AN ABSTRACT OF THE THESIS OF

Lauren Milliken for the degree of Master of Science in Robotics presented on August 18, 2017.

Title: Modeling Human Expertise for Providing Adaptive Levels of Robot Shared Autonomy

Abstract approved: ______________________________

Geoffrey A Hollinger

In shared autonomy, a robot and human user both have some level of control in order to achieve a shared goal. Choosing the balance of control given to the user and the robot can be a challenging problem since different users have different preferences and vary in skill levels when operating a robot. We propose using a novel formulation of Partially Observable Markov Decision Processes (POMDPs) to represent a model of the user's expertise in controlling the robot. The POMDP uses observations from the user's actions and from the environment to update the belief of the user's skill and chooses a level of control between the robot and the user. The level of control given between the user and the robot is encapsulated in macro-action controllers. The macro-action controllers encompass varying levels of robot autonomy and reduce the space of the POMDP, removing the need to plan over separate actions. As part of this research, we ran two users study, developed a method to automatically generate macro-action

controller values, and applied our user expertise model to provide shared autonomy on a semi-autonomous underwater vehicle.

In our first user study, we tested our user expertise model in a robot driving simulation. Users drove a simulated robot through an obstacle-filled map while the POMDP model chose appropriate macro-action controllers based on the belief state of the user's skill level. The results of the user study showed that our model can encapsulate user skill levels. The results also showed that using the controller with greater robot autonomy helped users of low skill avoid obstacles more than it helped users of high skill.

We designed a controller value synthesis method to generate the variables that control the levels of autonomy in the macro-action controllers. We found differences in how the users drive the robot using a decision tree generated from the data recorded in the first user study, and we used these differences to program simulated user "bots" that mimic users of different skill levels. The "bots" were used to test a range of variables for the controllers, and the controller variables were found from minimizing obstacles hit, time to complete maps, and total distance driven from the simulated data.

For our second user study, we looked at users' satisfaction without robot autonomy, with the highest amount of autonomy, and with the autonomy chosen by our expertise model. We found users we classified as beginners ranked the autonomy more favorably than those ranked as experts.

We implemented our expertise model on a Seabotix vLBV300 underwater vehicle and ran a trial off the coast of Newport, Oregon. During our trials, we recorded a user driving the vehicle to predetermined waypoints. When beginner actions were performed, the user expertise model provided an increased level of autonomy which either increased throttle when far from waypoints or decreased throttle when close to waypoints. This demonstrated an implementation of our algorithm on existing robot hardware in the field.

Modeling Human Expertise for Providing Adaptive Levels of Robot Shared Autonomy

by

Lauren Milliken

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented August 18, 2017
Commencement June 2018

ACKNOWLEDGEMENTS

I would like to acknowledge those who made this work possible. My advisor, Dr. Geoff Hollinger, and all my colleagues in the Robotic Decision Making Laboratory. I thank Matt Reuben for his guidance on how to run a user study, and my committee members Dr. Cindy Grimm and Dr. Julie A. Adams for their advice and guidance. Finally, I would like to give special thanks to the lab's wonderful babies, Robbie, Kelvin, and Toni. Your adorable pictures and videos often lifted my spirits.

This research is funded in part by grant NSF IIS-1317815, DOE contract DE-EE-0006816, and support from PCC Structurals, Inc. and Oregon Metals Initiative. User studies were performed under Oregon State University IRB 6656.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF TABLES

LIST OF ALGORITHMS

1 Introduction

When deploying robots in the field, there are many advantages to using shared autonomy, where both the human operator and the robot have some level of control. Direct teleoperation may be tedious or difficult, and assistance from the robot can greatly reduce the burden on the human user. However, different users may have varying needs when it comes to the level of autonomy the robot that should be provided. One way that users are different is in their level of expertise. A novice user might need a great deal of assistance in performing even basic tasks in order to complete the goal safely. With only direct teleoperation, a novice may take a long time to complete a task, experience higher levels of frustration and displeasure with the robot, and may be dangerous to themselves, others or the robot. On the other hand, a user with more experience could accomplish these tasks without as much assistance from the robot. In some cases, a user may be able to accomplish a task faster without robot assistance and might even dislike the lack of control in certain situations. Different users also have different learning curves. Some might quickly become familiar with the system, while others might need assistance for longer. For a robot to work for the greatest number of people, it should adapt to the different needs and preferences of different users.

Our work presents methods that model human expertise and choose the amount of control to share between the human and the robot's autonomy. The goal of this work is to develop a robotic system that can be used by a variety of users of different skill levels in ways that work well for them.

We chose to model expertise because there are many key differences in how a novice operates a robot and how an expert would. In some cases, robot autonomy hinder expert users. For example, commercial unmanned aerial vehicles (UAVs) come with auto-pilot features that make flying the UAV much simpler and prevents crashing in most scenarios. These features have enabled first-time UAV pilots the ability to fly more easily and safely, but these features may prevent an expert user from performing more complicated maneuvers. In some cases, if a user finds they do not want these features, they can shut them off. Our work seeks to do this automatically. By modeling user expertise, we can help ensure that beginner users will be able to control a robot successfully and safely. We can also make sure that expert users will not be hindered by undesired autonomy.

We apply our work modeling expertise in the domain of marine robotics. Missions involving teleoperated marine robots can be difficult for a number of reasons. Ships are expensive to launch and can only hold a limited number of people. Thus operators of marine robots are usually very highly trained and experience a high workload from long missions [17]. Assistive autonomy can help relieve the operator of an intense workload and require less training. Our work uses a novel expertise model on a tethered semi-autonomous underwater vehicle to provide different levels of vehicle control shared between the operator and the autonomy by observing how well the operator performs.

1.1 Thesis Contributions

In this thesis, we will present the following contributions to the field of shared autonomy:

- A method to predict human user expertise using a Partially Observable Markov Decision Process. We use macro-action controllers that encompass varying levels of robot autonomy into the robot's controllers in order to reduce the size of the action space of our model. By using macro-action controllers in this model, we can adjust the level of shared autonomy provided to a user based on their observed actions and belief that they are a beginner or expert. We ran a user study of fourteen participants in a robot driving domain. The results verified that our POMDP model was able to predict expertise. Through this study, we found that users who had a higher probability of being a beginner received more help when receiving higher levels of robot autonomy than users with a lower probability of being a beginner.

- A method for automatically generating the amount of robot autonomy in the macro-action controllers. Through data captured from the first user study, we use a decision tree to capture the different driving behaviors of beginners and experts. We then programmed simulated "bots" to act as those users. By running simulations with these bots, we used the results of simulations to find the controller values that gave the beginner and the expert bots the best performance.

- A study on the differences in user's perceptions of the robots and their performance while receiving no robot autonomy, a high amount of autonomy, and the amount of autonomy chosen by the expertise model with a second user study. This study compared the performance of the beginners and experts in a modified domain from our first study,

- We performed a field trial with a semi-autonomous underwater vehicle using our user expertise model. We showed the model choosing more robot assistance to provide the driver after the driver performed beginner actions.

Overall, the contributions made in this work provide a novel method for predicting human expertise and using that prediction to provide an appropriate level of shared autonomy to the user.

1.2 Thesis Summary

The work done in this thesis is organized as follows. Chapter 2 provides an overview of current methods and uses in shared autonomy. Additionally, Chapter 2 provides the fundamentals of the methods used in this work, the Markov Decision Process, the Partially Observable Markov Decision Process, and extracting data using decision trees. Chapter 3 goes over the user expertise model we developed and the first user study we ran to test it. Chapter 4 discusses our method of using user data to design Beginner and Expert automated "bots", and running simulations to generate robot autonomy levels used for controllers for beginner and expert users. Chapter 5 goes over the second user study we ran which looked to validate the results of our first study and observe user attitudes between users of different skill levels. Chapter 6 shows how we have implemented the expertise model on a semi-autonomous underwater vehicle and our results found from our first trial. Finally, Chapter 7 summarizes the contributions presented in this work and discusses the avenues for future work.

2 Background

This thesis builds on prior work in shared autonomy and human modeling. This first part of this chapter introduces readers to the past work done in shared autonomy and marine robotics. The second part of this chapter will provide the reader with background on Partially Observable Markov Decision Process, and Decision Trees.

We developed our algorithm for finding user expertise by modeling expertise as a Partially Observable Markov Decision Process, or POMDP. A POMDP is a generalization of a Markov Decision Process, or MDP. This chapter will first go over the basics of the MDP, then the POMDP [21].

2.1 Shared Autonomy

Many tasks benefit from collaboration between robots and humans, but often this collaboration comes with trade-offs. There are a number of ways to look at how much control should be shared between the human and a robot including: completing the goal optimally, reducing stress and fatigue for the human user, and maintaining human trust and satisfaction with the robot. There are numerous different methods to go about addressing these issues. Shared autonomy can reduce the workload for a human operator in tasks such as search and rescue [14], stable control of marine and aerial vehicles [12, 37], or driving semi-autonomous cars [3]. The robot can do much of the work that would have otherwise required human teleoperation, but the user still needs to maintain awareness and an understanding of what the robot will do. Maintaining awareness and understanding should not increase the workload for the human [38] so there is a balance in how much information the robot should provide a user without overloading them. However, what level of shared autonomy is needed in different situations is not always clear since the amount of control between the human and the robot can be very broad [4].

2.1.1 Adaption to Users

Improving human models provides better information that can inform how and/or when to provide robot assistance [31]. Much of the work in improving shared autonomy looks at being able to predict the user's intentions. Work from Dragan, Lee, and Srinivasa [8] showed that in human-robot collaboration, the robot needs the ability to predict the human user's intent and must also act in an intent-expressive way that enables the human to predict what the robot will do. Work done by Hauser [16] uses Gaussian mixture models to predict user intention in free-form tasks and a cooperative motion planner is used to generate trajectories based on the user's desired task. Predicting the user's intent has also been shown in assistive technology such as robotic wheelchairs [7, 32] in order to assist users with limited physical capabilities control their wheelchairs. In our work, rather than continuously predicting the user's actions, we use the prediction of their skill level in order to make a generalization of what actions the user will most likely perform. For instance, a user classified as a beginner is more likely to choose poor actions, so the model is used to choose an appropriate controller that provides greater assistance to the user.

Previous work has designed robots to adapt to different users based on their past actions. These methods work to model what the user prefers the robot to do. Robot autonomy often can decrease the workload and improve efficiency, but some users are unable or unwilling to cooperate with the robot. If the user is trying to work against the robot's actions, workload and frustration in the human will likely increase. Work done by Nikolaidis et al. [29] models the human's willingness to adapt to the robot's actions to improve the effectiveness of the team while retaining human trust in the robot. A method proposed by Javadani, Bagnell and Srinivasa [19] is used to find how the user reacts to assistive actions and how to choose the most assistive robot actions using the user model. Other work has looked at how robots can learn how to work best side-by-side with the human user. The cross-teaming method proposed by Nikolaidis et al. [30] is used to teach a robot to adapt to the preferences of the user. This method uses a Markov Decision Process to encode a mental model that learns the user's preferences in completing a certain task through training with the user. Our work similarly uses a Partially Observable Markov Process to model the human user, but we utilize the Markov process to find user expertise in a way that is general enough to be used on multiple systems.

2.1.2 Preference for Levels of Autonomy

Koo et al. [22] looked at how providing messages to a driver about how and why a shared autonomous vehicle would act affected performance and the driver's attitude on the system. Work done by Storms, Chen, and Tilbury [39] tested varying amounts of robot autonomy with a remotely operated ground vehicle with different amounts of communication delay. In their user study, they found that users felt they were better able to control events in the robot environment with autonomy located on the operator side at low time delay. However, at higher delay, users preferred having autonomy located onboard the robot over autonomy on the operator side. In our work, we looked at whether using expertise as a measure of how much autonomy to provide was the amount of autonomy the user prefers.

Previous work using shared autonomy has looked at the arbitration between the user's input and the robot's assistance [9, 20]. Such work has found that even when assistance from the robot decreases the task completion time, some users still preferred feeling in control of the robot. By encompassing the user's actions, both desirable and undesirable, the system can choose behaviors that both optimize reaching the goal and perform actions preferred by the user. A user study done by Takayama et al. [42] found that using assisted teleoperation resulted in fewer obstacles hit, but increased time for users to complete a map. The study also found that users with video game experience found the task with assistance more enjoyable and less physically demanding.

2.1.3 Mobile Ground Robots

Teleoperation of mobile ground robots is often used to control robots in areas that are dangerous or impossible for humans to go into. Mobile robots have been used in search and rescue [5] and inspection [41]. Shared autonomy can help reduce the workload of teleoperation for the human operators as well as utilize the capabilities of the human and robot to perform better than either could alone.

Semi-autonomous cars are an increasingly growing area of research in the field of shared autonomy. Vehicles using shared autonomy can intervene in dangerous situations [43] or reduce the amount of control the driver needs to provide [44]. In order for drivers to relinquish control to an autonomous car, they will need to understand and trust the autonomy. Work done from Koo et al. [22] looked at how providing messages to the driver about how and why the shared autonomous vehicle would act affected performance and the driver's attitude on the system. Using haptic feedback in shared autonomous cars has also been studied to improve performance when driver's actions conflict with the autonomous control [15].

2.1.4 Medical Robots

Shared autonomy is often used in medical robotics as well. The da Vinci [28] provides surgeons with improved visualization, enhanced dexterity, greater precision and ergonomic comfort, and reduces opportunities for human error. Work done by Shamaei et al. [35] has achieved shorter completion time of surgical tasks by automating repetitive tasks while having the surgeon act as supervisor. Rehabilitation robotics can assist in physical therapy with adaptive levels of shared autonomy that decrease assistance as the patient improves [10]. Shared autonomous wheelchairs can also improve driving for users with limited or unstable control [7, 32].

2.1.5 Aerial Robots

Aerial robots are a quick and affordable way to perform inspection, mapping, or delivery tasks in the air, and aerial robots are frequently bought by hobbyists. However, aerial robots can pose a danger if poorly controlled. Assistive shared autonomy has been used to improve the performance of the operator's control [46] or make it possible for one human to control multiple robots [12].

2.1.6 Marine Robots

Remotely operated vehicles (ROVs) can be used in a variety of underwater tasks including inspection, manipulation, and data gathering. Work in providing more autonomy to these vehicles is intended to reduce operator load and human error. Improved sensing capabilities and controllers can assist operators by having the robot perform some tasks autonomously [34]. Our work uses the semi-autonomous vehicle, the Seabotix vLBV300 shown in Figure 6.1. The autonomous capabilities of the vehicle, station keeping and waypoint following, can provide assistance to vehicle operators [24]. The Seabotix vLBV300 has been used in manipulation tasks, data collection, and search and recovery tasks. The Seabotix vLBV300 was used by Streenan and Du Toit [40], where they applied a model to predict diver movement underwater. Their method helps the Seabotix vLBV300 avoid potential collisions with the diver making it possible for safer human-robot teaming underwater.

2.2 Basic Theory

2.2.1 Markov Decision Process

An MDP is a framework used for decision making for an agent interacting synchronously with a world. The MDP is represented as a tuple $< S, A, T, R >$, where S is a set of states of the world, A is a set of actions, and T is the state-transition function. For each state, s, and action, a, there is a probability distribution over all states. Thus, if the world is currently in state s and action a is taken, there is a probability of $T(s, a, s')$ that the state will transition to state s'. R is the reward function. At each state, there is a reward for taking an action. If action a is taken at state s there is some reward $R(s, a)$. In an MDP model, the next state and the expected reward depend only on the previous state and the action taken. This property is known as the Markov property.

The purpose of using the MDP is to find a policy π that will maximize the total reward over a possibly infinite horizon. Thus, the goal is to maximize the equation:

$$\sum_{t=0}^{\infty} \gamma^t R_{a_t}(s_t, s_{t+1}), \tag{2.1}$$

where γ is a discount factor between 0 and 1. To find the optimal policy, various algorithms are used by finding the values of equations for the policy, π, and value, $V(s)$:

$$\pi_t^*(s) = \operatorname*{argmax}_a \left[R(s,a) + \gamma \sum_{s' \in} T(s,a,s') V_{t-1}^*(s')) \right] \tag{2.2}$$

$$V^* = \max_a \left[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^*(s') \right] \tag{2.3}$$

A commonly used method of solving a MDP is the value iteration method in which V is continuously calculated until the difference of V_t and V_{t-1} is below a set threshold.

2.2.2 Partially Observable Markov Decision Process

The Partially Observable Markov Decision Process (POMDP) is a generalization of the MDP where the current state s is not directly observable. To optimize the actions taken by the policy, the POMDP maintains a belief of the current state. Through observations mapped to actions, the POMDP maintains a belief, b, of probabilities over the set of states. The POMDP is described as the tuple $< S, A, T, R, \Omega, O >$, where S, A, T, and R are the components of the MDP. Ω is a set of observations the agent experiences that provides some information about the state, and O is the set of conditional observation probabilities for each action and resulting state. There is a probability $O(s', a, o)$ of making observation o given that the agent took action a and resulted in state s.

The key to the POMDP is the belief update that results from taking an action and an observation. The Markov property maintains that a belief over the states only requires the previous belief state, the action taken, and the observation. Given the prior belief $b(s)$, taking action a, and observing o, the belief is updated by the equation:

$$b'(s') = \eta O(o \mid s', a) \sum_{s \in S} T(s' \mid s, a) b(s), \tag{2.4}$$

where

$$\eta = \frac{1}{\sum_{s' \in S} O(o \mid s', a) \sum_{s \in S} T(s' \mid s, a) b(s)}. \tag{2.5}$$

The POMDP can be solved as an MDP, where the set of states is the set of belief states. We utilize this method to model the uncertainty of a human user's expertise. We set the states as Beginner or Expert, and so the belief will represent the certainty a user is either a Beginner or Expert. One of the drawbacks of POMDPs is that they are often computationally demanding to solve. For this reason, we designed our POMDP model to use macro-action controllers, rather than solving over all possible robot actions.

2.2.3 Learning From Small Amounts of Data

When deciding how to design simulated user bots to act like Beginners and Experts, we wanted to learn what ways the the users were different in how they controlled the robot. Much has been done in machine learning and classification relies on having large amounts of data, but gathering a large amount of data from humans can be difficult. Therefore, we looked into methods of learning and classification that perform well with small amounts of data.

One-shot learning is one such method developed to train with a smaller data set. One-shot learning makes use of knowledge transfer, which uses the parameters used in previous models to learn over fewer examples [11]. One-shot learning has been used to train neural networks for learning manipulation tasks [13] and with learning from demonstration to play tic-tac-toe and solder [45]. One-shot learning has been successful in reducing the amount of training needed to learn classifications.

Data can be organized into groups of similar features using clustering. Clustering is another learning method which does not require large amounts of data. One widely used method is K-means clustering [25]. With K-means clustering, the mean of clusters can be found for different groups, and these means can be used as the learned averages for the groups. Some drawbacks to K-means clustering are that it will attempt to cluster into groups even if no difference actually exists and it relies on the assumption that the grouping has roughly the same number in each group.

We use a decision tree to find the values to program into simulated user bots [26]. A decision tree is a decision-making device that assigns a probability to each choice based on the context of that decision. The decision tree provides an easy-to-parse format and clearly represents the differences found between the Beginners and the Experts from the data collected. Decision trees are often used to create a model that predicts the value of a target based on input variables. We use the decision tree to find how Beginners and Experts differ in their control, and we can then design simulated bots with those differences. With the average values found for the differences of Beginners and Experts, we design bots with the knowledge of how the bot would be classified in the decision tree based on its assigned behavior.

3 Modeling User Expertise

This chapter will present the POMDP user expertise model developed and the first user study that was run to test its functionality. The work presented in this chapter was published in [27].

The remainder of the chapter is organized as follows. We present the algorithms used to model user expertise and develop a policy for choosing levels of shared autonomy (Section 3.1). Then we present the domain that we ran the user study in (Section 3.2) and give a description of the macro-action controllers designed with varying levels of autonomy (Section 3.3). The study procedure is described (Section 3.4) and the results of the study are presented (Section 3.5). Finally, we discuss the implications of the results (Section 3.6).

3.1 Algorithm Description

3.1.1 User Model

We leverage the POMDP framework to learn a human's level of expertise and use this model to determine the level of autonomy to give the robot. The POMDP model is a tuple $<S,A,O,T,\Omega,R,b_o,\gamma>$. The set of states, S, encompasses the user's level of expertise. The set of observations, O, includes the observations from the environment as well as the actions performed by the user, which reflect either expert or novice skills. These observations may include how many times the user came close to hitting or hit an obstacle, how quickly they completed the goal, or if they are operating the robot at constant speeds or rapidly accelerating. The actions, A, are macro-actions controllers, controllers that encompass the level of control shared between the robot and the human. These controllers are developed for users at different skill levels or as transitional controllers when users are moving between skill levels. T, Ω, R, b_o, and γ represent the conditional transition probabilities between states, the conditional observation probabilities, the reward function, the initial belief, and the discount factor, respectively. Currently, the values of the POMDP tuple are hand-tuned, but could potentially be learned from data. The hand-tuned values were designed from results of pilot studies run on members of our lab and from our own intuition on how we wanted to define expertise.

3.1.2 Shared Autonomy Policy

With knowledge of the user's skill level, we can use controllers that will best assist the user based on what actions they are most likely to perform. For our experiments, we used the Approximate POMDP Planning Toolkit (APPL) as an offline POMDP solver [1]. APPL implements the SARSOP algorithm [23] to approximately solve the POMDP. Hand-tuned values of the POMDP tuple were input into APPL to generate a policy offline.

We use the POMDP and the policy generated offline to update the belief of the user's level of expertise as they operate the robot. At each time step, the policy π is used to select the macro-action controller $a \in A$ based on the current belief state b.

Algorithm 1 User Expertise Prediction and Controller Selection

```
1: procedure ChooseController($\pi$,$b$)
2:     $a \leftarrow \pi(b)$
3:     while not at update condition do
4:         Use macro-action controller, $a$
5:         Record observation, $o$
6:     end while
7:     $b' = \tau(b, a, o)$
8:     $b \leftarrow b'$
9: return $b$
10: end procedure
```

The controller provides some level of shared autonomy to the user. When the time step completes, the observation $o \in O$ is received based on the user's actions, the actions taken by the controller, and the observation of the environment. The belief state is updated based on the current belief, the controller used, and the observation given by $b' = \tau(b, a, o)$. The process repeats with the new belief state. The complete algorithm is described in Algorithm 1.

3.2 Domain

For our experiments, a robotic driving simulator was created using the pygame library of Python [36]. A picture of the simulator is shown in Figure 3.1. The user drives the robot using the WASD keys on the computer keyboard, with "W" accelerating forward, "S" accelerating in reverse, "A" rotating the robot counter-clockwise, and "D" rotating clockwise. The objective of the map is to reach the goal as fast as they can.

The states of the world represent the user's expertise and the difficulty of the map.

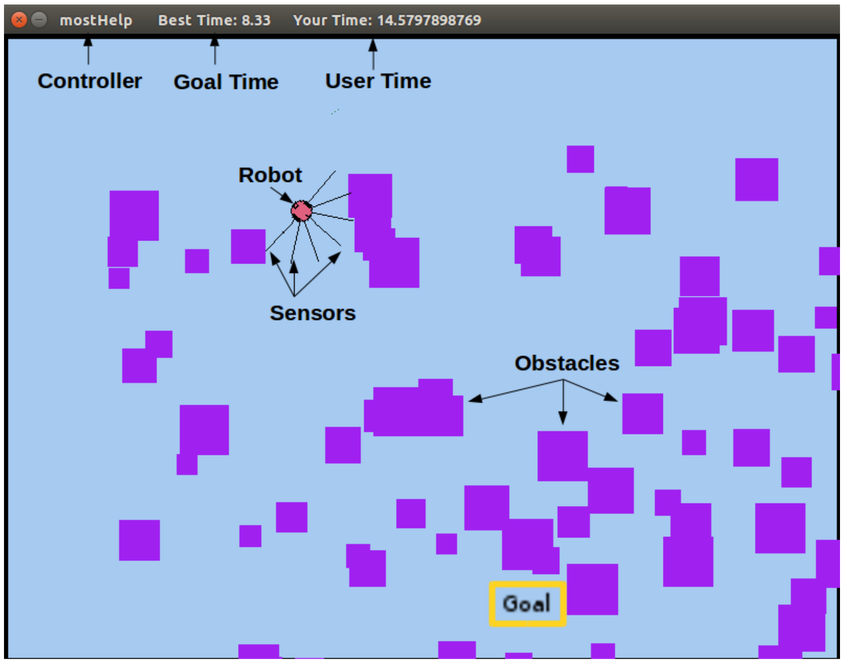


Figure 3.1: Simulated driving environment used to test macro-action POMDP.

Easy maps contained fewer, more spread out, obstacles while hard maps contained a greater number of obstacles that require finer maneuvering to avoid. The observations in the model represented the perceived difficulty of the map and the number of times the user collided with an obstacle. Four controllers were created that provided different levels of assistance in avoiding obstacles. These controllers are described in more detail below. Table 3.1 shows the states, actions, and observations chosen for our experiments. The values of the POMDP were set so that the policy assigns users with a high probability of being at the Expert state the "LeastHelp" controller. Those with a high probability of being at the Beginner state are assigned the "MostHelp" controller. When the probability is distributed more evenly between Beginner and Expert, the "SomeHelp" or "MoreHelp" controllers are assigned.

Table 3.1: POMDP for modeling user expertise

S	Beginner, Expert
A	LeastHelp, SomeHelp, MoreHelp, MostHelp
O	No Obstacles Hit-Easy, Few Obstacles Hit-Easy, Many Obstacles Hit-Easy, No Obstacles Hit-Hard, Few Obstacles Hit-Hard, Many Obstacles Hit-Hard

3.3 Robot Controllers

The macro-action controllers can be set up for different user levels and environmental states. By selecting macro-action controllers rather than low-level motions, the number of actions the POMDP must solve for can be greatly reduced [2]. In a human-robot system, some of the burden of modeling the state of the world is taken over by the human rather than relying on the POMDP alone. Our macro-action controllers combine the actions of the user and the robot so that the POMDP requires fewer states and observations to navigate through the environment. By encapsulating the user skill level in our model, we can predict a range of probabilities that certain actions will be performed by the user depending on their skill and the difficulty of the task.

The macro-action controllers were created with the possible differences between beginners and experts in mind. The principal difference being that beginners would be more likely to hit obstacles. We note that our model is general enough to incorporate many different controllers and observations across domains, and future work could incorporate more complex observations of novice and expert actions.

In order to assist the user in avoiding obstacles, the controllers use a potential field method similar to the method presented by Crandall and Goodrich [6]. When obstacles are within range of the robot, the controller attempts to slow the robot down and steer it away from the incoming obstacle. To balance the control between the user and the robot, a variable $U = [0, 1]$ defines the amount of user influence. As the user presses the control buttons on the keyboard, U increases. Once the button is released, U decreases. The user influence affects both the velocity and the heading of the system with U_v representing the user's influence over velocity (when pressing the W or S keys to accelerate) and with U_r representing the user's influence over rotation (when pressing the A or D keys to rotate). If they hold the key long enough, the user can override the robot's actions when it tries to slow or rotate. The variable v is the current speed of the robot, α_1 is the factor for the amount slowed when approaching an obstacle from any side, and α_2 is the variable for the amount slowed when approaching an obstacle from the front of the robot. The speed of the robot is calculated as

$$v_{new} = v - (1 - U_v)(\alpha_1 + \alpha_2)sign(v). \tag{3.1}$$

The angle of rotation from the user's control is defined as θ_U, the repulsion angle from the potential field of the obstacles is defined as θ_R, and β is a weighting factor for the amount of control given to the robot. Below is the final angle of rotation for the robot leveraging both the user's and the robot's steering commands:

$$\theta = U_r\theta_U + \beta(1 - U_r)\theta_R. \tag{3.2}$$

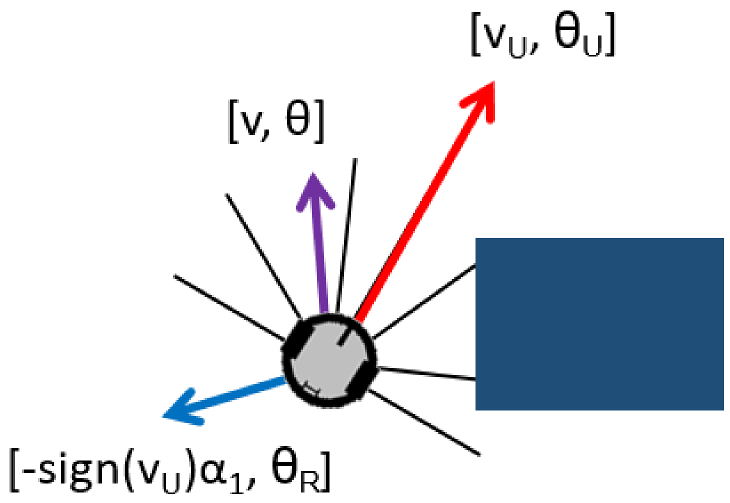


Figure 3.2: The effect of the POMDP macro-action controllers. v_U and θ_U are the velocity and rotation command from the user, θ_R is the robot's rotation command, and v and θ are the combined commands. If the user drives towards an obstacle, the force of the robot turns away from the obstacle and slows the robot down. Since the robot is approaching the obstacle from the side, α_1 is activated, but α_2 is not.

The controller's effect on the robot's velocity and steering is shown in Figure 3.2. While the controllers designed for the beginner states were able to avoid obstacles, they also took away some of the control from the user and decreased the speed of the robot. This decrease in control may be preferred by a beginner who may require the robot to assist them, but could slow down an expert user. The differences in the controllers are shown in Table 3.2. These parameters were derived from testing of the simulator in pilot trials. We found values for "MostHelp" that made it much less likely to hit obstacles, but were not so large that the user no longer had control. Then these values were decreased until reaching the "LeastHelp" controller that would only slightly slow down users when approaching obstacles directly in front of the robot.

Table 3.2: The parameters set for each of the controllers. The variable α_1 controls the decrease in speed when approaching an obstacle from the side, α_2 controls the decrease in speed when approaching an obstacle from the front, and β controls the amount to turn away from obstacles.

$Controller$	$Parameters$
MostHelp	$\alpha_1 = 2$, $\alpha_2 = 0.80$, $\beta = 2.0$
MoreHelp	$\alpha_1 = 2$, $\alpha_2 = 0.60$, $\beta = 1.5$
SomeHelp	$\alpha_1 = 1$, $\alpha_2 = 0.45$, $\beta = 1.0$
LeastHelp	$\alpha_1 = 0$, $\alpha_2 = 0.25$, $\beta = 0.0$

3.4 Study Procedure

A user study was designed to verify that (1) the POMDP model is able to make accurate predictions of user expertise, (2) users predicted by the POMDP as more likely to be experts perform better than the users predicted more likely to be beginners, and (3) using the different controllers will result in differences in user performance.

Fourteen able-bodied users (6 male, 8 female, mean age 24) were recruited to participate in the user study through emails and fliers distributed through Oregon State University. The participants were compensated $5 USD for the 30-minute experiment. The users were given a brief overview of the robot's level of autonomy and how to drive the robot using the WASD keys.

The participants were given the robot driving simulator described in Section III. The participants were given a view similar to Figure 3.1. At the top of the screen, the user is shown which controller they are currently using, a "best time", and their own time. The controller that is being used by the robot is shown to the user to give the robot's autonomy some transparency. Pilot trials of the simulator showed us that when the users were not shown which controller was being used, they would often become frustrated when a different controller was chosen. Often they would not realize a new controller was chosen and would drive the robot as if it would behave the same as the last map. This often led to mistakes before the user realized the controller was different. The user's time and the goal time are shown to motivate the users to complete the map quickly. The same maps were presented in the same order for every user.

The users first drove the robot through the "training maps", 30 pre-generated maps of varying difficulty. The initial belief state is set at the beginning of the trial with equal probability of the user was in the Beginner or Expert state. Each user also starts with the "SomeHelp" controller. The belief state is updated once the user completes the map, and the controller to be used in the next map is chosen by checking the policy with the new belief state.

After completing the training maps, the user drove through 5 "test maps". Instead of using the POMDP model to choose the controller, they were given either the "LeastHelp" or the "MostHelp" controller. The users then ran those same 5 maps a second time with the other controller. During the test maps, the controller used first is counterbalanced so that half the users started with the "LeastHelp" controller and half started with the "MostHelp" controller. Therefore, the results will not be skewed from seeing the maps a second time. We used the controllers with the most/least autonomy in the test maps so that we could compare how the amount of autonomy affected the users.

3.5 Results

3.5.1 Training Maps

Figure 3.3 shows the total number of obstacles hit versus the mean probability of the user being a Beginner during the training maps. Figure 3.4 shows the total time in seconds to complete the training maps versus the mean probability of the user being a Beginner. Though there is variance in the results, the plots show that the users who have a higher probability of being a Beginner did perform worse than users who had a lower probability. Users who had a high average probability of being a Beginner tend to hit more obstacles and take longer to complete the maps than those with a low average probability. The probability that the model predicts the user is an Expert over the training maps is shown for three of the users in Figure 3.6. User 1 has a high probability of being an Expert through most of the maps. User 2 has a low probability of being an expert for the first 15 maps, but then the probability increases, showing that they have improved. User 3 may have performed well on some maps, but never performed consistently enough for the POMDP to achieve a high probability of being an Expert by the end of the maps.

3.5.2 Test Maps

We pose three hypotheses regarding the validity of using the different controllers for different users. We looked at the number of obstacles hit, the total time to complete the maps, and the total amount of user input (i.e. the amount of time the user was

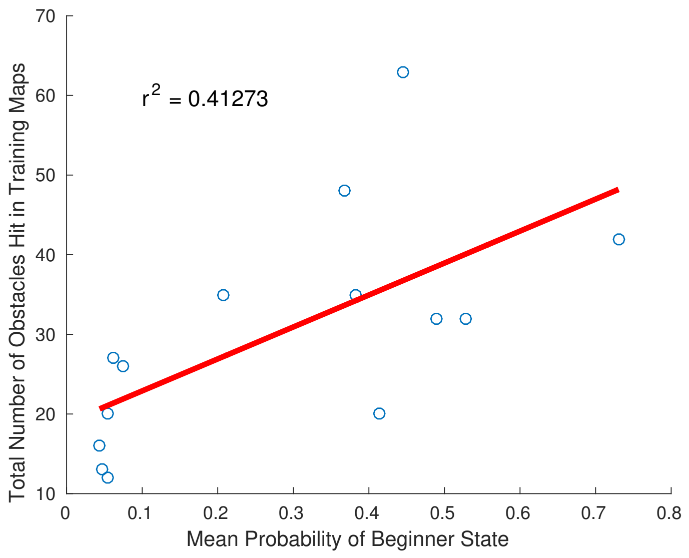


Figure 3.3: The total number obstacle hits in the training maps versus the mean probability of being a Beginner.

pressing one of the WASD keys). The “MostHelp” controller was designed with the intention of helping users avoid obstacles at the cost of slowing the robot more and giving more control to the robot. The “LeastHelp” controller was designed for users more skilled at avoiding obstacles. Since these users need little or no help to avoid obstacles, there is less need to slow down or give the robot more control. Because the robot’s autonomy was presented as the robot “helping”, the names Most, More, Some, and LeastHelp were given to show the user how much the robot would be helping them avoid obstacles. However, the robot’s increased autonomy also slowed the robot down more and requires more user input to counteract if the robot tries to turn away from obstacles against the user’s wishes. While the robot’s assistance does help to avoid obstacles, it can also increase the time and amount of user input needed to complete the

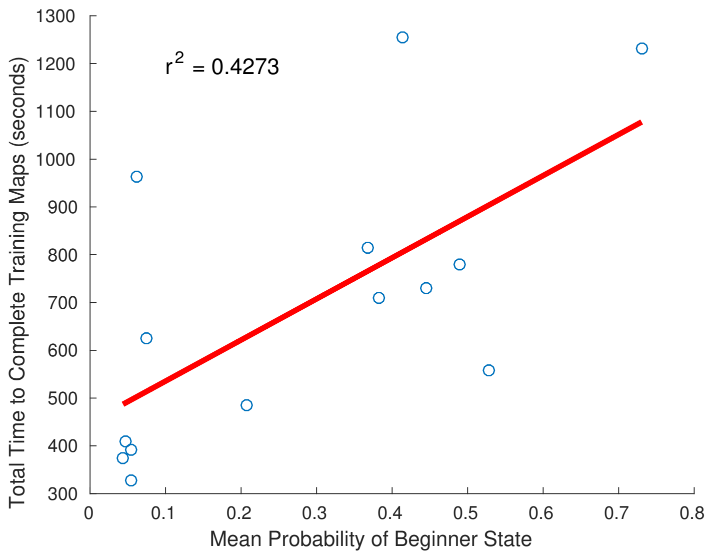


Figure 3.4: The total time to complete the training maps versus the mean probability of being a Beginner.

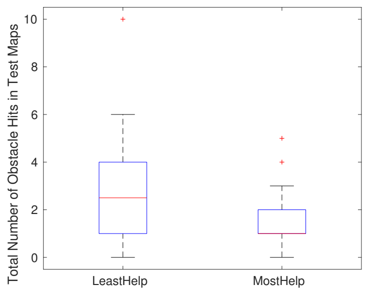
maps. Three hypothesis were proposed.

Hypothesis 1a *Using the "MostHelp" controller will result in fewer obstacles hit than using the "LeastHelp" controller.*

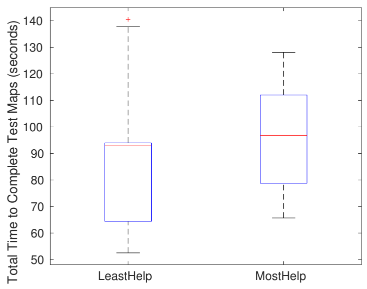
Hypothesis 1b *Using the "LeastHelp" controller will result in shorter time to complete the maps than using the "MostHelp" controller.*

Hypothesis 1c *Using the "LeastHelp" controller will result in less user input to complete the maps than using the "MostHelp" controller.*

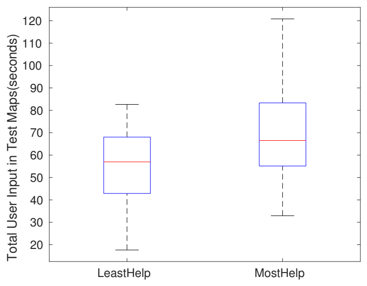
A paired t-test was conducted to compare the users' results from the test maps while using the "LeastHelp" controller and the "MostHelp" controller. We use a p-value<0.05



(a) Number of Hits (p = .033)



(b) Total Time (p = .095)



(c) User Input (p = .008)

Figure 3.5: The performance using the “LeastHelp” and “MostHelp” controllers. There is a statistically significant difference between the number of obstacle hits (3.5a) and the total user input (3.5c). We can take away from these results is that using the “MostHelp” controller reduces the number of obstacles hit, but using the “LeastHelp” controller reduces the amount of user input needed to complete a map.

as a threshold for statistical significance. The results are shown in Figure 3.5. There was a statistically significant difference in the number of obstacles hits for “LeastHelp” (M = 3.14, SD = 2.60) and “MostHelp” (M = 1.64,SD = 1.50) conditions; *t*(13)=2.39, *p*=.033. There was also a statistically significant difference in the total user input for “LeastHelp” (M = 55.68, SD = 18.10) and “MostHelp” (M = 71.97, SD = 25.88) conditions; *t*(13) = -3.16, *p* = .008. There was not a statistically significant difference in the total time for “LeastHelp” (M = 90.13, SD = 29.21) and “MostHelp” (M = 97.89, SD = 1.50) conditions; *t*(13) = -1.80, *p* = .095. Hypothesis 1a and 1c are supported, but Hypothesis 1b is not supported. These results suggest that while using the “MostHelp” controller, a user will hit fewer obstacles; however, when using the “LeastHelp” controller the amount of time and user input needed to complete the map decreases. For users who are able to avoid obstacles well on their own, the “LeastHelp” controller is more likely to be the better fitting option.

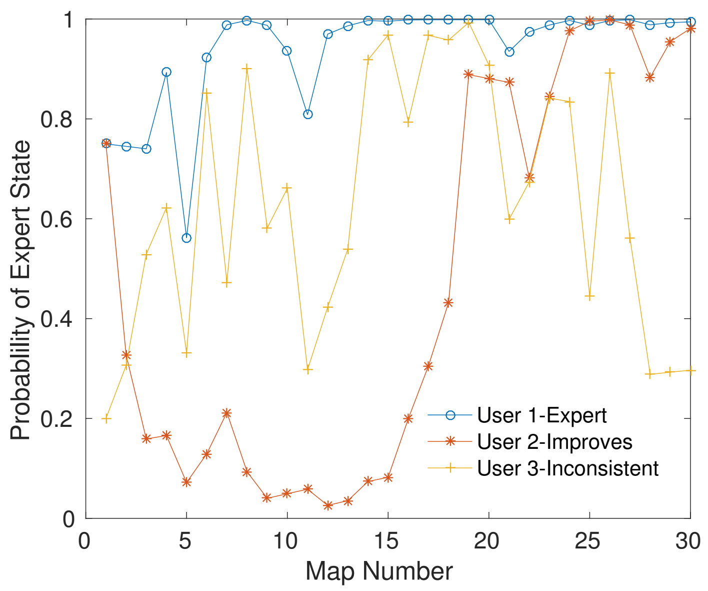


Figure 3.6: The POMDP's predicted probability of being at the Expert state over the training maps for 3 users.

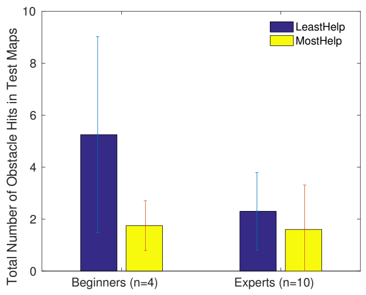
We compared the results of users at different skill levels to see if the two controllers made significant improvements for users of higher and lower expertise. We grouped the users into "beginners" and "experts" based on the mean probability of user state of maps 20-30 performed in the first part of the experiment. Users who had reached a mean Expert probability above 0.9 were classified as "experts" (10 users), while those below were classified as "beginners" (4 users). Looking at the Figure 3.6, those such as User 1 (high probability of Expert through most of the training maps) and User 2 (showed improvement and had a high probability of Expert at end of training maps) were classified as experts. Those such as User 3 (did not improve greatly) were classified as beginners. We posed three more hypothesis to compare the controllers between the different kinds of users.

Hypothesis 2a *The difference in the number of obstacles hit between using the "MostHelp" and the "LeastHelp" controller will be greater for users classified as beginners than users classified as experts.*

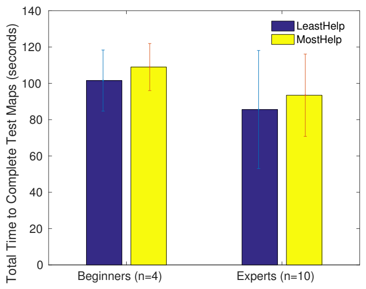
Hypothesis 2b *The difference in the time to complete the maps between using the "MostHelp" and the "LeastHelp" controller will be greater for users classified as beginners than users classified as experts.*

Hypothesis 2c *The difference in the user input to complete the maps between using the "MostHelp" and the "LeastHelp" controller will be greater for users classified as beginners than users classified as experts.*

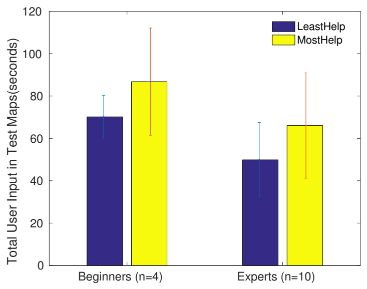
An unpaired t-test was conducted to compare the difference between using the "LeastHelp" controller and the "MostHelp" controller for the users classified as Beginners and Experts. Figure 3.7 shows the differences between the user types. There was a statistically significant difference in the difference of obstacles hits for Beginners (M = 3.50, SD = 3.11) and Experts (M = 0.70, SD = 1.49); *t*(12) = 2.34, *p* = .037. There was not a statistically significant difference in the difference of total time for Beginners (M = -7.45, SD = 9.49) and Experts (M = -7.88, SD = 18.58); *t*(12) = 0.043, *p* = .966. There was not a statistically significant difference in the difference in user input for Beginners (M = -16.61, SD = 29.49) and Experts (M = -16.17, SD = 15.72); *t*(12) = -0.037, *p* = .972. Hypothesis 2a is supported with statistical significance suggesting that for the Beginner users, using the "MostHelp" controller provides more improvement by reducing of the number of obstacles hit. Hypothesis 2b and 2c were not supported. What these results suggest is that users who are less skilled receive more assistance from the



(a) Number of Hits (p = .037)



(b) Total Time (p = .966)



(c) Total User Input (p = .972)

Figure 3.7: The performance from users grouped into Beginners and Experts. The error bars show the standard deviation. The change in performance from the two controllers is compared for Beginners and Experts. There is a statistical difference in the number of obstacles hits (3.7a). There is not a statical difference in the total time (3.7b) and total user input (3.7c). These results suggest that Beginners receive the most amount of assistance in avoiding obstacles when using the "MostHelp" controller, but the reduction of time and user input is not significantly greater for Beginners and Experts.

robot autonomy in avoiding obstacles than a more skilled user would receive. Since Experts do not improve as much as Beginners from the increase in the robot's control, a controller that lets them complete the maps more quickly may be the more appropriate controller to use.

3.6 Discussion

The results from the user trial show that our model can distinguish between users of different skill level. Our results have shown that the users that have lower probabilities of being an Expert hit more obstacles and take more time to complete the maps. The results also showed that our designed macro-action controllers were able to reduce the number of obstacles hit with the robot's assistance, but this assistance also increased the amount of time to complete the maps and the amount of input the user had to give. Though this was the case for all users, we found that the users with a lower probability of being an Expert received more help in avoiding the obstacles than those with a higher probability of being an Expert. There was no statistically significant difference between the two types of users when looking at the amount of time and user input, but for Expert users, a controller that allows the Expert to complete the maps faster should be chosen since there is little difference in the number of obstacles hit. These results support our idea that using a macro-action POMDP to predict user expertise can be useful when leveraging levels of shared autonomy.

4 Learning Controller Values for Beginners and Experts

In the previous user study, the controller values were hand-tuned. In this chapter, we present a method to automatically generate these controller values by utilizing our collected user data. With the data, we design simulated user "bots". These bots perform similarly to how the average Beginner and Expert users behaved. With the simulated bots, we can run many trials with a range of variables for the different controllers. We find which controller values minimized cost for Beginners and Experts and assign the newly generated controllers as the macro-action controllers for the POMDP model.

Using our method, the designer can change the importance of different features for the controller. For example, one can optimize for reducing the likelihood of hitting an obstacle over completing a map quickly. We used the data gained from the first user study in order to design Beginner and Expert bots. From the results of simulations done with these bots, we determined which controller values gave the best results for varying values of importance on time, distance, and obstacles hit.

This chapter is organized as follows. The method describing how the bots were created and how the controller values were synthesized is presented (Section 4.1). We next present two controllers generated from data from the first user study (Section 4.2). Finally, conclusions and future directions are discussed (Section 4.3).

4.1 Method

4.1.1 Designing Bots

We used the data collected from the user study performed in Chapter 3 to find auto-generated controller values for Beginners and Experts in the robot driving domain. From the data gathered from the user study, we used the average robot speed and the speed variance to create a decision tree. A decision tree assigns a probability to each choice, in our case is the user a Beginner or Expert, based on the context of that decision, the user's speed values. The label of Beginner and Expert were assigned to the data based on how they were classified in the user study. This data was used to generate the decision tree shown in Figure 4.1. On the tree, samples are the number of training samples found within that node of the tree, and value shows how many of those samples are in each group. The first value is the Expert users and the second values are the Beginners.

The bots were designed to match the user control used in the study and performed certain actions based on average speed and speed variances found for Beginner and Expert users in the decision tree. When a bot is created, i.e. before starting a simulation in each map, the bot is assigned an average speed and variance based on the map's difficulty. For example, if the map is hard and the bot is an expert, there is a 60% chance its average speed will be assigned as a value less than 6.33.

In each map, the bots are given a certain path to follow to reach the goal. Whether the robot speeds up, slows down, or drifts depends on: the average speed for the type of bot (Beginner or Expert), the speed variance for the bot, and if they need to turn.

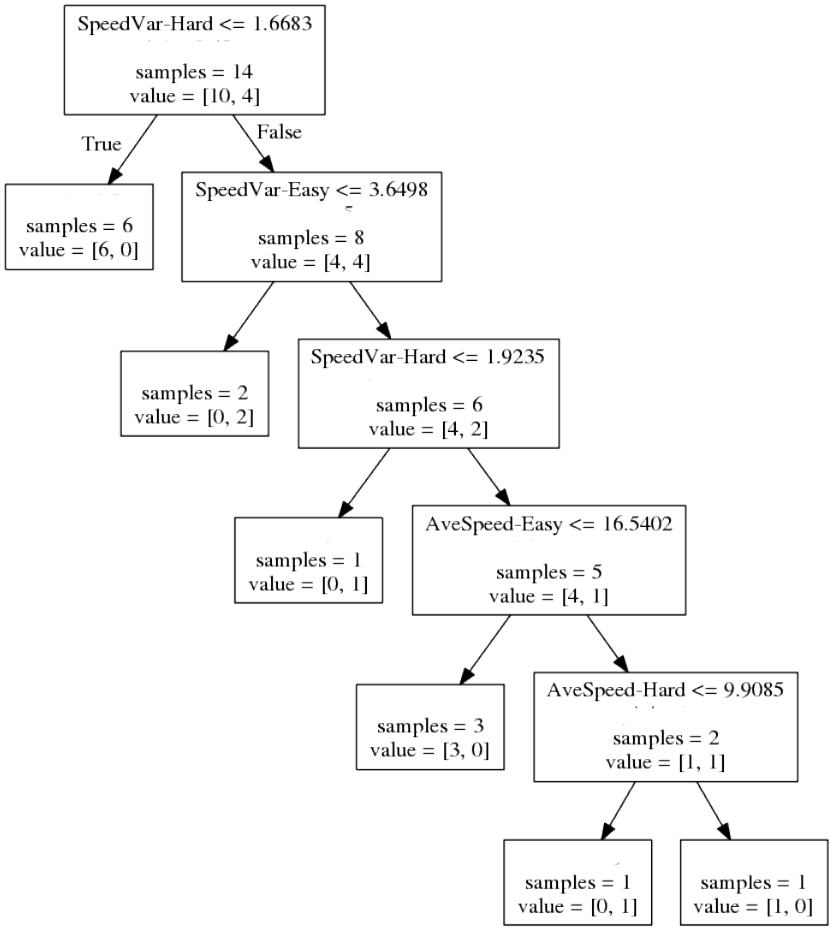


Figure 4.1: Decision tree generated from user data collected during the first user study.

Algorithm 2 shows how the bot chooses to speed up, slow down, or do nothing in an easy map.

Algorithm 2 Bot control in map

```
1: procedure BOT CONTROL MAP EASY(aveSpeed,stdSpeed)
2:     if $v < speed$ then
3:         SPEED UP
4:     else if $v \leq speed$ AND $v < (speedE + stdSpeed)$ then
5:         if RAND $>$ speedChangeProb then
6:             SPEED UP
7:         else
8:             Do Nothing
9:         end if
10:    end if
11:    if Close To Obstacle then
12:        if RAND $>$ turnProb then
13:            Turning = TRUE
14:        end if
15:    end if
16:    if $Turning$ AND $v > 0$ then
17:        if RAND $>$ speedChangeProb then
18:            SLOW DOWN
19:        end if
20:    end if
21: end procedure
```

4.1.2 Synthesizing Controllers

After the bots have run through the simulations, the data points are used to find the controller values for Beginners and Experts. The total distance traveled, D, the total time for the set of maps T, and the total number of obstacles hit, H, are taken from the simulated data from the controller settings. The variables α_1, α_2, and β are the controller values that encompass the amount of assistance provided by the robot. The level of importance of the different variables, d, t, and h, are values that when added equal 1.

The values of these variables can be altered based on the importance of its function in that intended controller. For instance, a controller designed for a Beginner may rank minimizing obstacles hit over time and distance, whereas this may be opposite for an Expert controller. Below is the minimization equation to assign the controller variables α_1, α_2, and β to the new controller for either a Beginner or Expert:

$$NewController(Expertise) = \min_{\alpha_1,\alpha_2,\beta} \left[d\hat{D} + t\hat{T} + h\hat{H} \right]. \tag{4.1}$$

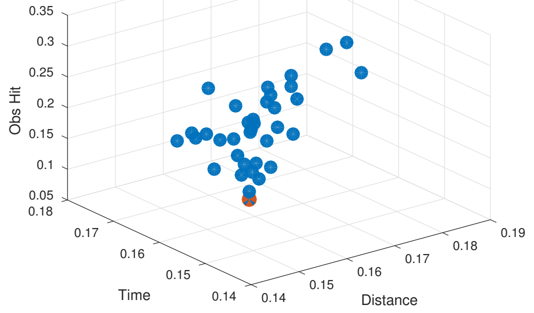
4.2 Results

We designed a Beginner bot and an Expert bot using the speed values found from the decision tree shown in Figure 4.1. The controller values of α_1, α_2, and β range from (0,3), (0,1), and (0,3), respectively. Figure 4.2 plots the normalized values of the total distance, time, and obstacles hit for each of the controller values.

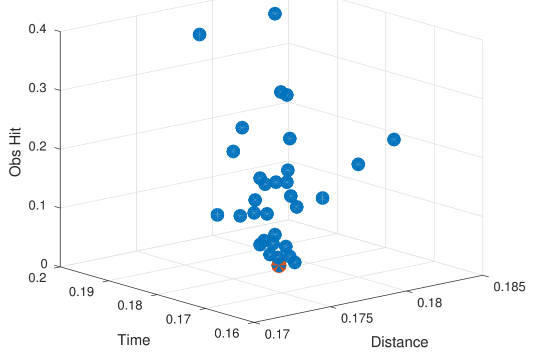
Table 4.1 shows the found controller values for various values of $d, t,$ and h. It can be seen from the table that the Beginner bot benefited most from a α_2 value of 0.75, while the Expert had better results at 0.5. For both the Beginner and Expert bots, time is optimized when there is no robot control.

4.3 Discussion

The work in this chapter provides a framework for automatically generating controller values for users of different expertise levels. This method does require user data from the system that one would want to design controllers for.



(a) Beginner



(b) Expert

Figure 4.2: The normalized values for the total time, distance, and obstacles hit with the bots. The red dots are found minimum for $(d, t, h) = (.33, .33.33)$

Table 4.1: Synthesized Controller Values

(d, t, h)	Beginner(α_1, α_2, β)	Expert(α_1,α_2, β)
.33, .33, .33	2, .75, 1	0, .5 , 1
1, 0, 0	2, .25, 1	1, .5, 1
0, 1, 0	0, 0, 0	0, 0, 0
0, 0, 1	1, .75, 2	1, .5, 1
0.25, 0.25, 0.5	1, .75, 2	1, .5, 1
0.5, 0.4, 0.1	2, .75, 1	0, .5, 1

So far, we have only designed controllers from the 14 subjects of the user study. Even then, we only had the data of 4 users classified as Beginners, and the data collected was not a comprehensive as it could have been. For example, we only had information on the average speed, and not how it related to when a user was near obstacles or the goal. The next chapter will present the work done on a second user study, we recorded the data from 28 users and recorded a great deal of data at each time step including user control and distance from obstacles and the goals. With this data, we can design more accurate bots, and therefore more accurate controllers.

5 Comparing Beginner and Expert Responses to Shared Autonomy

This chapter will present our second user study. This user study was designed to expand upon the user study done in Chapter 3. Using an expanded domain with new functionalities, we confirm the results of the first user study and observe subjective measures from users of different skill levels.

The results of the first user study showed the controller with more assistance the Beginner users hit fewer obstacles using the most assistive controller, but our tests were unable to show if the users were satisfied with the assistance. Even if a user's performance increases with higher levels of autonomy, they may choose not to use it if they do not like it. When choosing a level of shared autonomy for users, it should not only improve performance, it should also provide the amount of assistance that users are more satisfied with. These two criteria might not always match up, and the user study will look at how often they align.

This chapter is organized as follows. First, the domain used in the user study is presented (Section 5.1). The procedure of the user study is then provided (Section 5.2), and the results are presented after that (Section 5.3). Finally, conclusions and possible future directions discussed (Section 5.4).

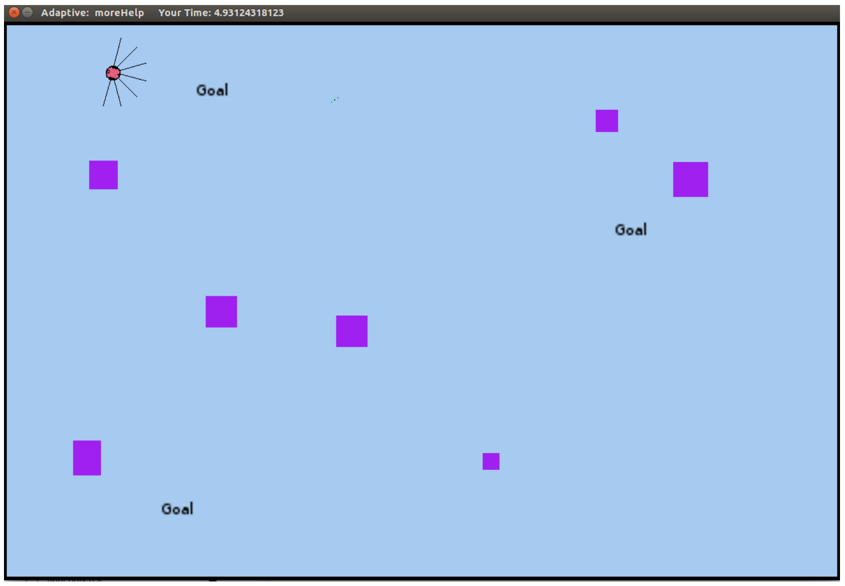


Figure 5.1: Robot driving scenario for the second user study.

5.1 Domain

For our experiments, we modified the pygame domain presented in Chapter 3.3. A picture of the simulator is shown in Figure 5.1. The control of the robot is changed to a Playstation 3 controller, with the left joystick being used to increase or decrease the speed and the right joystick used to change the heading of the robot. The objective in this domain is to drive to the three areas on the map marked “Goal” in any order.

The states in this domain remained the same as the one presented in Chapter 3, the user’s expertise and the difficulty of the map. The observations in the model represented the perceived difficulty of the map and instead of just counting obstacles hit, this model used a total Beginner score made up of 5 different actions the user may take. These actions include: hitting an obstacle, driving too quickly when close to a goal, having the front of the robot close to obstacles, moving slowly towards the goal, and not pointed towards the goal or pointed towards an obstacle. The total combination of how often the user performed these actions on the map contribute to a Beginner score, B. The number of times these actions occur, b_n, are given different weights, λ_n, indicating how much they indicate the user is a Beginner as well as how often they potentially occur.

$$B = \sum_{n=1}^{5} \lambda_n b_n \tag{5.1}$$

5.1.1 Controllers

For this study, we expanded the capability of the macro-action controllers. Along with slowing down when close to obstacles and turning the robot away from them, the autonomous control now will also: slow the robot down if approaching a goal point quickly (α_3), increase the speed of the robot when moving slowly and far from a goal point (α_4), and will turn the robot towards the goal (β). Table 5.1 are the values for each of the controllers used in the study. The values of α_1, α_2 and β are from the values found using the controller synthesis shown in Chapter 4.

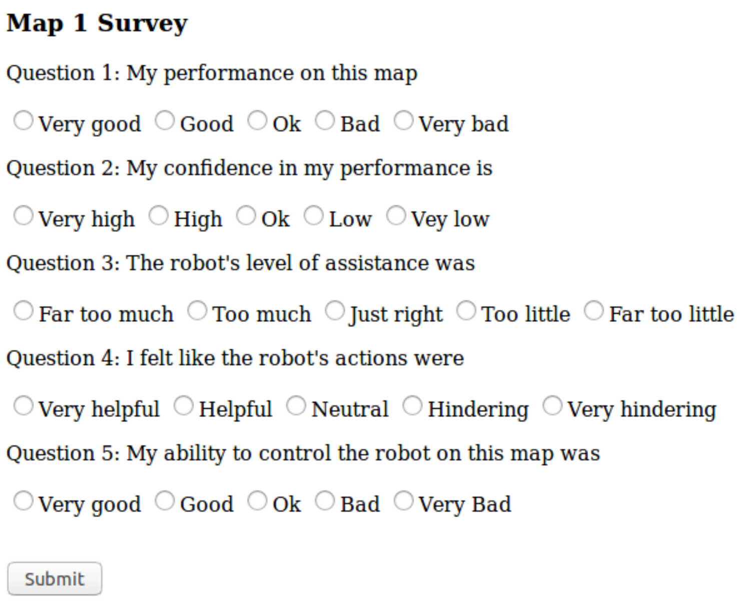
Table 5.1: The parameters set for each of the controllers.

$Controller$	$Parameters$
MostHelp	$\alpha_1 = 2$, $\alpha_2 = 0.75$, $\alpha_3 = 1$, $\alpha_4 = 2$, $\beta = 1.0$
MoreHelp	$\alpha_1 = 1.5$, $\alpha_2 = 0.50$, $\alpha_3 = 1$, $\alpha_4 = 2$, $\beta = 1.0$
SomeHelp	$\alpha_1 = 0.5$, $\alpha_2 = 0.25$, $\alpha_3 = 2$, $\alpha_4 = 0.5$, $\beta = 0.5$
LeastHelp	$\alpha_1 = 0$, $\alpha_2 = 0.0$, $\alpha_3 = 0$, $\alpha_4 = 0$, $\beta = 0.0$

5.2 Procedure

28 able-bodied users (17 male, 11 female, mean age 24.5) were recruited to participate in the user study through emails and fliers distributed through Oregon State University. The participants were compensated \$5 USD for the 30-minute experiment. The users were given an overview of the experiment. Participants were told how and when the assistive controllers would provide assistance, that the "Adaptive" condition would choose a controller based on their performance level and a description of the actions that were considered Beginner actions. The users were given two training maps, the first with "LeastHelp" and the second with "MostHelp", in order to see how to control the robot with the PS3 controller and how the robot would provide assistance.

After the training maps, the users drove through a series of 10 maps with three conditions, for a total of 30 maps. The 10 maps were given in the same order for every condition and for every user. The 10 maps were given in the order of the increasing number of obstacle blocks in the map. To successfully complete a map, a user would drive to the three areas of the areas marked "Goal" on the map. Once in the area, "Goal" would turn yellow and the user would need to stay in the area for one second. After 1 second, the word "Goal" would disappear from that area, and the user would repeat for the remaining two goal areas on the map. Once the user completed a map, the timer would stop and a questionnaire would open for the user to complete. The questionnaire, shown in Figure 5.2, asked 5 questions about the map they just completed. After finishing the questionnaire, the user went on to repeat this process for the remaining maps in the condition. At the end of the 10 maps in the condition, the user answers a set of questions (5.3) about the entire set of 10 maps for that condition. The user repeated the entire process for the other two conditions. At the end of all three conditions, the participants were given a questionnaire (Figure 5.4) asking them to rank the three conditions based on four statements, choosing the least and most helpful assistive actions from the robot, and answering questions on experience with robots, video games, and the PS3 controller.



Map 1 Survey

Question 1: My performance on this map

○ Very good ○ Good ○ Ok ○ Bad ○ Very bad

Question 2: My confidence in my performance is

○ Very high ○ High ○ Ok ○ Low ○ Vey low

Question 3: The robot's level of assistance was

○ Far too much ○ Too much ○ Just right ○ Too little ○ Far too little

Question 4: I felt like the robot's actions were

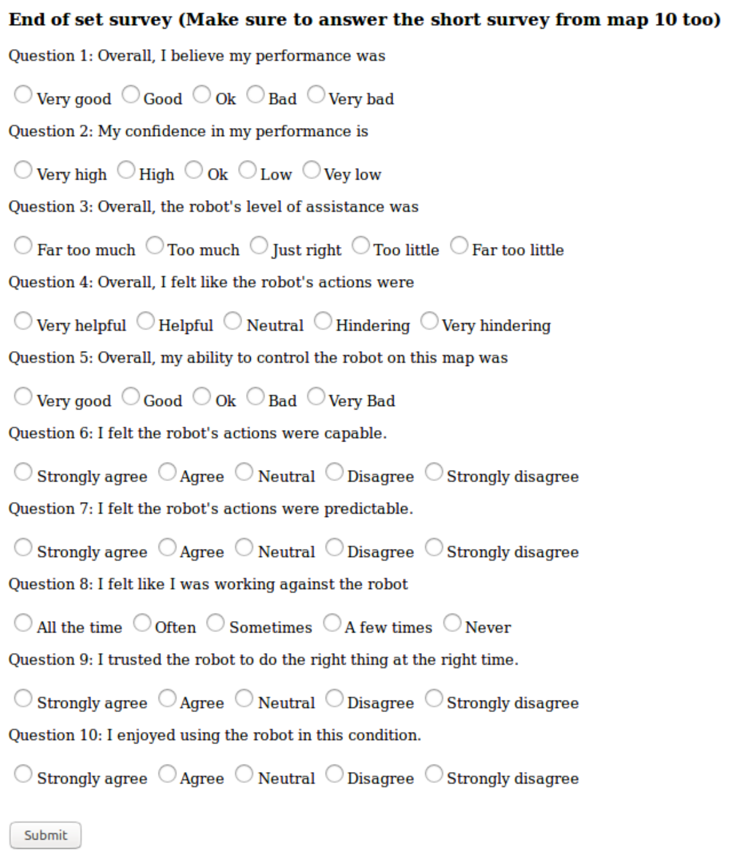
○ Very helpful ○ Helpful ○ Neutral ○ Hindering ○ Very hindering

Question 5: My ability to control the robot on this map was

○ Very good ○ Good ○ Ok ○ Bad ○ Very Bad

Figure 5.2: The questionnaire taken after every map in the second user study.

The participants were given the robot driving simulator described in Chapter 5.2. The participants are given the view as shown in Figure 5.1. At the top of the screen, the user is shown which controller they are currently using and their own time. The same maps were presented in the same order for every user. The 3 conditions were “LeastHelp”, using only the “LeastHelp” controller, MostHelp, using only the “MostHelp”



End of set survey (Make sure to answer the short survey from map 10 too)

Question 1: Overall, I believe my performance was

○ Very good ○ Good ○ Ok ○ Bad ○ Very bad

Question 2: My confidence in my performance is

○ Very high ○ High ○ Ok ○ Low ○ Vey low

Question 3: Overall, the robot's level of assistance was

○ Far too much ○ Too much ○ Just right ○ Too little ○ Far too little

Question 4: Overall, I felt like the robot's actions were

○ Very helpful ○ Helpful ○ Neutral ○ Hindering ○ Very hindering

Question 5: Overall, my ability to control the robot on this map was

○ Very good ○ Good ○ Ok ○ Bad ○ Very Bad

Question 6: I felt the robot's actions were capable.

○ Strongly agree ○ Agree ○ Neutral ○ Disagree ○ Strongly disagree

Question 7: I felt the robot's actions were predictable.

○ Strongly agree ○ Agree ○ Neutral ○ Disagree ○ Strongly disagree

Question 8: I felt like I was working against the robot

○ All the time ○ Often ○ Sometimes ○ A few times ○ Never

Question 9: I trusted the robot to do the right thing at the right time.

○ Strongly agree ○ Agree ○ Neutral ○ Disagree ○ Strongly disagree

Question 10: I enjoyed using the robot in this condition.

○ Strongly agree ○ Agree ○ Neutral ○ Disagree ○ Strongly disagree

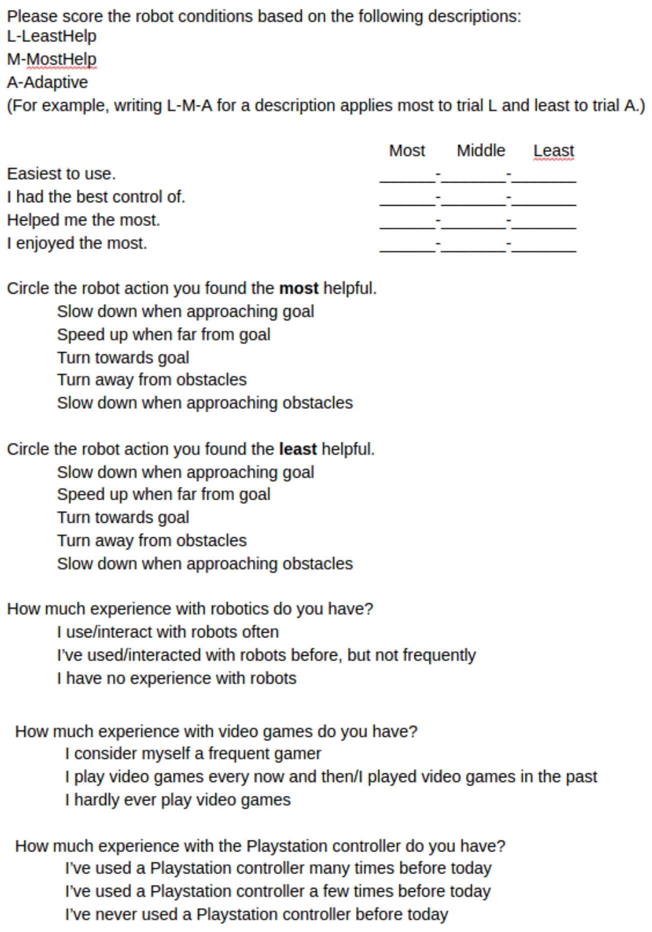
Submit

Figure 5.3: The questionnaire taken after the 10 maps in the second user study.

controller, and “Adaptive”, using the POMDP model to choose the controller based on the belief of expertise. The 3 conditions were counterbalanced in the study to account for order.

5.3 Results

We grouped our subjects into those considered Beginners and those considered Experts as shown in Figure 5.5. Those classified as Beginners had a mean probability of being



Please score the robot conditions based on the following descriptions:
L-LeastHelp
M-MostHelp
A-Adaptive
(For example, writing L-M-A for a description applies most to trial L and least to trial A.)

	Most	Middle	Least
Easiest to use.	______-	_______-	_______
I had the best control of.	______-	_______-	_______
Helped me the most.	______-	_______-	_______
I enjoyed the most.	______-	_______-	_______

Circle the robot action you found the **most** helpful.
- Slow down when approaching goal
- Speed up when far from goal
- Turn towards goal
- Turn away from obstacles
- Slow down when approaching obstacles

Circle the robot action you found the **least** helpful.
- Slow down when approaching goal
- Speed up when far from goal
- Turn towards goal
- Turn away from obstacles
- Slow down when approaching obstacles

How much experience with robotics do you have?
- I use/interact with robots often
- I've used/interacted with robots before, but not frequently
- I have no experience with robots

How much experience with video games do you have?
- I consider myself a frequent gamer
- I play video games every now and then/I played video games in the past
- I hardly ever play video games

How much experience with the Playstation controller do you have?
- I've used a Playstation controller many times before today
- I've used a Playstation controller a few times before today
- I've never used a Playstation controller before today

Figure 5.4: The questionnaire taken at the end of the second user study.

at the Expert state of less than 0.85 (8 participants) and those at or above 0.85 were classified as Experts (20). Figure 5.6 also shows the combined scores of the user's reported experience in robotics, video games, and with the Playstation 3 controller (the lower the score the higher the experience levels). From this data, we do see that 5 of the users classified as Beginners did report lower experience levels, but the other 3 users actually reported high experience levels. For our experiment, what constitutes a

Beginner or Expert is largely dependent on how the POMDP model was designed. The users who had high experience, but still had lower Expert probabilities, were possibly not as proficient as others at this particular task and for these metrics.

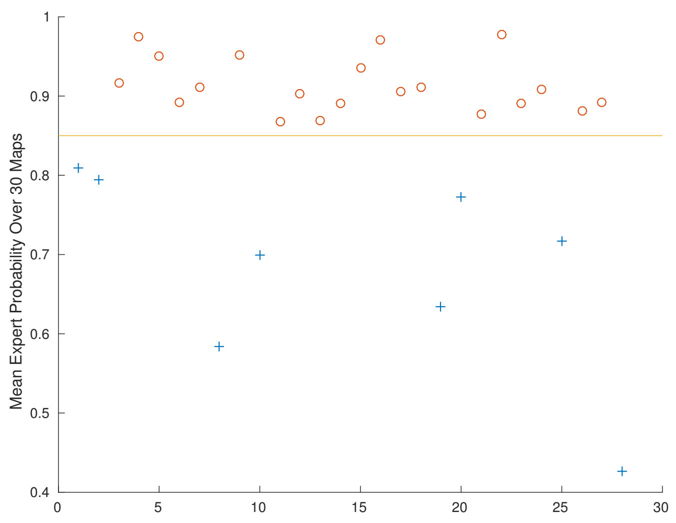


Figure 5.5: The mean probability of being an Expert for the participants over all 30 maps. We classified Experts as being at or above 0.85 and Beginners under it.

For the statistical analysis of the results of this study, we performed non-parametric tests on the data. We used non-parametric tests in this study because non-parametric tests do not rely on the assumption that the data has a normal distribution [18]. While parametric statistics compare the means of data sets, non-parametric tests compare the medians and are better suited to relatively small amounts of data. We use a p-value<0.05 as a threshold for statistical significance, which is shown on Figures with one star. Two stars represents a p-value<0.01 and three stars represents a p-value<0.001.

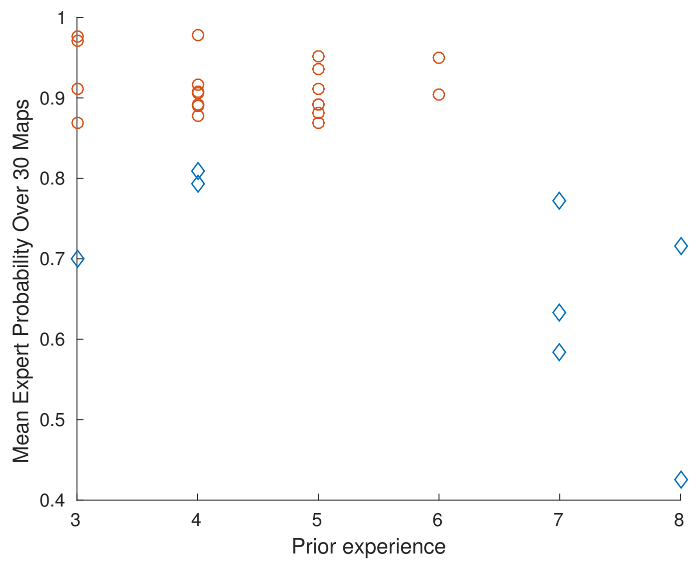


Figure 5.6: The users' combined ratings for their experience in robotics, video games, and the Playstation 3 controller versus their mean probability of being an Expert. Circles are users classified as Experts and diamonds are users classified as Beginners.

5.3.1 Performance Measures

We first hypothesize that we will see some learning effects the longer the participants use the system. We pose three hypothesis about how performance will improve over time.

Hypothesis 3a *Drivers will complete maps more quickly the longer they drive the robot.*

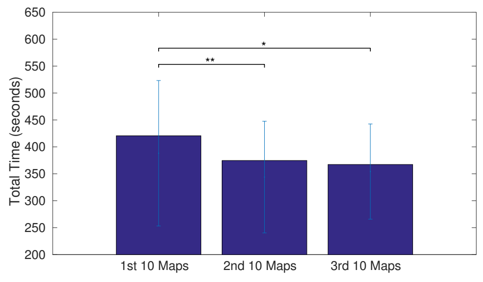
Hypothesis 3b *Drivers will hit fewer obstacles for a given amount of time the longer they drive the robot.*

Hypothesis 3c *The probability of a driver being in the Expert state will increase the longer they drive the robot.*

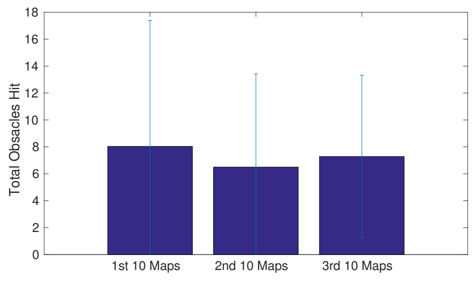
Figure 5.7 shows the differences in total time to complete, total number of hits, and the mean belief at state Expert for each set of ten maps in the order they were completed. Figure 5.7a shows that the amount of time to complete the map decreases after the first set of maps. There is a statistically significant decrease between the first 10 maps $(MDN = 388.20)$ and the second 10 maps $(MDN = 343.97)$; $p = 0.006$. There is also a statistically significant difference between the first 10 maps and the last 10 maps $(MDN = 354.17)$, $p = 0.015$. These results suggest that users tended to perform faster after seeing the maps once. Figure 5.7c shows that the belief the user is an Expert increases over time. There is a statistically significant increase in Expert belief between the first 10 maps $(MDN = 0.822)$ and the second set of 10 maps $(MDN = 0.909)$; $p = 0.004$, as well as in the third set of $(MDN = 0.926)$ set of 10 maps; $p = 0.023$. As seen in Figure 5.8b, there is not much difference in the number of obstacles hit between all three sets. These results support Hypothesis 3a and Hypothesis 3c. After seeing all the maps one time, participants finished the map quicker and the POMDP had a higher belief they were in the Expert state. However, the number of obstacles hit does not improve even after seeing the maps once.

We also compare these variables in each of the conditions, as shown in Figure 5.9. There is not a noticeable difference between the conditions for time (Figure 5.8a) and Expert probability (5.8c), but there is a decrease in the number of obstacles hit in between the LeastHelp condition $(MDN = 6.50)$ the MostHelp condition $(MDN = 3.50)$;$p = 0.015$. This suggests that the controller impacts the number of obstacles hit more when the participant has driven through 30 maps.

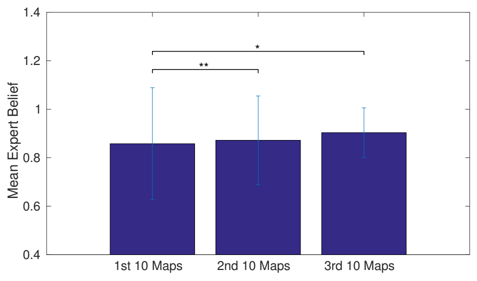
Figure 5.9 shows the total number of obstacles hit and the total time to complete the



(a) Total time to complete 10 maps.



(b) Total obstacles hit in 10 maps.

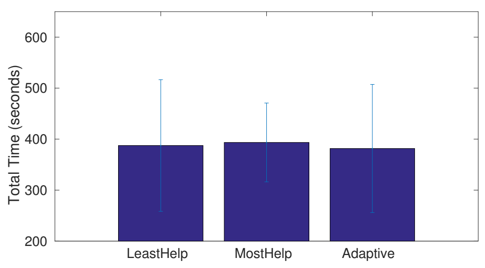


(c) Average belief for state Expert.

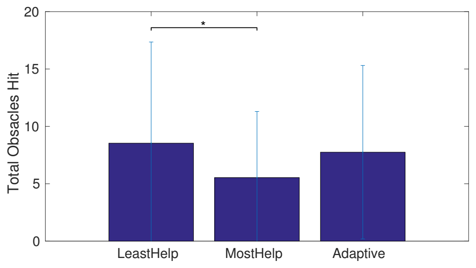
Figure 5.7: The differences in time, obstacles hit, and the belief percent of the Expert state after each set of 10 maps.

10 maps in each condition for the Beginner and Expert users. We compare the number of obstacles hit and the total time using a Wilcoxon sign rank test. There was no statistically significant difference between any of the conditions for Beginners, but for Experts, the number of obstacles hit between "LeastHelp" (MDN=7.50) and "MostHelp" (MDN=2.00) is statistically significant $(p = 0.023)$.

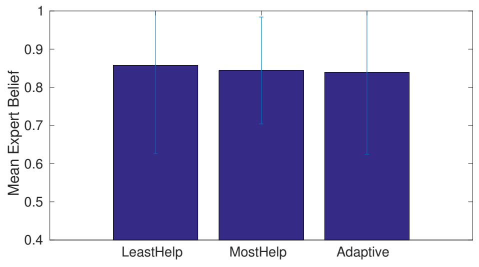
We again tested Hypothesis 2a from Chapter 3, *the difference in the number of obstacles hit between using the "MostHelp" and "LeastHelp" controllers will be greater for users classified as beginners than users classified as experts*. Figure 5.10 shows the difference between the number of obstacles hit between the Beginners and Experts. A



(a) Total time to complete 10 maps.



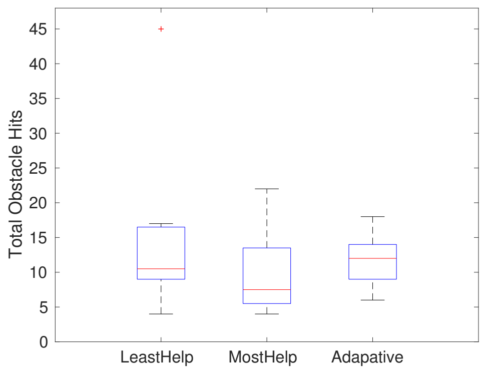
(b) Total obstacles hit in 10 maps.



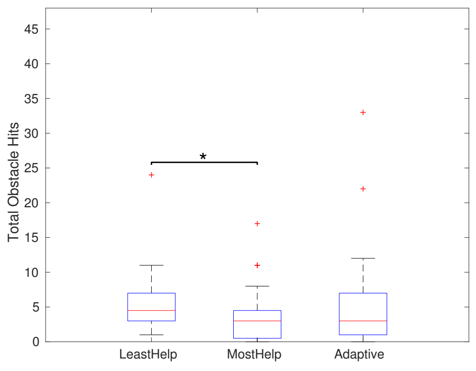
(c) Average belief for state Expert over 10 maps.

Figure 5.8: The differences of time, obstacles hit, and the belief percent of the Expert state over each of the conditions.

Wilcoxon rank sum test shows there was no statistically significant difference between the difference when comparing over all ten maps (Figure 5.10a). We also separated the groups by the maps. The easy maps are the first five maps each with less than or equal to 20 obstacles. The hard maps are the last five maps, each with over 20 obstacles. There is only a statistically significant difference between Beginners (MDN = 3.50) and Experts (MDN = 2.50) for the total number of obstacles hit for the easy maps (Figure 5.10b) . For both Beginners and Experts, the “MostHelp” controller helped some users avoid obstacles in the hard maps, but the negative values indicate that it sometimes caused the users to hit more obstacles with the robot’s assistance (Figure 5.10c). Though the



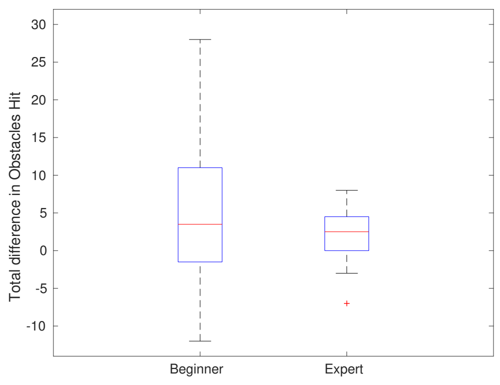
(a) Beginners



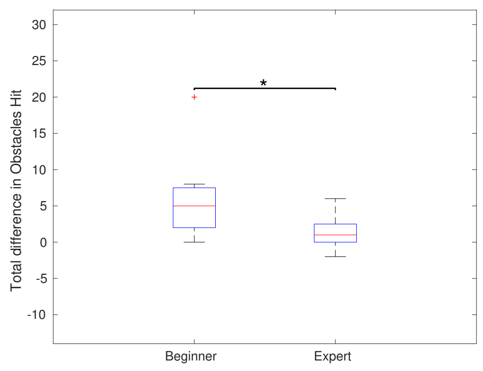
(b) Experts

Figure 5.9: The total number of obstacles hit between the three conditions for the group of Beginners and Experts.

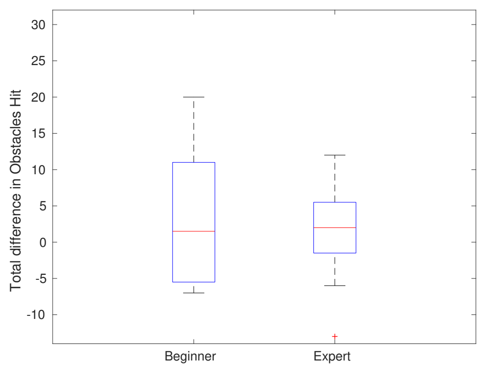
hypothesis is supported by the results of the easy maps, the results of the hard maps suggest that the robot's assistance may actually hinder users in very cluttered environments, despite their level of expertise.



(a) All maps



(b) Easy maps (1-5), Obstacles ≤ 20



(c) Hard maps (6-10), Obstacles > 20

Figure 5.10: The difference in the total number of obstacle hit between the "MostHelp" condition and "LeastHelp" condition for Beginner and Expert users.

5.3.2 Questionnaire Measures

We developed a set of hypotheses about the questionnaire results we would see in beginners and experts. We developed the following hypothesis for the participants classified as Beginners.

Hypothesis 4a *Beginners will feel they have a higher performance when in the Mos-*

tHelp and "Adaptive" condition than the "LeastHelp" condition.

Hypothesis 4b *Beginners will feel more confident in their performance in the MostHelp and "Adaptive" conditions than the "LeastHelp" condition.*

Hypothesis 4c *Beginners will feel the "LeastHelp" condition provides too little assistance compared to "MostHelp" and "Adaptive".*

Hypothesis 4d *Beginners will feel the "MostHelp" and "Adaptive" condition more helpful than hindering compared to "LeastHelp".*

Hypothesis 4e *Beginners will feel they have a better amount of control when in the MostHelp and "Adaptive" condition than the "LeastHelp" condition.*

We also developed the following hypothesis for the participants classified as Experts.

Hypothesis 5a *Experts will feel they have a higher performance when in the "LeastHelp" and "Adaptive" condition than the "MostHelp" condition.*

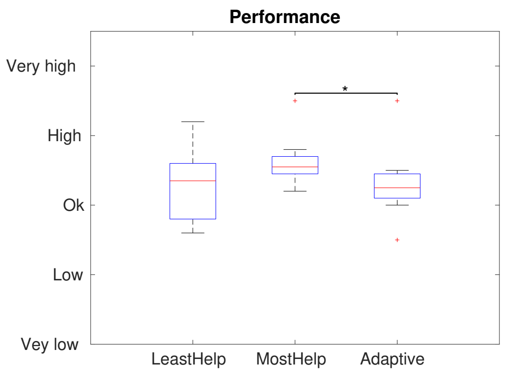
Hypothesis 5b *Experts will feel more confident in their performance in the "LeastHelp" and "Adaptive" conditions than the "MostHelp" condition.*

Hypothesis 5c *Experts will feel the "MostHelp" condition provides too much assistance compared to "LeastHelp" and "Adaptive".*

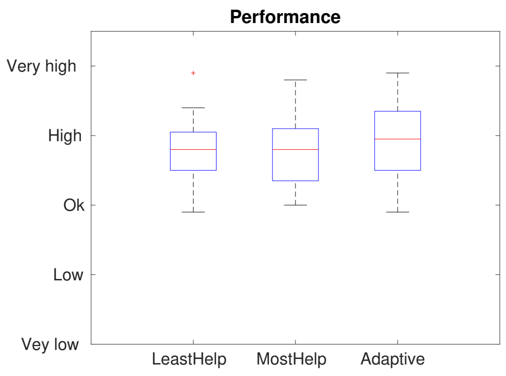
Hypothesis 5d *Experts will feel the "MostHelp" condition more hindering than helpful compared to "LeastHelp" and "Adaptive".*

Hypothesis 5e *Experts will feel they have a better amount of control when in the "LeastHelp" and "Adaptive" condition than the "MostHelp" condition.*

A Wilcoxon signed rank test was performed over the questionnaire answers from the questions answered after each map. The responses to the statements given in the questionnaire given after every map are shown in Figures 5.11-5.15.

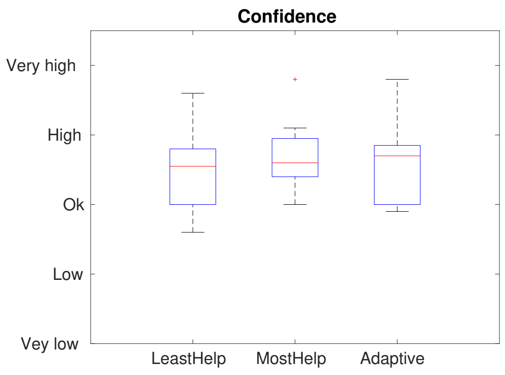


(a) Beginners

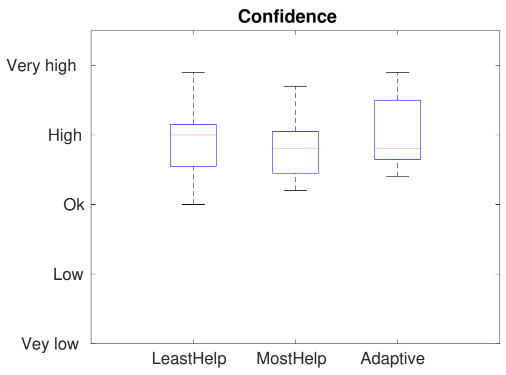


(b) Experts

Figure 5.11: Scores of Q1, My performance on this map was

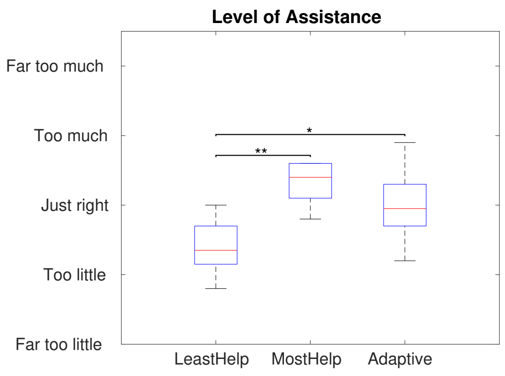


(a) Beginners

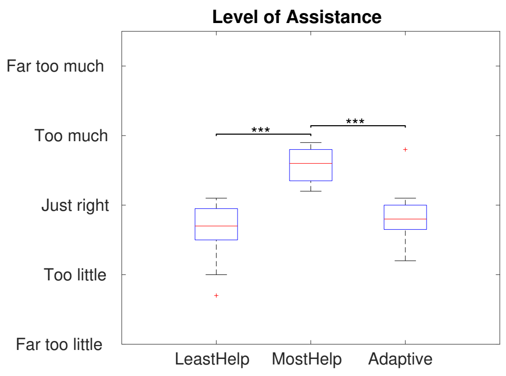


(b) Experts

Figure 5.12: Scores of Q2, My confidence in my performance is

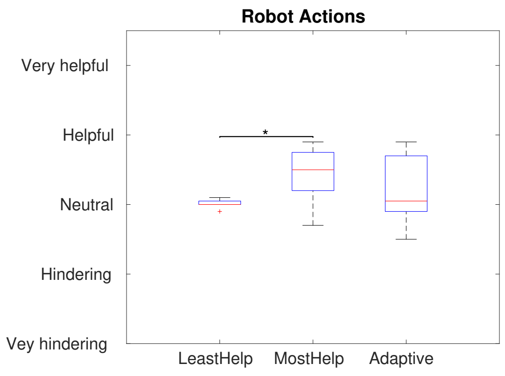


(a) Beginners

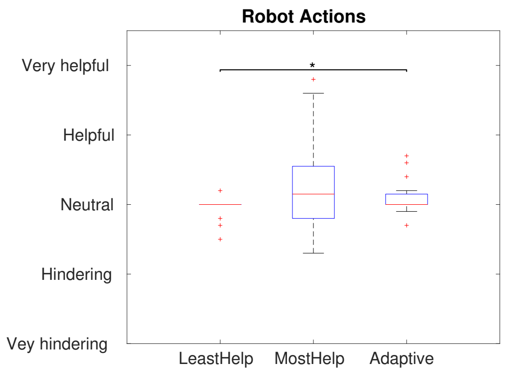


(b) Experts

Figure 5.13: Scores of Q3, The robot's level of assistance was

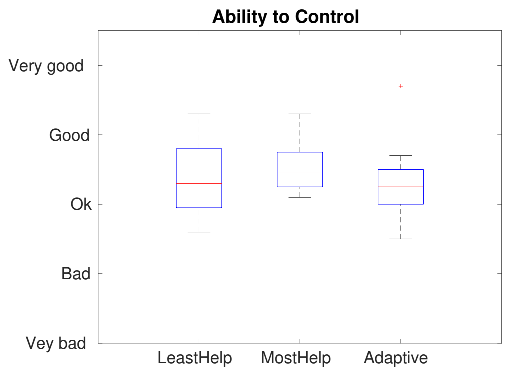


(a) Beginners

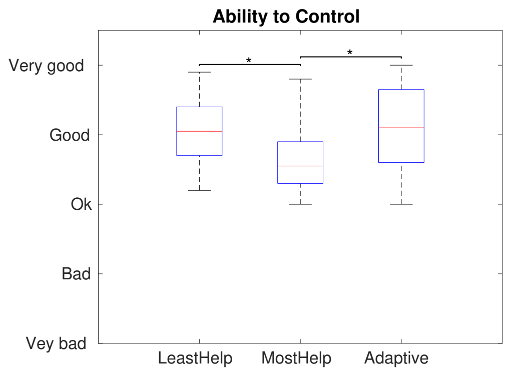


(b) Experts

Figure 5.14: Scores of Q4, I felt like the robot's actions were



(a) Beginners



(b) Experts

Figure 5.15: Scores of Q5, My ability to control the robot on this map was

When reporting performance (Q1), Beginners ranked the MostHelp condition the highest, and there is a statistically significant difference between “MostHelp” and “Adaptive” ($p = 0.016$). When reporting how they thought about the robot’s level of assistance (Q3), there is statistical significance between “LeastHelp” and “MostHelp” ($p < 0.01$) and “LeastHelp” and “Adaptive” ($p = 0.016$). This supports Hypothesis 4c. When reporting the robot’s actions (Q4), there is statistical significance between “LeastHelp” and “MostHelp” ($p = 0.031$). This seems to partially support Hypothesis 4d, but the ranking for “MostHelp” is only slightly above Neutral. The results did support Hypothesis 4b and Hypothesis 4e. Table 5.2 shows if the hypothesis were supported, partially supported, or not supported.

When reporting performance (Q1), Experts had no statistically significant differences in the conditions. This does not support Hypothesis 5a. This may instead imply that the controller used does not affect they feel about their performance. When reporting how their confidence in their performance (Q2), there was again no statistical significance. When reporting how they thought about the robot's level of assistance (Q3), there is statistical significance between "LeastHelp" and "MostHelp" ($p < .0001$) and "MostHelp" and "Adaptive" ($p < 0.0001$). This supports Hypothesis 5c. When reporting the robot's actions (Q4), there is statistical significance between "LeastHelp" and "Adaptive" ($p = 0.024$). This suggests that it is possible "Adaptive" gave the users the correct amount of assistance they needed over time, but "Adaptive" is ranked only slightly above Neutral. Expert users are likely to find the assistance to be more than they need. When reporting their feeling of level of control they had (Q5) there is statistical significance between "LeastHelp" and "MostHelp" ($p = 0.031$) and "MostHelp" and "Adaptive" ($p = 0.021$). This supports Hypothesis 5e.

Table 5.3 shows if the hypothesis were supported, partially supported, or not supported. These results show Beginners did tend to find the assistance from the "MostHelp" controller and the "Adaptive" controller more helpful, though their feeling of performance, confidence, and ability to control were not significantly different. Experts found the level of assistance provided by the "MostHelp" to often be more than desired and had a worse feeling of control when using the "MostHelp" controller.

Table 5.2: Comparison between controllers LeastHelp (L), MostHelp (M), and Adaptive (A). The p-values that support the developed hypothesis for Beginners. The columns show the hypothesis that are either partially supported, supported, or not supported and the p-values.

	H4a (Performance)	H4b (Confidence)	H4c (Assistance)
	Not Supported	Not Supported	Supported
L-M	ns	ns	.008
L-A	ns	ns	.016
M-A	.016	ns	ns

	H4d (Actions)	H4e (Control)
	Partially	Not Supported
L-M	.031	ns
L-A	ns	ns
M-A	ns	ns

Table 5.3: Comparison between controllers LeastHelp (L), MostHelp (M), and Adaptive (A). The p-values that support the developed hypothesis for Experts. The columns show the hypothesis that are either partially supported, supported, or not supported and the p-values.

	H5a (Performance)	H5b (Confidence)	H5c (Assistance)
	Not Supported	Not Supported	Supported
L-M	ns	ns	$<.0001$
L-A	ns	ns	ns
M-A	ns	ns	$<.001$

	H5d (Actions)	H5e (Control)
	Partially	Supported
L-M	ns	.031
L-A	.024	ns
M-A	ns	.021

Table 5.4: List of questions from the questionnaires.

		1	2	3	4	5
Q1	I believe my performance was	Very low	Low	Ok	High	Very good
Q2	My confindence in my performance is	Very low	Low	Ok	High	Very high
Q3	The robot's level of assistance was	Far too little	Too little	Just right	Too much	Far too much
Q4	I felt like the robot's actions were	Very hindering	Hindering	Neutral	Helpful	Very Helpful
Q5	My ability to control the robot on this map was	Very bad	Bad	Ok	Good	Very Good
Q6	I felt the robot's actions were capable.	Strong disagree	Disagree	Neutral	Agree	Strongly Agree
Q7	I felt the robot's actions were predictable.	Strong disagree	Disagree	Neutral	Agree	Strongly Agree
Q8	I felt like I was working against the robot	Strong disagree	Disagree	Neutral	Agree	Strongly Agree
Q9	I trusted the robot to do the right thing at the right time.	Strong disagree	Disagree	Neutral	Agree	Strongly Agree
Q10	I enjoyed using the robot in this condition.	Strong disagree	Disagree	Neutral	Agree	Strongly Agree

We also compare the answers between Beginners and Experts with a Wilcoxon rank sum test. The comparisons for the questionnaires taken after each map are shown in Table 5.5 for "LeastHelp", Table 5.6 for "MostHelp", and 5.7 for "Adaptive". When comparing Beginners and Experts, Experts did tend to give more positive ratings than Beginners when they used the "LeastHelp". Performance (Q1), Confidence (Q2), and Feeling of control (Q5) were all statistically significantly higher for Experts than Beginners.The results were not significantly different when using the "MostHelp" controller because Experts tended to rate this condition with less positive values and Beginners rating it with more positive values. In the Adaptive condition, Experts again had statistically significant differences from Beginners in rating their performance (Q1) and their feeling of control (Q5). This suggests that the controller preferred by user was more often chosen in Adaptive for Expert users than Beginner users. Another explanation may be that Beginners would switch controllers through the Adaptive condition, while Experts tended to only be assigned the "LeastHelp" controller. This could be an indication that the POMDP model we created would try to transition controllers from "MostHelp" to another controller too quickly for Beginners.

The questionnaire results after each condition are shown in Figure 5.16 for Begin-

Table 5.5: Scores in “LeastHelp” condition from the short questionnaire after each map.

		Q1	Q2	Q3	Q4	Q5
Median	B	3.500	3.550	2.350	3.000	3.300
	E	3.800	4.000	2.700	3.000	4.050
p		0.027*	0.044*	0.175	0.300	0.013*

Table 5.6: Scores in “MostHelp” condition from the short questionnaire after each map.

		Q1	Q2	Q3	Q4	Q5
Median	B	3.550	3.600	3.400	3.500	3.450
	E	3.770	3.800	3.600	3.150	3.550
p		0.444	0.473	0.062	0.212	0.540

Table 5.7: Scores in “Adaptive” condition from the short questionnaire after each map.

		Q1	Q2	Q3	Q4	Q5
Median	B	3.250	3.700	2.950	3.050	3.250
	E	3.950	3.800	2.800	3.000	4.100
p		0.007*	0.183	0.342	0.895	0.011*

ners and Figure 5.17 for Experts. The comparisons for the longer questionnaires taken after each condition are shown in Table 5.8 for “LeastHelp”, Table 5.9 for “MostHelp”, and 5.10 for “Adaptive”. From the significant values from these results, we see that Experts found the LeastHelp and Adaptive conditions more predictable (Q7). We also see that Experts found they were working against the robot more in the MostHelp condition, but Beginners working against it more in the Adaptive (Q8). Experts also had more positive ratings for performance (Q1) and control over the robot (Q5) during the Adaptive condition. These values are a result of Experts often receiving only the LeastHelp controller during the Adaptive condition, while Beginners would be assigned the other controllers more often.

Table 5.8: Questionnaire scores after completing “LeastHelp” condition.

		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
Median	B	3.625	3.500	2.375	3	3.625	2.625	3.125	1.125	2.625	3.375
	E	3.950	4.105	2.650	2.950	4.100	3.211	4.368	1.2	2.6	4
p		0.417	0.130	0.202	0.740	0.125	0.0678	.002*	0.867	0.936	0.077

Table 5.9: Questionnaire scores after completing “MostHelp” condition.

		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
Median	B	3.750	3.625	3.375	3.500	3.625	3.875	3.625	2.750	3.125	3.375
	E	3.800	3.900	3.750	3.000	3.750	3.400	3.700	3.300	3.000	2.900
p		0.840	0.310	0.071	0.208	0.843	0.310	0.623	0.044*	0.725	0.245

Table 5.10: Questionnaire scores after completing “Adaptive” condition.

		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
Median	B	3.000	3.625	3.125	3.250	3.000	3.500	2.875	2.625	3.250	3.125
	E	4.000	4.000	2.750	3.200	4.050	3.250	3.800	1.650	3.050	3.800
p		0.002*	0.216	0.150	0.786	0.002*	0.443	0.033*	0.018*	0.419	0.170

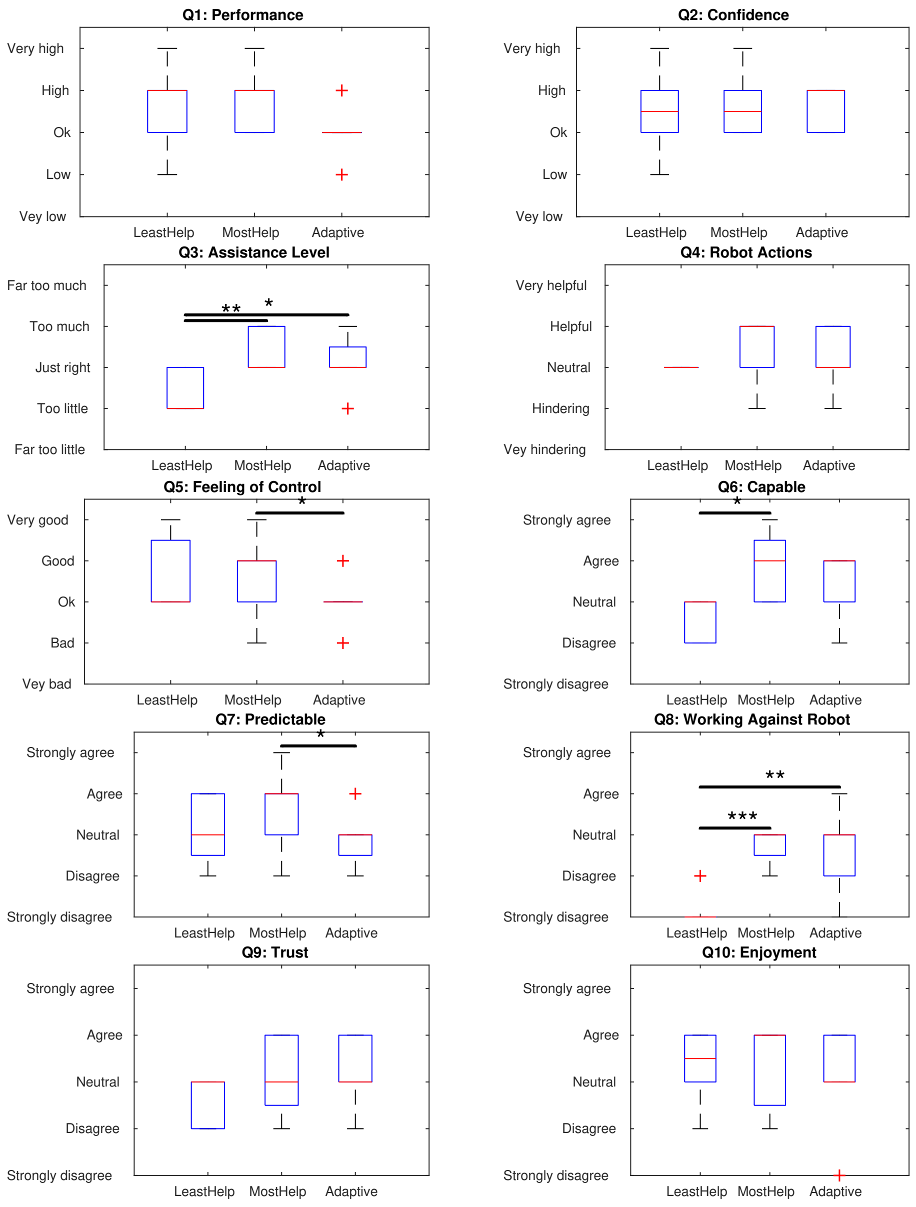


Figure 5.16: The questionnaire answers after each condition from the Beginner users.

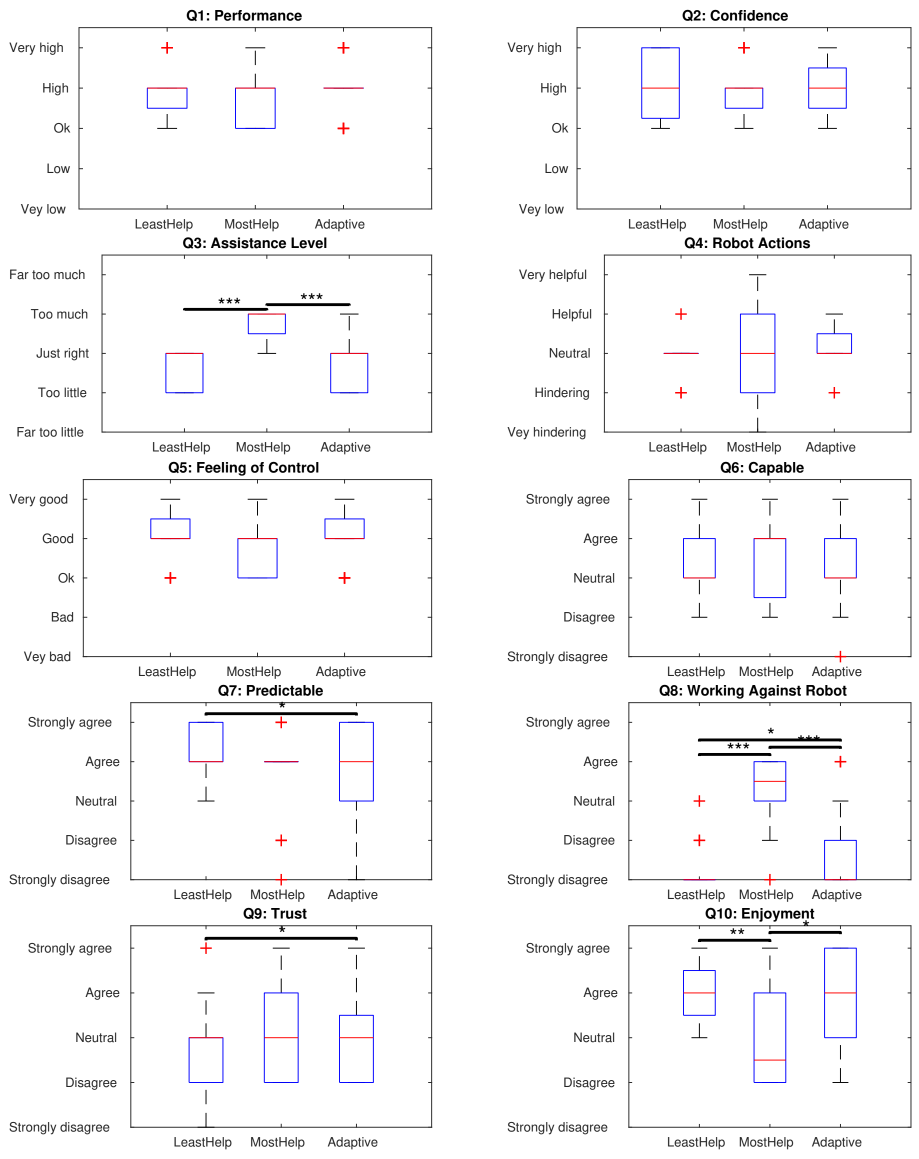
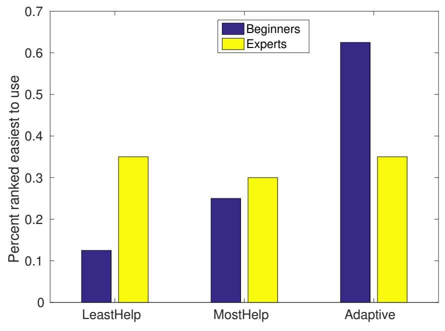
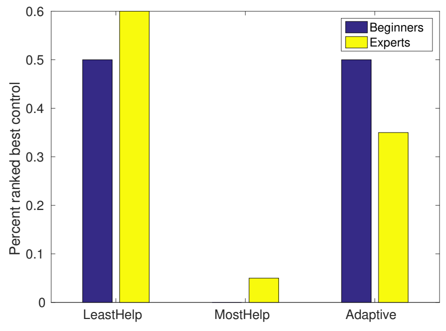


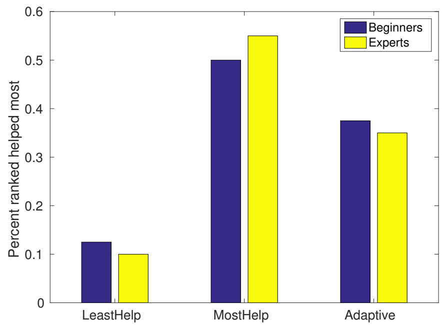
Figure 5.17: The questionnaire answers after each condition from the Expert users.



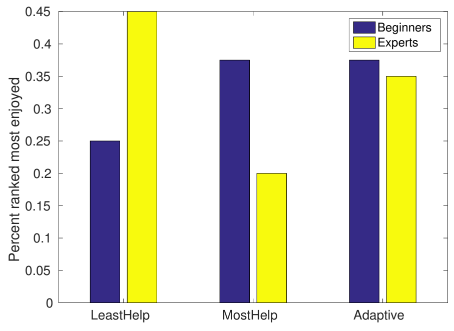
(a) Easiest to Use



(b) Had Best Control



(c) Helped Most



(d) Most Enjoy

Figure 5.18: Percentage of user rankings of the 3 conditions.

We had the participants rank the three conditions as shown in Figure 5.4. The data from the conditions that they ranked the highest are shown in Figure 5.18. The rankings for the conditions they had the best control in (Figure 5.18b) and which helped the most (Figure 5.18c) are fairly similar between Beginners and Experts. When ranking which was easiest to use (Figure 5.18a), they ranked all conditions similarly, but fewer Beginners ranked the “LeastHelp” condition easiest to use. In Figure 5.18d, most Experts either most enjoyed “LeastHelp” or “Adaptive”, but most Beginners found “MostHelp” the most enjoyable.

5.4 Discussion

With our second user study, we added assistive speed control to our domain to test the user expertise algorithm and confirmed what we found in the first study. We found again that more robot assistance helped the users avoid obstacles, though it sometimes increased the amount of time to complete the map. We also looked at how the users responded to the assistance. The results suggest that Beginner users felt that the assistance from the “MostHelp” controller provided a good amount of assistance. Beginners were also less likely to rank “LeastHelp” as the easiest controller to use and were more likely to rank “MostHelp” as the controller they most enjoyed.

The results from this study show that while the expert users ranked the “Adaptive” controller similarly to the “LeastHelp” as expected, but this was not the case with beginners. This may suggest that our POMDP model would too quickly assign a “LeastHelp” controller to users who actually still needed assistance. This means the belief state would transition to a higher belief of being at the Expert state too quickly. Future work can look at ways to use user data, such as the data from this study, in order to learn the transition functions for the user expertise POMDP model. This would be more accurate than hand-tuning the transitions values.

In our study, we saw there some expert users who enjoyed having more of the autonomous assistance and some beginners who preferred less assistance. Even though as the results showed, this is a less common case, it is important to keep in mind these users when designing a system. Preference and performance should be weighed and considered depending on the situation. For the example, we presented where more autonomy prevents collisions but can slow a user down. In the case where hitting obstacles would not cause much harm, it may be better to provide a beginner user with less assistance if you can find that is what they prefer. On the other hand, if you have an objective that is time critical, if the autonomous assistance would slow them down, it would be best not to provide it even if it is what they would prefer.

6 Adapting to Operator Expertise on an Underwater Vehicle

We performed a hardware demonstration of our POMDP model on the Seabotix vLBV300, a tethered underwater semi-autonomous vehicle shown in Figure 6.1. The vLBV300 platform has 6DOF. A user drives the vehicle by controlling depth, heading, and driving through the horizontal plane. Our demonstration with the shared autonomous system was off the docks in Newport, Oregon. The teleoperation of ROVs requires a high level of training and skill to operate. Missions involving these vehicles are often expensive and time sensitive, and a novice may pose a great risk to the safety of the vehicle. With our user expertise model, a novice user could potentially operate the vehicle more easily and safely if given the appropriate level of robot assistance, therefore increasing the number of people who can operate the vehicle on missions.

The rest of the chapter is organized as follows. First, we present the procedure for testing the user expertise model on the vehicle (Section 6.1). We then present the results of a demonstration off the coast of Newport (Section 6.2). Finally, we discuss the result and possible future work (Section 6.3).

6.1 Procedure

For the Seabotix vehicle demonstration, we set up a test scenario where an operator must drive to a set of predetermined waypoints using a Playstation 3 controller. Before

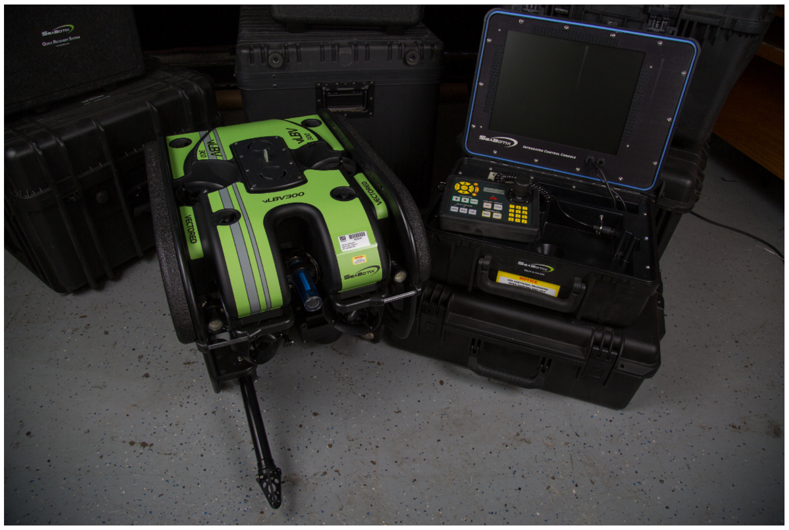


Figure 6.1: The Seabotix vLBV300 used in the field demonstration.

starting, the waypoints are determined, along with a close range, CR, and far range, FR, from a waypoint. The far range is the distance the vehicle is from a waypoint where the user should be driving the vehicle at a high speed and the close range is the distance from the waypoint where the operator drive slower as to not overshoot the waypoint. The area where the vehicle is considered “in” the waypoint must also be set. These conditions can vary depending on the distance between waypoints. We also set boundary areas or “no go zones” where the vehicle should not pass. This area could be a bounding box of the driving area or can mark a spot where there is a known obstacle. Figure 6.2 shows a representation of this scenario.

The shared autonomy for the Seabotix vLBV300 was developed with the use of the Robot Operating System (ROS) [33]. The past work done by Lawrance et al. [24] developed a ROS software package that serves as a communication bridge between the vehicle’s Lightweight Communications and Marshalling (LCM) messages and ROS nodes.

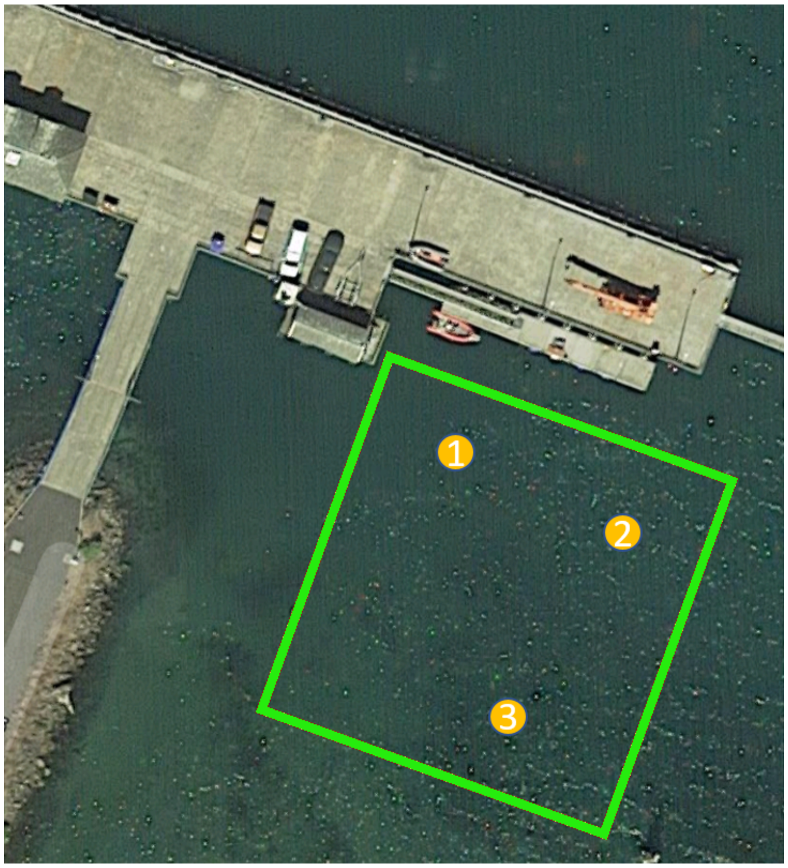


Figure 6.2: An example scenario of the Seabotix vehicle test domain at Newport, Oregon. The yellow circles represent the waypoints and the green box represents the “no go zones”.

To test the user expertise method for choosing levels of shared autonomy on the Seabotix vehicle, we created a ROS node to modify the PS3 joystick commands to the vehicle. The ROS node receives the vehicle’s speed, position, and the joystick commands from the user.

The POMDP model we used for the Seabotix vehicle is shown in Table 6.1. To simplify the scenario, we only use 2 controllers in our action set, with MostHelp providing

Table 6.1: POMDP for Seabotix

S	Beginner, Expert
A	LeastHelp, MostHelp
O	TooFast, TooSlow, NoGoZone, Good

assistance and LeastHelp not providing any robot assistance. The observation set includes “TooFast”, when the operator is driving too fast in the close range, “TooSlow”, when the operator is driving too slow in the far zone, “No go zone”, when the robot is out of bounds, and “Good”, when the operator is driving exactly how they are supposed to.

6.2 Results

The following are the results from the dock demonstration that was performed at Newport, Oregon. The test plan of driving to the 3 waypoints was completed. The testing was performed by the author demonstrating representative actions that would be performed by a Beginner or an Expert. Figure 6.3 shows the recorded data of driving to the 3 waypoints.

Figure 6.4 shows an example of the model working when the human keeps driving full speed when close to the waypoint. At around the 1000 mark, where the distance to the goal is small, the Beginner probability begins to rise, and the MostHelp controller modifies the driving command to be slower than the human’s original control. In Figure 6.5, while the driver is still far the waypoint and is driving slowly, the Beginner proba-

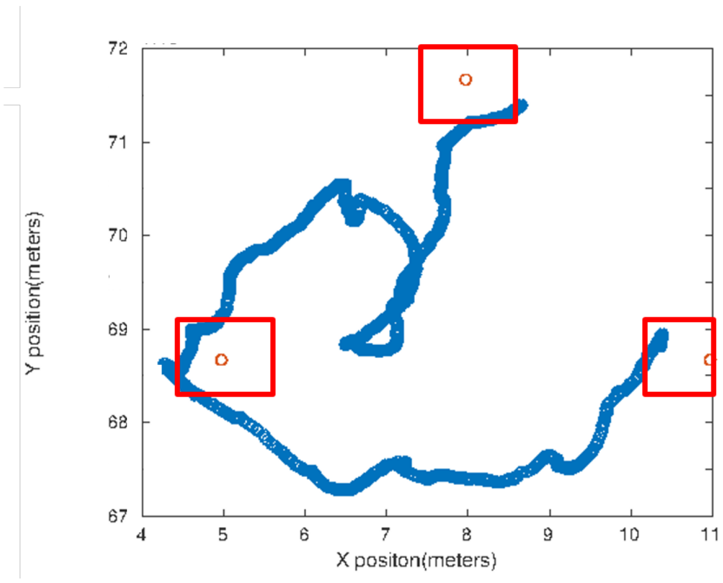


Figure 6.3: The recorded position of the Seabotix vehicle (blue) and the waypoints (red).

bility increases, and the controller increases the speed command so that the vehicle will drive faster towards the waypoint. These results show the POMDP model working on the Seabotix vehicle.

6.3 Discussion

We have implemented our developed expertise algorithm on a Seabotix ROV and developed a test scenario. In our preliminary demonstration, we were able to show that the POMDP model could detect “Beginner” behavior and switch controllers accordingly.

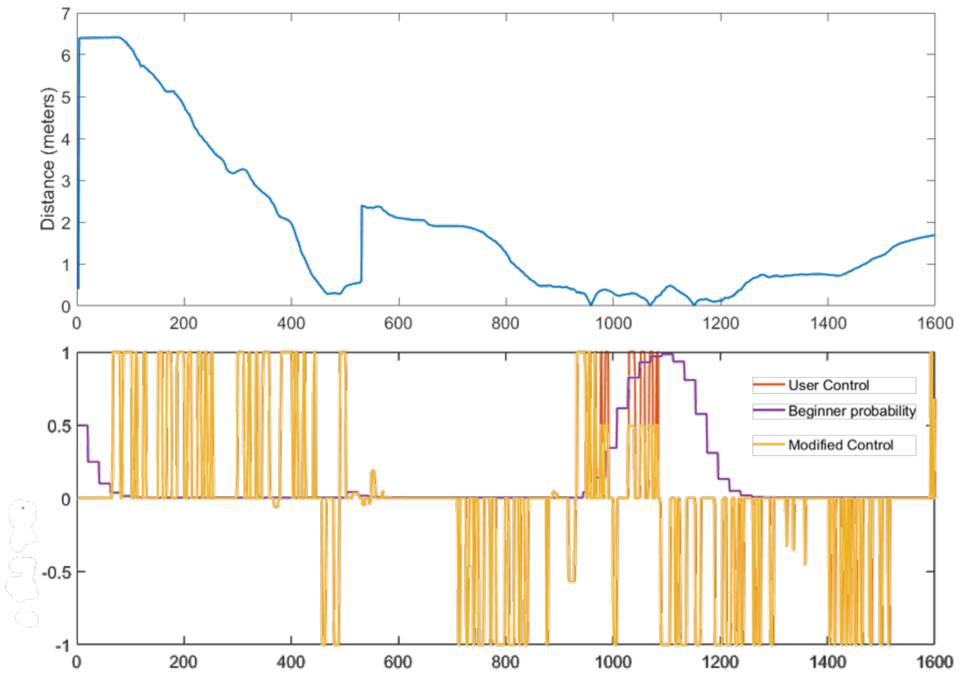


Figure 6.4: As the vehicle approaches the the waypoint around the 1000 mark, the Beginner probability rises, and the MostHelp controller limits the speed command.

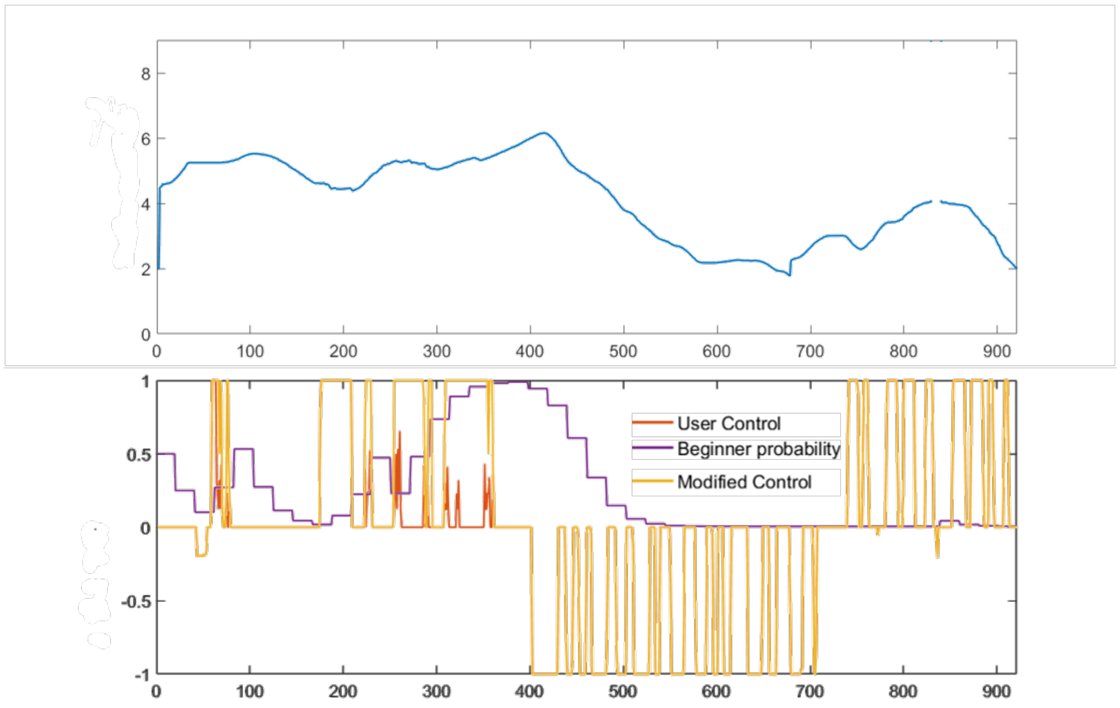


Figure 6.5: When the vehicle is still far from the waypoint and going slowly, the Beginner probability rises and the controller increases the speed command.

7 Conclusions and Future Directions

In this thesis, we presented a novel contribution to the area of shared autonomy by developing a POMDP model for human expertise. Our novel algorithmic accomplishments include:

- A framework that leverages POMDPs to model human expertise by using observations of the user's actions to maintain a belief of where that user lies between the states of Beginner or Expert.
- Macro-action controllers that encompass various levels of shared autonomy. These macro-action controllers are used by the POMDP model to assign the appropriate level of robot autonomy based on that user's expertise.

Along with the development of our POMDP model of human expertise we made the following additional contributions in this thesis:

- We ran a user study with fourteen participants which verified that our POMDP model was able to predict expertise. Through this study, we found that users with a higher probability of being a Beginner received more help when receiving higher levels of robot autonomy than users with a lower probability of being a Beginner.
- Through data captured from the first user study, we used a decision tree to capture the different driving behaviors of Beginners and Experts. We then programed

simulated “bots” to act as those users. By running numerous simulations with these bots, we used the results of simulations to find the controller values that gave the Beginner and the Expert bots the best performance.

- For our second user study, we studied the differences in how the users felt while receiving no robot autonomy, when receiving a high amount of autonomy, and when receiving the amount of autonomy chosen by expertise model. We found that the users classified as Beginner ranked the assistive controller better than the Experts did.
- We performed a field trial with a semi-autonomous underwater vehicle using our user expertise model. We were able to show the model switching to more robot assistance after the driver performed beginner actions.

7.1 Future Work

The user expertise model we developed is a general framework that it could be applied to a number of scenarios and can possibly be altered to capture more than just expertise such as a human’s rationality.

Much of the current state of modeling human actions in spaces shared with autonomous or semi-autonomous robots relies on the assumption that humans will always make rational decisions. Of course humans do not always act rationality. Humans that are distracted, fatigued, or simply inexperienced may choose irrational actions, and in the real-world, robots should know when a human is not behaving rationally so that it may plan accordingly. Rather than using the belief of a human being a beginner or an expert, the model can be modified with a belief of being either behaving completely irrational or rational.

Keeping track of user expertise may also be used in training humans. For example in video games, a player does not usually start off being able to use every tool or perform every move. Providing players with everything in the beginning makes learning how to play the game more difficult. Instead, by completing levels, thus proving mastery of what you have, the player unlocks more features as they go along. The same idea could be applied with our expertise model. Rather than trying to train a person with a robot with all the features all at once, training can start with the robot having control of those features in the beginning, and gradually allowing the human more control as the belief of their expertise increases.

The contributions made in this work provide a groundwork for utilizing knowledge of human expertise to improve the state of the art in shared autonomous human-robot interaction. In order for robots to be used by, worked with and around, and interact with a variety of people, they must be able to model the key aspects of human behavior. The work presented here provides a step forward towards helping robots understand these complexities.

Bibliography

[1] Approximate pomdp planning software. `http://bigbird.comp.nus.edu.sg/pmwiki/farm/appl`, 2014.

[2] C. Amato, G.D. Konidaris, A. Anders, G. Cruz, J. P. How, and L. P. Kaelbling. Policy search for multi-robot coordination under uncertainty. 35(14):1760–1778, 2015.

[3] S.J. Anderson, S.B. Karumanchi, and K. Iagnemma. Constraint-based planning and control for safe, semi-autonomous operation of vehicles. In *Proc. IEEE Intelligent Vehicles Symposium (IV)*, pages 383–388, 2012.

[4] J. Beer, A. Fisk, and W. Rogers. Toward a framework for levels of robot autonomy in human-robot interaction. *Journal of Human-Robot Interaction*, 3(2):74, 2014.

[5] J. Casper and R.R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(3):367–385, 2003.

[6] J. W. Crandall and M. A. Goodrich. Characterizing efficiency of human robot interaction: A case study of shared-control teleoperation. 2:1290–1295, 2002.

[7] E. Demeester, A. Hüntemann, D. Vanhooydonck, G. Vanacker, H. Van Brussel, and M. Nuttin. User-adapted plan recognition and user-adapted shared control: A bayesian approach to semi-autonomous wheelchair driving. *Autonomous Robots*, 24(2):193–211, 2008.

[8] A. Dragan, K. Lee, and S. Srinivasa. Teleoperation with intelligent and customizable interfaces. *Journal of Human-Robot Interaction*, 1(3):33–79, 2013.

[9] A. D. Dragan and S. S. Srinivasa. A policy-blending formalism for shared control. *The International Journal of Robotics Research*, 32(7):790–805, 2013.

[10] J. L. Emken, J. E. Bobrow, and D. J Reinkensmeyer. Robotic movement training as an optimization problem: designing a controller that assists only as needed. In *Proc. International Conference on Rehabilitation Robotics*, pages 307–312, 2005.

[11] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

[12] A. Franchi, C. Secchi, M. Ryll, H. Bulthoff, and P. Giordano. Shared control: Balancing autonomy and human assistance with a group of quadrotor uavs. *IEEE Robotics & Automation Magazine*, 19(3):57–68, 2012.

[13] J. Fu, S. Levine, and P. Abbeel. One-shot learning of manipulation skills with online dynamics adaptation and neural network priors. In *Proc. Intelligent Robots and Systems (IROS)*, pages 4019–4026, 2016.

[14] M. Gao, J. Oberländer, T. Schamm, and J. Zöllner. Contextual task-aware shared autonomy for assistive mobile robot teleoperation. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3311–3318, 2014.

[15] A. H. Ghasemi, M. Johns, B. Garber, P. Boehm, P. Jayakumar, W. Ju, and R. B. Gillespie. Role negotiation in a haptic shared control framework. In *Proc. International Conference on Automotive User Interfaces and Interactive Vehicular Applications Adjunct*, pages 179–184, 2016.

[16] K. Hauser. Recognition, prediction, and planning for assisted teleoperation of freeform tasks. *Autonomous Robots*, 35(4):241–254, 2013.

[17] G. Ho, N. Pavlovic, and R. Arrabito. Human factors issues with operating unmanned underwater vehicles. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 55, pages 429–433, 2011.

[18] T. Hoskin. Parametric and nonparametric: Demystifying the terms. *Mayo Clinic*, 2012.

[19] S. Javdani, J. A. Bagnell, and S. Srinivasa. Minimizing user cost for shared autonomy. In *Proc. ACM/IEEE International Conference on Human-Robot Interaction*, pages 621–622, 2016.

[20] S. Javdani, S. S. Srinivasa, and J. A. Bagnell. Shared autonomy via hindsight optimization. In *Robotics: Science and Systems*, Rome, Italy, 2015.

[21] L. P. Kaelblingand, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

[22] J. Koo, J. Kwac, W. Ju, M. Steinert, L. Leifer, and C. Nass. Why did my car just do that? explaining semi-autonomous driving actions to improve driver understanding, trust, and performance. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 9(4):269–275, 2015.

[23] H. Kurniawati, D. Hsu, and W.S. Lee. Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*. Zurich, Switzerland, 2008.

[24] N.R.J. Lawrance, T. Somers, D. Jones, S. McCammon, and G. A. Hollinger. Ocean deployment and testing of a semi-autonomous underwater vehicle. In *OCEANS*, pages 1–6, 2016.

[25] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, 1967.

[26] D.M. Magerman. Statistical decision-tree models for parsing. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 276–283. Association for Computational Linguistics, 1995.

[27] L. Milliken and G. A. Hollinger. Modeling user expertise for choosing levels of shared autonomy. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Singapore, 2017.

[28] M. E. Moran. The da vinci robot. *Journal of Endourology*, 20(12):986–990, 2006.

[29] S. Nikolaidis, A. Kuznetsov, D. Hsu, and S. Srinivasa. Formalizing human-robot mutual adaptation via a bounded memory based model. In *Proc. ACM/IEEE International Conference on Human-Robot Interaction*, pages 75–82, 2016.

[30] S. Nikolaidis, P. Lasota, R. Ramakrishnan, and J. Shah. Improved human–robot team performance through cross-training, an approach inspired by human team training practices. *The International Journal of Robotics Research*, 34(14):1711–1730, 2015.

[31] A. Pentland and A. Liu. Modeling and prediction of human behavior. *Neural computation*, 11(1):229–242, 1999.

[32] P. Pinheiro, E. Cardozo, and C. Pinheiro. Anticipative shared control for robotic wheelchairs used by people with disabilities. In *Proc. IEEE International Conference on Autonomous Robot Systems and Competitions*, pages 91–96, 2015.

[33] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5, 2009.

[34] P. J. Sanz, P. Ridao, G. Oliver, G. Casalino, Y. Petillot, C. Silvestre, C. Melchiorri, and A. Turetta. Trident an european project targeted to increase the autonomy levels for underwater intervention missions. In *Oceans*, pages 1–10, 2013.

[35] K. Shamaei, Y. Che, A. Murali, S. Sen, S. Patil, K. Goldberg, and A. M. Okamura. A paced shared-control teleoperated architecture for supervised automation of multilateral surgical tasks. In *Proc. Intelligent Robots and Systems (IROS)*, pages 1434–1439, 2015.

[36] Pete Shinners. Pygame. http://pygame.org/, 2011.

[37] T. Somers and G.A. Hollinger. Human–robot planning and learning for marine data collection. *Autonomous Robots*, 40(7):1123–1137, 2016.

[38] A. Steinfeld, T. Fong, D. Kaber, M. Lewis, J. Scholtz, A. Schultz, and M. Goodrich. Common metrics for human-robot interaction. In *Proc. ACM SIGCHI/SIGART conference on Human-Robot Interaction*, pages 33–40, 2006.

[39] J. Storms, K. Chen, and D. Tilbury. A shared control method for obstacle avoidance with mobile robots and its interaction with communication delay. *The International Journal of Robotics Research*, 36(5-7):820–839, 2017.

[40] A. Streenan and N.E. Du Toit. Diver relative auv navigation for joint human-robot operations. In *OCEANS*. San Diego, CA, 2013.

[41] T. Suzuki, T. Sekine, T. Fujii, H. Asama, and I. Endo. Cooperative formation among multiple mobile robot teleoperation in inspection task. In *Proc. IEEE Conference on Decision and Control*, pages 358–363, 2000.

[42] L. Takayama, E. Marder-Eppstein, H. Harris, and J. M. Beer. Assisted driving of a mobile remote presence system: System design and controlled user evaluation. In *Proc. IEEE Conference on Robotics and Automation*, pages 1883–1889, 2011.

[43] R. Vasudevan, V. Shia, Y. Gao, R. Cervera-Navarro, R. Bajcsy, and F. Borrelli. Safe semi-autonomous control with enhanced driver modeling. In *Proc. American Control Conference (ACC)*, pages 2896–2903, 2012.

[44] J. Wang, L. Zhang, D. Zhang, and K. Li. An adaptive longitudinal driving assistance system based on driver characteristics. *IEEE Transactions on Intelligent Transportation Systems*, 14(1):1–12, 2013.

[45] Y. Wu, Y. Su, and Y. Demiris. A morphable template framework for robot learning by demonstration: Integrating one-shot and incremental learning approaches. *Robotics and Autonomous Systems*, 62(10):1517–1530, 2014.

[46] X. Yang, K. Sreenath, and N. Michael. Online adaptive teleoperation via incremental intent modeling. In *Proc. ACM/IEEE International Conference on Human-Robot Interaction*, pages 329–330, 2017.