12-2015

Analysis and Synthesis of Effective Human-Robot Interaction at Varying Levels in Control Hierarchy

David Spencer
Clemson University, daspenc@clemson.edu

Analysis and Synthesis of Effective Human-Robot Interaction at Varying Levels in Control Hierarchy

---

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Mechanical Engineering

---

by
David A. Spencer
December 2015

Accepted by:
Dr. Yue "Sophie" Wang, Committee Chair
Dr. John Wagner
Dr. Laura Humphrey

Abstract

Robot controller design is usually hierarchical with both high-level task and motion planning and low-level control law design. In the presented works, we investigate methods for low-level and high-level control designs to guarantee joint performance of human-robot interaction (HRI). In the first work, a low-level method using the switched linear quadratic regulator (SLQR), an optimal control policy based on a quadratic cost function, is used. By incorporating measures of robot performance and human workload, it can be determined when to utilize the human operator in a method that improves overall task performance while reducing operator workload. This method is demonstrated via simulation using the complex dynamics of an autonomous underwater vehicle (AUV), showing this method can successfully overcome such scenarios while maintaining reduced workload. An extension of this work to path planning is also presented for the purposes of obstacle avoidance with simulation showing human planning successfully guiding the AUV around obstacles to reach its goals. In the high-level approach, formal methods are applied to a scenario where an operator oversees a group of mobile robots as they navigate an unknown environment. Autonomy in this scenario uses specifications written in linear temporal logic (LTL) to conduct symbolic motion planning in a guaranteed safe, though very conservative, approach. A human operator, using gathered environmental data, is able to produce a more efficient path. To aid in task decomposition and real-time switching, a dynamic human trust model is used. Simulations are given showing the successful implementation of this method.

Acknowledgments

I would like to first thank my advisor, Dr. Yue Wang, for her continued help and guidance over the course of this work. I would also like to thank my fellow researchers for their invaluable advice and assistance and for providing such a friendly work environment. I would like to especially thank Hamed Saeidi for his help and encouragement, especially in the early days of this work. Lastly I would like to give much thanks to my family whose moral and financial support allowed me to pursue this goal. I would not be here if not for you.

Table of Contents

Title Page . . . . . **i**

Abstract . . . . . **ii**

Acknowledgments . . . . . **iii**

List of Tables . . . . . **vi**

List of Figures . . . . . **vii**

1 Introduction **1**
1.1 HRI in Control 1
1.2 Trust in HRI 3
1.3 Overview 4

2 SLQR Suboptimal Human-Robot Collaborative Guidance and Navigation for Autonomous Underwater Vehicles **5**
2.1 Introduction 5
2.2 AUV Dynamic Modeling 7
2.3 Collaborative Manual & Autonomous Motion Guidance Strategy 10
2.4 Simulations Results 18
2.5 Conclusions 22

3 AUV Suboptimal Switching Between Waypoint Following and Obstacle Avoidance in Human-Robot Collaborative Guidance and Navigation **24**
3.1 Introduction 24
3.2 Modification of SLQR 26
3.3 Simulation 30
3.4 Conclusion 31

4 Trust-Based Human-Robot Interaction for Safe and Scalable Multi-Robot Symbolic Motion Planning **33**
4.1 Introduction 33
4.2 Human-Robot Interaction for Symbolic Motion Planning 35

4.3 Trust-Based Specification Decomposition 40
4.4 Real-Time Trust-Based Switching Between Human and Robot Motion Planning 45
4.5 Simulation 48
4.6 Conclusions 50

5 Conclusions 53
5.1 Discussion 53
5.2 Future Works 54

Appendices 55
A AUV Equations of Motion 56
B Linearized Dynamic Matrices 57

Bibliography 58

List of Tables

2.1 Deviation from desired waypoints in meters 19
2.2 Statistical results of varying disturbance parameters 19

List of Figures

2.1 Collaborative Control Scheme for the AUV. 10
2.2 3D results of autonomous LQR model verification 12
2.3 (a) AUV depth change under the LQR controller and (b) inputs α, β calculated by LQR. 12
2.4 Results of simulation under autonomous mode 20
2.5 Results of simulation under collaborative mode 21
2.6 Control mode according to time (a) and position (b). The solid bars represent the region of current while the dashed bars represent the location of waypoints 21

3.1 Collaborative Control Scheme for the AUV. 26
3.2 Results of simulation 31

4.1 Multiple robots must reach a set of destinations while avoiding obstacles and collisions with other robots, taking "riskier" paths between obstacles with human oversight when trusted to do so. 35
4.2 Plot of human performance when collaborating with a robot i. 40
4.3 Specification decomposition for two robots using A-G contracts. 44
4.4 (a) Safe robot motion planning in low trust scenario, and (b) advanced human motion planning in high trust scenario. 46
4.5 Collision avoidance scenario. 49
4.6 Progression of simulation (e - h) with corresponding path plans (a - d). . . . 50
4.7 Final paths of robots. 51
4.8 Trust evolution of robots. 51

Chapter 1

Introduction

Autonomy has made great strides over the history of robotics, dramatically decreasing physical and cognitive workload of operators and increasing task performance. This is especially prevalent in long-duration tasks such as search-and-rescue and reconnaissance. Despite these advances, however, automation has yet to surpass the adaptability and high-level cognitive reasoning of a human operator. A human operator can adapt and devise plans that are too complex or computationally expensive for autonomy to develop alone. However, as the human operator becomes fatigued, he or she is prone to mistakes. It is therefore desirable to devise novel methods of effective human-robot interaction (HRI) that take into account the strength of both autonomy (consistent and precise low-level control) and human operation (high-level cognitive planning and adaptability) by detecting scenarios difficult for autonomy and weighting that difficulty against an operator's abilities.

1.1 HRI in Control

There has been much work showing the benefits of HRI. For instance, [11, 39] show the benefit a human operator maintaining and assistively teleoperating a team of mobile

robots. Specifically in the case of [39] where human-robot teams are used to search for survivors in a search and rescue scenario, when the human operator assisted autonomy in low-level tasks such as teleoperation and identification, more survivors were found than in autonomy or manual control alone. In [31], an autonomous underwater vehicle (AUV) is used to search for mines. By having the operator take direct control only in navigation tasks too complex for autonomy, the operator could focus his or her attention on the data collection aspect, resulting in improved overall task performance. In [7], an HRI scheme is used in the control of a powered wheelchair where an autonomous system performed collision detection and avoidance, allowing the operator to solely focus on navigation.

HRI has not only shown benefits in low-level tasks. Robot controller design is usually hierarchical in nature with both high-level task and motion planning and low-level control law design. In [33], the authors study the effect of human involvement in operating teams of unmanned air vehicles (UAVs). Having the operator directly participate in all low-level tasks is shown to greatly increase operator workload when working with multiple robots. Removing the operator from the task completely, however, leads to poorer performance during automation failures. Having the operator actively give consent in a high-level supervisory role led to increased task performance and overall better workload levels of the operator. In [10], the authors give the scenario of an operator interacting with a swarm of autonomous vehicles in a patrol scenario. The authors present an adaptive control scheme where the level of interaction between human and swarm changes depending on the scenario, from defining search behaviors and locations during routine patrol tasks (high-level HRI) to controlling the swarm directly when an intruder is detected (low-level HRI).

Despite such interest in HRI for controlling autonomous robots, there appears to be a lack of work in incorporating human factors directly into the control scheme itself. In this body of works, novel control schemes are presented that incorporate not only the use of a human operator but also means of incorporating human factors (e.g. workload, trust, etc.)

into the controller itself. Doing so not only takes advantage of the performance aspects of autonomy and human control but considers when and how to best implement these aspects.

1.2 Trust in HRI

Human trust is an important factor to consider when designing and incorporating HRI. Trust is influenced by a variety of factors [19] that include robot performance characteristics, environmental characteristics, and human-related factors such as workload and prior experience. Trust is also highly influential in a human's acceptance and use of a robotic system. For example, [37] found that as trust in an adaptive cruise control system increased, brought about by the autonomy sharing its goals with the driver, acceptance of the autonomous system also increased. However, case studies have been conducted studying the misuse of automation, either through under-reliance or over-reliance, in railway and aircraft accidents [29]. From these studies it was found that trust in automation proved a major contributer to the human's decision to use (over-rely) or not use (under-rely) the automation in these accident scenarios. This, understandably, has led to an interest in quantifying trust and determining appropriate levels of trust for particular applications. For instance [8] uses trust as a variable in human-robot scheduling of multiple unmanned vehicles. In [32], operator's trust is determined by a function of prior reliability of the automation and is used to determine how much autonomy the system should implement. Trust has also been shown to be dynamic in nature and can change over the course of the interaction [42], leading to much interest in modeling this trust [42, 41] and using this dynamic trust to actively influence the interaction between the human and robot. By considering trust in HRI, especially in the case of interacting with multiple robots, the control scheme can better predict how a human operator will interact with the autonomy and assign tasks.

1.3 Overview

The works presented here provide novel additions to HRI at different levels of the controller hierarchy. Chapter 2[1] outlines a low-level HRI scheme where an optimal control policy is adapted to treat a human operator as a secondary mode to autonomy, embedding the human directly into the control scheme to determine when it is best to request human teleoperation in areas of difficult environmental disturbances. An extension of this work is given in Chapter 3 where the same optimal control policy is modified to incorporate sensor information to detect obstacles. Using the optimal control policy, the proposed method can determine when and if an operator needs to be notified to provide an updated path for high-level control to avoid the obstacle. Chapter 4[2] introduces a high-level HRI approach involving symbolic motion planning. Here, HRI is to switch between "safe", guaranteed, but inefficient autonomous motion planning and efficient, yet risky, human motion planning, demonstrated using a reconnaissance scenario. Trust is incorporated into the control scheme to aid in task assignment and decision making. Finally, conclusions regarding the work as a whole are presented in Chapter 5.

[1] This work has been accepted by the 2015 American Control Conference

[2] This work is currently under review for the 2016 IEEE Int. Conf. on Robotics and Automation

Chapter 2

SLQR Suboptimal Human-Robot Collaborative Guidance and Navigation for Autonomous Underwater Vehicles

2.1 Introduction

Despite advances in autonomous technology, humans and robots are often needed to collaborate and interact with each other in missions. For example, in [11] and [39], robot teams are maintained by a human operator with a focus on overall team performance. It is pointed out in [39] that the robot teams are able to search a wider area and identify a higher number of survivors when control is shared between robot and human in a search and rescue scenario than either full autonomous or full manual control alone. Often, the benefit of human-robot collaboration is an increase in task performance and a reduction of the human workload. In [7], Carlson et al. propose a collaborative control mechanism for a powered wheelchair that will help conduct collision avoidance, freeing up the operator for the more cognitive task of path planning. Extensions are also made for manual control of

Autonomous Underwater Vehicles (AUVs). In [31], AUVs are used to inspect harbors for mines. It is observed in [31] that by allowing the AUV to control movement the operator can focus on data processing. By only requiring operator input only when navigation tasks are difficult, the cognitive workload of the operator is decreased.

To tackle the issue of determining when a human operator should take control in collaborative human and robot navigation, we propose a control scheme based around the Switched Linear Quadratic Regulator (SLQR). The SLQR is an extension of the traditional Linear Quadratic Regular (LQR) problem that allows for the regulation of systems with multiple modes characterized by different dynamics or control inputs. One example of such a system is an automobile as it shifts between multiple gears. Research in the area of optimal control of systems with switchable modes using SLQR has achieved much attention [27, 46, 45, 47]. As compared to traditional LQR, SLQR finds the optimal control and switching sequence simultaneously. The primary issue in calculating the optimal solution to the SLQR problem is the need to account for every possible switching sequence, resulting in a control law that is very computationally heavy as the calculations of the Riccati equation grow exponentially. In [27], an offline method for calculating the optimal switching sequence and control input is proposed using a method that removes switching paths that are clearly suboptimal. In [46], Zhang et. al. introduce a method of reducing computational complexity of the discrete-time SLQR by removing branches of the Riccati mapping and keeping only solutions of the dynamic Riccati equation (DRE) that are significant. This is expounded upon in their later work [45] to produce a suboptimal, online method that can be extended into infinite horizon problems by repeatedly solving for reduced "equivalent" solution sets of the DRE. The SLQR has also been studied with regards to stochastic systems [47] and probabilistic switching [36] but such a case will not be considered here. In this paper, we use a modified version of the SLQR framework proposed by [45] as a means of optimal control that will produce switching between autonomous control and a manual

control model such that the design task is accomplished despite unexpected conditions with minimal human workload.

The organization of the rest of the chapter is as follows. Section 2.2 describes the derivation of the discrete-time linear dynamic model of the AUV. Section 2.3 outlines the control scheme implemented in this paper. Section 2.4 describes the simulations conducted in this paper and outlines the results with concluding remarks given in Section 2.5.

2.2 AUV Dynamic Modeling

The dynamics of an AUV can be described by the following form

$$\begin{aligned} M\dot{V} + C(V)V + D(V)V &= \tau(u, \alpha, \beta, rpm) \\ \dot{\eta} &= T(\eta)V \end{aligned}, \tag{2.1}$$

where M is the inertia matrix, $V = [u, v, w, p, q, r]^T$ are the linear and angular velocities, $C(V)$ is the matrix of Coriolis and centripetal terms, $D(V)$ is the damping matrix, τ is the vector of external forces and moments, u is the linear forward velocity of the AUV from V, α, β, and rpm are the horizontal and vertical fin angles and motor speed, respectively, $\eta = [x, y, z, \phi, \theta, \psi]^T$ are the inertia frame position and Euler angles, and $T(\eta)$ is a transformation matrix from the body-fixed frame to the inertia frame. The dynamics of the AUV were derived in the seminal paper [17] and have been utilized widely in robotics and control fields [44]. The equations provided in Eq (2.1), however, are highly nonlinear and coupled. Linearized equations of motion of AUVs are usually obtained for performance analysis and controller design [6]. Discretized linear models are also used [16].

Our objective is to guide the AUV through predesigned waypoints chosen for path planning. We assume that the AUV cruises between waypoints while maintaining a constant depth between each two waypoints. The AUV will adjust its yaw and depth after

reaching a waypoint to align with the next waypoint and depth requirement. Applications of such operations include maintaining a particular distance from the sea floor or taking samples at multiple depths between waypoints. Extension to varying depth between waypoints can be achieved simply by allowing the goal depth to be adjusted independently of the waypoints. Due to this style of guidance we can assume that the motion of the AUV consists of small deviations from a reference condition of steady cruise at a constant depth between every two waypoints. Therefore, the state variables are replaced by the reference states plus some small perturbations around the reference states, i.e., $\eta = \eta_0 + \Delta\eta$, $V = V_0 + \Delta V$.

The model is now linearized using small perturbation theory and Taylor series expansion neglecting terms with order greater than one. We choose the reference states as an equilibrium state of steady cruise, and hence the nominal accelerations $\dot{u}_0, \dot{v}_0, \dot{w}_0, \dot{p}_0, \dot{q}_0, \dot{r}_0$ and the nominal angular velocities p_0, q_0, r_0 are zero. Likewise, we set $\phi_0 = v_0 = \beta_0 = 0$ because the desired equilibrium state is to follow a straight forward path, and hence no lateral motion is expected. For the sake of simplicity the reference depth z_0 and yaw ψ_0 are also set to zero. Note that other nonzero values can be set and the rest of the analysis in the paper still holds. Finally, the values for $u_0, w_0, \theta_0, \alpha_0$, and rpm_0 were chosen such that the remaining constants are cancelled and the reference state trajectory can be obtained. In this paper, we consider an EcoMapper AUV model [38] and a nominal velocity u_0 is set to 2 m/s which is close to the maximum speed of the device. The equations of motion for the EcoMapper are given in Appendix A.

The linearized model of the AUV can now be described in the following discretized form

$$X_{k+1} = AX_k + BU_k \tag{2.2}$$

where A and B are the discretized state and input dynamics matrices, which can be found

in Appendix B, $X = [\Delta z, \Delta\theta, \Delta\psi, \Delta u, \Delta v, \Delta w, \Delta q, \Delta r]$ and $U = [\Delta\alpha, \Delta\beta, \Delta rpm]$. The states x and y are removed due to the ability to be controlled through the yaw angle ψ. The roll angle ϕ and roll velocity p are removed due to the EcoMapper's ability to self-stabilize its roll angle [38]. The choice of using the discrete-time model mainly arises from the digital nature of computer control as well as the means through which the control law will be derived. For this paper a time increment of 10 ms is used. The corresponding numerical values for the discretized state and input dynamics matrices are as follows

$$A = \begin{bmatrix} 1 & -.02 & 0 & 2.66*10^{-5} & 0 & .01 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0.1 \\ 0 & 0 & 0 & 0.99 & 0 & 0 & 7.34*10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 0.32 & 0 & 0 & -0.15 \\ 0 & 0 & 0 & 5.45*10^{-2} & 0 & 0.32 & 0.15 & 0 \\ 0 & 0 & 0 & 1.84*10^{-1} & 0 & -0.51 & .036 & 0 \\ 0 & 0 & 0 & 0 & 1.83 & 0 & 0 & 0.42 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -6.18*10^{-4} & 0 & 2.06*10^{-5} \\ 0 & -0.15 & 0 \\ -0.23 & 0 & 0 \\ -0.52 & 0 & 0 \\ 0 & -0.32 & 0 \end{bmatrix} \tag{2.3}$$

2.3 Collaborative Manual & Autonomous Motion Guidance Strategy

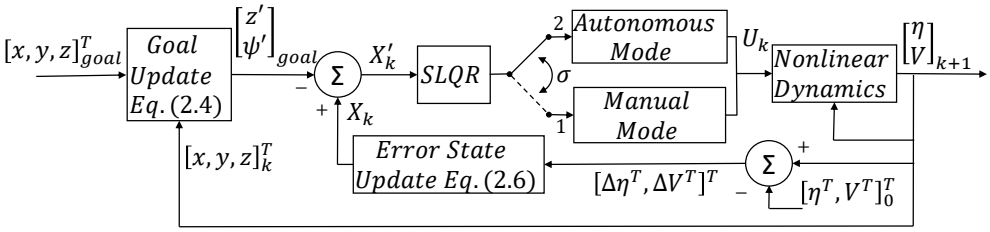

Figure 2.1: Collaborative Control Scheme for the AUV.

In this section, we propose a collaborative manual and autonomous motion guidance and navigation strategy in order to guide the AUV to desired waypoints while reducing human workload. We will design a suboptimal, online SLQR which will drive the linearized error system (2.2) to the goal state. The goal depth and heading can be altered by simply subtracting the goal depth and heading from the current depth and heading perceived by the AUV such that the desired state, $[z', \psi']^T_{goal}$, is the new zero state. As shown in Fig. 2.1, this is accomplished by inputting the error signal $\left[\Delta\eta^T, \Delta V^T\right]^T$, as defined by Eq. (2.5), into the error state update equation (Eq. (2.6)) to produce the state X_k. This state is then adjusted using Eq. (2.4) to align the zero state with the goal state, i.e $X'_k = X_k - [z'_{goal}, 0, \psi'_{goal}, 0, 0, 0, 0, 0]^T$. The SLQR is now implemented according to the dynamic equation given by Eq. (2.7) where A and B are the same as those used in Eq. (2.2). This holds since A and B are not dependent on the values of z and ψ. Depending on whether the manual or autonomous mode is chosen, the corresponding control law U_k will be calculated and substituted into the nonlinear dynamics (2.1) of the AUV to determine the states at the next time step $k+1$.

$$[z', \psi']^T_{goal} = [z_{goal}, tan^{-1}\left((x_{goal} - x_k)/(y_{goal} - y_k)\right)]^T \tag{2.4}$$

$$[\Delta\eta, \Delta V] = [\eta_k^T, V_k^T]^T - [\eta_0^T - V_0^T]^T \tag{2.5}$$

$$X_k = \left[\Delta\eta(3), \Delta\eta(5), \Delta\eta(6), \Delta V(1:3)^T, \Delta V(5:6)^T\right]^T \tag{2.6}$$

$$X'_{k+1} = AX'_k + BU_k \tag{2.7}$$

2.3.1 Linear Quadratic Regulator (LQR) for Nonlinear AUV Dynamics

To validate and study the effect of the SLQR control law for collaborative manual and autonomous strategy, as well as test the merits of the linearization, we first derive a suboptimal autonomous control scheme based on the traditional linear quadratic regulator (LQR). The goal is to have the AUV travel through multiple sets of complicated waypoints and depths autonomously. This scenario happens when there is no/little environmental disturbances and hence no need for human intervention. The LQR problem is solved using the typical discrete DRE with A and B, i.e., the state and input matrices, given by Eq. (2.3). The optimal input, U_k, as solved by the discrete LQR problem, is then fed back into the nonlinear AUV model (2.1) to calculate the actual change in state.

The results of the model verification are shown in Fig. 2.2 - 2.3a with suboptimal inputs given in Fig. 2.3b. The state and input weight matrices are set as $Q = diag([10,0,10,1,1,1,1,1])$ and $R = diag([1000,500,1])$. This choice of weights places the most emphasis on the depth and direction of the AUV while disregarding non-zero pitch angles, which would penalize the AUV changing depth.

Figure 2.2 shows the AUV trajectory as it moves in a three-dimensional figure 8

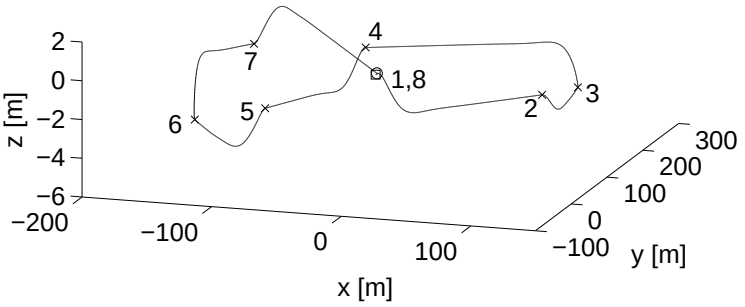

Figure 2.2: 3D results of autonomous LQR model verification

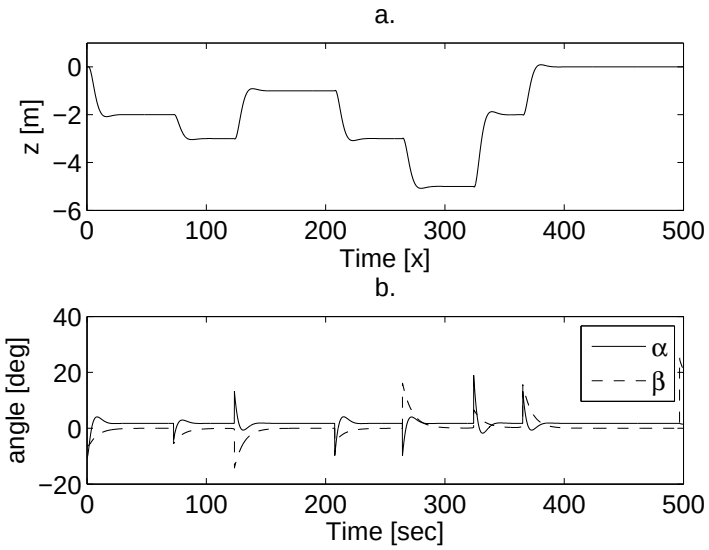

Figure 2.3: (a) AUV depth change under the LQR controller and (b) inputs α, β calculated by LQR.

pattern from the starting point (marked by a circle) to the end point (marked by a square) while passing through the desired waypoints (marked by x's). The depth profile z is shown in Figure 2.3a. The fin angles α and β (Figure 2.3b.) are the inputs into the AUV for controlling heading.

As shown in Fig. 2.2 and 2.3, the autonomous LQR control law derived from the linear model (Eq. (2.2)) is able to successfully drive the nonlinear dynamics (Eq. (2.1)) of the AUV to the desired waypoints while maintaining desired depths z (Fig. 2.3a) with forward velocity $u = 2$m/s under the fin angles α, β Fig. (2.3b) and motor speed $rpm =$ 1593rpm. For the purposes of this paper, the AUV is assumed to have successfully reached its goal if the AUV moves within a 5m radius of the desired waypoint.

2.3.2 Modeling Human System and Workload

The primary challenge with the utilization of a fully autonomous mode is that it lacks the capacity to accommodate higher level tasks that exceed the design expectations. Therefore, this paper proposes utilizing manual control as an ulterior "system" to which the autonomous mode can switch in situations beyond the autonomous controller's capability. To remain in this manual controlled mode, however, increases an operator's workload by requiring the operator to dedicate valuable man-hours to a single task. This can also cause cognitive fatigue if the operator must directly control the device for extended periods of time. Based on the LQR suboptimal control for autonomous mode developed in Section 2.3.1, here we propose a "switchable" human-robot collaborative system for the AUV guidance and navigation. The AUV will request human intervention only when higher level task management is required. This collaborative scheme will allow the AUV to achieve its task while minimizing the amount of operator workload as well as the total amount of time an operator must be dedicated to the control of one machine. This, in turn, allows the operator

to attend to multiple tasks, such as the monitoring of multiple robots in a similar method.

In order to utilize the SLQR problem for the control, the manual mode must first be modeled in a fashion similar to an LQR while maintaining the characteristics of a manual controller. To model the manual mode, the weights to the AUV system are first chosen as in the AUV LQR problem with $Q_1 = diag([10,0,10,1,1,1,1,1])$ and $R_1 = diag([1250,100,10])$. This reduces the cost on yaw input, representing a human operator's tendency to produce comparatively larger inputs to reach the goal state. Maintaining a precise depth and speed, however, would be difficult for a manual controller, suggesting an increase in cost for depth control, α, and motor speed, rpm.

To model the human workload the concept of the utilization ratio is used [35]. The utilization ratio, γ, is a numerical representation of perceived operator workload based on recent usage history. The value can range between zero, representing no recent usage and hence low workload, and one, representing complete usage and high workload of the operator. The dynamics of the utilization ratio is given by the discrete-time equation

$$\gamma_{k+1} = \left(1 - \frac{\Delta t}{\tau_\gamma}\right)\gamma_k + \frac{\Delta t}{\tau_\gamma} b_\sigma \tag{2.8}$$

where b_σ is either 1 or 0 depending on if the operator is or is not being utilized, respectively, Δt is the time step of discretization, and τ_γ represents sensitivity of the operator to the recent history with smaller values corresponding to an increased rate of change of γ. For the purposes of simulation, a value of $\tau_\gamma = 500$ was chosen as it will reach full utilization in approximately 1000 sec. Actual implementation can customize this value to a specific user. This equation corresponds to an increase in the utilization ratio during operator control and a decrease during autonomous control. In this work, the utilization ratio will be added to the cost function as part of the overall system dynamics to allow large operator workload, indicated by a high utilization ratio, to influence the switching dynamics

by correspondingly increasing the cost of manual mode. The dynamics of the utilization used in conjunction with the AUV dynamics are hereby represented in discrete linear form as follows

$$\begin{bmatrix} \gamma_{k+1} \\ 1 \end{bmatrix} = \begin{bmatrix} \left(1 - \frac{dt}{\tau_\gamma}\right) & \frac{dt}{\tau_\gamma} b_\sigma \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \gamma_k \\ 1 \end{bmatrix} \tag{2.9}$$

2.3.3 Switched Linear Quadratic Regulator (SLQR) for Human-Robot Collaborative AUV Guidance and Navigation

To model this "switched" human-robot collaborative operation of the AUV, a modification of the Switched Linear Quadratic Regulator (SLQR) will be implemented. SLQR is an expansion of the traditional LQR problem [36] that accounts for systems that contain multiple modes. The SLQR control law to be developed in this paper differs from the online SLQR problem presented by [45] in that our goal is to minimize a cost function around dynamics involving non-switching AUV dynamics, as opposed to the general switching dynamics in [45], as well as the addition of an operator workload parameter γ that influences the switching between manual and autonomous controllers. We also seek to use the SLQR to find a compromise between the workload of the operator, as defined above, while still completing the navigation task. As such, switching will be dictated by the AUV's effectiveness to complete a task autonomously, invoking a request for manual control when the controller deems it optimal to do so.

We associate the SLQR collaborative problem with the following quadratic cost function:

$$J(Z,U,\sigma) = Z_N^T Q_f Z_N + \sum_{k=0}^{N-1} Z_k^T Q_{\sigma(k)} Z_k + U_k^T R_{\sigma(k)} U_k \tag{2.10}$$

where $Z_k = \left[X_k^T, \gamma_k, 1\right]^T$ is the compound state vector, $\sigma = 1$ denotes the manual control mode and $\sigma = 2$ denotes the autonomous mode (as shown in Fig. 2.1), Q_f is the terminal

state cost weighting matrix and $Q_\sigma \succeq 0$ and $R_\sigma \succ 0$ are the chosen symmetric state and input cost weighting matrices respectively. As shown by [45] the solution can be found by applying the well-known discrete DRE, denoted here as $\rho_\sigma(P) : \mathbb{R}^{nxn} \to \mathbb{R}^{nxn}$, recursively in time with regards to the optimal mode σ. The mapping is shown in Eq. (2.11).

$$\rho_\sigma(P) = Q_\sigma + A_\sigma^T P A_\sigma - A_\sigma^T P B_\sigma (R_\sigma + B_\sigma^T P B_\sigma)^{-1} B_\sigma^T P A_\sigma \tag{2.11}$$

Due to the recursive nature of solving for the Ricatti mappings it is unknown prior to implementation which switching sequence will produce the optimal control sequence. Therefore, every possible switching sequence must be considered in the calculation of the Riccati mappings prior to actual implementation of the control scheme. To accommodate this uncertainty in switching, denote the set of all Riccati mappings moving from time $k+1$ to time k as $\mathscr{H}_k$, called the Switched Riccati Set (SRS) at time k. The sequence of these sets $\{\mathscr{H}_k\}_{k=N}^0$ are generated iteratively backwards in time in accordance to Eq. (2.12)

$$\mathscr{H}_k = \rho_{\mathbb{M}}(\mathscr{H}_{k+1}) = \{\rho_\sigma(P) : \text{for } \sigma = 1,2 \textit{ and } P \in \mathscr{H}_{k+1}\}. \tag{2.12}$$

where ρ_σ is the discrete DRE defined in Eq. (2.11).

Once all the Riccati sets have been computed offline and before implementation, the optimal mode and Riccati mapping can be determined online by solving the value function for the SLQR problem at each time step k according to the following equation.

$$V_k(Z_k) = \min_{P \in \mathscr{H}_k} Z_k^T P Z_k. \tag{2.13}$$

From Eq. (2.13), the optimal mode, σ, associated with the optimal mapping, $P \in \mathscr{H}_k$, can be determined. If the optimal mode is such that $\sigma = 1$, the control law will then make a request to the operator to engage in manual control. Otherwise, the AUV will continue in

autonomous control with the optimal input determined by the equation

$$U_k = -\left(R_\sigma + B_\sigma^T P B_\sigma\right)^{-1} B_\sigma^T P A_\sigma Z_k = -K_k Z_k \tag{2.14}$$

One noteworthy issue with this method is that when Eq. (2.13) reaches a value such that the control method deems it optimal to switch, the possibility exists that the value function will cause the mode to switch multiple times rapidly before settling on the true optimal mode. This type of switching may be acceptable in some completely autonomous systems, but systems requiring human interaction require human response time to be taken into account. To counteract this, an additional term $Q_\xi \xi$ is added to the cost function of the opposing mode where Q_ξ is a predetermined gain. Let ξ be a number between 0 and 1 with dynamics described by the equation

$$\xi_k = 1 - \frac{\Delta t}{T_\xi} \tag{2.15}$$

where $\Delta t \in [0, T_\xi]$ represents the time elapsed since the last mode switch and T_ξ is a predetermined time before allowing ξ to equal zero. This latter term should be chosen large enough such that rapid switching should be negated. This method was chosen over a strict time requirement as this method will still allow switching to manual mode in cases where the cost to stay in autonomous mode far outweighs the cost of manual mode.

A second issue with this approach can be clearly seen from the fact that the sets defined in Eq. (2.12) grow exponentially in size. To make the calculations computationally feasible, all matrices that can be considered algebraically redundant according to Lemma 1 in [45] can be removed during the computing of these sets without affecting the value function. Furthermore, Zhang, et al. show that an error term ε can be added to the tested matrix to further increase computational feasibility while proving that the effect on the

optimality of the solution is bounded.

The above mentioned Lemma can be solved using convex optimization algorithms. For the purposes of this paper, we utilize the DSDP semidefinite programming solver [4] offered through the free OPTimization Interface (OPTI) Toolbox for MATLAB [12] with an $\varepsilon - \textit{redundancy}$ value of $\varepsilon = 0.1$. For the following simulation, this resulted in the offline calculations of the SRSs being reduced to approximately 30 matrices at each time step. For instances where the reduced set is large, or for very large horizons, N, [45] shows that a "divide-and-conquer" approach can be used while still achieving an arbitrary suboptimal performance. The control method can now be summarized in Algorithm 1.

Compute SRSs offline according to Eq. (2.12) ;
while *AUV is operating* **do**
  Compare Current State with Goal State;
  Determine $P \in \mathscr{H}_k$ and σ that solves Eq. (2.13);
  if $\sigma = 1$ **then**
    Send request for manual input from operator;
  else
    Calculate input according to Eq. (2.14);
  end
  Implement control;
end

Algorithm 1: Implementation of SLQR-based collaborative optimal control

2.4 Simulations Results

Matlab simulations are now conducted to test the performance of the human-robot collaborative guidance and navigation strategy (the codes used for this simulation can be found at `http://people.clemson.edu/~yue6/papers/thesis/DAS_Thesis_Codes.pdf`). The simulations consist of the AUV following a set of 8 waypoints spaced 75m apart along the x-axis and arranged in a semi-circular path of radius 300m, as shown in Fig. 2.4. The AUV will start at the position $(-300, 0)$ with $u = 0$ m/s and ψ facing along the y-axis.

Table 2.1: Deviation from desired waypoints in meters

Mode	Point 1	Point 2	Point 3	Point 4	Point 5	Point 6	Point 7	Point 8	Avg.
Autonomous	0.16 m	1.78 m	3.33 m	5.84 m	NA	NA	NA	NA	2.78 m*
Manual	0.19 m	1.20 m	3.14 m	4.89 m	2.93 m	0.58 m	0.66 m	4.97 m	2.32 m
Collaborative	0.18 m	1.79 m	3.24 m	4.89 m	2.90 m	0.83 m	0.65 m	4.97 m	2.43 m

*Average of autonomous mode includes the first four points only.

Table 2.2: Statistical results of varying disturbance parameters

Mode	Point 1	Point 2	Point 3	Point 4	Point 5	Point 6	Point 7	Point 8
Mean	0.54 m	2.71 m	1.61 m	2.04 m	1.69 m	1.74 m	1.11 m	4.97 m
St. Dev.	0.58 m	1.49 m	1.25 m	1.51 m	1.37 m	1.37 m	0.78 m	0.01 m

The AUV must come within a radius of 5m of the desired waypoint before it is allowed to move to the next waypoint. The simulation environment consists of a nominal, still body of water for which the autonomous controller was designed. In the center of the simulation environment, a cross current of 0.3 m/s along the positive y-axis is introduced across a span of 200 m. This current simulates an unexpected change in the conditions that would benefit from operator intervention. The three scenarios that will be tested and compared are autonomous mode, manual mode, and human-robot collaborative mode where the AUV is controlled autonomously until conditions are such that human intervention is determined by the cost function as optimal.

The results of the simulations for the autonomous mode and the human-robot collaborative mode are shown in Figs. 2.4 and 2.5 respectively. From Fig. 2.4, it can be seen that the autonomous mode is not sufficient to successfully reach point 4 in the midst of the cross current, causing the AUV to become stuck around this point. As expected, this issue is not present under fully manual control. This manual control scenario, however, is labor intensive, requiring 470 sec of the operator's full attention for the entire duration of the task and a final utility ratio of 0.61 under an operator sensitivity of $\tau_\gamma = 500$.

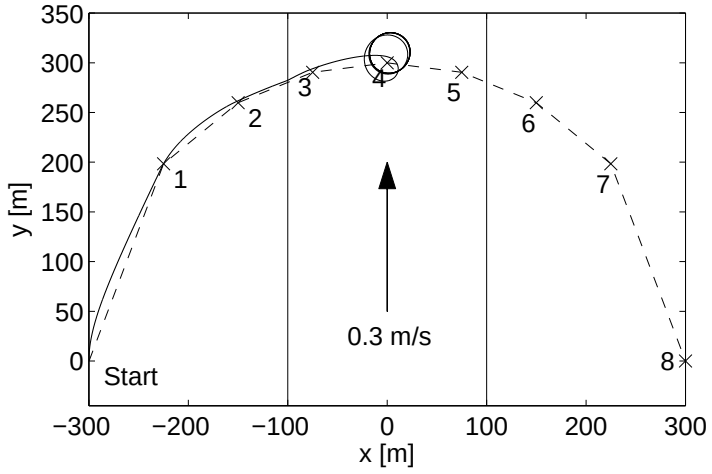

Figure 2.4: Results of simulation under autonomous mode

The simulation results of the cooperative control scheme are shown in Fig. 2.5 with the resulting mode switching scheme shown in Fig. 2.6 against both time (a) and position (b). Viewing Fig. 2.6b in conjunction with Fig. 2.5 it can be seen that the AUV begins in manual mode due to the large deviation from the direct path at the beginning of the simulation. As the AUV progresses and the path converges autonomous is deemed optimal, leading the AUV to switch modes. Almost immediately after reaching waypoint 1, however, the AUV shoots past the waypoint, causing a large deviation in the path. This overrides the switching buffer, outlined by Eq. (2.15) leading to a brief change in mode to compensate for the overshoot. The AUV then returns to autonomous mode until entering the disturbance region. Inside this region, the cost function dictates manual operation to be optimal, as desired, until just before the end of the disturbance region. The AUV then returns to autonomous mode until shooting past the next to last waypoint, leading to a switch to manual mode before finishing in autonomy.

The collaborative control method successfully navigated the AUV through the series of waypoints in 471 sec, of which the operator was engaged for 268 sec. This corresponds to a 43% reduction in engagement time. The maximum utilization ratio reached by

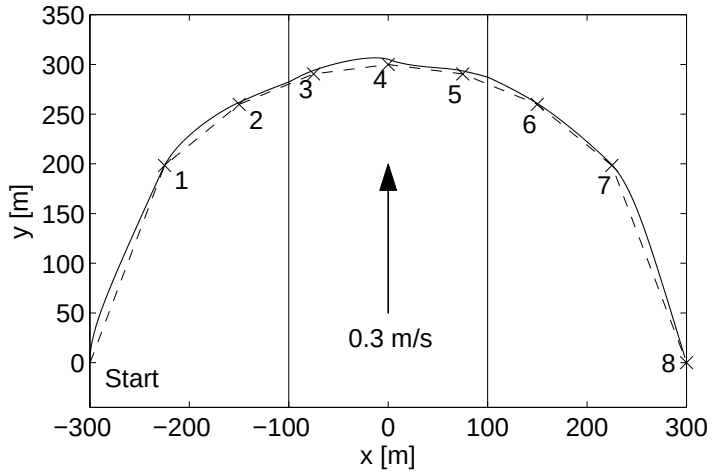

Figure 2.5: Results of simulation under collaborative mode

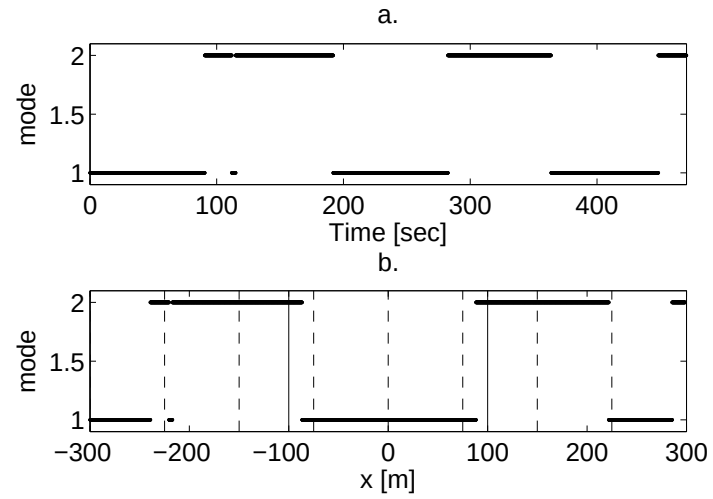

Figure 2.6: Control mode according to time (a) and position (b). The solid bars represent the region of current while the dashed bars represent the location of waypoints

the operator at any time during the simulation is also 0.34, much less than that of the fully manual mode. From Table 2.1 it can be seen that the collaborative control scheme achieved performance comparable to the fully manual mode despite the decrease in utilization time.

To test robustness of this method, further tests were conducted using variations of the disturbance parameters. The cross current was implemented using varying widths of 100m, 200m, and 300m centered around -150m, 0m, or 150m on the x-axis. The strength of the current was varied as 0.1m/s, 0.2m/s, or 0.3 m/s along the positive y-axis. The average distance and standard deviation of each point for the 27 simulations is listed in Table 2.2 with the averages comparable to the fully manual mode. The method also has an average 57% operator engagement time with standard deviation of 3.4% engagement time and an average maximum utilization ratio of 0.36 with standard deviation as 0.02. This shows this method has comparable performance for total average deviation to fully manual mode while reducing workload, despite the variation in the unknown disturbance. It can be seen that in Table 2.2 the mean deviation about point 3 is much lower than point 2, which is the opposite of what is seen in Table 2.1. This is likely due to the scenario in Table 2.1 having waypoint 3 almost immediately at the beginning of the cross current, allowing less time to correct the path. In the variations method in Table 2.2, waypoint 2 is inside the disturbance region more often than waypoint 3 as a result of one of the variations being the current centered about -150m, leading to a greater tendency to deviate from the intended path.

2.5 Conclusions

In this paper, a SLQR based optimal controller has been designed for human-robot collaborative tasks. A linearized model of the highly nonlinear dynamics of an AUV has been created and used in simulations to characterize the performance of this new collaborative control scheme. It was found that under the simulation environment the autonomous

LQR was unable to guide the AUV successfully to each waypoint under conditions beyond the design of the controller. By implementing the SLQR based control scheme the controller successfully guided the AUV to each waypoint while under the control of the operator 57% of the total time. The proposed controller also significantly reduced the workload experienced by the operator compared to the scenario of the AUV being guided manually for the entire mission.

Chapter 3

AUV Suboptimal Switching Between Waypoint Following and Obstacle Avoidance in Human-Robot Collaborative Guidance and Navigation

3.1 Introduction

There are many applications that involve autonomous underwater vehicles (AUVs) exploring unknown environments, such as data collection, surveying, shipwreck exploration, and mine detection. Autonomous control has made great strides in allowing autonomous vehicles to conduct these types of missions without the need of human interaction. However, in complex, dynamic, and uncertain environments, it is very likely there will be scenarios where the autonomy is unable to accommodate. In such uncertain environments, humans still have the advantage over autonomy via adaptability and higher level cognitive reasoning. One such scenario, the case of unknown environmental disturbances,

was explored in Chapter 2. There, the SLQR optimal control policy was used as a means of determining when it is best to request manual control to overcome such disturbances in a way that balances mission effectiveness and operator workload. While the work in Section 2 provides a novel way of using human-robot interaction (HRI) to overcome disturbances, there are scenarios that that method still cannot overcome.

In the normal situation, an AUV follows waypoints that are predefined to finish the navigation task. However, many of the before-mentioned applications involve scenarios where obstacles can block an AUV's designated path, especially in mine detection and shipwreck exploration where discovery of such obstacles is the goal. In such scenarios it is required that the AUV navigates around such obstacles while still making progress towards its remaining search waypoints. Despite advances in autonomous guidance mechanisms, however, such operations, especially in the presence of complex and uncertain environments, are still better suited for human guidance due to human adaptability and response flexibility [28]. For example, [28] emphasizes that it is common for rotorcraft to operate in terrain that lacks obvious structure and a priori knowledge, making it difficult for autonomous systems to plan and navigate. A skilled operator is able to study the environment while simultaneously flying the aircraft and planning his or her course of action, though at the cost of increased workload. Therefore, it is desired to devise a means of incorporating these abilities into the control of an AUV and to determine in a real-time fashion when it is best to utilize the human operator.

To incorporate human path planning abilities we expand upon the work presented in Chapter 2, creating a modification of the SLQR suboptimal human-robot collaborative control scheme to accommodate the scenarios where path changes are required. In this scenario, an LQR provides the optimal input for the AUV under its normal operating condition. Should an unexpected obstacle appear along the path, the SLQR will use sensor information to determine when it is best to stop and request operator assistance in the form

of path replanning. The operator will then assess the scenario, provide a revised path that efficiently circumvents the obstacle and allow the AUV to resume its task.

3.2 Modification of SLQR

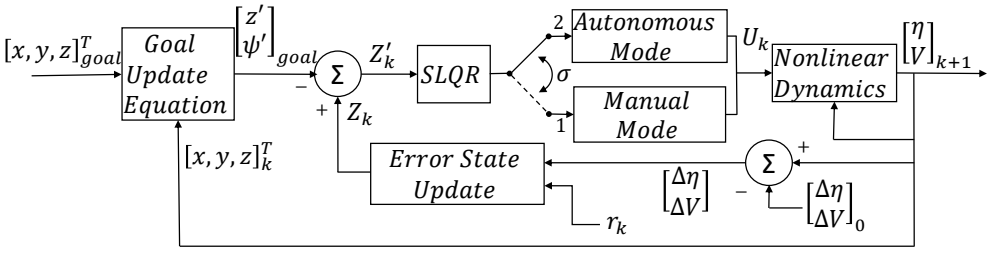

Figure 3.1: Collaborative Control Scheme for the AUV.

In order to integrate a human operator's path planning abilities, we first need to incorporate obstacle detection into the control scheme. To begin, the control scheme presented in Section 2.3 is modified to include the sensor reading r. This sensor reading detects and measures the distance of obstacles in front of the AUV within some specified sensing range r_S. When there are no obstacles within sensing range, an autonomous LQR controller is used to navigate the AUV. If an obstacle is detected, the updated control scheme, shown in Fig. 3.1, is engaged. The new error state update equation is now given as Eq. (3.1) with added sensor dynamics given in Eq. (3.2).

$$Z_k = \left[X_k^T, r_k\right]^T = \left[\Delta\eta(3), \Delta\eta(5), \Delta\eta(6), \Delta V(1:3)^T, \Delta V(5:6)^T, r_k\right]^T \tag{3.1}$$

$$r_{k+1} = r_k + u_k \Delta t \tag{3.2}$$

where $\Delta\eta$, ΔV, and X_k are defined as in Chapter 2. Here, r represents the sensor reading of the negative distance ($r_k \leq 0$) between the AUV and the detected obstacle. The sensor dynamics for autonomy when an obstacle is detected within its sensing range are modeled intuitively as approaching zero at the rate of forward velocity, u_k. The dynamic system is now given as

$$Z'_{k+1} = AZ'_k + BU_k \tag{3.3}$$

with A and B being the combined AUV and sensor state and input matrices, respectively, and Z' being the AUV states and sensor reading shifted according to the goal depth, heading, and sensor range, i.e. $Z'_k = Z_k - [z'_{goal}, 0, \psi'_{goal}, 0, 0, 0, 0, 0, -r_S]^T$ with z'_{goal} and ψ'_{goal} as defined in Chapter 2.

The cost function, set up similarly as in Chapter 2 using the new dynamics, is now given as

$$J(Z', U, \sigma) = {Z'}_N^T Q_f Z'_N + \sum_{k=0}^{N-1} {Z'}_k^T Q_{\sigma(k)} Z'_k + U_k^T R_{\sigma(k)} U_k \tag{3.4}$$

where $\sigma = 1$ denotes the manual control mode and $\sigma = 2$ denotes the autonomous mode (as shown in Fig. 3.1), Q_f is the terminal state cost weighting matrix and $Q_\sigma \succeq 0$ and $R_\sigma \succ 0$ are the chosen symmetric state and input cost weighting matrices respectively for the corresponding mode. This cost function differs from that in Chapter 2 by the addition of the sensor reading. The additional cost can be explicitly stated as

$$J_{r_k} = Q_{r_\sigma} {r'}_k^2 = Q_{r_\sigma} (r_k + r_S)^2 \tag{3.5}$$

where r' is the shifted sensor reading in Z' and Q_{r_σ} is the portion of Q_σ assigning weight to the sensor reading. This equation assigns added cost to moving closer to an obstacle via a positive finite weight with the goal being to have no obstacles within sensing range.

Therefore as r_k approaches zero, cost increases.

It is important to note from Eq. (3.3) that since the sensor and AUV dynamics are the same for manual and autonomy, the dynamics of the system is non-switching. Therefore, the weights Q_σ and R_σ must be chosen as representative of the advantages given by the two control modes for effective switching to occur. To begin, as the difference between the two modes is characterized by the detection of an obstacle, the weights on the AUV states, Q_X, is chosen to be the same for both manual and autonomous modes. To characterize the difference between manual and autonomy, different weights are given to Q_{r_σ}. For autonomy, this weight is chosen as a finite positive gain determined by assessing the trade-offs between the risk of colliding with the obstacle and reaching the next waypoint without detour; e.g., mine detection will have a higher value for Q_{r_σ} than shipwreck exploration. For manual, this weight is set to zero to represent the operators ability to plan around the obstacle. The final state weight matrices used in this work are now given as

$$Q_X = diag([10,0,10,1,1,1,1,1]) \tag{3.6}$$

$$Q_{r_\sigma} = \begin{cases} 0 & \sigma = 1 \\ 10 & \sigma = 2 \end{cases} \tag{3.7}$$

$$Q_\sigma = \begin{bmatrix} Q_X & 0 \\ 0 & Q_{r_\sigma} \end{bmatrix} \tag{3.8}$$

To characterize the two modes regarding the input weights, weights are first chosen for autonomy. For this application, we chose $R_2 = diag([50,50,1])$. Unlike in Chapter 2, the operator does not directly control the AUV, but instead provides a replanned path. To characterize this, added weight is assigned to the inputs for manual mode, representing the

AUV having to stop while it waits for an updated path. For this work, this added weight was chosen as $R_1 = 10R_2$.

The derivation of the SLQR is the same as in Chapter 2. Like in the previous method, U in autonomy ($\sigma = 2$) is calculated using the optimal input. In this case, however, manual mode does not constitute a request for operator control, but for the operator to give planning assistance. Therefore, when manual mode is chosen by the SLQR, the AUV will stop and send the information to the operator, who will process the information and provide the AUV with an updated path. Once the AUV receives the updated plan, it returns to autonomy and implements that plan until a new obstacle is sensed or all objectives are met. The new scheme can now be summarized in Algorithm 2.

$$U_k = \begin{cases} 0,\ \textit{Request path update}, & \sigma = 1 \\ -\left(R_\sigma + B^T P B\right)^{-1} B^T P A Z_k = -K_k Z_k, & \sigma = 2 \end{cases} \tag{3.9}$$

Compute SRSs offline according to Eq. (2.12) ;
while *AUV is operating* **do**
    Compare Current State and Sensor Data with Goal State;
    Determine $P \in \mathscr{H}_k$ and σ that solves Eq. (2.13);
    if $\sigma = 1$ **then**
        Stop motion and send request for path update from operator;
        Operator sends updated path;
        Return to Autonomous mode ($\sigma = 2$);
    else
        Calculate input according to Eq. (2.14);
        Implement control input;
    end
end

Algorithm 2: Extension of SLQR-based collaborative optimal control

3.3 Simulation

A Matlab simulation is now conducted to demonstrate the performance of the modified human-robot guidance and navigation strategy (the codes used for this simulation can be found at `http://people.clemson.edu/~yue6/papers/thesis/DAS_Thesis_Codes.pdf`). The simulation consists of the AUV following a set of 4 waypoints spaced 150m apart along the x-axis and arranged in a semi-circular path of radius 300m, as shown in Fig. 3.2a. The AUV will start at the position (-300,0) with $u = 0m/s$ and ψ facing along the y-axis. The AUV must come within a radius of 5m of the desired waypoint before it is allowed to move to the next waypoint. the simulation environment consists of a nominal, still body of water for which the autonomous controller was designed. However, throughout the environment are obstacles that are unknown before deployment and must be circumvented for the AUV to successfully complete its mission. These obstacles are indicated by the boxes shown in Fig. 3.2. To detect these obstacles, the AUV is equipped with a front facing sensor with range of $r_S = 25m$. The goal of the autonomy is to drive the AUV to the desired waypoints while maintaining distance between the AUV and any sensed obstacles.

The simulation results of this control scheme are shown in Fig. 3.2a-d. The AUV begins in autonomy heading towards waypoint 1. However, as the AUV progresses it senses an obstacle in the path. This sensor value is now provided to the SLQR to make the decision using the given weights to determine if and when the operator should be notified and a path update be requested. Here, the sensor determines to send the request approximately 20m from the obstacle, at which point the AUV stops its motion and requests an updated path from the operator. The operator supplies an additional waypoint to the AUV marked by the triangle in Fig. 3.2. This process is repeated in Fig. 3.2b. as the AUV detects a second obstacle after waypoint 2. The operator returns a new path denoted by the triangle and

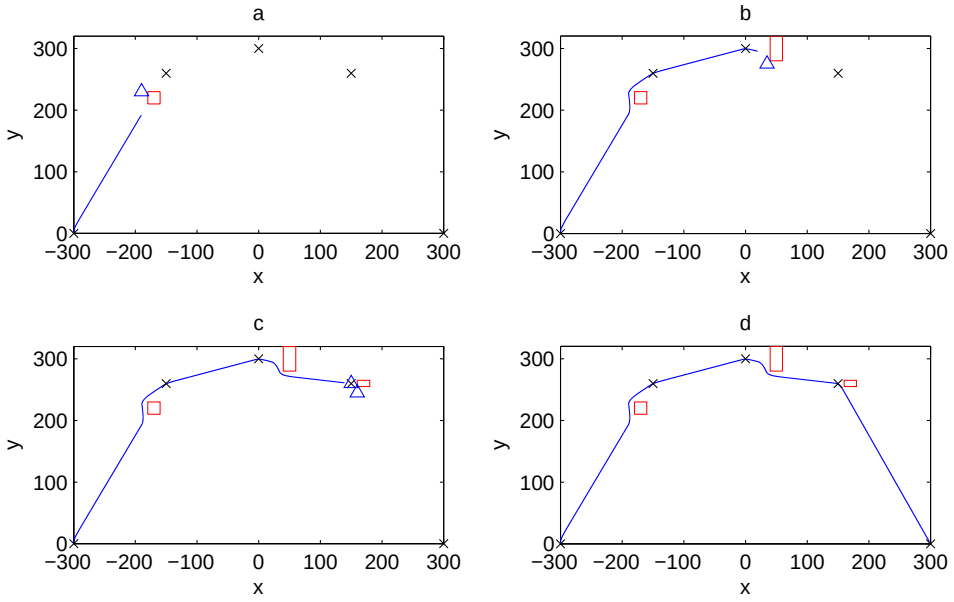

Figure 3.2: Results of simulation

the AUV successfully uses this new waypoint to circumvent the obstacle. In Fig. 3.2c. the AUV detects a third obstacle very close to the desired waypoint. To ensure the AUV reaches the desired waypoint while simultaneously avoiding the obstacle, the operator gives a series of waypoints to ensure the AUV can maneuver around the obstacle without missing the waypoint. The AUV then executes this path and finishes the route successfully in Fig. 3.2d.

3.4 Conclusion

In this work, a modification of the SLQR-based human-robot collaborative control scheme was created for the objective of obstacle avoidance. By using sensor readings and modifying the cost function, this method was able to take advantage of autonomy's low-level task performance and an operator's higher level path planning capabilities. The method was successfully applied to the highly nonlinear dynamics of an AUV and was

shown through simulation to successful accomplish the overall task with minimal human input.

Chapter 4

Trust-Based Human-Robot Interaction for Safe and Scalable Multi-Robot Symbolic Motion Planning

4.1 Introduction

Despite advances in autonomy for robotic systems, human collaboration is often still necessary to ensure safe and efficient operation. When designing robotic systems, it is thus important to consider factors related to human-robot interaction (HRI) [18]. However, development of effective schemes for HRI remains a challenge, especially for systems in which a single human must interact with multiple robots.

An important factor to consider with respect to HRI is human trust. Trust is a dynamic feature of HRI [26] that heavily affects a human's acceptance and use of a robotic system [19]. Consideration of trust is especially important in systems that require human supervisory control of multiple robots, since supervisory tasks must be carefully allocated to ensure time-critical issues are addressed while human workload is kept within acceptable

bounds [33].

Other important factors to consider include safety and performance. While many advances have been made in HRI, extant works in this area often lack quantitative models and analytical approaches that could be used to provide safety and performance guarantees. Some progress in addressing this deficiency has been made through the application of formal methods – i.e. mathematically-based tools and techniques for system specification, design, and verification [2] – to problems involving HRI [5], including symbolic robot motion planning [3]. In symbolic motion planning, a set of specifications for the robots is given, e.g. "go to locations A and B while avoiding obstacles," and plans are generated in a discretized representation of the workspace. These plans are then converted to reference trajectories and control laws in the continuous workspace such that the specification is satisfied in the discrete workspace. Though advances have been made in this area, challenges remain in addressing the scalability of these approaches for multi-robot systems and in incorporating models of human behavior to improve joint human-robot performance [14].

Here, we investigate methods for improving the scalability, safety, and performance of symbolic motion planning for multi-robot systems, taking into account the effects of human trust. Specifically, we explore (1) trust-based specification decomposition to address scalability, and (2) real-time trust-based switching between human and robot motion planning to address other concerns related to safety and performance. We explore these methods in the context of a multi-robot intelligence, surveillance, and reconnaissance (ISR) scenario.

The remainder of this chapter is organized as follows. Section 4.2 introduces symbolic motion planning and a computational model of dynamic trust. Section 4.3 outlines the method for trust-based specification decomposition, and Section 4.4 describes the method for implementing switching between human and robot control during plan execution. A simulation that demonstrates these methods is presented in Section 4.5, with concluding

remarks given in Section 4.6.

4.2 Human-Robot Interaction for Symbolic Motion Planning

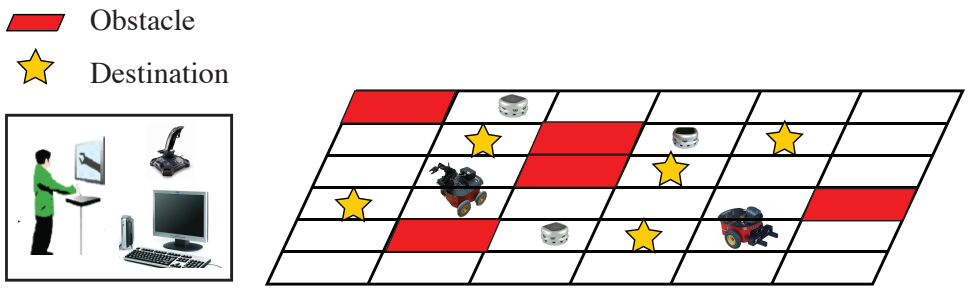

Figure 4.1: Multiple robots must reach a set of destinations while avoiding obstacles and collisions with other robots, taking "riskier" paths between obstacles with human oversight when trusted to do so.

We consider an ISR scenario in which a team of robots, supervised and potentially assisted by a human operator, must reach a set of goal destinations while avoiding collisions with stationary obstacles and with each other, as shown in Fig. 4.1. As is standard in symbolic motion planning problems, the workspace is discretized into polytopic regions that are labeled with relevant properties, e.g. whether they contain an obstacle or goal. Note this discretization can be performed to an arbitrary degree of accuracy; however, increasing the number of regions significantly increases the computational complexity of the planning problem. Most discretizations therefore significantly overapproximate certain features of the workspace, e.g. an obstacle may only take up a small portion of an "obstacle" region. The result is that planning through the workspace may be overly conservative, since "risker" paths that go through regions containing obstacles might be feasible in the continuous workspace.

In this scenario, we assume a set of goal destinations is given at the start, and each goal must be reached by at least one robot while collisions with obstacles and between robots are avoided. This set of requirements forms a formal specification for the scenario. To reduce computational complexity of the multi-robot planning problem, compositional reasoning approaches are used to decompose this specification. More specifically, the specification is decomposed such that each robot is assigned a subset of the goal destinations and individually synthesizes a plan to reach its assigned goals. Potential collisions are then handled locally as they are detected, with the involved robots implementing a collision avoidance protocol that requires synthesizing modified plans through collaboration between the robots. In this sense, the proposed planning scheme is implemented in a distributed manner. In addition, we assume obstacle locations are not known a priori; therefore, robots also synthesize new plans when they encounter obstacles, whose locations are shared with other robots when they come into communication range.

Throughout the scenario, a quantitative and dynamic trust model based on robot performance, human performance, and the environment is used to estimate human trust in each of the robots [34, 40]. This estimate of trust affects the specification decomposition, with more trusted robots assigned more destinations. Trust is also used to determine when the robot should suggest navigating between obstacles, as this requires real-time switching between human and robot motion planning, with the human planning the path between obstacles. Human consent for this switching is assumed to depend on current trust as well as whether or not the human is currently occupied with other tasks. Each robot is assumed to follow a simple first-order kinematic equation of motion

$$\dot{x}_i = u_i,\ i = 1, 2, \cdots, N \tag{4.1}$$

where N is the total number of robots.

4.2.1 Symbolic Motion Planning

In symbolic motion planning, the workspace is discretized into a set of regions or states S. This discretized workspace is often represented as a transition system. The definition of a transition system is as follows.

Definition 1 (Labeled Transition System) *A finite labeled transition system is a tuple* $TS = (S, R, s_0, \Pi, L)$ *consisting of (i) a finite set of states S, (ii) a transition relation* $R \subseteq S \times S$*, (iii) an initial state* $s_0 \in S$*, (iv) a set of atomic propositions* Π*, (v) and a labeling function* $L : S \to 2^{\Pi}$.

Let us define each robot's motion be described by the transition system $TS = (S, R, s_0, \Pi, L)$ and having a path as an infinite sequence of states $\sigma = s_0 s_1 s_2 \ldots$, where $s_i \in S$ is the robot's state at time $i = 0, 1, \ldots$ and pairs of sequential states ($s_i \to s_{i+1} \in R$) represent feasible transitions between states, i.e. direct transitions between states that are achievable in the continuous workspace. Each state will be labeled with all atomic propositions from the set Π that are currently true. For each path, a trace is then a corresponding infinite sequence of labels $L(\sigma) = L(s_0)L(s_1)L(s_2)\ldots$, where $L : S \to 2^{\Pi}$ is the mapping from states to atomic propositions.

Discretization of the workspace enables formalization of plan specifications in discrete logics such as linear temporal logic (LTL) [2]. LTL extends propositional logic – which has operators $\wedge$ "and," $\neg$ "not," $\vee$ "or," $\to$ "implies," etc. – with temporal operators such as $\Diamond$ "eventually" and $\Box$ "always." With respect to temporal operators, we restrict our attention to terms of the form $\Box p$, $\Diamond p$, and $\Diamond\Box p$. The term $\Box p$ is true for a trace if propositional formula p is true in every state of the trace, $\Diamond p$ is true if p is true in some state of the trace, and $\Diamond\Box p$ is true if p is true in some state of the trace and all states thereafter.

Given an LTL plan specification φ and a transition system TS that encodes all possible state transitions in the workspace, a plan that satisfies the specifications can be syn-

thesized using a model checking approach. Traditionally, a model checker verifies whether a system TS satisfies a specification φ, written $TS \models \varphi$. If not, it returns a counterexample trace $L(\sigma) \in traces(TS)$ that does not satisfy the specification, i.e. $L(\sigma) \not\models \varphi$. Model checking TS against the negation of the specifications returns a trace that satisfies the specifications, since $L(\sigma) \not\models \neg\varphi \rightarrow L(\sigma) \models \varphi$. For symbolic motion planning, this approach generally produces short paths relatively quickly [21]. Here, we generate plans using this approach with the NuSMV model checker [1].

For the ISR scenario illustrated in Fig. 4.1, we are interested in specifications of the form

$$\varphi = \underbrace{\bigwedge_{j \in \text{Goals}} \Diamond \pi_j \wedge \bigwedge_{j \in \text{Final Goals}} \Diamond\Box \pi_j \wedge}_{\text{Reachability}} \underbrace{\bigwedge_{j \in \text{Obs}} \Box \neg \pi_j}_{\text{Obstacle Avoidance}} \wedge \underbrace{\bigwedge_{i=1}^{N} (\pi_i^c \wedge \pi_i^o \rightarrow \neg \pi_i^u)}_{\text{Robot Collision Avoidance}} \tag{4.2}$$

where propositions of the form π_j label regions containing *Goals*, *Final Goals*, and *Obstacles*. Propositions of the form π_i^c, π_i^o, and π_i^u for *Robot Collision Avoidance* are further explained in Section 4.3.1.

4.2.2 Trust Model

Based on previous research in human-robot trust [19, 25, 20], we use the following time-series model to capture the dynamic evolution of trust:

$$\begin{aligned} T_i(k) = {} & AT_i(k-1) + B_1 P_{R_i}(k) - B_2 P_{R_i}(k-1) + \\ & C_1 P_H(k) - C_2 P_H(k-1) + D_1 F_i(k) - D_2 F_i(k-1) \end{aligned} \tag{4.3}$$

where $T_i(k)$ represents human trust in a robot i for $i \in \{1,\ldots,N\}$ at time step k, P_{R_i} represents robot performance, P_H represents human performance, and F_i represents faults made in the joint human-robot system. The coefficients $A, B_1, B_2, C_1, C_2, D_1, D_2$ can be determined by human subject testing. In this scenario, robot performance P_{R_i} is modeled as a function of "rewards" the robot receives when it identifies an obstacle or reaches a goal destination

$$P_{R_i}(k) = C_O N_{O_i}(k) + C_G N_{G_i}(k) \tag{4.4}$$

where N_{O_i} and N_{G_i} are the number of obstacles detected and goals reached by the robot i up to time step k, and C_O and C_G are corresponding positive rewards. This allows the robot to earn trust as it learns details of the environment.

Human performance is calculated based on workload and the complexity of the environment surrounding the robot with which the human is currently collaborating. The concept of the utilization ratio, γ, is used to measure workload based on recent usage history

$$\gamma(k) \quad = \quad \gamma(k-1) + \frac{\sum_{i=1}^{N} m_i(k) - \gamma(k-1)}{\tau} \tag{4.5}$$

where $m_i(k) = 1$ if the human is collaborating with robot i and 0 otherwise, and where τ can be thought of as the sensitivity of the operator. Assuming a human can only collaborate with one robot at a time, (4.5) allows workload to grow or decay between 0 and 1. Complexity of the environment is based on the number of obstacles that lie within sensing range r_i of collaborating robot i at time step k. A human is able to create more detailed paths in more complex environments, leading to increased performance in the presence of more obstacles, so that $P_H(k)$ is modeled as

$$P_H(k) = \begin{cases} 1 - \gamma(k)^{S_{o_i}(k)+1} & \text{if } m_i(k) = 1 \\ 1 - \gamma(k) & \text{if } m_i(k) = 0 \end{cases} \tag{4.6}$$

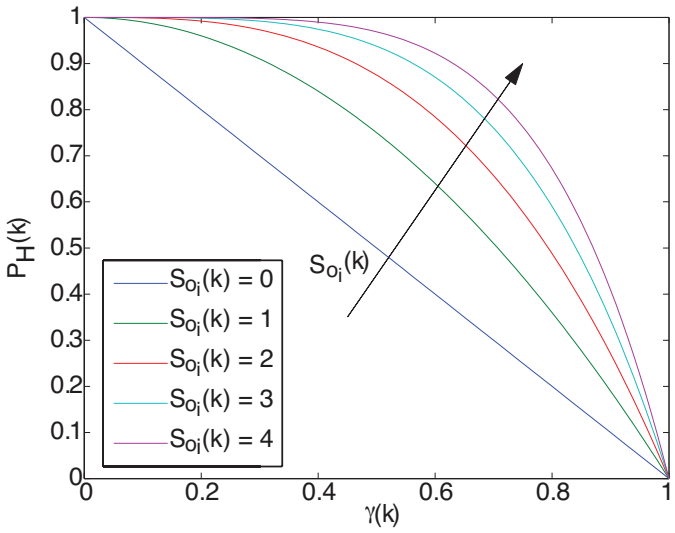

Figure 4.2: Plot of human performance when collaborating with a robot i.

where S_{o_i} is the number of obstacles within sensing range of collaborating robot i. Fig. 4.2 shows the change of human performance with respect to workload γ and environmental complexity S_{o_i}.

Faults in the system are modeled as

$$F_i(k) = C_H N_{H_i}(k) \tag{4.7}$$

where N_{H_i} is the total number of obstacle regions robot i has entered before sensing the corresponding obstacle up to time k, and C_H is the corresponding negative penalty. Note that faults can originate from both the robot and the human, i.e. human trust in a robot will decrease even if the robot enters an obstacle region under human planning.

4.3 Trust-Based Specification Decomposition

Available methods for multi-robot symbolic motion planning have mainly focused on fully autonomous systems and can be summarized into two types: centralized and de-

centralized solution approaches. Centralized solutions treat the robot team as a whole and have a large global state space formed by taking the product of the state spaces of all the robots [24, 23], resulting in a state space that is too large to handle in practice. Decentralized solutions tend to give local specifications to individual robots, which results in a smaller state space but often sacrifices guarantees on global performance [15], unless none of the specified tasks require direct collaboration between robots [30, 13]. Here, we propose a distributed solution for human-robot symbolic motion planning in which we separately address tasks that do not need direct collaboration between robots, i.e. *Reachability* and *Obstacle Avoidance*, and tasks that do need collaboration, i.e. *Robot Collision Avoidance*. We first present a method for addressing the collision avoidance task in Section 4.3.1 and then a method for decomposing the specification for individual tasks in Section 4.3.2

4.3.1 Specification Updates Based on Atomic Propositions for Communication, Observation, and Control

Here we consider robot collision avoidance tasks. This requires defining the atomic propositions π_i^c, π_i^o, and π_i^u in (4.2), which correspond to communication, observation, and control. A similar approach has been used in decentralized multi-robot tasking in [15], and here we extend the results to distributed multi-robot systems that must meet a global specification. The communication proposition π_i^c for robot i is true if another robot j is within its communication range ρ_i and false otherwise:

$$\pi_i^c(k) = \begin{cases} \|x_i(k) - x_j(k)\| \leq \rho_i & \text{true} \\ \|x_i(k) - x_j(k)\| > \rho_i & \text{false} \end{cases} \qquad (4.8)$$
$$j \neq i, \ = 1, 2, \cdots, N.$$

When π_i^c is true, robots i and j can communicate with each other to exchange sensing and path information, allowing them to learn features of the environment they have not yet explored themselves and resynthesize their plans to avoid obstacles if necessary. This information can also be used to detect possible collisions between the two robots, expressed in the proposition π_i^o. If it is observed that the current robot's motion plan will cause an immanent collision with the second robot, then π_i^o is true; otherwise π_i^o is false.

We next introduce the control proposition π_i^u for robot i. When π_i^u is true, robot i is executing a nominal linear quadratic regulator (LQR) control law; when false, the robot pauses or replans its path:

$$\pi_i^u(k) \quad = \quad \begin{cases} \text{LQR} & \text{true} \\ \text{wait or replan} & \text{false} \end{cases} . \tag{4.9}$$

We utilize an LQR control law to drive the robot to the midpoint of the next adjacent region in discrete path. This in conjunction with the simple first-order kinematic equation of motion given in Eq. (4.1) guarantees the robot will never enter an unplanned region while moving between sequential regions in the planned path, allowing us to establish a bisimulation relation between the continuous state space used for control of the robot and the discretized state space used for planning. That is, any paths planned in the discrete space will be guaranteed to be followed by the robots.

When both the communication and the observation propositions are true, i.e. $\pi_i^c \wedge \pi_i^o$, robot i has detected a potential collision and communicates its path with involved robot j. At this moment, the control proposition π_i^u is set to false, so that robot i either waits or replans depending on the collision type. If robot i's path is perpendicular to j's path, robot i waits until robot j passes. If robot i and j's paths are opposite to and coincide with each other, one of these robots will replan its path according to a prioritization policy.

While these two plans are able to resolve most intervehicle reactions, there still exists the possibility for a scenario to arise such that the vehicles cannot generate a successful plan or become stuck in a replanning loop. Should such a situation appear, the supervisor can intervene and use his or her higher level planning skills to resolve the conflict, lending to another advantage of including a human in a supervisory roll.

At each time step, the propositions are checked, and local specifications are dynamically updated. Through these propositions π_i^c, π_i^o, and π_i^u, we are able to decompose the robot collision avoidance task and guarantee there is no collision between the robots.

4.3.2 Compositional Reasoning Based on Assume-Guarantee (A-G) Contracts

Compositional reasoning based on A-G contracts formalizes the guarantees a system component can make given a set of assumptions about its environment – including other system components – in order to prove properties of the system as a whole in a computationally efficient manner [9]. We use the example shown in Fig. 4.3 to illustrate the use of A-G contracts to decompose a task specification for a two-robot team [30]. Assume that Robot 1 starts in region 2 and Robot 2 starts in region 6. Let the global specification be $\varphi =$ "One robot reaches region 4, one robot reaches region 8, and all robots avoid the obstacle in region 5." Without task decomposition, each individual robot needs to find a path that satisfies the whole specification φ. Instead, the specification can be decomposed by imposing an A-G contract on each robot. Let $\varphi_1 =$ "Robot 1 reaches region 4 and avoids region 5" and $\varphi_2 =$ "Robot 2 reaches region 8 and avoids region 5." Symbolically, the A-G

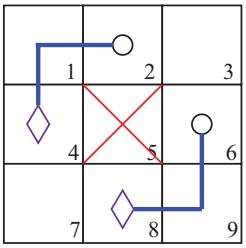

Figure 4.3: Specification decomposition for two robots using A-G contracts.

reasoning states that

$$\frac{\langle\varphi_2\rangle R_1\langle\varphi_1\rangle \qquad \langle\varphi_1\rangle R_2\langle\varphi_2\rangle}{\langle true\rangle R_1 \| R_2\langle\varphi_1\wedge\varphi_2\rangle} \tag{4.10}$$

Here, $R_1\|R_2$ denotes the composition of Robot 1 and Robot 2's transition systems, and $\varphi_1\wedge\varphi_2\Rightarrow\varphi$. Formulas of the form $\langle\varphi_2\rangle R_1\langle\varphi_1\rangle$ assert that R_1 guarantees φ_1 on the assumption that R_2 satisfies φ_2 and *vice versa* for R_2, so that $R_1\|R_2$ guarantees φ unconditionally. This allows us to deduce properties about $R_1\|R_2$ while reasoning about R_1 and R_2 separately, reducing the state space and making paths easier to compute and verify.

For the example in Fig. 4.3, one can verify the global specification $\varphi=\varphi_1\wedge\varphi_2$ using model checking with LTL with $\varphi_1=\Diamond region\ 4\wedge\Box\neg region\ 5$ and $\varphi_2=\Diamond region\ 8\wedge\Box\neg region\ 5$. Using model checking, it can be seen that a path exists such that R_1 satisfies φ_1. However, proving this also requires proving the remainder of the system cannot violate φ_1. Let $E_1=R_2$ represent the environment of Robot 1. Therefore, to prove R_1 satisfies φ_1, we must also show R_2, or in the case of multiple robots that the combination of all additional robots, does not violate φ_1. Since Robot 1 can be proven to reach region 4 individually, this proof entails showing no other robot enters region 5. Conducting model checking on Robot 2 shows it can satisfy φ_2, thereby proving it does not enter region 5. Therefore, it is shown $\langle True\rangle E_1\langle\Box\neg region\ 5\rangle$ and sequentially $\langle\neg region\ 5\rangle R_1\langle\varphi_1\rangle$. Repeating this process for R_2 likewise shows $\langle True\rangle E_2\langle\Box\neg region\ 5\rangle$ and $\langle\neg region\ 5\rangle R_2\langle\varphi_2\rangle$, proving the composition of

R_1 and R_2 always satisfies $\varphi_1 \wedge \varphi_2 = \varphi$.

Based on A-G contracts we can decompose the goal reaching and obstacle avoidance aspects of (4.2) into local subspecifications. First, each destination(s) in the set of *Goals* and *Final Goals* is assigned to one robot. Each robot then guarantees its assigned destinations are reached while avoiding all obstacles. This is specified by the following Lemma

Lemma 1 *Given the assumption that there exists at least one path to reach each goal, the robots are guaranteed to eventually generate a successful path to their goals individually provided there is no collision between individual robots.*

Proof Using the symbolic motion planning method outlined in section 4.2.1, every path generated by an individual robot will end in either reaching its goal or detecting an obstacle along the path. As this process is repeated, eventually the robot will gather a full map of the environment, meaning the final generated path will guarantee reaching its goal, provided there is no collision with another robot. ∎

This combined with the collision avoidance method in Section 4.3.1 ensures that the *Reachability* and *Obstacle Avoidance* portions of the global specification are met.

4.4 Real-Time Trust-Based Switching Between Human and Robot Motion Planning

In this section, we use trust in a real-time switching framework to switch between human and robot motion planning. Recall from Section 4.2 that although robot motion planning is guaranteed to be correct, it is usually conservative due to overapproximation of the environment. So while robot motion planning is safe, more efficient but riskier paths –

in this case, paths between obstacles in adjacent regions – may exist. If the human trusts a robot's ability to navigate between two obstacles, the human can choose to construct a more efficient path between the obstacles based on, e.g., sensory information about the obstacles supplied by the robot.

Fig. 4.4 shows an example simulation of how trust can benefit the path planning of an autonomous robot, generated by integrating the NuSMV model checker [1] with Matlab. The robot begins in region 5, represented by the circle, and must reach the goal in region 15, represented by the diamond. This must be done while avoiding obstacles, represented in the continuous space by filled polygons and represented in the discrete space by regions marked with "X"s. The robot planning mode plans safe paths that avoid regions marked by Xs, whereas the human planning mode can plan more efficient paths that go between obstacles. This tradeoff between safety and efficiency motivates the use of trust-based switching between human and robot motion planning.

As Fig. 4.4a shows, the robot motion planner generates a safe but lengthy path to the goal; while there is a gap between the two obstacles through region 10, the overapproximation prevents the robot from generating a path through this region. If the human operator trusts the robot's ability to follow a path between the obstacles and the human is not overloaded, the human planning mode will be activated and the benefits can clearly be seen in Fig. 4.4b.

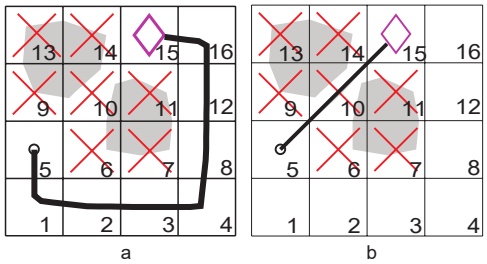

Figure 4.4: (a) Safe robot motion planning in low trust scenario, and (b) advanced human motion planning in high trust scenario.

This high trust path, however, is inherently risky since information regarding the environment in a typical scenario is limited to what the robot has sensed. It is therefore necessary to have an additional condition by which the robot will release the human plan should the robot detect that it is about to collide with an obstacle. The release condition here is represented by $\|x_i - x_O\| \leq r_O$, where x_i and x_O are the robot and obstacle positions and r_O is some minimally acceptable distance between them. Should the robot come within this distance, it will go back to the previous region and return to robot planning mode, triggering a fault. According to the trust equation (4.3), this lowers trust in the robot, leading to a re-evaluation of the assigned tasks. The result is that other more trusted robots may be re-assigned some of the destinations that were originally assigned to the robot that generated the fault. Once this re-evaluation is performed, the operator is free to continue working with the same or another robot depending on current levels of trust.

Theorem 2 *Given Lemma 1, the robot collision avoidance method detailed in Section 4.3.1, and the trust-based switching scheme detailed here, the human-robot team is guaranteed to satisfy global specifications of the type given in Eq. (4.2)*

Proof From the proof of Lemma 1 it is shown that given no collision between robots, each robot will generate a path that will eventually guarantee reaching its goal, with the bisimulation property given in 4.3.1 guaranteeing the robot will follow this path correctly. Section 4.3.1 outlines a scheme that guarantees there is no collision between robots, satisfying the condition in Lemma 1. The human generated plans provide a more efficient way of reaching goals, but can be risky if there is incomplete environmental information. However, using the release condition outlined above, the robot is guaranteed to never collide with an obstacle and will return to the proven safe method. This switching will continue until all goals are reached or a complete obstacle map is generated, at which time the human a path that is efficient and safe. ∎

4.5 Simulation

In this section, a simulation of the ISR scenario of Section 4.2 is used to demonstrate the methods described in the previous sections. The simulation is conducted in Matlab with model checking being computed using NuSMV, as mentioned previously, with the simulation codes provided at `http://people.clemson.edu/~yue6/papers/thesis/DAS_Thesis_Codes.pdf`. The environment is shown in Fig. 4.7 and consists of three robots, marked by numbered circles, and six goals, marked by diamonds. There are 16 obstacles in the environment, marked by "X"s, which are initially unknown by the robots until they are sensed. The sensor range of a robot is marked by a dashed circle around the robot. Once an obstacle is sensed, its position becomes known to that individual robot; it is communicated with other robots when they come within communication range. The goal of the human-robot team is to successfully reach each goal destination while avoiding all collisions, meeting the global specification (4.2). To address computational complexity, A-G contracts are used to decompose (4.2) into local subspecifications as described in Section 4.3.2. For this scenario, trust levels are assumed to be equivalent at the start of the simulation, leading each robot to be assigned an equal number of goals. In practice, initial trust levels can vary according to factors such as prior experience with a robot.

To find a path such that the local subspecification is satisfied, the model checking approach described in Section 4.2.1 is used by each individual robot. To further reduce computational complexity, each robot only computes its path over a local subspace. This subspace is determined using the robot's knowledge of its assigned goals and the location of obstacles that have been sensed. Using this method, every path generated by an individual robot will end in either reaching its goals or detecting an obstacle and will continue until all obstacles are detected. Therefore, the robots are guaranteed to eventually reach their goals as outlined in Lemma 1

To prevent collisions between the robots, the communication method detailed in Section 4.3.1 is used with the communication range ρ_i set as equal to the sensor range. An example scenario is outlined in Fig. 4.5 with the arrows representing the path information communicated between the robots. Here, all three robots are in collision scenarios. Robot 2 sees that it can avoid collision by simply waiting for robot 1 to pass. Robots 1 and 3 however must replan to avoid collision. Should a situation arise as described in Section 4.3.1 where conflict between two robots are not immediately solved, an additional option for π_i^u is interaction from the human supervisor whereby he or she generates a plan for the affected robots, guaranteeing the robots are able to navigate around each other without colliding.

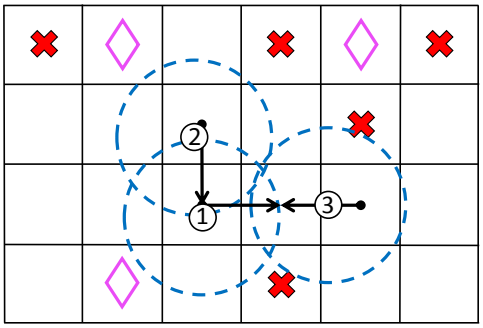

Figure 4.5: Collision avoidance scenario.

The results of the simulation can be seen in Fig. 4.7, with 4.6 showing key moments during the simulation and 4.8 showing the evolution of trust. From Fig. 4.8, it can be seen that both robots 1 and 2 quickly gain trust due to discovering an obstacle, but robot 2 gains the definitive lead in trust due to quickly observing two additional obstacles and having a relatively complex immediate environment when it reaches region (3,10) (Fig. 4.6 f). The operator is therefore paired with robot 2 and then designs a path through the two obstacles, through the first goal, and continues up and right towards robot 2's second goal (Fig. 4.6 b). Robot 2 follows this path until nearly colliding with the obstacle in region (4,10) (Fig. 4.6 g). This creates a fault, releasing the operator's plan and starting a return to robot control.

The fault causes a significant drop in trust, as can be seen in Fig. 4.8 around time step 300. As a result of this drop in trust, a reallocation of tasks is initiated. Robot 2 has its task load decreased, meaning it no longer has any more tasks to complete, and robot 1, which has the highest trust value, has its task load increased. Robot 1 is then paired with the operator since it has the highest trust and then finishes the task following the operator's plan (Fig. 4.6 d). Robot 3 also successfully completes its task using the safe control scheme, leading to the global specification being satisfied.

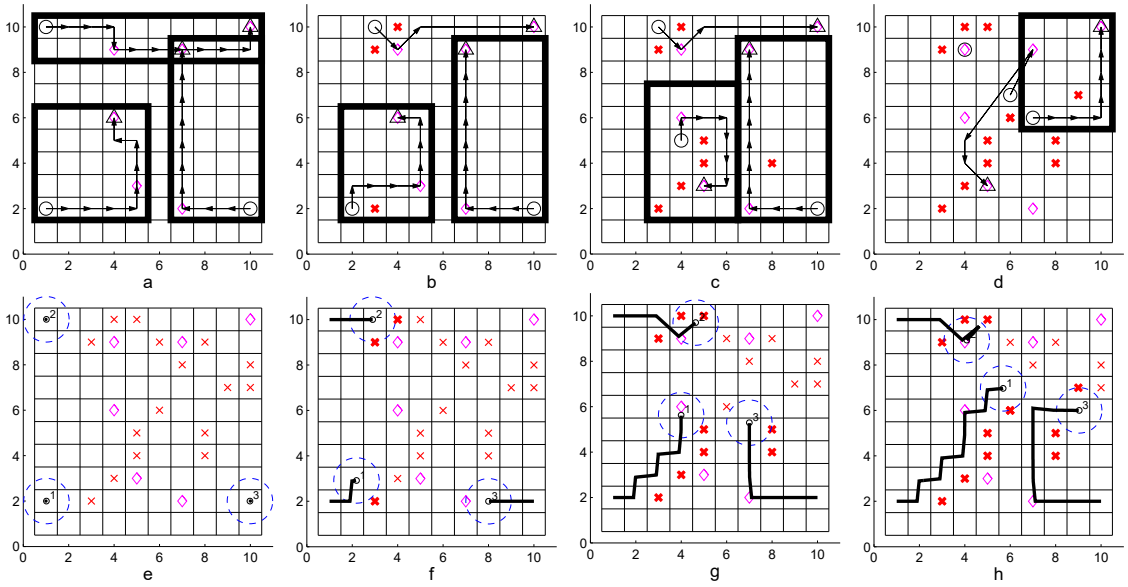

Figure 4.6: Progression of simulation (e - h) with corresponding path plans (a - d).

4.6 Conclusions

In this paper, we have explored methods to address scalability and safety of symbolic motion planning for multi-robot systems that interact with a human operator, with interactions affected by human trust. Scalability is addressed by decomposing portions of the global symbolic motion planning specification that do not require inter-robot collabo-

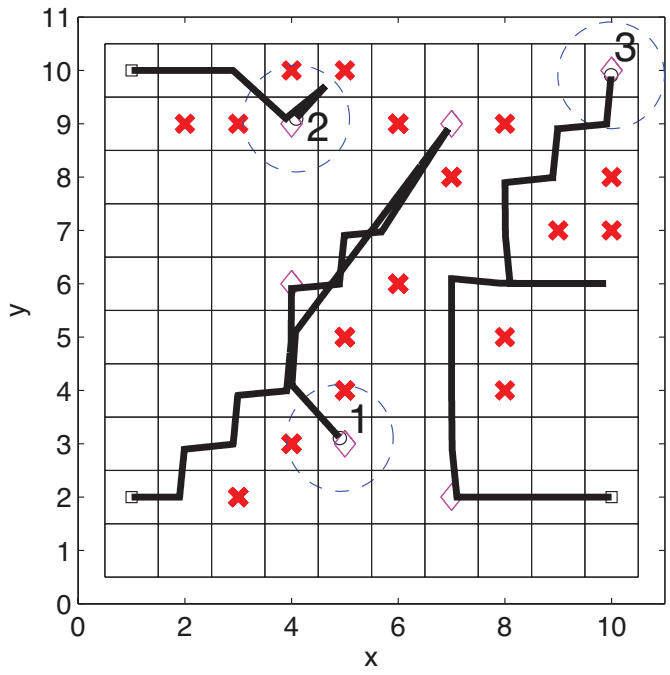

Figure 4.7: Final paths of robots.

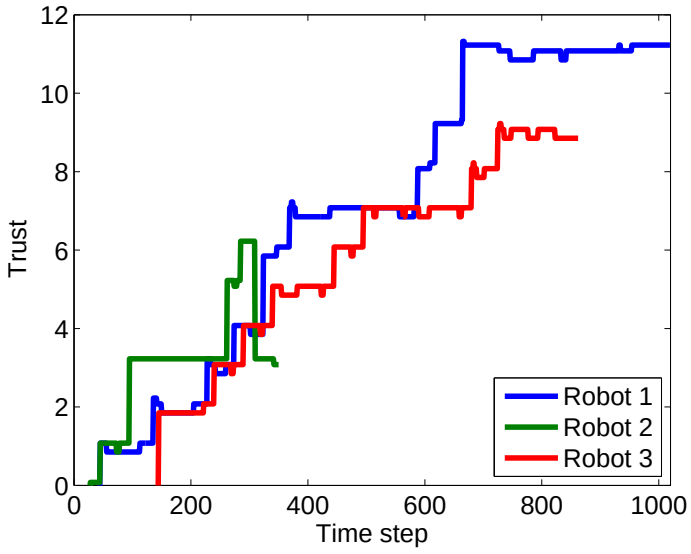

Figure 4.8: Trust evolution of robots.

ration using A-G contracts, while portions that require collaboration are addressed through a protocol that relies on local communication, observation, and control to modify plans as needed during execution. Trust affects this decomposition, with more trusted robots assigned more destinations, and with trust changing as robots generate faults by coming too close to obstacles. In addition to implementing methods for obstacle and collision avoidance, safety versus efficienty of planning is addressed by switching between a safe but conservative robot motion planning mode that avoids obstacles based on an overapproximation of the environment and a riskier but more efficient human planning mode that allows paths between obstacles, with switching mediated by human trust.

In the future, there are several other areas that could be explored. For instance, other human trust models could be used in this framework. The trust model we use here assumes trust in a specific robot evolves independently of all other robots, when in actuality, evolution of trust for each robot might be interdependent [22]. More complex probabilistic models of trust [43] could also be used. In addition, further proofs of correctness for the A-G contracts and robot collision avoidance protocols in more complex environments could be developed. Here, we have assumed that obstacles are placed such that all destinations are reachable by all robots, which would not be true in all environments and would require more careful assignment of destinations. In some cases, certain destinations might not be reachable by any robot, which would require revising the specification. We have also assumed that there is sufficient room for the robots to manuever when implementing the protocol for robot collision avoidance, and that it is unlikely for more than two robots to encounter each other a time. Collision avoidance protocols in complex environments with arbitrary numbers of robots is a challenging area of research that could be further developed within this framework.

Chapter 5

Conclusions

5.1 Discussion

Despite advances in autonomy, humans still offer invaluable input in many aspects of control. This has been shown by implementing HRI in the control of autonomous robots at varying levels in the control hierarchy. Implementation of HRI within the control scheme for low-level operations, such as the direct control of the AUV, has shown that a human can assess a scenario and compensate for autonomy's navigational shortcomings, while the operator benefits from a reduced workload during scenarios within the autonomy's capabilities. The SLQR has also been shown to be an effective tool, offering the ability to determine this switching scenario using optimal control policies not just at the low-level but for high-level as well. This was shown using the SLQR as a tool for HRI in obstacle avoidance with an AUV. By using the SLQR, autonomy is controlled via an optimal control policy until it encounters an obstacle along its planned path. Once the operator replans the path, he or she can return to the more important task of data processing.

HRI was also shown to provide useful advantages even for guaranteed and safe high-level symbolic motion planning operations. By incorporating a human operator, the

safe but inefficient symbolic motion planning can incorporate riskier paths only devisable by the operator. The operator also has the benefit of knowing that if his or her plan fails, the system will always fall back on a guaranteed autonomous method. The operator also provided the advantage of resolving conflict between multiple robots in cases that are too complex for decentralized motion planning. The conclusion of these works shows the advantages of not only using HRI, but of also incorporating it directly into the control schemes themselves, offering a way to take advantage of human factors to best incorporate the advantages of both human and autonomy.

5.2 Future Works

The works presented here show the beginning of promising topics of research and offer many paths for future work on the subject. Currently, the methods presented here are implemented individually with high-level and low-level control schemes separated. One major desire is to incorporate these concepts into the development of a comprehensive control scheme that takes advantage of HRI at all control levels simultaneously. Work is also currently underway to implement the trust-based HRI for multi-robot symbolic motion planning using real mobile robots. Future work will also incorporate larger teams, including multiple operators, and determine how trust plays a role in multi-operator control. It is also desired to extend this work to heterogeneous robot groups with different sensing and motion capabilities. Finally, with the current interest in dynamic trust modeling, the trust model used in this work will be constantly improved and refined. As shown in the introduction, correctly predicting and identifying the level of trust in automation can have a significant impact on proper use of autonomy and can even allow the autonomy itself adjust to this trust.

Appendices

Appendix A AUV Equations of Motion

$$\dot{x} = w(\sin(\phi)\sin(\psi) + \cos(\phi)\cos(\psi)sin(\theta)) - v(\cos(\phi)\sin(\psi) - \cos(\psi)\sin(\phi)\sin(\theta)) + u\cos(\psi)\cos(\theta)$$

$$\dot{y} = v(\cos(\phi)\cos(\psi) + \sin(\phi)\sin(\psi)\sin(\theta)) - w(\cos(\psi)\sin(\phi) - \cos(\phi)\sin(\psi)\sin(\theta)) + u\cos(\theta)\sin(\psi)$$

$$\dot{z} = w\cos(\phi)\cos(\theta) - u\sin(\theta) + v\cos(\theta)\sin(\phi)$$

$$\dot{\phi} = p + r\cos(\phi)\tan(\theta) + q\sin(\phi)\tan(\theta)$$

$$\dot{\theta} = q\cos(\phi) - r\sin(\phi)$$

$$\dot{\psi} = \frac{r\cos(\phi)}{\cos(\theta)} + \frac{q\sin(\phi)}{\cos(\theta)}$$

$$D_1 u - rv(\dot{Y}_v + m) + qw(\dot{Z}_w + m) + \dot{u}(\dot{X}_u + m) =$$
$$b_{x7}\alpha^2 u^2 + b_{x4}\alpha^2 u + b_{x8}\beta^2 u^2 + b_{x5}\beta^2 u + C_{prop} rpm^2 + b_{x6}u^4 + b_{x3}u^3 + b_{x2}u^2 + b_{x1}u$$

$$D_2 v + ru(\dot{X}_u + m) - pw(\dot{Z}_w + m) + \dot{v}(\dot{Y}_v + m) = b_{y2}\beta^3 u + b_{y4}\beta u^3 + b_{y3}\beta u^2 + b_{y1}\beta u$$

$$D_3 w - qu(\dot{X}_u + m) - pv(\dot{Y}_v - m) + \dot{w}(\dot{Z}_w + m) = b_{z3}\alpha^3 u + b_{z7}\alpha u^3 + b_{z5}\alpha u^2 + b_{z2}\alpha u + b_{z8}u^4 + b_{z6}u^3 + b_{z4}u^2 + b_{z1}u$$

$$D_4 p - rq(I_y - \dot{M}_q) + qr(I_z - \dot{N}_r) - wv(\dot{Y}_v + m) + vw(\dot{Z}_w + m) + \dot{p}(I_x + \dot{K}_p) = 0$$

$$D_5 q + rp(I_x - \dot{K}_p) - pr(I_z - \dot{N}_r) + wu(\dot{X}_u + m) - uw(\dot{Z}_w + m) + \dot{q}(I_y + \dot{M}_q) =$$
$$b_{\theta 3}\alpha^3 u + b_{\theta 7}\alpha u^3 + b_{\theta 5}\alpha u^2 + b_{\theta 2}\alpha u + b_{\theta 8}u^4 + b_{\theta 6}u^3 + b_{\theta 4}u^2 + b_{\theta 1}u$$

$$D_6 r + qp(I_x + \dot{K}_p) + pq(I_y - \dot{M}_q) - vu(\dot{X}_u + m) - uv(\dot{Y}_v - m) + \dot{r}(I_z + \dot{N}_r) = b_{\psi 2}\beta^3 u + b_{\psi 4}\beta u^3 + b_{\psi 3}\beta u^2 + b_{\psi 1}\beta u$$

Here, I is the moment of inertia in the specified direction, D is an entry in the damping matrix, $\dot{X}$, $\dot{Y}$, $\dot{Z}$, $\dot{K}$, $\dot{M}$, and $\dot{N}$ are entries corresponding to the hydrodynamic added inertia matrix, C_{prop} is the propeller coefficient, and b are coefficients corresponding to the torque inputs of the system.

Appendix B Linearized Dynamic Matrices

$$A = \begin{bmatrix} 1 & -\Delta t u_0 \cos\theta_0 - \Delta t w_0 \sin\theta_0 & 0 & -\Delta t \sin\theta_0 & 0 & \Delta t \cos\theta_0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & \Delta t \frac{1}{\cos\theta_0} \\ 0 & 0 & 0 & 1+\Delta t \frac{C_{uu}}{\dot{X}_u+m} & 0 & 0 & -\Delta t \frac{\dot{Z}_w+m}{\dot{X}_u+m} w_0 & 0 \\ 0 & 0 & 0 & 0 & 1-\Delta t \frac{D_2}{\dot{Y}_v+m} & 0 & 0 & -\Delta t \frac{\dot{X}_u+m}{\dot{Y}_v+m} u_0 \\ 0 & 0 & 0 & \Delta t \frac{C_{wu}}{\dot{Z}_w+m} & 0 & 1-\Delta t \frac{D_3}{\dot{Z}_w+m} & \Delta t \frac{\dot{X}_u+m}{\dot{Z}_w+m} u_0 & 0 \\ 0 & 0 & 0 & \Delta t \frac{C_{qu}}{I_y+M_q} & 0 & \Delta t \frac{\dot{Z}_w-\dot{X}_u}{I_y+M_q} u_0 & 1-\Delta t \frac{D_5}{I_y+M_q} & 0 \\ 0 & 0 & 0 & 0 & \Delta t \frac{\dot{X}_u+\dot{Y}_v}{I_z+\dot{N}_r} u_0 & 0 & 0 & 1-\Delta t \frac{D_6}{I_z+\dot{N}_r} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \Delta t \frac{2b_{x7}\alpha_0 u_0^2 + 2b_{x4}\alpha_0 u_0}{\dot{X}_u+m} & 0 & \Delta t \frac{2C_{prop} rpm_0}{\dot{X}_u+m} \\ 0 & \Delta t \frac{b_{y4}u_0^3 + b_{y3}u_0^2 + b_{y1}u_0}{\dot{Y}_v+m} & 0 \\ \Delta t \frac{3b_{z3}\alpha_0^2 u_0 + b_{z7}u_0^3 + b_{z5}u_0^2 + b_{z2}u_0}{\dot{Z}_w+m} & 0 & 0 \\ \Delta t \frac{3b_{\theta 3}\alpha_0^2 u_0 + b_{\theta 7}u_0^3 + b_{\theta 5}u_0^2 + b_{\theta 2}u_0}{I_y+M_q} & 0 & 0 \\ 0 & \Delta t \frac{b_{\psi 4}u_0^3 + b_{\psi 3}u_0^2 + b_{\psi 1}u_0}{I_z+\dot{N}_r} & 0 \end{bmatrix} \tag{1}$$

where

$$C_{uu} = -D_1 + 2b_{x7}\alpha_0^2 u_0^2 + b_{x4}\alpha_0^2 + 4b_{x6}u_0^3 + 3b_{x7}u_0^2 + 2b_{x2}u_0 + b_{x1} \tag{2}$$

$$C_{wu} = b_{z3}\alpha_0^3 + 3b_{z7}\alpha_0 u_0^2 + 2b_{z5}\alpha_0 u_0 + b_{z2}\alpha_0 + 4b_{z8}u_0^3 + 3b_{z6}u_0^2 + 2b_{z4}u_0 + b_{z1} \tag{3}$$

$$C_{qu} = (\dot{Z}_w - \dot{X}_u)w_0 + b_{\theta 3}\alpha_0^3 + 3b_{\theta 7}\alpha_0 u_0^2 + 2b_{\theta 5}\alpha_0 u_0 + b_{\theta 2}\alpha_0 + 4b_{\theta 8}u_0^3 + 3b_{\theta 6}u_0^2 + 2b_{\theta 4}u_0 + b_{\theta 1} \tag{4}$$

Here, Δt is the discrete time increment, m is the mass of the AUV, I is the moment of inertia in the specified direction, D is an entry in the damping matrix, $\dot{X}$, $\dot{Y}$, $\dot{Z}$, $\dot{K}$, $\dot{M}$, and $\dot{N}$ are entries corresponding to the hydrodynamic added inertia matrix, C_{prop} is the propeller coefficient, and b are coefficients corresponding to the torque inputs of the system.

Bibliography

[1] NuSMV: a new symbolic model checker. `http://nusmv.fbk.eu/`.

[2] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press Cambridge, 2008.

[3] Calin Belta, Antonio Bicchi, Magnus Egerstedt, Emilio Frazzoli, Eric Klavins, and George J Pappas. Symbolic planning and control of robot motion [grand challenges of robotics]. *IEEE Robotics & Automation Magazine*, 14(1):61–70, 2007.

[4] Steven J. Benson, Yinyu Ye, and Xiong Zhang. Solving Large-Scale Sparse Semidefinite Programs for Combinatorial Optimization. *SIAM Journal on Optimization*, 10(2):443–461, 2000.

[5] Matthew Bolton, Ellen Bass, and Radu Siminiceanu. Using formal verification to evaluate human-automation interaction: A review. *IEEE Trans. Systems, Man, and Cybernetics: Systems*, 43(3), 2013.

[6] Regardt Busch. *Modelling and simulation of an autonomous underwater vehicle*. PhD thesis, Stellenbosch: University of Stellenbosch, 2009.

[7] Tom Carlson and Yiannis Demiris. Collaborative control for a robotic wheelchair: evaluation of performance, attention, and workload. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 42(3):876–888, 2012.

[8] Andrew S Clare. *Modeling real-time human-automation collaborative scheduling of unmanned vehicles*. PhD thesis, Massachusetts Institute of Technology, 2013.

[9] Jamieson M. Cobleigh, Dimitra Giannakopoulou, and Corina S. Păsăreanu. Learning assumptions for compositional verification. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 331–346. Springer Berlin Heidelberg, 2003.

[10] Gilles Coppin and François Legras. Autonomy spectrum and performance perception issues in swarm supervisory control. *Proceedings of the IEEE*, 100(3):590–603, 2012.

[11] Jacob W Crandall, Michael A Goodrich, Dan R Olsen Jr, and Curtis W Nielsen. Validating human-robot interaction schemes in multitasking environments. *Systems, Man*

and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 35(4):438–449, 2005.

[12] Jonathan Currie and David I. Wilson. OPTI: Lowering the Barrier Between Open Source Optimizers and the Industrial MATLAB User. In Nick Sahinidis and Jose Pinto, editors, *Foundations of Computer-Aided Process Operations*, Savannah, Georgia, USA, 8–11 January 2012.

[13] Jin Dai and Hai Lin. Automatic synthesis of cooperative multi-agent systems. *Proc. IEEE Conf. Decision and Control*, 2014.

[14] Lu Feng, Clemens Wiltsche, Laura Humphrey, and Ufuk Topcu. Controller synthesis for autonomous systems interacting with human operators. In *Proc. ACM/IEEE 6th Int. Conf. Cyber-Physical Systems*, 2015.

[15] Ioannis Filippidis, Dimos V Dimarogonas, and Kostas J Kyriakopoulos. Decentralized multi-agent control from local LTL specifications. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pages 6235–6240. IEEE, 2012.

[16] D Fryxell, P Oliveira, A Pascoal, C Silvestre, and I Kaminer. Navigation, guidance and control of auvs: an application to the marius vehicle. *Control Engineering Practice*, 4(3):401–409, 1996.

[17] Morton Gertler and Grant R Hagen. Standard equations of motion for submarine simulation. Technical report, DTIC Document, 1967.

[18] M. A. Goodrich and A. C. Schultz. Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275, 2007.

[19] Peter A Hancock, Deborah R Billings, Kristin E Schaefer, Jessie YC Chen, Ewart J De Visser, and Raja Parasuraman. A meta-analysis of factors affecting trust in human-robot interaction. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 53(5), 2011.

[20] Kevin Anthony Hoff and Masooda Bashir. Trust in automation integrating empirical evidence on factors that influence trust. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 57(3):407–434, 2015.

[21] Laura R Humphrey, Eric M Wolff, and Ufuk Topcu. Formal specification and synthesis of mission plans for unmanned aerial vehicles. In *2014 AAAI Spring Symp. Series*, 2014.

[22] D. Keller and S. Rice. System-wide versus component-specific trust using multiple aids. *The Journal of General Psychology*, 137(1):114–128, 2009.

[23] M. Kloetzer and C. Belta. Automatic deployment of distributed teams of robot from temporal logic motion specifications. *IEEE Transactions on Robotics*, 26(1):48–61, February 2010.

[24] M. Kloetzer, S. Itani, S. Birch, and C. Belta. On the need for communication in distributed implementations of lti motion specifications. *IEEE Int. Conf. Robotics and Automation*, pages 4451–4456, 2010.

[25] John Lee and Neville Moray. Trust, control strategies and allocation of function in human-machine systems. *Ergonomics*, 35(10):1243–1270, 1992.

[26] D. Z. Levin, E. M. Whitener, and R. Cross. Perceived trustworthiness of knowledge sources: The moderating impact of relationship length. *Journal of Applied Psychology*, 91(5):1163–1171, 2006.

[27] Bo Lincoln and Bo Bernhardsson. Lqr optimization of linear system switching. *Automatic Control, IEEE Transactions on*, 47(10):1701–1705, 2002.

[28] Bérénice Mettler, Navid Dadkhah, Zhaodan Kong, and Jonathan Andersh. Research infrastructure for interactive human-and autonomous guidance. *Journal of Intelligent & Robotic Systems*, 70(1-4):437–459, 2013.

[29] Raja Parasuraman and Victor Riley. Humans and automation: Use, misuse, disuse, abuse. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 39(2):230–253, 1997.

[30] Alireza Partovi and Hai Lin. Assume-guarantee cooperative satisfaction of multi-agent systems. *American Control Conf. (ACC)*, 2014.

[31] Scott Reed, Jon Wood, and Chris Haworth. The detection and disposal of ied devices within harbor regions using auvs, smart rovs and data processing/fusion technology. In *Waterside Security Conference (WSS), 2010 International*, pages 1–7. IEEE, 2010.

[32] Thomas M Roehr and Yuping Shi. Using a self-confidence measure for a system-initiated switch between autonomy modes. In *Proceedings of the 10th International Symposium on Artificial Intelligence, Robotics and Automation in Space, Sapporo, Japan*, pages 507–514, 2010.

[33] H. A. Ruff, S. Narayanan, and M. H. Draper. Human interaction with levels of automation and decision-aid fidelity in the supervisory control of multiple simulated unmanned air vehicles. *Presence: Teleoperators and virtual environments*, 11(4):335–351, 2002.

[34] B. Sadrfaridpour, H. Saeidi, , J. Burke, K. Madathil, and Y. Wang. Modeling and control of trust in human-robot collaborative manufacturing. *Robust Intelligence and Trust in Autonomous Systems*, 2015.

[35] Ketan Savla and Emilio Frazzoli. A dynamical queue approach to intelligent task management for human operators. *Proceedings of the IEEE*, 100(3):672–686, 2012.

[36] JB Veneman. Switched lqr control: Design of a general framework. Master's thesis, Netherlands: Delft Unv. of Technology, 2012.

[37] Frank MF Verberne, Jaap Ham, and Cees JH Midden. Trust in smart systems sharing driving goals and giving information to increase trustworthiness and acceptability of smart systems in cars. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 54(5):799–810, 2012.

[38] Chuanfeng Wang, Fumin Zhang, and Dirk Schaefer. Dynamic modeling of an autonomous underwater vehicle. *Journal of Marine Science and Technology*, pages 1–14, 2014.

[39] Jijun Wang and Michael Lewis. Human control for cooperating robot teams. In *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, pages 9–16. IEEE, 2007.

[40] X. Wang, Z. Shi, F. Zhang, and Y. Wang. Dynamic real-time scheduling for human-agent collaboration systems based on mutual trust. *Cyber-Physical Systems (CPS), Taylor & Francis*, 2015.

[41] Yue Wang, Zhenwu Shi, Chuanfeng Wang, and Fumin Zhang. Human-robot mutual trust in (semi) autonomous underwater robots. In *Cooperative Robots and Sensor Networks 2014*, pages 115–137. Springer, 2014.

[42] Anqi Xu and Gregory Dudek. Towards modeling real-time trust in asymmetric human-robot collaborations. *Int. Sym. on Robotics Research (ISRR13)*, 2013.

[43] Anqi Xu and Gregory Dudek. OPTIMo: Online probabilistic trust inference model for asymmetric human-robot collaborations. In *Proc. ACM/IEEE Int. Conf. Human-Robot Interaction*. ACM, 2015.

[44] Junku Yuh. Design and control of autonomous underwater robots: A survey. *Autonomous Robots*, 8(1):7–24, 2000.

[45] Wei Zhang, Alessandro Abate, and Jianghai Hu. Efficient suboptimal solutions of switched lqr problems. In *American Control Conference, 2009. ACC'09.*, pages 1084–1091. IEEE, 2009.

[46] Wei Zhang and Jianghai Hu. Optimal quadratic regulation for discrete-time switched linear systems: A numerical approach. In *American Control Conference, 2008*, pages 4615–4620. IEEE, 2008.

[47] Wei Zhang, Jianghai Hu, and Jianming Lian. Quadratic optimal control of switched linear stochastic systems. *Systems & Control Letters*, 59(11):736–744, 2010.