Visual Homing for a

Car-like Vehicle

Kane Usher, B. Eng. (Mechanical) (Hons I)

A thesis submitted to the
School of Mechanical, Manufacturing & Medical Engineering
for the award of

Doctor of Philosophy

at
the Queensland University of Technology
Brisbane, Australia

2005

Keywords

mobile robots; nonholonomic systems; control of nonholonomic systems; control of car-like vehicles; pose stabilization; non-linear control; switching control; computer vision; omnidirectional vision; panoramic vision; colour segmentation; object tracking; visual homing.

Abstract

This thesis addresses the pose stabilization of a car-like vehicle using omnidirectional visual feedback. The presented method allows a vehicle to servo to a pre-learnt target pose based on feature bearing angle and range discrepancies between the vehicle's current view of the environment and that seen at the learnt location. The best example of such a task is the use of visual feedback for autonomous parallel-parking of an automobile.

Much of the existing work in pose stabilization is highly theoretical in nature with few examples of implementations on 'real' vehicles, let alone vehicles representative of those found in industry. The work in this thesis develops a suitable test platform and implements vision-based pose stabilization techniques. Many of the existing techniques were found to fail due to vehicle steering and velocity loop dynamics, and more significantly, with steering input saturation. A technique which does cope with the characteristics of 'real' vehicles is to divide the task into predefined stages, essentially dividing the state space into sub-manifolds. For a car-like vehicle, the strategy used is to stabilize the vehicle to the line which has the correct orientation and contains the target location. Once on the line, the vehicle then servos to the desired pose. This strategy can accommodate velocity and steering loop dynamics, and input saturation. It can also allow the use of linear control techniques for system analysis and tuning of control gains.

To perform pose stabilization, good estimates of vehicle pose are required. A simple, yet robust, method derived from the visual homing literature is to sum the range vectors to all the landmarks in the workspace and divide by the total number of landmarks — the Improved Average Landmark Vector. By subtracting the IALV at the target location from the currently calculated IALV, an estimate of vehicle pose is obtained. In this work, views of the world are provided by an omnidirectional camera, while a magnetic compass provides a reference direction. The landmarks used are red road cones which are segmented from the omnidirectional colour images using a pre-learnt, two-dimensional lookup table of their colour profile. Range to each landmark is estimated using a model of the optics of the system, based on a flat-Earth assumption. A linked-list based method is used to filter

the landmarks over time. Complementary filtering techniques, which combine the vision data with vehicle odometry, are used to improve the quality of the measurements.

Contents

List of Figures

List of Tables

List of Algorithms

List of Original Technical Publications

The following papers were published as a result of the work conducted in this thesis:

- Kane Usher, Peter Corke and Peter Ridley: A Camera as a Polarised Light Compass: Preliminary Experiments. *Proceedings of the Australasian Conference on Robotics and Automation*, Sydney, Australia,(2001), available at http://www.araa.asn.au/acra/acra2001/index.html

- Kane Usher, Peter Corke and Peter Ridley: Visual Servoing of a Car-like Vehicle – An Application of Omnidirectional Vision, *Proceedings of the Australasian Conference on Robotics and Automation*, Auckland, New Zealand (2002), available at http://www.araa.asn.au/acra/acra2002/index.html

- Kane Usher, Peter Corke and Peter Ridley: Home Alone: Mobile robot visual servoing, *International Conference on Intelligent Robots and Systems - Visual Servoing Workshop*, Lausanne, Switzerland (2002).

- Kane Usher, Matthew Dunbabin, Peter Corke and Peter Ridley: Sensing for Visual Homing, *Proceedings of the Australasian Conference on Robotics and Automation*, Brisbane, Australia (2003), available at http://www.araa.asn.au/acra/acra2003/index.html

- Kane Usher, Peter Corke and Peter Ridley: Landmark-based Visual Homing: Experiments with a Non-holonomic Vehicle, *Proceedings of the Conference on Field and Service Robots*, Yamanashi, Japan (2003). A modified version of this paper is also available in a post-conference proceedings publication in Springer Tracts on Automation and Robotics.

- Peter Corke, Dimitris Symeonidis and Kane Usher: Tracking Road Edges in the Panospheric Image Plane, *Proceedings of the International Conference on Intelligent Robots and Systems*, Las Vegas, USA (2003).

- Kane Usher, Peter Corke and Peter Ridley: Visual servoing of a car-like vehicle – an application of omnidirectional vision, *Proceedings of the International Conference on Robotics and Automation*, Taipai, Taiwan (2003).

Nomenclature

The most commonly used symbols and notation are listed below. Note that some symbols are used multiple times and their meaning is context dependent.

$\underline{x}$	a vector	
x_i	ith component of vector $\underline{x}$	
$\hat{x}$	an estimate of x	
x^*	a demanded value of x	
$\dot{x}$	the time derivative of x	
$\mathbf{A}$	a matrix	
$\underline{A}^T$	Transpose of $\underline{A}$	
α	equiangular mirror gain	
α_i	angle to landmark i measured with respect to the x-axis	$[\mathrm{rad}]$
$\underline{\alpha}$	the vector of absolute landmark angles	
β_i	angle to landmark i measured with respect to the vehicle	$[\mathrm{rad}]$
$\underline{\beta}$	the vector of relative landmark angles	
d	distance between front wheels	$[\mathrm{m}]$
δ	bearing angle to the origin relative to the x-axis, measured with respect to the vehicle	$[\mathrm{rad}]$
e	distance to the origin	$[\mathrm{m}]$
f	camera focal length	$[\mathrm{m}]$
ϕ	steering angle	$[\mathrm{rad}]$
g	image distance with respect to image centre	$[\mathrm{pixels}]$
Γ	a vector of gains	
γ_θ	complementary filter gain for orientation estimation	
γ_r	complementary filter gain for landmark range estimation	
γ_β	complementary filter gain for landmark bearing estimation	
$\underline{H}$	a homing vector	
$\mathbf{J}$	image Jacobian	
k_i	gain i	
K_v	gain for velocity loop response	
L	distance between front and rear axles	$[\mathrm{m}]$
$\underline{L_i}$	the range vector to landmark i (consists of $R_i \angle \alpha_i$)	
p	CCD pixel pitch	$[\mathrm{m/pixel}]$
θ	orientation of vehicle longitudinal axis with respect to the x-axis	$[\mathrm{rad}]$
ω	vehicle angular rate	$[\mathrm{rad/s}]$
ω_n	natural frequency of the steering response	$[\mathrm{rad/s}]$

r	equiangular mirror radius	$[\mathrm{m}]$
r_0	peak equiangular mirror radius	$[\mathrm{m}]$
r_{min}	minimum equiangular mirror radius	$[\mathrm{m}]$
r_w	radius of front left-hand wheel	$[\mathrm{m}]$
$\underline{R}$	the vector of landmark ranges	
R_i	the range to landmark i	$[\mathrm{m}]$
s	the s operator in the Laplace domain	
t_s	settling time	
τ_v	time constant for velocity loop response	
$\underline{U_i}$	the unit vector to landmark i (consists of $1\angle\alpha_i$)	
u	distance on CCD plane	$[\mathrm{m}]$
U	the potential energy of a system	
v	vehicle speed	$[\mathrm{m/s}]$
V	Lyapunov energy function	
x, y	Cartesian position	$[\mathrm{m}]$
ψ	bearing angle to the origin relative to the vehicle longitudinal axis	$[\mathrm{rad}]$
ζ	damping ratio of the steering response	

Further conventions used:

- $\ominus$ An operator which signifies angular subtraction with appropriate handling of wrap-around effects
- $\oplus$ The corresponding angular addition operator
- The *sign* of a variable is defined as:

$$\text{sgn}(a) \quad = \quad \begin{cases} 1 & \text{if } a \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{1}$$

- The *minimum* of two variables is defined as:

$$\min(a, b) \quad = \quad \begin{cases} a & \text{if } a \leq b \\ b & \text{otherwise} \end{cases} \tag{2}$$

- The *maximum* of two variables is defined as:

$$\max(a, b) \quad = \quad \begin{cases} a & \text{if } a \geq b \\ b & \text{otherwise} \end{cases} \tag{3}$$

- To represent *saturation* of a variable, the following function is used

$$\text{sat}(a, b) \quad = \quad \begin{cases} a & \text{if } |a| \leq b \\ b\text{sgn}(a) & \text{otherwise} \end{cases} \tag{4}$$

- The four-quadrant inverse tangent is:

$$\theta = \text{arctan2}(y, x) \tag{5}$$

where $\theta \in (-\pi, \pi]$.

Terms and Abbreviations

Following is list of the commonly used terms and abbreviations in the thesis.

ALV	Average Landmark Vector
APA	Australian Post-graduate Award
AT	the Autonomous Tractor, the experimental test-bed developed in this thesis
CCD	Charged Coupled Device
CLM	Concurrent Localization and Mapping
CMIT	CSIRO Division of Manufacturing and Infrastructure Technology
CSIRO	Australia's Commonwealth Scientific and Industrial Research Organization
DC	Direct Current
DFL	Dynamic Feedback Linearisation
dGPS	differential Global Positioning System
E-Stop	Emergency stop
exteroceptive	Pertaining to external stimuli, normally applied to biological systems but has been adopted by the robotics community
GPS	Global Positioning System
IALV	Improved Average Landmark Vector
IBVS	Image-based Visual Servoing
KLT	Kanade Lucas Tomasi (feature tracker)
LAN	Local Area Network
LED	Light Emitting Diode
LQR	Linear Quadratic Regulator
PD	Proportional and Derivative
PI	Proportional and Integral
PID	Proportional, Integral and Derivative
PLS	Proximity Laser Scanner

proprioceptive	Pertaining to stimuli on internal state, an example being an inertial sensor; normally applied to biological systems but is also used by the robotics community.
PWM	Pulse Width Modulation
QCAT	Queensland Centre for Advanced Technology
QUT	Queensland University of Technology
RTK-GPS	Real-time Kinematic Global Positioning System
SLAM	Simultaneous Localisation and Mapping
taxis	In biology, taxis is the reaction of a free organism to an external stimulus by movement in a particular direction.
VC	Vice Chancellor
VCC	Variable Constraint Control
VLSI	Very Large Scale Integration
WV	Weighted Vector

Statement of Originality

“The work contained in this thesis has not been previously submitted for a degree or diploma at any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.”

Signed: .....

Date: .....

Acknowledgments

Firstly, to my supervisors, Dr. Peter Corke and Dr. Peter Ridley, thank you for your guidance and support throughout this course of research. In particular, Peter Corke requires special mention for his tireless reading of draft material for this and other publications, his constant availability, his guidance (and patience) in broadening the horizons of a mechanical engineer, and his general enthusiasm for research. To Peter Ridley, thank you for making me clarify my thoughts and to express them more clearly.

Throughout the research, the CSIRO has provided outstanding technical and financial support without which this research would not have been possible. Acknowledgment is also due to QUT and the financial support provided through an APA and the VC grant. Jonathan O'Brien at UNSW is gratefully acknowledged for the loan of the Toro ride-on mower, subsequently modified for use in this thesis.

Special thanks to the Automation Team in the CSIRO Division of Manufacturing and Infrastructure Technology which has provided highly competent (and patient) technical support, educating a 'green' engineer in the ways of research and engineering outside of the university world. You have provided all the support a research student could ask for, and more.

Elliot Duff, thanks for teaching me much about vision systems and navigation, and for putting together the bi-variant histogram code. To Pavan and Craig, you have taught me the ways of computer and software engineering — Pavan, thank you for your tireless efforts at fixing software when I (frequently) managed to break it, and to Craig, thanks for keeping the computer system running and quickly fixing it when it didn't. To Jonathan Roberts, thanks for your help throughout, in particular with filtering methods and sensing in general. Matthew Dunbabin deserves special mention for his willingness to help out, for expanding my knowledge on control systems, and especially for the last minute reading of a draft of this work. To Les Overs, thanks for your efforts in the design stages of the Autonomous Tractor and for keeping it running in general. To Stuart Wolfe and Stephen Brosnan, thanks for your quick and competent technical support whenever I needed it. In

particular, thanks Stuart for your patience and for passing on some of the skills of your multi-faceted background. And finally to Graeme, thanks for keeping it all together and for quickly dealing with problems as they arose. In general, I would like to thank the Automation Team for providing such a stimulating work environment and for the many discussions on all topics, robotics and otherwise — you guys have taught me much about the world. It was a pleasure coming to work every day and I look forward to working with you all in the future.

To Gregg Buskey, you deserve special mention for sharing an office with me and for sharing the grief of working in experimental robotics — seeing your helicopter crash every couple of months kept my spirits up in an odd sort of way. Your, often dark, sense of humour and willingness to help out were invaluable over the last three years. May we one day stumble across a MATLAB goon!

To all my close friends, thanks for still calling me after months of idleness on my part. In particular, Melissa and Glenn, thanks for keeping me sane and (kind of) culturally educated — you guys have seen the highs and lows of my life and yet you still hang out with me! — I appreciate your friendship over the years.

To my family, thanks Mum and Dad for your wisdom, kindness, and willingness to support me in whatever I chose to do. Nanna, thanks for helping me out when I needed it, and when I didn't, we will always remember your plucky spirit; to Misty, thanks for always being supportive. To Kirsten's Mum and Dad, thanks for helping me (making me) fix all those things around the house that I've let slip over the last few months and for bringing such a wonderful person into the world.

Finally, to that wonderful person. Kirsten thank you for your unerring support and patience, your willingness to listen to me babble about things you didn't want to understand, and for giving me a life outside of robotics. Most importantly thanks for your love. You have truly turned my world around.

To Kirsten

Chapter 1

Introduction

1.1 Motivation for research

Autonomous mobile machines have potential applications in a diverse range of industries including mining, cargo handling, forestry, and construction. Such machines have the potential to improve productivity and yield, and to reduce safety incidents. For example, the operators of heavy mobile machinery are exposed to whole-body vibration often with sustained and awkward postures which puts them at significant risk to the development of musculo-skeletal disorders [Wilder and Pope, 1996; Zimmerman *et al.*, 1997; Donati, 2002]. The automation of such equipment provides a means of eliminating this exposure, with on-board machinery operators becoming remote machinery managers.

A further example is provided by the operation of machinery in environments which are either inhospitable or too dangerous for humans. The obvious benefit is the removal of operators from potentially dangerous work environments. An additional benefit is the potential to more fully exploit these operations with the accompanying increases in yield and productivity. Furthermore, autonomous mobile robots could open the door to the exploration of hostile environments — including those on other planets. The significant communication delays due to the massive distances between Earth and other interesting objects in the Universe restrict the usefulness of remotely-controlled vehicles. Already we have seen Mars through the eyes of the Mars rover; to extend the range of such missions requires significant improvements in mobile robotic technology.

1.2 Limitations of present mobile robots

Mobile robotics is a technically challenging field of research as it requires the integration of many spheres of knowledge. Progress over the last three decades has been steady but by no means astounding given the early expectations of the public and researchers alike. Before we can expect the likes of R2D2 of Star Wars fame to be an ubiquitous part of our lives, a leap of understanding in autonomous navigation and the related fields of Artificial Intelligence and cognition is required.

A significant factor hindering the development of mobile robot technology is a perceived need for complete metric models of the robot's workspace and for continuous estimates of the robot's position in the environment. This is a result of the classical approach to navigation which defines the problem in terms of [Levitt and Lawnton, 1990; McKerrow, 1991; Leonard and Durrant-Whyte, 1991]:

1. Where am I?
2. Where is everything else?
3. How do I get to other places?

This definition of navigation leads to a sense-model-plan-act architecture in which the robot's sensors are used to update, and build, a single global representation of the robot's 'world', from which, in combination with a mission specification, motor commands are derived — see Figure 1.1 for an illustration of this concept.

Building and maintaining complex metric representations of the robot's world is a difficult process which does not scale well to large environments. The computational overheads of sequentially abstracting sensor data in order to update the world model leads to a bandwidth limitation which, at times, leaves the mobile robot effectively 'blind' to the real world [Nehmzow, 2000; Thrun, 1998b].

Despite the near equivalence of presently available processing power to that available to a small insect, the best of our mobile robots are no match for an insect's system in terms

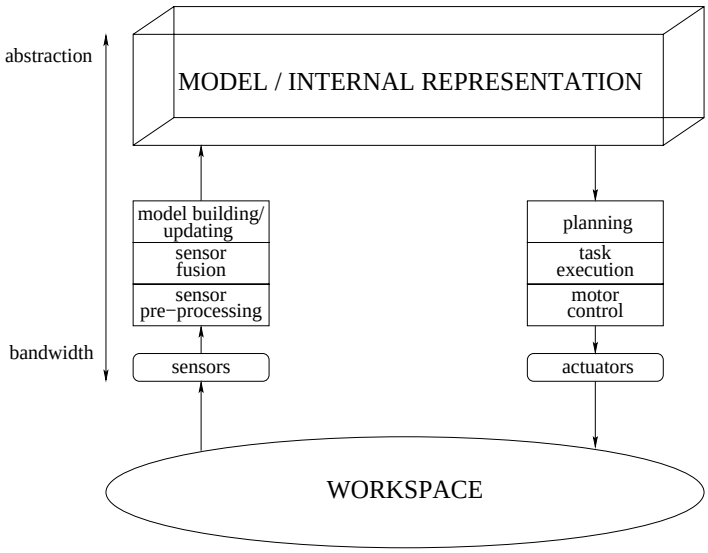


Figure 1.1: A typical hierarchical architecture used by classical navigation systems. Sensor data is passed up through the system, updating world models and the position estimate of the robot. In combination with a mission specification, the world model is then used by the path planning module, which compares estimated position to desired position. A series of tasks is then specified and the task execution module sends information to the motor control module which in turn sends commands to the robot's actuators (diagram adapted from [Corke, 1994] and [Nehmzow, 2000]).

of reliability, flexibility and performance [Srinivasan *et al.*, 1999]. Early attempts at mimicking insect behaviours resulted in the fields of behaviour-based or reactive robotics in which a tight coupling between sensing and action [Brooks, 1986; 1990; Arkin, 1995; 1989a] is used to bypass the computationally intense modelling stage in the classical approach. An example reactive architecture is shown in Figure 1.2. Instead of using a single global representation of the world, reactive systems refer directly to the environment for information; in the words of Brooks [Brooks and Flynn, 1989] 'The world really is a rather good model of itself'.

Although purely behaviour-based robots were very successful, eliciting purposive tasks from such robots was difficult. It soon became clear that in order to move forward, the classical and behaviour-based approaches had to be united [Arkin, 1989b; Gat, 1998; Jarvis, 1997]. Figure 1.3 shows an example of a hybrid system illustrating that integrating reactivity and deliberation can reduce the level of 'abstractness' required of a system. To-

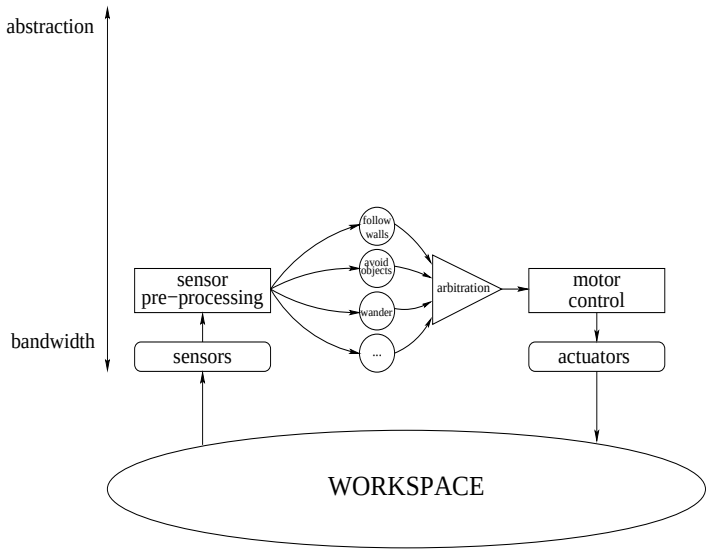


Figure 1.2: A typical architecture for a behaviour-based system. In a behaviour-based system there is a tighter coupling between sensors and actuators. Sensor information is processed in parallel by a selection of different controllers or 'behaviours'. Commands from these behaviours are then fused, through a weighted summation and/or switching strategies, and sent to the actuators. Note the difference in structure to that of a classical deliberative system, depicted in Figure 1.1.

day, nearly every autonomous mobile robot navigation system has a reactive component at some level.

A growing body of research has extended the insect and animal inspired navigation approaches to beyond the level of taxis[1] and obstacle avoidance. An example of a biologically-inspired development is visual homing in which discrepancies between target and current views of the workspace drive the agent toward the target location, without resorting to complex metric representations. Visual homing has a finite 'catchment area' limited by the range of view of the sensor used — it is a 'local' navigation technique. However, visual homing can be used as a tool for larger scale navigation through the use of topological maps. A topological map is a network of nodes and links, in which the nodes represent distinctive locales and the links represent the interconnections between distinctive locales; see Figure 1.4 for an example. Such maps provide a means of reducing

[1] In biology, taxis is the reaction of a free organism to an external stimulus by movement in a particular direction.

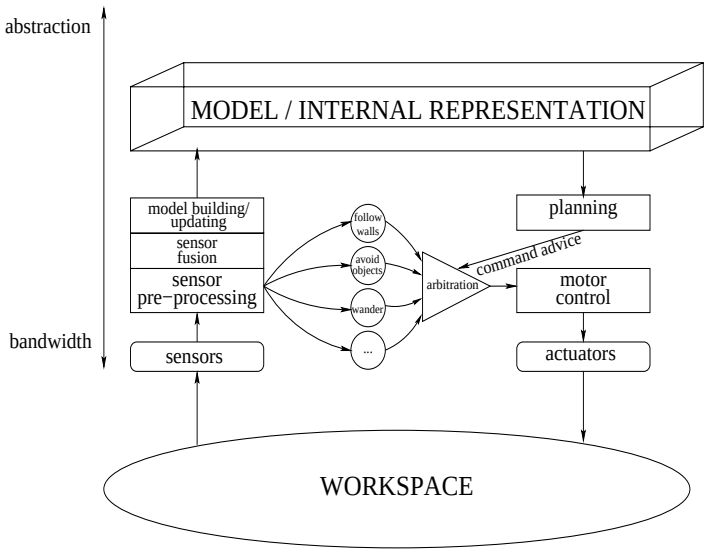


Figure 1.3: An example of a hybrid architecture. World modelling is used to influence the behaviour of the reactive control system, whether it be through selecting particular behaviours or by providing 'hints' or advice to the reactive level.

the computational overheads of navigation, while still providing adequate representational knowledge to facilitate purposive navigation.

However, as with many classical approaches, many of the biologically-inspired navigation methods have been implemented on mobile robots which are not representative of existing vehicles in the civil and industrial domains. 'Real world' vehicles such as cars, trucks, and the articulated vehicles found in the civil and mining industries have significant constraints on the ways in which they can move compared to traditional laboratory mobile robots. In fact, from a control-theory point of view, the control of these 'real world' vehicles is a very challenging non-linear control problem.

A particularly difficult problem is that of pose stabilization, i.e. moving from an initial position and orientation in the workspace to a desired position and orientation. Parallel parking of a car is a perfect example of this problem. This problem can be attacked by either an open-loop, sense-model-plan-act strategy or by a closed-loop, reactive strategy. The open-loop strategies are again computationally intense and sensitive to disturbances in the sensed environment and the model of the robot. In contrast, the feedback strategies provide robustness against uncertainties in the sensed environment and the mobile robot

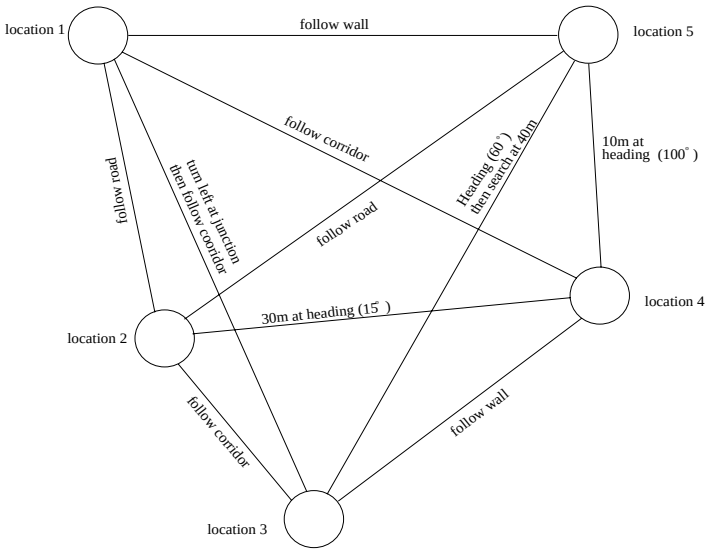


Figure 1.4: An example topological map. The nodes represent distinctive places, which are defined by salient features in the environment. Each node, or distinctive place, is connected to other nodes by the links which contain information linking the two nodes (e.g. wall-following).

itself. However, in the control of these vehicles, it is usually assumed that the full robot state is available from proprioceptive and exteroceptive sensors [Conticelli *et al.*, 1999], that is, reconstructed from models of the environment and the vehicle's motion. This state estimate can be corrupted by wheel-slip, errors in the vehicle model and noise. An alternative is to close the feedback loop at the sensor level, a candidate sensor being computer vision.

Vision is one of the most widespread and powerful sensors used in nature, where it serves for both high-level cognition and low-level motor control. Biological vision systems have a wide field of view and the ability to focus attention on 'interesting' objects in the environment. When combined with the neural and visual 'short-cuts' evolved over millions of years, biological vision systems are astonishingly effective.

From the earliest days of robotics, vision has also been a prominent sensor. However, vision has traditionally been used in a sense-model-plan-act manner, reconstructing a scene into a metric model from which navigation proceeds. Because of the huge amount of data provided by vision, this re-constructional approach places a huge demand on com-

putational resources. Thus, vision has largely been restricted to the open-loop control dictated by the hierarchical, sense-model-plan-act structure of many existing navigation systems. Similar paradigms were encountered with robot manipulators. In manipulators, the development of visual servoing techniques, in which vision is used as a feedback sensor [Corke, 1996; 1994], has proved to be a powerful technique for avoiding the sense-model-plan-act process. The work in this thesis uses similar processes for the *local* control of a car-like vehicle.

The advent of the omnidirectional camera has provided a means of mimicking the wide field of view found in many biological systems, and thus would seem to be an excellent sensor for the control of a mobile robot. In fact, omnidirectional cameras have been used for some time, usually in the context of visual homing. Again, these robots have generally had much simpler kinematics than those of a car-like vehicle.

1.3 Research questions and methodology

This thesis proposes and demonstrates the use of omnidirectional vision as the primary sensor for the closed-loop control of a car-like vehicle. It investigates insect-inspired visual strategies and the seemingly disparate field of nonholonomic pose control strategies. The primary contribution of this thesis is the integration of these two strategies allowing the pose stabilization of a car-like vehicle using omnidirectional visual feedback.

The central topic addressed in this thesis is:

> *Can omnidirectional visual feedback be used as the primary sensor for the pose stabilization of a nonholonomic, car-like vehicle?*

This central topic gives rise to the following questions, which are addressed in this thesis:

1. What are the relative strengths and weaknesses of the visual homing techniques and how can they be adapted for a car-like vehicle?

2. Can current vision techniques (feature extraction, blob tracking, feature correspondence, optic flow etc.) be adapted and developed for an omnidirectional camera?

3. Can the vehicle's motion model be used to assist omnidirectional camera feature tracking?

4. Can the vehicle be stabilized to a position with no constraint on its final orientation, in the spirit of the visual homing literature?

5. Which of the pose stabilization techniques can accommodate 'real world' vehicle kinematics, constraints, dynamics, and sensor limitations?

These questions give rise to an additional thesis objective:

> To develop a test platform representative of industrial vehicles[2], enabling testing of navigation and control methods for this and future land navigation research.

1.3.1 Scope of work

The scope of the problem addressed is limited to:

- No metric map — the robot finds the target location based upon discrepancies between a pre-stored view of the target location and the current view.
- The primary sensor used is an omnidirectional camera.
- The workspace is to be similar to an industrial or building site comprising of a ground-plane and approximately planar vertical surfaces.

1.4 The test-bed

The test-bed is a Toro ride-on mower, which has been fitted with the relevant actuators, microprocessors, sensors and computer hardware allowing for the testing of autonomous navigation techniques. The design and construction of this vehicle formed a large proportion of the initial stages of this research and as such, the vehicle is comprehensively described in Chapter 3.

[2] In this context, 'representative' means having similar motion constraints and significant dynamics in the steering and speed control loops.

1.5 Contributions

The contributions of this research include:

- Design and construction of an automated a ride-on mower which serves as an experimental platform representative of many existing industrial vehicles. This vehicle will also serve as a testbed for much future land navigation research at the CSIRO.
- Identification and characterization of the dynamic response of the vehicle's steering and velocity loops. Subsequently developed an accurate model of the vehicle using the bicycle model for the kinematics, and the experimentally determined models of the vehicle's response to control demands. The model also includes non-linear effects such as input saturation and rate-limiting.
- Simulated several different landmark-based homing strategies illustrating their strong dependence on landmark configuration. Developed a simple homing technique based upon a *visual servoing* framework.
- Extended the landmark-based ALV homing strategy to include range information, eliminating the landmark configuration dependence. Illustrated and characterized the sensitivity of the method to sensing errors.
- Developed a robust vision system to track a set of coloured landmarks in a workspace, using an omnidirectional camera.
  - Applied colour-based object segmentation techniques based upon a pre-taught two-dimensional colour profile of the target object colour.
  - Presented and validated a model for range estimation based on the optics of a catodioptric omnidirectional camera with an equiangular mirror, and using the 'flat-Earth' assumption.
  - Developed robust landmark tracking techniques based upon the vehicle kinematics and vehicle-object motion models.

  - Applied complementary filtering methods which combined data from the disparate sensors, improving the quality of the measurements.

- Developed a position control technique which allows a vehicle to servo precisely to a pre-learnt position based on the discrepancies between the vehicle's current view of the world and that at the pre-learnt location.

- Simulated and developed several pose control techniques for a car-like vehicle illustrating the negative effects that 'real world' characteristics such as saturated inputs have on many existing methods.

- Developed, simulated and experimentally tested switching control laws for the pose stabilization of a car-like vehicle. These laws are robust to significant velocity and steering loop dynamics, and input saturation.

- Demonstrated vision-based, state feedback position and pose stabilization of a car-like vehicle.

The research in this thesis is distinguished from other similar work in the following ways:

- The experimental vehicle is significantly larger than most other vehicles in the non-holonomic pose stabilization literature. Because of its larger size, the vehicle requires more space to execute the required motions.

- This work explicitly considers the dynamics of the vehicle's steering and velocity loops, and other factors such as input saturation. In general, larger vehicles are less responsive than smaller ones, and the slower dynamics have a significant influence on the success of pose stabilization methods.

- The larger vehicle motions place further demands on the vision system which is used to track landmarks and servo to the pre-learnt pose. Furthermore, this work demonstrates vision-based pose stabilization in an outdoor environment.

- This work is the first, to the author's knowledge, to use omnidirectional vision for the pose stabilization problem. The vision-based pose stabilization literature normally uses monocular cameras for which the additional constraint of keeping landmarks in the camera field of view must also be accommodated in the control design. In addition, most of this literature considers unicycles rather than car-like vehicles.

1.6 Structure of the thesis

The thesis aims to unite and develop results from control theory, mobile robotics (both the traditional and biologically-inspired literature) and computer vision. The work is structured as follows:

- **Chapter 2** begins with a comprehensive introduction to mobile robotics, giving a sense of the history of the field and the major obstacles to be overcome. Navigation of mobile robots in general is then addressed from which vision-based pose stabilization is identified as an interesting, and potentially useful, area of research. This is followed by a comprehensive review of the pose stabilization literature for nonholonomic systems. In particular, the vision-based pose stabilization literature is addressed, alongside the biologically-inspired approach to a similar problem — visual homing.

- **Chapter 3** presents the development of the test platform for the research. This includes descriptions of the vehicle's control system and the software used to interact with the vehicle. Also included is the identification and development of realistic models which accurately describe the system. These models have been extensively used in the thesis, expediting the selection, development and testing of vehicle control algorithms

- **Chapter 4** presents the sensing strategy used for the vehicle control. In this chapter, several landmark-based homing strategies are presented and compared, along

with a planar image-based visual servoing technique. These strategies are then demonstrated to have a significant dependence on landmark configuration — the introduction of landmark range is demonstrated to eliminate this dependence. A correspondence-free, landmark configuration independent homing method is then developed, and a method of using this technique for pose estimation described. The sensors and sensor processing used to implement the strategy, using vision and a magnetic compass, are then presented. Methods of isolating landmarks in a colour omnidirectional image are then described, and techniques for estimating landmark range are presented. In considering the sensing, complementary filtering techniques are developed which improve the quality of the desired measurements by combining vision and magnetometer data with vehicle odometry. Finally, errors associated with the sensing and sensing strategy are identified and characterized.

- **Chapter 5** presents the main contribution of the thesis in which control algorithms for position and pose stabilization are presented. The position controller relies on a discrete event supervisor to switch between a set of two control laws, bringing the vehicle to the origin with no constraint on its final orientation — this corresponds to the visual homing problem addressed in the biologically-inspired mobile robot literature. The problem of pose stabilization is then addressed. First, many existing algorithms are demonstrated to fail with the inclusion of characteristics found on real vehicles, such as dynamics and non-linearities. A controller which can cope with these effects is then developed. Like the position controller developed earlier in the chapter, the pose controller also relies on a discrete event supervisor, this time switching between a set of three control laws. This controller drives the vehicle to the origin *with the additional constraint on the vehicle's orientation*. Both controllers are extensively tested in simulation and implemented on the vehicle using the vision system for feedback.

- **Chapter 6** Summarises the work and presents the concluding remarks and directions for future research.

Chapter 2

Mobile Robot Navigation

Navigation is distinguished from other spatial behaviours, such as obstacle avoidance, by the requirement for attaining a goal. There is a hierarchy of navigation competencies that enable a goal to be reached. These competencies can be broadly categorised as local *and* global*, with the difference being that* global *navigation requires representation of 'things' that may be outside the current range of perception. The majority of mobile robot research has focused on the highest level in the navigation hierarchy, survey navigation, a* global *competency, which embeds spatial information pertaining to the environment in a common coordinate frame. This leads to sense-model-plan-act architectures using metric models of the environment, which to date have been infeasible for large-scale environments. In effect, the hardest task in the navigation hierarchy has been attempted first, without mastery of the lower level navigation competencies. Of the* local *navigation techniques, the attainment of a goal based on the spatial arrangement of surrounding objects is a fundamental competency if mobile robots are to perform purposive tasks. For car-like, and other nonholonomic vehicles, this requires fundamentally nonlinear approaches.*

2.1 Navigation

Before the age of accurate maps and charts, ship navigation was only possible through the accumulated knowledge of experienced sailors. These sailors, and their sailing notes known as *rutter's*, were highly sought after by the then dominant powers of the sea, Eng-

land, Spain, Portugal, and the United Dutch Provinces [Dash, 2002]. Their knowledge included that of the prevailing currents and winds for particular areas, known landmarks and hazards, expected journey times, and other characteristics such as what bird-life to expect. This knowledge was very closely guarded and often considered state secret.

With the advent of accurate maps and charts, and a method of determining longitude, ship navigation became more of a science than an art. Today, the Oxford English Dictionary defines navigation as:

> The art or science of directing the movements of ships on the sea, including more especially, the methods of determining a ship's position and course by the principles of geometry and nautical astronomy.

To do this, a ship's navigator performs three recurring tasks [Franz and Mallot, 2000]:

1. Determine the ship's position on a chart or map;
2. Relate this plotted position to the destination, reference points or hazards;
3. Based on this information, set the course of the ship.

Classically, mobile robot research has adopted a similar stance with many papers in the field concerned with the questions [Hill and Wilfong, 1990; Levitt and Lawnton, 1990; McKerrow, 1991; Leonard and Durrant-Whyte, 1991]:

1. Where am I?
2. Where are other things?
3. How do I get to other places?

This view of navigation clearly has its roots in the nautical approach, decomposing the problem into a logical set of subproblems. The classical answer to these questions relies on complete and accurate maps of the robot's operating environment and the robot's position in it. These methods are variously known as metric [Thrun, 1998b; Trullier *et*

al., 1997], absolute [Roberts *et al.*, 2000], or survey navigation [Franz and Mallot, 2000]. Such methods typically use a hierarchical, sense-model-plan-act architecture in which sensor data is sequentially abstracted to build or update a single global representation of the environment from which a plan is formulated and executed. Thus, control of the robot's actions is largely an open-loop process, relying on the accuracy of the world model and the estimation of the robot's state in the 'world'.

More and more mobile robotics researchers are discarding the nautically derived 'absolute' view of the navigation problem in favour of a broader view which includes all of the processes required to move a robot from one point to another, including what have traditionally been seen as lower-level tasks such as the guidance and control of the robot. In fact, the Institute of Electronics and Electrical Engineers defines navigation as *the process of directing a vehicle so as to reach the intended destination* [IEEE, 1983]. This definition does not imply a need to answer all, or even any, of the questions posed by the absolute approach. Observations of navigation in biology reveal a similar philosophy, with the most important question for a navigating organism being 'How do I reach the goal?' [Franz and Mallot, 2000]. This has lead to the development of mobile robot navigation systems which operate relative to the environment, referring to comparatively 'sparse' environmental representations, rather than complex metric models.

This chapter is organised as follows: Section 2.2 presents a brief history of mobile robotics; Section 2.3 briefly describes typical architectures for the navigation and control of mobile robots; Section 2.4 briefly summarizes the sensors available to mobile robots, outlining the relative strengths and weaknesses of different sensors; Section 2.5 presents a hierarchy of navigation competencies or behaviours; Section 2.6 introduces nonholonomic systems and reviews the literature on the topic of pose stabilization for mobile robots; and Section 2.7 completes the chapter with some concluding remarks.

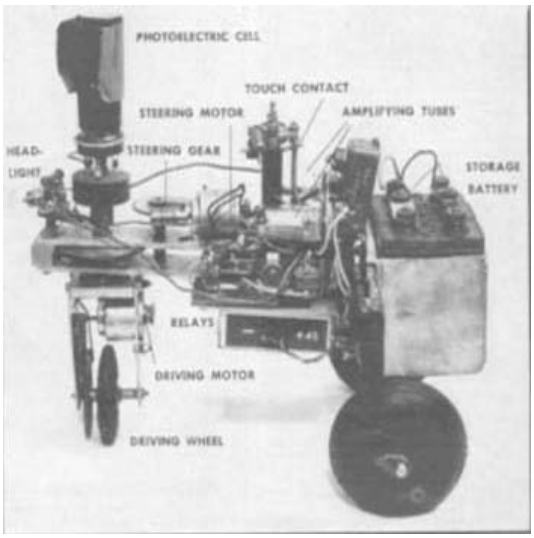


Figure 2.1: Elsie, one of the earliest autonomous robots. Elsie was a three wheeled, front-wheel steered and driven device equipped with a light sensor and a proximity sensor. Two vacuum tubes provided for control of the robot's actuators. Energy was supplied by a telephone DC battery. Image courtesy of http://epub.org.br/cm/n09/historia/documentos_i.htm.

2.2 A brief history of mobile robotics

The earliest autonomous mobile robot is attributed to W. Grey Walter who, in 1948, built a robotic 'tortoise' he called 'Elsie'. Elsie was capable of avoiding large obstacles and finding a re-charging station signalled by the presence of a light [Walter, 1953]. Dr. Grey Walter was a neuro-physiologist, interested in reflex actions and complex behaviour arising from neural connections [Holland, 1997]. Elsie, see Figure 2.1, provided the foundations for the more modern field of behaviour-based mobile robotics [Arkin, 1999].

The early 1960s saw further development of mobile robotics with the creation of the Hopkins 'beast', see Figure 2.2, by the Johns Hopkins University Applied Physics Laboratory. It was equipped with sonar which enabled it to wander corridors, and was able to feel walls for a re-charging socket when its batteries were low. Later modifications included the addition of a photo-cell that enabled the 'beast' to *see* re-charging sockets by means of the contrast between the colour of the socket and a wall.

Shakey, see Figure 2.3, created in the late 1960s at the Stanford Research Institute, saw

Figure 2.2: Two versions of the Hopkins 'beast', the larger of which is re-charging at the wall socket. Image courtesy of the Robotics Institute at Carnegie Mellon University http://frc.ri.cmu.edu/~hpm/book97/index.html.

a move towards robots capable of more deliberative action — the birth of what is termed in this thesis the absolute, deliberative approach to mobile robot navigation. Shakey could plan using a symbolic world model, with computer vision providing information to maintain and update the world model's representations. This robot was very large, while the controlling computer filled a room. It took several hours to plan and achieve the nominal task of finding and moving a coloured block using vision to find the block and a laser scanner to determine its range. Encoders on the locomotion motors allowed the robot to estimate its position in the world with respect to a global coordinate system [Nillson, 1969].

Also using the absolute, deliberative approach, the Stanford Cart, built in the late 1970s at the Stanford University Artificial Intelligence Laboratory, used stereo vision to locate objects in three dimensions. Using a model built from this information, it could plan an obstacle-avoiding path to a goal. However, it was only reliable for short runs and took upwards of fifteen minutes to move a distance of one metre [Moravec, 1990]. Morovac later moved on to build the Carnegie Mellon University (CMU) Rover, and, with Elfes [Moravec and Elfes, 1985], pioneered the occupancy grid-based methods of

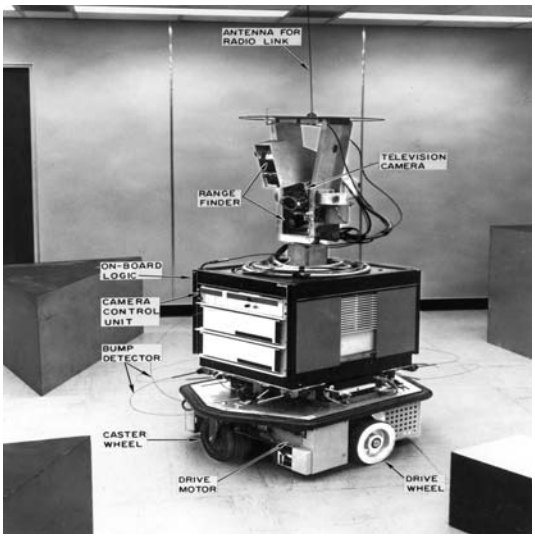


Figure 2.3: The Stanford Research Institute's Shakey. This robot was one of the first to use what is termed the absolute navigation method. Image courtesy of the Robotics Institute at Carnegie Mellon University http://www.frc.r.cmu.edu/'hp,/book98/fig.ch2/p027.html.

navigation.

In 1977, HILARE was created at the Laboratoire d'Automatique et d'Analyse des Systèmes (LAAS) in France, see Figure 2.4. It was a highly successful robot still in use over a decade later. This robot continued with the absolute, deliberative approach, with planning conducted in a multi-level representational space consisting of geometric models representing distances and measurements of its world (an office-like environment), with a relational model expressing the connectivity of rooms and corridors [Giralt *et al.*, 1979]. This relational model is an early instance of a topological map.

In 1984, Braitenberg re-ignited interest in behaviour-based robotics with a series of thought experiments designed to explore the seemingly complex behaviours that could emerge from a simple set of tight sensor-motor couplings [Braitenberg, 1984]. Shortly later in 1986, Brooks also took inspiration from biology, creating his *subsumption architecture* [Brooks, 1986]. This architecture was implemented on many subsequent robots with a great deal of success, outperforming the best that classical mobile robotics could offer at the time, culminating in Mataric´'s robot Toto, which was able to construct maps and plan within a behaviour-based architecture [Mataric´, 1990; 1992].

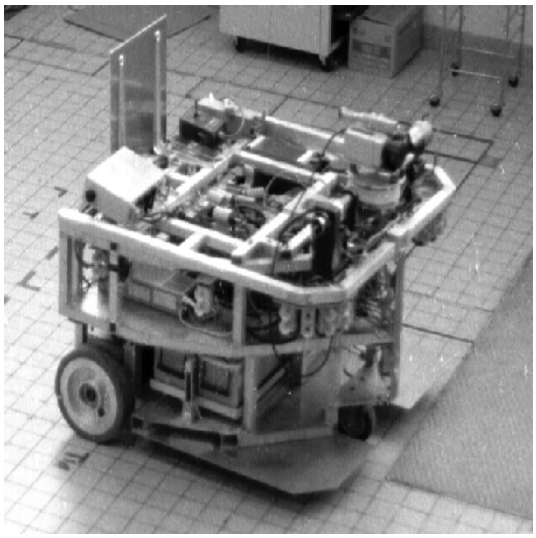


Figure 2.4: HILARE, an early robot built at LAAS. It has a laser range finder, sixteen ultrasonic sensors, odometry, four on-board computers and weighs 400kg. Image courtesy of LAAS at http://www.laas.fr/~matthieu/robots/hilare. The LAAS Internet web site reports that the robot now supports a table.

Arkin's motor-schema behaviour-based architecture soon followed [Arkin, 1987], with a recognition shortly after by many researchers for a need to unite the behaviour-based and the classical, deliberative approaches [Arkin, 1989b; Gat, 1998; Jarvis, 1997]. Today, few robots are built without consideration of both behaviour-based and deliberative control.

2.3 Typical architectures

The brief history of mobile robotics presented in Section 2.2 highlighted several different navigation architectures. In this section, the deliberative and behaviour-based approaches are explored more fully in terms of their relative strengths and weaknesses and the desirability of uniting the two in a 'hybrid' architecture.

2.3.1 Deliberative / Hierarchical

The typical control / navigation architecture for a system based on the absolute, deliberative approach relies on the sequential processing of data resulting in a hierarchical

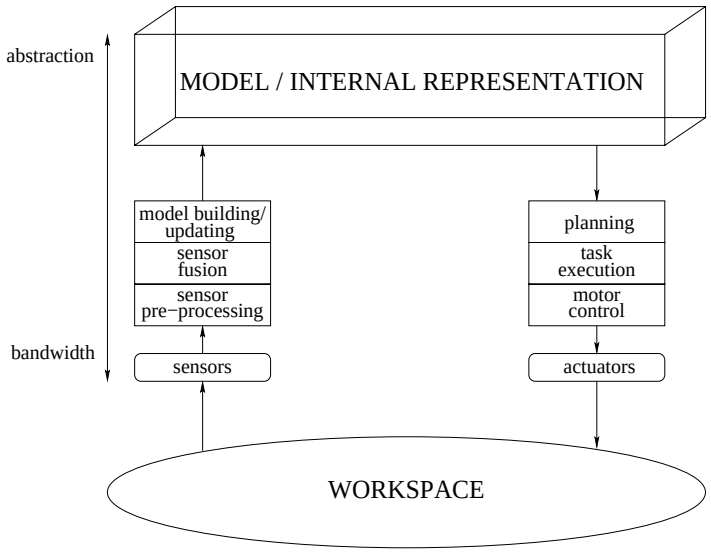


Figure 2.5: A typical architecture for a classical navigation system. Sensor data is passed up through the system, updating world models and the position estimate of the robot. In combination with a mission specification, the world model is then used by the path planning module, which compares estimated position to desired position. A series of tasks is then specified and the task execution module sends information to the motor control module which in turn sends commands to the robot's actuators (diagram adapted from [Corke, 1994] and [Nehmzow, 2000]).

architecture [Brooks, 1986; Nehmzow, 2000], as shown in Figure 2.5. The sense-model-plan-act cycle is realised through the deliberative architecture. No action is taken by the system without an analysis of its consequences. Central to the system is a model of the environment, usually in the form of a metric map.

Sensor data is fed through the system, updating and sometimes building the map (also called the model), while estimating the robot's position on the map. Position is notoriously difficult to track using odometry (see e.g. [Borenstein *et al.*, 1996; 1997; Everett, 1995] for a detailed explanation of sensors and their associated problems) so data from different sensing modalities is often fused using a Kalman filter or similar techniques (see e.g. [Smith and Cheeseman, 1986; Nebot and Durrant-Whyte, 1999; Dissanayake *et al.*, 1999; Adam *et al.*, 1999; Aono *et al.*, 1998; Chung *et al.*, 2001]). The path planning module uses the world model and the mission specification to determine a path from the start to the goal location by performing a search which produces a least

cost path. Least cost paths can be defined in terms of shortest distance, minimum energy usage, maximum distance from obstacles, or any combination of relevant measures. One of the functions of the task execution module is to maintain a given path by comparing the robot's estimated position with the desired position. This and any other required tasks are processed into actuator commands by the motor control module. The robot itself and the actuators must be modelled, allowing prediction of the consequences and interactions of the proposed actions.

Throughout the deliberative process, the robot is responding to the estimated model of the environment, not directly to what it perceives. A robot using deliberative techniques requires relatively complete knowledge of the world, using this to analyse the outcomes of its actions. This allows it to optimise its performance, and in terms of navigation, generally produces least-cost or optimal paths. However, if the information used for planning, i.e. the model, is incorrect, then the robot's actions may also be incorrect for the task at hand. This is why sensor data is used to continuously update the model of the world and state of the robot [Nehmzow, 2000; Arkin, 1999].

Deliberative / hierarchical systems are well suited to highly structured and predictable environments in which the outcome of actions is predictable. However, unstructured and uncertain environments present problems particularly when it comes to world modelling. It is difficult to track and model dynamic objects, reason about their behaviour and produce acceptable control commands in real-time[1] [Nehmzow, 2000; Arkin, 1999]. In some cases, sensor data may not be processed fast enough for the actuators to respond due to the bottleneck in the system caused by the need for world modelling and state estimation. This effectively leaves the robot 'blind' to the real world.

2.3.2 Behaviour-based / Reactive

In contrast to the deliberative architecture, behaviour-based systems have a much tighter coupling between sensing and action [Brooks, 1986; Arkin, 1999; Nehmzow, 2000].

[1] Real-time operation refers to a level of computer responsiveness that a user senses as sufficiently immediate or that enables the system to keep up with some external process.

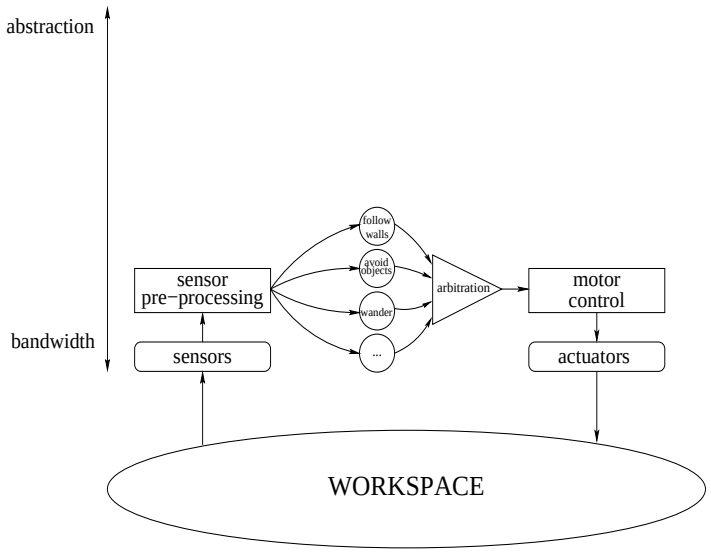


Figure 2.6: In a behaviour-based system there is a tighter coupling between sensors and actuators. Sensor information is processed in parallel by a selection of different controllers or 'behaviours'. Commands from these behaviours are then fused, through a weighted summation and/or switching strategies, and sent to the actuators. Note the difference in structure to that of the classical, deliberative system depicted in Figure 2.5.

Behaviour-based navigation systems are structurally different to their deliberative counterparts — behaviour-based systems process information in parallel rather than sequentially. Figure 2.6 shows a typical behaviour-based architecture. Note the difference in information flow to that of the deliberative system of Figure 2.5.

Sensor data is distributed to individual, parallel modules, each of which performs a specific task, for example 'avoid obstacles' or 'follow walls'. In a sense, each behaviour is competing to control the actuators. Hence, there must be a mechanism for arbitrating the commands. In Brooks' [1986] subsumption architecture, higher level behaviours could examine data from lower levels and either inhibit or subsume signals from the lower level layers. The command arbitration module in Figure 2.6 is shown merely to illustrate that commands from different behaviours must be combined in some manner, whether it be subsumption-based, a voting scheme, or some form of command weighting [Arkin, 1999].

The difference between an individual 'behaviour' and a traditional control-loop, whether it be open or closed-loop, is slim and in many cases the terms could be used interchange-

ably. Arkin [1999] defines an individual behaviour as:

> a stimulus/response pair for a given environmental setting that is modulated by attention and determined by intention.

Attention focuses sensory resources and prioritises tasks based upon current environmental context. Intention determines which set of behaviours should be active for the robot to achieve its goals and objectives.

Purely reactive systems use no explicit abstract model of the world, instead referring to the environment directly for information [Brooks, 1986; Arkin, 1999; Brooks and Flynn, 1989; Brooks, 1990]. In the words of Brooks and Flynn [1989]

> The world really is a rather good model of itself.

This is particularly important in unknown or uncertain environments as the robot can react directly to stimulus rather than performing the time consuming and error-prone process of building an abstract model and reasoning about it [Arkin, 1999]. In particular, reactive systems have a much higher bandwidth because of the lack of abstraction used. Actuators can more readily respond to sensors without the intervening stage of world modelling and state estimation. Because of this tight link, behaviour-based systems are capable of operating in real-time with vastly less computational ability than required for hierarchical systems.

The design of behaviour-based systems is a modular process; behaviours can be added as desired without necessarily redesigning or discarding the existing system. An advantage of purely behaviour-based systems is robustness — failure of a single or even multiple behaviours leads to a degradation in performance rather than total robot failure. Behaviours can simply be built on top of each other, gradually extending the robots capability.

Because of this ‘system modularity’, it has been claimed that these systems are easier to design and debug. However, researchers have found that the behaviour interactions are hard to predict and control. Planetary rovers were seen as a potential application for purely

behaviour-based systems. Desai and Miller [1992] showed that a sensor impoverished system could perform useful tasks if it was designed for its environment. However their work served to highlight a key limitation to purely reactive systems — taskability. As behaviours were built into the control structure, specification of a different task would require an almost complete redesign of the system.

Although successful in particular environments, purely reactive navigation has several limitations. Firstly, to achieve a different task or behaviour, the robot control system needs to be redesigned. Each level of behaviour is designed to respond to stimuli in particular ways: changing the task would require changing the (task-achieving) behaviours [Reignier *et al.*, 1997]. Secondly, it is very difficult to express plans in a behaviour-based architecture [Nehmzow, 2000]. If behaviours respond directly to stimuli, they are unable to respond to externally specified instructions. In short, purely reactive systems are not taskable, and it is difficult to express plans, as we know them, to such a system [Nehmzow, 2000; Gat, 1998; Arkin, 1995; 1999]. Also, the assumption of the world being its own best model only holds when it is sensed often and accurately enough. In the real-world, sensors are noisy and can be unreliable [Kuipers and Byun, 1991]. The performance ceiling of purely reactive navigation was quickly reached [Gat, 1998].

2.3.3 Hybrids

In response to the weaknesses of both the purely deliberative and behaviour-based architectures, many researchers saw the need to unite the two [Arkin, 1989b; Gat, 1998; Jarvis, 1997]. Many different forms of hybrid architectures exist, with the majority of systems incorporating reactive behaviours for low-level tasks such as obstacle avoidance. In general, there are three ways in which deliberative and reactive architectures can be tied [Lyons and Hendriks, 1992]:

1. Hierarchical integration of planning and reaction — A deliberative system is 'piggy-backed' onto a reactive system. Whether the system is *planning* or *reacting* depends on the situation at hand.

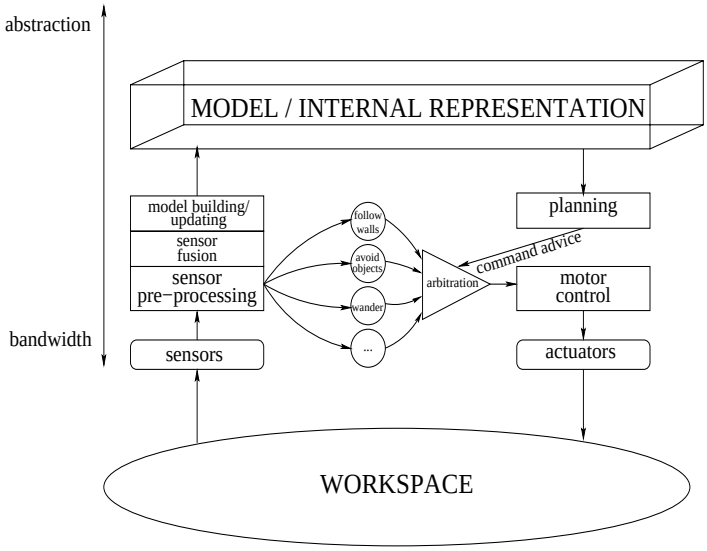


Figure 2.7: An example of a hybrid architecture. World modelling is used to influence the behaviour of the reactive control system, whether it be through selecting particular behaviours or by providing 'hints' or advice to the reactive level.

2. Planning to guide reaction — The deliberative layer configures and sets parameters for the reactive layer or provides 'advice'. Everything is executed by the reactive layer and the reactive layer may or may not follow the instructions or advice given by the deliberative layer.

3. Coupled planning and reaction — Deliberative and reactive layers are coupled, each guiding the other and acting concurrently.

An example architecture is shown in Figure 2.7. This architecture is of the 'planning to guide reaction' form, allowing the robot to respond to the environment directly while providing a means for representing knowledge of the world. The robot is free to act based upon its sensor readings, and is guided by the deliberative level of the architecture. Figure 2.7 represents just one of the various ways in which a deliberative and reactive architecture can be combined. Which element, deliberative or reactive, should dominate the architecture is still a subject of debate. Hybrid architectures can fuse the best of the deliberative and purely reactive approaches; on the other hand they can also fuse the worst of both methods if care is not taken [Arkin, 1989b].

2.3.4 Discussion

Today, virtually no mobile robot is built without considering both planning and reactivity. Methods of best combining planning and reactivity are on ongoing field of research which is not specifically addressed within this thesis. However, it is recognised that a measure of both planning and reactivity is essential for successful navigation systems.

To date, the incorporation of reactive elements in navigation architectures has normally been restricted to spatial behaviours in general, for example obstacle avoidance and wall-following, rather than navigation. Navigation is distinct from other spatial behaviours in that it requires the robot to reach, and recognise, some goal position or pose. One of the major objectives of this thesis is to 'close-the-loop' at a higher level in the navigation architecture allowing a car-like vehicle to attain a particular pose, based on visual feedback. This will allow a robot to achieve a goal pose without 'thinking' about it, freeing up resources for higher level functions.

2.4 Sensing

Fundamental to any navigation system is the ability to sense the state of the vehicle and its surrounds. Just as we have senses to give us information about the environment we inhabit, robots too require a means of perceiving the operating environment. Sensors fall into two categories: internal and external sensors. Internal sensors monitor the state of the robot itself, whereas external sensors provide data about the operating environment or the 'world'[2]. Within the category of external sensors, there are active and passive types. Active sensors project energy into the environment and interpret the return signal. For example, an ultrasonic sensor projects acoustic waves and interprets the time-of-flight for the return signal. Passive sensors are stimulated by the environment directly. For example, a colour camera receives information regarding the intensity and colour structure of the immediate environment which is of course dependent on lighting conditions.

[2]In the biological literature, these categories are known as *proprioceptive* and *exteroceptive* sensors respectively.

The type of sensors available to the designer of a mobile robot influence the control strategies available for autonomous navigation. Each type of sensor has inherent limitations and advantages that need to be compensated for and exploited respectively. Commonly, data from several different sensors is 'fused' to improve accuracy and reliability. However, this usually relies on an accurate model of the vehicle kinematics, dynamics, and sensor behaviour.

This section briefly discusses the different sensor modes, and how each relates to navigation. The discussion is not intended to be comprehensive, serving merely as an introduction to the types of sensors available and their relative strengths and weaknesses. For a comprehensive treatment of sensors for mobile robots refer to [Everett, 1995; Borenstein *et al.*, 1996; 1997].

2.4.1 Internal sensors

Internal sensors monitor the state of the robot, traditionally measuring such things as joint angles in robotic manipulators. In mobile robots, internal sensors are commonly used for path integration, also known as dead-reckoning or odometry. However, computer vision-based path integration has also been demonstrated [Chahl and Srinivasan, 1996; Srinivasan *et al.*, 1997; Mallet *et al.*, 2000].

Although dead-reckoning techniques are well researched, none can match some of the astonishing feats of biological systems. Take for example the desert ant *cataglyphis bicolor*, which forages at distances exceeding $200\,\mathrm{m}$ from its nest, returning to a nest opening of less than $5\,\mathrm{mm}$ [Wehner and Wehner, 1990]. An example of such a foray is given in Figure 2.8 (figure from [Franz and Mallot, 2000]). The high ground temperatures encountered exclude the use of odour trails and hence, it is hypothesised that this ant navigates through a combination of path integration and visual homing [Wehner and Wehner, 1990; Franz and Mallot, 2000; Collett and Collett, 2000]. Note, in the figure, the almost direct route back to the nest on conclusion of the out-bound section of the foraging journey. If we assume that path integration is the dominant behaviour in effect here, this ant's path integration system navigates with a drift rate of less than 0.0025% of distance travelled.

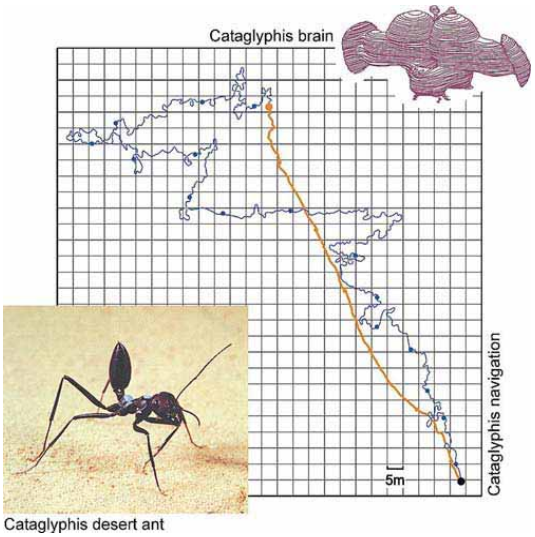


Figure 2.8: An example of the amazing navigation feats of the humble ant. Diagram courtesy www.neuroscience.unizh.ch/e/groups/wehner00.htm, Rudiger Wehner.

Compare this with currently available high end Inertial Measuring Units with drift rates of the order of 0.1% of distance travelled [Everett, 1995]. Of course, just as the ant can presumably compensate for accumulated errors in its odometry system based upon previously seen landmarks, so too the information from our man-made internal sensors can be corrected with an external reference.

Odometry and Dead-reckoning

In (wheeled) mobile robots, shaft encoders can be used to measure wheel rotations and the relative orientation of steering wheels (steering angle). Given a method of measuring wheel rotation, distance travelled and velocity can be calculated with knowledge of the wheel radius and the measuring time increment. Such measurements are highly prone to error due to wheel slip, variations in rolling wheel diameter (e.g. due to wear, pressure and temperature variations or inaccurate measurements), and to inaccuracies in wheel base measurement. The fundamental idea of odometry is the integration over time of incremental motion information, which leads to an unbounded accumulation of error [Borenstein *et al.*, 1997; Hague *et al.*, 2000]. These errors can be categorised as follows [Borenstein *et al.*, 1997]:

1. Systematic — result from kinematic imperfections of the robot and inexact measurements.

2. Non-systematic — result from interactions of the terrain with the wheels, e.g. wheel-slip.

Systematic errors can be accounted for, but not completely eliminated by, improved measurements of wheel diameters and a more accurate vehicle and sensor model. Non-systematic errors are unpredictable and cannot be corrected without resorting to another sensing modality — usually an external reference such as landmarks.

Odometric sensors such as shaft encoders are cheap and can provide a high sample rate. Although widely criticised, they have seen extensive use.

Inertial sensors

Inertial sensors measure either rate of rotation or acceleration from which position information can be derived. As for the odometric sensors, error is again unbounded due to the accumulated error from integrating velocity, or double integrating acceleration measurements [Borenstein *et al.*, 1997; Hague *et al.*, 2000]. Any small error in the sensor is magnified by the integration process.

Internal sensors are generally good for short periods or distances — much research has focussed on extending the period for which they are considered accurate. This is normally done by referring to an external sensor to correct accumulated error, also referred to as drift (see e.g. [Borenstein *et al.*, 1997; Hague *et al.*, 2000; Dissanayake *et al.*, 1999]).

2.4.2 External sensors

Examples of external sensors include magnetic compasses, global positioning systems (GPS), active beacons, vision, and range-finders (e.g. ultra-sonics, laser scanners, and radar). Generally, external sensors are much more reliable than internal sensors because they measure angle and distance directly rather than integrating from velocity and accel-

eration measurements. However, there are problems when accessing the external environment for perception including:

- line of sight requirements;
- interference due to occlusion and reflections, or, in the case of compasses, the presence of steel structures, buried pipes, cables and even certain rocks;
- perceptual aliasing.

Perceptual aliasing occurs when separate locations in an environment return similar sensor data; it can be difficult to distinguish one location from another [Nehmzow, 2000].

External sensors can measure an absolute quantity such as *magnetic North*, or they can measure a relative quantity like the *range* to another object. Within the category of external sensing, there are two sensing modes [Hebert, 2000]:

1. Active sensors — actively emit a beam of energy, measuring some quantity in the environment, usually range. Examples include ultra-sonics, laser range-finders and radar.
2. Passive sensors — receive information from the environment without interfering with it. Examples include computer vision and GPS.

External sensors usually have a lower update rate than internal sensors. However, the measurement is referenced to the environment, and hence is not subject to the accumulation of error experienced by internal sensors.

2.4.2.1 Active sensors

Active beacons

Active beacon systems use a set of beacons mounted at known locations in an environment. These beacons transmit signals that enable a robot equipped with a receiver to estimate its pose through trilateration[3] techniques [Borenstein *et al.*, 1997; Everett, 1995]. A

[3] In robotics, *trilateration* is the process of determining the vehicle's position based upon distance measurements to beacons at known positions. *Triangulation* is the process of determining position based upon

minimum of three beacons are required, at least three of which must be ‘visible’. Active beacon systems provide accurate position information with a minimum of processing. However, modification of the operating environment is not always possible and active beacon systems can be expensive to install and maintain.

Range-finders

To date the most commonly used range sensor for mobile robotics has been ultra-sonic range-finders. This is due to their relatively low cost through extensive development in the photographic industry. However, laser range-finders are becoming cheaper and finding ever increasing application due to their superior accuracy and range over ultra-sonics.

Range-finders work by transmitting an energy pulse and measuring either the time-of-flight or phase-shift of the beam on its return. Spurious measurements can occur in the presence of obscurants such as rain or dust.

2.4.2.2 Passive sensors

Orientation sensors

Error in orientation is the most insidious of positioning errors leading to large displacement errors over long traverses. Magnetic compasses provide a means of measuring absolute orientation and hence finding the orientation of a vehicle. However, magnetic compasses measure the Earth’s local magnetic field and are susceptible to fluctuations caused by man-made objects such as ferrous structures and the dynamically induced fields produced by electric motors. The magnetic field direction also varies over the Earth’s surface.

An alternative method for providing a reference direction is to use the polarisation pattern produced by the atmospheric scattering of sunlight [Lambrinos *et al.*, 1997; 2000]. This technique was derived from observations of insect behaviour [Rossel and Wehner, 1986]. Early work in this thesis research showed that a CCD camera with a polarising filter could be used to observe this pattern [Usher *et al.*, 2001]. First, a look-up table

angle measurements. Triangulation can also refer to to the process of determining the range to some object, based upon two angle measurements and the distance between the measurement locations, as in stereo vision.

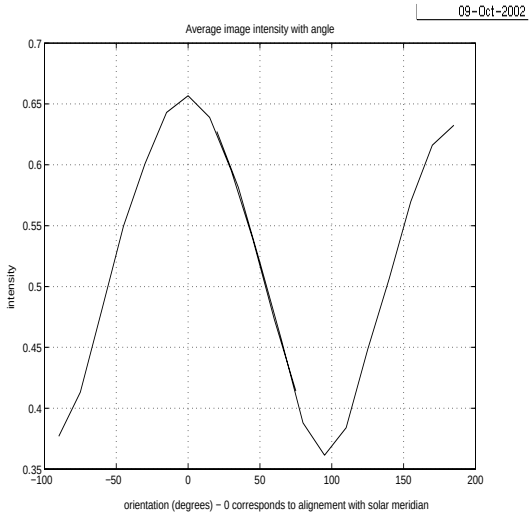


Figure 2.9: The variation of average light intensity with changes in the orientation of a polarising filter as measured by a CCD camera focussed on the sky. By comparing these values to a current average light intensity, an estimate of orientation can be determined.

of the average sky intensity at different orientations of the polarising filter is created, see Figure 2.9. Then, by comparing the average intensity of the polarised image at an arbitrary orientation to the look-up table of intensity values, the current orientation can be estimated. However, a look-up table is only valid for a relatively short period, and the method is also dependent on a significant amount of blue sky being visible. In addition, the symmetry of the polarisation pattern gives an ambiguity at 0° and 180° with respect to the solar meridian. This ambiguity leads to four possible solutions for orientation[4]. Lambrinos *et al.* [1997] overcame this ambiguity by using a set of three photo-diodes with polarising filters, with filter angles at 120° to each other. They present an analytical method of determining orientation based on these three measurements without the need for a look-up table. However, the method is still brittle with respect to lighting conditions and restricted to operating at certain times of day.

Motivated by the requirements of planetary rovers operating on other planets which may not have a magnetic field, NASA developed an orientation measuring system that

[4]The data in Figure 2.9 was collected for orientations from -90° to 180° and only shows three of the possible four solutions.

tracks the Sun throughout the day, providing a reference direction through knowledge of its motion [Trebi-Ollennu *et al.*, 2001]. In contrast to the experiments of [Lambrinos *et al.*, 1997; 2000; Usher *et al.*, 2001], the Sun was tracked with a CCD rather than using the polarising pattern of the sky. Although the system was shown to be reliable with a reasonable amount of cloud cover, excessive cloud cover is clearly a problem and of course such a system cannot function at night.

Global Positioning Systems

GPSs provide longitude, latitude and altitude position information through triangulation techniques using signals from a number of suitable satellites. Some units can also provide velocity and heading information (provided the GPS receiver is moving). Global positioning systems are becoming more common due to consumer demand in the automotive and maritime industries. The use of GPS in robots is also now more common but of course restricted to those operating in an outdoor environment.

The accuracy of GPS can be affected by [Everett, 1995]:

- the number of satellites available due to occlusions from buildings or a lack of satellites in the area;
- multi-path reflections;
- satellite position — geometric dilution of precision;
- ionospheric and tropospheric effects.

Another disadvantage of GPSs are their relatively slow update rates (generally of the order of $1\,\mathrm{Hz}$, although current high-end units can operate at up to $10\,\mathrm{Hz}$).

Differential GPS (dGPS) has an increased accuracy over that of standard GPS by correlating the error of a dGPS receiver with that of a signal received at a fixed reference station. However, the overall accuracy and reliability of GPS prevents it from being of significant use in the work presented in this thesis. Of significant influence in this decision

is the target operating environment, an industrial site, in which the presence of buildings leads to the potential loss of signal. For example, as of November 1999, less than 5% of the city of Tokyo, Japan, was covered by GPS [Chen, 1999]. Of course, this figure will improve with time but it is representative of the often overstated usefulness of GPS for precise positioning.

In addition, a GPS capable of the required positional precision (of the order of $10\,\mathrm{cm}$) is not within the budget of this project. However, in the latter stages of the project, an RTK-GPS[5] became available and has been used where possible for ground-truth purposes.

Computer vision

With few exceptions, all animals have eyes but, in general, have extremely complex and powerful parallel processors and dedicated low-level feature extraction 'circuits' which have evolved over millions of years to handle the complexity of visual information. For example, in humans, from the moment an image reaches the eye, low-level processes extract 'useful' information, such as the location of edges. This information, in parallel with a multitude of other extracted features and the image itself, is passed to the brain, where the information is used, for example, to construct internal representations of scenes, to identify and track objects, and, because we are highly social animals, to identify people. In fact, in biology, the eye has independently evolved in life forms ranging from octopi, insects and mammals — in each case, a large proportion of the brain is dedicated to vision processing and image understanding. Clearly, vision is a powerful skill for navigating and performing tasks in the animal kingdom and would seem to be a good choice of sensor for a mobile robot.

Vision arguably provides the most comprehensive information of all sensors available to mobile robots but because of its complexity and sensitivity to factors such as lighting conditions, it is difficult to use effectively. Many decades of research have given us some very effective vision systems but, by and large, these are still embryonic when compared

[5] Real Time Kinematic GPS which can give position information to a stated best accuracy of $2\,\mathrm{cm}$ at $10\,\mathrm{Hz}$.

to the the feats of biology.

Computer vision is in itself a stand-alone field of research but the links to applications in mobile robotics are clear. Recently, there has been much interest in omnidirectional camera systems (also known as panospheric) as they can provide a panoramic view of a scene. In general there are three ways to provide a panospheric image [Chahl and Srinivasan, 1997b]:

1. Rotate a conventional camera about a vertical axis and combine the resulting montage of images.

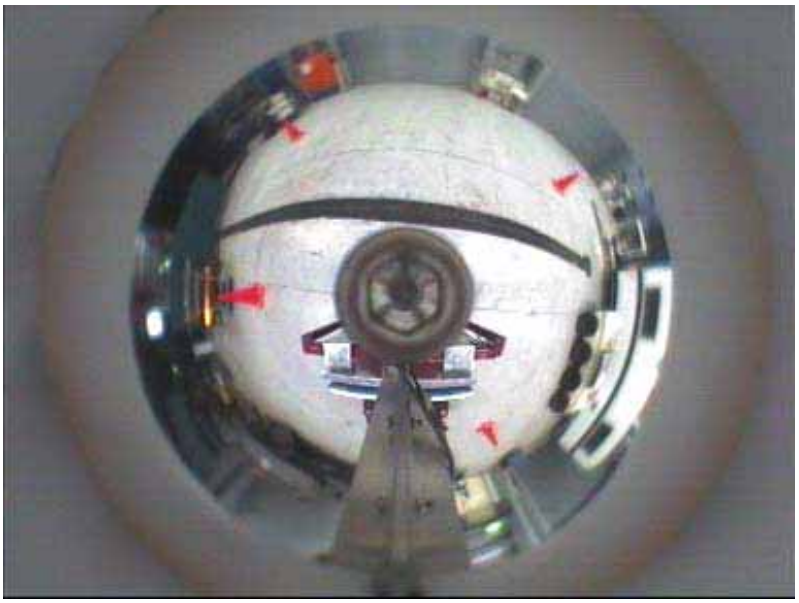
2. Fish-eye (wide-angle) lens systems which can provide a hemispherical view of a scene, although with some distortion.

3. Catadioptric systems which consist of a camera aimed at a mirror, where the mirror usually resembles a cone or sphere.

The first method requires significant post-processing. In addition, it is difficult to acquire the image set quickly, limiting the usefulness of the approach for dynamic applications [Chahl and Srinivasan, 1997b]. Fish-eye lens system are generally much more expensive because they are difficult to manufacture with the required accuracy. Poor control of lens tolerances can lead to significant image distortion. Even with a well manufactured lens, distortion away from the optical axis is very significant [Chahl and Srinivasan, 1997b]. Catadioptric systems are in general much cheaper because they are currently easier to manufacture. In addition, a panoramic image can be obtained from a single image, giving this method an advantage over rotating a conventional camera. Mirror shapes for such systems vary but are essentially a compromise between resolution, field of view, and elevational and radial gain (see, for example, Baker and Nayer [1999] for further explanation).

A subject which divides researchers in omnidirectional vision is whether the omnidirectional image should be ‘un-wrapped’ (or de-warped) to the panoramic view more easily interpreted by humans. The un-wrapping process consumes computational resources that

(a) A catadioptric omnidirectional system.



(b) A sample image.

Figure 2.10: An omnidirectional camera and a sample image.

could otherwise be dedicated to more useful tasks, as the image content is unchanged. In fact, the un-wrapping process often leads to a degradation in image quality but, on the other hand, it does allow the use of existing image processing techniques.

Likened to insect vision, omnidirectional cameras are proving to be a powerful tool for mobile robotics. An example is Srinivasan *et al.* [1997] use of an omnidirectional camera for dead reckoning using the image motion observed by the camera to calculate distance travelled. Others have used omnidirectional vision for visual homing [Hong *et al.*, 1991; Franz *et al.*, 1998a] and appearance-based navigation[6] [Krose *et al.*, 2001; Ulrich and Nourbakhsh, 2000; Krose *et al.*, 2000].

A key advantage of omnidirectional cameras over their monocular counterparts, is that landmarks, or targets, are always within the field of view of the camera, provided they are within range. This is particularly important for the problem of vision-based feedback pose stabilization addressed in this thesis. If a monocular camera is used, specific control mechanisms must be provided to ensure that landmarks remain in the field of view of the camera. When using an omnidirectional camera, these viewing constraints are relaxed.

2.4.3 Sensor fusion

Because of the somewhat complementary strengths and weaknesses between different sensing modalities, sensor data is often fused to reduce the uncertainty in the overall measurement [Smith *et al.*, 1990; Chatila and Laumond, 1985; Smith and Cheeseman, 1986]. In particular, dead-reckoning sensors are accurate over short distances and provide a relatively high update rate but their fundamental principle of operation involves a single or double integration leading to an unbounded accumulation of error. Reference-based sensors, e.g. landmark detection using a laser range-finder, generally have a slower update rate but are grounded in the environment and hence are not subject to drift. One method of combining sensor data is the Kalman filter or the Extended Kalman filter. Figure 2.11 shows the typical operation of a sensor data fusion loop. Dead-reckoning data is fed through the system and gives a prediction of the pose estimate based upon a kinematic model of the vehicle. This prediction is then compared with data from the reference-based sensor, generating an error signal which is used to compensate for any error in the

[6] Appearance-based navigation is a technique of localisation through matching of a currently viewed scene with a database of images of previously visited locations.

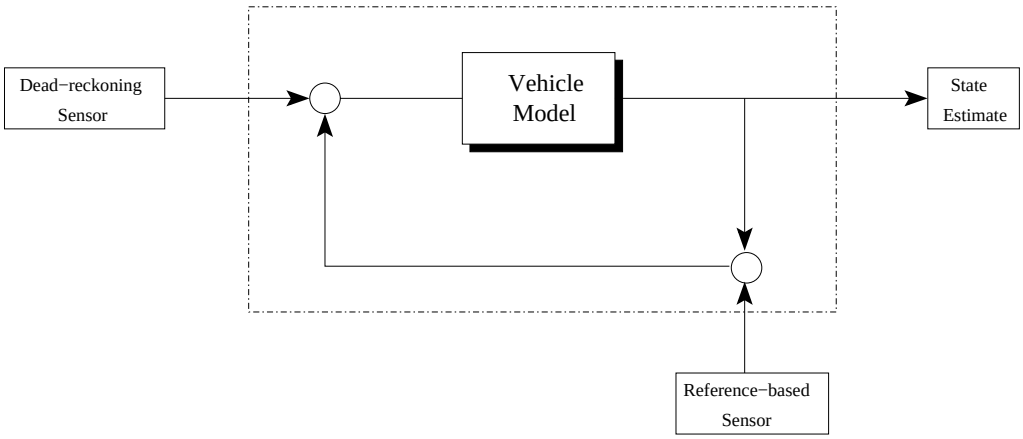


Figure 2.11: Outline of a sensor data fusion loop (adapted from [Nebot and Durrant-Whyte, 1999]).

dead-reckoning sensors.

Many different sensing modes have been combined, for example: computer vision and odometry[Adam *et al.*, 1999; Hague and Tillet, 1996]; GPS and dead reckoning[Aono *et al.*, 1998]; odometry and a high accuracy fibre-optic gyro[7] [Chung *et al.*, 2001]; omnidirectional camera and a map [Das *et al.*, 2001]; vision, odometry and a magnetic compass [Hague *et al.*, 1997]; and twin loops of GPS/IMU and millimetre wave radar and encoders [Nebot and Durrant-Whyte, 1999].

2.4.4 Discussion

All real sensors are affected by noise arising from inherent sources as well as from the environment. They suffer from trying to decompose a multi-dimensional environment into a lower dimensional representation. Much research over recent years has focussed on ways of best combining and fusing various sensor sources.

The sensors available for this project are wide ranging in terms of mode of operation, bandwidth and update rates. The primary sensor used will be a catadioptric omnidirectional camera, given that to date it has not been fully exploited for the pose stabilization problem. However visual homing also relies on a reference direction, for which a mag-

[7] Both dead-reckoning sensors.

netic compass will be used. All sensors are noisy — to reduce this noise, simple techniques will be presented based upon the concept of *complementary filtering*, leading to much smoother and hence more useful measurements. In addition, an RTK-GPS will be used where possible to provide a ground-truth *but serves no function in the control of the vehicle*.

2.5 Navigation competencies

The IEEE definition of navigation, that is, *the process of directing a vehicle so as to reach the intended destination*, captures a broader spectrum of behaviours than that implied by the classical, absolute approach. What sets navigation apart from other spatial behaviours (for example wandering or obstacle avoidance) is the ability to recognise a goal when it is reached [Franz and Mallot, 2000].

Navigation behaviours are divided into two fundamentally different categories: *local* and *global* navigation[8]. *Local* navigation behaviours require the recognition of a single location — the goal. The only required sensory perceptions are those that are in the immediate environment; there is no need for representation of 'things' outside of this 'sensory horizon' [Franz and Mallot, 2000; Trullier *et al.*, 1997]. In contrast, *global* navigation behaviours require the recognition of multiple goals, some of which may not be available in the current range of perception [Franz and Mallot, 2000; Trullier *et al.*, 1997]. A means of representing 'things' external to the immediate sensory horizon is a prerequisite for this range of behaviours.

Within *local* and *global* navigation there are a range of behaviours and competencies, which Franz and Mallot [2000] summarised into a hierarchy of navigation competencies; these are shown in Table 2.1. This hierarchy is based upon an original scheme presented by Trullier *et al.* [1997], modified and extended to more aptly apply to mobile robots. The first four entries in Table 2.1 correspond to *local* navigation behaviours. Within this group, each successive competency depends upon the behavioural prerequisites of prior

[8]Way-finding is the more popular term in the literature on biological navigation; in mobile robotics, global navigation is more common and is the term adopted here.

Table 2.1: The navigation hierarchy as presented by Franz and Mallot [2000]. The first four competencies do not require sensory perceptions, or representation, of 'things' outside of the immediate environment and are thus grouped as *local* navigation competencies. The last three competencies require representation of 'things' which may lie outside of the current sensory horizon and are thus grouped as *global* navigation competencies.

Competency	Behavioural prerequisite	Navigation competence
Search	Ability to move and recognize the goal	Finding the goal without goal direction knowledge
Instruction following[9]	Ability to follow course instructions	Finding the goal from one approach direction
Aiming	Ability to align goal direction with direction of motion	Finding a salient goal from a catchment area
Homing[10]	Ability to attain spatial relation to surrounding objects	Finding a goal defined by relation to the surroundings
Recognition-triggered response	Associate a set of stimuli to a set of responses	Following fixed routes
Topological navigation	Integration of fixed routes and planning	Flexible concatenation of route segments
Survey (metric) navigation	Embedding environment representation into a common reference frame	Finding paths over novel terrain

competencies. The last three entries belong to the *global* navigation category. Competencies within this group do not necessarily require all of the *local* navigation competencies [Franz and Mallot, 2000]. Each competency is briefly explained in the following sections, based upon the original presentation by Franz and Mallot.

2.5.1 Search

The *search* competency requires only the ability to move and the ability to recognise the goal. No knowledge of the goal direction is used for this behaviour, the agent simply wanders the environment until the goal location is recognised. Figure 2.12 illustrates such a

[9] Franz and Mallot refer to this as *direction following* in the sense of following a set of instructions. This term could be confused with direction in the compass sense and hence the term has been altered here to *instruction following* to avoid confusion.

[10] Franz and Mallot refer to this as guidance. Guidance in the robotics literature refers to trajectory generation, and the kinematics, dynamics and feedback control of a vehicle [Cox and Wilfong, 1990]. Hence, the term has been altered here to *homing* to avoid confusion.

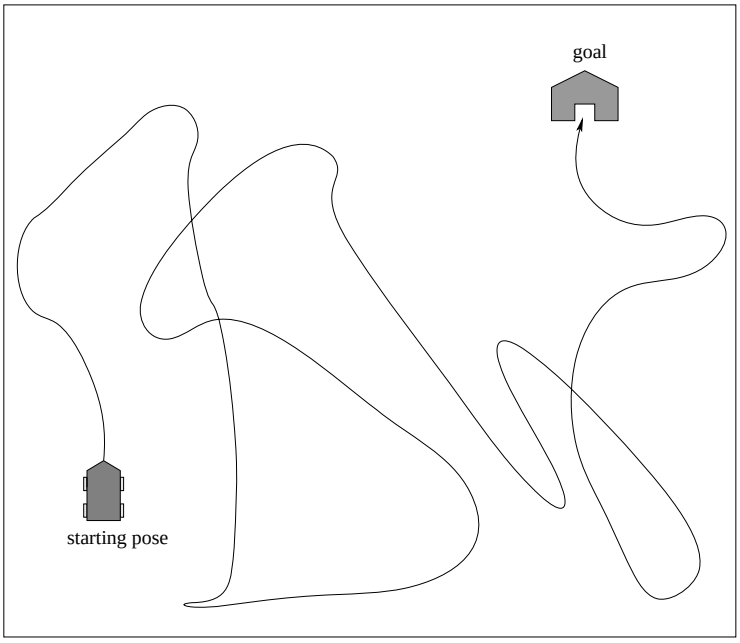


Figure 2.12: Illustration of the *search* competency (adapted from [Franz and Mallot, 2000]).

competency in which the agent searches the environment until the goal is attained. Shown here is an almost random pattern of motion but the search competency can use more sophisticated motion patterns to improve the chances of finding the goal, for example spiral motions.

In mobile robotics, this is not a popular method due to its inefficiency. The search behaviour is of little use but could perhaps be used if the robot became 'lost'. In contrast to a search behaviour, *exploration* requires the robot to wander the environment while it *does not* recognise a location. Wandering behaviours have been implemented on many mobile robots, enabling the robot to explore an environment while avoiding obstacles (see e.g. [Brooks, 1986; Yamauchi *et al.*, 1998; Mataric´, 1992; 1990; Donnart and Meyer, 1996; Diaz *et al.*, 2001]). With the current interest in Simultaneous Localisation and Mapping[11] (see e.g. [Smith *et al.*, 1990; Guivant *et al.*, 2000; Thrun *et al.*, 2000; Rencken, 1994; Bailey, 2002; Leonard *et al.*, 1992]), exploration competencies have become an integral part of many navigation systems.

[11] Also known as Concurrent Localisation and Mapping.

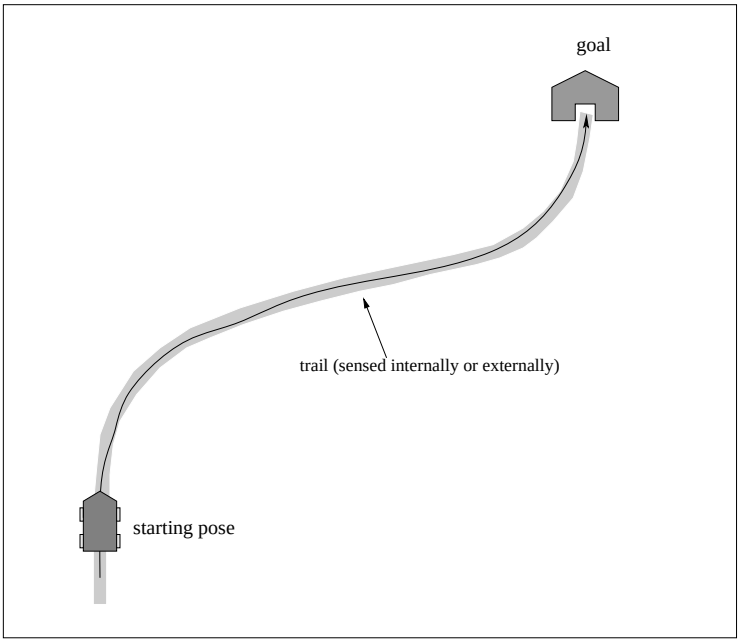


Figure 2.13: Illustration of the *instruction following* competency (adapted from [Franz and Mallot, 2000]).

2.5.2 Instruction following and path integration

This class of behaviour requires the agent to follow a set of instructions defined in terms of an internal or external reference, for example '*head North for 200 metres*'. The goal does not necessarily need to be visible for the entire journey, merely recognised when it is reached. Importantly, this competency only allows a goal to be approached from a single direction.

Internal references could take the form of odometry or inertial sensing which can be used for *path integration*, also known as dead-reckoning. In path integration, the distance and direction travelled are integrated to estimate position with reference to some starting position. The use of a compass direction, which is an external reference, can greatly improve path integration, vastly reducing errors in orientation. Other external references such as a wall or a road edge can also be used in a set of instructions for finding a goal. An example of the *instruction following* competency is shown in Figure 2.13, where the vehicle follows a a trail, which has some defining characteristic, leading to the goal. Even

for a vehicle with complex kinematics, *instruction following* in this context is relatively straightforward.

As far back as the 1950s, Autonomous Guided Vehicles have been operating in factories. Early versions followed a guide wire and were able to traverse fixed routes through an environment. In the 1970s this technology was extended to allow AGVs to follow buried guide wires (detected via magnetic inductance), and later, painted lines on the ground. The buried wires required significant modification to the environment but on the other hand were reliable and permanent. Painted lines allowed for greater flexibility and faster route generation but required higher maintenance to ensure reliability against wear and fading [Bailey, 2002]. Even later still, the installation of retro-reflective lines or lines of lights in mining tunnels has allowed Load-Haul-Dump vehicles to autonomously navigate along the fixed routes defined by the tunnels [Roberts *et al.*, 2000]. Similar systems exist for navigating along corridors in office settings.

Edge-following is another well developed technique with a great deal of research having been conducted using a variety of sensors including computer vision, ultrasonics, and laser range-finders. The main edges used have been those associated with walls and roads. Road-following techniques have enabled car-like vehicles to autonomously follow many thousands of kilometres across Europe and the United States of America (see e.g. [Pomerleau, 1993; Thorpe, 1990; Dickmanns *et al.*, 1994; Rodriguez *et al.*, 1998]). Others have used computer vision to track crop rows [Ollis, 1997; Hague *et al.*, 2000; 1997; Hague and Tillet, 1996; 2001] and laser range-finders in the reactive, wall-following navigation of a Load-Haul-Dump vehicle in a mining application [Roberts *et al.*, 2000].

Wall-following techniques using monocular computer vision have been developed for indoor applications [Horswill, 1993; Vassalo *et al.*, 2000; Kosecka, 1997] and wall-following using omnidirectional computer vision has been developed [Das *et al.*, 2001; Gaspar *et al.*, 2000; Winters *et al.*, 2000]. However, the work of Gaspar *et al.* [2000] and Winters *et al.* [2000] uses a vehicle which can spin on its own axis, and the work of Das *et al.* [2001] uses a toy remote control truck with computer processing performed off-board.

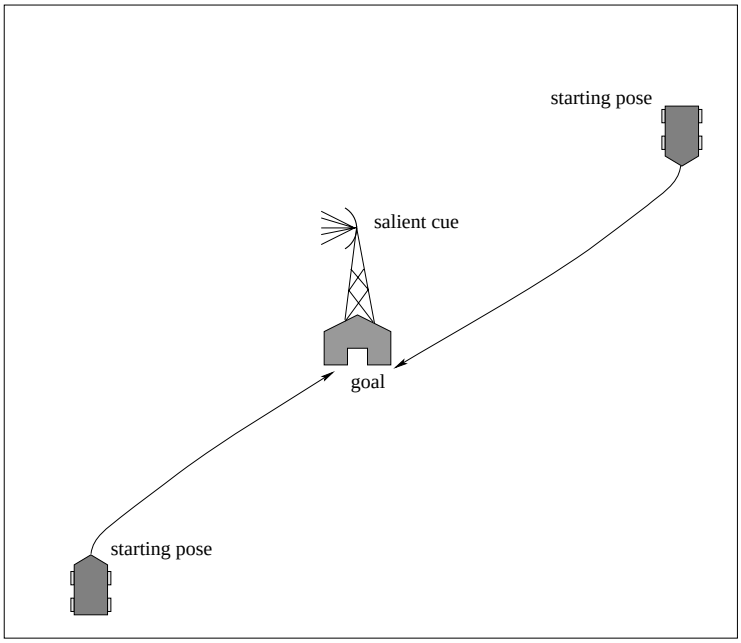


Figure 2.14: Illustration of the *aiming* competency (adapted from [Franz and Mallot, 2000]).

2.5.3 Aiming

The *aiming* competency requires the agent to orient its body such that the goal is in the direct path of the vehicle. As such, the goal has to be associated with some salient cue. *Aiming* is distinguished from *instruction following* and *path integration* by the fact that the goal can be approached from any direction. Not every location can be associated with *aiming*, as the goal is required to have a salient cue. An illustration of the *aiming* competency is given in Figure 2.14, where the vehicle heads to the goal based on the presence of a salient cue at the goal. This occurs regardless of the approach direction, as the goal is defined in terms of some cue which is visible from all locations within a 'catchment area'. The catchment area is the extent of the region of attraction, in this case limited by the visibility of the salient cue at the goal.

Aiming is one of the simplest navigation competencies to implement with examples dating back to the earliest days of robotics. Elsie, introduced in Section 2.2 is probably the earliest instance of autonomous *aiming*. This robot was capable of approaching a

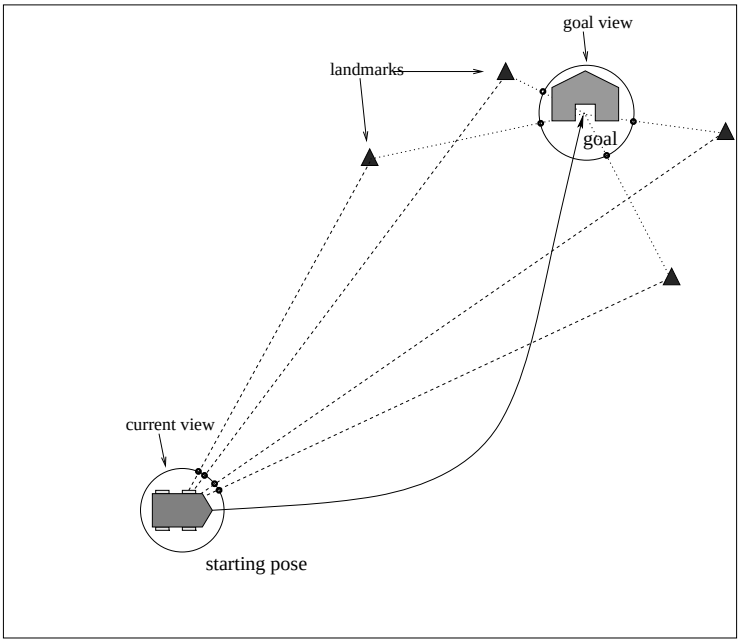


Figure 2.15: Illustration of the *homing* competency (adapted from [Franz and Mallot, 2000]).

charging station based on sensing of a light using a photo-cell. In 1984, Braitenberg re-ignited interest in low-level sensor-motor couplings with a series of thought experiments. One such experiment included *aiming* behaviours which have since been the foundation of many robotic implementations.

2.5.4 Homing

Homing allows an agent to position itself with respect to salient objects in the environment. In order to do this, a means of perceiving the environment is required, which distinguishes *homing* from *instruction following* and *path integration*. A goal is defined with respect to salient cues in the surrounding environment rather than at the goal itself, as in *aiming*. An example of the homing competency is shown in Figure 2.15. In this example, the agent attempts to minimise the error between what it currently perceives and a 'remembered' view at the goal location.

In this thesis, the main topic addressed will be visual homing. However, other sensing modalities have been successfully used with the same basic characteristics (e.g. Nehmzow

and Owen [2000] who used omnidirectional ultra-sound). Homing techniques have several advantages as a navigation method. Firstly, there is a tight coupling between sensing and action — local planning and control are closely intertwined. Secondly, it is possible to formulate homing techniques that have inherent obstacle avoidance (see e.g. [Weber *et al.*, 1998]). Lastly, complex models of the operating environment are not required — the environment itself is used for navigation.

Drawbacks of homing techniques include:

1. Perceptual aliasing — different locations in the environment cannot be guaranteed to be be perceived uniquely. This problem is however not isolated to homing, it is common to most navigation methods which 'look' to the environment for information, whatever 'look' may mean for a particular system.

2. Feature extraction — finding features that can reliably be extracted from a location with robustness to occlusions, lighting conditions and dynamic environments. Also, matching features with previously viewed features, known as *correspondence*, can be difficult.

3. Agent must be in the catchment area — landmarks surrounding the target location must be visible from the agent's current location.

4. Paths generated can be sub-optimal — this drawback again is not unique to homing. The only way of generating optimal paths is with complete knowledge of the environment and a competent global planner.

Visual homing, like reactive navigation (see e.g. [Brooks, 1986]), evolved out of a recognition of the fact that using three-dimensional representations of the world to navigate is time consuming and computationally expensive in both acquiring and using the model [Hong *et al.*, 1991]. To date, most homing algorithms allow the robot to attain a home *position* rather than a home *pose*. The main contribution of this thesis is to solve the pose stabilization problem, drawing upon results from the homing literature. Visual homing is reviewed in greater detail in Section 2.6.3.4.

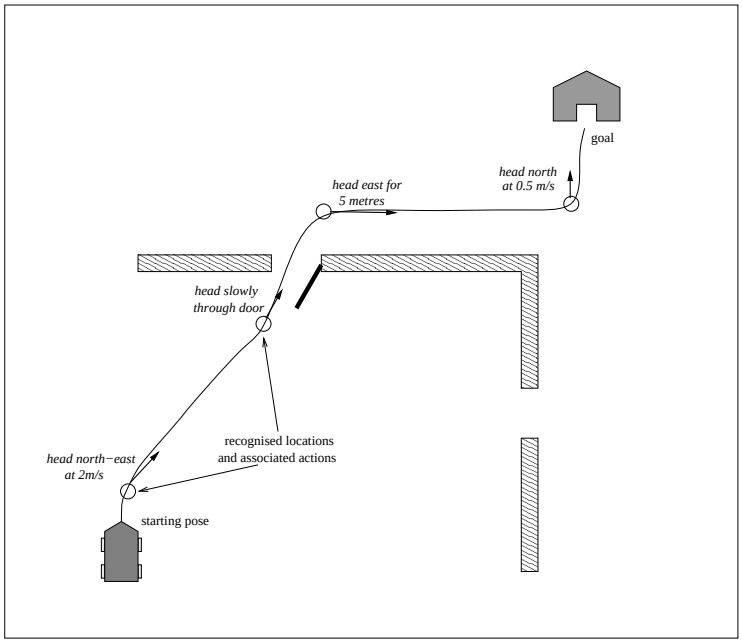


Figure 2.16: The *recognition-triggered response*. Circles indicate a recognised location, while the arrows indicate the associated action. Figure adapted from [Franz and Mallot, 2000].

2.5.5 Recognition-triggered response

Recognition-triggered response is the simplest of the global navigation techniques. Here, recognition of a starting location triggers a set of actions which take the robot to the goal location. The recognition of both a start and goal location, which may not both lie within the sensory horizon at all times, distinguishes *recognition-triggered response* from the *local* navigation competencies. The *recognition-triggered response* competency is illustrated in Figure 2.16. In this example, the robot associates a particular response with the recognition of each 'place' signified by a circle in the diagram. For example, at the door, the associated response is to head forward through the opening. Past the door, the associated response is to head for the next distinctive place by turning right.

Recognition-triggered responses can be used to generate 'routes' through an environment by connecting a series of sub-goals, where each sub-goal triggers actions which enable the next sub-goal to be reached. No planning is involved and the robot responds in an inflexible manner. In reaching the goal, the robot is confined to using the same se-

quence of sub-goals. If a route is blocked, the method will fail and an alternative strategy must be initiated.

2.5.6 Topological navigation

When using the routes generated by recognition-triggered responses, the robot is confined to using the same set of sub-goals to reach the goal. Topological[12] navigation allows the use of different sets of sub-goals to reach the same goal. This requires that multiple paths passing through a goal or sub-goal be identified and the possibly different sensory associations for that particular goal or sub-goal need to be merged, see Figure 2.17. Here, if the door was approached and found to be closed, a new plan could be formulated taking the robot along a different set of way-points. If such a situation was encountered when using a *recognition-triggered response* type competency, the robot would become stuck.

Like recognition-triggered responses, topological mapping associates a particular distinctive place with an action — in the figure the associated actions are shown as arrows. Unlike recognition-triggered responses, each distinctive place may have multiple actions associated with it. Thus, multiple 'distinctive places' can be connected by the actions required to reach other distinctive places. This can be represented graphically with a topological or nodal map, where the nodes represent distinctive places, and the links represent the actions required to reach other distinctive places; see Figure 2.18 for an example.

The concept of topological maps stems from studies of human and biological navigation (see [Kuipers, 1977; Trullier *et al.*, 1997] for reviews). Such a map provides a parsimonious environmental representation which can be used to navigate relative to the environment rather than to an absolute frame of reference. Take as an example walking from Brisbane central railway station to the Regent cinema on Queen Street. It is enough to know your location to the accuracy of a block, maintaining a position relative to the footpath. On reaching street corners, or some other defining locale in the environment, you re-localise, following a set procedure to reach the intended destination. In contrast,

[12] Topology is the science of place but the term has come to refer to the way in which constituent parts are arranged or inter-related [OED, 2000].

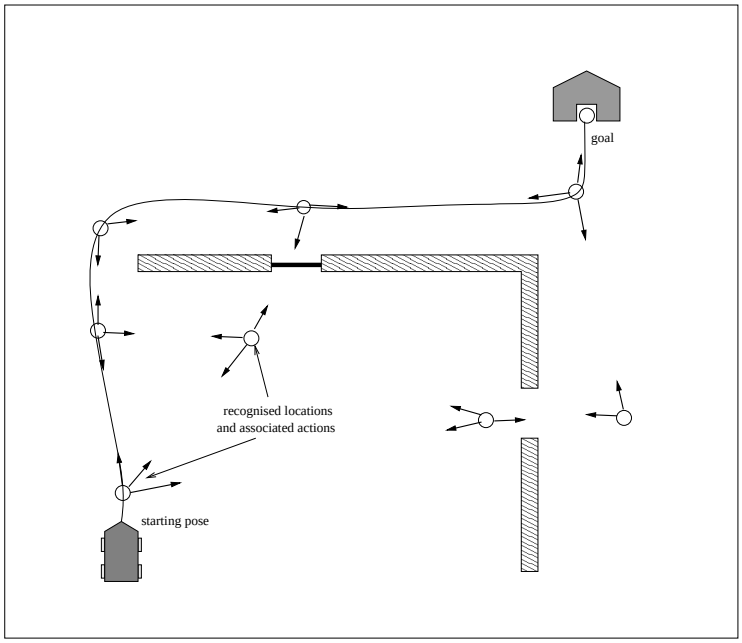


Figure 2.17: The *topological* competency. Circles indicate a recognised location, while the arrows indicate the associated actions. Unlike *recognition-triggered responses*, the *topological* competency associates many different actions with a distinctive location. Thus, the *topological* competency allows for re-planning, whereas the *recognition-triggered response* competency is limited to fixed routes. Figure adapted from [Franz and Mallot, 2000].

on entering a doorway, much more precise information is needed to prevent a collision and the environment is referenced directly for information [Gaspar *et al.*, 2000].

The strength of topological maps lies in the lack of requirement for rigorous metric information or grounding in an absolute coordinate system, although they are often augmented with metric information. This idea of 'multi-level representations' was presented by Chatilla and Laumond [Chatila and Laumond, 1985] in the mid 1980s. Many topological models to date have been constructed from an initial geometric map of the environment (see e.g. [Nehmzow and Owen, 2000; Golfarelli *et al.*, 2001; Thrun *et al.*, 1998; Thrun, 1998b]). Because of the presence of the full metric representation, they suffer from many of the drawbacks of metric systems, namely the computational overheads. The museum tour guide robot of Thrun *et al.* [1998], uses a topological map built from an evidence grid representation and is probably one of the more successful robots to date,

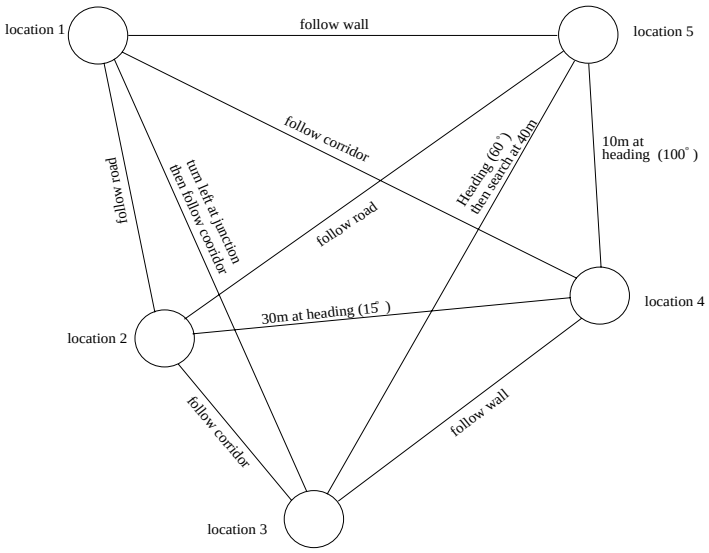


Figure 2.18: An example topological map. The nodes represent distinctive places, which are defined by salient features in the environment. Each node, or distinctive place, is connected to other nodes by the links which contain information linking the two nodes (e.g. wall-following). Of course, the linking information can be context dependent, a route between two distinctive places would usually contain information linking the nodes in *both* directions.

although it is somewhat limited in the scale of environment it can traverse.

A further example of the usefulness of topological maps is the work of Roberts *et al.* [2000] who demonstrated just how powerful topological maps can be. Their system allows a 30 tonne articulated (nonholonomic) mining vehicle to autonomously navigate through a network of tunnels using a topological map and a reactive wall-following behaviour. Mining tunnels have much in common with an indoor environment in that the robot is confined to move between the walls — a topological map is ideally suited to an environment with such a structure.

Topological maps allow a particular goal to be reached by multiple routes and thus a level of planning is required for topological navigation. This gives greater flexibility over the fixed routes generated by the recognition-triggered response technique. Maps are simple and approximate, making them relatively easy to construct and change. However, topological navigation is restricted to the 'known' (or learnt) paths connecting distinctive places, and the generation of paths over novel (unvisited) terrain is not possible. However,

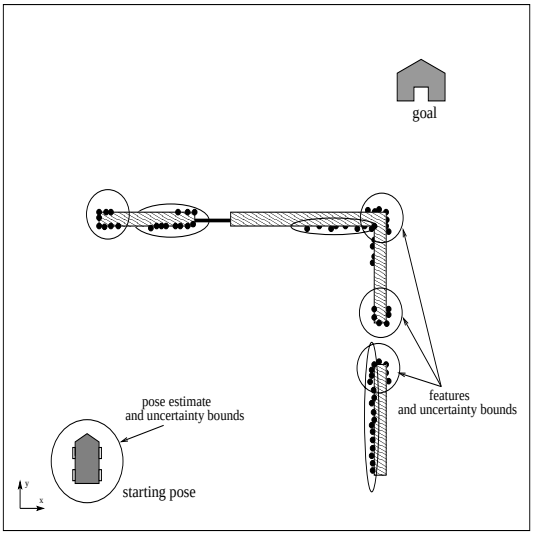


Figure 2.19: A typical feature-based map. Features used typically include: edges, corners and lines. Range and bearing sensors are the most common means of sensing features.

in practice this is not a significant restriction in the majority of industrial situations or applications. Integral to the use of a topological map is the ability to find and recognise distinctive places, and traverse the expanses that connect them. The main weakness with topological navigation is the robust recognition of these distinctive places.

2.5.7 Survey (metric) navigation

Survey or metric navigation embeds known locales or features in a common coordinate frame and thus allows the generation of paths over novel terrain. This method of navigation has received the most attention from the robotics community. The approaches to survey navigation are numerous with the most popular environmental representations being the grid-based and feature-based methods [Jensfelt, 1999]. However, some systems use a topological-like representation, grounded in a global coordinate system. Each is discussed in the following paragraphs.

Feature-based methods

The feature-based methods use easily identifiable objects in the environment, parameterising them with reference to, for example, colour, width, length and position [Jensfelt,

1999]. These features are then embedded into a common coordinate system [Betge-Brezetz *et al.*, 1996; Rencken *et al.*, ; Leonard *et al.*, 1992; Chatila and Laumond, 1985; Smith *et al.*, 1990]. These methods usually represent the state of the robot, and of the features in the environment, with an estimated state-vector and an associated covariance matrix. An extended Kalman filter is typically used to track the pose of the robot, with predictions of the robot's future pose provided by odometry and a vehicle model. Odometry is subject to an unbounded accumulation of error and hence, as the robot moves through the environment, the covariance grows. By periodically sensing features in the environment, and matching these to map features, the robot relative feature location can be used to improve the estimate of the state-vector and decrease its covariance [Bailey and Nebot, 2001]. An example feature-based map is shown in Figure 2.19.

The main advantages of this technique are its efficiency and the provision of an optimal global estimate of pose. However, this method is brittle in the face of modelling errors and the computational complexity becomes unmanageable for large-scale environments due to the large number of vehicle-to-feature and feature-to-feature correlations. Critically:

- Features are generally required to be viewpoint invariant; and
- Correct correspondences between (feature) observations and the map are required.

Currently, feature-based methods are limited to maps of approximately one thousand features.

In contrast to the active beacon systems mentioned in Section 2.4, feature-based navigation relies on the detection of known, salient and distinctive features in an environment whether they be natural or artificial. Natural landmarks may be man-made objects or features that serve a function other than for robot navigation, e.g. the corner of a wall [Borenstein *et al.*, 1997]. Artificial landmarks are essentially passive beacons, specifically placed in the environment for the purpose of aiding robot navigation, e.g. a pattern or bar-code [Borenstein *et al.*, 1997]. Landmarks, or features, can be unique or anonymous. A unique

landmark is defined by some distinctive characteristic, such as a barcode, which unambiguously distinguishes that landmark from other similar landmarks in the workspace. Anonymous landmarks are, of course, detectable in the workspace but have no individually defining characteristics — an example of such a landmark is an edge detected in an image. Once detected, landmarks then need to be matched to pre-known landmarks given an approximated position and a map, from which the robot can more precisely estimate its pose. Typical sensors used include computer vision, laser range-finders, and ultra-sonic range-finders.

The main task in feature-based or *landmark* navigation is feature recognition. It requires substantially more processing than that required for the active beacon systems. However, natural landmarks require no environmental modification and artificial landmarks are inexpensive compared to their active beacon counterparts [Everett, 1995; Borenstein *et al.*, 1997]. Robotic sensors often perceive the environment differently than we do. For example, using an ultrasonic sensor, it is impossible for a robot to 'see' a door. It is easier for such a sensor to 'see' the doorways [Nehmzow, 2000]. Thus, care is required when selecting relevant landmarks. Thrun [1998a] developed a technique which allows a robot to learn its own landmarks. However, the learning phase was quite long and the technique did not allow a human to influence landmark selection.

Grid-based methods

The grid-based representation of the environment, also known as an occupancy grid, was initially proposed by Moravec and Elfes [1985] and now, is probably the most popular means of environmental representation. Essentially, the environment is represented by a matrix of cells, each of which contains information on whether that particular cell is occupied by an object. An example grid-based map is shown in Figure 2.20. Moravec and Elfes matched locally constructed occupancy grids to a global one using a correlation technique. This method of localising was computationally intense and the robot could not move continuously. Schultz and Adams [1998] presented a similar system, however rather than waiting until a position error was detected, a locally constructed occupancy

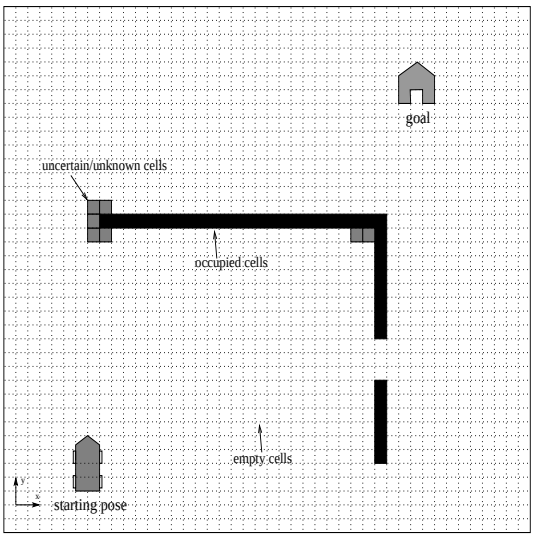


Figure 2.20: A typical occupancy grid representation. Here, the environment is represented by a matrix of cells. Each cell represents a small region in the environment and is assigned a probability of being occupied by an object at that location.

grid was matched to a global one continuously. Thrun *et al.* [1998] attacked the problem from a statistical view point with each cell in the grid continually updated with an estimate of the probability that the robot is in that cell.

More recent developments in the grid-based methods include the introduction of framed quad-trees [Yahja *et al.*, 1998; 2000]. The framed quad-tree approach places a frame of small cells around each quad (large cell). This allows many more exit directions for segments connecting the cells and therefore produces paths that are closer to optimal, as the paths generated for fixed grid sizes can be highly quantised. Other researchers have looked at fuzzy rules for classifying terrain (see e.g. [Seraji, 2000]) and multi-levelled maps with differing cell resolution (see e.g. [Hsu and Hwang, 1998]). More popularly, grid-based methods now incorporate probabilities of grids being occupied — otherwise known as occupancy grids (see e.g [Moravec and Elfes, 1985; Thrun *et al.*, 2000]).

Grid-based approaches are conceptually the simplest means of representing the environment and it is relatively easy to integrate multiple sensing modes into the one representation. However, distinguishing similar environments is difficult because of the reduced information content available, and the computational costs of localising the robot are high

[Thrun, 1998b]. Also, the memory requirements of such a representation are proportional to the area of the environment unless a sacrifice in resolution is made [Thrun, 1998b].

Absolute topological methods

Some systems use topological maps grounded in an absolute coordinate system. Nehmzow et al. [Duckett and Nehmzow, 1999a; 1999b; 2000; Nehmzow and Owen, 2000] experimented with such a strategy in rather large-scale indoor environments. Distinctive places were defined by a local occupancy grid[13]. These distinctive places were linked by distance and heading information and, using this metric information, each distinctive place was assigned coordinates in an absolute coordinate system. The method was robust to sensor imperfections and was demonstrated on a real robot, albeit in an indoor environment.

2.5.8 Discussion

An advantage of the metric or survey systems over a purely reactive system is the fact that path planning methods can be used to determine in advance whether the goal is achievable [Laubach *et al.*, 1998; Latombe, 1991; Choset and Burdick, 2000]. Furthermore, as the map used is usually of the same form we would use, the goal is easily specified and it is easy to communicate with the robot. Paths are usually optimal or nearly optimal, which is important when energy considerations are required, such as for planetary rovers.

Systems using survey navigation require a considerable amount of computational effort for updating and maintaining world models and these approaches have been criticised due to their heavy reliance on computational resources [Brooks, 1986; Brooks and Flynn, 1989; Desai and Miller, 1992]. This is particularly true of grid-based methods in which memory and computational requirements are proportional to the area of the environment, unless a sacrifice in resolution is made [Thrun, 1998b]. Consider Yamauchi's robot which requires three computers to navigate and map a small office/corridor environment — two off-board computers (Sparcstation 20's) deal with localisation and world modelling sep-

[13] *Local* here refers to containing information available to a current sensor reading

arately, with an additional on-board computer dealing with motor control and low-level sensor processing [Yamauchi *et al.*, 1998]. Of course, the available computational power is ever increasing, and methods which were untenable a decade ago are now well within reach.

A more relevant criticism for survey navigation is its reliance on accurate and continuous position estimation [Nehmzow, 2000; Brooks, 1986; Donnart and Meyer, 1996; Nagatani *et al.*, 1998]. Classical survey navigation techniques are essentially blind to the real world, seeing only abstract representations. Maintaining continuous estimates of position is a burden on the navigation system due to the difficulty and uncertainty in tracking position [Smith and Cheeseman, 1986; Borenstein *et al.*, 1997]. Furthermore, world models are generally hard to obtain, unreliable and difficult to maintain — clearly, dynamic environments in particular are problematic [Nehmzow, 2000; Arkin, 1995].

Many researchers see the answer to these problems in Simultaneous Localisation and Mapping (SLAM), also known as Concurrent Localisation and Mapping (CLM). This involves the somewhat circular process of localising the robot and exploring an environment while building a map (see e.g. [Smith *et al.*, 1990; Guivant *et al.*, 2000; Thrun *et al.*, 2000; Rencken, 1994]). In the words of Rencken [1994]:

> This is similar to the famous question of *which came first the chicken or the egg?*

In order to localise, the robot needs a map — in order to build a map, the robot needs to know its position on the map. A particularly difficult problem is that of cyclic environments with long traverses. Much research has focussed on map-matching and pose estimation from sensor readings in order to cope with cycles in the environment (see e.g. [Gutmann and Konolige, 2000; Lu and Milios, 1997; Yamauchi *et al.*, 1998; Scott *et al.*, 2000]. Pose estimation (localisation) is still a key problem for robots operating in real-world environments, particularly when the environments contain cycles. In addition, despite advances in computing technology, systems relying on metric representations are limited in the scale of environments they can traverse by the inefficiency of such repre-

sentations.

On the other hand, systems which navigate relative to the environment, such as topological navigation systems, can be built such that they do not have a need for continuous knowledge of the robot's position. This is not to say that relative systems cannot 'get lost', merely that because the robot is not continually localised, one mode of failure is almost eliminated. Relative systems can be augmented with metric data where it is appropriate, but do not rely solely on the abstract representations central to the operation of an absolute, metric system. Because the time-consuming and error prone abstraction stage is by-passed, relative systems can operate in real-time dealing with real environments.

Relative navigation can be more than a set of way-points or distinctive places, it can provide a means of mapping an environment in the sensory domain rather than in terms of geometry [Krose and Bunschoten, 1999]. Gaspar *et al.* [2000; 2000] use a set of omnidirectional images for long range navigation, and visual path following for more precise tasks such as door entry. Krose and Bunschoten [1999; 2001] pursue a similar line of research but instead use probabilistic techniques, concentrating on determining where the robot is in a given image set. Both of these systems map the image sets into Eigen-space, reducing the dimensionality of the image data, and have so far only been implemented on indoor systems. This form of navigation is also known as *appearance-based* navigation which correlates directly to relative navigation. More recently Milford and Wyeth [2003] presented a promising SLAM type technique based on hippocampal models of rat navigation — the method did not explicitly use metric information, rather the relations between distinctive regions in an environment were embedded in a neural network architecture.

However, there are some drawbacks with relative navigation systems. First, truly autonomous operation would have robots' exploring and building up representations of the environment that allow the subsequent definition and achievement of some goal. Relative systems do not refer to a 'hard' map, making any self-built environmental representation subjective and perhaps hard to interpret for a human. Further disadvantages include the fact that distinctive locales are difficult to robustly recognise.

Relative navigation is not hindered by a need for complete world modelling. It can draw on many of the control techniques developed for the absolute systems. A relative system can move away from the classical geometric environment representations to representation that are based in the sensory domain. Relative navigation systems can be built without a need for continuous knowledge of the robot's position, and hence, one mode of failure is almost eliminated. Relative navigation allows the redefinition of navigation from [Franz and Mallot, 2000]:

the science of course and position estimation

to

determining and maintaining a course to a goal location.

The aim of this thesis is to demonstrate feedback control at a relatively high-level in the navigation architecture, allowing a vehicle with significant motion constraints, such as a car, to stabilize to a target pose based upon what it can presently 'see' in the environment. Whether using a relative or absolute navigation method, this can free up computing resources for use by the higher level tasks, such as, for example planning paths through complex environments. The next section provides some background material on nonholonomic systems and reviews the literature on the pose stabilization problem.

2.6 Nonholonomic control systems

Nonholonomic systems have been studied in classical mechanics for over 150 years but it is only recently that the control of such systems has received significant attention [Astolfi, 1996]. Nonholonomic systems arise when there are constraints on allowable local motion in the system but not on the globally reachable configurations. In other words, such systems have restrictions on allowable velocities which do not impact on the allowable configurations which they can attain. Mathematically speaking, nonholonomic systems have velocity constraints that cannot be integrated to position constraints.

The classic example of a nonholonomic control problem is parallel parking of an automobile. In control terms, this is a *stabilization* problem in which the goal is to drive

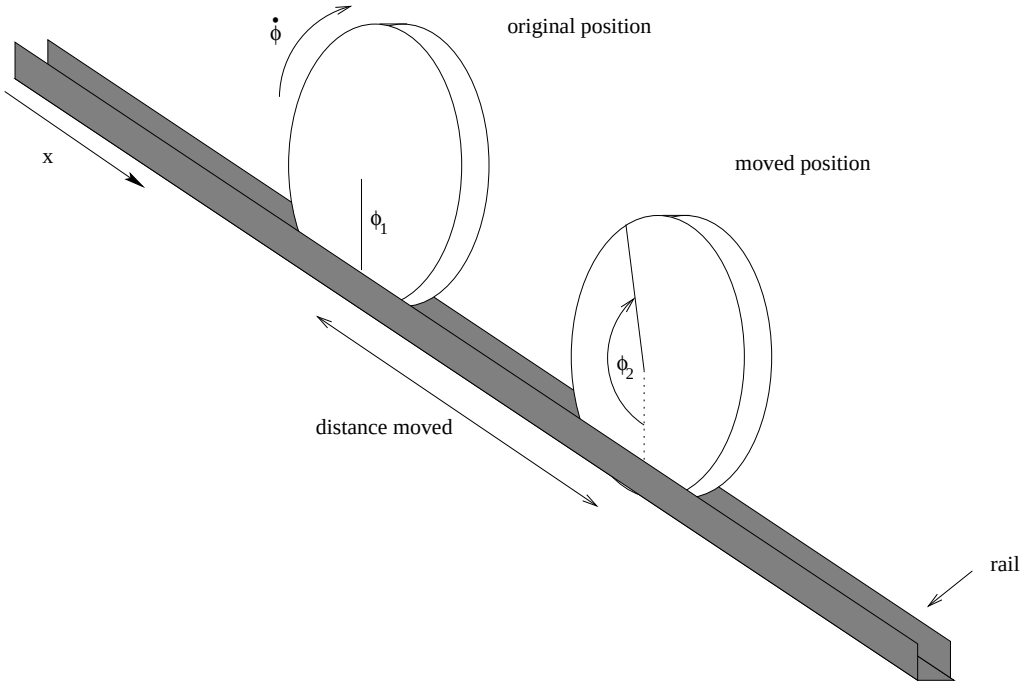


Figure 2.21: A disc on a rail. The disc's motion is constrained by geometry but the velocities of the system can be integrated to get the position of the disc – a holonomic system.

the system configuration to some desired point. As anyone who has driven a car knows, this is indeed possible for a car-like vehicle but may require considerable back and forth manoeuvring.

This section addresses the literature on the stabilization of nonholonomic control systems, looking in particular at car-like and similar vehicles. The section begins with an explanation of nonholonomy, introducing the kinematics of a commonly used platform in the the literature, the unicycle. Car-like vehicle kinematics are then introduced, and the stabilization control of nonholonomic systems is then reviewed. Finally, a review on visual pose stabilization and visual homing is presented.

2.6.1 Nonholonomy explained

Consider the disc, see Figure 2.21, rolling in a one-dimensional track without slip. Two coordinates can describe the disc's state — angle of rotation, ϕ, and distance moved along

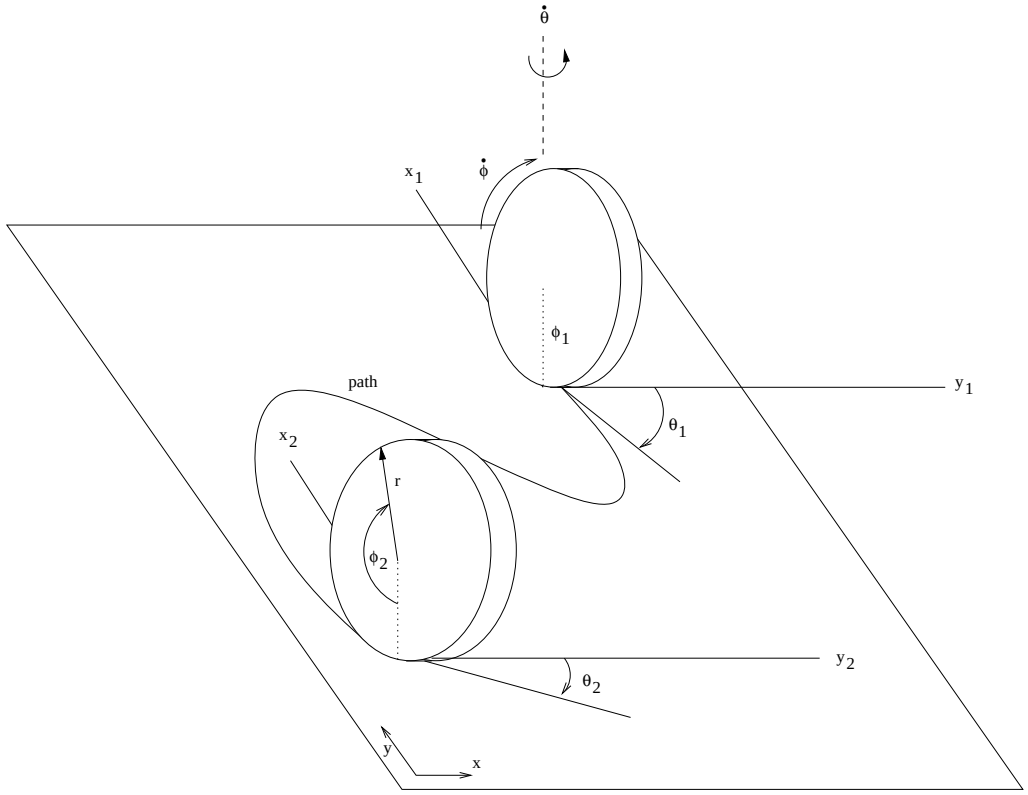


Figure 2.22: A disc on a plane. The disc's motion is constrained to the plane but its configuration is determined by the path taken to the present position. The velocities of the system *cannot* be integrated to obtain the system configuration – a nonholonomic system.

the track, x; the system has one input, $\dot{\phi}$. A relationship between distance travelled and the angle of rotation of the disc can easily be established which reduces the number of coordinates required to describe the state of the disc. The rate of angular rotation, $\dot{\phi}$, can be integrated to completely describe the state (also called configuration) of the system. If we know the current angular position of the disc and we were to roll the disc along the track, on restoring the original angle of rotation (restoring the *internal* state), the disc would be at the same position on the track. In other words, the state of the system is not path dependent and the system is *holonomic*.

Contrast this to a disc moving without sideways slip on a two dimensional surface, see Figure 2.22. The disc's configuration is described by the point of contact on the plane, (x, y), its roll angle, ϕ, and its yaw angle, θ. Demands to the system are the rate of roll,

$\dot{\phi}$, and rate of yaw, $\dot{\theta}$. The no-slip condition gives rise to constraints on the allowable velocities in the system:

$$\dot{x} - r\dot{\phi}\cos\theta = 0 \tag{2.1}$$

$$\dot{y} - r\dot{\phi}\sin\theta = 0 \tag{2.2}$$

where r is the rolling radius of the disc. These constraints are not integrable and there is no limitation on the configurations which can be attained by the disc [De Luca *et al.*, 1997]. When the generalised velocity of a mechanical system satisfies an equality constraint that cannot be written as an equivalent condition on the generalised position, the system is called a nonholonomic system [Wen, 1995; Latombe, 1991]. The disc-on-a-plane system is *nonholonomic* — for a nonholonomic system, the state of the system is *path dependent*.

The disc-on-a-plane system is also known as a *unicycle* for which the kinematic equations of motion are [De Luca *et al.*, 1997]:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{2.3}$$

where the demands are $v = r\dot{\phi}$ and ω, and the remaining coordinates are as previously defined. No reduction in the number of coordinates (3) required to describe the system is available, and the number of control inputs (2) is *less* than the number of generalised coordinates. When a nonholonomic system has fewer control inputs than generalised coordinates, the system states cannot be driven to a desired configuration using smooth, state feedback laws [Brockett, 1983]. For a vehicle system, driving all the system states towards a desired configuration is known as *pose stabilization*. Unicycle systems are well-studied in the nonholonomic pose stabilization literature.

So, in determining whether a system is nonholonomic, the important question is:

- Are the system velocity constraints integrable to position constraints?

If a system *is* nonholonomic, and the system is to be controlled, the important additional question is:

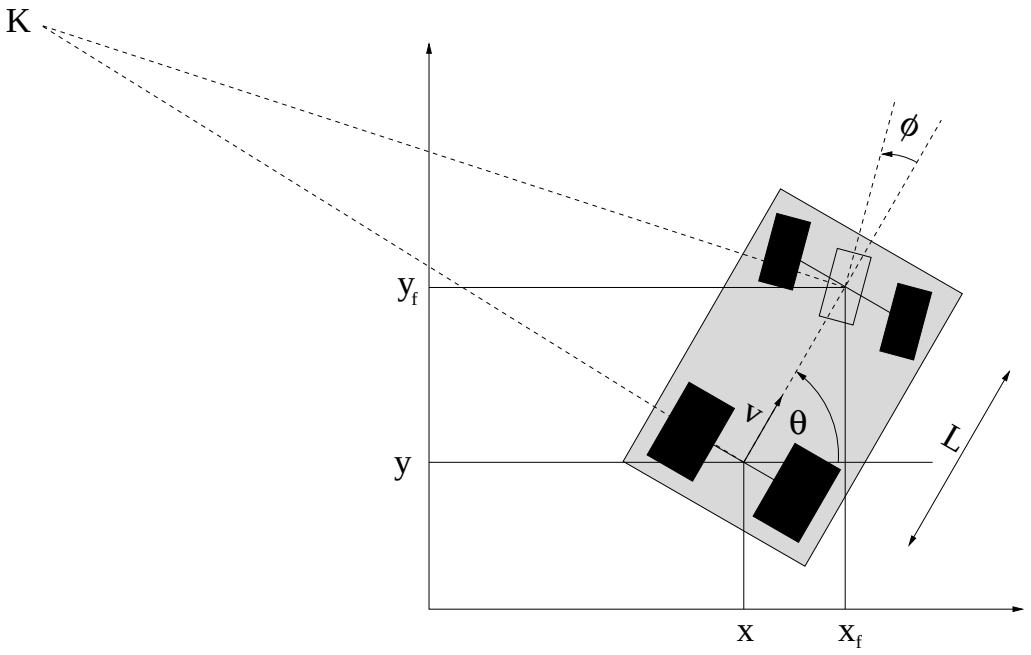


Figure 2.23: Car-like vehicle kinematics. Like the disc on a plane, the car's motion is constrained to the plane but its configuration is determined by the path taken to it's present position. The velocities of the system *cannot* be integrated to obtain the system configuration – a nonholonomic system.

- Are the number of inputs *less* than than the number of generalised coordinates?

Such systems present challenging control problems as they allow the attainment of virtually any configuration available in the workspace but the attainment of a particular configuration is intimately related to the path taken.

2.6.2 Car-like vehicle kinematic model

Figure 2.23 shows a car-like vehicle moving in a plane with velocity v and steering angle ϕ. When modelling such a vehicle, it is normal to collapse the front and rear wheels into a single, virtual wheel at the centre of each axle, the *bicycle* model. The front wheel can be steered while the rear wheel orientation is fixed relative to the vehicle. The generalised coordinates for this system are $q = (x, y, \theta, \phi)$ where x and y refer to the coordinates of the midpoint of the rear-axle (i.e the position of the rear *virtual* wheel), θ is the orientation of the longitudinal axis of the vehicle with respect to the x-axis, and ϕ is the vehicle's

steering angle. This geometry constrains movement of the vehicle to rotation about point K, the instantaneous point of rotation, provided there is no sideways slip of the wheels. The no-slip condition gives rise to two kinematic constraints, one for each (virtual) wheel:

$$\dot{x}_f \sin(\theta + \phi) - \dot{y}_f \cos(\theta + \phi) = 0 \tag{2.4}$$

$$\dot{x} \sin(\theta) - \dot{y} \cos(\theta) = 0 \tag{2.5}$$

where x_f and y_f refer to the coordinates of the mid-point of the front-axle.

These constraints impose zero lateral velocity for both wheels. Using the rigid-body constraint[14]:

$$x_f = x + L\cos\theta \tag{2.6}$$

$$y_f = y + L\sin\theta \tag{2.7}$$

where L is the distance between the vehicle's front and rear axles, the first kinematic constraint (Equation 2.4) becomes:

$$\dot{x} \sin(\theta + \phi) - \dot{y} \cos(\theta + \phi) - \dot{\theta} L \cos\phi = 0 \tag{2.8}$$

These constraints can be shown to be non-integrable using a corollary of the Frobenius theorem (see Latombe [1991] for a more complete explanation) *so the car-like system is nonholonomic.*

For a rear-wheel drive vehicle, the kinematic model is given by [De Luca *et al.*, 1997]:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{L} \\ 0 \end{bmatrix} v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{2.9}$$

where v_1 is the vehicle's forward velocity (measured at the centre axle of the rear wheels) and v_2 is the steering angle rate. This system is a non-linear, driftless[15] system with fewer

[14] A *holonomic constraint.*

[15] The term driftless refers to the fact that no motion takes place under zero input. On a flat plane, with no control inputs a car-like vehicle remains in equilibrium. This is not the case for a flying vehicle which, in general, will fall to the ground if no control inputs are provided.

control inputs (2) than generalised coordinates (4) [De Luca *et al.*, 1997] — *a car-like system cannot be feedback stabilized using smooth, state feedback laws* [Brockett, 1983; De Luca *et al.*, 1997; Samson, 1993; Latombe, 1991; Samson and Ait-Abderrahim, 1991; Campion *et al.*, 1991; Bloch and McClamroch, 1989].

In the control community, the problem of stabilizing a mobile robot to a specific pose has generally been approached from two directions; the open-loop and closed-loop strategies. For the task at hand, we argue the case for the closed-loop control methods, with vision used as the primary sensor. However, it is clear that a measure of open-loop planning is also usually required in order to prevent deadlock situations from occurring. The next section explores the open and closed-loop strategies of pose control, together with a review of vision for the position and pose control of mobile robots.

2.6.3 Pose stabilization

In this section, the literature on pose stabilization is reviewed. This literature is dominated by results from the control community, usually using mobile robots with unicycle-like kinematics. Solutions to the problem largely rely on complete state knowledge of the robot, and as will be shown, real experiments are few. Following the review on pose stabilization, a review of the literature which exclusively deals with vision-based control is presented. This includes *visual homing* which is based upon the *homing* competency discussed in Section 2.5.4. Visual homing, in essence, solves the position stabilization problem, where the robot is driven towards a goal position, with no requirement on its orientation, using visual feedback. Results in this field of the literature largely come from the 'biologically-inspired' school of thought, with many solutions attempting to emulate the visual and control 'shortcuts' exhibited by insects in particular. Our aim is to unite the two fields of research, solving the pose stabilization problem using vision as the primary sensor.

As noted in the Section 2.6.2, the pose stabilization problem has been attacked from two major fronts, the open-loop approach and the closed-loop feedback approaches. We begin the discussion with the open-loop approaches.

2.6.3.1 Open-loop control

Open-loop control, or path-planning, and mobile robots have been synonymous areas of research for many years due to the sense-model-plan-act approach common to mobile robot systems. For a summary of early work refer to Latombe [1991] and Li and Canny [1993].

Open-loop strategies seek to find a bounded sequence of control inputs, driving the vehicle from an initial position to some arbitrary position, usually working in conjunction with a motion planner (see e.g. [Latombe, 1991; Murray and Sastry, 1993]). Finding this sequence of control inputs is difficult for the nonholonomic case and, in the event of disturbances or modelling inaccuracies, a new plan has to be formulated [Oriolo *et al.*, 1998]. Computing power is ever increasing, however the compromise between optimality and real-time operation still does not favour these approaches, particularly in our case were we wish to run vision and control on a single on-board computer.

Early work on open-loop control concentrated on search-based techniques which essentially performed an exhaustive search of all possible control input variations, satisfying the nonholonomic constraints, which stabilized a vehicle to a desired configuration (see e.g. [Barraquand and Latombe, 1989]). They usually operate through some decomposition of the configuration space into cells, after which some graph is constructed which connects a starting and goal configuration. A search algorithm, such as A^*, is then used to identify the 'best' path from a starting configuration to the goal configuration. 'Best' paths are described in terms of distance, minimal control inputs, shortest time, or any other relevant cost measure or combination of cost measures.

These methods are computationally very expensive which motivated research on more efficient techniques. Murray and Sastry [1993] looked at more efficient and elegant methods based on the tools from geometric control theory, applied to a car-like vehicle. More recently, Kelly and Nagy [2002] looked at on-line trajectory generation with a promising parametric optimal control based approach, but again published results have so far been limited to simulation studies. Oriolo *et al.* [1998] turned to learning-based control

to overcome the computational costs but have so far been restricted to simulation studies. Other work has looked at path-space iteration (see e.g. [Divelbiss and Wen, 1997; Lizarralde and Wen, 1996]) in which the problem is transformed to a root-finding problem and solutions performed in an iterative, feedback type manner.

Open-loop control in general fails when the robot is subjected to disturbances or if there are inaccuracies in the plant model. The on-line approaches overcome some of the shortcomings of open-loop control, enabling the robot to re-plan in the event of model inaccuracies or other disturbances which affect the true path taken by the vehicle. However, many of these strategies are computationally very expensive, preventing the running of other required processes, in our case computer vision, on the same computer processor. As computer power increases, providing more resources and time for running other required processes, these on-line strategies will become more attractive — the open-loop strategies can offer optimal, or close to optimal, solutions to the problem and can more reliably deal with other constraints, such as the presence of obstacles in the workspace.

2.6.3.2 Closed-loop control

The closed-loop strategies consist of designing a feedback loop using proprioceptive and exteroceptive sensors to provide estimates of the vehicle's state (see e.g. [De Luca *et al.*, 1997]). Feedback control systems are generally more robust to uncertainty and disturbances when compared to their open-loop counterparts. All real mobile robots and sensors are subject to noise and uncertainty — feedback control would thus seem essential.

Brockett's theorem states that for a closed-loop system to be stabilizable to a desired configuration using smooth, state feedback laws, the number of control inputs must be greater than or equal to the number of generalised coordinates. *For a vehicle with car-like kinematics this is not the case* and fundamentally non-linear feedback techniques must be used. However, as the experience of anyone who has driven an automobile shows, the system *is* controllable and stabilizable. For mathematical proofs for the general case of driftless, nonholonomic systems the reader is referred to [Brockett, 1983] and for the case of mechanical systems, refer to [De Luca *et al.*, 1997; Samson, 1993; Latombe,

1991; Samson and Ait-Abderrahim, 1991; Campion *et al.*, 1991; Bloch and McClamroch, 1989].

Although smooth state feedback cannot be used to stabilize nonholonomic robots to a pose, nonlinear feedback laws do exist. These laws rely on time-varying strategies, discontinuous control or combinations thereof [Kolmanovsky and McClamroch, 1995]. For a comparison of the three strategies, refer to Canudas de Wit *et al.* [1993] or Kim and Tsiotras [2002].

Here, the literature is discussed based upon the above taxonomy, restricting the discussion, unless necessary, to mobile robot systems rather than nonholonomic systems in general. Importantly, much of the literature does not address what has been found in this study to be a significant limitation of many control algorithms for the closed-loop pose stabilization of car-like vehicles — input saturation and the dynamics of the steering and velocity loops. Further to this, few researchers have implemented systems on *real* robots.

Discontinuous, time-invariant control

Discontinuous techniques operate by providing either piecewise continuous feedback laws, or through a non-smooth transform of coordinates in which a smooth controller is designed in the new coordinate system — when transformed back to the original coordinate system, these controllers are no longer smooth. The discontinuous techniques have by far seen more implementations on real systems and demonstrate considerably faster convergence rates than their time-varying counterparts. However, most real implementations have been on unicycle-like robots which, although nonholonomic, are much simpler in their kinematics.

The existence of piecewise smooth feedback controllers for nonlinear control systems was shown by Sussman [1979] and later demonstrated on several examples of low-dimensional systems by Canudas de Wit and Sørdalen [1992a]. The piecewise smooth techniques operate by splitting the configuration space into sub-manifolds. A supervisory level in the controller then switches a set of time-invariant low-level controllers such that stabilization to each sub-manifold of the configuration space proceeds, eventually leading

to stabilization of the system to the desired configuration. This can occur with sliding mode type controllers or with discrete-event switching.

Bloch *et al.* [1992] were one of the earliest researchers to present this type of approach, using a discrete event supervisor which switched between low-level, smooth, time-invariant controllers. In this approach, there are two stages of design:

- First, find an open-loop strategy which will achieve the desired vehicle pose from any initial configuration.
- Second, transform the motion sequence into a succession of equilibrium manifolds, each of which can be stabilized by feedback.

Such an approach is very simple provided a suitable motion sequence can be determined. This simplicity of design can allow the use of linear control tools for each motion phase, with the resulting control being much easier to analyse and predict. However, the technique relies on the ability to devise an open-loop strategy for the system stabilization — such strategies may not be obvious for some nonholonomic systems. In addition, disturbances to states which are *not* controlled in the current phase of control cannot be corrected at the time they occur unless another phase of control is initiated [De Luca and Oriolo, 1995].

Later work by Lee *et al.* [1999] used a similar switching technique rooted in this philosophy, which allowed a car-like vehicle to stabilize to a pose. Feedback in this work was provided by inertial sensors which were integrated to give robot pose. As discussed in Section 2.4.1, current inertial sensors are subject to drift, which upon integration to a position estimate, leads to a significant accumulation of error if an outside sensing source is not used for correction. However over the short distances in these experiments, the method was successful at stabilizing the vehicle to a desired pose.

Sliding mode controllers for the problem of stabilizing a unicycle-like mobile robot to a pose were presented by Guldner and Utkin [1994]. Later work by Lu *et al.* [2000] extends sliding mode control to a car-like vehicle in simulation. Implementation on a real vehicle has not been demonstrated.

Some of the switching type controllers[16] are susceptible to 'chatter', in which the system oscillates around discontinuities in the system. Of course there are means to alleviate such problems, for example, by providing boundary layers for the switching surfaces. An alternative approach to nonholonomic pose stabilization is to use a coordinate transformation. Systems, which operate with smooth controllers in a transformed coordinate system avoid the chattering encountered in systems which rely on switching or sliding mode strategies.

Changes in coordinates were initially exploited by Badreddin and Mansour [1993] who used a fuzzy tuned state feedback control law based in the polar representation of a unicycle-like vehicle's pose. This transformation allows the design of smooth controllers in polar-space, but creates a discontinuity at the origin. Their method involved the fuzzy selection of gains, based upon measurement of the robot state. They present simulation results but claim that the method has been functioning on a real robot for some time.

Aicardi *et al.* [1995] and many later authors (see e.g. [Pourboghrat, 2002; Aguiar *et al.*, 2000; Kantor and Rizzi, 2003; De Luca *et al.*, 2002]) have based their system on the original Cartesian to polar transformation of Badreddin and Mansour. Aicardi *et al.* present more explicit proofs of the stability of the method and use fixed gains to stabilize a unicycle-like robot to the origin in simulation. Later work includes the use of vision as a feedback sensor, using switching strategies to ensure that visual targets remain in view [Conticelli *et al.*, 1999; 2000]. Indiveri [1999] attempted to adapt the model to the bicycle model, commonly used for car-like vehicles, including analysis of saturation of the velocity input. However, for a car-like vehicle, the important input in terms of saturation is the steering angle and that work did not address this issue. In addition, all experiments were simulated.

Parallel to the work of Aicardi *et al.*, Astolfi [1995] developed techniques applicable to all chained form nonholonomic systems and explicit methods to design discontinuous controllers based upon a suitable coordinate transform. Astolfi [1995]; [1996] addresses

[16] Sliding mode control is not strictly a discrete state switching controller but it is conceptually similar, operating by suppressing terms in the control law rather than switching between controllers.

car-like vehicles but again, it is assumed that the modelled vehicle has no steering angle or velocity input saturation. Although Astolfi experimented with a unicycle-like robot, all experiments on a car-like vehicle were simulations. Canudas de Wit and Sørdalen [1992b] also use a change of coordinates, designing a smooth feedback law in this new coordinate system to stabilize a simulated unicycle to a pose. Their change of coordinates is based upon a circle tangent to the desired configuration, rather than a transform designed to put the system into chained form. This work provided excellent stabilization times and avoided 'chattering' at the discontinuities of the piecewise continuous control law however, all experiments were simulations.

Time-varying control

Time-varying strategies use continuous feedback strategies combined with an exogenous time variable which essentially gives the system control energy at configurations where it would otherwise become stuck in 'local minima'. One of the earlier researchers to use time-varying strategies was Samson [1991]. An exogenous time variable combined with a smooth, static state feedback law allowed a simulated mobile robot to stabilize to a pose using either velocity or torque inputs. Based upon this result, Coron [1992] proved that all controllable, driftless systems could be stabilized to a pose using smooth periodic, time varying feedback but did not provide any feedback laws. Feedback law design approaches using a similar method were presented by Pomet [1992] and Samson and Ait-Abderrahim [1991]. The former work was based upon transferring the system state representation into a chained form. The latter work was based upon following the trajectory of a 'reference cart'. Samson [1993] later also incorporated the idea of transforming coordinates to help controller design.

More recent results in the time-varying literature include analysis of saturated inputs for a unicycle-like robot [Jiang *et al.*, 2001]. In the case of this work, the convergence rate is again exceedingly slow for the stabilization problem and control inputs are, as with most of the time-varying literature, highly oscillatory which could lead to problems on 'real' robots. Others, have used adaptive control techniques in an effort to deal with plant

uncertainty (see e.g. [Pourboghrat and Karlsson, 2002]).

Kim and Tsiotras [2002] studied several time-varying controllers on a Khepera robot, and found that all were very slow, in comparison to other types of controllers, and in one case divergent. In general, real, non-simulated studies of time-varying strategies have come to similar conclusions.

Hybrid discontinuous and time-varying control

A criticism of time-varying control is the slow convergence rate. M'Closkey and Murray [1997] built on the results of Pomet [1992], presenting a design approach to constructing time-varying feedback laws with exponential convergence rates when stabilizing to a desired configuration. This is based upon a non-smooth state feedback combined with an exogenous time variable. Their results included application to a real, car-like robot, in which their controller compared favourably with earlier work in time-varying control, including Pomet [1992]. The mobile robot used was attached to a mechanical linkage which provided the state feedback used for control.

Sørdalen and Egeland [1995] presented a method which combined a discontinuous controller with an exogenous time variable, applying it to general nonholonomic systems which could be transformed to the chained form. They present results for a simulated car-like vehicle showing excellent convergence rate. However, as with much of the work in the field, results were simulated and the effects of saturated control inputs were not considered.

2.6.3.3 Visual pose stabilization

Much of the work in pose stabilization has been on simulated models of vehicles, rather than real vehicles. In this section, the research which has produced real working vehicles is reviewed, focusing on those systems which use vision for feedback.

Many implementations have been in indoor environments, taking advantage of the structure and the ground plane constraint (i.e. the ground in the environment is completely flat allowing an estimation of the range of an object from its vertical position in the image

— Horswill [1993] was one of the earlier researchers to take advantage of this).

In addition, most of the vision-based pose stabilization literature uses monocular cameras for which the controller must also maintain the target in the camera field of view, as well as stabilizing the vehicle. In order to do this, most researchers use switching control techniques similar to the work of Bloch *et al.* [1992]; [1990]. Here, a discrete-event supervisor chooses the appropriate controller from a suite of controllers, based upon the current state of the robot. Of course, for such a strategy to work, the target features must be in view at the robot's initial pose, unless a further 'feature search' type behaviour is provided at the initial pose. This constraint restricts the applicability of monocular cameras to the pose stabilization problem. A sensor more suited to the problem is the omnidirectional camera. An omnidirectional camera avoids the requirement for control mechanisms to maintain the target features in the field of view because the field of view is panoramic. However, like monocular cameras, target occlusion can still occur.

Kantor and Rizzi [2003] experimented with a small unicycle-like robot equipped with a colour, monocular camera. They use an engineered coloured cube with each face painted a different colour, allowing full pose estimation from a single monocular image. Their work is based upon the Variable Constraint Control (VCC) paradigm which divides control problems into a natural sequence of subproblems which can be solved using feedback — this technique is used to stabilize the robot and to maintain the 'beacon' within the image frame during robot motion. As noted earlier, the use of an omnidirectional camera would avoid this latter requirement because targets do not 'disappear' as easily from such an image due to the panoramic field of view. They report results from a stabilization experiment but, because the strategy is also maintaining the landmark in the field of view, convergence to the goal configuration is relatively slow.

Similarly, Conticelli *et al.* [1999] rely on a supervisory level which switches between low-level controllers to ensure features are maintained in the image, and the robot is stabilized to a pose. Like much of the work in this field, their vision system relies on tracking of corner or edge-based features, in other words, distinctive regions in an image of high intensity curvature. In this case, Conticelli *et al.* [1999] use corner tracking to

track a landmark against a white background. The low-level controllers in this work were based upon the polar representation pioneered by Badreddin and Mansour [1993]. Later work by Conticelli *et al.* [2000] simulates a robot and vision system, again using features extracted from the artificial environment, combining vehicle odometry with vision to estimate vehicle-object relative positioning. However this work relies on more complicated switching between low-level controllers. Extensions to this work using a similar switching strategy were experimentally implemented on a unicycle-like robot [Murrieri *et al.*, 2002]. Here, corner features were tracked using multiple temporal windowing and filtering of the grabbed images — this process increases the signal-to-noise ratio of the vision system. However as with the work of Kantor and Rizzi [2003], convergence is slow and necessarily oscillatory to maintain features in the camera's field of view.

'Artificial' visual cues on a wall are a popular means of providing systems with 'landmarks'. Hashimoto and Noritsugo [1997] use such visual cues to control a car-like, remote-control vehicle. They use blob extraction techniques to find the features and assume knowledge of the world coordinates of these features. All processing occurs off-board, based upon vision transmitted from a monocular camera mounted on the vehicle. Their controller is an extension of that presented by Canudas de Wit and Sørdalen [1992b], who used a discontinuous transform of coordinates based on a virtual circle with a tangent at the origin. They transform this representation to an image-based form. In combination, these transformations can produce a singularity which prevents full pose stabilization (i.e. if an orientation error exists on reaching the target position, there is not enough control effort to resolve it). To overcome this, Hashimoto and Noritsugo [1997] introduce a switching strategy. This switching strategy does not provide a specific mechanism to ensure that features remain in the image — in this case such a mechanism is not required because their controller is resolving the robot pose error in image coordinates. However, it must be assumed that when the robot begins its motion that the features are already in view. Resulting vehicle motion is highly oscillatory at the origin although the time to convergence is relatively fast. However, given the size of the vehicle, the residual error on reaching the target pose is relatively large, of the order of $0.15\,\mathrm{m}$.

Lietmann and Lohmann [2000] also use artificial visual cues on a wall, again in a visual servoing framework. The features used are four 'blobs' which are extracted from the image using thresholding and blob-analysis. They combine an image-based visual servoing system, which does not cope with the nonholonomy of the system, with a trajectory generator, which constantly provides 'virtual' goal points which are reachable given the nonholonomic constraints of the vehicle. These virtual goal points lead the vehicle towards the desired configuration.

An alternative to an omnidirectional camera is to fit the vehicle with a manipulator mounted monocular camera[17] which can be used to track features during the vehicle's motion. Using a hybrid time-varying and discontinuous control law rather than a switching strategy, Tsakiris *et al.* [1997] stabilize a car-like vehicle fitted with a monocular camera mounted on a robot arm. Because the camera was mounted on a robot arm, the target features could be tracked and the stabilization of the mobile base did not require specific motions to keep the target in view. Off-board VLSI chips were used to process the vision data, relying on the concept of active-windowing to track regions of interest — a Kalman filter was attached to each region of interest, performing tracking. The object providing the corners for tracking was a fixed target, the size of which was assumed to be known. They present methods and results for three different tasks: stabilizing the robot arm-mounted camera to a pose, stabilizing the mobile base to a pose (while tracking the image features by servoing the robot arm) and, stabilizing the mobile base and camera pose. Their method is based upon the earlier work of Samson [1995], in hybrid time-varying and discontinuous controllers. The convergence rate of this type of control is slow, and results presented by Tsakiris *et al.* [1997] are for very modest initial conditions (the vehicle's initial distance from the goal was less than $25\,\mathrm{cm}$ for which it took $188.4\,\mathrm{seconds}$ to stabilize).

Das *et al.* [2001] experimented with a small toy truck equipped with an omnidirectional camera. Images are transported to an off-board computer and the processed data used to control the position of the truck. The controller is based upon a transformation of

[17] A pan-tilt camera would be just as effective.

coordinates, and position control is achieved with the use of a reference cart. It is noted that this research does not solve the pose stabilization problem, it merely allows the vehicle to attain a goal position with no requirement on goal pose, similar to the visual homing literature. However, earlier work by Das *et al.* [1998] involved simulating formations of mobile robots and parallel parking based upon a 'follow-the-leader' concept in which the leader followed some optimal path derived off-line, and the follower vehicles mimicked the leader's motion using feedback control techniques.

Other systems rely on optic flow techniques alone to provide information on time-to-collision and also stabilization with respect to a visually rich target (i.e. a target with a large number of edges allowing good optic flow calculations), see for example [Santos-Victor and Sandini, 1994].

Another means of providing visual feedback is from an off-board camera. De Luca *et al.* [2002] use a calibrated camera attached to the ceiling to track the robot. The robot is equipped with a top mounted board containing three LEDs which form an isosceles triangle pointing to the front of the vehicle. The LEDs are found in the image using thresholding and blob-analysis, from which the position of the triangle can be deduced and the pose of the vehicle estimated. They compare two sets of controllers, the first based on the polar representation of Badreddin and Mansour [1993] and the second based on Dynamic Feedback Linearisation (DFL). The DFL method augments the robot state by addition of an integrator on the linear velocity input leading to a PD type controller. The DFL controller compares favourably but careful selection of gains is required to avoid singularities at the origin.

The discussion so far has been limited to robots operating indoors as the majority of research conducted in pose stabilization is for small indoor robots, usually unicycles. An exception is the work of Minten *et al.* [2001] who looked at docking type behaviours, in indoor and outdoor settings. Their work is based upon potential-field type approaches to illicit the required response from the vehicle. The potential field is designed to give the robot the correct orientation on reaching the goal — in this case, the task is to dock a small robot with a 'mother robot'. The vehicle uses a monocular colour camera to track

a landmark which contains two horizontally adjacent rectangles of different colours. In contrast to the techniques described above, the control law is simply based on the pixel counts of each rectangle. For example, if the ratio of right to left pixel count is greater than one then turn left. Vehicle velocity is set according to the total number of pixels. Minten *et al.* [2001] conducted trials in a variety of indoor and outdoor environments of differing lighting conditions, demonstrating a relatively high success rate, albeit from relatively close starting positions.

2.6.3.4 Visual homing

Stabilization has been researched by two disparate sets of groups. The first is the control community, who usually solve the problem with full robot state knowledge, with few instances of real functioning systems. However, as the previous section demonstrated, there is a growing body of literature on vision-based pose stabilization for mobile robots. The second group take inspiration from biology, seeking solutions which are much simpler, using visual and control 'shortcuts' to solve the problem of positioning a mobile robot with respect to its surroundings — in this literature no constraint is placed upon the robot's goal orientation. In other words, this literature solves the *position* rather than *pose* stabilization problem, and this behaviour is referred to as *homing*, as discussed in Section 2.5.4. The vast majority of the literature on this problem is based upon the use of visual feedback and is often inspired by biology, in particular insect navigation. The use of omnidirectional vision is much more common in this literature, and avoids the problems discussed in the previous section of maintaining features within the camera's field of view.

Much like early work in reactive navigation, visual homing was driven by the inadequacies of classical path planning approaches and their need for complete knowledge of the operating environment. There are two major schools of thought on navigation using vision-based feedback [Weber *et al.*, 1998]:

1. Landmark-based homing — salient features in the current and target views are extracted with the differences used to derive a homing vector

2. Image-based homing — differences in the raw images taken at the current and target locations are used to derive a homing vector.

This categorisation is based upon competing hypotheses for the mechanisms of insect navigation — *Do insects use features within a retinal image or the retinal image itself to find locations in an environment* [Möller, 2001]*?* For robotics, the former is the more popular.

Image-based techniques rely on matching whole images obtained at the goal and starting location. Early work using this type of approach includes that of Franz *et al.* [1998a; 1998b; 1997] who use a slice of an omnidirectional view of the world at the horizon. This current view of the world is then distorted in such a way as to bring it closer in appearance to the home view, from which a homing direction is derived. They implemented their system on a small robot operating in an artificial 'town like' environment. They report good results for starting locations which are close to the target location ($< 15\,\mathrm{cm}$) but success drops away dramatically as the robot is displaced further from the goal.

More recently, Zeil *et al.* [2003] performed experiments with an omnidirectional camera mounted to a gantry which allowed three-dimensional translations. They showed that through simple image subtraction, using whole images rather than a reduced view relying on a slice at the horizon, the system was able to return to a target location within a fairly small 'region of attraction' ($1\,\mathrm{m}$). Importantly, their system operated outdoors in unmodified environments. However, as with most of the image-based techniques, the catchment area (or region of attraction) is fairly small in comparison to the landmark-based techniques.

Turning to the landmark-based approach, Hong *et al.* [1991] were one of the earliest researchers to experiment with a mobile robot equipped with an omnidirectional camera for homing. Their paper titled 'Image-based homing' is more aptly categorised as a landmark-based approach as their method relies on the detection and extraction of salient features in images rather than the raw images themselves to derive a homing vector. Their work was implemented on a small robot in an office environment and showed

that homing to a set of 17 distinctive places at a spacing of approximately $30\,\mathrm{cm}$ allowed a robot to traverse an environment. Hong *et al.* [1991] used a 360° grey-level view of the world obtained from a catadioptric omnidirectional camera. This view was condensed into a one-dimensional location signature which contained the required information for the robot to home-in on a series of target locations. This one dimensional signature was essentially a slice of the panospheric image taken at the horizon. Location signatures were distinguished by a list of characteristic points (in Hong's experiment 15 were used) ranked according to the gradient of change in intensity of the image measured over an angular interval in the image. Normalised correlation functions were used to match location signatures of current and target views. To deal with illumination changes, they assumed that the difference between the target signature and the current signature at the same location was some affine transformation of the brightness profile. To help deal with perceptual aliasing, Hong's robot was not able to rotate; each image taken had essentially the same perceptual frame of reference.

Much subsequent work in visual homing relies on similar visual processes, i.e. sampling a one-dimensional view at the horizon of a panoramic image, obtained either using an omnidirectional camera (see for example [Franz *et al.*, 1998a; 1998b; Möller *et al.*, 1998; Hafner and Saloman, 2002; Hafner and Möller, 2001]), or a conventional camera combined with a rotating behaviour (see for example [Gaussier *et al.*, 2000; Weber *et al.*, 1998; 1999]), or a ring of photo-diodes (see [Möller, 2000]). The advantages of sampling this one-dimensional subspace of the original image at the horizon include [Franz *et al.*, 1998a]:

1. Smaller number of pixels reduces processing time.

2. Landmarks at the horizon do not leave the field of view during rotation and translation unless they are occluded by other objects.

3. The 360° view is more robust to changes affecting only part of the view, for example moving objects.

4. Places can be encoded with a single view without having to 'stitch together' multiple representations.

Hong *et al.* [1991] assumed that the camera orientation was fixed with respect to the environment — subsequent work makes similar assumptions or use another sensor to provide a reference direction so that the world can be viewed at the same orientation.

There is very little literature on visual homing for robots outside the confines of indoor laboratories. Lambrinos *et al.* [2000] performed a simple visual homing experiment in the Saharan desert but used man-made black cylinders as artificial landmarks, which were very easy to segment from the stark background. The calculation of the homing vector was conceptually much simpler than the approaches of previous authors, and probably more effective, requiring no correspondence or matching of landmarks between views.

More recent visual homing systems adopted by the robotics community include Bianco and Zelinsky [1999] who use colour template matching to generate 'reliable' landmarks in an image which are then subsequently tracked. They initially used a monocular camera but later moved to a catadioptric omnidirectional camera, using an unwrapped image [Thompson *et al.*, 1999; 2000; Thompson and Zelinsky, 2002]. A similar region matching system was implemented by Gaussier *et al.* [2000], inspired by the biological literature. They used a neural network architecture which learned to associate actions with a set of images from around the goal area. Their system showed that navigation could proceed completely independently of any metric information.

A similar strategy, which used automatically selected landmarks, was also presented by Rizzi *et al.* [2000]. In this work, a monocular camera was used, so it must be assumed that the robot is facing (or tracking) the goal location and it's initial position is reasonably close to the goal location.

As noted earlier, corners can also be used as landmarks. Argyros *et al.* [2001] used corners detected in an image for landmarks then tracked them using a KLT corner tracker. Their method was similar in concept to Hong *et al.* [1991], homing to a sequence of 'intermediate points' to reach a goal location. Their work showed that 'corners' could

only be used for a relatively short period of traversal when using omnidirectional vision. To overcome this, they used 'intermediate points' which switched to tracking new locally available corners for landmarks, linking the intermediate points in a path eventually leading to 'home' — this is very much in the vein of the *recognition-triggered response* competency discussed in Section 2.5.5 where the robot has to follow a fixed route to the target location.

In a loose sense, visual homing is a form of the broader field of visual servoing. Visual servoing is usually associated with the control of the pose of a fixed robot's end effector using visual information in a closed feedback loop [Hutchinson *et al.*, 1996]. An error signal, being the differences in the current and desired images, is used to drive the plant in such a manner as to minimise this error term. Hutchinson *et al.* [1996] give a detailed explanation of visual servoing in the context of position-based and image-based systems. Visual homing corresponds most closely with image-based visual servoing in that features of the desired and target images are used directly to define an error signal. Position-based visual servoing uses image features to derive robot pose (via a sensor and object model) with the differences in current and desired pose defining the error signal. Thus, position-based visual servoing is more closely related to survey navigation in which models of the environment are used to determine the mobile robot pose and deduce navigation instructions [Corke, 2001].

In fact, visual navigation in general can be seen as a planar version of visual servoing. Corke [2001] presented an image-based visual servoing technique, in simulation, for an omnidirectional robot (i.e. a holonomic system in which the vehicle can independently translate and rotate) based upon an omnidirectional view of the environment. This method has many interesting parallels with visual homing and the control theory-based approach to pose stabilization. It relies on the use of a snapshot of the landmarks at a target location, from which the feature bearing error between the target view and the current view are used to drive the vehicle towards the target location. However, the 'catchment area' of the method as presented by Corke [2001] is smaller than for the landmark-based visual homing strategies.

2.6.4 Discussion

The majority of the visual homing literature assumes that the camera is either rotationally invariant (i.e. the camera on the vehicle maintains a constant heading irrespective of the base) or that a reference direction is supplied by some other means. Omnidirectional vision is commonplace in this literature. Importantly, the panoramic view provided by such a sensor prevents the need for specific control mechanisms to keep landmarks in the camera field of view.

In contrast, the pose stabilization literature using visual feedback does not assume this knowledge of camera orientation but extracts it from the image itself. However, omnidirectional imaging is much rarer in this literature. Recently published studies involve the use of monocular cameras for which the controller must be designed to ensure that landmarks are kept in the field of view. Common to the literature reviewed is a lack of research on a realistically sized vehicle using vision to solve the full pose stabilization problem. In fact, as Kelly and Nagy [2002] note, in the field of pose stabilization for nonholonomic systems, there is an abundance of theoretical work but few instances of real implementations.

In terms of vision-based systems, pose stabilization for nonholonomic vehicles has not been achieved using feedback from an omnidirectional camera. Work on real vehicles has so far been restricted to ‘laboratory-like’ environments and ‘laboratory-like’ robots.

In summary, the following major points were highlighted in this literature review:

- Many existing control strategies have not been tested outside of simulation studies which consider only the kinematics of the system. Dynamic effects, such as steering loop actuator dynamics, and additional nonlinear effects, such as input saturation, may prevent the physical implementation of these strategies.

- There are few instances of physical implementations of pose stabilization strategies on car-like vehicles.

- A growing body of literature is emerging on the application of vision for feedback

pose stabilization of mobile robots. These robots have so far been limited to small laboratory unicycle-like vehicles. Although in the visual homing literature, which solves the position rather than pose stabilization problem, omnidirectional vision is popular, there is little literature from the control community using such a sensor.

- Many existing visual pose stabilization techniques may not scale well to larger vehicles because larger vehicles require more space to execute the necessary motions for stabilization — this additional space means that vision systems must be able to sense landmarks from a longer range. Furthermore, techniques which attempt to maintain the landmarks in the image while also servoing to the desired pose may encounter problems with the larger motions required to stabilize bigger vehicles.

The major contribution of this thesis is the development and implementation of a vision based pose stabilization scheme operating on a real car-like vehicle with significant actuator dynamics, also dealing with nonlinearities such as input saturation.

2.7 Conclusions

The fundamental behavioural prerequisites for any of the navigation competencies presented in Table 2.1 are the ability to move through an environment and to recognise a goal when it is acquired. For flexibility and manoeuvrability purposes, many mobile robots to date have kinematics which allow them to rotate about their own vertical axis. If a goal location is attained, such robots can simply rotate to provide the correct orientation, and hence pose. This approach is suitable for indoor applications, where robotic technology will probably be implemented on 'new' vehicle designs. However, industrial applications of mobile robotics will probably involve retro-fitting systems to existing vehicles or vehicle designs. Many industrial vehicles have kinematic constraints which prevent them from rotating on their own vertical axis and thus, the attainment of a particular pose in an environment can only be achieved with proper path selection. If a mobile robot is to perform useful tasks, interacting with the environment, it must have the ability to reach a destination *with a particular orientation*, i.e. it must be able to stabilize to a pose.

A car-like vehicle is constrained to moving in a series of arcs and lines and thus the attainment of a particular pose is highly dependent on the path taken. For nonholonomic systems with less control inputs than generalised coordinates, such as car-like vehicles, Brockett [Brockett, 1983] showed that there are no smooth, state feedback stabilization laws. The pose stabilization problem for car-like vehicles can be solved by open or closed-loop strategies. The open-loop strategies attempt to find a bounded sequence of control inputs which take the vehicle to the desired pose, given an initial pose. These usually operate in conjunction with a motion planner, which is normally associated with the classical, absolute navigation methods. Such systems are computationally expensive, and in the event of a disturbance due to inaccuracies in the environmental or vehicle model, a new plan needs to be formulated. In contrast, the closed-loop systems operate by minimising the error between sensed and desired pose, through the application of some feedback control law. Brockett's result requires that the control law be discontinuous, time-varying, or a combination of both. An alternative is to relax the constraint on orientation and stabilize to a position rather than a pose. This has been the strategy used by much of the biologically-inspired homing strategies.

The work of this thesis unites results from the visual homing and pose stabilization literature, using an omnidirectional camera as the primary sensor. In addition, a comparatively larger vehicle will be used than found in previous studies and the control must be able to cope with the associated dynamics and limitations in the steering and velocity loops.

Chapter 3

The Test-bed: the Autonomous Tractor

The Autonomous Tractor is a ride-on mower which has been fitted with an array of actuators, sensors and a computer system allowing the implementation and testing of control and navigation algorithms. The vehicle is car-like and is driven by a petrol engine which also supplies power to all on-board systems. The vehicle's design is such that it can operate in three modes: manual, remote or automatic. The vehicle can operate as a stand-alone unit but can also communicate to a network of computers through a wireless LAN connection. For simulation purposes, the vehicle's kinematics are represented by the bicycle model, while the velocity and steering loops are modelled using experimentally derived dynamic models which also encompass non-linearities such as input saturation.

3.1 Introduction

An important aspect in field robotics is the energy source for the robot. Many current mobile robots rely on chemical batteries or a tethered connection to an external power supply [Debenest *et al.*, 2002]. Mobile robots consume a considerable amount of power for actuating levers, powering sensors, and running communications and computer systems. In the case of battery-powered robots, the normal tasks of the robot quickly consume the available energy, limiting operating times. In the case of tethered robots, operating times are essentially unlimited but manoeuvrability and range of operation are limited by the tether. In addition, both methods require significant electrical infrastructure, either to

Figure 3.1: The Autonomous Tractor.

charge batteries or to provide the external power source transmitted by the tether [Debenest *et al.*, 2002]. Such infrastructure may not exist in the desired operating environment.

The research platform developed and used in this thesis is a Toro ride-on mower, a rear-wheel driven, front steered vehicle with car-like kinematics. This vehicle is representative of many existing vehicles in industry yet is small enough to enable experiments to proceed in relative safety. It is also driven by a petrol engine which supplies all the necessary power for the on-board systems. Figure 3.1 shows the automated vehicle, which is referred to as the Autonomous Tractor (AT).

The modified vehicle is capable of three modes of operation:

- **Manual** — the vehicle is driven manually by a person sitting on the vehicle.
- **Remote** — the vehicle's controls are electrically actuated with demands supplied by a person using a hand-held remote control transmitter.

- **Automatic** — the vehicle's controls are electrically actuated with demands supplied by the on-board computer system.

The design brief for the automation of the vehicle called for minimal obvious or visible modification — it should look largely unchanged. Further to this, the vehicle's levers should move when operated by the automation system and the automation system should not preclude the manual operation of the vehicle.

This chapter outlines the design of the automation system and also describes the kinematic and dynamic modelling of the vehicle.

3.2 The original vehicle

The ride-on mower is a $16\,\mathrm{hp}$ Toro Wheelhorse on loan to CSIRO Manufacturing and Infrastructure Technology from the School of Civil and Environmental Engineering, Faculty of Engineering, at the University of New South Wales. It is an Ackerman steered vehicle (i.e. it is a car-like, nonholonomic vehicle), capable of forward and reverse motion. The vehicle's rear wheels are powered by a hydraulic drive (hydro-drive), which in turn is pulley driven by the petrol engine. The hydraulic drive permits proportional control of the vehicle's speed. Top speed in the forward direction is approximately $12\,\mathrm{kmh}^{-1}$. For the initial stage of automation, the mower blades have been removed. However, the design of the automation system is such that the blades can be re-fitted when desired.

Figure 3.2 shows the main vehicle controls. When driven by a human operator (**manual** operation), the vehicle is controlled using the:

- Steering wheel — Provides control over the heading of the vehicle.
- Forward-reverse (speed) pedal — Operated by the right foot, provides control of the vehicle's speed and direction of motion.
- Brake pedal — Operated by the right foot (for safety reasons the foot has to be removed from the speed pedal to operate the brake). This brake disengages the belt

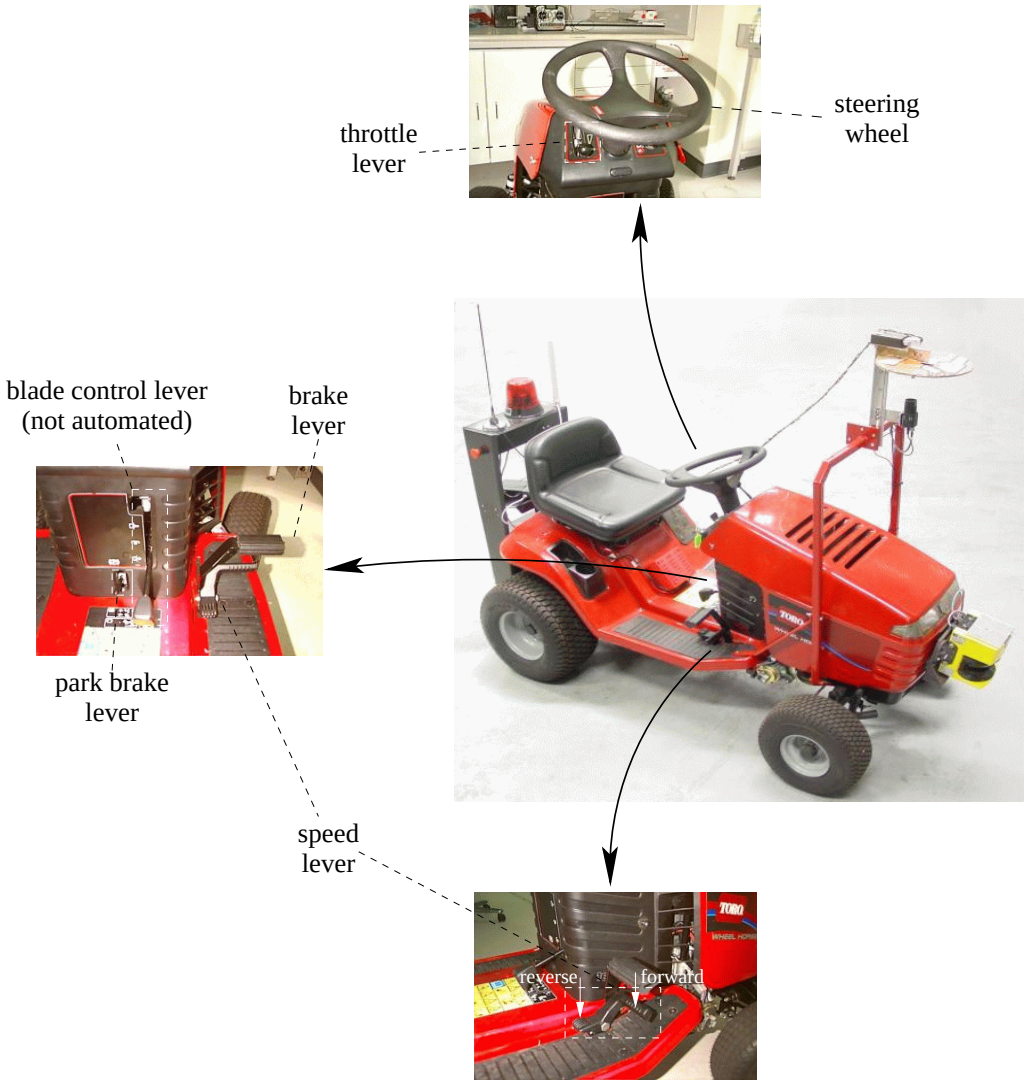


Figure 3.2: The AT's control levers.

which drives the hydro-drive, and also operates a series of friction pads on one of the rear axles.

- Park brake pedal — Mechanically locks the brake pedal in the 'on' position once the brake pedal has been depressed.

- Throttle — A lever operated by the left hand. Essentially it is a three position lever: slow, fast, and choke, although there is a limited range of proportionality between the slow and fast positions. This lever operates a cable connected to the engine

carburettor and sets a nominal engine speed.

- Blade engage — This lever engages the mowers blades (not automated).
- Blade adjust — This lever adjusts the cutting height of the blades (not automated).

The speed pedal controls the vehicle's forward-reverse direction and rate of motion in an approximately proportional manner. The throttle sets the vehicle's engine speed, which effects the amount of power transmitted to the hydraulic drive. Higher engine speed means more power to the hydraulic drive which, in turn, means a higher vehicle speed for a given speed pedal position. Under normal operation, the throttle is set on high.

3.3 Control system design

The AT has been fitted with an array of actuators, sensors, and a computer system, allowing complete control of the vehicle's function. To satisfy the requirement of not precluding manual operation, a means of disengaging the steering automation system has also been fitted, resulting in a total of six axes of control.

An overall schematic of the entire AT control system is shown in Figure 3.3. The system consists of a set of HC12 microprocessors which control the individual actuators for each axis of control. The HC12 processors can receive demands from either the hand-held radio transmitter or the computer system. However, computer system demands can only be sent to the HC12 system if the computer senses that the *key-switch* is in **automatic-ready** mode, the *automatic* pushbutton has been pressed, and the software switch for the particular axis of control has been activated.

The following description is divided into two main sections. First, the low-level control system including the HC12 system, actuators and low-level feedback sensors are described. Second, the computer system hardware and software is presented. Video footage of the AT's first remote control test is included in the supplementary material contained in Section A.1.

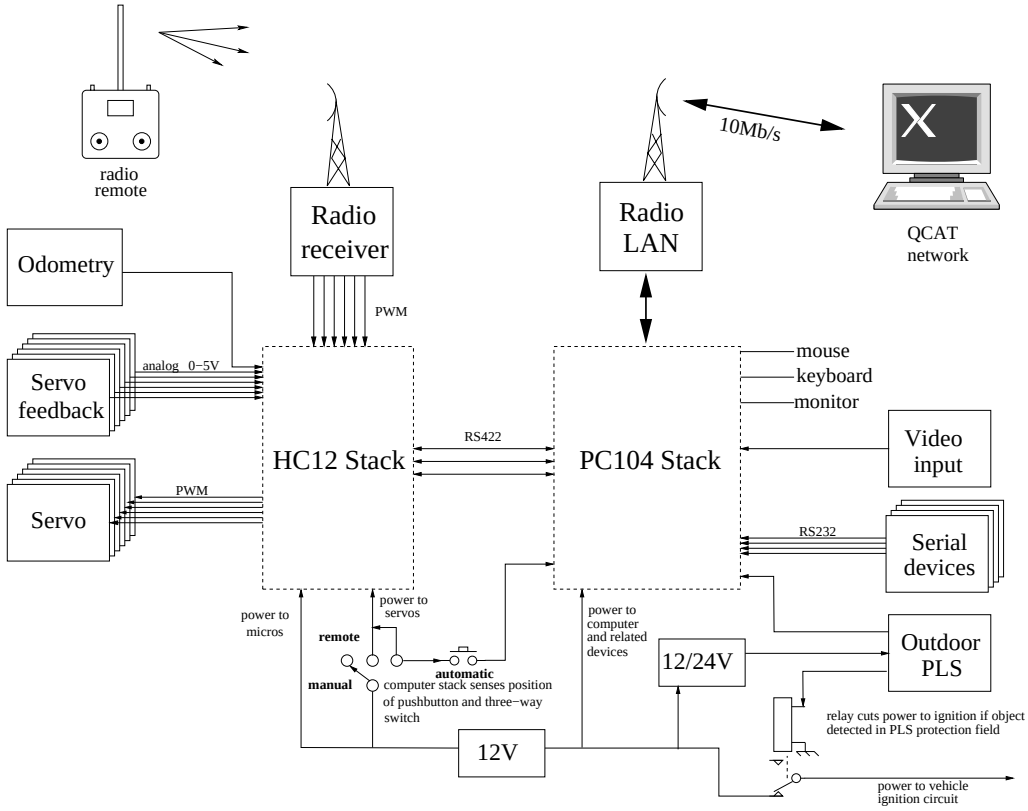


Figure 3.3: A schematic of the entire control system on the AT. Included is the odometry system which was fitted with the rest of the control system.

3.3.1 Low-level control

The six vehicle inputs are monitored and controlled by a stack of three, independent Motorola HC12D60 microprocessors, each of which controls two axes. These are grouped as follows:

- steering / engage
- brake pedal / park brake
- throttle lever / speed pedal

Figure 3.4 shows a schematic of the HC12 stack. Each axis of control has four common elements:

1. An actuator.

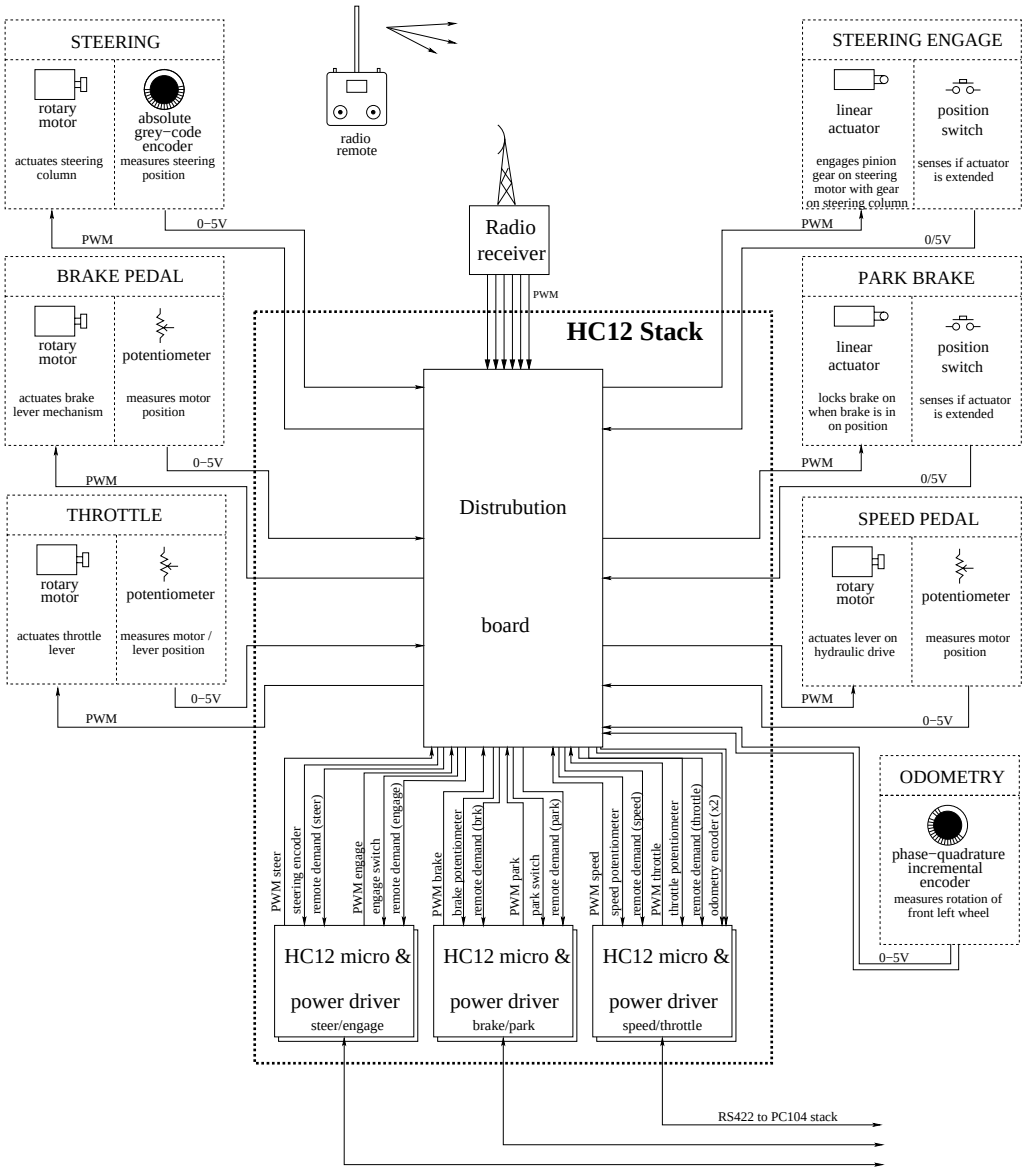


Figure 3.4: A schematic of the HC12 stack. The stack consists of 3 pairs of microprocessor and driver boards, each of which is responsible for two axes of control.

2. A feedback sensor (actuator or lever position).

3. A controller.

4. A means of providing a control demand.

A schematic of an individual HC12 microprocessor / power drive pair is illustrated in Figure 3.5 (a), also showing the common elements of each control axis. Feedback for

each axis of control is in the form of an analogue or logical $0 - 5\,\mathrm{V}$ signal. Control demands to the HC12 are provided by the radio remote or via a serial link to the on-board computer. When the system is in **remote** mode, demands are supplied from the hand-held radio transmitter. When the system is in **automatic** mode, each individual control axis can be controlled by either the computer or the remote. Switching between the two demand sources is selected by the computer, with the default source being the remote control, see Figure 3.5 (b). When the AT is in **remote** or **automatic mode**, the HC12's compare the desired demands with the feedback sensor, applying a control law to the resulting error signals. These control inputs are then amplified and converted to high frequency PWM signals (simulating a variable DC voltage) which drive the actuators. All the control actuators are nominally $12\,\mathrm{V}$.

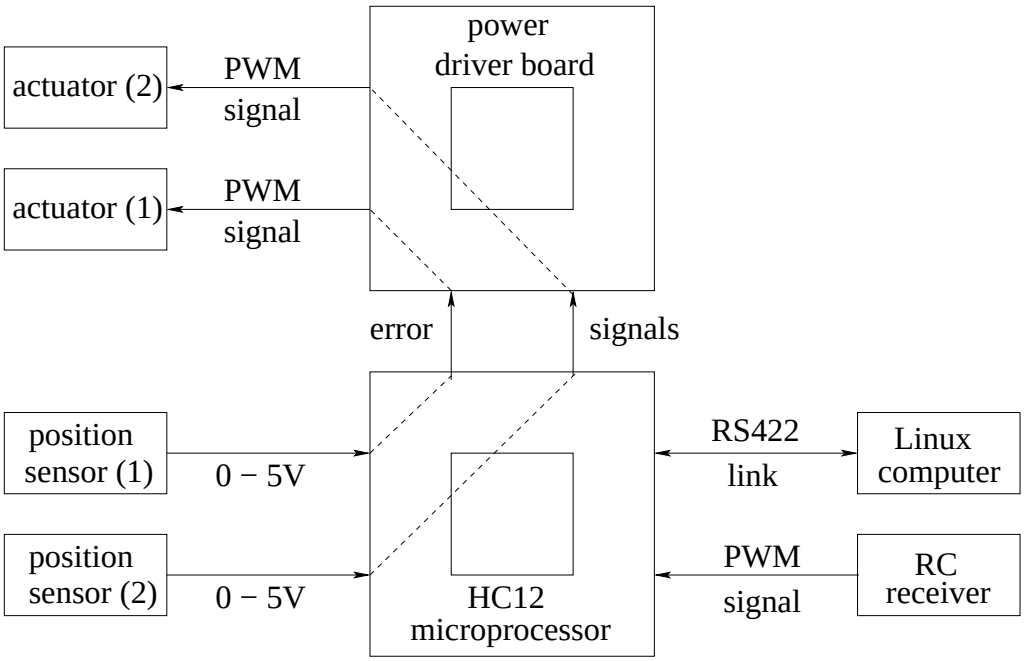
Each axis of control has different requirements in terms of the type of feedback, actuator, and control law used. The following sections briefly describe the individual axes of control, including a section on the other modifications made to the vehicle. A pictorial overview of the modifications made to the vehicle is given in Figure 3.6.

3.3.1.1 Steering

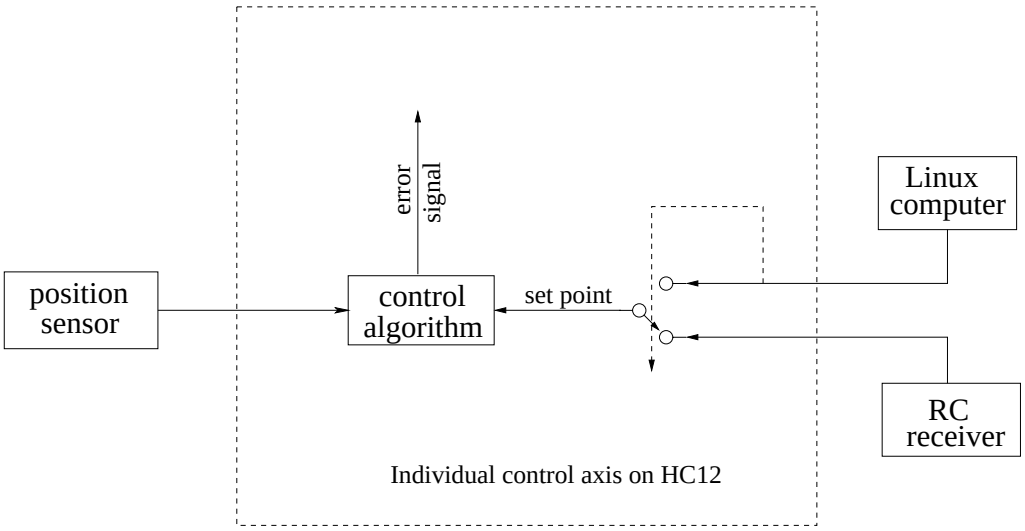
The original steering mechanism on the AT consisted of a gear and linkage system operated by turning the steering wheel. To automate the steering, the steering column has been fitted with a spur gear which is driven by a rotary motor-gear set[1], as shown in Figure 3.7. Steering angle feedback is provided by an absolute encoder mounted on the pivot point of the front left-hand wheel, see Figure 3.8 (a).

The geometry of the steering linkage is such that the left and right-hand wheels are rotated equally. Because of this, the vehicle is not strictly Ackerman steered, and the front wheels will slip slightly when the vehicle is turning. However, the vehicle is only capable of relatively slow speeds and the slip has not been found to be significant. The steering angle range is approximately $\pm 30^{\circ}$. When the steering actuation system is engaged, and the system is in **remote** or **automatic** mode, the steering angle of the vehicle is controlled

[1] A wind-screen wiper motor from an automobile.



(a) Each HC12 microprocessor controls two of the control axes.



(b) The source of the demand on the HC12 is switched by the computer. The default is remote control. Each axis can be switched individually.

Figure 3.5: Data flow through an individual HC12 microprocessor / power driver pair.

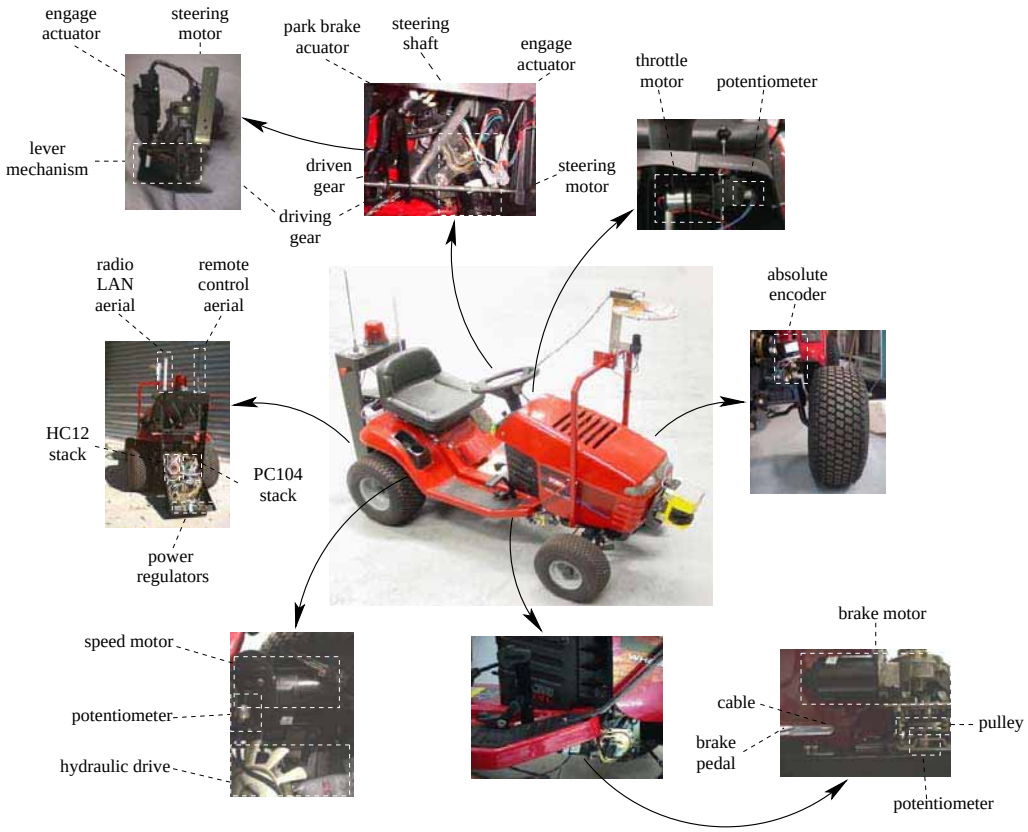
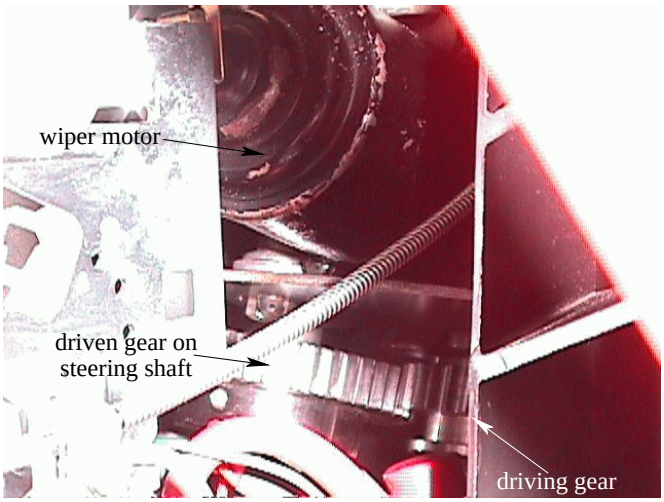


Figure 3.6: The control system as fitted to the AT illustrating where particular items are situated.

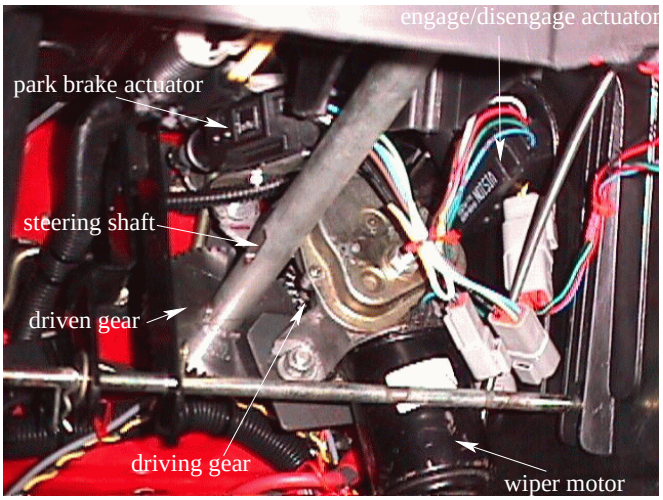
by a PID loop implemented on the HC12 microprocessor.

3.3.1.2 Steering actuator engage

Because of the high gearing in the steering actuation system, **manual** operation of the steering wheel is difficult when the motor-gear set is engaged. Thus, a mechanism has been devised which allows the gear on the motor to 'disengage' from the gear on the steering column. Engaging and disengaging of the driving gear is achieved by sliding the gear along the splined output shaft of the rotary motor. The motion of this gear along the splined shaft is controlled by a lever attached to a two-position, linear actuator from a standard automobile central-locking mechanism, as shown in Figure 3.8 (b). When engaged, a spring holds the gear in position. When disengaged, a set of rare-earth magnets holds the driving gear clear of the gear on the steering shaft. To ensure meshing of the gear



(a) Side view of the steering actuation system.

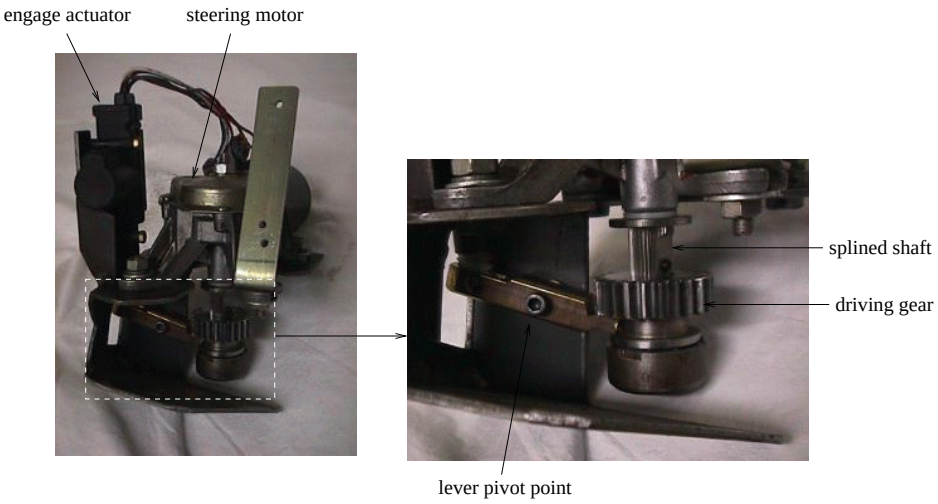


(b) Top view of the steering actuation system. Also shown is the steering engage/disengage actuator and the park brake actuator.

Figure 3.7: The steering actuation system. Note how ‘cramped’ this area is.

teeth, on an engage / disengage command, the linear actuator is pulsed, giving a relatively high frequency back-and-forth motion of the gear along the shaft. Simultaneously, the steering actuator is pulsed, giving the driven gear a back and forth motion. This tactic ensures that the gear teeth mesh on engagement, and de-mesh on disengagement.

(a) The absolute encoder measures the rotation of the front left-hand wheel.



(b) The steering motor and the engage/disengage mechanism. The driving gear slides along a splined shaft, depending on the input from the engage mechanism.

Figure 3.8: The absolute encoder and steering disengage mechanisms.

Feedback is provided by a switch internal to the actuator mechanism indicating the state of the actuator, extended or retracted. When in **remote** or **automatic** mode, the steering actuation system can be engaged or disengaged with the state of the actuator controlled by a state-machine on the HC12 microprocessor.

3.3.1.3 Brake pedal

Power is transmitted from the AT's petrol engine to its wheels via a pulley system connected to the hydraulic drive. When the brake is depressed, the initial portion of the pedal stroke takes power off of the hydraulic drive by slackening the drive belt. The final portion of the pedal stroke operates a mechanism which applies a set of friction pads to one of the output shafts of the hydraulic drive, i.e. one of the driving wheel axles. To actuate the brake pedal, a rotary motor[2] drives the pedal through a simple cable-and-pulley system, see Figure 3.9. Brake actuation is not fail-safe — if the actuator or cable fails, the brake will not be applied.

Pedal position feedback is provided by a potentiometer connected to the output shaft of the brake rotary motor. When in **remote** or **automatic** mode, a PID loop on the HC12 microprocessor controls the motor position.

3.3.1.4 Park brake

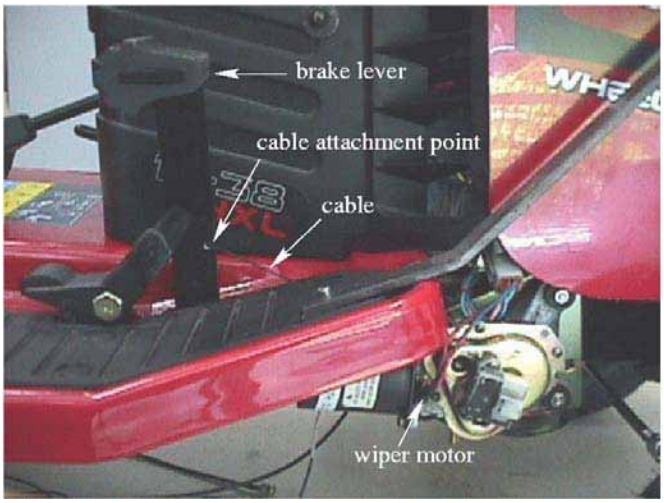
Manual operation of the park brake lever simply locks the brake pedal in the 'ON' position. This lever can only be used when the brake is 'ON'. Actuation to this lever is again provided by a two position linear actuator from a standard automobile central-locking mechanism.

Feedback is provided by a switch internal to the actuator mechanism which indicates the state of the actuator, extended or retracted. When in **remote** or **automatic** mode, a state-machine on the HC12 controls the park brake. This state-machine first activates the brake pedal, then applies the park brake. Figure 3.7 (b) illustrates where this actuator is fitted on the AT.

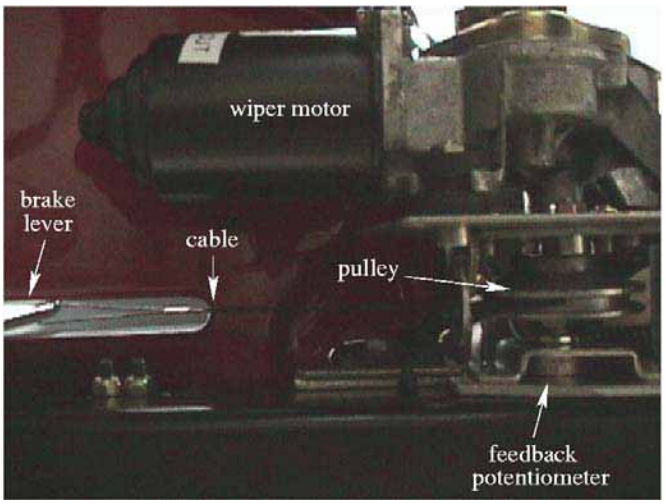
3.3.1.5 Speed pedal

The petrol engine supplies power, through a set of pulleys, to the hydraulic drive, which in turn powers the rear wheels. Forward and reverse with proportional speed control, is provided by a foot-pedal which, through a linkage system, moves a lever on the hydraulic

[2] A wind-screen wiper motor from an automobile.



(a) Side view of the brake actuation system.

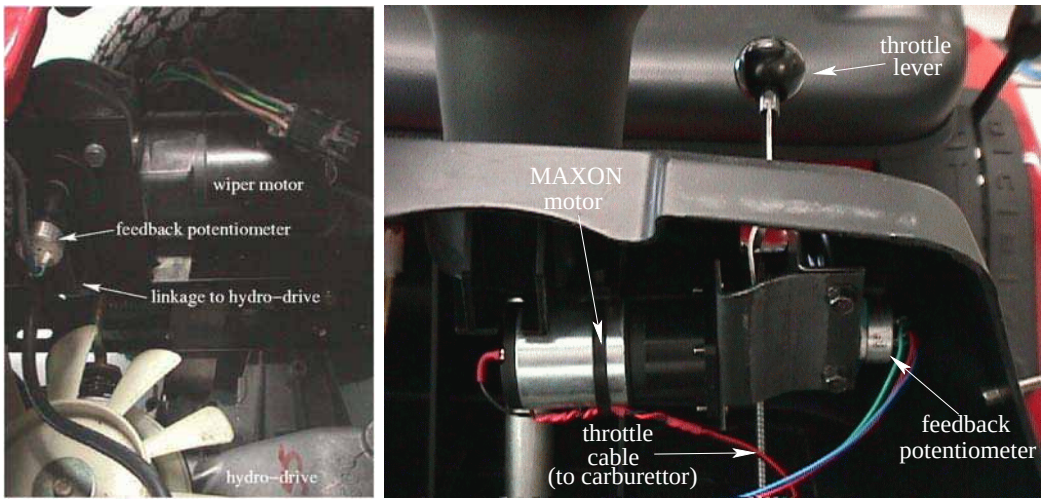


(b) Bottom view of the brake actuation system.

Figure 3.9: The cable-and-pulley system which actuates the brake pedal.

drive (controlling the separation distance of a fluid coupling — separation distance determining level of power transmission). An automatic spring return is fitted to the mechanism so that when not in use, the hydraulic drive returns to the neutral position and the vehicle halts.

Automation of the speed pedal directly was not possible due to space constraints.



(a) The wiper motor used to control the vehicle's speed and direction.

(b) The motor actuating the throttle is coupled directly to the axis of rotation of the throttle lever.

Figure 3.10: The motors used for the control of the speed pedal and throttle lever.

Instead, the hydraulic drive lever is actuated by a rotary motor[3] through a simple two-link mechanism. This system also actuates the speed pedal. Feedback to the motor is provided by a potentiometer connected directly to the motor shaft, see Figure 3.10 (a). When in **remote** or **automatic** mode, a PID loop on the HC12 microprocessor controls the motor position.

3.3.1.6 Throttle

The throttle lever operates a cable connected to a butterfly valve on the carburettor. There are three throttle positions: choke, fast, and slow. The existing throttle lever and bracket were replaced with similar items that could accommodate a small MAXON motor-gear head combination driving a shaft at the pivot point of the throttle lever. Throttle lever position feedback is provided by a potentiometer coupled to the pivot point of the throttle adjustment lever and motor output shaft. Figure 3.10 (b) shows a photograph of the throttle control system. When in **remote** or **automatic** mode, a PID loop on the HC12

[3] Again, a wind-screen wiper motor from an automobile.

microprocessor controls the throttle lever position.

3.3.1.7 Safety systems

The first safety system is the *key-switch* and *automatic* button system. This system allows switching between the modes of operation, **manual**, **remote** or **automatic**. The *key-switch* has three positions. In the first, or **manual** position, no power is supplied to the actuators and hence the vehicle cannot be controlled by the HC12 system. In the second, or **remote** position, power is supplied to the actuators but the HC12 stack can only accept demands from the hand-held radio transmitter. In the third, or **automation-ready** position, power is again supplied to the actuators and the HC12 can accept demands from the hand-held radio transmitter. A safety card on the PC104 stack senses the position of the *key-switch* and the *automatic* button. If the *key-switch* is in the **automation-ready** position and the *automatic* button has been pressed, then the system is in **automatic** mode. Demands can then be sent to the HC12 stack from the computer. Additionally, the computer monitors software switches which allow each axis of control to be individually switched to **automatic** mode if the safety card conditions described previously are met. For example, in the majority of experiments conducted in this research, when the system was in **automatic** mode, control of the speed pedal and steering wheel was given to the computer, while the user maintained control of the vehicle's braking functions through the hand-held radio transmitter.

A SICK Proximity Laser Scanner (PLS) has been mounted to the front of the vehicle. This sensor provides a 180° range sweep at 0.5° intervals up to a range of $50\,\mathrm{m}$. It also has relays which are activated when an object enters custom programmed 'fields' in the sensor's field of view. On the AT, this sensor's primary purpose is collision avoidance and its fields have been programmed to activate a relay. This relay cuts the vehicle's ignition on sensing an object within a 120° forward facing arc at $1.5\,\mathrm{m}$ from the vehicle, as shown in Figure 3.11. Of course, because the sensor is facing forward, this feature only provides protection from frontal collisions. Range data from this sensor is also fed into the AT's computer system and can be accessed for navigation purposes.

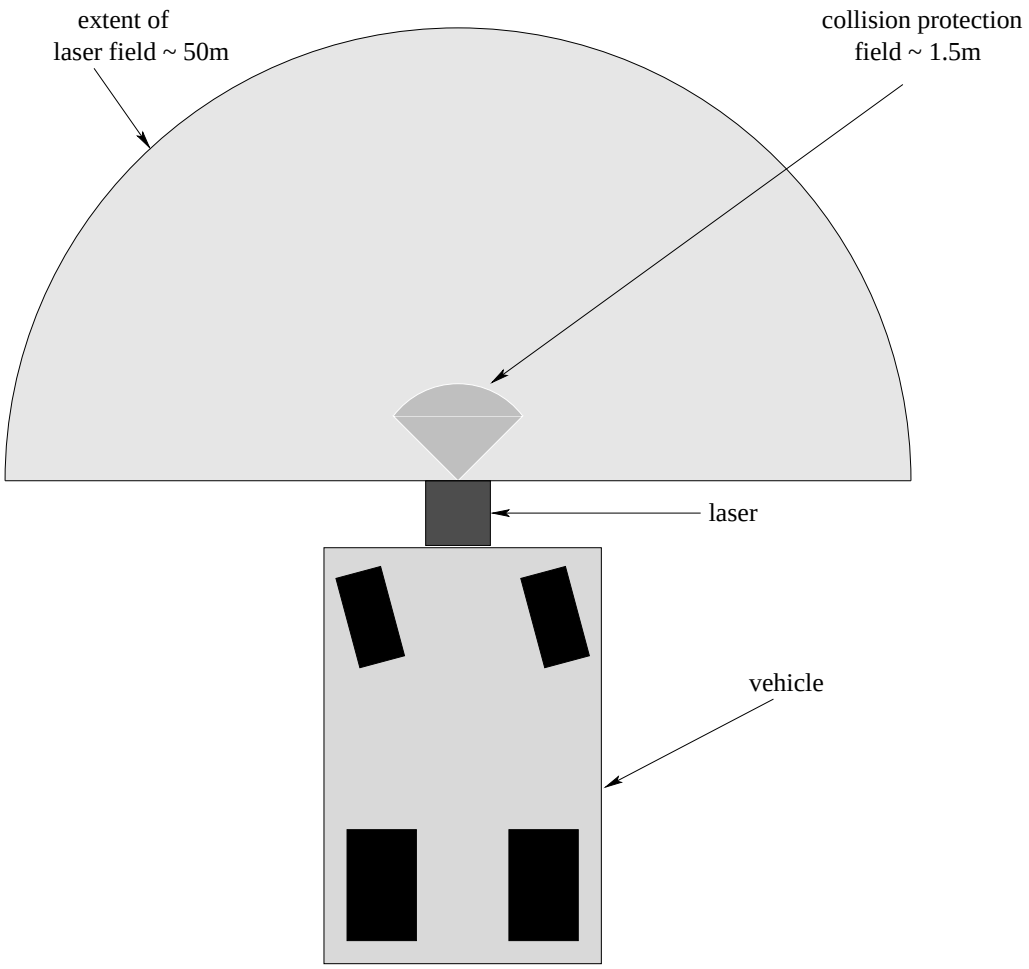


Figure 3.11: The SICK laser field of view showing the collision detection field.

Two Emergency Stop (E-Stop) buttons have been fitted on each side of the rear of the vehicle, see Figure 3.13. These buttons are also connected to a relay which immediately cuts the vehicle's ignition on activation of either of the E-Stops. Additionally, a rotating red warning light has been fitted to the rear of the vehicle, see Figure 3.13. It is activated when the key-switch described earlier is in **remote** or **automation-ready** mode.

3.3.1.8 Other modifications

Odometry

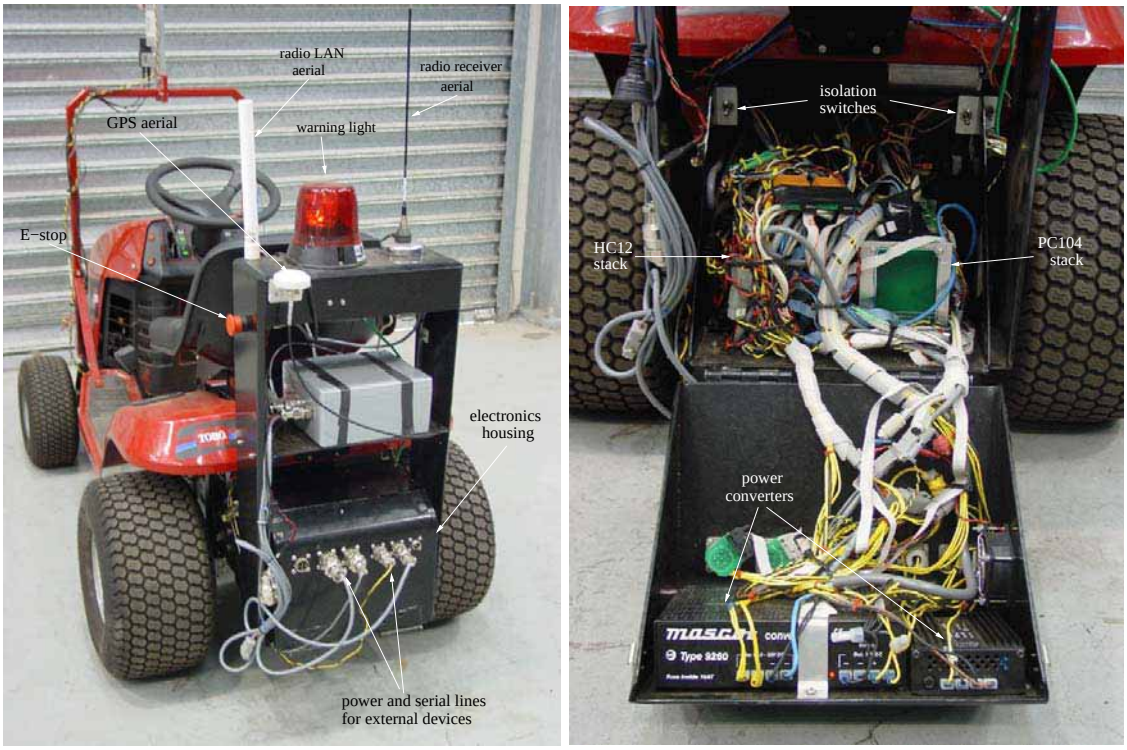
An odometry system has been fitted to the tractor. It consists of a pair of proximity switches which are used to monitor the revolution of a specially manufactured disc (two

Figure 3.12: The odometry system fitted to the tractor. Used together with the absolute encoder fitted to the steering system, relative heading and distance information can be determined.

switches are required to determine which direction the disc is rotating). This disc has 40 'teeth' on its face. The disc has been fitted to rotate about the same axis as the front-left-hand wheel, see Figure 3.12. This system is, in effect, a rather coarse phase-quadrature incremental encoder, its main purpose in this work is to provide feedback of the vehicle's speed. The proximity sensors measure the rising and falling edges of the disc's teeth, resulting in an angular resolution of $160\ \frac{\text{counts}}{\text{revolution}}$.

Heavy-duty alternator

Power consumption tests revealed that the AT's original alternator, which supplied $3\,\text{Amps}$ at the vehicle's nominal operating voltage of $12\,\text{Volts}$, was not sufficient to supply power to the retro-fitted items and maintain the charge to the vehicle's existing on-board battery. When in **remote** or **automatic** mode, this would in effect limit the vehicle's operating time. Rather than upgrading the battery to extend the vehicle's operating time, the alternator was upgraded to $16\,\text{Amps}$, which is more than sufficient to maintain the battery and supply power to all of the retro-fitted electrical items.



(a) The rear of the vehicle showing the electronics housing.

(b) The H12 and PC104 stacks.

Figure 3.13: The electronics housing and other equipment fitted to the rear of the vehicle.

Miscellaneous modifications

Other modifications include the fitting of various brackets and housings to accommodate the electronics, and to mount motors and sensors, as shown in Figure 3.13. An electronics box, which holds the HC12 stack, the PC104 stack and various other electrical items, has been fitted to the rear of the vehicle. Above the box is a series of shelves to which the radio LAN and remote control aerials have been fitted, along with the warning light and E-Stop buttons.

Various navigation sensors have also been fitted to the vehicle including:

- EyeSee 360 omnidirectional camera: the primary navigation sensor in this work.
- CROSSBOW high-speed orientation sensor: used for a compass direction.

Figure 3.14: Mounting points of the PLS, omnidirectional camera, and the orientation sensor.

- Real Time Kinematic (RTK) GPS: used for a ground truth in this work.

The aerial for the optional GPS is fitted to the electronics housing, as shown in Figure 3.13, while the GPS unit is housed on the shelf (the grey box in Figure 3.13). The camera and the orientation sensor perform the primary sensing for the work in this thesis and are described in more detail in Section 4.3 — their mounting locations are shown in Figure 3.14, together with the location of the SICK PLS.

3.3.2 Low-level sensing

The HC12 stack receives feedback on:

- brake pedal actuator position: a potentiometer with a resolution[4] of $1024\ \frac{\text{counts}}{\text{revolution}}$
- park brake actuator position: a two-position switch, ON or OFF
- speed pedal actuator position: a potentiometer with a resolution of $1024\ \frac{\text{counts}}{\text{revolution}}$
- throttle position: a potentiometer with a resolution of $1024\ \frac{\text{counts}}{\text{revolution}}$
- steering angle: an absolute encoder with a resolution of $4096\ \frac{\text{counts}}{\text{revolution}}$
- steering mechanism engage position: a two-position switch, ON or OFF
- front left-hand wheel rotation (odometry): a phase-quadrature encoder with a resolution of $160\ \frac{\text{counts}}{\text{revolution}}$

all of which are sent to the computer.

In this work, anti-clockwise rotation of the steering wheels is defined as positive for which the vehicle will undergo a left-hand turn when travelling forwards, and a clockwise rotation is defined as negative for which the vehicle undergoes a right-hand turn when travelling forwards.

The Ackerman configuration on the vehicle is designed to limit slip of the front wheels when the vehicle is undergoing a turn by rotating the inner-front-wheel on the turn slightly more than the outer-front-wheel [Everett, 1995]. As described in Section 2.6.2, car-like vehicles are commonly modelled by the so-called bicycle model in which the wheels on the front and rear axles are collapsed to a single 'virtual' wheel on each axle. Because the vehicle speed and steering measurements are that of the front left-hand wheel, corrections need to be made to calculate the steering angle at the mid-point of the front-axle and the vehicle speed at the mid-point of the rear-axle. For example, if the vehicle undergoes a left-hand turn, the measured speed of the front-left-wheel will be slightly slower than

[4]Determined by A/D conversion.

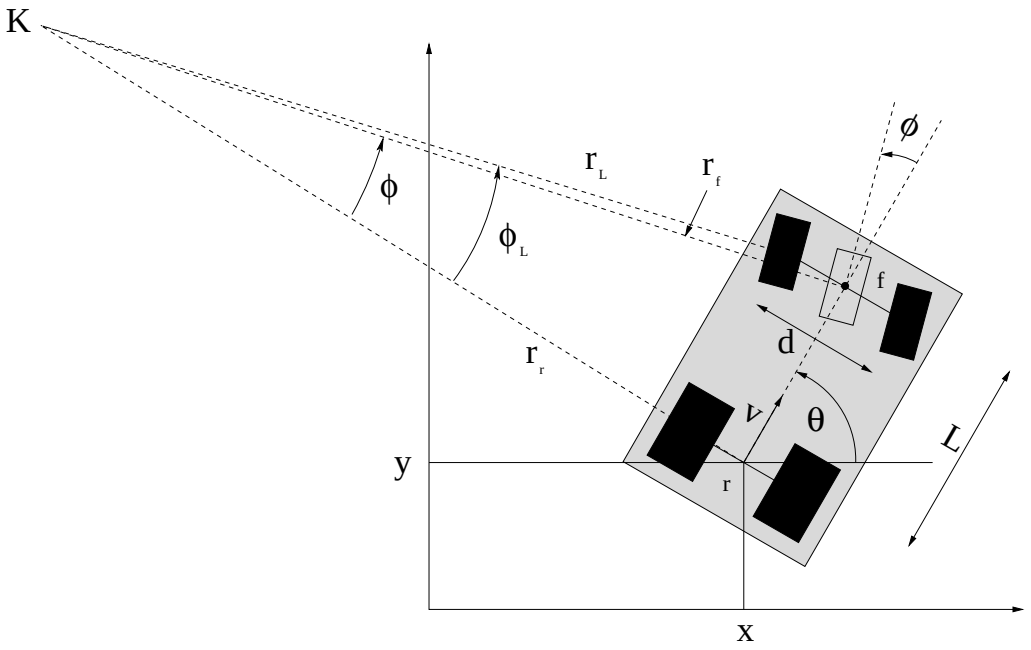


Figure 3.15: The AT's steering geometry.

the true speed of the vehicle. Likewise for a right-hand turn, the speed measured will be slightly higher than the actual vehicle speed.

The steering measurement is first converted from an encoder reading to an angle in radians, referenced to a previously determined centre position:

$$\phi_L = \frac{(encoder_{centre} - encoder_{current})2\pi}{4096} \tag{3.1}$$

with an anti-clockwise rotation defined as positive. This sensing arrangement gives a resolution on steering angle of $\pm 0.0879°$. The steering measurement on the left wheel is translated to the mid-point of the front axle using the well-known Ackerman equation [Everett, 1995] (recalling that turning the wheels to the left is defined as a positive steering angle):

$$\phi = \arctan\left(\frac{1}{\cot\phi_L + \frac{d}{2L}}\right) \tag{3.2}$$

where ϕ_L refers to the angle as measured at the left wheel, $d = 0.74\,\mathrm{m}$ is the separation distance of the wheels on the front-axle and $L = 1.2\,\mathrm{m}$ is the distance between the front and rear axle.

The odometry measurement allows calculation of the vehicle's speed and relative distance travelled. The odometry measures the rotation of the front left-hand wheel and is translated to distance travelled through knowledge of the wheel diameter:

$$distance = \frac{(encoder_{current} - encoder_{ref})2\pi r_w}{160} \tag{3.3}$$

where $r_w = 0.175\,\mathrm{m}$ is the rolling wheel radius. Velocity can be calculated through knowledge of the measuring time increment between $encoder_{ref}$ and $encoder_{current}$:

$$v_L = \frac{distance}{\Delta t} \tag{3.4}$$

At a frequency of $1\,\mathrm{Hz}$, the resolution on velocity is $\pm 0.0069\,\mathrm{ms}^{-1}$. However, for control purposes, velocity needs to be measured at a higher frequency, resulting in poorer resolution. A frequency of $2\,\mathrm{Hz}$ was found to be a good compromise, providing a velocity resolution of $\pm 0.0137\,\mathrm{ms}^{-1}$.

The velocity measurement also requires correction to translate the measurement to the mid-point of the rear axle. Again referring to Figure 3.15, using the fact that when turning, each point on the vehicle has the same angular velocity:

$$v_f = \frac{\sin\phi}{\sin\phi_L} v_L \tag{3.5}$$

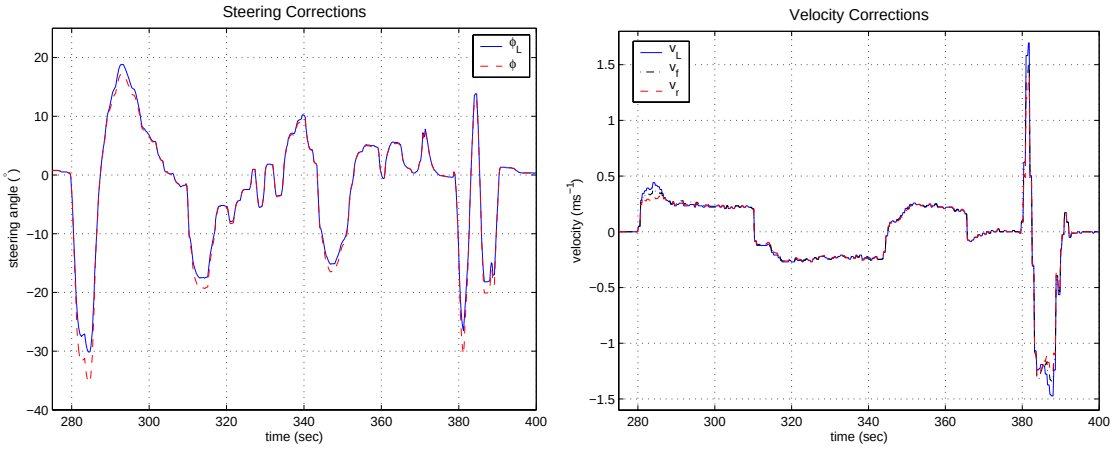
where v_f refers to the velocity of the 'virtual' wheel at the midpoint of the front axle, and v_L refers to the velocity as measured at the left-hand front wheel, and ϕ is the effective steering angle calculated using Equation 3.2.

The velocity of the mid-point of the rear axle, shown as point 'r' in Figure 3.15, is easily calculated by again recognising that every point on the vehicle rotates about point K with an equal angular velocity:

$$v_r = v = v_f \cos\phi \tag{3.6}$$

Figure 3.16 compares measured speed and steering angle at the left-hand wheel with the corrected values showing that the importance of the correction is fairly limited for the majority of the range of operation for these vehicle inputs. In practice, these corrections

have been found to make little difference to the control of the vehicle as the object of the thesis is to use feedback control which, if effective, will compensate for inaccuracies in the plant estimation.



(a) Steering angle correction. The subscript 'L' refers to the measurement at the front-left wheel.

(b) Velocity correction. The subscript 'L' refers to the measurement at the front-left wheel, 'f' refers to the virtual wheel at the midpoint of the front axle and 'r' refers to the virtual wheel at the midpoint of the rear axle.

Figure 3.16: Steering angle and velocity correction. Note that the corrections are relatively minor except for the case of large inputs.

3.3.3 Computing

The PC104 stack is a stripped down, miniaturised personal computer, built for operation in rugged environments. It is the 'brains' of the navigation system, passing instructions to the HC12 stack which in turn controls the vehicle's actuators.

3.3.3.1 Hardware

The PC104 stack is connected by radio LAN to the CSIRO Pinjarra Hills computing network allowing remote communication with the AT's computer system. The AT's computer runs under the Linux operating system. Figure 3.17 shows a schematic of the computer system together with its connection to the HC12 stack. The essential components

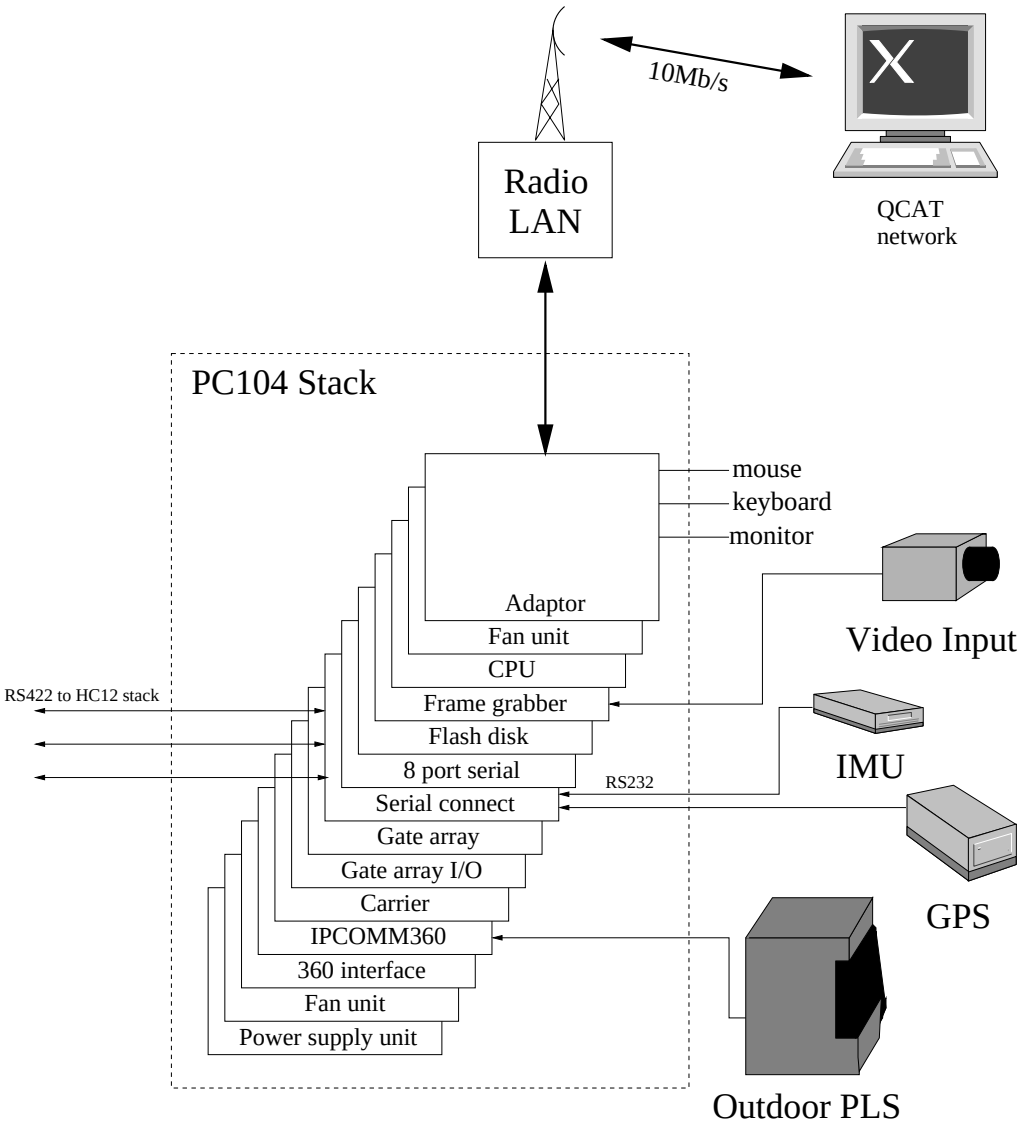


Figure 3.17: Schematic of the AT's PC104 stack.

of the computer stack are:

- a CPU — Crusoe $800\,\mathrm{MHz}$ processor
- hard-drive — a $256\,\mathrm{Mb}$ flash disk (solid state hard-drive)
- frame grabber — allows connection of two cameras
- serial port board — allows connection of up to eight serial devices

- adaptor board – allows connection of a keyboard and monitor to the PC104 stack

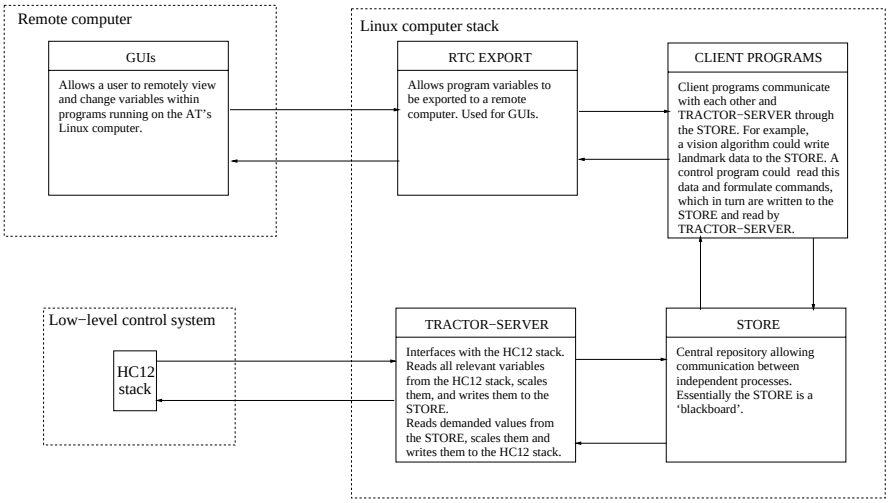
Currently, three of serial ports on the 8-way serial board are used for communication with the HC12 stack (i.e. one for each HC12 processor in the stack). The remaining ports are available for connecting sensors, currently an orientation sensor and a GPS. An omnidirectional camera is connected to the frame-grabber, while data from the SICK laser is also made available to the computer.

3.3.3.2 Software

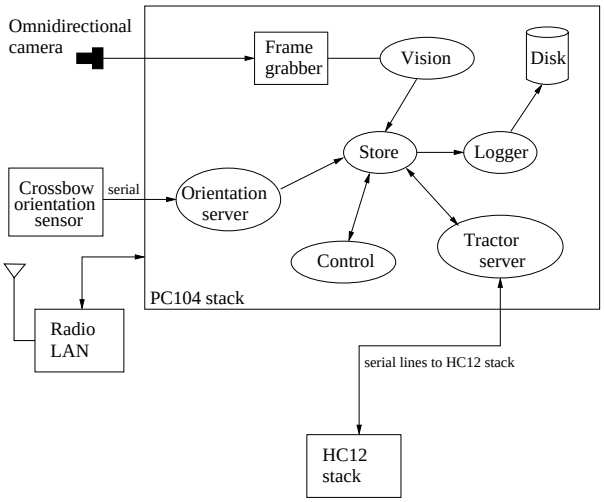
At the heart of the AT's software system is the **store**. The **store** provides a shared memory block which is used as a data repository. Client processes can create data structures in this repository and then read or write data to these structures. This allows communication between individually running processes, provided each of the processes connects to the **store** and registers a variable of the same type with the same handle. A logger client process allows time-stamped recording of data written to the **store**.

Figure 3.18 (a) illustrates the communication mechanisms available to client processes. For communication with the HC12s, **tractor-server** reads data coming from the HC12s, applies scaling if required and places it in an appropriate structure, here called `tr_pos`, which is then written to the **store** for use by other processes. Client processes can read the state of the HC12s by accessing `tr_pos` in the **store**. These processes can then make command decisions and write demands to the **store** in the appropriate structure, here called `tr_demand`. The demand structure is then in turn read by the **tractor-server**, scaled, and commands issued to the HC12 stack.

Although the AT's on-board computer has the facility to connect a monitor and keyboard, under normal operation this is not available. Data can be imported and exported from the AT computer to a remote computer through the use of the Remote Tool Control facility, **rtc-export**. This facility allows interfacing with remote computers through Tcl/Tk scripts, enabling monitoring of variables and adjustment of parameters within processes running on the AT's PC104 stack. An example of such a Graphical User Interface, running on a remote machine is shown in Figure 3.19.



(a) General structure of the software.



(b) An example of a set of processes and how they interact.

Figure 3.18: An overview of the AT's software systems.

A typical set of processes running on the AT computer is shown in Figure 3.18 (b). In this case, sensor data is read and processed by the **vision** and **orientation server** processes, and written to the **store**. The **control** program reads this processed data, making command decisions based on this data, and perhaps the current state of the HC12s which is also read from the **store**. On making the command decisions, the **control** program

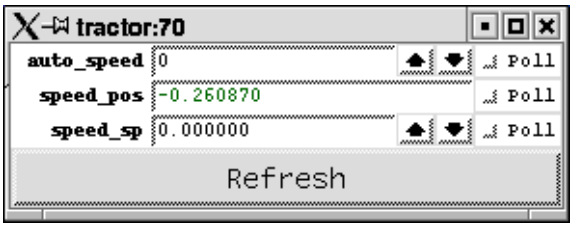


Figure 3.19: An example of a GUI running on a remote machine. Shown is an interface for the speed lever. The `auto_speed` field is used to place the axis of control into **automation** mode, provided the key-switch and pushbutton conditions are met. The `speed_pos` field is the current pedal position and the `speed_sp` is the pedal set-point which can be changed by the user, and also read from the control program.

places the data into the demand structure which is written to the **store**. **tractor-server** reads this demand data, scales it and issues commands to the relevant boards in the HC12 stack. All data written to the **store** can also be logged to disk.

The *tractor-server*

In addition to facilitating communication between the computer and the HC12s, the **tractor-server** also monitors the position of the three-way switch and push-button on the dash of the AT, which determines the state of the system (i.e. **manual**, **remote** or **automatic**). When the state of the system is **automatic**, the control system can accept demands from the computer. Contained in the the demand structure written to the **store** is the software switch for each individual control axis. If this software switch is activated, then **tractor-server** can send the appropriate demand to the HC12 stack (recall Figure 3.5 (b)).

As noted earlier, after data is read from the HC12s, it is scaled before being written to the **store**. For example, the true demand for the steering control is an encoder value in the range 0 to 4096 counts[5]. However, users of the AT do not necessarily need to know the encoder position, rather a steering angle in radians is more useful. Likewise when demands are read from the **store**, they need to be scaled to the original HC12 form before they are written to the HC12s. In effect, the **tractor-server** takes care of the details

[5]Because the steering angle range is limited to $\pm 30°$, the range of encoder values is in fact limited to 0 to 342 counts, with some offset to place the range away from the end of the encoder count, preventing problems with wrapping.

of communicating with the HC12s allowing the user to demand and read the following vehicle states:

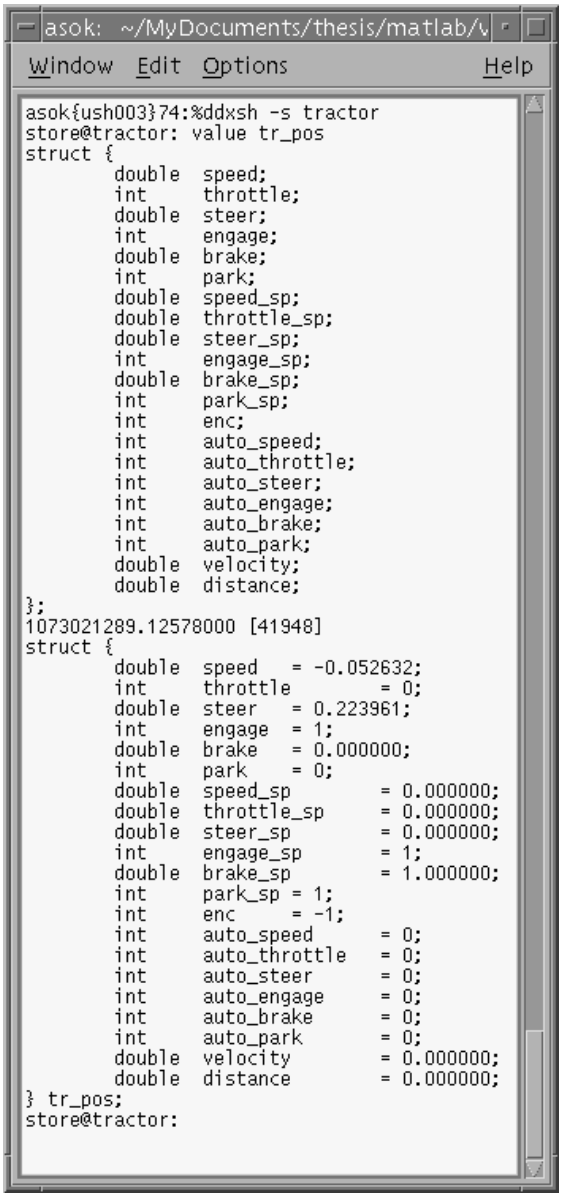
- steering angle: proportional in the range $[-0.52, 0.52\,\mathrm{rad}]$
- steering engage: ON or OFF
- brake: ON or OFF
- park brake: ON or OFF (also applies the brake if brake is OFF)
- throttle: CHOKE, HIGH or LOW
- speed pedal, either
  - vehicle speed: proportional in the range $[-1, 2\,\mathrm{ms}^{-1}]$ or
  - speed pedal position: proportional in the range $[-1, 1]$

Figure 3.20 (a) gives an example of the structure containing data read from the HC12s, scaled and written to the **store**. Figure 3.20 (b) shows the demand structure which is read from the **store** by **tractor-server**, and subsequently scaled back to the form required by the HC12s.

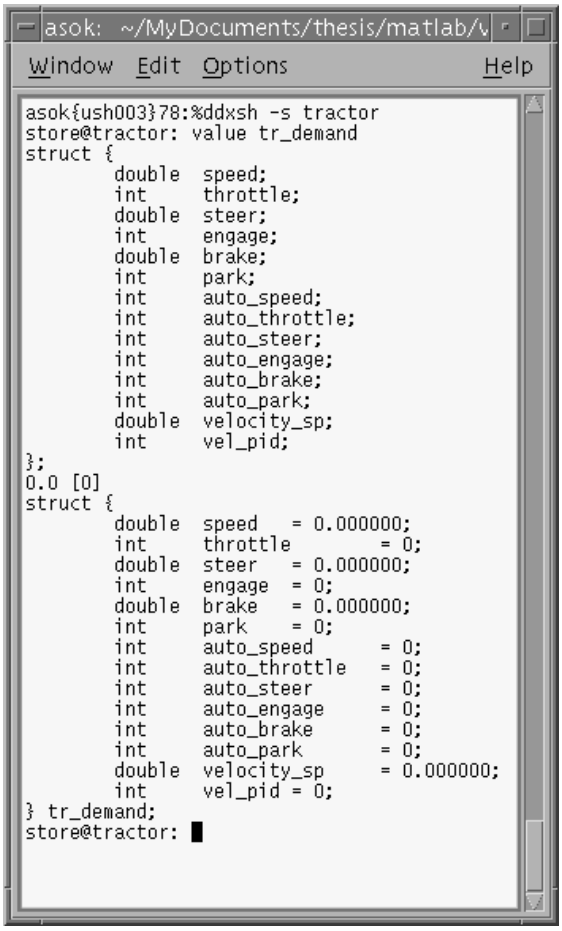
The **tractor-server** module also calculates the vehicle's speed, based on the odometry readings. In the case of the speed pedal, it is important to note that **tractor-server** sets the position of this lever, which is then maintained by a PID loop on the HC12, rather than specifically setting a vehicle velocity. A program switch allows the **tractor-server** process to switch between controlling the vehicle's velocity or speed pedal position directly (in the demand structure this software switch is the variable `vel_pid`). The velocity loop is implemented with a PID loop running at the **tractor-server** level. The output of this PID loop is the speed pedal position set-point, which is written to the HC12.

3.3.3.3 Speed control

Speed control loops for other mechanically similar vehicles have relied on fuzzy logic approaches [Kodagoda *et al.*, 2002; Trebi-Ollennu *et al.*, 1999] and PD control with a



(a) The structure containing the scaled data read from the HC12s and written to the **store**.



(b) An example of the demand structure written to the **store** by a client process. This structure is then read by **tractor-server**, scaled, and written to the HC12s.

Figure 3.20: Examples of the **store** variables used for reading from and writing to the HC12s.

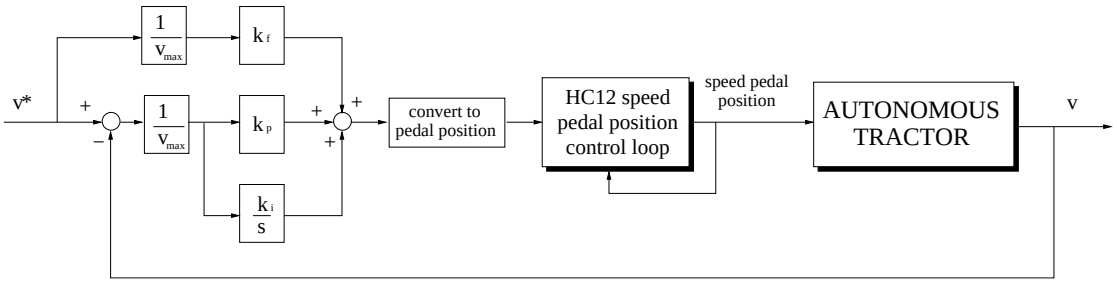


Figure 3.21: The speed control loop.

Speed (ms^{-1})	k_f	k_p	k_i
$\lvert v\rvert > 0.13$	0.0	0.25	0.45
$0 < v \leq 0.13$	0.05	0.25	0.85
$-0.13 < v \leq 0$	0.1	0.25	0.85

Table 3.1: Manually tuned gains for the speed control loop.

discontinuous switching ON-OFF term [Debenest *et al.*, 2002]. Our approach is similar to that used by Liubakka *et al.* [1993], who use PI control with gains based upon the vehicle speed (amongst other parameters). However, for the control of the AT, an additional feed-forward term is used which helps with stiction of the speed pedal, particularly at low speed ($|v| < 0.1\,\mathrm{ms}^{-1}$). A block diagram of the velocity control loop is illustrated in Figure 3.21. Although not shown in the diagram, the gain values are adapted to the demand. Control gains were tuned experimentally and are shown in Table 3.1.

Responses to various step inputs for low and high speed demands are shown in Figure 3.22. Tracking is good for the demand of $\pm 1\,\mathrm{ms}^{-1}$ with little overshoot observed. For the demand of $\pm 0.1\,\mathrm{ms}^{-1}$, tracking is oscillatory. This is due to quantization of the speed sensing, as discussed in Section 3.3.2. Sampling at $1\,\mathrm{Hz}$ gives a speed resolution of $0.0069\,\mathrm{ms}^{-1}$. However, this sampling rate is too low to give adequate control and has been increased to $2\,\mathrm{Hz}$, for which the resolution on speed is $0.0137\,\mathrm{ms}^{-1}$ — hence the oscillatory behaviour around the speed set-point. In addition, both responses exhibit a large latency, of the order of $1\,\mathrm{seconds}$ for the first demand, and $3\,\mathrm{seconds}$ for the second. This latency can be attributed to the relatively low sampling rate, the poor sensing resolution

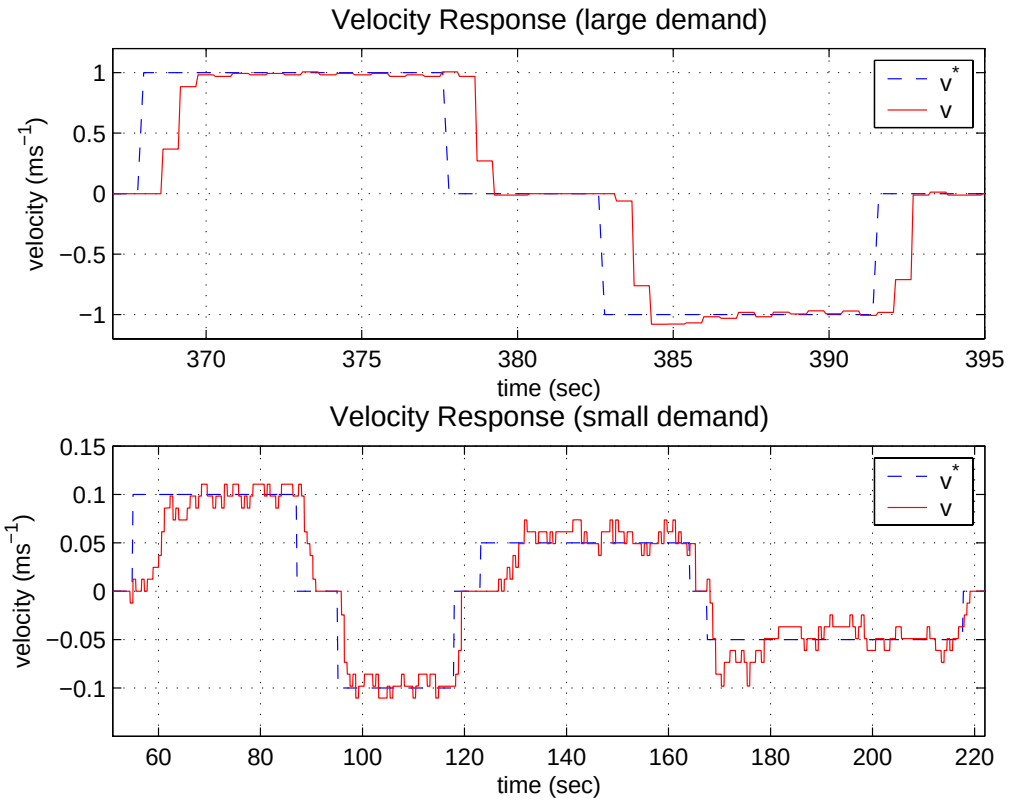


Figure 3.22: Results from the speed control loop, demand is v^* and response is v. Note that the oscillations are largely attributable to quantisation of the speed measurement.

and the communication delays between issuing a demand on the computer and reading back from the HC12s. Also contributing is stiction effects on the speed pedal — this is particularly important for low speed demands and is the reason a feed-forward term is required.

Despite the limitations, this approach is sufficient for the purposes of the work in this thesis. However, a higher resolution encoder on the speed sensing would certainly improve the ease of control of the vehicle.

3.4 Modelling the vehicle

To facilitate the development of various control and navigation algorithms, mathematical models of the vehicle's kinematic and dynamic response to demands from the computer

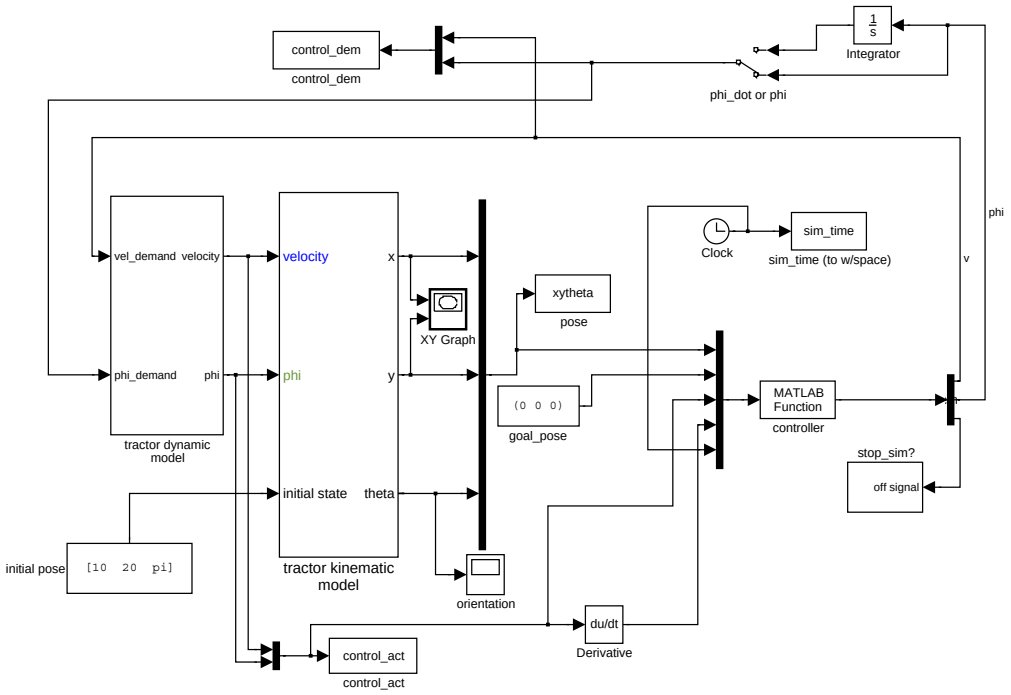


Figure 3.23: The SIMULINK model of the AT system.

system were developed. The standard kinematic model for a rear-wheel driven car-like vehicle was presented in Section 2.6.2. This model can be used for path integration purposes and for simulating the vehicle's motion. Dynamic models of the vehicle are specific to this vehicle given the uniqueness of the automation system and the platform itself — these dynamic models were acquired through experimentation. Both the kinematic and dynamic aspects of the modelling are used in MATLAB and SIMULINK for developing, and analysing, control and navigation algorithms expediting the physical experimentation on the AT.

The overall model of the vehicle is shown in Figure 3.23. It consists of a *dynamic* block which models the response of the steering and velocity loops, and a *kinematic* block which models the geometry and motion of the vehicle. The model allows us to gauge the performance of different controllers in a realistic but convenient environment. In most cases, the controller also feeds back a *STOP_SIMULATION* switch variable which stops the simulation when a particular event has occurred, for example when the vehicle

is within a tolerance distance of the goal pose.

When implementing software for the AT, most of the processes have been written such that they can be run in an *on-line* or *off-line* manner. *On-line* refers to the process using real data in real time whereas *off-line* processing is used for testing of processes with either simulated data or data collected on-line. In particular, in the control code, an equivalent version of the MATLAB / SIMULINK model is integrated in the code for *off-line* testing. This ensures accurate translation of controllers developed in MATLAB to the 'C' code on the vehicle.

The following sections describe the development of the kinematic and dynamic models of the vehicle.

3.4.1 Kinematic model

A geometric model of a car-like vehicle is shown in Figure 3.24. It is usual to assume for such a vehicle that the two wheels on each axle collapse to a single wheel — the *bicycle* model. Referring to Figure 3.24, the kinematic equations for a car-like vehicle with rear tyres aligned with the vehicle and front tyres allowed to rotate about the vehicle's vertical axis are:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{L} \end{bmatrix} v \tag{3.7}$$

where v is the vehicle's forward velocity (measured at the midpoint of the rear axle), L is the axle separation distance, ϕ is the steering angle, and the point (x, y) refers to the coordinates of the midpoint of the rear axle. When commanding the vehicle, the demands v and ϕ are multiplicatively coupled in Equation 3.7 which can present problems in some cases[6] . An alternative representation with four state variables leads, to the following

[6]For example, if the vehicle requires a negative angular velocity, this can be achieved by setting $v > 0$ and $\phi < 0$ or vice versa.

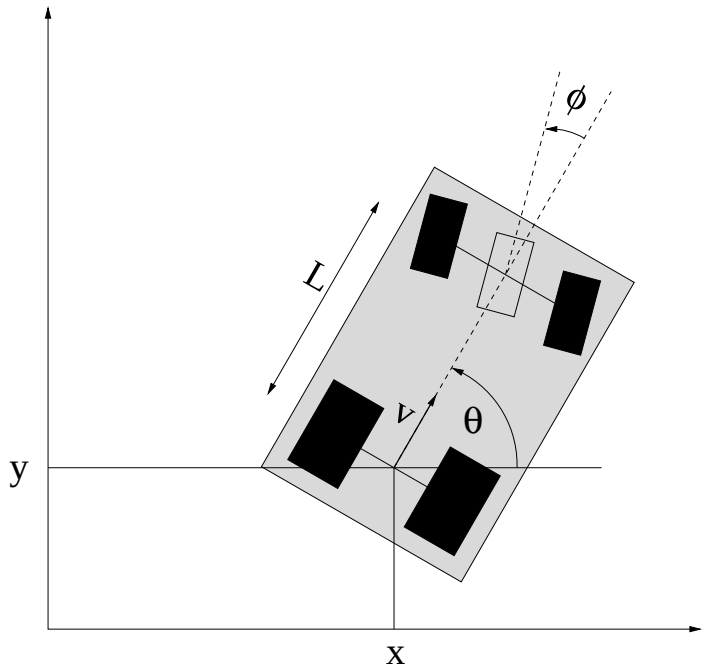


Figure 3.24: The vehicle and the coordinate system used. All angles are counter-clockwise positive.

kinematic representation [De Luca *et al.*, 1997]:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{L} \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \dot{\phi} \tag{3.8}$$

In equation 3.8, the inputs to the system are the translational velocity v and the steering angle rate $\dot{\phi}$.

The different approaches are distinguished by the input to the steering loop. For simulation purposes, a *kinematics* block was created in MATLAB/SIMULINK, as shown in Figure 3.23. The subcomponents of this block are shown in Figures 3.25-3.28.

In Chapter 5, several different controllers are tested, some of which require ϕ as a demand and others which require $\dot{\phi}$. In SIMULINK, it is a simple matter to integrate the input before it enters the kinematic block (this is implemented with a manual switch in the simulation, as shown in the top right of Figure 3.23).

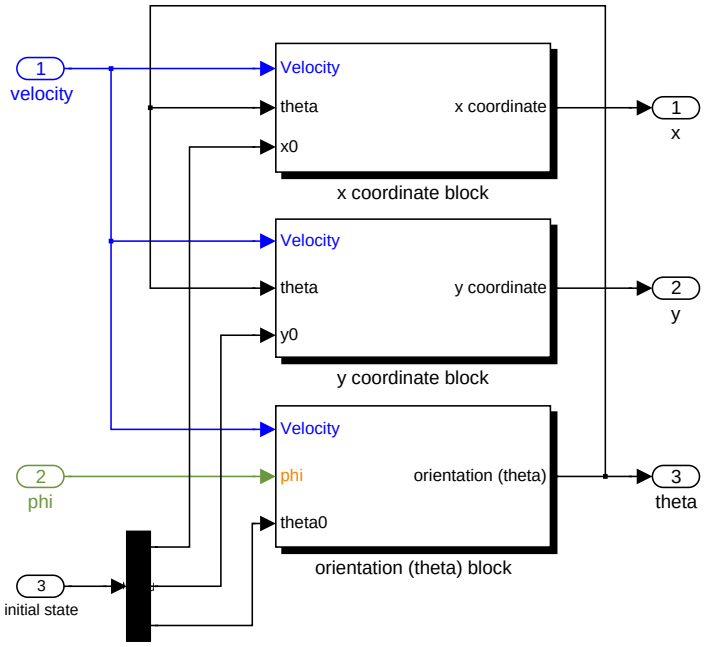


Figure 3.25: SIMULINK model 'AT kinematics'.

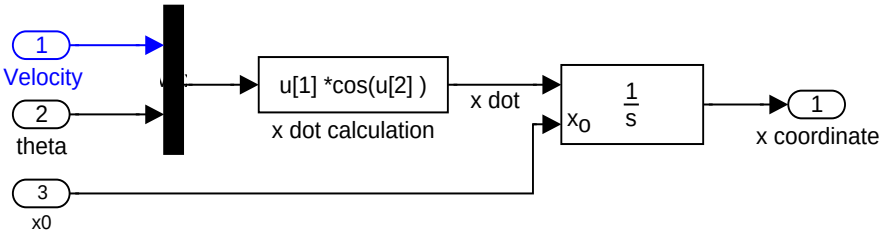


Figure 3.26: SIMULINK model x-coordinate block: calculates x position of the AT using initial x position (x_0), velocity v and vehicle orientation θ.

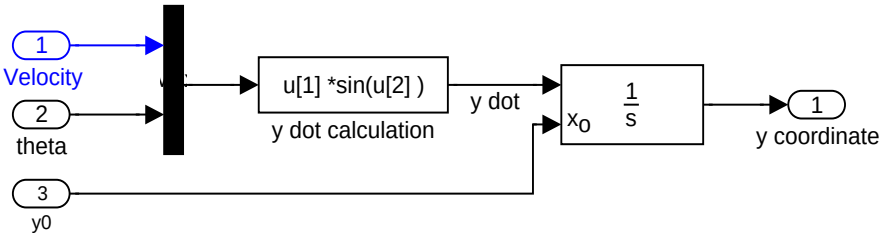
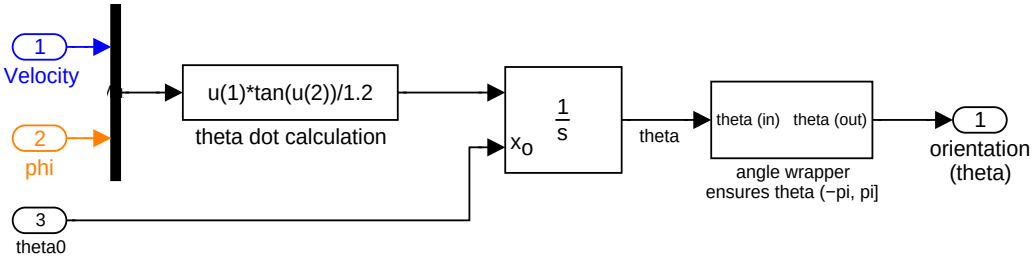
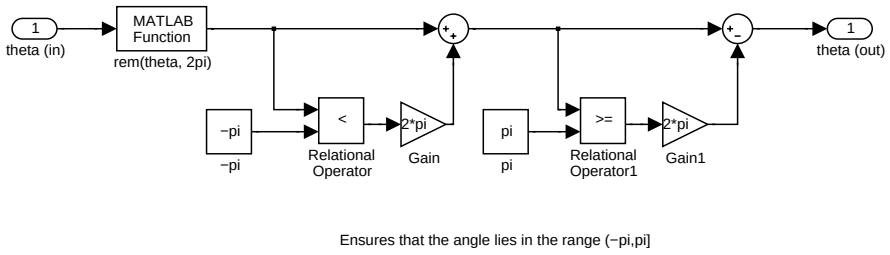


Figure 3.27: SIMULINK model y-coordinate block: calculates y position of the AT using initial y position (y_0), velocity v and vehicle orientation θ.



(a) SIMULINK model theta block



(b) SIMULINK model angle wrapper.

Figure 3.28: SIMULINK model theta block: calculates orientation (θ) of the AT using initial orientation (θ_0), velocity v and steering angle ϕ. The angle wrapper ensures that the vehicle orientation θ remains in the range$(-\pi, \pi]$.

3.4.2 Dynamic modelling

Many control algorithms for car-like vehicles fail to consider the realities of implementation on a vehicle, for example steering angle limitations and the dynamics of the actuators which drive the vehicle. In this work, models of the vehicle's dynamic response to demands are used, including non-linear effects such as input saturation. This allows the realistic testing and development of control algorithms in simulation. Although the presented models are rather simple, they are sufficient to capture the general behaviour of the system. Their use has significantly reduced controller development time.

The demands to the vehicle are, for the purposes of modelling, limited to steering angle and velocity. Of particular importance is to note that on the real vehicle, the velocity is set by a loop around the speed pedal and the odometry system, as described in Section

$v(\,\mathrm{ms}^{-1})$	$\omega_n(\,\mathrm{rads}^{-1})$	ζ
0.1	0.57	0.78
0.3	0.72	0.78
0.5	0.72	0.78
1.0	0.82	0.80
1.5	0.82	0.80

Table 3.2: Variation of steering loop parameters with vehicle velocity.

3.3.3.3.

3.4.2.1 Steering models

An approximate model of the steering dynamics was experimentally identified from the response of the AT's steering loop to unit step changes in desired steering angle (provided from the computer). The response was determined to be approximately second-order of the form (in the Laplace domain):

$$\frac{\phi(s)}{\phi^*(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n + \omega_n^2}$$

The parameters ω_n and ζ where found to vary due to the complexity of the interactions between the terrain and the wheels on different surfaces and also varied with the vehicle's translational speed. Table 3.2 shows the experimentally derived parameter values for different vehicle speeds, where the vehicle was driven on a flat concrete surface. In effect, the response of the steering loop is marginally 'faster' for higher vehicle speeds, with greater control effort required at lower vehicle speeds resulting in slower actuation. As the velocity of the vehicle approaches zero, the steering loop becomes uncontrollable — a large steering demand cannot be effected as the friction in the system is greater than the available control effort. For the purposes of simulation, constant parameter values corresponding to a velocity of $0.5\,\mathrm{ms}^{-1}$ are used.

Figure 3.29 shows a plot of the experimental and model response of the steering loop to a unit step. For the experimental data, the vehicle was travelling over smooth, flat concrete at a speed of $1\,\mathrm{ms}^{-1}$. Although not exact, the match between experimental data and the model is adequate for the purposes of the work in this thesis.

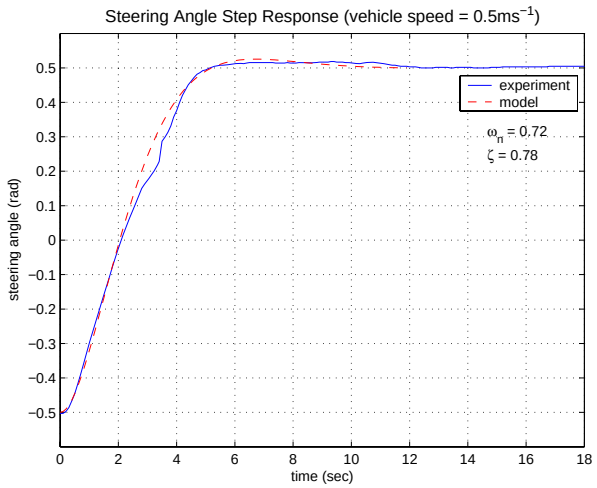


Figure 3.29: Step response of the steering loop. Model data is plotted as a dashed line and experimental data is plotted as a solid line.

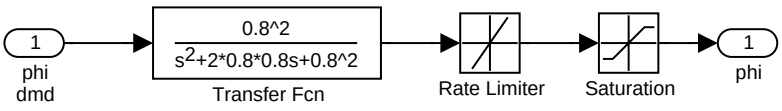


Figure 3.30: SIMULINK model of the steering loop.

The steering angle on the AT is limited to $\pm 0.523\,\mathrm{rad}$ ($\pm 30°$) which is modelled in SIMULINK using a non-linear **saturation** block. Also modelled is the limitation on the rate of change of steering angle, with a rising and falling limit of $\frac{\pi}{6}\,\mathrm{rad\,s^{-1}}$. Rate limiting is simulated using the **rate limit** block in SIMULINK. The full SIMULINK model of the steering loop is shown in Figure 3.30.

3.4.2.2 Velocity models

The velocity loop was empirically determined to have a first-order response which is represented in the Laplace domain as:

$$\frac{v(s)}{v^*(s)} = \frac{K_v}{\tau_v s + 1}$$

Figure 3.31 illustrates the AT's response to a unit step change in velocity while travelling on level ground (concrete). Also shown is the response of the first-order model

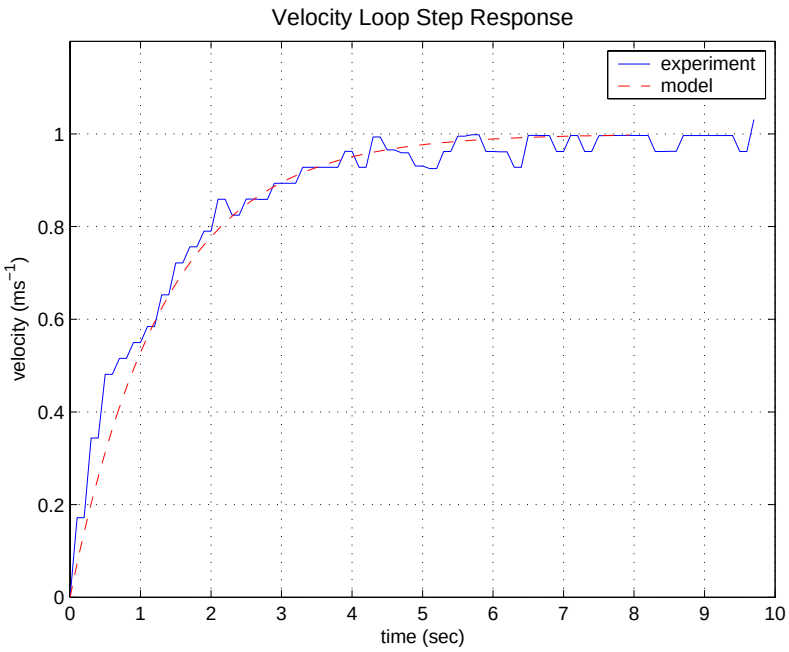


Figure 3.31: Unit step response of the speed loop. Model data is plotted as a dashed line and experimental data is plotted as a solid line.

where $K_v = 1$ and $\tau_v = 1.33$. Note the 'quantisation' of the experimental velocity measurements[7].

The parameters (K_v and τ_v) will again vary slightly due to differences in the velocity response on sloping terrain, different surfaces and under different loading conditions. For simulation purposes, the above parameter values are used.

The forward speed of the AT is limited to approximately $3\,\mathrm{ms}^{-1}$ while the reverse speed is limited to approximately $1.5\,\mathrm{ms}^{-1}$. Velocity saturation is modelled in SIMULINK using a non-linear **saturation** block. Also modelled is the limitation on the rate of change of velocity, which was empirically determined to be $5\,\mathrm{ms}^{-2}$ on acceleration and $2\,\mathrm{ms}^{-2}$ on deceleration. Rate limiting is simulated using the **rate limit** block in SIMULINK. The full SIMULINK model of the velocity loop is shown in Figure 3.32.

[7]For the model identification task, the velocity loop rate was increased to $10\,\mathrm{Hz}$, as opposed to the normally used for control. The normal loop rate of $2\,\mathrm{Hz}$ provides for finer control over velocity at low speeds.

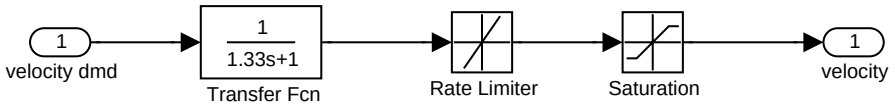


Figure 3.32: SIMULINK model of the velocity loop.

3.4.3 Software modelling

As noted earlier, although the models of the vehicle presented here have been based around MATLAB / SIMULINK blocks, the vehicle kinematics and dynamics have also been modelled within the vehicle control code. This serves two purposes:

1. It enables testing of control code as it is written in the accompanying process.

2. It allows forward modelling / simulation of the vehicle behaviour within the controller itself which can be useful for obstacle avoidance and other control tasks.

In essence, this type of simulation involves two steps; predicting future values of the vehicle velocity and steering angle, and predicting the vehicle pose. The former task involves the solution to the differential forms of the models of the steering and velocity loops, while the later task uses the vehicle kinematics and Runga-Kutta integration.

3.5 Conclusion

This chapter has outlined the design of the Autonomous Tractor which serves as a test-bed for the work presented in this thesis. The vehicle can be driven either manually, by remote control, or it can be controlled by the on-board computer. Unlike many current mobile robots, it is a fully self-contained vehicle which can operate for indefinite periods provided it is regularly refuelled. Also presented in this chapter was a speed controller designed to control the translational speed of a petrol-powered, hydraulically driven vehicle.

Models of the vehicle's kinematics were also presented along with dynamic models of the vehicle's response to inputs. These models were then validated against experimental data. Importantly, the work in this thesis goes beyond much of the current mobile robot

control literature and considers the dynamic aspects of controlling a vehicle's heading and speed. Included in this analysis are limitations on speed, acceleration, steering angle and steering angle rate. The models developed in this chapter are used for simulations which serve to expedite the experimentation on the AT, eliminating unreliable or unworkable solutions before they are tried on the real vehicle. These vehicle models are also used within the control software, providing a means of testing the software off-line and also providing a useful tool for obstacle avoidance and prediction of the AT's future motion based on the current demands.

Chapter 4

Visual Homing and Pose Estimation

Bearing-only visual homing methods have a strong dependence on landmark configuration. By incorporating landmark range, this dependence can be eliminated and a full vehicle pose estimate with respect to some target pose obtained. A reference direction, required for techniques which use anonymous landmarks, is provided by a magnetic compass. Vast improvements can be made to the orientation measurement by combining it with an estimate of the vehicle's angular rate, provided by odometry and a vehicle model. Landmarks are found using an omnidirectional camera and extracted based upon a prelearnt look-up table of the landmark colour profile. Landmark range can be estimated using the ground-plane constraint and a geometric model of the omnidirectional camera's optics. Tracking over time is achieved by using vehicle-object motion models which rely on vehicle odometry. These same models are then used in complementary filters to improve the estimates of landmark range and bearing.

4.1 Introduction

In this chapter, a comparison of several landmark-based visual homing strategies is presented, showing that when using bearing only strategies, it is difficult to obtain consistent performance for varying landmark configurations. For this reason, these strategies are not suitable for the control of a car-like vehicle. A technique which incorporates landmark range, eliminating the landmark configuration dependence, is then developed.

A description of the sensing system used to implement the homing strategy is then provided. The Autonomous Tractor is fitted with a magnetometer which is used to provide the reference direction required for homing. The vehicle is also fitted with a colour omnidirectional camera. This is used to track a set of coloured objects in the environment which serve as landmarks in this work. Object range is estimated using a model of the omnidirectional camera optics and the ground-plane constraint. The chapter concludes with an analysis of possible error sources in the range and bearing based homing strategy.

4.2 Visual homing and pose estimation

In this section, a method for estimating a vehicle's pose with respect to a target pose is developed based upon the discrepancies between the current 'view' of the workspace and some pre-stored view of the target pose. First, a comparison of several landmark-based homing techniques is presented, followed by the development of a range and bearing based technique.

All methods considered here assume that landmarks are homogeneous, i.e. landmarks *do not* necessarily need to be uniquely identified. Importantly, no *a priori* knowledge of landmark positions is required — this knowledge is embedded in a snapshot of the landmark configuration at the target location. In addition, these methods require that the landmarks are visible throughout the vehicle's journey. The methods considered here also require an external reference direction which effectively allows the environment to be viewed with a constant orientation.

An alternative to the requirement for a reference direction is to introduce unique landmarks from which vehicle orientation can be determined, provided the requirements of triangulation are satisfied. However, this can introduce further sources of error and adds a higher level of sophistication to the vision system requirements.

Throughout the following discussion, it is assumed that the target location, i.e. the goal, is the origin of the local coordinate system, and the coordinate system is defined as a right-handed trihedral frame with the the x and y axes defining the ground-plane and

the z axis pointing upwards.

4.2.1 Homing strategies

Here, a comparison of several homing techniques is presented. These are the Weighted Vector (WV) method, a planar Image-Based Visual Servoing (IBVS) technique, and the Average Landmark Vector (ALV) method. Based on this analysis, a new homing method is developed which overcomes some of the shortcomings of these and other similar methods by incorporating landmark range.

4.2.1.1 WV method

The Weighted Vector method[1] was initially presented by Weber *et al.* [1999] and is an extension of the earlier work of Hong *et al.* [1991]. Hong *et al.* [1991] relied on the summation of unit vectors, based upon landmark bearings, to derive the homing vector. Rather than unit vectors, Weber *et al.* [1999] use the difference in landmark bearings to weight each contributing vector. A similar weighting method was independently presented by Lambrinos *et al.* [2000]. Both methods bear similarity to the hypotheses on *bee* navigation presented by Cartwright and Collett [1983].

The WV method requires storage of a list of the landmark bearings as seen from the target location. This list is then matched with the currently viewed landmark bearings, from which a homing vector is calculated. Weber *et al.* present several different methods of obtaining correspondence between the current and target lists, ranging from 'optimal' pairing using an exhaustive search, through to voting schemes. The method used here involves least square matching with preservation of landmark order — this method matches the exhaustive search method in terms of homing success rate [Weber *et al.*, 1999].

Current and target landmark bearings are correlated by finding the minimum of the following matching function (preserving adjacency in the target and current bearings lists):

$$\sum_{i=1}^{n} |\alpha_i^* \ominus \alpha_i|^2 \tag{4.1}$$

[1] A name applied here, the original authors referred to the technique simply as 'homing'.

where n is the number of landmarks, $\alpha_i^* \in (-\pi, \pi]$ is the landmark bearing viewed from the target and $\alpha_i \in (-\pi, \pi]$ is the currently viewed landmark bearing. The $\ominus$ operator refers to the shortest angular distance around a circle between the two angles, referenced to the first angle eliminating problems caused by discontinuities in angle representations. The $\oplus$ operator represents the corresponding angular addition. Note, throughout this work, unless otherwise specified, all angles are defined in the range $(-\pi, \pi]$.

After matching, or *landmark correspondence*, is performed, the homing vector is calculated according to the following *vector* additions:

$$\underline{H_{WV}} = \sum_{i=1}^{n} \underline{V_i} \tag{4.2}$$

where a vector is expressed as a magnitude and angle pair ($M\angle\theta$) and the component vectors, $\underline{V_i}$, are given by:

$$\underline{V_i} \;=\; |\alpha_i^* \ominus \alpha_i| \;\angle\eta_i \tag{4.3}$$

and

$$\eta_i \;=\; \begin{cases} \alpha_i^* \oplus \frac{\pi}{2} & \text{if } \alpha_i < \alpha_i^* \\ \alpha_i^* \ominus \frac{\pi}{2} & \text{otherwise} \end{cases} \tag{4.4}$$

where $\eta_i \in (-\pi, \pi]$. An illustrative example of WV homing vector calculation is shown in Figure 4.1 Note in the example that the homing vector, $\underline{H_{WV}}$, is not directed exactly towards home. As the current position approaches the target, this discrepancy is reduced. Figure 4.2 illustrates the results of applying the WV technique at discrete points on a grid covering a workspace (in this and all cases cited in this section, the workspace is defined as $x, y \in [-20, 20]$ and the grid spacing is 1). Figure 4.2 (a) contains landmarks which are well distributed around the the target position — a *distributed* landmark configuration. In this case, the home vectors provide a close approximation to the true target direction and a scaled version of the distance to the target location. In addition, the WV method appears to have an inherent obstacle avoidance type characteristic, as can be seen from the vectors at positions which are 'behind' landmarks with respect to the target. However, the situation

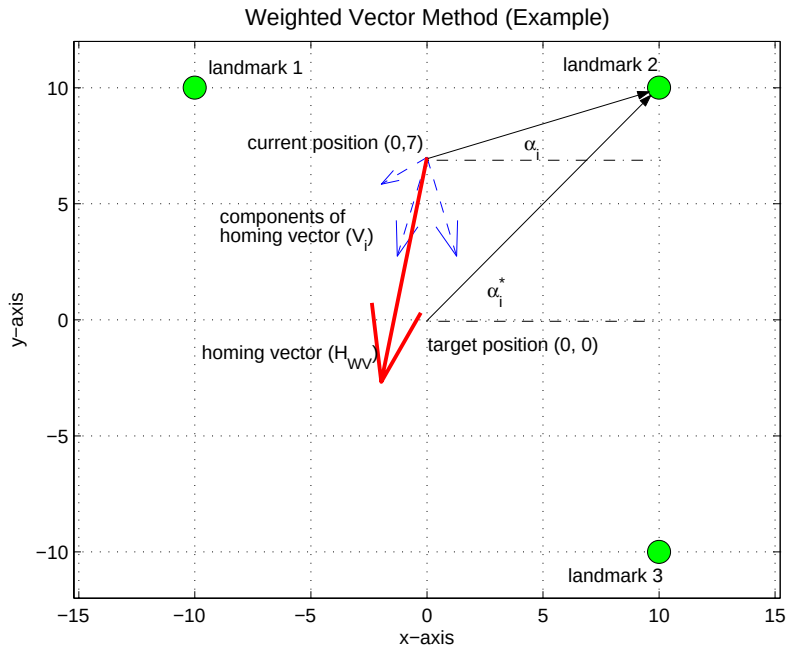
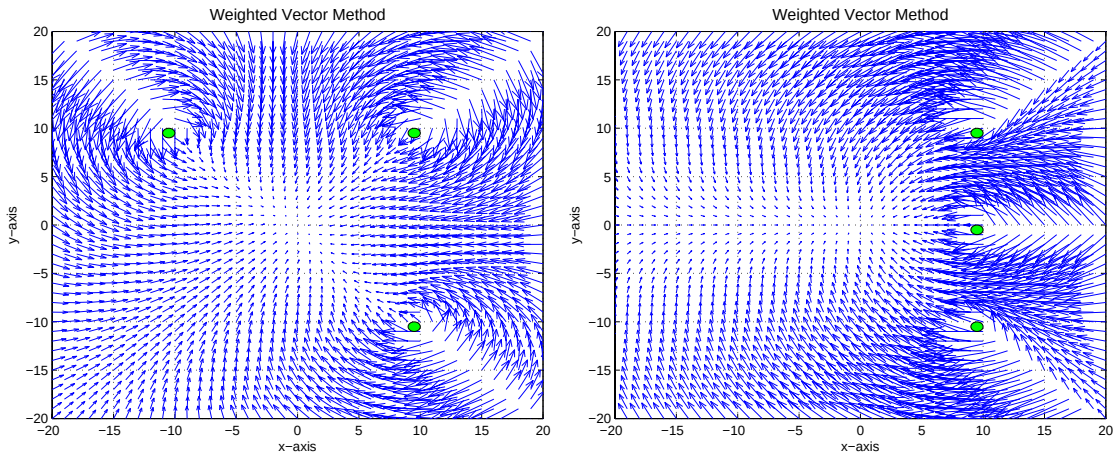


Figure 4.1: Example of the WV method. The vectors have been scaled by a factor of 10 for visibility.

is different for the case where landmarks are less radially distributed with respect to the target position — here called a *clustered* landmark configuration, as in Figure 4.2 (b). For this case, the quality of the homing vector over the workspace deteriorates, even without considering the sensing process itself (e.g. landmarks leaving the *perception horizon* and inaccuracies in determining landmark bearing). The obstacle avoidance characteristic is preserved but vectors near obstacles have a higher magnitude which is counter-intuitive. In addition, far away from the landmarks, the homing vector magnitude diminishes.

4.2.1.2 Planar IBVS method

An interesting approach to homing and *local* navigation is to treat the problem as a planar version of the well-known technique of *visual servoing* [Corke, 2001]. Image-based visual servoing is an approach to controlling robots using visually defined tasks rather than tasks defined in Cartesian coordinates (see, for example, [Hutchinson *et al.*, 1996]) — it has traditionally been applied to manipulator type robots. The parallels to mobile robotic



(a) WV method with distributed landmark layout.

(b) WV method with clustered landmark layout.

Figure 4.2: WV method applied at discrete points covering a workspace, illustrating the effect of landmark configuration. No scaling has been applied to the homing vectors.

navigation were explored by Corke [2001] where IBVS techniques were used to derive control laws to stabilize a (simulated) omnidirectional mobile robot to a pose. Here the development of these control laws, in a homing context, is presented.

This particular application of visual servoing relates the bearing angle changes of selected landmarks to desired motion through a simple 'Image Jacobian' [Corke, 2001]:

$$\dot{\beta}_i = \begin{bmatrix} \frac{\sin(\beta_i+\theta)}{R_i} & -\frac{\cos(\beta_i+\theta)}{R_i} & -1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{4.5}$$

where x and y are the current position, θ is the current orientation with respect to the x-axis, β_i is the robot relative landmark orientation, and R_i is its range; see Figure 4.3 for a description of the coordinate system. Note that in this work, the range to each landmark is assumed to be a constant as in Corke [2001] — this is quite similar to the bearing-only homing strategies (like the WV method) which also by default presume a constant landmark range. A difference here from the work of Corke [2001] is that each landmark range is set equal to the landmark range as observed from the target location rather than

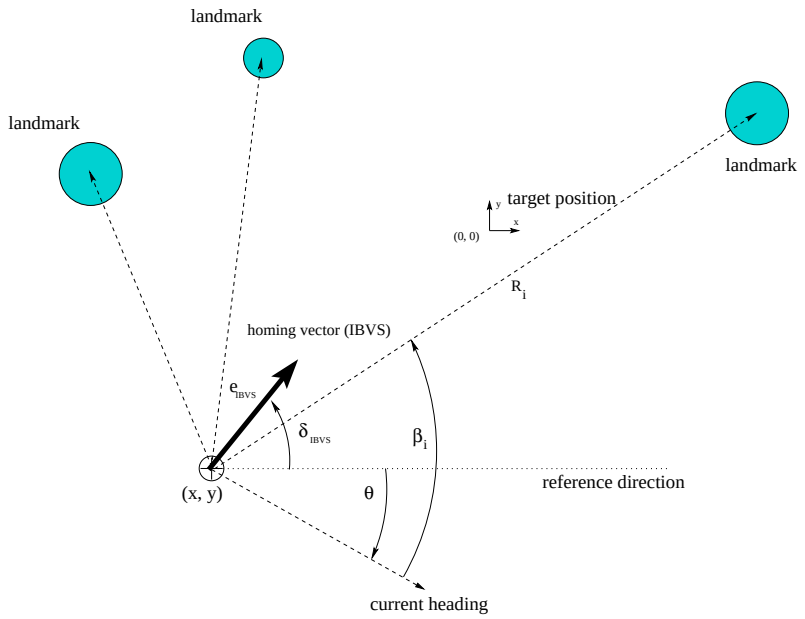


Figure 4.3: The coordinate system and conventions used for planar IBVS.

set to some arbitrary positive constant, i.e.:

$$R_i = R_i^* \tag{4.6}$$

where R_i is the current landmark range and R_i^* is the landmark range observed at the target location. This strategy produces more accurate homing vector directions over a wider array of initial positions. Incidently, using the correct current landmark range in the calculation produced vectors with larger direction errors throughout the workspace.

For three or more features, the equations can be stacked:

$$\dot{\underline{\beta}} = \mathbf{J}\dot{\underline{x}} \tag{4.7}$$

where $\underline{x} = \begin{bmatrix} x & y & \theta \end{bmatrix}^T$. By finding the inverse (three landmarks) or the Moore-Penrose pseudo-inverse (more than three landmarks) an explicit relationship for desired velocity based on the feature velocity can be obtained. The feature velocity is chosen to be proportional to the feature error, which is given by the difference between the desired feature bearings and the current feature bearings:

$$\dot{\underline{\beta}} = \Gamma(\underline{\beta}^* \ominus \underline{\beta}) \tag{4.8}$$

where Γ is a positive definite gain matrix (here, $\Gamma = [0.1, 0.1, 0.1]^T$), $\underline{\beta}^*$ is the desired feature bearing vector found from a snapshot at the target location, and $\underline{\beta}$ is the current feature bearing vector.

Intuitively, Equation (4.5) is resolving two errors, a translational error (given by $\dot{x}$ and $\dot{y}$) and a rotational error (given by $\dot{\theta}$). For homing, only the translational error terms are required. By using the solutions for $\dot{x}$ and $\dot{y}$ found from Equation (4.5) a magnitude, corresponding to the desired velocity:

$$e_{IBVS} = \sqrt{\dot{x}^2 + \dot{y}^2} \tag{4.9}$$

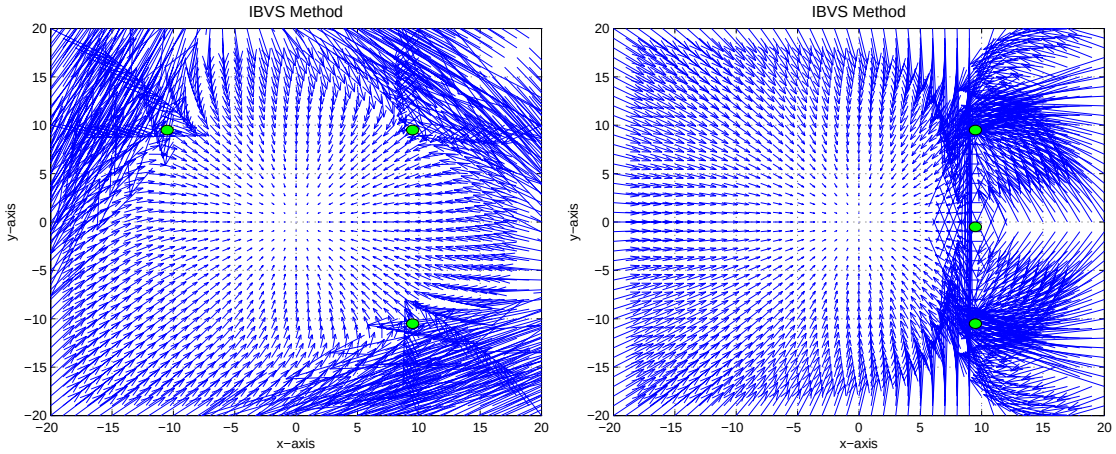
and a direction, corresponding to the desired heading angle:

$$\delta_{IBVS} = \text{arctan2}(\dot{y}, \dot{x}) \tag{4.10}$$

can be found. Here, $\text{arctan2}(y, x) \in (-\pi, \pi]$ is the four-quadrant inverse tangent function. This resolves the translational error, taking the vehicle towards the desired location but like most homing strategies *it does not resolve orientation error on reaching the target*.

Figure 4.4 shows the application of this strategy at discrete points on a grid covering a workspace as defined in the previous section. For the case of Figure 4.4 (a), homing is successful from within an area defined by the polygon made by the landmarks. In fact, the error between the true target direction and that given by the IBVS method, when within the polygon defined by the landmarks, is far superior to the WV method. Outside this polygon, the vectors are not as 'well behaved'. Unlike the WV method, this method produces homing divergence at positions near and around landmarks, rather than providing obstacle avoidance. Figure 4.4 (b) indicates that this method breaks down dramatically at positions near and around landmarks, with divergent behaviour expected from starting positions behind the landmarks with respect to the target position. However, for other starting positions, the IBVS method generates excellent homing vectors.

Corke [2001] observed that the IBVS method displayed divergent behaviour for starting positions outside the polygon formed by the landmarks in the workspace. The performance of the IBVS method should deteriorate for the landmark configuration of Figure



(a) Planar IBVS method with distributed landmark layout.

(b) Planar IBVS method with clustered landmark layout.

Figure 4.4: Planar IBVS method applied at discrete points covering a workspace, illustrating the effect of landmark configuration. No scaling has been applied to the homing vectors.

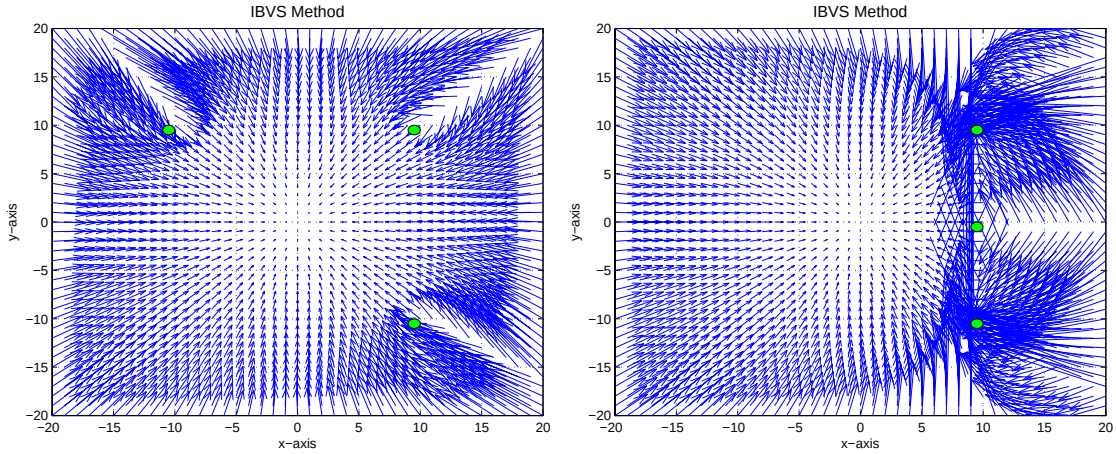
4.4 (b) whereas in fact the behaviour is better than for the distributed landmark configuration. Perhaps the reason for this previously observed divergent behaviour is that the method is driving the bearing angle error of, at least, three landmarks towards zero; for some cases, bearing error must actually increase before it can be driven towards zero and the method is ill-posed for these cases[2]. When the landmarks are aligned, as in Figure 4.4 (b), the landmark bearing errors can be driven to zero immediately (i.e. errors do not need to increase before they can decrease) over a much wider region of the workspace. For some initial positions on the right-hand side of the plot, the bearing errors must increase before they can decrease, and the homing vector behaviour for these starting positions is divergent.

Interestingly, by removing orientation considerations, Equation 4.5 becomes:

$$\dot{\alpha}_i = \begin{bmatrix} \frac{\sin(\alpha_i)}{R_i} & -\frac{\cos(\alpha_i)}{R_i} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \tag{4.11}$$

where $\alpha_i = \beta_i + \theta$. In this case, the minimum number of landmarks is now two (as there

[2]Peter Corke, personal communication, 22 December 2003.



(a) Planar IBVS method with distributed landmark layout.

(b) Planar IBVS method with clustered landmark layout.

Figure 4.5: Planar IBVS method with the removal of orientation from the calculation, applied at discrete points covering a workspace, illustrating the effect of landmark configuration. No scaling has been applied to the homing vectors.

is now only two unknowns $\dot{x}$ and $\dot{y}$). Using similar techniques to the case for $\underline{\dot{\beta}}$, $\dot{x}$ and $\dot{y}$ can be found. For this case, the bearing error is given by

$$\underline{\dot{\alpha}} = \Gamma(\underline{\alpha}^* \ominus \underline{\alpha}) \tag{4.12}$$

where $\Gamma = [0.1, 0.1]$, and $\underline{\alpha}^*$ and $\underline{\alpha}$ are the vectors of target and current absolute bearing angles respectively. Figure 4.5 shows results from the distributed and clustered landmark configurations. From the plots, it can be seen that the homing vectors near and around landmarks appear to flow around the landmarks, as for the WV method, and that divergence no longer occurs. Note that the results of Figure 4.4 and 4.5 presume that the landmark correspondence problem has been solved which in reality is a reasonably difficult problem.

4.2.1.3 ALV method

The Average Landmark Vector method was initially presented by Lambrinos *et al.* [2000] and is an extension of a hypothesis on the navigation behaviour of the desert ant, *cataglyphis*

bicolor. Like the WV method, it is a bearing-only strategy. However, the only storage required for this method is a single vector which summarises the view at the target location — landmark correspondence is not required. Correspondence is a notoriously difficult problem and the simplicity of the ALV approach can result in a very robust system provided stable landmarks are available.

An ALV for a particular location is found by vectorially summing the unit vectors towards each visible landmark:

$$\underline{ALV} = \frac{1}{n}\sum_{i=1}^{n}\underline{U_i} \tag{4.13}$$

where $\underline{U_i}$ is the unit vector towards landmark i, and n is the number of landmarks currently seen. The homing vector is formed by subtracting the (stored) target ALV, $\underline{ALV}^*$, from the ALV at the current location, $\underline{ALV}$:

$$\underline{H}_{ALV} = \underline{ALV} - \underline{ALV}^* \tag{4.14}$$

An illustrative example of homing vector calculation is shown in Figure 4.6, while application of the ALV method at discrete points covering a workspace is shown in Figure 4.7. Again, the effect of landmark distribution with respect to the target position is quite marked. As with the previous two methods, from the same position, the magnitude and direction of the homing vector is dependent on the landmark configuration — this can make it difficult to control vehicles with significant motion constraints, as for a car-like vehicle.

4.2.1.4 Improved Average Landmark Vector method

All methods considered so far have a considerable dependence on the landmark configuration. This work has found that by incorporating landmark range, this dependence can be eliminated. The ALV strategy relies on the summation of the unit vectors towards each landmark. If instead the *range* vectors are summed, a measurement of position with respect to the centroid of the landmarks in the workspace is obtained. This method, developed from the work in this thesis, is termed the Improved Average Landmark Vector method.

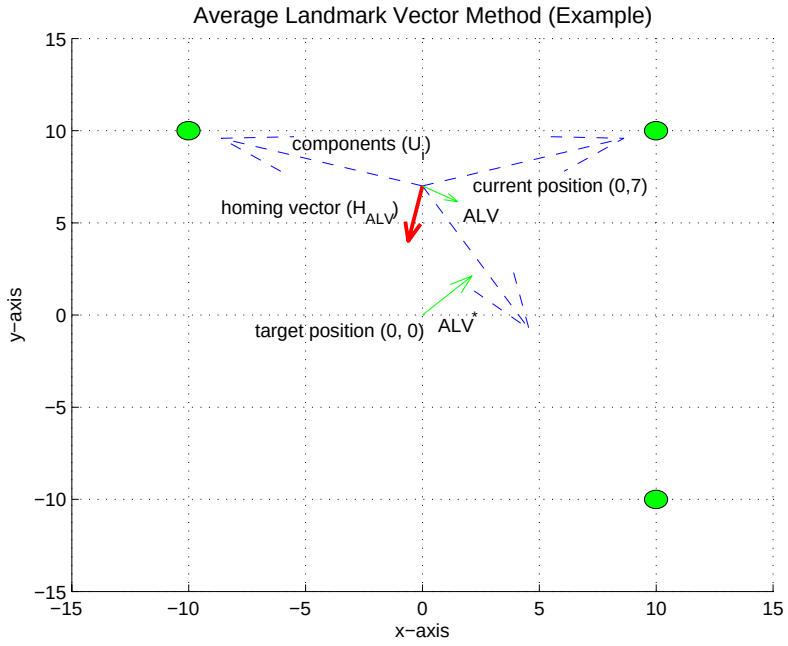
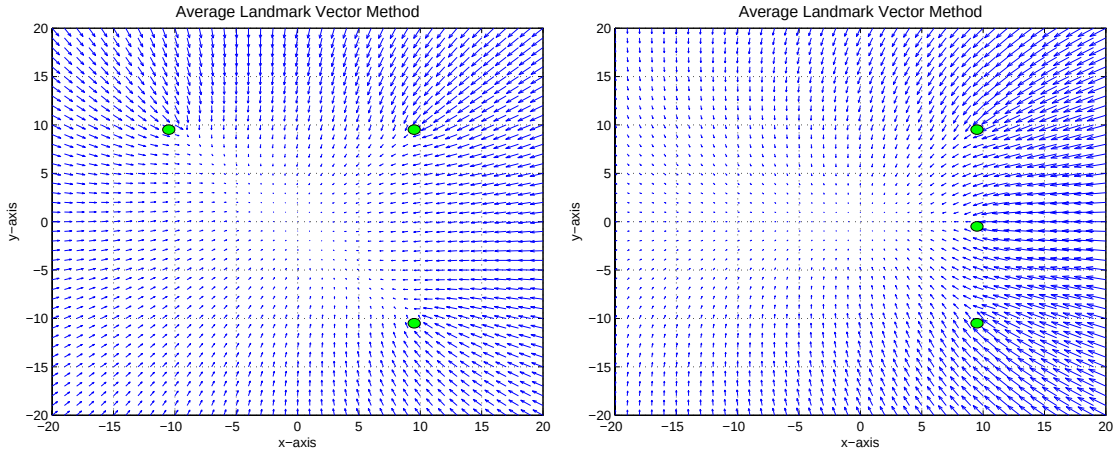


Figure 4.6: Example of the ALV method. The vectors have been scaled by a factor of 10 for visibility.



(a) ALV method with distributed landmark layout.

(b) ALV method with clustered landmark layout.

Figure 4.7: ALV method applied at discrete points covering a workspace, illustrating the effect of landmark configuration. No scaling has been applied to the homing vectors.

Many of the bearing-only homing strategies are inspired by observations of navigation behaviours in biology, particularly in insects. It is interesting to note that many insects and animals are thought to estimate range through a variety of cues, including 'height in image' (see for example [Zeil, 1998; Kral, 1999; Sekuler and Blake, 1994]) as is used in this work. Thus the introduction of range information does not necessarily move away from the 'biologically inspired' approaches.

Using the same type of strategy as for the ALV method, an IALV for a particular location is found by vectorially summing the *range* vectors towards each visible landmark:

$$\underline{IALV} = \frac{1}{n}\sum_{i=1}^{n}\underline{L_i} \tag{4.15}$$

where $\underline{L_i}$ is the range vector towards landmark i (consisting of the range R_i and the bearing α_i), and n is the number of landmarks.

When calculating the homing vector, landmark correspondence is again not required and the only storage needed is the IALV at the target location, $\underline{IALV^*}$. As for the ALV method, the homing vector is formed by subtracting the target IALV from the IALV at the current location, $\underline{IALV}$:

$$\underline{H_{IALV}} = \underline{IALV} - \underline{IALV^*} \tag{4.16}$$

An illustrative example of homing vector calculation is shown in Figure 4.8, while Figure 4.9 shows the application of the IALV method at discrete points covering the workspace. Note that in the figure the homing vectors have been scaled for visibility purposes — the magnitude of each vector is in fact the distance to the target location. In contrast to the previous methods, the incorporation of range information has eliminated the effects of landmark configuration, at least for the case considered here where 'perfect' sensing is available.

4.2.2 Comparing the methods

In this section, the *clustered* landmark configuration of Figure 4.9 (b) is used to compare the effectiveness of the homing methods presented in the preceding sections. The homing vector magnitude and the angle errors over the workspace for the IALV method are shown

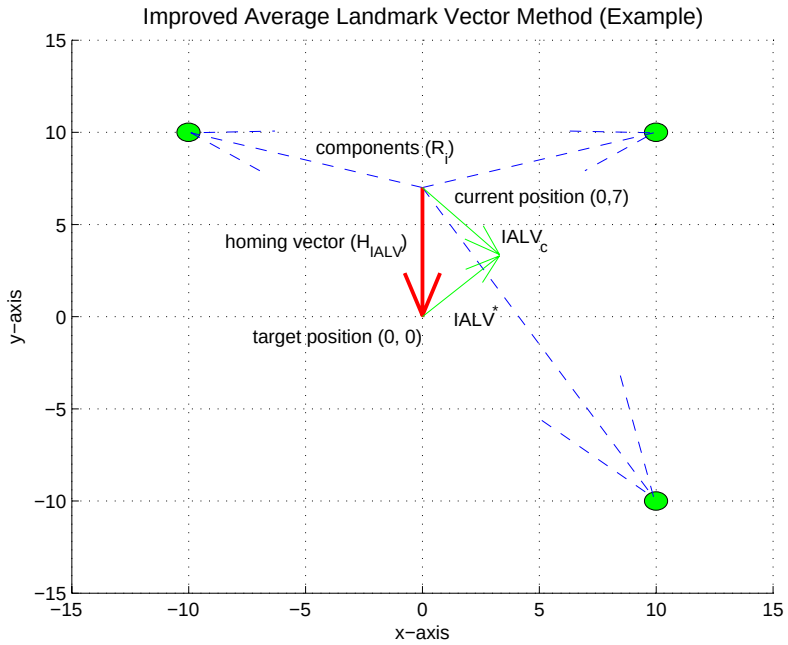
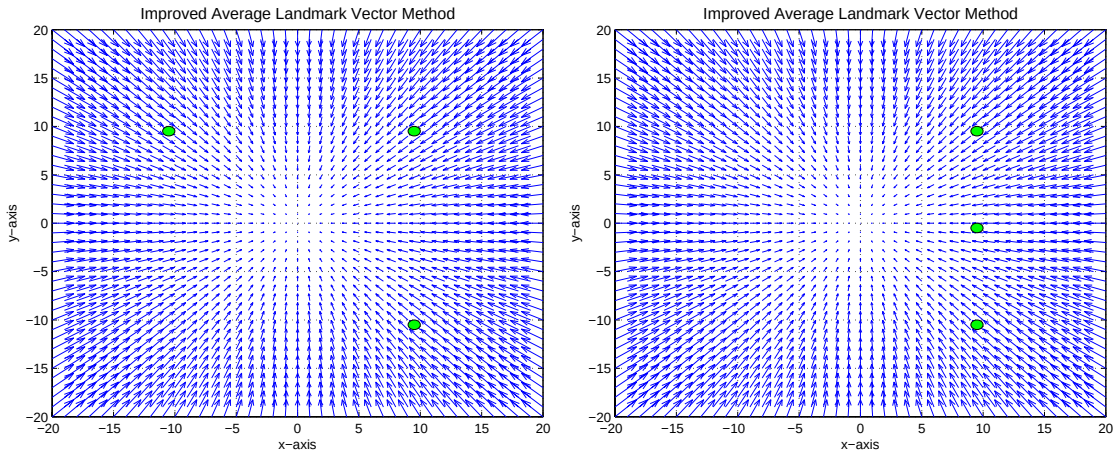


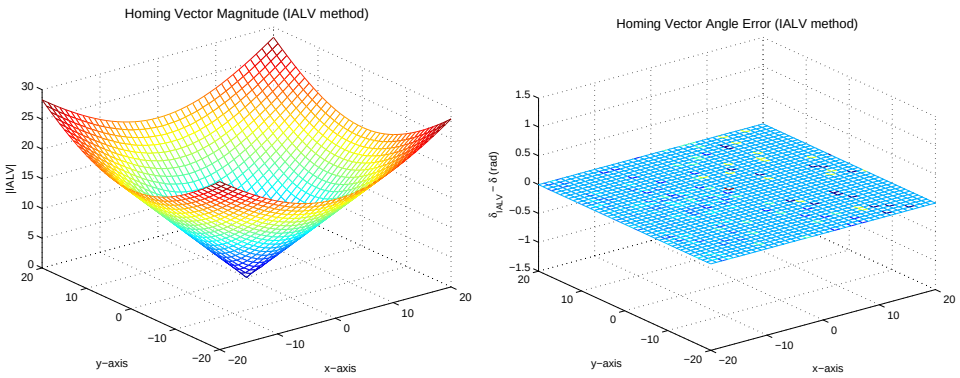
Figure 4.8: Example of the IALV method. All vectors are to scale.



(a) IALV method with symmetrical landmark layout.

(b) IALV method with asymmetrical landmark layout.

Figure 4.9: IALV method applied at discrete points covering a workspace, illustrating the effect of landmark configuration. The IALV is *independent* of landmark configuration. All vectors in the plot have been scaled, the magnitude of each vector is in fact the distance to 'home'.



(a) IALV homing vector magnitude. (b) IALV homing vector angle error.

Figure 4.10: IALV method vector magnitude, and homing vector direction error, for the landmark configuration of Figure 4.9 (b).

in Figure 4.10, and for the WV, IBVS, and ALV methods the same information is shown in Figure 4.11. First, it is noted that the IALV method of Figure 4.10, provides the exact distance and orientation to home. The shape of the 'attractive well' for the magnitude of the IALV homing vector is conical and 'even' throughout the workspace — it does not depend on landmark configuration. A mathematical sketch of this observation follows. Using similar terminology to Möller [2000], if the Cartesian position of landmark i is represented by the vector:

$$\underline{x_i} = (x_i, y_i) \tag{4.17}$$

the target position by the vector:

$$\underline{x}^* = (x^*, y^*) \tag{4.18}$$

and the current position by the vector:

$$\underline{x} = (x, y) \tag{4.19}$$

where x_i and y_i are the Cartesian position of landmark i, x^* and y^* are the Cartesian position of the target location, and x and y are the current Cartesian position, the IALV strategy calculates the homing vector as (effectively):

$$\underline{H_{IALV}} \quad = \quad \underline{IALV} - \underline{IALV_t}$$

$$= \frac{1}{n}\sum_{i=1}^{n}(\underline{x}_i - \underline{x}) - \frac{1}{n}\sum_{i=1}^{n}(\underline{x}_i - \underline{x}^*) \tag{4.20}$$

which reduces to:

$$\underline{H}_{IALV} = \underline{x}^* - \underline{x} \tag{4.21}$$

This is in fact equivalent to the negated gradient of the attractive potential field produced by a parabolic well, commonly found in the Artificial Potential Field literature (see Latombe [1991]):

$$\underline{H}_{IALV} \equiv -\nabla U \tag{4.22}$$

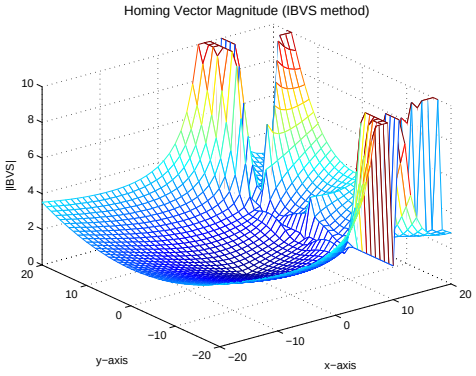
where

$$U = \frac{1}{2}(\underline{x} - \underline{x}^*)^2 \tag{4.23}$$

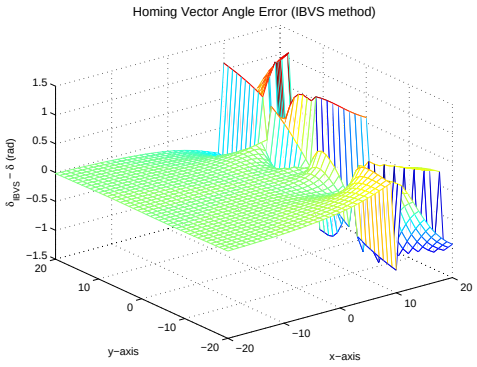
for a parabolic well. For the case of the IALV strategy, there is no mathematical dependence on landmark configuration in the homing vector calculation.

For the IBVS method, the second technique involving the absolute bearing angles $\dot{\alpha}$ is used for comparison purposes. The magnitude of the homing vector varies vastly at positions near the landmarks as does the error in homing vector direction, as shown in Figures 4.11 (a) and (b). However, an attractive well is clearly visible at positions well away from the landmarks and the error in homing direction is well behaved at these starting positions.

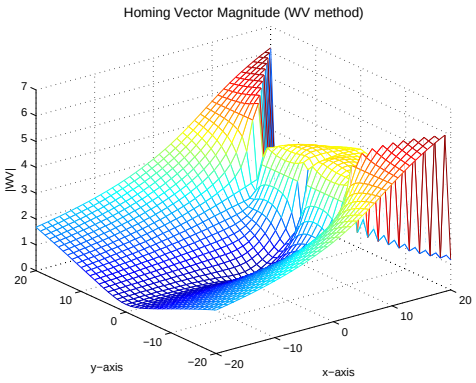
For the bearing-only strategies, the situation is quite different as the shapes of the homing vector magnitude 'attractive wells' are highly distorted for all cases. In the case of the WV method, Figures 4.11 (c) and (d), the distortion in the homing vector magnitude near the landmarks can be explained by the inherent obstacle avoidance characteristic of this method. However, even far from the landmarks, the well for the WV method is not particularly evenly shaped. Also, the difference in true direction to the target location and the direction calculated by the WV method is marked over the entire workspace. For the case of the ALV method, Figures 4.11 (e) and (f), the shape of the attractive well for the magnitude of the homing vector is equally as bad as for the WV method. However, the direction of the ALV homing vector does not appear to deviate quite as much as for the WV method.



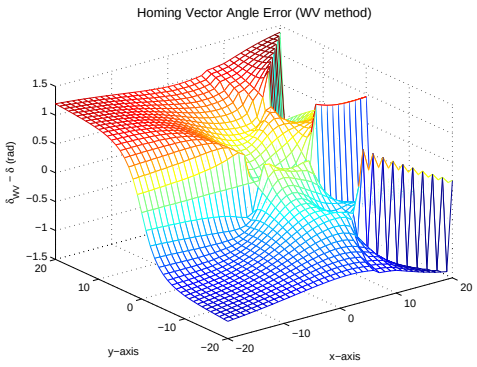
(a) IBVS homing vector magnitude (truncated at a value of 10).



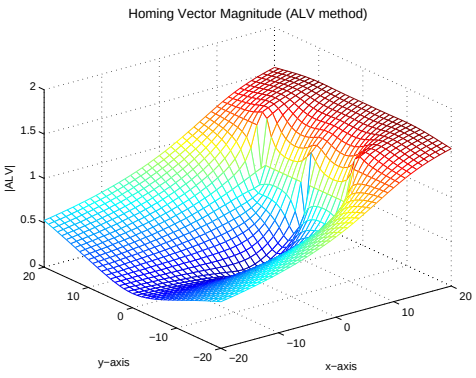
(b) IBVS homing vector angle error.



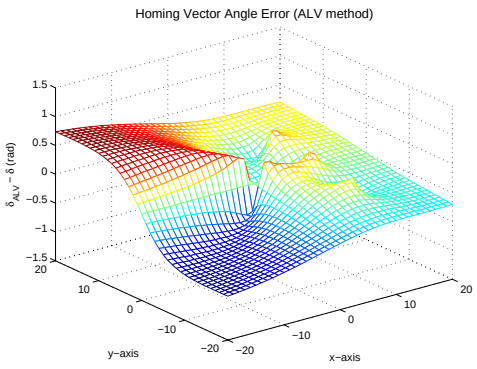
(c) WV homing vector magnitude.



(d) WV homing vector angle error.



(e) ALV homing vector magnitude.



(f) ALV homing vector angle error.

Figure 4.11: Comparison of the methods, for the landmark configuration of Figure 4.9 (b).

A mathematical sketch of the above observations for the ALV strategy follows. Using the same terminology as for the IALV case, the ALV homing vector is, effectively, calculated as:

$$\begin{aligned} \underline{H}_{ALV} &= \underline{ALV} - \underline{ALV}_t \\ &= \frac{1}{n}\sum_{i=1}^{n}\frac{(\underline{x}_i - \underline{x})}{\|\underline{x}_i - \underline{x}\|} - \frac{1}{n}\sum_{i=1}^{n}\frac{(\underline{x}_i - \underline{x}^*)}{\|\underline{x}_i - \underline{x}^*\|} \end{aligned} \tag{4.24}$$

Rearranging terms leads to:

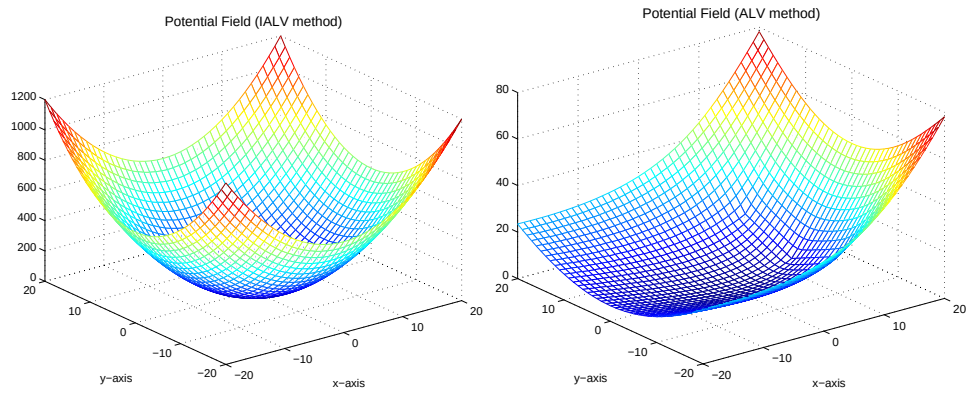
$$\begin{aligned} \underline{H}_{ALV} = \frac{1}{n}\sum_{i=1}^{n}\Bigg[&\left(\frac{\underline{x}_i}{\|\underline{x}_i - \underline{x}\|} - \frac{\underline{x}_i}{\|\underline{x}_i - \underline{x}^*\|}\right) + \\ &\left(\frac{\underline{x}^*}{\|\underline{x}_i - \underline{x}^*\|} - \frac{\underline{x}}{\|\underline{x}_i - \underline{x}\|}\right)\Bigg] \end{aligned} \tag{4.25}$$

In this calculation, the $\underline{x}_i$ terms, which represent the landmark positions, do not cancel as they did in the IALV case. When landmarks are evenly distributed around the workspace, the summation introduces some term cancellation, effectively eliminating the effects of the x_i terms. However, for pathological landmark distributions, this term cancellation due to the summation does not occur, leading to attractive fields with steep gradients at some workspace locations and very shallow gradients in others, as shown in Figure 4.11 (e). However, convergence still occurs and, as the goal is approached, the $\|\underline{x}_i - \underline{x}^*\|$ and $\|\underline{x}_i - \underline{x}\|$ terms (which are in fact the landmark range at the goal, and the current landmark range) approach each other and convergence improves due to the decreased influence of the $\underline{x}_i$ terms — the convergence behaviour approaches a scaled version of the parabolic Artificial Potential Field (Equation 4.23). Incidently, the Artificial Potential Function for the ALV case is given by [Möller, 2000]:

$$U = \sum_{i=1}^{n}\|\underline{x}_i - \underline{x}^*\| - \frac{\underline{x}_i - \underline{x}}{\|\underline{x}_i - \underline{x}^*\|}(\underline{x}_i - \underline{x}) \tag{4.26}$$

The *potential wells* for the IALV and ALV cases, with the clustered landmark configuration, are shown in Figure 4.12, highlighting the difference in convergence behaviour.

The bearing-only methods are, by design, approximate and cheap, only requiring sensing of landmark bearing and a reference direction. Here, the landmark configuration has



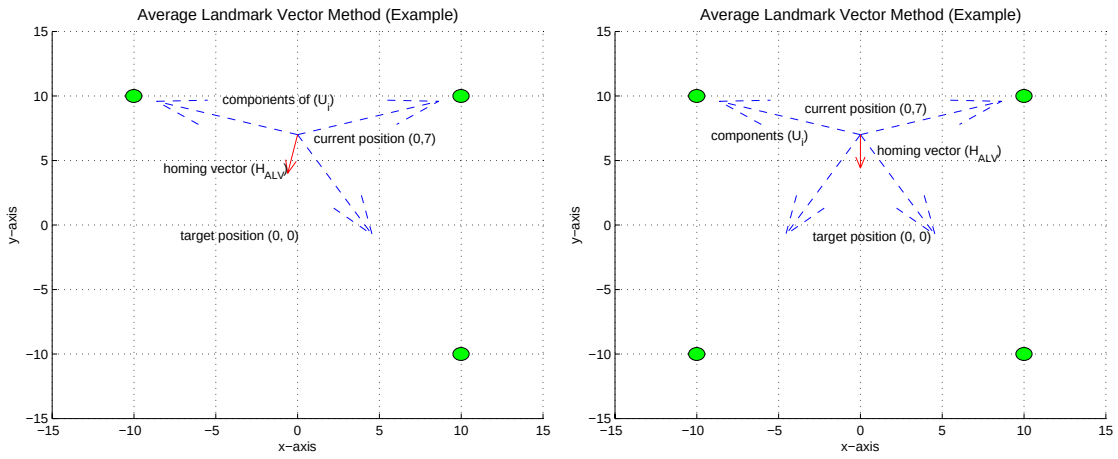
(a) IALV potential function.

(b) ALV potential function.

Figure 4.12: IALV and ALV potential function value over the workspace, for the landmark configuration of Figure 4.9 (b).

been chosen to highlight the shortcomings of these strategies. However, less pathological landmark configurations show similar characteristics, although not quite as markedly. The key point is that bearing-only strategies are highly dependent on landmark configuration. If one is to use a bearing-only strategy, particularly for the 'simple' homing strategies as compared to the more sophisticated SLAM-based approaches (see for example [Shimshoni, 2002; Fitzgibbons and Nebot, 2002; Bailey, 2003; Deans and Hebert, 2000]), careful consideration of the landmark distribution in the workspace is required as the convergence behaviour to the target location may vary depending on the starting location.

Figure 4.13 shows the difference in homing vector, using the same target and current positions, with different landmark configurations. This illustrates the difficulty of applying consistent commands to a vehicle given the same displacement from the target location. The homing vector provides a representation of the *direction* to the target location, and a scaled version of the *distance*. If these quantities vary with landmark configuration, it is difficult to make intelligent decisions on the control of the vehicle. For example, if the vehicle's initial position is $(x, y, \theta) = (0, 2, 0)$, because of the limitation on the steering angle, the vehicle cannot reach the goal using a simple linear controller



(a) ALV homing vector for the case of three landmarks.

(b) ALV homing vector for the case of four landmarks with same target and current positions as (a).

Figure 4.13: Homing vector dependence on landmark configuration.

— a back and forth motion is required. In contrast, if the vehicle is a long way from the goal, it has plenty of time and space to align its longitudinal axis with the homing vector direction and homing to the target location would more than likely be successful. In this latter case, varying the landmark configuration merely changes the path followed by the vehicle.

If the estimates of the distance and heading to the goal vary with landmark configuration how can intelligent decisions as to the type of motion required be made? Of course, if using 'natural' landmarks, control of landmark configuration is not always possible. When controlling a car-like vehicle, which has significant local motion constraints, the use of these strategies is not viable.

The IBVS method performed comparatively well at certain regions in the environment but near and around landmarks, the homing vector direction and magnitude can be unpredictable. Although convergence to the target position still occurs, the requirement for landmark correspondence, and the need for at least two landmarks (three for the case of relative landmark bearings) leaves the IALV strategy as the preferred option. The IALV

strategy *does not* require landmark correspondence and at a minimum requires just one landmark.

In this work, it is assumed that the workspace is relatively flat and that all landmarks used for navigation will lie on the ground-plane. The next section discusses the sensing system used to provide a reference direction and the landmark range and bearings required for the IALV strategy.

4.3 Sensing

The previous section identified the sensing required for the IALV strategy as:

- Measurement of vehicle orientation.
- Measurement of landmark range and bearing.

The AT has facility for carrying a variety of sensors. The primary sensors used for the work in this thesis are a magnetic compass (in the form of a CROSSBOW high-speed orientation sensor) and an omnidirectional camera. As mentioned is Section 3.3.1.7, a SICK PLS is used for low-level collision avoidance but the range data is also fed into the computer, making it available for high level navigation functions.. These sensors, and their location on the vehicle, are indicated in Figure 4.14.

This section presents the sensing system used to estimate the required parameters for the IALV strategy. Included is an analysis of the effects of sensing errors on the strategy, and a means of obtaining the full vehicle pose estimate (x, y, θ) with respect to a target pose.

4.3.1 CROSSBOW orientation measuring system

A reference direction is critical for any homing system. In this work, a magnetic compass is used for this purpose, as it was simple and available.

A CXM543 high-speed digital output orientation measuring system has been fitted to the front of the vehicle. The sensor can operate in either *angle* or *vector* modes. When in

Figure 4.14: An overview of the AT's sensors.

angle mode, the output is in the form of roll, pitch and azimuth, all measured in radians. When in *vector* mode, the output is the raw data from the three-axis accelerometer and three-axis magnetometer. This sensor is connected to the computer by a serial line.

In this work, the Crossbow unit is used in *angle* mode and data is fed into the **store** for use by other processes. The azimuth reading is filtered within the Crossbow unit, and

corrected for roll and pitch. Due to the highly vibratory environment, and the presence of EM fields produced by the vehicle's actuators, the vehicle's engine and alternator, the unit's azimuth reading can be extremely noisy, varying by as much as $\pm 20^\circ$ when the vehicle's engine is running, even when the vehicle is stationary — see the raw azimuth readings plotted in Figure 4.15 for an example. This noise is probably caused by the vibration of the vehicle which excites the accelerometers in the CROSSBOW unit. In *angle* mode, the unit uses these acceleration measurements to estimate the direction of the gravity vector in order to determine the magnetic field vector in the horizontal plane.

4.3.1.1 Complementary filtering

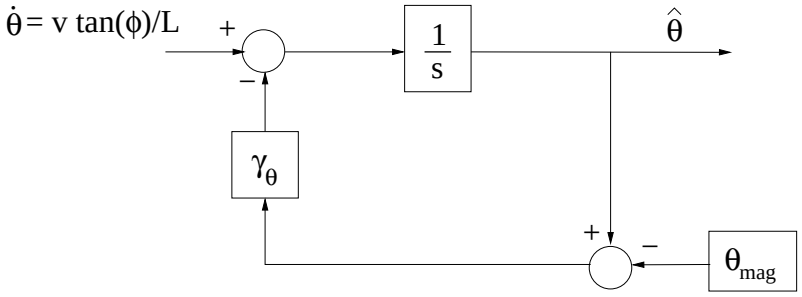
To reduce noise and improve the accuracy of the measured variable, the azimuth reading from the Crossbow unit is combined with an estimate of the vehicle's angular rate of rotation found from vehicle odometry. Recalling the standard vehicle kinematic model presented in Section 2.6.2, estimates of the vehicle's angular rate are given by the well-known kinematic equation of the angular rotation rate of a car-like vehicle:

$$\dot{\theta} = \frac{v \tan \phi}{L} \tag{4.27}$$

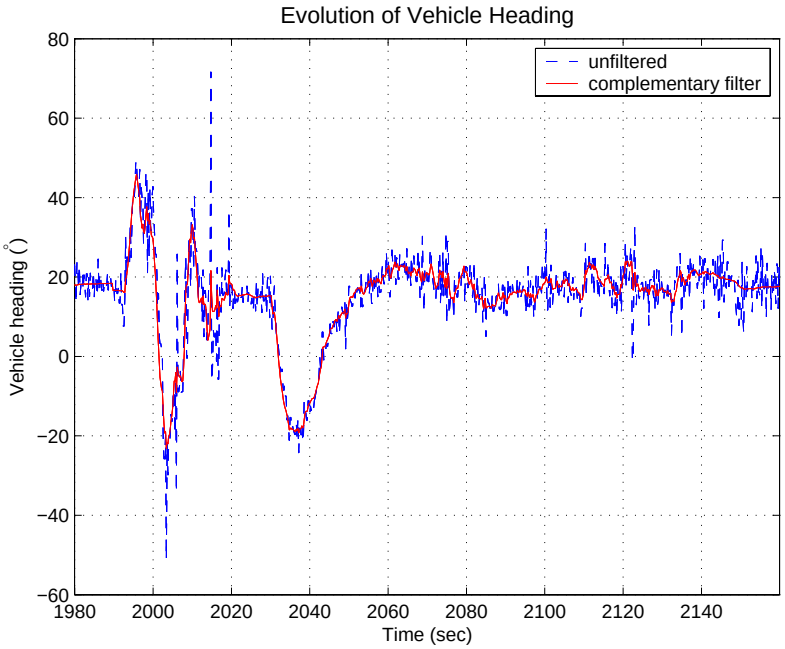
where $L = 1.2\,\mathrm{m}$ as measured on the vehicle, v is the velocity of the vehicle's rear-axle midpoint, and ϕ is the steering angle, both of which are measured.

Figure 4.15 (a) illustrates the complementary filter used for heading estimation. This filter operates by subtracting the heading reference signal, supplied by the magnetometer, θ_{mag} from the estimate, $\hat{\theta}$ [Buskey *et al.*, 2003]. This error signal is then scaled by γ_θ, which includes a component representing the sampling time, and subtracted from the angular rate signal calculated from vehicle odometry, $\dot{\theta}$. The resulting value is then integrated to give the heading estimate, $\hat{\theta}$. This process in effect provides low-pass filtering of the heading reference, θ_{mag}, and high-pass filtering of the vehicle angular rate measurement, $\dot{\theta}$. Here, $\gamma_\theta = 1.4$, at a magnetometer sampling rate of $5\,\mathrm{Hz}$, provides a good compromise on the reliance of each measurement signal. Figure 4.15 (b) shows the result of the application of the filter on real AT data, along with the original raw azimuth

reading.



(a) The complementary filter on vehicle heading.



(b) Comparison of raw vehicle heading measurement and filtered version.

Figure 4.15: The complementary filter on vehicle heading together with its application to real AT data.

As Figure 4.15 illustrates, the complementary filtering is highly successful at rejecting noise in the system with very little phase lag. However, in practice a much 'cleaner' azimuth reading is extracted by using the raw magnetometer readings from the Crossbow unit in *vector* mode, rather than the azimuth signal from the *angle* mode, which has been

corrected for roll and pitch using the accelerometer readings. This tactic assumes that the AT is operating on a relatively flat surface, which is one of the underlying assumptions of this work.

4.3.1.2 Discussion

As discussed in Section 2.4.2.2, the drawback of using a magnetic sensor for orientation estimates is that such a sensor measures the Earth's local magnetic field which can be distorted by, for example, the presence of ferrous structures and the dynamically induced fields produced by electric motors. The magnetic field direction also varies over the Earth's surface.

The sensor is mounted as far as physically possible from the AT's electric system and electric motors in order to minimise 'self-induced' interference. Distortions to orientation readings can still occur, as much of the intended operating environment will contain ferrous structures which interfere with the compass. However, for the purposes of this work, sensing orientation using a magnetic compass is adequate.

An alternative method to determine orientation would be to use unique landmarks which would enable vision to completely determine the pose of the vehicle, provided the requirements of triangulation are satisfied. Of course, this requires a more sophisticated vision system.

4.3.2 Tractor vision

Vision as related to pose stabilization and visual homing was reviewed in Sections 2.6.3.3 and 2.6.3.4. In this section, the vision system and the techniques used for tracking a set of coloured landmarks, estimating their range and bearing are presented.

Vision and robotics have been synonymous for many years. For reviews see for example [Batlle *et al.*, 2000; DeSouza and Kak, 2002]. The system developed here is relatively simple but is robust. Because the system can sense landmarks at a relatively long range when compared to other vision systems for pose stabilization, the vehicle is given a relatively large 'region of operation'. This is important because the AT is significantly larger

than most platforms used for pose stabilization research and requires the additional space to perform the necessary motions.

The vision system is designed to extract and track objects, relying on colour segmentation using a pre-learnt look-up table of the desired object colour. For testing purposes, red witches hats (also known as road cones) are used as landmarks for the visual homing process. However, the system is designed to track multiple sets of objects with differing colour profiles. Objects are assumed to lie on the ground-plane, from which their range can be estimated using a geometric model of the camera-mirror optics. A vehicle-object motion model is then used to track objects over time, and a complementary filter is used to reduce noise.

An overview of the algorithm follows:

1: Grab first image
2: Extract objects
3: Calculate object locations relative to image centre
4: Estimate object ranges
5: Initialise object list for tracking
6: **loop**
7: Grab image
8: Extract objects
9: Calculate object image location relative to image centre
10: Estimate object ranges
11: Match currently extracted objects to those in the list updating the relevant object {*temporal* filtering}
12: Perform complementary filtering
13: Write object list to the **store**
14: **end loop**

Following a description of the hardware, the sections which follow discuss each of the major processes in this loop. First, colour segmentation and range estimation are discussed.

The temporal filtering process is then described and a description of the complementary filtering process is provided. A method for obtaining a full vehicle pose estimate is then presented, based upon the IALV strategy and the storage of an additional characteristic of the target pose, the orientation at the target location. The section concludes with an analysis of the likely errors when using the IALV method and the these sensing techniques.

4.3.2.1 Hardware

An omnidirectional camera system, made by EyeSee 360 Inc., has been fitted to the front section of the vehicle, mounted above the engine as shown in Figure 4.14. Images from the camera are grabbed and fed to the computer using a Sensorray frame-grabber. The image formats supported by the camera are RGB or YCrCb colour images, or grey-level images.

The camera-mirror system consists of a WATEC LCL-217 colour camera, targeted at a curved mirror. The mirror is shaped such that each pixel in the image spans an equal elevation angle increment irrespective of its distance from the centre of the image [Ollis *et al.*, 1999]. Such mirror shapes are known as 'equiangular mirrors'. To maximise the available field of view, and to minimise vibration effects, a number of modifications to the camera-mirror assembly were made. These are discussed below.

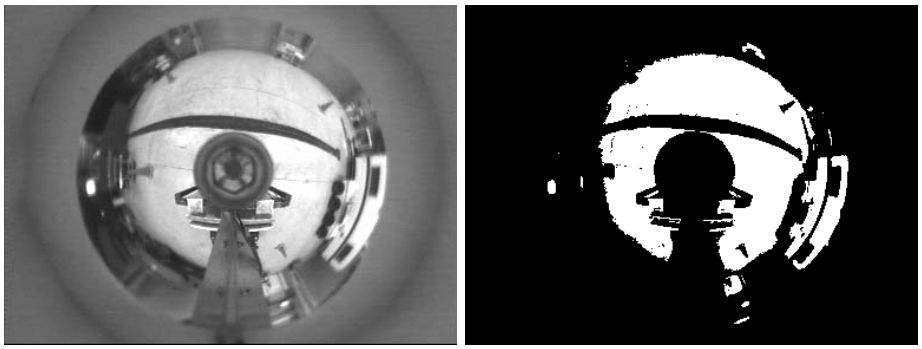
Firstly, the assembly has been stiffened to minimise relative motion between the mirror and camera caused by vibration on the Autonomous Tractor. This modification has reduced the radial field of view to approximately $330°$, due to the presence of the additional bracing.

Secondly, the camera has been shifted to sit closer to the mirror. By moving the camera closer to the mirror, the elevational view is reduced from the manufactured value of $140°$ to approximately $125°$. However, the portion of the image which contains the ground-plane has been increased, giving a better view of the immediate environment around the AT. The large elevational field of view also lead to problems when working outdoors due to saturation effects when the Sun, or any overhead light for that matter, shone directly into the camera or reflected from the mirror. To reduce these effects, a 'sun

shade' was fitted above the camera-mirror system which further restricts the field of view in elevation to approximately $110°$.

4.3.2.2 Colour segmentation

If landmarks are to be used for navigation, a means of extracting them from the omnidirectional images is required. An image is simply a matrix, each cell of which is a digital representation of the corresponding pixel on a camera CCD. Each cell of the matrix has some value representing the intensity of light for that particular pixel. In a grey level image, once digitised, this value is a number $\in [0, 1]$ or an integer $\in [0, 255]$. In segmenting grey-level images, it is common to use *thresholding* techniques in which all pixels in an image are set to ON if they exceed the threshold value and to OFF otherwise. An example of this process is shown in Figure 4.16 (b), where a grey-level image from the omnidirectional camera has been thresholded at 180 — the original image is shown in Figure 4.16 (a). Thresholding is but one tool in the process of *image segmentation*. For a complete



(a) The original intensity image. (b) Image thresholded at 180.

Figure 4.16: Example of image thresholding of a grey-level image, threshold level set to 180 where image intensity is represented by an integer $\in [0, 255]$.

treatment of image segmentation processes, refer to any of the commonly available texts, such as [Russ, 1992]. Here a brief explanation in relation to colour imaging is given.

Image segmentation is the process of partitioning an image into a set of non-overlapping regions whose union is the whole image [Haralick and Shapiro, 1992]. In order to be use-

ful, these partitioned regions should have some meaning to the application at hand. In this work, colour is used as the main identifying characteristic for landmarks. Hence, the meaningful regions in an image are those which contain the desired colour (or colours) and those that do not. For example, red road cones (also known as witches hats) are used as 'artificial landmarks' and meaningful regions of the image for control purposes are those that correspond to the colour of red witches hats. Of course, it is easy to 'confuse' such a simple system as it extracts all objects matching the colour description. Other cues can then be used to reinforce the selection — for example, in this work the expected motion of the objects is used to track, and hence filter, the objects over time.

Extraction of 'interesting' objects from images has been a goal of image processing for over three decades, with advances mainly coming in the form of increased processing power. The amount of data to analyse in an image makes real-time processing difficult, particularly with colour images. Much of the work in the field assumes that lighting is sufficient for good exposure, diffuse and time invariant. In addition, the object to be segmented normally consumes a significant proportion-of the image — an example is the 'face tracking' literature, see e.g. [Bradski, 1998]. When using colour, further difficulties arise as the colour of an object varies with the illuminant colour, reflectance of the object, illumination geometry, viewing geometry and other sensor specific parameters [Batlle *et al.*, 2000].

However, the use of colour can enable the determination of the *identity* of an object, and also the *location* within an image of objects with known colour profiles [Swain and Ballard, 1991]. When using colour as a defining characteristic, this can be achieved without knowledge of the shape of the object. This latter point is particularly important for mobile robots because the shape of objects varies with viewpoint — omnidirectional vision distorts objects with viewpoint even further.

A colour space is a mathematical coordinate system used for assigning numerical values to colours. Colour spaces are usually used in relation to computers, televisions and other graphical interfaces. There are many different colour spaces, all claiming particular advantages over each other for their usefulness in particular situations. As Batlle *et al.*

[2000] note, the question of which colour space to use does not have a single or perfect solution.

The most well known colour space is RGB, in which each pixel in an image is represented by a vector, whose elements are the intensity in the Red, Green and Blue parts of the light spectrum [Russ, 1992]. Closely related to the RGB colour space is YCrCb space. In this space, the Y component combines the red, blue and green components in proportion to the human eyes sensitivity to each, giving a representation of image intensity, and the Cr and Cb components represent the chrominance. Both the RGB and YCrCb colour spaces are hardware oriented schemes [Russ, 1992], with the latter space used for colour computer displays. Both colour spaces are cubic.

Other colour spaces are more closely related to human perception, for example the Hue, Saturation and Value (HSV) colour space. This colour space involves a non-linear transform from the RGB space and is represented by a cone in three dimensional space. In this space, hue is a representation of colour as described by wavelength, saturation is the *amount* of colour present, and value is a representation of the light intensity.

Thresholding techniques can also be used for colour images, where each component of the colour image is thought of as an individual image. Thresholds can then be set on the components of the colour vector, and the resulting images combined with a pixel-by-pixel AND operation.

Histogramming and variations of this concept are also popular in the colour segmentation and tracking literature and the related problem of Image Retrieval from large databases of images. A one-dimensional histogram of an image is obtained by counting the number of times each intensity level occurs in the image array — such a histogram can be used to set threshold levels for example. Similarly, a colour histogram is obtained by discretizing the image colours, and counting the number of times each colour occurs in the image [Swain and Ballard, 1991] — for an RGB, or any 3-space image, this can occur in three-dimensions. Such histograms can be used to identify and locate objects.

Early work by Swain and Ballard [1991] was based upon three-dimensional histograms of ‘quantised’ colour, with object identification and localisation performed based

on pre-taught histograms. Many later techniques build upon the work of Swain and Ballard [1991]. For example, Rasmussen *et al.* [1996] use three-dimensional histogramming but fit a model to the pre-taught data from which image segmentation proceeds. Improvements to Swain and Ballard [1991] histogramming technique have used the relationship between the colours of neighbouring pixels to improve object / image recognition (see for example [Huang *et al.*, 1999; Chang and Krumm, 1999]).

Others have used techniques based on two-dimensional tables which are adaptive to illumination, rather than capturing the three-dimensional colour data, see for example [Lee *et al.*, 2001]. This of course substantially reduces the computational overheads of storing a three-dimensional histogram and the associated costs of looking up such a representation.

Initial investigations

In this work, the initial investigations in segmentation were centred on the RGB colour space, relying on an image subtraction between the $\underline{R}$ and $\underline{G}$ components of an image and the subsequent thresholding of the resulting image to find the desired object. The process used is described below, where $\underline{R}$, $\underline{G}$ and $\underline{B}$ represent the matrices of the colour components vectors, also known as the colour planes.

1: Apply a spatial mask to the image, eliminating the camera-mirror assembly, mounting brackets and the AT.
2: Subtract the green colour component from the red ($\underline{R} - \underline{G}$), scaling to $\in [0, 1]$.
3: Threshold the image at a manually tuned threshold level.
4: Perform morphological opening on the thresholded image with a 3x3 mask — this reduces the occurrence of isolated 'noisy' pixels.
5: Perform morphological closing on the the resulting image with a 3x3 mask — this reduces the occurrence of 'holes' in blobs.
6: Extract blobs and perform image labelling.
7: Eliminate blobs which are too large or too small.

Over a series of images, an 'optimum' threshold could be obtained but was highly dependent on lighting conditions. Figure 4.17 shows the steps in the process.

Figure 4.18 shows the results, for the same image set, of segmenting a series of images in which the AT was driven towards a set of five witches hats. The figure illustrates the sensitivity to the thresholding levels. The first set of results were obtained with a well-tuned threshold, while the next two sets show the results of setting the threshold too low and too high respectively. Segmentation in this colour space is quite susceptible to changing lighting conditions, requiring repeated 'tuning' of the threshold.

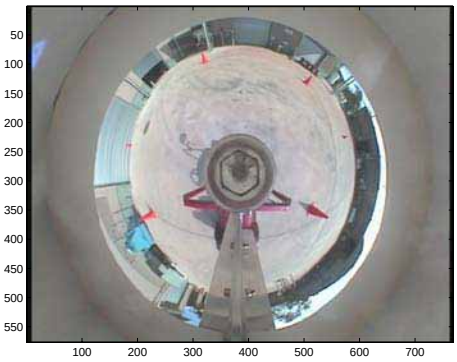
This lead to the investigation of segmentation in HSV space. Although much more successful under varying lighting, the transformation from RGB to the HSV colour space is nonlinear and was far too expensive in terms of processor use and processing time.

Two-dimensional loop-up table segmentation

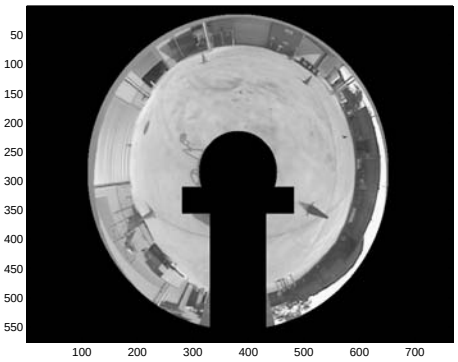
The susceptibility to lighting conditions in RGB space, and the computational overheads of converting to HSV space lead us to investigate segmentation based upon a two-dimensional histogram method in YCrCb space. This colour space is provided directly by the frame-grabber at no additional computational cost. Segmentation in this colour space provided the best compromise between speed and segmentation effectiveness.

The Cr and Cb colour components from the image are used to create a bi-variant histogram on the desired colour or colours, with the first axis being Cr and the second Cb. This histogram is created through a Graphical User Interface in which the user clicks on objects (regions) of the desired colour in a training image. The Cr and Cb counts for the selected regions are then recorded in a two-dimensional histogram. This histogram itself can be viewed as an image, and standard image processing tools, such as morphological processes, can be used to manipulate it. The histogram is transformed to a simple binary bi-variant look-up table by applying a threshold to the colour counts in the histogram. The bi-variant look-up table is then used for image segmentation.

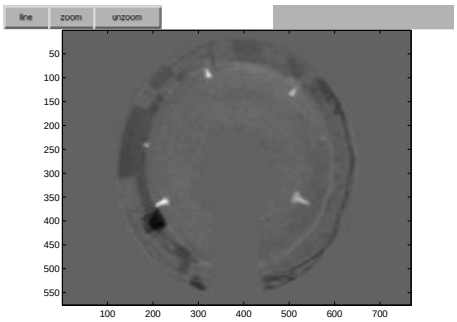
A two-dimensional (or bi-variant) look-up table can be viewed as a means of relating two input variables to a single output. In this case, the input variables are the Cr and Cb



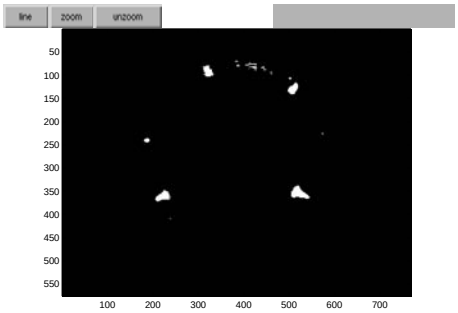
(a) The original image.



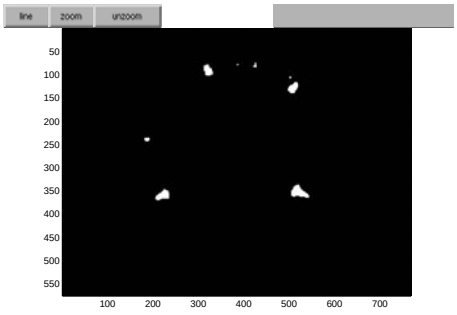
(b) The masked image.



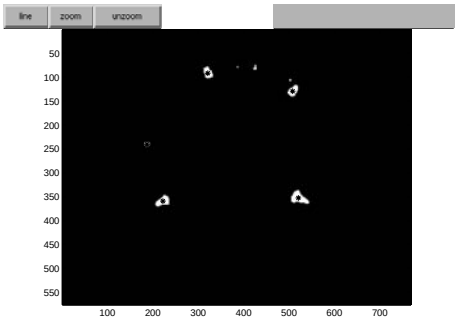
(c) Red component minus the green component.



(d) The image in (c) is thresholded, here 0.12 was used.

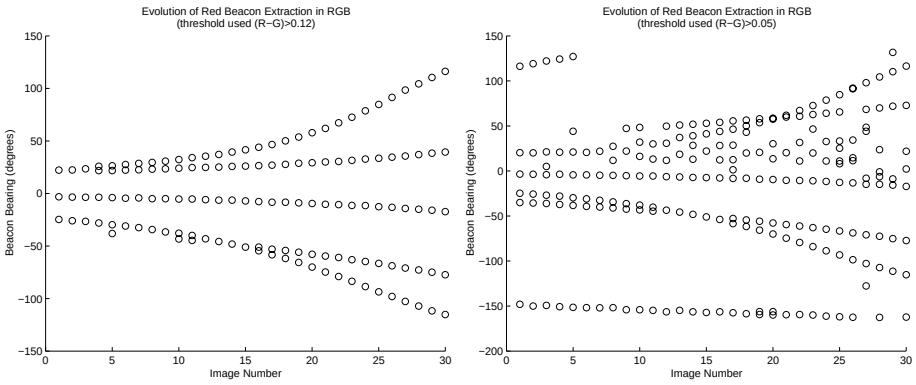


(e) Morphological opening and closing of image resulting from the image in (d).



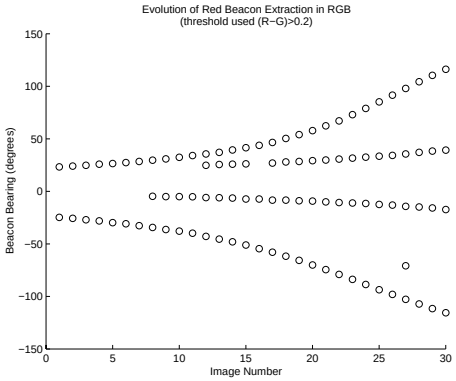
(f) Results of blob extraction with very big and very small blobs eliminated. The star indicates the centroid of the blob.

Figure 4.17: The segmentation process in the RGB colour space.



(a) For this image set a threshold of 0.12 is optimal. Note the appearance of the lower beacon as it becomes distinguishable.

(b) Results from a lower than optimal threshold.



(c) Results from a higher than optimal threshold.

Figure 4.18: Evolution of landmark bearing over a series of images showing the effect of different threshold levels. Note in image (b) the landmarks at the start of the image sequence at a bearing of between $100°$ and $150°$. This is in fact a fire hydrant which is in the middle of the testing area.

values of the current image, and the output is whether the particular pixel is defined as being 'ON' or 'OFF'.

After training the system for a particular colour or group of colours, colour segmentation proceeds as described in Algorithm 1. Figure 4.19 (a) shows the input image, while Figure 4.19 (b) illustrates the results of the segmentation process described in Algorithm

Algorithm 1 Colour segmentation of an image.

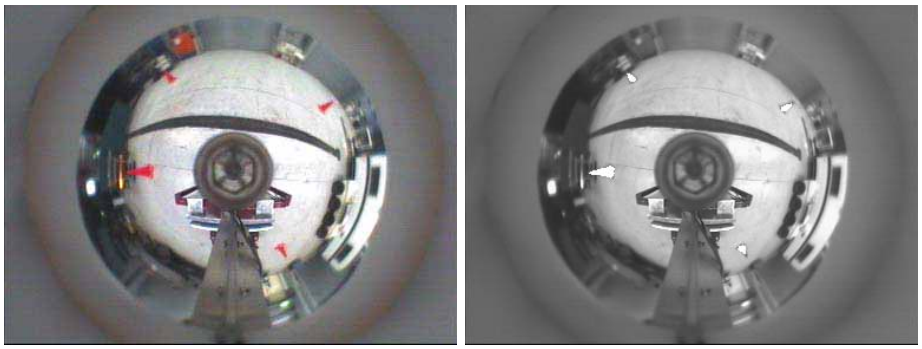
1: Apply the mask, eliminating superfluous elements in the image.
2: Perform look-up using current image and the 2-d look-up table of the desired colour.
3: Extract blobs from the resulting binary image and perform image labelling.
4: Eliminate blobs which are too large or too small.
5: Calculate individual blob properties (position of blob centroid in image, pixel count etc.).
6: Calculate blob pixel distance and bearing w.r.t image centre.
7: Estimate blob range using geometric model of camera-mirror and the flat-Earth assumption (described in Section 4.3.2.3).

1. Figure 4.19 (c) shows the look-up table used.

4.3.2.3 Range estimation

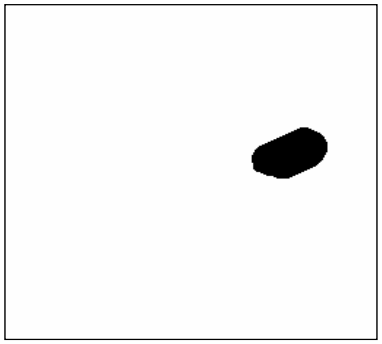
Using a flat-Earth assumption, an estimate of range can be determined in a similar manner to Horswil's range from height-in-image method [Horswill, 1993; Das *et al.*, 2001] if the geometry of the camera-mirror system is known. An alternative to using the geometry of the system is to determine an empirical relationship between ground-plane range and radial image distance. Another alternative method to determine range is to use image reconstruction techniques based upon the vehicle's motion, see for example Srinivasan et al. [Chahl and Srinivasan, 1996; Srinivasan *et al.*, 1997; Nagle and Srinivasan, 1996; Chahl and Srinivasan, 1997a]. However, initial investigations with simple versions of these optic flow type techniques proved that the AT odometry measurements were too coarse to achieve accurate estimates of range.

The mirror in the omnidirectional camera-mirror assembly has equiangular optics meaning that for a given angle of incidence into the mirror, the reflected ray is elevated by a particular gain depending on the camera-mirror separation distance [Chahl and Srinivasan, 1997b]; for an illustration of this point refer to Figure 4.20. The details of this mirror's design are given in [Ollis *et al.*, 1999]. This mirror is slightly different in shape to the equiangular optic design originally presented by Chahl and Srinivasan [1997b] which relies on a small angle approximation. However, Srinivasan's mirror shape design has a closed form solution as opposed to the numerical method required to produce the



(a) The image prior to segmentation.

(b) Segmented image.



(c) An example lookup table of the red witches hats contained in the image of Figure (a). Image inverted for clarity.

Figure 4.19: Segmentation using the two-dimensional look-up table.

shape of the EyeSee 360 mirror. This closed form solution is used in this work, and it is noted that the designers of the EyeSee 360 mirror (i.e. [Ollis *et al.*, 1999]) also used this closed form solution for analysis purposes.

The equation describing the surface of equiangular mirrors is [Chahl and Srinivasan, 1997b]:

$$\left(\frac{r}{r_o}\right)^{-\frac{1+\alpha}{2}} = \cos\left[\frac{\theta(1+\alpha)}{2}\right] \tag{4.28}$$

where the parameters are defined with reference to Figure 4.20.

As received from the manufacturer, the mirror was designed to operate with $r_o = 14\,\mathrm{cm}$ and with an α value of 11. As noted earlier, the camera has been moved closer to

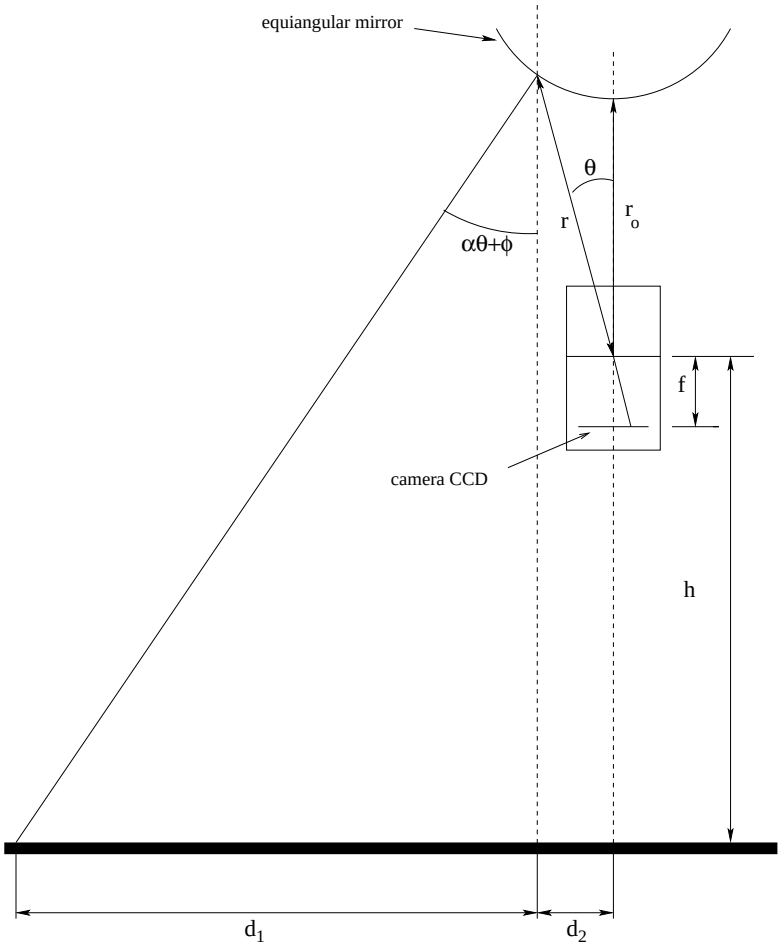


Figure 4.20: The geometry of the omnidirectional camera system related to the ground-plane. Diagram adapted from [Chahl and Srinivasan, 1997b]

the mirror (reducing r_o) in order to increase the proportion of the image containing the ground-plane — this in effect reduces the angular magnification, α, and can have an effect on the assumption of the constancy of α at high angular elevations [Chahl and Srinivasan, 1997b]. However, α remains constant over *most* of the angular range of the mirror [Chahl and Srinivasan, 1997b].

Using the geometry of the system, the ground-plane range can be estimated using a flat-Earth assumption, given a pixel distance from the centre of the image. Referring to Figure 4.20, the distance d_1 can be calculated by:

$$d_1 = (h + r\cos\theta)\tan(\alpha\theta + \phi) \tag{4.29}$$

and the distance d_2 is given by:

$$d_2 = r \sin\theta \tag{4.30}$$

Adding these together[3] gives the theoretically determined ground-plane distance, $\hat{R}$. Combining equations 4.28, 4.29 and 4.30, the equation for $\hat{R}$ is:

$$\begin{aligned}\hat{R} = &\ d_1 + d_2 \\ = &\ r_o \left[\cos\left(\frac{\theta(1+\alpha)}{2}\right)\right]^{-\frac{2}{1+\alpha}} \sin\theta + \\ &\left[h + r_o \left[\cos\left(\frac{\theta(1+\alpha)}{2}\right)\right]^{-\frac{2}{1+\alpha}}\right] \tan(\alpha\theta + \phi)\end{aligned} \tag{4.31}$$

where the parameters are defined with reference to Figure 4.20 and $\hat{R}$ refers to the range estimate (i.e. $d_1 + d_2$).

Now to map a ground-plane distance into the image plane a relationship between θ and radial pixel distance g is required. Referring to Figure 4.21:

$$\tan\theta = \frac{u}{f} = \frac{gp}{f} \tag{4.32}$$

where u is the distance from the centre of the image measured on the image plane, g is the distance in the image measured in pixels, p is the pixel pitch and f is the focal length of the camera. Using the known radius of the mirror at its outer edge r_{min}, and the corresponding pixel distance of this edge in an image g_0, a relation for f is determined as follows:

$$\frac{g_0 p}{f} = \frac{r_{min}}{r_o + r_{min}} \tag{4.33}$$

Here $r_{min} = 3.68\,\mathrm{cm}$, and the distance of its edge as it appears in the image is $g_o = 172\,\mathrm{pixels}$. The pixel pitch p is $17.639 \times 10^{-6}\,\mathrm{m}$ found from the image size ($384\,\mathrm{pixels} \times 288\,\mathrm{lines}$)[4] and the CCD diagonal which is $\frac{1}{3}\,\mathrm{inches}$. This leads to the following relationship for f:

$$f = \frac{p g_o (r_o + r_{min})}{r_{min}} \tag{4.34}$$

[3] In fact $d_2 \ll d_1$ and d_2 could be omitted with little error.

[4] For a YCrCb image, the intensity image (Y), is double the resolution of the chrominance images (Cr and Cb). The size specified here is the chrominance image size.

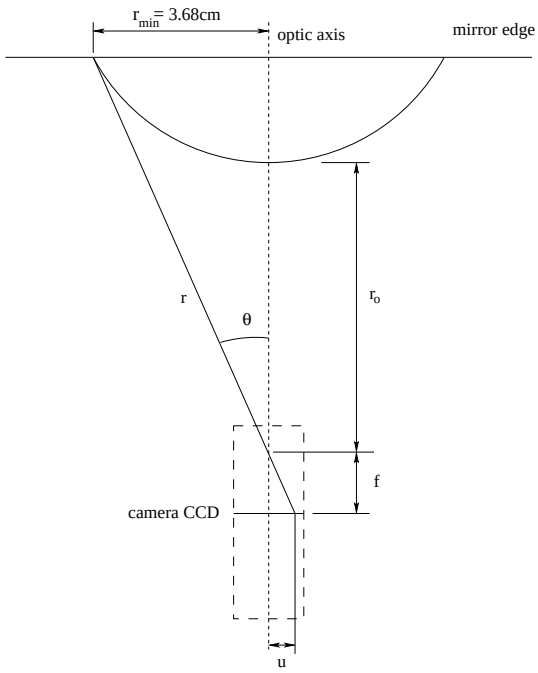


Figure 4.21: Relationship between f and u.

and the relationship for θ at any pixel distance g (measured with respect to the centre of the image) is then given by:

$$\theta = \arctan\left[\frac{gr_{min}}{g_0(r_{min} + r_0)}\right] \tag{4.35}$$

Substituting the relevant values leads to:

$$\theta = \arctan\left[\frac{21.395 \times 10^{-3} g}{36.8 \times 10^{-3} + r_o}\right] \tag{4.36}$$

where r_o is given in metres and g in pixels.

Determining imaging parameters

A method for estimating the unknown camera parameters (r_0, ϕ, α and h) is now presented. These must be identified in order to use Equation 4.31 for ground-plane range estimation. To determine these parameters, a series of images of a red road cone at specific distances from the camera was gathered and the corresponding pixel distance from the centre of the image determined. The centre of the image was found by fitting a circle

Parameter	Value
r_o	$0.0836\,\mathrm{m}$
h	$1.2961\,\mathrm{m}$
α	7.1411
ϕ	$0.1680\,\mathrm{rad}$

Table 4.1: Parameter values as determined using `fminsearch` in MATLAB.

to the outer edge of the mirror, visible in the image. Numerical optimisation (MATLAB's `fminsearch`) was used to adjust these imaging parameters so that the estimated range $\hat{R}$ matched the true, measured ground-plane range R. The error (or cost) function used to optimise the parameters was:

$$J(\underline{R}, \underline{\hat{R}}) = \sum_{i=1}^{n} \left(\left| R_i - \hat{R}_i \right| (R_i < 5) w_1 \right) + \sum_{i=1}^{n} \left(\left| R_i - \hat{R}_i \right| (R_i > 5) w_2 \right) \tag{4.37}$$

where R was the actual measured range for a particular pixel distance, $\hat{R}$ was the estimated range for a particular pixel distance, n was the number of measurements, and $w_1 = 5$ and $w_2 = 1$ were weighting terms for ranges below and above $5\,\mathrm{m}$ respectively. A weaker weighting was placed on $R_i > 5\,\mathrm{m}$ as these were less accurate due to the reduction in resolution with rising angular elevation in the image. The resulting parameters values for this system are shown in Table 4.1. These values correlate well with the parameters which can be estimated directly in the system, i.e. the height of the camera is approximately $1.3\,\mathrm{m}$ and the camera mirror separation is approximately $0.1\,\mathrm{m}$. Also, $\alpha = 14$ for the camera-mirror system as manufactured — by moving the camera closer to the mirror, α is expected to decrease as indicated by these results. In addition, the focal length of the lens (Cosmicar TS812A) is specified as $8\,\mathrm{mm}$. Substituting the estimated imaging parameters into Equation 4.34 gives a focal length of $9.9\,\mathrm{mm}$, which is also a relatively close match.

Range estimation results

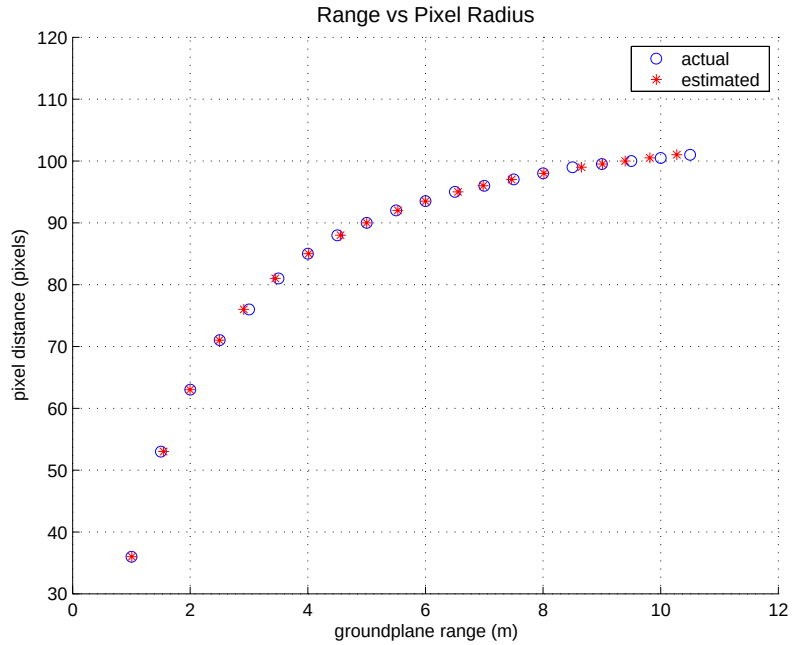
Figure 4.22 (a) shows a plot of estimated range, as determined using Equations 4.31 and 4.36, versus pixels distance. Range was estimated using the parameter values of Table 4.1. Also shown is the true ground-plane distance, versus pixel distance. Figure 4.22 (b) illustrates the difference between the range estimate and the measured value. The match is excellent for ranges of up to $9\,\mathrm{m}$, with the error between experimental data and estimated range being less than $0.1\,\mathrm{m}$ for ranges less than $8\,\mathrm{m}$. Inaccuracies at greater ranges can be attributed to measuring errors and to degradation of the constant α assumption at high angular elevations [Chahl and Srinivasan, 1997b], i.e. high radial distances.

Note that the maximum range of the sensor for this particular object (a large red witches hat) is approximately $12\,\mathrm{m}$. This limitation is largely due to the reduction in angular resolution with rising elevation in the mirror. Similar range limitations would be encountered with a monocular camera. The minimum range is limited to approximately $1\,\mathrm{m}$ due to the presence of the vehicle itself, and the central bolt which holds the mirror to the camera assembly.

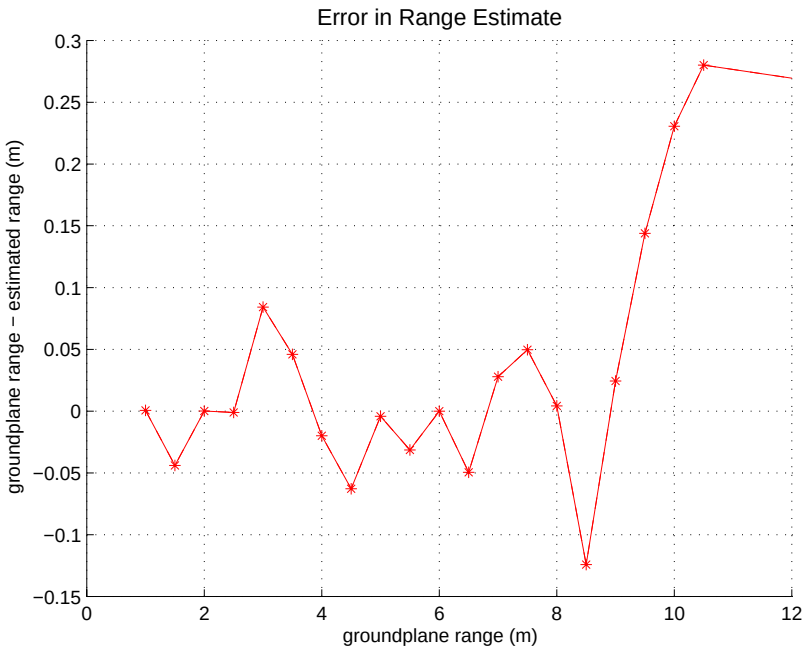
Variation with landmark bearing

This range estimation technique relies on the flat-Earth assumption and perpendicular alignment of the camera-mirror system vertical axis with the ground-plane. Deviations from these assumptions can lead to distortions in the range estimate. Most of the terrain encountered by the vehicle will be relatively flat. Hence, the major source of error is camera-mirror ground-plane perpendicular misalignment. This can lead to landmarks at equal ground-plane ranges but different bearings having different range estimates. Perfect alignment of the camera-mirror system is difficult because:

- The ground is not always ‘flat’.
- The platform itself vibrates considerably.
- The camera-mirror system may be poorly mounted — even small errors can have a



(a) Estimated and measured range plotted against pixel radius. The estimated data is plotted as an '*', while the actual ground-plane range is plotted as a 'o'.



(b) Error between estimated and measured range plotted against measured range.

Figure 4.22: Range estimation results.

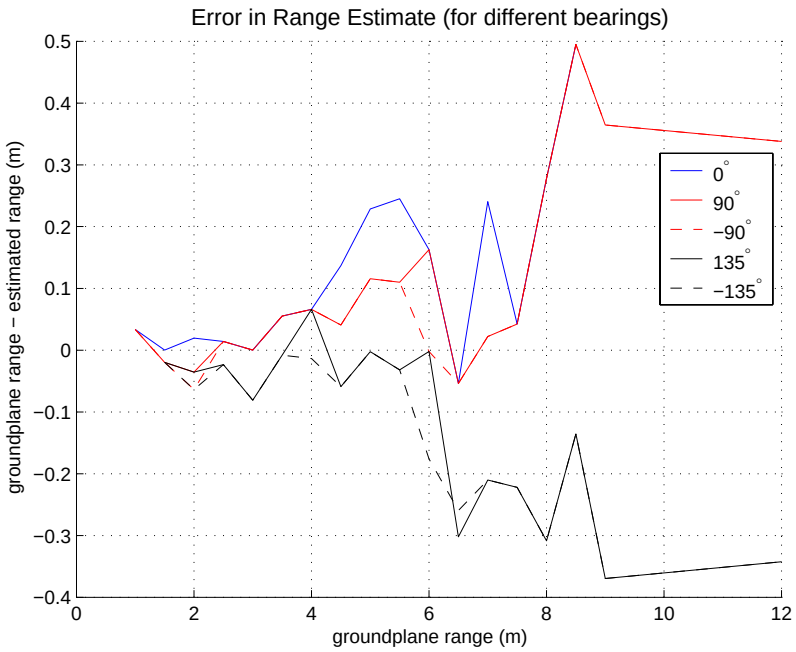


Figure 4.23: Variation of range estimate with landmark bearing.

considerable influence.

In addition, small aberrations on the mirror surface can lead to large range estimate variations. At larger ranges, a difference of a single pixel in the image leads to a difference of several metres in ground-plane range — see for example Figure 4.22 for which ground-plane ranges of $8 < R < 11\,\text{metres}$ show a pixel distance change of only $3\,\text{pixels}$ (i.e. a pixel distance of 98 to $101\,\text{pixels}$ in the figure).

Figure 4.23 illustrates the differences in range estimation for five different bearings, taken under conditions where the AT was on a relatively flat surface with the engine off. Here, the system was aligned with the local ground-plane as best as physically possible. Redesigning the camera-mirror system to accommodate more sophisticated alignment adjustments may be a fruitless exercise because of the highly vibratory environment and the fact that small misalignments lead to considerable differences in range estimation.

In operation, the error in the range estimate will have two sources:

1. Systematic error caused by misalignments between the camera-mirror system and the local ground-plane, and camera-mirror aberrations.

2. Non-systematic error caused by vibrations to the camera-mirror platform from the engine and motion of the vehicle.

Non-systematic errors seems to 'average' out over several measurements and do not significantly affect navigation performance. However, the systematic errors lead to some profound results in position and pose stabilization. In terms of this sensing arrangement it is much easier to stabilize to a *pose* because all systematic errors in the system will be negated as the target image will 'look' like the current image as the vehicle gets closer to home. For position stabilization this is not the case because the vehicle may approach the home position with *any* orientation meaning that the image at the target will not have the same systematic errors as the images currently seen when the vehicle approaches the target position.

4.3.2.4 Temporal filtering (tracking)

After segmentation and range estimation, the colour objects are tracked over time in order to reduce the effects of incorrect image segmentation. In this work, a rather simple but effective means of tracking objects over time was developed based upon the frequency and recent history of sightings, and the expected evolution of the relative object position. This is termed *temporal* filtering, as distinct from the *spatial* filtering discussed in Section 4.3.2.2.

Given that the Autonomous Tractor is a nonholonomic, car-like vehicle its means of translation and rotation are constrained. Thus, the motion of objects relative to the vehicle can be predicted with knowledge of the vehicle's translational and rotational velocities. For example, if an object is ahead and to the right of the vehicle, and the vehicle maintains a straight course, the object's bearing should evolve clockwise with respect to the vehicle. The temporal filter is based upon a model of this vehicle-object relative motion. Referring to Figure 4.24, the equations for the motion of an individual object are (for a car-like vehicle):

$$\dot{R} = -v\cos\beta \tag{4.38}$$

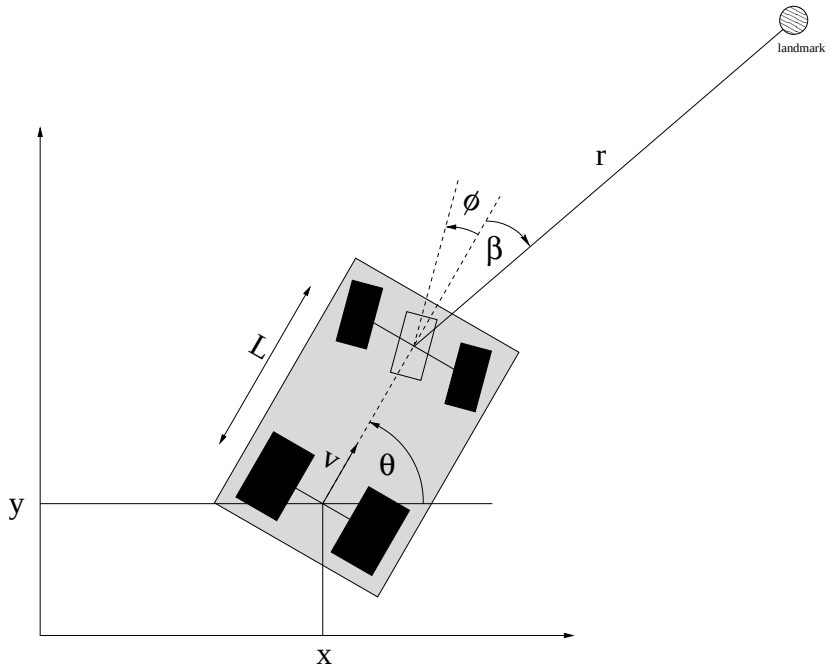


Figure 4.24: The vehicle, a landmark and the coordinate system used. All angles are counter-clockwise positive.

$$\dot{\beta} = \frac{v \sin \beta}{R} - \frac{v \tan \phi}{L} \tag{4.39}$$

where R is the object's ground-plane range relative to the AT, v is the vehicle's velocity, β is the relative orientation of the object with respect to the AT, ϕ is the vehicle's steering angle, and L is the length of the vehicle. If v, ϕ and L, are known or measured, $\dot{R}$ and $\dot{\beta}$ can be calculated and used to predict future values of R and β through simple Euler integration. In practice, this method is used to predict future object positions. This allows us to get 'correspondence' between tracked points but this correspondence is not a necessary feature of the system, it just helps with noise reduction in the event of falsely detected beacons resulting from poor image segmentation.

To track objects over time, linked lists are used where each node in the list represents an object. Each node consists of the following information:

– relative bearing

– pixel distance w.r.t image centre

– estimated range

– time first seen

– time last seen

– number of times seen

– status (good or bad)

The first image captured is used to bootstrap the list with all extracted objects assumed to be 'good'. The list is maintained as each subsequent image is processed, with the 'good' objects being stored for use by other processes (e.g. to control the vehicle). The *temporal* filtering process is described in Algorithm 2. For landmark tracking, we wish

Algorithm 2 Temporal filtering

1: Grab the first image
2: Extract the beacons from the image
3: Boot strap the list with the first set of beacons
4: **loop**
5: Grab next image
6: Extract beacons from image using Algorithm 1
7: Match beacons in list to current beacons using Equations 4.38 and 4.39 and Euler integration for prediction of object positions
8: **if** no match **then**
9: add another node to the list
10: **end if**
11: Upgrade relevant nodes to 'good' status, based on times seen and when last seen
12: Demote relevant nodes to 'bad' status, based on when last seen
13: Eliminate old nodes, based on when last seen
14: Perform complementary filtering using vehicle odometry (described in Section 4.3.2.5)
15: Write properties of 'good' beacons to the STORE for use by other processes
16: **end loop**

the image to be as large as possible because the beacons assume a relatively small proportion of the image (at $8\,\mathrm{metres}$ a large road cone occupies approximately $10\,\mathrm{pixels}$ in a $768\,\mathrm{pixels} \times 576\,\mathrm{lines}$ colour image[5].) This large image size limits the processing speed to

[5]This is the maximum image size of the *intensity* plane of the image (i.e. the Y component). The colour

approximately $7\,\mathrm{Hz}$ including image acquisition. However, the loop rate has been reduced to $5\,\mathrm{Hz}$ to ensure that there is adequate time for other processes to run.

The system is designed to enable the use of multiple lists, and hence tracking of different groups of colour objects. Each list can use its own set of parameters for promotion and demotion etc. For example, these different types of objects could be landmarks and obstacles. Within the algorithm there are several tuning parameters:

- Object age
  - Age of promotion
  - Age of demotion
  - Age of elimination
- Constraints on object's relative position evolution
  - Depends on accuracy of vehicle motion knowledge and errors in extraction of object relative position (e.g. vehicle vibration may induce errors in where the object appears in the image).

When tracking landmarks for positioning purposes, the parameters are adjusted so that landmarks will remain stable in the list. When tracking obstacles, the parameters are adjusted so that list maintenance is much more responsive when adding and deleting items from the relevant list.

For an example of the temporal filtering process, refer to the raw range and bearing results in Figures' 4.25 (b) and 4.26 (b), which show tracking of a witches hat over a period of approximately $70\,\mathrm{s}$.

4.3.2.5 Complementary filtering

As described in Section 4.3.1.1, the idea of a complementary filter is to fuse the complementary features of different sensing sources to produce a more accurate measurement

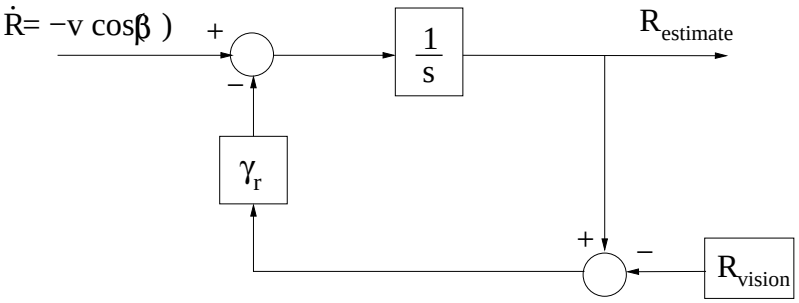
planes (CR and Cb) have *half* this resolution.

of the desired quantity. In the case of *range* estimation, an estimate of range is available from the vision system. An estimate of how a particular object's range should change based on the motion of the vehicle and the relative orientation of the landmark is also available, as given by Equation 4.38. This quantity is obtained from vision and odometry. Object range as determined from vision is combined with the rate of change of range measurement using a complementary filter as shown in Figure 4.25 (a). Some representative results of this process are given in Figure 4.25 (b). For these results, the gain was set equal to $\gamma_r = 1.3$ at a vision sampling rate of $5\,\mathrm{Hz}$. In practice, it was found that filter performance is improved at low vehicle speeds by setting the gain to a lower value — for $|v| < 0.1\,\mathrm{ms}^{-1}$, $\gamma_r = 0.1$.

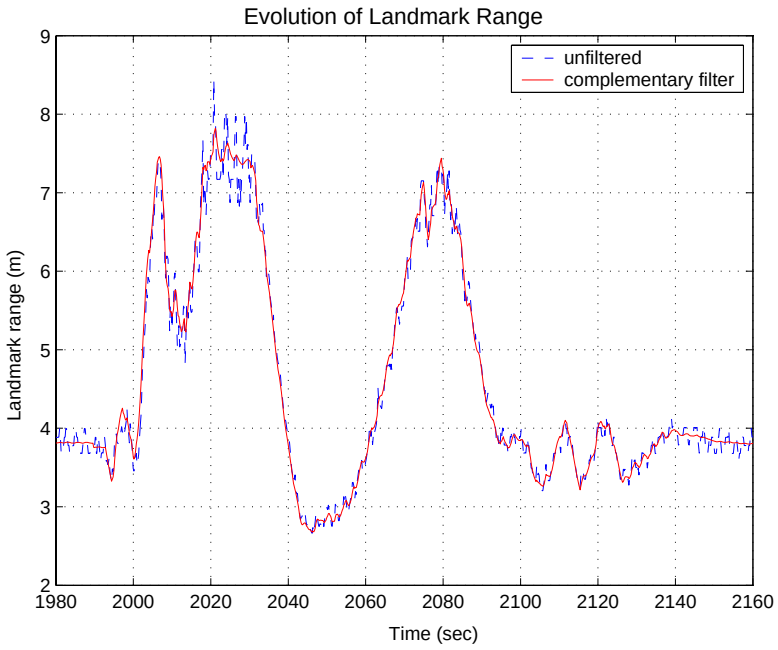
Similarly, for the bearing estimate of an object the bearing angle from vision can be combined with the rate of change of bearing angle given by Equation 4.39, as shown in Figure 4.26 (a). Sample results of this process are shown in Figure 4.26 (b). Here it is noted that although the data has been smoothed somewhat, the filtering process on the bearing measurement actually gives a phase lead to the estimate. This is probably due to the fact that the measurements of β and $\dot{\beta}$ are coupled. The filter gains are as used for the range estimate.

In both cases the complementary filter was compared with a second-order Butterworth filter on the range and bearing measurement, empirically tuned with a cut-off frequency of $1.25\,\mathrm{Hz}$. The improvement in signal-to-noise ratio provided by the Butterworth filtering did not justify the problems that would be encountered by the large phase-lag introduced by the filter (of the order of $1\,\mathrm{second}$). In any case, the complementary filtering has far superior performance in terms of noise reduction and introduced almost no phase lag. An Extended Kalman Filter was also applied to the combined data (range, bearing and odometry) but it was found that tuning and software implementation was less favourable against the single parameter complementary filters presented here.

For an illustration of the complete landmark tracking process, please refer to the video *lm_tracking_sequence.avi* contained in the supplementary material in Section A.1. This video shows the AT moving through an industrial shed-like environment, tracking a set of



(a) The complementary filter on landmark range.



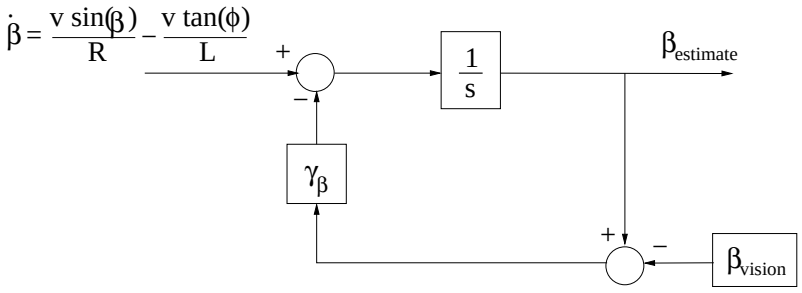
(b) Evolution of the landmark range with respect to the AT as the vehicle moves through a workspace.

Figure 4.25: The complementary filter on landmark range, with an example of the technique applied to real AT data.

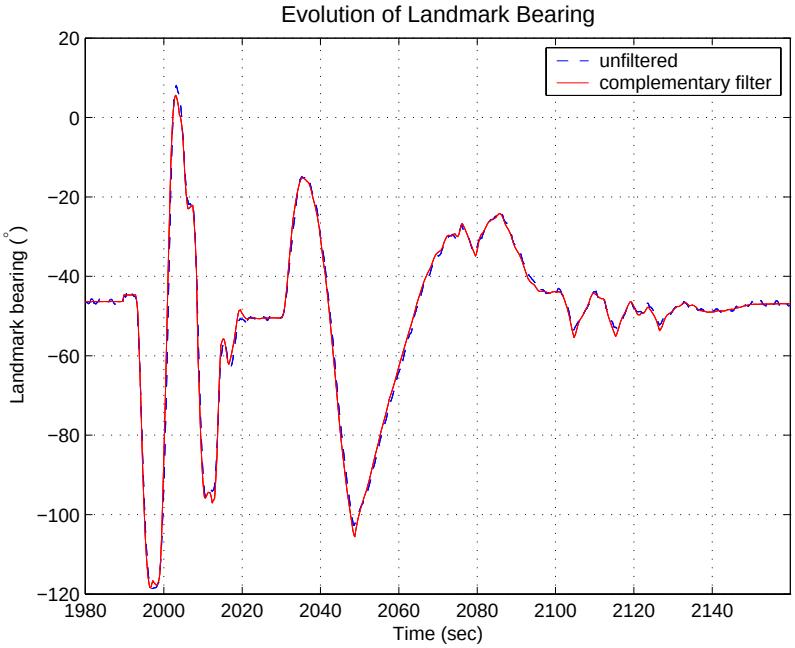
witches hats.

4.3.3 Implementing the IALV and determining vehicle pose

Obtaining a pose estimate using the IALV strategy is straight-forward as it provides the distance and heading to the target location, relative to some reference direction. As de-



(a) The complementary filter on landmark bearing.



(b) Evolution of the landmark bearing with respect to the AT as the vehicle moves through a workspace.

Figure 4.26: The complementary filter on landmark bearing, with an example of the technique applied to real AT data. Tracking for this filter is not as good and in fact *leads* the original signal. This is due to coupling in the filtering equations as β appears in the calculation of $\dot{\beta}$.

tailed in Section 4.3.2.1, the camera-mirror system is mounted over the centre of the front axle, and is capable of finding coloured objects and returning their bearing with respect to the vehicle, β_i, and their ground-plane range, R_i. The magnetometer on the vehicle

returns the vehicle's current orientation, ${}^{N}\theta$, with respect to North.

The IALV strategy defines the target position as the local origin of the coordinate system. Because the desired vehicle orientation at the target pose is not always aligned with North, the vehicle orientation at the target pose also needs to be stored. All subsequent vehicle orientation measurements are then made with respect to this target orientation, that is:

$$\theta = {}^{N}\theta \ominus {}^{N}\theta^{*} \tag{4.40}$$

where $\theta \in (-\pi, \pi]$, ${}^{N}\theta$ is the current vehicle orientation with respect to North, and ${}^{N}\theta^{*}$ is the target orientation with respect to North — the target orientation defines the x-axis. Referring to Figure 4.27, the range vectors to the individual landmarks in the workspace are given by:

$$\underline{L_i} \;=\; R_i \angle(\theta \oplus \beta_i) \tag{4.41}$$

To determine vehicle pose with respect to a target pose, the vehicle is first driven to the desired pose and the target orientation and IALV are captured and stored. Subsequently, the IALV method is used to find the range and bearing to the target location, e and δ, where δ is measured with respect to the x axis. This can then be transformed to a Cartesian representation:

$$x_f \;=\; -e\cos\delta \tag{4.42}$$

$$y_f \;=\; -e\sin\delta \tag{4.43}$$

where (x_f, y_f) are the coordinates of the front-axle. However, the vision sensor is mounted over the midpoint of the front-axle — for control purposes, the position of the midpoint of the rear-axle is required. With knowledge of the vehicle's length, L, and its orientation, θ, the position of the midpoint of the rear-axle is given by the rigid-body constraint:

$$x \;=\; x_f - L\cos\theta \tag{4.44}$$

$$y \;=\; y_f - L\sin\theta \tag{4.45}$$

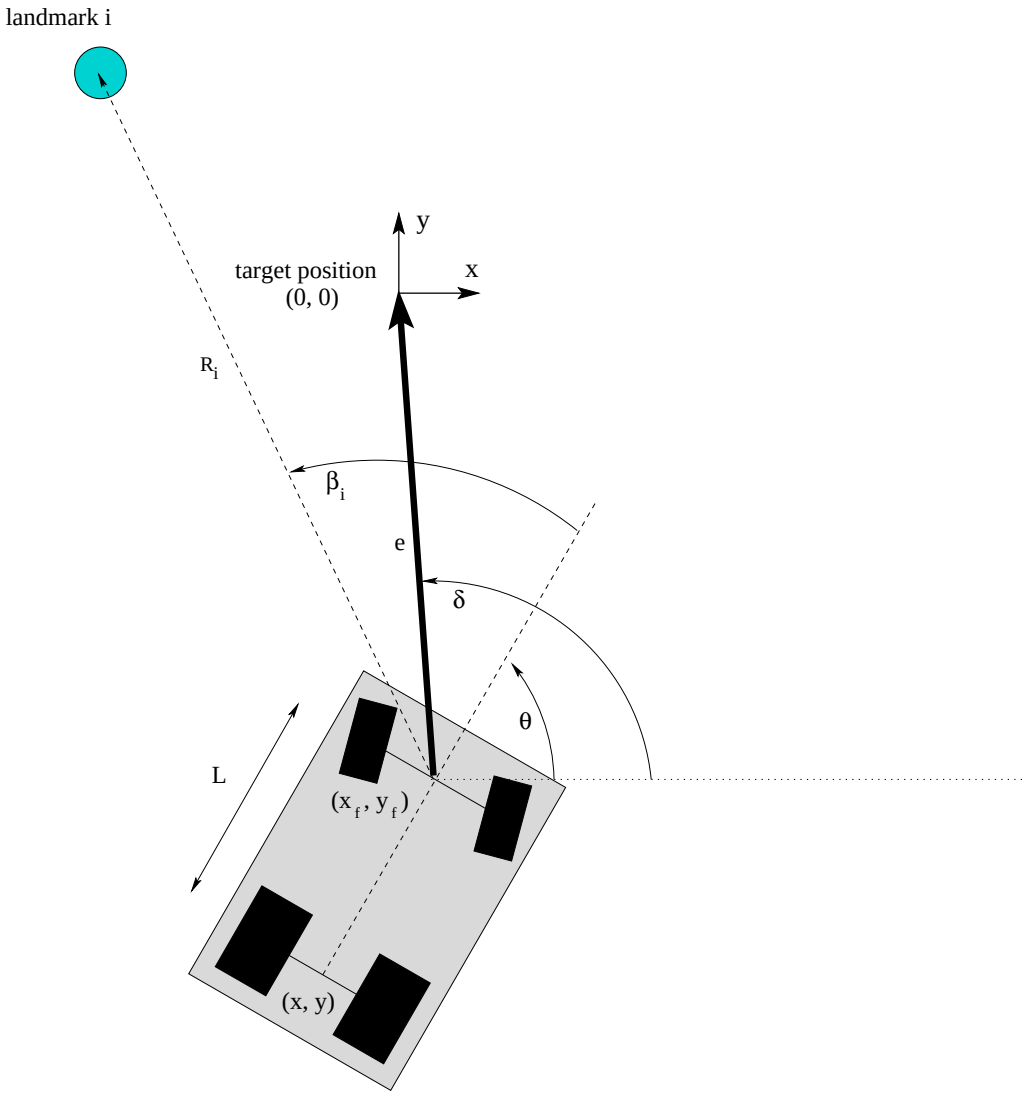


Figure 4.27: IALV method used to determine vehicle pose (x, y, θ) with respect to the target pose.

where (x, y) are the coordinates of the rear-axle. Thus, the full vehicle pose estimate, (x, y, θ), with respect to the target pose, can be determined using the IALV method and data from the magnetometer.

4.3.4 IALV range and bearing error sensitivity

Determining landmark range is relatively straight-forward with an omnidirectional camera however, as shown in Section 4.3.2.3, range estimation with this sensor does not pro-

vide an accurate measurement when compared to other dedicated range sensors such as, for example a SICK scanning laser range-finder. In addition, when using the ground-plane constraint, object apparent range can vary depending on object bearing with respect to the camera due to imperfections in the camera-mirror system and when the camera axis is not perpendicular to the ground-plane (recall Figure 4.23). Landmark bearing estimation is straight-forward and less prone to error than range estimation.

Here, a simple experiment is performed to illustrate range error effects on the IALV strategy. The workspace used is as previously defined in Section 4.2.1 (i.e. $x, y \in [-20, 20]$ with an overlaid grid of spacing 1) with landmarks located at $(x_1, y_1) = (-10, 10)$, $(x_2, y_2) = (10, 10)$ and $(x_3, y_3) = (10, -10)$ as in Figure 4.9 (a). First, it is assumed that the target 'snapshot' is obtained correctly but a scale error occurs on every subsequent landmark range measurement such that $R = 0.8R_{true}$ for each landmark. The effect of this offset can be seen in Figures 4.28 (c) and (d), while the 'perfect case' is shown in Figures 4.28 (a) and (b). As the figure shows, for such an error homing occurs to a point displaced from the true target location.

Range and bearing error effects can be explained very simply. Recall the Cartesian version of the IALV homing vector calculation:

$$\begin{aligned} \underline{H}_{IALV} &= \underline{IALV} - \underline{IALV}^* \\ &= \frac{1}{n}\sum_{i=1}^{n}(\underline{\hat{x}}_i - \underline{x}) - \frac{1}{n}\sum_{i=1}^{n}(\underline{x}_i - \underline{x}^*) \end{aligned} \tag{4.46}$$

where $\underline{\hat{x}}_i$ has been introduced to refer to the current landmark estimated position and all other terms are as defined in the preceding sections. $\underline{\hat{x}}_i$ consists of the actual landmark position, and an error component $\underline{\epsilon}_i$ caused by landmark range and bearing errors, i.e.:

$$\underline{\hat{x}}_i = \underline{x}_i + \underline{\epsilon}_i \tag{4.47}$$

$\underline{\epsilon}_i$ encapsulates all errors associated with determining where a landmark is. This includes random errors such as those associated with vibration of the vehicle, and errors caused by sensor misalignments.

For 'perfect' range and bearing estimation, such that $\underline{\epsilon}_i = 0$ for all landmarks, Equation 4.46 reduces to:

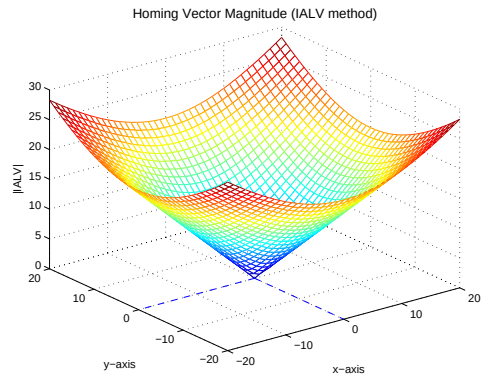
$$\underline{H}_{IALV} = \underline{x}^* - \underline{x} \tag{4.48}$$

However, for imperfect range and bearing estimation, complete term cancellation does not occur leaving:

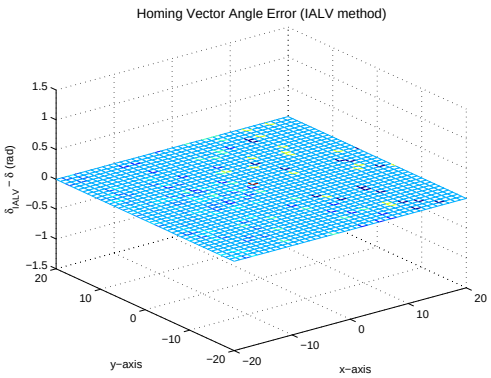
$$\begin{aligned} \underline{H}_{IALV} &= (\underline{x}^* - \underline{x}) + \frac{1}{n}\sum_{i=1}^{n}(\underline{\hat{x}}_i - \underline{x}_i) \\ &= (\underline{x}^* - \underline{x}) + \frac{1}{n}\sum_{i=1}^{n}\underline{\hat{\epsilon}}_i \end{aligned} \tag{4.49}$$

The terms in the summation produce the homing vector offset. The contribution from these terms is dependent of course on the magnitude of the range and bearing errors but also depends on the radial distribution of the landmarks with respect to the target location. Some landmark configurations will diminish the effects of these errors, as for landmarks which are equally radially distributed around the target location, while other configurations will magnify the effects. In practice, systematic sensing errors, such as the dependence on bearing for the range estimate discussed in Section 4.3.2.3, form the bulk of the error term while random errors, such as those caused by vibration, tend to cancel due to averaging effects.

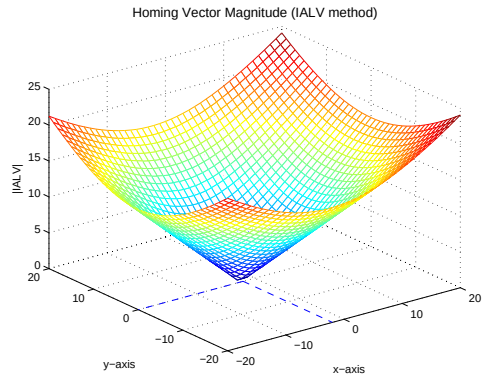
As mentioned earlier, the use of this sensing arrangement and such a 'simple' localisation strategy has some important consequences when servoing to a position as opposed to a pose — the world can 'look' different from the same location when viewed at different orientations when using an omnidirectional camera which is not perpendicularly aligned with the local ground-plane. When servoing to a pose, the goal is approached with the same vehicle orientation as the 'target snapshot' and any vision sensing errors appear at the same location in the snapshot and hence, cancel out. However, when servoing to a position with no constraint on the vehicle final orientation, the goal can be approached with *any* vehicle orientation and the current snapshot of the environment appears different to the target snapshot — the world 'looks' slightly different and the vehicle homes to the incorrect position. This homing error is theoretically equal to $\frac{1}{n}\sum_{i=1}^{n}\underline{\hat{\epsilon}}_i$.



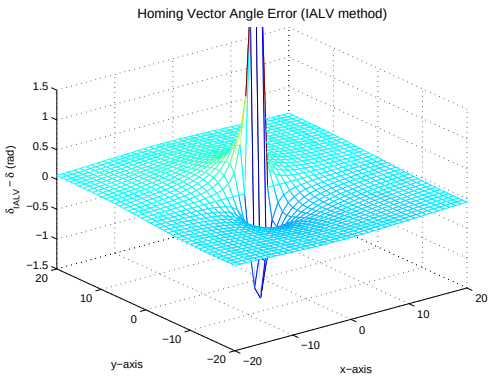
(a) IALV homing vector magnitude, no range error.



(b) IALV homing vector angle error, no range error.



(c) IALV homing vector magnitude, range error on all measurements after target 'snapshot'.



(d) IALV homing vector angle error, range error on all measurements after target 'snapshot'.

Figure 4.28: Effect of range errors on IALV homing vectors for the landmark configuration of Figure 4.9 (a).

Finally, although increasing the number of landmarks may help reduce problems with landmark sensing errors, it can also limit the available workspace. As explained in Section 4.3.2.3, the vision system is limited to sensing landmarks within a range of 1 to $12\,\mathrm{m}$. Because of this limitation, and the simple strategy used, the usable navigation area becomes quite limited as the number of landmarks increases. Figure 4.29 gives an example of a workspace containing three fairly widely spaced landmarks and the associated limitation to the available space for which the IALV strategy can be used with this range limited

sensor. For this reason, most of the experiments on the vehicle were conducted with one

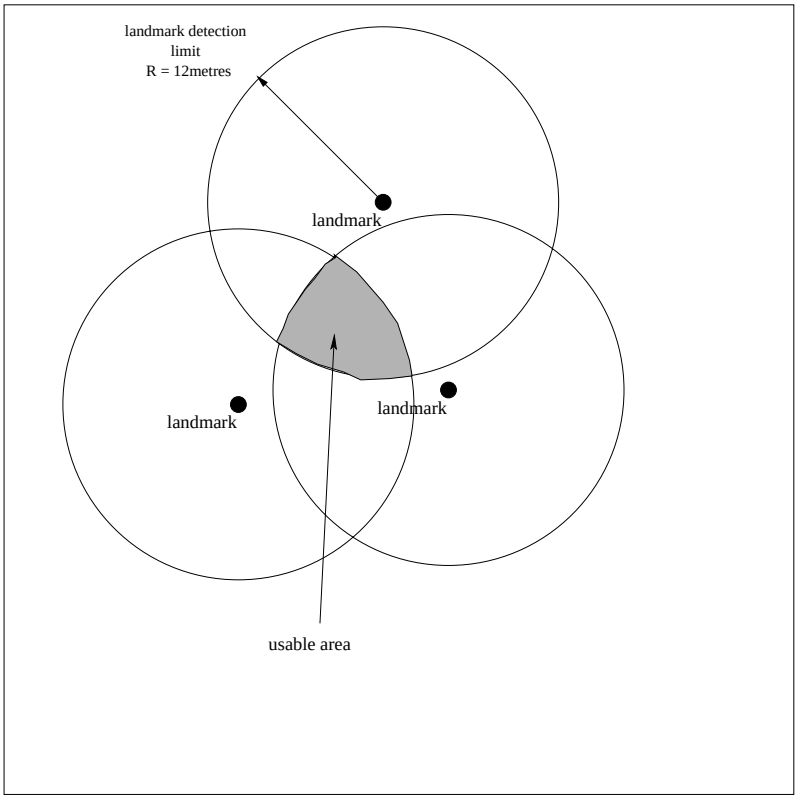


Figure 4.29: Useful navigation space when using the IALV strategy.

landmark in the workspace.

4.4 Conclusion

In this chapter, several landmark-based homing strategies were investigated. These strategies rely on comparing the current view of the environment with some remembered view of a target location to drive the robot towards the target position. In this work, it has been shown that the bearing-only landmark-based homing strategies have a strong dependence on landmark configuration with respect to the target position. By introducing range information to such strategies, this dependence can be eliminated. The Improved Average Landmark method developed in this work includes landmark range information and is also free of the need for correspondence between features in the current and target view.

By storing the vehicle orientation at the the target location, a full vehicle pose estimate with respect to the target pose can also be obtained.

The sensors and sensing strategies used to implement the IALV homing method were then presented. First, a reference direction is provided by a CROSSBOW high-speed orientation sensor. To reduce noise effects, measurements from this sensor were combined with estimates of the angular rate of the vehicle found from vehicle odometry. These measurements are combined using a complementary filter which provides significant signal-to-noise ratio improvements, without introducing any significant delays in the system. Landmarks are sensed using an omnidirectional camera. Landmarks are found in an image based on a pre-learnt look-up table of their colour profile. Landmark range is estimated using a model of the omnidirectional camera's optics, and the landmarks are subsequently tracked using a vehicle-object motion model. These models are also used in complementary filters to improve the estimates of landmark range and bearing.

Importantly, the IALV method is correspondence-free — there is no need for matching of the current and target landmarks. In addition, it is simple, requiring only vector addition and subtraction of landmark range vectors. The method requires a minimum of one landmark as opposed to the minimum of three required by the bearing-only strategies. Also, the method is simple to implement in terms of sensing, using a magnetic compass for a reference direction and an omnidirectional camera to determine landmark range and bearing. Furthermore, unwrapping of the omnidirectional image is not required, providing a further computational saving.

However, the success of this visual pose estimation strategy is somewhat limited by the sensors and the simple sensing strategies used. For example, small range errors caused by non-perpendicular alignment of the omnidirectional camera with the ground-plane can lead to significant errors in position. Thus, the method is suited to pose stabilization, in which errors between the current and target views cancel out.

Chapter 5

Position and Pose Stabilization — Visual Homing for a Car-Like Vehicle

The position stabilization problem for car-like vehicles can be solved by using a two-stage state feedback approach: first stabilize the relative heading to the goal with respect to the vehicle's longitudinal axis; then stabilize the relative heading and *the distance to the goal. Likewise, the pose stabilization problem for such a vehicle can also be solved with a multistage state feedback strategy: First home to the goal region; second, servo to the x-axis with the desired final orientation; finally, servo to the goal. In practice, when using the IALV strategy with the sensing arrangement described in the previous chapter, pose stabilization is more successful than position stabilization because errors in the target and current views cancel out. When stabilizing to a position, this does not always occur.*

5.1 Stabilization of mobile robots

If mobile robots are to perform useful tasks, they will require the ability to stabilize to specified positions and poses in the environment — these competencies are central to a mobile robot's ability to do useful things in a workspace. For example, if a mobile robot is to perform a drilling task, it must position and orient itself in a precise manner with respect to the hole to be drilled. No matter how detailed the map it is provided, it still must possess the ability to accurately manoeuvre itself. Although computing power is ever increasing, solutions based upon path-planning and optimal control are still beyond

the requirements of operating in real-time, particularly if high-level sensing is also to occur on-board. This is important in the case when the robot is required to replan on-line because of accumulated errors associated with robot motion, errors in the model of the robot, and mapping errors.

In many industries, robotic systems will be retrofitted to existing vehicles, and hence, the control must be able to cope with the dynamics and constraints of the vehicle. The example used in this thesis is that of a car-like vehicle which is limited to moving in a series of clothoids and lines. The vehicle has significant dynamics in the steering and velocity loops (as detailed in Section 3.4), leading to significant delays between a demand being issued and the vehicle's response. Similar challenges are faced in automating industrial vehicles.

This chapter is divided into two sections; the first section concentrates on the problem of position stabilization in which the aim is to stabilize[1] the vehicle to a particular position in a workspace without a constraint on the vehicle's final orientation; the second section adds the constraint of stabilizing the vehicle to a particular position *with a particular final orientation*, i.e. pose stabilization.

5.2 Position stabilization

First, the problem of *position* stabilization is considered in which the goal is to stabilize a car-like vehicle to a target position with no constraint on its final orientation. Here, the IALV method and sensing techniques of the previous chapter are used to estimate the vehicle pose with respect to a target pose, based on the discrepancies between the current and target views of the workspace.

Like pose stabilization, position stabilization can be attacked on several fronts. The open-loop strategies seek to find the bounded sequence of control demands which will bring the vehicle to the target position from some arbitrary position in the workspace. The closed-loop strategies require the design of a feedback law which provides the control

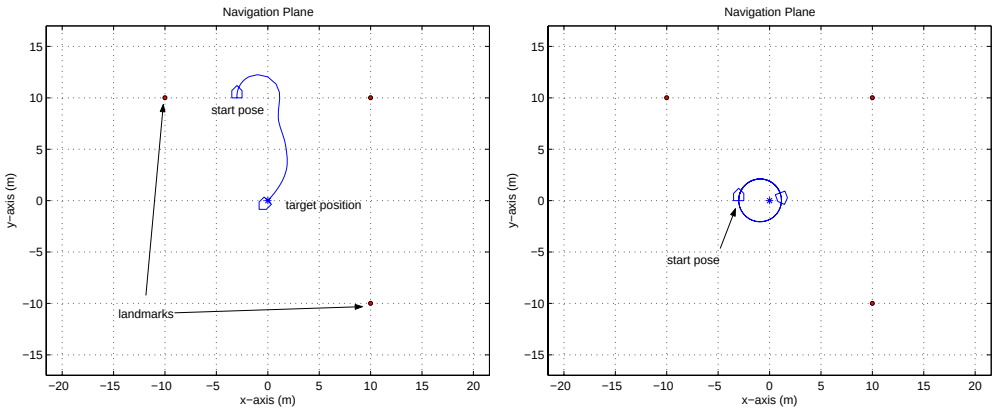
[1]The term 'stabilize' is used to describe driving a state, or set of states, towards zero.

demands based upon the current robot state. In this work, the closed-loop strategy is used as it is more robust to model and sensor uncertainty.

The visual homing strategies discussed in Section 4.2.1 fall into the closed-loop control category. These methods operate by providing a *homing vector*, using the vector's magnitude and direction to set the vehicle speed and direction of travel. This strategy works well for vehicles which can rotate about their own vertical axis (i.e rotate on the spot), provided that the vehicle lies within some 'catchment' area of the target location. However, a car-like vehicle cannot rotate about its own vertical axis and the visual homing strategies are not as effective, particularly if the vehicle's starting location is relatively close to the goal position — this is because a car-like vehicle needs significant 'space' to align itself with the homing vector and it is more likely to get 'stuck' in local control minima where the vehicle rocks back and forth between competing homing vectors, or cannot resolve errors in the final approach stages. Figure 5.1 (a) shows a successful homing run in which the ALV strategy was used to set the vehicle speed and heading angle. Figure 5.1 (b) gives an example of a failure case, in which the vehicle was unable to resolve the final error. This example demonstrates the limitations of linear controllers for the control of nonholonomic vehicles. Of course, in this example, the vehicle could reverse and align itself with the goal but with the inconsistent information provided by the ALV strategy, it is difficult to make these decisions 'autonomously'.

Besides the visual homing type literature, this problem has received little attention in the literature. Similar competencies have been studied in terms of providing behaviours for vehicles and characters in the computer gaming industry (see for example [Reynolds, 1999]). In the control community, position stabilization is little studied. An exception is the early work of Samson and Ait-Abderrahim [1991], who simulated a unicyle-like vehicle, illustrating the controllability of the system and demonstrating stabilization to a position. However, they do not consider the dynamic effects of steering and velocity loop responses.

Here a non-linear feedback law is designed which allows a vehicle to stabilize to a position without getting 'stuck' at local control minimums while also naturally accom-



(a) ALV homing, starting pose $(x, y, \theta) = (-3, 10, \frac{\pi}{2}\,\mathrm{rad})$.

(b) ALV homing, starting pose $(x, y, \theta) = (-3, 1, \frac{\pi}{2}\,\mathrm{rad})$.

Figure 5.1: Example of position control with ALV strategy. Vehicle starts at the specified poses, and the ALV strategy is used to set vehicle speed and heading to drive it towards the origin.

modating the dynamics of the system, such as the response of the steering and velocity loops, and non-linearities such as input saturation. Such a competency could be used, for example, to ensure that a mobile robot passes precisely through a navigation 'way-point'.

In developing the control law, extensive testing was conducted using the simulation model of the vehicle (refer to Section 3.4 for details of the vehicle model) and it is assumed that pose information is available using the techniques described in Section 4.2.1.4.

5.2.1 Control law design

Repeating the kinematic equations for a car-like vehicle in Cartesian space:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ v\frac{tan\phi}{L} \end{bmatrix} \tag{5.1}$$

where the demands for the system are v and ϕ, and x, y and θ are defined with reference to the target position and the x-axis as shown in Figure 5.2. The relation

$$\omega = v\frac{tan\phi}{L} \tag{5.2}$$

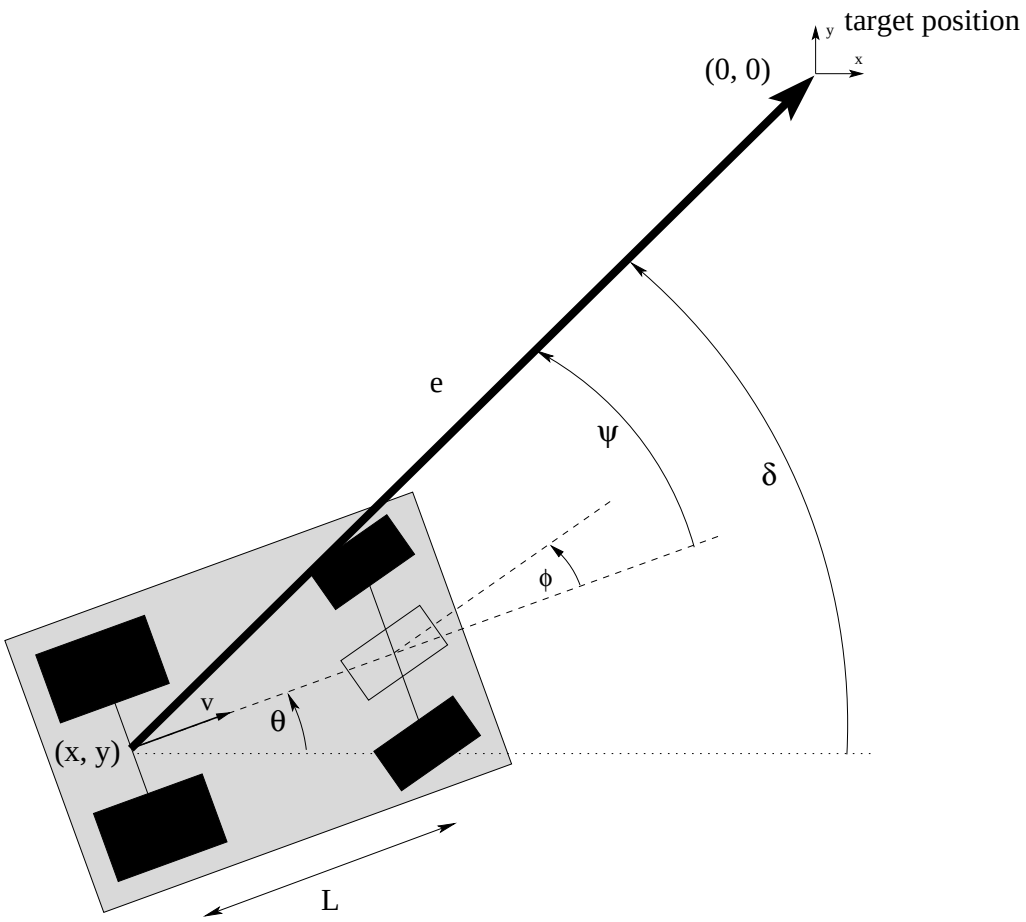


Figure 5.2: Coordinate system for control.

will be substituted during the analysis for simplification, it is an easy matter to transform between the two representations.

Transforming the Cartesian representation into a vehicle relative target representation, see Figure 5.2, leads to the following vehicle state equations:

$$\begin{bmatrix} \dot{e} \\ \dot{\psi} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} -v\cos\psi \\ \frac{v\sin\psi}{e} - \omega \\ \frac{v\sin\psi}{e} \end{bmatrix} \tag{5.3}$$

where e is the distance to the target position, $\psi \in (-\pi, \pi]$ is the bearing to the target position relative to the vehicle's longitudinal axis and $\delta \in (-\pi, \pi]$ is the bearing to the target position relative to the x-axis. Note the similarity to the vehicle-object motion

equations presented in Section 4.3.2.4. These equations are quite a 'natural' form for a vehicle moving through a workspace and are commonly found in the literature on the pose control of unicycle-like vehicles (see e.g. [Badreddin and Mansour, 1993; Aicardi *et al.*, 1995; 1994; Pourboghrat, 2002]).

The control strategy used here is to break the state-space into sub-manifolds, each of which is then sequentially stabilized [Bloch and McClamroch, 1989; Bloch *et al.*, 1990]. To stabilize a car-like vehicle to a point, the first stage of control stabilizes the bearing of the target position relative to the vehicle's longitudinal axis ψ. The second stage of control then stabilize the heading ψ *and* the distance to the target position e. That is,

- First stage (bearing stabilization): $\psi \rightarrow 0$
- Second stage (homing): $\psi \rightarrow 0$ and $e \rightarrow 0$

To achieve this, the individual controllers of each stage are designed using suitably chosen Lyapunov functions, based upon the well-known Lyapunov stability theory (see for example Slotine and Li [1991]).

5.2.1.1 First stage: bearing stabilization

For the first stage of control, a suitable Lyapunov function is:

$$V = \frac{1}{2}\psi^2 \tag{5.4}$$

which is a positive semi-definite function. The time derivative of this function is:

$$\begin{aligned} \dot{V} &= \psi\dot{\psi} \\ &= \psi\left(\frac{v\sin\psi}{e} - \omega\right) \end{aligned} \tag{5.5}$$

where the state equation for $\dot{\psi}$ from Equation 5.3 has been substituted. For the system to converge to $\psi \rightarrow 0$, $\dot{V} \leq 0$. This can be achieved through the judicious choice of v and ω, where for the time being the dynamics are ignored (i.e. $v^* = v$ and $\omega^* = \omega$).

This stage of control aims to stabilize the vehicle to a line defined by $\psi = 0$ and hence v can be chosen in an essentially open-loop manner depending on the vehicle's initial orientation with respect to the goal.

$$v^* = \begin{cases} k_1 & \text{if } \cos\psi_{initial} \geq 0 \\ -k_1 & \text{otherwise} \end{cases} \tag{5.6}$$

where $k_1 > 0$. The choice for the vehicle velocity direction, that is the sign of v, has some important consequences for stability, particularly when the angular velocity is saturated — this is discussed further in the section on input saturation.

Now to ensure that $\dot{V} \leq 0$, ω^* is chosen as:

$$\omega^* = k_2\psi + \frac{v^* \sin\psi}{e} \tag{5.7}$$

where $k_2 > 0$. The actual steering demand to the vehicle is the steering angle which is given by rearranging Equation 5.2:

$$\phi^* = \arctan\left(\frac{\omega^* L}{v^*}\right) \tag{5.8}$$

where ϕ^* refers to the demanded steering angle. Substituting the choice for ω^* back into Equation 5.5 gives:

$$\begin{aligned} \dot{V} &= \psi\left(\frac{v\sin\psi}{e} - \omega\right) \\ &= \psi\left(\frac{v\sin\psi}{e} - \left\{k_2\psi + \frac{v^*\sin\psi}{e}\right\}\right) \\ &= -k_2\psi^2 \end{aligned} \tag{5.9}$$

which is negative semi-definite (i.e. $\dot{V}(0) = 0$ and $\dot{V}(\psi) \leq 0$) and uniformly continuous; by the Lyapunov stability theory and using Barbalat's lemma (required because $\dot{V}$ is negative semi-definite), the Lyapunov function (Equation 5.4) is therefore stable and $\psi \to 0$.

Analysis so far has assumed perfect response of the v and ω control loops (i.e. $v = v^*$ and $\omega = \omega^*$). The next section studies the effects of dynamics in these loops together with an analysis of saturated inputs.

Dynamic effects and gain tuning

Recall from Section 3.4 that the response of the vehicle's velocity loop is first-order and given by:

$$\frac{v(s)}{v^*(s)} = \frac{K_v}{\tau_v s + 1} \tag{5.10}$$

where v^* indicates the demand, v is the response, and $K_v = 1$ and $\tau_v = 1.33$ as determined in Section 3.4. In state space form, Equation 5.10 becomes:

$$\dot{v} = \frac{1}{\tau_v}(v^* - v) \tag{5.11}$$

Again, recall from Section 3.4 that the response of the vehicle's steering loop is second-order and given by:

$$\frac{\phi(s)}{\phi^*(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{5.12}$$

where ϕ^* indicates the demanded steering angle, ϕ is the response, and ω_n and ζ are the natural frequency and damping ratio of the steering loop (refer to Section 3.4 where it was found for this system that $\omega_n = 0.8$ and $\zeta = 0.8$). In addition, the steering angle is limited to $\phi = \pm 30^\circ$. Over this angular range, linearization gives:

$$\tan\phi \simeq \phi \tag{5.13}$$

with approximately 10% error at the extremities. Thus, the vehicle angular velocity from Equation 5.2 can be approximated by:

$$\omega = \frac{v}{L}\phi \tag{5.14}$$

and Equation 5.8 becomes:

$$\phi^* = \frac{\omega^* L}{v^*} \tag{5.15}$$

The equation relating demanded angular velocity ω^* and actual angular velocity ω is (combining Equations 5.12, 5.14 and 5.15):

$$\frac{\omega(s)}{\omega^*(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{5.16}$$

which can be reduced to two state equations, through the introduction of an intermediary state:

$$\dot{\rho} = \omega_n^2 (\omega^* - \omega) \tag{5.17}$$

$$\dot{\omega} = \rho - 2\zeta\omega_n\omega \tag{5.18}$$

Linearising the system equations 5.3, 5.11, 5.17 and 5.18 about $\psi = 0$, and assuming that the vehicle is relatively far from the origin (for which $e \gg v^*$) leads to the following system state equations:

$$\begin{aligned} \dot{e} &= -v \\ \dot{\psi} &= -\omega \\ \dot{\delta} &= 0 \\ \dot{\rho} &= \omega_n^2 (\omega^* - \omega) \\ \dot{\omega} &= \rho - 2\zeta\omega_n\omega \\ \dot{v} &= \tfrac{1}{\tau_v} (v^* - v) \end{aligned} \tag{5.19}$$

Linearising the feedback laws about the same point gives:

$$v^* = k_1 \tag{5.20}$$

$$\omega^* = k_2\psi \tag{5.21}$$

Inserting the linearised feedback laws in to Equation 5.19 leads to the closed loop state equation:

$$\begin{bmatrix} \dot{e} \\ \dot{\psi} \\ \dot{\omega} \\ \dot{\rho} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -2\zeta\omega_n & 1 & 0 \\ 0 & \omega_n k_2 & -\omega_n^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{\tau_v} \end{bmatrix} \begin{bmatrix} e \\ \psi \\ \omega \\ \rho \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \frac{k_1}{\tau_v} \end{bmatrix} \tag{5.22}$$

The equations for $\dot{e}$ and $\dot{v}$ are independent of the equations for $\dot{\psi}$, $\dot{\omega}$ and $\dot{\rho}$, and will not be considered further (in any case, the sub-loop for $\dot{e}$ and $\dot{v}$ is not controlled in this stage of control).

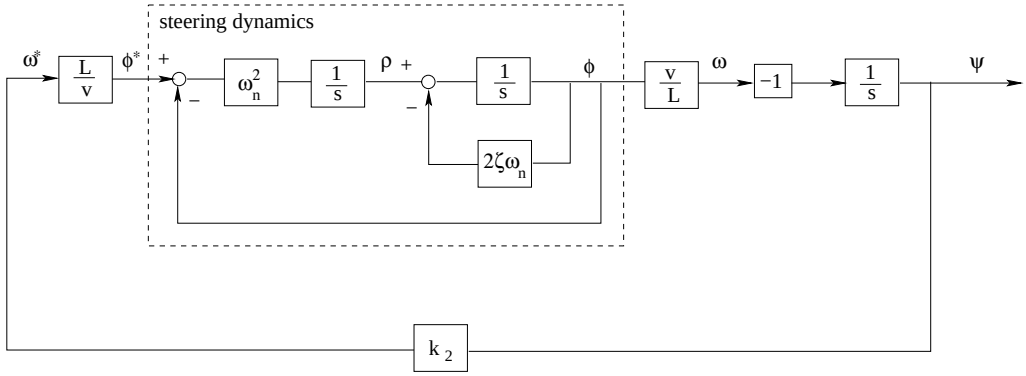


Figure 5.3: Block diagram for ψ loop in first stage position control.

In closed-loop form, the linearised system is then represented by:

$$\begin{bmatrix} \dot{\psi} \\ \dot{\omega} \\ \dot{\rho} \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & -2\zeta\omega_n & 1 \\ k_2\omega_n^2 & -\omega_n^2 & 0 \end{bmatrix} \begin{bmatrix} \psi \\ \omega \\ \rho \end{bmatrix} \tag{5.23}$$

The block diagram for the system is given in Figure 5.3. and the *characteristic equation* of the system is:

$$s^3 + 2\zeta\omega_n s^2 + \omega_n^2 s + k_2\omega_n^2 = 0 \tag{5.24}$$

The stability of the system with reference to the k_2 gain is analysed by rearranging the characteristic equation (to the form $1 + k_2 GH$):

$$1 + k_2 \frac{\omega_n^2}{s^3 + 2\zeta\omega_n s^2 + \omega_n^2 s} = 0 \tag{5.25}$$

for which the root locus plot is shown in Figure 5.4. The root locus plot indicates that the system is stable for $0 < k_2 < 1.28$.

To select k_2, the problem is treated as a parameter optimization problem, in the spirit of Linear Quadratic Regulator (LQR) control and other optimised control design techniques. Here, an unconstrained non-linear optimizer is used, the *Nelder-Mead simplex search* method for multidimensional parameter optimisation (of course, here there is only a single parameter and a multidimensional search is not required but this technique is still applicable). This is a function commonly available in mathematical packages such

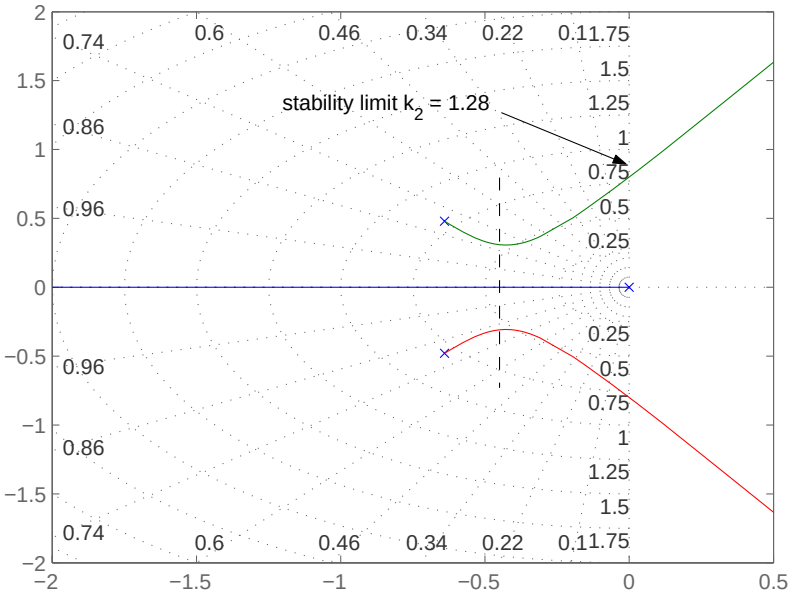


Figure 5.4: Root locus for first stage of position control, where k_2 is varied.

as MATLAB. It operates by minimising some cost function, J, given a starting vector for the desired parameters, in this case the single parameter k_2.

For this problem, the response of the linearised system is simulated using the closed-loop state equations (Equation 5.23) given some arbitrary initial state. The cost function is based upon the speed of convergence of the ψ state and the amount of overshoot and settling time, where k_2 is adjusted to obtain the desired system behaviour. The cost function must be carefully chosen to ensure that the desired behaviour is obtained — the design of cost functions requires a good understanding of the problem and can be an iterative process in order to obtain the desired system convergence. Here, it was found that the following cost function produces the desired results:

$$J(\psi) = \int_{t=0}^{t=T} \left(\frac{\psi(t)}{\psi(0)} \right)^2 dt + \xi_1(\underline{\psi}) + \xi_2(\underline{\psi}) \tag{5.26}$$

where $\psi(t)$ is the state value at time t, $\underline{\psi}$ is the vector of the evolution of the ψ state throughout the simulation, $\xi_1(\underline{\psi})$ is a function which calculates the maximum overshoot in ψ and $\xi_2(\underline{\psi})$ is a function which calculates the settling time — these latter two quantities are defined in Figure 5.5. Overshoot is defined for the general state x as the maximum

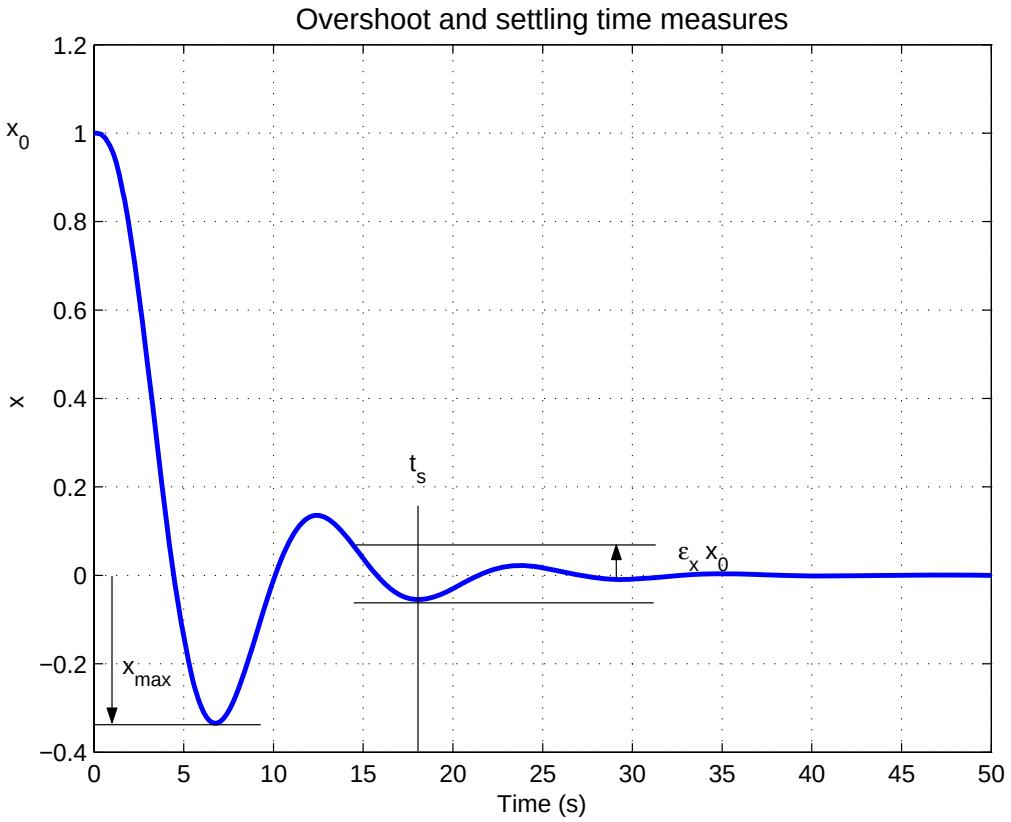


Figure 5.5: Illustration of the overshoot and settling time where x is an arbitrary state variable, in this case ψ.

excursion beyond zero from the starting point x_0, referenced to the initial value of the state i.e.:

$$\xi_1(\underline{x}) \quad = \quad \begin{cases} 100 \left| \frac{x_{max}}{x_0} \right| & \text{if a zero crossing occurs} \\ 0 & \text{otherwise} \end{cases} \tag{5.27}$$

where the quantities are as defined in Figure 5.5. Settling time for the general state x is defined as the time taken to settle to within a defined percentage, ϵ_x, of the initial state value, x_0, that is:

$$\xi_1(\underline{x}) \quad = \quad t_s \tag{5.28}$$

where t_s is the time taken for the transients in the evolution of the state x to settle to within predefined bounds of the final state, relative to the initial state, x_0. These bounds are given by $\epsilon_x x_0$, where for the case of the stabilization of ψ, $\epsilon_\psi = 0.02$.

ψ_0	(k_2)
2	0.2139
1	0.2139
0.5	0.2139
0.1	0.2139

Table 5.1: Optimisation results for k_2 in Stage 1 position control determined using `fminsearch` in MATLAB, for different initial target angle (ψ) errors.

Having defined the cost function, the `fminsearch` function in MATLAB is used to simulate the behaviour of the system, minimising the cost function by numerically searching the parameter space of k_2 — the process terminates on minimisation of the cost function. Table 5.1 summarises the results of the optimisation procedure to a range of initial values for the variable ψ. The cost function produces similar values for k_2 over a wide range of initial state values and initial values for the parameter search.

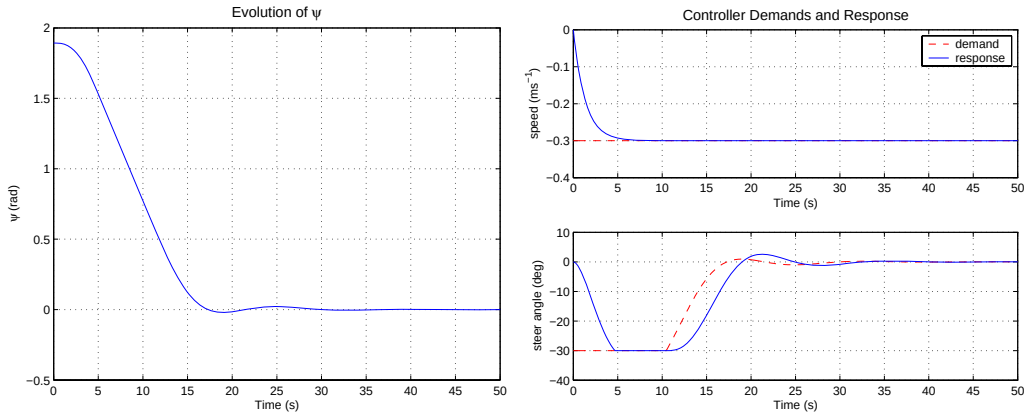
In terms of system stability, the gain selection of $k_2 = 0.2139$ leads to the following roots for the characteristic equation of the system (Equation 5.24):

$$-0.5870 \quad \text{and} \quad -0.3465 \pm 0.3364i$$

indicating that the poles for the closed-loop system are well-placed for this gain selection.

For the velocity law of Equation 5.6, we select $k_1 = 0.3$. Selecting the gain for the velocity control was an iterative process, requiring a balance between minimised time to stabilize, and minimised 'space' to stabilize, given that a higher velocity will require more area for the vehicle to turn within.

The simulated response of the vehicle to the demands supplied by this stage of control using the full non-linear vehicle model developed in Section 3.4 is now considered. The evolution of ψ and the associated demands and response of the velocity and steering loops are illustrated in Figure 5.6 for the case where the initial pose of the vehicle is $(x, y, \theta) = (-30, -10, -\frac{\pi}{2}\,\text{rad})$. Note from these plots, that the ψ loop is stable and converges to zero even for initial values of ψ well outside the linear assumptions used in the analysis. It is also noted that although there is overshoot in the steering angle demand



(a) Evolution of ψ.

(b) Control demands and vehicle response.

Figure 5.6: Response in Stage 1 of the position control loop. Initial vehicle pose is $(x, y, \theta) = (-30, -10, -\frac{\pi}{2}\,\text{rad})$.

and response, the overshoot in the ψ state variable is minimal.

The observed effects, in simulation, of increasing the k_2 gain is increased oscillation in the evolution of ψ, and a consequently longer settling time eventually leading to instability.

Input saturation

The limits of the steering angle ($|\phi_{max}| = 30°$), and velocity ($|v_{max}| \simeq 2\,\text{ms}^{-1}$) can lead to a saturation of ω, the vehicle's angular velocity. In this case, the magnitude of the velocity demand is limited to k_1 by the control law of Equation 5.6, and hence:

$$|\omega_{max}| = \frac{k_1 \tan \phi_{max}}{L} \tag{5.29}$$

$$= 0.1443 \tag{5.30}$$

Saturation of ω can lead to stability problems. These problems can be observed and predicted by analysing Equation 5.5 for the cases were ω saturates, i.e.:

$$\dot{V} = \psi\dot{\psi}$$

$$= \psi\left(\frac{v^* \sin\psi}{e} - \text{sat}\left(\omega^*, \omega_{max}\right)\right) \tag{5.31}$$

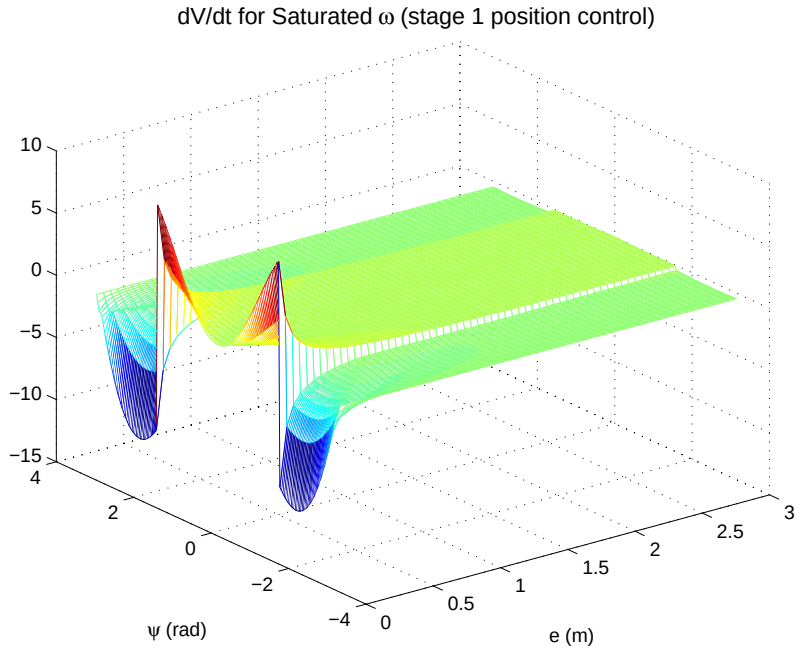


Figure 5.7: Behaviour of $\dot{V}$ for saturated angular velocity ω.

A surface plot of the variation of $\dot{V}$ for the space defined in $e \in [0, 4]$ and $\psi \in [-\pi, \pi]$ is shown in Figure 5.7. In this plot, v^* and ω^* have been calculated for each point using Equations 5.6 and 5.7, and $\dot{V}$ is calculated using Equation 5.31. From the plot, it can be seen that for some combinations of e and ψ, $\dot{V}$ is positive meaning the system is unstable and $\psi \not\to 0$.

Because the vehicle is stabilizing about $\psi = 0$, and v^* is set in an essentially open-loop manner, this saturation can be overcome through the correct selection of the velocity direction (i.e. the sign of v^*). One way to do this is to always choose the vehicle velocity as negative — this has the effect of minimising the angular velocity demand because the two terms in Equation 5.7 are then given an opposite sign. However, this is not practical for all cases. A better technique is to pre-calculate the demands, v^* and ω^* (using Equations 5.6 and 5.7), and then check the value of $\dot{V}$ using Equation 5.31. If $\dot{V} > 0$, then the sign of the velocity demand should be changed to negative:

$$v^* = -k_1 \qquad \text{if } \dot{V} > 0 \tag{5.32}$$

The angular velocity demand, ω^*, is then recalculated using Equation 5.7.

Figures 5.8 (a) and (b) illustrate results for the case with *no saturation logic* in which the sign of v is chosen incorrectly leading to the vehicle simply driving in a circle around the target position for certain starting positions. Figures 5.8 (c) and (d) illustrate results from the same starting position with *saturation logic* which leads to the correct selection of the velocity direction. The starting pose for both examples is $(x, y, \theta) = (-1, 0, \frac{\pi}{4}\,\mathrm{rad})$ which ensures immediate saturation of the angular velocity ω. Only one case is presented here, the Monte Carlo simulations presented in Section 5.2.2.1, with the full switching control law, will further demonstrate the validity of these design decisions.

5.2.1.2 Second stage: homing

The second stage of control aims to stabilize $e \to 0$ and $\psi \to 0$[2] given that ψ has already been stabilized by the first stage of control. This stage of control corresponds to homing where an estimate of the distance and direction to the target location are used to drive the vehicle towards this location. Again, using a Lyapunov-based design approach, a suitable Lyapunov function is:

$$V = \frac{1}{2}e^2 + \frac{1}{2}\psi^2 \tag{5.33}$$

which is a positive semi-definite function. The time derivative of this function is:

$$\begin{aligned} \dot{V} &= e\dot{e} + \psi\dot{\psi} \\ &= -ev\cos\psi + \psi\left(\frac{v\sin\psi}{e} - \omega\right) \end{aligned} \tag{5.34}$$

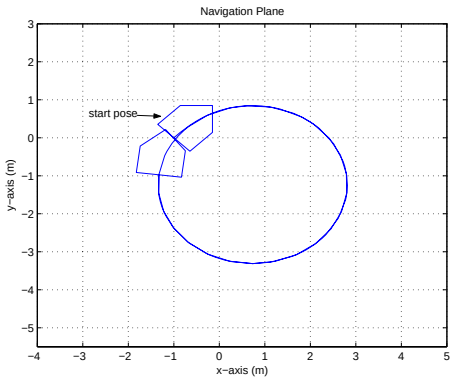
where the state equations for $\dot{e}$ and $\dot{\psi}$ from Equation 5.3 have been substituted.

Now to minimise the first term in Equation 5.34, v^* is selected as:

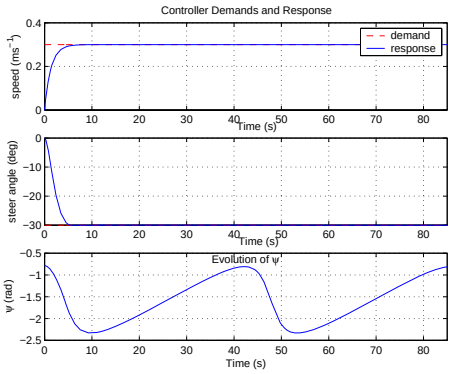
$$v^* = k_1 e\cos\psi \tag{5.35}$$

where $k_1 > 0$. In this stage of control, it is assumed that ψ has already been stabilized to zero by the first stage of control and thus, the $\cos\psi$ term in Equation 5.35 could be

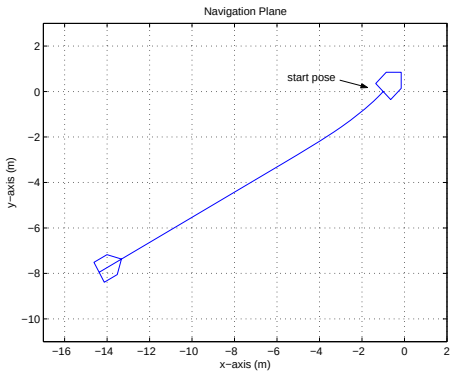
[2] ψ is also controlled here to counteract disturbances.



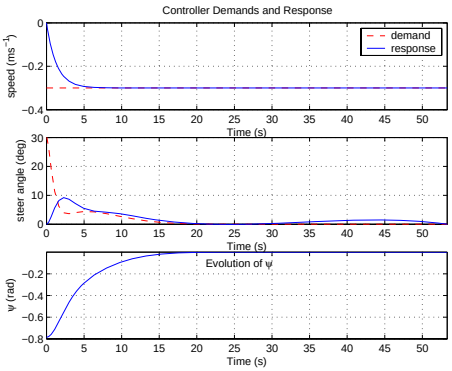
(a) Ground-plane motion of vehicle, no saturation logic.



(b) Control demands and vehicle response, no saturation logic. Also shown in the bottom plot is the evolution of ψ.



(c) Ground-plane motion of vehicle, with saturation logic



(d) Control demands and vehicle response with saturation logic. Also shown in the bottom plot is the evolution of ψ.

Figure 5.8: Response in Stage 1 of the position control loop where the angular velocity (steering angle) is saturated and there is insufficient space to resolve the error (a) and (b). Figures (c) and (d) show the results of selecting the correct initial velocity direction, based on pre-calculation of the vehicle demands. Initial vehicle pose is $(x, y, \theta) = (-1, 0, \frac{\pi}{4}\,\mathrm{rad})$ for both cases.

linearised at this stage. However, it is retained in order for the controller to be able to cope with excursions outside the assumed linear region of operation.

ω is again chosen as in Equation 5.7:

$$\omega^* = k_2\psi + \frac{v^* \sin\psi}{e} \tag{5.36}$$

where $k_2 > 0$ which ensures that the second term in Equation 5.34 is negative. Again, the true demand to the system is ϕ, rather than ω — we convert between the two as required.

Substituting these choices for v^* and ω^* back into Equation 5.34 leads to:

$$\begin{aligned} \dot{V} &= -e\cos\psi\,(k_1 e\cos\psi) + \psi\left(\frac{v\sin\psi}{e} - \left\{k_2\psi + \frac{v^*\sin\psi}{e}\right\}\right) \\ &= -k_1 e^2\cos\psi^2 - k_2\psi^2 \end{aligned} \tag{5.37}$$

which is again negative semi-definite and uniformly continuous meaning that with this choice of demands, $V \to 0$ and hence $e \to 0$ and $\psi \to 0$ again by using the Lyapunov stability theory and Barbalat's lemma. Again, these demand selection assume that $v^* = v$ and $\omega^* = \omega$. Next, the dynamic effects in the steering and velocity loops are considered, followed by an analysis of the effects of input saturation.

Dynamic effects and gain selections

From the first stage of control (Section 5.2.1.1), the following state equations for the steering and velocity loops were derived:

$$\begin{aligned} \dot{v} &= \frac{1}{\tau_v}(v^* - v) \\ \dot{\rho} &= \omega_n^2(\omega^* - \omega) \\ \dot{\omega} &= \rho - 2\zeta\omega_n\omega \end{aligned} \tag{5.38}$$

Again, linearising Equation 5.3 about $\psi = 0$:

$$\begin{bmatrix} \dot{e} \\ \dot{\psi} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} -v \\ \frac{v\psi}{e} - \omega \\ \frac{v\psi}{e} \end{bmatrix} \tag{5.39}$$

The feedback laws are linearised about the same point; Equation 5.35 becomes:

$$v = v^* = k_1 e \tag{5.40}$$

and Equation 5.36 becomes:

$$\begin{aligned} \omega^* &= k_2\psi + \frac{v^*}{e}\psi \\ \omega^* &= k_2\psi + k_1\psi \end{aligned} \tag{5.41}$$

Combining Equations 5.38 and 5.39, and inserting the linearised feedback laws leads to:

$$\begin{aligned} \dot{e} &= -v \\ \dot{\psi} &= \tfrac{v\psi}{e} - \omega \\ \dot{\delta} &= \tfrac{v\psi}{e} \\ \dot{\rho} &= \omega_n^2\left((k_2\psi + k_1\psi) - \omega\right) \\ \dot{\omega} &= \rho - 2\zeta\omega_n\omega \\ \dot{v} &= \tfrac{1}{\tau_v}\left(k_1 e - v\right) \end{aligned} \tag{5.42}$$

There is considerable multiplicative coupling between the states v, ψ and e in the above equations. In order to eliminate the effects of this coupling, in terms of the analysis and the control, a slight modification to the velocity control law of Equation 5.35 is made such that $v^* \simeq v$. This is achieved by limiting the rate of change of v as follows:

$$v_{rl}^* = v + \text{sat}\left(\frac{v^* - v}{\tau_2}, \frac{\dot{v}_{max}}{\tau_2}\right) \tag{5.43}$$

where v^* is as given in Equation 5.35, which when linearised about $\psi = 0$ gives:

$$v_{rl}^* = v + \text{sat}\left(\frac{k_1 e - v}{\tau_2}, \frac{\dot{v}_{max}}{\tau_2}\right) \tag{5.44}$$

In this work, $\tau_2 = \tau_v$ and $\dot{v}_{max} = \tau_v^2$. Because the velocity loop dynamics are significantly faster than the steering loop dynamics, this change in the velocity control law allows us to now make the assumption that $v_{rl}^* \simeq v$. This simplifies the analysis and also further stabilizes the control. With the above assumption, the velocity loop dynamics can be ignored for which the closed-loop, linearised system equation becomes:

$$\left[\begin{array}{c} \dot{e} \\ \hline \dot{\psi} \\ \dot{\omega} \\ \dot{\rho} \end{array}\right] = \left[\begin{array}{c|ccc} -k_1 & 0 & 0 & 0 \\ \hline 0 & k_1 & -1 & 0 \\ 0 & 0 & -2\zeta\omega_n & 1 \\ 0 & \omega_n^2\left(k_1 + k_2\right) & -\omega_n^2 & 0 \end{array}\right] \left[\begin{array}{c} e \\ \hline \psi \\ \omega \\ \rho \end{array}\right] \tag{5.45}$$

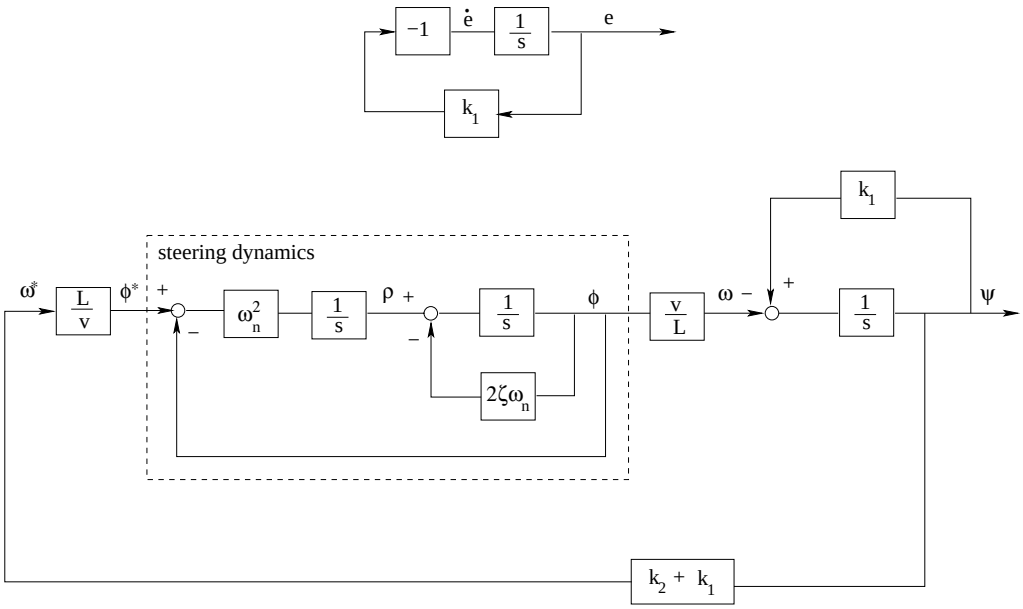


Figure 5.9: Block diagram for the ψ and e loops in the second stage of position control.

where the lines separating the e loop from the lower three equations indicate that, through linearization and the assumption that $v = v^*$, it has been made independent of the other equations — it is retained in the state matrix equations for convenience. The block diagram for the system is shown in Figure 5.9. In this system, there are two parameters which can be independently adjusted to alter the response of the system, k_1 and k_2.

First, the stability of the system with respect to these two gains is assessed, noting that the state equation for e is independent of all other states (this is due to the term cancellation in the linearised state equations and feedback laws). The characteristic equation for the lower part of the system, that is the lower three equations of Equation 5.45, is:

$$s^3 + (2\zeta\omega_n - k_1)\, s^2 + \left(\omega_n^2 - 2k_1\zeta\omega_n\right) s + \left(k_2\omega_n^2\right) \;=\; 0 \tag{5.46}$$

Rearranging for the k_2 gain allows us to use MATLAB's `rlocus` tool:

$$1 + k_2 \frac{\omega_n^2}{s^3 + (2\zeta\omega_n - k_1)\, s^2 + (\omega_n^2 - 2\zeta\omega_n k_1)\, s} \;=\; 0 \tag{5.47}$$

Figure 5.10 illustrates the root locus for the case where k_2 is varied and k_1 is fixed at 0.1, 0.2 and 0.5. Increasing the k_1 gain pushes the two poles towards the origin, eventually resulting in an inherently unstable system. The characteristic equation for the k_1 gain is:

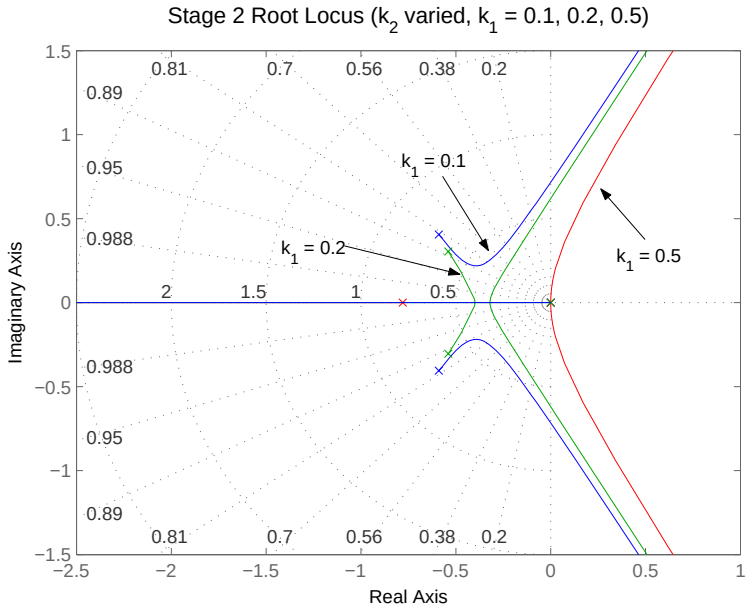


Figure 5.10: Root locus for second stage of position control, where k_2 is varied and $k_1 = 0.1,\ 0.2,\ 0.5$.

$$1 + k_1 \frac{(-s^2 - 2\zeta\omega_n s)}{s^3 + 2\zeta\omega_n s^2 + \omega_n^2 s + k_2\omega_n^2} \quad = \quad 0 \tag{5.48}$$

and the root locus for the case where k_1 is varied and k_2 is fixed at 0.1, 0.5 and 1.4 is shown in Figure 5.11. Increasing the k_2 gain beyond 1.4 leads to an inherently unstable system.

Given an indication of the stability behaviour of the system, tuning of the system gains can be addressed. As for the first stage of control, the technique used here is to formulate the problem as a parameter optimisation problem. However, for this stage of control, the parameter search occurs in two dimensions, k_1 and k_2.

The behaviour of the system is simulated using the closed-loop state equation (Equation 5.45) given some arbitrary initial state. Again a cost function is created based upon the speed of convergence of the states e and ψ, and the amount of overshoot in stabilizing these states, adjusting k_1 and k_2 to obtain the desired system behaviour. Here, the cost function is designed to obtain quick convergence of the states with minimal overshoot —

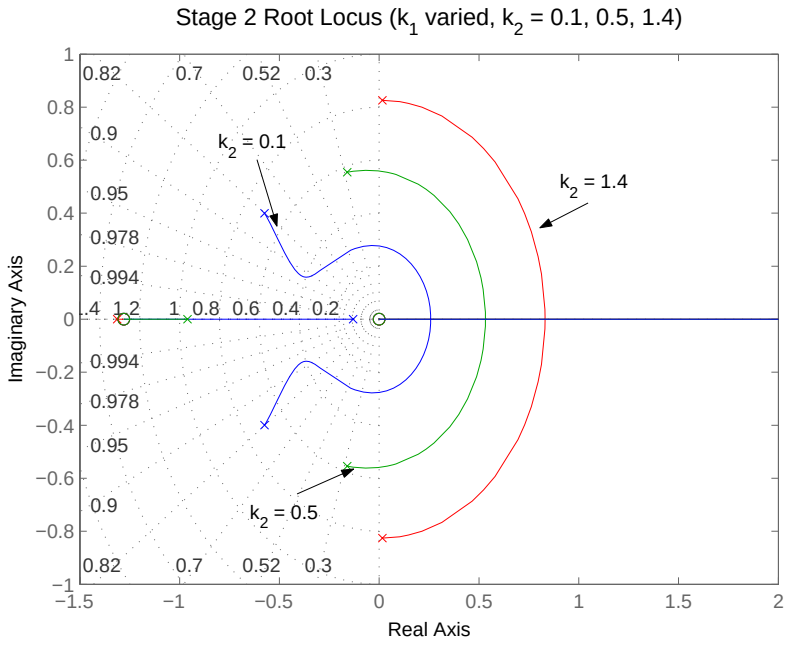


Figure 5.11: Root locus for second stage of position control, where k_1 is varied and $k_2 = 0.1,\ 0.5,\ 1.4$.

the cost function used to achieve this is:

$$J(e,\psi) \quad = \quad w_1 J(e) + J(\psi) \tag{5.49}$$

where

$$J(e) \quad = \quad \int_{t=0}^{t=T} \frac{e(t)}{e(0)}^2 dt + \xi_1(\underline{e}) + \xi_2(\underline{e}) \tag{5.50}$$

and

$$J(\psi) \quad = \quad \int_{t=0}^{t=T} \frac{\psi(t)}{\psi(0)}^2 dt + w_2\xi_1(\underline{\psi}) + \xi_2(\underline{\psi}) \tag{5.51}$$

where $e(t)$ and $\psi(t)$ are the state values at time t, $\underline{e}$ and $\underline{\psi}$ are the state evolution vectors throughout the simulation, and ξ_1 and ξ_2 are functions which calculate the overshoot and settling time for each state, as defined in Figure 5.5, and Equations 5.27 and 5.28. w_1 and w_2 are weighting factors, here set equal to 0.5 and 1.2 respectively. These weighting factors allow a higher 'importance' to be placed on the ψ state and on the requirement for

(e_0, ψ_0)	(k_1, k_2)
$(50, 1)$	$(0.1087, 0.1715)$
$(10, 1)$	$(0.1087, 0.1715)$
$(10, 0.1)$	$(0.1087, 0.1715)$
$(5, 0.3)$	$(0.1087, 0.1715)$

Table 5.2: Optimised Stage 2 position control gains determined using `fminsearch` in MATLAB. Note that the initial values for the other states were set to $\omega = 0$ and $\rho = 0$ for all optimization runs.

minimal overshoot. As for the first stage of control, for the function ξ_2, $\epsilon_e = 0.02$ and $\epsilon_\psi = 0.02$.

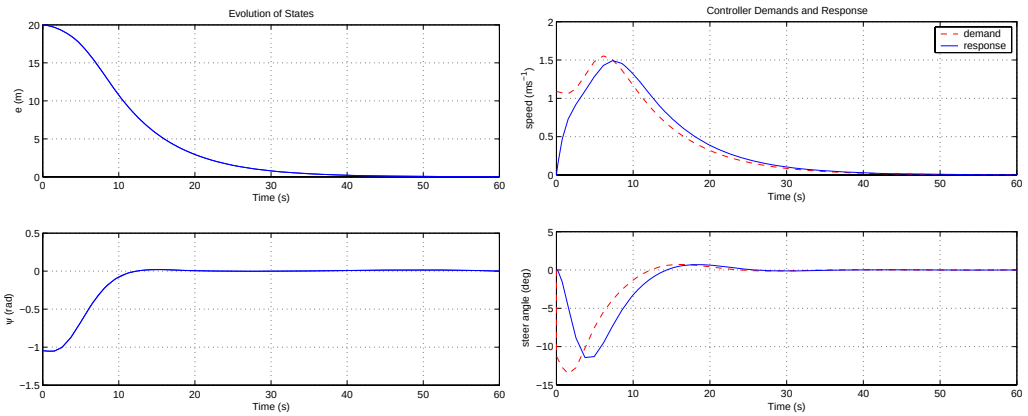
Having defined the cost function, the `fminsearch` function in MATLAB is used to simulate the behaviour of the system, minimising the cost function by numerically searching the parameter space of the gains — the process terminates on minimisation of the cost function. Performing the optimization produces similar gains for a wide range of initial conditions of the state e and ψ, a snapshot of which is provided in Table 5.2.

In terms of system stability, the gain selection of $k_1 = 0.1087$ and $k_2 = 0.1715$ leads to the following roots for the characteristic equation of the system (Equation 5.46):

$$-0.6671 \quad \text{and} \quad -0.2521 \pm 0.3178i$$

indicating the system poles are well-placed for this selection of gains..

The simulated response of the second stage of control on the full model of the vehicle (i.e. including *all* dynamics and non-linearities as described in Section 3.4), with the above tuned gains is shown in Figure 5.12. In this simulation, the initial orientation error is quite large ($-60°$), illustrating the robustness of the controller, even outside the 'linear' region of operation. In addition, the assumption that the velocity loop dynamics were insignificant is validated because the stabilization of the e loop closely follows a first-order response which would not occur if the velocity loop dynamics were significant (significant contribution by the first-order model of the velocity loop would make the e loop dominately second-order). Additionally, the state ψ converges relatively quickly to



(a) Evolution of e and ψ.

(b) Control demands and vehicle response.

Figure 5.12: Response in Stage 2 of the position control loop. Initial vehicle pose is $(x, y, \theta) = (-20, 0, \frac{\pi}{3}\,\text{rad})$.

zero and does not appear to be affected by the inclusion of the velocity loop.

Saturation of the steering angle for this stage of control is not significant, as this stage is only initiated upon stabilization to $\psi = 0$ by the first stage of control. Saturation of the velocity controller actually helps stability as it limits the available velocity used in the ψ stabilization loop. For the velocity loop, saturation is not significant in terms of stability, it merely limits the rate at which e stabilizes to 0.

5.2.1.3 Control supervision

In order to obtain the desired behaviour from the system, that is position stabilization, a control supervisor needs to be designed which switches between the Stage 1 and Stage 2 controllers upon detection of discrete events in the state-space. This is relatively easy for this simple case but requires the provision of a mechanism to *push* the control through the discontinuities introduced by switching between controllers, otherwise the system 'chatters' about the switching boundaries. In addition, due to sensing inaccuracies and the asymptotic nature of these controllers, ψ and e will never reach zero in finite time and thus, thresholds are required to switch between the controllers and to determine when the vehicle has reached the origin.

The control supervisor operates as described in Algorithm 3. In the algorithm, ϵ_ψ and ϵ_e are the tolerances on ψ and e respectively, η_{finish} is the tolerance on the time elapsed before the goal is considered to be reached, and η_2 is the tolerance on the time elapsed before the `ControlState` variable is changed to 'Stage 2'. The variables `goalReached-Timer++`, `stageOneTimer` and `stageTwoTimer` record the time elapsed since detecting the particular state. The default state of the system is 'Stage 1'.

Algorithm 3 Control supervision for position controller.

```
1:  ControlState = Stage 1 {default state}
2:  GoalReached = FALSE {default state}
3:  while (NOT GoalReached) do
4:     {First check if the goal has been reached}
5:     if (|e| < ε_e) then
6:        if (goalReachedTimer++> η_finish) then
7:           GoalReached = TRUE
8:           v* = 0
9:           ω* = 0
10:       end if
11:       {if no, check if Stage 1 is required}
12:    else if (|ψ| < ε_ψ) then
13:       if (stageTwoTimer++> η_2) then
14:          ControlState = Stage 2
15:          stageOneTimer = 0
16:          goalReachedTimer = 0
17:       end if
18:       {if no, Stage 1}
19:    else
20:       ControlState = Stage 1
21:    end if
22:
23:    {The rest of the loop implements the appropriate controller}
24:    .
25:    .
26:    .
27: end while
```

Recapping, the control laws for each stage are:

- Stage 1

$$v^* = \begin{cases} k_1 & \text{if } \cos\psi_{initial} \geq 0 \\ -k_1 & \text{otherwise} \end{cases}$$

where $k_1 = 0.3$, and

$$\omega^* = k_2\psi + \frac{v^* \sin(\psi)}{e}$$

where $k_2 = 0.2139$. To avoid instability due to input saturation, check for $\dot{V} > 0$ using

$$\dot{V} = \psi\left(\frac{v^* \sin\psi}{e} - \text{sat}\left(\omega^*, \omega_{max}\right)\right)$$

where $\omega_{max} = \frac{k_1 \tan\phi_{max}}{L} = 0.1443$. If this condition is true, then the vehicle velocity is set to:

$$v^* = -k_1$$

and the angular velocity demand ω^* is recalculated.

- Stage 2

$$\begin{aligned} v^* &= k_1 e \cos\psi \\ \omega^* &= k_2\psi + \frac{v^* \sin(\psi)}{e} \end{aligned}$$

where $k_1 = 0.1087$, $k_2 = 0.1715$ and rate-limiting using Equation 5.43 is applied to the velocity control law.

5.2.2 Experiments

Simulation results are now presented illustrating that the switching controller does indeed work over the entire workspace. In these experiments, the vehicle was started from different positions in the workspace, all with a common orientation. Then, the vehicle was started from a common position, with different initial orientations. The final simulation is

a Monte Carlo simulation in which the controller is presented with 1000 random initial positions. In these, and all, simulations, the vehicle kinematics and dynamics are modelled as described in Section 3.4. A single landmark is used, from which the IALV strategy is used to derive the vehicle's pose relative to the target pose, nominally the origin. By way of comparison, results from experiments on the real vehicle are then presented.

5.2.2.1 Simulations – covering the state space

In this first experiment, simulation results were obtained for the vehicle servoing to the origin from starting poses located on a circle of radius $5\,\mathrm{m}$ at angular intervals of $\frac{\pi}{4}\,\mathrm{radians}$, all with an initial vehicle orientation of $0\,\mathrm{radians}$. The simulated vehicle paths are plotted in Figures 5.13 and 5.14. Note the uniformity of behaviour for mirrored starting positions.

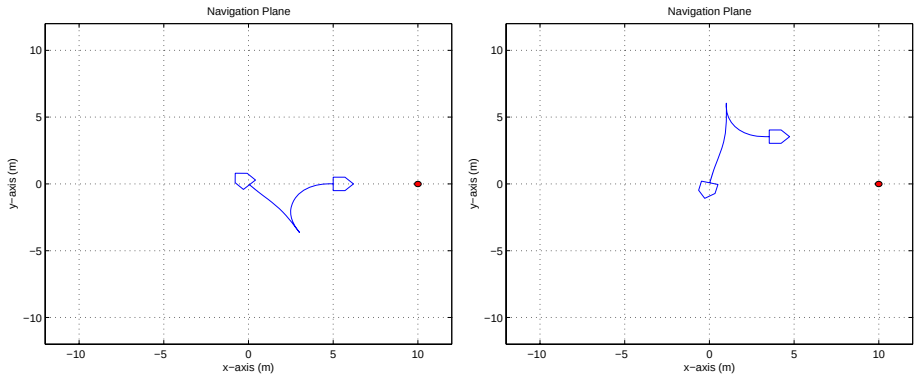
The next case to analyse is a constant starting position with a varying initial orientation, results from these simulation are shown in Figures 5.15 and 5.16. Again, these experiments show that the controller is robust to initial conditions, the only requirement being that the landmarks in the workspace are visible.

Figure 5.17 illustrates results from a Monte Carlo simulation in which position control was simulated from 1000 random initial poses in the range $x, y \in [-12, 12]$ and $\theta \in (-\pi, \pi]$. Figure 5.17 (a) shows the resulting final vehicle positions for each of the random initial poses, and Figure 5.17 (b) shows the time taken to stabilize to the target position for each simulation. This further demonstrates the validity of the control law for position stabilization.

5.2.2.2 Experiments on the vehicle

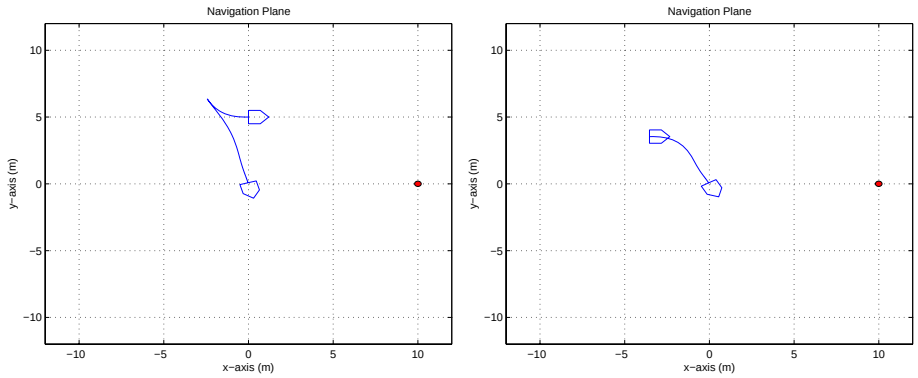
Here, the results from experiments conducted on the vehicle are analysed. A detailed analysis of one particular experiment from a particular starting position is presented, comparing results with the predicted results from a simulation run with the same initial conditions.

The experiment in position control was conducted in an outdoor environment, in



(a) Path generated from initial pose of $(x, y, \theta) = (5, 0, 0)$.

(b) Path generated from initial pose of $(x, y, \theta) = (-3.5356, 3.5356, 0)$.

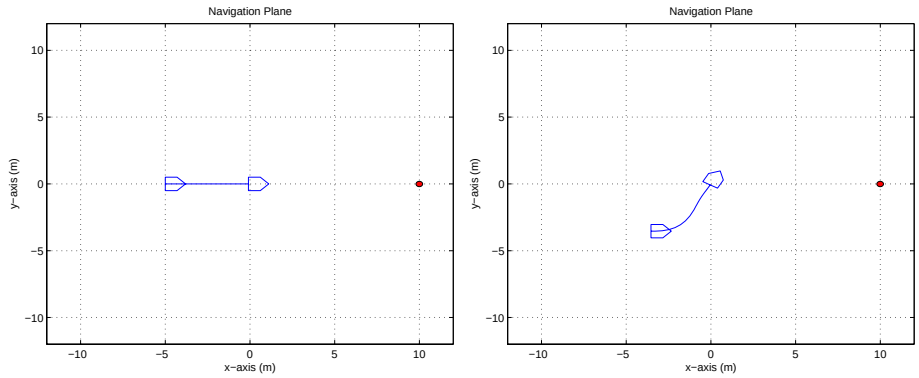


(c) Path generated from initial pose of $(x, y, \theta) = (0, 5, 0)$.

(d) Path generated from initial pose of $(x, y, \theta) = (3.5356, 3.5356, 0)$.

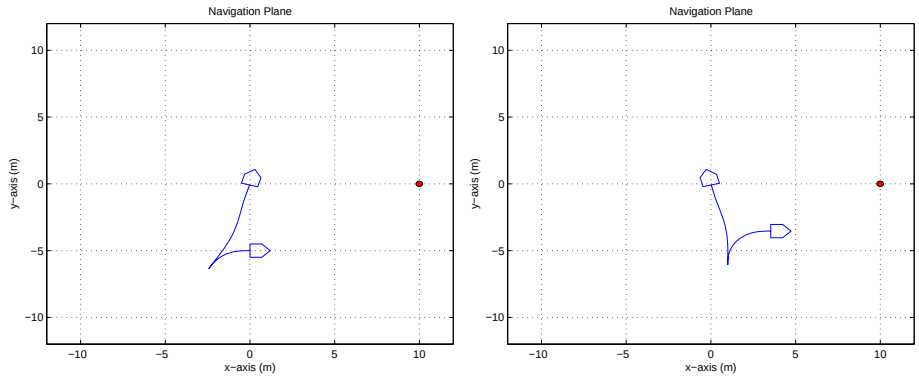
Figure 5.13: Effect of initial position. Vehicle servoing to position $(x, y) = (0, 0)$ from different starting locations, with a constant orientation.

which the vision system was trained to track a red 'witches hat' located within a relatively flat workspace. A landmark was placed at approximately $(x, y) = (5, -1)$ with respect to the intended target pose. The vehicle was manually driven to the target pose, at which point the IALV strategy was initiated giving a pose estimate with respect to this target position. The vehicle was then driven to the 'initial condition' position at $(x, y, \theta) = (-0.5, 2.85, 3^\circ)$, from which control was activated. For ground-truth purposes, data from an RTK-GPS mounted on the vehicle was logged, along with all the relevant



(a) Path generated from initial pose of $(x, y, \theta) = (5, 0, 0)$.

(b) Path generated from initial pose of $(x, y, \theta) = (-3.5356, -3.5356, 0)$.



(c) Path generated from initial pose of $(x, y, \theta) = (0, -5, 0)$.

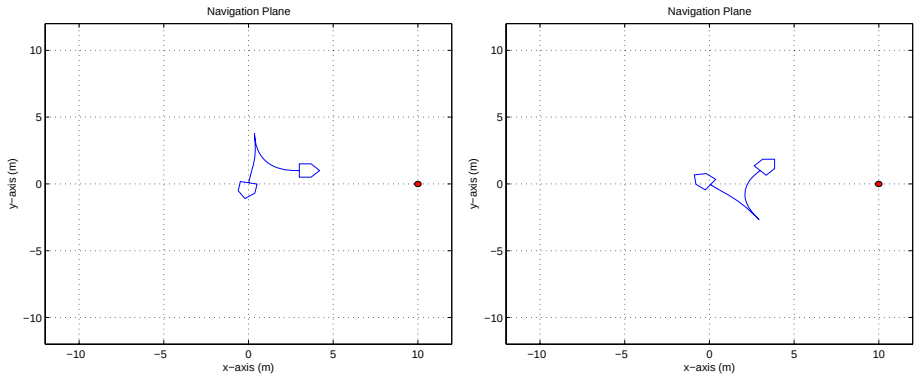
(d) Path generated from initial pose of $(x, y, \theta) = (-3.5356, -3.5356, 0)$.

Figure 5.14: Effect of initial position. Vehicle servoing to position $(x, y) = (0, 0)$ from different starting locations, with a constant orientation

data from the vision and control system. *Note that the GPS is used for ground-truth only and is not at any stage used for control.*

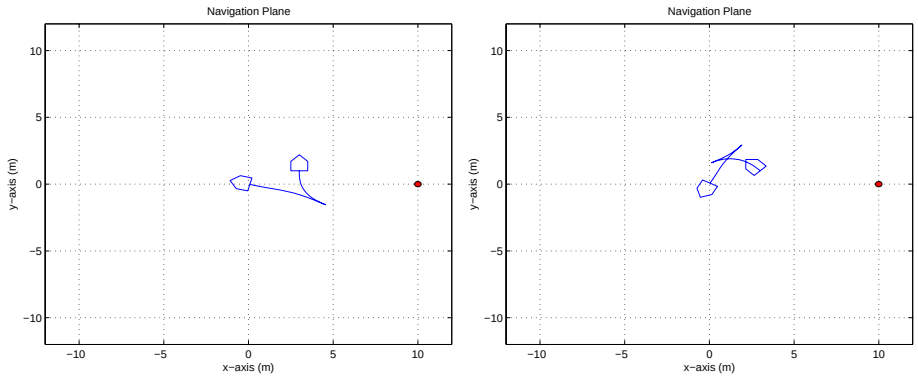
Figure 5.18 (a) shows the ground-plane path of the vehicle, illustrating the differences in position information obtained from the vision-magnetometer system using the IALV strategy, the ground-truth position information obtained from the GPS, and the simulated data.

These results indicate that the simulation of the vehicle is more conservative in the



(a) Path generated from initial pose of $(x, y, \theta) = (3, 1, \pi)$.

(b) Path generated from initial pose of $(x, y, \theta) = (3, 1, -\frac{3\pi}{4})$.



(c) Path generated from initial pose of $(x, y, \theta) = (3, 1, -\frac{\pi}{2})$.

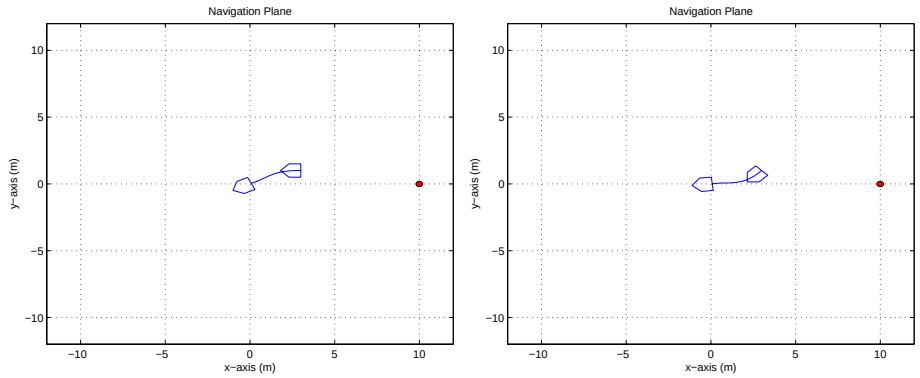
(d) Path generated from initial pose of $(x, y, \theta) = (3, 1, \frac{\pi}{4})$.

Figure 5.15: Effect of starting orientation. Vehicle servoing to position $(x, y) = (0, 0)$ from the starting poses indicated.

limitation of the angular rate of the vehicle (brought about through the second-order model on the steering loop and saturation effects) — the vehicle is in fact capable of 'turning faster' than predicted by the model. Two factors contribute to this:

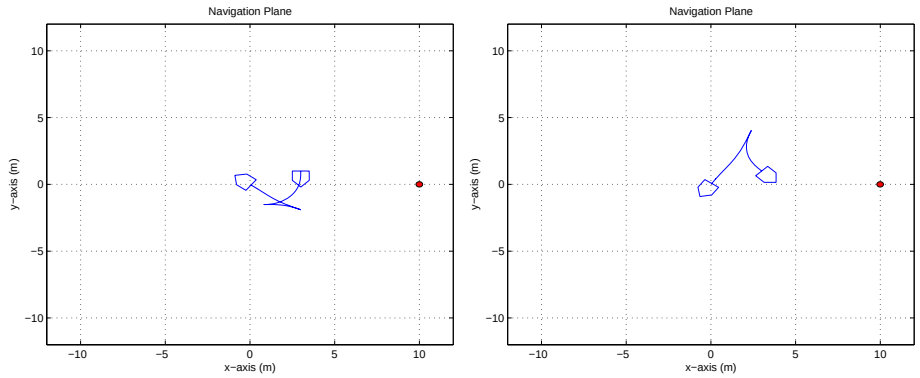
- The simulation uses constant values for ω_n and ζ in the second-order steering loop model whereas in fact, these parameters vary with the vehicle speed.

- Differences in frictional effects between the parameters experimentally determined for the vehicle modelling, and those found under the current experimental condi-



(a) Path generated from initial pose of $(x, y, \theta) = (3, 1, 0)$.

(b) Path generated from initial pose of $(x, y, \theta) = (3, 1, \frac{\pi}{4})$.



(c) Path generated from initial pose of $(x, y, \theta) = (3, 1, \frac{\pi}{2})$.

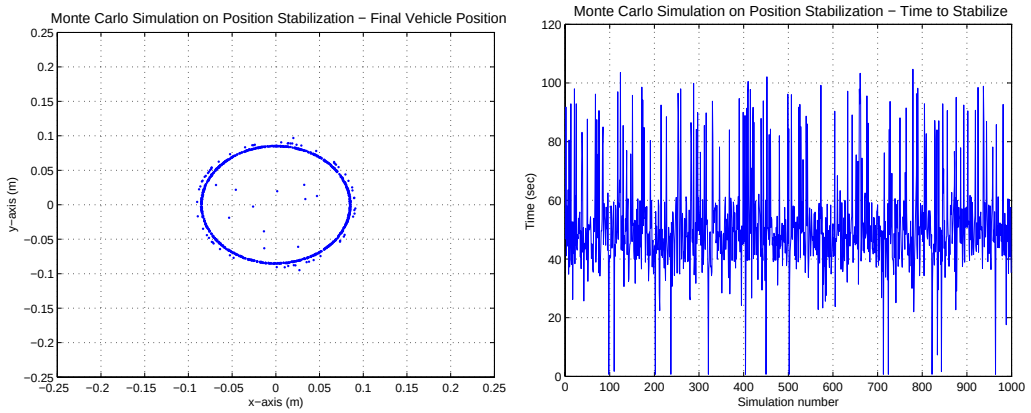
(d) Path generated from initial pose of $(x, y, \theta) = (3, 1, \frac{3\pi}{4})$.

Figure 5.16: Effect of starting orientation. Vehicle servoing to position $(x, y) = (0, 0)$ from the starting poses indicated.

tions (for example, increased tyre wear, different surface, differences in temperature etc.).

This serves to emphasise the difficulty in using open-loop strategies, as model errors significantly impact on the outcome of control — if this model was used in an open-loop controller, significant errors would result if no on-line re-planning occurred.

The next point to note is the difference between the ground-truth reading found from the GPS and the ground-plane path according to the vision-magnetometer system. There



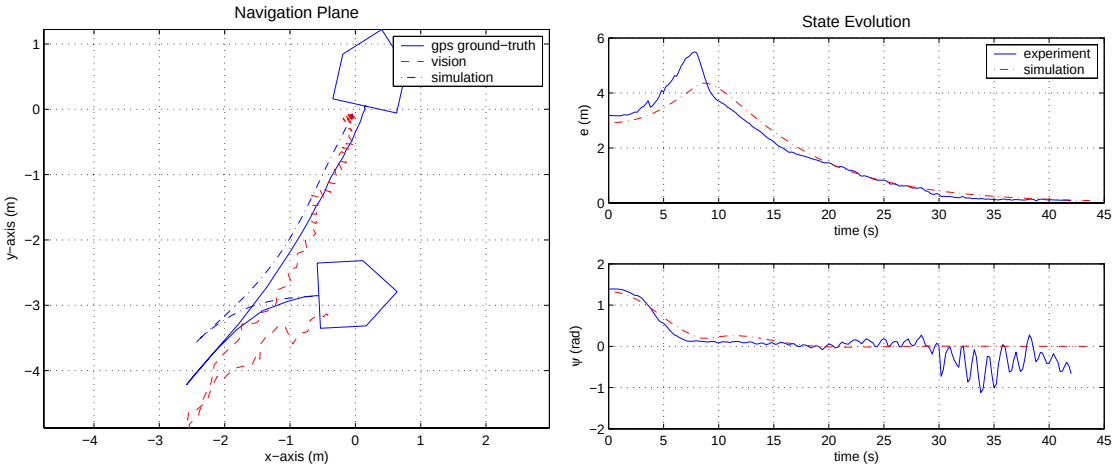
(a) Final vehicle position.

(b) Time to stabilize to position.

Figure 5.17: Final position results and stabilization times from Monte Carlo simulation on position control. In this simulation, 1000 random initial poses were presented to the position controller, all simulations brought the vehicle to within the specified tolerance of the goal, $e < 0.1$.

are several factors in effect here. Recapping, the pose estimate used for control is a function of range and bearing estimates from the vision system and an orientation estimate from the magnetometer. Firstly, the landmark range and bearing estimates are rather noisy, as shown in Section 4.3.2.3. The second factor to consider is the variation of estimated range with object bearing (again this was illustrated in Section 4.3.2.3). Range estimation through the omnidirectional image is a highly nonlinear process, complicated by misalignments in the camera-mirror system and in the camera axis perpendicular alignment with the ground-plane. An additional likely source of error is the possible corruption of the magnetometer readings which are used to provide a reference direction for the IALV strategy. Reinforcing steel is often used in the construction of large concreted areas, such as the workspace of this experiment. This could introduce unpredictable influences on the compass reading, further corrupting the pose estimate.

Consider next the evolution of the state variables e and ψ, shown in Figure 5.18 (b). Experimental and simulation results for the distance to the target, e, correlate well with the exception of the 'bump' in the experimental results at $t = 7\,\text{seconds}$. This sudden rise can



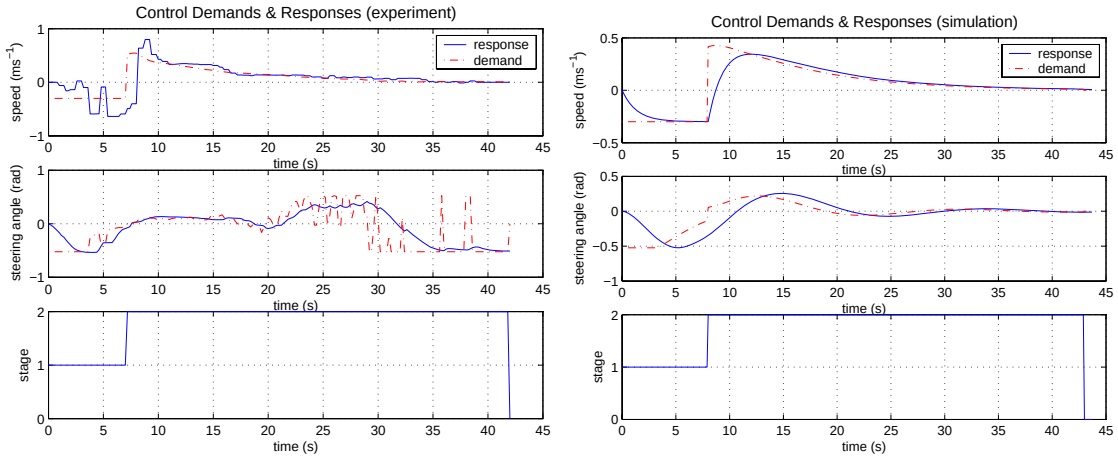
(a) Ground-plane path tracked for position control.

(b) Evolution of the states e and ψ for experiment (vision data shown) and simulation.

Figure 5.18: Comparisons of the response of the experimental and simulated systems. Initial position of the vehicle is $(x, y, \theta) = (-0.5, 2.85, 3^\circ)$.

be explained by the fact that the vehicle initially moves away from the target position and the landmark. The landmark is then further away, for which the estimate of range becomes more inaccurate (in the vision system an image distance of $1\,\mathrm{pixel}$ can equate to a distance of $1\,\mathrm{m}$ in the ground-plane range estimate for ranges over about $9\,\mathrm{m}$). However, after this initial bump, the correlation between experiment and simulation is very good. Correlation between the experimental and simulated values of ψ are extremely good until the vehicle gets close to the target position when the experimental reading starts to oscillate. Again, this can be explained by noise in the system. When the vehicle gets close to the target position, noise in the system can throw the estimation of the vehicle's position from one side of the target position to other, with the associated large changes in the relative angle to the goal, ψ. This effect does not occur with the e variable, as it is a distance measurement which is always positive. When close to the goal, this distance is very small, no matter which side of the goal the vehicle 'apparently' is according to the IALV strategy.

The oscillation in the ψ state estimate as the goal position is neared, influences the steering demand and response in the experiment, as shown in Figure 5.19 (a). Here, the



(a) Demands and response for the experimental system.

(b) Demands and response for the simulation.

Figure 5.19: Comparisons of the response of the experimental and simulated systems. Initial position of the vehicle is $(x, y, \theta) = (-0.5, 2.85, 3^\circ)$. Bottom plot in both figures is the *stage* of control.

steering angle response follows the demand quite nicely in the experimental system for the initial stage but when e becomes smaller and the estimate of ψ becomes unreliable, the steering angle demand starts to oscillate. However, when this occurs, the vehicle is already close to the target position and the effect on position stabilization is minimal. In simulation, the behaviour of the steering demand and response is much smoother but is of a similar form to that seen in the experiment, see Figure 5.19 (b).

In the experiment, the velocity *demand* is of a similar form to that seen in the simulation but the *response* of the vehicle's velocity is quite different. Firstly, when reversing, there is a large oscillation about the demand — a momentary frictional effect could have caused a build up of the integral component in the PID controller on the vehicle. This characteristic is again observed when the velocity swings to the forward direction — there is quite a delay between the issue of a forward demand, and the execution (see Figure 5.19 (a) at $t \simeq 7\,\text{seconds}$). The fact that the simulation and experimental results for the state evolution and ground-plane path of the vehicle are so similar, despite these marked differences in the predicted and actual velocity response, could be due to the similarity

in the time it takes to respond to the demands (when going from a negative to a positive demand). From the plots, it is observed that after $t = 10\,\text{seconds}$, the experimental and simulation results for the velocity response correlate very well.

In addition, it is noted that the time taken to stabilize to a position in the experiment and simulation correlates very well ($42\,\text{seconds}$ and $43\,\text{seconds}$ respectively). Also, switching between the controllers occurs at approximately the same time.

Further examples of position control are provided in the supplementary material in Section A.1. These videos illustrate several position stabilization runs in an outdoor environment for several different initial poses. In these videos, note the changing lighting conditions and the relative robustness of the vision system. Also note the number of other 'red' objects in the environment, and the systems ability to maintain a pose estimate, and drive the vehicle back to the target position.

5.2.3 Summary – Position stabilization experiments

From the experiments on the vehicle it can be inferred that the vehicle modelling is representative of the actual motion and response of the vehicle. Although there were quite marked differences in the velocity response between the simulation and the experiment, the overall outcome of the velocity response (which of course integrates to a position response) was very similar. The simulation results are therefore a reliable indicator that the position stabilization control law developed here is suitable for a car-like vehicle with significant actuator dynamics, allowing 'homing' from any position within the environment.

Experiments on the vehicle highlighted the limitations of the sensing strategy, with the vision system often failing due to poor segmentation of the witches hats. In addition, the homing repeatability, or accuracy, is affected by sensing errors, mainly due to the camera axis of the omnidirectional camera not being precisely perpendicular with the ground-plane. This will lead to a degradation in the homing accuracy, and the vehicle will stabilize to within a wider tolerance of the goal than that specified in the control law — this point was addressed in Section 4.3.4. In addition, because the sensing-localising strategy is so simplistic, there is the constraint in the real-world experiments that landmarks must

be visible. This makes this controller, when combined with the IALV method for position estimation, a highly *local* behaviour.

5.3 Pose stabilization

The topic of pose stabilization, which adds a constraint on the final orientation of the vehicle on reaching the target, is now considered. Pose stabilization for nonholonomic systems, such as unicycle and car-like vehicles, has received much attention in the literature due to the fundamentally nonlinear nature of these systems. The literature on the subject was reviewed in Section 2.6.3 where three major points were highlighted. First, many existing control algorithms may not be physically realizable because they do not consider the dynamics of 'real' systems, such as velocity and steering loop dynamics and further non-linearities such as input saturation. Secondly, there are few instances of implementations of these algorithms on real vehicles, particularly for car-like vehicles (this probably relates back to the first point). Thirdly, there is a growing body of research which is attempting to apply vision in the feedback loop but these robots have so far been limited to small laboratory unicycle-like vehicles and there is little literature on the use of *omnidirectional* vision for the problem.

The work in this section addresses these points. In the first instance, the simulation model of the vehicle developed in Section 3.4 is used to test several existing pose stabilization strategies, illustrating when and why these methods can fail. Then a controller is presented which does cope with the characteristics of 'real' systems. An instability in this controller is addressed, and the controller is further developed for application with the landmark-based pose estimation strategy, the IALV method.

5.3.1 Control law comparisons

In this section, several different pose controllers are presented, highlighting the deficiencies of many nonholonomic control algorithms when they are applied to 'real' systems with significant dynamics in the velocity and steering loops. In particular, overcoming

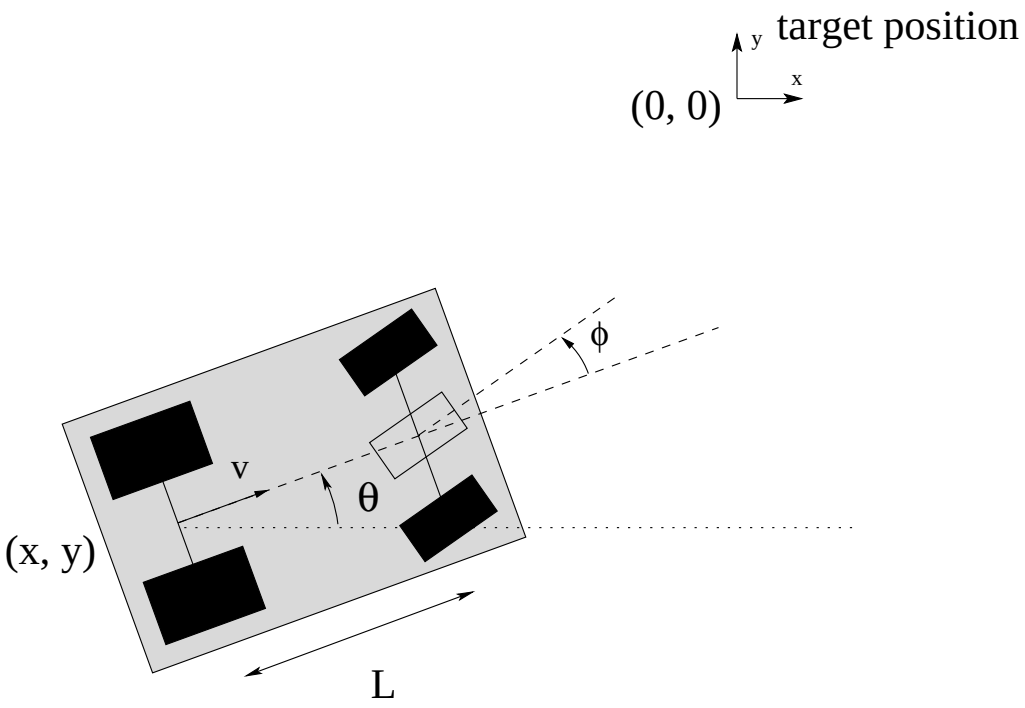


Figure 5.20: Coordinate system for control.

steering input saturation is the key to stabilizing a car-like vehicle to a desired pose. In this section, the nomenclature of the original authors is preserved were possible.

5.3.1.1 Chained form for a car-like vehicle

Before exploring these controllers, the car-like system kinematics are revisited and a common canonical representation of the system, the *chained form*, is presented.

Referring to Figure 5.20, and repeating the equations presented in Section 2.6.2, the system kinematic equations are:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{L} \\ 0 \end{bmatrix} v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{5.52}$$

where the demands to the system are the translational velocity, $v_1 = v$, and the steering angle rate, $v_2 = \dot{\phi}$. Canonical forms for nonholonomic systems are useful for the systematic development of control strategies, both open and closed-loop [De Luca *et al.*, 1997].

One of the more useful canonical structures is the *chained form*. Although nonlinear, the chained form has a strong underlying linear structure [De Luca *et al.*, 1997]. Methods for the systematic conversion of nonholonomic systems to a chained form were presented by Sørdalen [1993]. Here, the conversion for the case of a car-like vehicle is presented. The first step is a change of coordinates:

$$\begin{aligned} x_1 &= x \\ x_2 &= \tfrac{\tan\phi}{L}\sec^3\theta \\ x_3 &= \tan\theta \\ x_4 &= y \end{aligned} \tag{5.53}$$

followed by the input transformation:

$$\begin{aligned} v_1 &= \tfrac{u_1}{\cos\theta} \\ v_2 &= -\tfrac{3\sin\theta}{L\cos^2\theta}\sin^2\phi u_1 + L\cos^3\theta\cos^2\phi u_2 \end{aligned} \tag{5.54}$$

leading to the chained form representation:

$$\begin{aligned} \dot{x_1} &= u_1 \\ \dot{x_2} &= u_2 \\ \dot{x_3} &= x_2 u_1 \\ \dot{x_4} &= x_3 u_1 \end{aligned} \tag{5.55}$$

This structure is commonly used in the literature on nonholonomic mobile robot systems as it is easily extendible to higher-order problems, such as a car-like vehicle towing n-trailers. Other canonical forms exist (see e.g. Huo and Gei [2001]) but the chained form is by far the most widely used.

5.3.1.2 Continuous, time-varying control

Time-varying control for car-like vehicle pose stabilization is based upon the intuitive notion that an approximately periodic forward / reverse motion with appropriate steering demands will bring the vehicle to the desired pose. This first controller considered here is constructed from a smooth function of the robot state combined with an exogenous

time-varying term provided by a 'heat' function [Samson, 1995; De Luca *et al.*, 1997], exploiting the internal structure of the chained from representation of the car-like vehicle system. Controller design occurs in two stages. First, it is assumed that one control input is given; the second control is then designed around the reduced sub-vector of the system state. The second stage of design aims to stabilize the remaining variable, i.e. the variable not contained in the sub-vector of the system state, using the first control input.

First, for convenience, the variables are reordered:

$$\underline{\chi} = (\chi_1, \chi_2, \chi_3, \chi_4) = (x_1, x_4, x_3, x_2) \tag{5.56}$$

bringing the chained system to the following form:

$$\begin{aligned} \dot{\chi_1} &= u_1 \\ \dot{\chi_2} &= \chi_3 u_1 \\ \dot{\chi_3} &= \chi_4 u_1 \\ \dot{\chi_4} &= u_2 \end{aligned} \tag{5.57}$$

De Luca *et al.* [1997] introduce the modified chained form, aiming to put the system into a *skew-symmetric* form. The first step involves the further change of coordinates:

$$\begin{aligned} z_1 &= \chi_1 \\ z_2 &= \chi_2 \\ z_3 &= \chi_3 \\ z_4 &= k_1 \chi_2 + \chi_4 \end{aligned} \tag{5.58}$$

leading to the skew-symmetric form of the chained system:

$$\begin{aligned} \dot{z_1} &= u_1 \\ \dot{z_2} &= u_1 z_3 \\ \dot{z_3} &= -k_1 u_1 z_2 + u_1 z_4 \\ \dot{z_4} &= -k_2 u_1 z_3 + w_2 \end{aligned} \tag{5.59}$$

where w_2 is a new input signal

$$w_2 = (k_1 + k_2)\, u_1 z_3 + u_2 \tag{5.60}$$

and $k_1,\ k_2 > 0$.

De Luca *et al.* [1997]; Samson [1995] prove the asymptotic stability of the system through the following choice of control inputs:

$$u_1 = -k_{u_1} z_1 + \eta\left(z_2, z_3, z_4, t\right) \tag{5.61}$$

where

$$\eta\left(z_2, z_3, z_4, t\right) \;=\; \begin{cases} k_\eta & \text{if } z_2^2 + z_3^3 + z_4^4 \geq \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{5.62}$$

with ϵ being some tolerance around the goal pose, and

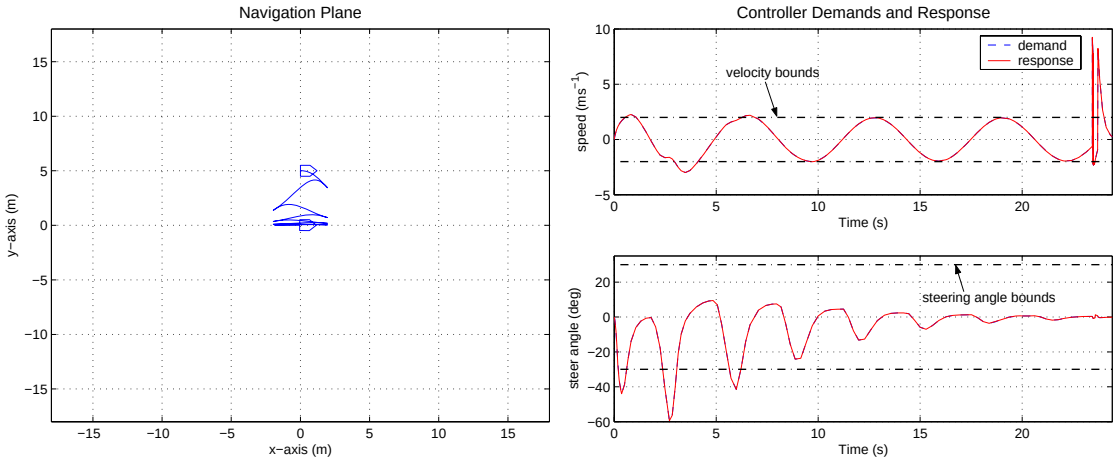
$$u_2 = -k_{w_2}^{'} z_1 |u_1| z_4 - (k_1 + k_2) u_1 z_3 \tag{5.63}$$

The gains, k_1, k_2 and $k_{w_2}^{'}$ are chosen based upon an analysis of the eigenvalues of the system, for details refer to De Luca *et al.* [1997]. For these simulations, the following gain values were used:

$$k_1 = \frac{1}{3}, \quad k_2 = \frac{8}{3}, \quad k_{w_2} = 3, \quad k_{u_1} = 5, \quad k_\eta = 20 \text{ and } \epsilon = 10^{-3} \tag{5.64}$$

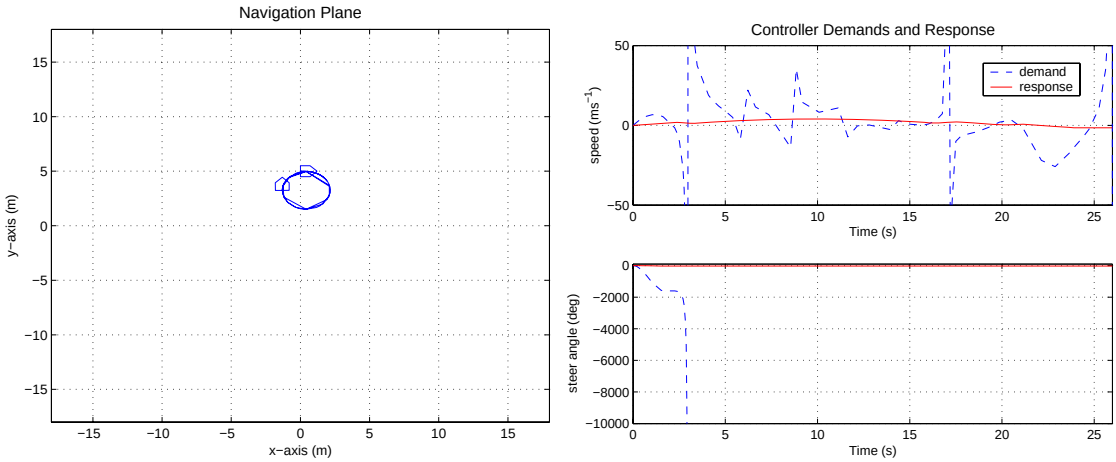
Figure 5.21 illustrates the results of the above control law applied to the car-like vehicle simulation *with no dynamics or input saturation*. Note the quite natural behaviour of the system at the start of the journey and the similarity to what intuitively, the path would appear when parallel parking a car. However, as the vehicle gets closer to the goal, the forward / reverse motion is rather erratic. In addition, note the large excursions outside the allowable demand for the steering angle.

The first problem with this controller is an instability for an initial orientation of $\theta = \pm\frac{\pi}{2}\,\mathrm{rad}$ leading to either a computational error or wild robot motion — in either case a failure. This is due to the chained form representation. The chained transforms, Equations 5.53 and 5.54, both contain a $\frac{1}{cos\theta}$ term, which approaches infinity for $\theta \rightarrow \frac{\pi}{2}$. In practice, this 'state' is hardly likely to occur on a real system due to noise and in any case could be avoided through the use of logic.



(a) Path followed (simulation). (b) Demands and responses (simulation).

Figure 5.21: Response of the smooth, time-varying control, first simulation. Initial position of the vehicle is $(x, y, \theta) = (0, 5, 0)$. No dynamics or input saturation included.



(a) Path followed (simulation). (b) Demands and responses (simulation).

Figure 5.22: Response of the smooth, time-varying control, second simulation. Initial position of the vehicle is $(x, y, \theta) = (0, 5, 0)$. Dynamics and input saturation included.

The inclusion of dynamic effects or input saturation leads to complete failure of the system, see Figure 5.22. Re-tuning the gains of the controller did not improve the behaviour of the system and including the vehicle dynamics in the controller design appears

to be a daunting task given the complexity of the controller. In addition, even for the case of a 'perfect' car-like vehicle, the resulting motion of the vehicle is highly oscillatory and contains a discontinuity for $|\theta| \rightarrow \frac{\pi}{2}$ due to the chained from transformation.

5.3.1.3 Piece-wise continuous, time-varying control

Next the controller developed by De Luca *et al.* [1997] is presented. This controller was in turn was based upon the work of Sørdalen and Egeland [1995]. Again, this controller is time-varying, similar in many ways to the first controller, except that the feedback law depends, in addition to the exogenous time-varying function, on a piecewise continuous function of the state. Design proceeds using the back-stepping method, which is a common technique for chained form integrator systems.

First, repeating the chained form system equations,

$$\begin{aligned} \dot{x_1} &= u_1 \\ \dot{x_2} &= u_2 \\ \dot{x_3} &= x_2 u_1 \\ \dot{x_4} &= x_3 u_1 \end{aligned} \tag{5.65}$$

the state vector is partitioned as $\underline{X} = (x_1, \underline{X}_2)$, where $\underline{X}_2 = (x_2, x_3, x_4)$. u_1 is chosen in an open-loop fashion and is updated as a function of state at discrete instants of time. It also contains the exogenous time-varying term which gives the system control energy at points it would otherwise have little or none. Given a sequence of equi-distance time instants $\{t_0, t_1, t_2, \ldots\}$:

$$t_h = hT \tag{5.66}$$

where $T = t_{h+1} - t_h > 0$, the first control is

$$u_1 = k(X(t_h))f(t) \tag{5.67}$$

which is a function of the state $\underline{X}$ at time-step $t = t_h$, and during the interval, is defined in open-loop. $k(X)$ is given by:

$$k(X) = \text{sat}\left(-\frac{\omega}{\pi}\left[x_1 + \text{sgn}(x_1)\kappa\sqrt{x_2^2 + x_3^2 + x_4^2}\right], K\right) \tag{5.68}$$

where the states are provided at $t = t_h$. The function $f(t)$ is:

$$f(t) \;=\; \frac{1 - cos(\omega t)}{2},\;\; \omega = \frac{2\pi}{T} \tag{5.69}$$

The parameters $\omega,\ \kappa,$ and K are positive, real values, used to tune the system response.

Design of the control law for the second input proceeds using the back-stepping method on the lower part of the partitioned system,

$$\underline{\dot{X}_2} = \begin{bmatrix} \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{bmatrix} = \begin{bmatrix} u_2 \\ k(X(t_h))x_2 \\ k(X(t_h))x_3 \end{bmatrix} \tag{5.70}$$

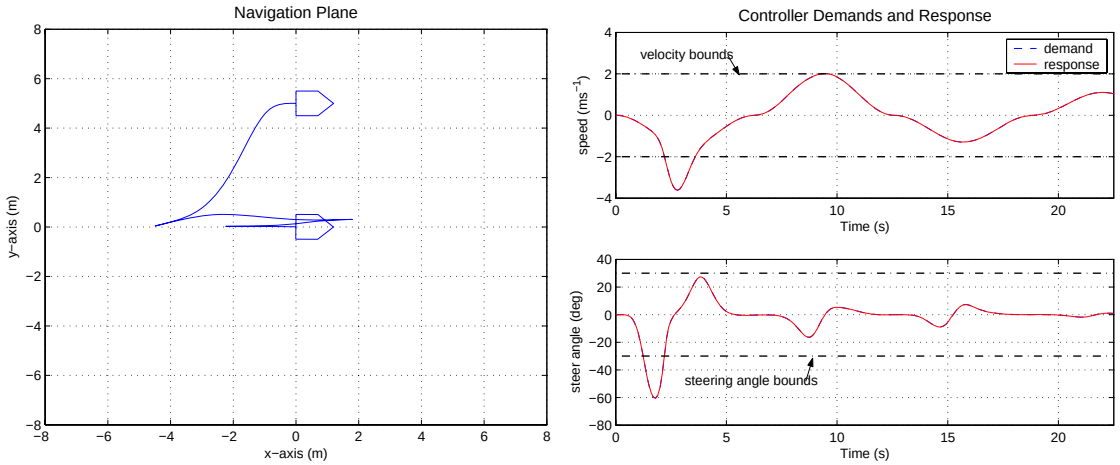
Back-stepping is a general method for controlling systems in cascaded form[3] (see e.g. [Krstić and Kokotović, 1996]). Essentially, back-stepping takes the last equation in the system of Equation 5.70, and treats x_3 as a dummy variable which is used to drive the state x_4 towards the target state ($x_4 \rightarrow 0$). The next variable in the chain, x_2, is then considered as the dummy variable, and is used to drive x_3 towards the target state. This process is repeated until the top equation is reached which contains the true input, u_2. Details of the derivation for this controller are given in De Luca *et al.* [1997], where stability and convergence proofs are also presented. The final form of the second input is:

$$u_2 = \begin{bmatrix} \Gamma_2(k(X(t_h)) & \Gamma_3(k(X(t_h)) & \Gamma_4(k(X(t_h)) \end{bmatrix} \begin{bmatrix} x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{5.71}$$

where

$$\begin{aligned} \Gamma_2 &= -\lambda_2 + f^2(t)g_{23} \\ \Gamma_3 &= \frac{f(t)\left[\lambda_2 f(t)g_{23} + 2\dot{f}(t)g_{23} + f(t)\dot{g}_{23} + f^2(t)g_{24}\right]}{k(X(t_h))} \\ \Gamma_4 &= \frac{f(t)\left[\lambda_2 f(t)g_{24} + 2\dot{f}(t)g_{24} + f(t)\dot{g}_{24} + f^2(t)g_{25}\right]}{k^2(X(t_h))} \end{aligned}$$

[3]Cascaded form systems are those in which the state equations are linked by a chain of integrators.



(a) Path followed (simulation).

(b) Demands and responses (simulation).

Figure 5.23: Response of the non-smooth, time-varying control, first simulation. Initial position of the vehicle is $(x, y, \theta) = (0, 5, 0)$. No dynamic or input saturation included for this simulation.

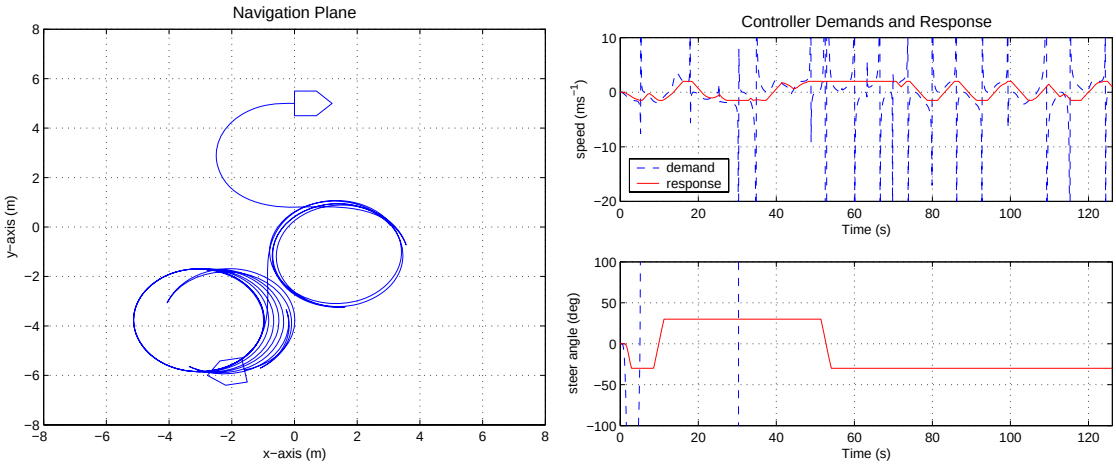
and

$$\begin{aligned} g_{23} &= -\lambda_3 - \lambda_4 f^2(t) \\ \dot{g}_{23} &= -2\lambda_4 f(t)\dot{f}(t) \\ g_{24} &= -\lambda_4 \left(\lambda_3 f^4(t) + 4\dot{f}(t)\right) \\ \dot{g}_{24} &= -4\lambda_3\lambda_4 f^3(t)\dot{f}(t) - 4\lambda_4 f(t) \\ g_{25} &= 0 \end{aligned}$$

The parameters used for this simulation are (as in De Luca *et al.* [1997]):

$$K = 1, \quad \omega = 1, \quad \kappa = 3, \quad \lambda_2 = \lambda_3 = \lambda_4 = 1 \tag{5.72}$$

Figure 5.23 illustrates the response for an initial pose of $(x, y, \theta) = (0, 5, 0)$. This non-smooth (piecewise continuous), time-varying controller elicits a much smoother response from the vehicle than the continuous, time-varying controller presented in Section 5.3.1.2. The excursions outside the allowable demand bounds are also modest. However, similar problems arise when dynamics and input saturation are introduced. Figure 5.24 illustrates



(a) Path followed (simulation).

(b) Demands and responses (simulation).

Figure 5.24: Response of the non-smooth, time-varying control, second simulation. Initial position of the vehicle is $(x, y, \theta) = (0, 5, 0)$. Dynamics and input saturation are included for this simulation.

the response for an initial pose of $(x, y, \theta) = (0, 5, 0)$, where input saturation and dynamics have been included in the model. Even with significant gain tuning, a satisfactory behaviour could not be obtained. In the case of no vehicle dynamics or input saturation, shown in Figure 5.23, the demands were not significantly outside the allowable bounds. this indicates that it is the combination of dynamics *and* input saturation which causes problems. As with many of the strategies which attempt to resolve pose stabilization in one step, when dynamics and input saturation are included, gain tuning is often highly dependent on initial conditions, and with the complexity of this controller, very difficult to do in an analytical fashion.

5.3.1.4 Discontinuous control (coordinate transform induced)

The next strategy tested was developed by Astolfi [1996]. This strategy also relies on a coordinate transformation, not unlike that used to transform nonholonomic systems into the chained from. It allows for the design of smooth controllers in the new coordinate system, which upon transformation back to the original system, again become discontinuous.

Repeating the original system kinematic equations:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{L} \\ 0 \end{bmatrix} v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{5.73}$$

Astolfi [1996] then applies the standard input transformation as used for chained form systems:

$$v_1 = \frac{u_1}{\cos\theta} \tag{5.74}$$

$$v_2 = -\frac{3\sin\theta}{L\cos^2\theta}\sin^2\phi u_1 + L\cos^3\theta\cos^2\phi u_2 \tag{5.75}$$

The states of Equation 5.73 are then transformed using the σ process [Astolfi, 1996]:

$$\begin{aligned} \xi_1 &= x \\ \xi_2 &= \tfrac{\tan\phi}{L}\sec^3\theta \\ \xi_3 &= \tfrac{\tan\theta}{x} \\ \xi_4 &= \tfrac{y}{x^2} \end{aligned} \tag{5.76}$$

Like the chained form transformation of Equations 5.53, this transformation contains a discontinuity for $|\theta| \rightarrow \frac{\pi}{2}$ but here, a further discontinuity exists for $x \rightarrow 0$. In any case, in the new coordinate system, the kinematics of the car-like vehicle are described by:

$$\begin{aligned} \dot{\xi_1} &= u_1 \\ \dot{\xi_2} &= u_2 \\ \dot{\xi_3} &= \tfrac{\xi_2 - \xi_3}{\xi_1} u_1 \\ \dot{\xi_4} &= \tfrac{\xi_3 - 2\xi_4}{\xi_1} u_1 \end{aligned} \tag{5.77}$$

For *almost exponential* stabilization, Astolfi [1996] proposes the following feedback law, presenting proofs of convergence for the entire state-space with the exception of $\xi_1 = 0$:

$$\begin{aligned} u_1 &= -k\xi_1 \\ u_2 &= p_2\xi_2 + p_3\xi_3 + p_4\xi_4 \end{aligned} \tag{5.78}$$

where $k > 0$ and p_i are system gains. In choosing values for p_i and k, the lower three equations in Equation 5.77 are considered. Substituting the feedback laws (Equation 5.78), in matrix form the closed-loop system is:

$$\begin{bmatrix} \dot{\xi}_2 \\ \dot{\xi}_3 \\ \dot{\xi}_4 \end{bmatrix} = \begin{bmatrix} p_2 & p_3 & p_4 \\ -k & k & 0 \\ 0 & -k & 2k \end{bmatrix} \begin{bmatrix} \xi_2 \\ \xi_3 \\ \xi_4 \end{bmatrix} = \underline{AX} \tag{5.79}$$

p_i and k are chosen such that the Eigenvalues of the matrix A are negative. Using this strategy, for this case, the values of these parameters are:

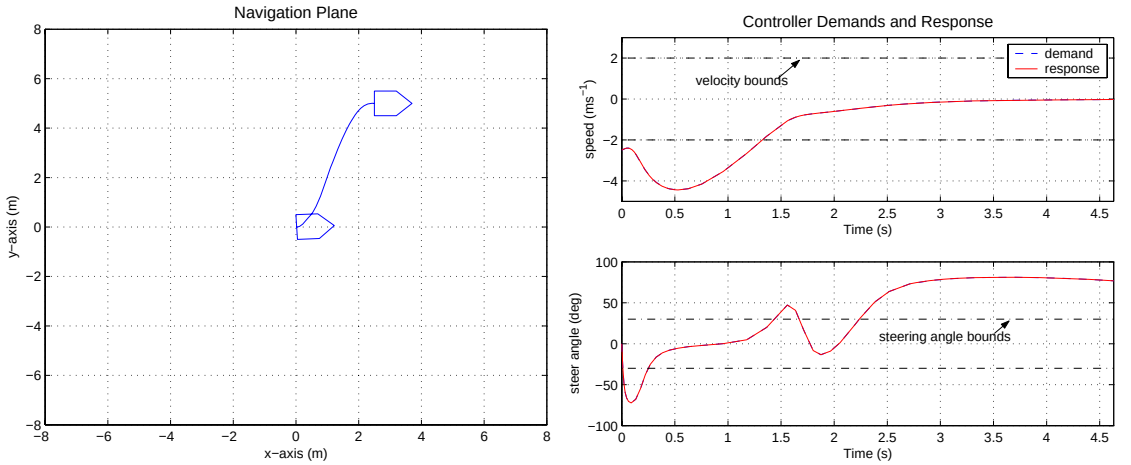
$$k = 1, \quad p_2 = 1, \quad p_3 = 3, \quad p_4 = 5 \tag{5.80}$$

The first simulation could not be conducted with an initial condition of $(x, y, \theta) = (0, 5, 0)$, as this particular controller is not defined for $x = 0$. Instead, the initial position is moved to $(x, y, \theta) = (-2.5, 5, 0)$. The results of the application of the control to the system in the absence of any dynamic or saturation effects are shown in Figure 5.25. From the plots, for both the demands, excursions outside the allowable values are significant.

Figure 5.26 illustrates the response for an initial pose of $(x, y, \theta) = (-2.5, 5, 0)$, where dynamic and saturation effects have been included. Again, these results indicate that the controller fails to cope with the characteristics of a real vehicle. Tuning of the system to accommodate these dynamics is difficult due to the complexity of the controller. In any case, the controller design is such that it excludes the entire y-axis. This is probably of limited importance in a real system as, due to noise, this state is unlikely to occur exactly. Nonetheless, erratic behaviour results on approach to the y-axis and although theoretically interesting, this controller is unacceptable for further investigation.

5.3.1.5 Discussion

In summary, the controllers presented here are representative of the range of controllers experimented with early in the thesis work. These controllers were unable to cope with the characteristics found in a real vehicle, namely velocity and steering loop dynamics



(a) Path followed (simulation).

(b) Demands and responses (simulation).

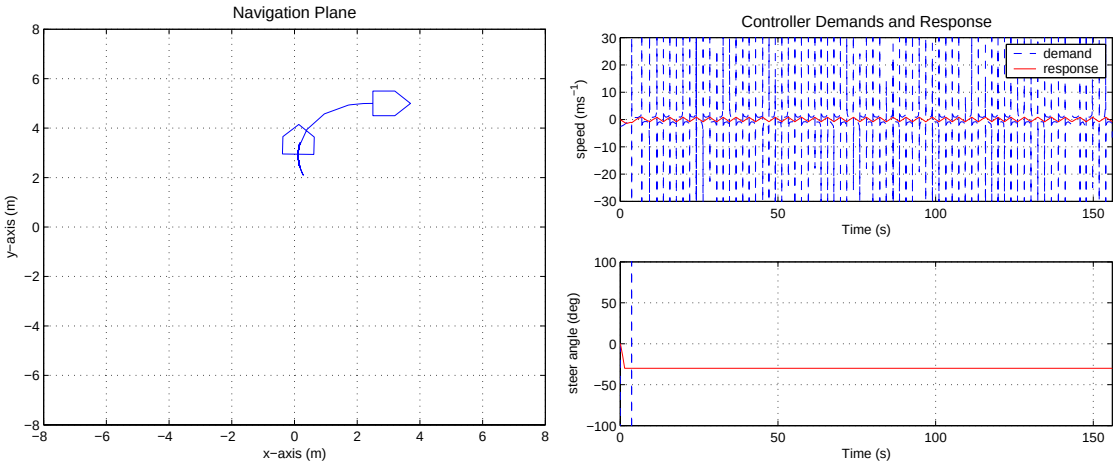
Figure 5.25: Response of the coordinate transform based controller, first simulation. Initial position of the vehicle is $(x, y, \theta) = (-2.5, 5, 0)$. No dynamic or input saturation included for this simulation.

and input saturation. The analysis of the system with the inclusion of these effects is difficult due to the inherent complexity of these controllers. Failure occurs because pose stabilization is attempted in essentially one step — on saturation of the steering input wild control demands are elicited.

In addition, the required coordinate transforms introduce discontinuities. In practice, few of these controllers have been implemented on a *car-like* vehicle, rather the unicycle-like system is more popular for experimentation. In Cartesian space, the kinematics of the unicycle are (repeating from Section 2.6.2):

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} \tag{5.81}$$

where the demands for the system are v and ω, and x, y and θ are the coordinates of the vehicle. This system can readily be transformed into a *polar* form ([Badreddin and



(a) Path followed (simulation).

(b) Demands and responses (simulation).

Figure 5.26: Response of the coordinate transform based controller, second simulation. Initial position of the vehicle is $(x, y, \theta) = (0, 5, 0)$. Dynamics and input saturation are included for this simulation.

Mansour, 1993]):

$$\begin{bmatrix} \dot{e} \\ \dot{\delta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -v\cos\psi \\ \frac{v\sin\psi}{e} - \omega \\ \frac{v\sin\psi}{e} \end{bmatrix} \tag{5.82}$$

in which linear controllers can be designed. The discontinuity introduced in the unicycle-like system is usually restricted to the origin rather than occupying an entire axis in Cartesian space as for the car-like vehicle (see Equations 5.54 and 5.53), and thus, experiments with 'real' unicycles have been relatively successful (see e.g. [Astolfi, 1996]).

The next section explores the use of a switching control strategy consisting of a set of piecewise continuous controllers, linked together by a control supervisor. This overcomes many of the problems with existing controllers, illustrated in this section. It also allows for the use of linear tools for system analysis and gain tuning.

5.3.2 Control law design

The idea of splitting the state-space into sub-manifolds, each of which is sequentially stabilized, was proposed by Bloch and McClamroch [1989]; Bloch *et al.* [1990]. This approach consists of two stages [Bloch and McClamroch, 1989; Bloch *et al.*, 1990; De Luca and Oriolo, 1995]:

- First, find an open-loop motion strategy which can achieve the desired behaviour of the system.
- Second, transform the motion sequence into a succession of equilibrium manifolds, each of which can then stabilized by feedback.

Overall, the resulting feedback is discontinuous, due to switching between the different control laws for each of the manifolds [De Luca and Oriolo, 1995]. The method is simple, and the development of open-loop motion sequences for many systems in straightforward. The simplicity of the method allows for the use of linear control tools for each phase of control and dynamic and input saturation effects can readily be accommodated.

However, the technique relies on the ability to form an open-loop motion strategy for the system stabilization — such strategies may not be obvious for some nonholonomic systems. In addition, disturbances to states which are not controlled by the current phase of control cannot be corrected at the time they occur unless another phase of control is initiated [De Luca and Oriolo, 1995]. This has not been a significant limitation in this work.

Although not recognised as such, this idea was successfully applied to a differentially steered robot which was artificially constrained (through software) to mimic the motion of a car-like vehicle by Lee *et al.* [1999]. The controller presented here is based on the work of Lee *et al.* [1999] and operates by splitting the state-space into a set of sub-manifolds, each of which is sequentially stabilized, as was done in Section 5.2.1.

The controller is designed in Cartesian space, as shown in Figure 5.27, and does not require any coordinate or input transformations. Without loss of generality consider the

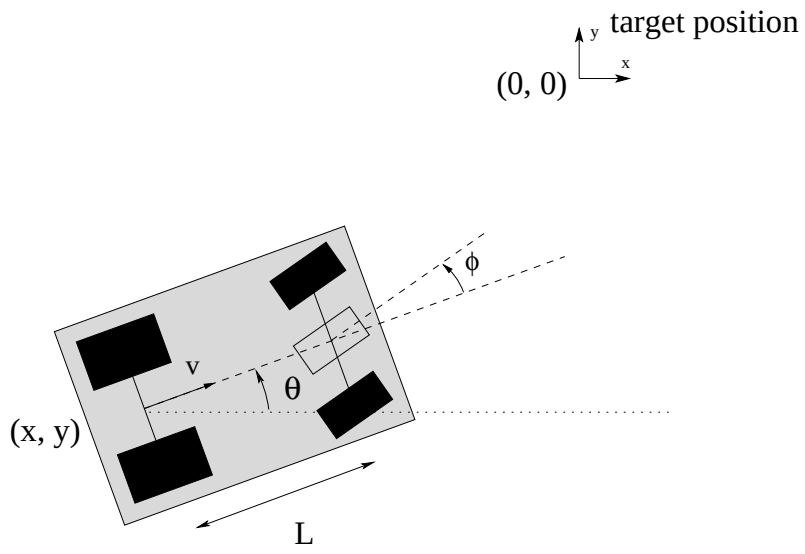


Figure 5.27: Coordinate system for pose control.

target pose to be $(x, y, \theta) = (0, 0, 0)$. In the original work of [Lee *et al.*, 1999], the control operated in two distinct phases: stabilize the vehicle to the x-axis with a zero vehicle orientation (i.e. stabilize $y \rightarrow 0$ and $\theta \rightarrow 0$), then stabilize to the origin (i.e. $x \rightarrow 0$). Here, an additional stage to the controller has been added which aims to minimise the space used in servoing to a pose by using a position stabilization strategy (as used in Section 5.2.1) which is activated when the vehicle is 'far from home', reverting to the pose stabilizing controller on reaching some tolerance distance to the home location. Note that this stage of control is rarely required as the range of operation of the vehicle is limited by its sensing capacity. Once within this 'home zone', the vehicle cannot exit the area, again minimizing the required space for pose stabilization.

The switching strategy is summarised as:

- First stage (homing): If $e > r_e$, $e \rightarrow r_e$ and $\psi \rightarrow 0$
- Second stage (servo-to-line): $y \rightarrow 0$ and $\theta \rightarrow 0$
- Third stage (servo-to-point): $x \rightarrow 0$

where r_e is the radius of the area 'homed' to, see Figure 5.28.

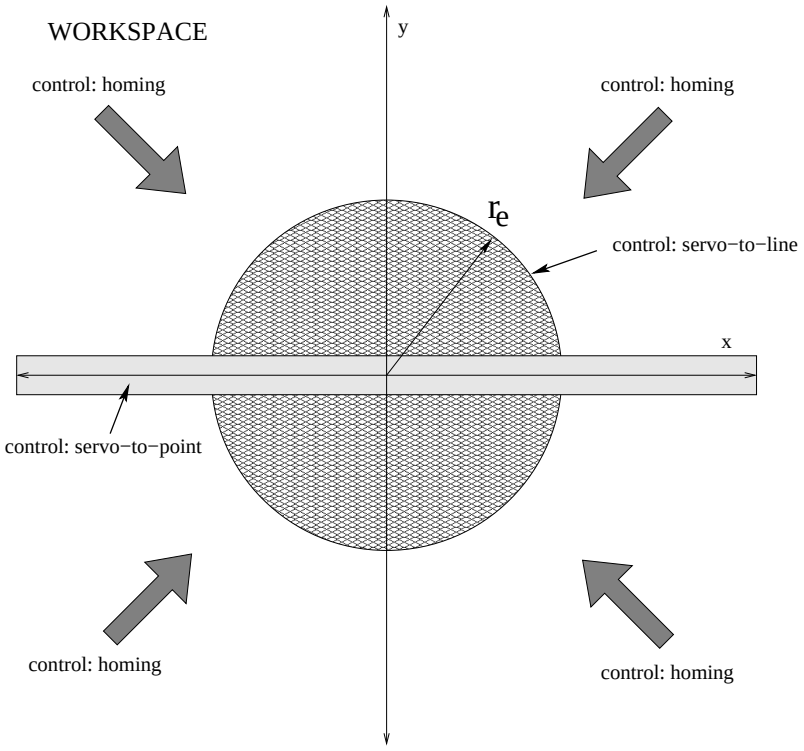


Figure 5.28: The pose stabilization strategy. The first stage takes the vehicle to within a tolerance distance of the goal, the second stage stabilizes the vehicle to the x-axis, facing $\theta = 0$, and the third stage stabilizes to the point $x = 0$.

5.3.2.1 First stage: homing

This first stage of control uses the position controller presented in Section 5.2.1 and, as such, in effect has two sub-stages of control. Gain tuning and control design proceed as described in Section 5.2.1 — the difference here is that homing occurs to a much broader tolerance of the goal position, of the order of $r_e = 6\,\mathrm{metres}$. Once this region is reached, the next stage of control is initiated.

5.3.2.2 Second stage: servo-to-line

First, repeating the system kinematic representation, with ϕ rather than $\dot{\phi}$ as the second input:

$$\begin{aligned} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= v\frac{\tan\phi}{L} \end{aligned} \tag{5.83}$$

where the demands for the system are v and ϕ, and x, y and θ are defined with reference to the target position and the x-axis as shown in Figure 5.27. As for the design of the position controller, the relation

$$\omega = v\frac{tan\phi}{L}$$

will be substituted during the analysis for simplification, it is an easy matter to transform between the two representations.

This stage of control aims to take the vehicle to the x-axis with an orientation of $\theta = 0$. A suitable Lyapunov function to do this is:

$$V = \frac{1}{2}k_1 y^2 + \frac{1}{2}\theta^2 \tag{5.84}$$

which is radially unbounded and positive semi-definite. Its derivative is:

$$\dot{V} = k_1 y\dot{y} + \theta\dot{\theta} \tag{5.85}$$

$$= k_1 y v\sin(\theta) + \theta\omega \tag{5.86}$$

By choosing:

$$\omega^* = -\left(k_2\theta + k_1 v^*\frac{\sin(\theta)}{\theta}y\right) \tag{5.87}$$

where $k_1,\ k_2 > 0$, the derivative of the Lyapunov function is always negative and, by the Lyapunov stability theory combined with Barbalet's lemma, the system is asymptotically stable, with y and θ stabilizing to zero [Lee *et al.*, 1999].

Vehicle velocity v for this stage of control is chosen in an open-loop manner, based upon the initial orientation of the vehicle with respect to the goal. If the vehicle is facing

the goal, it is given a positive velocity, else it is given a negative velocity:

$$v^* = \begin{cases} k_3 & \text{if } \cos\psi_{initial} \geq 0 \\ -k_3 & \text{otherwise} \end{cases} \tag{5.88}$$

where $k_3 > 0$. Motion direction (i.e. forward or reverse) v^* can be switched at any time, depending on obstacle layout and requirements to minimise the space used by the vehicle in stabilizing to a pose.

A characteristic of the original controller presented by Lee *et al.* [1999] is that the selection of the initial direction can lead to a large motion away from the goal. Sensing limitations make this characteristic undesirable — in this case, if the vehicle moves too far away for the goal, landmarks will start to disappear and pose estimation will break down. This can be overcome with a set of logical rules that switch the velocity direction in the second stage of control, based upon current robot state. This is also useful if, for example, the velocity needs to be reversed due to the presence of obstacles.

The analysis has so far assumed perfect vehicle response (i.e. $v^* = v$ and $\omega^* = \omega$). Next, an analysis of the system with the inclusion of the system dynamics and saturation effects is presented.

Dynamic effects and gain tuning

The following state equations for the steering and velocity loops were derived in Section 5.2.1:

$$\begin{aligned} \dot{v} &= \frac{1}{\tau_v}(v^* - v) \\ \dot{\rho} &= \omega_n^2(\omega^* - \omega) \\ \dot{\omega} &= \rho - 2\zeta\omega_n\omega \end{aligned} \tag{5.89}$$

where, as in Section 5.2.1, the state equations for the steering loop are considered in the angular rate form given by combining Equations 5.12, 5.14 and 5.15. Parameters values are as experimentally derived in Section 3.4.

Linearising the system equations 5.83 about $\theta = 0$:

$$\begin{aligned} \dot{x} &= v \\ \dot{y} &= v\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{5.90}$$

and linearising the feedback law for ω^* (Equation 5.87) about the same point:

$$\begin{aligned} \omega^* &= -\left(k_2\theta + k_1 v^* \frac{\sin(\theta)}{\theta} y\right) \\ \omega^* &= -\left(k_2\theta + k_1 v^* y\right) \end{aligned} \tag{5.91}$$

where l'Hoptial's rule has been applied (i.e. for $\theta \to 0$, $\frac{\sin(\theta)}{\theta} \to 1$).

Inserting the linearised feedback law into the state equations (Equations 5.89 and 5.90) leads to:

$$\begin{aligned} \dot{x} &= v \\ \dot{y} &= v\theta \\ \dot{\theta} &= \omega \\ \dot{\rho} &= \omega_n^2 \left(-\left(k_2\theta + k_1 v^* y\right) - \omega\right) \\ \dot{\omega} &= \rho - 2\zeta\omega_n\omega \\ \dot{v} &= \frac{1}{\tau_v}\left(v^* - v\right) \end{aligned} \tag{5.92}$$

In these equations, there is multiplicative coupling between v and y. This is eliminated using the same approach as for Section 5.2.1.2. That is, v^* is set such that $v \simeq v^*$,

$$v_{rl}^* = v + \text{sat}\left(\frac{v^* - v}{\tau_2}, \frac{\dot{v}_{max}}{\tau_2}\right) \tag{5.93}$$

where v^* is as set by Equation 5.88.

Again, $\tau_2 = \tau_v$ and $\dot{v}_{max} = \tau_v^2$. Because the velocity loop dynamics are significantly faster than the steering loop dynamics, this change in the velocity control law allows us to now make the assumption that $v_{rl}^* \simeq v$. This simplifies the analysis and also further stabilizes the control. With the above assumption, the velocity loop dynamics can be ignored. The simulations and experiments will verify the validity of this assumption. The

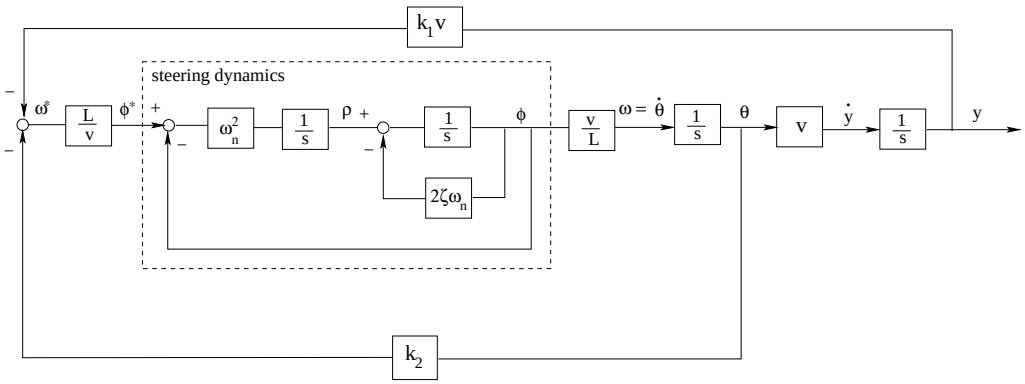


Figure 5.29: Block diagram for the y and θ loops in the second stage pose control.

closed-loop, linearised system equation becomes:

$$\begin{bmatrix} \dot{y} \\ \dot{\theta} \\ \dot{\omega} \\ \dot{\rho} \end{bmatrix} = \begin{bmatrix} 0 & v & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -2\zeta\omega_n & 1 \\ -k_1 v\omega_n^2 & -k_2\omega_n^2 & -\omega_n^2 & 0 \end{bmatrix} \begin{bmatrix} y \\ \theta \\ \omega \\ \rho \end{bmatrix} \tag{5.94}$$

The block diagram for the system is shown in Figure 5.29. In this system, there are two parameters which can be independently adjusted to alter the response of the system, k_1 and k_2.

First, the stability of the system is assessed with respect to these two gains. The characteristic equation is:

$$s^4 + 2\zeta\omega_n s^3 + \omega_n^2 s^2 + k_2\omega_n^2 s + k_1 v^2 w_n^2 \;=\; 0 \tag{5.95}$$

Rearranging for the k_2 gain allows the use MATLAB's `rlocus` tool to generate the root locus plots:

$$1 + k_2 \frac{\omega_n^2 s}{s^4 + 2\zeta\omega_n s^3 + \omega_n^2 s^2 + k_1 v^2 w_n^2} \;=\; 0 \tag{5.96}$$

Figure 5.30 illustrates the root locus for the case where k_2 is varied and k_1 is fixed at 0.01, 0.05, and 0.1. In this plot, each root locus for a fixed value of k_1 is shown. Increasing the k_1 gain results in an increasingly unstable system. Now, the stability of the system is

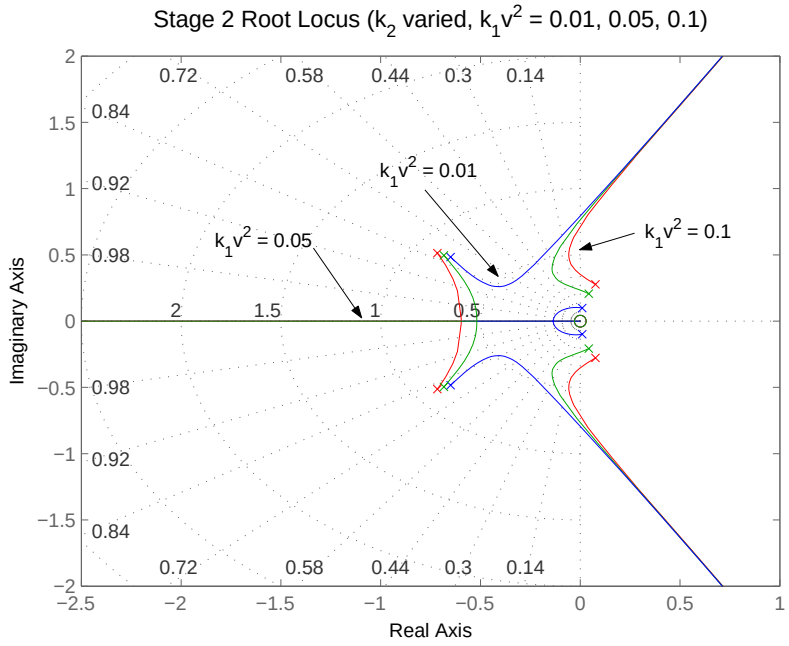


Figure 5.30: Root locus for second stage of position control, where k_2 is varied and $k_1 = 0.01,\ 0.05,\ 0.1$.

strongly influenced by the $k_1 v^2$ term in the characteristic equation:

$$1 + k_1 v^2 \frac{\omega_n^2}{s^4 + 2\zeta\omega_n s^3 + \omega_n^2 s^2 + k_2 w_n^2 s} \quad = \quad 0 \tag{5.97}$$

and the root locus for the case where k_1 is varied and k_2 is fixed at 0.1, 0.5 and 1.4 is shown in Figure 5.31. Increasing the k_2 gain beyond 1.4 leads to an inherently unstable system. Note that to keep the system poles at fixed locations, k_1 must be adjusted with the vehicle speed v.

Given an indication of the stability behaviour of the system, the system gains can be tuned. As for the design of the position controller in Section 5.2.1, the technique used here is to formulate the problem as a parameter optimisation problem. For this stage of control, the parameter search occurs in two dimensions, k_1 and k_2.

System behaviour is simulated using the closed-loop state equation (Equation 5.94) given some arbitrary initial state. Again, a cost function is created based upon the speed of convergence of the states y and θ, and the amount of overshoot in stabilizing these states, adjusting k_1 and k_2 to obtain the desired system behaviour. Here the cost function

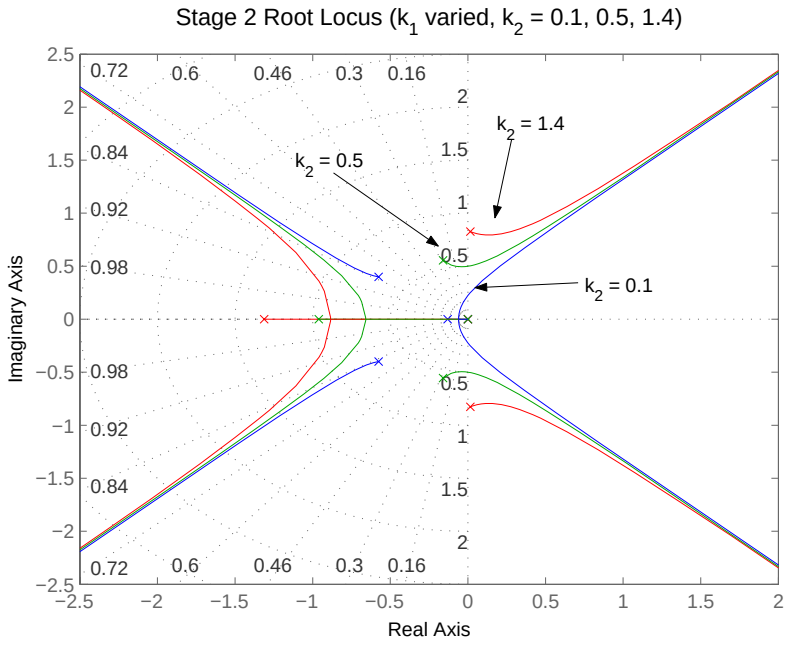


Figure 5.31: Root locus for second stage of pose control, where k_1 is varied and $k_2 = 0.1,\ 0.5,\ 1.4$.

is designed to ensure quick convergence of the states with minimal overshoot — the cost function used to achieve this is:

$$J(y,\theta) \quad = \quad w_1 J(y) + w_2 J(\theta) \tag{5.98}$$

where

$$J(y) \quad = \quad \int_{t=0}^{t=T} \frac{y(t)}{y(0)}^2 dt + \xi_1(\underline{y}) + \xi_2(\underline{y}) \tag{5.99}$$

and

$$J(\theta) \quad = \quad \int_{t=0}^{t=T} \frac{\theta(t)}{\theta(0)}^2 dt + \xi_1(\underline{\theta}) + \xi_2(\underline{\theta}) \tag{5.100}$$

where $y(t)$ and $\theta(t)$ are the state values at time t, $\underline{y}$ and $\underline{\theta}$ are the state evolution vectors throughout the simulation, and ξ_1 and ξ_2 are functions which calculate the overshoot and settling time for each state, as defined in Figure 5.5, and Equations 5.27 and 5.28. w_1 and w_2 are weighting factors, here set equal to 1 and 1 respectively. As for the position control design, for the function ξ_2, $\epsilon_y = 0.02$ and $\epsilon_\theta = 0.02$.

(y_0, θ_0)	$(k_1 v^2, k_2)$
$(5, 0.0)$	$(0.03980.3115)$
$(5, 0.1)$	$(0.0389, 0.3080)$
$(5, 0.2)$	$(0.0374, 0.3042)$
$(5, 0.3)$	$(0.0371, 0.3018)$
$(5, 0.5)$	$(0.0357, 0.2966)$
$(5, 1.0)$	$(0.03350.2875)$
$(1, 0.0)$	$(0.0399, 0.3115)$
$(1, 0.1)$	$(0.0357, 0.2966)$
$(1, 0.2)$	$(0.0335, 0.2875)$
$(1, 0.3)$	$(0.0323, 0.2815)$
$(1, 0.5)$	$(0.0603, 0.4916)$
$(1, 1.0)$	$(0.0617, 0.6226)$

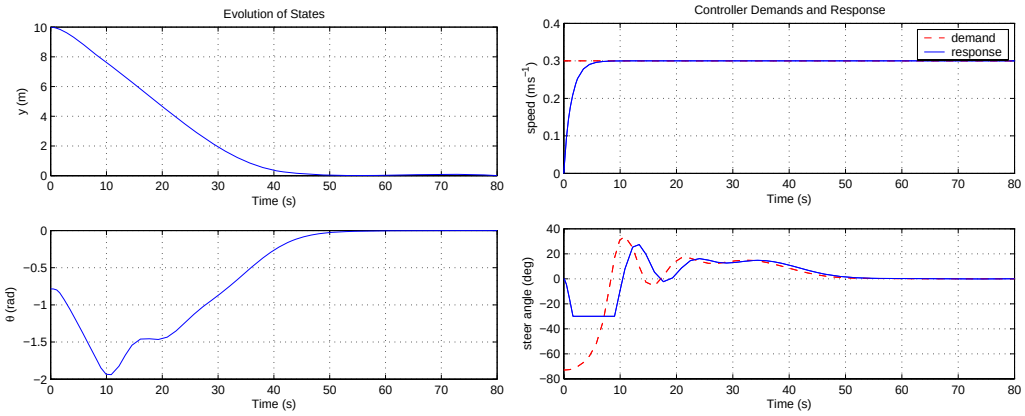
Table 5.3: Optimised Stage 2 pose control gains determined using `fminsearch` in MATLAB. Note that the initial values for the other states were set to $\omega = 0$ and $\rho = 0$ for all optimization runs.

Having defined the cost function, the `fminsearch` function in MATLAB is used to simulate the behaviour of the system, minimising the cost function by numerically searching the parameter space of the gains — the process terminates on minimisation of the cost function. Performing the optimization produces similar gains for a wide range of initial conditions of the state y and θ, a snapshot of which is provided in Table 5.2. From the table, it is noted that unlike for the cases considered in Section 5.2.1, the optimised parameters depend considerably on the initial conditions. This is probably due to the strongly non-linear nature of this particular system — recall that the system has been linearised about $\theta = 0$ for which $\dot{y} = v\theta$ rather than $\dot{y} = v \sin\theta$ for the nonlinear case. However, by choosing:

$$k_1 v^2 = 0.035 \tag{5.101}$$

$$k_2 = 0.3 \tag{5.102}$$

the desired response from the system can be elicited from a wide array of initial conditions, as the simulations and experiments will show. Note that k_1 must be adjusted with



(a) Evolution of y and θ.

(b) Control demands and vehicle response.

Figure 5.32: Response in Stage 2 of the pose control loop. Initial vehicle pose is $(x, y, \theta) = (-10, 10, \frac{\pi}{4}\,\mathrm{rad})$.

vehicle speed to keep the system poles at a fixed location. For the velocity law of Equation 5.88 $k_3 = 0.3$. This gain allows the vehicle to servo to the x-axis in a reasonable time and within a reasonable space — it was determined through an iterative process of observing the outcome of simulations for a variety of initial conditions.

In terms of system stability, the gain selection of $k_1 v^2 = 0.035$ and $k_2 = 0.3$ leads to the following roots for the characteristic equation of the system (Equation 5.95):

$$-0.6448, \quad -0.3345 \quad \text{and} \quad -0.1504 \pm 0.3274i$$

indicating the system poles are well-placed for this selection of gains.

The simulated response of the second stage of control on the full model of the vehicle (i.e. including *all* dynamics and non-linearities as described in Section 3.4), with the above tuned gains is shown in Figure 5.32. In this simulation, the initial orientation error is quite large (initial pose is $(x, y, \theta) = (-10, 10, \frac{\pi}{4})$) illustrating the robustness of the controller, even outside the 'linear' region of operation. Additionally, the state θ converges relatively quickly to zero and does not appear to be affected by the inclusion of the velocity loop.

Input saturation

As for the position controller presented in Section 5.2.1, saturation of the angular velocity ω can lead to stability problems and the associated non-convergence to the desired states. In this case, the problem occurs for large initial offsets from the x-axis, i.e. for large $|y|$. This instability results in the vehicle simply turning in a circle. In this case, the magnitude of the velocity demand is limited to k_3 by the control law of Equation 5.88, and hence:

$$|\omega_{max}| = \frac{k_3 \tan \phi_{max}}{L} \tag{5.103}$$

$$= 0.1443 \tag{5.104}$$

These problems can be observed and predicted by analysing Equation 5.86 for the cases were ω saturates, i.e.:

$$\begin{aligned} \dot{V} &= \dot{y(y)} + \theta\dot{\theta} \\ &= k_1 y v^* \sin(\theta) + \theta\omega \\ &= k_1 y v^* \sin(\theta) + \theta \text{sat}\left(\omega^*, \omega_{max}\right) \end{aligned} \tag{5.105}$$

A surface plot of the variation of $\dot{V}$ for the space defined in $y \in [-30, 30]$ and $\theta \in [-\pi, \pi]$ is shown in Figure 5.33. In this plot, v^* and ω^* have been calculated for each point using Equations 5.88 (assuming $\psi = 0$) and 5.87, and $\dot{V}$ is calculated using Equation 5.105. It can be seen that for many combinations of y and θ, $\dot{V}$ is positive meaning the system is unstable and, $y \not\to 0$ and $\theta \not\to 0$. In practice however, unless the initial value of the y state is very large (i.e. $|y_0| \gg 0$), the system seems to self-correct. *Because this mode of control is only used once the vehicle has homed to the goal region, the instability is of little significance to the overall switching control law*. Nevertheless, a means of dealing with the instability is presented.

By examining of Equation 5.105, with some rearranging and a substitution of the maximum available angular rate, ω_{max}, it can be determined that the instability arises under the following conditions:

$$y > \left| \frac{\theta \omega_{max}}{v^* k_1 \sin(\theta)} \right| \tag{5.106}$$

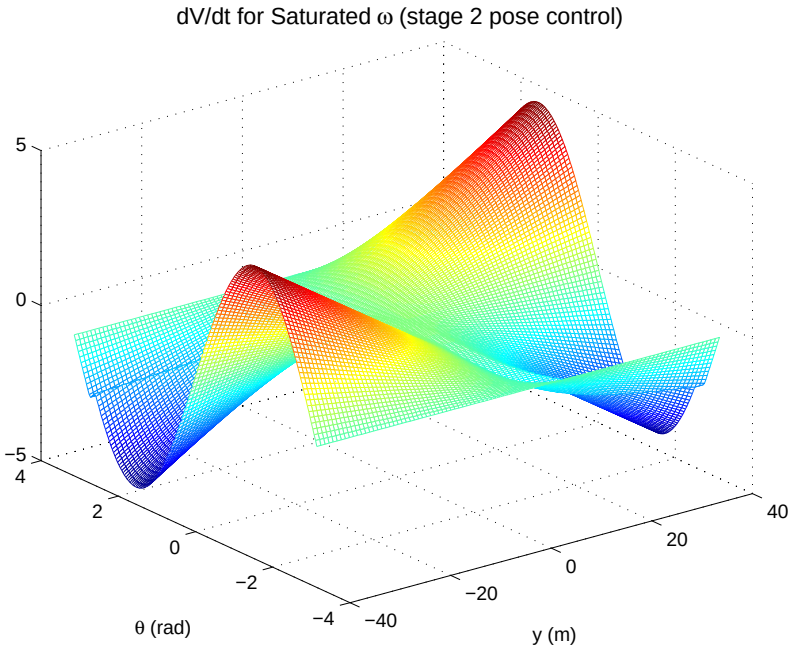


Figure 5.33: Behaviour of $\dot{V}$ for saturated angular velocity ω.

where $\omega_{max} = \left|\frac{v_{max}\tan(\phi_{max})}{L}\right|$, in which for this case, $|v_{max}| = k_3$ for this stage of control, and $|\phi_{max}| = 30^\circ$ as discussed in Section 3.4.

This instability is addressed by testing for the above condition and adjusting the value of k_1 in such a manner as to ensure that the instability condition is avoided. Optimally, this means heading towards the x- axis in the most direct manner possible, i.e. with an orientation of $\pm\frac{\pi}{2}$ depending on which side of the x-axis the vehicle is located. Re-examining the control law on angular rate:

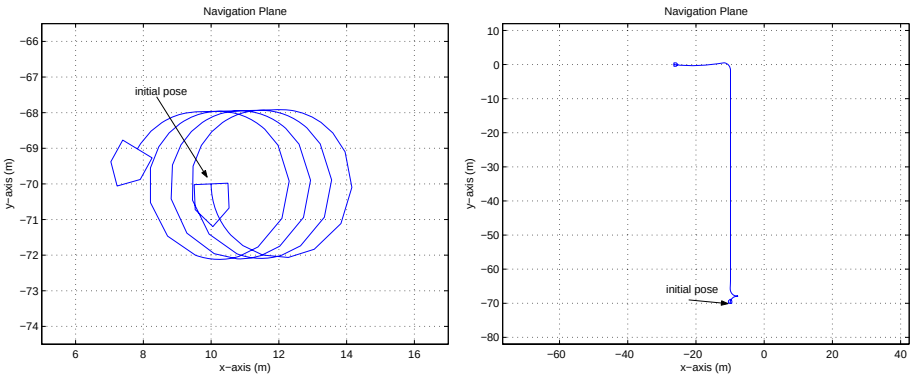
$$\omega^* = -\left(k_2\theta + k_1 v^* \frac{\sin(\theta)}{\theta} y\right)$$

the vehicle's heading can be driven to the desired value of $\pm\frac{\pi}{2}$ by setting:

$$k_1 = k_2 \frac{\frac{\pi}{2}\theta}{v^* y \sin(\theta)} \tag{5.107}$$

In addition, the velocity of the vehicle is set as:

$$v^* = -\text{sgn}(y\theta)k_3 \tag{5.108}$$



(a) Ground-plane motion of vehicle, no saturation logic.

(b) Ground-plane motion of vehicle, with saturation logic

Figure 5.34: Response in Stage 2 of the pose control loop where the angular velocity (steering angle) is saturated. Note the distance from the x-axis ($70\,\mathrm{m}$) means that saturation is of little practical significance. Initial vehicle pose is $(x, y, \theta) = (-10, 70, \frac{\pi}{2}\,\mathrm{rad})$ for both cases.

to ensure that the vehicle heads towards, rather than away from, the axis. Figures 5.8 (a) and (b) illustrate results for the case with *no saturation logic* leading to the vehicle simply driving in a circle around the initial position for certain starting positions, namely $|y| \gg 0$. Figures 5.8 (c) and (d) illustrate results from the same starting position with *saturation logic* which adjusts the value of k_1 leading to the vehicle heading directly for the x-axis. The starting pose for both examples is $(x, y, \theta) = (-1, 0, \frac{\pi}{4}\,\mathrm{rad})$ which ensures immediate saturation of the angular velocity ω. As noted earlier though, for the purposes of this work, this saturation condition has little significance because it occurs so far from the x-axis. The homing stage brings the vehicle to a location where the instability is insignificant and in any case, the sensing used in this work limits the vehicle's displacement from the target. Only one case is presented here, the Monte Carlo simulations presented in Section 5.2.2.1, with the full switching control law, will further show that this instability effect is insignificant.

5.3.2.3 Stage three control: servo-to-point

On completion of Stage two of control (i.e reaching the x-axis with an orientation of $\theta = 0$), Lee *et al.* [1999] proposed the following control laws to servo the vehicle to $x = 0$

$$v^* = -k_3 x \tag{5.109}$$

$$\phi^* = \omega^* = 0 \tag{5.110}$$

where $k_3 > 0$. Steering angle should not need to be controlled in this phase as the vehicle is already on the x-axis. However, in practice, it was found necessary to control the vehicle's steering angle because disturbances to the system, usually caused by sensing noise or error, can lead the vehicle astray. If the steering angle or the angular rate of the vehicle is not controlled, then there is no means to correct for these disturbances. Therefore, the control law for the angular rate of the vehicle as developed for the second stage of control, is used:

$$\omega^* = -\left(k_2\theta + k_1 v \frac{\sin(\theta)}{\theta} y\right) \tag{5.111}$$

where $k_1,\ k_2 > 0$,

Dynamic effects and gain selection

The preceding stages of control bring the vehicle into Stage 3 control for which the system dynamics are almost trivial. The analysis is presented for completeness. Repeating the velocity and steering loop equations from Section 5.2.1, the state equations for the steering and velocity loops are:

$$\begin{aligned} \dot{v} &= \frac{1}{\tau_v}\left(v^* - v\right) \\ \dot{\rho} &= \omega_n^2\left(\omega^* - \omega\right) \\ \dot{\omega} &= \rho - 2\zeta\omega_n\omega \end{aligned} \tag{5.112}$$

where, as in Section 5.2.1, the state equations for the steering loop are considered in the angular rate form given by combining Equations 5.12, 5.14 and 5.15. Parameters values are as experimentally derived in Section 3.4.

Again, linearising the system equations 5.83 about $\theta = 0$ gives:

$$\begin{aligned} \dot{x} &= v \\ \dot{y} &= v\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{5.113}$$

Linearising the feedback laws for v^* and ω^* (Equations 5.110 and 5.111) about the same point gives:

$$v = -k_3 x \tag{5.114}$$

$$\omega^* = -\left(k_2\theta + k_1 v y\right) \tag{5.115}$$

where l'Hoptial's rule has been applied as in Section 5.3.2.2.

Inserting the linearised feedback laws into the state equations (Equations 5.112 and 5.113) leads to:

$$\begin{aligned} \dot{x} &= v \\ \dot{y} &= v\theta \\ \dot{\theta} &= \omega \\ \dot{\rho} &= \omega_n^2\left(-\left(k_2\theta + k_1 v y\right) - \omega\right) \\ \dot{\omega} &= \rho - 2\zeta\omega_n\omega \\ \dot{v} &= \frac{1}{\tau_v}\left(-k_1 x - v\right) \end{aligned} \tag{5.116}$$

Again, there is multiplicative coupling between v and θ, which is eliminated using the same approach as for Section 5.2.1.2. That is, v^* is set such that $v \simeq v^*$,

$$v_{rl}^* = v + \text{sat}\left(\frac{v^* - v}{\tau_2}, \frac{\dot{v}_{max}}{\tau_2}\right) \tag{5.117}$$

where v^* is set by Equation 5.110.

Again, $\tau_2 = \tau_v$ and $\dot{v}_{max} = \tau_v^2$. Because the velocity loop dynamics are significantly faster than the steering loop dynamics, this change in the velocity control law allows us to now make the assumption that $v_{rl}^* \simeq v$. This simplifies the analysis and also further stabilizes the control. With the above assumption, the velocity loop dynamics can be ignored. The simulations and experiments will verify the validity of this assumption. The

closed-loop, linearised system equation then becomes:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\omega} \\ \dot{\rho} \end{bmatrix} = \begin{bmatrix} -k_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & v & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -2\zeta\omega_n & 1 \\ 0 & -k_1 v\omega_n^2 & -k_2\omega_n^2 & -\omega_n^2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \\ \omega \\ \rho \end{bmatrix} \tag{5.118}$$

The x dynamics have been made independent of the rest of the system. The gains for the steering loop are chosen using the same method as for Section 5.3.2.2, for which:

$$k_1 v^2 = 0.035 \tag{5.119}$$

$$k_2 = 0.3 \tag{5.120}$$

Tuning of the gain for the x loop is trivial, it is selected as:

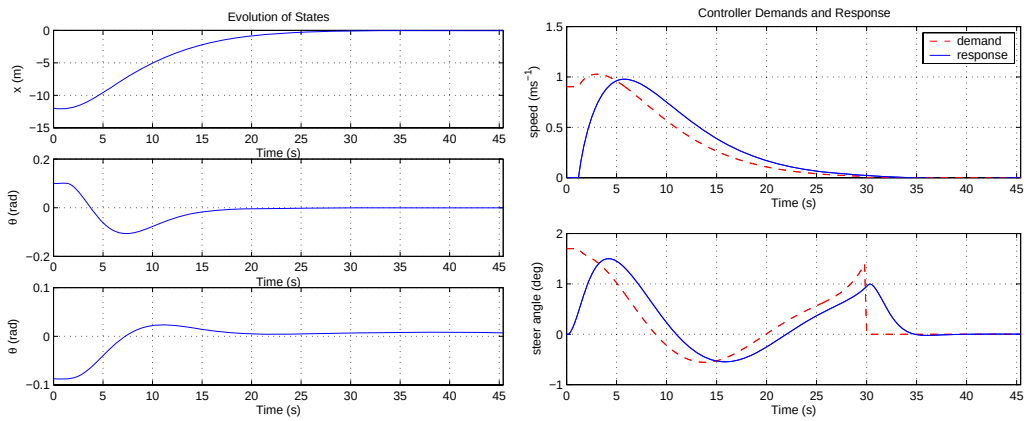
$$k_3 = 0.1 \tag{5.121}$$

The simulated response of the third stage of control on the full model of the vehicle (i.e. including *all* dynamics and non-linearities as described in Section 3.4), with the above tuned gains is shown in Figure 5.35. The aim of this stage of control is to stabilize to the origin, i.e. $x = 0$, given that prior stages of control have stabilized the vehicle to $y \simeq 0$ and $\theta \simeq 0$. In this simulation, the initial offset and orientation errors are modest (initial pose is $(x, y, \theta) = (-10, 0, \frac{\pi}{36})$) but the controller still manages to stabilize all three states. Note in the figure at $t = 30\,\mathrm{sec}$, the steering demand drops to 0 — this is due to the steering law being switched off when the vehicle has reached a position within a tolerance distance of the goal.

Input saturation for this stage of control is not a significant issue.

5.3.2.4 Control supervision

In combination, these control laws stabilize the vehicle to the desired pose from any initial condition. The design of the control-supervisor is now considered. As for the position controller of Section 5.2.1, the switching laws are straightforward but requires the



(a) Evolution of y and θ.

(b) Control demands and vehicle response.

Figure 5.35: Response in Stage 3 of the pose control loop. Initial vehicle pose is $(x, y, \theta) = (-10, 0, \frac{\pi}{36}\,\text{rad})$.

provision of a mechanism to *push* the control through the discontinuities introduced by switching between controllers — otherwise 'chattering' about the switching boundaries can occur. In addition, due to sensing inaccuracies and the asymptotic nature of these controllers, x, y, and θ will never reach zero in finite time — thresholds are required to switch between the controllers and to determine when the vehicle has reached the origin.

The control supervisor operates as described in Algorithm 4. In the algorithm, ϵ_y and ϵ_θ are the tolerances on y and θ respectively, η_{finish} is the tolerance on the time elapsed before the goal is considered to be reached, η_2 is the tolerance on the time elapsed before the `ControlState` variable is changed to 'Stage 2', and η_3 is the tolerance on the time elapsed before the `ControlState` variable is changed to 'Stage 3'. The variables `goalReachedTimer`, `stageOneTimer`, `stageTwoTimer` and `stageThreeTimer` record the time elapsed since detecting the particular state. The default state of the system is 'Stage 1'.

Recapping, the control laws for each stage are:

- Stage 1: homing

  The laws for this stage are as described in Section 5.2

Algorithm 4 Control supervision for pose controller.

{Initialise the state of the sytem}
`ControlState` $= Stage1$;`GoalReached` $= FALSE$; `EntereredStageTwo` $= FALSE$; `DirectionStageTwo` $= FALSE$
while (NOT`GoalReached`) **do**
  {First check if the goal has been reached}
  if ($|e| < \epsilon_e$) **then**
    if (`goalReachedTimer++` $> \eta_{finish}$) **then**
      `GoalReached` $= TRUE$
      $v^* = 0$; $\omega^* = 0$
    end if
    {if no, check if Stage 1 is required}
  else if ($e > r_e$) & (NOT`EntereredStageTwo`) **then**
    if (`stageOneTimer++` $> \eta_1$) **then**
      `ControlState` $= Stage1$; `stageTwoTimer` $= 0$; `stageThreeTimer` $= 0$; `goalReachedTimer` $= 0$
    end if
    {if no, check if Stage 3 is required}
  else if ($|y| < \epsilon_y$) & ($|\theta| < \epsilon_\theta$) **then**
    if (`stageThreeTimer++` $> \eta_3$) **then**
      `ControlState` $= Stage3$; `stageOneTimer` $= 0$; `stageTwoTimer` $= 0$; `goalReachedTimer` $= 0$
    end if
    {if no, Stage 1}
  else
    if (`stageTwoTimer++` $> \eta_2$) **then**
      `ControlState` $= Stage2$; `stageOneTimer` $= 0$; `stageThreeTimer` $= 0$; `goalReachedTimer` $= 0$
    end if
  end if
  {The rest of the loop implements the appropriate controller}
  .
  .
end while

- Stage 2: servo-to-line

  On first entering this stage, set `EnteredStageTwo` $= TRUE$, then set the open-loop velocity according to:

$$v^* = \begin{cases} k_3 & \text{if } \cos\psi_{initial} \geq 0 \\ -k_3 & \text{otherwise} \end{cases}$$

where $k_3 = 0.3$. Subsequently, if $e > r_e$ then set $v^* = -v^*$.

For the steering demand,

$$\omega^* = -\left(k_2\theta + k_1\frac{v^*\sin\theta}{\theta}\right)$$

where $k_1 v^2 = 0.035$ and $k_2 = 0.1715$.

- Stage 3: servo-to-point

$$\begin{aligned} v^* &= -k_3 x \\ \omega^* &= -\left(k_2\theta + k_1\frac{v^*\sin\theta}{\theta}\right) \end{aligned}$$

where $k_1 v^2 = 0.035$, $k_2 = 0.1715$ and $k_3 = 0.1$

Rate-limiting on the velocity, using Equation 5.43, is applied to all the velocity demands v^*.

5.3.3 Experiments

Simulation results are now presented, illustrating that the switching controller does indeed work over the entire workspace. In these experiments, the vehicle is started from different positions in the workspace, all with a common orientation. The vehicle is then started from a common position, with different initial orientations. The final simulation is a Monte Carlo simulation in which the controller is presented with 1000 random initial positions. In these, and all, simulations, the vehicle kinematics and dynamics are modelled as described in Section 3.4. A single landmark is used from which the IALV strategy is used to derive the vehicle's pose relative to the target pose, nominally the origin. By way of comparison, results from experiments on the real vehicle are then presented.

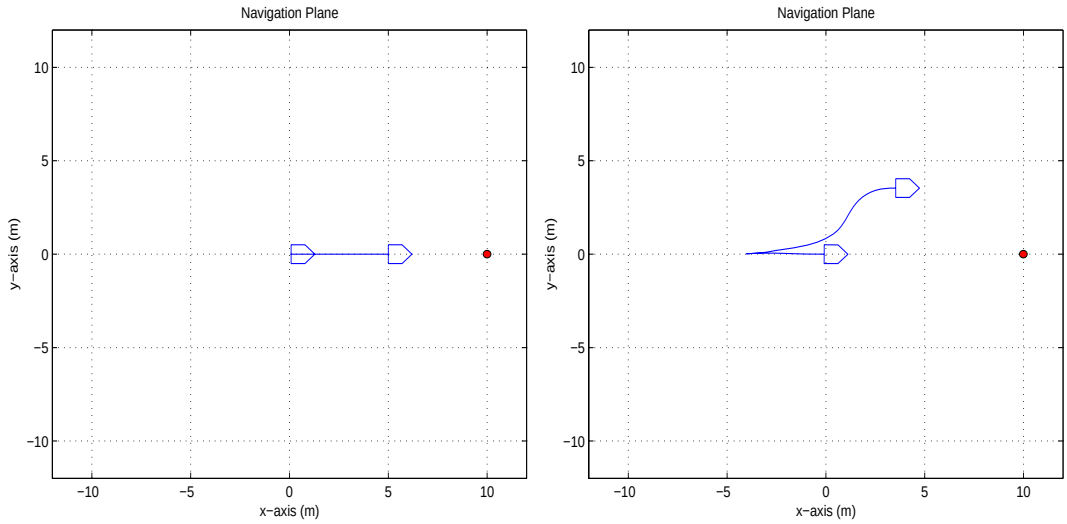
5.3.3.1 Simulations – covering the state space

In this first experiment, simulation results are obtained for the vehicle servoing to the origin from starting poses located on a circle of radius $5\,\mathrm{m}$ at angular intervals of $\frac{\pi}{4}\,\mathrm{radians}$, all with an initial vehicle orientation of $0\,\mathrm{radians}$. The simulated vehicle paths are plotted

in Figures 5.36 and 5.37. Note the uniformity of behaviour for mirrored starting positions.

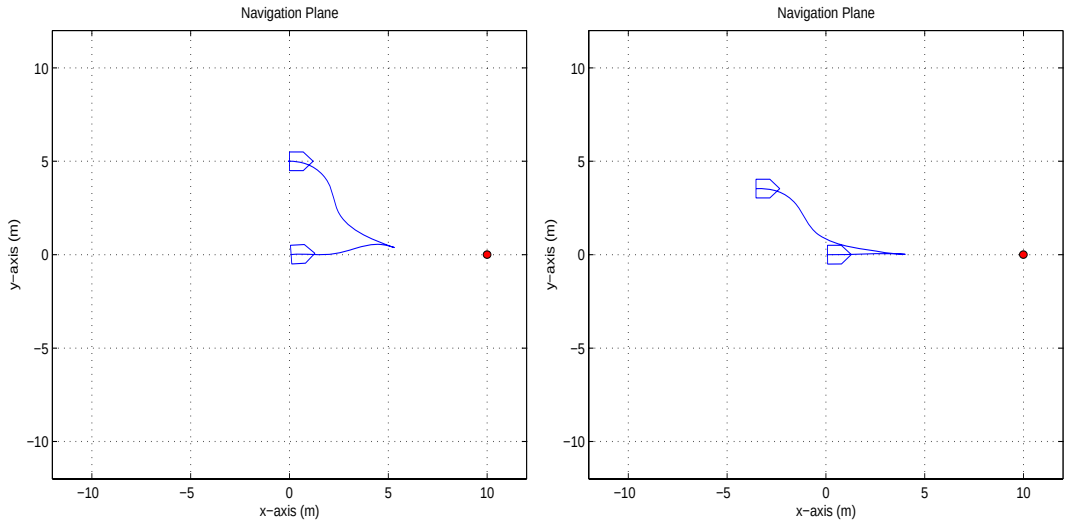
The next case to analyse is a constant starting position with a varying initial orientation, results from these simulation are shown in Figures 5.38 and 5.39. Again, these experiments show that the controller is robust to initial conditions, the only requirement being that the landmarks in the workspace are visible.

Figure 5.40 illustrates results from a Monte Carlo simulation in which pose control was simulated from 1000 random initial poses in the range $x, y \in [-12, 12]$ and $\theta \in (-\pi, \pi]$. Plotted are the resulting final vehicle positions for each of the random initial poses, Figure 5.40 (a), the final orientation 5.40 (b), and the time taken to stabilize to the target position for each simulation, Figure 5.40 (c). All experiments successfully brought the vehicle to within a tolerance of the goal pose, with the majority being within that tolerance specified by the controller $e < 0.1$ and $\theta < 0.1$. Note the difference between this and the corresponding plot for the position control experiments (Figure 5.17). In these experiments, the final position points are clustered on two lines at the distance tolerance for control completion (i.e. at $x = \pm 0.1$) — for the position controller, these points were distributed on a circle around the origin. Also note in this plot that some of the final positions lie outside the tolerance along the x-axis — this is probably caused by the rate-limiting on the velocity demand.



(a) Path generated from initial pose of $(x, y, \theta) = (-5, 0, 0)$.

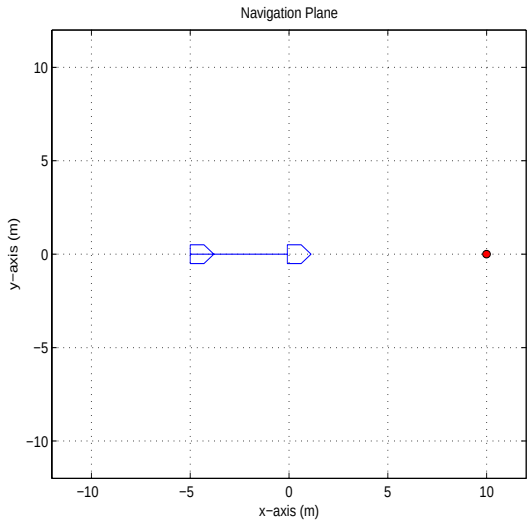
(b) Path generated from initial pose of $(x, y, \theta) = (-3.5356, 3.5356, 0)$.



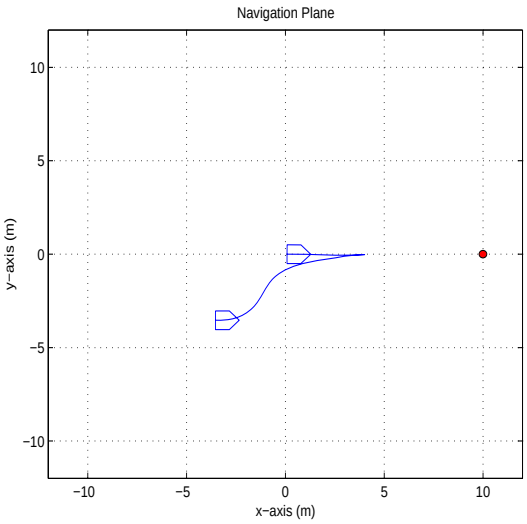
(c) Path generated from initial pose of $(x, y, \theta) = (0, 5, 0)$.

(d) Path generated from initial pose of $(x, y, \theta) = (3.5356, 3.5356, 0)$.

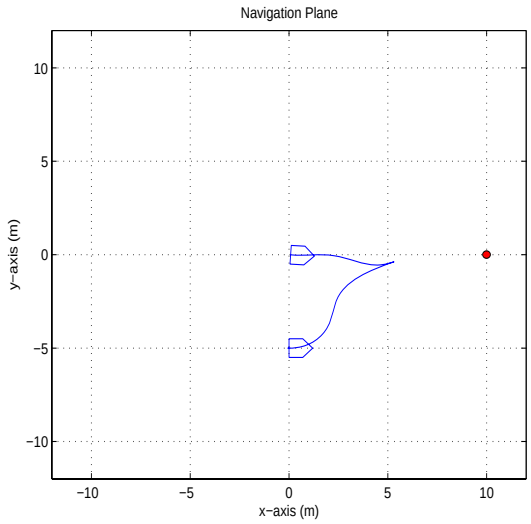
Figure 5.36: Effect of changing initial position on pose stabilization algorithm. Vehicle servoing to pose $(x, y, \theta) = (0, 0, 0)$ from the specified starting positions each with an initial orientation of $\theta = 0$.



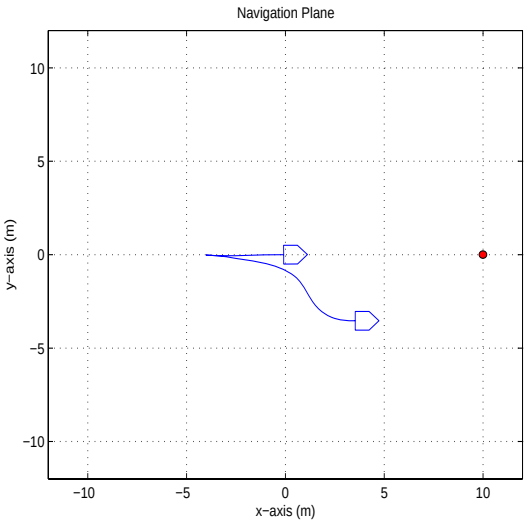
(a) Path generated from initial pose of $(x, y, \theta) = (-5, 0, 0)$.



(b) Path generated from initial pose of $(x, y, \theta) = (-3.5356, 3.5356, 0)$.

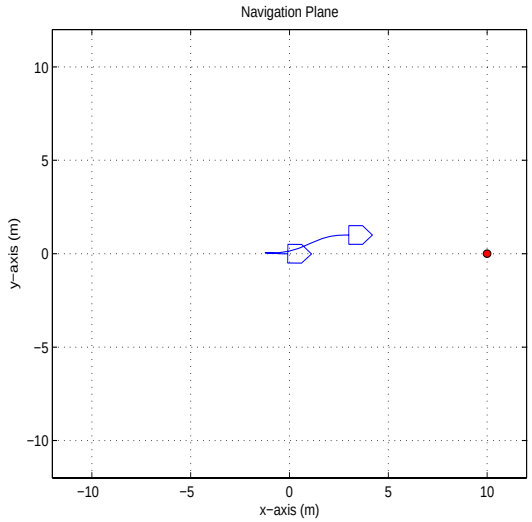


(c) Path generated from initial pose of $(x, y, \theta) = (0, 5, 0)$.

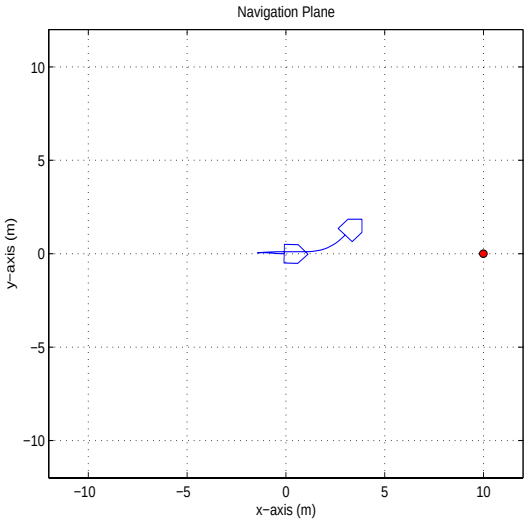


(d) Path generated from initial pose of $(x, y, \theta) = (3.5356, 3.5356, 0)$.

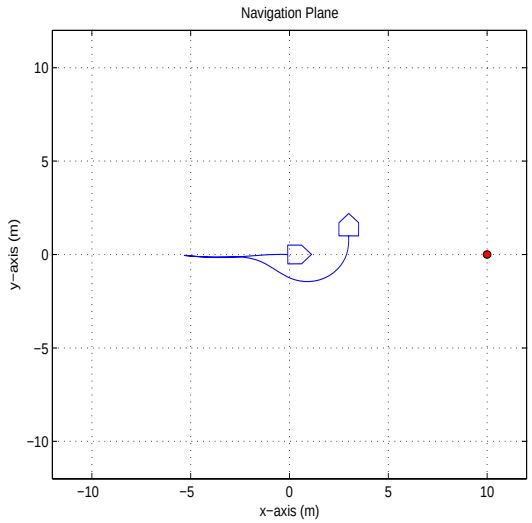
Figure 5.37: Effect of changing initial position on pose stabilization algorithm. Vehicle servoing to pose $(x, y, \theta) = (0, 0, 0)$ from the specified starting positions each with an initial orientation of $\theta = 0$.



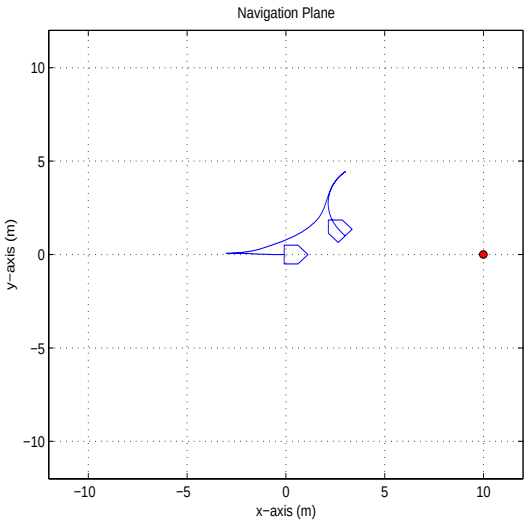
(a) Path generated from initial pose of $(x, y, \theta) = (3, 1, \pi)$.



(b) Path generated from initial pose of $(x, y, \theta) = (3, 1, -\frac{3\pi}{4})$.

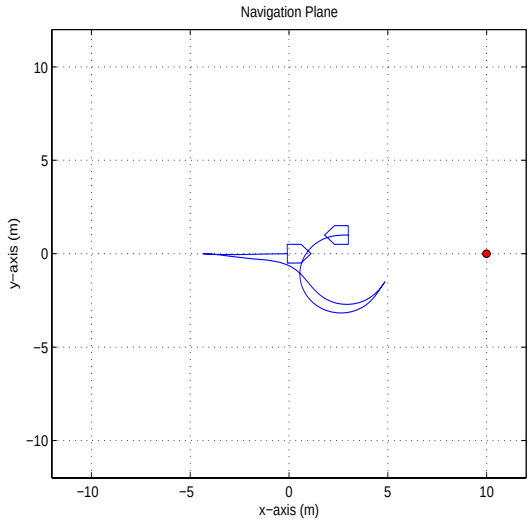


(c) Path generated from initial pose of $(x, y, \theta) = (3, 1, -\frac{\pi}{2})$.

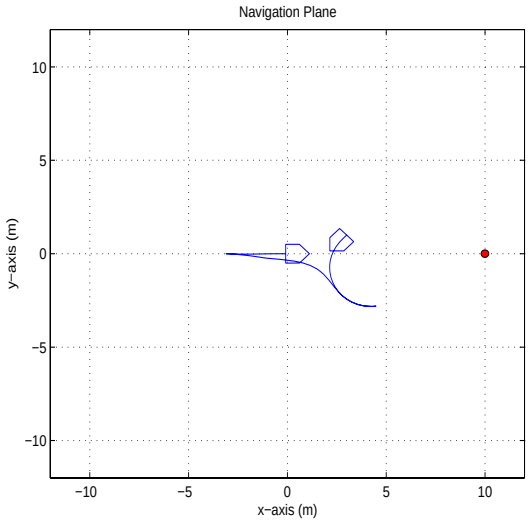


(d) Path generated from initial pose of $(x, y, \theta) = (3, 1, \frac{\pi}{4})$.

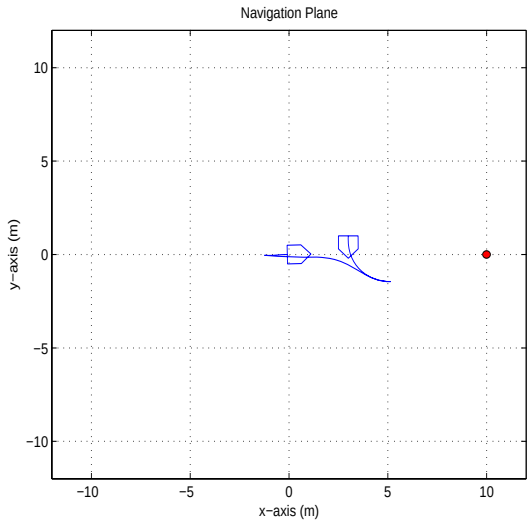
Figure 5.38: Effect of different starting orientations on pose stabilization. Vehicle servoing to pose $(x, y, \theta) = (0, 0, 0)$ from the specified starting poses.



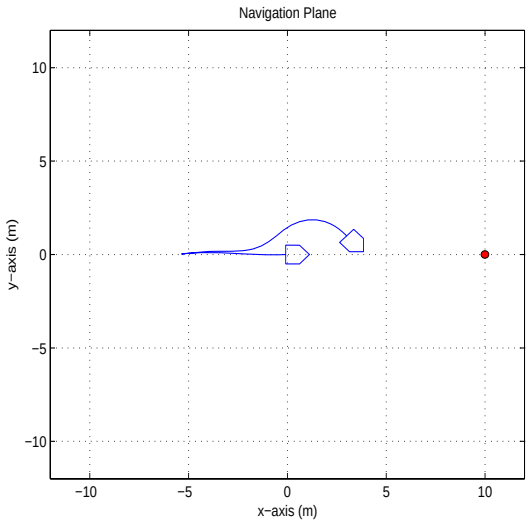
(a) Path generated from initial pose of $(x, y, \theta) = (3, 1, \pi)$.



(b) Path generated from initial pose of $(x, y, \theta) = (3, 1, -\frac{3\pi}{4})$.

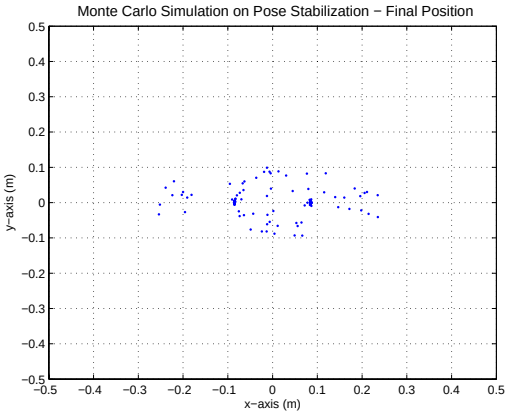


(c) Path generated from initial pose of $(x, y, \theta) = (3, 1, -\frac{\pi}{2})$.

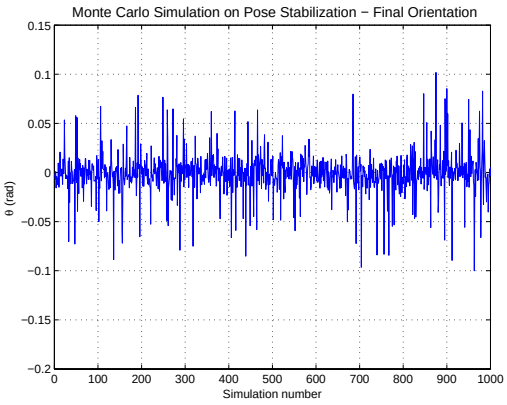


(d) Path generated from initial pose of $(x, y, \theta) = (3, 1, \frac{\pi}{4})$.

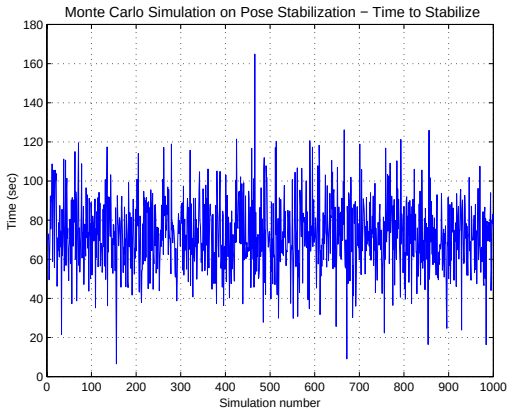
Figure 5.39: Effect of different starting orientations on pose stabilization. Vehicle servoing to pose $(x, y, \theta) = (0, 0, 0)$ from the specified starting poses.



(a) Final vehicle position (x, y).



(b) Final vehicle orientation (θ).



(c) Time to stabilize to pose.

Figure 5.40: Stabilization times from Monte Carlo simulation on position control. In this simulation, 1000 random initial poses were presented to the pose controller, all simulations brought the vehicle to within the specified tolerance of the goal, $e < 0.1$.

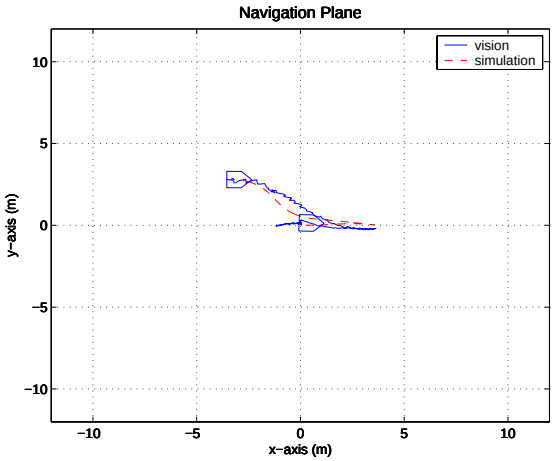
5.3.3.2 Experiments on the vehicle

In Section 5.2.1, the validity of the simulation model was demonstrated by comparing predicted motion with the vision and GPS data. In the experiments in this section, the motion of the vehicle as determined from the vision which is used for control, is compared with the predicted motion from the simulations.

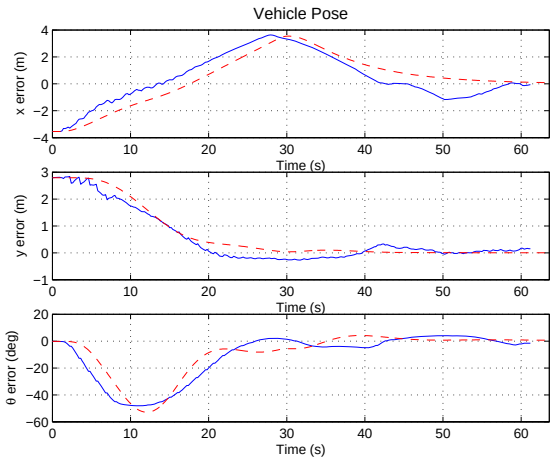
The first experiment starts the vehicle with an initial pose of $(x, y, \theta) = (-3.54, 2.79, 0)$, as determined by vision. A simulation was also conducted with the same starting pose. Figure 5.41 shows the results for the ground-plane motion of the vehicle, along with the state evolution, for the experiment and the simulation, and the vehicle demands and response. As with the position control experiments, there is good correlation between the simulated and real system. The states evolve as predicted by the simulation, and the experimental path of the vehicle is very close to that predicted by the simulation. In addition, the time to stabilize is very similar in the simulated and experimental runs. These results further validate the model of the vehicle.

However, as in Section 5.2.2.2 for the position control experiments, it is observed that the system demands and response are similar in *general* behaviour but there are some marked differences. These differences can again be accounted for by unmodelled effects such as stiction and friction of the speed pedal. Furthermore, recall that the pose determined by the IALV strategy (using vision and a magnetic compass) is subject to error due to the effects described in Section 5.2.2.2 — like the position control experiments, when the vehicle gets close to the origin, sensor noise can make the pose estimates rapidly change sign. In addition, sensor noise can lead to the vehicle entering a phase of control prematurely or incorrectly — such an effect is evident at $t \simeq 43\,\mathrm{sec}$. However, unlike the position control experiments, *systematic* sensor errors, such as the dependency on bearing for landmark range, cancel out.

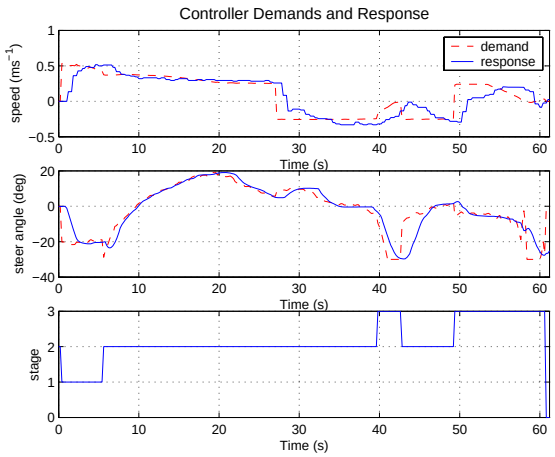
The next experiment investigates a more difficult manoeuvre, a three-point turn in which the vehicle's initial pose is close to the origin but it is facing the wrong way. The results are illustrated in Figure 5.42 where again the simulations and experimental results



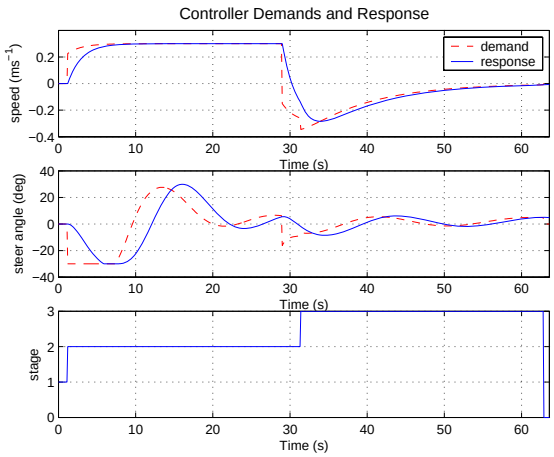
(a) Ground-plane path tracked.



(b) Evolution of states.



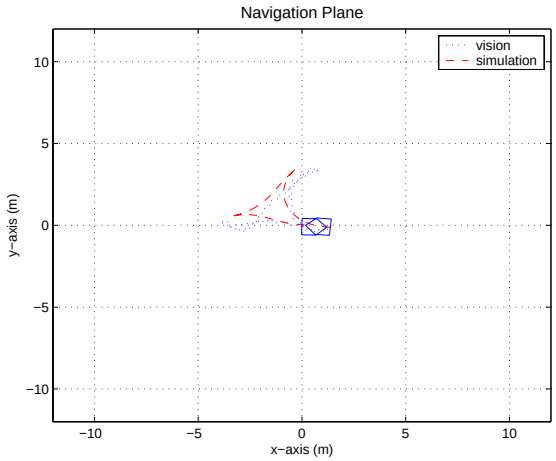
(c) Demands and response for the experimental system.



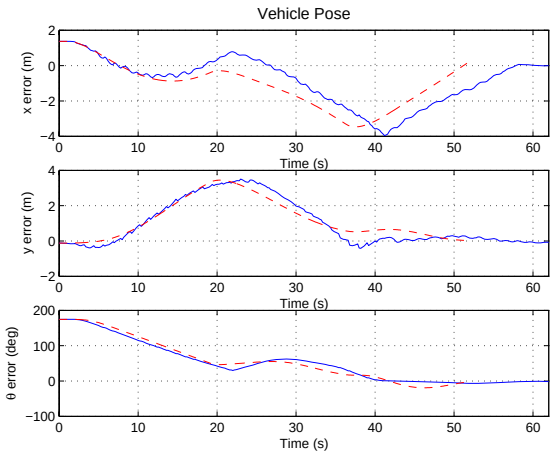
(d) Demands and response for the simulation.

Figure 5.41: Ground-plane path, state evolution and demands/responses for pose control, first experiment. Initial pose is $(x, y, \theta) = (-3.54, 2.79, 0)$. Goal pose is the origin, $(x, y, \theta) = (0, 0, 0)$.

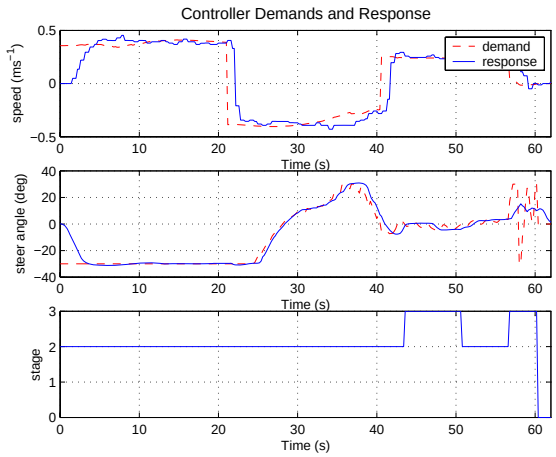
correlate extremely well. The experimental system took approximately $10\,\mathrm{sec}$ longer to stabilize — this is probably due to factors already discussed including sensing errors. At $t \simeq 50\,\mathrm{sec}$, the experimental system jumps from Stage 3 control to Stage 2. In contrast, the simulated system does not even enter Stage 3 control. This could explain the dif-



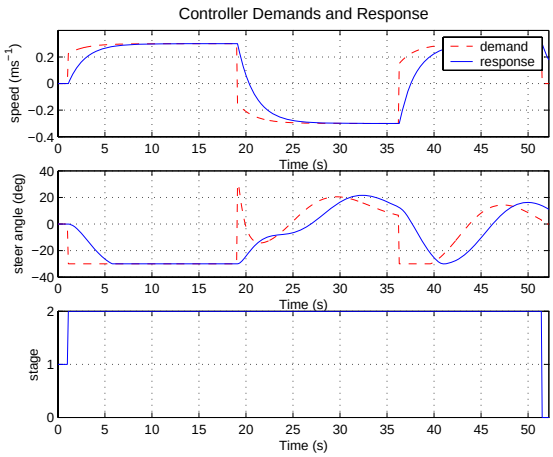
(a) Ground-plane path tracked.



(b) Evolution of states.



(c) Demands and response for the experimental system.



(d) Demands and response for the simulation.

Figure 5.42: Ground-plane path, state evolution and demands/responses for pose control, second experiment. Initial pose is $(x, y, \theta) = (1.37, -0.12, 3.05)$. Goal pose is the origin, $(x, y, \theta) = (0, 0, 0)$.

ference in stabilization times. In practice, this three-point-turn type manoeuvre can lead to landmarks 'disappearing' from the vehicle's view — under certain circumstances, the landmark can disappear for too long and the controller fails as the vision loses the landmark. However, given the difficulty of the manoeuvre, the performance of this vision-

based feedback controller is remarkable.

Further examples of pose control are provided in the supplementary material in Section A.1. These videos illustrate several pose stabilization runs in an outdoor environment for several different initial poses. In these videos, again note the changing lighting conditions and the relative robustness of the vision system.

5.3.4 Summary – Pose stabilization experiments

Again it has been shown that the vehicle modelling is representative of the behaviour of the system and accurately predicts the motion of the vehicle. In practice, sensing errors can lead to slightly different decisions being made in the experimental and simulated systems. However, the overall behaviour is, in general, well described by the modelling.

From the simulations and the experimental results, it can be inferred that the pose stabilization controller developed here is capable of driving the vehicle to a desired pose from *any* initial pose in the workspace. Controller performance is limited only by the capability of the sensing system.

5.4 Conclusions

In this chapter, controllers were developed enabling a car-like vehicle to stabilize to a *position* and a *pose* using state feedback. Both controllers were based upon the use of switching logic to sequentially stabilize the vehicle to sub-manifolds of the desired states. Both controllers were designed with the limitations of real vehicles in mind, namely steering and velocity loop dynamics, and input saturation.

The validity of these controllers was demonstrated through extensive simulations. The simulations demonstrate that the controllers are stable over the entire state-space, supporting the presented theory. Results from experiments on the vehicle were also presented. These latter results demonstrate two things:

1. The modelling of the vehicle is representative of the behaviour of the system;

2. The tasks can be achieved using the IALV strategy with sensing provided by omnidirectional vision and a magnetic compass.

Vision-based feedback pose stabilization for a nonholonomic car-like vehicle with significant dynamics in the steering and velocity loops was demonstrated in this chapter, representing the major contribution of the thesis.

Chapter 6

Conclusions

The principal topic addressed in this thesis is:

> *Can omnidirectional visual feedback be used as the primary sensor for the pose stabilization of a nonholonomic, car-like vehicle?*

In short, the answer to this question is *yes*. Within the limits of the sensor, omnidirectional visual feedback strategies can be used to stabilize a car-like vehicle with significant steering and velocity loop dynamics to a pre-learnt pose, based upon the discrepancies between the view from an arbitrary location and that seen at the target pose. To cope with dynamics and input saturation, several different stages of control are required:

- Initially the vehicle should stabilize to the line which has the target orientation and contains the target point.
- On stabilizing to the line, servo to the target point.

A discrete event supervisor switches between the different stages of control, based upon the state of the vehicle, with state-machines used to overcome chattering problems, pushing the control through the inherent discontinuities. This control strategy overcomes the dynamics and limitations of 'real', car-like systems by choosing subsets of the degrees of freedom in the problem.

For providing the visual feedback, visual homing strategies were analysed and extended to allow full pose estimation for the car-like vehicle, based upon feedback from

an omnidirectional camera and a compass, using the differences between the currently sensed scene and that at a pre-learnt target pose. These pose estimates were then fed into the switching controller, allowing the vehicle to stabilize to the pre-learnt pose using omnidirectional vision. However, although the control method is stable, when combined with extremely noisy visual pose estimates, chattering-type problems can appear in which the vehicle oscillates about a switching boundary. In this case, the presented feedback pose stabilization method will fail.

6.1 Research questions

Several subproblems to the principal topic of investigation were highlighted in the introduction to this thesis (Chapter 1). These research questions are now re-examined:

1. *What are the relative strengths and weaknesses of the visual homing techniques and how can they be adapted for a car-like vehicle?* Most existing pose stabilization strategies require full vehicle pose estimates and it is usually assumed in the literature that these are readily available. Visual homing strategies provide a homing vector which drives the agent towards some pre-learnt location, based upon the discrepancies between the current view and a remembered view of the target location. This homing vector is a scaled representation of the robot's position with respect to the target position, with different homing methods providing varying qualities of homing vector 'accuracy'. Chapter 4 compared two prominent bearing only homing techniques, the Weighted Vector method, and the Average Landmark Vector method. The superiority of the ALV method for providing a homing vector under varying landmark configurations was demonstrated and it was highlighted that the ALV method does not require correspondence between landmarks at the current and target views. A contribution of this thesis was the introduction of landmark range information to the ALV method — the Improved Average Landmark Vector method — which provides the distance and orientation to the target location exactly. When combined with the compass sense required for homing vector calculation, a full

vehicle pose estimate, with respect to the target pose, is provided.

2. *Can current vision techniques (feature extraction, blob tracking, feature correspondence etc.) be adapted and developed for an omnidirectional camera?* A catodioptric omnidirectional camera provides a panoramic view of the environment. A common paradigm in omnidirectional vision is to 'unwrap' the view to one more easily interpreted by humans but this is an unnecessary waste of processing, as the form of the image is unimportant to the computer interpreting the image. Objects in omnidirectional images appear warped and change shape with viewpoint much more dramatically than with conventional images. Colour-based segmentation provides a means of identifying and localising objects in an image without reference to object shape. Additional recognition cues can then be introduced such as object size (the amount of space it consumes in the image). Using the geometry of the system, and a flat-Earth assumption, object range can readily be estimated, based on distance from the image centre. More sophisticated means of range estimation, for example optic flow type techniques, were not possible with the coarse on-board velocity measurements of this particular experimental system.

3. *Can the vehicle's motion model be used to assist omnidirectional camera feature tracking?* Once localised in the image, object tracking based on vehicle-object relative motion models is then used to predict future object positions. This can improve object recognition, eliminating objects falsely detected based on their colour alone, effectively providing *temporal filtering*. Further to this, *complementary filtering* techniques can be used to combine data from the vehicle-object motion models based on odometry with the measurements from the vision system, vastly improving the quality of object range and bearing measurements.

4. *Can the vehicle be stabilized to a position with no constraint on its final orientation, in the spirit of the visual homing literature* Yes, using a two-stage switching control law combined with the sensing strategies developed in previous chapters,

it is possible to stabilize a car-like vehicle to a position. However, the success of this process is limited by errors in the sensing induced by incorrect perpendicular alignment between the camera-mirror assembly and the ground-plane — this leads to a bearing dependent range estimate which in turn affects the pose estimate. In contrast, when servoing to a pose, these errors cancel out and pose stabilization is more precise than position stabilization, despite the increase in complexity of the controller

5. *Which of the pose stabilization techniques can accommodate 'real world' vehicle kinematics, constraints, and dynamics?* Pose stabilization methods were tested via two means: implementation on the CMIT Autonomous Tractor, which was specifically developed through the work of this thesis, and through a detailed simulation model consisting of the AT's kinematics, and experimentally determined models of the dynamics of the steering and velocity loops and the associated non-linearities such as input saturation. These were used to test several pose stabilization strategies, illustrating that many existing feedback strategies cannot cope with the characteristics of 'real' vehicles. Significantly, many failed on saturation of the steering demand — real car-like vehicles have significant limitations on the available steering angle. A strategy that does deal with these effects is to split the state-space into sub-manifolds, sequentially servoing to specifically chosen states in each sub-manifold. For the car-like vehicle case the strategy is:

   - Servo to the line which has the target orientation and contains the desired point.
   - On stabilizing to the line, servo to the desired point.

   This strategy can accommodate dynamics of the system and overcomes saturation of the steering demand. It also allows the use of linear control tools for system analysis. Servoing to a line has a reduced degree of freedom when compared to servoing to a pose. Once on the line, acquiring the goal position is almost trivial,

if good pose estimates are available. Despite the extensive work on improving the pose estimate, measurements of the vehicle state were still quite noisy. Strategies to overcome this noise included the use of timers to *push* through the switching boundaries which together with a discrete event supervisor, decide the state of the system based upon the frequency of a particular combination of states at a particular time.

6.2 Contributions of the thesis

In answering the above questions, the thesis provided a comprehensive background in mobile robotics, reviewed the literature on pose stabilization and vision-based sensing, created a suitable test-platform with the associated kinematic and dynamic models of the platform, and demonstrated pose stabilization in simulation and through extensive experimentation.

In addressing the main research questions, the following key contributions were made:

- **Background material on navigation** was presented, highlighting the slow shift away from complete metric representations of the environment to simpler, more manageable representations in the form of *topological maps*. A review of the existing literature on the navigation problem in general was provided, based upon the taxonomy presented by Franz and Mallot [2000].

- **Reviewed the pose stabilization literature** based upon the taxonomy of Kolmanovsky and McClamroch [1995], highlighting the shortage of 'real' experimental results, particularly for car-like vehicles.

- **Reviewed the use of computer vision** to solve the problem of both position and pose stabilization, including a review of the visual homing literature.

- **Developed a test platform** representative of an industrial vehicle — the CMIT Autonomous Tractor. The AT is an ideal platform for testing navigation and control algorithms and has a role for much future land navigation research.

- **Developed accurate models** of the AT using the well-known bicycle model for the vehicle kinematics and experimentally determined models of the vehicle's response to control demands, including velocity and steering loop dynamics and non-linear effects such as input saturation. In terms of testing pose control algorithms, few, if any, authors use realistic vehicle models.

- **Developed a simple, yet robust, pose estimation technique** using omnidirectional computer vision and a compass. The camera, which mimics an insects wide field of view, was used to track a set of homogeneous, coloured landmarks, from which the pose of the vehicle could be estimated. This strategy is called the Improved Average Landmark Vector method, a derivative of a bearing-only strategy, enriched with range information. It allows the estimation of the vehicle's pose relative to some pre-learnt target pose, using the discrepancies between the current view and the view at target location. This method is free of the the need for *landmark correspondence* and does not require image unwrapping. Another contribution in this area was the characterisation of the sensitivity to sensing errors.

- **Developed a robust vision system** to implement the homing/pose stabilization algorithms. The vision uses a colour-based object segmentation techniques based upon a pre-taught two-dimensional look-up table of the target object colour. Range estimation techniques were developed based on the flat-Earth assumption and the equiangular mirror optics. A novel tracking method was presented based upon vehicle-object motion models.

- **Developed complementary filtering** techniques which combine data from disparate sensing sources, producing higher quality measurements. In this case, vehicle odometry and vehicle-object motion models were combined with the range and bearing estimates from the the vision system. Also, vehicle odometry and a kinematic model was combined with a compass sensor.

- **Investigated and implemented position stabilization strategies** through simula-

tion and experimentation.

- **Investigated and implemented pose stabilization strategies** through simulation and experimentation. A workable pose stabilization technique was implemented on the AT, with further developments including fixing a stability problem with an existing discontinuous controller, and providing techniques to reduce control 'chatter'.

- **Demonstrated that pose stabilization is easier than position stabilization** with the sensing arrangement used in this thesis. This is due to the fact that the world looks different when viewed at the same location with different orientations — it is near impossible to perfectly align an omnidirectional camera with the ground-plane, and even if this is achieved, lens / mirror aberrations could still produce distortions. On stabilizing to a pose, the target location is approached with a similar orientation to the target view, and hence any distortions in the view are nullified. In contrast, This is not the case when stabilizing to a position.

6.3 Closing remarks

Although the vision-based pose stabilization technique presented in this thesis was successful, it was somewhat limited by the sensing which required the vehicle to travel relatively slowly (still much faster than comparable work). With the ever increasing availability of computing power, the open-loop strategies for pose stabilization are becoming more attractive when used in an on-line feedback type manner, more reliably providing for additional constraints such as obstacle avoidance. One of the reasons these on-line, open-loop approaches were not used in this work was that all processing was to occur on-board the AT, a difficult task when also running computer vision for pose estimation. An interesting direction of future research would be to investigate, optimise and implement the on-line open-loop strategies, perhaps initially using a less computationally burdensome pose estimation sensor.

Pose stabilization is a highly *local* behaviour but a necessity if we are to see autonomous industrial vehicles 'do' anything in a workspace. However, it is but one of the suite of required competencies for performing *global* tasks. In order to perform global tasks, autonomous vehicles need to be able to recognise different locations, stored in some internal representation from which coherent plans of action can be formulated. The current paradigm is to provide, or have the robot build, complete metric maps. This approach is limited in scale due to the high computational burdens of maintaining metric representations of large workspaces — this is particularly true of outdoor mobile robot systems. It is also wasteful, as much of the workspace is not 'interesting' to a mobile robot. A more parsimonious representation is the topological map which consists of a series of distinctive places, connected by control instructions, rather than through complete metric mapping. Of course, if required, local metric maps can be integrated into topological representations. Key to the use of topological maps is the ability to recognise distinctive places and to traverse the spaces between them. Another key aspect is the human/robot interface: How do we communicate, in a flexible manner, what we wish the robot to do?

The investigation of these additional competencies, which would allow for more *global* behaviours, provides further research opportunities, especially in the context of topological mapping. Vision, and in particular omnidirectional vision, provides a promising avenue of research for recognising different locations through matching of colour histogram representations of pre-learnt distinctive places. Edge following (e.g. a road) is a technique that could be used to connect such distinctive places. These techniques are well developed using monocular cameras and range sensors but the use of omnidirectional vision for edge following, particularly in outdoor environments [Corke *et al.*, 2003], is another promising avenue of research. The work in this thesis focused on the use of 'artificial' colour landmarks but the idea of using natural landmarks and fusing different visual features including edges, colour, and optic flow type measurements is also a rich source of future work. However, an 'image' is not restricted to computer vision and could be acquired from, or fused with, other sources such as a scanning laser or radar.

The methods and techniques developed in this thesis are equally applicable to other robotic platforms. Of particular interest would be stabilizing a flying and/or submersible vehicle to a specific target pose based upon a pre-learnt view of the target location using principles similar to those used in this thesis. These vehicles are also nonholonomic systems, with additional degrees of freedom and sources of disturbances. An additional difficulty is that flying and swimming vehicles are not *driftless*[1] systems, as for the car-like vehicle investigated in this thesis.

[1] On a flat plane, with no control inputs a car-like vehicle remains in equilibrium. This is not the case for a flying vehicle which, in general, will fall to the ground if no control inputs are provided.

Appendix A

Supplementary Material

A.1 Video Material

The attached CD-ROM contains video footage of the Autonomous Tractor. The CD-ROM is organized as detailed in Table A.1.

Directory and Title	Brief Description
chapter3/first_remote_run.mov	The AT's first run under remote control
chapter3/first_remote_run_short.mov	A shorter version of the above video
chapter4/lm_tracking_sequence.mov	A short sequence showing tracking of a set of witches hats
chapter5/position_example_1.mov	A position stablization run (example 1)
chapter5/position_example_2.mov	A position stablization run (example 2)
chapter5/pose_example_1.mov	A pose stablization run (example 1)
chapter5/pose_example_2.mov	A pose stablization run (example 2)
chapter5/early_pose_stabilization_run.mov	A pose stabilization run from early in the thesis work

Table A.1: Table of contents for the supplementary material on the CD-ROM.

Bibliography

[Adam *et al.*, 1999] Amit Adam, Ehud Rivlin, and Héctor Rotstein. Fusion of fixation and odometry for vehicle navigation. In *International Conference on Robotics and Automation*, pages 1638–1643, Detroit, Michigan, 1999. IEEE.

[Aguiar *et al.*, 2000] António Pedro Aguiar, A. Nazir Atassi, and António Pascoal. Stabilization of a nonholonomic dynamic wheeled mobile robot with parametric modelling uncertainty using Lyapunov functions. In *Proceedings of CONTROLLO'2000 - 4th Portuguese Conference on Automatic Control*, Guimares, Portugal, October 2000.

[Aicardi *et al.*, 1994] Michele Aicardi, Giuseppe Casalino, Aldo Balestrino, and Antonio Bicchi. Closed loop smooth steering of unicycle-like vehicles. In *Proceedings of the 33rd Conference on Decision and Control*, pages 2455–2458, Lake Buena Vista, Florida, USA, December 1994.

[Aicardi *et al.*, 1995] Michele Aicardi, Giuseppe Casalino, Antonio Bicchi, and Aldo Balestrino. Closed loop steering of unicycle-like vehicles via Lyapunov techniques. *IEEE Robotics and Automation Magazine*, pages 27–35, March 1995.

[Aono *et al.*, 1998] Toshihiro Aono, Kenjiro Fujii, Shintaro Hatsumoto, and Takayuki Kamiya. Positioning of a vehicle on undulating ground using GPS and dead reckoning. In *International Conference on Robotics and Automation*, pages 3443–3448, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Argyros *et al.*, 2001] Antonis A. Argyros, Kostas E. Bekris, and Stelios C. Orphanoudakis. Robot homing based on corner tracking in a sequence of panoramic

images. In *Computer Vision and Pattern Recognition (CVPR 2001)*, Hawii, USA, December 2001.

[Arkin, 1987] Ronald C. Arkin. Motor schema based navigation for a mobile robot: An approach to programming by behavior. In *IEEE Conference on Robotics and Automation*, pages 264–271, 1987.

[Arkin, 1989a] Ronald C. Arkin. Motor schema based mobile robot navigation. *International Journal of Robotics Research*, pages 92–112, August 1989.

[Arkin, 1989b] Ronald C. Arkin. Towards the unification of navigational planning and reactive control. Working notes of the AAAI Spring Symposium on Robot Navigation, March 1989. Stanford University.

[Arkin, 1995] Ronald C. Arkin. Reactive robotic systems. In M. Arbib, editor, *Handbook of Brain Theory and Neural Networks*, pages 793–796. MIT Press, 1995.

[Arkin, 1999] Ronald C. Arkin. *Behaviour-based Robotics*. MIT Press, 1999.

[Astolfi, 1995] A. Astolfi. Exponential stabilization of a car-like vehicle. In *Proceedings of the International Conference on Robotics and Automation*, pages 1391–1396, Nagoya, Japan, 1995. IEEE.

[Astolfi, 1996] Alessandro Astolfi. *Asymptotic stabilization of nonholonomic systems with discontinuous control*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 1996.

[Badreddin and Mansour, 1993] E. Badreddin and M. Mansour. Fuzzy-tuned state-feedback control of a non-holonomic mobile robot. In *Proceedings of IFAC World Conference*, pages 577–580, Sydney, Australia, 1993.

[Bailey and Nebot, 2001] Tim Bailey and Eduardo Nebot. Localisation in large scale environments. *Robotics and Autonomous Systems*, 37(4):261–281, December 2001.

[Bailey, 2002] Tim Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, ACFR, Department of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, August 2002.

[Bailey, 2003] Tim Bailey. Constrained initialisation for bearing-only SLAM. In *International Conference on Robotics and Automation*, Taipai, Taiwan, May 2003. IEEE.

[Baker and Nayer, 1999] Simon Baker and Shree Nayer. A theory of single-viewpoint catadioptric image formation. *International Journal of Computer Vision*, 35(2):1–22, 1999.

[Barraquand and Latombe, 1989] Jérome Barraquand and Jean-Claude Latombe. On nonholonomic mobile robots and optimal manuevering. In *Proceedings of the intelligent control workshop*, pages 340–347, Albany, NY, USA, September 1989.

[Batlle *et al.*, 2000] J. Batlle, A. Casals, J. Freixenet, and J. Martí. A review on strategies for recognizing objects in colour images of outdoor scenes. *Image and Vision Computing*, 18:515–530, 2000.

[Betge-Brezetz *et al.*, 1996] S. Betge-Brezetz, P. Hebert, and R. Chatilla. Uncertain map making in natural environments. In *Proceedings of the International Conference on Robotics and Automation*, pages 1048–1053, Minneapolis, Minnesota, USA, April 1996. IEEE.

[Bianco and Zelinsky, 1999] Giovanni Bianco and Alexander Zelinsky. Biologically-inspired visual landmark learning and navigation for mobile robots. In *International Conference on Intelligent Robotics and Systems*, Kyongjo, Korea, October 1999. RSJ/IEEE.

[Bloch and McClamroch, 1989] A. M. Bloch and N. H. McClamroch. Control of mechanical systems with classical nonholonomic constraints. In *Proceedings of the 28th IEEE Conference on Decision and Control*, pages 201–205, 1989.

[Bloch *et al.*, 1990] A. M. Bloch, N. H. McClamroch, and M. Reyhanoglu. Controlability and stabilizability properties of nonholonomic control systems. In *Proceedings of the 29th IEEE Conference on Decision and Control*, pages 1312–1314, Honolulu, Hawaii, USA, 1990. IEEE.

[Bloch *et al.*, 1992] A. Bloch, M. Reyhanoglu, and N. H. McClamroch. Control and stabilization of nonholonomic dynamic systems. *IEEE Transactions on Automatic Control*, 37(11):1746–1757, 1992.

[Borenstein *et al.*, 1996] J. Borenstein, H. R. Everett, and L. Feng. Where am I? Technical report, University of Michigan, April 1996. Edited by J. Borenstein.

[Borenstein *et al.*, 1997] J. Borenstein, H. R. Everett, L. Feng, and D. Wehe. Mobile robot positioning: Sensors and techniques. *Journal of Robotic Systems*, 14(4):231–249, 1997.

[Bradski, 1998] Gary R. Bradski. Real time face and object tracking as a component of a perceptual user interface. In *Proceedings of 4th International Workshop onApplications of Computer Vision*, pages 214–219, Princeton, New Jersey, USA, October 1998.

[Braitenberg, 1984] Valentino Braitenberg. *Vehicles: Experiments in Synthetic Psychology*. MIT Press, Cambridge, Mass., 1984.

[Brockett, 1983] R.W. Brockett. Asymptotic stability and feedback stabilization. In R. W. Brockett, R. S. Millman, and H. J. Sussman, editors, *Differential Geometric Control Theory*, pages 181–191. Birkhauser, Boston, USA, 1983.

[Brooks and Flynn, 1989] Rodney A. Brooks and Anita M. Flynn. Fast, cheap and out of control: A robot invasion of the solar system. *Journal of the British Interplanetary Society*, 42:478–485, 1989.

[Brooks, 1986] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, March 1986. Also in MIT AI Memo 864.

[Brooks, 1990] Rodney A. Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6:3–15, 1990.

[Buskey *et al.*, 2003] G. Buskey, J. Roberts, P. Corke, P. Ridley, and G. Wyeth. Sensing and control for a small-size helicopter. In B. Siciliano and P. Dario, editors, *Experimental Robotics*, volume VIII, pages 476–487. Springer-Verlag, 2003.

[Campion *et al.*, 1991] G. Campion, B. d'Andrea Novel, and G.Bastin. Controlability and state feedback stabilizability of non holonomic mechanical systems. In C. Canudis de Wit, editor, *Advanced Robot Control: Proceedings of the International Workshop on Nonlinear and Adaptive Control: Issues in Robotics*, number 162 in Lecture Notes in Control and Information Sciences, pages 106–124. Springer Verlag, 1991.

[Canudas de Wit and Sørdalen, 1992a] C. Canudas de Wit and O. J. Sørdalen. examples of piecewise smooth stabilization of nonlinear systems with less inputs than states. In *Proceedings of the IFAC nonlinear control design symposium (NOLCOS)*, Bordeax, France, 1992.

[Canudas de Wit and Sørdalen, 1992b] C. Canudas de Wit and O. J. Sørdalen. Exponential stabilization of mobile robots with nonholonomic constraints. *IEEE Transactions on Automatic Control*, 37(11):1791–1797, November 1992.

[Canudas de Wit *et al.*, 1993] C. Canudas de Wit, H. Khennouf, C. Samson, and O. J. Sørdalen. Nonlinear control design for mobile robots. In Yuan F. Zheng, editor, *Recent Trends in Mobile Robots*, volume 11 of *World Scientific Series in Robotics and Automated Systems*, chapter 5, pages 121–156. World Scientific Publishing, 1993.

[Cartwright and Collett, 1983] B. A. Cartwright and T. S. Collett. Landmark learning in bees. *Journal of Comparative Physiology*, 151:521–543, 1983.

[Chahl and Srinivasan, 1996] J. S. Chahl and M. V. Srinivasan. Visual computation of egomotion using an image interpolation technique. *Biological Cybernetics*, 74:405–411, 1996.

[Chahl and Srinivasan, 1997a] J. S. Chahl and M. V. Srinivasan. Range estimation with a panoramic visual sensor. *Journal of the optical society of America*, 14(9), September 1997.

[Chahl and Srinivasan, 1997b] J. S. Chahl and M. V. Srinivasan. Reflective surfaces for panoramic imaging. *Applied Optics*, 36(31):8275–8285, November 1997.

[Chang and Krumm, 1999] Peng Chang and John Krumm. Object recognition with color cooccurence histograms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 498–504, 1999.

[Chatila and Laumond, 1985] R. Chatila and J. Laumond. Position referencing and consistent world modelling for mobile robots. In *International Conference on Robotics and Automation*, pages 138–145. IEEE, 1985.

[Chen, 1999] T. Chen. Development of a vision-based positioning system for high density area. In *Proceedings of the Asian Conference on Remote Sensing*, Hong Kong, China, November 1999.

[Choset and Burdick, 2000] Howie Choset and Joel Burdick. Sensor-based exploration: The hierarchical generalized voronoi graph. *International Journal of Robotics Research*, 19(2):96–125, February 2000.

[Chung *et al.*, 2001] Hakyoung Chung, Laura Ojeda, and Johann Borenstein. Accurate mobile robot dead-reckoning with a precision calibrated fiber-optic gyroscope. *IEEE Transactions on Robotics and Automation*, 17(1):80–84, February 2001.

[Collett and Collett, 2000] Matthew Collett and Thomas S. Collett. How do insects use path integration for their navigation? *Biological Cybernetics*, 83:245–259, 2000.

[Conticelli *et al.*, 1999] Fabio Conticelli, Bendetto Allota, and Pradeep K. Khosla. Image-based visual servoing of nonholonomic mobile robots. In *Proceedings of the 38rd Conference on Decision and Control*, pages 3496–3501. IEEE, Phoenix, Arizona, USA, December 1999.

[Conticelli *et al.*, 2000] F. Conticelli, D. Prattechizzo, F. Guidi, and A. Bicchi. Vision-based dynamic estimation and set-point stabilization of nonholonomic vehicles. In *International Conference on Robotics and Automation*, pages 2771–2776, San Francisco, California, USA, 2000. IEEE.

[Corke *et al.*, 2003] Peter Corke, Dimitris Symeonidis, and Kane sher. Tracking road edges in the panospheric image plane. In *Proceedings of the 2003 International Conference on Intelligent Robots and Systems*, Las Vegas, USA, October 2003. IEEE/RSJ.

[Corke, 1994] Peter I. Corke. *High-Performance Visual Closed Loop Control*. PhD thesis, Department of Mechanical and Manufacturing Engineering, University of Melbourne, July 1994.

[Corke, 1996] Peter I. Corke. *Visual Control of Robots — high-performance visual servoing*. Robotics and Mechatronics. Research Studies Press, 1996.

[Corke, 2001] Peter Corke. Mobile robot navigation as a planar visual servoing problem. In *10th International Symposium of Robotics Research*, pages 217–223. IFRR, 2001.

[Coron, 1992] Jean-Michel Coron. Global asymptotic stabilization for controllable systems without drift. *Mathematics of Control, Signals and Systems*, 5:295–312, 1992.

[Cox and Wilfong, 1990] Ingemar J. Cox and Gordon T. Wilfong. Introduction. In Ingemar J. Cox and Gordon T. Wilfong, editors, *Autonomous Robot Vehicles*, pages xix–xxi. Springer Verlag, 1990.

[Das *et al.*, 1998] Jaydev P. Das, Jim Ostrowski, and Vijay Kumar. Controlling formations of multiple mobile robots. In *1998 IEEE International Conference on Robotics and Automation*, pages 2864–2869, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Das *et al.*, 2001] A.K. Das, R. Fierro, V. Kumar, B. Southall, J. Spletzer, and C.J. Taylor. Real-time vision based control of a non-holonomic mobile robot. In *International Conference on Robotics and Automation*, pages 1714–1719, Seoul, Korea, May 2001. IEEE.

[Dash, 2002] Mike Dash. *Batavia's Graveyard*. Crown Publishing Group, 2002.

[De Luca and Oriolo, 1995] Alessandro De Luca and Giuseppe Oriolo. Modelling and control of nonholonomic mechanical systems. In J. Angeles and A. Kecskemethy, editors, *Kinematics and Dynamics of Multi-Body Systems*, CISM Lecture Notesno. 360, chapter 7, pages 277–342. Springer Verlag, 1995.

[De Luca *et al.*, 1997] Alessandro De Luca, Giuseppe Oriolo, and Claude Samson. Feedback control of a nonholonomic car-like robot. In Jean-Paul Laumond, editor, *Planning Robot Motion*, chapter 4. Springer Verlag, 1997.

[De Luca *et al.*, 2002] Alesandro De Luca, Giuseppe Oriolo, Luca Paone, and Paolo Robuffo Giordano. Experiments in visual feedback control of a wheeled mobile robot. In *International Conference on Robotics and Automation*, pages 2073–2078, Washington DC, USA, May 2002. IEEE.

[Deans and Hebert, 2000] Matthew Deans and Martial Hebert. Experimental comparison of techniques for localization and mapping using a bearing-only sensor. In *Proeedings of the 7th International Symposium on Experimental Robotics*, 2000.

[Debenest *et al.*, 2002] Paulo Debenest, Edwardo F. Fukushima, and Shigeo Hirose. Development and control of a buggy robot for operations on unstructured terrain. In

Proceedings of the 2002 International Conference on Intelligent Robots and Systems, pages 763–768, Lausanne, Switzerland, October 2002. IEEE/RSJ.

[Desai and Miller, 1992] Rajiv S. Desai and David P. Miller. A simple reactive architecture for robust robots. In *International Conference on Robotics and Automation*, May 1992. Workshop on Intelligent Architectures for Robots.

[DeSouza and Kak, 2002] Guilherme N. DeSouza and Avinash C. Kak. Vision for mobile robot navigation: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):237–267, February 2002.

[Diaz *et al.*, 2001] Jonathan F. Diaz, Alexander Stoytchev, and Ronald C. Arkin. Exploring unknown structured environments. Submitted to FLAIRS 2001, 2001.

[Dickmanns *et al.*, 1994] E. D. Dickmanns, R. Behringer, R. Dickmanns, T. Hildebrant, M. Mauer, F. Thomanek, and J. Shielhlen. The seeing passenger car 'VaMoRs-P'. In *Proceedings of the International Symposium on Intelligent Vehicles*, Paris, 1994.

[Dissanayake *et al.*, 1999] Gamini Dissanayake, Salah Sukkarieh, Eduardo Nebot, and Hugh Durrant-Whyte. A new algorithm for the alignment of inertial measurement units without external observation for land vehicle applications. In *International Conference on Robotics and Automation*, pages 2274–2279, Detroit, Michigan, 1999. IEEE.

[Divelbiss and Wen, 1997] Adam W. Divelbiss and John T. Wen. A path space approach to nonholonomic motion planning in the presence of obstacles. *IEEE Transactions on Robotics and Automation*, 13(3):443–450, June 1997.

[Donati, 2002] P. Donati. Survey of technical preventative measures to reduce whole-body vibration effects when designing mobile machinery. *Journal of Sound and Vibration*, 253(1):169–183, 2002.

[Donnart and Meyer, 1996] Jean-Yves Donnart and Jean-Arcady Meyer. Spatial exploration, map learning, and self-positioning with monalysa. In P. Maes, M. Mataric´,

J. A. Meyer, J. Pollack, and S. W. Wilson, editors, *From Animals to Animats 4 - Proceedings of the 4th International Conference on Simulation of Adaptive Behaviour*, 1996.

[Duckett and Nehmzow, 1999a] Tom Duckett and Ulrich Nehmzow. Exploration of unknown environments using a compass, topological map and neural network. Proceedings of International Symposium on Computational Intelligence in Robotics and Automation, 1999. pp 312-317.

[Duckett and Nehmzow, 1999b] Tom Duckett and Ulrich Nehmzow. Knowing your place in real world environments. In *Proceedings of EUROBOT'99, 3rd European workshop on Advanced Mobile Robots*, Zurich, Switzerland, 1999.

[Duckett and Nehmzow, 2000] Tom Duckett and Ulrich Nehmzow. Performance comparison of landmark recognition systems for navigating mobile robots. In *Proceedings AAAI*, 2000.

[Everett, 1995] H. R. Everett. *Robot Motion Planning - Sensors for Mobile Robots - Theory and Application*. A.K. Peters Ltd, 1995.

[Fitzgibbons and Nebot, 2002] Trevor Fitzgibbons and Eduardo Nebot. Bearing-only SLAM using colour-based feature tracking. In *Proceedings of the Australian Conference on Robotics and Automation*, Auckland, New Zealand, December 2002. published via CDROM.

[Franz and Mallot, 2000] Matthies O. Franz and Hanspeter A. Mallot. Biomimetic robot navigation. *Robotics and Autonomous Systems*, 30:133–153, 2000.

[Franz *et al.*, 1997] M. Franz, B. Scholkopf, and H. Bulthoff. Homing by parameterized scene matching. In *Proceedings of the 4th European Conference on Artificial Life*, 1997.

[Franz *et al.*, 1998a] Matthias O. Franz, Bernhard Scholkopf, Hanspeter A. Mallot, and Heinrich H. Bulthoff. Learning view graphs for robot navigation. *Autonomous Robots 5*, pages 111–125, 1998.

[Franz *et al.*, 1998b] Mattias O. Franz, Bernhard Schölkopf, Hanspeter A. Mallot, and Heinrich H. Bülthoff. Where did i take that snapshot? scene-based homing by image matching. *Biological Cybernetics*, 79:191–202, 1998.

[Gaspar *et al.*, 2000] Jose Gaspar, Niall Winters, and Jose Santos-Victor. Vision-based navigation and environmental representations with an omnidirectional camera. *IEEE Transactions on Robotics and Automation*, 16(6):890–898, December 2000.

[Gat, 1998] Erran Gat. On three-layer architectures. In David Kortencamp, R. Peter Bonnasso, and Robin Murphy, editors, *Artificial Intelligence and Mobile Robots*. AAAI Press, 1998.

[Gaussier *et al.*, 2000] P. Gaussier, C. Joulain, J. P. Banquet, S. Lepetre, and A. Revel. The visual homing problem: An example of robotics / biology cross fertilization. *Robotics and Autonomous Systems*, 30:155–180, 2000.

[Giralt *et al.*, 1979] Georges Giralt, Ralph Sobek, and Raja Chatila. A multi-level planning and navigation system for a mobile robot. In *IJCAI-79*, pages 335–337, 1979.

[Golfarelli *et al.*, 2001] Matteo Golfarelli, Dario Maio, and Stefano Rizzi. Correction of dead-reckoning errors in map building for mobile robots. *IEEE Transactions on Robotics and Automation*, 17(1):37–47, February 2001.

[Guivant *et al.*, 2000] Jose Guivant, Eduardo Nebot, and Stephen Baker. Autonomous navigation and map building using laser range sensors in outdoor applications. *Journal of Robotic Systems*, 17(10):565–583, 2000.

[Guldner and Utkin, 1994] J. Guldner and V. Utkin. Stabilization of nonholonomic mobile robots using lyapunov functions for navigation and sliding mode control. In *Pro-*

ceedings of the 33rd Conference on Decision and Control, pages 2967–2972, Lake Buena Vista, Florida, USA, December 1994.

[Gutmann and Konolige, 2000] Jens-Steffen Gutmann and Kurt Konolige. Incremental mapping of large cyclic environments. CIRA, 2000.

[Hafner and Möller, 2001] Verena V. Hafner and Ralf Möller. Learning of visual navigation strategies. In M. Quoy, P. Gaussier, and J. Wyatt, editors, *Proceedings of the European Workshop on Learning Robots (EWLR-9)*, pages 47–56, 2001.

[Hafner and Saloman, 2002] Verena V. Hafner and Ralf Saloman. Evolving neural controllers for visual navigation. In *Proceedings of the Fourth Congress on Evolutionary Computation (CEC-2002), IEEE World Congress on Computational Intelligence.*, 2002.

[Hague and Tillet, 1996] T. Hague and N. D. Tillet. Navigation and control of an autonomous horticultural robot. *Mechatronics*, 6(2):165–180, 1996.

[Hague and Tillet, 2001] T. Hague and N. D. Tillet. A bandpass filter based approach to crop row location and tracking. *Mechatronics*, 11(1):1–12, 2001.

[Hague *et al.*, 1997] T. Hague, J. A. Marchant, and N. D. Tillet. Autonomous robot navigation for precision horticulture. In *International Conference on Robotics and Automation*, pages 1880–1885, Alburquerque, New Mexico, USA, 1997. IEEE.

[Hague *et al.*, 2000] T. Hague, J. A. Marchant, and N. D. Tillet. Ground based sensing systems for autonomous agricultural vehicles. *Computers and Electronics in Agriculture*, 25(1):11–28, 2000.

[Haralick and Shapiro, 1992] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision*, volume 1. Addison Wesley Publishing Company, 1992.

[Hashimoto and Noritsugo, 1997] Koichi Hashimoto and Toshiro Noritsugo. Visual servoing of nonholonomic cart. In *International Conference on Robotics and Automation*, pages 1719–1724, Albuqueque, New Mexico, USA, April 1997. IEEE.

[Hebert, 2000] Martial Hebert. Active and passive sensing for robotics. In *International Conference on Robotics and Automation*, pages 102–110, San Francisco, California, USA, 2000. IEEE.

[Hill and Wilfong, 1990] Murray Hill and Gordon T. Wilfong. Introduction. In *Autonomous Robot Vehicles*. Springer Verlag, 1990.

[Holland, 1997] Owen E. Holland. Grey walter: the pioneer of real artificial life. In Christopher Langton, editor, *Alife V: Proceedings of the 5th International Workshop on Artificial Life*, pages 34–44, Cambridge, 1997. MIT Press.

[Hong *et al.*, 1991] Jiawei Hong, Xiaonan Tan, Brian Pinette, Richard Weiss, and Edward M. Riseman. Image-based homing. In *International Conference on Robotics and Automation*, pages 620–625, Sacramento, California, USA, April 1991. IEEE.

[Horswill, 1993] I. Horswill. Polly: A vision-based artificial agent. In *Proceedings of the eleventh national conference on artificial intelligence (AAAI'93)*, Washington DC, USA, July 1993. MIT Press.

[Hsu and Hwang, 1998] Jane Yung-Jen Hsu and Liang-Sheng Hwang. A graph based exploration strategy of indoor environments by an autonomous mobile robot. In *International Conference on Robotics and Automation*, pages 1262–1268, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Huang *et al.*, 1999] Jing Huang, S. Ravi Kumar, Mandar Mitra, Wei-Jinj Zhu, and Ramin Zabih. Spatial color indexing and applications. *Internation Journal of Computer Vision*, 35(3):245–268, 1999.

[Huo and Gei, 2001] Wei Huo and S. S. Gei. Exponential stabilization of non-holonomic systems: an ENI approach. *Internation Journal of Control*, 74(14):1492–1500, 2001.

[Hutchinson *et al.*, 1996] Seth Hutchinson, Greg Hager, and Peter Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.

[IEEE, 1983] IEEE. *Navigation Aid Terms*, 1983.

[Indiveri, 1999] Giovanni Indiveri. Kinematic time-invariant control of a 2s nonholonomic vehicle. In *Proceedings of the 38rd Conference on Decision and Control*. IEEE, Phoenix, Arizona, USA, December 1999.

[Jarvis, 1997] Ray Jarvis. Intelligent robotics - where are we at and where are we going? In *Proceedings of the International Conference on Field and Service Robotics*, pages 21–25, Canberra Australia, 1997.

[Jensfelt, 1999] P. Jensfelt. Localization using laser scanning and minimalistic environmental models. Licentiate thesis, Automatic Control, Royal Institute of Technology, SE-100 44 Stockholm, Sweden, 1999.

[Jiang *et al.*, 2001] Zhong-Ping Jiang, Erjen Lefeber, and Henk Nijmeijer. Saturated stabilization and tracking of a nonholonomic mobile robot. *Systems and Control Letters*, 42:327–332, 2001.

[Kantor and Rizzi, 2003] George Kantor and Alfred A. Rizzi. Feedback control of underactuated systems via sequential composition: Visually guided control of a unicycle. In *Proceedings of the International Symposium of Robotics Research*, Sienna, Italy, October 2003. to appear.

[Kelly and Nagy, 2002] A. Kelly and N. Nagy. Reactive nonholonomic trajectory generation via parametric optimal control. *International Journal of Robotics Research*, 22(7/8):583–602, 2002.

[Kim and Tsiotras, 2002] ByungMoon Kim and Panagiotis Tsiotras. Controllers for unicycle-type wheeled robots: Theoretical results and experimental validation. *IEEE Transactions on Robotics and Automation*, 18(3):294–307, June 2002.

[Kodagoda *et al.*, 2002] K. R. S. Kodagoda, W. S. Wijesoma, and E. K. Teoh. Fuzzy speed and steering control of an agv. *IEEE Transactions on Control Systems Technology*, 10(1):112–120, 2002.

[Kolmanovsky and McClamroch, 1995] Ilya Kolmanovsky and N. Harris McClamroch. Developments in nonholonomic control problems. *IEEE Control Systems Magazine*, 15:20–36, December 1995.

[Kosecka, 1997] Jana Kosecka. Visually guided navigation. *Robotics and Autonomous Systems*, 21:37–50, 1997.

[Kral, 1999] K. Kral. Binocular vision and distance estimation. In F.R. Prete, H. Wells, P.H. Wells, and L.E. Hurd, editors, *The Praying Mantis*, pages 114–140. The Johns Hopkins University Press, 1999.

[Krose and Bunschoten, 1999] B. J. A. Krose and R. Bunschoten. Probabilistic localization by appearance models and active vision. In *International Conference on Robotics and Automation*, pages 2255–2260, Detroit, Michigan, 1999. IEEE.

[Krose *et al.*, 2000] B. J. A. Krose, N. Vlassis, and R. Bunschotten. Omnidirectional vision for appearance-based robot localization. In Greogory D. Hager, Henrik I. Chistensen, Horst Bunke, and Rolf Klein, editors, *Sensor Based Intelligent Robots*, number 2238 in Lecture Notes in Computer Science, pages 39–50. Springer, 2000.

[Krose *et al.*, 2001] B. J. A. Krose, N. Vlassis, R. Bunschoten, and Y. Motomura. A probabilistc model for appearance-based robot localization. *Image and Vision Computing*, 19(6):381–391, April 2001.

[Krstić and Kokotović, 1996] Miroslave Krstić and Petar V. Kokotović. Adaptive nonlinear control. In William S. Levine, editor, *The Control Handbook*, The electrical engineering handbook series, chapter 57: Design Methods. CRC Press, 1996.

[Kuipers and Byun, 1991] Benjamin Kuipers and Yung-Tai Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, 8:47–63, 1991.

[Kuipers, 1977] Benjamin J. Kuipers. *Representing Knowledge of Large-Scale Space*. PhD thesis, Mathematics Department, Massachusetts Institute of Technology, June 1977.

[Lambrinos *et al.*, 1997] Dimitrios Lambrinos, Hiroshi Koayashi, Rolf Pfeifer, Marinus Maris, Thomas Labhart, and Rudiger Wehner. An autonomous agent navigating eith a polarized light compass. *Adaptive Behavior*, 6(1):131, 1997.

[Lambrinos *et al.*, 2000] Dimitrios Lambrinos, Ralf Möller, Thomas Labhart, Rolf Pfiefer, and Rudiger Wehner. A mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems*, 30:39–64, 2000.

[Latombe, 1991] Jean-Claude Latombe. *Robot Motion Planning*. Kluwer Academic, 1991.

[Laubach *et al.*, 1998] Sharon L. Laubach, Joel Burdick, and Larry Matthies. An autonomous path planner implemented on the rocky 7 prototype microrover. In *International Conference on Robotics and Automation*, pages 292–297, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Lee *et al.*, 1999] Sungon Lee, Manchul Kim, Youngil Youm, and Wankyun Chung. Control of a car-like mobile robot for parking problem. In *International Conference on Robotics and Automation*, pages 1–6, Detroit, Michigan, 1999. IEEE.

[Lee *et al.*, 2001] Yong-Beom Lee, Bum-Jae You, and Seong-Whan Lee. A real-time color-based object tracking robust to irregular illumination variations. In *International Conference on Robotics and Automation*, pages 1659–1664, Seoul, Korea, May 2001. IEEE.

[Leonard and Durrant-Whyte, 1991] John J. Leonard and Hugh F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3):376–382, June 1991.

[Leonard *et al.*, 1992] J. Leonard, H. Durrant-Whyte, and I. Cox. Dynamic map building and localization for an autonomous mobile robot. *International Journal of Robotics Research*, 11(4):286–292, 1992.

[Levitt and Lawnton, 1990] Tod S. Levitt and Daryl T. Lawnton. Qualitative navigation for mobile robots. *Artificial Intelligence*, 44:305–360, 1990.

[Li and Canny, 1993] J. Li and J. F. Canny, editors. *Nonholonomic motion planning*. Kluwer, 1993.

[Lietmann and Lohmann, 2000] Thorsten Lietmann and Boris Lohmann. Docking procedure for a non-holonomic mobile platform based on visual servoing. In *Proceedings of the Sixth International IFAC Symposium on Robot Control (SYROCO)*, pages 301–306, Vienna, Austria, 2000.

[Liubakka *et al.*, 1993] M. K. Liubakka, D. S. Rhode, J. R. Winkelman, and P. V. Kokotović. Adapative automotive speed control. *IEEE Transactions on Automatic Control*, 38(7):1011–1020, July 1993.

[Lizarralde and Wen, 1996] Fernando Lizarralde and John T. Wen. Feedback stabilization of nonholonomic systems in presence of obstacles. In *Proceedings of the International Conference on Robotics and Automation*, Minneapolis, Minnesota, USA, April 1996. IEEE.

[Lu and Milios, 1997] Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2d range scans. *Journal of Intelligent and Robotic Systems*, 18(3):249–275, March 1997.

[Lu *et al.*, 2000] Jiangzhou Lu, Sepanta Sakhavat, Xie Ming, and Christian Laugier. Sliding mode control for nonholonomic mobile robot. In *Proceedings of the International Conference on Control, Automation, Robotics and Vision*, pages 465–470, 2000.

[Lyons and Hendriks, 1992] D. Lyons and A. Hendriks. Reactive planning. In S. Shapiro, editor, *Encyclopedia of Artificial Intelligence*. Wiley, New York, USA, 2 edition, 1992.

[Mallet *et al.*, 2000] A. Mallet, S. Lacroix, and L. Gallo. Postion estimation in outdoor environments using pixel tracking and stereovision. In *International Conference on Robotics and Automation*, pages 3519–3524, San Francisco, California, USA, 2000. IEEE.

[Mataric´, 1990] Maja J. Mataric´. A distributed model for mobile robot environment-learning and navigation. Master's thesis, MIT Artificial Intelligence Laboratory, 1990.

[Mataric´, 1992] Maja J. Mataric´. Integration of representation into goal-driven behaviour-based robots. *IEEE Transactions on Robotics and Automation*, 8(3):304–312, June 1992.

[McKerrow, 1991] Phillip John McKerrow. *Introduction to Robotics*. Addison Wesley, 1991.

[M'Closkey and Murray, 1997] Robert T. M'Closkey and Richard M. Murray. Exponential stabilizatio nof driftless nonlinear control systems using homogeneous feedback. *IEEE Transactions on Automatic Control*, 42(5):614–628, 1997.

[Milford and Wyeth, 2003] Michael Milford and Gordon Wyeth. Hippocampal models for simultaneous localisation and mapping on an autonomous robot. In *Proceedings of the Australasian Conference on Robotics and Automation*, Brisbane, Australia, December 2003. published via CDROM.

[Minten *et al.*, 2001] Brian W. Minten, Robin Murphy, Jeff Hyams, and Mark Micere. Low-order complexity vision-based docking. *IEEE Transactions on Robotics and Automation*, 17(6):922–930, d 2001. December.

[Möller *et al.*, 1998] Ralf Möller, Dimitris Lambrinos, Rolf Pfiefer, and Rúdiger Wehner. Insect strategies of visual homing in mobile robots. In *Proceedings of the Computer Vision and Mobile Robotics Workshop, CVMR'98*, pages 37–45, Heraklion, Greece, 1998.

[Möller, 2000] Ralf Möller. Insect visual homing strategies in a robot with analog processing. *Biological Cybernetics*, 83:231–243, 2000.

[Möller, 2001] Ralf Möller. Do insects use templates or parameters for landmark navigation? *Journal of Theoretical Biology*, 210:33–45, 2001.

[Moravec and Elfes, 1985] H. P. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *IEEE International Conference on Robotics and Automation*, pages 116–121, St. Louis, Missouri, USA, 1985. IEEE Computer Society Press.

[Moravec, 1990] Hans Moravec. The Stanford Cart and the CMU Rover. In I. J. Cox and G. T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 407–41. Springer-Verlag, 1990.

[Murray and Sastry, 1993] Richard M. Murray and S. Shankar Sastry. Nonholonomic motion planning: Steering using sinusoids. *IEEE Transactions on Automatic Control*, 38(5):700–716, May 1993.

[Murrieri *et al.*, 2002] Pierpaolo Murrieri, Daniele Fontanelli, and Antonio Bicci. Visual-servoed parking with limited view angle. In *Preprints of the International Symposium on Experimental Robotics*, Sant'Angelo d'Ischia, Italy, 2002.

[Nagatani *et al.*, 1998] Keiji Nagatani, Howie Choset, and Sebastion Thrun. Towards exact localization without explicit localization with the generalized voronoi map. In *International Conference on Robotics and Automation*, pages 342–348, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Nagle and Srinivasan, 1996] M. G. Nagle and M. V. Srinivasan. Structure from motion: determining the range and orientation of surfaces by image interpolation. *Journal of the optical society of America*, 13(1):25–34, January 1996.

[Nebot and Durrant-Whyte, 1999] Eduardo M. Nebot and Hugh Durrant-Whyte. A high integrity navigation architecture for outdoor autonomous vehicles. *Robotics and Autonomous Systems*, 26:81–97, 1999.

[Nehmzow and Owen, 2000] Ulrich Nehmzow and Carl Owen. Robot navigation in the real world: Experiments with manchester's FortyTwo in unmodified large environments. *Robotics and Autonomous Systems*, 33:223–242, 2000.

[Nehmzow, 2000] Ulrich Nehmzow. *Mobile Robotics: A Practical Introduction*. Springer, London, 2000.

[Nillson, 1969] N. Nillson. A mobile automation: An application of artificial intelligence techniques. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1969.

[OED, 2000] OED. *Oxford English Dictionary*. Oxford University Press, 2000.

[Ollis *et al.*, 1999] Mark Ollis, Herman Herman, and Sanjiv Singh. Analysis and design of panoramic stereo vision using equi-angular pixel cameras. Technical Report CMU-RI-TR-99-04, The Robotics Institute, Carnegie Mellon University, January 1999.

[Ollis, 1997] Mark Ollis. *Perception Algorithms for a Harvesting robot*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, August 1997.

[Oriolo *et al.*, 1998] Giuseppe Oriolo, Stefano Panzieri, and Giovanni Ulivi. An iteratuve learning controller for nonholonomic mobile robots. *International Journal of Robotics Research*, 17(9):954–970, 1998.

[Pomerleau, 1993] Dean A. Pomerleau. *Neural Network Perception for Mobile Robot Guidance*. Kluwer Academic Publishers, 1993.

[Pomet, 1992] J. B. Pomet. Explicit design of time-varying stabilizing control laws for a class of controllable systems without drift. *Systems and Control Letters*, 18:139–145, 1992.

[Pourboghrat and Karlsson, 2002] Farzad Pourboghrat and Mattias P. Karlsson. Adaptive control of dynamic mobile robots with nonholonomic constraints. *Computers and Electrical Engineering*, 28:241–253, 2002.

[Pourboghrat, 2002] Farzad Pourboghrat. Exponential stabilization of nonholonomic mobile robots. *Computers and Electrical Engineering*, 28:349–259, 2002.

[Rasmussen *et al.*, 1996] Christopher Rasmussen, Kentaro Toyama, and Gregory D. Hager. Tracking objects by color alone. Technical Report DCS-TR-1114, Yale University, USA, 1996.

[Reignier *et al.*, 1997] Patrick Reignier, Volker Hansen, and James L. Crowley. Incremental supervised learning for a mobile robot reactive control. *Robotics and Autonomous Systems*, 19:247–257, 1997.

[Rencken *et al.*,] Wolfgang Rencken, Wendelin Feiten, and Martin Soika. Large consistent geometric feature maps. Powerpoint presentation, Siemens Corporate Technology - Information and Communications.

[Rencken, 1994] W. D. Rencken. Autonomous sonar navigation in indoor, unknown, and unstructured environments. In *Proceedings of International Conference on Intelligent Robots and Systems*, pages 431–438. IEEE/RSJ, 1994.

[Reynolds, 1999] Craig Reynolds. Steering behaviors for autonomous characters. In *Proceedings of the Game Developers Conference*, 1999.

[Rizzi *et al.*, 2000] A. Rizzi, D. Duina, S. Inelli, and R. Cassinis. A novel visual landmark matching for a biologically inspired homing. *Pattern Recognition Letters*, 22:1371–1378, 2000.

[Roberts *et al.*, 2000] Jonathan M. Roberts, Elliot S. Duff, Peter I. Corke, Pavan Sikka, Graeme J. Winstanley, and Jock Cunningham. Autonomous control of underground mining vehicles using reactive navigation. In *International Conference on Robotics and Automation*, pages 3790–3795, San Francisco, California, 2000. IEEE.

[Rodriguez *et al.*, 1998] F. Javier Rodriguez, Manuel Mazo, and Miguel A. Sotelo. Automation of an industrial forklift truck, guided by artificial vision in open environments. *Autonomous Robots*, 5:215–231, 1998.

[Rossel and Wehner, 1986] S. Rossel and R. Wehner. Polarization vision in bees. *Nature*, 323:128–131, 1986.

[Russ, 1992] John C. Russ. *The Image Processing Handbook*. CRC Press, 1992.

[Samson and Ait-Abderrahim, 1991] C. Samson and K. Ait-Abderrahim. Feedback control of a non-holonomic wheeled cart in cartesian space. In *International Conference on Robotics and Automation*, pages 1136–1141, Sacramento, California, USA, April 1991. IEEE.

[Samson, 1991] Claude Samson. Velocity and torque feedback control of a nonholonomic cart. In C. Canudis de Wit, editor, *Advanced Robot Control: Proceedings of the International Workshop on Nonlinear and Adaptive Control: Issues in Robotics*, number 162 in Lecture Notes in Control and Information Sciences, pages 125–151. Springer Verlag, 1991.

[Samson, 1993] Claude Samson. Time-varying feedback stabilisation of car-like wheeled mobile robots. *International Journal of Robotics Research*, 12(1):55–64, 1993.

[Samson, 1995] Claude Samson. Control of chained systems: Application to path following and time-varying point stabilization of mobile robots. *IEEE Transactions on Automatic Control*, 40(1):64–77, January 1995.

[Santos-Victor and Sandini, 1994] José Santos-Victor and Guilio Sandini. Visual behaviours for docking. Technical Report TR 2/94, LIRA-Lab – DIST University of Genova, Via Opera Pia 13 – 16145 Genova, Italy, June 1994.

[Schultz and Adams, 1998] Alan Schultz and William Adams. Continuous localization with evidence grids. In *International Conference on Robotics and Automation*, pages 2833–2839, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Scott *et al.*, 2000] Alexander Scott, Lynne E. Parker, and Claude Touzet. Quantative and qualative comparison of three laser ranges mapping algorithms using two types of laser scanner data. In *Proceedings of Systems, Man and Cybernetics*, 2000.

[Sekuler and Blake, 1994] Robert Sekuler and Randolph Blake. *Perception*. McGraw Hill, third edition, 1994.

[Seraji, 2000] Homayoun Seraji. Fuzzy traversability index: A new concept for terrain-based navigation. *Journal of Robotic Systems*, 17(2):75–91, 2000.

[Shimshoni, 2002] Ilan Shimshoni. On mobile robot localization from landmark bearings. In *International Conference on Robotics and Automation*, pages 3605–3611, Washington DC, USA, May 2002. IEEE.

[Slotine and Li, 1991] Jean-Jacques E. Slotine and Weiping Li. *Applied Nonlinear Control*. Prentice-Hall, 1991.

[Smith and Cheeseman, 1986] Randall C. Smith and Peter Cheeseman. On the representaion and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4):56–68, 1986.

[Smith *et al.*, 1990] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In Ingemar J. Cox and Gordon T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag, 1990.

[Sørdalen and Egeland, 1995] O. J. Sørdalen and O. Egeland. Exponential stabilisation of nonholonomic systems. *IEEE Transactions on Automatic Control*, 40(1):35–50, January 1995.

[Sørdalen, 1993] O. J. Sørdalen. Conversion of the kinematics of a car with n trailers into a chained form. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 382–387, Atlanta, Georgia, USA, 1993.

[Srinivasan *et al.*, 1997] M. V. Srinivasan, J. S. Chahl, and S. W. Zhang. Robot navigation by visual dead reckoning: Inspirations from insects. *International Journal of Pattern Recognition and Artificial Intelligence*, 11(1):35–47, 1997.

[Srinivasan *et al.*, 1999] M. V. Srinivasan, J. S. Chahl, K. Weber, S. Venkatesh, M. G. Nagle, and S. W. Zhang. Robot navigation inspired by principles of insect vision. *Robotics and Autonomous Systems*, 26:203–216, 1999.

[Sussman, 1979] H. J. Sussman. Subanalytic sets and feedback control. *Journal of differential equations*, 31(31-52), 1979.

[Swain and Ballard, 1991] Michael J. Swain and Dana H. Ballard. Color indexing. *Internation Journal of Computer Vision*, 7(1):11–32, 1991.

[Thompson and Zelinsky, 2002] Simon Thompson and Alexander Zelinsky. Accurate local positioning using visual landmarks from a panoramic sensor. In *International Conference on Robotics and Automation*, pages 2656–2661, Washington DC, USA, May 2002. IEEE.

[Thompson *et al.*, 1999] Simon Thompson, Alexander Zelinsky, and Mandyam Srinivasan. Automatic landmark selection for navigation with panoramic vision. In *Proceedings of the Australian Conference on Robotics and Automation*, Brisbane, Australia, March 1999.

[Thompson *et al.*, 2000] Simon Thompson, Toshihiro Matsui, and Alexander Zelinsky. Localisation using automatically selected landmarks from panoramic images. In *Proceedings of the Australian Conference on Robotics and Automation*, Melbourne, Australia, August 2000.

[Thorpe, 1990] Charles E. Thorpe, editor. *Vision and Navigation: The Carnegie Mellon Navlab*. Kluwer Academic Publishers, 1990.

[Thrun *et al.*, 1998] S. Thrun, A. Bucken, W. Burgard, D. Fox, T. Frohlinghaus, D. Hennig, T. Hofmann, M. Krell, and T. Schimdt. Map learning and high-speed navigation in rhino. In D. Kortenkamp, R. P. Bonasso, and R. Murphy, editors, *AI-based Mobile Robots: Case studies of successful robot systems. MIT Press, Cambridge, MA*, pages 21–52. MIT Press, cambridge, MA, 1998.

[Thrun *et al.*, 2000] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot mapping and 3d mapping. In *International Conference on Robotics and Automation*, pages 321–328, San Francisco, California, USA, 2000. IEEE.

[Thrun, 1998a] Sebastion Thrun. Finding landmarks for mobile robot navigation. In *International Conference on Robotics and Automation*, pages 958–963, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Thrun, 1998b] Sebastion Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intellegence*, 99:21–71, 1998.

[Trebi-Ollennu *et al.*, 1999] Ashitey Trebi-Ollennu, John Dolan, and Pradeep Khosla. Adaptive fuzzy throttle control for an all terrain vehicle. In *Proceedings of the IASTED International Conference on Intelligent Systems and Control*, October 1999.

[Trebi-Ollennu *et al.*, 2001] Ashitey Trebi-Ollennu, Terry Huntsberger, Yang Cheng, E. T. Baumgartner, and Brett Kennedy. Design and analysis of a sun sensor for plan-

etary rover absolute heading detection. Technical Report NASA CR-2001-210800, NASA, JPL and California Institute of Technology, Pasadena, January 2001.

[Trullier *et al.*, 1997] Olivier Trullier, Sidney I. Weiner, Alain Berthoz, and Jean-Arcady Meyer. Biologically based artificial navigation systems: Review and prospects. *Progress in Neurobiology*, 51:483–544, 1997.

[Tsakiris *et al.*, 1997] Dimitris P. Tsakiris, Patrick Rives, and Claude Samson. Applying visual servoing techniques to control nonholonomic mobile robots. In *Workshop on "New Trends in Image-based Robot Servoing", International Conference on Intelligent Robots and Systems*, Grenoble, France, September 8-12 1997. IEEE/RSJ.

[Ulrich and Nourbakhsh, 2000] Iwan Ulrich and Illah Nourbakhsh. Appearance-based place recognition for topological localization. In *International Conference on Robotics and Automation*, pages 1023–1029, San Francisco, California, USA, 2000. IEEE.

[Usher *et al.*, 2001] Kane Usher, Peter Corke, and Peter Ridley. A camera as a polarised light compass: Preliminary experiments. In *Proceedings of the Australian Conference on Robotics and Automation*, Sydney, Australia, November 2001. published via CDROM.

[Vassalo *et al.*, 2000] Raquel F. Vassalo, Hans J. Schneebeli, and Jose Santos-Victor. Visual servoing and appearance for navigation. *Robotics and Autonomous Systems*, 31:87–97, 2000.

[Walter, 1953] W. Grey Walter. *The Living Brain*. Norton, New York, 1953. reprinted 1963.

[Weber *et al.*, 1998] Keven Weber, Svetha Venkatesh, and M. V. Srinivasan. An insect based approach to robotic homing. In *International Conference on Pattern Recognition*, pages 297–299, Brisbane, Australia, 1998. IEEE Computer Society Press.

[Weber *et al.*, 1999] Keven Weber, Svetha Venkatesh, and Mandyam Srinivasan. Insect-inspired robotic homing. *Adaptive Behavior*, 7(1):65–97, 1999.

[Wehner and Wehner, 1990] R. Wehner and S. Wehner. Insect navigation: Use of maps or Ariadne's thread. In *Ethology, Ecology and Evolution 2*, pages 27–48, 1990.

[Wen, 1995] John Ting-Yung Wen. Control of nonholonomic systems. In William S. Levine, editor, *The Control Handbook*, The electrical engineering handbook series, chapter 76, pages 1359–1368. CRC Press and IEEE Press, 1995.

[Wilder and Pope, 1996] D. G. Wilder and M. H. Pope. Epidemiological and aetiological aspects of low back pain in vibration environments — an update. *Clinical Biomechanics*, 11(2):61–73, 1996.

[Winters *et al.*, 2000] Niall Winters, Jose Gaspar, Gerard Lacey, and Jose Santos-Victor. Omni-directional vision for robot navigation. IEEE Workshop on Omni-directional Vision (OMNIVIS'00), June 2000. South Carolina, USA.

[Yahja *et al.*, 1998] Alex Yahja, Anthony Stenz, Sanjiv Singh, and Barry L. Brummit. Framed quadtree path planning for mobile robots operating in sparse environments. In *International Conference on Robotics and Automation*, pages 650–655, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Yahja *et al.*, 2000] Alex Yahja, Sanjiv Singh, and Anthony Stentz. An efficient on-line path planner for outdoor mobile robots. *Robotics and Autonomous Systems*, 32:129–143, 2000.

[Yamauchi *et al.*, 1998] Brian Yamauchi, Alan Schultz, and William Adams. Mobile robot exploration and mapping strategy with continuous localization. In *International Conference on Robotics and Automation*, pages 3715–3720, Katholieke Universiteit Leuvun, Leuvun Belgium, 1998. IEEE.

[Zeil *et al.*, 2003] Jochen Zeil, Martin I. Hofmann, and Javaan S. Chahl. Catchment areas of panoramic snapshots in outdoor scenes. *Journal of the optical society of America*, 20(3), March 2003.

[Zeil, 1998] Jocken Zeil. Homing in fiddler crabs (*uca lactae* and *uca vomeris:ocypodidae*). *Journal of Comparative Physiology A*, 183:367–377, 1998.

[Zimmerman *et al.*, 1997] Chris. L. Zimmerman, Thomas M. Cook, and John C. Rosecrance. Work-related musculoskeletal symptoms and injuries among operating engineers: A review and guidelines for improvement. *Applied Ocupational and Environmental Hygiene*, 12(7):480–484, July 1997.