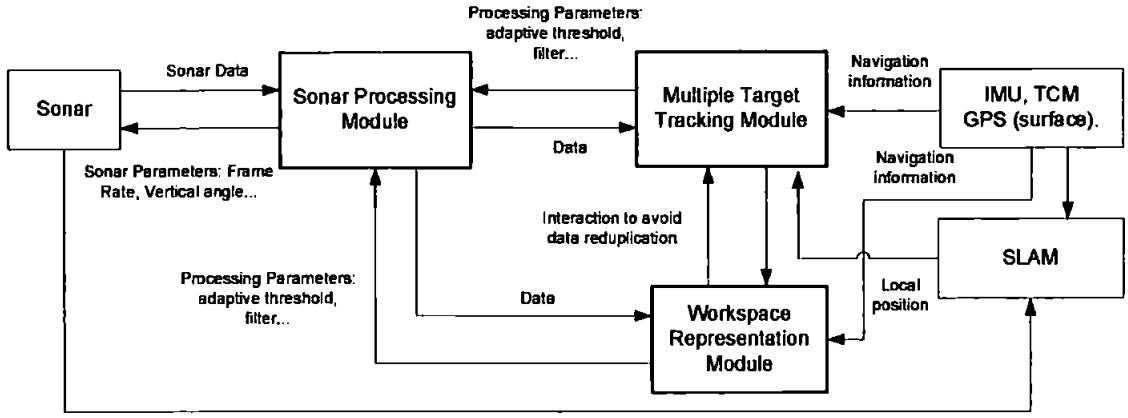


Figure A.1: A block diagram of a generic collision detection unit of an AUV.

In essence, each submodule plays an indispensable role of sustaining the overall functioning of an obstacle detection system. Sonar data processing entails processing of the sonar raw data, which is known for its aberrant, unpredictable and highly noise infested nature. On the other hand, the MTT submodule is needed for tracking important targets. Tracking allows the system to anticipate the movement of the obstacles, and to an extend, quantify the predictability of their motion which is in fact, intimately linked to the risk of collision. Indeed, this feature is highly beneficial for the motion planning of an AUV in a dynamic environment. Even static obstacles must be tracked, static obstacles with invariant parameter such as shape, are potential candidates to be employed as landmarks. The SLAM technique is capable utilising several reliable static landmarks for localised position calibration (Tomatis *et al.* 2001, Rikoski *et al.* 2002). The final module used is for workspace representation. The limited detection envelope of the sonar sensor necessitates a scheme to represent and memorise the environment structure for the purpose of navigation.

This chapter presents and proposes several techniques to be employed in sonar data processing and workspace representation. It commences with descriptions of the sonar transducer, sonar data processing, occupancy grid for workspace representation and finally ends with quasi-simulations using real world data. It is highly unfortunate that in the aim to ensure that the following study remains in a manageable scope, one will not consider the design of the MTT submodule. The MTT is a rather challenging subject in its own right, and has been pursued vigorously by both industry and academae alike. MTT techniques are used predominantly in radar and sonar applications. Avid readers are recommended to solicit the following books by Black-

man (1986), Bar-Sharlom and Li (1995) and Blackman and Popoli (1999) for further details. All the experimental results presented herein, were acquired via the AT500 sonar. The heading measurement was provided by the TCM-2, tilt-compensated magnetic compass. In practice, AUVs are not able to access the GPS positioning data when submerged. For this reason, a SLAM system is essential for the recalibration of the position of an AUV with reference to a local map in a short term basis.

A.1 The AT500 Sonar

The AT500 is a type of multi-beam formed, forward looking sonar especially designed for use in an obstacle detection task for an AUV. A detailed specification of the AT500 can be seen in Appendix 2. Figure A.2 indicates some important parts of the AT500 sonar.

Figure A.2: The AT500 sonar and its important parts.

Fundamentally, the sonar produces a 'ping', and the echoes (reflections) are captured by the receiver. The 'ping' is at a frequency of 500 kHz. The strength of the echoes

(amplitude) are located into a 8 bit resolution bin. The time of the echoes return corresponds to the distance of the detected object. The returned time is mapped into a discrete bin which has a resolution of 8 bit. It must be noted that all this sensing are done in polar coordinate. The beam formed characteristic of the AT500 allows one to consider the sonar as producing 64 beams with a overall horizontal coverage of 60° (Fig A.3). During the 'pinging' process, the sonar must be inhibited (blinded) for a brief period of time to avoid detecting ping as a spurious echo returns. This feature is achieved in the sonar hardware via a time varying gain (TVG). This 'blind' (blank) time is a function of the duration and strength of the transmit pulse. Currently, the sonar has a $1/s$ frame rate, although this might be altered in the future according to the manufacturer. The AT500 has two main user parameters, the gain and the vertical beam angle. The gain pertains to the sensitivity of the receiver, increasing the gain will extend the sonar capability to detect smaller objects at further range but at the expense of increasing noise. The vertical beam angle can be configured in real-time either for a 10° or 20° setting. In the experiments, only a 20° setting is employed. It is important to note that the larger the vertical beam angle, the higher the detection volume. As a result, a small object has less tendency to disappear from the sonar detection envelope. However, if the AUV is travelling near to the sea bed, then a false detection due to seabed reverberation will likely to increase.

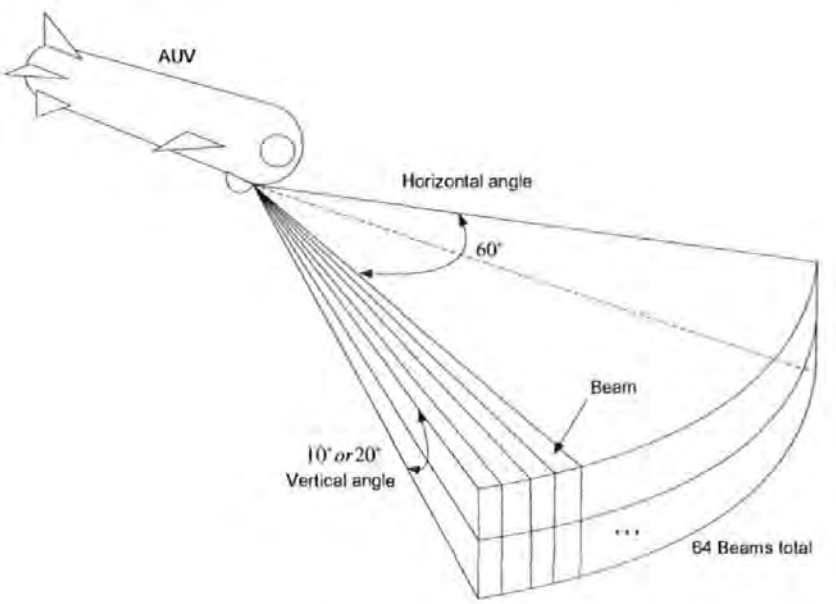


Figure A.3: AT500 sonar detection envelope

A.2 Multi-Path Reflection and Reverberation

Two undesired phenomena that occur when using underwater sonar are multi-path echoes and reverberation (echoes scattering) (Newman and Durran-Whyte 1998). Multi-path echoes are induced when the ping is deflected by adjacent objects in such a way that it returns after travelling several paths (Fig A.4). The sonar will receive simultaneously the target echo and that of its image or images. This image, befittingly, called a 'ghost', is actually non-existent in the physical world. The real returned ping of the object is typically termed as the 'principal reflection'. The consecutive echoes that occur after the principal reflection are multi-path reflections. The cause of this phenomenon can be attributed to the structure or shape of the environment. Likewise, it can also be induced when AUV is travelling near to the surface or seabed (Chevalier 2002). Discrimination of images from real targets is difficult at best, hence to be safe, it is advised to assume 'ghosts' as potential obstacles instead.

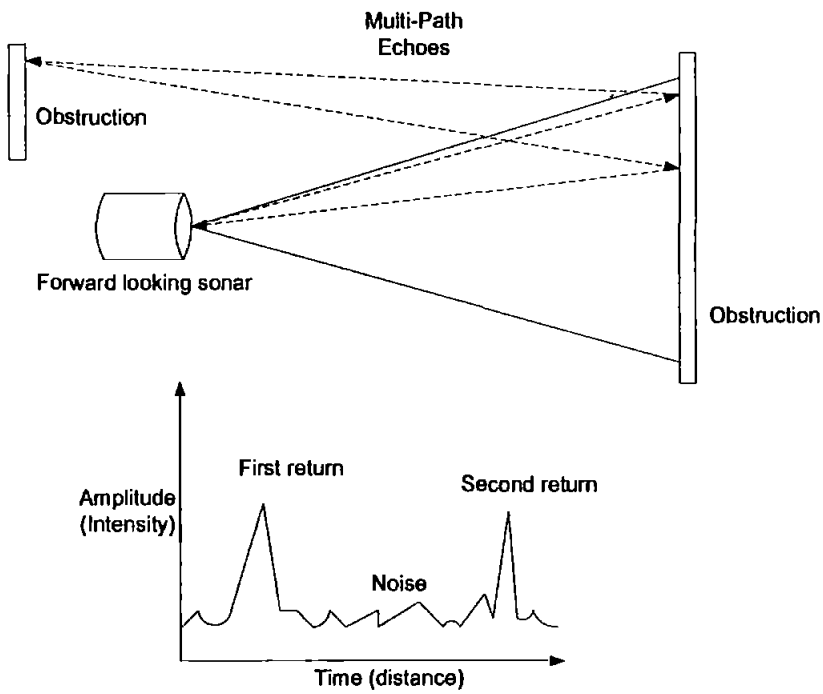


Figure A.4: Multi-path reflection phenomenon

Reverberation, also known as the clutter in radar terminology, occurs when the ping hits an object that has a non-uniform texture surface. The surface tends to scatter

the returning acoustic signal in multiple directions, invoking significant noise in the sonar data. Figure A.5 illustrates a scenario when the reverberation persists. This problem can be mitigated, to an extend, by passing the sonar data through a low pass filter such as median or Gaussian. Clearly, the severity of the reverberation depends primarily on the type of the environment. Surroundings that have more texture such as coral reef and an undulating sea-bottom will increase the occurrence of this phenomenon.

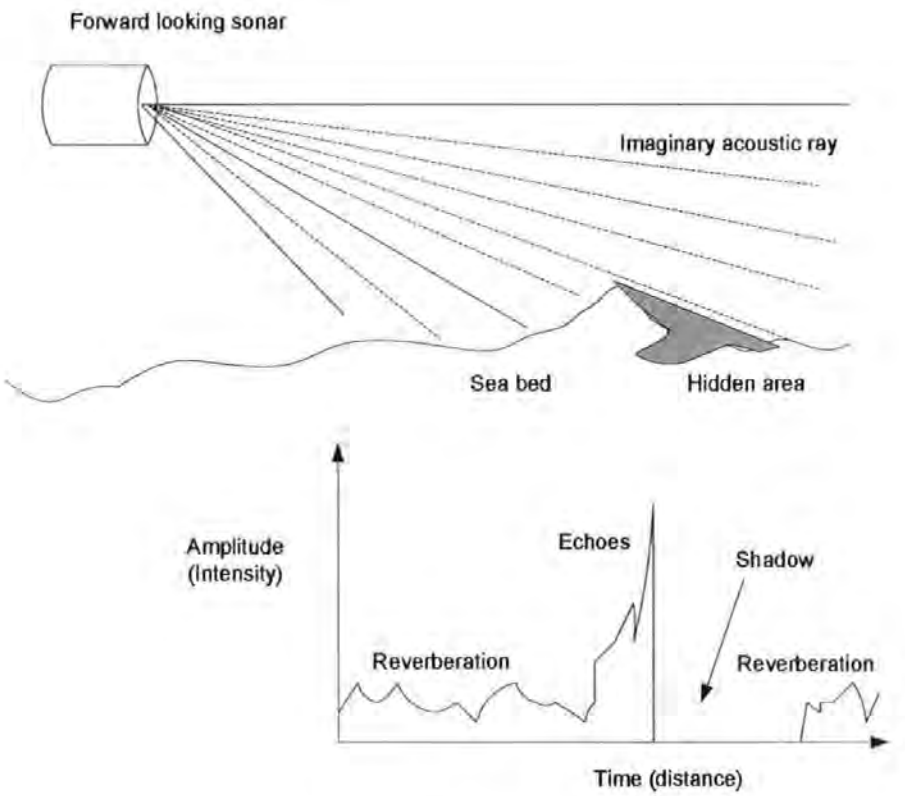


Figure A.5: Simplistic diagram illustrating the side view of a sonar observing a scene and the resulting time intensity return (Tena Ruiz 2001)

An experiment was carried in the University of Plymouth's laboratory tank to evaluate the characteristic of the sonar. Dimensionally, the tank is 2.5 m in length, 1.5 m in width and 0.6 m in depth (Fig A.6(b)). The image returned by the sonar in polar coordinate form is depicted in Fig A.6(a) with each alphabet corresponds to a particular item or lack thereof. The objective here is to assess if the sonar can detect the wall of the tank and also a PVC pipe of diameter 0.15 m, located approximately midway between the sonar and the wall. In this experiment, the sonar was set to 5

m range and a gain of 15. Again, referring to Fig A.6(a), it is clear that label (a) does not coincide with any actual physical objects. Object (a) is a 'ghost', created by multi-path reflection of the walls. Object (b) is also a combination of reflection from tank bottom and the large protruding pipe. With careful observation, a ghost can be discriminated from the actual target by altering the position and heading of the sonar. Evidently, the data is rather noisy, this is one of the prevalent characteristics pertaining to data acquired from a sonar.

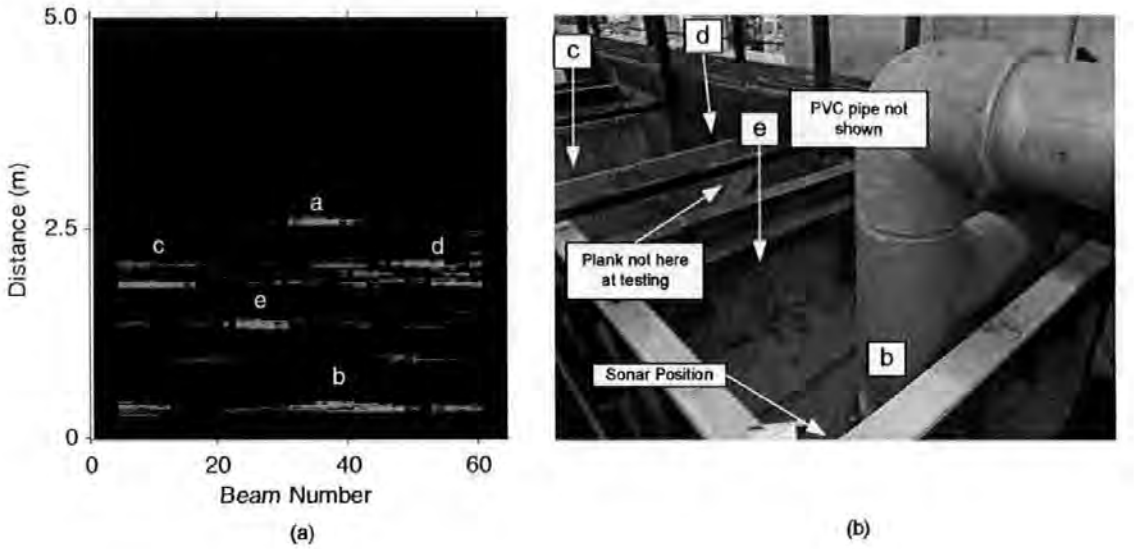


Figure A.6: (a)Image acquired by the sonar (b)The experiment facility

A.3 Sonar Data Processing

The main function of the sonar data processing module is signal-peak detection. It is defined as a quantitative way to discriminate the signal from noise (Kolawole 2002). The underlying paradigm here can be construed as a way to find an appropriate threshold, one above which noise pulses seldom rise and below which signal pulses seldom fall. However, prior to applying the threholding process, certain auxiliary processes must be introduced to prepare the data such that it has a higher chance of separating the signal from the noise.

It has been mentioned before that the approach adopted by this thesis is based on the context of an image processing methodology. The sonar data processing consists

of several steps. A complete flow chart of the procedures required are shown in Fig A.7. The first step is image acquisition, this has been achieved by employing a sonar in this case. The second step, involves the preprocessing of the image. A filter is introduced to attenuate the noise, improving the signal-to-noise ratio and the chance for correct signal and noise discrimination. The next stage deals with image segmentation. Image segmentation partitions the data into its constituent parts, categorising the signal and eliminating the noise according to a predefined threshold. Subsequently, a morphological process termed as dilation is introduced to improve the size of the detected target to accommodate the effect of the AUV dynamics, sonar and navigation sensor inaccuracy. Since the data is still in the native polar coordinate form, transformation to Cartesian is compulsory, in order to be compatible with other navigation data. Transformation is then introduced to scale and rotate the data to facilitate workspace representation. Finally, the data is sent to the workspace representation module for processing.

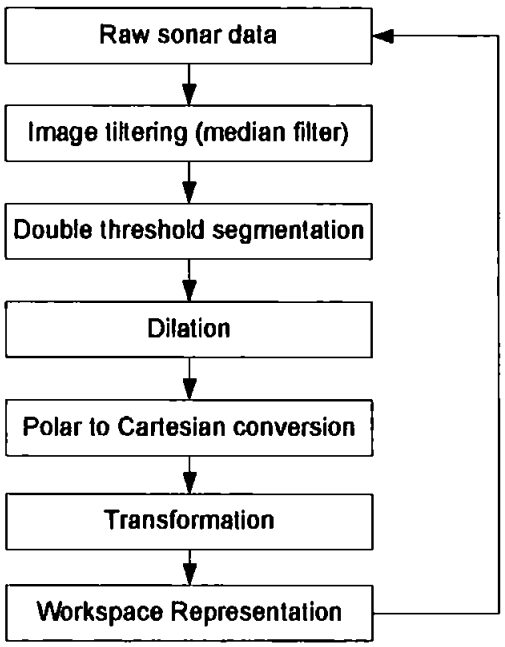


Figure A.7: Sonar data processing flow chart

A.3.1 Signal and Noise

Herein, the attention is drawn to understand the nature of signal and noise that exist in sonar data. The noise in the sonar can be decomposed into two dominant types. The first one is sensor noise (endogenous) and the other one is reverberation, as induced by the environment (exogenous). A sonar receiver inherently generates 'thermal' noise, the noise is clearly discernible when the gain is amplified or when the sensor gets warm. The sensor noise can be approximated by an exponential distribution (Toomay 1989). When a returned signal is received, the convolution between the signal (pulse type with mean S) and the receiver noise creates a signal which has a Gaussian distribution at mean S (Fig A.8). Consulting Fig A.8, the crux of the concept here is to design the transducer such that the S is always much greater that N so that the distortions that result from operating close to the zero point will be negligible. The signal transmitted can also be distorted in the form of scattering by the environment. The power of this clutter or scattering may distort target echoes by multiple folds, and is commonly a dominant noise in the sonar when the AUV travels near the sea bottom. This noise changes from region to region making estimation of its characteristic nontrivial.

One can also refer to Fig A.8 to understand more about the probability distribution of the noise in the sonar context. Here one will use the term detection probability, P_d and false alarm probability, P_{fa}. Ideally, P_d should be maximised while P_{fa} should be minimised, but without resorting to redesigning the sonar, one is permitted only to set the threshold so that arbitrary amounts of noise would be rejected. This implies that the P_d and P_{fa} are fixed for a particular x, increasing P_d will also increase the P_{fa} and vice versa, so a trade-off between the two must be met.

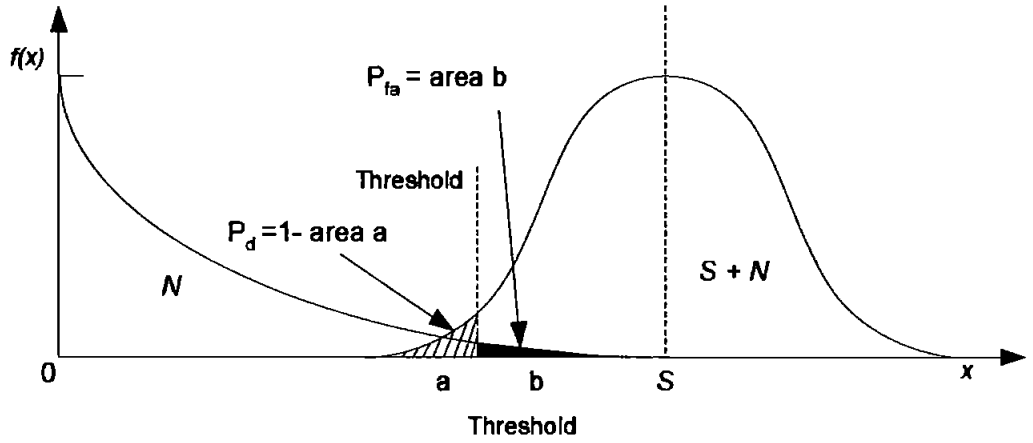


Figure A.8: Probability of detection and false alarm probability

A.3.2 Filtering

Filtering is normally the first operation applied in image processing. The objective here is to massage the data such that noise are attenuated to allow better signal-to-noise ratio for latter signal extraction processes. Two common ones are the median and Gaussian filters. The median filter evaluates each pixel in the image in turn and looks at its nearby neighbour, ranks them and decides whether or not it is representative of its surroundings. The Gaussian smoothing operator is a 2-D convolution operator that uses a Gaussian 'bell-shaped' kernel (Petrou and Bosdogianni 1999).

Using the data acquired from the experimental tank, one can observe in Fig A.9(a) that the raw data is corrupted with noise. The noise has certain characteristics, they are of a high signal, scattered, impulse type. Several filters were applied to the data to check their effectiveness. It was found the median filter with size $[3 \times 3]$ is superior as shown in Fig A.9(b). The outcome of using Gaussian filter with size $[3 \times 3]$ and $[7 \times 7]$ is shown in Fig A.9(c) and (d). The median filter provides better filtered data and simultaneously preserves the signal. Tena Ruiz (2001), however, suggested using a Gaussian filter owing to the computational saving. This is true, as sorting and ranking is a time consuming process. On the other hand, the small size of the median filter and the data used in this study did not create any drastic computational burden worthy of concern. Hence, the median filter was employed for the remaining simulations.

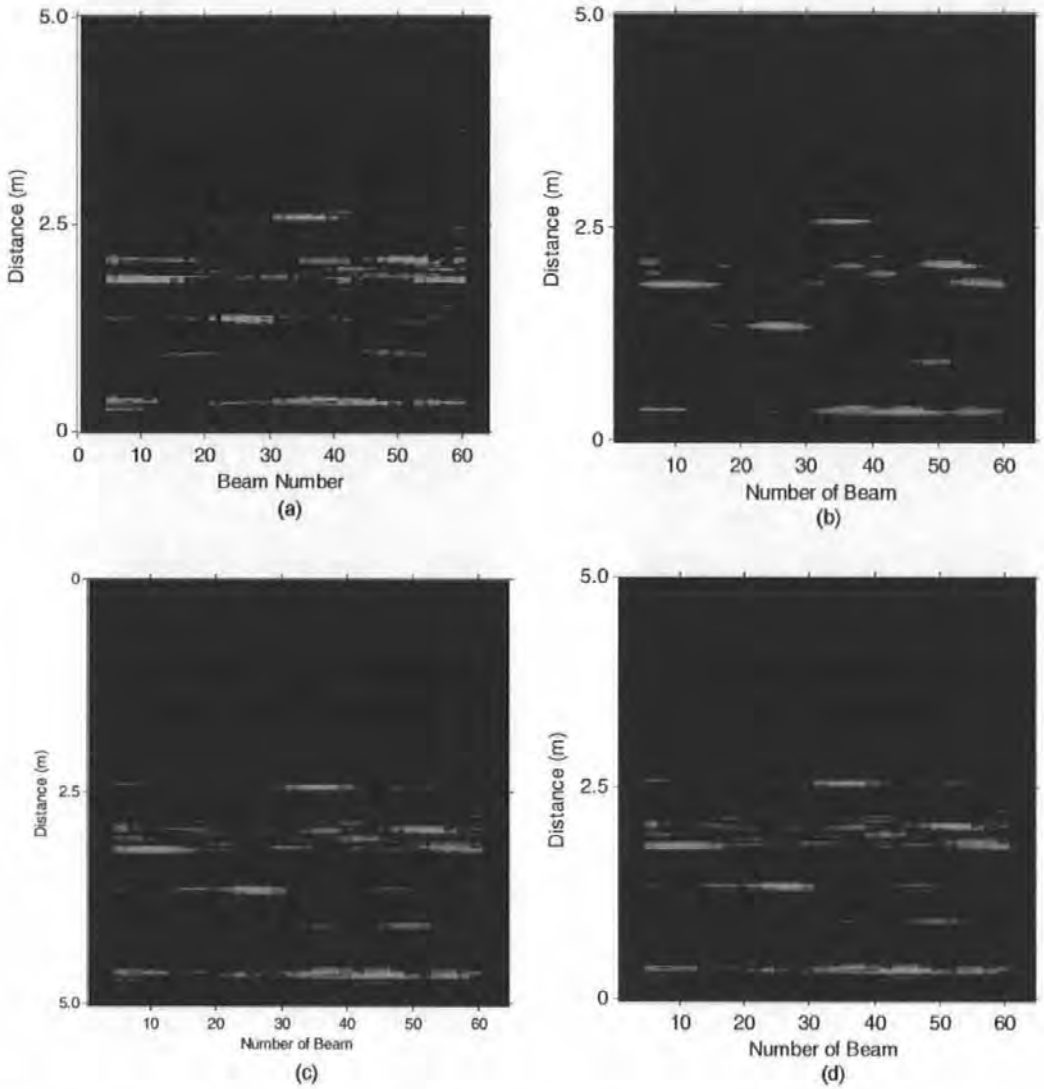


Figure A.9: Effect of applying different filters (a)Sonar data (b)Median [3 × 3] (c)Gaussian [3 × 3] (d)Gaussian [7 × 7]

A.3.3 Segmentation (thresholding)

Once the data has been filtered, thresholding will be applied to extract the signal from the noise. Thresholding is a form of segmentation technique, and its effect can be global or local. The simplest thresholding functions by making all the values above the predefined threshold one and setting the lower value to zero, in other words, it separates the data into two regions. It also converts the data into a binary form, hence an alternative name for this process is binarization.

The threshold is normally fixed *a priori* if the performance characteristic of the sonar and environmental effect is known. It can be adaptively change by analysing the histogram. A more robust double threshold algorithm is advocated by Tena Ruiz (2001) and also applied by Pettilot *et al.* (2001).

The idea is to partition the histogram into three regions using two thresholds, $t1$, high threshold and $t2$, low threshold, instead of two regions as in the simple threshold algorithm. It is at this instance one is allowed to segment the region in between the two thresholds, assuming that some of the returns are connected to the region $> t1$. An illustration demonstrating a double threshold application to a simple 1-D curve is given in Fig A.10.

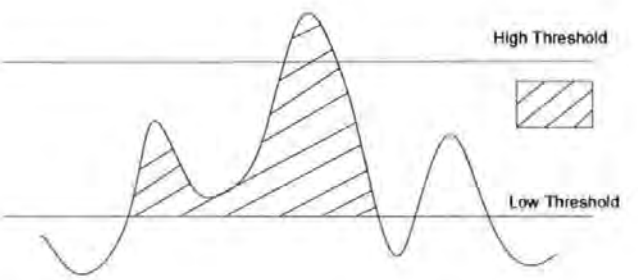


Figure A.10: Example of application of the double threshold algorithm to a simple 1D curve. The regions selected by the algorithm are sectioned.

Using the previous sonar data from the tank experiment, a test was conducted to check the performance of various thresholding algorithms. The histogram of the filtered data was studied to provide a starting point of selecting $t1$ and $t2$. Figure A.11(a) shows the filtered version and (b) the unfiltered version. Note that the median filter eliminate significant amount of noise at intensity 0 to 100. The results of the segmented data are displayed in Fig A.12. The neighbourhood of the double threshold algorithm was set to $[3 \times 3]$, $t1 = 130$ and $t2 = 60$. The single threshold algorithm uses only one value for segmentation. It is apparent, that the results of a double threshold is superior compared to the others. The objects (pure white) are properly segmented. Figure A.12(c) shows a case of threshold value set too low, resulting in noise being categorised as objects. Conversely, when the threshold value is set to high, the objects break up and are not distinct (A.12(d)). The actual segmented data is in binary form, white and black, but the noise is included here for visualisation.

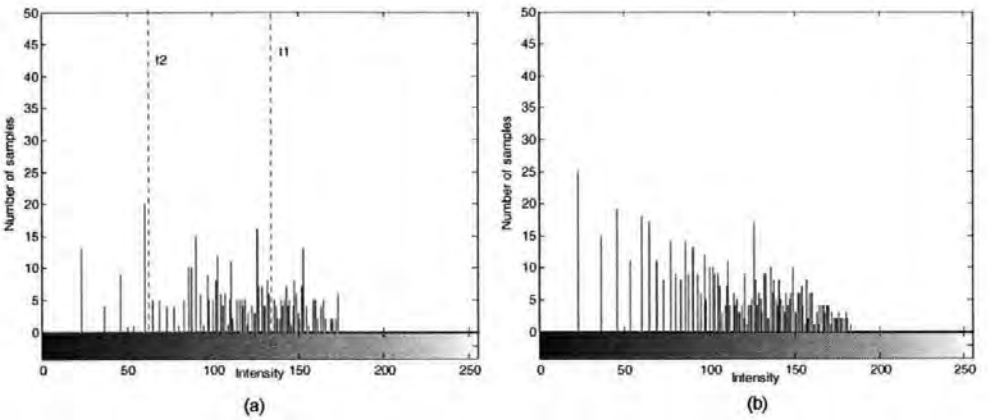


Figure A.11: Image histogram (a) Median filtered (b) Not filtered

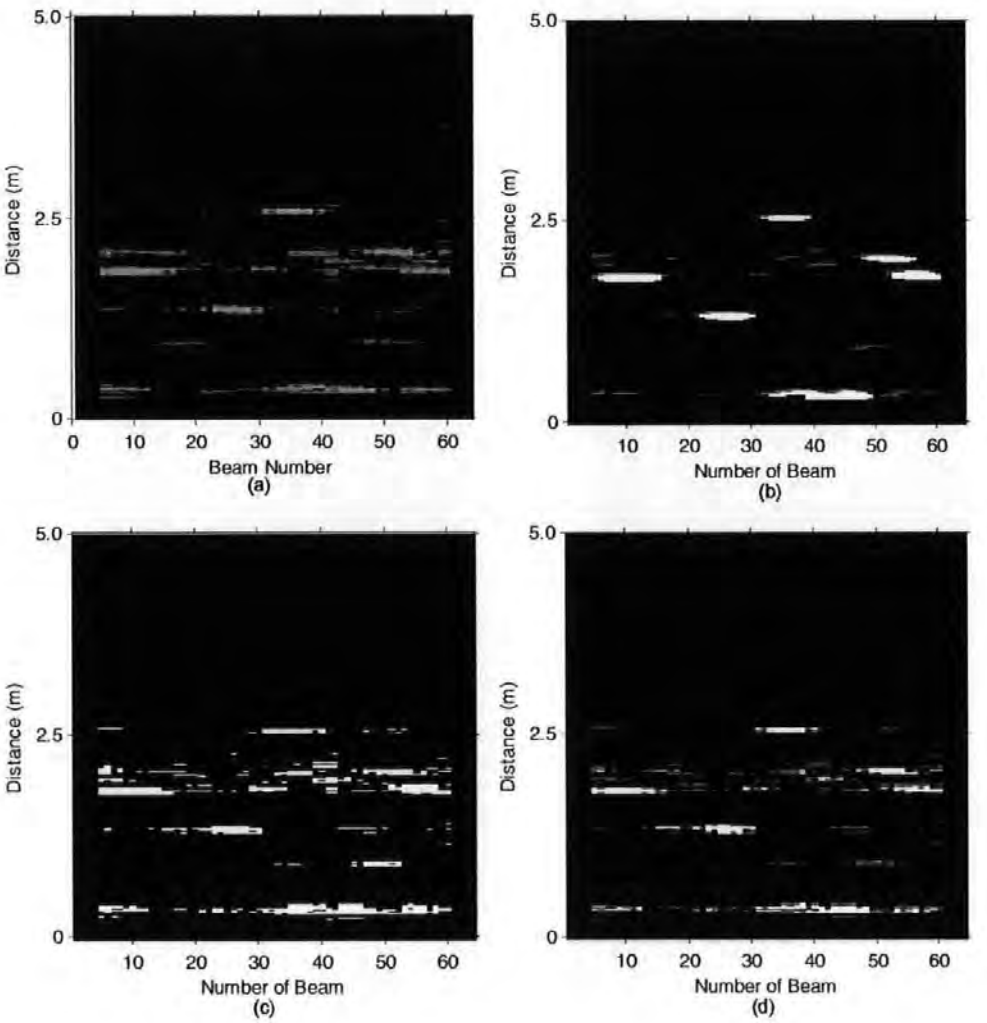


Figure A.12: Type of different thresholding methods (a)Sonar data (b)Double threshold (c)Low threshold($t = 60$) (d)High Threshold($t = 130$)

The current, state-of-the-art, thresholding algorithm used by the military is of the adaptive version. One version, is known as the Cell-Averaging Constant False Alarm (CACFA) algorithm (Drumbheller and Lew 2001). It utilises the mean, variance or median of the neighbourhood cells to determine the local threshold value, and maintains a constant false-alarm rate. This algorithm is rather computationally intensive, but will be an interesting topic of research. In must also be remembered that unlike a stationary sonar, the AUV is constantly travelling and its displacements, headings, and environment are unpredictable, therefore the clutter mapping feature commonly used in fixed scanning sonar cannot be employed here.

A.3.4 Dilation

Dilation is one of the two basic operators in the area of mathematical morphology, the other being erosion. The basic effect of the operator on a binary image is to enlarge gradually the boundaries of regions of foreground pixels. Thus areas of foreground pixels grow in size while holes within those regions become smaller (Giardina and Dougherty 1987).

It was found that without the dilation operation, some of the targets are very small when mapped into Cartesian coordinates, especially those nearer to the receiver. This distortion effect is implicated by the native polar coordinate to cartesian mapping. Slight changes in angle (polar coordinate) will effectuate a large discrepancy in the Cartesian displacement for object located further from the sonar. This distortion is aggravated the further the target from the sonar. The noise in navigation data can create jumps in the heading measurement, making positioning the target in workspace representation rather difficult. By increasing the apparent size of the target, this allows them to overlap easily, and is useful in incrementing the value of the occupancy map.

A structuring element of a disc with a radius of 5 pixel was found to be adequate for this purpose. Fig A.13 shows the before and after effect of applying dilation operation. Note that operation is applied to the polar coordinate data.

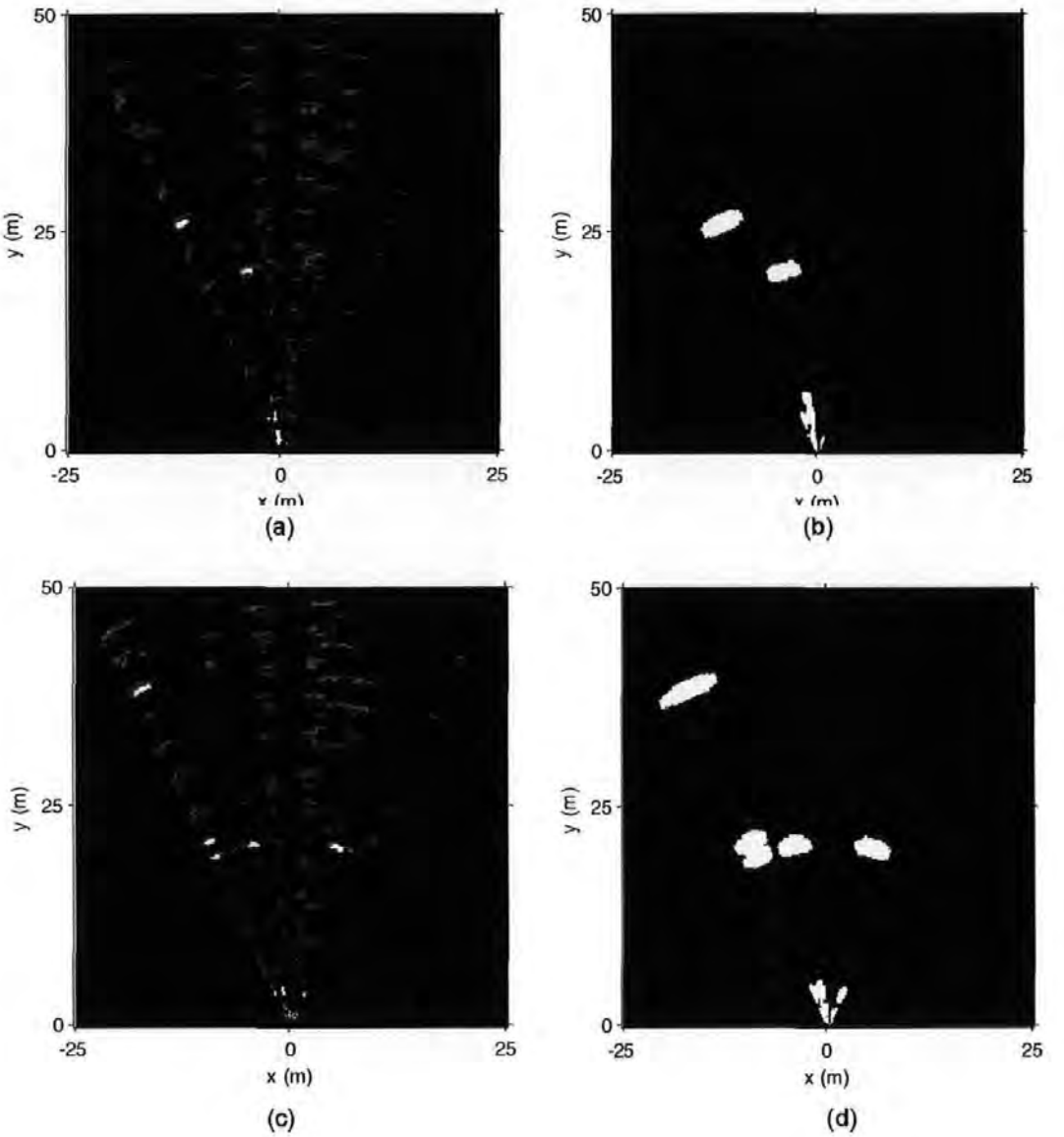


Figure A.13: Dilation operation on sonar data (a)Not dilated Frame 13 (b)Dilated Frame 13 (c)Not dilated Frame 15 (d)Dilated Frame 15

A.3.5 Coordinate transformation

Sonars similar to radars operate natively in polar coordinate. It is a convention that an area map uses a Cartesian coordinate as this is more intuitive for the human but is also more compatible with other navigation measurements. Therefore, this coordinate must finally be transformed to a Cartesian form for further post processing.

The transformation introduce several effects to the data, first the Gaussian noise distribution is converted into a Rayleigh distribution. Secondly, interpolation for required at certain cells, causing distortion to the data at times. The process can be computationally intensive. Two popular methods for polar to Cartesian coordinate conversion are the polar formatting algorithm and the inverse transformation.

The one preferred by industry is based on polar formatting algorithm. It is very accurate, less distortional to data, but computationally intensive. It is based on the concept of the 'Fourier space of the scene being imaged' and the fact that the phase history data represents a surface in this space. The surface is then projected onto a plane, where it is resampled to a rectangular grid for easy processing by 2-D FFT algorithms (Wahl *et al.* 1996). Due to the high requirement demanded by the military radar and sonar applications, dedicated digital processors for this task have been implemented.

Another simpler algorithm, yet still very popular, where accuracy is less important is the inverse transformation. The crux of the idea is that instead of mapping each bin in the polar scan to a cell in the Cartesian plot, one proceeds to map each cell in the Cartesian coordinates to the polar coordinates and decides which value it should take. One can increase the accuracy of the neighbourhood cell by applying sub-pixel interpolation. A simple pseudocode of the algorithm is given below:

1. Set the size of the Ccartesian plot (image size)
2. Find angle: $\tan^{-1}(y, x)$ and range: $\sqrt{x^2 + y^2}$
3. Check the value from polar plot for angle and range and put it x and y
4. Go to the next cell
5. Repeat 3 to 5 until the all the cells are calculated

In implementation, the use of transcendental function is very demanding for computers, so a simple way to abrogate this calculation is to use a table. All the values are mapped into tables and a look up routine is used. Significant improvement is observed if the data is large. Even with a size of 256 with 256 pixels, a 10 times speed improvement was recorded. Figure A.14 shows an example of image from a

polar coordinate being converted into a Cartesian coordinate format. Notice that the shape of the object changes with the conversion.

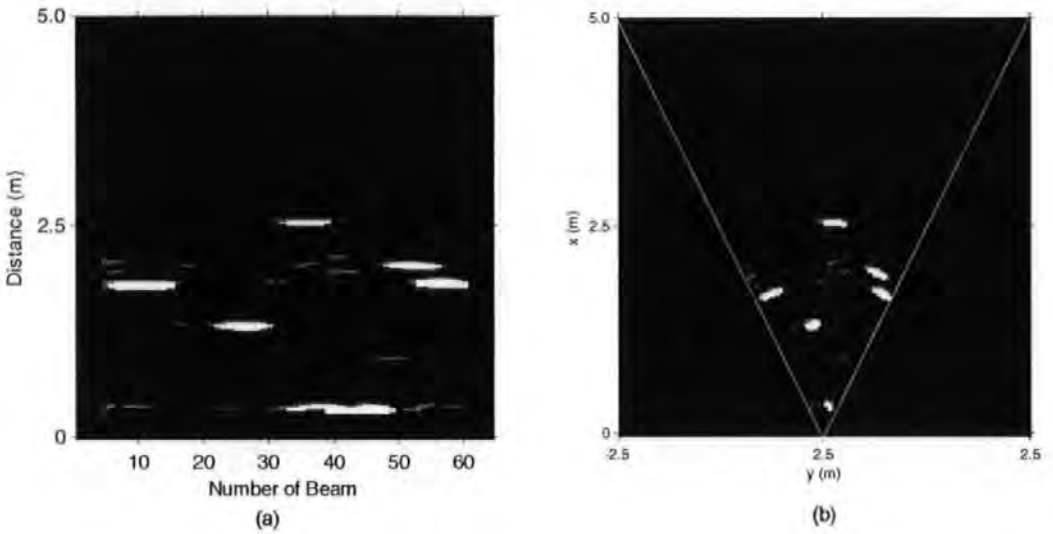


Figure A.14: Polar-to-Cartesian coordinate image conversion (a)Polar form (b) Cartesian form

A.3.6 Transformation

Linear transformation is a common operation applied to image processing. Here the sonar data is down scaled to a $[50 \times 50]$ size using the nearest neighbour algorithm. This means that each pixel represents a 1 m cell in the real world and gives more than adequate resolution and accuracy for collision avoidance purpose. Scaling operations can also use bilinear and bicubic algorithms. Although they are more accurate and computational intensive than nearest neighbour algorithm, they have not been adopted since their high accuracy is not required herein. Finally, the image is then rotated and displaced according to NED convention in order to be aligned with the workspace grid.

A.4 Sea Trial Experiment

This section describes the experiment conducted at the Coxside site of the University of Plymouth. The location recorded by the GPS with reference to the WGS84 datum, is $N50°21.952'$, $W004°07.922'$. The objective of the experiment is to scan the area, store the data, process and use it for environment mapping. It is also a test to assess the performance and feasibility of the sonar to be used for collision avoidance purposes of AUVs. Figure A.15 and Fig A.16 depict the panoramic view of the designated area for the experiment. The targets are labelled in the figure alphabetically. The location of the sonar can be seen in Fig A.15 with label (a). Since, it is the usual convention to use NED coordinate system for internal workspace representation of the AUV, a conceptual area plan of the targets is illustrated in Fig A.17 for the sake of clarity.

The trial was achieved by dipping the sonar using a special made fixture into the sea to a depth of 0.5 m. The TCM-2 compass is fixed to the pole such that the heading can be acquired. The sonar scanned by rotating it manually, starting from approximately North-West to South-East and back again to North-West. The sequence was recorded by the software, tagged with corresponding heading data (Table A.1). The vertical beam of the sonar was set to 20°, the gain to 18 and the range to 50 m.

Figure A.15: Sea trial location with sonar position shown

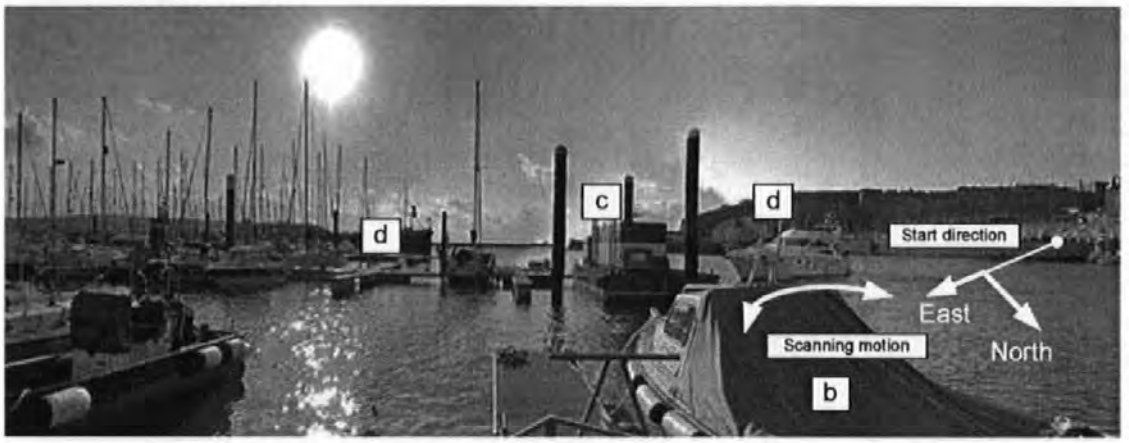


Figure A.16: Sea trial location

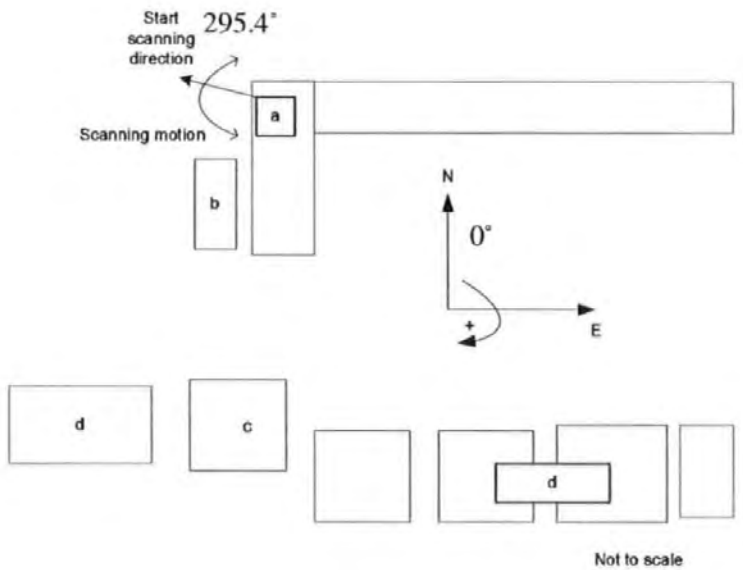


Figure A.17: Sketched area plan in NED coordinate

Sequence	Heading (°)	Sequence	Heading (°)
1	295.4	17	169.8
2	286.9	18	173.9
3	274.4	19	192.5
4	262.6	20	212.6
5	251.8	21	232.3
6	236.0	22	242.7
7	223.3	23	250.8
8	211.9	24	251.9
9	203.6	25	261.7
10	199.1	26	274.1
11	197.0	27	286.4
12	188.1	28	295.2
13	170.9	29	305.8
14	151.1	30	311.6
15	135.8	31	319.2
16	150.3	32	324.7

Table A.1: Sonar frames and their corresponding heading for the Coxside trial

It was found that the data extracted from the trial contained anomalies at the further and nearest bin of the range (Fig A.18(a)). This was later informed by J&S Marine Ltd that this is an inherent defect of the sensor, causing it to produce high values for certain cells. Since this error is a persistent fault and in fixed location, the data is eliminated by forcing it down to zero, assuming no detection. The corrected data is shown in (Fig A.18(b)).

Figure A.19 shows the histograms of Frame 15 and Frame 16 of the collected filtered data. The threshold setting, $t1$ and $t2$ were tuned to the values 154 and 90 respectively. This setting, achieved heuristically, was a good compromise between a reasonable detection probability and a false alarm probability. Figure A.20 reveals the difference between unprocessed, raw sonar data and the processed (segmented) version. The targets are clearly defined for the latter version.

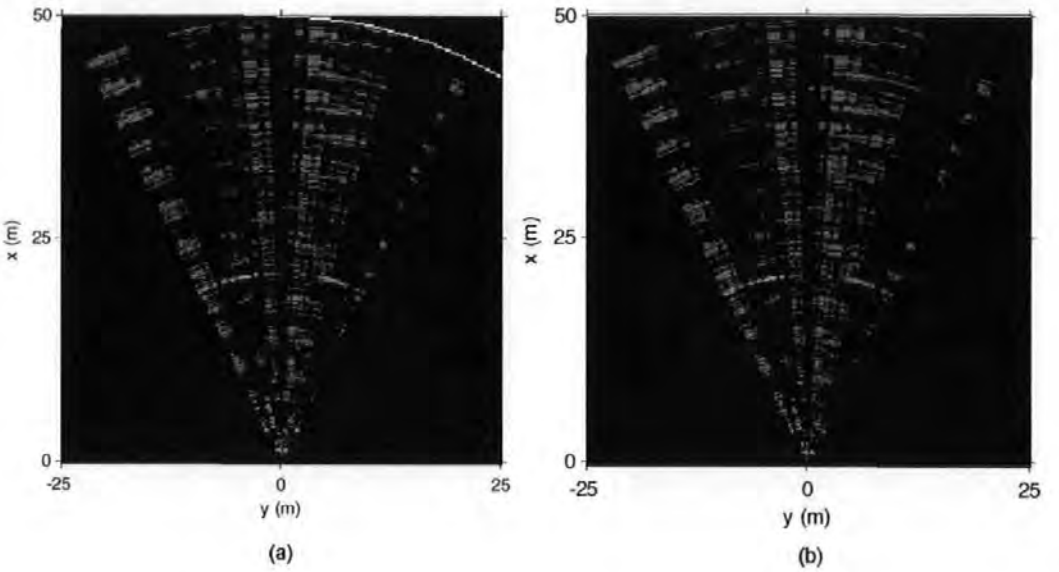


Figure A.18: (a)Sensor defect in upper and lower detection range (b)Corrected data

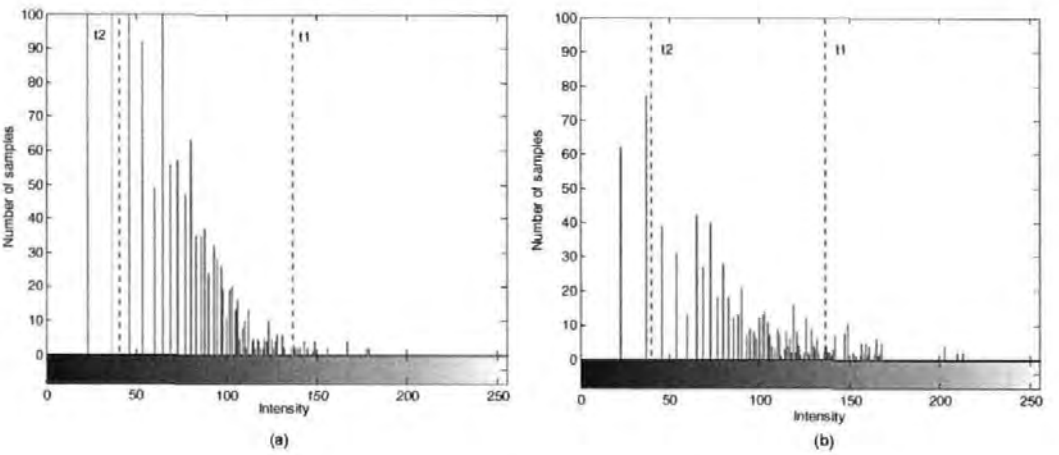


Figure A.19: (a)Trial histogram for Frame 15 (b)Frame 16

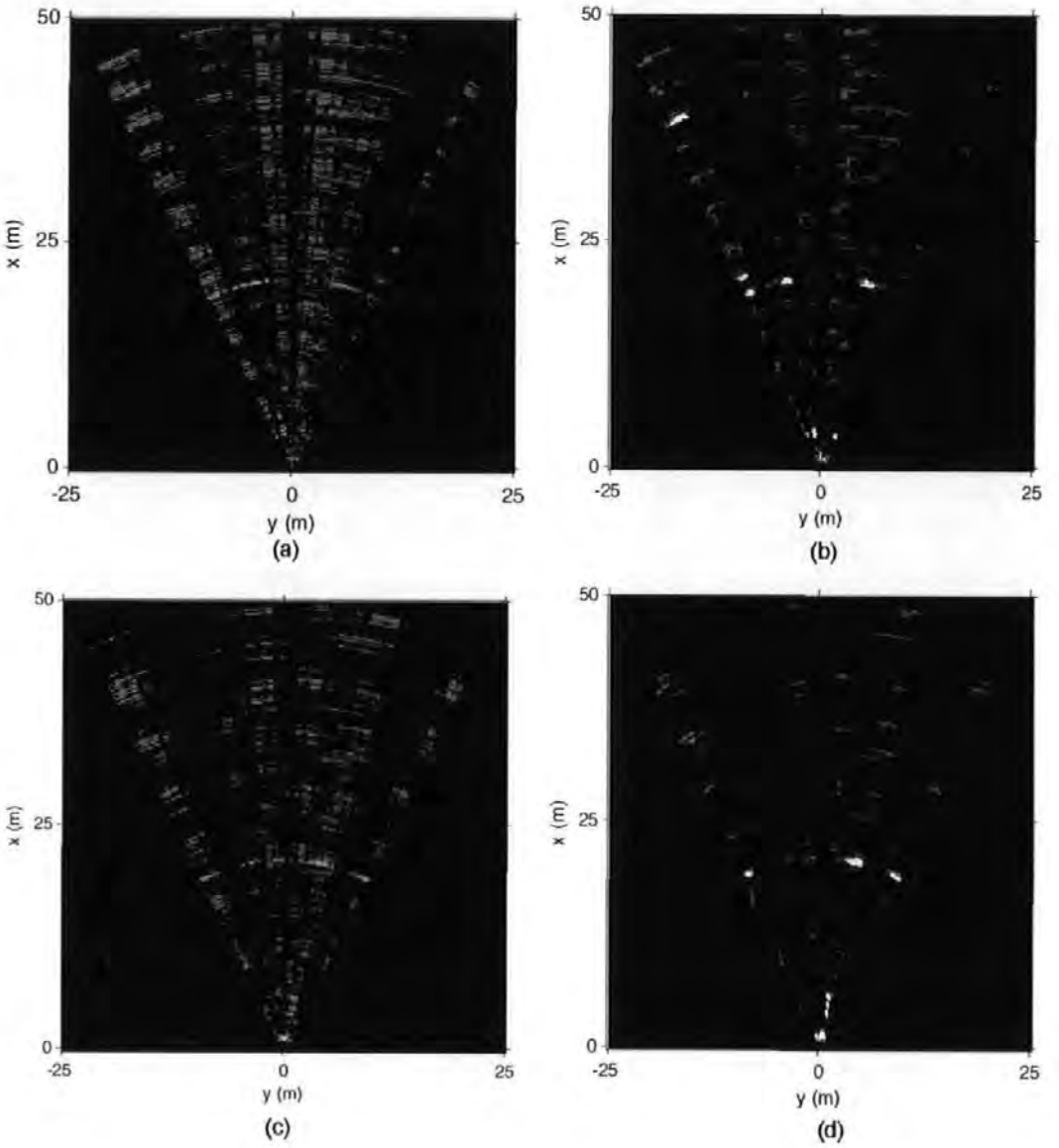


Figure A.20: (a)Unprocessed data Frame 15 (b)Unprocessed data Frame 15 (c)Processed data Frame 16 (d)Processed data Frame 16

A selected sequence of the processed frames with their associated frame number labelled below are presented in Fig A.21 and A.22. The rest of the frames are not shown because they do not contain any interesting targets. The median filter, size [3×3], and double thresholding algorithm with settings as mentioned above was applied to the frames. The pure white objects are the segmented targets. The noise in the frames are shown here only for clarity of exposition. It is apparent from this sequence, that the sonar data is rather aberrant and noisy by nature, sometimes displaying false targets and multi-path phenomenon.

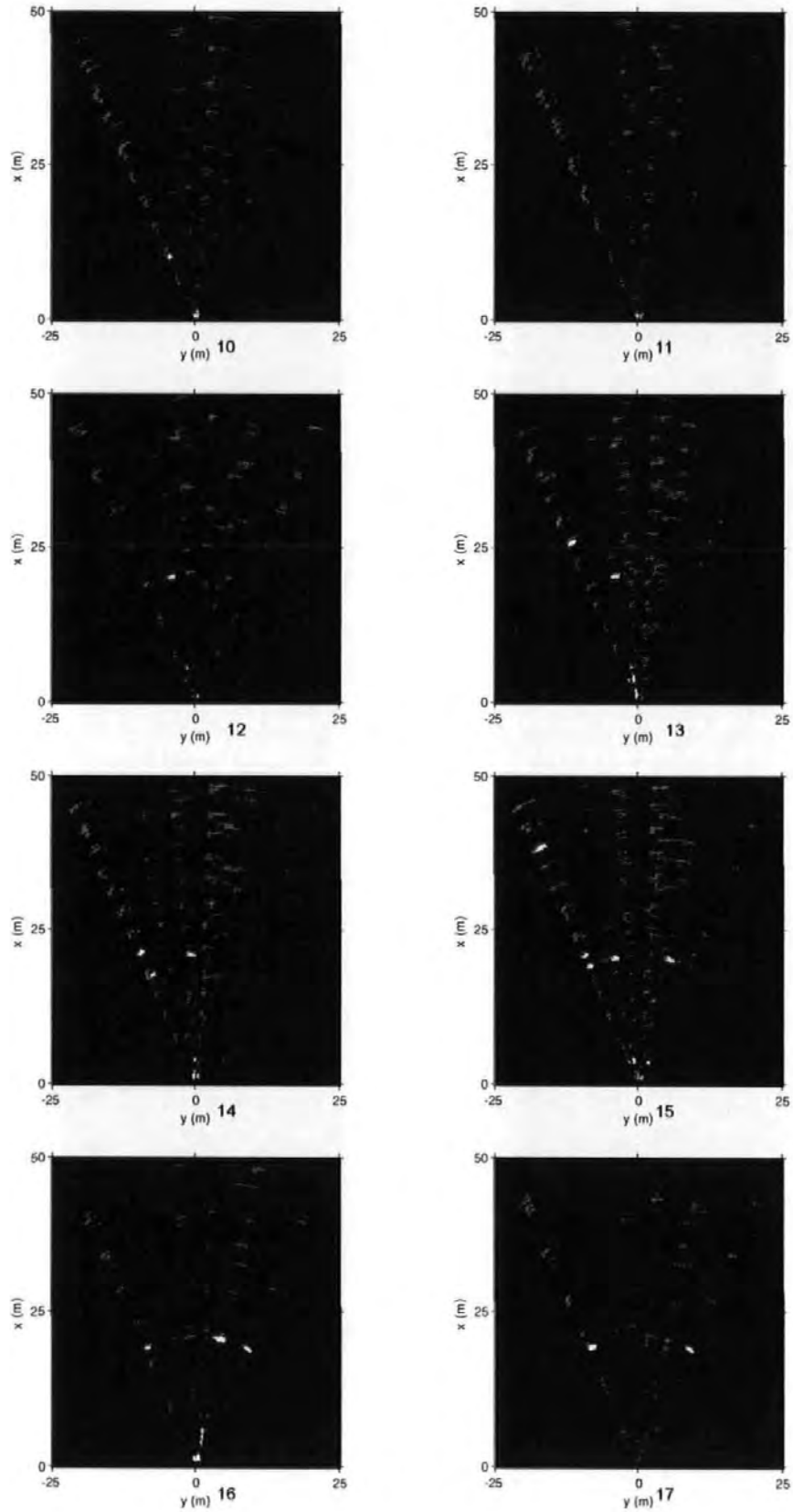


Figure A.21: Sonar sequence from 10 to 17

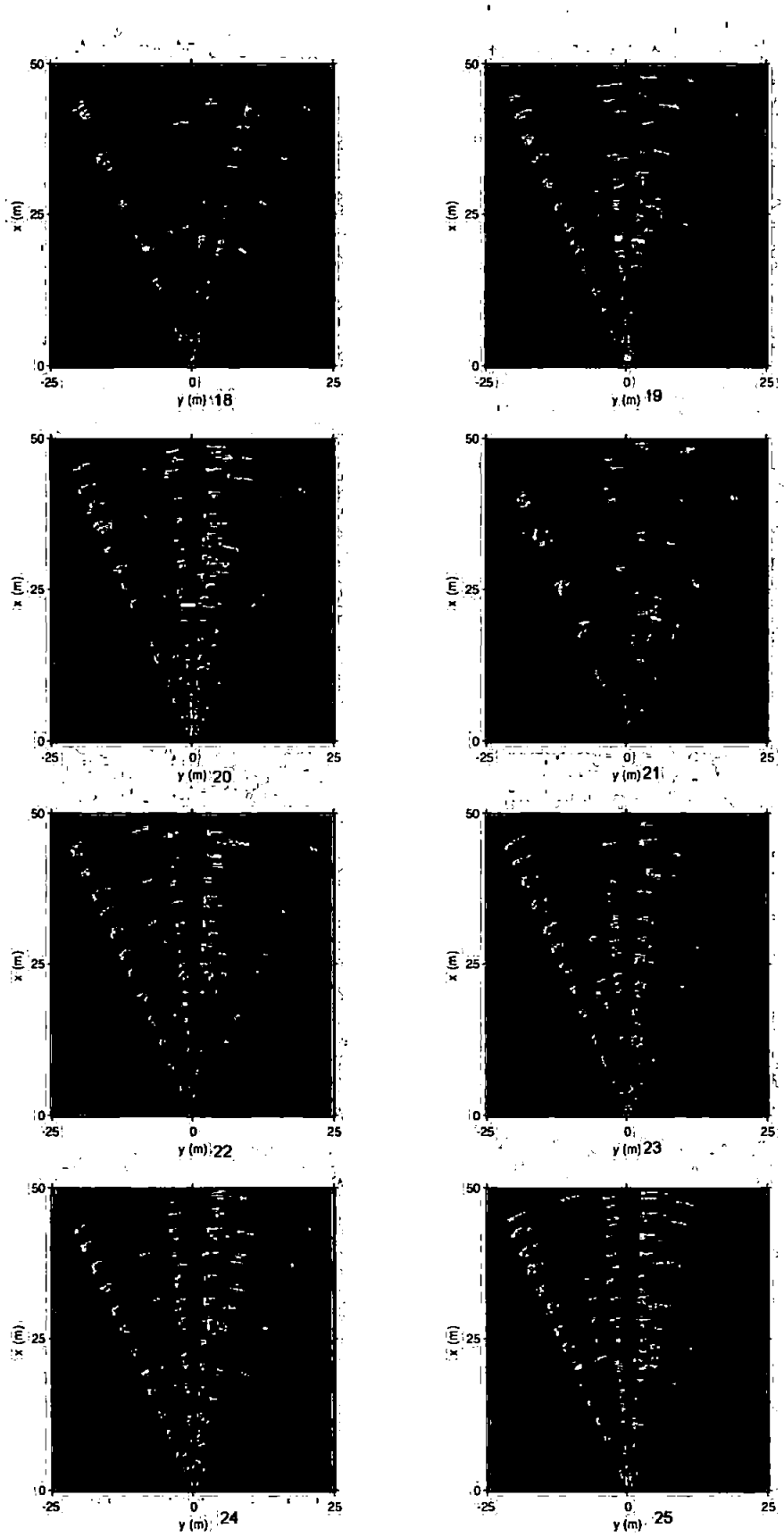


Figure A.22: Sonar sequence from 18 to 25

A.5 Workspace Representation

The data acquired from the sonar would not be useful unless it could be processed into a much compact and functional form. Moreover, the sonar measurements were incremental by nature due to the limited detection envelope. This necessitated a method to represent the environment for use in motion planning. A more detail exposition regarding workspace representation is provided in Subsection 2.3.4.

A.5.1 Occupancy map implementation

The approach here was to employ the occupancy grid, a variant of spatial decomposition scheme for workspace representation. In essence, an occupancy grid represents the environment space by partitioning it into uniform, non-overlapping grids or cells in a spatial lattice. Each cell can be allocated with user defined attributes such as confidence of obstacle presence, terrain geometry and safety factor (Movarec and Elfes 1985).

Figure A.23 shows the implication of applying the dilation operation on an occupancy grid. It must be understood that an occupancy grid functions by maintaining the cells that overlap. Note that without dilation, Fig A.23(a), a case of inadequate overlap, it creates a map with small targets. These targets have a tendency to disappear and reappear, behaving more like noise. The figure is labelled alphabetically that corresponds to the targets as shown in Fig A.15 and Fig A.16.

Figure A.24 shows the flow chart of the implemented occupancy grid. Unlike, Ridao *et al.* (2000) who used only pings that have not expired after a preset time interval are recorded into a coarser map, and used for path planning. In this study, an alternate approach was employed, upon detection of a new target, its cell was set to a heuristically determined, starting value of 6. This value was tuned according to the confidence of the probability of detection and probability of false alarm of a particular sonar. An excessive starting value can result in a system that is prone to accept noise as targets. A value too low will eliminate the target too quickly from the map, resulting an incomplete view of the surrounding environment.

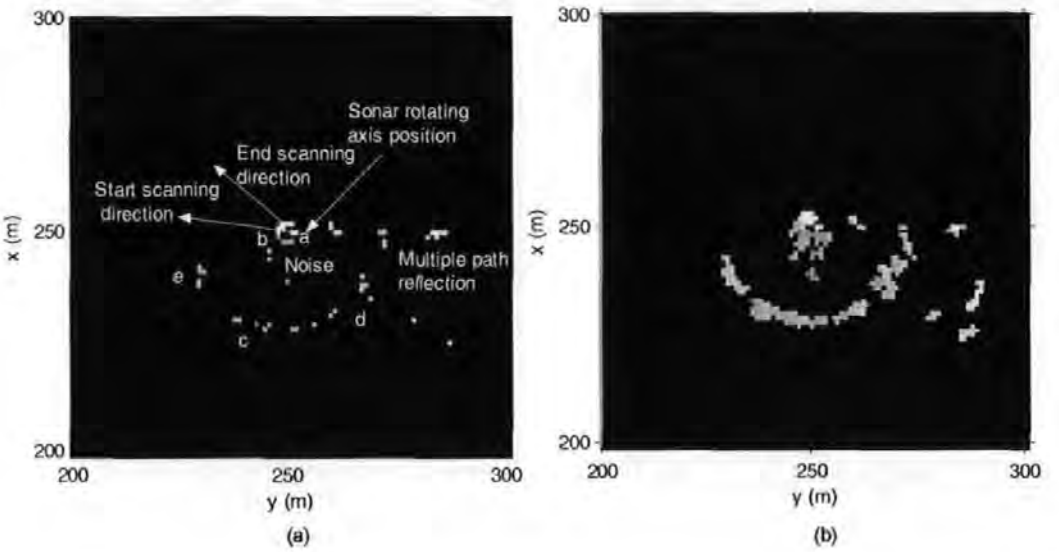


Figure A.23: Comparison between the effect of dilation (a)Non-dilated and (b)Dilated image

Also, instead of using a simple linear function to maintain the 'life' of the cells, herein, the algorithm used a quadratic function. This implies that any cell value will be squared if they overlapped with that sonar data. This produces an effect that frequently overlapped cells will be quickly preserved and maintained. Conversely, those that do not overlap sufficiently often their values are decremented by using a linear function, at every time step, one unit in every second. Note that a maximum bound of a value of 60 was set for this map. The value from the quadratic function, if not bounded would reach an excessively high value in a few scans. Such a high value tends to decay very slowly. This value must be set according to the accuracy of the vehicle navigation sensors. When submerged, it is necessary for an AUV to performs dead reckoning to navigate the environment. Sensor drift tends to cause the AUV displacement to be inaccurate. This inaccuracy will then propagate to the occupancy grid. AUVs that are equipped with SLAM technology are expected to have better performance in this aspect, therefore allowing a higher maximum occupancy grid value to be used.

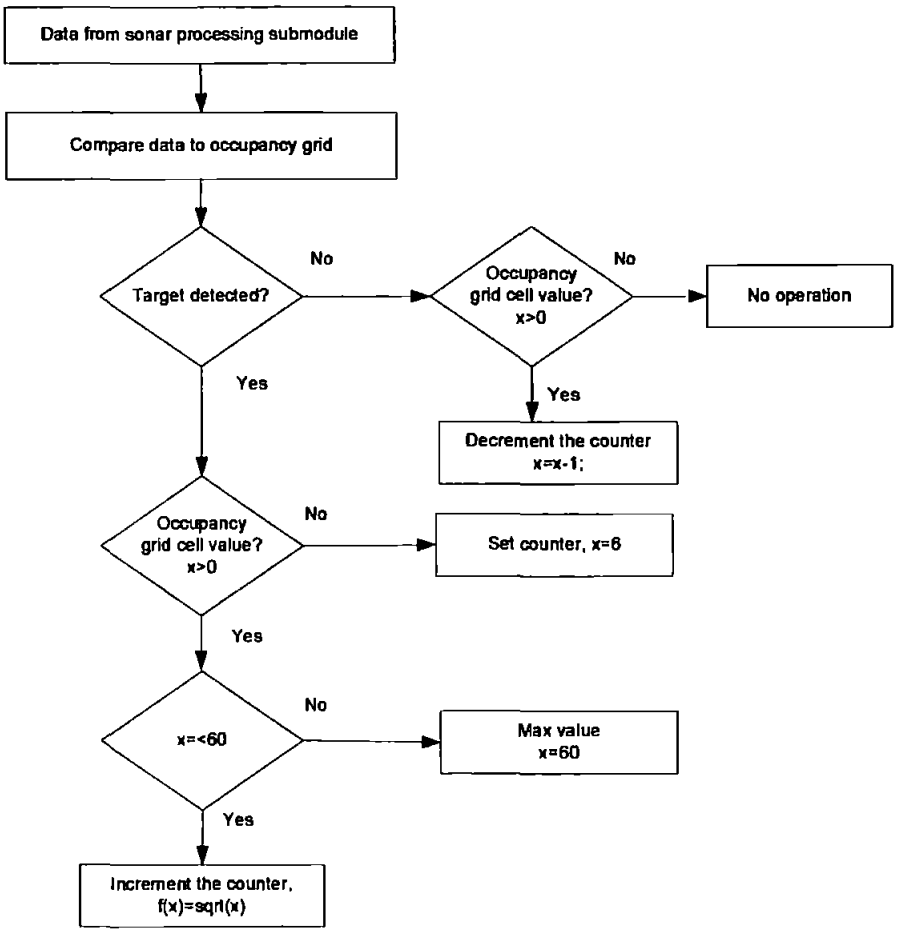


Figure A.24: Occupancy grid flow chart

A.5.2 Discussion

A selected sequence of the occupancy grid frames are presented in Fig A.25 and A.26 with their associated frame number labelled below. The white sector represents is the detection envelop of the AT500 sonar. The heading of the sonar for each frame is given in Table A.1. For ease of reference, the frame number coincides with the sonar frames given in Fig A.21 and Fig A.22. Again, one should consult the alphabetical labels in Fig A.15, Fig A.16 and Fig A.23 for better understanding of the targets position relative to the NED convention.

From Frame 10 to 17, one can observe that target (c) was detected, but the fast scanning motion of the sonar, achieved manually, did not allow sufficient time for the

target to be redetected, this explains its low value (light blue). Similarly, from Frame 14 to 16, target (d) was detected and its value was reinforced as the frame progressed (changing of the colours to light green). Notice that from Frame 12 to 16, several false alarms or targets caused by multi-path reflection and noise appear. Beginning with Frame 17, the direction of scanning of sonar was reversed. At Frame 18, target (c) was redetected, and its value increased, then it was continuously reinforced from Frame 18 to 25. Interestingly, a fortuitous event occurred, a vessel (target (e)) was manoeuvring into the harbour at that particular time. The vessel was detected by the sonar as shown in Frame 21, it is discernable as a small light blue blip at the furthest left of the frame. The occupancy grid manages to eliminate the multi-path reflection and incorrect noise signal discrimination, detected at Frame 12 to 16. These 'targets' were then suppressed after Frame 18. Although, impressive and practical, the occupancy grid only works well if the navigation sensors are reasonably accurate. Furthermore, without a tracking module, it is unable to predict the target velocity and trajectory, resulting in deteriorating performance in highly dynamic environment.

It was discovered that the maximum distance of AT500 sonar is 40 m and not 50 m as stated in the specification. Detection of a moving target is very difficult. The sonar is very sensitive to the exposed cross section of the target. When the boat was heading straight to the sonar, the detection is very weak, almost nil. But as the boat turns to be perpendicular to the sonar, detection probability increases, this can be explained by the kneel of the boat which provides sufficient large area to reflect the sonar acoustic energy.

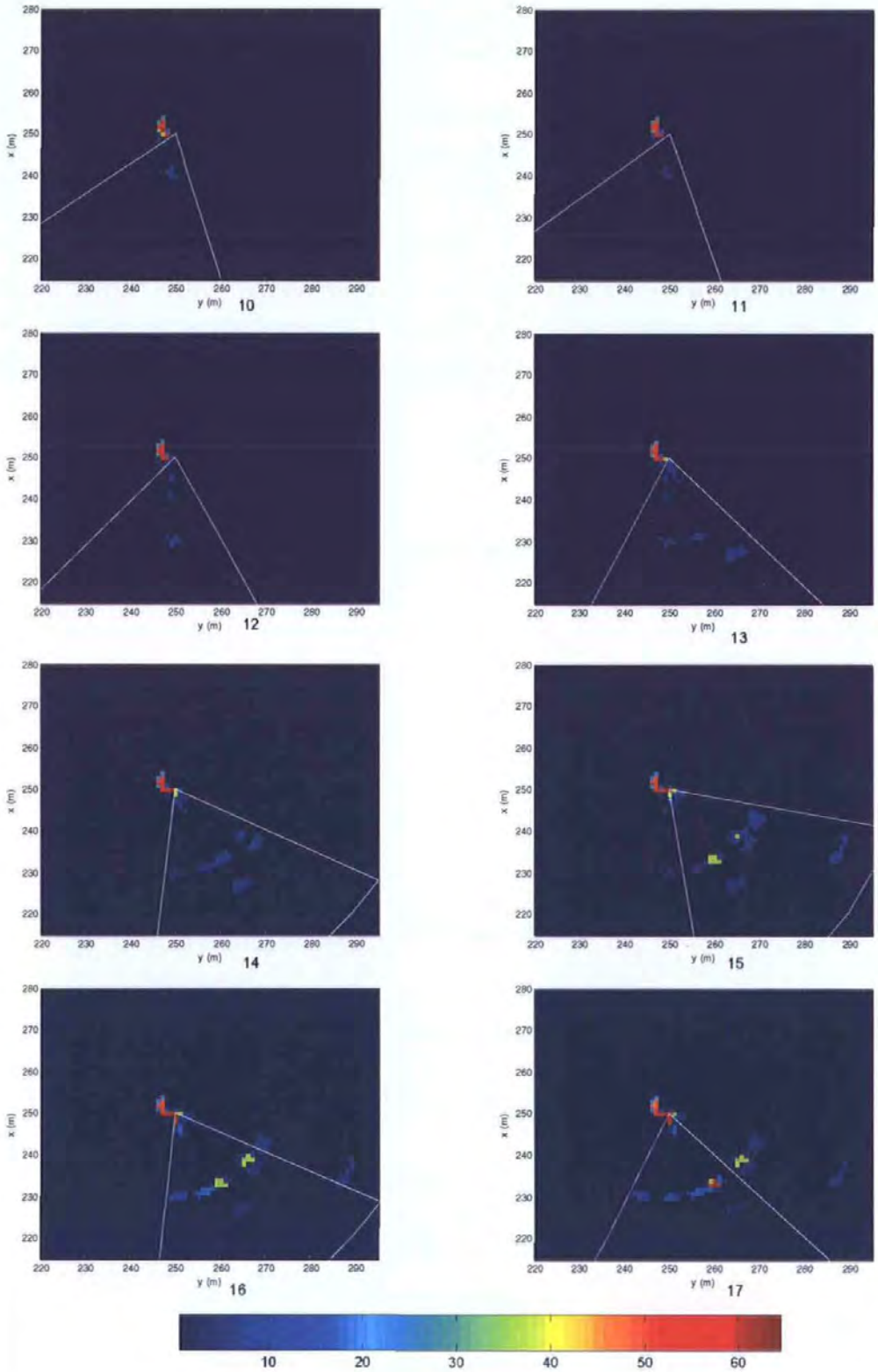


Figure A.25: Map sequence from 10 to 17

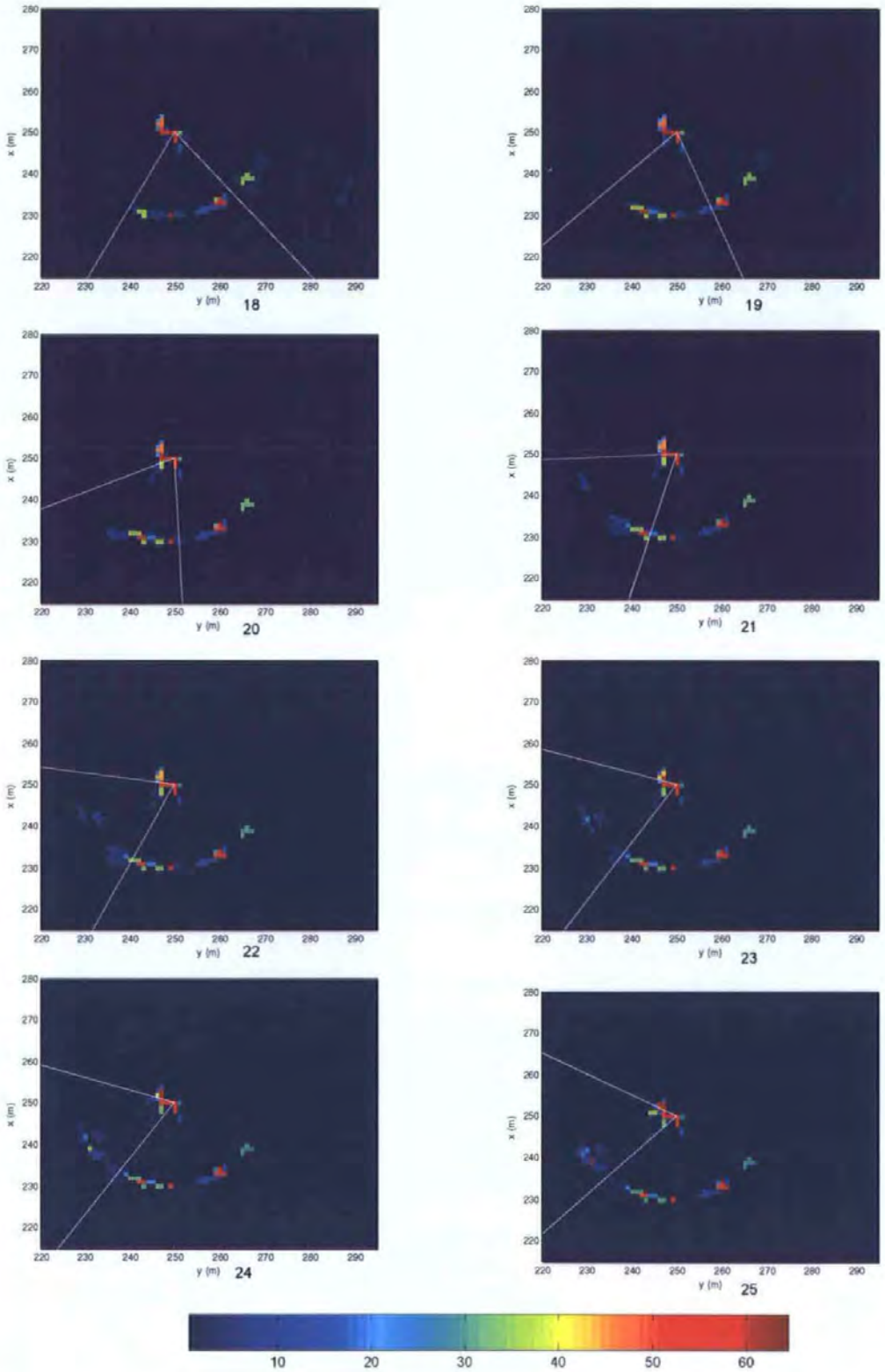


Figure A.26: Map sequence from 18 to 25

The final occupancy grid of Frame 32 was used for the following motion planning simulation (Fig A.27). Note that multi-path reflection and noise have been eliminated and what is left are the actual targets. The 3-D plot of the occupancy map is shown in (Fig A.27(b))

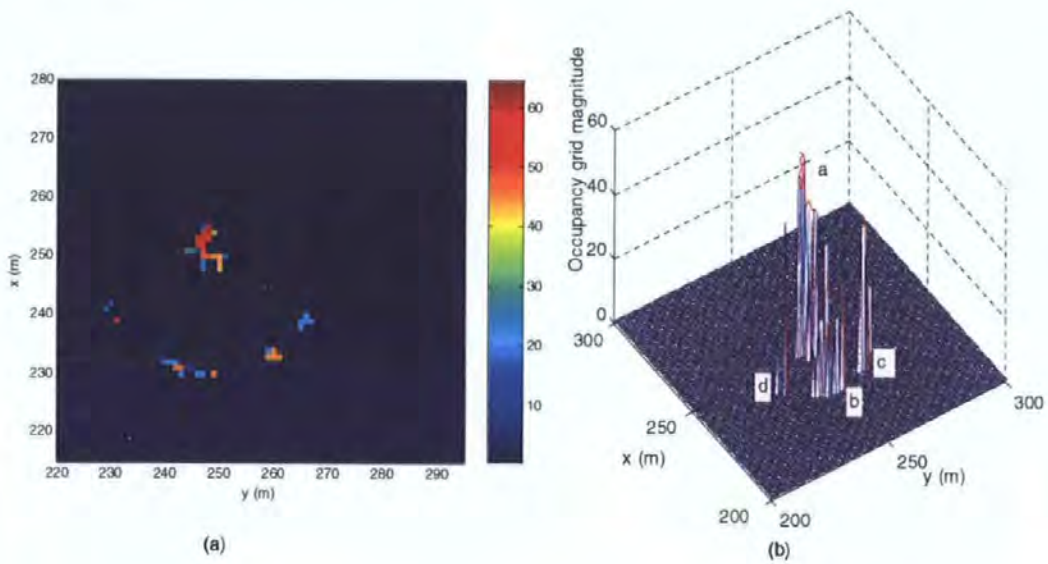


Figure A.27: (a) Occupancy grid of Frame 32 (b) 3-D plot of the occupancy map

A.5.3 Quasi-simulation of collision avoidance scenario

For these simulations, the sampling space was set to 250 m $\times$ 250m, as illustrated in Fig A.28. The remanding simulations were hosted in the MATLAB 6.5 /SIMULINK environment, on a 2.1 GHz Pentium IV machine, with 512 MB of RAM and running Windows XP.

Additionally, the assumptions required are outlined below :

- The AUV uses only the occupancy grid map of the Frame 32.
- The AUV has performed a scanning manoeuvres to scan the surroundings.
- The AUV model is of *AUTOSUB* and it uses the MA+RRT algorithm detailed in Chapter 5.

- The AUV are moving at a cruising speed of $2m/s$ and the initial position was set to [300 240 3.1] and the goal is [170 230 κ], according to the format $[x(m)\ y(m)\ \psi(rad)]$. κ is a variable. It should be worth mentioning that the goal, is an element from a collection of the mission waypoints.
- The occupancy grid uses NED convention.

The triangles shown in Fig A.28 and A.29 represent the *AUTOSUB* AUV, enlarged twice the original size to improve visualisation.

Discussions

Figure A.28 shows a near optimal trajectory found by the algorithm in 0.11 s, the AUV avoids the obstacles by moving through the gap in between them. This trajectory might not be a practical one, for a large AUV as the physical size of the AUV and its slow dynamics, consequently, the AUV has a high chance of colliding with the nearby obstacle. Nonetheless, this manoeuvre is ideally suitable for small agile AUV conducting clandestine and reconnaissance missions in a hostile territory. For the case of a large AUV, and to deter the algorithm from discovering 'risky' trajectory, a dilation operation can be introduced to enlarge the obstacles. This is in fact, similar to the concept known as configuration-space patching proposed by (Lozano-Perez and Wesley 1979).

Figure A.29 depicts a similar scenario but using different seeds of the quasi-random generator. The trajectory was found in 0.46 s. This time, the AUV turns away, in an aggressive manner from the obstacles. Then, it executes a considerable large detour before arriving to the target. Qualitatively speaking, this is obviously a safer trajectory for the *AUTOSUB* AUV, owing to its size and slow dynamics. As expected, the trajectory total distance when compared to the former simulation trajectory is substantially larger, hence this provoked the question of energy efficiency.

Here, one can notice the consequence of dynamics quantisation, which restricted the admissible dynamics of the original system. A trade-off between computational efficiency, computational tractability and performance degradation that one must be willing to pay for the adoption of this approach. Secondly, the AT500 sonar, owing to

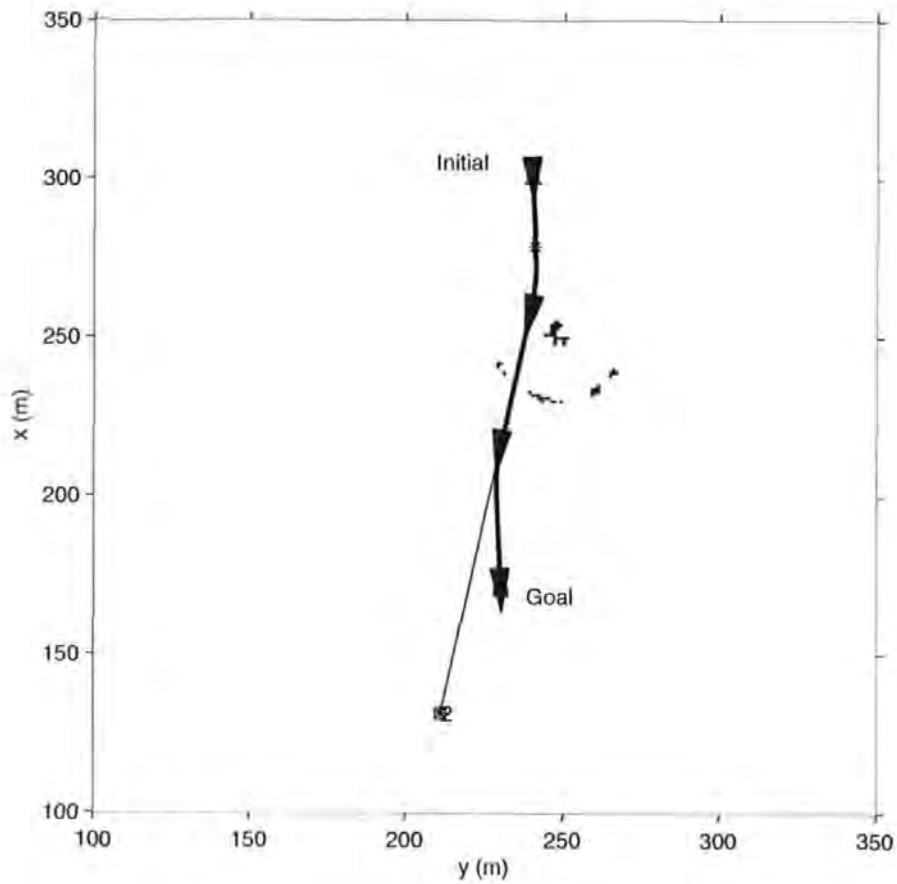


Figure A.28: Collision avoidance simulation 1

its limited detection range of only 40 m, is not suitable for large AUVs, with sluggish dynamics. Their minimum turning radius are too large, and by the time an obstacle is detected and confirmed, it will already be too late for any pre-emptive action to be taken.

A.6 Concluding Remarks

This chapter presented the development of the sonar data possessing and workspace representation subsystems of an AUV. The two submodules play an important role in obstacle detection. The sonar data from the AT500 sonar was processed in the context of image processing, and the occupancy grid was employed for workspace representation. An experiment was conducted at the Coxside site of the University of Plymouth, to gather real world data for the development of the aforementioned

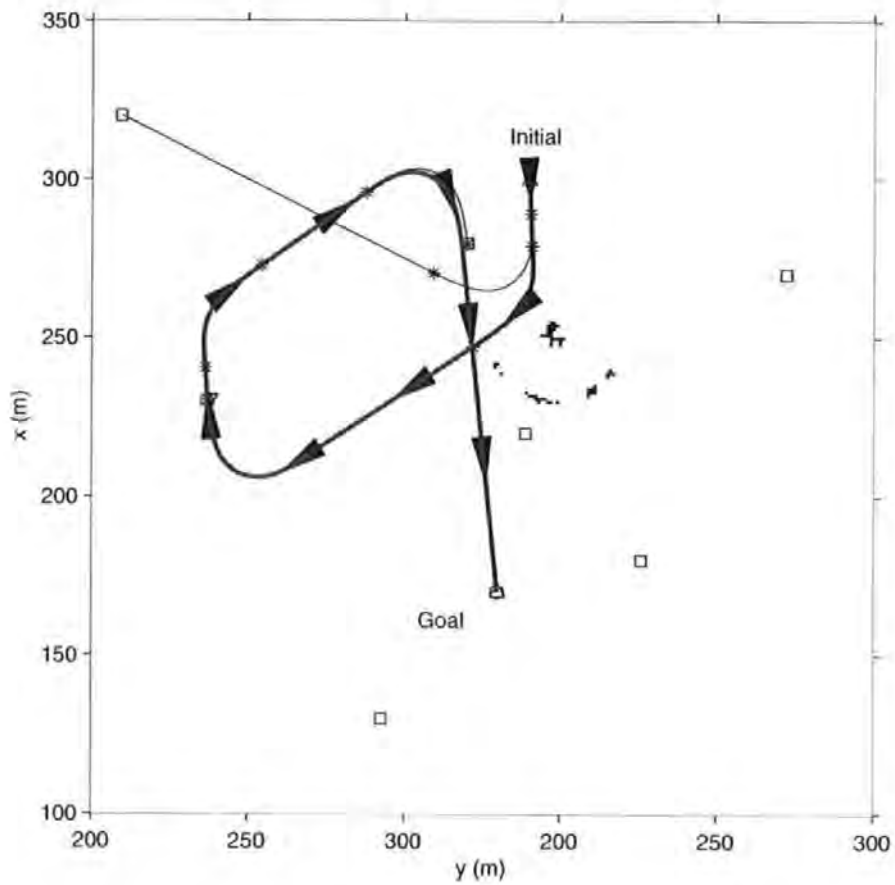


Figure A.29: Collision avoidance simulation 2

submodules. Since the AT500 is a prototype sonar, this chapter has documented original work on its application. The developed obstacle avoidance system is adequate for a static environment case. Later, the tracking module if designed, will result in a fully comprehensive obstacle detection system, that can be used in static and dynamic environments alike.

Appendix B

Publications

The work within this thesis has contributed significantly to the underwater research literature via the following list of publications. This includes all the papers which have either been published, accepted or under preparation.

Journal Papers

- Tan, C. S., R. Sutton and J. Chudley (2005). CRASH, BANG,AVOIDANCE: Collision Avodiance Systems. *Oceanology today* (5),22-23.
- Tan, C. S., R. Sutton and J. Chudley (2004). Collision Avoidance Systems for Autonomous Underwater Vehicles, Part A: Obstacle Detection. *Journal of Marine Science and Environment, IMarEST* (C2), 39-50.
- Tan, C. S., R. Sutton and J. Chudley (2004). Collision Avoidance Systems for Au- tonomous Underwater Vehicles, Part B: Obstacle Avoidance. *Journal of Marine Science and Environment, IMarEST* (C2), 51-62.

Refereed Conference Papers

- Tan, C. S., R. Sutton and J. Chudley (2005). Quasi-Random Manoeuvre-based Motion Planning Algorithm for Autonomous Underwater Vehicles. *IFAC 16th World Congress, Prague* pp. 483-488.

- Tan, C. S., R. Sutton and J. Chudley (2004). An Incremental Stochastic Motion Planning Technique for Autonomous Underwater Vehicles. *IFAC Conference on Control Applications in Marine Systems (CAMS 2004), Italy* pp. 483-488.

- Robinson, P., C. S. Tan and C. Morris and R. Sutton (2003). Low Power Intelligent Sonars for Autonomous Underwater Vehicle Navigation and Collision Avoidance, *In Proceedings of Underwater Defence Technology (UDT) Europe, UK.*

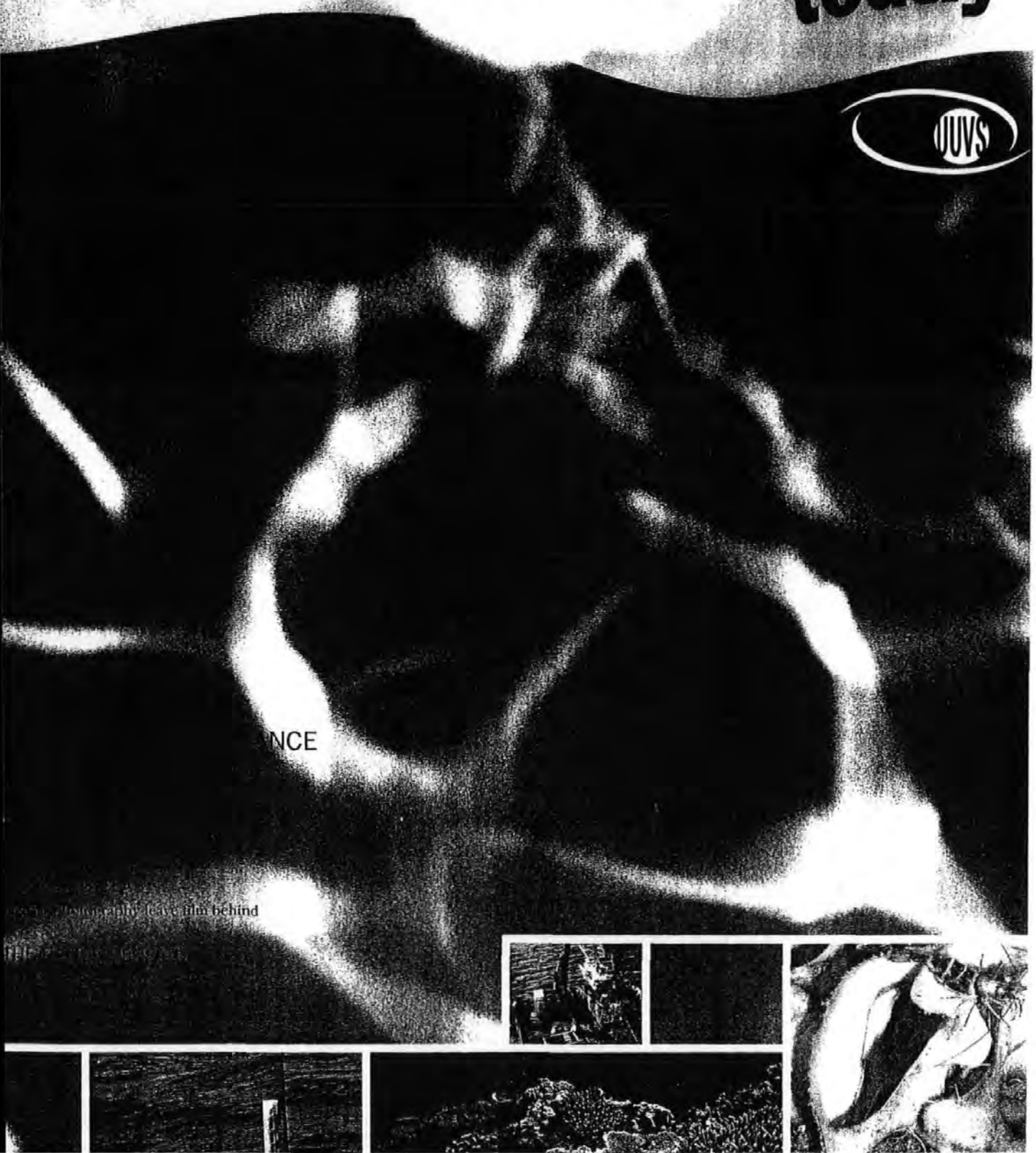
- Tan, C. S., R. Sutton, J. Chudley and S. Ahmad (2003). Autonomous Underwater Vehicle Retrieval Manoeuvre Using Artificial intelligence Strategy. *Proc. 1st IFAC Workshop on Guidance and Control of Underwater Vehicles (GCUV 2003)* pp. 143-148.

NO.5
AUTUMN 2005

www.oceanologytoday.com

Oceanology today

INCORPORATING THE OFFICIAL UUVS CATALOGUE 2005



CRASH, BANG, AVOIDANCE

Collision avoidance systems for autonomous underwater vehicles: A review of obstacle detection

C S Tan, R Sutton & J Chudley, University of Plymouth

SINCE MAN'S FORAY INTO THE OCEANS, mid sea collision has been a frequent occurrence. The poor navigation technology at that time was partially to blame for the situation. Various regulations or "rules of the road" were enacted in the hope to mitigate the occurrence of such incidents.

The prolific deployment of radar, a significant technology, in commercial vessels at the end of World War II was hailed as the solution to a centuries-old problem. However, much to the surprise of the maritime community even this innovative technology, regrettably, failed to prevent collisions from happening and it is painfully clear that most of the collision causes were not technology related but due to gross human errors, particularly the incorrect application of the technology and collision avoidance regulations. Undeniably every collision at sea has serious environmental, economical and human life implications.

The exploration of the oceans is extremely difficult to perform but is desirable for the advancement of economic, political, scientific and military purposes. Consequently, over the last view decades there has been an exponential growth in applications of unmanned underwater vehicles (UUV's) particularly Autonomous Underwater Vehicles (AUVs). Cost reduction and mitigation of the risk of human life have become the impetus for UUV exploitation. Despite advances AUV's still suffer from numerous inherent technical difficulties, specifically from power, sensing, communication and reliability limitations. One of the areas that particularly needs addressing, is that of collision avoidance, which is required to maintain the structural integrity of the AUV in a very hostile environment. An AUV collision in the ocean is intolerable for two main reasons; the recovery process can be arduous and the replacement process in terms of cost and time can be prohibitive.

Although pioneering efforts in establishing certain public laws concerning AUV operations1, 2, 3 have been made what is urgently lacking is an authority who can implement these instituted rules. One reason for the absence of interest in enforcing these rules is probably due to insufficient risk justification, especially the risk to human life. Unlike an unmanned aviation vehicle, which shares the same space with civil airspace as commercial aircraft, an AUV will conduct its mission under the water where the chance of encountering another AUV or submarine is extremely unlikely. However, the current scenario is about to change. By working cooperatively and via mutual information sharing these AUV's will be able to complete missions such as oceanographic sampling and mine hunting with substantial reduction in both operational time and cost. This, as a result necessitates a set of proper "Rules of the road" in order to safely and successfully conduct a multi-AUV mission.

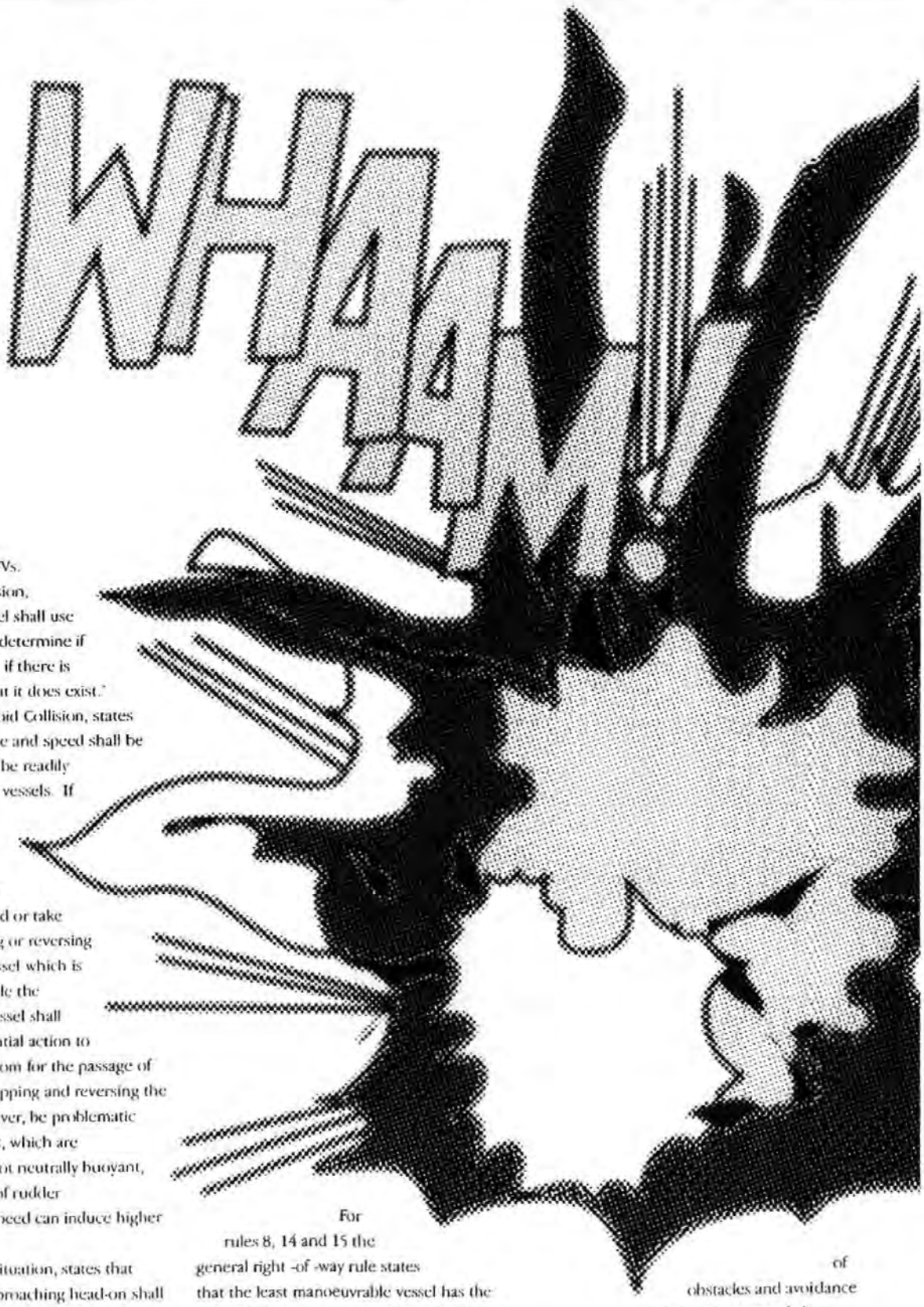
The Rules of the road pertain to a set of protocol applied to assist in tackling a collision predicament. Incorporating these ideas into an automatic collision system is not entirely new but implementations have historically been restricted to surface vessels. These selection of guidelines relevant to AUVs are derived from the International Regulations for Preventing Collisions at Sea.4, 5

Rule 2 Responsibility, requires that 'due regard shall be given to all dangers of navigation and collision.' This rule allows an AUV to depart from all the rules as necessary to avoid the immediate danger of collision.

Rule 4 Lookout, requires that 'every vessel shall at all times maintain a proper lookout by all available means appropriate in the prevailing circumstances so as to make a full appraisal of the situation and of the possible risk of collision.'

This is the primary task of an obstacle detection unit, where the primary lookout sensor employed is the sonar. There is even a suggestion that future AUVs shall be equipped with a system similar to the Identify Friend or Foe (IFF) unit commonly used in military aircrafts.

Rule 6 Safe Speed, requires that 'every vessel shall at all times proceed at a safe speed so that she can take proper and effective action to avoid collision and be stopped within a distance appropriate to the prevailing circumstances and conditions.' The speed of an AUV will be determined by these factors: the detectability, traffic density, manoeuvrability of the vessel with special reference to stopping distance and turning ability, the state of the sea, current, and proximity of navigational hazards. Slow speed, however, can affect the



manoeuvrability of AUVs.

Rule 7 Risk of Collision, states that 'every vessel shall use all available means to determine if risk of collision exists; if there is any doubt, assume that it does exist.'

Rule 8 Action to Avoid Collision, states that 'changes in course and speed shall be large enough so as to be readily apparent to the other vessels. If necessary to avoid collision or allow more time to assess the situation, a vessel shall slacken her speed or take all way off by stopping or reversing her propulsion. A vessel which is required not to impede the passage of another vessel shall take early and substantial action to allow sufficient sea room for the passage of the other vessel.' Stopping and reversing the propulsion can, however, be problematic for a majority of AUVs, which are *under*actuated, and not neutrally buoyant, for example the loss of rudder effectiveness in low speed can induce higher collision risk instead.

Rule 14 Head-On Situation, states that 'vessels which are approaching head-on shall alter course to starboard (right-hand-side) so each will pass port (left-hand-side) to port.'

Rule 15 Crossing Situation, states that 'when two vessels are crossing so as to involve risk of collision, the vessel which has the other vessel on her starboard side shall keep out of the way, and shall, if the circumstances of the case admit, avoid crossing ahead of the other vessel.'

For rules 8, 14 and 15 the general right -of -way rule states that the least manoeuvrable vessel has the right of way. For the case of a surface vehicle it is apparent that these manoeuvres occur in the planar domain. Whilst AUVs operate in a 3-D domain, for the moment, their avoidance manoeuvres are still limited to only planar motion owing to the restriction imposed by conventional 2-D obstacle avoidance sonar.

Any developments in collision avoidance systems for AUV both in terms of detection of obstacles and avoidance of obstacles can only be proposed in the expectation that forthcoming AUV's will, at least, be compliant with these regulations, hence minimising the risk of collision.

This is taken from a technical paper published in The Journal of Marine Science and Environment, Proceedings of the IMarEST, Part C2

C2 2004 ISSN 1741-1165

PART C2

Journal of Marine Science and Environment

Proceedings of The Institute of Marine Engineering, Science and Technology

IMAREST Publications

Collision avoidance systems for autonomous underwater vehicles Part A: a review of obstacle detection

C S Tan, R Sutton, and J Chudley
Marine and Industrial Dynamic Analysis Group
School of Engineering
The University of Plymouth, UK

An underlying requirement of an autonomous underwater vehicle (AUV), irrespective of the mission plan, is to navigate an unknown, dynamic and unstructured terrain. Usually, the AUV is fitted with a variety of advanced and expensive sensor suites dedicated to a particular mission. For this reason, the loss of an AUV due to collision is unjustifiable both in terms of the replacement cost and time. To prevent the occurrence of such an unfortunate event, one requires an effective and robust collision avoidance system which is able to preserve the AUV structural integrity. The aim of this paper is to discuss the techniques for designing collision avoidance systems and their corresponding advantages and disadvantages. The subject material has been divided into two parts for clarity of exposition. Here, in Part A, emphasis is placed on the description of obstacle detection aspects of collision avoidance systems. Additionally, a brief introduction concerning AUV control architectures and their relevant collision avoidance submodules are also presented.

AUTHORS' BIOGRAPHIES

Chiew Seon Tan, is a research student in the School of Engineering at the University of Plymouth. Prior to that he worked for a number of years as a mechanical engineer specialising in both automated, semiconductor and agricultural machinery designs. His main interests include digital signal processing (DSP), fuzzy logic, optimisation algorithms, and optimal control systems. In addition, he is also a recipient of a Stanley Gray Fellowship award from IMarEST, the Frederic Barnes Waldron Best Student Prize (2002) and the Best Mechanical Engineering Project Prize (2002) from IMechE.

Robert Sutton is Professor of Control Systems Engineering in the School of Engineering at the University of Plymouth. His main research interests lie in the application of advanced control engineering and artificial intelligence techniques to control problems. Ongoing research is concerned with the use of fuzzy logic, neurofuzzy algorithms, and adaptive search algorithms in the design of industrial and marine plant.

John Chudley is currently the Director of Knowledge Transfer for the University of Plymouth. Prior to taking up this position he was the Head of the Department of Mechanical and Marine Engineering at that university. His main research interests are in the design of marine propulsion systems including performance prediction, composite propellers, integrated navigation systems and marine renewable energy.

INTRODUCTION

It is surprising to know that approximately 97% of the space for Earthbound organisms are in the oceans, thereby, supporting the fact that oceans play a vital role in sustaining the Earth's ecology. Even with current technology advancement, less than 5% of the oceans have been characterised to the same degree of resolution as the planet Mars. One obvious reason is that the exploration of this environment is extremely difficult to perform, however it is still desirable for the advancement of economic, political, scientific and military purposes.

Consequently, over the last few decades there has been an exponential growth in the applications of unmanned underwater vehicles (UUVs), particularly in the field of science, the offshore industry and the military. Cost reduction and mitigation of the risk of human life have become the impetus for UUV exploitation. UUVs can be used for sea bottom exploration, repairing, surveying, policing exclusive economic zones, mine-hunting, seabed mapping, scientific data, and intelligence gathering. In this context, the term 'unmanned underwater vehicle' is considered as a generic expression to describe both an autonomous underwater vehicle (AUV) and a remotely operated vehicle (ROV). An ROV can be considered as a human-operated, highly-manoeuvrable underwater vehicle that is connected via an umbilical cable to a surface vessel. However, this does severely limit its operating range. Unlike the ROV, the AUV without the restraint of an umbilical, is a free swimming vehicle of higher autonomy, capable

of performing missions that require longer operating range without human intervention. For clarity of exposition, the term AUV will be used for the remainder of this paper because of its more challenging and rigid requirements. Nonetheless, the ideas discussed are still applicable to a wide range of vehicles. It is worth reviewing the recent trends in AUV applications, to better appreciate their contributions.

The offshore and scientific communities, who were especially sensitive to financial constraints, were quick to seize the opportunity in exploiting the potential of AUVs. Some of the commercial AUVs for offshore survey are *Hugin* (Norway),[1] *Aqua Explorer* 1000 (Japan),[2] and *Theseus* (Canada).[3] This has been reinforced by the recent placing of orders to purchase AUVs by Fugro-Geos Ltd, C&C Technologies and Racal Survey Ltd.[4]. In the case of the scientific community, *Autosub*,[5] and *Theseus*,[6] AUVs have demonstrated their ability to navigate under polar ice caps while the *Autonomous Benthic Explorer* (ABE),[7] has performed a fine-scale sea floor survey in a rugged deep-ocean terrain. All of these have been achieved at significant financial cost saving. These impressive achievements further strengthen the belief that AUV applications will continue to escalate as the realisation of the importance of ocean resources unfolds.

Recently, the military have shifted their focus from blue-water to brown-water warfare. This was instigated by the increase propensity for littoral water operations and the attendant focus on amphibious power projection.[8] Technically, the littoral zone is a subdivision of the benthic province that lies between the high and low tide marks and can be considered as an extension of the shoreline to 600ft (183m) out into the water. The importance of accessing the littoral zone is critical if a successful amphibious launch is to be achieved. The littoral zone is an intricate area to navigate by default, with unpredictable natural effects such as bioluminescence*, internal waves, coastal currents, changing beach profile, reefs and artificial objects. This is made increasingly difficult by the prolific deployment, by defending countries, of inexpensive underwater mines which have the effect of retarding or halting any military advancement. Consequently, this has prompted a search for the most effective countermeasure that culminated with the employment of AUV technology. Currently, the US employs the highly portable and cost-effective *REMUS* AUV for mine hunting as illustrated in the recent 2003 Iraq conflict.[9] Moreover, the design of a more advanced, stealth AUV, codename *Manta*, is already in progress,[10] and the long-term mine reconnaissance system (LMRS) is also due in service soon. Elsewhere, the UK is using the highly modular and reconfigurable *Marlin*,[11] that is also submarine-launch capable and a smaller, more manoeuvrable, *Gambit*,[12] for mine countermeasure mission.

From the aforementioned, the impression may have been given that AUVs are going to become the panacea for a number of subsea activities. This is certainly not the case, as AUVs still suffer from numerous inherent technical difficulties, specifically from power, sensing, communication, and reliability limitations. The increased exploitation of AUVs, as mentioned above, has also demanded a more robust and autonomous capability. One of the areas that needs particular addressing is collision avoidance, which is required to maintain the structural integrity of the AUV in a very hostile environment. Clearly, once a collision occurs, and assuming that the AUV structure is breached, then it is inevitable that a catastrophic failure will follow. Thus an AUV collision in the ocean is intolerable for two main reasons; the recovery process can be arduous, and the replacement process in terms of cost and time can be prohibitive.

In this paper, collision avoidance is defined as the ability of a vehicle to detect and avoid colliding with both static and dynamic obstacles, while still attempting to accomplish the current mission objective. The processes involved also encompass obstacle detection, digital map building (workspace representation), motion planning and reflexive obstacle avoidance. To date, AUV collision avoidance schemes have been somewhat ad hoc whereby only a simple reflexive module is added depending on the customer requirements. The problem with this approach is that the resulting AUV system is not fully integrated with its various subsystems which leads to suboptimal behaviour in terms of manoeuvrability and capability to navigate in a dynamic, unknown and unstructured terrain. Moreover, under operational conditions, this is exacerbated still further by the poor sensor performance and highly complex non-linear dynamic nature of an AUV. Previous AUVs were designed as efficient swimmers with very limited manoeuvrability, but some recent AUVs are equipped with auxiliary vector thrusters† to achieve more sophisticated manoeuvres such as hovering and pure sway movement. As a consequence, a new collision avoidance system that is capable of addressing all these factors is required.

To ease comprehension, this paper adopts an approach where the pertinent theory and concept are emphasised before proceeding to their specific implementations. Each of the modules is also presented in accordance to the sequel of the collision avoidance process. Starting with the following section, a review of concepts and techniques used in designing a collision avoidance system for an AUV are provided. It commences by providing a brief outline of different types of AUV control architectures. Clearly, this is vital since the control architecture functions as a supporting platform for the collision avoidance mechanism. Various collision avoidance architectures are surveyed. Subsequently, several modules that form the basis of a generic collision avoidance system are highlighted. The third section of this paper delves into the obstacle detection module which is comprised of a forward-looking sonar, a sonar signal processing module and finally a map builder module. The theory, principle and implementation behind each individual submodule are then elaborated upon. Then, concluding remarks are given, stressing the limitations of the current techniques. The obstacle avoidance aspects of this subject are addressed in the Part B paper which follows this paper.[14]

* Refers to the light-producing ability of certain surface organisms. Any provocation of the organisms will cause them to emit light. Thus to maintain stealthiness, AUVs must take extra precautions when travelling on or near the surface.

† One example is the novel flexible foil propulsor called *Nektor*.[13] These flexible thrusters are capable of maximising AUV agility by transforming it into a holonomic vehicle.

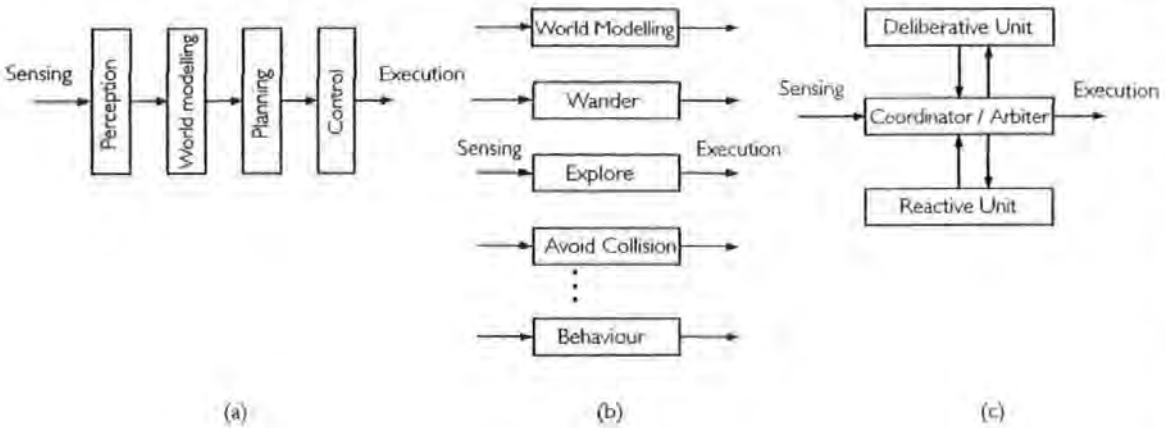


Fig 1: Types of control architecture; (a) generic deliberative control architecture (b) generic reactive control architecture (c) generic hybrid architecture

COLLISION AVOIDANCE SYSTEM ARCHITECTURE

AUV control architecture

A control architecture is a framework which manages the sensorial and actuator system in order to enable the AUV to undertake a user-specified mission. This is a major topic of research, and different approaches to AUV control architectures are discussed in the literature.[15,16,17] This section intends to elaborate on three major types of control architectures.

Deliberative architecture

This architecture is also known as a top down, structured, symbolic, goal-driven, model-based, hierarchical or sense-plan-act approach. Deliberative architecture always maintains internal representations of its surroundings and this allows it to make reasoning, prediction and inferencing concerning the environment. The information flow direction is depicted in Fig 1(a). This scheme represents a well-defined, tightly-coupled structure thus simplifying the process of designing, debugging and evaluating the system. However, the amount of information flow from sensors to the centralised computing resources can be significant. Exacerbating the situation is the synchronisation difficulty of workspace representations and the environment. Owing to the computationally intensive nature of the architecture, there is a tendency to exhibit unresponsive or erratic behaviours in unpredicted situations. This architecture is employed in the the *EAVE*,[18] and the *OTTER*.[19]

Reactive architecture

Also known as a bottom up, sensor-driven, layered, forward-inferencing, subsumptive, reflexive or sense-react approach. The theory of reactive architecture was initiated by Arbib,[20] and implemented by Brooks.[21] It is based on a parallel structure where each individual sensor is used to sense the environment, providing its own perception and activating its own behaviour, refer to Fig 1(b). A global behaviour is produced by coordinating the parallel execution of individual behaviour. Its performance is excellent, particularly in unforeseen situations. Furthermore, this scheme is known for its flexible and modular nature. However, its propensity to demonstrate elusive behaviour when subjected to conflicting sensor information is a major concern. Also, its nondeterministic nature does not lend itself to a straightforward performance evaluation. Lastly, its deficiency in global mapping and in relation to workspace objects, often results in simplistic behaviours which tend to get trapped in certain cases. This architecture is employed in the *Sea Squirt*,[22] and the *Twin Burger*,[23] AUVs.

Hybrid architecture

In the search for a superior architecture than the two previously discussed, the hybrid architecture was born through the amalgamation of both the above architectures. Generally, it is decomposed into three task specific layers: deliberative, reactive and execution layer, refer to Fig 1(c). In military parlance, it is called the strategic, tactical and execution layer. Abstraction and real-time responsiveness varies correspondingly at each level. The deliberative layer is in charge of high level planning (non time-critical) while the reactive layer is responsible for real-time issues. The execution layer acts as supervisor to facilitate interlayer interactions. Due to its apparent advantages, most recent AUVs have employed a variant of this architecture. The *Garbi*,[24] the *SAUVIM*,[25] and the *Phoenix*,[26] AUVs are examples that exploit this architecture.

Collision avoidance system architecture

As stated previously, pure deliberative and reactive architectures do not function adequately for a collision avoidance task. As hybrid control architecture provides an ideal platform for integrating the functionality of the individual submodules, it is not surprising that it is applied in the majority of the proposed obstacle avoidance architectures.[27-31]

Before proceeding to an in-depth discussion pertaining to the individual submodules, it would be more enlightening to provide a simple descriptive of a collision avoidance process to better elucidate the utility of each submodule. A typical collision avoidance task can be considered like this. First and foremost, a target must be acquired by the forward-looking sonar. Classification of static or dynamic targets are then performed. Depending on the types of object, their information will then be fused with AUV navigation data such as velocity, depth and altitude in order to represent the object into a digital map. From the digital map, a motion planning technique is employed to steer the AUV safely to its predefined goal or subgoal. Motion planning is computationally expensive and not very suitable for tackling unexpected objects. Therefore, the reflexive obstacle avoidance submodule is employed to provide the AUV with a time-critical, in situ response to an unexpected object. Once the obstacle has been successfully avoided, the AUV should resume its preplanned mission. The bolded phases denote critical processes in collision avoidance. These processes are highly dependent, particularly the one in the lowest of the process chain such as motion planning. For this reason, a method of designing an efficient, optimal and practical collision avoidance system

requires a perfect integration of these processes. On the whole, a collision avoidance system can be decomposed into two principal functional modules; the obstacle detection module and the obstacle avoidance module, where both of them comprise further submodules.

Obstacle detection module
1. Forward-looking sonar
2. Sonar processing submodule
3. Navigation submodule
4. Map builder (Workspace representation submodule)

Obstacle avoidance module
1. Motion planner and waypoint generator
2. Trajectory tracker. (Autopilot and actuator controller)
3. Reflexive submodule.

A detailed discussion of aspects of obstacle avoidance is given in the Part B paper. Fig 2 illustrates the interconnection of the submodules of a generic collision avoidance system. The arbiter is used to co-ordinate the activation and inhibition of various submodules.

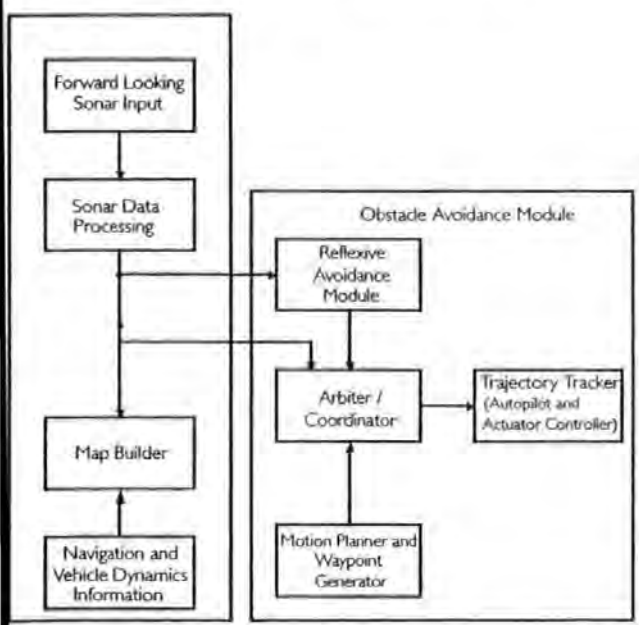
Forward-looking sonar

A forward-looking sonar is frequently used for AUV obstacle detection. Recently, the advent of digital signal processing (DSP) technology has increased the popular usage of cost-effective, high resolution, electronic beam formed sonar for obstacle detection purposes. Besides providing adequate bearing and range resolution, its rapid scanning rate (frame rate) also permits temporal information extraction, which is vital for motion planning in a dynamic environment.

Sonar processing submodule

Despite the acoustic sensor performance being unprecedented in underwater applications, obtaining high quality and reliable sonar data is still problematic. Reverberation, reflection, refraction and scattering tend to corrupt the data and cause frequent false alarms, hence subsequent processing of the data is required. This submodule is also responsible for object discrimination, verification and tracking.

Fig 2: Generic collision avoidance architecture



Navigation submodule

A navigation submodule typically comprises an inertial measurement unit, digital compass, depth sensor, altimeter, and a GPS unit (when surfaced). When submerged, the AUV is deprived of any global frame of reference and dead-reckoning is the only viable method for localisation.

Map builder

Deprived of any global frame of reference, it is critical to have an online map which is incrementally developed to assist an AUV in navigating the unknown terrain. A digital map is also required for localisation and motion planning processes. Clearly, there are numerous methods of representing the AUV environment; three well-known methods are cell decomposition, geometrical representation and topology representation.

Motion planner and waypoint generator

A motion planner is used to assist an AUV in navigating through an unstructured and unknown environment via the generation of a time-parameterised path, whilst simultaneously taking into account several factors such as AUV safety, kinematics, dynamics and energy constraints. An interested reader is directed to Part B of this paper,[14] for more details regarding this matter.

Trajectory tracker

A trajectory tracker is also popularly known as the autopilot. The actuators to be controlled can be a rudder, hydroplane or motor. In essence, its main responsibility is to ensure that the AUV output follows the desired input. This is not a trivial assignment when one needs to take into consideration the effect of the vehicle dynamics, modelling uncertainty, sensor noise and external disturbances. Presently, this area is a major topic of research.

Reflexive submodule

A reflexive submodule function is similar to a backup system in the unfortunate event of motion planner failure. The failure can either be a system malfunction or a failure to meet the predefined time constraint, which is a more common occurrence than the former. Unlike the motion planning submodule, this submodule is highly capable of responding to unforeseen circumstances. This submodule is activated when an object intersects a predefined virtual boundary.[27,32]

OBSTACLE DETECTION MODULE

Introduction

As previously outlined, an obstacle avoidance module consists of a forward-looking sonar, a navigation submodule and a map builder. The navigation submodule will not be reviewed since this has been provided by Loebis et al.[33] The primary function of an obstacle detection module is to detect, discriminate and represent the object information into a digital map for disposal by the obstacle avoidance module.

Forward-looking sonar

In the underwater domain, radio waves and vision suffer from inherent limitations. Radio waves are virtually useless underwater due to its high attenuation, while vision effectiveness is restricted to a range of a few metres, and is highly dependent on the turbidity of the water. This is caused by the scattering effect of light by suspended matter. Obviously, one method is to employ a higher intensity light source to offset the light attenuation, but this only results in a massive power drain.

Unlike radio waves and optical energy, sound transmission is the single most-effective means of directing energy transfer over long distances in seawater. Consequently, an acoustic sensor in the form of sonar is largely employed underwater. There are numerous sonar types such as bathymetric sonar, side scan sonar, tow-array sonar and etc, which are all applications specific. One type that is commonly employed for obstacle detection is the forward-looking sonar. The main purpose of a forward-looking sonar is to provide spatial information such as the range, bearing and size of an object via some processes of signal processing and data fusing.

A forward-looking sonar is required to detect objects at the longest range possible in order to allow for further information processing before an avoidance manoeuvre can be initiated. However, at moderate ranges of several hundred metres, sonar paths can be distorted significantly because of continuous refraction from sound speed variation caused by changes in water temperature, salinity, and pressure. To aggravate the situation, sonar range is also highly frequency-dependent. For long range detection, a low-frequency sonar is required. Nonetheless, low frequency results in poor acoustic resolution. In the case of shallow water (200m or less) and when an AUV is cruising near to the sea bed (pipe tracking or terrain following), this issue is exacerbated by the the combined effect of boundary reverberation noise, multi-path returns and bottom clutter.[34] Increasing the acoustic resolution, on the other hand, can significantly enhance an AUV's ability to perform boundary reverberation discrimination while obtaining a more precise bearing on echo returns. Besides, high acoustic resolution is also critical for the purpose of map building and optimal path generation.[35] For this reason, there is a constant trade-off between operating range and acoustic resolution; proper selection should be based on the AUV mission. Some of the desired qualities of an AUV sonar are listed below:

- Low power consumption.
- High resolutions with adequate detection range (depending on the AUV operating speed).
- Scanning rate.
- Cost.
- Embedded clustering or classification logic (optional).
- Embedded static and dynamic objects tracker (optional).

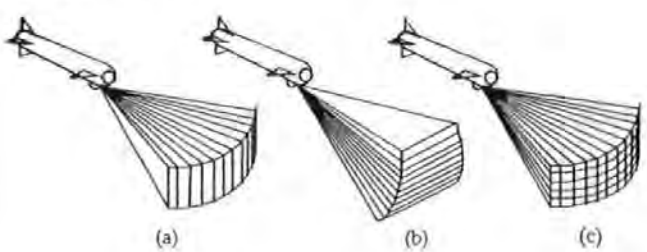
Types of forward-looking sonar

A detail review of different types of forward-looking sonars can be found in Loggins.[36] Lately, the advent of DSP technology has culminated in the development of high performance, low cost electronic beam-formed sonar. In principle, beamforming[37,38] is a process of listening to or transmitting energy (sound, in this case) from an array at selected angles. The core concept is to sum the incoming signal such that those that are coming from a given direction are added coherently, resulting in maximum magnitude response, while those signals arriving from other directions are attenuated as a result of the self-destructive interference effect. The two main approaches in beam-forming are the time-domain and the frequency-based methods. Typically, a number of fixed directional receiver beams are formed simultaneously to cover the ensonified region in order to obtained better directional resolution while maximising the scanning rate.[39] One obvious advantage of this sonar is its high scanning rate (frame rate), rendering it less susceptible to platform motion disturbance. Besides, the high scanning rate can be exploited for temporal and spatial information.[43] The lack of mechanical moving parts also increases the reliability of the sonar.

There are several types of sonar that are based on slightly different operational principles. Neglecting this distinction, as a whole they may be categorised into a 1-D, 2-D and 3-D sonar. The most primitive form of sonar is a 1-D sonar such as an echo-sounder or depth-sounder, which is capable of only providing range (altitude) information. Two examples of 2-D sonar, with different configuration are shown in Fig 3(a) and (b). The Fig 3(a) sonar configuration is commonly employed by most commercial AUV forward-looking sonars. Here, range and bearing information is acquired, but not depth. This makes it suitable for AUVs performing mid-sea surveying and mine-searching missions, where the environment is uncluttered or sparse. However, discrimination of object depth can be difficult. Taking into account the worst-case scenario, and assuming that the obstacle is on the same plane as the AUV, this certainly restricts the AUV to perform only planar evasive manoeuvres. A planar manoeuvre might not be the most effective action in certain circumstances, since a better approach might be to climb or dive over the obstacle if possible. Another advantage of multibeam sonar is its capability to ensonify the illustrated region (Fig 3(a)) simultaneously in a single ping, while the pencil beam sonar is limited to scanning the entire sector incrementally.

Fig 3(b) shows an alternative sonar configuration which is similar to the former but with the transducers being rotated through 90deg. This configuration is commonly employed by a surface vessel for collision avoidance purposes. It provides depth and range information at the expense of bearing information. This mode tends to put more emphasis on discriminating objects in the vertical direction (terrain) than the horizontal (suspended object). This configuration is also beneficial for an AUV that is performing terrain-hugging manoeu-

Fig 3: Multibeam sonar type: (a) horizontal scanning 2-D sonar (b) vertical scanning 2-D sonar (C) horizontal and vertical scanning 3-D sonar



vres, such as in a pipeline tracking and seafloor surveying missions, as it needs to estimate the terrain gradient in advance to facilitate a successful manoeuvre.

The most advanced forward-looking sonar is of the 3-D type, (Fig 3(c)). This is achieved using an array of transducers. It possesses the capability to discern an object in a spatial domain by providing range, bearing and depth information simultaneously. This type of sonar is needed to exploit fully the true capability of some motion planning algorithms in generating optimal, 3-D trajectory. Nevertheless, the cost of this type of sonar can be prohibitive. To circumvent this issue, one method is to utilise the Fig 3(a) configuration by making the vertical beam-width thinner and steerable in the vertical direction. Assuming a sufficiently-fast steering (sweeping) rate is obtainable, it can then be extended into a pseudo 3-D sonar.

Sonar signal processing

The raw data obtained tends to be corrupted with noise, mainly because of the operating principle and operating environment of a sonar. This imposes further processing stages to obtain better representation of the environment. Pettilot et al[40] reported that sonar signal processing can be classified into four distinctive processes:

- Filtering and segmentation.
- Feature extraction.
- Tracking.
- Map building (workspace representation).

Filtering and segmentation

The first step in the sonar signal processing typically entails the elimination of noise and backscatter of sonar images caused by the scattering and reverberation effect. This can be achieved by applying a simple Gaussian, median or mean filter to the image. A median filter is the most effective in elimination of backscatter noise, however, the Gaussian filter is sometimes used due to its lower computational requirement.[40]

Segmentation, a process of regioned pixel extraction, is applied to enhance object background discrimination in order to increase the robustness and accuracy of the tracking process. The most popular and simple is the thresholding technique, also known as binarisation. In principle, thresholding is a process of defining a limit so that any colour above the limit will be converted into black, while those below the limit will be converted into white. It is effective when the intensity levels of the objects fall squarely outside the range of levels in the background. A more sophisticated version, called the adaptive threshold technique, uses a switching function-integration to provide an improved result.[41] The use of a unsupervised hierarchical Markov random field (MRF) model together with contextual information has also been reported.[42] Simulated annealing has also been attempted, but the algorithm is applied to segmentation of synthetic aperture radar images,[43] and not sonar. Both the algorithms are very computationally intensive, making real-time implementation very difficult.

Segmentation processes can be very costly in terms of computational requirement and, as such, some authors advocate using selective, multi-rate/multi-depth filtering and data compressing techniques. In the selective approach, the static and dynamic part of the image is discriminated using a frequency domain method (one-dimensional Fast Fourier Transform (FFT)) or a time-domain method (moving average).[44] Once the dynamic object is detected, it will be tracked and segmented only at the particular region of interest. For a static object, only new objects need to be segmented. In contrast, the multi-rate/multi-depth technique tries to redistribute the computational load by sampling the area at various rates depending on the degree of their importance.[35] Clearly, those regions adjacent to the AUV are more critical and deserve a higher sampling rate. Instead, Zanoli et al[32] attempted to compress the sonar data before filtering, significantly reducing the processing requirement.

Feature extraction

Feature extraction is a process that is intimately linked with object classification.[41,45] In the case of image processing, feature extraction entails accurate measurement of object features. Ideally, the feature selected should be invariant under various circumstances while extracting maximum information regarding the object. Such features can be object size, such as area, perimeter, surface and centre of mass, which can be easily obtained by counting pixels of the object, or more complicated parameters such as moments, mean, variance, and median used to describe statistical distributions.

Tracking

In this context, tracking is a process where object attributes such as position, velocity and estimation confidence level are estimated and recorded. In video processing, one tries to correlate a predetermined feature with subsequent frame features and noting their difference. Tracking is typically a forward-looking process, requiring a computer to anticipate the object position and velocity ahead of time. The accuracy of the predicted target attributes play a critical part in characterising its behaviour, hence making it indispensable for the motion planning process. Lane et al[46] applied an optical flow with an associative searching trees technique while Moran et al[47] advocated using a multiple hypothesis for object tracking. Multiple hypothesis technique is effective in cases where multi-modal representation is required, such as in the presence of background clutter, self-occlusions and complex dynamics. However, both these pixel-based schemes are very computationally expensive thus precluding their application in time-critical applications.

Alternatively, the classical Kalman filter has been applied with success in sonar tracking systems.[35,48,49,50] To simplify the analysis and lighten the computational requirement, Williams et al[48] employed two different Kalman filters for tracking dynamic and static objects while Henriksen[35] preferred using a total of five separate Kalman filters to track the corresponding states, but this is not without problems. One inherent limitation of the Kalman filter, due to its derivation, is the assumptions of a linear model, Gaussian white noise, constant noise distribution and uni-modal representation. If certain discrepancies exist between the ideal case and practical case, then its effectiveness can be greatly affected. Its performance is also plagued by the 'curse of dimensionality'. In other words, its real-time performance degrades tremendously when the number of objects being tracked increases. These are a few pertinent problems that need to be addressed in this

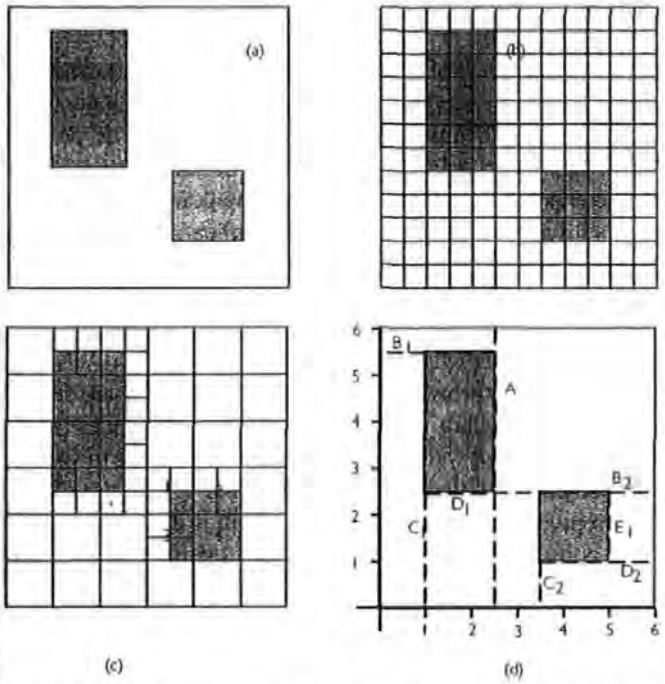


Fig 4: Diagram showing: (a) workspace (b) uniformed grid (b) quadtree representation and (c) binary spatial partitioning

research area.

Map building (workspace representation)

Knowledge representation is one of the key elements that determines the capabilities and performance of machine intelligence. This is particularly true for map building or workspace representation processes, which can be defined as a process of generating models that represents the vehicle environment via sensor measurements. The generated model or digital map, other than containing metric information, can also be embedded with supplementary user-defined information to better characterise the environment. This information is vital for motion planning, obstacle avoidance and localisation processes.

Nevertheless, the generated models tend to be a simplified version of the real environment for three obvious reasons:

1. This is limited by the AUV sensorial perceptions.
2. This is to comply with the computational requirement.
3. The condensed information is more suitable for high-level symbolic manipulation and model inference.

Conversely, the aggregate of discarded information and sensor induced errors, can be considered as noise, and is detrimental to the overall system performance. In the case of an AUV, the sensor drift in a dead-reckoning scheme tends to degrade the map reliability after a certain time period, whilst increasing the model-fidelity will definitely enhance the system performance, but at the expense of memory and computational requirements.

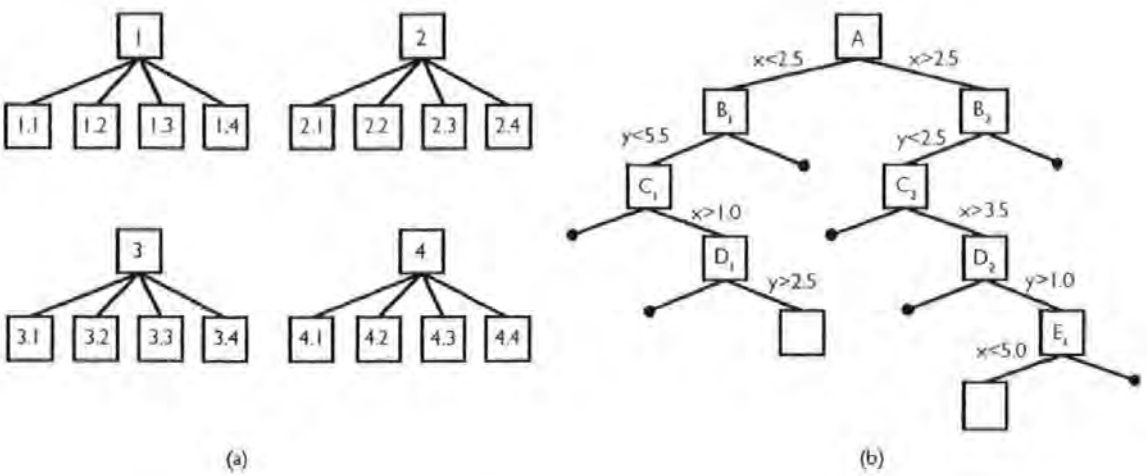
For this purpose, map building can be considered as a trade-off between model-fidelity, memory requirement, robustness, computational efficiency, implementation simplicity and expansibility.[51] There are three fundamental schemes in workspace representation; metric-based spatial decomposition, geometry representation and non-metric-based topological representation.

Spatial decomposition

Spatial decomposition is a scheme of representing space via a discrete sampling process; division of space into non-overlapping cells. Either only the free space is taken into account, or only objects are mapped and free space is found by implication. There are various variants of the spatial decomposition method.

The most conceptually simple and yet prevalent scheme in the field of mobile robotics is where the environment space is partitioned into uniform, non-overlapping grids or cells in a spatial lattice (Fig 4(a)). Each cell can be allocated with user defined attributes such as confidence of obstacle presence, terrain geometry and safety factor. This scheme conventionally employs probabilistic sensor interpretation models to update the cell value[52]. Due to its popularity, it is known by different names, such as evidence grids, probability grids, certainty grids or occupancy grids. Hyland[27] and Allison et al[53] have implemented this scheme in their AUV simulations. One overriding constraint concerning this approach is the high memory requirement, such that it can be characterised by the space complexity of $O(a^n)$, where a is a constant and n denotes the dimension. It must be understood that the number of cells employed to approximate a model are finite, hence, decreasing the cell size will definitely improve the model fidelity but at the

Fig 5: A tree representation of (a) quadtree (b) binary spatial partitioning



(a) (b)

expense of escalating the cell quantity. This problem is intensified for cases of higher dimensional space.

As a result, various researchers have resorted to dual resolution maps; each map using different resolution. Ridao et al[54] describe using a high resolution map to record sonar pings for the *SAUVIM* AUV. Only pings that have not expired after a preset time interval are recorded into a coarser map, and used for path planning. Fundamentally, the high resolution map is functioning like a low pass filter to eliminate false alarms. Similarly, Moitie et al[30] employed a low resolution map for global path planning and a more detailed local map when executing local motion planning.

To solve the memory and computational inefficiency of a uniform cell map, a type of multi-resolution algorithm has been proposed.[55] It is known as a quadtree and octree in their 2-D and 3-D forms respectively (Fig 4(b)). A quadtree is fundamentally a recursive data structure with a hierarchical representation property. It attempts to exploit the occupancy of adjacent cells by clustering them much like a data-compressing algorithm. It adaptively subdivides into smaller cells in order to improve the modelling accuracy, while the minimum cell size determines the depth of the tree and the accuracy of the mapping. Fig 5 shows how it is represented in the form of a tree to facilitate quick searching. Its efficiency is much superior to that of the former method, particularly for environments that are sparsely populated with objects.[56] However, its performance suffers significantly for the case of a dynamic object due to constant tree structure changes.

Another efficient workspace representation scheme,[57] highly popular in the computer graphic domain, is the binary spatial partitioning (BSP) scheme. A BSP tree is a hierarchical representation structure that exploits the recursive subdivision by hyperplanes (Fig 4(d)). Since there is no restraint on the types of hyperplane used, exact polyhedra and polygon representations are possible. All of this information is then compactly encoded in the form of a binary tree structure, as shown in (Fig 5(b)), ready for subsequent implementation of path-finding algorithms. Unlike the quadtree, the BSP tree structure is preserved by affined and perspective transformations, which result in its capability to incorporate dynamic objects without resorting to changing the tree structure. This scheme has been exploited by Arinaga et al[28] for the *Umihico* AUV workspace representation.

Geometric map

Probably the oldest of the workspace representation methods is the geometric map. As suggested by the name, the geometric map tends to use geometric primitives such as points, lines, polygons, polyhedrals and polynomial functions to characterise the environment. One of the compelling advantages of a geometric map, assuming application of appropriate modelling primitives, is its capability to model complex objects with a very low memory requirement. This concise mathematical representation also facilitates a rapid and accurate collision-checking process.[58]

The simplest primitives, such as point and line, are rarely used in isolation but as a preliminary form of model inference. Leal[59] employed points in what he referred to as the sampled environment map (SEM) scheme. Unlike a uniform grids representation, here the environment is divided into discrete point locations. Then a decision theoretic scheme is used to adapt a geometrical model from the sampled environment distribution. Brutzman et al[60] employed this scheme to trace incrementally the obstacle contour by aggregating piecewise, linear lines into polygons. Caccia et al[61] and Moran et al[47] developed modules to process and classify sonar data into corresponding geometrical features. The systems, however, are constrained to function only in a partially man-made environment since it is more geometrically distinctive compared to the non-homogeneous features found in nature. Of all the polygons, the circle or sphere has particularly interesting attributes such as simple formulation, orientation invariance, convex shape and ease of manipulations, thus explaining its popularity. This method of representation is employed by Fox, Garcia,[62,63] and Wang et al.[64] Others prefer to approximate the obstacles as polygons, particularly convex types.[65,66] Convex attributes are vital in simplifying the implementation of various motion planning algorithms while fostering faster convergence. Unsatisfied with the limitation of simple polygonal representation, Lane et al[39] resorted to using a constructive solid geometry(CSG) method, a technology extensively used in the CAD industry. The CSG method allows explicit representation of objects using simple primitives such as spheres and cuboids via Boolean operators: subtraction, intersection and union. One key attribute is the lack of ambiguity between the inner and outer part of the object. To ease the implementation of the optimisation algorithm, Wang et al[64] restricted themselves to using only sphere and

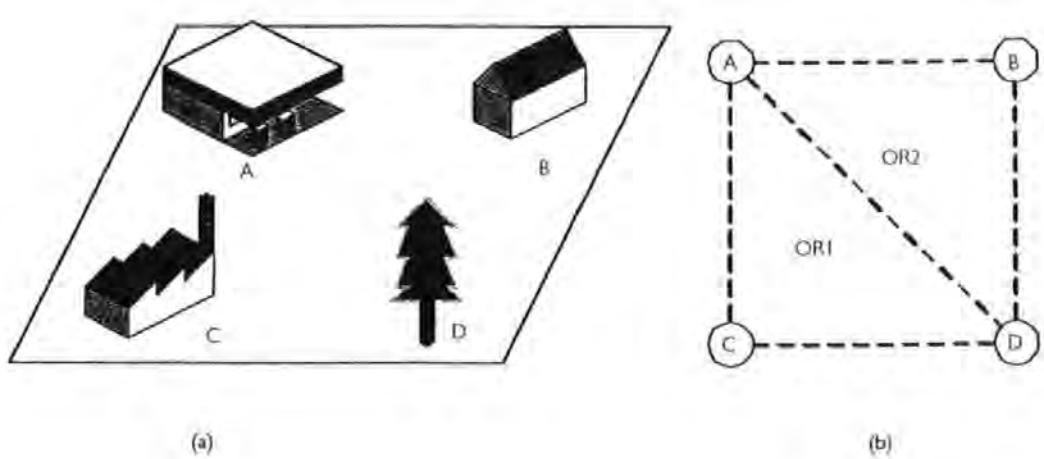


Fig 6: (a) landmarks in workspace (b) topology representation of the workspace

ellipsoid primitives. Alternatively, one can try to approximate the seabed surface using a surface modelling technique.[67]

Notwithstanding the above advantages, one of the obvious shortcomings of the geometry map is its difficulty in making inference from noisy, measured-sensor information which has a great impact on its reliability. Furthermore, a stochastic model can rarely be described in a simple parameterised, geometric manner. Attempts to do so have achieved limited success. Other problems are also encountered, such as lack of stability and lack of expressive power to model the object.[51] Lack of stability is due to parameters that are sensitive to variation, causing additional erratic model shape changes. On the other hand, lack of expressive power is caused by using oversimplified geometric models which severely restrict their approximating capability.

Topological map

Topology is concerned particularly with the global connectivity of an object by considering how the object is connected locally. A topology map represents the environment as graphs, where nodes correspond to distinct places (landmarks), and arcs represent adjacency or orientation. Fig 6(a) illustrates a hypothetical workspace with landmarks and (b) the topology representation of the workspace. The orientation regions (OR) are used for localisation. The key to topological representation is its compactness and high immunity to noise since it is less dependent on metric information. This compact representation also facilitates high-level symbolic reasoning for map-building, navigation, planning, and communication.

Since this scheme is not especially susceptible to noise, one obvious application of it would be to the problem of simultaneous localisation and mapping (SLAM)[68] to assist an AUV to navigate uncharted waters. In spite of this, its effectiveness is currently being outperformed by a metric approach using an extended Kalman filter (EKF). This scheme is also known as stochastic mapping.[69,70]

In reality, the topology mapping scheme performance is far from outstanding, hence explaining its unpopularity. One particular downfall is its excessive dependence on the presence of landmarks. A landmark can be defined as an individual or a cluster of objects that have distinctive feature relationships. Other desirable criteria are ease of identification, uniqueness and repeatability. Zimmer[71] advocates embedding local metrical map patches with a globally-consistent topological map. Then again, his results tend to be biased as it was simulated using a land-based robot in an indoor environment where proper landmarks can easily be identified. These landmarks are particularly difficult to find in a non-homogeneous and 'noisy' environment where an AUV operates, thus severely limiting its usage.

Hybrid representation

Owing to the apparent advantages and disadvantages of each method of representation, some researchers resorted to using hybrid representations such as a combination of metric-based and topological paradigms.[68,71,72] Hino[73] utilised a uniform grid scheme for preliminary terrain representation. A data reduction scheme is introduced to convert the previous map into a contour-like map, with significant memory saving. Nevertheless, the transformed contour map lacks flexibility for further modification. Zanoli et al[32] employed spatial decomposition techniques for local map building, but all AUV obstacle avoidance tasks are conducted using a geometry model of the environment. To conclude, hybrid representation provides the user with better flexibility, simplicity and robustness that is difficult to achieve using an individual type representation. However, extra precautions are required in synchronising and maintaining the data integrity in between different representations.

CONCLUDING REMARKS

An overview of AUV control architectures and collision avoidance system submodules has been presented. The obstacle detection module has been divided into four distinctive sub-units; a forward-looking sonar, a signal-processing submodule, a map builder and a navigation submodule.

The one major problem with the contemporary forward-looking sonars is that they are not specifically designed for AUV obstacle avoidance. Most sonars employed by AUVs are 2-D mechanical scanning variants commonly employed by surface vessels. As such, the development of a novel, cost-effective, energy-efficient, obstacle-avoidance sonar should be forthcoming in order to address this critical issue. In the aspect of target tracking, the norm today is to employ the Kalman filter. Its performance in this particular case can be unreliable if certain assumptions are violated. Furthermore, the simultaneous tracking of a large number of dynamic targets can rapidly overload the system. To circumvent these disadvantages, a deviation or modification from the conventional Kalman filter is envisaged. It is clear from the previous discussion, that none of the single workspace representation methodology is superior in all circumstances. Significant improvement in performance can be attained by merging both the cell decomposition and geometrical map. Then again, such a hybrid scheme does, indeed, necessitate further research, particularly in the aspects of data synchronisation.

Deducing from the preceding discussion, there exists an obvious trade-off between energy efficiency, computational requirement and performance aspect amongst all the aforementioned submodules. Hence, a balance between all the requirements needs to be achieved in order to develop an effective AUV obstacle avoidance system. Nonetheless, a successful obstacle detection system is not the end of the story, it must also be accompanied by prudent evasive actions. This is the subject of obstacle avoidance and is delivered in Part B, which follows.

ACKNOWLEDGEMENT

Mr CS Tan would like to thank sincerely the IMarEST for its support through a Stanley Gray Fellowship Award.

REFERENCES

1. Vestård K. *Seabed Surveying with the HUGIN UUV.* IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 11th, pp 63-74, 1999.

2. Kato N, Ito Y, Kojima J, Asakawa K, and Shirasaki Y. *Guidance and Control of Autonomous Underwater Vehicle AQUA EXPLORER 1000 for Inspection of Underwater Cables.* IEEE Int Symposium on Unmanned, Untethered,

Submersible Technology, 8th, pp 195-211, 1993.

3. Butler B and den Hertog V. *Theseus: A Cable-laying AUV*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 8th, pp 1-6, 1993.

4. Anon. *AUV Orders Show Industry Commitment*. Int Ocean Systems Design, Sep/Oct 1999.

5. Griffiths G, Stevenson P, Webb AT, Millard NW, McPhail SD, Pebody M, and Perrett JR. *Open Ocean Operation Experience with the Autosub-1 AUV*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 11th, pp 1-12, 1999.

6. Ferguson J, Pope A, Butler B, and Verrall R. *Theseus AUV-Two Record Breaking Missions*, Sea Technology Magazine, pp 65-70, Feb 1999.

7. Yoerger DR, Bradley AM, Walden BB, and Cormier M. *Fine-Scale Seafloor Survey in Rugged Deep-Ocean Terrain with an Autonomous Robot*, IEEE Int Conferenceon Robotics and Automation, pp 34-55, 2000.

8. Foxwell D. Unmanned Vehicles, *Submarine-launched UUVs Swims Ahead*, WarshipTechnology, pp 6-7, May 2000.

9. Jordan K. *Remus AUV Plays Key Role in IraqWar*, Underwater Magazine, July/August 2003.

10. Lisiewicz JS and French DW. *The Naval Undersea Warfare Centre's Unmmaned Undersea Vehicle Initiative*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 11th, pp 86-93, 1999.

11. Tong A. Marlin, *The UK Military UUV Programme: A Programme Overview*, Proc. of Int UUV Symposium, pp 31-38, 2000.

12. Morrison NJ, Evans BS, James TS, and Allen KD. Gambit MCM AUV: *Overview and System Performance*, OCEANS 2003 Marine Technology and Ocean Science Conference, pp 1723-1729, Sept 2003.

13. Hobson B, Murray M, and Pell C. *Pilotfish: Maximizing Agility in An Unmanned Underwater Vehicle*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 11th, no 41-51, 1999.

14. Tan CS, Sutton R, and Chudley J. *Collision Avoidance Systems for Autonomous Underwater Vehicles, Part B: Obstacle Avoidance*, Journal of Marine Science and Environment, Part C2. IMarEST London, 2004.

15. Ridao P, Batlle J, Amat J, and Roberts GN. *Recent Trends in Control Architectures for Autonomous Underwater Vehicles*, Int Journal of System Science, Vol 30, No 9, pp 1033-1056, 1999.

16. Valavanis PK, Gracanin D, Matijasevic M, Kolluru R, and Demetriou GA. *Control Architecture for Autonomous Underwater Vehicles*, IEEE Control System Magazine, pp 48-64, 1997.

17. Caccia N, Virgili P, and Veruggio G. *Architectures for AUVs: A Comparative Study of Some Experiences*, AUVS'95 Int Symposium and Exhibition, pp 164-178, 1995.

18. Blidberg DR, Chappel S, Jaibert J, Turner R, Sedor G, and Eaton P. *The EAVE AUV Program at the Marine Systems Engineering Laboratory*. Proc of the 1st Workshop on: Mobile Robots for Subsea Environments, pp 33-42, Oct 1990.

19. Rock SM, Wang HH, and Lee MJ. *Task-Directed Precision Control of the MBARI/Stanford OTTER AUV*, Proc of the Int Program Development in Undersea Robotics & Intelligent Control (URIC): A Joint US/Portugal Workshop, pp 131-138, 1995.

20. Arbib M. *Perceptual Structures and Distributed Motor Control*, Handbook of Physiology the Nervous System II, pp 1449-1456, 1981.

21. Brooks R. A *Robust Layered Control System for a Mobile Robot*, IEEE Transactions on Robotics and Automation, pp 14-23, 1986.

22. Bellingham JG, Consi TR, and Beaton RM. *Keeping Layered Control Simple*, IEEE Proc Symposium Autonomous Underwater Vehicle Technology (AUV), pp 3-8, 1990.

23. Fujii T and Ura T. *Development of an Autonomous Underwater Robot 'Twin-Burger' for Testing Intelligent Behaviours in Realistic Environments*, Autonomous Robots, Vol 3, pp 285-296, 1996.

24. Ridao P, Batlle J, and Carreras M. *O^2CA^2 A New Object Oriented Control Architecture for Autonomy. The Reactive Layer*, Control Engineering in Practice Journal, 2001.

25. Yuh J and Choi SK. *Semi-Autonomous Underwater Vehicle for Intervention Missions*, Sea Technology Magazine, Vol 40, No 10, pp 31-40, 1999.

26. Healey AJ, Marco DB, McGhee RB, Brutzman DP, and Cristi R. *Evaluation of The NPS PHOENIX Autonomous Underwater Vehicle Hybrid Control System*, Proc American Control Conference, pp 477-484, 1995.

27. Hyland JC. *Optimal Obstacle Avoidance Path Planning For Autonomous Underwater Vehicles*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, pp 226-278, Jun 1989.

28. Arinaga S, Nakajima S, Okabe H, Ono A, and Kanayama Y. *Motion Planning Method for an AUV*, IEEE Proc Symposium Autonomous Underwater Vehicle Technology (AUV), pp 477-484, 1996.

29. Antonelli G, Chiaverini S, Finotello R, and Morgavi E. *Real-Time Path Planning and Obstacle Avoidance for an Autonomous Underwater Vehicle*, IEEE Int Conference on Robotics and Automation, Vol 1, pp 78-83, 2001.

30. Moitie R and Seube N. *Guidance Algorithms For UUVs Obstacle Avoidance Systems*, IEEE Oceans Conference Record, Vol 3, pp 1853-1860, 2000.

31. Lane DM and Trucco E. *Embedded Sonar and Video Processing for AUV Application*, Proc. Annual Offshore Technology Conference, Vol 2, pp 599-607, 2000.

32. Zanoli SM and Affaitati F. *Obstacle Avoidance with Scanning Sonar for Bottom Navigation*, IEEE Proc Symposium Autonomous Underwater Vehicle Technology (AUV), pp 537-545, 1999.

33. Loebis D, Sutton R, and Chudley J. *Review of Multisensor Data Fusion Techniques and Their Application to Autonomous Underwater Vehicle Navigation*, Journal of Marine Engineering and Technology, Vol 1, Part A1, pp 3-14, 2002.

34. Nussbaum F, Stevens GT, and Kelly JG. *Sensors for a Forward-Looking High Resolution AUV Sonar*, IEEE Proc.Symposium Autonomous Underwater Vehicle Technology (AUV), pp 141-145, 1996.

35. Henriksen L. *Real-Time Underwater Object Detection Based on an Electrical Scanned High-resolution Sonar*, IEEE Proc Symposium Autonomous Underwater Vehicle Technology (AUV), pp 99-104, 1994.

36. Loggins CD. *A Comparison of Forward-Looking Sonar Design Alternatives*, IEEE Oceans Conference Record, Vol 3, pp 1536-1545, 2001.

37. Veen BV and Buckley K. *Beamforming: A Versatile Approach to Spatial Filtering*, IEEE ASSP Magazine, Vol 5, pp 4-24, 1988.

38. Curtis TE and Ward RJ. *Digital Beamforming for Sonar*, IEE Proc Communication, Radar and Signal Processing, Vol 127, No Part F, 1980.

39. Lane DM, Chantler MJ, and Dai DY. *Robust Tracking of Multiple Objects in Sector Scan Sonar Image Sequences using Optical Flow Motion Estimation*, IEEE Journal of Oceanic Engineering, Vol 23, pp 31-46, Jan 1998.

40. Petillot Y, Ruiz IT, and Lane DM. *Underwater Vehicle Path Planning using a Multi-beam Forward Looking Sonar*, IEEE Proc Symposium Autonomous Underwater Vehicle Technology (AUV), pp 1194-1199, 1998.

41. Lane DM, Chantler MJ, Roberston EW, and McFadzean AG. *A Pyramidal Architecture for Knowledge-Based Sonar Image Intepretation*, IEE Colloquium on Transputers for Image Processing Applications, Feb 1989.

42. Mignotte M, Collet C, P'erez P, and Bouthemy P. *Sonar Image Segmentation using an Unsupervised Hierarchical MRF Model*, IEEE Transactions on Image Processing, Vol 9, No 7, pp 1216-1231, 2000.

43. Stewart D, Blacknell D, Blake A, Cook R, and Oliver C. *An Optimal Approach to SAR Image Segmentation and Classification*, DERA, Vol 9, No 7, pp 1216-1231, 2000.

44. Dai DY, Chantler MJ, Lane DM, and Williams N. *A Spatial-Temporal Approach for Segmentation of Moving and Static Objects in Sector Scan Sonar Image Sequences*, IEE 5th, Int Conference on Image Processing and Its Applications, No 163-167, 1995.

45. Lane DM, Chantler MJ, Roberston EW, and McFadzean AG. *Distributed Problem Solving Architecture for Sonar Image Interpretation*, Underwater Technology, Vol 14, pp 12-18, Fall 1988.

46. Lane DM, Chantler MJ, Dai DY, and Williams N. *Motion Estimation and Tracking of Multiple Objects in Sector Scan Sonar using Optical Flow*, IEE Colloquia on Autonomous Underwater Vehicles and Their Systems - Recent Developments and Future Prospects, pp 1-11, May 1996.

47. Moran BA, Leonard JJ, and Chryssostomidis C. *Geometric Shape from Sonar Ranging*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, pp 370-383, 1993.

48. Williams GN, Lagace GE, and Woodfin A. *A Collision Avoidance Controller for Autonomous Underwater Vehicles*, IEEE Proc Symposium of Autonomous Underwater Vehicle Technology (AUV), Jun 1990.

49. Ruiz IT, Petillot Y, Lane DM, and Bell J. *Tracking Object in Underwater Multibeam Sonar Images*, IEE Colloquium on Motion Analysis and Tracking, pp 1-7, May 1999.

50. Trucco E, Pettilot YR, Plakas K, and Lane DM. *Feature Tracking in Video and Sonar Subsea Sequences with Applications*, Computer Vision and Image Understanding, vol 79, pp 92-122, May 2000.

51. Dudek G and Jenkin M. *Computational Principles of Mobile Robotics*, Cambridge, UK: Cambridge University Press, 2000.

52. Movarec HP and Elfes A. *High Resolution Maps from Wide Angle Sonar*, IEEE Int Conference on Robotics and Automation, pp 116-121, 1985.

53. Allison JL, Watson DP, and Cook TM. *Intelligent Waypoint Transiting In Complex AUV Environment*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 6th, pp 246-257, Jun 1989.

54. Ridao P, Yuh J, Batlle J, and Sugihara K. *On AUV Control Architecture*, IEEE Int Conference on Intelligent Robots and Systems, Vol 2, pp 885-860, 2000.

55. Kambhampati S and Davis LS. *Multiresolution Path Planning for Mobile Robots*, IEEE Journal on Robotics and Automation, Vol 2, pp 135-145, 1986.

56. Yahja A, Stentz A, Singh S, and Brumitt BL. *Framed-Quadtree Path Planning for Mobile Robots Operating in Sparse Environments*, IEEE Journal on Robotics and Automation, pp 650-655, 1988.

57. Naylor BF. *Constructing Good Partitioning Trees*, Graphics Interface, pp 181-91, May 1993.

58. Lin MC, Manocha D, Cohen J, and Gottschalk S. *Collision Detection: Algorithms and Applications*, In Proc of the Algorithms for Robotics Motion and Manipulation, pp 129-142, 1996.

59. Leal J. *Stochastic Environment Representation*, PhD Thesis, Australia: University of Sydney, 2003.

60. Brutzman DP, Compton MA, and Kanayarna Y. *Autonomous Sonar Classification using Expert Systems*, IEEE Oceans Conference Record, pp 554-559, Oct 1992.

61. Caccia M, Bono R, and Veruggio G. *2-D Acoustic World Reconstruction*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 9th, pp 299-305, 1995.

62. Fox R, Garcia A Jr, and Nelson ML. *A Generic Path Planning Strategy for Autonomous Vehicles*, The University of Texas -Pan American, Department of Computer Science Technical Report CS-00-25, August 2000.

63. Garcia A Jr. *Overview of Path Planning Algorithms for Autonomous Vehicles*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 11th, pp 426-431, 1997.

64. Wang Y and Lane DM. *Subsea Vehicle Path Planning using Non-linear Programming and Constructive Solid Geometry*, IEE Proc Control Theory and Applications, Vol 144, pp 143-152, March 1997.

65. Mckendrick JD. *Autonomous Knowledge-based Navigation in an Unknown Two-dimensional Environment with Convex Polygon Obstacles*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, pp 258-265, 1989.

66. Liu C, Ang MH Jr, Hariharan K, and Yong LS. *Virtual Obstacle Concept for Local Minimum Recovery in Potential Based Navigation*, IEEE Int Conference on Robotics and Automation, Vol 2, pp 983-988, 2000.

67. Subramaniam LV and Bahl R. *Segmentation and Surface Fitting of Sonar Images for 3-D Visualization*, IEEE Int Symposium on Unmanned, Untethered, SubmersibleTechnology, 9th, pp 290-298, 1995.

68. Tomatis N, Nourbakhsh I, and Siegwart R. *Simultaneous Localisation and Map Building: A Global*

Topological Model with Local Metric Maps, IEEE Int Conference on Intelligent Robots and Systems, pp 421-426, Oct 2001.

69. Rikoski RJ, Leonard JJ, and Newman PM. *Stochastic Mapping Frameworks*, IEEE Int Conference on Robotics and Automation, Vol 1, pp 426-433, Oct 2002.

70. Carpenter RN. *Concurrent Mapping and Localization with FLS*, IEEE Proc Symposium on Autonomous Underwater Vehicle Technology (AUV), pp 113-148, 1998.

71. Zimmer UR. *Embedding Local Metrical Map Patches in a Globally Consistent Topological Map*, Proc of Underwater Technologies, May 2000.

72. Simhon S and Dudek G. *A Global Topological Map Formed by Local Metric Maps*, IEEE Int Conference on Intelligent Robots and Systems, Vol 3, pp 1708-1714, Oct 1998.

73. Hino JH. *Intelligent Planning for a Search and Surveillance Unmanned Underwater Vehicle*, IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 6th, pp 236-245, Jun 1989.

Collision avoidance systems for autonomous underwater vehicles Part B: a review of obstacle avoidance

C S Tan, R Sutton, J Chudley
Marine and Industrial Dynamic Analysis Group
School of Engineering
The University of Plymouth, UK

This paper reviews a variety of techniques that can be employed in an autonomous underwater vehicle (AUV) obstacle avoidance module. On the whole, there are two types of obstacle avoidance schemes. One is the motion planning approach and the other is the reflexive technique. Motion planning is employed for the generation of an optimal, obstacle-free trajectory. The reflexive avoidance technique, on the other hand, tends to be more ad hoc, focusing only on avoiding collisions. Both techniques are indispensable if one intends to realise a truly effective AUV collision avoidance system. The paper also includes a short survey concerning the 'Rules of the Road' that may be relevant to AUVs. The significance of these regulations cannot be underestimated, as one is seeing an increase of multiple AUV operations.

INTRODUCTION

In the context of an AUV, obstacle avoidance is concerned with the ability to avoid colliding with obstacles while navigating along a predetermined track. Unlike the term 'collision avoidance' which tends to denote both the detection and avoidance process, the obstacle avoidance process neglects the detection aspect as it is considered here as a separate process just for clarity of exposition. The interested reader should refer to Part A[1] for further details on obstacle detection. Perhaps, before delving deeper into the technical content, it is worth reviewing briefly the history of collisions, both at sea and in the air.

Since man's early foray into the oceans, mid-sea collision has been a frequent occurrence. The poor navigation technology at that time was partly to be blamed for this situation. Various regulations, or 'Rules of the Road', were enacted in the hope to mitigate the occurrence of such incidents. The prolific deployment of radar, a significant technology, in commercial vessels at the end of World War II was hailed as the solution to this centuries-old problem. However, much to the surprise of the maritime community even this innovative technology, regrettably, failed to prevent collisions from happening, as exemplified by the *Andrea Doria/Stockhom* disaster in 1956.[2,3] It was painfully clear subsequently, that most of the collision causes were not technology-related but were due to gross human errors, particularly the incorrect application of the technology and collision avoidance regulations. More recently, such unfortunate events have also been extended to the air domain, as confirmed by the mid-air collision between a DHL Boeing 757 and the Bashkirian Airlines Tupolev 154 near the Swiss/German border in July 2002.[4,5]

Undeniably, every collision either at sea or in the air has serious environmental, economical, and human life implications. To make matters worse, the current forecast indicates an escalating trend in the number of vessels, aircraft and their operations in the near future. This is set to increase the probability of collision as a result. Fortunately, this predicament has not been taken lightly by the UK's Civil Aviation Authority (CAA) who just published the CAP 722 report,[6] outlining some of the regulations pertaining to the legal and safety operations of unmanned aerial vehicles (UAVs). In the United States, NASA, being slightly more pragmatic, is employing a UAV named *Proteus* as a test bed for state-of-the-art collision avoidance technology.[7] It is envisaged that *Proteus* will be able to fly reliably and autonomously in national civil airspace within two years.

Although there exists some pioneering efforts in establishing certain public laws concerning AUV operations,[8,9,10] what is urgently lacking is an authority who can implement these instituted rules. One reason for the absence of interest in enforcing these rules is probably due to insufficient risk justification, especially the risk to human life. Unlike a UAV, which shares the same civil airspace as commercial aircraft, an AUV normally conducts its mission under the water where the chance of encountering another AUV or submarine is extremely unlikely at the moment.

AUTHORS' BIOGRAPHIES

See the preceding Part A of this paper

In spite of this, the current scenario is about to change as there is a sudden surge of interest in the field of multi-agent underwater robots. By working co-operatively and via mutual information sharing, these AUVs will be able to complete missions such as oceanographic sampling[11] and mine hunting with substantial reduction in both operational time and cost. This, as a result, necessitates a set of proper 'Rules of the Road' in order to safely and successfully conduct a multi-AUV mission. Starting with next section of this paper, some 'Rules of the Road' relevant to the AUV will be presented. In the section following it, different motion planning methodologies are surveyed with reference to both their advantages and disadvantages, and their implementation in AUVs. Motion planning is normally a computationally intensive process which explains its propensity to fail in time-critical conditions. As a corollary, one proceeds to investigate reflexive avoidance techniques in the penultimate section. Its primary function is to act as a contingency or backup system in the unfortunate event of motion planning failure. The final section provides the concluding remarks.

It is understandable that most of the collision avoidance algorithms implemented in an AUV are derived from the field of land-based mobile robotics. This is to be expected since the underlying perception is that both systems share significant similarities. However, it must be noted that terrain-based mobile robots do not suffer as severely from the problems that plague underwater vehicles, such as extreme non-linear dynamics, high degree of freedom, noisy sensor data, and the subjection to an unknown and varying environment.

In the forthcoming text, several notions – such as configuration space, holonomic system, non-holonomic system and under-actuated system – will be used pervasively. Therefore it is felt that an explanation of these notions is in order. Configuration space (C-space) is a fundamental tool introduced in the late 1970s to address the basic motion planning problem.[12] C-space is a set of all possible configurations of a robot or, to be more precise in this case, a vehicle. The dimensionality of a C-space is equivalent to the number of degrees of freedom (independent parameters) of the vehicle. For instance, Fig 1 shows a four-wheeled vehicle constrained to plane movement. One can describe the vehicle configuration using three variables; (x,y,φ), two translations and one rotation, concluding that this is a three dimensional C-space. Unlike a workspace, in a C-space the vehicle shape is 'patched' to the obstacles. Subsequently, the vehicle can be represented as a point which has the effect of simplifying the path planning process. In general, the high dimensionality of the C-space of nontrivial devices is perceived as the principal reason behind the complexities of a motion planning problem.[13]

Generally, a system is considered to be *fully actuated* when it has the same number of independent inputs as the configuration variables. Alternatively, the term holonomic is used to describe a system that has constraints as a function of configuration variables and time. Elsewhere, a non-holonomic system arises when the system has fewer control inputs than its configuration variables. These are generally characterised by non-integrable constraint equations involving the time derivatives of the system configuration variables. As stated in the configuration space section, the four-wheeled vehicle has only two control inputs (v, velocity and $\dot{\varphi}$, angular velocity) contrasting to three configuration variables, which results in a non-holonomic system. A first order non-holonomic relation can normally be written in the form of a time-invariant ordinary differential equation (ODE),

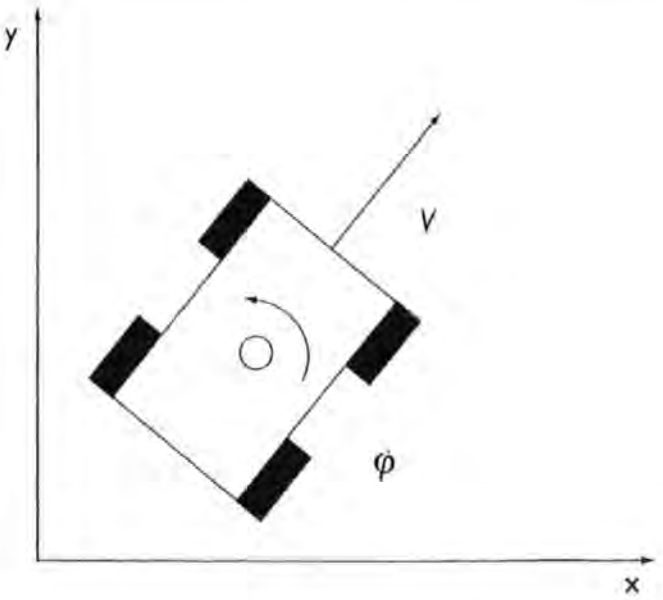


Fig 1: Diagram showing a four-wheeled vehicle configuration variables and control inputs

$$\dot{\mathbf{x}} = f(\mathbf{x}(t), \mathbf{u}(t)) \qquad (1)$$

that deals with only non-integrable velocity, where $\mathbf{x}$ is the state vector and $\mathbf{u}$ is the input vector. On the other hand, the second-order non-holonomic relation, can be written as,

$$\ddot{\mathbf{x}} = f(\dot{\mathbf{x}}(t), \mathbf{x}(t), \mathbf{u}(t)) \qquad (2)$$

which deals with non-integrable acceleration. This type of problem is also known as a kinodynamics problem[14, 15] and is frequently found in *under-actuated* systems such as surface vessels, spacecraft, manipulators and underwater vehicles. Suffice to say that controlling and motion planning for these systems is significantly more challenging than for the holonomic cases. A thorough review of non-holonomic controls and planning can be found in Laumond.[16]

RULES OF THE ROAD RELEVANT TO AN AUV

As stated earlier, the 'Rules of the Road' of AUVs pertain to a set of protocols or regulations applied to assist in tackling a collision predicament. Ironically, both marine vehicles and aircraft employ very similar regulations. The idea of incorporating these rules into an automatic collision avoidance system is not entirely new and has been essayed by various researchers.[17,18] Even so, their implementations are restricted to surface vessels. These selection of guidelines, as presented below, are derived from the International Regulations for Preventing Collisions at Sea.[19,20]

- **Rule 2** *Responsibility*, requires that 'due regard shall be given to all dangers of navigation and collision.' This rule allows an AUV to depart from all the rules as necessary to avoid the immediate danger of collision.

- **Rule 4** *Lookout*, requires that 'every vessel shall at all times maintain a proper lookout by all available means appropriate in the prevailing circumstances so as to make a full appraisal of the situation and of the possible risk of collision.' This is the primary task of the obstacle detection unit, where the primary lookout sensor employed is the sonar. There is even a suggestion that future AUVs shall be equipped with a system similar to the Identify Friend or Foe (IFF) unit commonly used in military aircrafts.

- **Rule 6** *Safe Speed*, requires that 'every vessel shall at all times proceed at a safe speed so that she can take proper and effective action to avoid collision and be stopped within a distance appropriate to the prevailing circumstances and conditions.' The speed of an AUV will be determined by these factors: the detectability, traffic density, manoeuvrability of the vessel with special reference to stopping distance and turning ability, the state of the sea, current, and proximity of navigational hazards. Slow speed, however, can affect the manoeuvrability of AUVs.

- **Rule 7** *Risk of Collision*, states that 'every vessel shall use all available means to determine if risk of collision exists; if there is any doubt, assume that it does exist.'

- **Rule 8** *Action to Avoid Collision*, states that 'changes in course and speed shall be large enough so as to be readily apparent to the other vessels. If necessary to avoid collision or allow more time to assess the situation, a vessel shall slacken her speed or take all way off by stopping or reversing her propulsion. A vessel which is required not to impede the passage of another vessel shall take early and substantial action to allow sufficient sea room for the passage of the other vessel.' Stopping and reversing the propulsion can, however, be problematic for a majority of AUVs which are underactuated and not neutrally buoyant, for example the loss of rudder effectiveness in low speed can induce higher collision risk instead.

- **Rule 14** *Head-On Situation*, states that 'vessels which are approaching head-on shall alter course to starboard (right-hand-side) so each will pass port (left-hand-side) to port.'

- **Rule 15** *Crossing Situation*, states that 'when two vessels are crossing so as to involve risk of collision, the vessel which has the other vessel on her starboard side shall keep out of the way, and shall, if the circumstances of the case admit, avoid crossing ahead of the other vessel.'

Note that for rules **8**, **14** and **15**, the general right-of-way rule states that the least manoeuvrable vessel has the right-of-way. For the case of a surface vessel, it is apparent that these manoeuvres occur in the planar domain. Whilst AUVs operate in a 3-D domain, for the moment their avoidance manoeuvres are still limited to only planar motion owing to the restriction imposed by the conventional 2-D obstacle avoidance sonar.

The airline industry is currently employing the Traffic Alert/Collision Avoidance System (TCAS). The concept is to create a virtual bubble around the aircraft and alert the pilot if there is any incursion to the protected zone around the aircraft. The simplest system, TCAS I only alerts the pilot on incoming threats and is referred to as a tactical advisory (TA) system. TCAS II[21] incorporates further feature enhancement to actually propose resolution advice (RA) in order to synchronise the vertical avoidance manoeuvre of both aircraft. This is achieved by the transmitting and receiving of interrogating signals, using a transponder, with the nearby aircraft. The latest, TCAS III, provides the pilot with a horizontal manoeuvre resolution advisory capability. The airline TCAS implicates the importance of a system or regulations that can propose complementary manoeuvres such that a collision can be avoided. Hence, to be truly effective, a consensus of these rules needs to be implemented in all AUVs.

MOTION PLANNING TECHNIQUES

Both motion planning and path planning can be defined as a problem of the form: given a configuration space, find a continuous sequence of configurations that leads from a start to a goal configuration while respecting certain constraints. However, the distinction is that motion planning tends to denote the generation of time parameterised solutions (trajectories) while, on the other hand, path planning neglects the time parameter. Simply stated, path planning does not take into consideration the vehicle dynamics. Both these terms will be used alternately depending on their suitability in a different context.

Owing to the inclusion of differential constraints, motion planning can also be considered as a search in a state space for a control input that can bring a system from an initial state to a goal state. Employing this perspective, one can directly associate a motion planning problem to a control engineering problem. Indeed, this promotes better problem assimilation and understanding. There is no dearth of literature regarding the theory of motion planning.[21,23] Thus, only a limited number of motion planning techniques that are associated with AUVs will be surveyed. Broadly speaking, motion planning approaches can be classified into three fundamental categories: cell-decomposition, road-map, and potential fields.

Cell-decomposition

One of the most popular motion planning schemes is the cell-decomposition. It is strictly related to the spatial decomposition scheme for workspace presentation. The fundamental idea is to represent the adjacent relation between the free cells with efficient structures such as a connectivity tree or a graph. They are then searched from the start to the goal state to find a sequence of states (path) that connects both the start and the goal state together. Various search algorithms that are based on dynamic programming exist for performing this routine. A few of the prevalent ones are; breadth-first search, depth-first search, best-first search, A^*, single-source shortest-distance algorithm (Dijkstra's algorithm)[24] and their unlimited variants.

The breadth-first search entails searching the neighbourhood cells, and expanding the list as it goes, while the depth-first search keeps probing in one path until an end is met, before trying the alternatives. Both search algorithms are exhaustive (complete), which means ultimately, all free space will be searched for solutions. For cases where multiple solutions exist, and optimality (shortest distance) is not a concern,

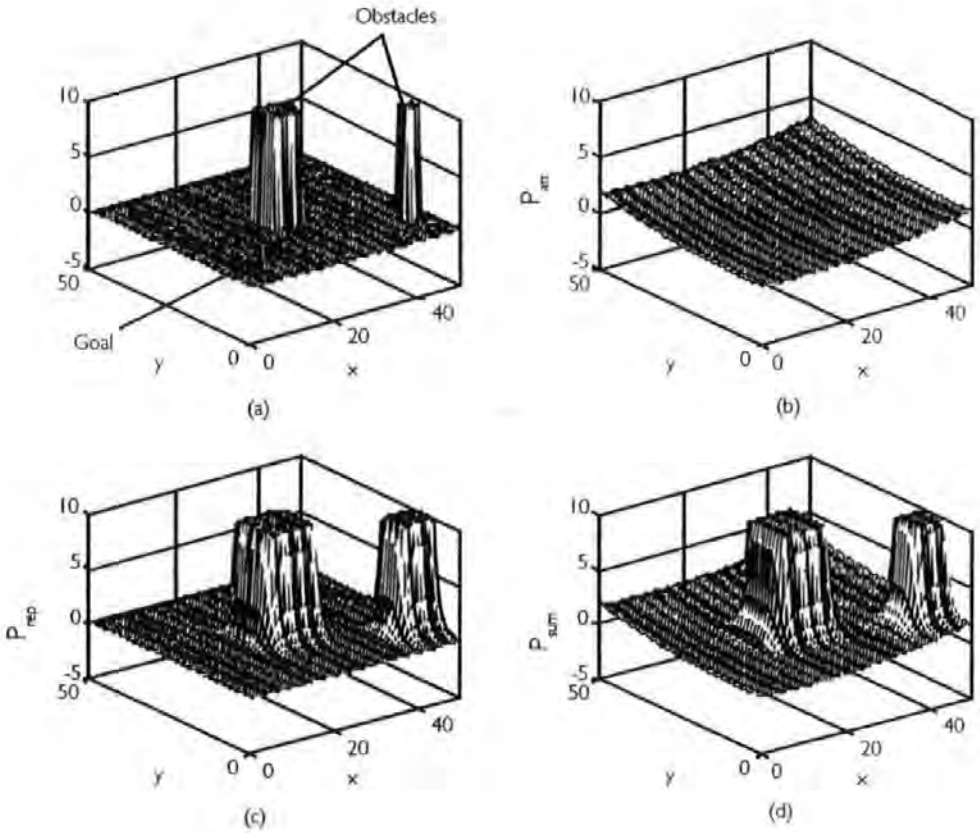


Fig 2: Potential field simulation; (a) workspace (b) attraction field (c) repulsive field (d) combined field

the depth-first search tends to have a lower memory requirement while providing a quicker answer. However, a depth-first search can be deceived into searching a long list of cells, or states, even when the goal may be very near. The Dijkstra's algorithm shares some resemblances with a breadth-first search, but unlike a breadth-first search, all the cells are encoded with distance from the goal which assists it in finding the shortest path. This search algorithm was applied in complement with a binary spatial partitioning scheme for the global path planning of the *Umihico* AUV.[25]

Nonetheless, in most circumstances, searching the entire free space can be too computationally-demanding. Unsatisfied with the performance of the former systematic search algorithms, some heuristically-enhanced versions have been devised. Heuristic information is normally encoded in an evaluation function (cost function). The distance to the Euclidean path (line-of-sight) from start to goal state is chosen as in the case of a best-first search. This scheme is efficient and fast when a proper evaluation function is provided, but for cases when this cannot be found, then its performance degrades significantly. The best-first search tends to provide suboptimal solutions since it neglects the cost of the solution path. The A^*[26] is a combination of the best-first search and the breadth-first search, which attempts to find a solution that minimises the total length of the solution path. The A^* method takes into account both the distance from the cell in question to the finish, and also the total distance taken from the start to the current cell. The evaluation function can be written as:

$$f(node) = g(node) + h(node) \qquad (3)$$

where $f(node)$ is the total cost, which is the evaluation function, $g(node)$ is the path cost to the current cell, and $h(node)$ is an estimate of the remaining cost to the goal state. A^* is guaranteed to find the shortest path if the $h(node)$ does not overestimate the cost to the solution.

Hyland[27] incorporated a 3-D A^* path planner with a reflexive obstacle avoidance module in his AUV simulation. The entire path is replanned by the path planner every time the vehicle completes a flat turn manoeuvre. Also, Hyland[28] provided a detail comparison between the breadth-first and the A^* search method for an AUV obstacle avoidance task. However, the results were inconclusive, as neither the A^* nor the breadth-first search shows any significant advantages in this case. Others, like Allision et al,[29] proposed a sensor-based exploration approach where a three-valued occupancy grid is coupled with the A^* algorithm evaluation function that is biased to search the unexplored region. It must be noted that these algorithms mentioned above, do not function optimally for cases when the environment is dynamic, partially known or unknown. The D^*, also known as Dynamic A^*[30], has been developed to address these issues. Owing to the nature of the problem, a substantial difference in performance can be obtained if one sets the initial search point as the start or the goal. Hence, some authors[31] prefer to use a bidirectional motion planning approach. Arinaga et al[25] employed this method for local path planning of the *Umihico* AUV. Their method involved moving the real AUV forward at the start point and a virtual AUV backward at the goal point simultaneously. Upon meeting, the real AUV is assigned to track the sequence of configurations created by the virtual AUV. Their method does necessitate a reflexive module for obstacle avoidance.

One of the apparent limitations of these search algorithms is the unrealistic computational requirement as the number of cells increase, a phenomenon known as the 'curse of dimensionality' or 'combinatorial explosion'. This might be caused by the increase of configuration space dimensionality or the scene complexity. For a heuristically-enhanced algorithm like the *A**, its performance is highly dependable on the selected evaluation function, which can be difficult to define for complicated problems.

Potential field

The potential field method utilises a very interesting approach. In essence, an artificial potential field is defined to emulate the space structure surrounding the vehicle.[32] It consists of representing the goal with an attractive field and the obstacles with a repulsive field, as shown in Fig 2. A new field emerges through the interaction of both the former fields. Eventually, the vehicle is required to just follow the local gradient of the new field to reach the goal.

The mathematical equations pertaining to the potential field method can be found in Appendix A. Fig 2(a) shows a simulated workspace representation of the vehicle. Using equation (A.3), an attractive field for the corresponding goal is simulated in Fig 2(b). Notice that the goal is the global minima, which is true for an ideal case. Using equation (A.5), Fig 2(c) shows the repulsion field exerted by the obstacles. Ultimately, Fig 2(d) illustrates the combined repulsive and attractive potential field as stated in equation (A.2). One major advantage of this method is its low computational requirement which makes it very suitable for real-time implementation.

Yoerger[33] employed a potential field local planner in the *Benthic Explorer* for a fine-scale rugged sea-floor surveying mission. The implementation is restricted to using an asymmetric potential field to alter the vehicle's forward and vertical speeds. One problem which is inherent to the potential field method is its tendency to get trapped in local minima. For this reason, it is normally used only as a local path planner, and in most implementations, it is combined with another global path planner that will be invoked when trapped. Its performance is also strictly linked to suitable definitions of heuristic potential functions, and this is not easily found when confronted with natural obstacles and differential constraints. Warren[34] proposed a hybrid method that involves two major stages. The first stage generates a preliminary straight path from current to goal configuration. Then, in the second stage, a method of path relaxation is introduced, the path is iteratively modified under the influence of the adjacent potential field in order to produce a feasible path. He argued that by considering the problem in such a global approach, the tendency of local minimum entrapment is significantly reduced. Instead of the conventional gradient descent method, Lane et al[35] reported using a preliminary tree search technique, to be specific, the best-first search to find the global minima. Local minima traps are avoided using the backtracking feature of this algorithm.

Some researchers encourage the use of a minimum-free function that is based on a harmonic potential field. The problem formulation is analogised to solving a fluid-flow problem such as fluid moving from the source location to the goal.[36,37] Nevertheless, this technique is not practical for a high dimensional problem as it is too costly in terms of computation demand.

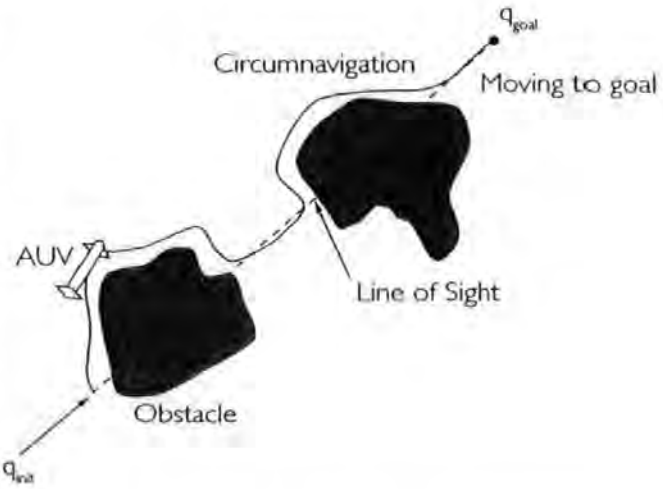


Fig 3: An AUV employing the bug algorithm to navigate the environment

Bug algorithm

The bug algorithm, also known as tangent bug or edge follower, is a particularly simple and yet remarkable path planning scheme. The main principle behind it is to trace the obstacle boundary and this is continued until the obstacle no longer blocks the desired path[38] (Fig 3). Fundamentally, the algorithm constitutes of only two modes: moving to the goal, and circumnavigating the obstacle. Note that this algorithm does not need any a priori information regarding its environment. Furthermore, it is guaranteed to find a solution if it exists. This makes the bug algorithm suitable for dealing with unknown environments.

Bennett et al[39] implemented the algorithm in the *Phoenix* AUV. A forward-looking sonar is used to detect the obstacle boundary, then it is approximated by aggregating piecewise linear lines before applying the bug algorithm. Alternatively, Cornforth and Croff[40] applied a wall-following algorithm in the *Autolycus* with the help of a side-facing sonar. Unfortunately, their current results were unsatisfactory but they anticipated further improvement can be realised by empirically tuning the controller gain. They envisaged using the *Autolycus* in environment-sensitive navigation. Better still, Laubach et al[41] devised a more memory-efficient approach that utilises only obstacle boundary endpoints. Their concept, however, is exemplified in a planetary exploration rover and not an AUV. This algorithm, although simplistic in concept, is extremely diffcult to be implemented in practice. Firstly, the influence of sensor drift in a dead-reckoning system tends to limit its effectiveness. In addition, this algorithm also assumes that the vehicle is holonomic and operating in a static environment, which is not entirely true for the case of an AUV.

Evolutionary computation (EC)

Evolutionary computation encompasses several types of heuristic and stochastic optimisation schemes that are fundamentally based on the concept of natural selection. Some of the proposed schemes are the evolutionary algorithm (EA), genetic algorithm (GA), evolutionary programming, evolutionary strategy and artificial life. The EC has shown signifi-

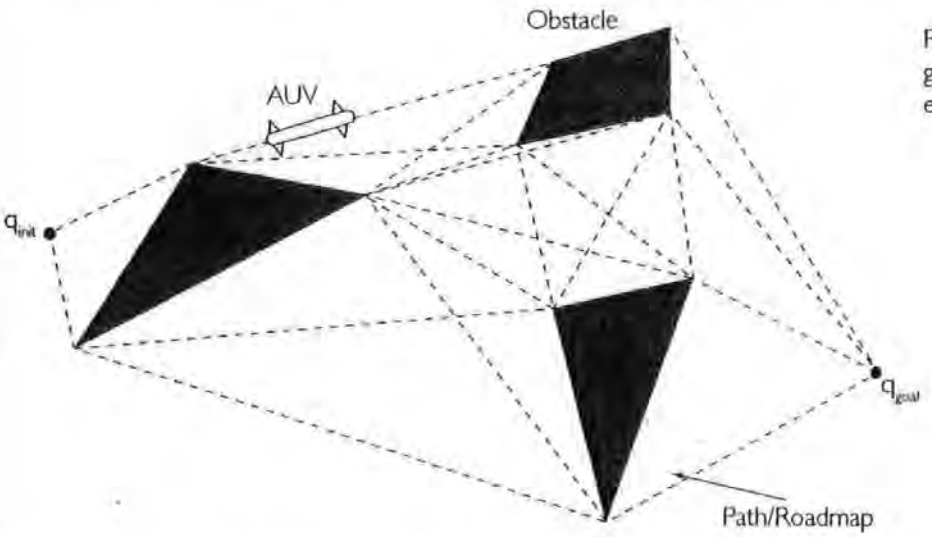


Fig 4: An AUV employing the visibility graph technique to navigate the environment

cant capability in solving complicated, highly-constrained, large-scale optimisation problems that have discontinuities on the response surface. Unlike the potential field method, EC is known to be highly resistant to becoming trapped in local minima. Another exceptional attribute of EC is its ability to offer solutions whenever it is interrupted.

Schultz[42] proposed using the GA for on-line collision avoidance and local navigation of an AUV. Promising simulation results of an AUV successfully navigating through both a static and dynamic minefield are presented. Similarly, Fogel et al[43] simulated 2-D optimal routing of multiple AUVs using an EA. Their simulations, although confined to only two-dimensional routings, still managed to demonstrate intriguing results. The AUV exhibited very intelligent behaviour by trying to avoid the detection region and, if that was not possible, the AUV proceeded at slower speeds to remain stealthy and speed up when it was a distance away from the detection site. Multiple AUV cases are also addressed. They argued that 'sophisticated' genetic operators such as crossover tends to disrupt the link between parent and offspring as coding structures become large. Sugihara[44] proposed a local GA 3-D path planner that is capable of functioning in a partially known environment for the *SAUVIM* AUV. He employed a method of discretisation, where the 2-D maps are partitioned into cells, and each corresponding cell is then encoded with a binary string as a sequence of pairs of direction and distance. Then, the three 2-D sequences of connected cells (paths), one in each respective plane, xy-plane, xz-plane and yz-plane, are merged via projection, into a single 3-D path.

Recently, there have been several attempts to hybridise evolutionary computation with other algorithms. Dozier et al[45] combined fuzzy inference along with tournament selection to select the best candidate paths based on several criteria. They claimed that the methodology does not only provide significant performance enhancement, but also obviates the need of explicit multi-objective evaluation function development. Alternatively, Vadakkepat et al[46] endeavoured to merge a potential field planning method with evolutionary programming to derive an optimal potential field function. The resulting algorithm is not only capable of solving the local minima problem but also handling dynamic obstacles.

One glaring weakness inherent in all evolutionary computation algorithms is its high computational requirement. Although, some authors purported to solve it using a distributed processing approach, this is not applicable in the case of an AUV, due to the energy limitation of the batteries. Furthermore, their long convergence time makes them unsuitable for a highly dynamic environment. Other weaknesses include difficulty in finding the exact global optimum and their performances are highly dependable on how the problem is structured and encoded.

Visibility graph

The visibility graph is a subset of the road-map approach to path planning. The operation of this scheme can be depicted as following: initially, all the vertices of all polygonal obstacles – including the start and goal – that are in a line-of-sight with respect to each other are connected as shown in Fig 4. This process is simplified if one limits the environment with only convex obstacles. Then, by representing the vertices as nodes, and the path as an edge, any tree search algorithms, as elaborated in the cell-decomposition sub-section of this paper, can be used to find the shortest path. Unfortunately, one obvious disadvantage of using a visibility graph is the assumption that all of the obstacles are known. Furthermore, a visibility graph has the tendency to generate paths that are very close to the obstacles' edges. One simple solution is to 'patch' the obstacles in order to take into account the vehicle geometry. However, this is not a trivial process if the vehicle considered is under-actuated.

McKendrick[47] applied a visibility graph method in an unknown 2-D environment with convex polygonal obstacles. To be realistic, the AUV was simulated with a limited sensor range. An exploration phase is then required for information acquisition. The simulation demonstrated that the path is highly inefficient, taking long detours and, as such, a simple bug algorithm easily surpasses its performance.

Probabilistic road-map planner (PRM)

This is still a relatively new approach to path planning, where the construction of the road-map is done probabilistically instead of deterministically. The main concept is to generate a number of nodes (vertices) randomly, eliminate those nodes in the obstacles, and then connect all the adjacent nodes with straight lines. The primary reason that only adjacent nodes are connected is to avoid saturating the configuration space

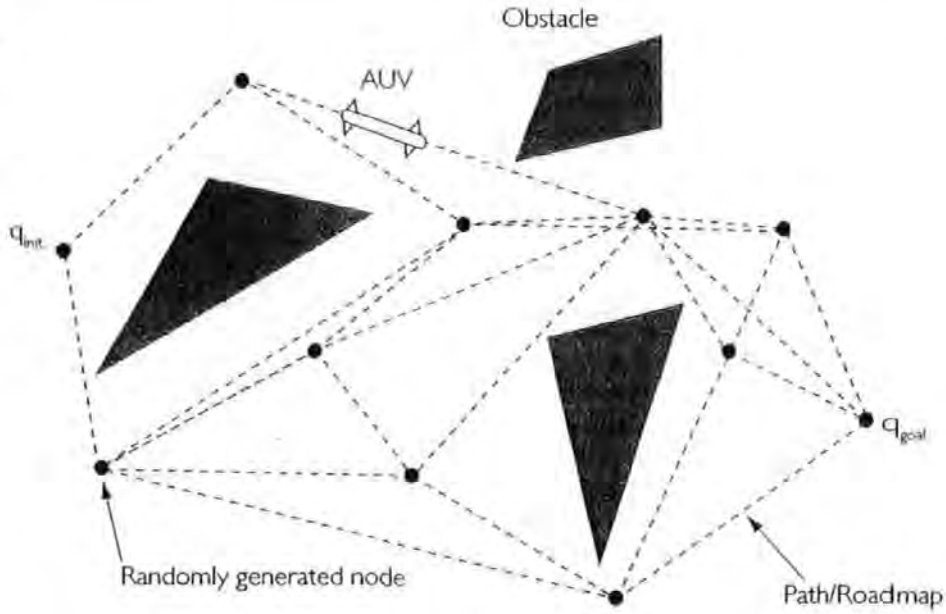


Fig 5: An AUV using the probabilistic road-map method to navigate the environment

with too many paths. Later, the resulting road-map is searched from the start point to the goal point for the shortest path (Fig 5). Unlike other motion planning methods, its randomised nature tends to make its performance less susceptible to the effect of configuration space dimension.[48] However, it does compromise solution optimality for enhanced robustness. Consequently, this method generally produces suboptimal solutions. PRM is also notoriously known for its long running-times and difficulty in finding a path in a configuration space that has a small passage. Therefore, some heuristically-enhanced PRMs such as visibility PRM,[49] lazy PRM,[50] obstacle-based PRM,[51] and Gaussian sampling PRM,[52] have been proposed to improve the generic PRM performance. These methods mostly differ from their sampling strategies. Fox et al[53,54] proposed an enhanced PRM that shares some similarity with the lazy PRM. They envisaged using it as a generic path planner for an AUV. Their method relies on generating a line-of-sight path from the initial point to the goal point. When an obstacle intersection occurs, a few points are generated at the obstacle's vicinity to reroute the path. This process is then repeated until a solution is found or a time-limit is reached. Their scheme assumes a static environment, thus a reactive planner is required for avoiding a dynamic obstacle.

Rapidly-exploring random tree (RRT)

The majority of path planning algorithms, as mentioned above, only take into account the algebraic constraints (caused by obstacle) and not differential constraints (caused by kinematics and dynamics). The paths generated are typically rigid with abrupt directional changes, making it not viable for vehicle actuator operations due to saturation. This becomes an impetus for a search for a scheme that can address both the path planning and control issues simultaneously. Several potential benefits arise from the integration of the path planning and control process. Assuming no disturbance, the paths generated are guaranteed to be reproducible in practice. A link optimisation for both path distance and control effort can also be achieved. Cases of differential constraints can easily be incorporated, thus extending its application into fault-tolerant and reconfigurable systems for an AUV.[55,56]

Most systematic search algorithms do not function well in high-dimensional space. This has prompted the introduction of the RRT[57,58] which can be considered as an incremental

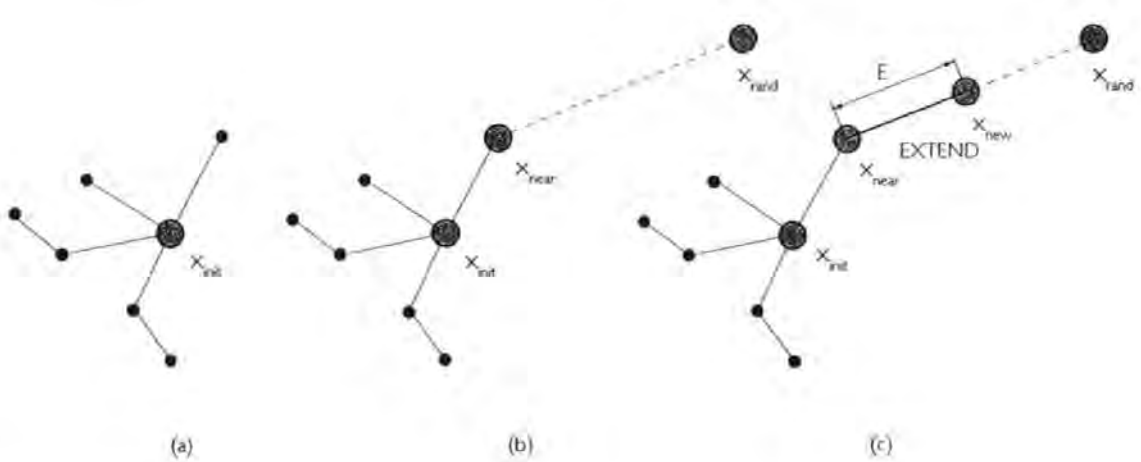


Fig 6: RRT operations; (a) shows a partial RRT tree with its initial state, x_{init}(b) shows the inclusion of random state, x_{rand} and selection of the nearest state, x_{near} (c) shows the addition of new state, x_{new} and its connection to near state, x_{near}

form of PRM and is designed to search efficiently non-convex high-dimensional spaces. It possesses a few fascinating properties.

1. It is biased to the free-space and exploits a probabilistic search method.
2. It has also been proven to be probabilistically complete.[59]
3. The simplistic nature of the algorithm facilitates performance analysis.
4. Lastly, it allows one to take into account both algebraic and differential constraints simultaneously, which is vital for motion planning.

To better appreciate the RRT, one needs to understand the principle of operation behind the algorithm: starting from an initial state, x_{init}, a tree is grown through a process of adding edge and vertex in each time step (iterations). Assuming that one has a partial RRT tree as shown in Fig 6(a), the second stage is to introduce a random state, x_{rand}, Fig 6(b). The EXTEND function determines the 'nearest' state to the random state to be extended. The 'nearest' state is typically defined by a metric. As shown in Fig 6(c), a new state, x_{new}, that is E-distance away from x_{near} is then computed and added into the RRT. The computation is required to find a suitable input, u_{new}, that is applied for a time increment, Δt so that it can bring x_{near} to x_{new}. Moreover, one is also required to determine if x_{new} has reached the goal state, or failed to find a suitable state. The process is then repeated.

Cheng et al[60] employed the RRT to optimise the trajectories of autonomous automobiles and spacecraft. The simulations show the viability of the method. Toussaint[61] tried to combine motion planning using the RRT with non-linear control employing the H_∞ technique for an under-actuated vehicle. Not only did he utilise a H_∞ filter for improving the planned motion of the vehicle, but he also attempted to address multiple vehicles' planning problems. Again his simulation is limited to only planar motion. The RRT has also been applied to solve non-linear control problems in hybrid systems. Frazzoli et al[62,63] provided some realistic simulations of unmanned-helicopter motion planning that employed the RRT. Unlike other path planning algorithms, he mentioned that the RRT is capable of exploiting fully the manoeuvrability of the helicopter.

Nonetheless, the RRT is not without problems. Firstly, as a novel algorithm, its capability is still not well characterised. Furthermore, its performance is highly sensitive with respect to the chosen metric. An incorrect metric will substantially deteriorate its performance. Cheng et al[64] described a technique to render the RRT less sensitive to the metric effect. In all of the experiments and simulations mentioned above, a known environment is assumed, and this is unrealistic for AUVs, hence supporting the fact that there is still room for further improvement.

REFLEXIVE AVOIDANCE TECHNIQUES

Reflexive avoidance techniques provide an AUV with a fail-safe mechanism in the case of motion planner failure. The failure can either be a system malfunction or a failure to meet the predefined time constraint, which is a more common occurrence than the former. Technically, these techniques are fundamentally based on the reactive control approach,[65] where the information from the sensors is sent directly to the actuator without passing through the high-level modules. This makes them amazingly fast and capable of handling dynamic environments especially in cases where in situ response is needed. Unfortunately they suffer from the identical problems that plague reactive control systems. Some of the problems are; highly non-optimal action, non-deterministic performance and, lastly, they are prone to get trapped in a canyon-like environment.

Neural network

The neural network approach has been intensively researched over the last 20 years, and it still ongoing. The core of the neural network concept is based on a mathematical emulation of simplified human brain mechanisms. One obvious benefit of the neural network* is its intrinsic ability to model a very complex, multi-input/multi-output, and strongly coupled non-linear system such as an AUV. It is also highly renowned for its 'generalisation' capability. Unlike other algorithms that are programmed, a neural network* is trained by exposing it to related input/output data. Properly trained, the neural network can provide an elegant solution to a very challenging problem.

DeMuth et al[66] proposed a neural-network-based obstacle avoidance controller for an AUV. They used two neural networks, one for a static object and another for a dynamic object classification. Then, a Boolean combiner is employed to reconcile the appropriate manoeuvre to be taken. Paradoxically, their neural network controllers were not trained but the weights were heuristically determined. Clearly, this is only applicable for a very simple case. In trying to exploit the adaptive nature (training ability) of a neural network, Sayyadi et al[67] applied a stochastic real value reinforcement learning method to the collision avoidance controller of the *Twin Burger*. They divided the obstacle avoidance mission into a targeting behaviour and an avoiding behaviour. Nonetheless, no tangible results were given concerning the avoiding behaviour performance as the research is still at a premature stage.

Interestingly, one tends to find more neural network applications in low-level controller design[68,69] than at the higher level. This could be caused by the black box characteristic of neural network, which precludes vital information extraction that can be crucial for problem understanding. Furthermore, a high-level controller typically exerts more influence over the entire system performance where a small error tends to amplify quickly. This, in turn, demands a transparent system for analysis purpose which is not offered by such a network. The training processes can also be very time consuming, depending much on the 'suitableness' of the selected training data. To foster rapid convergence, another data pre-processing stage is also found to be compulsory.

Virtual Force Field (VFF)

The virtual force field method tries to simulate an artificial force field which can be two- or three-dimensional, surrounding the vehicle. Thus any contact with neighbourhood objects will cause a deformation of the force field. The aim here is to minimise these deformations by locally modifying the control

* The majority of neural networks such as the popular multi-layer perceptron and radial basis network do require supervised training.

vector. Its computational efficiency and fast reaction make it a valuable technique in a dynamic environment. Recalling the fact that the TCAS employed by the aviation industry also utilises a form of virtual bubble, which is a variant of VFF, Zapata et al[70] addressed the collision avoidance and bottom following problems of an AUV using a VFF scheme. Their results, however, are confined to computer simulations only.

VFF shares some similarity with the potential field method, both trying to simulate the interaction of an artificial field between objects and the vehicle. However, VFF only considers the forces in a limited neighbourhood domain thus it is highly susceptible to being trapped in local minima. Worse still, it can also lead to unstable behaviour of the vehicle when surrounded with obstacles or travelling in a narrow passage. To solve this problem, a high-level planner is usually introduced.

Vector field histogram (VFH)

The vector field histogram[71] was invented to solve some of the problems of the VFF algorithm concerning detail spatial distribution information loss. The VFH is a data reduction process algorithm that can be decomposed into three distinct phases. The first phase entails representing the vehicle workspace as a two-dimensional grid. The second phase involves constructing the vehicle surrounding into local polar histogram form where each sector represents obstacle density. The last phase involves a selection of the sector of the lowest obstacle density and alignment of vehicle heading to the selected sector. Its performance in most circumstances is better than the VFF.[72]

Williams[73] proposed a 3-D collision avoidance controller that has the intrinsic functionality of VFH. He exploited a merit function that defines a field that takes into account obstacle bearing and distance, as well as the vehicle's own heading, depth and the goal direction. For the three-dimensional case, a data reduction process is performed to transform the presentation into an image showing different obstacle density. Consequently, the vehicle is just required to align its heading vector to the lowest obstacle density area. Antonelli et al[74] attempted to integrate the VFF and geometrical approach proposed by Hyland,[27] and implemented it in the *RAIS* AUV. Their approach takes into account the polar information instead of only the Cartesian, making the algorithm very similar to VFH. They also employed a high-level path planner to detect the high risk area that can trap an AUV.

The VFH tends to take into account the workspace Cartesian information, hence it is less affected by the local minima entrapment problem as suffered by the VFF method. However, this does not mean that it is impervious to local minima entrapment. Furthermore, by discarding all explicit distance information and representing them implicitly, the vehicle is sometimes deceived into assuming that there are no clear sectors when it is surrounded with only distanced obstacles.

Fuzzy logic

Fuzzy theory was introduced by Zadeh[75] in 1965 as an alternative technique for tackling complicated problems that are difficult to solve using conventional differential-equation-based approaches. In essence, fuzzy logic is a rule-based, multi-value logic inference system that attempts to take into account the uncertainty and imprecision of the real world. Its intrinsic operational principle bears substantial resemblance with human cognition. In fact, the fuzzy logic ability to quantify abstract expert knowledge, has made it a choice in solving complicated systems. Consequently, the control decisions of an expert can be formulated into an algorithm to control the desired plant. One example of fuzzy rule base in a collision avoidance context is:

IF *Target Direction* is *Left* AND *Target Range* is *Very Near* THEN *Heading* is *Hard Right*.

Clearly, the rule above is self-explanatory which facilitates problem understanding. As such, a set of rules can be promptly constructed without resorting to complex mathematical techniques. Its transparent nature and excellent immunity to both noise and error also contributes to its popularity.

Shinjo et al[76] proposed a collision avoidance system that is based on a combination of sensor-based navigation and fuzzy logic control. The fuzzy logic inference system provides a mapping framework to transform the acquired object information such as range, position, and size, into the respective control commands for heading and vertical movement of the vehicle. A short-term memory is also used to store successive obstacle avoidance processes with the objective to reduce abrupt changes or chattering of the control command outputs. This is achieved via reducing the degree of membership of the last executed fuzzy conclusion in order to reduce its dominance. The analysis of the algorithm was performed on a full 6-DOF *Ocean Voyager* AUV simulator. Instead of encoding the problem directly as in the former approach, Vasudevan et al[77] attempted a hybrid reasoning scheme by aggregating fuzzy rule sets and case-based reasoning to function as a high-level dynamic path selector. In what was called Reasoning from Fuzzy Indexed Cases Scheme (REFIC), it attempts to exploit the a priori information such as the prestored cases to assist in determining a promising vehicle-heading and also in selectively activating a subset of navigational behaviours. The simulated example proved to be very robust in navigating in the presence of noisy sensor data and cluttered obstacle distributions. Liu et al[78] tried to tackle AUV navigation in an unknown environment by creating a virtual boundary and incorporating some heuristic rules via fuzzy logic. The vehicle emergence behaviour turns out to be very similar to the bug-following algorithm. However, the effects of local minima and sea current were neglected in the simulation. Ridao et al[79] implemented a collision avoidance controller in the *Garbi* AUV using a combination of VFF and fuzzy logic behavioural encoding techniques. The implementation is, however, limited to the horizontal plane. The vehicle is surrounded with several circular force fields of varying radii and each particular region is then mapped into the corresponding behaviours such as go-to, spin, avoid, keep depth and avoid bottom. A simulation study has verified that the vehicle is able to exhibit 'intelligent' manoeuvres such as circumnavigating and escaping from a canyon-like trap.

Although, many praises can be made regarding a fuzzy logic system, there is also an equivalent amount of criticisms. Owing to the heuristic method upon which the fuzzy logic paradigm is fundamentally based, a multitude of incoherent and diverse viewpoints exist regarding the types and fuzzy operators used. The lack of a solid framework also tends to

make fuzzy logic appear to be an ad hoc approach to finding a solution. Although, a fuzzy system is renowned for its transparency property, it is virtually mathematical intractable and can complicate analysis.

CONCLUDING REMARKS

A review of collision avoidance systems in AUVs, tackled in two parts, has been presented. Part A concentrates on the obstacle detection aspects, while in Part B the emphasis is on the obstacle avoidance techniques. The obstacle avoidance module consists of two crucial components, the motion planning and reflexive avoidance submodules. A motion planning submodule is required to generate an 'optimal' path. Strictly speaking, the majority of the motion planning methodologies mentioned are, in fact, considered to be path planning methodologies. They do not take into account the dynamics of an AUV. Systematic search techniques, although extremely popular in the robotics community, unfortunately, do not function well in higher-dimensional search space which is found so commonly in most practical systems. The more-recent probabilistic techniques are more robust with respect to the dimensionality effect. Then again, their performance is erratic and can be unreliable occasionally. Owing to their more recent discovery, their attributes are not very well characterised, hence vindicating the need for more in-depth research. In terms of reflexive avoidance techniques, the fuzzy logic methodology outshines the rest by providing transparency, ease of use, flexibility and robustness attributes in one complete package. Its inherent rule-based capability could be exploited in conjunction with the 'Rules of the Road' to achieve a seamless high-level controller integration. These rules are proposed in the expectation that the forthcoming AUVs will, at least, be compliant with these regulations, hence minimising the risk of collision.

In conclusion, it is anticipated that with the fusion of these different methodologies, as presented in Part A and Part B, a robust and computationally efficient collision avoidance system can be realised. In designing such a system, one must consistently bear in mind that an effective collision avoidance system derives its success through the synergistic interaction of submodules, and not because of a particular submodule functionality. Similarly, the application of these methodologies could also offer potential technological advances in the field of AUV collision avoidance while simultaneously benefiting the marine industry.

ACKNOWLEDGEMENT

Mr CS. Tan would like to thank sincerely the IMarEST for their support through a Stanley Gray Fellowship Award.

REFERENCES

1. Tan C, Sutton R, and Chudley J. *Collision Avoidance Systems for Autonomous Underwater Vehicles, Part A: Obstacle Detection*. IMarEST Journal of Marine Science and Environment, Part C2. 2004.

2. Moscow A. *Collision Course, the* Adrea Doria *and the* Stockholm. New York: Putum and Sons, 1959.

3. Cahill R, *Collisions and Their Causes*. US: Nautical Books, 2nd ed, 1997.

4. BBC News, *Relatives visit jet crash site*. BBC News, 2002.

5. Harro R and Fabian L. *Accident description*. Aviation Safety Network, 2002.

6. UK Civil Aviation Authority (CAA). *Cap 722: Unmanned aerial vehicle operations in UK airspace-guidance*. Directorate of Airspace Policy, 2002.

7. Aerotech News and Review. *Flight Demonstrations Evaluate UAV Collision Avoidance Technology*. Online Journal of Aerospace and Defence Industry News, April 2003.

8. *The Operation of AUVs: Recommended Code of Practice*. Vol 1. Society of Underwater Technology (SUT), UK, 2000.

9. *The Operation of AUVs: Report On the Law*. Vol 2. Society of Underwater Technology (SUT), UK, 2000.

10. *The Operation of AUVs: The Law Governing AUV Operations-Questions and Answers*. Vol 3. Society of Underwater Technology (SUT), UK, 2000.

11. Chappell S, Turner R, and Turner E. *Cooperative Behaviour in an Autonomous Oceanographic Sampling Network: MAUV Project Update*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, pp 375-384, 1997.

12. Lozano-Perez T and Wesley M. *An Algorithm for Planning Collision-free Paths among Polyhedral Obstacles*. Communications of the ACM, Vol 22, pp 560-570, Oct 1979.

13. Hwang YK and Ahuja N. *Gross Motion Planning - A Survey*. ACM Computing Surveys, Vol 24, pp 219-291, Sep 1992.

14. Amato N and Wu Y. *A Randomized Roadmap Method for Path and Manipulation Planning*. IEEE Int Conference on Robotics and Automation, pp 113-120, Jan/Feb 1996.

15. LaValle S and James K. *Randomized kinodynamic planning*. IEEE Int Conference on Robotics and Automation, pp 473-479, 1999.

16. Laumond J. *Robot Motion Planning and Control*. Springer, 1998. ISBN 3 540 76219 1.

17. Pietzykowski Z. *The Analyis of a Ship Fuzzy Domain in a Restricted Area*. IFAC Con Control Applications in Marine Systems, pp 1-6, 2002.

18. Tran T, Harris C, and Wilson P. *A Vessel Management Expert System*. Proc Institute of Mechanical Engineers: Engineering for the Maritime Environment, vol 216, Part M:J, pp 161-177, 1997.

19. Brown H, *Brown's Rule of the Road Manual*. Brown, Son & Ferguson, Ltd, 7th ed, Glasgow, UK 1983.

20. Cockcroft A and Lameijer J. *A guide to the Collision Avoidance Rules*. Butterworth-Heinemann, Oxford, UK, 5th ed, 2001.

21. Abdul-Baki B, Baldwin J, and Rudel MP. *Independent validation and verification of the TCAS II collision avoidance subsystem*. IEEE, AIAA Annual Digital Avionics Systems Conference, 1999.

22. Latombe J. *Robot Motion Planning*. Kluwer Academic Publishers, USA, 1991.

23. Fujimura K, *Motion Planning in Dynamic Environments*. Springer-Verlag, Tokyo,1991.

24. Dijkstra E. *A Note on Two Problems in Connection*, Numerical Mathematics, Vol 1, pp 269-271, 1959.

25. Arinaga S, Nakajima S, Okabe H, Ono A, and Kanayama Y. *Motion Planning Method for an AUV*. IEEE

Proc Symposium Autonomous Underwater Vehicle Technology (AUV), pp 477-484, 1996.

26. Hart P, Nilsson J, and Raphael B. *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*. Transactions on Systems, Man and Cybernetics, pp 100-107, 1968.

27. Hyland J. *Optimal Obstacle Avoidance Path Planning For Autonomous Underwater Vehicles*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, pp 226-278, Jun 1989.

28. Hyland J. *A Comparison of Two Obstacle Avoidance Path Planning For Autonomous Underwater Vehicles*. IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV), Jun 1990.

29. Allison J, Watson D, and Cook T. *Intelligent Waypoint Transiting In Complex AUV Environment*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 6th, pp 246-257, Jun 1989.

30. Stentz A, *Optimal and Efficient Path Planning for Partially-Known Environments*. IEEE Int Conference on Robotics and Automation, No 4, pp 3310-3317, 1994.

31. Arai H, Tanie K, and Shiroma N. *Time-scaling Control of An Underactuated Manipulator*. IEEE Int Conference on Robotics and Automation, pp 525-536, May 1998.

32. Krogh B and Thorpe C. *Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles*. IEEE Int Conference on Robotics and Automation, pp 1664-1669, 1986.

33. Yoerger D, Bradley A, Walden B, and Cormier M. *Fine-Scale Seafloor Survey in Rugged Deep-Ocean Terrain with an Autonomous Robot*. IEEE Int Conference on Robotics and Automation, pp 34-55, 2000.

34. Warren C. *A Technique for Autonomous Underwater Vehicle Route Planning*. IEEE Journal of Oceanic Engineering, Vol 15, pp 199-204, October 1990.

35. Lane D and Trucco E. *Embedded Sonar and Video Processing for AUV Application*. Proc Annual Offshore Technology Conference, Vol 2, pp 599-607, 2000.

36. Li G and Bui TD. *Robot Path Planning using Fluid Model*. Journal of Intelligent and Robotic Systems, Vol 21, pp 29-50, 1998.

37. Louste C and Liegeois A. *Near Optimal Robust Path Planning for Mobile Robots*. Journal of Intelligent and Robotic Systems: Theory and Applications, Vol 27, No 1-2, pp 99-112, 2000.

38. Kamon I, Rimon E, and Rivlin E. *A New Range-Sensor Based Globally Convergent Navigation Algorithm for Mobile Robots*. CIS-Center of Intelligent Systems 9517, Computer Science Dept Technion, Israel, 1995.

39. Bennet A and Leonard J. *Behaviour-Based Approach to Adaptive Feature Detection and Following with Autonomous Underwater Vehicles*. IEEE Journal of Oceanic Engineering, Vol 25, No 2, pp 213-226, 2000.

40. Cornforth W and Croff K. *Development of an Environment-Sensitive Navigation System for the AUV Autolycus*. Marine Technology, Vol 37, pp 238-245, October 2000.

41. Laubach S and Burdick J. *An Autonomous Sensor-Based Path-Planner for Planetary Microrovers*. IEEE Int Conference on Robotics and Automation, pp 1-8, 1999.

42. Schultz A. *Using a Genetic Algorithm to Learn Strategies for Collision Avoidance and Local Navigation IEEE Int Symposium on Unmanned*, Untethered, Submersible Technology, pp 213-215, Sep 1991.

43. Fogel D and Fogel L. *Optimal Routing of Multiple Autonomous Underwater Vehicles Through Evolutionary Programming*. IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV), Jun 1990.

44. Sugihara K. *GA-based On-line Path Planning for SAUVIM*. Int Conference on Industrial Engineering & Applications of Artificial Intelligence & Expert Systems, Vol 2, pp 329-338, 1998.

45. Dozier G, McCullough S, Homaifar A, Tunstel E, and Moore L. *Multiobjective Evolutionary Path Planning via Fuzzy Tournament Selection*. IEEE Int Conference on Evolutionary Computation (ICEC'98), pp 684-689, 1998.

46. Vadakkepat P, Tan KC, and Wang M. *Evolutionary Artificial Potential Fields and Their Application in Real Time Robot Path Planning*. Congress of Evolutionary Computation (CEC'2000), pp 256-263, Jul 2000.

47. Mckendrick J. *Autonomous Knowledge-based Navigation in an Unknown Two-dimensional Environment with Convex Polygon Obstacles*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, pp 258-265, 1989.

48. Overmars M. *Recent Developments in Motion Planning*. Int Conference on Computational Science (3), pp 3-13, May 2002.

49. Simeon T, Laumond J, and Nissoux C. *Visibility-based probabilistic Road-maps for Motion Planning*. Advanced Robotics Journal, Vol 14, No 6, pp 371-386, 2000.

50. Bohlin R and Kavraki L. *Path Planning using Lazy (PRM)*. IEEE Int Conference on Robotics and Automation, pp 521-528, May 1998.

51. Amato N, Bayazit O, Dale L, Jones C, and Vallejo D. *OBPRM: An Obstacle Based PRM for 3-D Workspaces*. Workshop on Algorithmic Foundations on Robotics (WAFR), May 1998.

52. Boor V, Overmars M, and Stappen A. *The Gaussian Sampling Strategy for Probabilistic Road-map Planners*. IEEE Int Conference on Robotics and Automation, pp 1018-1023, 1999.

53. Fox R, Garcia A, and Nelson M. *A Generic Path Planning Strategy for Autonomous Vehicles*. The University of Texas-Pan American, Department of Computer Science Technical Report CS-00-25, August 2000.

54. Garcia A. *Overview of Path Planning Algorithms for Autonomous Vehicles*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 11th, pp 426-431,1997.

55. Perrault D and Nahon M. *Fault-Tolerant Control of an Autonomous Underwater Vehicle*. Oceanic Engineering International, Vol 3, No 2, pp 85-94, 1999.

56. Sutton R, Pearson A, and Tiano A. *A Fuzzy Fault Tolerant Control Scheme for an Autonomous Underwater Vehicle*. IEEE Int Conference on Methods and Models in Automation and Robotics, pp 595-600, 2001.

57. LaValle S. *Rapidly-exploring Random Trees: A New Tool for Path Planning*. TR 98-11 computer Science Dept, Iowa State University, Oct 1998.

58. LaValle S and Kuffner J. *Rapidly-exploring Random Trees: Progress and Prospects*. In; Workshop on the

Algorithmic Foundations of Robotics, pp 293-308, Oct 2000.

59. Cheng P and LaValle S. *Resolution Complete Rapidly-Exploring Random Trees*. IEEE Int Conference on Robotics and Automation, pp 267-272, 2002.

60. Cheng P, Shen Z, and LaValle S. *RRT-based Trajectory Design for Autonomous Automobiles and Spacecraft*. Control Sciences, Vol 11, No 3-4, pp 167-194, 2000.

61. Toussaint G. *Motion Planning for Non-linear Underactuated Vehicles using H_∞ techniques*. PhD Thesis. Urbana-Champaign: University of Illinois, 2000.

62. Frazzoli E, Dahleh M, and Feron E. *Hybrid Control Architecture for Aggressive Manoeuvring of Autonomous Helicopters*. Proc of The IEEE Conference on Decision and Control, Vol 3, pp 2471-2476, 1999.

63. Frazzoli E, Dahleh M, and Feron E. *Real-Time Motion Planning for Agile Autonomous Vehicles*. Journal of Guidance, Control, and Dynamics, Vol 25, pp 116-129, Jan/Feb 2002.

64. Cheng P and LaValle S. *Reducing Metric Sensitivity in Randomised Trajectory Design*. IEEE Int Conference on Intelligent Robots and Systems, pp 43-48, 2001.

65. Brooks R. *A Robust Layered Control System for a Mobile Robot*. IEEE Transactions on Robotics and Automation, pp 14-23, 1986.

66. DeMuth G and Springsteen S. *Obstacle Avoidance using Neural Networks*. IEEE Proc Symposium Autonomous Underwater Vehicle Technology (AUV), pp 213-215, 1990.

67. Sayyadi H, Ura T, and Fujii T. *Collision Avoidance Controller for AUV Systems using Stochastic Real Value Reinforcement Learning Method*. Proc SICE, Japan, pp 218-224, 2000.

68. Ishii K, Fujii T, and Ura T. *An On-line Adaptation Method in Neural Network-based Control System for AUV's*. IEEE Journal of Oceanic Engineering, Vol 20, April 1995.

69. Wettergreen D, Gaskett C, and Zelinsky A. *Autonomous Guidance and Control for an Underwater Robotic Vehicle*. Int Conference on Field and Service Robotics, 1999.

70. Zapata R and Lepinay P. *Collision Avoidance and Bottom Following of A Torpedo-like AUV*. IEEE Oceans Conference Record, Vol 2, pp 571-575, 1996.

71. Borenstein J and Koren Y. *The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots*. IEEE Journal on Robotics and Automation, Vol 7, No 3, pp 278-288, 1991.

72. Koren Y and Borenstein J. *Potential Field Methods and Their Inherent limitations for Mobile Robot Navigation*. IEEE Int Conference on Robotics and Automation, pp 1398-1404, April 1991.

73. Williams G, Lagace G, and Woodfin A. *A Collision Avoidance Controller For Autonomous Underwater Vehicles*. IEEE Proc Symposium Autonomous Underwater Vehicle Technology (AUV), Jun 1990.

74. Antonelli G, Chiaverini S, Finotello R, and Morgavi E. *Real-Time Path Planning and Obstacle Avoidance for an Autonomous Underwater Vehicle*. IEEE Int Conference on Robotics and Automation, Vol 1, pp 78-83, 2001.

75. Zadeh L. *Fuzzy Sets*. Information. Control, Vol 8, pp 338-353, 1965.

76. Shinjo N and Graeme J. *Sensor-Based Fuzzy Obstacle Avoidance System for Autonomous Underwater Vehicles*. IEEE Int Symposium on Unmanned, Untethered Submersible Technology, 9th, pp 271-298, 1995.

77. Vasudevan C, Smith S, and Ganesan K. *Collision Avoidance by Dynamic Path Selection and Fuzzy Behaviour Control*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, pp 262-270, 1995.

78. Liu X, Zhang S, Li Y, and Shang Y. *A New Method for AUV's Collision Avoidance*. IEEE Int Symposium on Unmanned, Untethered, Submersible Technology, 11th, pp 587-591, Aug 1999.

79. Ridao P, Batlle J, and Carreras M. O2CA2: *A New Object Oriented Control Architecture for Autonomy. The Reactive Layer*. Control Engineering in Practise Journal, 2001.

80. Khatib O. *Real-Time Obstacle Avoidance for Manipulators and Mobile Robots*. Int Journal of Robotics Research, Vol 5, No 1, pp 90-98, 1986.

APPENDIX A

The potential field method is a very popular tool for motion planning. The equations below are used to generate the simulation results as illustrated in Fig.2. The related mathematical definitions are listed as follows:[80]

The field of artificial forces $\vec{F}(q)$ in C is produced by a differentiable function, $U : C_{free} \to R$ with:

$$\vec{F}(q) = -\vec{\nabla}U(q) \tag{A.1}$$

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{A.2}$$

where U_{att} is the attractive potential associated with the goal configuration q_{goal} and U_{rep} is the repulsive potential associated with the C-obstacle region.

The attraction field can be formulated as below:

$$U_{att}(q) = \frac{1}{2}\xi\rho_{goal}^{\ 2}(q) \tag{A.3}$$

The attraction force can be formulated as below:

$$\vec{F}_{att}(q) = -\xi(q - q_{goal}) \tag{A.4}$$

where ξ is the positive scaling factor, ρ is the Euclidean distance, q is the current configuration and q_{goal} is the goal configuration.

The repulsion field can be formulated as below:

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)^2 & \text{if } \rho(q) \le \rho_0 \\ 0 & \text{if } \rho(q) > \rho_0 \end{cases} \tag{A.5}$$

The repulsion force can be formulated as below:

$$\vec{F}_{rep}(q) = \begin{cases} \eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(q)}\vec{\nabla}\rho(q) & \text{if } \rho(q) \le \rho_0 \\ 0 & \text{if } \rho(q) > \rho_0 \end{cases} \tag{A.6}$$

where η is a positive scaling factor, $\rho(q)$ is the distance to the obstacle and ρ_0 is the distance of influence.

Selected Plenaries, Milestones and Surveys

praha 2005

16th IFAC World Congress

July 3-8, 2005 Prague, Czech Republic

Edited by
P. Horacek
M. Simandl
P. Zitek

QUASI-RANDOM, MANOEUVRE-BASED MOTION PLANNING ALGORITHM FOR AUTONOMOUS UNDERWATER VEHICLES

Chiew Seon Tan,[1] Robert Sutton, John Chudley
Marine and Industrial Dynamics Analysis Group
School of Engineering
The University of Plymouth,
PL4 8AA, UK

Abstract: This paper presents an approach using a hybrid modelling technique known as Manoeuvre Automaton (MA) to capture the key dynamics of a nonlinear autonomous underwater vehicle (AUV) in such a way that high-level tasks such as optimal motion planning can be computationally simplified, while still allowing it to perform complicated manoeuvres when the situation arises. With respect to motion planning in an obstacle filled environment, an incremental stochastic technique derived from the Rapid-exploring Random Tree (RRT) algorithm is applied. This paper proposes a multiple nested node version of RRT and also addresses the case of a time varying final state. Simulation results as presented, using a 3 degree-of-freedom (DOF) nonlinear AUV model in order to prove the viability of the concept. *Copyright*© 2005 *IFAC*

Keywords: Hybrid system, Manoeuvre Automaton, Dynamics quantisation, Optimal motion planning, Motion primitives

1. INTRODUCTION

In the last few years, AUVs are frequently being employed for sea bottom exploration, mine-hunting, scientific data gathering and reconnaissance missions. The requirement for the successful accomplishment of all the above tasks has manifested itself into an urgent demand for an increase in AUV autonomy. In fact, one area that needs particular attention is collision avoidance. Due to its obvious complexity and the limited length, this paper shall concentrate on addressing only the motion planning issues of an AUV.

Lately, there has been a sudden paradigm shift by the scientific communities from AUV deep sea exploration missions to deployment in littoral waters. The littoral zone is important for scientific research since it houses the bulk of ocean based organisms. Likewise, the navies have demonstrated a keen interest in exploiting AUV technology as a potential force multiplier to complement their amphibious power projection plans. Obviously, one can envisage numerous important military missions such as port infiltration and mine-hunting, where it is crucial for an AUV to be able to navigate in an unknown and hostile terrain.

Several inherent characteristics of an AUV particularly its highly nonlinear coupled dynamics, underactuated, non-driftless, non-minimum phase behaviour and its subjection to unpredictable exogenous disturbances makes controller design nontrivial. The last attribute is especially relevant for small, lightweight AUVs such as *Remus* (Prestero, 2001). Typically, a few linearised models are utilised for the AUV controller design, thus artificial operational constraints must be imposed to avoid violation of the linearity assumption. Consequently, this introduces additional restric-

[1] The authors would like to sincerely thank Dr. E.Frazzoli, for his various enlightening explanations and inputs. C. S. Tan is partly supported by IMarEST Stanley Gray Fellowship Award.

tion to the system performance envelope. Alternatively, a more preferable approach is to "quantise" the AUV dynamics, transforming a continuous dynamic model governed by complex nonlinear differential equations into a hybrid model which not only possesses higher levels of abstraction but is also more beneficial computationally. With an added advantage, this approach also conveniently permits the incorporation of complex and aggressive manoeuvres into the AUV. This process is achieved via the Manoeuvre Automaton (MA) representation.

Most path planning techniques introduced to date are based firmly on deterministic methods and graph searches. Unfortunately, due to their sample space discretisation issues, the generated trajectories need extra smoothing and interpolation. Notwithstanding this, the system dynamics are also neglected to avoid state-explosion effect. Collectively, these factors ensue a very conservative system performance. Randomisation methods are becoming popular as they are inherently more robust to the state explosion effect. One version is the Rapid-exploring Random Tree (RRT), which this paper extends and integrates with the MA.

This paper begins with a brief outline of the MA concept and its merits in Section 2. A brief theoretical foundation on how MA can be extended to solve optimal motion planning problem for cases without obstacles is provided. Section 3 discusses the AUV dynamic model and the generation of motion primitives. Implicitly, the latter process is critical as it will dictate the achievable behaviour of the AUV, *per se*. Section 4 is devoted to the integration of the RRT algorithm with the MA for the motion planning problem in the case of an obstacle filled environment. Discussion of the simulation results are contained in Section 5. Finally, Section 6 contains concluding remarks and future work.

2. THE MANOEUVRE AUTOMATON (MA)

The MA, a form of finite state machine, is proposed by Frazzoli *et al.* (1999) as a unified framework for formalising the control of nonlinear systems with symmetries. In essence, the main idea is to generate a complete trajectory via sequential combination of the copies of motion primitives from a library set. These motion primitives are extracted from the vehicle in an open-loop mode.

The approach relies primarily on two different types of motion primitives: trim trajectories and manoeuvres. Trim trajectories (relative equilibria) correspond to steady state behaviour in situation when the velocities in body-axes and inputs are constants. On the other hand, manoeuvres can be seen as finite time motion primitives that interconnect two trim trajectories together.

Similar to a differential or difference equation, a MA transcription, describes a dynamic system, differing only in that it has hybrid elements in both its control inputs (τ, p), and state vector $(\mathbf{x}, q)$. MA evolves in so-called "dense time" by either continuous flows or discrete transitions. Consequently, at each particular moment, the system is constrained to be either in a trim condition q or performing a manoeuvre p. Thus the system behaviour can also be explicitly formulated as below.

- An MA system H starting at state vector $(\mathbf{x}_i, q_i)$ in trim trajectories, evolve according to $f_q(\cdot)$ as determined by the length of the τ_k, which can be infinite. Where $f_q(\cdot)$ is the governing differential equation at the specific discrete state q_k. The hybrid state then evolves as:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_k \dot{\mathbf{x}}_q \quad (1)$$
$$q_{k+1} = q_k \quad (2)$$
$$t_{k+1} = t_k + \tau_k \quad (3)$$

  where $\dot{\mathbf{x}}_q$ is the time rate of change of the vehicle's continuous state variables and k is the "stage" number.
- In the case of performing a manoeuvre p, the vehicle leaves the trim trajectory q_1 for a finite length of time before settling to the trim trajectory q_2. Mathematically, the manoeuvre is initiated by the control action p, which is discrete, and is described by a fixed duration Δt_p and displacement $\Delta \mathbf{x}_p$ in the continuous state space, as illustrated in Fig. 1 for a $SE(2)$ case. In reality, the control history of the continuous state-space system is implicitly encoded in the control action p. As such, when manoeuvering, the hybrid state evolves as:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}_p \quad (4)$$
$$q_{k+1} = q_2 \quad (5)$$
$$t_{k+1} = t_k + \Delta t_p \quad (6)$$

Although, the hybrid control input at instant k can be described by a vector $(\tau, p)_k$, however only one input, either τ or p can be active at any moment.

By having the AUV continuous behaviour encoded as a discrete state q, its configuration can be described by an element of the Lie group G of rigid motions in $\mathbb{R}^2$ or $\mathbb{R}^3$, called $SE(2)$ or $SE(3)$, respectively. In planar situations, where the altitude is constant, $SE(2)$ will be employed. The reason for restricting the formulation to $SE(2)$ has real practical significance, and will be elaborated

in Section 3. The group $SE(2)$ can be expressed using the homogenous coordinates as follows:

$$g = \begin{bmatrix} \cos\psi & -\sin\psi & x \\ \sin\psi & \cos\psi & y \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

The Lie algebra elements $\xi \in SE(2)$ are represented as matrices in $\mathbb{R}^{3\times3}$ and for a special case of $\omega = 0$, one yields Equation 8 to describe the configuration change after τ length of time in trim trajectory. The subscript k is omitted for clarity.

$$e^{\xi\tau} = \begin{bmatrix} 1 & 0 & v_x\tau \\ 0 & 1 & v_y\tau \\ 0 & 0 & 1 \end{bmatrix} \tag{8}$$

One can also describe the configuration change resulted of a manoeuvre p using Equation 7. As a result of applying the MA representation, one can now mathematically describe the system configuration by concatenating these motion primitives as expressed below:

$$g_f = g_0 \left[\prod_{k=1}^{N} e^{(\xi_k,\tau_k)} g_k \right] e^{\xi_{k+1}\tau_{k+1}} \tag{9}$$

Where g_0 and g_f is the initial and final configuration. $e^{(\xi_k,\tau_k)}$ and g_k represent the transformation of applying the k-th trim trajectory and the k-th manoeuvre, respectively.

The MA representation allows one to express easily the optimal motion planning problem. This is true, for certain cost functions that share the symmetry properties of the system, such as minimum time, minimum length and minimum control effort. For the special case of the minimum time cost functional, one can formulate it as Equation 10.

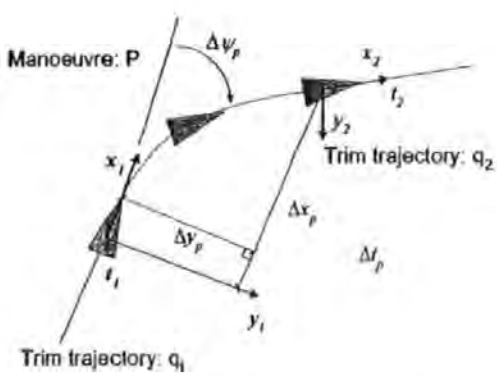


Fig. 1. Displacement of configuration variables and its time duration for manoeuvre p

$$\min_{p_k,\tau_k} \sum_{k=1}^{N} (\Delta t_{p(k)} + \tau_k) \tag{10}$$

such that Equation 9 is satisfied and $\tau \geq 0$.

The above optimisation problem can be solved using a dynamic programming (DP) technique (Frazzoli, 2001; Schouwenaars *et al.*, 2003). For the unique case when $\dot{\psi} = 0$, such that when all the trim trajectories are translations, the cost is linear with respect to the coasting variables τ, hence one can employ the linear programming (LP) method instead (Frazzoli, 2002*a*).

3. AUV DYNAMIC MODEL AND MA IMPLEMENTATION

In this section, the AUV model and techniques for synthesising the motion primitives are presented. This is done, in order to convert the continuous system model into the MA representation. The AUV model was supplied by QinetiQ, based on the *Autosub* vehicle (Millard *et al.*, 1998), which has a torpedo shaped hull. Dimensionally, the vehicle is 7 m long, and approximately 1 m in diameter and has a nominal displacement of 3600 kgs. In this paper, the model is restricted to only latitude dynamics and yaw control limited to the locked bow rudders. The vehicle has a maximum rudder deflection of $\pm 25.2°$ and a rudder rate limit of $9.9°/s$. Including these two components into the model resulted in a nonlinear system. The pitch and roll effects are neglected. The main reason for concentrating only on a latitude model is due to the limitation imposed by a forward looking sonar. To elaborate, most commercial forward looking sonars are only capable of providing a projection of 2D image of the terrain, hence determination of object depth is extremely difficult. Accordingly, to mitigate any risk of collision, the AUV must avoid all the perceived obstacles.

Figure 2 shows a possible MA representation of the *Autosub* dynamics. Both 2 m/s and 5 m/s of cruising speeds are illustrated. Unfortunately, the above model lacks propulsion dynamics, here the forward velocity was held constant by an Proportional-Integral (PI) controller during the experiments. Therefore, this constrains the following simulations to only one speed regime which was selected to be 5 m/s. Normally, for the purpose of generating trim trajectories, a velocity augmentation loop must first be designed into the system. Nonetheless, this process is redundant since the AUV is already assumed to be cruising at a constant velocity. The *Autosub* model is discretised using the zero-hold method and a sampling frequency set to 10 Hz.Referring to Figure 2 again, it shows clearly a library that constitutes manoeuvres of 15°, 30°, 60°, 120°. Since the manoeuvres

are symmetry, the opposite direction manoeuvres are not shown. The manoeuvres should encompass the important performance envelope of the AUV, and their generation can be attained via a human operator or a controller input. The latter method is selected for the following study. A Proportional-Derivative (PD) autopilot is designed so that one can extract the manoeuvres by input step inputs. Obviously, a more advanced controller can also be applied to extract better performance out of the AUV. The input and the state histories are recorded. Table 1 shows a few manoeuvres with their associated execution time duration and displacements.

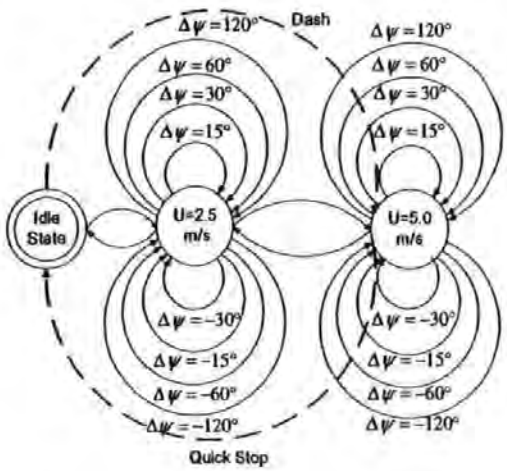


Fig. 2. AUV dynamics in MA representation

Table 1. Manoeuvre Library, $q = 5\ m/s$

P(index)	$\Delta T(s)$	$\Delta x(m)$	$\Delta y(m)$	$\Delta\psi(o)$
P_{15}	5.5	27.2	3.3	15
p_{30}	8.2	38.5	12.2	30
p_{60}	6.6	26.8	15.8	60
p_{120}	23	-8.4	93.4	120

4. QUASI RANDOM RAPID-EXPLORING RANDOM TREE

The approximate cell-decomposition methods such as A^*, dynamic programming and breath-first search are highly susceptible to the curse of dimensionality. Therefore, it is reasonable for one to concentrate on randomised algorithms. These algorithms do not have the completeness[2] and optimality properties of the previous algorithms. However, their robustness to the "curse of dimensionality" tends to make them preferable in practical and real-time applications. One version of this algorithm is the RRT (Lavalle, 1998). It is a form of incremental stochastic search technique that has been devised to search efficiently nonconvex high-dimensional state space.

4.1 Quasi Random Generator

Here, the quasi random (sub-random) generator based on the Halton sequence (Halton, 1960) is utilised instead of a pseudo random generator. Theoretically, the former generator possess certain desirable properties such as low discrepancy and improved uniformity over the sampling space. The generation of an element of a one-dimensional Halton sequence within the interval [0, 1] is calculated using Equation 11 and Equation 12. Different prime numbers starting from the smallest are used for the multi-dimensional sampling case.

$$x_i = \sum_{k=0}^{\infty} n_{k,i} p^{-k-1} \tag{11}$$

with $i > 0, p = 2$ and $n_{k,i}$ determined by the following equation:

$$i = \sum_{k=0}^{\infty} n_{k,i} p^k; \quad 0 \leq n_{k,i} \leq p; \ n_{k,i} \in \mathbb{N} \tag{12}$$

4.2 Motion Planning Algorithm

Frazzoli (2002*b*) advocates enhancing the RRT algorithm by fusing it with the MA to solve motion planning problem with obstacles. The algorithm assumes that one has an embedded planner, that can plan an optimal trajectory in an obstacle free environment between two arbitrary states (Equation 9). The approach in this paper is that of multiple nested nodes. Since every state in a trim trajectory can be considered as a starting point of a manoeuvre. This algorithm generates child nodes at every connection point between a trim trajectory and manoeuvre. This improves the RRT branching capability, thus increasing the probability of finding a solution. A brief explanation of the the algorithm with reference to figure 3 is outlined below:

(1) Generate a subgoal (R1) using the quasi-random generator and attempt to connect to it using the embedded planner based on a minimum time criterion.
(2) If there is no collision, then generate an edge with new vertices at all interconnecting points of trim trajectories and manoeuvres. For all the new vertices, attempt to connect directly to the goal (greedy algorithm).
(3) If failed, generate another random subgoal (R2). Sort the shortest time trajectories to R2 from all vertices in an ascending order and attempt to connect to it. Apply this to only the first few near-optimal trajectories to avoid vertices saturation.

[2] A property where the algorithm will return a solution if such a solution exists

(4) The whole process is repeated until a feasible trajectory to the final state is found, maximum vertices size or time limit is reached. Figure 3 shows that vertex (nc1) has connected successfully to the final state.

4.3 Error Mitigation

Few researchers have expressed their concern regarding the prescribed error generated by RRT algorithm. Due to the discretised nature of the inputs in the original RRT algorithm, when the input history is applied, there will exist some errors in the final state. Hence, Kim and Ostrowski (2003) attempt to circumvent the problem by introducing a subconnection process. Similarly, Cervern *et al.* (2004) introduces error mitigation scheme to reduce the error caused by the concatenation effects. The former approach relies on "integrating" the dynamic model using the acquired input history to ascertain the final state. One disadvantage, in this approach is the requirement of an accurate dynamic model of the system. This might not be true in practice, due to model complexity, or nonexistence of a mathematical model. In fact, this error can be considered as a form of disturbance, and a robust controller can be designed to track the nominal trajectory instead. The design of the tracking controller is non-trivial due to the multi-input-multi output (MIMO) and underactuated behaviour of the AUV, as such it will be addressed in the near future.

4.4 Time-varying final condition

The case of a time-varying final condition is particularly interesting. This problem is commonly met when an AUV is conducting an interception mission such as docking with a moving mother submarine (Tan *et al.*, 2003). The problem of addressing the time-varying final condition using RRT was first pursued by Cervern *et al.* (2004). His approach is based on embedding time variable into the system state vector. Evidently, the immediate effect is the increase in the state vector dimension. A simpler solution proposed here is to adopt an iterative subroutine commonly known as the "false-position" method. Fundamentally, assuming that the target is moving at a constant velocity, the concept is to use a predict-correct process to converge within a tolerance of the final state. Nonetheless, there is no guarantee of convergence, thus an upper bound to the iteration count is needed to terminate the loop as a contingency.

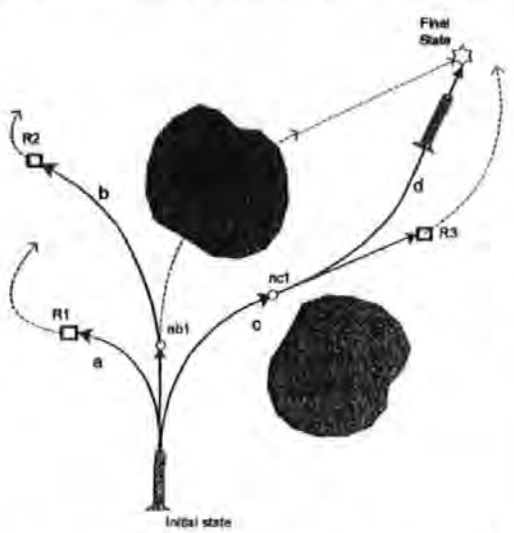


Fig. 3. Simplified illustration of the RRT Algorithm operation

5. SIMULATION RESULTS AND DISCUSSIONS

The algorithm is implemented in MATLAB with the GNU Linear Programming Kit ver. 4.4 (GLPK) [3] in an 2.1 GHz Pentium IV machine, with 512 MB of RAM and running Windows XP. The environment, based on the North-East-Down coordinate, is set to 300 $\times$ 300m in dimension. The simulations assume an ideal case where *a priori* information of the environment is provided and there is no external disturbance from the environment. The simulations are run with 200 maximum nodes, 300 maximum iterations, and a 2 seconds time constraint, terminating when either criterion is reached or if a solution is found.

Pertaining to simulation 1, the AUV initial state is set to [1 1 0.1], while the final state is [170 145 κ] where κ denotes an unconstrained variable. Here the individual variables are [x y ψ], displacements in meters and heading in radians, the goal tolerance is defined as a 7m radius. Figure 4 illustrates one of the trajectories found by the algorithm. The dotted lines are candidate trajectories where the continuous line is the feasible trajectory. The triangles symbolise the AUV, enlarged twice for reason of clarity. The numeric values denote the vertices. For simulation 2 (Figure 5), the final state is set to [150 70 2.2] and moving at 2 m/s, simulating a moving submarine. Again, a trajectory is found as depicted in Figure 5.

Due to the probabilistic nature of the algorithm, a sample of 100 simulations are run to compile the statistics (Table 2). From the median statistic, it is observed that the majority of the solutions are less than a fraction of a second for both simulations. The failure rate of simulation 2 is higher due to the time varying state issue. Moreover, in certain

[3] Obtained from http://www.gnu.org /software/ glpk/ glpk.html

cases, the AUV will intercept the target in a head on position instead of a tail-chase fashion. This is prevented by wrapping the heading angle and constraining the final state heading.

6. CONCLUDING REMARKS

The primary objective of this paper is to verify the feasibility of employing the MA representation and the RRT algorithm to an AUV to solve the motion planning problem. The simulation results obtained are very encouraging, although additional detail studies are warranted. Its very short computational time makes it an ideal algorithm for real-time applications. Additionally, a simpler algorithm for solving time-varying final state has been proposed. As aforementioned, the algorithm is intrinsically a feedforward controller, therefore a robust low-level feedback controller will be designed in the near future to track the prescribed trajectory.

Table 2. Statistics from 100 samples run

Statistics	Sim. 1	Sim. 2
Average time taken,s	0.41	1.22
Median,s	0.27	0.56
Standard deviation,s	0.25	0.41
Success rate	88/100	68/100

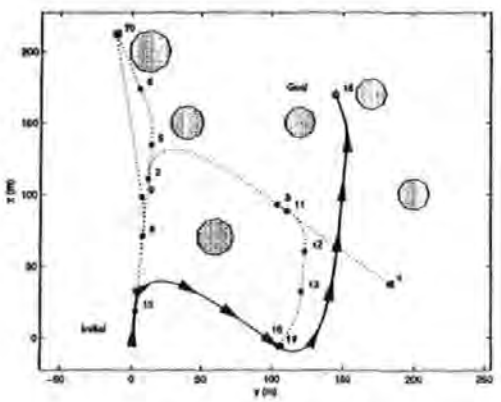


Fig. 4. Simulation 1. Environment with static obstacles

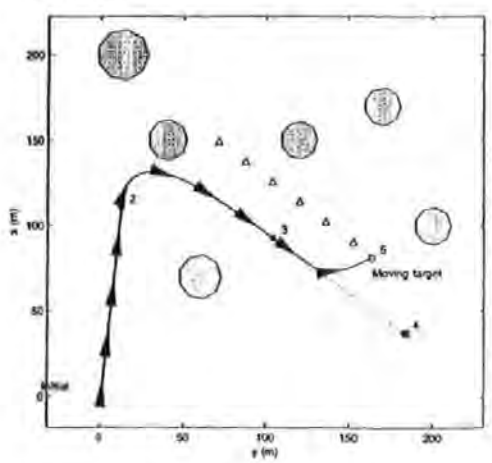


Fig. 5. Simulation 2. Environment with time-varying final state

REFERENCES

Cervern, W. T., F. Bullo and V. L. Coverstone (2004). Vehicle Motion Planning with Time-Varying Constraints. *Journal of Guidance, Control, and Dynamics* **27**, 506–509.

Frazzoli, E. (2001). Robust Hybrid Control for Autonomous Vehicle Motion Planning. *P.h.D Thesis, Massachusetts Institute of Technology, Cambridge, MA.*

Frazzoli, E. (2002*a*). Manoeuvre-Based Motion Planning And Coodination for Single and Multiple UAV's. *Proc. of AIAA/IEEE Digital Avionics Systems Conference* **2**, 8D31–8D312.

Frazzoli, E. (2002*b*). Quasi-Random Algorithms for Real-Time Spacecraft Motion Planning and Coordination. *Acta Astronautica* **53**, 485–495.

Frazzoli, E., Munther A. Dahleh and Eric Feron (1999). Hybrid Control Architecture for Aggressive Maneuvering of Autonomous Helicopters. *Pro. of The IEEE Conference on Decision and Control* **3**, 2471–2476.

Halton, J. H. (1960). On the Efficiency of Certain Quasi-random Sequences of Points in Evaluating Multi-dimensional Integrals. *Numerical Mathematics* **2**, 84–90.

Kim, J. and J. P. Ostrowski (2003). Motion Planning of Aerial Robot using Rapidly-exploring Random Trees with Dynamic Constraints. *IEEE Int. Conference on Robotics and Automation.*

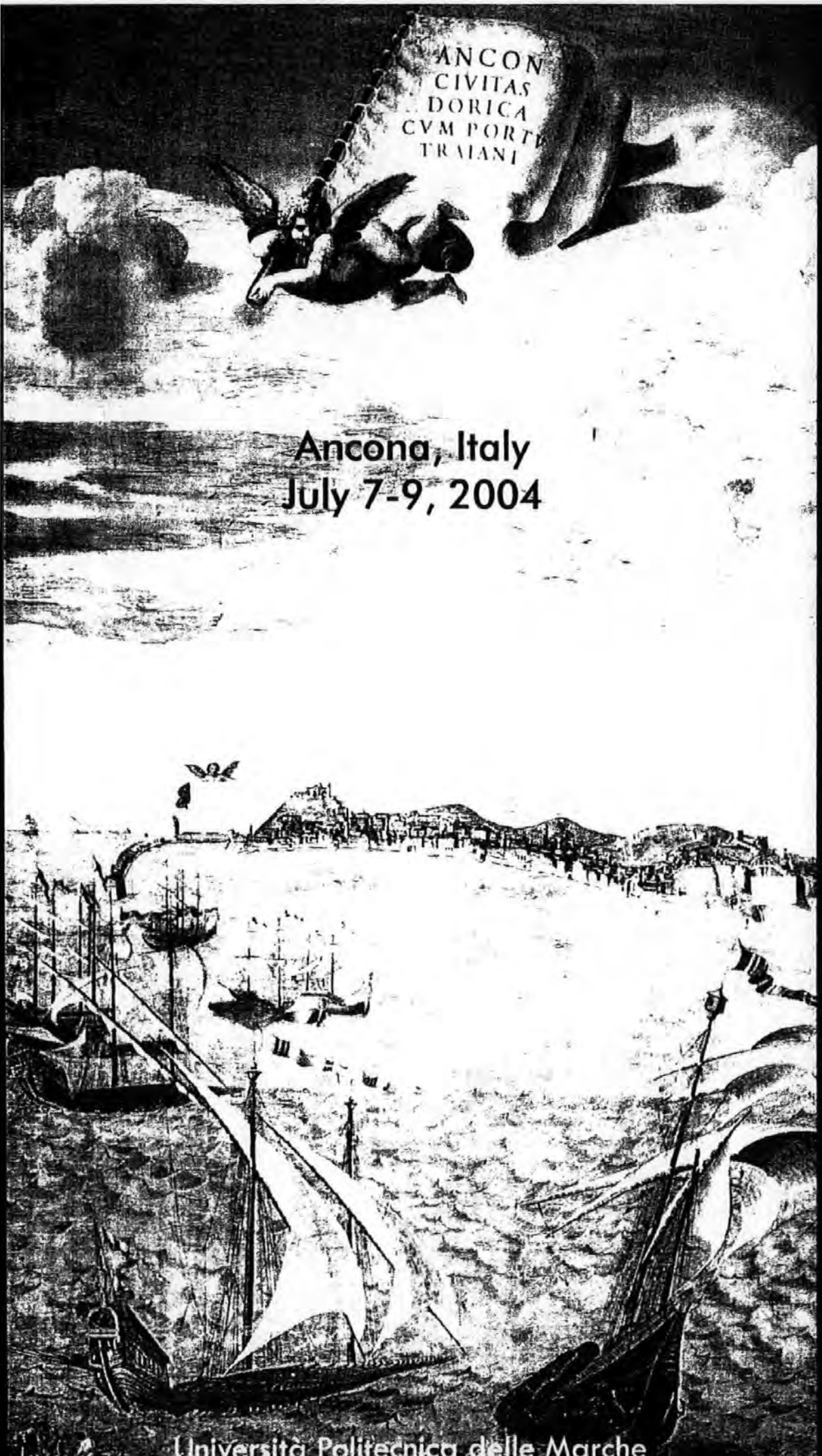
Lavalle, S. M. (1998). Rapidly-exploring Random Trees: A New Tool for Path Planning. *TR 98-11 Computer Science Dept., Iowa State University.*

Millard, N.W., G. Griffiths, Finegan, S. D. McPhail, D. T. Meldrum, M. Pebody, J.R. Perrett, P. Stevenson and A.T. Webb (1998). Versatile Autonomous Submersibles-The Realising and Testing of A Practical Vehicle. *Underwater Technology* **23**(1), 7–17.

Prestero, T. (2001). *Verification of a Six-Degree of Freedom Simulation Model for the REMUS Autonomous Underwater Vehicle, Master Thesis.* MIT. Massachusetts, USA.

Schouwenaars, T., B. Mettler, E. Feron and P. Jonathan (2003). Robust Motion Planning Using a Manoeuvre Automaton with Built-In Uncertainties. *Proc. of the American Control Conference* **3**, 2211–2216.

Tan, C. S., R. Sutton, J. Chudley and S. Ahmad (2003). Autonomous Underwater Vehicle Retrieval Manoeuvre Using Artificial intelligence Strategy. In: *Proc. of GCUV' 2003 Conference.* IFAC. Newport, UK. pp. 143–148.



Ancona, Italy
July 7-9, 2004

IFAC Conference on

CONTROL APPLICATIONS IN MARINE SYSTEMS

Università Politecnica delle Marche

AN INCREMENTAL STOCHASTIC MOTION PLANNING TECHNIQUE FOR AUTONOMOUS UNDERWATER VEHICLES

Chiew Seon Tan, Robert Sutton, John Chudley

Marine and Industrial Dynamics Analysis Group
School of Engineering
The University of Plymouth,
PL4 8AA, UK

Abstract: This paper presents a variant of an incremental stochastic motion planning technique for autonomous underwater vehicles (AUVs) that considers both the algebraic constraints, caused by the environment obstacles, and differential constraints, induced by the vehicle dynamics. The term kinodynamic planning is used to describe this type of motion planning. The majority of AUV path planners tend to adopt an approach where the vehicle differential constraints are neglected with the aim to simplify the path planning process. However, these techniques frequently resulted in paths that are only executable by holonomic vehicles, whereas underactuated vehicles like AUVs are unable to track the prescribed paths exactly. To circumvent this problem, an incremental stochastic planning technique based on a Rapid-exploring Random Tree (RRT) algorithm is used to take into account both types of constraints simultaneously. This paper proposes embedding the RRT algorithm with a reconnection technique to enhance the quality of the generated trajectory. Simulation results as presented below, using a 3 degree-of-freedom AUV model, show the viability of the concept.

Keywords: Kinodynamic planning, motion planning, Rapid-exploring Random Tree, stochastic, reconnection.

1. INTRODUCTION

Autonomous underwater vehicles (AUVs) are not recent inventions, in fact the technology has been around since the past three decades. Nowdays, AUVs are frequently employed for sea bottom exploration, mine-hunting, seabed mapping and scientific data gathering. It is obvious that for AUVs to accomplish successfully such diverse missions, a robust and effective motion planning strategy should be forthcoming.

A majority of the proposed path planning techniques (Fogel and Fogel, 1990; Fox *et al.*, 2000) do not take into account the AUV dynamics. These paths are normally computed as the interconnection of polynomials or splines. Since these paths are independently computed without AUV dynamics, it cannot be guaranteed that they are executable in practice. Typically, very conservative constraints are imposed on the derivatives of the flight path in order to avoid violating the low-level feedback controller operating regime. To further simplify the process, the AUV body geometry is frequently neglected by shrinking it into a point via the application of the configuration space concept. This assumption is highly valid if the vehicle considered is operating in a sparse environment. Most deep sea terrain can be categorised as such. Lately, there has been a sudden paradigm shift by the scientific and naval communities from AUV deep sea exploration missions to deployment in

littoral waters. The littoral zone is a subdivision of the benthic province that lies between the high and low tide marks and can be considered as an extension of the shoreline to 600 feet(183 meters) out into the water. The littoral zone is important for scientific research since it houses the bulk of ocean based organisms. Likewise, the navies have demonstrated a keen interest in exploiting the AUVs technology to complement their amphibious power projection plans. The littoral zone is an intricate area to navigate by default, with unpredictable disturbances such as internal waves, coastal currents, changing beach profile, reefs and artificial objects. Therefore, it is crucial for an AUV to exploit its dynamics to actually navigate the unknown terrain. One class of motion planning algorithm called Rapid-exploring Random Tree (RRT) is particularly well suited for this type of application.

The paper begins with a brief outline of the AUV dynamic model in Section 2. Section 3 attempts to explicate the advantages and the internal mechanisms of the RRT whilst Section 4 discusses several factors that can seriously affect the RRT performance. Section 5 introduces a tree reconnection algorithm to improve the quality of the generated trajectory whereas Section 6 elaborates upon the simulation results for both static and dynamic obstacles. The idea is also extended to a crippled AUV case in order to justify its suitability as a subcomponent to be integrated into a fault tolerant system. Lastly, Section 7 gives the concluding remarks and future extension of the study.

2. AUV DYNAMIC MODEL

A 3-DOF REMUS AUV dynamic model by Prestero (2001) is employed for the simulation study. In matrix form, the dynamics of the vehicle can be defined as:

$$\begin{bmatrix} m - Y_{\dot{v}_r} & 0 & 0 \\ 0 & I_{zz} - N_{\dot{r}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{v_r} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_{v_r} & Y_r - mU_o & 0 \\ N_{v_r} & N_r & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_r \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} Y_\delta \\ N_\delta \\ 0 \end{bmatrix} \delta_r(t) \tag{1}$$

Terminology

- $Y_{\dot{v}_r}$ coefficient for added mass in sway
- Y_{v_r} coefficient of sway force induced by side slip
- Y_r coefficient for sway force induced by yaw
- $N_{\dot{v}_r}$ coefficient for added mass moment of inertia in sway
- $N_{\dot{r}}$ coefficient for added mass moment of inertia in yaw
- N_{v_r} coefficient of sway moment from side slip
- N_r coefficient of sway moment from yaw
- Y_δ rudder input coefficient for sway
- N_δ rudder input coefficient for yaw
- I_{zz} moment of inertia at z axis
- v_r sway velocity
- r yaw rate
- δ_r rudder deflection
- ψ heading angle w.r.t inertial frame, $[0, 2\pi)$

This model is linearised at a constant surge velocity, u_r of $1.5m/s$ (cruising speed) to avoid violating the model fidelity (Equation 1). A pole placement feedback controller is designed for the AUV to improve its dynamic response. The pitch and roll effects are neglected. Clearly, for this simple model an orthogonal transformation matrix can be used to convert the body reference velocities to the velocities in the inertial frame (Equation 2). Here, c_ψ and s_ψ denote $\cos(\psi)$ and $\sin(\psi)$ respectively. $[p\ q]^T$ are the surge and sway velocity while $[X\ Y]^T$ are the vehicle position (configuration) in $\mathbf{R}^2$. There is no singularity problem for this trivial case but this effect will need to be considered if the Euler angle formulation is used in $SE(3)$. By combining both Equation 1 and Equation 2, one obtains a dynamic equation of state vector, $\mathbf{x} = [u_r\ v_r\ r\ X\ Y\ \psi]^T$.

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \end{bmatrix} = \begin{bmatrix} c_\psi & -s_\psi \\ s_\psi & c_\psi \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} \tag{2}$$

The vehicle has a maximum rudder deflection of $\pm 13.6°$ and a rudder rate limit of $18°/s$. Embedding these two components into the model resulted in a nonlinear system. The nominal dimensions of the vehicle are $1.4m$ in length and $0.3m$ in diameter. This information is required by the collision detection algorithm.

3. RAPID-EXPLORING RANDOM TREE

It has been known that, the approximate cell-decomposition methods such as A^*, dynamic programming and breath-first search are highly susceptible to the curse of dimensionality. Therefore, it is reasonable for one to concentrate on randomised algorithms (Branicky *et al.*, 2002). These algorithms do not have the completeness[1] and optimality properties of the previous algorithms. However, their robustness to curse of dimensionality tends to make them preferable in practical and real-time applications.

One version of this algorithm is the RRT (Lavalle, 1998). It is a form of incremental stochastic search technique that is devised to search efficiently non-convex high-dimensional state space, $\mathbf{X} \subset \mathbf{R}^n$. In essence, the RRT algorithm attempts to build

[1] A property where the algorithm will return a solution if such a solution exists

a graph structure, or to be precise, a tree that describes the free state space, $\mathbf{X}_{free}$ of the system. Each node is implanted with the system state, $\mathbf{x}$, where $\mathbf{x} \in \mathbf{X}$. The tree is grown incrementally by picking the closest (Euclidean metric, ρ) node, $\mathbf{x}_{near}$ on the tree to the random node, $\mathbf{x}_{rand}$. Then the best constant input, $\mathbf{u}$ from a finite predetermined set $\mathbf{U}$ is chosen by propagating each input through the system differential equation, $\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)$ for a predetermined time increment, Δt. If no collisions are found, a new child node, $\mathbf{x}_{new}$ is added to the tree and the whole process is then repeated. Note that the time increment, Δt is not the same as the system equation time, δt.

4. RRT PERFORMANCE ENHANCEMENT

4.1 Bias

RRT performance can be improved significantly by the introduction of certain biasing techniques. One such technique is to employ a Gaussian distribution function such that the expected value is located at the goal state. Likewise, one can use a function to return either the goal state or a random state depending on a preset bias coefficient as implemented in this paper. Low discrepancy sequence (quasi-random) such as the Halton sequence has been argued to be more efficient. Despite the fact that its merit has been proven for the Probabilistic Road Map (PRM) method but its effect on RRT is still nonconclusive.

4.2 Computational Bottlenecks

Perusing through the algorithm sequence, one will notice that the two major bottlenecks of the RRT algorithm are the nearest neighbour subroutine and collision detection subroutine. Of the two, the former, particularly the naive version which requires all the nodes to be checked, is the most computationally intensive. As such, it is prudent to substitute it with approximate nearest neighbour (ANN) algorithm by Arya *et al.* (1998) or KD-Tree which are more efficient. For the collision detection routine, this paper uses only non-convex polygons such as rectangle and circle to describe the obstacles thus avoiding the computational issues. High quality collision detection libraries will be needed if nonconvex obstacle models in three dimension are used.

4.3 Metric Sensitivity

A proper metric is essential for the successful operation of RRT. Suitable metric varies from problem to problem. Nonetheless, the metric ρ in the form of cost function or performance index, defined as,

$$\rho^* = min\left(\phi[\mathbf{x}(t_f), t_f] + \int_{t_0}^{t_f} \xi[\mathbf{x}(t), \mathbf{u}(t), t]dt\right) \tag{3}$$

while satisfying the differential constraint,

$$\mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), t] - \dot{\mathbf{x}}(t) = 0$$

where ρ is a function of $(\mathbf{x}_{near}, \mathbf{x}_{rand}, \mathbf{u})$, has been found to be a very suitable for RRT. Equation 3 assumed an obstacle free environment. It has been discovered that the RRT performance tends to degrade as ρ and ρ^* diverge. Typically, numerical method is used to solve the above optimal control problem. The solution can be time consuming since it entails solving a two-point boundary-value problem. In holonomic cases, the differential constraints disappeared and a simple weighted Euclidean metric can be used. On the other hand in nonholonomic cases, a weighted Euclidean metric is also frequently employed however tuning of the parameters (weights) can be nontrivial. Depending on the structure of the vehicle dynamics, incorrect weighting frequently introduces bias into RRT, diverging the search from the goal. Consequently, Cheng and Lavalle (2001) have devised several methods to render RRT less sensitive to metric effects. Two of their proposed methods are also incorporated into the following simulations. The first method is to record the used (expanded) inputs, thus avoiding any states duplications while the second method is to extract environment information concerning obstacles by recording the state collisions frequency. This information is kept in the form of constraint violation frequency (CVF). The objective here is to avoid expanding the state in the region where collision is bound to happen, hence biasing the search to the free space. Since RRT is a form of randomised algorithm, the solution obtained can be far from optimal. As such, this paper proposes a process called reconnection to mitigate this effect. A detail exposition of the process is given in Section 5.

4.4 Multiple Trees, Tree Pruning and Subconnection

Other researchers advocates using multiple trees and tree pruning techniques to improve the RRT performance (Li and Shie, 2003). Indeed, multiple trees RRT version does provide fast solution but its effectiveness is limited to problems with algebraic constraints. This is caused by the difficulty of connecting the tree without obvious gap. Another interesting characteristic of the single tree RRT is its tendency to grow a few major branches at the initial states thus making connection with the terminal states very problematic. To circumvent this problem, one method is to introduce another start tree, with different time

increment and metric when it is at the proximity of the goal states, thus improving the probability of connection (Kim and Ostrowski, 2003).

4.5 Hybrid Planner

As mentioned in Section 4.3, the RRT tends to degrade as the ρ and ρ^* diverge. One promising technique initiated by Frazzoli *et al.* (2002) is to combine an optimal planner with the RRT. The optimal planner which exploits the precomputed trajectory primitives is used to plan an obstacle free path, while the RRT attempts to reroute the path if there are obstacles. Other researches prefer to merge RRT with collocation and nonlinear programming (Karatas and Bullo, 2001). The trajectory obtained shows substantial improvement compared to the individual methods.

5. RECONNECTION

The objective of the kinodynamic planning problem is to find a trajectory $\pi : [t_0, t_f] \rightarrow \mathbf{X}_{free}$ from an initial state $\mathbf{x}_{init}$ to a goal state $\mathbf{x}_{goal}$ within the tolerance τ. However, it can also be formulated in such a way as the problem to find an input function $u : [t_0, t_f] \rightarrow \mathbf{U}$ that results in a collision free trajectory connecting both $\mathbf{x}_{init}$ and $\mathbf{x}_{goal}$. In most cases, it is also appropriate to select a path that optimises certain cost function, such as the time to reach $\mathbf{x}_{goal}$ or the control effort which corresponds to the energy consumption of the system. However, due to its randomised nature, the generated path will tend to be suboptimal.

Hence, this paper proposes a process termed as reconnection where the algorithm is initially executed to obtain a feasible trajectory which is then trimmed at a certain point and reexecuted again. This method exploits two inherent properties of RRT: (1) Its propensity to grow a few major branches from the initial point where these major branches are potential suboptimal trajectories. (2) Reconnecting the RRT entails recycling some of the residual branches thus achieving certain computational advantages compared to initiating a new tree. Certainly, two components need to be addressed: (1) The location of the trimming point and (2) the number of reruns required.

Once the first feasible trajectory is found, it is backtracked to the initial point. The trimming point is selected from 0.4 to 0.7 of the trajectory length. A value of 1.0 is equivalent to starting a new run since the whole core branch is trimmed. A too high value risk destroying important branches and a too low value will not provide substantial improvement as the RRT will attempt to just reconnect the trimmed branch. Multiple runs can be conducted, but, it has been experimentally determined that two to three runs are sufficient to obtain the best suboptimal trajectory.

6. SIMULATION RESULTS AND DISCUSSIONS

For these simulations, the environment is set to $200 \times 200m$ in dimension. Here, we assume an ideal case where *a priori* information of the environment is provided and there is no external disturbance from the environment. The convention here is to take the heading angle to start from the x axis (inertial) and positive when turn counter clockwise. The algorithm is implemented in C on a 2.1 GHz Pentium IV machine, with 512 MB of RAM and running Windows XP. The simulations are run with 2000 maximum nodes and 4000 iterations, terminating when either criterion is reached or if a solution is found. The AUV initial states is set to [1.5 0 0 0 1] (angle in radian), while the goal state is [1.5 κ 150 100 κ] where κ denotes a variable (unconstrained). Goal tolerance is defined as a $5m$ radius. This accuracy can easily be achieved via a modern GPS employing a Wide Area Augmentation System (WAAS). The time increment and dynamic equation time are set to $3s$ and $0.1s$ respectively. Runge-Kutta method is used to propagate the dynamic equation. Instead of assuming a constant input for Δt, the input is linearly interpolated as it propagates through the state equations. This method allows one to employ larger time increment while easily taking into account the input rate constraint. All of the simulations use only the Euclidean distance as the metric.

Figure 1 shows the consequence of applying the reconnection algorithm. Notice the first feasible path (grey colour) has been found and trimmed. The trimming coefficient is set to 0.5 in this case. The intrinsic RRT property of selecting the nearest node resulted in the extension of the longer untrimmed trajectory (dark colour). Both the trajectories are compared to select the shortest amongst the two. Figure 2 shows a histogram plot of 100 simulation samples comparing both the enhanced RRT and a generic RRT performance. It is clear from the histogram that the enhanced algorithm returns the solution in shorter time, particularly for the first feasible trajectory. In Figure 2, it is observed that the enhanced algorithm outperforms the generic RRT in terms of time response while returning the best suboptimal trajectory, shortest distance in this case (Figure 3). The two means are compared using a t-test, it is significance at the 0.01 level alpha whilst the 99% confidence interval for the true difference in means is [36.7 60.5]. However, one must be aware that trajectory selected remains suboptimal, future extension of the algorithm is likely to take this factor into consideration. A detail descriptive statistics comparison of both the algorithms can be found in Table 1. The table indicates that the

generic RRT has a higher failure rate. The failures are caused by the program reaching the predefined maximum nodes number or maximum iterations. Once again, the enhanced RRT performance is superior in comparison to the generic version.

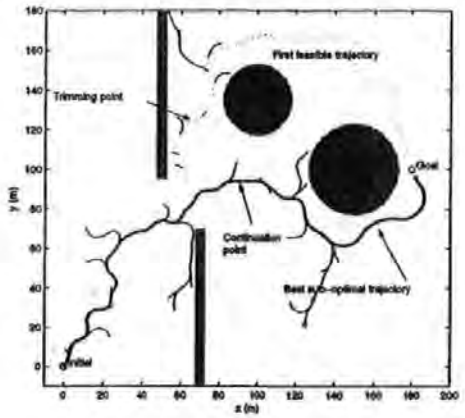


Fig. 1. Motion planning with RRT in environment populated with static obstacles. The reconnection process is also being depicted

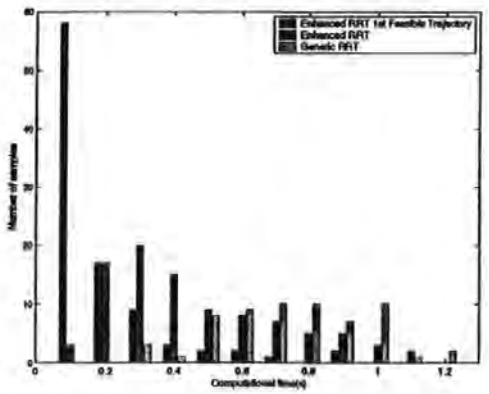


Fig. 2. Histogram plot of computational time

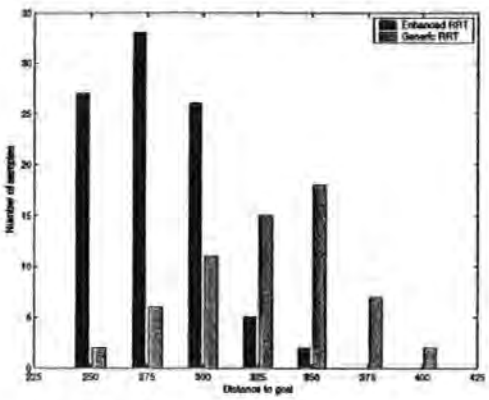


Fig. 3. Histogram plot of total distance

Figure 4 shows a more challenging environment where both the dynamic and static obstacles are presented. The dynamic obstacles are assumed to have constant velocity. The uncertainty of their position as time progresses can be considered by expanding the obstacles size through time. Figure 5 shows a feasible trajectory (dark colour) found

Table 1. Descriptive statistics collected from 100 samples run

Parameters	Enhanced RRT	Generic RRT
Number of nodes used(mean)	870.5	1254.5
Number of nodes used(std)	397.5	515.5
Computational Time(mean, s)	461.8	743.8
Computational Time(std, s)	245.0	211.2
Total distance(mean, m)	278.7	327.3
Total distance(std, m)	23.8	33.0
Number of failures/100 runs	6/100	39/100

by the RRT. To assist in visualising the dynamic effect, it is plotted with respect to time in the z-axis. The trimmed trajectory is not shown to avoid cluttering the view. Figure 6 illustrates a plot of the CVF magnitude. Notice an increase in CVF value of the corresponding node when it collides with an obstacle. This information allows RRT to behave in a more 'intelligent' way by avoiding extension near to the collision area.

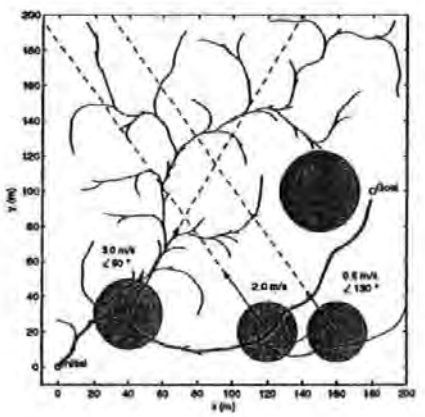


Fig. 4. Position of the static and dynamic obstacles with their corresponding velocities

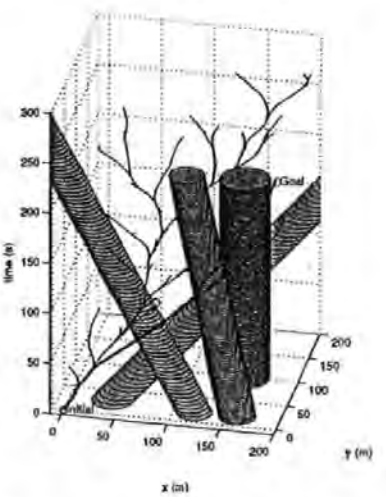


Fig. 5. Plot of configuration w.r.t time for an environment with both static and dynamic obstacles

Recently, there is a sudden increase of interest in developing systems having very high fault tolerant capacity. One of a subcomponent of these systems

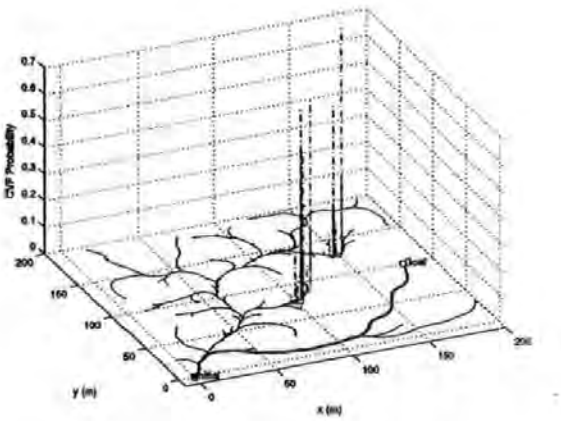


Fig. 6. Plot of the CVF superimposed with the trajectories

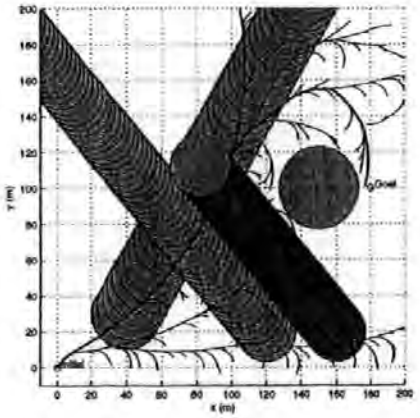


Fig. 7. Plot showing a feasible trajectory (bold) for a crippled AUV

is the fault tolerant control (FTC) system . These systems have very high reliability and is important in terms of operational cost and mission safety. Figure 7 shows a simulation run of a case where the rudder is partially jammed, due to seaweed entanglement or ice built-up. Instead of the normal rudder deflection range of $\pm 13.6°$, it is constrained to operate from $-0.5°$ to $+13.6°$. The effect can be deduced from the shape of the trajectories. The enhanced RRT algorithm found the goal in 3930 iterations and 1922 nodes in approximately $1.2s$. This clearly demonstrates RRT as a promising subunit of a FTC system.

7. CONCLUDING REMARKS

The objective of this paper is to introduce the enhanced RRT algorithm as a motion planner for AUVs. The algorithm is capable of generating feasible trajectories, satisfying both the algebraic and differential constraints. Its very short computational time makes it an ideal algorithm for real-time applications. The reconnection algorithm has been demonstrated to provide shorter trajectories. Nonetheless the trajectory is still suboptimal and this issue will be addressed in the future. Since the algorithm is inherently a feedforward controller, a robust low-level feedback controller is needed to track the prescribed trajectory when subjected to external disturbance. In addressing the case of navigating in an unknown environment, a sensor-based motion planning method will be merged with the enhanced RRT. The algorithm will be extended to a 6-DOF AUV model and it is anticipated that the developed algorithm will be implemented in a research AUV in the near future.

REFERENCES

Arya, S., D. M. Mount, N. S. Netanyahu, R. Silverman and A. Y. Wu (1998). An Optimal Algorithm for Approximate Nearest Neighbour Searching Fixed Dimensions. *Journal of ACM* 45(6), 891–923.

Branicky, M. S., S M. LaValle, K. Olson and L. Yang (2002). Deterministic vs. Probabilistic Roadmaps (summited). *IEEE Transactions on Robotics and Automation* pp. 1–13.

Cheng, P. and S. M. Lavalle (2001). Reducing Metric Sensitivity in Randomised Trajectory Design. *IEEE Int. Conference on Intelligent Robots and Systems* pp. 43–48.

Fogel, D. B. and L. J. Fogel (1990). Optimal Routing of Multiple Autonomous Underwater Vehicles Through Evolutionary Programming. *IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV)* pp. 44–47.

Fox, R., A. Garcia, Jr. and M. L. Nelson (2000). A Generic Path Planning Strategy for Autonomous Vehicles. *The University of Texas - Pan American, Department of Computer Science Technical Report CS-00-25.*

Frazzoli, E., M. A. Dahleh and E. Feron (2002). Real-Time Motion Planning for Agile Autonomous Vehicles. *Journal of Guidance, Control, and Dynamics* 25(1), 116–129.

Karatas, T. and F. Bullo (2001). Randomized Searches and Nonlinear Programming in Trajectory Planning. *Pro. of The IEEE Conference on Decision and Control* 5, 5032–5037.

Kim, J. and J. P. Ostrowski (2003). Motion Planning of Aerial Robot using Rapidly-exploring Random Trees with Dynamic Constraints. *IEEE Int. Conference on Robotics and Automation.*

Lavalle, S. M. (1998). Rapidly-exploring Random Trees: A New Tool for Path Planning. *TR 98-11 Computer Science Dept., Iowa State University.*

Li, T-Y and Y-C Shie (2003). An Incremental Learning Approach to Motion Planning with Roadmap Management. *IEEE Int. Conference on Robotics and Automation.*

Prestero, T. (2001). *Verification of a Six-Degree of Freedom Simulation Model for the REMUS Autonomous Underwater Vehicle, Master Thesis.* MIT. Massachusetts, USA.

9-11 April 2003 Newport, South Wales, UK

INTERNATIONAL FEDERATION OF
AUTOMATIC CONTROL

IFAC

1st IFAC Workshop

Guidance and Control of Underwater Vehicles

GCUV2003

UWCN

AUTONOMOUS UNDERWATER VEHICLE RETRIEVAL MANOEUVRE USING ARTIFICIAL INTELLIGENT STRATEGY

C. S.Tan, R. Sutton, S. Ahmad and J. Chudley
(chiew.tan ,r.sutton, s.ahmad, j.chudley) @plymouth.ac.uk

Marine and Industrial Dynamic Analysis Group
Department of Mechanical and Marine Engineering
The University of Plymouth, PL4 8AA, UK

Abstract: This paper focuses on the application of employing fuzzy logic technique in an autonomous underwater vehicle (AUV) docking manoeuvre involving a moving platform. The core of the paper delves on the implementation of an enhanced fuzzy logic controller for the docking of an AUV in a simulated environment. The implemented fuzzy logic inference system (FIS) functions by mapping the individual fuzzy areas or cells (AUV positions) into a format that contains critical information regarding AUV speed and heading requirements such that the AUV could be recursively driven to the docking station.

Keywords: Artificial intelligence, fuzzy logic, high level control, docking

1. INTRODUCTION

There is increasing interest in the use of autonomous underwater vehicles (AUVs) as force multipliers for submarine supported maritime expeditionary operations. However, aggravating the problem is the limited battery technology of an AUV, which does not provide adequate servicing range. Likewise, the high bandwidth data transmission requirement for most surveillance task makes it compulsory for an AUV to upload its data intermittently. Thus, the concept for an AUV docking with a station/submarine for recharging, downloading data or even servicing purposes is an attractive proposition. Until recently, most research (Cowen *et al.*, 1997; Yoerger *et al.*, 1991) has been concentrated on the docking process with a static docking station or hitching post and not with a moving object. This paper investigates the viability of employing a fuzzy logic technique to assist the AUV docking with a moving target.

An AUV retrieval manoeuvre can be partitioned into two distinct phases, which are interception and docking. In this case, the interception phase employed the popular proportional navigational guidance (PNG) algorithm. However, once the AUV reaches a predefined circumference of acceptance of the target, the docking phase that utilised fuzzy logic will be activated. Very few scientific papers can be found regarding AUV docking with a moving target. Smith, *et al.* (1993) are seen as the only proponents in this area of research. It was reported that they successfully employed a separate linear function for the speed profile controller and a fuzzy logic system to provide recursive 'goal' for the AUV during the docking phase simulation. The advantage of using a fuzzy rule base system is its non-dependence of accurate information about the dynamics of the complete AUV ocean system. The fuzzy system approached the virtual "funnel" by providing new waypoint so that more aggressive AUV manoeuvre can be initiated. The main goal is to dock reliably in various uncertain conditions instead of trying to achieve docking in minimum time.

The remainder of the paper begins with Section 2, which provides a brief outline of the AUV dynamics used in the simulation. Section 3, delves into the speed profile controller. Section 4 discusses the Virtual Target Strategy (VTS), a proposed method for redirecting AUV heading. Section 5 provides a detail discussion regarding the integration of the speed profile controller and VTS into one compact AUV docking controller using a FIS. By integrating both elements, a simple collision avoidance system can be established to reduce the risk of the AUV colliding with the target during docking. Section 6 provides simulations result. Lastly, Section 7 gives concluding remarks regarding the study.

2. AUV DYNAMICS MODEL

This simulation was hosted in the MATLAB / Simulink environment and utilised the Subzero II AUV dynamics. Subzero II is a research AUV from University of Southampton, much of the identification of the AUV dynamics has been produced by Lea (1998). Non-linear heading dynamics was used for simulation, while depth dimension was neglected (assuming constant depth). The non-linear heading dynamics is valid for the range of 0.8 to 1.3m/s AUV cruising speed. The speed of the AUV is adequately controlled by a proportional derivative controller. For this simulation, an adaptive network-based fuzzy inference system (ANFIS) (Jang, 1993) has been developed as the heading controller (Craven, *et al.* 1998). The state space equation of the heading dynamics is given as:

$$\begin{bmatrix} \dot{v} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = u \begin{bmatrix} -1.61 & 0.29 & 0 \\ 6.12 & -6.69 & 0 \\ 0 & u^{-1} & 0 \end{bmatrix} \begin{bmatrix} v \\ r \\ \psi \end{bmatrix} + u^2 \begin{bmatrix} 0.63 \\ -8.34 \\ 0 \end{bmatrix} \delta r \qquad (1)$$

v = Sway u = Speed
r = Yaw rate δr = Rudder deflection
ψ = Yaw Euler angle

3. SPEED PROFILE CONTROLLER

As anticipated, the AUV velocity must decrease steadily as it approaches the target. This can be accomplished by decreasing the velocity proportional to the distance between the AUV and target using linear function (Smith, *et al.* 1993). Instead of varying the velocity linearly, a more sophisticated version is to map the docking zones with various speeds as achieved in the integrated fuzzy docking controller.

4. VIRTUAL TARGET STRATEGY

The VTS is proposed to assist in dictating the AUV heading in order to increase the likelihood of docking success. The VTS utilises a simple concept of offsetting the real target to ensure the AUV homes to 'virtual target' (fig. 1). The offset target causes the AUV to centralise itself quicker relative to the mother submarine. For performance comparison, both the VTS and non-VTS AUV were simulated. Both AUVs demonstrates desired tail chase trajectory. The VTS AUV managed to constrain itself to dock in a tail chase configuration earlier but requires longer time to achieve docking. In the simulation, the offset value was constrained to be in line with the target vector and both linear and Gaussian profiles were tested. The Gaussian profile provides the smoothest response with AUV exhibiting remarkable agility. The guidance algorithm used in the docking phase is the line-of-sight (LOS) and not PNG algorithm. This is to avoid introducing additional parameters thus complicating the process of evaluating VTS performance.

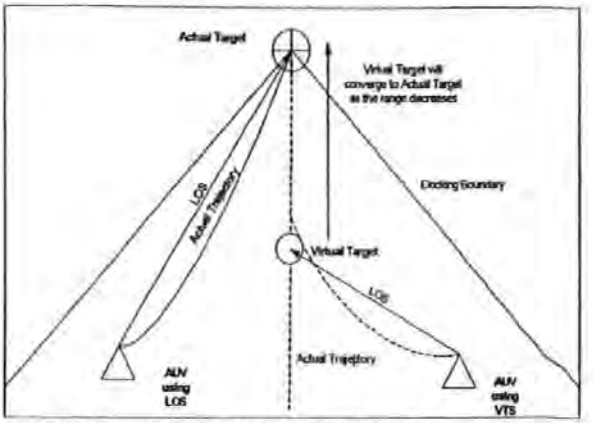


Fig. 1. Virtual Target Strategy (VTS)

5. INTEGRATED FUZZY DOCKING CONTROLLER

5.1 Introduction

The main objective for this part is to unify the functionality of the speed profile controller and VTS into one integrated docking controller through the use of a FIS. The FIS employed is a multi-input multi-output (MIMO) system with two inputs (range and theta) and two outputs (speed coefficient and waypoint offset). To achieve this manoeuvre it must be assumed that the AUV has accurate relative position information for itself and the dock throughout the docking procedure.

5.2 Cell Partitions

The inputs are derived from area partitions, or cells as shown in fig. 2. The area under investigation is the side zone of submarine, assuming that the docking process is constrained to occur in only a tail chase configuration.

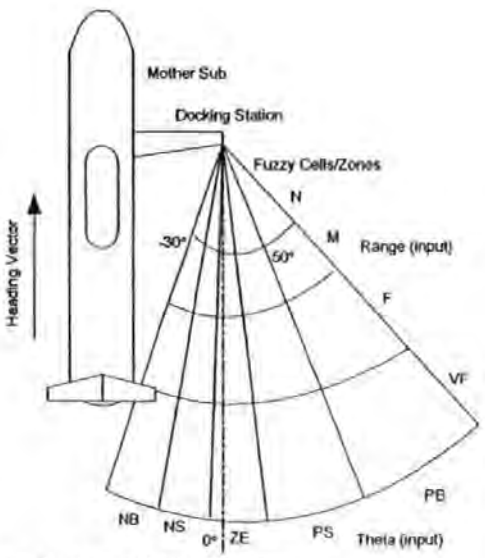


Fig. 2. Fuzzy Cell Partitions

The fuzzy docking controller functions by mapping the area behind the target into different cells. Each individual fuzzy cell incorporates essential heuristic information regarding speed and heading (waypoint offset) requirement of the AUV. Evidently from fig. 2, the partitioned cells are asymmetric and vary in size. Cells that are near to the submarine are classified as dangerous zones. It is crucial to

incorporate 'danger cells' into the docking algorithm in order to reduce the possibility of collision. The embedded FIS is expected to execute collision avoidance manoeuvre by dictating the speed and heading of the AUV. The Mandani's style fuzzy inference system (Mandani and Assillian, 1975) was chosen due to its extraordinary capability to intuitively embed abstract human knowledge into the controller.

5.3. Fuzzy Inputs/Outputs

Figure 3 illustrates the membership functions of the two inputs, range (m) and theta (degree) that comprise of triangular and trapezoidal membership functions. There are four membership functions for range input and five for theta input.

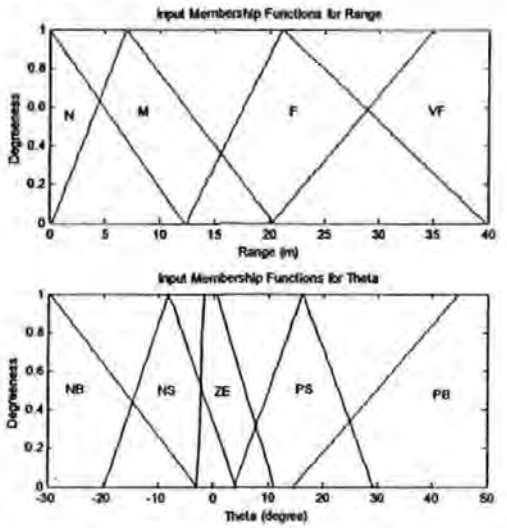


Fig. 3. Inputs Memberships for Fuzzy Docking Algorithm

A reasonable amount of overlapped, 25% or above (Yan, *et al.*, 1994) is required for a fuzzy system to produce smooth output and to avoid any discontinuity. A trapezoidal shape membership function has been claimed to bring robustness to fuzzy logic controller (Yan, *et al.*, 1994). In this case, the trapezoidal shape allows the fuzzy system to have a 'dead band' effect, so that the controller is less sensitive to external noise. Triangular membership function normally creates a 'peak' in the controller output surface plot, causing actuator demand to fluctuate when there exists small perturbations. Figure 4. shows the output membership functions of speed coefficient (for controlling the AUV speed) and waypoint (for redirecting AUV to the virtual target). The membership function, MF3, for waypoint output was added to provide a smoother output.

5.4 Fuzzy Rule Base

The fuzzy rule base has the following form:

IF Theta is *T* **AND** Range is *R* **THEN** Speed Coefficient is *S* **ALSO** Waypoint is *W* (2)

Where *T*, *R*, *S* and *W* are crisp inputs and outputs.

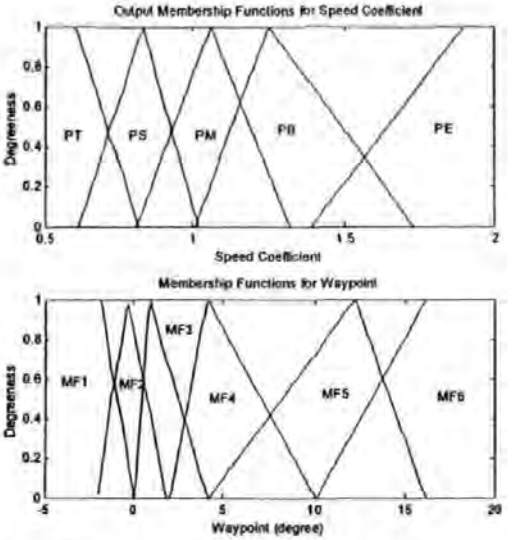


Fig. 4. Outputs Memberships for Fuzzy Docking Algorithm

Table 1 and 2 shows the fuzzy rule base in matrix form. The AND method used was Product. For composition rule, the most popular type was the MAX-MIN but various authors (Andreeva, *et al.*, 1999; Negnevistsky, 2002 and Yan, *et al.*, 1994) have discovered that the MAX-DOT in general provides more appealing result since there are less information loss. This can be attributed to the use of Larsen product operator instead of Min operator. Consequently, both MAX-MIN and MAX-DOT composition rules were tested to observe their difference.

Table 1. Speed Coefficient Output Matrix

Theta\ Range	N	M	F	VF
PB	PS	PM	PB	PE
PS	PS	PB	PE	PE
ZE	PM	PB	PE	PE
NS	PM	PS	PS	PE
NB	PT	PS	PS	PE

Table 2. Waypoint Output Matrix

Theta\ Range	N	M	F	VF
PB	MF3	MF4	MF5	MF6
PS	MF2	MF3	MF4	MF5
ZE	MF1	MF1	MF2	MF3
NS	MF2	MF3	MF4	MF5
NB	MF3	MF4	MF5	MF6

Inputs Abbreviations

PB	=Positive Big	N	=Near
PS	=Positive Small	M	=Medium
ZE	=Zero	F	=Far
NS	=Negative Small	VF	=Very Far
NB	=Negative Small	MF	=Membership Function

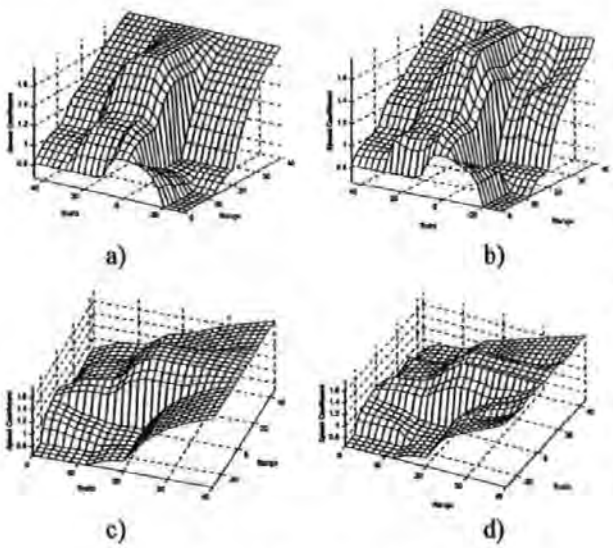


Fig. 5. Fuzzy Surface Plot (Speed Coefficient)

Figure 5 a) and c) are output surface plots for the speed coefficient using MAX-DOT composition rule, while fig. 5 b) and d) used MAX-MIN composition rule. Notice that the MAX–DOT composition rule produces a smoother response surface compares to the MAX-MIN version. This is especially noticeable near the peak of the surface. MAX-MIN surface is clustered with bumps and ridges. The undulating surface causes the controller to have a tendency to produce erratic response in noisy environment. The speed coefficient output is then multiplied by the target speed to obtain the demanded AUV speed at different zones. The asymmetric nature of the output surface plot is due to the integration of the collision avoidance feature into the fuzzy expert system. The abrupt changes in surface gradient at negative theta zones are compulsory to decelerate the AUV and to avoid collision. This is preceded by the assumption that the submarine was located at the left hand side of the AUV (fig. 2).

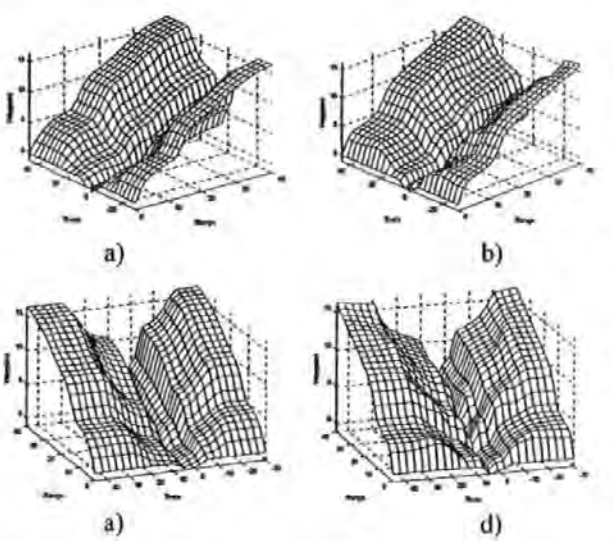


Fig. 6. Fuzzy Surface Plot (Waypoint Offset)

Figure 6 a) and c) are surface plots for the output waypoint using MAX-DOT composition rule, while fig. 6 b) and d) used MAX-MIN composition rule. Again, the MAX–DOT composition rule produces smoother response surface compare to the MAX-MIN version. The waypoint output was used to offset the distance of the actual target thus creating a 'virtual' target for the AUV to home on. The surface was created with the assumption that if the AUV was in the ideal trajectories or desired zone, then the offsetting feature was not required. This explained the reason why the centre surface produces lower waypoint offset value. Conversely, if the AUV is not in the desired zone probably due to underwater disturbances, then the offset waypoint magnitude is increased to guide the AUV to desired zone. For the later simulation run, the MAX-DOT composition rule was selected for the fuzzy docking algorithm due to its better response.

6. SIMULATIONS

The fuzzy docking algorithm was tested relative to a target as illustrated in fig. 7 and fig. 10. The fuzzy docking algorithm was tested from both left and right side of the submarine to evaluate its performance. For analytical tractability, each simulation was compared with the ideal, no disturbance model. The noise simulating a simple underwater disturbance encountered in practical situation was generated using uniform random number, smoothed and constrained to ± 1 m magnitude. The noise was then added in the X-axis direction (fig. 7 and fig. 10). The AUV was considered to dock successfully if it managed to reach 1 m from the point target. The performances of AUV with and without noise disturbance are compared with reference to the following:

I) Engagement trajectories
II) Heading angle requirement
III) Actuator demand
IV) Docking speed and
V) Miss-distance

Again, the emphasis shall be focused on the AUV's ability to dock in uncertain conditions and not the minimum time required.

6.1 Simulation A

The first simulation was run with an initial target positioned at [100,100], heading 90° with speed 0.8m/s, and AUV initial position at [150,0]. Both resulted in a successful docking (fig. 7), although the disturbance model requires longer time to dock. The AUV with noise disturbance changes heading constantly to accommodate for the disturbance effect as shown in the yaw history (fig. 8). Furthermore, the actuator demand shows no rudder saturation for both the cases, clearly demonstrating the superiority of fuzzy controller. The sudden heading change clearly noticeable at time 90s in the yaw history is caused by the engagement of the docking sequence. The sudden offset of the waypoint causes the AUV to react aggressively. This phenomenon is not desirable and will be investigated in future work.

For the noise model, there were abrupt AUV speed changes during docking process as shown in fig. 9. The speed changes are compulsory in order to avoid any collision with the submarine when the AUV has drifted into the danger zone. The miss distance (fig. 9) shows an increase of the range at approximately 180m due to the deceleration of the AUV. Clearly, the low precision potential of the fuzzy logic has been utilised to provide robustness to the overall docking process for this simulation.

6.2 Simulation B

The second simulation was tested with a initial target positioned at [100,100], heading 90° with speed 0.18m/s, and AUV initial position was altered to [50,0]. Again, both show successful docking (fig. 10). However, the docking process takes longer compared to the first simulation, since the AUV collision avoidance feature requires the AUV to be in a 'safety' zone before accelerating to the target. The AUV with disturbance changes heading constantly to accommodate for the swaying as shown in the yaw history (fig. 11). The controller also shows no rudder saturation for both the cases. The sudden heading change at time 90s is caused by the initialisation of the docking algorithm.

Rapid AUV speed changes during docking process as shown in fig. 12 was even more pronounce compare to the first simulation. Docking from the left side of the submarine posed a higher risk of collision due to the submarine structure. The speed change is to decelerate the AUV and allowing it to enter the safety zone, before accelerating to the target. The miss distance (fig. 12) shows an increase of range at approximately 150m and 180m due to the deceleration of the AUV. Evidently, the fuzzy logic capability to integrate expert knowledge, knowledge of docking and collision avoidance in this case has been fully exploited in this case.

7.0 CONCLUSION and REMARKS

The FIS performed an excellent task of unifying both the speed profile controller and VTS into one controller. Although, the data incorporated are still subjective and depends much on human knowledge, at least the fuzzy rules and membership functions provides a methodology for integrating and modifying the knowledge.

Other simulation models were also tested but the results were unsatisfactory. The AUV failed to dock completely when imposed to a directional sea disturbance. This was caused by the proportional-derivative optimised ANFIS heading controller; with no integrator, the model is vulnerable to setpoint and steady state error (Fossen. 1985). Higher magnitude of current disturbance model was also tested but the AUV fuzzy docking algorithm demonstrated oscillatory and unstable behaviour in that situation. The problem was partly caused by flaws in the expert knowledge. One of the major problems is the increase of accuracy required as the AUV approaches the target. Slight sway relative to the target at near range causes the AUV to drift into neighbourhood cells, which induces the oscillatory behaviour. An effective solution to this would be to employ a multistage FIS. Moreover, the Subzero II is a small research torpedo shaped AUV with 0.1 m diameter and length of 1m, this makes it unsuitable for real underwater disturbance simulation.

Overall, the simulation results obtained shows viability and attractiveness in the techniques employed. However, the problems encountered confirm that more detail analysis is needed for the controller to function appropriately in practical situation.

REFERENCES

Andreeva, P., Atanasova T., Zaprianov J. (1999), The model ASIS for process control application, Institute of Control and System Researches, BG. *http://www.enable.evitech.fi/enable99/papers/andreeva/andreeva.html.*

Cowen, S., Briest, S, and Dombrovoski, J. (1997), Underwater docking of autonomous undersea vehicles using optical terminal guidance, *IEEE Oceans '97 Halifax.*

Craven, P.J., Sutton, R. and Burns, R.S. (1998), Control strategies for unmanned underwater vehicles, *Journal of Navigation*, **Vol.51**, No.1, pp.79-105.

Fossen, T.I (1985), *Guidance and Control of Ocean Vehicles*, John Wiley & Sons, New York.

Jang, J.-S.R. (1993), ANFIS: Adaptive network-based fuzzy inference systems, *IEEE Transactions on Systems, Man and Cybernetics*, **Vol. 23**, No. 3, pp. 665-685.

Lea, R.K. (1998), *Control of a tethered underwater flight vehicle*, PhD thesis, University of Southampton.

Mamdani, E.H., and Assillian, S. (1975), An experiment in linguistic synthesis with a fuzzy logic controller, *International Journal of Man-Machine Studies*, **Vol. 7**, 1-13.

Negnevitsky, M. (2002), *Artificial Intelligent, A Guide to Intelligent Systems*, Addison-Wesley, England.

Smith, S.M. Rae, G.J.S, Anderson, D.T. and Shein, A.M. (1993), Fuzzy logic control of an autonomous underwater vehicle, *Control Engineering Practice*, 1994, **Vol. 2**, No. 2, pp. 321-331.

Yan, J., Ryan, M. and Power, J. (1994), *Using Fuzzy Logic*, Prentice Hull International (UK) Limited.

Yoerger, D.E., Bradley, A.M., and Walden, B.B. (1991), A deep ocean AUV for scientific survey- The autonomous Benthic Explorer, *Unmanned Systems*, **Vol.9**, No. 2, pp. 17-23.

Simulation A

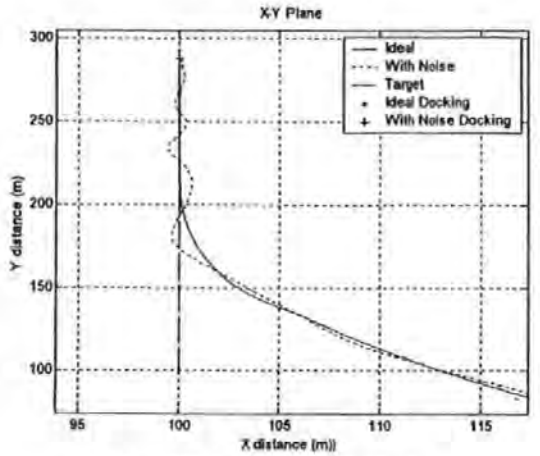


Fig. 7. AUV Docking Trajectories

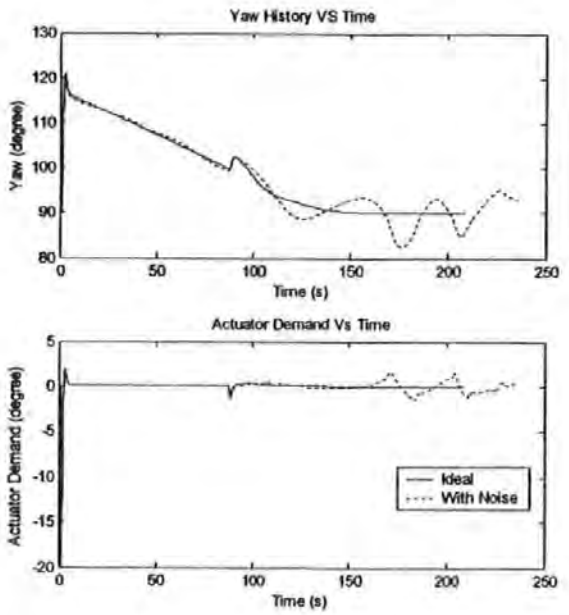


Fig. 8. Yaw History and Actuator Demand

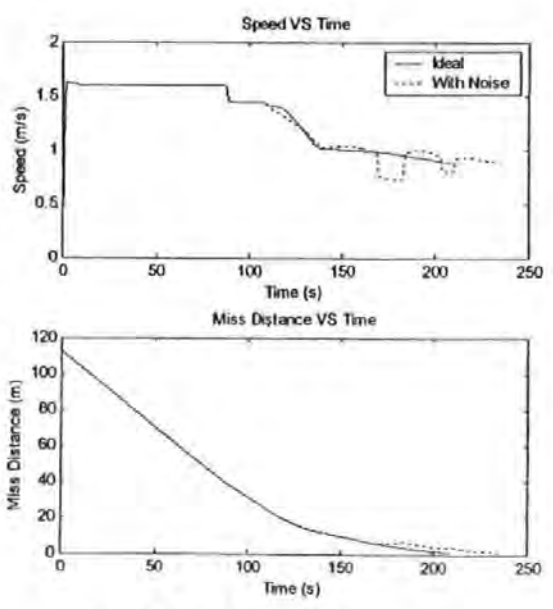


Fig. 9. Speed and Miss Distance

Simulation B

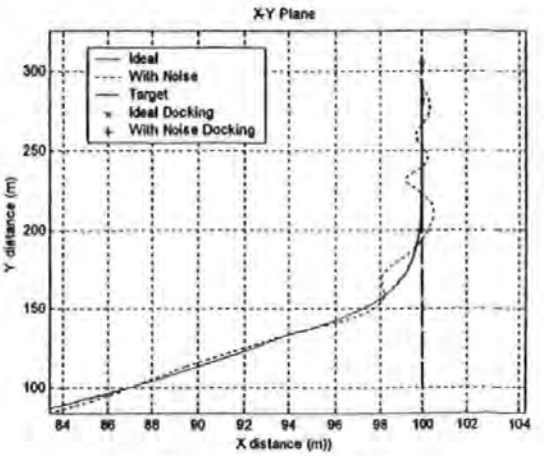


Fig. 10. AUV Docking Trajectories

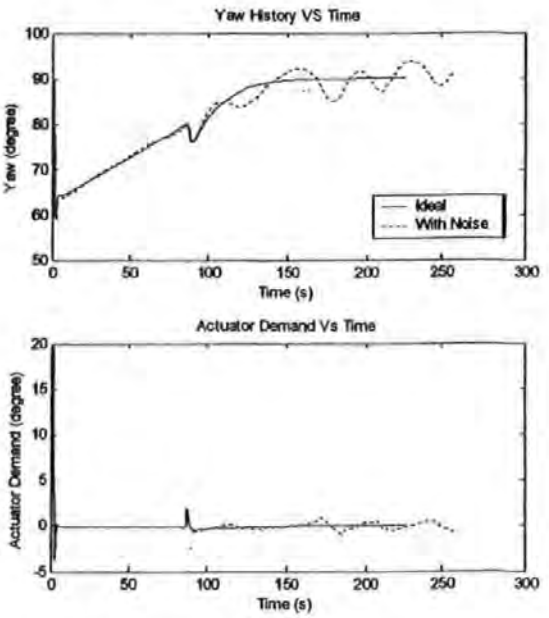


Fig. 11. Yaw History and Actuator Demand

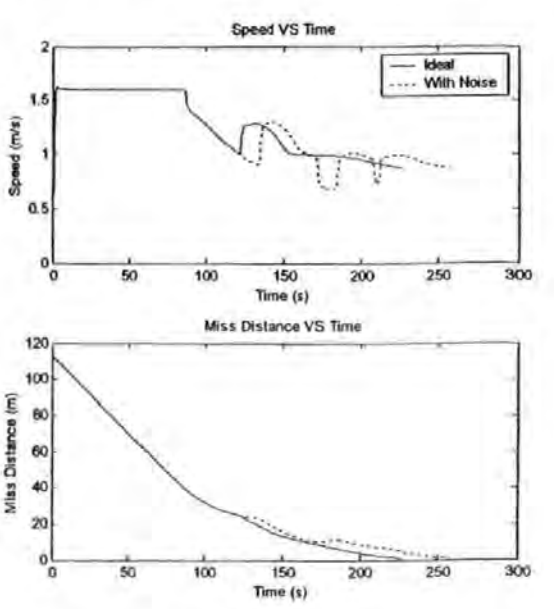


Fig. 12. Speed and Miss Distance

Low Power Intelligent Sonars for AUV Navigation and Collision Avoidance

P. Robinson, C. S. Tan[†], C. Morris, R. Sutton[†]
{peterr, clivem}@jsmarine.co.uk [†] {chiew.tan, r.sutton}@ plymouth.ac.uk

J&S Marine Ltd, UK
[†] The University of Plymouth, UK

Abstract: Traditional approaches to AUV collision avoidance involve the adaptation of COTS electronic scanning sonars that have been designed for applications where low power is not a consideration. This has an adverse impact on the endurance and performance of the AUV. The navigation and obstacle avoidance processing is typically performed by the vehicle control processor, acting on large volumes of data fed at high speed from the sonar. This results in a heavy burden on the AUV control processor. A novel approach to the sonar is described which addresses both of these issues. The sonar used achieves power consumption figures that are an order of magnitude less than typical sonars fitted to AUVs. The sonar also uses its own DSP to perform the data analysis functions, thus reducing the processing burden on the vehicle computer and allowing the use of a low speed serial link between the sonar and the AUV.

INTRODUCTION

Hammerhead (Figure 1) is an Autonomous Underwater Vehicle (AUV) with adaptive tracking and navigation capability. This EPSRC funded project commenced in October 2001. A deep mobile target (DMT) has been converted into a rudimentary AUV. Its small size and modular construction make it an ideal platform to use as a low cost technology demonstrator. In this project, teams from the Universities of Plymouth and Cranfield in the UK are working together to integrate navigation, control and vision systems into a single vehicle for a seabed inspection task. The navigation system will be based on a MSDF algorithm which will produce accurate navigation information continuously in real-time. During an actual surveying task this subsystem will be enhanced by data from a vision system in order that mission parameters can be changed based on changing mission and environmental conditions. Once the navigation data has been suitably processed it will be fed to the guidance and control system for appropriate action.

As an extension to the original work programme the University of Plymouth has undertaken a separate study of the use of sonar for collision avoidance. During the latter part of 2002 a search was made for a suitable sonar for this purpose. At an early stage in this search, studies on update rates ruled out the use of mechanically scanned sonars. This meant that a suitable electronically scanned sonar had to be found. This proved problematical for several reasons: size, power consumption and price. The solution was found in the adaptation of an existing low power sonar offered by one of the Industrial Steering Group members, J&S Marine Ltd.

AN ELECTRONIC SCANNING SONAR DESIGNED FOR AUV USE

Some of the desired qualities of AUV obstacle avoidance sonar are listed below:

- Low power consumption
- High resolution with adequate detection range (depending on the AUV manoeuvrability)
- High Scanning rate
- Low Cost
- Embedded clustering or classification logic
- Embedded static and dynamic object tracker

J&S Marine Ltd are a UK based manufacturer of sonar equipment including transducers, underwater communications systems, diver navigation systems and electronic scanning sonars. J&S Marine are involved in the Hammerhead project as industrial advisers. It was decided that an existing electronic scanning sonar would be adapted for use on the vehicle. This is seen by J&S Marine as an experimental situation in which feedback will be obtained from trying different scanning configurations and the results used to define a catalogue AUV sonar product. The adapted product is shown in Figure 2. The sonar comprises an externally mounted transducer assembly (shown on top of the vehicle in Figure 2), and an internally mounted electronics unit. The sonar has the ability to operate in several different scanning modes due to the use of two transmitter systems, one fixed and one steered. This will allow the University of Plymouth to experiment with different methods of collision avoidance and sea bed terrain following. The transducer assembly is designed to rotate to provide a horizontal or a vertical scanning capability. In the experimental version this rotation angle must be set and fixed prior to each run. In the production model this feature will be software selectable. Normally, the sonar would be incorporated into the nose of the vehicle, but Hammerhead has a drop weight in the nose which is released if the vehicle floods.

System Overview

Figure 3 shows the system block diagram. The sonar receiver is a narrow band electronically scanned system using an FFT beamformer. There are two transmitters, one having fixed steer and the other using a set of phase shifters to steer the beam. All aspects of the sonar are controlled by a low power DSP – a Texas TMS320LF2407A. This device is a 16 bit fixed point DSP running at 40 MIPS. It has 32k of onchip FLASH memory and 2.5k of onchip RAM. Whilst the device was originally targeted at motor control, the fact that it has extensive I/O capabilities makes it ideal for embedded applications. Each sonar transmission is initiated by the DSP which selects the appropriate transmitter, and controls the frequency and duration of the transmit pulse. During the subsequent reception period, the DSP controls the digitisation of the return signals from the 16 elements in the linear receiver array. The returns are amplified, quadrature detected and converted into digital form prior to storage in FIFO memory. The DSP reads the samples from memory and performs a 64 point complex FFT on them to form a set of quarter beams. The DSP will typically digitise 256 sets of beams across the operating range of the sonar. At each of these range bins there are 64 beams across the 60 degree sector. The DSP can then perform amplitude detection on the target information in order to determine if there are obstacle ahead of the AUV. Target detection information is passed to the AUV control processor by means of an RS485 serial data link. The control processor can adjust the operation of the sonar over the serial link by means of a set of commands to determine operating range, scanning configuration and detection parameters.

Receiver array

Figure 4 shows the transducer configuration. The receiver array consists of 16 elements spaced at

one wavelength intervals. This results in a receive beam which is just over 3 degrees wide and which can be steered over a 60 degree sector. In the non – steered direction the elements are curved to produce an acceptance angle of +/- 30 degrees.

Fixed transmitter array

Figure 4 shows the location of the fixed transmitter array. It produces a beam which is 60 degrees wide in the direction of scan. The beam cuts off sharply at the edges of the scan sector to reduce the aliasing effects of the under sampled receiver array. In the non scanned direction the beam is 25 degrees wide.

Steered transmitter array

This array is similar to the receiver array. Each element is driven by an individual power amplifier coupled to a digital phase generator. The effect of this is to allow the 3 degree wide transmit beam to be steered over a 60 degree sector in a direction orthogonal to the steer direction of the receiver. Figure 5 illustrates this.

Operating Modes

Figure 6 shows the ability of the mounting arrangement to allow the sonar to be operated in either a horizontal or a vertical scan configuration. Figures 7, 8 and 9 show the three modes of operation. These are, respectively:

- Mode 1: Fixed, broad vertical beam transmitter, horizontally scanned receiver
- Mode 2: Steered, narrow vertical beam transmitter, horizontally scanned receiver
- Mode 3: Steered, narrow horizontal beam transmitter, vertically scanned receiver

Mode 1: Fixed beam mode is commonly employed by most commercial forward looking sonars. In this mode, the sonar is capable of ensonifying a large region in a single ping, which is vital for the purpose of obstacle detection. Here, range and bearing information are acquired but not depth. This makes it suitable for an AUV performing mid-sea surveying and mine-searching missions, where the environment is uncluttered or sparse. However, discrimination of object depth can be difficult. In the case of shallow water (200m or less) and when the AUV is cruising at low altitude, near to the sea bed performing pipe tracking or terrain following mission, the combined effect of boundary reverberation noise, multi-path returns and bottom clutter can rapidly degrade the sonar data [1].

Mode 2: By incrementally sweeping the fan-like horizontally beams vertically, as much as 3° every ping, for a total angle of 60°, the sonar performance in object discrimination and characterisation in the spatial domain are significantly enhanced. Furthermore, this feature provides an ability to focus on certain region of interest, which is critical in tracking applications. Ultimately, the temporal and spatial information extracted from this mode, allows the AUV to conduct more complicated missions such as mine-hunting and optimal motion planning in 3D.

Mode 3: Mode 3 is similar to mode 2, but with the transducer being rotated ninety degrees. The non-steerable version of this mode is commonly employed by surface vessel for collision avoidance purposes. Unlike mode 2, this mode tends to put more emphasis on discriminating objects in the vertical direction (terrain) rather than in the horizontal direction (suspended objects). An AUV performing a terrain hugging manoeuvre such as pipeline tracking or seafloor surveying, needs to estimate the terrain slope in advance in order to initiate effective manoeuvre, hence justifying the use of this mode.

LOW POWER TECHNIQUES FOR AUV SONARS

Figure 10 contrasts the approach adopted for the low power AUV sonar with a conventional electronically scanned sonar. The conventional approach uses a sonar head which typically will consume about 25W – 50W of power from the

vehicle supply. The output of the sonar head is fed to a PC for processing and display. The PC may be an embedded board, in a format such as PC104. The PC will typically consume a further 5W – 10W of power.

In the approach described here, the sonar achieves a power consumption of 5W by a combination of the following techniques:

1) Reduced number of receive channels
2) Analogue front end power control
3) Low power fixed point DSP
4) No PC used for processing or display

1) The small number of receive channels saves power by reducing the number of analogue amplifiers and analogue to digital converters required. In a conventional imaging sonar the use of only 16 channels can compromise the image quality by reducing the number of receive beams which can be displayed. However, in a collision avoidance sonar, there is no need to present an image to a human operator, and hence subjective image quality is not an issue. What does matter is the ability to detect and localise an object which the AUV must avoid. To do this the sonar must have an operating range and scanned sector which are sufficiently large to guarantee obstacle avoidance within the operating envelope of the vehicle.

2) The use of power switching on the analogue circuits and analogue to digital converters ensures that power is not wasted during the time between sonar transmissions. This technique relies on the fact that the DSP requires a finite time to process the detected signals from the sonar and relay the target detection information to the AUV control processor. During this time the analogue circuits are switched off and experience shows that this produces a considerable power saving.

3) The DSP used in the AUV sonar has inherently low power consumption. It consumes less than 0.5W. The compromise that must be accepted when using such a device is limited processing speed. At 40MIPS the device is capable of processing about 4 frames of sonar data per second in a conventional imaging sonar mode. In the AUV sonar it is possible to reduce the number of operations performed by the DSP when compared to those required for presentation of the image to a human operator. Figure 11 illustrates this point. The processing chain illustrated at the top of the figure is representative of a typical electronic imaging sonar used for collision avoidance. The darker coloured processing blocks would typically be carried out by a DSP, the lighter coloured blocks represent the operations performed on a PC to present the image to an operator. In this case the target detection and tracking operations are performed in software running on the PC. The output is usually some form of serial data stream which carries target position data to the control computer. In the case of the low power AUV sonar, the blocks at the bottom of Figure 11 all run on the DSP. It is also possible to reduce the number of the blocks by eliminating those (such as dynamic range compression and PPI transformation) which are solely concerned with presentation of data to a human operator. By making these reductions in the processing required it is possible to use a low power DSP of limited performance and still achieve frame rates comparable to those achieved by a conventional approach using much higher power.

4) The observations made in the previous section on eliminating the need for an MMI or PPI display also mean that there is no need to use a PC as a display device. This saves a large amount of power and simplifies the system architecture and software.

TARGET DETECTION AND OBTACLE AVOIDANCE TECHNIQUES

Raw sonar data tends to be corrupted with noise. This requires additional processing stages to obtain a better representation of the environment.

Sonar signal processing can be decomposed into the processes shown below [2]:

- Filtering and Segmentation
- Feature Extraction
- Tracking
- Map Building

Filtering and Segmentation

The first step of sonar signal processing typically entails the elimination of noise and backscatter of sonar image caused by scattering and reverberation. This can be achieved by applying simple Gaussian, median or mean filter to the image. Then, a form of segmentation process, popularly known as thresholding is then applied to the image to enhance object background discrimination. Typically, this is sufficient but for performance critical application, a more sophisticated version, which is adaptive via switching function-integration can also be implemented [3].

Most segmentation algorithms are very computationally expensive and time consuming. As a result, some authors advocate using selective, data compressing and multirate / multidepth filtering techniques. In a selective approach, the static and dynamic parts of the image are discriminated using the frequency domain method (1 dimensional Fast Fourier Transform (FFT)) or time domain method moving average [4]. Once the dynamic object is detected, it will be tracked and segmented only in the region of interest. For static objects, only new objects need to be segmented. The multirange / multirate approach attempts to redistribute the computational load by sampling the area at various rates depending on their importance. As illustrated in Figure 12, the region adjacent to the AUV is more critical and requires a higher sampling rate [5]. As an alternative, Zanoli et al. [6] attempted to compress the sonar data before filtering, significantly reducing the processing requirement. A combination of these methodologies will be applied to this prototype sonar.

Feature Extraction

Feature extraction is a process that is intimately linked with object classification. In image processing, feature extraction entails accurate measurement of object features. Ideally, the feature selected should be invariant under various circumstances while extracting maximum information regarding the object. Such features can be object dimensions such as area, perimeter, or more complicated parameters such as moments, mean, and variance used to describe statistical distributions.

Tracking

In this context, tracking is a process whereby the object attributes such as position and speed, prediction accuracy are estimated, compared and recorded. In video processing, one tries to correlate a predetermined features with subsequent frame features, at the same time noting their differences. The accuracy of the predicted target attributes plays a critical part in characterising its behaviour. These attributes are vital for successful collision avoidance as they assist in quantifying the risk. Lane *et al.* [7] employed optical flow with associative searching trees while Moran *et al.* [8] proposed using multiple hypotheses for object tracking. Nonetheless, both these pixel based schemes are very computational expensive, precluding their application in time-critical applications.

Alternatively, the classical Kalman filter has also been applied successfully to sonar tracking systems [5, 9, 10]. Currently, we are on the stage of employing two different Kalman filters for tracking dynamic and static objects. Isolating the two simplified the analysis and renders it less computational demanding. However, this is still at a preliminary stage, and this methodology is likely to change depending on the outcome. For the case of a dynamic obstacle, the implemented tracking filter is employed to track the state vector

$S_k = [S_k \quad \dot{S}_k]^T$ for each of the four state variables S_x, S_y, S_z, (position) and additional S_r which is the radius of the circle circumscribing the object. It is used to characterise the predictability, hence the collision risk of the obstacle. Alternatively, including object area and change of area in the state vector can also enhance the tracking performance [10]. Tracking of static obstacles is the responsible of another simplified Kalman filter that possess identical state variables as the former except for the derivative terms.

In is a well known fact that Kalman filters suffer from the curse of dimensionality. Increasing the number of tracked obstacles will rapidly overload the filter. Thus for those objects that shift out of the field of view, they should be tracked for only a predefined time. Once the predefined period has expired, an action needs to be taken for both static and dynamic targets. Initially, the static object will be stored and the countdown counter set. Storing the object permits quick retrieval in case the AUV is navigating the area again. However, without a global frame of reference, the AUV is relying on a dead reckoning system that is susceptible to constant sensor drift, making the information unreliable. Once the counter countdown is reached, the object is deleted. Dynamic objects are too unpredictable, and should not be tracked more than necessary. The lifespan of a dynamic object is determined by a countdown counter, however, in this case with a shorter timeframe.

Map Building

This is still an open topic and will be updated as the research progress. Initially, we envisage using a geometrical representation scheme to characterise the AUV environment. Here, dynamic obstacles will be considered as spheres and static obstacles as cylinders (vertical projection of a circle). Nevertheless, other forms such as spatial decomposition will also be investigated. A hybrid method also sounds promising, where each of the scheme's advantages can be exploited.

Collision Avoidance Techniques

Reflexive avoidance techniques are based on a sensor-based approach, where the information from the sensors is sent directly to the actuator without passing through the high-level modules. This makes them fast and capable of handling dynamic situations. Lately, some AUVs have used more advanced reflexive modules that are integrated with short-term memory, enabling the AUVs to exhibit simple projective planning capability, such as escaping from trap circumstances.

Fuzzy logic is a rule based, multivalued logic inference system that tries to take account of uncertainty and in the real world. Fuzzy logic's ability to quantify abstract expert knowledge has made it a choice in solving complicated systems. Complex rules can be promptly constructed without resorting to obscure mathematical techniques. Its transparent nature, excellent immunity to noise and error, and real-time performance makes it an ideal choice for designing both high and low level controllers.

Various researches have successfully implemented fuzzy logic systems in their collision avoidance modules [11, 12]. The proposed collision avoidance system is based on a hybrid of fuzzy logic and virtual force field paradigms. The virtual force field paradigm seeks to simulate an artificial force field surrounding the vehicle. Invariably, any contact with neighbourhood objects will cause deformation of the force field. The objective is to minimise these deformations by locally modifying the control vector. Fuzzy logic will be used to assist in mapping different parameters such as obstacle position (zones), speed, risk factor, etc into behaviours. These behaviours such as "maintain height", "circumnavigation" and "avoid" will then initiate the corresponding actuator actions.

One inherent weakness of the reflexive technique is its propensity to get trapped in certain

circumstances. Furthermore, the trajectory generated is highly non-optimal. Because of this, the proposed reflexive avoidance technique is to be complimented with advance motion planning techniques (research phase) that is capable of generating near optimal paths and escaping from trapped conditions. A block diagram of the proposed collision avoidance architecture is illustrated in Fig 13. The arbiter is used to coordinate the activation and inhibition of various modules.

CONCLUSIONS

The sonar presented has been designed for use on an AUV, rather than simply being an existing imaging sonar fitted to an AUV. Power savings have been shown to result from this approach which have a corresponding benefit to the endurance of the vehicle. The sonar has been designed to operate in several scan modes which are intended to improve its usefulness in vehicle navigation and obstacle avoidance. The resulting equipment is to be fitted to the Hammerhead vehicle as an experimental platform for assessing the optimum scanning system for an electronic AUV sonar. A series of experiments will be carried out over the next 12 months during which time the sonar performance will be evaluated and refined. It is the intention of J&S Marine to offer the resulting system as a commercial product for the AUV market within the next 18 months.

ACKNOWLEDGEMENT

The authors wish to thank Dr. S.Tetlow of the University of Cranfield for his help in the design of the AUV sonar.

REFERENCES

[1] F. Nussbaum, G. T. Stevens, and J. G. Kelly, "Sensors for a Forward-Looking High Resolution AUV Sonar," *IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV)*, pp. 141-145, 1996.

[2] Y. Petillot, I. T. Ruiz, and D. M. Lane, "Underwater Vehicle Path Planning Using a Muti-beam Forward Looking Sonar," *IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV)*, pp. 1194-1199, 1998.

[3] D. M. Lane, M. J. Chantler, E. W. A. Roberston, and A. G. McFadzean, "A Pyramidal Architecture for Knowledge-Based Sonar Image Intepretation," *IEE Colloquium on Transputers for Image Processing Applications*, Feb 1989.

[4] D. Y. Dai, M. J. Chantler, D. M. Lane, and N. Williams, "A Spatial-Temporal Approach for Segmentation of Moving and Static Objects in Sector Scan Sonar Image Sequences," *IEE 5 th International Conference on Image Processing and Its Applications*, no. 163-167, 1995.

[5] L. Henriksen, "Real-Time Underwater Object Detection Based on an Electrical Scanned High-resolution Sonar," *IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV)*, pp. 99-104, 1994.

[6] S. M. Zanoli and F. Afftaitati, "Obstacle Avoidance with Scanning Sonar for Bottom Navigation," *IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV)*, pp. 537-545, 1999.

[7] D. M. Lane, M. J. Chantler, and D. Y. Dai, "Robust Tracking of Multiple Objects in Sector Scan Sonar Image Sequences Using Optical Flow Motion Estimation," *IEEE Journal of oceanic Engineering*, vol. 23, pp. 31-46, Jan 1998.

[8] B. A. Moran, J. J. Leonard, and C. Chryssostomidis, "Geometric Shape from Sonar Ranging," *IEEE International Symposium on Unmanned, Untethered,*

Submersible Technology, pp. 370-383, 1993.

[9] G. N. Williams, G. E. Lagace, and A. Woodfin, "A collision avoidance controller for autonomous underwater vehicles," *IEEE Proc. Symposium Autonomous Underwater Vehicle Technology (AUV)*, Jun 1990.

[10] E. Trucco, I. T. R. Y R. Pettilot, K. Plakas, and D. M. Lane, "Feature Tracking in Video and Sonar Subsea Sequences with Applications," *Computer Vision and Image Understanding*, vol. 79, pp. 92-122, May 2000.

[11] N. Shinjo and J. S. R. Graeme, "Sensor-Based Fuzzy Obstacle Avoidance System for Autonomous Underwater Vehicles," *IEEE International Symposium on Unmanned, Untethered, Submersible Technology*, 9 th , pp. 271-298, 1995.

[12] X. Liu, S. Zhang, Y. Li, and Y. Shang, "A New Method for AUV's Collision Avoidance," *IEEE International Symposium on Unmanned, Untethered, Submersible Technology, 11 th* , pp. 587-591, Aug 1999.

Figure 1 The Hammerhead AUV

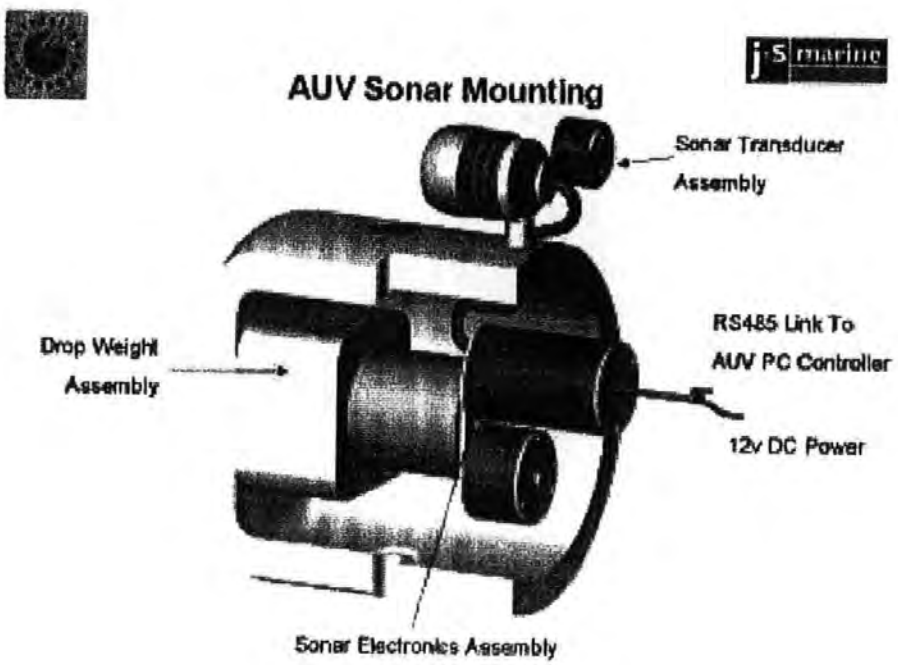


Figure 2 :

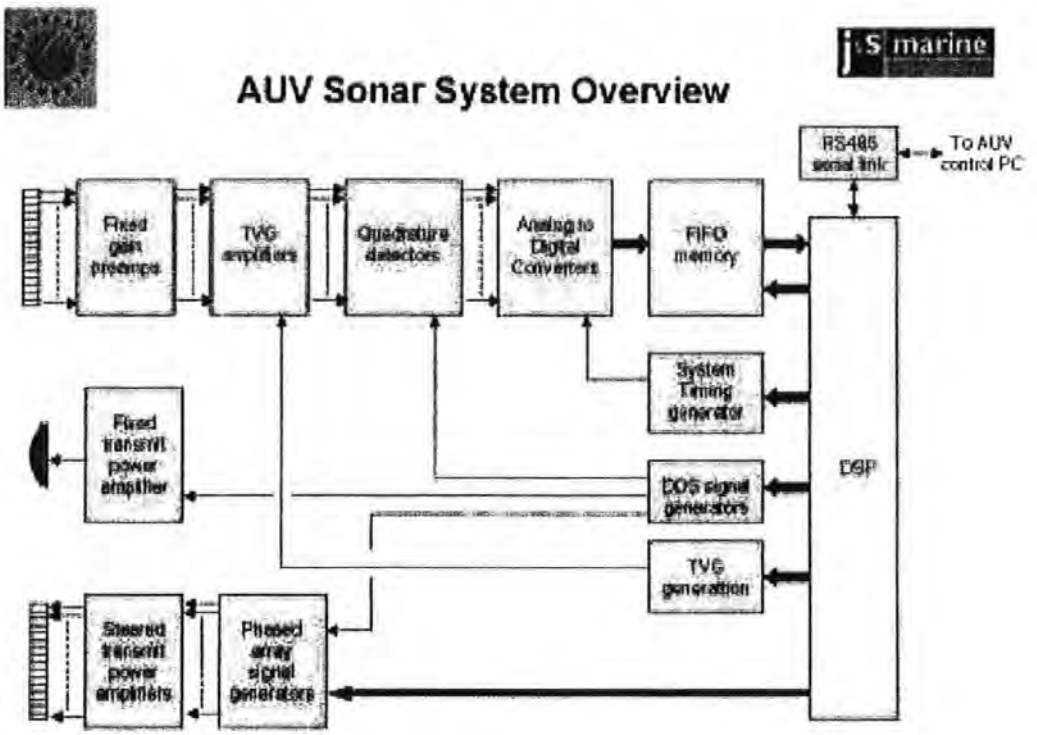


Figure 3

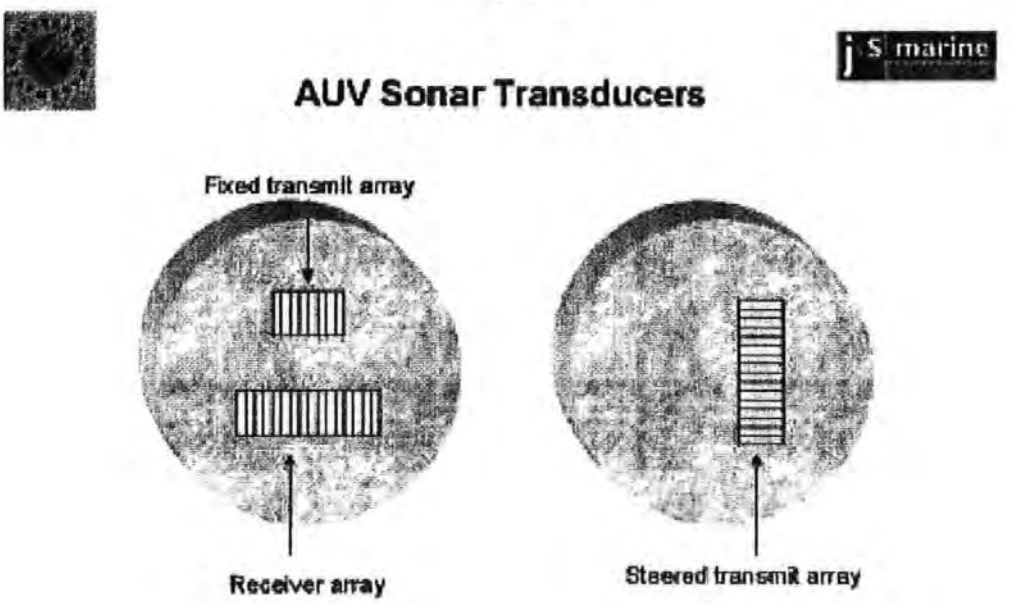


Figure 4

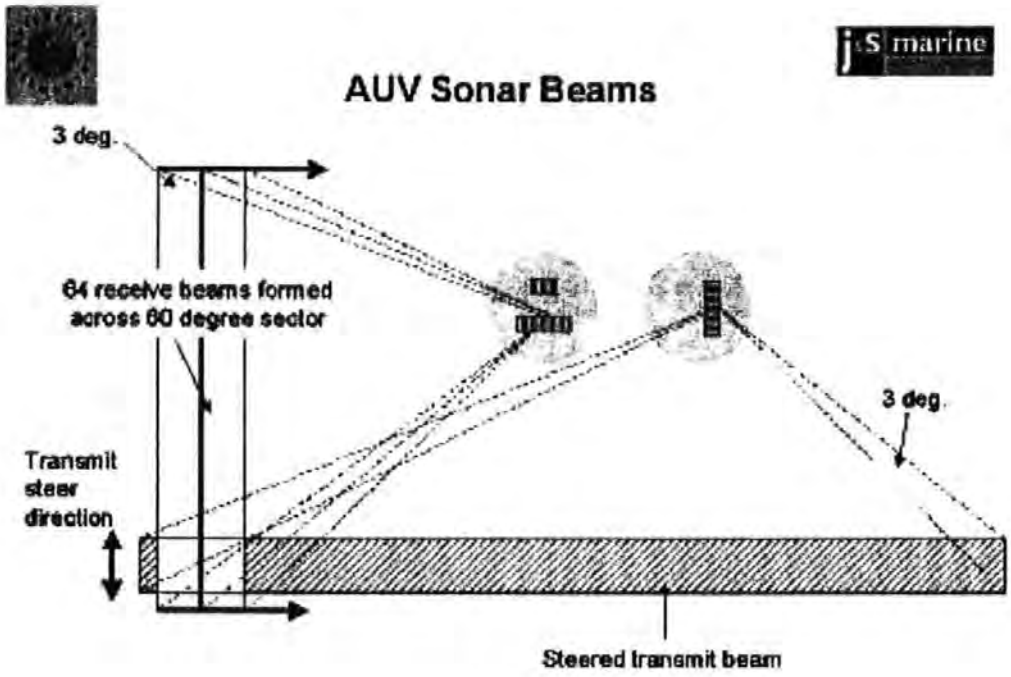


Figure 5

Figure 6

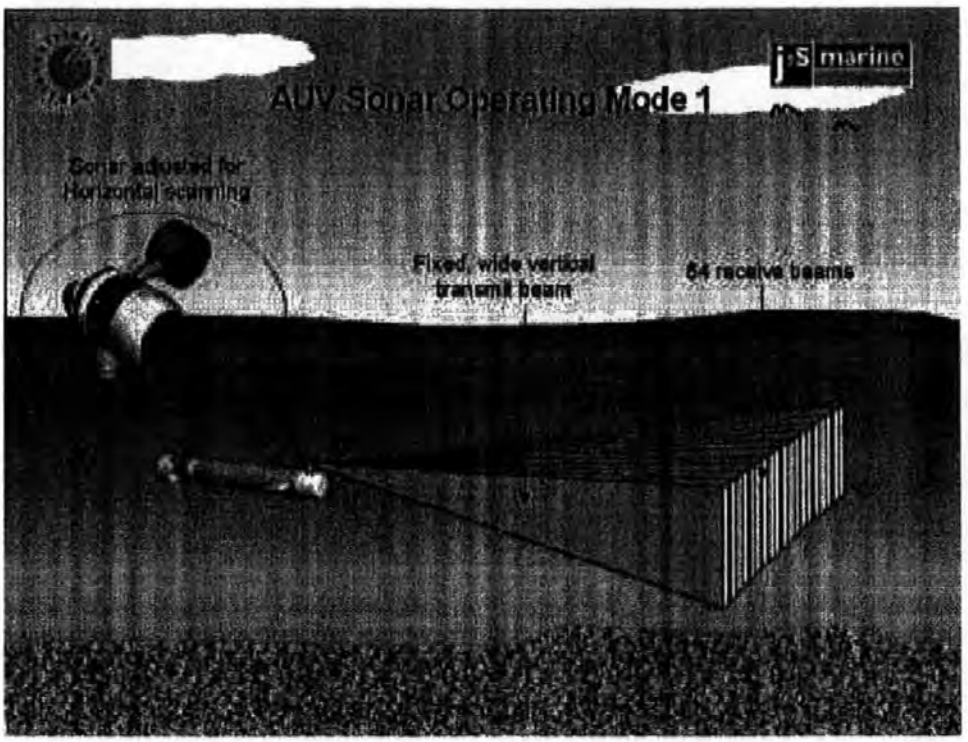


Figure 7

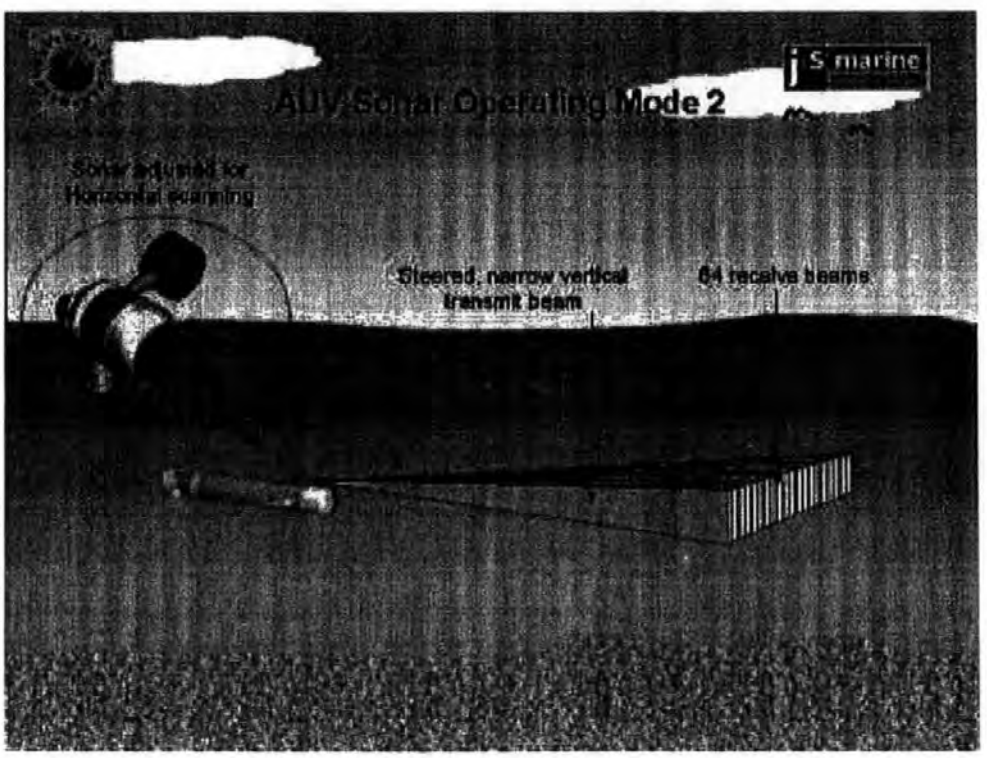


Figure 8

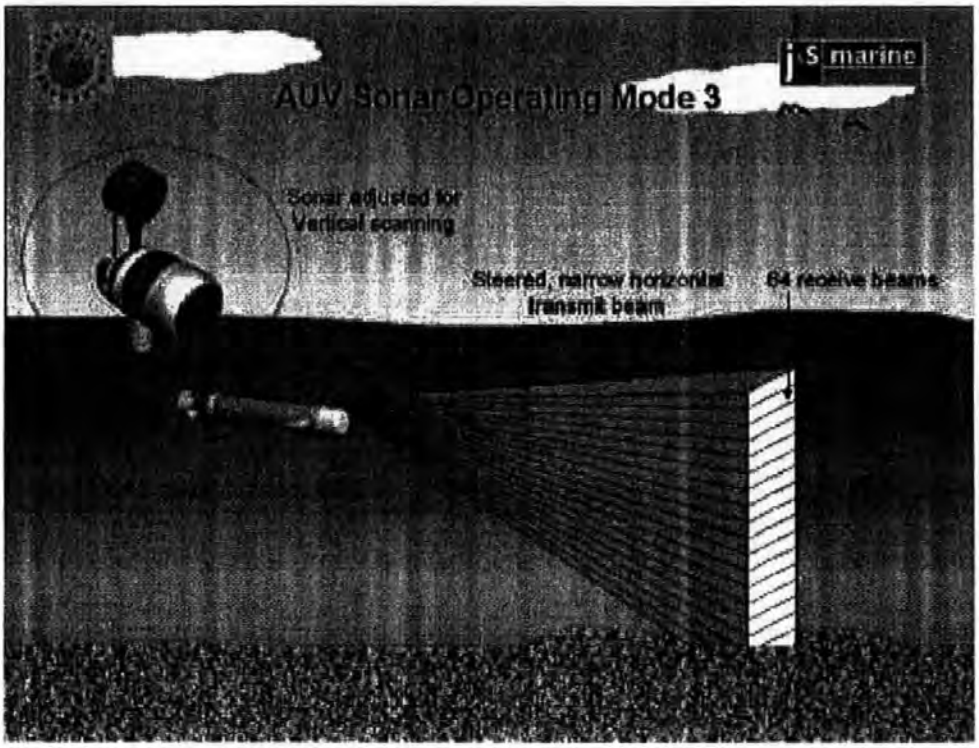


Figure 9

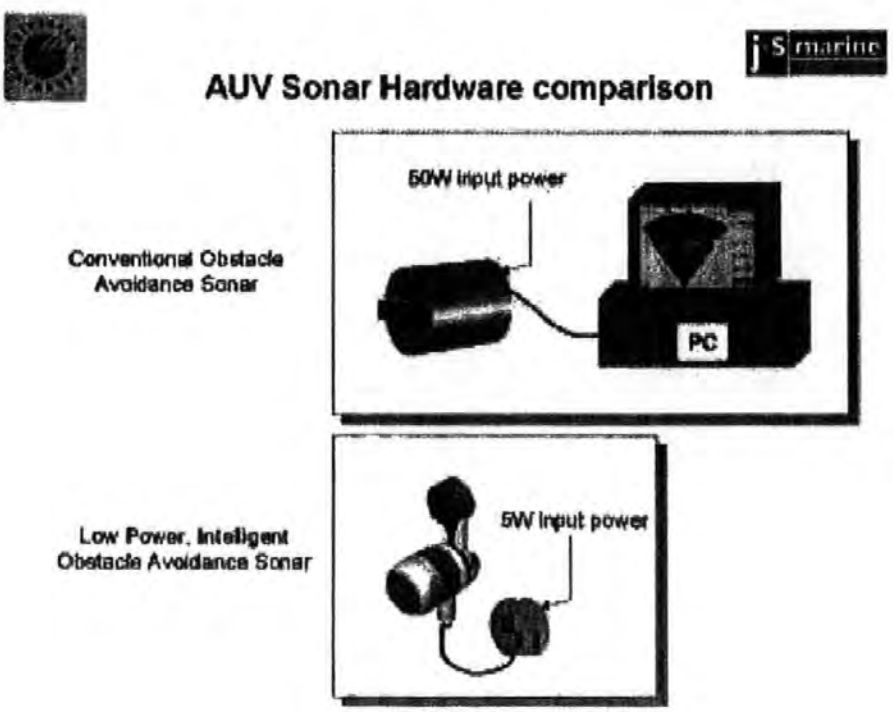


Figure 10

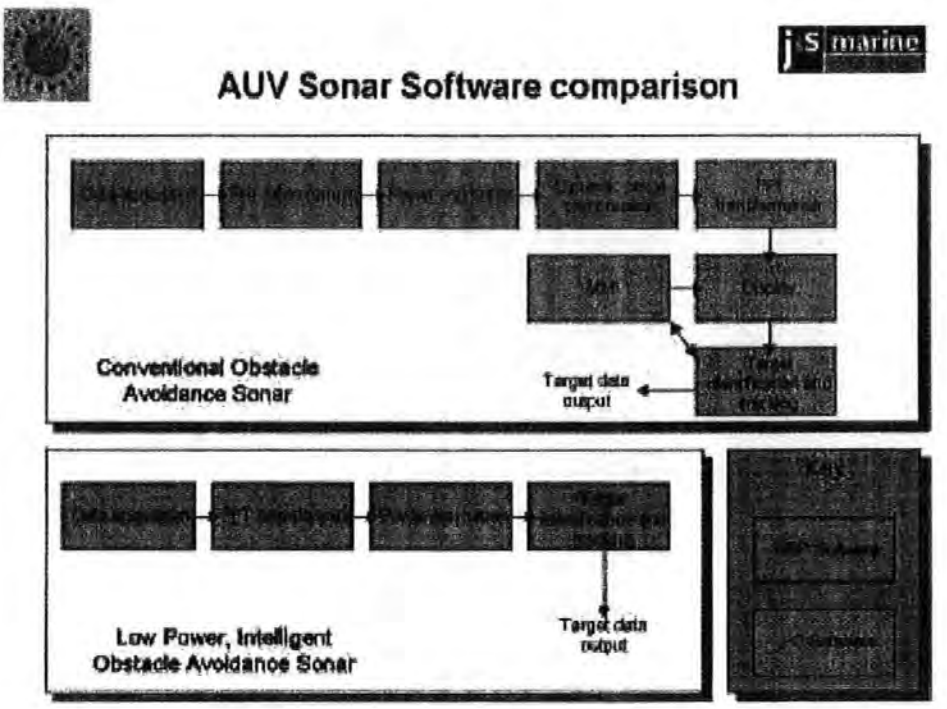


Figure 11

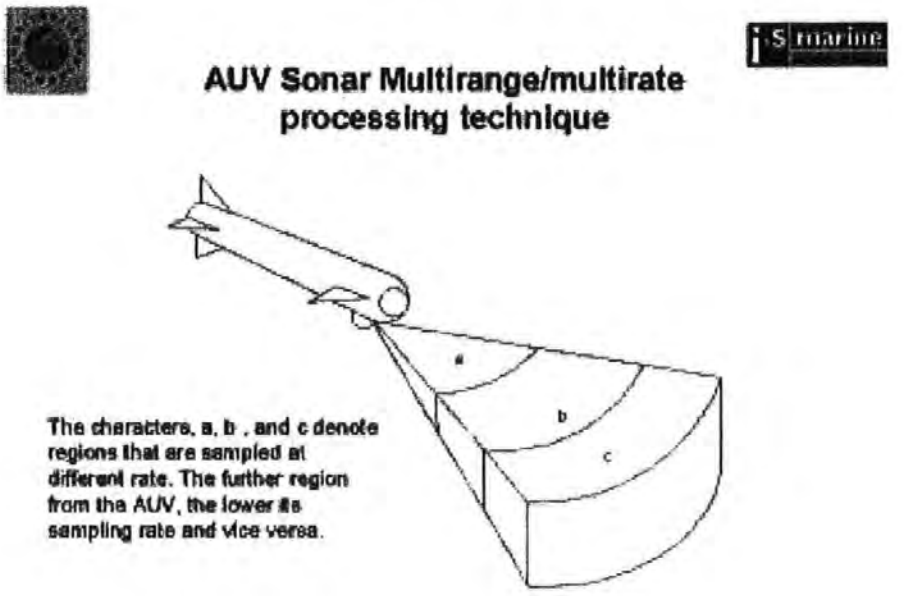


Figure 12

AUV Sonar Collision Avoidance Architecture

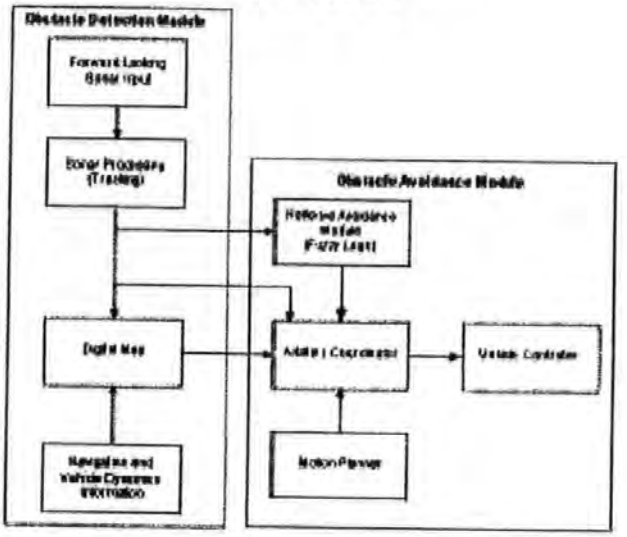


Figure 13

Appendix C

AT500 Sonar Data Sheet

Electronically Scanned Collision Avoidance Imaging Sonar

AT500

Description

AcuTrak PSE Ltd has developed the AT500 imaging sonar for use on Remotely Operated Vehicle (ROV) and Autonomous Underwater Vehicle (AUV). The AT500s use of the latest DSP control and data processing electronics has resulted in a highly compact yet powerful imaging sonar which can be used to aid the ROV pilot and for collision avoidance in AUV. By interfacing the AT500 to a PC, sonar images can be displayed in real-time and logged for later editing and analysis.

FEATURES

- **Electronically scanned Sonar**
- **Rapid scan rate**
- **60° horizontal scan angle**
- **Switchable vertical beam width**
- **Compact size**
- **Real-time Range and Bearing information**

SYSTEM OVERVIEW

Sonar

The AT500 uses a transducer array to electronically scan for underwater object up to 75m away over a 60° arc in front of the sonar.

PC Software

Interfaced to a PC, AcuTrak software allows images from the at500 to be displayed in real-time. Clicking on the sonar display give information on range and bearings of targets.

Controls

Simple on-screen buttons are used to configure the AT500. The operator has control over range, gain, clipping, and beam width.

Data Logging

The PC software can be used to capture sequences of sonar scans into a log file. This data can then be replayed, edited and analysed.

Construction

The sonar is housed in a lightweight anodised aluminium cylinder. The end caps are edged in polyurethane to give resistance to impacts. An external connector is located at the rear of the housing for power input and data communications

Environment

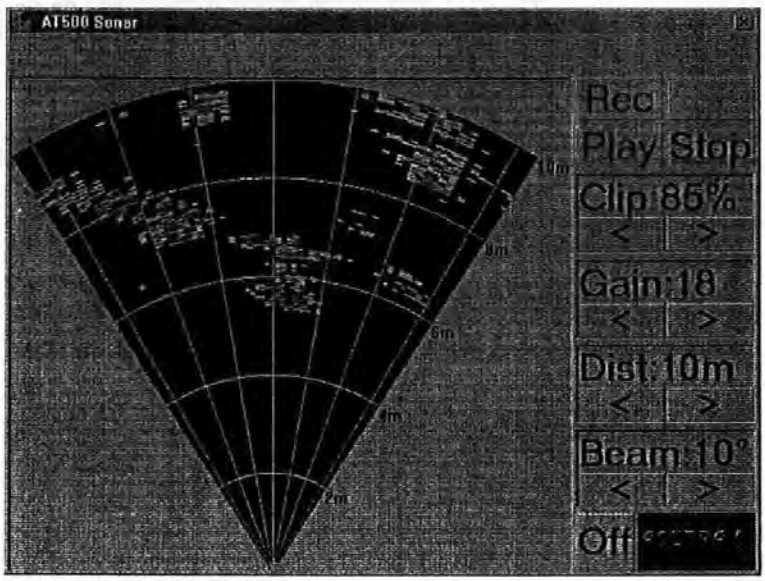
The AT500 has been design to withstand a harsh environment. It is rated to a standard operational depth of 1000m.

AUV option

As an option, the AT500 can be used in stand-alone mode without the need for a separate PC. This mode is intended for use as a collision avoidance and navigation sonar for AUVs. The AT500 detects obstacles in the AUV's path and sends target range and bearing data to the AUV navigation system via a serial link.

SPECIFICATIONS

Parameter	Value
Operating Frequency	500kHz
Beam Number	64°
Vertical Scan Angle (Minimum)	3.3°
Vertical Scan Angle (Maximum)	13.4°
Horizontal Scan Angle	60°
Horizontal Beam Width	2.7°
Maximum Range	75m
Range Resolution	0.1m
Dimensions	Ø100 mm x 190 mm
Weight in water	Slightly Negative
Connector	SubConn® MCBHRA8M
Power	24V DC



CONTACT

Peter ROBINSON
AcuTrak Precision Survey Equipment Ltd
AcuTrak House
Ironworks Way
Warton Road
Carnforth
Lancs. LA5 9EU
United Kingdom

- Tel: +44 1524 736973
- Fax: +44 870 164 1665

e-mail peterr@acutrak.demon.co.uk