PLANNING FOR HUMAN-ROBOT INTERACTION: REPRESENTING TIME AND HUMAN INTENTION

Frank Broz

CMU-RI-TR-08-49

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

December 2008

ABSTRACT

THIS thesis proposes a novel approach to planning for a specific class of human-robot interaction domains: those in which robots engage in tasks with humans that are governed by social conventions. When humans perform these social tasks, they try to achieve their own goals in an environment that they share with other people. Each participant has their own individual goals, and the interaction is neither purely cooperative nor adversarial in nature. However, the actions of others may have a direct impact on each person's ability to achieve their goals.Social conventions exist as a guideline for how to interact with others so that all parties involved can achieve their goals efficiently without interfering with one another. Recognizing what goals others are trying to achieve and performing actions at the appropriate time in the interaction are critical abilities for social competence.

Adding a robot into these systems without upsetting the social equilibrium is challenging. The approach to this problem taken in this thesis focuses on creating more accurate models of social tasks for planning. Because the human participants are modeled as a part of the environment, the world state in these problems is dynamic and partially observable. Specifically, the intentions of the humans are represented as hidden state in a partially observable Markov decision process (POMDP), and the time-dependence of action outcomes are explicitly modeled for both the humans and the robot. A model structure designed by a human expert is combined with experimental data of humans performing the task in question. The resulting models are large and complex. State aggregation over the time dimension of the state space is used to trade off between the quality of the representation of time and the model's size in order to find sufficiently expressive models that can also be solved tractably. The performance of the policies obtained are evaluated both in simulation and in interactions with human participants. The utility of this approach is demonstrated by using it to implement a controller for a mobile robot that rides elevators with people and an agent in a driving simulator that attempts to take the Pittsburgh left at an intersection with human drivers. Performance is evaluated by comparing the policies obtained using the proposed modeling technique to policies developed using less expressive representations, and by evaluating objective performance criteria and people's subjective responses to interacting with the agent in the simulated driving task. The policies for time-dependent POMDP models with human intention as hidden state outperformed the policies of the less expressive models, achieving both higher rewards during interaction and more positive evaluations for naturalness and social propriety of behavior.

ACKNOWLEDGEMENTS

Thanks to my advisors, Illah Nourbakhsh and Reid Simmons, for many years of guidance and interesting discussion. Thanks to Geoffrey Gordon for making time for me in a busy schedule and always having a fresh perspective to offer. And thanks to Sven Koenig for all his valuable feedback and advice.

So many people made an impact on this research (and on me during my time at CMU) that it would be impossible to list them all. But I would like to single out a few whose contributions were especially significant. Thanks to Trey Smith for releasing the excellent ZMDP software package and going above and beyond in offering support in its use. Thanks to Aaron Steinfeld for his useful suggestions about the analysis of my human subject experiment. Thanks to Suzanne Lyons Muth for making being an RI grad student easier to navigate in countless ways. Thanks to all of the members of the semi-Markov and social robots reading groups for many intellectually stimulating discussions that helped me to form the ideas behind this work. Thanks to Brennan Sellner all his help in the lab on a day-to-day basis and for day-to-day commiseration. And thanks to Maayan Roth for many conversations about POMDPs and basically everything else that helped to keep me going on the long path to graduation.

Thanks to all the friends inside and outside of CMU who helped keep me sane. Special thanks to the other members of the bands I was lucky enough to be a part of, No One Has to Know and Soldiers & Sailors, for providing me with a creative outlet and many hours of fun. And, last but definitely not least, thanks to my parents for the unfailing support they offered in just about every possible form.

DEDICATION

For my parents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATION

- $\mathcal{S}$: the state space
- $\mathcal{S}_t$: the set of states at timestep t
- s_t : the state at time t
- r_t : the reward at time t
- $\mathcal{A}$: the action space
- a_t : the action at time t
- $Pr\{\}$: the probability of an event
- π : a policy
- δ : a one-step policy mapping each state to an action
- γ : a discounting factor
- $\mathcal{O}$: the observation space
- $\mathcal{E}$: the set of state variables representing the environment
- $\mathcal{T}$: the (singleton) set representing the time variable in the state space
- $\mathcal{S}_P$: the state space defined by the environment state variables

CHAPTER 1

Introduction

1.1. Motivation

HUMANS are social animals. It is hard to imagine thinking that a person has achieved basic competency at a task if they can perform it only when the environment is cleared of other people. This obvious statement has deep implications for the design of robots whose purpose is to act as helpmates to humans. Almost any task performed in a populated environment will involve some amount of interaction with people. In order to be useful, such robots must have a way to create policies that respect human behavioral norms.

As mobile robots move into working environments such as offices, hospitals, and nursing homes, they face new challenges in human-robot interaction. In order to be a help rather than a hindrance, robots must be able to navigate through crowded areas and engage with people without disrupting the flow of human traffic. In these social tasks, people's intentions cannot be directly observed by the robot, but they are easily inferred through observation by other humans. A robot needs its own models of social behavior to be able to respond intelligently in these ambiguous situations.

The mental models people have of social behavior includes the role of time in the way an interaction evolves. Time may be critical to social interaction in a number of ways. Because these interactions usually take place in dynamic, changing environments, people may reason about time in order to coordinate their behavior with the occurrence of significant environmental events. But people are also concerned about the amount of time that an interaction takes even in the absence of external events. As time passes, the way that they behave in an interaction may change as well, as they lose interest or patience. Because humans are constantly reasoning about time, it is important that an agent that interacts with them be able to reason about it as well. Unfortunately, in many computational models used for action selection, the explicit representation of time is abstracted away for the sake of computational efficiency. Approaches based on these models may be unable to perform satisfactorily in many social tasks.

At the most general level, this problem can be thought of as one of acting in a particular kind of multi-agent system: one in which self-interested agents pursue their own goals in a shared environment where the other agents are acting according to some set of guidelines for behavior. In most cooperative and many adversarial domains, the goals of the other agents are assumed to be known. In a domain where the agents are pursuing one of a number of possible goals, the intent of another agent may not immediately be clear. By observing the actions of the other agents, their intention can be inferred, which will improve an agent's ability to coordinate its actions and achieve its own goal. Unlike many multi-agent systems, however, there is only the opportunity to control one agent out of the population. In order to be preserve the patterns of behavior that allow all of the agents to coordinate their actions to achieve their individual goals, this agent must act in a way that conforms to the rest of the existing system. This leads to a criteria for ″successful″ behavior that is broader and more holistic than the more common notion associated only with a single agent's goal achievement. Behaviors that are myopically successful at reaching an agent's immediate goal but violate norms for socially acceptable behavior may damage the stability of the entire system. Other agents, losing trust that social guidelines will be followed, may also act without consideration for others, creating conflicts between agents that reduces the opportunities for success for all the agents involved.

In situations where the agents are all people and the goals are achievement of everyday tasks, these guidelines for behavior and their interactions are called ″etiquette″. Social interactions are stable systems in which people play different roles based on their goals. The constraints on acceptable behavior in social situations make the interactions manageable to model. Familiar, easily recognized patterns of behavior make the actions of others' comprehensible and predictable. Following behaviors that they have observed to be successful for others saves people the trouble of coming up with a novel way of achieving their goal on the spot. The acceptable behaviors result in relatively ″fair″ outcomes because people know that in a later interaction they may be in the other person's role. People act not only out of self-interest, but in a manner that that is beneficial to the people with whom they interact. By acting to maximize the joint benefit of the interaction, they improve the long-term benefit to themselves by reinforcing these social rules. Even when people's goals are in conflict, social guidelines allow these conflicts to be resolved in one person's favor in a way that doesn't cause a complete breakdown in interaction.

The central focus of this thesis is to produce a practically useful computational model for the purpose of controlling an agent in *situated social tasks*, which are defined to have these characteristics:

- Goals that are rooted in the physical world and whose attainment is objectively verifiable (e.g., ″get to the end of the hallway″, as opposed to ″make friends with this person″)
- Performed in the presence of other social actors and require some level of interaction or coordination in order to be successful for all agents involved in the long term
- Familiar protocols for the achievement of goals are assumed to be known by all participants

As a motivating example, consider an autonomous robot riding an elevator with human passengers, a social task of time-dependent interactions with both the environment and people. To be successful, the robot must plan not only to get itself into, or out of, the elevator while the doors are open, but also to infer the intentions of other passengers so as not to interfere with them. If people want to exit the elevator as the robot is entering, it is probably best for the robot to yield to them (as etiquette dictates) in order to avoid colliding with someone or blocking them from exiting the elevator. There is a danger of deadlock if the robot doesn't move to enter the doorway at the right time. It should not wait too long, however, or else the doors will close before it can get on. Similarly, when on the elevator, the robot should try to infer the intentions of the others on board in order to know where it should position itself so as to not prevent them from exiting. People may also behave more assertively or brusquely if they believe the doors may close before they can get on or off, so their actions are also time-dependent.

Driving in traffic is also a social domain in which self-interested agents are following a set of guidelines for interaction. The intentions of the other drivers are not directly observable. Exogenous events in the environment, such as the changing of traffic lights, provide external cues that people use to coordinate the interaction. In many cases, driving interactions are governed by social rules in addition to traffic laws. One example of such an interaction is the ″Pittsburgh left″. The Pittsburgh left is a technically illegal, yet locally common driving maneuver that seems to have developed in response to Pittsburgh's many narrow, two-lane intersections that have neither a turn lane nor a left turn signal (Wereschagin, 2006). If the oncoming car opposite it chooses to yield, a car making a left turn takes the turn immediately after the light turns green (as if there were a left turn signal). This action allows the cars behind the turning car to pass through the intersection, rather than having to wait for the next light as they would if the turning car waited until all oncoming cars had passed. Whether the Pittsburgh left will be taken is negotiated between the two cars through a variety of nonverbal signals, ranging from creeping towards the intersection while the light is red to flashing their headlights at each other.

Standing in line is another task in which others' intentions may be ambiguous because there can be uncertainty as to who is, or is not, in line. Both spatial and temporal cues can help – people leaving too much space (compared to their personal space) are probably not in line, and neither are people who do not move up within a reasonable amount of time after the rest of the line moves. The uncertainty is in how much space is "too much" and how much time to allow before deciding that the person is not in line. Of course, one can always ask the person about their intentions, but one would prefer to use that sparingly, since it is intrusive and potentially annoying. Thus, the ability to reason about the utility of information gathering actions is important.

There are a number of domains that do not include physical robots interacting with people that share many characteristics with the previous tasks. One example is dialog management: the conversational goals of a person will determine what they communicate, but which goal a person has may not be known to the conversational agent beforehand. Also, in conversation, time can provide valuable information. The amount of time it takes a person to respond can give the agent insight into whether its last response was appropriate or something they were not expecting.

In order to solve these kinds of problems, a model-based planning approach is taken. The models used represent the relationship between people's observable actions and their unobservable intentions. Models are created from a combination of domain-specific rules and data of humans performing the same task. Because of the importance of time in these interactions, models that allow the representation of time-dependent action outcomes are employed.

1.2. Acting in Social Domains

One might question whether it is necessary to use planning to arrive at a policy for action. Shouldn't domain experts be able to hand-code policies for interaction, especially since social behaviors tend to follow regular patterns? While social interaction is regular in a way that makes it easy to describe people's roles in an interaction at a high level, human behavior is still highly variable within these roles. Hand-coded policies are likely to be brittle and would require a great deal of tweaking to get reasonable performance from. Also, there is no reason to expect that any part of a hand-written policy can be reused, even on related tasks set in the same environment under similar conditions. It would be preferable to have a model that could be specified in terms of high-level rules and used to create policies that respond reasonably to a wide range of behavior.

Another approach to developing policies for interaction is to learn them directly from interacting with people using reinforcement learning methods (Sutton and Barto, 1998). But learning typically requires large amounts of data. It may take prohibitively long, or be prohibitively expensive, to collect enough data to reach an adequate level of performance. Also, unless the interaction is one that already involves one participant taking on a teaching role, having the agent learn the interaction online may fundamentally change the way that people perform their part in the interaction versus how they would if they were interacting with a human peer. By using an existing model for planning (based on both expert knowledge and available data), policies for action can be found that perform well without an online learning phase.

The planning paradigm on which this work is based is decision-theoretic planning using probabilistic graphical models. Probabilistic graphical models are typically used for planning in situations where there is uncertainty about the state of the world, such as the outcomes of actions an agent may take. This uncertainty is represented by probability distributions. Decision-theoretic planning creates a policy by finding the actions that maximize the expected reward (also commonly referred to as cost for negative values) that can be obtained according to a given model. Decision-theoretic approaches have proven themselves useful in a variety of real world planning domains, particularly in robotics, because of the flexibility of defining agents' goals in terms of a utility function and the ability to model the uncertainty of action outcomes and observations that so often arises in physical domains (Nourbakhsh et al., 1995; Simmons and Koenig, 1995).

The particular model this thesis focuses on, the partially observable Markov decision process (POMDP), is capable of representing situations in which the entire state of the world is not directly observable and making use of this hidden state information while planning. The probabilistic relationship between underlying state and the aspects of the state that can be observed by the agent is included in the model. Rather than a mapping from states to actions, a POMDP policy maps beliefs (probability distributions over the states) to actions. This belief distribution is maintained during execution according to the observations that the agent receives while acting. This model is of particular interest for social interaction domains, because even if an agent were able to sense its physical surroundings perfectly, the mental states of the people with which it interacts remain unobservable.

This thesis is also concerned with the importance of representing time's role in social interactions. This is a challenge because most decision theoretic models for control make simplifying assumptions to represent action outcomes as dependent only on the current state for the sake of computational efficiency. Semi-Markov models (which allow more sophisticated representations of the relationship between time, state, and action) exist, but are much less widely used (Puterman, 1994). Their application is less straightforward than fully Markov models, particularly in the case of POMDPs, where finding solutions to problems of a realistic size requires the use of sophisticated heuristic algorithms that may not readily translate to semi-Markov extensions. A significant portion of this work concerns representing time-dependent action outcomes in a fully Markov model that can be used with existing state-of-the-art POMDP solvers. Representing time dramatically increases the size of the state space, which can also significantly increase the time required to solve the model for a policy. In order to mitigate this effect, a reward-based state-aggregation method that operates on the time dimension of the state space was developed. Simulation experiments conducted using models of the elevator riding and the Pittsburgh left domains demonstrated the ability of time-state aggregation to achieve significant improvements in policy solution time while maintaining high-quality task performance.

The approach proposed in this thesis was inspired by observing the characteristics of the problem of planning for a robot to effectively navigate in populated spaces and perform goal-oriented social tasks with people in a socially appropriate manner. Experience with these problems suggested that the most common ways of modeling problems using decision-theoretic planning formalisms were inadequate to express their fundamental characteristics. Models that assign reward to an agent only for purely self-interested action are unlikely to produce policies for socially acceptable behavior. Additionally, models should include information about the relationship between people's unobservable intentions and their observable action in order for an agent to be able to interpret their behavior. And because of the role time plays in these types of interactions, a model that allows a rich representation of time-dependence is required. But representing hidden state and time-dependent action outcomes are design decisions which may dramatically increase the complexity of acquiring policies for action when compared to less expressive representations. In order to justify this increase in model complexity, it is necessary to demonstrate that it does actually have a significant positive impact on the quality of the policies obtained. For this reason, a significant portion of this thesis is concerned with the evaluation of the proposed representation relative to simpler representations that could alternatively be used.

Statement of Hypothesis: This thesis describes a general framework for modeling and planning for situated social tasks. The intentions that direct people's behaviors are modeled as hidden state and the time-dependent aspects of the interaction are explicitly modeled. The hypothesis is that planning for social interaction with a model that represents time dependent action outcomes and allows for planning over uncertainty in others' intentions will achieve better results when compared to similar models that make fixed assumptions about people's intentionality or abstract away time-dependence in action outcomes.

The superiority of this modeling technique will be demonstrated in two ways:

- For a global reward that combines the rewards obtained by all interaction participants, the policies will obtain higher reward.
- The policies produced will be subjectively more acceptable to people who interact with the agent.

In order to demonstrate the effectiveness of a planning approach for social interaction, evaluating policy performance through interaction with humans is vital. The Pittsburgh left domain was used to conduct this evaluation. People performed the task in a driving simulator with agents controlled by policies from the time-state aggregated POMDPs and from less expressive time-indexed MDPs and POMDPs without time in their state space. The time-state aggregated POMDPs achieved statistically significantly better results than the other models, both according to the reward obtained and to people's subjective impressions of their performance as measured by a survey. The results for the time-state aggregated POMDPs were also the closest to the results obtained for humans interacting with other humans while performing the same task. These results demonstrate both the relative superiority of these representation choices and the overall effectiveness of the general approach to planning for social interaction.

1.3. Contributions

This thesis demonstrates a novel framework for modeling social systems involving people and robots and designing controllers for robots in human-robot interaction domains using decision-theoretic planning. Because reasoning about the intentions of others is so central to acting in social domains, the representation chosen is that of a POMDP with human intention as part of its hidden state. The models are created by combining a prior model compiled from domain-specific probabilistic action rules with data of humans engaging in the interaction. The data collection is conducted in such a way that a person's intention (usually an unobservable quantity) can be reliably associated with each data trace. The robot's intentions are represented by reward structure that places an ordering over the the relative desirability of various outcomes. The reward structure also includes reward for the human's goal achievement, in order to produce action motivated by an interest in the common good for both interacting parties. This general framework can be used as a template for creating models for a wide variety of social interaction tasks, not restricted to cases where the robot and human share a common immediate goal. It's effectiveness is demonstrated by using it to create a model of the Pittsburgh left interaction task which produced polices that performed well during real interactions with humans, as well as in simulation.

A novel time-state aggregation algorithm for POMP models with time-dependent state transitions is also introduced. Time-indexed models are used as a simple and flexible alternative to semi-Markov models for representing time-dependent action outcomes. This representation potentially leads to very large state spaces, which could be a major concern for the practicality of its use with POMDP models. In order to mitigate this effect on state space size, a reward-based state aggregation method was developed that operates exclusively on the time dimension of the state space. By varying the threshold used to select states for aggregation, an explicit tradeoff can be made between the fidelity of the model's time representation and the size of the state space. Guidelines are given for choosing a useful aggregation threshold for a particular model. The effectiveness of this method at reducing model solution time while producing high-quality policies is experimentally verified.

Additionally, this research contributes to the field of human-robot interaction by experimentally evaluating the performance of the policies developed through interactions with people. Policies from the time-dependent POMDP models are compared to policies from time-indexed MDP models and POMDP models without time in their state. Quantitative measures of performance are used to assess how successful the policies are and how similar their performance is to that of human participants in the same social task. Subjective evaluations of the policies' performance by the human participants who interact with them at this task provide confirmation of whether the model design framework produces natural and socially appropriate behavior. The superiority of the time-dependent POMDP model policies according to these subjective and quantitative measures has implications for the future design of agents for social tasks. Experimental results demonstrate that an inadequate representation of time in a model can produce policies that exhibit behavior that is both socially inappropriate and irrational given an understanding of the time-dependent aspects of the problem. Additionally, experimental results show the benefit of using POMDP models to create policies capable of acting based on beliefs about a person's intentions. People interacting with these POMDP policies found their behavior to be more natural and socially appropriate than policies based on simpler models that acted according to fixed assumptions about the human's intention.

1.4. Outline

- Chapter 1 introduces the focus of the thesis and explains the motivation for the work.
- In Chapter 2, aspects of the study of social behavior in psychology and related fields with relevance to human-robot social interaction are discussed and a general overview of Markov decision processes and their variants is presented.
- Chapter 3 discusses related work.
- Chapter 4 presents a technical overview of the research problem and the general framework for its solution and introduces the problem domains that will be used for experiments: elevator riding and the Pittsburgh left.
- Chapter 5 describes the data collection experiment for the Pittsburgh left driving simulation task in detail. A summary of the data and survey results are presented and interpreted.
- Chapter 6 describes how the POMDP models are created from their rule-based description, how the reward structure for the models is designed according to the agent's intentions, and how the human data is combined with the prior model.
- Chapter 7 describes the reward-based time-state aggregation method used to reduce the state size of the time-indexed POMDP models. The impact of this state-space reduction on solution time is demonstrated for a variety of models from both of the example domains.
- In Chapter 8, experimental results are analyzed for the models' policies both in simulation and in experiments with human participants in the Pittsburgh left driving simulation task.
- Chapter 9 concludes and summarizes the thesis and discusses possible future directions for this line of research.

CHAPTER 2

Background

2.1. Modeling Social Interaction

2.1.1. Psychology

THERE are a number of disciplines in the humanities that are concerned with people's social behavior and interaction, with social psychology being the one that primarily focuses on the individual rather than large social systems and institutions. The majority of social psychology research focuses on how social norms and prejudices influence people's attitudes and behavior rather than on how social interactions occur and progress over time. The aspects of social interaction that social psychologists have primarily been interested in measuring do not provide a great deal of guidance to roboticists as to how to design social robots. However, the methodologies they have employed to measure human behavior are extremely useful in order to design experiments that evaluate whether robots meet human criteria for social interaction.

One researcher who influenced social psychology and, through it, human-robot interaction, is Edward T. Hall. Hall, an anthropologist, was among the first researchers of nonverbal aspects of communication. He invented a field of study he named *proxemics*, which was concerned with the social role of spatial relationships between people. Hall detailed the physical distances between people that were commonly considered acceptable for different levels of social engagement, and measured the way that these distances differed across cultures (Hall, 1968). Hall's work provides a theory that can easily be implemented in concrete terms and is relevant to a wide variety of everyday interactions. Because of these characteristics, Hall's research has been used as the basis for a number of approaches to socially aware navigation for mobile robots which will be discussed in in Section 3.1.1.

While most of social psychology still focuses on the internal mental states of the individual, there has been a movement within the field to incorporate an ecological approach, emphasizing the relationship between social perception and social action and focusing on the interaction as the basic unit of study rather than the individual (Good, 2007). Observational studies of this sort reveal that there is a rich social protocol even for the process of avoiding and ignoring others in a socially appropriate manner, such as when riding a crowded elevator with strangers (Hirschaur, 2005). This way of looking at interaction as a system embedded in a specific physical environment that provides its context, is an inspiration to the modeling approach taken in this thesis.

Cognitive psychology has addressed a topic of relevance to how people conceptualize social interaction, the relationship between human behavior and people's goals and intentions. Recognizing goal-directed intentional behavior in others seems to be a fundamental human social ability, and one that develops very early in life. In a study conducted on 18-month old children, the subjects watched an adult try and always fail to perform a variety of simple puzzle manipulation tasks (Meltzoff, 1995). The children were later able to produce the intended behavior themselves without ever witnessing a successful task completion. When the same failure behaviors were demonstrated by a simple non-humanoid manipulator, the children did not go on to exhibit the intended behavior themselves, suggesting that they did not think of the manipulator's actions as failed attempts at a goal. Intentions have been conceptualized as part of a hierarchical action production framework linking high-level desires to concrete plans for action (Bratman, 1990). Baldwin and Baird provide an overview on research in cognitive science that explores how and under what conditions people interpret the actions of others as intentional, using the studies discussed to support a hypothesis that people use generative models of intention-guided action to interpret human behavior and detect people's intentions (Baldwin and Baird, 2001). Some researcher have gone so far as to state that it is people's ability to hold shared intentions (rather than just recognizing the intentions of others) that separates adult humans from children and primates and enables the richness of social behavior that human society enjoys (Tomasello et al., 2005). These studies show that humans use the concept of intention to reason about and interpret the actions of others. Furthermore, they suggest that intentions are conceptualized as something distinct from (and most likely hierarchically above) immediate goals, as intentions can be recognized even when the goal most closely associated with an intention's desired outcome has not been achieved.

Social interaction has also become a subject of research in neuroscience, with researchers attempting to understand where and how we process information about other people and how it relates to how we process information about ourselves. Research indicates that not only do people recognize goal-oriented action in others, but that they reason about the goals of others in a way that is similar to the way that we reason about our own goals. Frith and Frith present experimental results which indicate that the same structures in the brain used to monitor our own behavior and represent our own goals are also used to form a model of others' mental states during social interaction (Frith and Frith, 2001). In another fMRI study where subjects watched an animated human figure perform grasping actions, the reaction to goal-directed and non-goal-directed actions could be distinguished in parts of the brain known to be related to processing the actions of others (Pelphrey et al., 2004). An additional study observed differences in brain activation when watching chasing behaviors between instances where the pursuer merely followed the target versus when it seemed to predict the target's path during pursuit (Schultz et al., 2004). These studies suggest that the human brain has mental apparatus specially devoted to recognizing intentional, goal-directed behavior. People seem to organize their understanding of the actions of others based on their intentions. The intention-based organization of action motivates the modeling approach taken in this thesis, in which actions and rewards are described in relation to their associated intention.

There is further experimental evidence that there is something special about how people process social interaction beyond how they process other intentional human behavior. An FMRI study where people observed others engaged in independent or social tasks suggests that there are areas of the brain concerned primarily with reasoning about intentions of people in social interactions (as opposed to more generally about intentions of isolated actors) (Walter et al., 2004). A survey of studies on shared mental representations of social tasks reveals that the task model is richer than just the agent's and other agents' goals, and captures the way that these interactions are situated in their environment. The shared task model allows people to predict other's actions based on events occurring in environment rather than just on their previous actions (Sebanz et al., 2006). It is commonly understood that people use domain knowledge about the timing of events in a dynamic environment to anticipate their occurrence. If people also use these events to predict the future actions of others, it is clear that representations that allow for reasoning about time are important for capturing the intricacies of social interaction. Additionally, people's mental models of social tasks appear to situate the actions of others in their specific environmental context. The way that that people's actions are modeled in this thesis as part of the evolution of the overall environment surrounding the agent is inspired by this way of thinking about social behavior.

2.1.2. Social Behavior and Rationality

There has also been research on social interaction that attempts to explain social behavior as arising from people's ability for rational decision-making. Drawing on ideas from decision theory and game theory, Lewis defined social conventions as a way for a population to select which strategy they will all play in a coordination game with multiple equilibria (Lewis, 1969). Each member of a population has a reasonable expectation about which strategy the other member's will play (the strategy that they all played to achieve the equilibrium last time), allowing them to coordinate future behavior. Lewis also introduced the concept of common knowledge, which was later formally defined by Aumann (Aumann, 1976). Common knowledge is information known to all players, that all players know the other players also know (Geanakoplos, 1992). Referring to common knowledge can short-circuit the potentially infinite recursion of trying to reason about what you know about what I know about what you know... and so on. While this research is similar to the approach in this thesis in that it conceives of social behaviors as coordinated action arising from rational decision-making, a classical game-theoretic approach relying on finding equilibria is difficult to apply to realistic problem domains. The social interactions discussed in this research would, if represented as games, be sequential stochastic games with imperfect information. Practical approaches to solving games of this type are in their very early stages (Rosemary Emery-Montemerlo and Thrun, 2004). This type of game theoretic modeling becomes even less applicable when one considers the time-dependence of these domains, as issues of time representation are largely unaddressed in most work on game theory.

In some human social interactions, certain players may choose a strategy that appears to be dominated in terms of the immediate payoff to the agent. This "altruistic" behavior in game theory is often explained by using mechanisms such as reputation to enforce cooperation (Mui et al., 2002). There are real-life situations, however, that are clearly not governed by these mechanisms and in which seemingly "altruistic" behavior still occurs (any number of anonymous interactions that follow rules of etiquette, for example). In evolutionary game theory, altruistic behavior can be explained as rational if it increases the possibility of passing on some of the agents' genes through relatives who benefit even if it does not benefit that particular agent (Maynard Smith, 1998). But human altruism is often based on social factors other than blood relation. Some researchers in the social sciences see this as a failure of game theory to accurately describe and predict human behavior. An article by Coleman in *Behavioral and Brain Sciences* proposing a "psychological game theory" and its rebuttals provide a perspective on the differing views of psychologists, biologists, philosophers, and game theorists on the ability of game theory to explain human social behavior (Colman, 2003). It is important for the purposes of this research to note that there are cases in which the strategies of interacting agents seem to be in what we would intuitively think of as an equilibrium, but which would be difficult or impossible to show to be a game theoretic equilibrium using the types of world models currently used in game theory. In such cases, existing game-theoretic models are of limited utility if the goal is to accurately model people's behavior rather than to describe what idealized, perfectly rational agents would do. The work in this thesis sidesteps the issue of finding equilibria in social interaction domains or modeling how such equilibria arise. Instead, the focus is on modeling the social behaviors observed in a way that allows the agent to find a socially appropriate rational response. How the people arrive at the intentions that they may have for an interaction (which may include preferring ”altruistic” goals), is beyond the scope of this thesis.

In the area of multi-agent systems, formal definitions of social convention based on the work of Lewis have been developed, and the conditions under which they are adopted and propagate have been explored through simulation (Shoham and Tennenholtz, 1997). Agent simulations of the propagation of conventions has shown that even large groups of agents can converge on a convention quickly in cases where their social network's structure has a few characteristics typical of real large-scale social networks (Delgado, 2002). Berninghaus and Ehrhart present experimental results which show that players of a coordination game are more likely to play the Pareto dominant equilibrium when there are larger numbers of repetitions of the game (Berninghaus and Ehrhart, 1998). They hypothesize that this is the case because players are more tolerant of early losses that will lead to higher payoffs over time. Norms are also modeled in certain multiagent systems as a mechanism to punish agents to who fail to cooperate in the achievement of group goals or to restrict the available behaviors through social laws (Boella and van der Torre, 2003),(Felicissimo et al., 2006). This simulation research is interesting way of exploring hypotheses about how social behaviors develop, but is not immediately relevant to the control of a social agent acting in realistic domains with human actors. Additionally, multi-agent research on social convention focuses on coordination for purely cooperative tasks, not addressing the more general case where social conventions may coordinate behavior for tasks in which the agents have potentially conflicting goals.

2.1.3. Social Behavior and Time

People are sensitive to the passage of time in social interaction. Studies indicate that even infants show lower levels of attentiveness to video feeds of their mothers if their mother's responses are delayed by a second (Striano et al., 2006). In fact, the idea of social interaction as something that unfolds through reciprocal actions over time is so deeply ingrained that the duration and structure of an interaction with another person can influence whether or not we conceive of it as a social interaction. In one series of experiments, human subjects played coordination games where each made their decisions either simultaneously or pseudo-sequentially (meaning that that they made decisions one after another but couldn't observe earlier decision makers' choices) (Abele et al., 2004). When playing pseudo-sequentially, people were more likely to cooperate than when playing the simultaneous version of the game. The authors explained this difference in behavior as the effect of people conceptualizing the game as a social interaction when the decisions were made over time and as a game when all decisions were made simultaneously.

Knowledge of timing is an important part of the knowledge about an interaction task both for predicting others' future actions and for taking appropriate actions yourself. Analysis of turn-taking behaviors in conversation have shown that the length of pauses and overlaps in these interactions is task-dependent, with different patterns for problem-solving tasks than for social chatting (Bosch et al., 2004). This suggests that in conversational tasks, the amount of time that passes before a response can provide information about the cognitive load placed on the individual, which may provide evidence to an agent about whether its statement or question to the human was expected. Also, people use the occurrence of environmental events to predict the future behavior of other agents (as mentioned previously in research on people's task models of social interactions (Sebanz et al., 2006)). The time at which they expect these events to occur influences their own actions and their predictions about the future actions of others. Because the human's actions make up part of the environment model in the approach taken in this thesis, a representation is necessary that can accurately model how people's actions may change in anticipation of events as well as in response to them.

2.2. Markov Decision Processes for Planning

The planning approach used in this thesis is decision theoretic planning with probabilistic graphical models. In particular, models which can represent partially observable state spaces and time-dependent action outcomes are used and compared to less expressive models. This section begins by introducing the *Markov decision process* (MDP), which is the most basic form of the class of models to be discussed. Then the *partially observable Markov decision process* (POMDP) is presented. After this the discussion proceeds to models and policies that represent action in terms of the time in a problem as well as the state. *Non-stationary* policies and the model representations that may be used to create them are discussed. Finally, the *semi-Markov decision process*, an extension of the MDP that allows a for a less restrictive model of the time spent in a state, is introduced and its limitations relative to the time-indexed state space is discussed.

2.2.1. Markov Decision Processes

The most common representations for sequential decision models in decision-theoretic planning are Markov decision processes. A *Markov decision process*, or MDP, is made up of the tuple $< \mathcal{S},\ \mathcal{A},\ T(s,a,s'),\ R(s,a) >$: a set of states, a set of actions, a transition function defining the probability of moving from one state to another for a given action, and a reward function defining the value of each action when taken at each state. Actions are taken at fixed decision epochs in the continuous time case or timesteps in the discrete case (this thesis will be concerned only with discrete time models).

This model is so named because it takes advantage of the Markov property as a way of limiting the complexity of the planning problem. The *Markov property* is defined as:

$$Pr\{s_{t+1} = s', r_{t+1} = r | s_t, a_t, r_t, ..., r_1, s_0, a_0\} = Pr\{s_{t+1} = s', r_{t+1} = r | s_t, a_t\}$$

The environment's response, given by the state transition and reward functions, at time $t + 1$ depends only on the state and actions at time t, and the rest of the execution history can be ignored. Systems that obey the Markov property may also be referred to as *memoryless* for this reason.

A *policy* is a mapping from states to actions for every state in the model at every timestep at which an agent may act. MDPs may have either a finite or an infinite planning horizon. For *finite horizon* problem, the system will terminate after a pre-defined number of timesteps. The *horizon length* of a planning problem is the number of timesteps from the time the agent is currently acting at to the timestep at which the agent makes its final decision and the problem terminates. The horizon length is distinct from the *chronological time* in the system, which describes the number of timesteps which have passed since the problem began. A policy for a finite-horizon model is dependent on the horizon length because it determines which states it is possible to reach and receive reward from before the problem terminates.

The finite horizon policy for action in a MDP is directly related to the more commonly used *infinite horizon* solution. First the way that a finite horizon policy is built from one-step action will be explained, and then the infinite horizon case will be presented. Allow δ to be a 1-step policy that maps each state to a corresponding action. A n-step finite horizon policy will be made up of n 1-step policies, each corresponding to the number of timesteps before termination of the problem at which it should be used.

$$\pi_n = \{\delta_n, ..., \delta_1\}$$

Given a policy, every state has a value that corresponds to the expected reward that can be obtained by following the policy when acting in the model. The variable γ is a discounting factor, a number between 0 and 1 that is used to reduce the value of less immediate rewards. For finite horizon problems, γ is most frequently set to 1. The value for each state for the planning horizon t can be computed from the state values for the $t-1$ horizon, along with the reward and state transition function for the problem.

$$V_t^{\delta_t}(s) = R(s,a) + \gamma * \sum_{s'} T(s, \delta_t(s), s') * V_t^{\delta_{t-1}}(s')$$

For a model with an *infinite horizon*, there is no fixed time at which the model terminates and an agent may continue to act within the model forever. This means that the same states (and therefore rewards) are reachable from a state no matter what time that state is entered. Policies for infinite horizon problems are therefore not dependent on the horizon length.

$$\pi_\infty = \{\delta_1, ..., \delta_n, ...\} = \delta \ \ \forall i,j \ \delta_i = \delta_j$$

The form of the equation for state values for an infinite horizon model is similar to that of the finite horizon model. One difference is that the discounting factor γ for an infinite horizon problem must be less than 1 so that the value of the states will converge to a finite number.

$$V_\pi(s) = R(s,a) + \gamma * \sum_{s'} T(s, \pi(s), s') * V_\pi(s')$$

Both finite horizon and infinite horizon models may have absorbing states (states with no outgoing transitions) to indicate that a state has been reached where the task that the model describes is completed or otherwise over. These states are typically of zero reward, so they have no effect on the value of the policy. Infinite horizon problems are more commonly used for planning problems with a long planning horizon because the resulting policies are significantly more compact. This is because, unlike the finite horizon policy, the same actions are chosen in each state regardless of the timestep at which the agent is acting.

Under certain conditions, an infinite horizon policy can be used as a solution to a finite horizon problem. The infinite horizon policy is the policy that a problem converges to eventually as its horizon length is increased. For any timestep whose horizon length is longer than the horizon length at which the infinite horizon converges, the infinite horizon policy may be used, because it is the optimal solution at that timepstep. The number of timesteps that it takes for a problem to converge to the infinite horizon solution is often referred to as the *planning horizon* of a problem. In episodic problems of a fixed length, an infinite horizon policy may be used rather than a finite horizon as long as the agent is always guaranteed to reach an absorbing end state prior to the end of the episode (so that all possible future reward-giving states are actually reachable throughout the course of the episode).

2.2.2. Partially Observable Markov Decision Processes

The *partially observable Markov decision process* (POMDP) is an extension of the Markov decision process in which the true state of the system is not directly observable by the decision maker. Instead, the decision maker receives observations about features of the environment that arise probabilistically from the underlying state depending on what action was taken. A POMDP is defined as the tuple $< \mathcal{S},\ \mathcal{A},\ \mathcal{O},\ O(s,a,o),\ T(s,a,s'),\ R(s,a) >$, where $\mathcal{O}$ is a set of observations that the agent may receive about the state it is in and the function O gives the probability of perceiving each observation for each state and action prior to entering the state. Each state in a POMDP model corresponds to a dimension in its belief space (Kaelbling et al., 1998). Planning in a POMDP can be thought of as planning over the continuous belief space of the underlying MDP, the probability distribution over which state the agent is actually in when it receives an observation. The Markov property holds for POMDPs in the sense that the belief state (not the current underlying true state) is a sufficient statistic of the history of the system. The degree of belief for a state s' can be computed from an existing belief distribution over states and a new action and observation as follows (η is a normalizing factor for $b'()$).

$$b'(s') = \eta * O(s',a,o) * \textstyle\sum_{\mathcal{S}} T(s,a,s') * b(s)$$

The value function for a finite horizon problem of horizon k is piecewise-linear and convex and can be computed directly from the value function for the horizon $k-1$. The value function can be used as a representation of the policy by keeping track of the belief state during execution and looking up the best action given the current belief state. Unfortunately, the finite horizon policy is often very large, because a separate value function must be stored for each intermediate value of the horizon in order to choose the correct action for the agent at that timestep.

The value function for the infinite horizon can be computed in the same iterative manner. If an optimal infinite horizon solution can be found, the value functions of two successive horizon lengths will be the same and the policy will have converged. As in the MDP case, the convergence of the value function for the infinite horizon case relies upon the discount factor for the reward being less than one. Even if the problem does not converge to the exact optimal solution, it is possible to get arbitrarily close by extending the horizon length. The infinite horizon solution is a stationary policy, so the last value function computed is retained and the earlier ones are discarded. The infinite horizon policy may also be represented by a policy graph whose nodes represent actions and whose edges are labeled with the possible observations. This graph may be traversed during execution to select the appropriate action, making belief tracking unnecessary.

Finding the solution to a finite horizon POMDP is PSPACE complete (Papadimitriou and Tsitsiklis, 1987) and finding an infinite horizon solution is undecidable (Madani et al., 1999). The difficulty of solving a POMDP can not be predicted by the size of the underlying state space, though very large state spaces can make belief tracking expensive. The number of actions, observations, and the planning horizon all have an impact on the complexity of the solution. In the worst case, the optimal policy graph may have $|\mathcal{A}|^{|\mathcal{O}|^{|\mathcal{H}|}}$ nodes.

The model's flexibility and power make it attractive for use in many domains that involve an agent basing its decisions on noisy or ambiguous data. Though solving for even approximately optimal POMDP policies is very expensive, there are now a number of POMDP solution algorithms that can achieve good performance on reasonably large models (Pineau et al., 2003a; Poupart and Boutilier, 2004; Smith, 2007a). Trey Smith's approximate POMDP solution algorithms were used to solve the models in this thesis. These algorithms are competitive with or faster than other state-of-the-art approximation algorithms for a wide variety of benchmark problems, and are publicly released for use as part of the ZMDP software package (Smith, 2007b), implemented in C++. The two approximate POMDP solution algorithms that are used in in this thesis are described below.

2.2.2.1. HSVI. Heuristic search value iteration (HSVI) is an approximate POMDP solution algorithm that provides provable bounds on the reward obtained by the polices it produces with respect to the optimal policy (Smith and Simmons, 2004). The algorithm stores a compact representation of the upper and lower bounds of the value function over the belief state. After initializing the lower bound according to the worst case reward for each action and the upper bound by assuming full observability, the bounds are refined through heuristic search by making local updates to specific beliefs. The beliefs to update are chosen by heuristic depth-first search of a tree that represents how beliefs evolve according the actions and observations. The commonly used heuristic of exploring the action with the highest upper bound is used for action selection. To select the observation, the novel heuristic of selecting the observation with the greatest *excess uncertainty* (a measure related to the width of the bound on the belief) is used. The policy is obtained directly from the piecewise linear convex representation of the value function given by the lower bound. HSVI can be modified to yield an anytime algorithm by iteratively adjusting the convergence threshold for the bounds, though in this thesis a fixed threshold was always used.

2.2.2.2. FRTDP. Focused real-time dynamic programming (FRTDP) is another heuristic search-based approximation algorithm that is similar to HSVI in that it maintains upper and lower bounds on the value function. FRTDP differs in that it uses a different search strategy, choosing states based on cached priority information in order to avoid revisiting states that do not improve. FRTDP stores an *explicit graph*, a finite data structure holding information about states which have already been visited during search. While HSVI chooses an observation outcome based on the excess uncertainty of a state compared to its immediate successors, FRTDP compares a state to its distant successors on the fringe of the explicit graph. As in HSVI, the goal of the heuristic is to achieve the greatest reduction in the width between reward bounds. This algorithm also uses an adaptive termination criteria to prevent unnecessary exploration of long, low reward trials. FRTDP is typically faster to converge than HSVI, at the expense of having far greater memory costs because of the need to store the explicit graph. A description of FRTDP applied to MDP models is given in (Smith and Simmons, 2006).

2.2.3. Non-stationary Policies and Models

Most work with MDP and POMDP models focuses on finding stationary policies for action. In a *non-stationary* policy the action selected is dependent on the time as well as the state that the agent is acting in. Finite horizon policies are non-stationary because they depend on time in the form of the planning horizon. These are non-stationary policies that can arise from the solution of a stationary model (meaning that none of the functions that describe the model's structure change over time).

The MDPs and POMDPs discussed in the previous sections were stationary models. A *non-stationary* model is one in which one or more of the functions in its definition are dependent on time. This is not the same as a finite horizon problem, where only the policy is dependent on time in the form of the number of time-steps left until termination. For example, a model may have a non-stationary state transition function $T(s, a, t, s')$, in which the state transition depends on chronological time (the number of timesteps that have passed since the problem began, not the number of timesteps left until termination). Representations of this type may be necessary to achieve good performance in domains where time has a significant effect on action outcomes in a state. In these cases, models that do not represent the time-dependence accurately and solution techniques that do not allow an agent to choose different actions in a state based on time are likely to fail to capture important characteristics of the problem.

One way of converting a non-stationary representation of a problem into a stationary model is to augment the state space to add the necessary time information to the state. Consider, for example, a problem in which the state-transition function depends upon the chronological time of the system as well as the state. If a duplicate copy of each state is created for each timestep at which the system may be in that state, a new state-transition function can be defined for these augmented states that has equivalent behavior to the time-dependent state transition function for the original model. For a problem with a time index in the state space and a zero-reward absorbing state that is guaranteed to be reached after a certain number of time steps, it can be shown that the infinite horizon solution can always be used in place of the finite horizon solution. To illustrate this, consider an arbitrary state S in a problem that has N timesteps. The planning horizon from the start states at chronological time zero is N, because we know by definition that the system will be in an absorbing state in N timesteps. Because the number of timesteps left until the end is equal to the planning horizon, the optimal action for the infinite horizon model is the true optimal action. At time 1, it is impossible for the system to be in any state that has a time index of anything other than one, and the planning horizon is $N - 1$, with $N - 1$ timesteps left until reaching the absorbing state. Continuing this line of reasoning, it can be seen that the infinite horizon solution to a finite horizon problem with a time-indexed state space is equivalent to the finite horizon solution.

There is a difference, however, which may make the time-indexed solution strictly more compact than the finite horizon solution for certain problems. In systems where there are a set of start states that do not include all of the states in the state space, not every state is reachable at every (chronological) timestep. Note that this is separate from the notion of what states are reachable given the planning horizon. In the time indexed model, only the states reachable at a certain timestep need be included in order to find the correct infinite horizon solution. In order to find the corresponding finite horizon solution for an equivalent model without a time index in the state space, the optimal action must be computed for every state for each planning horizon, because there is no notion of chronological time and its impact on the reachability of states in the model.

2.2.4. Semi-Markov Decision Processes

Semi-Markov decision processes (SMDPs) are a generalization of MDPs that allow for the representation of limited non-Markov time-dependent aspects of the state. An SMDP is defined as the tuple $\langle \mathcal{S},\ \mathcal{A},\ T(s,a,s'),\ R(s,a),\ F(s,a)\rangle$, where F defines distributions over the possible duration of state occupancy. The time spent in a state is modeled by an arbitrary probability distribution $f()_t$ and the state transition probabilities are dependent on the time i_t that has been spent in that state (Puterman, 1994).

$$Pr\{s_{t+1} = s'|s_t, a_t, ..., s_0, a_0\} = Pr\{s_{t+1} = s'|s_t, a_t, i_t, f()_t\}$$

In an MDP, time spent in a state can be modeled only as a self-transition, which has a geometric distribution (or an exponential distribution in the continuous time case). SMDPs improve the accuracy of the representation of the time spent in the state, modeling temporally extended actions by allowing action selection only at decision epochs defined by the state transitions. Action selection may typically be based only on the current state, not on time. Extensions to SMDPs have been designed that allow actions to be chosen in a state at any time and that allow the action selection to be time-dependent (Cantaluppi, 1984; Chitgopekar, 1969; Stone, 1973). However, these variants of the SMDP model have not been used in planning applications, probably due to the complexity of developing a practically useful implementation. SMDPs, in the form that they are most commonly used, are incapable of representing time-dependent changes to the probability of action outcomes in a state. This limitation makes them unsuitable for representing the many problems for which the time-dependence of action outcomes is a critical characteristic.

Semi-Markov models can be approximated by fully Markov model using the equivalent rates method, where a semi-Markov state is replaced by a Markov state with a mean time-in-state that matches the mean of the original state's distribution (Sing, 1980). A more accurate technique is to replace the semi-Markov state by a series of Markov states corresponding to a distribution that approximates the original state's distribution. A *phase-type distribution*, a distribution over the time to absorption in a Markov chain, can approximate any positive-valued distribution. The accuracy of the approximation is dependent on the number of states making up the distribution’s Markov chain, so an equivalent model may require adding a very large number of additional states (Limnios and Oprisan, 2001).

CHAPTER 3

Related Work

3.1. Human-Robot Social Interaction

THE majority of current research in social robotics is more concerned with the physical design and the development of lower-level behaviors that support face-to-face human-robot interaction than with high-level planning for social interaction (Fong et al., 2003). However, there is a great deal of current work that focuses specifically on developing behavior for interaction using a wide variety of approaches. Social guidelines are built into interaction behaviors at the level of constraints on low-level action and on the production of high-level plans. Machine learning algorithms are employed to acquire socially-aware models of tasks directly from the people the robot interacts with. POMDPs are used to model the inherent uncertainty and ambiguity present in social interaction. Different representations of time and its impact on social interaction are also explored.

3.1.1. Navigation and Social Interaction

Much work on social norms in human-robot interaction focuses on lower-level tasks such as navigation and combine navigation planning with additional social constraints to produce socially appropriate behavior. Much of this work relies on Hall's concepts of personal space and its impact on interaction. In one such application, socially appropriate line-standing behavior is implemented on a mobile robot using a vision system that reason's about personal space to detect lines of people and find the end of the line for the robot to join (Nakauchi and Simmons, 2002). Constraints based on social norms for maintaining appropriate distances and yielding to the right were also built into the goal-directed navigation system for a mobile robot to design a low-level motion controller for navigating populated hallways (Christensen and Pacchierotti, 2005). A study on socially appropriate following behaviors for a mobile robot compared different methods for choosing the bearing for a robot that followed behind a human while staying just outside of their zone of personal space (Gockley et al., 2007). These approaches are similar to the work in this thesis in that they address social tasks that are not purely collaborative and in which the robot has its own goals distinct from those of the humans they are interacting with. This work concentrates on accurately modeling specific tasks based on their social spatial characteristics rather than attempting to develop a general framework for describing social interaction capable of describing social tasks that fall outside the realm of Hall's theories about face-to-face interaction (such as social interactions that occur while operating vehicles). Additionally, these approaches make assumptions about human intention based on available information rather than reasoning explicitly about their ambiguity.

3.1.2. Learning for Social Interaction

As opposed to approaches such as the one in this thesis where models of human social behavior are encoded by experts, there is also work on human-robot interaction that attempts to learn behaviors directly through interaction with people. Much work on learning for social interaction focuses on leveraging the analogy of human learning, with the human acting as a teacher to a robot which is designed to appear young and likable to elicit a caregiver response from the human. The robot Kismet is one of the earliest examples of this approach. While Kismet did not engage in learning, its design was motivated by the idea of creating a social structure for learning by engaging people in parent-infant interaction with the robot (Breazeal and Scassellati, 2000). Kismet's behavior was governed by low-level drives that caused the robot to perform actions to engage the human in interaction but avoid overstimulation. Later work by Breazeal with the robot Leonardo used verbal and nonverbal social cues from humans to guide task learning for a button-pushing task (Lockerd and Breazeal, 2004). The robot also communicates its belief about the task state back to the human teacher with nonverbal social signals. More recent work incorporates spatial cues from the way the teacher positions themself in the space and manipulates objects during a blocks play task (Breazeal and Berlin, 2008). While this work makes use of nonverbal social aspects of the teaching interaction to provide feedback to the robot, the tasks being taught are not social tasks. This approach also relies heavily on the teacher-student analogy, and would be difficult to apply to tasks where the analogy does not hold. In many social interaction tasks, people do not have experience teaching the interaction to others while performing the interaction itself. Many social behaviors are more commonly either explained at some time before the interaction is engaged in (in the case of formal etiquette or local social conventions that must be explained to outsiders) or picked up by observing people interacting (in the case of simple interactions or subtle ”unspoken rules”).

Learning by imitation typically concerns an agent learning a sequence of actions to perform a task by observing another agent performing it. Purely imitation-based techniques are difficult to apply to interaction, because if the person being interacted with exhibits actions that were not observed during training, the agent has no model of how to respond. Researchers have made the case that the behaviors to be imitated should be considered in their social context in order to determine what aspects of the behavior should be imitated and when these imitated behaviors may be applied (Dautenhahn and Nehaniv, 2002). Such context can provide a link between the actions that make up the task and the goals and intentions that motivate them. Demeris relates work on intent recognition in robotics and related fields to cognitive theories about how people recognize intention when observing behavior to imitate (Demiris, 2007). These intent recognition techniques require agents to have a model of human behavior (which may be learned or designed) relating the observable actions to goals or intentions. The modeling approach taken in this thesis allows the robot to reason about and recognize the human’s intention while acting. Rather than relying entirely on (possibly sparse) human data to model the interaction task, the available data is combined with a prior model that describes the relationship between the human’s actions and their intentions.

Reinforcement learning (RL) techniques, which can create policies without prior models, have been applied to social domains. In a collaborative robotic foraging task, robots were able to learn social rules to coordinate their behavior by learning locally while receiving reward both for goals they achieved themselves and for goals they observed teammates achieving (Mataric, 1997). Agents have also been trained to interact with people in social situations by receiving reward directly from human users. Cobot, a reinforcement learning-based chatbot, was trained to perform amusing actions through interaction with lambdaMOO users over a period of months (Isbell et al., 2001). The agent learned separate models for individuals and based its action choices on who was present in the channel, personalizing its behavior. More recent research on the use of reinforcement learning for social robotics has explored the way that people understand and try to use reward in RL tasks. In an experiment where people taught an agent in a computer game to make a cake, Thomaz observed that people were biased towards giving positive reward and that they attempted to associate rewards with objects of interest in the environment in order to guide the agent’s actions (Thomaz et al., 2006). While these results are important to the approach of using RL as a social learning interface, the human’s interaction with the agent in this case was limited to teaching the task rather than being a participant in it. In most of these applications, the tasks learned were not social interaction tasks with people, either because the robots interacted only with other robots or because a non-interactive task was learned using reinforcement provided by a human. Cobot is an exception to this, but the agent in this case functioned as a social "toy" for the users of the MOO rather than as a peer. Both the difficulty of practically applying RL to learn even relatively simple non-interactive behaviors for mobile robots (Smart and Kaelbling, 2002) and the way in which human user's intuitive understanding of how reward should be allocated differs from the way RL algorithms expect reward to be assigned suggest that there are great challenges to using RL techniques to learn full social interaction tasks with people. Humans make use of mental models of behavior to understand social interactions and are able to explain behavior in terms of action and intention. This thesis takes advantage of this fact to create models that can be used to perform social interaction even in the absence of training data. If task performance data is available, it can be combined with the prior model. But because the model describes behavior that may not be observed in the training data, there is no requirement that large amounts of data must be collected to cover the space of possible interactions, as would be the case for RL techniques.

3.1.3. Planning for Social Interaction

There is work on planning for human-robot interaction that specifically deals with the requirements of planning for social situations. One such approach is an architecture for the tour guide robot Rackham based on high-level symbolic planning to support human-robot interaction for collaborative goal achievement (Alami et al., 2005). The human and robot are conceptualized as a team with a common goal, and human-aware task level planning for social interaction is motivated by the need to ensure that the plans arrived at are acceptable and understandable to people. In this system, socially acceptable behavior is defined as a set of constraints on undesirable states or action sequences.

POMDPs have also been applied to a number of planning tasks involving the understanding of human behavior. Hoey et. al. developed a POMDP-based agent to assist elderly people with dementia with a handwashing task by observing their behavior and offering advice when they seemed to experience confusion about what to do next (Hoey et al., 2007). Fern, Judah, and Tadepalli use a method that is somewhat similar to the one described in this thesis to model tasks for an assistive agent in the sense that the goals of the human user are included in the model as hidden state(Fern et al., 2007). Action selection for the assistive agent is accomplished using a heuristic value based on an MDP formulation of the model. In both of these systems, the tasks are purely assistive and the agent's goals are the goals of the user, rather than having independent goals. While these applications demonstrate the strength of the POMDP model's ability to relate human action to high-level goals in a way that allows an agent to reason about a person's intention, they do not address how to model situations in which an agent has its own goals to achieve during the course of an interaction. POMDPs have also been used to plan for agents that have their own goals distinct from those of the other agents or humans they interact with. Pearl, the nursebot, used a POMDP planner to perform everyday tasks and offer information to residents in a nursing home (Pineau et al., 2003). While Pearl acted independently, interactions with residents were primarily conversational and assistive in nature, with the POMDP planner being used to disambiguate users' speech acts so the robot could respond appropriately. In all of these systems, the emphasis was on producing assistive behavior rather than behavior that was socially appropriate, so the modeling techniques used to do address how to create an agent that is an effective social actor.

Models inspired by POMDPs' ability to represent unobservable mental states in people and reason based on beliefs have been used to model social behaviors in agents. The PsychSim software uses a multi-agent MDP planner with models inspired by POMDP planning (in the sense that beliefs about unobservable aspects of the state are modeled) to simulate bullying behaviors in a classroom setting (Pynadath and Marsella, 2005). Each agent's beliefs include a simplified model of the beliefs of the other agents. In this system, the agents' choose actions by bounded look-ahead in which the other agents' behaviors are produced by sets reactive rules determined by an agent's "stereotypical" belief about the characteristics of other the agents. Each agent can update its beliefs about other agents based on their actions, but they do not reason about uncertainty in their beliefs when looking ahead to determine immediate actions. Thespian is a system for interactive story-telling based on PsychSim. Conversational social norms are enforced in Thespian both by explicitly representing social obligations (such as responding immediately to a greeting or a question) in the agents' state and by adjusting the weights of various goals to encourage or discourage certain behaviors for a particular agent (Si et al., 2006). While this method is useful for producing socially appropriate behavior over short-term interactions (such as at the level of pieces of conversation or the choice of individual immediate actions), it is unlikely to be effective for producing appropriate interaction in domains where the rewards are delayed until late in the course of an interaction. The use of finite-horizon lookahead is justified by people's bounded rationality in decision-making, but the type of simplifying assumptions that people use when making rational decisions is likely not well modeled by a fixed finite-horizon lookahead. This system also does not take into account uncertainty in the agents' beliefs when selecting actions.

3.1.4. Time and Social Interaction

Researchers, recognizing the impact of time on social interaction, have addressed modeling the effects of time explicitly in planning, learning, and producing reactive behaviors for coordinating interaction. Work has been conducted in the field of multi-agent planning to address the temporal issues faced by certain social interactions which take place in dynamic environments (Allouche et al., 2000). In a domain where a group of agents collaboratively monitor a fleet of buses, the way task dependencies among agents change over time during an interaction is modeled, allowing the agents to coordinate their behavior during this cooperative task. Like the work in this thesis, this system takes into account the fact that many social tasks are carried out in a changing environment, though the social actors performing the task are all artificial agents rather than humans.

Other work with robots that interact with people investigates the role of time in how people recognize and respond to action in a social context. Gold and Scassellati model the temporal aspects of social interaction at the level of action and response in face-to-face interaction. A humanoid robot uses a simple learning algorithm to learn the window of time over which an action's effects may occur and uses this timing information to reason about the causality of events it observes. This learning algorithm is applied both to the effects of the robot's own actions and the social responses of a human interacting with the robot (Gold and Scassellati, 2006). The designers of the Keepon robot focus on a different aspect of the relevance of time to human-robot social interaction. Rather than looking at time in terms of a chronological sequence of events, they focus on the repetitive, rhythmic aspect of face-to-face interaction. In an dance-oriented play interaction with children, the robot matches its dancing behaviors to the rhythm of the children's movements (as measured by sensor data) as well as the rhythm of music (Michalowski and Kozima, 2007). These approaches focus on the role of time in low-level, reactive social behaviors that establish a rapport with humans rather than its role in modeling and reasoning about goal-oriented social tasks.

3.2. Decision-Theoretic Models and Representations of Time

While one of the strengths of the MDP as a planning model is the reduction in solution complexity afforded by the Markov property, there are some problems that cannot be well-modeled without including information about the time spent in states or time-dependent action effects. Various methods have been proposed to add time information to a model while still keeping the representation computationally tractable. These modifications often take the form of extending certain aspects of the representation to allow for more accurate representation of time's effect on transitions from individual states (such as is done in the SMDP model) or of adding the needed information about time directly to the state space, resulting in an augmented model that is still fully Markov.

An augmented state space was used to develop an algorithm for maximizing probability of winning (rather than reward) for finite horizon MDPs in a robotic soccer application (McMillen and Veloso, 2007). The MDP's state space is expanded by adding the time remaining and intermediate reward acquired to the state and the resulting model is solved. In order to manage the increase in the size of the state space, they use heuristic compression methods that compress the time horizon according to a fixed-criteria (such as compressing uniformly or compressing the early timesteps). While the policies obtained by this method are time-dependent, the policy of the opponent and the environment described in the model are not. Additionally, the state compression method used, unlikely the method proposed in this thesis, combines states across timesteps using a fixed time-based criteria rather than a method that takes into account the potential differences in these states' values. A discrete MDP representation that includes a continuous time dimension has also been explored for the purpose of planning trips that include the use of various forms of public transportation (Boyan and Littman, 2000). This allows for the accurate representation of time-varying state transitions and rewards as well as extended action durations, producing a model with more representational power than the typical SMDP formulation. This representation proved faster to solve than a model that discretized the time dimension. In this approach, reward functions are restricted to a piece-wise linear representation for the sake of computational efficiency. The continuous time representation used in this method makes it impossible to use for general POMDP models, which have a discrete state space.

Semi-Markov models are starting to be used in activity recognition applications, as researchers recognize that more accurate representation of time allows for better representation of human action. Switching hidden semi-Markov models have been used for recognition of everyday household activities (Duong et al., 2005). The performance of the exact model is compared with an approximate version that uses the discrete Coxian distribution (a kind of phase type distribution) to approximate the semi-Markov state distribution. The approximate models were found to perform well while offering a considerable computational advantage. These models are capable of representing temporally extended actions, but not time-dependent action outcomes. Non-stationary hidden semi-Markov models, hidden semi-Markov models with time-dependent state transition probabilities, have also recently been used for activity recognition, showing that recognition performance can be improved by using models that take into account the dependence of action outcomes on the time in the state as well as the overall duration of time spent (Marhasev et al., 2006). These systems demonstrate the how more accurate time representations can improve the recognition of human behavior, but do not address the issue of how an agent might act in response to these behaviors.

Semi-Markov models are useful for representing time-dependence in the environment model, but it is often preferable to transform a semi-Markov model into a fully Markov model for computational reasons. Approximate solutions for generalized semi-Markov decision process models can be found by replacing each semi-Markov state with a phase type distribution, a series of fully Markov states that approximate a continuous distribution, and then solving the resulting continuous-time Markov decision process (Younes and Simmons, 2004). This work has only been applied to fully observable planning problems. Semi-Markov states have also been used as a modification to POMDPs to represent temporally extended macro-actions in a robot navigation task (Theocharous and Kaelbling, 2004). These macro actions are used as a form of hierarchical abstraction, and a custom, grid-based method is used to solve the model. The resulting model is more compact than the equivalent POMDP model, but can not be solved using general POMDP algorithms. This method also assumes that the environment the agent is acting in is static, with the agent's own action outcomes being the only time-dependent aspect of the model. In contrast, this thesis uses a time-indexed state representation that is capable of modeling time-dependent action outcomes both for the agent and its environment.

3.3. State Aggregation for Markov Decision Processes

The state aggregration method presented in this thesis is similar in concept to existing approaches to state aggregation for MDPs, which compare reward-based state values and next state distributions in order to decide which states should be merged. Dean and Givan present a general technique for doing state aggregation in MDPs based on a model minimization algorithm for finite automata and explain how this method is closely related to several other state abstraction algorithms (Dean and Givan, 1997). Stochastic dynamic programming dynamically aggregates states according to an estimate of the value function (Boutilier et al., 2000). In a method relevant for applications where goals are associated with physical locations, rectangular regions in a spatial state space are aggregated based on values propagated back from a goal-based reward function (Dietterich and Flann, 1995). Li and colleagues provide an overview and analysis of various approaches to state abstraction in fully-observable Markov decision processes (Li et al., 2006).

There are also a number of state aggregation methods for POMDP planning, many of which focus on hierarchical decomposition of the state space. The PolCA+ algorithm performs state abstraction on a POMDP given a human-designed task hierarchy, first fixing a policy for the lowest levels of the task hierarchy before finding abstract representations for the higher levels (Pineau et al., 2003b). Temporally extended actions, represented as semi-Markov states, have also been used as a form of hierarchical abstraction in POMDP (Hansen and Zhou, 2003; Mahadevan et al., 2003; Theocharous and Kaelbling, 2004). One drawback to using abstraction methods that create semi-Markov models is that a POMDP with semi-Markov states cannot be solved using general POMDP solution techniques. All of these techniques also assume that the agent's environment is stationary, so they do not address the issue of how to do aggregation on models of a time-dependent dynamic environment.

Another way of doing state abstraction is to find a smaller set of bases to represent or approximately represent the state space. Roy's Exponential PCA algorithm does nonlinear compression in the belief state by projecting the reachable part of the belief state onto a low dimensional manifold (Roy and Gordon, 2002). The projected model is represented and solved as a belief state MDP. The VDC algorithm, in addition to using abstract decision diagrams or decision trees for factored representations of the state space and reward, also uses Krylov functions to represent an exact or approximate linear decomposition of the state space (Poupart and Boutilier, 2004). The approximate models achieved by limiting the number of Krylov bases are more accurate over a short planning horizon, so this technique may prove ineffective in domains where reward is delayed (as in the tasks addressed in this thesis).

Other state abstraction techniques attempt to identify and abstract away states or state variables that do not have an impact of the achievable reward. The CIVA algorithm compresses a POMDP with a factored representation by removing conditionally irrelevant state variables from the model, producing a policy equivalent model with a flat representation that is exponentially smaller than the original (Smith et al., 2007). Conditional relevancy is determined by finding the values of observable state variables that are reachable according to the model, and then identifying which other state variables either do not effect reward, observation, and state transitions or can have their value reconstructed in later relevant states using other state information. This method assumes that a subset of the state variables are observable and have deterministic transitions, which may not hold in the general case. Very preliminary work has been done on finding smaller abstract POMDP representations given an an original model using a method inspired by MDP model minimization (Wolfe, 2006). Predictive State Representation (PSR)-style tests are used to identify state partitions that have observations whose differences are irrelevant to predicting future states. These methods of state abstraction are similar to the one proposed in this thesis in that they rely inspecting reachable portions of the state space and identifying and abstracting away state information which will not effect the future reward that can be achieved.

These general aggregation techniques are useful in cases where the source of potentially irrelevant state information in the model is unknown. Many of these approaches rely on a factored state space to reduce the size of the state representation. Factorization is of limited use for time-dependent representations, as the relationships between state variables may change over time, requiring multiple different factored models. Instead, this thesis uses a flat, time-indexed state space to represent time dependence. The large size of the models is due to the addition of the time index to the state. It is very likely that many of these time-states are redundant or irrelevant to finding a high quality policy for action. Rather than a general abstraction technique which may not be effective for this type of problem representation, a novel state abstraction technique is developed that operates exclusively on the time dimension of the state space and is able to take advantage of the time-indexed state space's special structure.

CHAPTER 4

Technical Approach

THIS thesis is concerned with the problem of how to create appropriate policies for action for robots performing *situated social tasks*. This class of tasks, as mentioned previously, is characterized by: goals based in the physical environment, the need to coordinate behavior with other social actors, and the existence of familiar protocols for goal achievement that are known to all participants. The approach proposed for creating robots that perform these tasks is to model the interactions as time-indexed POMDPs in which the intention of the human is part of the hidden state. In order to make the time required to find solutions to these potentially large and complex models tractable, a reward-based method for aggregating states in the time dimension of the state space was designed. These models are then solved to create policies for action. The resulting policies are evaluated by using them as controllers for a robot that performs an interaction task with human subjects.

4.1. Planning for Social Interaction

For tasks that humans perform regularly and it is possible to collect data of humans performing, why plan at all? Should it not be possible for robots to learn their policies for action entirely from observing human data? If the domain were one in which a person performs a narrowly definable task in which the rest of the environment is not highly variable, this is a feasible approach. But for situations in which people interact with other people, it is less so. Human behavior may have many subtle differences from one individual to the next, even when performing the same task. Unless the quantity of data available is virtually unlimited, it is likely that an robot will encounter a human who is performing the task in a recognizably socially acceptable manner that is still not exactly the same as any of the data examples it has witnessed. A common way of responding to this difficulty in data-based methods is to choose an action that corresponds to the ”nearest” state to the one currently being observed from the data set. But for complex state spaces, coming up with a sensible definition of what constitutes ”near” that leads to reasonable behavior can be difficult. For realistic social interaction tasks, the complexity of human behavior makes the use of solely data-based methods prohibitively expensive.

This work takes a different approach that uses domain knowledge about social tasks to produce a complete model of the interaction. This model contains all of the actions that a person performing the task might choose, as well as all possible changes in the environment that may occur. It also provides a probability distribution over these action outcomes for all of the robot's possible actions. The relationship between the visible actions of the other robot and their unobservable intentions are encoded in the model as hidden state. The model's reward structure assigns values to states based on their desirability as goals, given the robot's intentions, and also penalizes states that correspond to a violation of social guidelines. The accuracy of this model is improved for the most frequently visited parts of the state space by updating the probability distributions using data collected from humans performing the task. In this way, a model is created that POMDP solution techniques can solve for a policy that takes into account the uncertainty about a person's intentions when selecting actions for the robot.

One might wonder if a simpler approach could be employed, in which the problem of inferring a person's intention and of planning to act given that intention can be separated. In situations where the best course of action is partially dependent on another's intentions, it may be tempting to think of the problem as that of observing the other person until you determine what their intention is, and then choosing the correct response. But this approach separates action from reasoning about beliefs in a way that allows for no possibility of recovery if the decision made about the other person's intentions turns out to be incorrect. Also, during ambiguous interactions, the time at which such a system is confident enough to respond may be delayed beyond the point of acceptability. For a policy of action to be robust, it must constantly make use of the best current belief about the intentions of the other person. In order to evaluate the performance of planning over this uncertainty rather than choosing actions based on a fixed belief of the human's intentions, policies from the POMDP models are compared to policies from MDP models that assume an intention for the human being interacted with in human subject experiments for the Pittsburgh left driving simulation task.

4.2. Modeling Social Interactions

4.2.1. Terminology

In order to discuss the approach taken to modeling social interaction clearly, it is useful to define a few terms. A *goal* can be thought of as a desired outcome, event, or ordering over a series of events. A person or robot's *role* in an interaction is a recognized grouping of behaviors associated with a common goal or set of similar goals. An robot's *intention* is a preference ordering over the possible goals for the role they are in.

Intention, for the purpose of this thesis, is defined as fixed over the course of an interaction. Using the example of turning left in Pittsburgh, imagine that you are a Pittsburgh-left-hating driver making a left turn and you intend to wait until oncoming traffic has passed before you turn. To your dismay, when the light changes, the oncoming car just sits there. When you fail to start turning, the driver flashes their lights at you. In order to avoid holding up traffic, you reluctantly make the turn. Despite the fact that you ended up making a Pittsburgh left, your preference for not making one never changed. You just chose the best course of action available to you, given your beliefs about how the interaction would play out, based on your perception of the other driver's intention. Your most preferred goal given your intention (to yield to the oncoming driver), was unlikely to be achievable because the other driver would not cooperate. Instead, you chose to act to obtain the less preferred goal of taking the left turn first and leaving time for the oncoming driver to go through the intersection before the light turned. It is also possible that you could have waited longer to see if the other driver would give up waiting and drive through. But this increases the likelihood that only the other driver (or only you, if you finally decided to turn late) would make it through the intersection in time, both of which are less socially desirable outcomes than ones that involve both drivers crossing through.

An alternate way of thinking about intention would be to imagine a person's intention changes over the course of the interaction as they decide to pursue different goals based on the circumstances. In this modeling approach, a person's intention would essentially be the same as the goal they are currently pursuing, and the modeling power of having a higher-level explanation for behavior that organizes the conditions under which these different goals are pursued is lost. When people change their goals over the course of an interaction, they do not explain this change after the fact by claiming that what they ended up doing was what they most preferred to do. Instead, they explain that they chose to pursue a less desirable goal that they expected to be able to achieve considering the intentions of the person with whom they were interacting. Treating intention as a fixed quantity over the course of an interaction has an additional benefit when it comes to modeling social tasks. Intention is an internal mental state that is not directly observable. If it changed constantly over the course of an interaction, there would be no way to measure it accurately to describe the relationship between intention and action for task performance data. But because a person's intention remains the same over the course of the interaction, their intention can be determined before or after an episode of interaction to be associated with their observed actions.

4.2.2. Building the Prior Models

The interplay between the intentions of all the participants is of central importance to the way that an interaction plays out. While this modeling method can be applied to interactions between a robot and multiple people, most of the discussion here will assume an interaction between a robot and a single person for the sake of simplicity. To create policies for a robot performing a given role in a task, a separate model is created for each of its possible intentions in that role. Each of these models contains all of the possible intentions for the person being interacted with as hidden state. A rule-based prior model is created of the person's actions and the changes in the environment that relates the observable action effects to the unobservable intentions. The robot's own intentions are encoded as a separate reward structure for each model that represent the desirability of various states of the world. Additional reward structure that penalizes actions that are socially undesirable is common to all of the robot's POMDP models. Data examples from human-human interactions where the person in the robot's role matches the model's intention are used to update each model's state transitions. By basing the model and reward structure on intentions, complete models are created that allow a robot to choose reasonable, goal-directed actions in response to a wide range of human behavior.

An alternate way of modeling social interactions would be to create a model in which the action space is made up of the joint actions of both the human and the robot. In a model created in this manner, policies would be found that specify the actions of both participants in the interaction. During execution, the actions for the role played by the robot could be selected from the policy's joint actions. Such a modeling approach would yield a policy for an interaction that could be used to act for any role that an agent engaging in the interaction could take on, as opposed to the approach taken in this thesis which produces a policy for only one role from each model. However, it would be be far more computationally expensive to solve a joint action model of these types of interactions, which we will see are already very expensive to solve for realistic problems even when choosing actions for only a single agent. Also, because the joint action space is far less constrained, it would be much harder on a practical level to design models that produce socially appropriate behavior. In the single-agent-action modeling approach, the socially appropriate behavior for the role played by the human agent is built directly into the model. The model is then solved to produce a complete policy of action that attempts to achieve the goals the model designer specifies for the robot in the way that is the best response to the described policy for human social behavior.

In order to model this human social behavior, a designer may want to make use of multiple sources of information, both explicitly describing people's actions in terms of their effects and motivations and adding data from examples of people performing the interaction. Even in a simulated environment, data collection is expensive and time-consuming. The data collected will cover only a small portion of the space of possible interactions. Fortunately, the regularity of social interactions makes it possible to specify a reasonably accurate prior model. This model is built by combining the outcomes of the robot's possible actions with the effects of the human's behavior, changes in the environment, and any effects caused by interactions between the robot, the human, or the environment.

Rather than modeling a joint action space (because the robot only has control over its own actions), the effects of the human's behavior in response to each of the robot's actions at each timestep are built into that action's next-state transition structure. The robot's actions' effects and state changes caused by the behavior of the human or the environment are expressed as a set of probabilistic rules with state-based preconditions governing the conditions under which they can be applied. The composition of these rules creates a full model of the interaction, giving the actions available to the robot and the distribution of their outcomes based on the expected behavior of the human and the environment.

Rules are expressed in terms of state-value based preconditions, probabilistic effects, and a weight that represents the relative likelihood of an effect. Rules are classified as describing the effects of actions by the *robot*, the *other* person, the *environment*, or the *interactions* of any of the previously named categories. These rules are applied to a state in a cascading manner to produce a probabilistic action effect for each of the robot's actions that includes all the possible responses of the other person and the environment and the probability that they will occur. No distinction is made between observable and unobservable states, so rules can have preconditions that apply to either or both. Rules may express low-level information about actions (such as the probability that moving at a certain speed will cause the robot to travel a certain distance) or high-level information about the relationship between intention and behavior (such as how likely it is that a signaling action will be performed for a given intention). Because time is part of the state space, rules may also be applicable only over a certain range of time.

Because one of the purposes of this work is to evaluate the impact of planning over the uncertainty in intention, MDP models are also created from the rule-based description to be used for comparison. A separate MDP model is created for each combination of robot and human intention. The state space for the MDP model is identical to that of the POMDP models, except that the human's intention may take on only one value. For the domain used for the human subject experiments, POMDP observations are assumed to be aliased rather than noisy (i.e., the state is partially observable, but the part that can be observed can be observed perfectly). The domain was chosen to have this property and the model was set up in this manner to make the most fair comparison possible between the MDP policies and the POMDP policies, and to isolate the impact of specifically modeling intention as the hidden state. If the domain were noisy and the POMDPs modeled that noise with states that were absent from the corresponding MDP model, it would be an almost forgone conclusion that the POMDP policies would perform better.

4.2.3. Intention, Action, and Reward

The reward structure for the model assigns values to possible goals according to their desirability given the robot's intention. Negative reward values are also used to penalize events that violate the social guidelines for the task. This reward structure allows the robot to take reasonable, goal-directed actions in response to any interaction it may encounter, even in cases where the behavior of the human in the interaction was not observed during data collection. This approach hypothesizes that human social behavior is, overall, relatively efficient. Therefore, if the reward structure and model are accurate representations of the problem, the policy that is found by solving them should be relatively similar to human behavior.

The robot's goals and intentions include the outcomes for the person they interact with as well, with the amount of reward allocated for the achievement of the human's goals relative to the robot's determined by the level of ”altruism” associated with a particular intention. Considering this from the standpoint of humans interacting with one another, in the future they will inevitably engage in the same interaction in the role of the person they're currently interacting with. Balancing self-interested and altruistic behavior is one of the reasons that social norms exist, and people act to reinforce these norms for their own long term benefit. In order to act in a socially appropriate manner, a robot's behavior must also reflect this balance between self interest and an interest in what is best for all participants.

It is worth noting that it is possible to think of people sharing the space who aren't directly engaged in the interaction as also being participants, in that the outcome of the interaction may affect their ability to achieve their own goals. One such example is the drivers behind both cars in a Pittsburgh left interaction. People often report that they will decide whether to give another driver the Pittsburgh left based on how many cars are waiting behind them. Reasoning about the benefits to others, beyond the primary actors, is part of how people choose what their intention for an interaction is, but that meta-reasoning is outside the scope of this thesis.

Another type of meta-reasoning that is not represented in this modeling paradigm is belief about the beliefs of other agents. In some game-theoretic modeling approaches, second-order (beliefs about what the other agents believe the acting agent will do) or even higher order self-referential beliefs are represented (Geanakoplos et al., 1989). Because of the class of interactions being modeled in this thesis, this level of meta-reasoning is not necessary to produce appropriate behavior. The strategies that agents engaged in a social interaction use are common knowledge to all of the participants, so the effects of actions on higher order beliefs can be implicitly represented in the model through the representation of the agents' behavior. For example, consider the action of flashing one's lights at a car that may be trying to take the Pittsburgh left in order to indicate a willingness to yield. This action changes the beliefs of the turning car, causing them to become confident that the straight car is yielding. As a result, the turning car becomes more likely to take the turn before the straight car moves to enter the intersection. The important effect of the light flashing action from the standpoint of action selection is that it increases the likelihood that the other car will move to cross the intersection first. As long as this correlation is represented in the action effects for the model, the reasoning process that leads to it need not be explicitly represented. While this implicit representation of the effects of changing higher order beliefs is adequate for a wide variety of social domains, it may fail to produce realistic behavior in domains where agents have differing levels of knowledge about the interaction they are engaging in or where the interaction is very dependent on communicative actions that may have a high rate of failure.

4.2.4. Using Human Data

In order to create an accurate model of an interaction and to gain insight into how people negotiate it with one another, the initial stage of model design includes a data collection experiment. During this experiment, subjects are randomly assigned a role and an intention for each episode of interaction. Subjects are instructed to behave as if the given intention is the one that they have naturally arrived at and to behave as they would during a real interaction if the person with whom they are interacting is not cooperating with their preferred course of action (the given intention of the person with whom they are interacting with is not known by either subject). The subjects then perform an episode of interaction according to their assigned roles and intentions. This process is repeated multiple times with each set of subjects for each possible combination of role and intention to build up a data set of interaction traces.

This instruction to "act as if" serves multiple purposes. First, it should be understood that the reasoning people employ to decide what their intention is before entering an interaction is frequently based on information that is outside of the scope of the interaction itself. Their choice may be based on personal preferences, the role the interaction plays in a larger goal they are pursuing, characteristics of the surrounding environment, or any number of other factors. This unobservable mental process is difficult to model, and often does not provide insight into the way the interaction is actually performed once an intention is chosen. Additionally, having the experimenter specify the conditions of the interaction ensures that a balanced number of combinations of each role and intention are observed for each set of subjects. And finally, it allows each data trace to be associated with an intention, rather than requiring the intention to be inferred from the data after the fact (as it would have to be if subjects chose their own intentions for each trial). Setting up the data collection in this manner reduces the model fitting problem from one of inferring values of hidden state variables to one of counting the occurrences of observed ones. It also creates a model in which the hidden state is known to correspond to an interpretable property that was specified by the designer. Once obtained, this data is added to the prior model using simple frequency counts of state, actions, and next state outcomes.

4.3. Representing Time

4.3.1. Selecting a Representation

The class of situated social tasks this thesis is concerned with also exhibit the property that the next-state distributions for actions are time-dependent, both because of dynamic aspects of the environment in which they take place and because the people the robot interacts with consider the passage of time when deciding on actions (and therefore act according to non-stationary policies).

One common way of representing arbitrary distributions over time spent in a state is with semi-Markov models. However, semi-Markov models have limitations that make them less than ideally suited for use in complex POMDP domains. One is that existing fast approximate POMDP solution algorithms were designed for fully Markov POMDPs and may be difficult or impossible to extend efficiently to semi-Markov models. Even assuming the existence of fast approximate semi-Markov POMDP solvers, semi-Markov models still have modeling limitations. Semi-Markov models model only restricted forms of time dependence. The versions used in common practice model only non-Markov time spent in the state and are not capable of representing time-dependent action outcomes. Semi-Markov models also necessitate making an *a priori* assumption about what kind of distributions best model the form of time dependence exhibited by the problem. Specifying the correct structure for the semi-Markov state of a problem may be difficult for complex problems, and fitting data to the model may be complicated. Given these issues, a simpler representation that maintains a fully Markov property would be preferable.

One very straightforward way of modeling time-dependent action outcomes is to add time to the state space of the problem. For episodic problems with a discrete state space (as is common for most MDP and virtually all commonly used POMDP models), this is as simple as adding a time variable that is an index of the timesteps in an episode. This creates a separate copy of all of the states for each timestep, which may be connected to the states in the following timestep in any way that the problem requires. This representation makes no *a priori* assumptions about the structure of the model, at the cost of greatly increasing the size of the state space.

Unfortunately, this approach makes finding a way of modeling time-dependent action outcomes that produces models of tractable size difficult. Many current approaches to MDP and POMDP modeling use factored state representations to reduce the size of the state space. But this is of limited use for a time-indexed model, because it may be necessary to represent a separate copy of the factored representation for each timestep in an episode of the problem. It is also possible that the independence relationships between variables could differ over time, which would also need to be specified in the structure of the model. In light of these issues, factored representations are not very useful in this case.

An insight that can reduce the size of the state space somewhat is that not all states may be reachable at every timestep of the model, and it is therefore unnecessary to represent them at that timestep. The rule-based method used to construct the prior models takes this into account. However, the reduction in the number of states from representing only the reachable states at each timestep will usually be small relative to the increase in the state space caused by including the time index.

4.3.2. Time-State Aggregation

The number of time-dependent states could be reduced by having certain ranges of time use the same state transition probabilities, but these ranges would have to be specified *a priori*. This creates the possibility that the pre-defined time-dependent structure of the model could end up preventing a high quality policy from being found if the structure is a poor match to the actual time-dependence in the problem (similar to the problem of choosing a correct semi-Markov structure). Using a pre-defined method to combine time-states, such as every k timesteps, or only combining states for the early timesteps, it is possible to combine states where the same action in each state may achieve very different reward values.

Ideally, the model should represent the time-dependent structure that is necessary to create a good policy for action. Representing time-dependent differences in next-state distributions is only useful for planning if it helps achieve a higher level of reward than if these differences were not represented. Acting on this intuition, a method for reward-based state aggregation of time-indexed POMDPs was developed. Exact state aggregations methods for MDPs only combine states when their immediate rewards and state transitions are all equivalent (Dean and Givan, 1997; Li et al., 2006), though this criterion can miss opportunities to combine states that may not have an effect on the optimal policy. More aggressive methods for state aggregation in MDPs combine states according to their expected value under a fixed MDP policy (Boutilier et al., 2000). However, this is impossible in the POMDP case, as policies for action are expressed in terms of beliefs rather than states. Treating the model as a MDP and using a policy that only ever selects one action in each state (as is the case for MDP policies) to calculate the expected state value would find values that fail to represent the uncertainty of acting in a partially observable state space. There is no reason to assume that the action chosen by an optimal POMDP policy when acting in a state during execution would correspond to the optimal MDP action, and many different actions could be chosen in a state by the same POMDP policy based to the current belief state. Instead, we calculate state values using a conservative measure of their future reachable reward based on the performance of a given policy. The policy used for the example domains tested in this thesis was a random policy which selects all actions with equal likelihood. This policy choice results in state values that depend on all of the rewards that are reachable by taking actions starting from a state given the model's transition structure.

Once the state values have been calculated, states that are identical except for their time index are combined across time steps when their values do not differ more than a predetermined threshold. This state aggregation reduces the size of the state space (and therefore the planning time). Adjusting the threshold for determining whether states should be combined varies the amount of state-aggregation that occurs, allowing the trade-off between planning time and model accuracy to be varied according to the user's needs. The performance of the policies for a version of the POMDP model without a time index and a fully observable MDP version of the time-indexed POMDP model can be used to help determine what amount of aggregation gives the best solution time improvement while still providing acceptable performance.

Two different methods of calculating state values were explored. In one method, a distribution over possible future rewards for the given policy was calculated for each state, and states were compared using the Kullback-Liebler divergence of their distributions. For a problem where the reward structure corresponds to a discrete set of goals, this is simple to represent, but it becomes complicated to apply to more complex reward structures. In this case, the expected reward for each state may be used instead. This method is less expressive in terms of capturing possible differences in next-state structure, but simpler to represent than a full distribution.

The time-state aggregated models were solved using approximate POMDP solution algorithms. Both value representations were evaluated for a variety of models from the elevator riding domain. The expected reward based value representation was used for additional experiments with larger, more complex models from the Pittsburgh left domain to further explore the relationship between state-aggregation threshold, policy solution time, and policy quality. In both domains, time-state aggregation was found to be capable of producing models that were significantly faster to solve while producing policies that performed close to optimally (or what is believed would be optimally in the case where the original model cannot be solved in a reasonable timeframe) when executed on the fully time-indexed model in simulation. Time-state aggregation was also used for the time-dependent POMDP models whose policies were evaluated in the human subject experiments.

4.4. Evaluation with Human Subjects

In order to evaluate this thesis's hypothesis that reasoning about people's intentions and represent time-dependence improves the performance and social acceptability of polices for social tasks, the performance of policies developed by solving time-indexed, state-aggregated POMDPs was compared to the performance of policies developed using less expressive models. POMDP models of the problem were developed from the same rule-based description and data as the time-based POMDPS, the only difference being that they lack a time-index in the state space. Time-indexed MDPs (one for each combination of robot and user intentions) were also created.

A robot car controlled by these three model variants interacted with human users in the role of the turning car in a Pittsburgh left driving simulator experiment similar to the one used for data collection. The policies received observations (or state in the case of the MDP) from the driving simulator and returned their chosen actions at a rate of half a second per update. The robot car used a low-level controller to translate the actions selected by the policies into steering, brake, and throttle commands. The human participant took on the role of the straight driver, driving against each policy variant for a set of trials. After each trial the participant filled out the same survey that was used by subjects from the data collection experiment to report their impressions of performing the interaction with another person.

The rewards obtained by the model variants were analyzed, and statistically significant differences in reward were found between the time-dependent POMDP policies and both other model variants (which achieved less reward). Post tests revealed statistically significant differences between the rewards for the time-dependent POMDP and MDP policies for one combination of intentions, and between the time-dependent POMDP and the non-time-dependent POMDP polices for two combinations of intentions. The causes of these differences were also qualitatively visible in the policies. The MDP policies caused the car to crash into the human driver far more frequently than either other model variant. The non-time-dependent POMDP ran the red light in certain situations, even though this behavior was penalized by the reward structure for all of the models.

The survey responses for each model variant were compared to determine which model variant's behavior was perceived as the most natural and socially appropriate by people. Significant differences were found between the responses for the time-dependent POMDP polices and the other policies for multiple survey statements, with the time-dependent POMDPs judged more positively. Comparison to the responses for human drivers gave additional insight into the overall success of the model variants, with the time-dependent POMDPs getting the same median response as the human subjects for all survey statements (both the other model variants had median responses that were lower for at least two statements). The time-dependent POMDP policies were judged to be more natural and socially appropriate than those of the other model variants by statistically significant differences in the survey responses.

4.5. Example Domains

Of the numerous possible examples discussed while introducing the class of socially situated tasks, two were selected as domains for implementation. These problem domains are described in greater detail below. The design of their corresponding models is explained in Chapter 6 and the details of their state, action, and observation spaces are included in Appendix A.

The models of the elevator domain were tested in simulation only and did not use human data for modeling. For the main problem domain, the Pittsburgh left, task data was obtained from human subjects during a data collection experiment and combined with the rule-based prior model. The policies obtained from these models were later evaluated in an additional experiment through real-time interaction with human subjects.

4.5.1. Elevator Domain

The elevator domain is a simplified version of the task of a mobile robot entering an elevator occupied by human passengers. The state space is made up of the robot's and people's positions, whether each person intends to exit, and the current timestep. The robot receives an observation at every timestep that indicates that the closest person in front of it is moving, that the person is stopped, or that the elevator is empty of people. A person may hesitate for several timesteps before starting to exit the elevator. The robot's action choices are to wait or move forward. The move action is assumed to succeed unless the robot attempts to move into a position currently occupied by a person or into the doorway when the door is closing. If the robot and a person meet in the doorway, they will both be blocked until the door closes and the trial ends. The robot receives a one-time reward based on whether it succeeded in getting on the elevator. Because the robot should not improve its performance at the expense of those it is interacting with, it is also penalized for each person it prevents from exiting. Models of different state size are created by varying the number of people, the period of time that each person may hesitate before moving, the length of time the door stays open, and the resolution of the regions defined over the elevator and lobby. This model will be described in greater detail in Section 6.6.1.

4.5.2. Pittsburgh Left Driving Domain

The Pittsburgh left has been described previously in Section 1.1. The Pittsburgh left problem domain is a driving task from the perspective of the driver attempting to make a left hand turn. The turning car starts several car lengths away from a two-lane, four-way intersection with a traffic light that is initially red. There is an oncoming car opposite it that is the same distance from the intersection that is assumed to be driving straight through the intersection. The two cars are the only actors in the environment. The state is made up of the physical state of both cars (their positions, velocities, body angles, headlight states, and whether a collision has occured), whether the oncoming car intends to allow the Pittsburgh left, the traffic light color, and the current timestep. The physical state of the other car and the traffic light are observable, but the other car's intention is not. The turning car's action choices are to move at one of 3 speeds (stopped, slow, or fast) or turn. Each interaction ends when the traffic light turns red again. This model will be described in greater detail in Section 6.6.2.

CHAPTER 5

The Pittsburgh Left Driving Simulation Task

PRIOR to discussing the method used to construct POMDP models for situated social tasks, the Pittsburgh left interaction task is introduced in greater detail. Before the model for this task was constructed, an experiment was conducted where humans performed this interaction with other humans. The purpose of this experiment was to gain greater insight into how humans perform this task, collect task performance data to incorporate into the POMDP model, and investigate people's subjective impressions of the people with whom they engaged in this interaction.

5.1. Experiment Design

The purpose of this experiment was both to gain insight into how people negotiate the Pittsburgh left with one another and to collect data traces that could later be used to create probabilistic graphical models representing this interaction. Because of the difficulty of obtaining quality data from real driving situations, a driving simulator was used. The driving simulator also serves as a common environment in which humans and agents can perform an identical task, with humans providing a baseline to which the agents' performance can later be compared. The Pittsburgh left driving simulator developed was based on TORCS, an open-source driving game (TORCS, 2005). A custom game level was designed to simulate driving in a suburban environment, including a 4-way intersection with functioning traffic lights. The game engine was extended to allow for control of the environment, automation of the experiment, and data collection. Participants controlled their cars in the driving simulator using an off-the-shelf steering wheel and pedal game controller. In addition to steering and braking, buttons on the controller's steering wheel allowed subjects to flash the car's headlights, honk their horn, or operate the turn signal.

In order to build an accurate POMDP model, data had to be collected of people performing the Pittsburgh left. Getting enough data to reliably learn a partially observable

Figure 5.1. Subject using the driving simulator

model is difficult, as the hidden state must typically be estimated from the observations using expectation maximization (EM) or related methods (Bengio and Frasconi, 1996). In this research, the way that the unobserved variable of user intention was defined avoids that issue. The user's intention corresponds to their original goal for the interaction, regardless of what outcome they are eventually able to achieve. Therefore, for the purpose of the experiment, subjects could be instructed to "act as if" they had independently arrived at a certain intention and behave accordingly. This allowed each data trace for each driver to be associated with the intention assigned to them for that trial (and only that intention), so the driver's intention could be treated as a fully observable part of the state when it was later used to create the POMDP model of the interaction.

Allowing drivers to make their own decision about what their intention was to be for each trial could reduce the diversity of the interactions witnessed from each pair of drivers, reducing the coverage of the data. For example, consider a hypothetical series of interactions between two drivers where one driver always intended not to yield (perhaps because they were an aggressive driver) and the other driver always intended to yield to their partner. Only a subset of the possible interactions that either driver may engage in during real world driving situations would be observed. Based on anecdotal evidence collected verbally from local drivers, people typically decide whether they intend to take or allow a Pittsburgh left based on surrounding traffic conditions such as the number of cars behind the turning and the oncoming driver, conditions that are not present in the driving simulator. Understanding and modeling the external conditions that cause drivers to make a decisions about their own intentions in order to induce them to make varied choices is beyond the scope of this thesis.

The basic interaction unit of the experiment, the trial, was comprised of a single episode of both drivers attempting to cross through the intersection during one cycle of a traffic light. Each driver was assigned a role to perform for the trial: turn left, drive straight, or turn right. The right turn role was included to prevent the drivers from automatically assuming that the other car would be taking a left turn because they had been assigned the role of driving straight for a particular trial, possibly creating expectations that could lead to unrealistic interactions. The right turn was always associated with the intention not to yield, as is the case in normal driving situations. Each driver was also, as previously described, assigned one of two possible intentions, to yield or not to yield to the other driver. For the turning car the intention to not yield corresponded to intending to take the Pittsburgh left, while the intention to yield corresponded to intending to take a "regular" left turn after the other car had passed through the intersection.

5.2. Experiment Procedure

Experiment subjects were recruited through advertisements placed on the campuses of Carnegie Mellon University and the University of Pittsburgh. Participants were prescreened in order to ensure that they drove in Pittsburgh on a regular basis and were familiar with the Pittsburgh left. Each participant was paid five dollars upon completion of the experiment. Thirty-two subjects participated in sixteen experiment sessions over the course of three weeks.

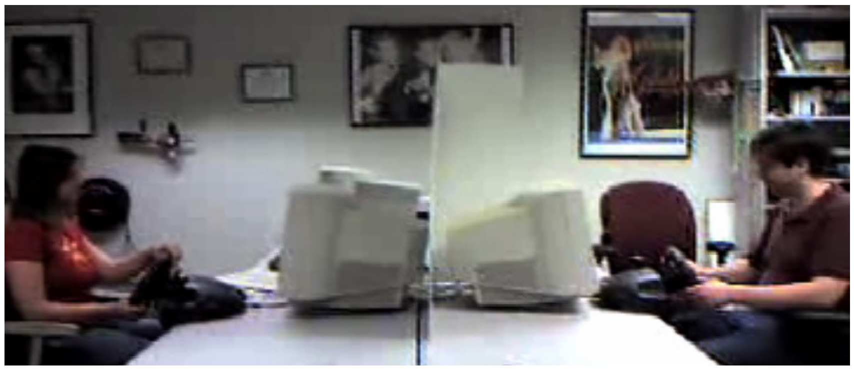


Figure 5.2. The experiment setup with subjects driving

Subjects were seated at two workstations as illustrated in Figure 5.2, facing one another with a divider separating the monitors. This prevented subjects from interacting non-verbally outside of the game (such as by gesturing or making eye contact). Subjects were also asked to refrain from speaking to one another during the course of the experiment. Prior to beginning the experiment, participants were briefed on use of the controllers and the experiment's overall structure. They were informed that it was possible that their assigned intention about yielding might conflict with the other driver's, and that if that seemed to be the case they should behave as they would in a real driving situation. They

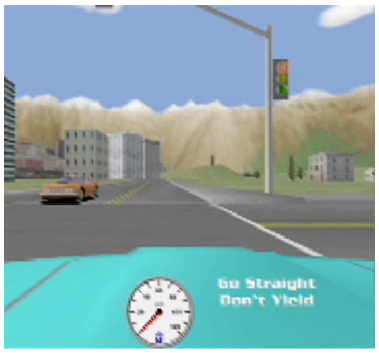


Figure 5.3. Subject POV while using the driving simulator

were also instructed to respect the traffic light and to try to avoid collisions with the other car.

Throughout each trial, a message displayed on the lower right corner of the screen informed each participant of what role they were instructed to take (turning left, going straight, or turning right), as well as whether their intention for that trial should be to yield or not yield to the other car. See Figure 5.3 for an example of the driver point of view within the simulator.

After being briefed, participants performed a set of eight training trials to give them a chance to familiarize themselves with the experiment setup and practice driving. The roles and intentions assigned during the training trials were combinations that would not lead to conflict over the right of way, so that the participants could focus on acquiring driving skills within the simulator rather than on decision making. The experiment itself consisted of four sets of eight trials each, with a one minute rest between sets. Each experiment took approximately thirty-five minutes to complete, including briefing, training, and filling out a paper survey afterwards.

The roles and intentions for both drivers were selected randomly without replacement from a vector of nine possibilities before the start of each trial. This vector contained two instances each of the four possible combinations of driver intentions, once for each player to take the left turn role while the other took on the drive straight role, and one instance where one driver took a right turn role while the other drove straight (in this case both drivers were assigned the intention not to yield as there is no possibility of conflict). This method of assigning roles and intentions assured that particular combinations of role and intention would be encountered in a random order each set. The drivers were guaranteed to experience each possible combination of intentions at least once during a set, and their roles would be reversed if they experienced a combination a second time.

Each trial lasted thirty seconds and began with each subject's car several carlengths from the intersection, with the traffic light on red. During a trial, the traffic light remained red for the first fifteen seconds (enough time for both cars to approach and stop at the intersection), then changed to green for eleven seconds, then to yellow for the final four seconds of the trial. There was a possible delay of up to one second on the light change due to variability in the timing of when events were handled in the driving simulator.

Data for both drivers was recorded at a rate of 20 Hz. The variables logged for each driver were: position(x,y), body angle, front wheel angle, velocity (x,y, angular), acceleration (x,y), and controller values for steering wheel angle, turn signals, lights, horn, brake, and throttle. The environmental variables of current traffic light color and whether a collision between the cars had been detected were also logged. Each log was annotated with the driver's roles and intentions, which were constant for a single trial.

After the driving portion of the experiment, subjects were given a short survey to evaluate their subjective impression of how well they were able to control the car during the experiment, how similar their interaction was to real world driving situations, and how natural and socially appropriate they found the other participant's driving behavior. Four statements were presented, and a Likert-type scale was used to measure the subjects' level of agreement with each statement.

5.3. Results

5.3.1. Outcomes and Events

The outcome of the experiment trials, grouped by the combination of driver intention and role, are shown in Table 5.1. The outcomes examined were the frequency with which each car succeeded in navigating through the intersection before the end of a trial. Because the Pittsburgh left was the driving interaction of interest to this study, trials in which one of the drivers was assigned the role of turning right were discarded from analysis. It can be easily seen from the data that driver intention (and the interactions between the drivers' intentions) had a significant impact on what happened during the course of an interaction. The case corresponding to legal traffic law (Regular Left, No Yield) was the most successful in terms of the percentage of the times that the outcome corresponded to the intention of both drivers. The case corresponding to a cooperative execution of the "law of the land" (Pittsburgh Left, Yield) was also highly successful. From the other two cases, one can infer that social expectations about the Pittsburgh left biased people towards certain outcomes when their intentions were in conflict. In both cases where the drivers did not agree on who would be given the right of way (Pittsburgh Left, No Yield and Regular Left, Yield), the most prevalent outcome was for the turning driver to go first.

Table 5.1. Summary of trial outcomes for Pittsburgh Left data collection experiment

Driving Instruction		Outcome				
Turning Car (T)	Straight Car (S)	T First	S First	T Only	S Only	Neither
Pittsburgh Left	Yield	**92%**	4%	2%	1%	1%
Regular Left	No Yield	2%	**97%**	0%	1%	0%
Pittsburgh Left	No Yield	**59%**	40%	0%	1%	0%
Regular Left	Yield	**51%**	29%	11%	2%	7%

It is also interesting to note that the case in which both drivers were trying to be ”polite” was also the least efficient in terms of both drivers navigating successfully through the intersection before the trial ended. There are two possible interpretations for this. One is that neither participant would deviate from their plan of action in time to make sure that both could clear the intersection. The other is that either one, or both, of the participants were willing to go first despite their intention to yield, but they were unable to successfully communicate this willingness (or recognize it in the other driver), resulting in deadlock. Both cases that involved the straight car yielding resulted in noticeably more outcomes where at least one driver failed to clear the intersection before the light changed. This seems to suggest that the straight car's intention to yield was not always communicated unambiguously to the turning car, leading to uncertainty and failure of coordination.

To get a more clearer picture of how these interactions played out, the frequency of occurrence of significant events were also examined (as shown in Table 5.2). There were certain actions available to both cars that were of use exclusively as signaling behaviors, such as using a turn signal, flashing the headlights, and honking the horn. Of these potentially informative events, neither the turn signal or the horn proved to be useful for understanding the way that drivers negotiated the Pittsburgh left. The turn signal was used by the driver making a left turn in all but six examples (this result may cause readers familiar with driving in Pittsburgh to question how representative this data is of real world interactions). Additionally, prior to the experiment it had been hoped that horn honking would provide a reliable indicator of conflict between drivers and breakdowns in social coordination. Unfortunately, the prevalence of honking during an interaction seemed to be dominated by factors outside of the experiment's realm of study. Subjects who knew each other outside of the experiment and signed up as a pair honked at each other far more frequently than subjects who were strangers that had been paired up by the researcher.

Table 5.2. Summary of event occurrences for Pittsburgh Left data collection experiment

Driving Instruction		Event		
Turning Car (T)	Straight Car (S)	Lights (T)	Lights (S)	Collision
Pittsburgh Left	Yield	10%	**39%**	1%
Regular Left	No Yield	10%	6%	1%
Pittsburgh Left	No Yield	9%	11%	**10%**
Regular Left	Yield	23%	**74%**	0%

The use of the headlights did prove to be informative, however. Looking at Table 5.2, one can see clear differences in the frequency of use of the headlights based on the drivers' intentions. Flashing the headlights was used often by the straight driver to indicate that they were yielding to the turning driver. As can be seen by comparing the frequency of use between the cases (Pittsburgh Left, Yield) and (Regular Left, Yield), the straight car was more likely to flash its lights the longer the turning car hesitated in making the turn. The approximate rate of 10% use that was present for both drivers across all combinations of intentions may have been due to user error caused by the unfamiliar location of the control for the lights. The headlights were operated via a button on the steering wheel placed near the button for the left turn signal and opposite the one for the horn, rather than a lever to the side of the steering wheel. The only time that the use of the headlights by the turning car was above this baseline was in the (Regular Left, Yield) case. While some turning drivers indicated their intention to yield to the straight car in this manner, it was a relatively uncommonly used behavior for this role.

Collision was another event whose occurrence was highly dependent on the combination of both drivers' intentions. Not very surprisingly, collisions occurred most frequently for the (Pittsburgh Left, No Yield) case, when both drivers were trying to go first. The frequency of collisions is undoubtedly high compared to what it would be if these interactions occurred in the real world rather than a driving simulator. This could be due to problems with low-level control of cars in the simulator. But that seems unlikely, given that collisions are still relatively uncommon even for this case. People's self-reports on the survey also did not indicate that they had significant issues controlling the car. Rather, drivers' lack of fear of running into each other would seem to be an unavoidable consequence of performing the experiment in simulation, and it also gives a useful indicator of participants' level of driving aggression.

5.3.2. Survey results

The purpose of the short survey given to participants was to assess their impressions of the experimental setup and the interactions they had with the other driver. The survey consisted of four declarative statements that participants were asked to rate their level of agreement with on a 5 point Likert-type scale. A blank copy of the survey is included in Appendix B. The frequencies with which participants reported levels of agreement with each statement are shown in Figures 5.4 through 5.7.

Interpreting the results of individual Likert scale responses as interval data is controversial, therefore the responses were treated as ordinal and the median was chosen as the statistic for measurement. Ninety-five percent bootstrap confidence intervals were also calculated from the data (this is a non-parametric method that makes no assumptions about the distribution of the response).

The first statement, ”I felt that I was able to control the car in the simulation to do what I wanted it to”, was intended to confirm that participants did not experience significant difficulty driving in the simulator. Anecdotally, the major complaint of participants was that it was difficult to keep the car entirely in the proper lane while turning onto the narrow two-lane roads (this kind of driving imprecision had no effect on task performance). A very small number of participants seemed to have significant difficulty controlling the car. There were no comments offered by participants that indicated why this was the case, but it may have been due to a general lack of familiarity or comfort with computer driving games. The median response was ”somewhat agree” with a 95% bootstrap confidence interval for the median result also covering only the ”somewhat agree” response. This indicates that subjects were fairly satisfied with their ability to control the cars in the simulator, which is also borne out by the previously shown results for trial outcomes, which showed that drivers were highly successful at navigating through the intersection while avoiding collisions overall.

The second statement, ”the actions of the other car seemed natural to me”, was asked to confirm that not only was the simulator sufficient to allow a participant to control a car naturally, but that control translated into behavior that was perceived as natural by the other participant. Once again, participants reported a high level of agreement overall. It is worth noting that the number of low-agreement responses to the first and second question were very similar. This is most likely because the drivers who had difficulty with control of their car exhibited behavior that seemed abnormal to their partner. As for the first

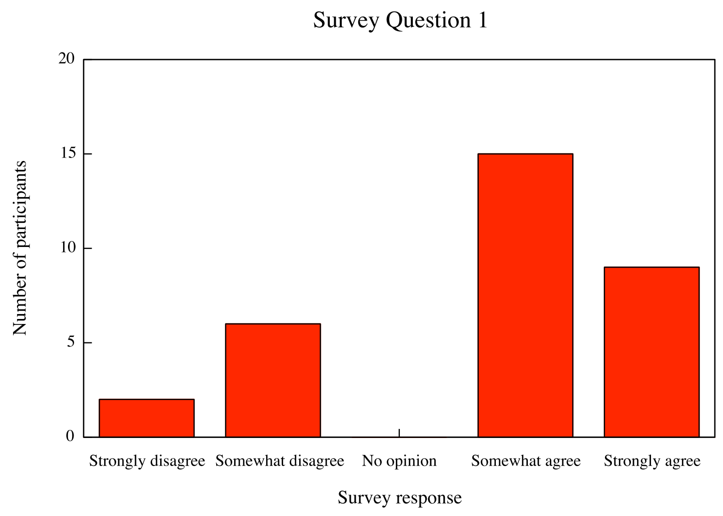


Figure 5.4. Post-experiment survey results for the statement, "I felt that I was able to control the car in the simulation to do what I wanted it to."

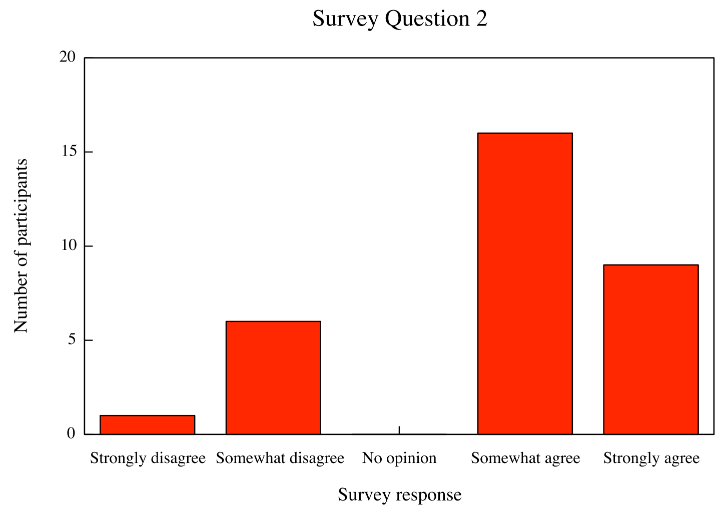


Figure 5.5. Post-experiment survey results for the statement, "The actions of the other car seemed natural to me."

statement, the median response was "somewhat agree", with a 95% bootstrap confidence interval for the median covering just the "somewhat agree" response.

The purpose of the third statement, "The interactions we engaged in were similar to the way I interact with other drivers taking the Pittsburgh left in real life", was to gauge the success of the experiment at capturing the essence of the interaction. Once again, the level of participant agreement was high. In fact, the number of participants who responded with "strongly agree" was greater than for the two previous questions. This seems to indicate

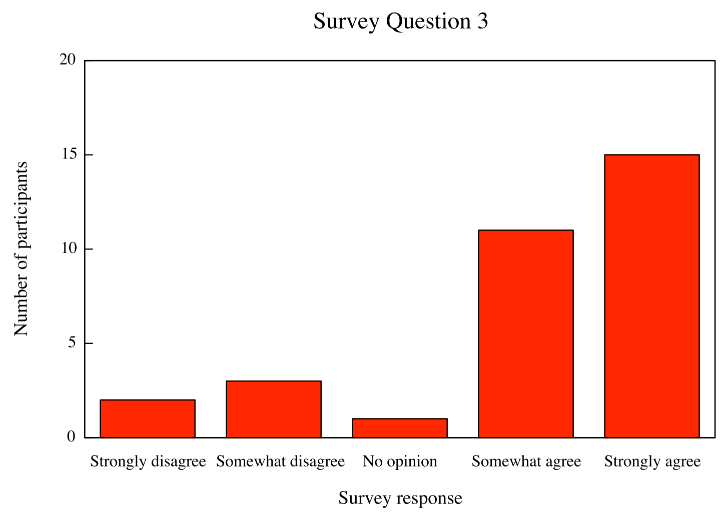


Figure 5.6. Post-experiment survey results for the statement, ”The interactions we engaged in were similar to the way I interact with other drivers taking the Pittsburgh left in real life.”

that minor control issues did not detract from the ability of the participants to interact in ways that felt natural and realistic to them within the simulator. The median response was ”somewhat agree”, with a 95% bootstrap confidence interval for the median result covering the range from ”somewhat agree” to ”strongly agree”.

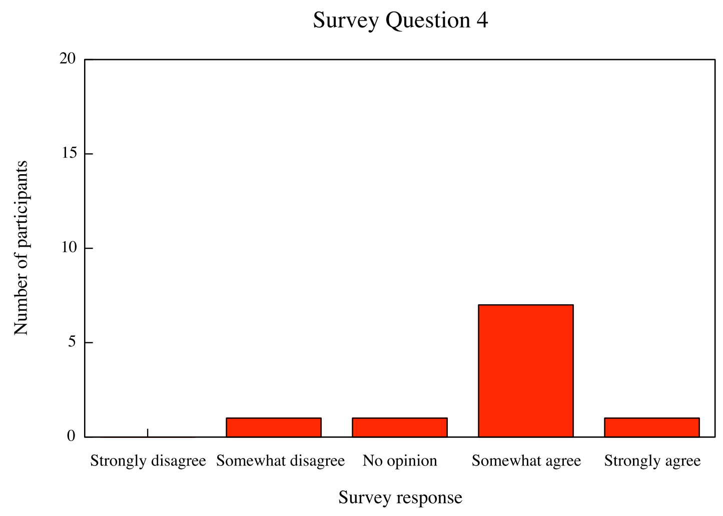


Figure 5.7. Post-experiment survey results for the statement, ”I felt that the other driver followed proper driving etiquette for taking and yielding to the Pittsburgh left.”

The fourth statement, ″I felt that the other driver followed proper driving etiquette for taking and yielding to the Pittsburgh left″, was designed to measure participants' impression of the social appropriateness of the other driver's behavior. This question was added to the survey during the course of the experiment, so results are only available for 10 of the participants. However, the strong peak indicates that a general impression can be inferred even from the small amount of data available. Overall, it seems that subjects had a positive impression of the way that the other driver negotiated the Pittsburgh left, but did not find them to be close to ″perfect″ according to their internally held standards for how it should be conducted. The median response was ″somewhat agree″, with a 95% bootstrap confidence interval for the median covering the range from somewhere just between ″no opinion″ and ″somewhat agree″ to ″somewhat agree″.

5.4. Conclusion

The interaction outcomes and survey results for this task confirm that people found this simulated driving interaction to be a reasonable approximation of the true Pittsburgh left driving interactions with which they were familiar. The results collected provide a useful baseline for behavior that will be compared to the performance of a robot car engaging in the same interaction with human drivers in Chapter 8. Data traces collected during the experiment interactions will also be added to the models of the task. Now that the specifics of how this particular interaction task is performed by humans have been discussed, the next chapter addresses how to model this task and others like it as a time-indexed POMDP, using a general framework that produces socially appropriate policies for interaction for a robot.

CHAPTER 6

Model Design

IN model-based planning, a good representation of the problem is critical to the creation of effective policies for action. In complex domains, acquiring good models directly from data can be difficult because of the large quantities of data necessary to accurately estimate the state transitions. This issue is even more difficult for models of partially observable domains, where it is frequently necessary to estimate the probabilities for the unobservable state transitions using the observations through methods such as expectation maximization (EM) (Bengio and Frasconi, 1996).

When planning to interact with another agent (such as a person), data sparsity can present a serious problem. In many domains, it can be very time consuming and potentially expensive to collect human task performance data. Because human behavior is highly variable, humans performing the same task may follow many slightly different trajectories through the state space. With limited training data, it is possible that many trajectories that people would define as qualitatively very similar to the trajectories in the data may never have been observed.

A good prior model of the problem can solve many of these issues. Here is where the regularity of human social behavior can be a great asset for planning in social domains. High level characteristics of people's goal oriented behavior can be encoded by a domain expert and used to create a model that provides a reasonable state transition structure over the entire space of possible behaviors. The accuracy for those parts of the state space that correspond to frequently observed trajectories in the training data can be improved by using this data to update the state transition probabilities.

We specify models using probabilistic rules that define the effects of actions. The applicability of these rules is determined by state-based preconditions. This allows the changes to the world state encoded in these action effects to be described in terms of both the observable aspects of the physical state and the unobservable intentions motivating the human's behavior. The domains being described are dynamic, both because the actions of the people being interacted with are part of the state transition model and because the environment the interactions take place in often have properties that change over time. In order to allow the complex interactions between robot, person, and environment to be expressed in a concise manner, each of these potential sources of change in state are described in terms of the actions available to them and the effects of those actions. The next state transitions are computed for each of the robot's actions by modeling both the person and the environment as agents which are also allowed to select an action at each timestep. The resulting model includes all of the possible states that are reachable given the initial conditions and action descriptions of the robot, human, and environment. The state transition probabilities are based on the relative weights given to action outcomes in the prior model and evidence from state trajectories recorded from humans performing the interaction with one another, when available. Because of the time-dependent nature of these types of problems, a time-indexed state space is used (the rationale behind this representation will be discussed in Chapter 7). This means that action rules may have time-based preconditions and that human data added to the model will include information about when the actions were performed.

The human social behaviors that this research focusses on are assumed to be both goal oriented and relatively efficient. Therefore, reward-based planning can produce policies for behavior that are reasonable policies for interaction. In order to do so, however, it is critical that the reward structure reflect the value of states both in terms of task achievement and their social value. For example, a robot trying to perform the task of entering an elevator that is currently occupied by human passengers could simply enter the elevator the moment the doors opened. While this may result in the robot succeeding at the task of getting on the elevator, it would also delay or possibly prevent the exit of the human passengers. While this policy succeeds according to a myopic task-oriented performance criteria, it fails at the greater goal of performing the task in a socially appropriate manner. In order to lead to socially appropriate action, the reward structure must reflect the preference for states that allow all of the agents involved to achieve their goals when possible. Reasoning about this balance between self-interest and the common good is a characteristic of human social decision-making.

6.1. Creating Model Structure

The purpose of the prior model is to specify a state-space which includes all of the states that could possibly be reached over the course of the interaction (while omitting possible state variable combinations that are meaningless in context) and to provide a reasonable estimate of the probabilities of action outcomes over this entire state space. For state trajectories commonly observed in human task performance data, these probabilities can be refined. But given the large state spaces of realistic problems, many states will not be visited often enough to get accurate (or possibly any) estimate of the state transition probabilities directly from the data. Because of this, it is critical that the prior model provide reasonably accurate estimates about the characteristics of all the possible states so that the robot can form robust policies even when data is sparse.

While the actions available to the robot determine its policy, it is important to note that the robot is not the only source of changes to the state in the environment. It is not even the only agent in the environment that is selecting actions in a goal-oriented manner. But the robot does not have any direct control over the actions chosen by the human it interacts with or many of the ways in which the state of the environment might evolve. These effects on the next state must be modeled as a part of the effects of the robot's own actions. Both the human and the environment can also be thought of as agents taking actions to affect the state. The results of these actor's actions may also interact in a way that has an effect on the state outcome. In order to simply express this complex relationship, the robot, the human, the environment, and their side effects are all described in terms of sets of action rules with state-based preconditions and weighted effects. These action rules are then composed at each timestep to yield a probabilistic joint action effect for all of the robot's actions.

The overall structure of the domain description file is given in Table 6.1. The rules used to parse the domain description language are presented in Table 6.2. The following subsections describe the details of how a model is specified using this language.

6.1.1. State and Actions

The state is defined in terms of a set of variables and the range of values these variables may take. The state space is time-indexed, so the number of timesteps that an episode of the problem takes place over must be specified. The time index is represented in the state by a special variable named "time" which may be referred to in the preconditions of action rules or reward rules, but may not be affected by any action rules. This state variable's value always refers to the current timestep. The rest of the state space is defined by a list of state variable names with the corresponding range of integer values that each variable may take.

The action space is defined in the same manner. In the action space, three special action variables are defined, "Other", "Env", and "SideEffect". These actions are used to

TIMESTEPS
<VAL>
number of timesteps
corresponds to special state variable named "time"

STATES
<STATEVAR DEF LIST>
state variable name and the range of values it can take on

ACTIONS
<ACTVAR DEF LIST>
action name and the range of values it can take on
includes the special action variables "Other", "Env", and "Side Effects"

OBSERVATIONS
<STATEVAR LIST>
subset of the state variables which are directly observable

<ACTION RULE LIST>
rules that describe the effects of actions on the state

<REWARD RULE LIST>
rules that describe the reward value of certain states

START
<STARTVAL LIST>
state variable values for the set of possible start states

Table 6.1. Structure for the domain description file used to create models.

specify the action rules that describe the actions of the human (the other agent), changes in the environment, and side effects that are caused by the interaction of other individual action effects from the previously named sources. Because these actions are automatically taken in response to the actions of the robot at each timestep, the value for these action variables (which would be used to specify an action that the action effect is caused by) is ignored. These action effects do not correspond to an action choice that may be selected by the robot, they are used to produce the world's response to the effects of robot's actions.

6.1.2. Action Rules

Action rules are described using a simple language that specifies the action identifier, the preconditions that determine whether the action is applicable in a state, the possible effects of that action on the state variable values, and a weighting factor that specifies the relative likelihood of those outcomes. This action description language provides a simple and flexible way to express the state transition structure in terms of small subsets of relevant variables and their values.

Rules:

```
<ACTION RULE LIST> → <ACTION RULE LIST> <ACTION RULE> | <ACTION RULE>

<ACTION RULE> → <UNWEIGHTED ACTION RULE> |
                <UNWEIGHTED ACTION RULE> WEIGHT <WEIGHT>

<UNWEIGHTED ACTION RULE> → RULE <ACTION DEF>
                           EFFECTS <ACTION EFFECT LIST>
                           CONDITIONS <PRECONDITION LIST>

<ACTION DEF> → <ACTVAR NAME> <VAL> <ID>

<ACTION EFFECT LIST> → <ACTION EFFECT> <ACTION EFFECT LIST> |
                       <ACTION EFFECT>

<ACTION EFFECT> → <ACTION VAR EFFECT> <ACTION EFFECT> |
                  <ACTION VAR EFFECT>

<ACTION VAR EFFECT> → <REL ACTION VAR EFFECT> | <ABS ACTION VAR EFFECT>

<REL ACTION VAR EFFECT> → <STATEVAR NAME> REL <VAL>

<ABS ACTION VAR EFFECT> → <STATEVAR NAME> ABS <VAL>

<PRECONDITION LIST> → <PRECONDITION> <PRECONDITION LIST> |
                      <PRECONDITION>

<PRECONDITION> → <STATEVAR NAME> <VAL LIST>

<VAL LIST> → <VAL> <VAL LIST> | <VAL>

<STATEVAR DEF LIST> → <STATEVAR DEF> <STATEVAR DEF LIST> | <STATEVAR DEF>

<STATEVAR DEF> → <STATEVAR NAME> <VAL> <VAL>

<ACTVAR DEF LIST> → <ACTVAR DEF> <ACTVAR DEF LIST> | <ACTVAR DEF>

<ACTVAR DEF> → <ACTVAR NAME> <VAL> <VAL>

<REWARD RULE LIST> → <REWARD RULE> <REWARD RULE LIST> | <REWARD RULE>

<REWARD RULE> → R̄EWARD <VAL> CONDITIONS <PRECONDITION LIST>

<STARTVAL LIST> → <STATEVAL LIST> <STARTVAL LIST> | <STATEVAL LIST>

<STATEVAL LIST> → <STATEVAL> <STATEVAL LIST> | <STATEVAL>

<STATEVAL> → <STATEVAR NAME> <VAL>
```

Terminals:

```
<WEIGHT> - integer
<VAL> - integer
<ID> - string
<STATEVAR NAME> - string
<ACTVAR NAME> - string
```

Table 6.2. Description of general language for domain models.

For each action rule, the action variable and variable value that the rule describes is specified. A single action variable may correspond to one or multiple actions. For example, an action variable that represents setting a robot's speed may correspond to several distinct actions, one for each possible speed setting. In order to uniquely identify each action rule an additional identifier is also given. This is necessary because the same combination of action variable and value may have different effects and different weights depending on the preconditions (these IDs could be assigned automatically by the parser, but if may be useful to the modeler to assign meaningful IDs to express the differences or similarities between effects). Multiple action rules may be necessary in order to specify the full range of possible outcomes and their relative weights for a single action.

For each action rule, possible changes to state variables caused by that action are specified as a list of action effects. Action effects may affect only a single state variable or multiple state variables simultaneously. Each action effect corresponds to a single setting of the state variables, so an action rule with more than one action effect listed has a non-deterministic effect. All of the action effects for an action rule are given equal weight. The weight, if specified, is an integer that specifies a ratio of how much less likely the action effects listed are than the default. For example, a weight of 2 would mean that all of the action effects are half as likely, with a value of 0.5. If no weight is specified, the weight for all action effects are assumed is assumed to be the default value 1. Non-deterministic effects with different weights on the outcomes can be specified by defining multiple action rules with the same preconditions. Weights were chosen rather than exact probabilities because they are more intuitive to use given the way that action rules are composed to create the probabilistic state transition structure.

An example action rule describing the behavior of the human agent is shown in Table 6.3. The "Other" action name specifies that this action rule describes the behavior of the uncontrollable human agent. Because the robot cannot choose the actions taken by this other agent, the action value is 0, with "b4" providing a unique identifier for this action rule. This action rule has a single possible effect, to turn on the headlights of the human's car. The condition list describes the state variable values that determine when this action rule may take effect. It makes sense to apply this action only in states where the headlights are off. The human may turn on the lights at any time before they move into the intersection. The goal condition restricts this rule to cases where the human has the intention *not* to yield to the turning driver. The velocity 0 condition reflects the fact that people were only observed to turn on their headlights when their cars were not moving. The weight of 100 for this action means that this action effect is $\frac{1}{100}$ as likely to occur as an outcome with the default weight. This weight corresponds to the fact that it is very rare for a car that does not intend to yield to flash its headlights.

```
RULE
Other 0 b4
EFFECTS
Light_S ABS 1
CONDITIONS
Light_S 0
Pos_S 0 1
Goal_S 1
Vel_S 0
WEIGHTS
100
```

Table 6.3. An example action rule for the human agent describing a light flashing behavior in the Pittsburgh left model.

6.1.3. Observations, Reward Rules, and Initial States

The observations are defined by a list of the subset of state variables that are directly observable for the domain. This subset is used to automatically create an observation for each state. Observations defined in this manner are aliased rather than noisy (each state has only one corresponding observation that may correspond to multiple different states). This way of specifying observations was used for the Pittsburgh left domain, in which the only source of state uncertainty is the unobservable intentions of the human. For domains with noisy observations, the language could be easily extended to define observation variables and specify a probabilistic relationship between the observation variables and the state variables that they arise from.

The elevator domain models, while created using a similar rule-based description, were not implemented using custom-designed scripts rather than this domain description language. The observations for the elevator model did not directly correspond to any of the underlying state variables and were defined in terms of a probability distribution corresponding to the state. Details of this implementation will be given in Section 6.6.1.

Rewards are expressed as a numeric value, with state-based preconditions of the same form as used in the action rules specifying what states should be assigned the given value. In the example domains given, reward is associated with a particular state rather than a state-action pair.

After the keyword ”START” vectors of all the state variable names and their corresponding values are listed. Each line specifies a single state. Each state listed is assumed to have an equal probability of being the starting state.

6.2. Producing Next-state Transitions

The state transitions possibly caused by one of the robot's actions represents more than the immediate effect of the action selected by the robot. They also encode all of the possible effects of actions that may be taken by the human in response to that action, the possible changes to the environment that may occur, and side effects that are the result of the interactions of any of these changes in state. In this model, the description how the world and the human's behavior evolve over time is built into the robot's actions' state transition structure. These complex probabilistic transitions are created by sequentially composing the action rules that make up the model.

The composed action effects are created as follows: for each robot action, all relevant action rules are applied to a start state in order to produce a set of intermediate next-states with weights determined by the weights of the action effects. For each of these intermediate states, the human's responding behavior is created by applying the relevant action rules for the human. A new set of intermediate states is produced, and the weights associated with these states are determined by combining the weights of their predecessors and the weights of the human's action effects. The same procedure is applied to these states for the environment's action rules and the action rules for side effects of the interactions. The final next-states and weights are stored in a dictionary representing the state action transitions for the model at that timestep. This procedure is repeated for each start state until the full set of reachable states at the next timestep have been created. Then the process is applied to this set of states, and each following set of states for each timestep until the penultimate timestep has been reached. This procedure is described in psuedocode in Table 6.4.

Once the action rules are composed for all timesteps, the resulting data structures are used to create the state transition matrices for the robot's actions. All of the states in the model are obtained by adding the appropriate time index to the sets of states for each timestep in S and empty state transition matrices are initialized. The matrix values are obtained from the weights stored in *successors*. Were it not for the weights that reduce the likelihood of some outcomes relative to others, the prior model's structure would function as a uniform Dirichlet prior over all the states and action effects that could be encountered by the robot. However, because task data not be available or may only sparsely cover many parts of the state space, it is necessary to have a prior model that represents the the problem well without the addition of data. Actions for which there are no existing transitions from a state are actions that cannot occur according to the model. In this case, the action will transition to a special "failure" state that transitions immediately to the end state. If there is no data to be added to the rule-based model, the matrices will be normalized so that the weights are expressed as probabilities. If there is data that can be used, it will first be added to the model as described in Section 6.4.

```
Creating state transition structure
Robot, Other, Env, and SideEffects are lists of action rules
Robot contains the rules for the actions that may be chosen by the robot
successors is a dictionary that stores the weighted state transitions
    for each action for every timestep
$S$ is a list of sets of states that occur at each timestep,
    initialized to empty sets

$S_1$ = start states
for $t$ in 1:$T$
    for $s$ in $\mathcal{S}_t$
        successors[$t$][$s$] = []
        for rule in Robot
            if rule.applicable(s,t)
            act = rule.getactname()
            states = rule.apply(s)
            for agent in {Other, Env, SideEffects}
                nextstates = []
                for agentrule in agent
                    for $ns$ in states
                        if agentrule.applicable(ns,t)
                            nextstates.append(agentrule.apply(ns))
                states = nextstates
            successors[$t$][$s$][$act$].insert(states)
            $S_{t+1}$.insert(states)
            $S_{t+1}$.setweightstodefault()

rule.apply(s)
    newstatelist = []
    for $e$ in rule.effects
        newstate = e.apply(s)
        newstate.weight = s.weight $*$ rule.weight
        newstatelist.append(newstate)
    return newstatelist
```

Table 6.4. Description of how action rules are composed to produce the state transition structure.

6.3. Representing Intention and Reward

The central tenent of this modeling approach is that the behaviors exhibited during an interaction are dependent on the intentions of the interacting agents. Because decision-theoretic controllers act to maximize reward, it stands to reason that the greatest reward must correspond to the outcome intended by the acting agent in order to produce the desired behavior. It is common to think of reward in tasks like this as being associated with an agent's achievement of its own goals, so that the policies will produce self-interested behavior. Purely self-interested behavior is unlikely to result in socially appropriate action, however. Considering the elevator domain, the robot could always achieve its own goal of getting onto the elevator by moving to enter as soon as the doors open without letting any passengers out. But this would inconvenience the people on the elevator and potentially prevent them from exiting. This behavior would be viewed as antisocial and inappropriate.

In order to create reasonable policies for social action, the reward structure should also contain a notion of the public good of actions. The agent prefers states in which both it and the other agent achieve their desired outcomes. When their goals are in conflict, the agent prioritizes its own goals over those of the other agent's, but still prefers better outcomes for the other agent over poor outcomes. For example, if the turning car intends to take the Pittsburgh left, it gets the greatest reward for successfully going through the intersection before the other car does. But it receives more reward for turning left after the other car goes through the intersection than it does for going through the intersection first in such a way that it prevents the other car from getting through the intersection at all before the light changes.

The robot should receive reward for all states which correspond to either the achievement of the robot's goals or the human's achievement of their own goals. The values of these reward states should be determined by the intentions of the robot, with the highest reward corresponding to the robot's most desired states. Recall that for the purposes of this thesis an intention is represented as a preference ordering over possible goal or outcome states for a task. In order that the rewards reflect social concerns, the goals that the intention gives an ordering over should be described in terms of outcomes and events for both the human and the robot. For example, in the Pittsburgh left domain, if the robot's intention is to make the Pittsburgh left, then it should receive a reward for states that correspond to the event of its car crossing through the intersection first. But the robot should also receive a reward for states that correspond to the human's car crossing through the intersection second, so that it will prefer courses of action for which one of the human's goals is also achieved. If the interaction plays out so that the human crosses the intersection first despite the robot's intention to take the Pittsburgh left, the robot receives a smaller reward for the human's goal achievement, because this goal was in conflict with it's own interests. The rewards assigned for goals in the Pittsburgh left interaction are chosen so that the combinations of outcomes for the human and the robot have a value ordering that matches the desirability of these outcomes given the robot's intention. These rewards for the turning car (T) driven by the robot and the straight (S) car driven by the human are shown for each possible outcome in Table 6.5.

Table 6.5. Rewards for possible interaction outcomes in Pittsburgh left domain

Intention	Outcome				
	T First	S First	T Only	S Only	Neither
Pgh left	T(15) + S(20) = 35	T(15) + S(10) = 25	T(15) = 15	S(10) = 10	0
Reg left	T(15) + S(10) = 25	T(25) + S(10) = 35	T(15) = 15	S(10) = 10	0

In the Pittsburgh left domain, the robot and human's intention may be in direct conflict, so the human's intentions are not taken into account in the reward, only the fact that it is preferable to the human to be able to cross through the intersection rather than to be unable to cross. Differences in the reward for goal achievement by the human reflect preferences based on the robot's intentions, not the human's. For domains where the robot's and human's goals do not potentially conflict, it may be desirable to assign rewards to state according to the human's intentions as well, so that the robot will be motivated to use policies that are the most mutually beneficial. In general, the amount of reward that the human's goals contribute and the structure of that reward is dependent on the domain and on the kind of robot behavior desired. At one extreme is a purely self-interested robot that acts only to achieve its own goals according to its intentions (which is the most common way of modeling reward for robots acting in populated environments). At the other is a purely altruistic robot that acts only to help the human it is interacting with achieve their goals according to their intentions (this would be identical to some of the POMDP-based assistants discussed in the related work).

Undesirable outcomes are penalized with negative rewards. States may be physically undesirable because they have an unquestionably high cost to the participants, such as collision between two cars in the Pittsburgh left domain. But there are also states that may be undesirable for social reasons, even if they pose no risk to the participants. Again, consider the simplified driving task of the Pittsburgh left interaction. There is no cross-traffic, so a car running a red light would not be risking collision. However, human participants know that this is an illegal driving maneuver, and there is a social stigma to running the red even when there are no cars present to yield to. People will sometimes jump the light or go through an intersection while a light is turning from yellow to red, and these boundary conditions are seen as more acceptable. The reward structure for this problem captures the social cost of running the red light in a way that is accurate across these different situations. A penalty is given for each timestep that the robot spends in the intersection while the light is red. Moving into or out of the intersection as the light is changing incurs a small penalty, but driving through the intersection during the middle of the red light (which takes several timesteps) has a greater cost.

6.4. Making Use of Human Data

If state trajectories of humans performing the task are available, these data can be added to the prior model in order to produce a state transition matrix that represents more human-like behavior. Though, for reasons already discussed, this data will most likely not cover the entire state space of the prior model, it will improve the estimates of the state transitions in the parts of the model most commonly visited by the population of people who produced the data. These data will likely contain information about nuances of human behavior or characteristics of the environment that may not be modeled in the prior model, allowing a higher quality policy to be found.

Because the data is given in terms of state trajectories, it can be added to the model using a simple frequentist approach that adds a default transition weight of 1.0 to the transition matrix for each observed transition. For states that are visited often in the data, the data transitions predicted by the data will dominate the relative weights given by the prior model. Information about which discrete actions were chosen may not be available for human task performance data (because the human controls were not given in terms of these actions, for example). In this case, the action corresponding to the transition between two states can be inferred by comparing their state variable values and selecting an action that is capable of producing the observed change in state. In cases where more than one of the actions may have produced the change, the observed transition should be added to the transition matrices for all of the possible actions.

It may be the case that there are states present in the data that correspond to state variable combinations that are not present in the prior model, or transitions between states for an action that are not present in the state transition structure of the prior model. This should not be a common occurrence for a reasonably complete prior model. One exception is in the case where there are certain states and actions corresponding to behaviors exhibited by humans in the role the robot will be taking that a design decision has been made to prevent the robot from performing. One example of this in the Pittsburgh left domain is the action of cutting the corner on the left turn. Humans sometimes turned directly into the wrong lane of the perpendicular street when making the turn, either to make the turn more quickly or due to difficulty in controlling the car. In addition to this being improper driving behavior, at the resolution of control available to the robot this is likely to result in instances of the robot driving onto the sidewalk, which would appear highly unnatural. The human data for these transitions between states that don't exist in the prior model are ignored, though transitions between existing states from the same trajectories are added as normal.

6.5. Less Expressive Representations

One of the major goals of this research is to evaluate the benefits of using a time-indexed POMDP representation to model social tasks. It is hypothesized that both explicitly representing time-dependent action outcomes and reasoning about the human participant's unobservable intentions allow a robot to act to achieve their goals in a more socially appropriate manner. In order to test this hypothesis, two different types of less expressive models are created from the same domain descriptions used to produce the time-indexed POMDP models. In order to test the effects of reasoning about human intention, sets of MDP models are created for each POMDP model, which model the fully observable aspects of the state in relation to a single fixed intention for the human participant. In order to evaluate the effects of the time-indexed state representation, POMDP models are produced that do not have an explicit time representation as part of their state space. The performance of these less expressive models is compared to that of the time-dependent POMDP models in a human subject experiment described in Chapter 8.

6.5.1. MDP Models

The state space of a POMDP model is not directly observable during execution. POMDP policies choose actions based on a belief about which states the agent is acting in, rather than on a particular state. If a human's intention makes up part of this unobservable state, the policies may behave differently for different beliefs about the human's intentions given the the observations received. For the Pittsburgh left domain in which the human subject experiments are carried out, the human's intention is the sole unobservable part of the state space. The rest of the physical characteristics of the underlying state are relayed to the agent directly through the observations, or can be inferred from the action effects. Because of this characteristic, it is simple to produce a closely related MDP model by eliminating the human's intention from the state space and making the state fully observable.

There are a couple of different ways this could be done, and the model should be chosen to allow for the most fair and reasonable comparison to the POMDP model. One option would be to simply eliminate the human's intention both from the state space and from the rules that produce the state structure. This would create an MDP that combined the actions of the human participants' different intentions into one aggregate model. More than not being able to reason about people's intentions, this model would not represent people's behavior as intentional at all. The model would suggest that because it is common to see humans flash their headlights (when they intend to yield) and accelerate quickly into the intersection (when they don't intend to yield), it would also be likely to see a person to flash their headlights and then immediately accelerate into the intersection. A model of this form is a poor representation of people's behavior.

An alternative MDP representation is to still represent people's behavior in terms of their intentions, but to create a separate MDP model that describes people's behavior for each possible intention. A policy produced by solving one of these models effectively assumes that the person it is interacting with holds this intention and acts accordingly. It is not an uncommon occurrence in human-human social interaction for someone to incorrectly infer the intention of the person they are interacting with, and people are accustomed to recognizing and compensating for these situations. By using a policy for interaction based on an assumption about the human's intention (that may or may not be correct), the onus of figuring out the other agent's intention and responding appropriately is shifted onto the human participant. This is in contrast with the POMDP policies, which maintain a belief about the human's intention based on observations and choose actions according to that belief.

The MDP models are created by first creating the states and state transition structure for the time-indexed POMDP. The resulting state space is then partitioned according to the values of the variable for the human's intention. The state transition structures for the separate state spaces are taken directly from the state transition structure of the POMDP model. Because the states of the MDP models are fully observable, the observation information for the states is discarded.

6.5.2. Non-Time-Dependent POMDP Models

The POMDP models produced without a time index as part of their state space are referred to as *non-time-dependent* POMDP models because they cannot accurately represent time-dependence in action outcomes. All MDP and POMDP models have some sort of time-based structure by virtue of the fact that they are sequential models. For example, if one state always occurs immediately before another in a particular model and never after it, there will be state transitions from the first state to the second but none from the second to the first. These models can therefore be thought of as *naively sequential*. They are capable of representing the order in which states are encountered, but represent the effects of time on these state transitions with limited or no accuracy. The amount of time spent in a state can be represented only using a self-transition. The resulting time in state will have an exponential distribution, which may be a poor match for the actual distribution of the time spent in the state. If the state transition probabilities change over the time spent in the state, these changes cannot be represented by the model. The state transition probabilities are restricted to being an average over the different time-dependent probabilities for that state.

The non-time-dependent models are produced using the same procedure described for the time-indexed model (Table 6.4), with one major exception. The time index variable is not a part of the resulting state space. The action rules are still applied in a time-dependent manner (if a rule has time-dependent preconditions it will still be applied only at the appropriate timesteps), but the new states resulting from the rule's application will not have a state variable indicating the timestep at which they were created. Also, rather than having separate dictionaries to store the state action transitions for each timestep, a single dictionary is used for each action across all timesteps. The resulting models represent the time-dependent aspects of the domain description as accurately as they can without an explicit representation of time in the state space. The resulting models have fewer states and more dense state transition matrices than their equivalent time-indexed models.

6.6. Domain Implementation

In order to prove the effectiveness of this modeling approach, models were implemented and used to create policies for two situated social task domains. For the elevator task, a simple probabilistic rule-based description of human behavior and the environment was developed that could be used to create a wide variety of models with differing task characteristics and resolution. The second implementation, the Pittsburgh left domain, was a more complex task that corresponded to an interaction that was performed both by pairs of human drivers and later by humans interacting with artificial agents. Models were created by combining a rule-based prior model with human task performance data. These models were used to produce policies that interacted with humans to perform the same driving task that was used for data collection.

6.6.1. Elevator Domain

The POMDP models used in the experiments for the elevator domain represent a simplified version of an elevator entry task for a mobile robot. The robot starts at the far end of a lobby with an open elevator door in front of it. The elevator and lobby are divided into regions of a fixed size along the dimension parallel to the doorway (The number of regions can be changed to create problems of varying size). Passengers inside the elevator are either attempting to get off the elevator or staying on to ride to another floor. If a person intends to exit, they may hesitate for several timesteps before starting to move, and may hesitate again for a timestep before exiting the elevator if the robot is a grid square or less away from the other side of the doorway (this is intended to simulate the possibility that someone may stop if they become confused by the actions of the robot). The robot receives an observation at every timestep that indicates that the closest person in front of it is moving, the closest person in front of it is stopped, or that the elevator and doorway are empty of people. The observations of the closest person's motion are noisy, with a 20% chance of being incorrect. At each timestep, the robot can choose to either wait or move forward. The move action is assumed to succeed in moving into the next region unless the robot attempts to move into a place currently occupied by a person or into the doorway when the door is about to close. The doorway is a bottleneck, and if the robot and a person meet at it, they will both be stuck until the door closes.

This task is modeled as episodic and of finite length. A fixed amount of time passes before the door closes, the episode ends, and the robot receives a one-time reward based on whether the robot succeeded in getting on the elevator (if the robot fails to get on in time it is penalized). The robot also receives a reward for each human passenger that achieves their goal (Whether it is to stay on the elevator or to exit). Because it is a breach of etiquette not to allow exiting passengers out of an elevator, it is penalized for each person intending to exit that it prevents from exiting. The reward is undiscounted, so the robot is not penalized for not getting on the elevator as quickly as possible, as long as it achieves its goal before the end of the episode. Each possible outcome (the success or failure of the humans and the robot at their goals) is assigned a reward value based on its desirability to the robot.

The state space is made up of the robot's position, the people's positions, whether each person intends to exit, and the current timestep. Models of different state sizes can be created by varying the number of people in the elevator, the period of time for which each person may hesitate before moving, the length of time that the door stays open, and the number of regions defined over the elevator and lobby. The model definition for this domain is described in greater detail in Appendix A.

6.6.2. Pittsburgh Left Domain

The POMDP models for the Pittsburgh Left driving task model the interaction in the driving simulator with the robot performing the role of the driver turning left and a human performing the role of a driver going straight. A separate model was created for each of the possible intentions of the turning driver (to take a Pittsburgh left or to take a regular left and yield to the oncoming car). Each model represents both of the possible intentions for the straight driver (to go through the intersection first or to yield to the turning car). The same domain description file was used to produce both models, with the intention for the robot driver producing a different reward structure for each model.

The continuous values that make up the state of the interaction are coarsely discretized to produce a finite state space of tractable size. The state of each model is made up of variables for: the human's intention, the robot car's position, the human's car's position, the robot car's turning angle, the robot car's speed, the human's car's speed, whether the human has flashed their headlights, whether a collision between the cars has occurred, the color of the traffic light, and the current timestep. There are also two special states, an absorbing failure state that is the outcome of all illegal actions, and an end state. The action space for the robot is made up of three possible speeds (stopped, slow, and fast) and a command to turn the car. The model is defined over 60 timesteps, each corresponding to one half-second of the thirty second interaction.

The observation space is made up of a subset of the state variables: the robot's position, the human's position, the human's speed, whether the human has flashed their lights, and the color of the traffic light. Because the observations arise directly from the state value, they are aliased rather than noisy.

When the robot exits the intersection, it receives a one-time reward. When the human exits the intersection, the robot receives another one-time reward with a value determined by the robot's intention. If the robot's intention is to yield rather than taking the Pittsburgh left, the robot receives a greater reward if the human exits the intersection first rather than second, and vice versa if the robot intends to take a Pittsburgh left. If a collision occurs, the robot receives a one-time reward with a large negative value. If the robot is positioned in the intersection while the traffic light is red, it receives a reward with a small negative value. This reward will be received at every timestep that the robot is in the intersection during the red light. Reward in these models is discounted by a factor of 0.99. This discount factor was chosen to give preference to earlier goal achievement while still maintaining the ordering of the goals' values regardless of when they were achieved.

Data from the experiments of two humans interacting in the driving simulator was discretized according to the state definition. This data, which was recorded at the rate of 20 Hz, was downsampled to the 2 Hz rate of the control model. Each experimental trajectory was used to create 10 data examples by adding an offset of 0 to 9 to the starting timestep for the downsampling. This was done to increase the number of training examples available from the sparse data and to capture the minor variability in the time at which actions may be taken.

The model design is described in greater detail in Appendix A. Additionally, the model description file defining the models used to produce the policies for the experiment in which the robot interacted with human drivers is included.

CHAPTER 7

Time-State Aggregation for POMDPs

7.1. Representing Time-dependence in POMDPs

In Markov decision process models, any time-dependent differences in action outcomes in a state are abstracted away for the sake of computational efficiency. But in many realistic domains this assumption does not hold. In these cases, it is important to include some sort of explicit representation of time in the state space in order to represent important time-dependent aspects of the problem. Failure to do so could result in models that are incapable of producing policies that perform well due to the representational limitations of the model.

The most common extension used to model time dependence are semi-Markov models. The states of these models use a probability distribution to represent the duration of time spent in that state. Some variants of semi-Markov representations also allow the state-transition function to be time-dependent as well. While this extension provides a powerful way of explicitly representing time, it also requires that *a priori* assumptions be made about the form of the distribution that best represents the time in state for that domain. It may be difficult to do this accurately for domains with time-dependent characteristics that are complicated or poorly understood. Additionally, the algorithms that are used to solve fully Markov models cannot be applied directly to semi-Markov models. This is of particular concern when using POMDP models. Most existing fast approximation algorithms for POMDPs are designed for fully Markov models. The existing algorithms that address semi-Markov models use semi-Markov states to do hierarchical state abstraction rather than addressing the issue of time-dependent action outcomes. In order to take advantage of state-of the art POMDP solution algorithms when solving problems with nonstationary state transitions, a way of representing time dependence in a fully Markov model is needed.

The use of a time-indexed state space is a simple way to represent time-dependence that is also agnostic to the particular time-dependent characteristics in a domain (aside from the need to choose a granularity for the time discretization that is appropriate for representing that domain). The straightforwardness of this representation and the fact that it produces a fully Markov model makes it preferable to semi-Markov models for complex problem domains. The downside of this representation is that adding a time index to the state may dramatically increase the size of the state space. This could result in POMDP models that are too large to solve in a reasonable amount of time, even using fast approximation algorithms.

Because a time-indexed model makes very few *a priori* assumptions about the form of the time-dependence in the model, it is highly likely that many of the time-indexed states are redundant or have little impact on the choice of the optimal policy for the model. Ideally, one would like to aggregate these unnecessary states prior to solving the model, yielding a more compact model that will be less computationally expensive to solve. But predicting which states can be abstracted away without knowing the form of the optimal policy is a difficult problem in and of itself. Fortunately, existing work on state aggregation can suggest approaches to the more specific problem of abstracting time in the state space.

Many state aggregation algorithms for MDPs compare reward-based state values and next state distributions to determine which states to merge. The expected reward is typically calculated based on the an MDP policy (that selects one action in each state), so applying these approaches to the POMDP models may not result in policies that perform well in partially observable domains. There are existing methods for state abstraction in POMDPs that approximate the state space or belief space with a small set of basis functions. This basis function representation performs best for short planning horizons, and may be less effective in domains with delayed rewards.

Evaluating the effects of representing time-dependent action outcomes is one of the main goals of this thesis research. An aggregation method that acts simultaneously over all dimensions of the state space makes it difficult, or impossible, to determine whether differences in performance between policies for aggregated models are due to differences in time representation or due to aggregation of other dimensions of the state space. Also, an aggregation method that operates on the time dimension of the state space alone can also take advantage of the special structural properties of time-indexed state spaces.

If one knew which states were needed to model time-dependent transitions in order to achieve good performance, one could abstract away the time information in the states that did not matter. Acting on that intuition, a state aggregation algorithm was designed that operates exclusively in the time dimension of the state space. States are scored with a value based on their future achievable reward, and these values are used to identify sets of states to be combined. This algorithm attempts to minimize the effect of the aggregation on the quality of the policy obtained from the resulting model by only combining states that are likely to lead to similar future rewards.

In this chapter, the time-indexed state representation for POMDP models is described in detail. Then the time-state aggregation method is explained, including the possible representations for the state values used to perform the aggregation. Guidelines for choosing a practically useful aggregation threshold (which determines how aggressively a model is aggregated) are discussed. Finally, experimental results are presented for both of the example domains used in this thesis.

7.1.1. Time-indexed POMDPs

A POMDP is defined as a tuple made up of sets of states, actions, and observations and the matrices of probabilities over the observations, state transitions, and rewards.

$$\{\mathcal{S}, \mathcal{A}, \mathcal{O}, O(o, s, a), T(s, a, s'), R(s, a)\}$$

In a time-indexed POMDP model, the state can be thought of as being made up of two distinct sets of state variables, those that describe the state of the environment , $sv_i \in \mathcal{E}$, and a singleton set, $\mathcal{T}$, of the variable that represents the time index.

$$\mathcal{S} = \mathcal{E} \bigotimes \mathcal{T} \ where \ sv_1, ...sv_n \in \mathcal{E} \ and \ t \in \mathcal{T}$$

The resulting POMDP contains many states that have the same values for all state variables except t. States having this property will be referred to as *physically identical*. The state space created without the addition of the time index will be referred to as the physical state space. These states are the states that make up the non-time-dependent model that is roughly equivalent to the time-indexed model.

$$\mathcal{S}_P = \mathcal{E} \ where \ sv_1, ...sv_n \in \mathcal{E}$$

Note that in practice, just as many possible combinations of state variable values may not actually exist as states of the world, not all of the physical states will exist at every timestep.

For the model formulations used in this thesis, observations are assumed to arise from characteristics of the physical state. Therefore, observations are assumed not to be time dependent, so if two states $S1$ and $S2$ are physically identical, their observation probabilities will also be identical.

$$O(o, S1, a) = O(o, S2, a) \forall a \forall o$$

A sequence of physically identical states that transition from one to the next under a certain action represent the amount of time that will be spent in that physical state when that action is chosen. These states are equivalent to one semi-Markov state with a step function for a time distribution that transitions from 0 to 1 after n timesteps, where n is the length of the state sequence. This is illustrated in Figure 7.1 for $n = 3$. In a Markov representation without a time-indexed state space, the expected time in a state can be modeled only as a self-transition, which follows an exponential distribution. Setting the probability of self-transition to $\frac{n-1}{n}$ will make the time spent in the state equal n in expectation, but the variance will be n^2.

The time-indexed state space representation has different strengths and weaknesses than a semi-Markov representation. A time-indexed state space allows action outcomes to be time-dependent, which is not possible to represent using the common definition of semi-Markov state. However, temporally extended actions can not be represented in a time-indexed state space without adding additional state information. This is because the outcome for an action depends on the chronological timestep, not on the timestep in which that action was initially selected in that state. A time-indexed state space can be augmented to represent temporally extended actions by adding a ”clock” variable to keep track of the progress of an action selected in a state. As temporally extended actions were not a characteristic of the domains modeled in this thesis, this representation was not explored. The ability to model time-dependent action outcomes in a straightforward manner and the computational tractability of the resulting models' fully Markov state representation are both clear strengths of time-indexing relative to semi-Markov models.

7.2. Time-state Aggregation Algorithm

In this research, a novel state aggregation method is developed that performs state aggregation using a heuristic estimate of state values. This aggregation method takes advantage of the special structure of time-indexed models to compute these state values quickly. While this method is inexact (in that the resulting aggregated models are not guaranteed to have the same optimal reward and policy as the original model), the state combination criterion used is more appropriate for partially observable models than similar criteria used to perform inexact state aggregation in MDP models. This method also allows the aggressiveness of the aggregation performed to be pre-determined by the user by choosing a threshold for state value comparison.

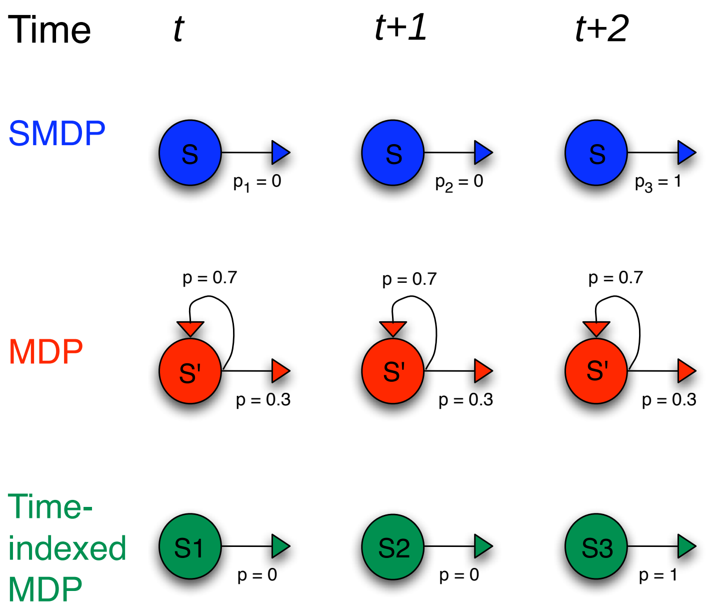


Figure 7.1. A single Markov state approximates a semi-Markov state's time distribution with a self-transition. A sequence of time-indexed states represents an equivalent time distribution.

Many MDP-based state aggregation algorithms assign a value to a state based on the expected reward for the action selected by an MDP policy for a state or the current estimate of the value for the optimal action (Li et al., 2006). These approaches do not make sense for POMDPs, as the action selected when acting in a state is based on the agent's belief distribution over states given its history, not the state it is currently acting in (which cannot be directly observed). That means that any number of different actions could be chosen in a state during execution based on the current belief. If the state was evaluated based only on a single action, states which had similar rewards for that action but very different rewards for all other actions might be combined, with the consequence of possible suboptimal performance for policies created by a model aggregated in this manner. Dean and Givan's bisimulation method of state abstraction performs exact state aggregation in MDP models, combining states only when they have identical state transition and reward functions (Dean and Givan, 1997). While this method could be extended to POMDP models by using identical observation probabilities for states as an additional criteria, the requirement for equivalence leads to extremely conservative state abstraction.

This state aggregation method is based on the transformation of sets of physically identical states into a single state. The new state's state transition probabilities for each action are the average of that action's transition probabilities from each state of the original set, with the exception that all state transitions from one state in the set to another are replaced by self-transitions in the new state. Once the new transition matrix including the combined states is constructed, it is repaired so that any transitions into deleted states are redirected to the combined states that they were replaced by. The details of the algorithm are given in Table 7.1. The $combine()$ function compares the states according to some state value to decide whether they should be merged. The metrics used to evaluate states based on reward will be discussed in Section 7.3.

There are two ways of selecting which physically identical states should be compared for possible combination. One is to only consider states for combination if one state is a direct successor of the other (or of another state that has already been combined). This can be thought of as collapsing chains of physically identical states across timesteps. Physically identical states may also be considered for combination whether one is an immediate successor of another or not. It is possible that the same physical state can be arrived at from a number of different actions and parent states, which means that physically equivalent time-states may arise in non-contiguous parts of the state space. If a time-indexed model has this characteristic, it will be necessary to compare and possibly combine non-successor states in order to produce an aggregated model which is close in size to the non-time-indexed state space. For very large models, this may be necessary in order to find a solution in a reasonable amount of time. Combining non-successor states has a greater impact on the state transition structure than collapsing chains of successor states, because the next state transitions will lead to both of the (potentially non-contiguous) parts of the state space that the original states lead to. Combining such states could lead to some of the same representation problems that occur in the non-time-indexed representation, where physically identical states that occur at different points in the interaction (and may require different action choices to interact successfully) are combined.

This is why the reward-based state values are so important. Ideally, similar values suggest that the risk of treating the separate time-states as the same for the purpose of creating a policy is low. In the case where non-successor states may be combined, combining all physically identical states regardless of value will lead to a model that has the same number of states as the non-time-indexed model. If only successor states are compared, the model obtained by combining all physically identical states may still be significantly larger than the non-time-indexed model, depending on how frequently the same physical states appear at non-contiguous points in the time-indexed state space.

```
**Time-state Aggregation Algorithm**
$T_{new}$, a state-transition matrix, is initialized to all zeroes
$O_{new}$, an observation probability matrix, is initialized to all zeroes
$deleted$ is an empty dictionary of states and the state they are combined with

**Combine states**
for $t$ in 1:$T$
    for $s$ in $\mathcal{S}_t$
        if $s$ in $deleted$
            $s = deleted(s)$
        for $s'$ in $succs(s)$
            $combinestates(s,s')$

If also aggregating nonsuccessor states, then:
for $t$ in 1:$T$
    for $s$ in $\mathcal{S}_t$
        if $s$ not in $deleted$
            for $t_n$ in t:$T$
                for $s'$ in $\mathcal{S}_{t_n}$
                    if $s'$ not in $deleted$
                        $combinestates(s, s')$

function $combinestates(s, s')$
        if $comparestates(s, s')$
            for $a$ in $\mathcal{A}$
                $T_{new}(s, a, s) += T(s, a, s')$
                for $o$ in $\mathcal{O}$
                    $O_{new}(o, a, s) = O(o, a, s')$
            $deleted(s') = s$
        else
            for $a$ in $\mathcal{A}$
                $T_{new}(s, a, s') += T(s, a, s')$

function $comparestates(s, s')$
    if $physically\,indentical(s, s')$ and
      $compare(s, s', a) < threshold\;\forall a \in \mathcal{A}$
        return $true$
    else
        return $false$

**Repair state transition matrix**
for $s \in \mathcal{S}$
    if $s$ not in $deleted$
        for $a$ in $\mathcal{A}$
            for $s'$ in $succs(s, a)$
                $s'' = s'$
                while $s''$ in $deleted$
                    $s'' = deleted(s'')$
                $T_{new}(s, a, s'') = T(s, a, s')$

Remove rows and columns of deleted states from $T_{new}$ and $O_{new}$
Renormalize $T_{new}$ so it is a well-formed probability matrix
```

Table 7.1. Description of the time-state aggregation for an unspecified method of state value comparison.

7.3. Estimating State Values

When selecting states for aggregation, it would be best to combine states when identical beliefs would lead to the same actions being taken in those states under the optimal policy. However, there is no way to know if that is the case without solving the original model, which (if it could be done) would render an aggregation method unnecessary. Instead, if a relatively computationally inexpensive and reliable method of evaluating states could be devised, the state values could be compared in order to select states for combination. This idea is the basis of a number of state aggregation algorithms, as discussed in Section 3.3.

7.3.1. Future Reachable Rewards

The time-state aggregation method aggregates states based on state values that represent an estimate of each state's *future reachable reward*. This method of evaluating states provides a metric that summarizes all of the possible rewards that are achievable by acting from that state. This allows the value to represent both the best and the worst possible future outcomes. This approach is based on the insight that, because action outcomes for a state may vary over time, the future states (and therefore possible rewards) reachable from a physical state may vary from timestep to timestep. If these changes in next-state transition probabilities do not have a significant impact on what rewards are accessible to a state at later timesteps, they can probably be abstracted away without adversely effecting performance. But if these time-varying differences lead to different parts of the physical state space with different rewards, these time-states should be kept distinct in the problem representation.

The values for future rewards are calculated with respect to some policy for action. A POMDP policy is not very useful in this case for a number of reasons. A policy expressed in terms of belief states cannot be applied in a straightforward manner to acting in states without tracking beliefs unless unnatural simplifying assumptions are made (such as assuming that the belief is as concentrated on one particular state and choosing the corresponding action, for example). Treating the model as fully observable and using the optimal MDP policy ignores the fact that state uncertainty plays an important role in what the optimal action selection is in the true model. Using an MDP policy could result in ignoring the impact of actions that will actually be chosen in certain states by the POMDP policy. In fact, any policy that eliminates the possibility of certain actions runs the risk of ignoring actions that may be chosen by the optimal policy for the time-indexed model in that state.

Ignoring the effects of certain actions in certain states when evaluating states may cause states to be combined when the long-term effects of those actions are different and may have a significant impact on performance.

An agnostic way of evaluating action effects without being dependent on the form of the policy is to evaluate states based on a random policy that selects each action in each state with equal weight. The state values based on this policy are likely to be highly suboptimal, but that is by design. The purpose of these value estimates is to distinguish states that have different reward structure, either due to differing immediate rewards or due to having transitions which lead to different parts of the state space where differing rewards are available. The value of an action in a state under the random policy is the expected value of taking that action and then following a random policy from that point on. This approach guarantees that the value of taking an action in a state represents a combination of the rewards of all the states that can be reached after taking that action. The manner in which these rewards may be represented is discussed in the following section.

In order to calculate reward-based state values (such as when applying the Bellman equations for MDP models), it is typically necessary to iteratively recompute the state values based on a state's neighbors until the values converge. Because of the acyclic structure of the time-indexed state space, this iterative approach is unnecessary. Instead, the value for states at the last timestep are computed first and then the values are propagated backwards to states at the previous timestep. At each timestep, state values are calculated based on the state value of its successors and any immediate reward that may be received in that state. Because no time-state can transition to any states other than the ones in the next timestep, it is possible to compute all the state values during one backwards sweep from the last timestep to the first.

7.3.2. Evaluating State Differences

In order to evaluate the differences between states, a succinct way of representing the reachable future rewards as state value is necessary. Two different representations were investigated. One approach is to represent a distribution over the possible reward. This representation expressively captures the relative possibility of obtaining different rewards. In cases where rewards are sparse (such as when they are associated with relatively small sets of possible goals), such distributions are simple to represent. But for more complex reward structures, representing the full distribution over reward may be complicated. Another potential problem with using the full reward distribution as a measure of the reward is that random noise in the reward function could make states with similar or identical expected reward values appear very different. The full distribution representation assumes that there is no irrelevant variance in the reward function, which does not hold in all domains. In these cases, the expected reward may be used as an alternative that is both simple to compute and store and less sensitive to variance in reward. This less expressive representation cannot distinguish certain differences in reward that can be detected by comparing full distributions. But the flexibility of this representation makes up for its slightly reduced expressive power. Empirical results (to be discussed in Section 7.6.1) indicate that both methods performed similarly when used on models from the elevator riding domain.

7.3.2.1. Distributions of Reward. The distribution of reward for a state-action pair represents the probability of receiving rewards if taking that action in that state and then following a random policy afterwards. This gives information both about the range of possible future rewards and the relative proportion of future states from which they are achievable. Distributions can be compared to one another for similarity by calculating their Kullback Liebler (KL) divergence. The way the distributions are calculated and backed up as state and state-action values for the case of a small set of rewards given at the end of an episode is shown in Table 7.2. Early in the development of this aggregation method, the effect of excluding states from possible aggregation based on characteristics of individual distributions, such as entropy, were explored. The intuition behind the use of these metrics was that low-entropy states with similar distributions were ”safer” to combine than high entropy states, as the reward value that could be achieved was more predictable. However, no benefit was seen from making use of this additional information, so this approach was discarded.

In domains such as the elevator riding problem, where rewards are associated with end states that correspond to different interaction outcomes, it is easy to represent a discrete distribution over the probability of each possible reward. In cases where reward may take on a wide range of values, a different representation would be preferable, such as a histogram over ranges of continuous reward values or a parametric distribution. Also, if reward is given for events throughout the course of an interaction rather than being assigned at the end, it is necessary to update the distributions for state-action pairs with new rewards while backing up values, rather than just combining the reward distributions for future states. These updates are straightforward in the case where the reward is restricted to a small number of possible values, but for more general reward structures the

```
**Distributions with KL divergence (KL)**
Let the rewards take on one of $M$ values, $r_1$ to $r_m$
Let $V_s(j) = Pr\{R(s) = r_j\}$ for $s$ in $S_T$

for $t$ in $T$-1:*downto*:1
    for $s$ in $\mathcal{S}_t$
        for $j$ = 1 to $M$
            for $a$ in $\mathcal{A}$
                $V_{s,a}(j) = \frac{\sum\limits_{s' \in \mathcal{S}_{t+1}} T(s,a,s') * V_{s'(j)}}{\sum\limits_{M} \sum\limits_{s' \in \mathcal{S}_{t+1}} T(s,a,s') * V_{s'}(m)}$
            $V_s(j) = \frac{\frac{1}{|\mathcal{A}|} \sum\limits_{a \in A} V_{s,a}(j)}{\sum\limits_{M} \frac{1}{|\mathcal{A}|} \sum\limits_{a \in A} V_{s,a}(m)}$
$compare(s, s', a)$
        $KL(V_{s,a}(), V_{s',a}())$
```

Table 7.2. State value calculation and comparison using reward distributions and KL divergence.

updates could be more complicated depending on the method used to represent the reward distribution. In many cases, it would be more convenient to use a statistic describing the distribution of future rewards, rather than the distribution itself. This simpler representation would also have the benefit of not needing to adjust the representation of the distribution from problem to problem in order to ensure that it effectively represents the reward structure for that domain.

7.3.2.2. Expected Reward. Expected reward is the most commonly used statistic for evaluating reward in Markov models. The expected reward under the random policy is simple to calculate for arbitrary reward structures. The way that state values are backed up and compared using expected reward is shown in Table 7.3.

```
**Expected reward (ER)**
Let $V(s) = R(s)$ for $s$ in $S_T$

for $t$ in $T$-1:*downto*:1
    for $s$ in $\mathcal{S}_t$
        for $a$ in $\mathcal{A}$
            $V(s,a) = R(s,a) + \sum\limits_{s' \in \mathcal{S}_{t+1}} T(s,a,s') * V(s')$
        $V(s) = \frac{1}{|\mathcal{A}|} \sum\limits_{a \in A} V(s,a)$
$compare(s, s', a)$
        $abs(V(s,a)) - V(s',a))$
```

Table 7.3. State value calculation and comparison using expected reward.

The downside of this representation is that it is less expressive than storing a full distribution, so it may result in states being combined in cases where their future actions could lead to differing levels of reward. Consider the example shown in Figure 7.2. At timestep t, state S may choose one of two actions, each leading to a different state. In one of the successor states, one action achieves a reward of 15 and the other a reward of 5. In the other state, both actions receive a reward of 10. Because the state values are calculated based on future actions under the random policy, the expected reward for acting in these states is the same. Distribution-based state values capture that action b will lead to a reward of 10 with probability 1, while action a has a 50% chance of receiving either of the two possible rewards for the successor state. The actions that lead to each successor state are reversed at the next timestep, $t + 1$. Based on the expected reward state values, state S would be combined across these two timesteps because the value for each of its actions remains the same. If the state value is based on the full reward distribution instead, the different possible effects of the actions in each of the timesteps can still be distinguished. A possible way of distinguishing these situations would be to also represent the first few moments of the reward distributions for each state, such as the variance of the reward in addition to the expected reward.

The relative simplicity of this representation makes it attractive despite this loss of representational accuracy. The example given above is synthetic, and changes in the reward distribution that produce the same expected reward may not be a very common occurrence in realistic problem domains. Empirical results for the elevator domain indicate that the performance of this representation is comparable to that of the distribution-based representation. Based on these experimental results, an expected reward state value representation was used for the models in the Pittsburgh left domain. Further experiments in this domain confirmed that the expected reward statistic was effective in producing time-state aggregated models whose policies performed well relative to the original time-indexed model.

7.4. A Simple Example

Not representing time dependent action outcomes for a problem with time-dependent characteristics can result in policies that perform poorly. However, by using time-state aggregation, a model smaller than the time-indexed model can be created whose policies perform similarly to the policies of the original model. This is best illustrated through a brief example. For simplicity of explanation, a fully observable model (MDP rather than POMDP) will be used. Consider the models in Figure 7.3. The model on top consists of three states, S, S', and S'', whose action outcomes change over the course of three time

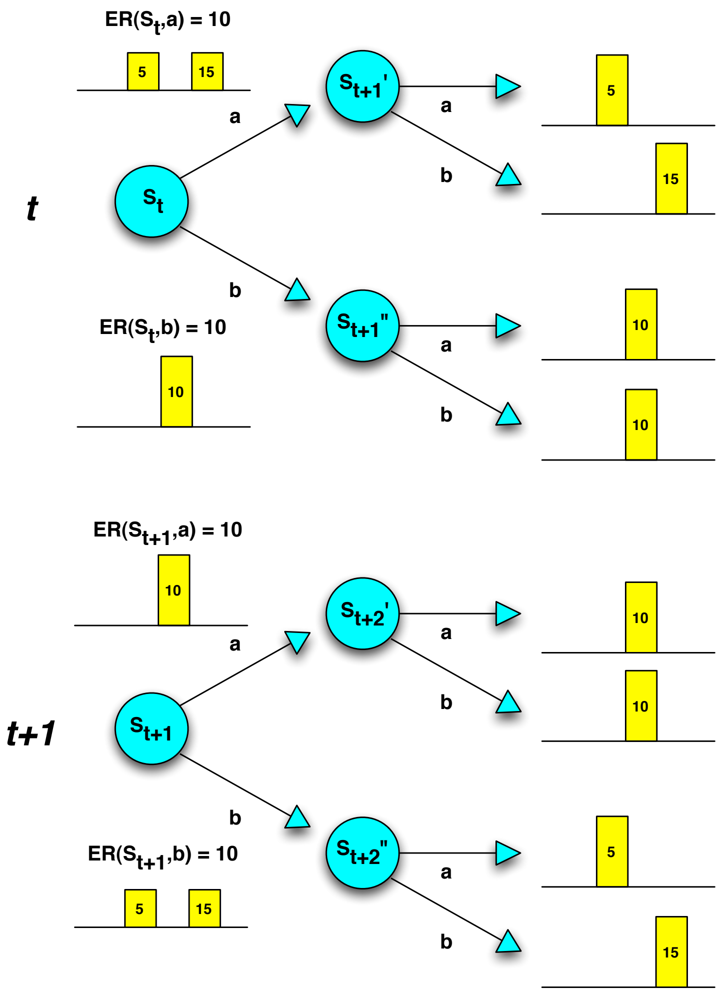


Figure 7.2. A time dependent transition difference that is distinguished by a distribution-based state value representation but not by an expected reward representation.

steps. The model on the bottom represents the same physical states and actions without the time index. States $S1$, $S2$, and $S3$ are equivalent to state S in the non-time-dependent model. Notice that the deterministic, time-dependent actions in the time-indexed model become non-time-dependent actions with probabilistic outcomes in the non-time-indexed model. The non-time-dependent model's optimal policy (always execute action a) is different from the optimal policy for the time-indexed model, which requires b to be the action chosen at the second timestep. The expected reward for state S (or $S1$) for the non-time-dependent model's policy is 10 when executed in either of the models. But the time-indexed model's optimal expected reward for state $S1$ is 50.

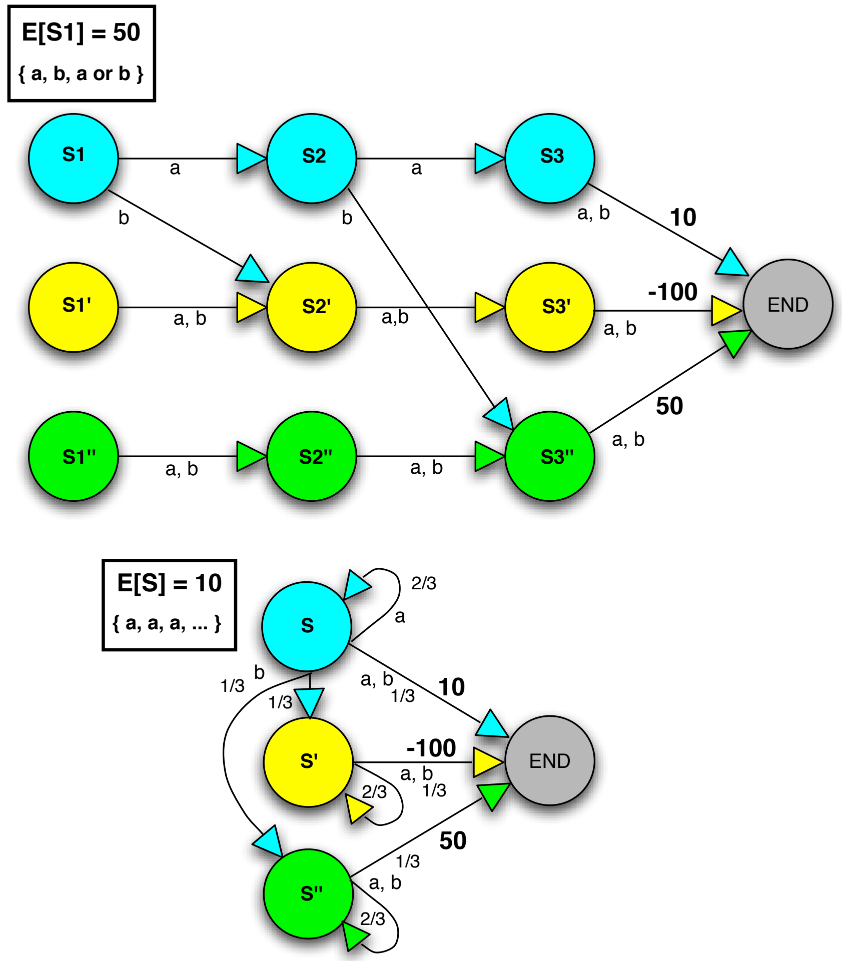


Figure 7.3. A model (top) and its equivalent model without a time index (bottom) and their optimal policies.

Expected reward state values and a state value comparison threshold of 1 are used to aggregate states of the time-indexed model from Figure 7.3 in Figure 7.4. Note that in this case the optimal policy and expected reward for this model are identical to the original's. It is not guaranteed that the optimal policy for a time-state aggregated model will be identical to that of the original model, or that it will achieve the same reward. But experimental results show that, in practice, these policies perform reasonably close to optimal compared with the original models at levels of aggregation that give significant speed-ups in solution time.

7.5. Determining a Useful Aggregation Threshold

The state aggregation threshold (used in the *comparestates* function from Table 7.1) can be adjusted to yield a desired reduction in state size. Both the reduction in size of

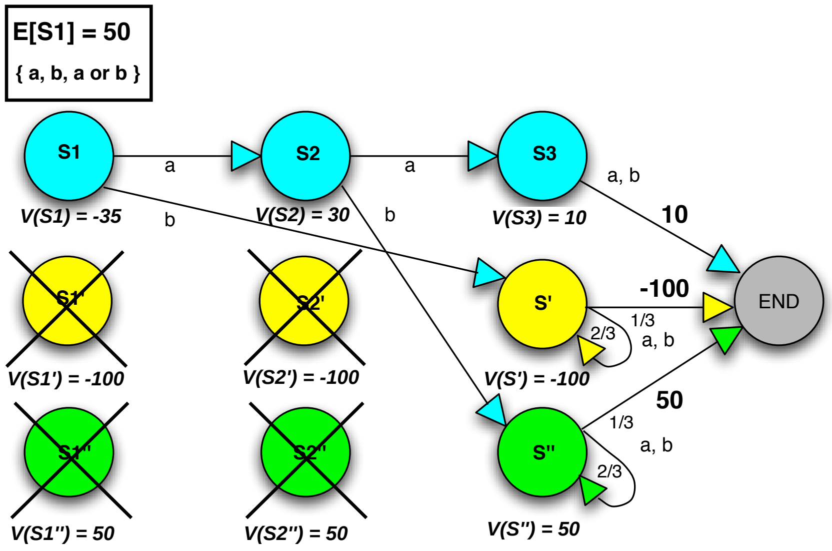


Figure 7.4. The model from Figure 7.3 (top), after time-state aggregation.

the model and the amount of time required to solve models of a certain size are highly dependent on the specific structure of the model. How does one select an appropriate threshold for aggregation? There is no way to calculate the threshold based on the loss of reward that one is willing to accept apriori. If one is capable of solving the full-time indexed model in a reasonable amount of time (so that one knows what reward can be obtained from the full POMDP), there is no reason to use an aggregated model. But models that are closely related to the time-indexed POMDP model may be solved to obtain approximate bounds on the performance of the time-state aggregated models. Using these bounds, one can test and identify a level of aggregation that gives acceptable performance within a tolerable solution time.

A lower bound can be found by solving a non-time-indexed version of the POMDP model. Note that it is possible to create a model without a time-index by running the state aggregation method on the time-indexed model with a threshold greater than the greatest possible difference between rewards. However, if data is added to the prior models, the state transition probabilities for the fully aggregated time-indexed model will most likely be different from the probabilities for a model created in the same manner as the time-indexed model (only without a variable representing time as part of the state space). This is because during the creation of the time-indexed model, the evidence from data in each time indexed state will be normalized before the states are combined during the aggregation process. In a non-time-indexed representation, all of the data corresponding to a state (regardless of the timestep at which it was observed) is added directly to that state prior to normalization. If it is possible to construct a model in this manner, rather than by aggregation, that model should be used instead as it more accurately represents the problem expressed in a non-time-dependent manner.

Note that when evaluating the non-time-dependent model, it is not sufficient to look at the expected reward of the optimal policy for that model. The policy should be simulated with observations and rewards arising from the original time-indexed model. Policies for time-state aggregated models should also be evaluated in this manner. This is because the time-indexed model is assumed to be the most accurate representation of the true time-dependent characteristics of the problem. The models without the full-time index are approximations, and should not be used for evaluation.

An upper bound can be obtained by solving an MDP version of the fully time-indexed model. This MDP model (or models) will provide an optimistic estimate of how well the task could be performed if the state were fully observable. It is important to note that the performance of the POMDP model, even without aggregation, typically will not actually be able to achieve the reward obtained by the MDP. Still, this provides a plausible upper bound on performance that can be calculated quickly relative to solving a POMDP model. Once these bounds are obtained, a starting threshold should be selected to create an aggregated model. As creating the model is cheap relative to solving it, it is recommended that a number of thresholds be tried until one that yields a promising level of state aggregation is found. In order to ensure a significantly reduced solution time, an aggregation threshold that results in a model with less than half of the number of states in the time-indexed model is recommended. Keep in mind that the aggregated model should also have more states than the non-time-dependent model if its performance is to be expected to be better.

After the model is solved, the policy obtained should be compared to the performance of the bounding policies. If the performance is acceptable, you are done. If not, the desired performance and the time required to solve the last policy should be used as guidelines to select a smaller threshold to create a larger model. This process should be repeated until a model with acceptable performance is found or until the time alloted to obtain a policy is expended. Note that while it is possible that several models may need to be solved, the time required to solve multiple smaller models will typically still be less than the time needed to solve the original time-indexed model.

7.6. Results

7.6.1. Elevator Domain Experiments

Time-indexed POMDP models were generated for 12 variations of the elevator riding problem ranging from around 4000 to 10,000 states in size. The thresholds for aggregation were varied to obtain varying levels of compression. Because both of the state comparison criteria (expected reward and KL divergence) performed similarly in this domain, only the results for KL divergence are reported, with the understanding that the same overall trend also applied when using expected reward. For a comparison of the number of states in the original and aggregated models, refer to Table 7.4. The "no time" models are the models in which the time index was completely removed from the state space. The non-time-dependent models were on average less than 10% of the number of states of their corresponding time-indexed models. All of the aggregation thresholds used also obtained a large reduction in state size, even for very small thresholds, suggesting that a significant number of the time-states in these models may be redundant or irrelevant.

Table 7.4. Comparison of the sizes of the state-aggregated models as percentages of the number of states of original models for the elevator domain.

Threshold	Avg %	(Min %, Max %)
KL 0.001	33	(20, 42)
KL 0.02	22	(12, 30)
KL 0.045	19	(10, 26)
No time	9	(5, 12)

The models were solved for their optimal policies using the ZMDP solver with the FRTDP heuristic (Smith, 2007b). The time to convergence of the policies (defined as the time at which the upper and lower bounds on the policy become approximately equal) are compared at various levels of compression in Figure 7.5.

The figure clearly shows that the time-state aggregated models converge to a solution considerably faster than the original. Other aggregation thresholds were also evaluated, and their performance followed the trend of this representative subset. Interestingly, the models at the intermediate aggregation level (KL 0.02) consistently converged the fastest. Policies for models with aggregation thresholds greater than KL 0.045 (these threshold values are not shown in the figure) converged more and more slowly until their performance matched that of the models without a time index, which failed to converge to an optimal policy within the 1,000 second time limit set for these experiments. The reason for this somewhat surprising result is most probably based on the structure of the original time-indexed models. In this synthetic example domain, the action outcomes are relatively

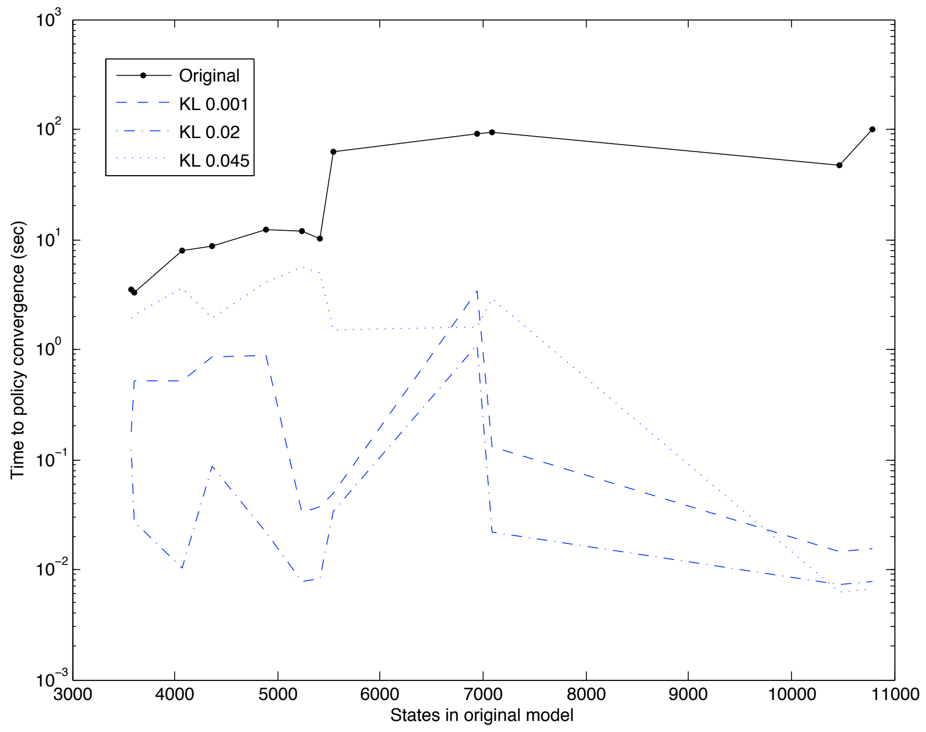


Figure 7.5. The relationship between the state aggregation threshold for merging states and their solution time for the elevator domain POMDP models

deterministic. When sets of states were combined, the state transitions out of each original state became state transitions out of the new state. The self-transition that the new state uses to represent the expected time in the state also increased the uncertainty of that action's outcome. At some point, the cost of the larger number of action outcomes and greater uncertainty outweighed the benefit of having fewer states in the model.

The policies from the time-state aggregated and non-time-dependent models were executed in the original time-indexed model for 1000 simulation trials each. All policies performed identically or very close to the policy of the time-indexed model. Because it could not be determined if the small variations seen in the rewards were due to differences in policy performance rather than variance in the simulation, these results are not shown. These similar results seem to be due to the relative simplicity of the elevator domain and the controller for the simulated people's behavior.

In order to determine if differences in performance between the time-dependent and the non-time-dependent models could be demonstrated in this domain, additional models were created and evaluated. These models were constructed to be more difficult versions of the task than the original dozen models, with a smaller number of timesteps given for

Table 7.5. Number of states and average reward over 1000 runs in simulation (with 95% bootstrap confidence intervals) for a set of elevator domain models.

Original		KL 0.02		No Time	
states	reward	states	reward	states	reward
292	15.0 [14.0,16.1]	252 86%	15.1 [14.0,16.2]	174 60%	9.6 [8.2, 10.9]
2701	39.9 [39.2,40.6]	1741 65%	39.6 [38.9,40.3]	1035 38%	37.1 [36.2,38.1]
3799	39.8 [39.0,40.5]	2484 65%	40.8 [40.1,41.5]	1494 39%	36.7 [35.8,37.7]
5141	39.8 [39.1,40.6]	2921 57%	40.1 [39.3,40.8]	1884 37%	36.9 [35.9,37.8]
5537	31.6 [30.8,32.5]	3823 69%	32.4 [31.5,33.2]	2931 53%	28.9 [27.9,30.0]
8933	32.2 [31.4,33.1]	6473 73%	32.0 [31.2,32.8]	5172 58%	29.5 [28.5,30.6]
8988	32.9 [32.1,33.8]	6170 69%	31.2 [30.3,32.1]	4372 49%	30.1 [29.1,31.0]
13053	32.4 [31.5,33.3]	8379 64%	32.3 [31.4,33.2]	6456 50%	29.8 [28.9,30.7]

the task to be completed. Table 7.5 shows the differences in performance of the policies obtained from a set of 40 compressed models between 200 and 40,000 states in size when they were run in simulation using observations and rewards generated from the original model. The 0.02 aggregation threshold for KL divergence was used because it led to the fastest convergence times overall for the previous set of models. The policies from the models compressed using this reward-based criteria consistently achieve optimal or near optimal performance (judged by comparing their policy's performance to that of the time-indexed model's) when acting in the environment of the original model in simulation. In fact, for the elevator problem models, policies from the compressed models at all compression levels performed close to the policy of the original model. Results were only reported for models where the difference between the largest and smallest average reward was greater than 2.0, because smaller differences than that seemed to be a product of the variability of the simulation runs rather than the performance of the policies.

Based on these results, it appears that the time-dependent representation does not improve significantly improve performance in this domain. It is possible that the similarity in performance across all model variants is the result of limitations in the design of the elevator problem. Because the control policy for the people in the simulator was extremely simple, the robot had relatively little opportunity to influence the behavior of the people through its own actions, and the aspect of the problem that was most dependent on time was the closing of the elevator door. A more realistic model of the interaction might better demonstrate the strengths of this method of modeling and compression, but it is difficult to construct an accurate model of human behavior without data of people performing the task. Even for problems where the difference in performance between the policies at various thresholds is not great, time-state aggregated models have the benefit of taking less time to solve than the original model. In this particular domain, the aggregated models for a large range of thresholds also converged significantly faster than the non-time-indexed models.

7.6.2. Pittsburgh Left Domain Experiments

Several time-state aggregated models were created from a version of the time-indexed POMDP models for the Pittsburgh left driving domain. These models were created from a somewhat different set of domain rules than the models that were later used in the human subject experiments. These models are large and complex, but the final models used in the second human subject experiment are larger still and take significantly longer to solve. The same domain rules were used to produce a naively sequential model with no time index in its state space. These models were solved to provide a grounds for comparison for the performance of the time-state aggregation method on the time-indexed models.

Information about the convergence time and characteristics of the polices computed for these models is shown in Table 7.6. The policies were solved for a regret bound of 5.0, and the average reward obtained by the policy over 100 simulation runs on the full time-indexed model was recorded (along with a 95% bootstrap confidence interval). The time-indexed model was initially solved using the FRTDP algorithm. However, in the course of solving the time-state aggregated models, the FRTDP algorithm frequently ran out of memory before reaching the final regret bound for many of the models. It is believed that the denser state transition structure of these models posed a problem for this algorithm because of the amount of information it stores during search. To avoid this difficulty, HSVI was used instead. The HSVI algorithm places less demands on memory, at the cost of exploring more states. The results for the full time-indexed models show that the HSVI algorithm was considerably slower than FRTDP. In fact, HSVI was terminated early for the full time-indexed models due to time constraints. The Pittsburgh Left model was terminated when the policy was at a regret bound of 6.15, and the Regular Left model's policy was terminated at a regret bound of 10.65. The reward results for the policies in simulation show that the impact on performance of this early termination was minimal.

The non-time-indexed models were solved for a regret bound of 5.0 using the HSVI algorithm. The reward results for these models show the effect of the time representation on the performance of their policies. For the Pittsburgh Left model, the policy for the model version without a time index performs far worse than the policy for the full time-indexed model. These results suggest that the time index allows the model to represent time-dependent characteristics of the interaction that are critical to acting successfully when the robot holds this intention. The rewards for the Regular Left models, on the other hand are fairly similar for both the time-indexed and non-time-indexed model. This suggests that a more accurate time representation is less important when the robot is acting according to this intention.

Table 7.6. Comparison of the size, solution time, and policy performance for the time-indexed and non-time-dependent model variants.

Model	Alg	States	Time	Alpha Vectors	Reward
Full Index (Pgh left)	FRTDP	64298	742032	2803	20.54 [18.53, 21.85]
Full Index (Reg left)	FRTDP	64298	1865400	5261	18.22 [16.06, 19.97]
Full Index (Pgh left)	HSVI	64298	3618380*	1716	20.75 [19.78, 21.61]
Full Index (Reg left)	HSVI	64298	3618800*	2966	17.75 [16.625, 18.87]
No Index (Pgh left)	HSVI	3714	764	333	-19.926 [-25.11, -14.42]
No Index (Reg left)	HSVI	3714	3411	528	15.555 [14.00, 17.08]

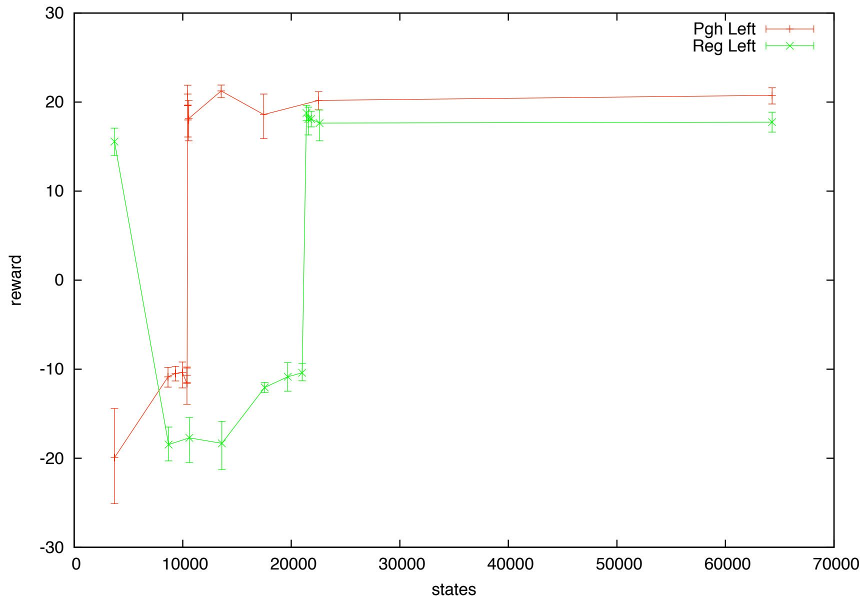
7.6.2.1. Aggregation of Nonsuccessor States. Both the model representing the Pittsburgh left agent intention and the regular left agent intention were aggregated using a number of thresholds using both the aggregation variant that combines nonsuccessor states and the variant that combines only successors. Expected reward-based state values were used for aggregation. The results for the nonsuccessor variant will be discussed first. Time-state aggregation was applied to both of the time-indexed models using aggregation thresholds of the integer values ranging from 1 to 5. Additional thresholds were used to create models for values where the reward obtained by the models underwent significant changes (the threshold level at which this occurred differed for the two models).

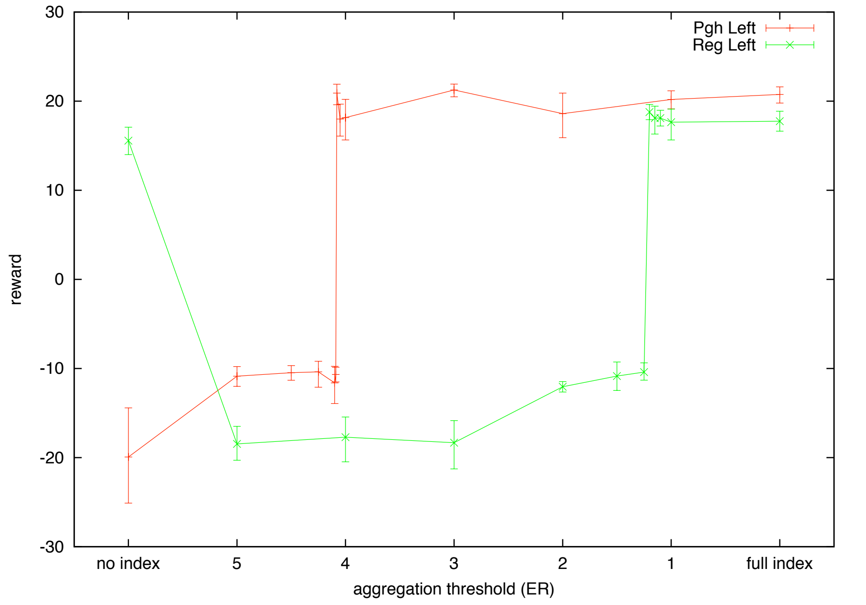
The relationship between the performance of the policies for the number of states in the model and the aggregation threshold are shown in the graphs in Figure 7.6. The reward was measured by executing the policies against the original, full time-indexed model for 100 simulation trials. The mean reward and 95% confidence intervals are shown. The results indicate that the reward obtained for both models are very similar to the reward for the original time-indexed version until the aggregated models became smaller than about 25,000 states. Both models could be reduced in size by over 50% without significantly degrading the performance of their policies. For both models, at some point, the reward obtained dropped dramatically over a very small range in aggregation thresholds. This seems to indicate that, at least for this domain, rather than the combination of more states having a gradual impact, there are certain states that, once combined, make it impossible for the aggregated model's policies to perform well in the environment defined by the original model. We will see when comparing results for the aggregation version that combines successor states only that this drop in performance is most likely due to combining non-successor states. The point at which this drop occurs is different for each model, both in terms of the number of states in the aggregated models and their aggregation threshold. This result is something important to keep in mind when using time-state aggregation. While the amount of aggregation can be set to any level by adjusting the threshold, the performance of the aggregated policies is highly dependent on the characteristics of the particular model.

The reward results for the Regular Left model demonstrate that in cases where the benefit of using a time-indexed representation is small, aggressive state aggregation has the potential to produce models whose policies perform poorly even relative to the policies of the non-time-indexed model. The Pittsburgh Left model provides an example of an interaction where representing time-dependent action outcomes has a significant impact on the reward. In this case, the policies for all of the time-state aggregated models created outperform the policy for the non-time-indexed model.

These experiments have demonstrated that time-state aggregated models can produce policies that perform similarly to the policies for the original model for a range of aggregation thresholds. But what is the effect of this state reduction on solution time? If the aggregation levels that achieve good performance do not also lead to significant reductions in solution time for the policies, then this approach is of limited utility. Fortunately, for the models considered in both the elevator domain and in the Pittsburgh left domain, aggregated models produce good policies that are also significantly faster to solve than the original time-indexed models. The graphs in Figure 7.7 show the time in seconds required to solve policies versus the number of states in the aggregated model and the aggregation threshold used. The models were solved using the HSVI algorithm, with a convergence threshold of 5.0 (the time reported for the full time-indexed model is the early termination time from Table 7.6). For the Pittsburgh left model, the smallest model with good policy performance (ER threshold 4.08) took just under thirty minutes to converge. The largest aggregated model tested (ER threshold 1.0) took about ten hours. The policy for the original model was terminated early after running for almost 6 weeks using the HSVI

Figure 7.6. Reward obtained by polices for time-state aggregated models in the Pittsburgh left domain, plotted versus the number of states in the models and the aggregation threshold used.





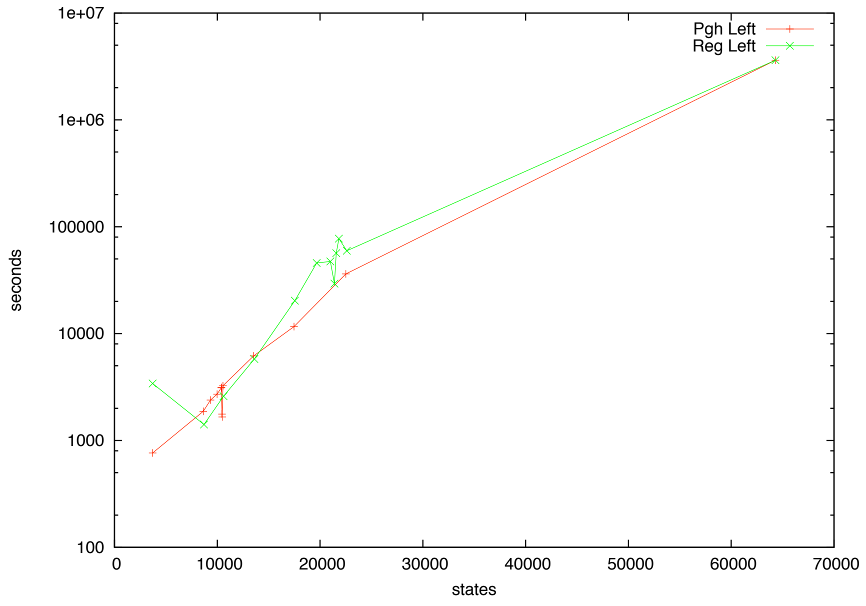
algorithm (using the FRTDP algorithm, it converged after 8 and a half days). The Regular left model's performance became worse for policies at smaller aggregation thresholds (as seen in Figure 7.6, rewards dropped for thresholds slightly larger than 1 while thresholds of up to 4 performed well for the Pittsburgh left model). It might be expected that this translated into smaller reductions in solution time. This is not the case. While the models took longer to solve than the Pittsburgh left models at the corresponding aggregation threshold, the original time-indexed model also took longer to converge than the original model for the Pittsburgh left case (though the time-indexed models were terminated after the same amount of time, the regret bound for the Regular Left model was significantly larger). The smallest model with good policy performance (ER threshold 1.2) took about 8 hours to converge. The largest aggregated model tested (ER threshold 1.0) took about 17 hours. Both of these policies performed slightly better than the non-time-indexed policy, which converged in a little under an hour. These convergence times may be surprisingly large considering the number of states even in the original model, but large number of observations (such as these models have) can make even relatively small POMDPs difficult to solve for a wide variety of POMDP approximation algorithms, not only the ones used for these experiments.

Interestingly, for both models, the solution times dipped at the last threshold value or two that still achieved good policy performance and then went back up for the models whose policies performed very poorly compared to the original model. This may reflect the fact that when the models are compressed beyond the point where the problem is well represented, the combination of certain states may create a model with a state transition matrix that makes the problem harder to solve in addition to being less accurate. This phenomenon may be related to (though less pronounced than) the saddlepoint in solution time versus aggregation level that was observed in the results for the elevator domain in Figure 7.5.

The relationship between the number of states and the aggregation threshold is shown in Figure 7.8. It can be seen from the graph that the amount of state aggregation that occurs at the various thresholds is very similar for both models. Therefore, differences in the amount of time required to solve the aggregated versions of the Pittsburgh left and Regular left models are due to differences in the structure of the models, not in differences in the state size of the models at the different aggregation levels.

This interpretation is reinforced by considering the number of alpha vectors in the policies plotted against the number of states in the models and their aggregation thresholds, shown in Figure 7.10. The number of alpha vectors needed to represent a policy can be

Figure 7.7. Time until policy convergence for time-state aggregated models in the Pittsburgh left domain, plotted versus the number of states in the models and the aggregation threshold used.



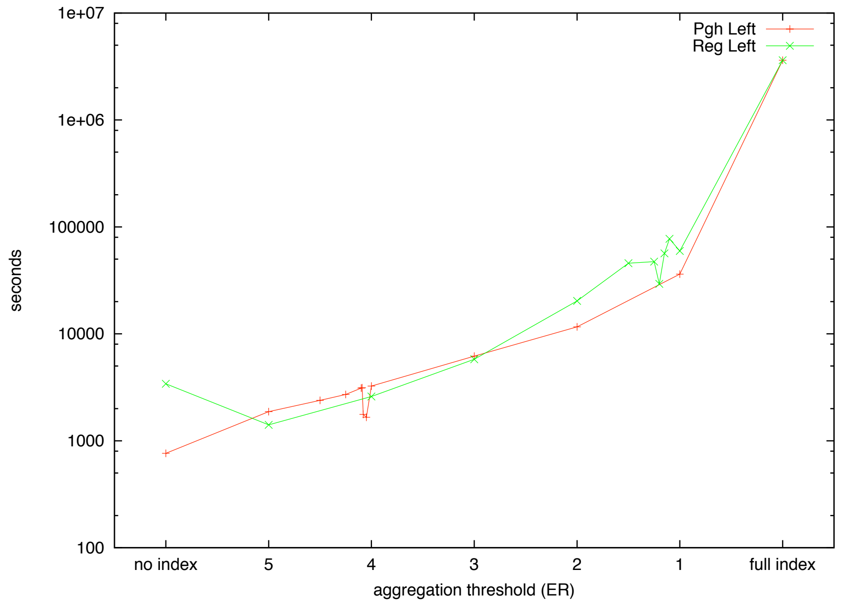
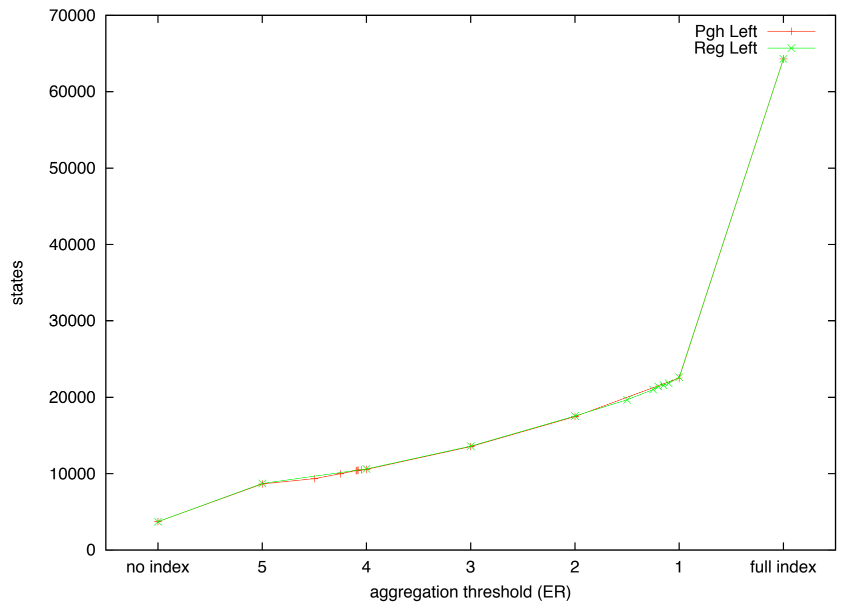


Figure 7.8. The relationship between aggregation threshold and state space size for time-state aggregated models in the Pittsburgh left domain.

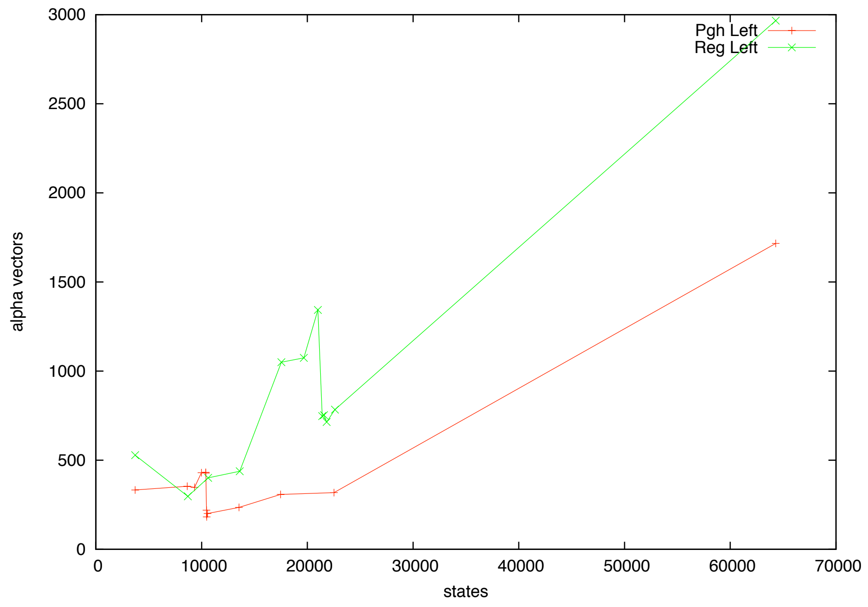


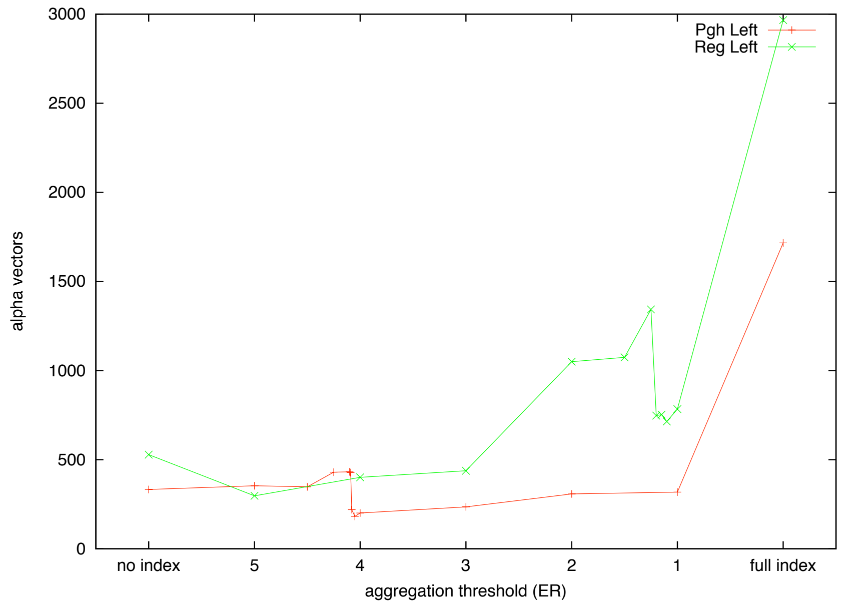
used as a measure of the policy's complexity. The Regular Left model, which took longer to solve than the Pittsburgh Left model for aggregation thresholds of 2 and smaller, also had policies with many more alpha vectors for these thresholds. It is interesting to note that for both models, the policies for the aggregation levels that achieve good performance all have a similar number of alpha vectors despite the differences in their number of states. At the aggregation levels where performance initially decreases, the number of alpha vectors actually increases. The increase in alpha vectors relative to the better performing aggregation levels indicates that these aggregated models were actually more difficult to solve despite their smaller size. This is likely because states were combined in a way that increased the uncertainty in the model. This increase in policy complexity explains the differences in solution time that occurred at the transition point from high performing to low performing policies as seen in Figure 7.7.

7.6.2.2. Aggregation of Successor States Only. In order to compare the performance of the aggregation method when only immediate successors are considered for aggregation, results for a few significant aggregation thresholds are shown in Table 7.7. The

Figure 7.9

Figure 7.10. The number of alpha vectors in the policies for time-state aggregated models in the Pittsburgh left domain, plotted versus the number of states in the models and the aggregation threshold used.





ER 1000 threshold is far greater than the largest difference between reward values, therefore the results for this threshold show the maximum amount of state aggregation that can be achieved for these models by combining only immediate successors. While the models produced are larger and slower to solve than many of the models created by combining nonsuccessors during aggregation, they are also much closer in performance to the original time-indexed model than many of them. This suggests that this state combination criteria is the more conservative choice, and may be a good alternative when combining nonsuccessors produces models whose policies differ greatly in performance from the original even at small aggregation thresholds. The policies of the ER 4 threshold models perform well for both the Pittsburgh left and Regular left models, unlike for the nonsuccessor method, where the Regular left model's policy performs very poorly at this aggregation level. However, for both models the nonsuccessor variant produces policies that perform similarly but take considerably less time to solve than the successor only variant. When faster solution time is the primary concern, it is preferable to use the nonsuccessor variant and fall back to using the successor only variant if models cannot be found that produce policies with good performance.

Table 7.7. Comparison of the sizes, solution time, and policy performance of time-state aggregated models in the Pittsburgh left domain where only successor states were considered for combination.

Model	States	Time(s)	Alpha Vectors	Reward
ER 1000 (Pgh left)	20339	24830	224	20.91 [19.50, 21.94]
ER 1000 (Reg left)	20344	31122	305	16.88 [14.79, 18.47]
ER 4 (Pgh left)	29623	73290	420	20.40 [18.69, 21.84]
ER 4 (Reg left)	29690	120660	740	19.09 [17.30, 20.44]

7.7. Conclusion

This chapter elaborated on the idea of modeling time-dependent action outcomes using fully Markov models with a time-indexed state space. In order to avoid the potentially great increase in solution time that comes with adding a time-index to the state space, a state aggregation method was developed that acts exclusively on the time dimension to remove redundant or irrelevant states. Time-state aggregation uses state values based on a state's future reachable rewards to select physically identical states at different timesteps for aggregation. The amount of aggregation performed can be adjusted by varying the threshold used for state value comparison. The state value representations explored were the full reward distribution and expected reward. In empirical results from two domains, time-state aggregation proved to successfully produce models whose policies converged significantly faster than, and performed comparably to, the original time-indexed models.

In the elevator domain, time-state aggregation was applied to a set of models at varying aggregation thresholds using distribution-based state values and the results were compared to the time-indexed and non-time-dependent versions of these models. The time-state aggregated models' policies converged faster and performed comparably to the time-indexed models' for a wide range of aggregation thresholds. In this domain, the aggregated models were also quicker to converge than the non-time-dependent models, due to the increased uncertainty in the models without explicit time representation in the state, and achieved higher reward in the cases where the reward obtained by the non-time-dependent policies were less than that obtained by the time-indexed policies. For many models tested, the reward results did not differ. It is believed that this is due to the relative simplicity of the time-dependence in this synthetic example.

In the Pittsburgh left domain, large models of a time-dependent task that included human data were evaluated. Time-state aggregation was applied to both models for a number of aggregation thresholds using expected reward state values. Over a range of thresholds, the time-state aggregated models' policies achieved similar rewards and converged far faster than the policies for the time-indexed models. The performance of these policies were also far superior to those for the non-time-dependent model for the Pittsburgh Left interaction model, indicating that representing time-dependent structure was important for this intention and that time-state aggregation preserved this structure. The performance benefit for the Regular Left interaction model was less pronounced, and restricted to a smaller range of aggregation thresholds. This result suggests that modeling time-dependence is not critical to acting effectively in most situations where the robot holds this intention. The effect of aggregating only successor or nonsuccessor time-states on solution time and policy performance was investigated. A method for selecting an effective aggregation threshold by using the performance of less expressive model variants as loose bounds was also discussed.

CHAPTER 8

Human Subject Experiment

8.1. Experiment Design

IN order to evaluate the relative benefits of the representation of time dependence and reasoning about human intention, the effects of these representational choices were evaluated through the creation of control policies for a realistic situated social task. An experiment was conducted to compare the performance of policies created from models built according to these principles and similar less expressive models. This experiment was closely related in structure and implementation to the data collection experiment described in Chapter 5. In this experiment, a robot car controlled by these policies performed the role of the turning driver, and a human driver took on the role of the car driving straight, using the same simulated driving environment as in the earlier experiment (Chapter 5) .

The experiment had a 3x2x2 factorial incomplete repeated-measures design, with the model variant, the human's intention, and the robot's intention as the independent variables. The dependent variable was the reward achieved by the robot during an interaction (as specified by the reward structure for the models). Additionally, the same paper survey given to participants in the data collection experiments was given to the participants in this experiment after driving with the controller for each model variant. These subjective responses were analyzed to determine which model was considered to have produced the most natural and socially appropriate behavior.

8.1.1. Models

In order to test the experimental hypothesis about the importance of the representation of time-dependence and human intention, three variants of a model based on the same rules and human data were produced. Four MDP models that assumed a fixed intention for the human driver, one for each combination of human and robot driver intentions, were

Table 8.1. Model and policy information for experiment POMDP models.

Model	Intention	States	Alpha Vecs	Time (s)	Reward
time POMDP	Pgh Left	37982	879	154592	19.6 [19.1,20.1]
time POMDP	Reg Left	37430	2152	219484	19.0 [18.3,19.5]
no time POMDP	Pgh Left	4398	281	1485	8.5 [7.9,9.0]
no time POMDP	Reg Left	4398	635	3924	15.0 [14.1,16.0]

created. These MDP models had a full time index in their state space. Two time-indexed POMDP models, one for each robot intention, were also created. These POMDP models, with 72,024 states each, were too large and complex to solve within a reasonable amount of time, so time-state aggregation was used to produce smaller, practically solvable time-dependent POMDP models. Two non-time-dependent POMDP models were also created without explicit time information in their state space.

The models used in these experiments were created from a revised version of the model description files used to produce the models evaluated in Section 7.6.2. Revisions were made after preliminary testing in the driving simulator identified cases for which action rules needed to be added or revised to produce reasonable behavior. These POMDP models have 4 actions and 724 observations. For the time-dependent POMDP model, a time-state aggregation threshold of 4.0 was used and only successor states were considered for aggregation. While it is most likely possible that a larger threshold or aggregation that considered nonsuccessor states could have produced a smaller model with similar performance, this amount of aggregation was deemed sufficient to produce a solution in an achievable time frame while being conservative with respect to the amount of aggregation. This decision was made in order to give the aggregated time-dependent model the fairest comparison possible to the time-indexed model used for the MDP. The models were solved using the HSVI algorithm with a convergence threshold of 5.0. The size of the models, the size of their resulting policies, and their convergence times are shown in Table 8.1. Reward results for simulation of 1000 trials against the time-indexed POMDP are also shown for each policy, with 95% bootstrap confidence intervals. The non-time-dependent policies perform noticeably worse, indicating that an explicit representation of time is necessary to capture aspects of the problem that enable successful performance of the task.

The time indexed MDP models were solved using the FRTDP algorithm with a convergence threshold of 0.1. A different algorithm (which cannot practically be used for the POMDP models) and tighter convergence threshold were used to realistically reflect the

Table 8.2. Model and policy information for experiment MDP models.

Robot Intention	Human Intention	States	Time (s)	Reward Bound
Pgh Left	Yield	36152	862	[21.5, 21.6]
Pgh Left	No Yield	35876	684	[19.5, 19.6]
Reg Left	Yield	36152	1128	[19.4, 19.5]
Reg Left	No Yield	35876	670	[21.9, 22.0]

advantages of using a simpler, less computationally expensive representation. The number of states in each model, their policy convergence times, and their solution bounds are given in Table 8.2.

8.1.2. Controller Implementation

During execution, the robot car communicates with a policy execution program using named pipes. Within the simulator, variables of interest are discretized using the boundaries specified during model design. These discrete values are used to construct the observations and states passed to the policy execution software. At the start of each trial, the policy execution software is passed a message announcing the beginning of a new trial, along with the policy that should be used for execution of that trial. Every half second, a new message is passed to the execution software. This message consists of an observation (or the state including the assumed intention for the human, in the MDP case) and the true state of the simulator to be logged for later comparison. The policy execution software passes the actions chosen by the policy back to the driving simulator. In the driving simulator, a simple PID controller is used to translate the model actions into low level controllers to the steering, throttle, and brake.

The MDP policies were represented as state-action lookup tables. The POMDP policies, however, were represented as alpha vectors, which meant that belief tracking had to be performed during execution in order to select an action. Policy execution was performed using the ZMDP software (Smith, 2007b). Most of the time, the half second control loop was more than adequate for belief tracking and action selection. However, an intermittent problem occurred when executing the POMDP models that caused the policy executor to not return an action selection for several seconds at a time. Whenever this occurred, synchronization was lost between the policy execution software and the driving simulator. Because this problem was always limited to the last few timesteps of the trial and the source of the error couldn't be determined, the decision was made to cut off the control after the 55th timestep (out of 60), or the last 2.5 seconds of interaction. After this point, the low level controller drove according to the last high-level action it had received from the policy executor. While it was not necessary to do this for the MDP policy, the same cutoff was applied so that the policies would be compared fairly.

8.2. Experiment Procedure

Experiment subjects were recruited using the same methods as for the data collection experiment. As before, the participants were pre-screened to assure that they were familiar with the Pittsburgh left. Thirty-five participants performed the experiment over the course of two weeks. Of these participants, five were eliminated from consideration, four for software problems which occurred during the course of the experiment that may have invalidated their results and one for failure to follow directions. Data from the remaining thirty subjects were used for analysis.

Subjects were seated at a workstation (one of the two used in the previous experiment). Prior to beginning, participants were briefed on use of the controllers and the experiment's overall structure. They were told that they would be driving with three different opponent drivers during the experiment, that the behavior of these drivers may differ from one another, and that they were to give their impressions of their interaction with each driver by filling out a short survey after completing each set of trials. They were informed that it was possible that their assigned intention about yielding might conflict with the other driver's, and that if that seemed to be the case they should behave as they would in a real driving situation. They were also instructed to respect the traffic light and to try to avoid collisions with the other car.

After being briefed, drivers engaged in a short training session of four trials to familiarize themselves with the controls. This training time had half the number of trials of the training for the first experiment, in which the subjects trained to perform both roles rather than just the role of the straight driver. During the training, the other car stayed stationary at its start position. The experiment began once the driver asked the experiment administrator any questions they had and confirmed that they were ready.

The experiment was divided into three sets of eight trials each. During each set, the driver interacted with a robot car controlled by policies from one of the three model variants tested: the time-dependent POMDP models, the time-indexed MDP models, and the non-time dependent POMDP models. The order in in which the human driver encountered each policy group was randomly selected at the start of the experiment. As in the data collection experiment, the robot's and driver's intentions were assigned to them at the beginning of a trial. Each possible combination of intentions for the two roles was experienced twice, in a random order determined at the beginning of the set of trials. For the POMDP controllers, the policy corresponding to the robot's intention for that trial was used for execution. Because the MDP models were separate for each combination of robot and human intention, the model corresponding to a given human intention may or may not have corresponded to the intention assigned to the human for that trial. The human's intention was selected randomly. After a set of trials was completed, the subject had two minutes to fill out the survey before beginning the next set.

8.3. Data Analysis

Statistical analysis was performed on the reward results and the responses to the paper survey. The comparisons made were planned contrasts of non-orthogonal data because the time-dependent POMDP results were compared to both the MDP results and the non-time-dependent POMDP results. The sequential Dunn-Sidak adjustment for k comparisons ($k = 2$ in this case) was used to adjust the p value required to reject the null hypothesis for the statistical tests making pairwise comparisons between the time-dependent POMDPs and the other models. Using this method, the smallest of the p values must be smaller than the smallest adjusted p value threshold, or no other results with larger p values can be accepted as significant. This criteria holds for all tests up to k. The adjusted p values for $\alpha = 0.05$, computed according to their formulas, are:

$$\begin{aligned} p_1 &= 1-(1-\alpha)^{\frac{1}{k}} \\ &= 1-(1-0.05)^{\frac{1}{2}} = 0.02532 \\ p_2 &= 1-(1-\alpha)^{\frac{1}{k-1}} \\ &= 1-(1-0.05)^{\frac{1}{2-1}} = 0.05 \end{aligned}$$

An analysis of variance (ANOVA) was conducted to compare the mean reward obtained by the groups of policies for each model variant for all combinations of intentions for both human and robot. Because the hypothesis of this experiment concerns the performance of the time-dependent POMDP model versus the other models, the *post-hoc* tests used are planned contrast t-tests of the time-dependent POMDP with each of the other model variants. In cases where the elements of the interaction cannot be expressed in terms of these planned contrasts, Tukey's HSD (Honestly Significant Differences), a *post-hoc* test that partitions the means into groups based on a more conservative test of statistical significance, is used instead.

The paper survey administered was the same survey used for the experiment in Chapter 5. The responses to the survey were a level of agreement with the statements presented,

Table 8.3. Tests of Effects

Source	Sum of Squares	df	F	p value
Model	33837.42	11	18.2124	<0.0001*
Error	119548.07	708		
C. Total	153375.48	719		
Group	2341.040	2	6.9322	0.0010*
Robot Intention	3608.049	1	21.3680	<0.0001*
Human Intention	76.048	1	0.4504	0.5024
Group*Robot Intention	4819.241	2	14.2705	<0.0001*
Group*Human Intention	3216.261	2	9.5238	<0.0001*
Human Intention*Robot Intention	12826.437	1	75.9621	<0.0001*
Group*Human Int.*Robot Int.	6940.339	2	20.5514	<0.0001*

expressed on a Likert scale. As before, the responses were treated as ordinal and the median was chosen as the statistic for measurement. Ninety-five percent bootstrap confidence intervals were also reported. Planned comparisons were made between the time-dependent POMDP trial and the time-indexed MDP trial and between the time-dependent POMDP trial and the non-time-indexed POMDP trial, as for the reward results. Because of the potential variability in people's subjective responses, a matched pairs test was used to evaluate the relative preferences for one model over another for the group of experimental subjects. The Wilcoxon signed rank test , a non-parametric test analogous to the paired t-test, was used to determine statistical significance in the cases where the median responses differed.

8.4. Reward Results

The F-test results for all main effects and interactions of the ANOVA are given in Table 8.3. It should not be very surprising that virtually all of the effects were significant, given different combinations of intentions resulted in very different interactions between humans in the data collection experiment. These results also support the hypothesis that differences in representation lead to very different behavior by the policies. The ANOVA results for the reward will be examined and discussed in this section to identify the sources of variation among the groups of policies of the different model variants. The reasons for these differences in reward will be more closely investigated when the outcomes and events observed during the experiments are examined in Section 8.5.

Table 8.4. Mean rewards for Groups of policies by model variant with planned contrast posthoc t-tests.

Group	Mean	t Ratio	p value
time POMDP	15.873		
full time MDP	11.937	t = 3.319	p = 0.001* ($< p_1$)
no time POMDP	12.171	t=3.121	p = 0.002* ($< p_1$)

Table 8.5. Mean rewards for the intention of the robot and human with significance results.

Robot Intention			
Pgh left	Reg left	t Ratio	p value
11.088	15.565	4.623	< 0.0001*
Human Intention			
No yield	Yield	t Ratio	p value
13.002	13.650	0.671	0.5024

8.4.1. Main Effects

The primary purpose of measuring the reward obtained by the policies in this experiment was to test the hypothesis that modeling human intention as hidden state and representing time-dependent action outcomes would result in better performing policies than those of models that lacked these representations. This hypothesis was supported by the results for the "Group" main effect, as shown in Table 8.4. On average, the polices for the time-dependent POMDP model outperformed the policies for both the time-indexed MDP and non-time-dependent POMDP models. The *posthoc* t-tests confirm that these differences were very statistically significant. This is the key result for the reward-based portion of the experiment analysis. Further discussion of the results will examine the similarites and differences among the policies for the model variants and the conditions under which they succeeded or failed at achieving good performance.

The mean rewards for the other two main effects, "Robot Intention" and "Human Intention", shown in Table 8.5, give insight into the aspects of the problem domain that proved most difficult for all model variants. The mean reward for trials during which the robot intended to take the Pittsburgh left was lower than for trials in which the robot intended to yield and turn left after the human driver crossed the intersection. Attempting the Pittsburgh left often results in more complicated interactions (with more risk of collision) than yielding does, so this result is not surprising. The human's intention, however, did not have a significant effect on the mean reward when considered in isolation. An examination of higher level interaction effects will yield more insight into the role humans' intentions played in the rewards obtained.

Table 8.6. Tukey's HSD posthoc test results for the Group x Robot Intention interaction.

Group	Robot Int.	Mean		
POMDP	Pgh left	17.257	A	
MDP	Reg left	16.436	A	
NT POMDP	Reg left	15.770	A	
POMDP	Reg left	14.490	A	
NT POMDP	Pgh left	8.571		B
MDP	Pgh left	7.437		B

8.4.2. Two-way Interactions

The interaction between the robot's intention and the model variant that produced the polices for its controller are shown in Figure 8.1. This interaction shows one of the major differences in performance between the model variants. The groups of means judged to be significantly different from one another given a Tukey's HSD posthoc test are presented in Table 8.6 (each letter in the table represents a group of values that the test determined to be significantly different from the values outside of that group). Both the MDP and the non-time dependent POMDP model produced policies that performed significantly worse than the time-dependent POMDP model when the robot had the intention to take the Pittsburgh left. The time-dependent POMDP's policy actually performed slightly better in this case, though this difference in performance was not statistically significant. These results suggest that only the time-dependent POMDP model was capable of representing the interaction in a way that produced high quality policies for both possible robot intentions.

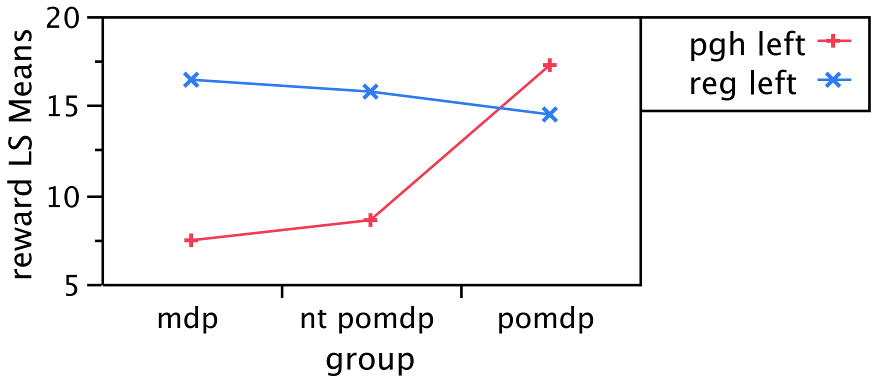


Figure 8.1. Interaction of model variant policy groups with robot intention for mean reward.

Table 8.7. Tukey's HSD posthoc test results for the Group x Human Intention interaction.

Group	Human Int.	Mean			
POMDP	No yield	17.690	A		
MDP	Yield	15.139	A	B	
POMDP	Yield	14.058	A	B	
NT POMDP	No Yield	12.581		B	C
NT POMDP	Yield	11.760		B	C
MDP	No yield	8.735		B	C

The interaction between the human's intention and the model variant that produced the polices for its controller are shown in Figure 8.2. This interaction reveals that there were differences between the POMDP and the MDP models in how successfully they responded to the humans' intentions. Though the time-dependent POMDP performed relatively well whether the human intended to yield to them or not, its performance was significantly better when the human did not intend to yield. The MDP model performed far worse in the case where the human intended to yield, the reverse of the relationship between intentions seen for the time-dependent POMDP case. The non-time-dependent POMDP performed similarly regardless of the human's intention. The groups of means judged to be significantly different from one another given a Tukey's HSD posthoc test are presented in Table 8.7.

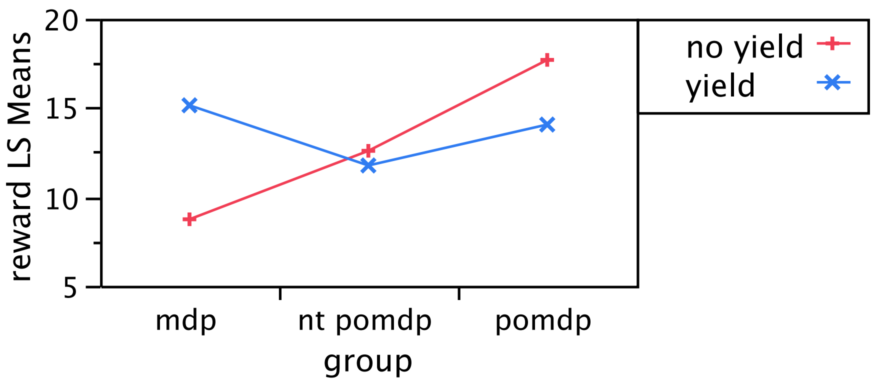


Figure 8.2. Interaction of model variant policy groups with human intention for mean reward.

Results for the interaction between the human's intention and the robot's intentions are shown in Figure 8.3. Tukey's HSD found all of the means to be statistically significantly different from one another. This interaction shows that the performance of the policies differed for each combination of human and robot intentions in a complex way. The performance of each group of policies for each of these combinations will be explored in the next section.

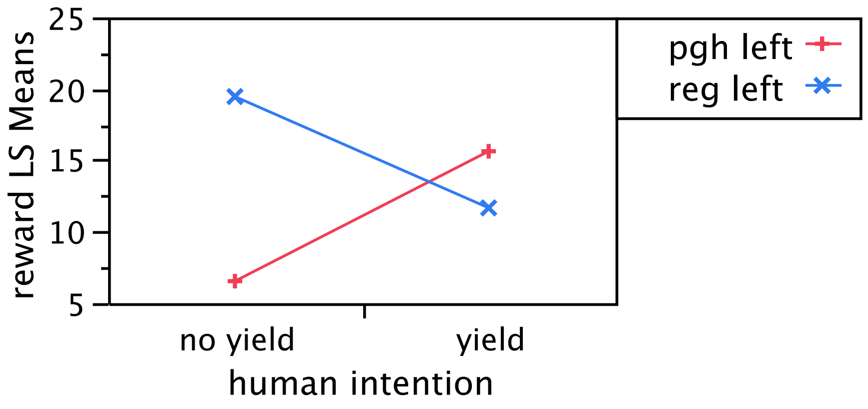


Figure 8.3

8.4.3. Three-way Interaction

The rewards for each combination of intentions grouped by model variant, are shown in Figure 8.4. At this level of analysis, the source of the differences in performance between the groups of policies begins to become clear. For certain combinations of intentions, the performance of the model variants were very similar, while for others they were radically different.

The mean reward for each group, with t-tests for significance of the differences, are reported for each combination of intentions in Table 8.8. These differences, when they occur, are due to what representation is necessary in order to perform that particular interaction successfully. For the case where the human intended to go first through the intersection and and robot intended to yield (Regular Left, Yield), all of the policies performed similarly well. This is most likely because this is a relatively uncomplicated interaction where all that is typically required in order to coordinate behavior with the human is to wait until they have cleared the intersection.

In the case where both the human and the robot intended to yield to one another (Regular Left, Yield), all of the model variants also performed similarly. They also all performed worse than in the previous case. This suggests that none of the model variants were

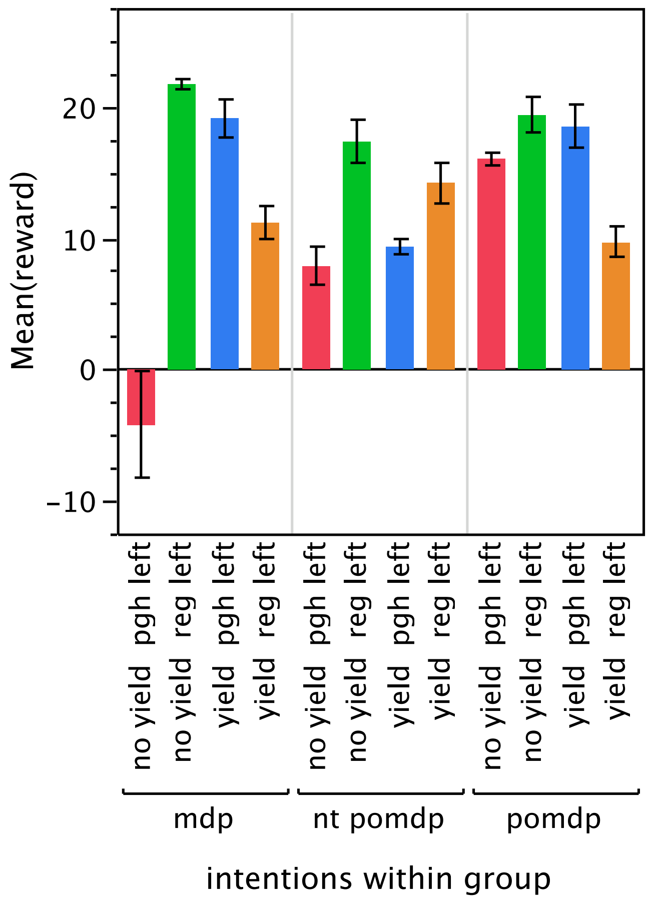


Figure 8.4. Rewards with standard error for each combination of robot and human intentions, grouped by the model variant of the robot's policies.

as successful as they could have been at negotiating this interaction, which indicates that this failure may have been caused by an aspect of the model or effect of policy execution that was common to all representations. This possibility will be explored in more detail in Section 8.5.

For the case in which the robot intended to take a Pittsburgh Left and the other car intended to yield (Pittsburgh Left, Yield), the policy for the time-dependent POMDP outperformed the policy for the non-time-dependent POMDP by a significant amount. The performance of the MDP policies were similar to that of the time-dependent POMDP. This result indicates that time representation was critical to the success of this particular interaction. The MDP policies seemed to perform well in this case whether or not their assumption about the intention of the human participant was correct.

Table 8.8. Reward and posthoc t-test results for each combination of robot and human intentions.

Reg left, No Yield			
Group	Mean	t Ratio	p value
POMDP	19.344		
MDP	21.708	-0.997	0.319
NT POMDP	17.331	0.848	0.397
Reg left, Yield			
Group	Mean	t Ratio	p value
POMDP	9.637		
MDP	11.164	-0.644	0.520
NT POMDP	14.208	-1.927	0.0544
Pgh left, Yield			
Group	Mean	t Ratio	p value
POMDP	18.478		
MDP	19.113	-0.268	0.789
NT POMDP	9.311	3.864	0.0001*
Pgh left, No Yield			
Group	Mean	t Ratio	p value
POMDP	16.035		
MDP	-4.239	8.545	<0.0001*
NT POMDP	7.832	3.458	<0.0001*

The case where the robot intended to take the Pittsburgh left and the human did not intend to yield (Pittsburgh Left, No Yield) is the most interesting one in terms of the differences between the groups. The time-dependent POMDP performed considerably better than either of the other models. This result suggests that both time-dependence and reasoning about people's intentions played a role in the policy's success. While both of the other model variants performed worse, the performance of the MDP models' policies were far worse than any models for any other combination of interactions. Though this difference in representation did not have an impact in the other cases, in this case it seems to have produced a catastrophic failure in interaction. This will be considered further in the next section.

8.4.4. MDP model results

Rather than planning for both of the human's possible intentions in one policy as in the POMDP models, the MDP policies were created from separate models for each possible human intention. Which policy to use with respect to the human's intention was randomly selected at the beginning of the trial, causing the robot to effectively "assume" that the human's behavior would correspond to the selected intention and act accordingly. In order to understand the MDP model variant's failure in the (Pgh left, No Yield) case, the trials

Table 8.9. Reward for the MDP model policies, divided into trials in which the model's intention did and did not match the human's intention, with t-tests of the mean differences.

Pgh left, No Yield			
Match	Mismatch	t Ratio	p value
-26.992	15.670	-11.89	<0.0001*
Pgh left, Yield			
Match	Mismatch	t Ratio	p value
18.707	19.404	-0.192	0.848
Reg left, No Yield			
Match	Mismatch	t Ratio	p value
22.486	21.226	0.342	0.733
Reg left, Yield			
Match	Mismatch	t Ratio	p value
11.205	11.133	0.020	0.984

in which the robot's assumption matched the human's intention and the trials in which they were a mismatch must be considered separately. The reward for the other cases were also compared in order to determine if whether the robot's assumption was correct had an impact on performance. These comparisons are presented in Table 8.9.

The only statistically significant difference between the reward obtained for trials in which the policy's model held the correct assumption about the human's intention and the trials in which there was a mismatch between the assumed intention and the true intention occurred in the (Pittsburgh Left, No Yield) case. Surprisingly, the policies with the correct assumption performed worse. This difference in performance is caused by an unexpected consequence of using human task performance data in the model. For the model where the robot intends to perform the Pittsburgh left and the human does not intend to yield, there were numerous data traces added to the model where both of the cars drove into the intersection and narrowly missed colliding with each other because one of the cars stopped in time. Recall that the robot's controller issues new action commands only twice a second, and that the overall ability of the robot to steer and modulate its speed are at a coarse resolution compared to the human's continuous control. The addition of human data to the model gave the robot an overly optimistic prediction of its ability to take last-minute maneuvers to avoid collision. For the model where the robot intended to take the Pittsburgh left and the other driver intended to yield, the vast majority of the data traces had the human wait before the intersection while the turning car made the left. When this model (of the incorrect intention) was used to interact with a human driver that did not intend to yield, the human's actions of moving into the intersection took it into a part of the state space where there was little, if any, human data in the model. This meant that the robot's actions were determined by the predicted chance of collision according to the prior model, which was pessimistic about the chance of collision relative to the human data. This resulted in more cautious behavior (the robot's car stopped when the human's entered the intersection) that successfully avoided collisions.

Model inaccuracy is always going to be a potential issue when doing model-based planning in any realistic domain. The policies for the POMDP model, because they planned over a belief space in which either intention for the human driver was possible, were more robust to these model inaccuracies.

8.4.5. Summary of results

Overall, the policies for the time-dependent POMDP model variants performed significantly better than either of the other modeling representations. All of the model variants performed similarly to one another in the cases where the robot intended to take a regular left turn after the human crossed the intersection. However, the interactions in which the robot intended to take the Pittsburgh left showed the superiority of the time-dependent POMDP representation. The non-time-dependent POMDP policies performed worse in both of the cases involving the Pittsburgh left intention, suggesting that it was not possible to produce quality policies for this intention without representing time-dependent action outcomes.

The MDP policies performed significantly worse than the time-dependent POMDP policies only in the case where the robot intended to take the Pittsburgh left and the human did not intend to yield. But in this case, their performance was far worse than any of the other models' policies for any other case. The cause of this low reward was multiple collisions with human experiment participants. This behavior was most likely caused by differences between the actual capabilities of the robot versus the capabilities of humans in the human data that was partially used to construct the model. The POMDP policies were more robust to the inaccuracies in the model. The time-dependent POMDP policies achieved results as good or better than the other model variants for all combinations of interactions. Additionally, it should not be assumed that policies that achieve similar rewards are subjectively similar to the humans interacting with them. There were observable differences in the driving styles of the model variants. These differences will be discussed in the next section, and their consequences will be seen in the discussion of the survey results.

8.5. Outcome and Event Results

In order to obtain a clearer picture of the differences in performance between the polices for the model variants, the frequency of possible trial outcomes and significant events are shown in Table 8.10. The outcome frequencies for the data collection experiment of humans interacting with other humans are also given for comparison. Upon looking at the frequency of various outcomes, it is notable that the relative frequency at which one car or the other went through the intersection first is less balanced for all the robots than it was for the human drivers. While the outcomes for the non-time-dependent POMDP model appear to be most similar to the human outcomes, the way in which they were achieved was very different. The non-time-dependent POMDP ran the red light whenever its intention was to take the Pittsburgh left. While this behavior resulted in it crossing the intersection before the human driver, the penalties for this socially undesirable action were the cause of the relatively low rewards it received. The reason that the policies behaved in this manner will be discussed later with a more detailed examination of the outcomes for this model variant's policies.

The results for the time-dependent POMDP and time-indexed MDP models were similar, with the exception of the occurrence of collisions, which was far higher for the MDP model. The relative frequency with which the turning car crossed the intersection first was very similar, which suggests the possibility of a common cause. The large penalty for collisions most likely resulted in behavior that was relatively conservative when compared to people's. But the relatively high frequency of collisions when the MDP model's policy drove in the manner that human data suggested was possible (as discussed in Section 8.4.4) indicates that this conservative behavior may have been necessary to prevent collisions given the robot's reduced sensing and control capabilities relative to humans.

For all of the robots, the number of trials during which one or more of the cars failed to cross the intersection were greater than for humans driving with other humans. There are a number of possible explanations for this difference. One is that the early cutoff of high-level control several timesteps before the end of each trial may have eliminated last-second actions by the robots that would have resulted in them clearing the intersection before the trial ended. In cases where both drivers were attempting to yield, it is possible that model inaccuracies caused the robot to overestimate the possibility that the human driver would begin moving, both allowing the robot to potentially achieve its intended outcome of allowing the human driver to go first and increasing the risk of collision if the robot were to start to turn across the intersection.

Table 8.10. Summary of trial outcomes for Pittsburgh Left experiment

Model	Outcome					Event	
	T First	S First	T Only	S Only	Neither	Lights (S)	Collisions
human	51%	43%	3%	2%	1%	32%	3%
time POMDP	30%	50%	4%	9%	8%	22%	1%
full time MDP	28%	56%	3%	9%	5%	21%	8%
no time POMDP	57%	28%	2%	13%	1%	13%	$< 1\%$

In order to better understand the ways the policies of the model variants differed from one another, as well as from human behavior, the results for each combination of driver intentions will be considered separately. The results for the trials in which the robot intended to take the Pittsburgh left and the human driver intended to yield are shown in Table 8.11. This combination of intentions is non-conflicting. In this case, all the models achieved outcome results relatively similar to those of human drivers. For the non-time-dependent POMDP case, notice that the human driver used their lights far less frequently than with any other model variant (or with humans). This is because the robot typically ran the red light, making signaling the driver to take the Pittsburgh left unnecessary. It is believed that the non-time-dependent POMDP model was unable to accurately represent the true risk of collision between the robot and the human driver because of its smaller, less expressive state space. In order to avoid this exaggerated risk of collision, it frequently chose to run the red light. While people occasionally jump the red by a very small amount of time (typically only after the perpendicular traffic light has also turned red), the non-time-dependent model is unable to represent this distinction. As a result, it frequently ran the red light long before it changed, resulting in very unnatural-appearing behavior. Also note the relatively high incidence of outcomes where only the driver going straight crossed the intersection. The non-time-dependent POMDP, lacking a explicit representation of the duration of an episode of interaction, would often reach a terminal state before the final timestep of the interaction. It is unclear why this occurred so early for a subset these cases. To the human, these early terminations looked like the turning car yielding and waiting at the intersection.

While both the full time MDP and the time-dependent POMDP performed this interaction successfully, the way in which they achieved their outcomes differed in their responsiveness to the human participant. The MDP driver was more aggressive, usually immediately taking the left when the light turned. The POMDP driver was more cautious, and would often pause when the light changed. If the human flashed their lights, the POMDP controller would begin the turn, having its belief converge to the case that the person intended to yield by that observation. If the person did not flash their lights, the POMDP would wait slightly longer to converge on the belief that they intended to yield. While both of these behaviors are within the range of what is normally exhibited and socially correct for the Pittsburgh left, the POMDP controller responds to a signaling behavior by the human driver in the way that is commonly understood for this interaction.

Table 8.11. Summary of trial outcomes for Pittsburgh Left, Yield interactions

Pittsburgh Left, Yield (non-conflicting intentions)							
Model	Outcome					Event	
	T First	S First	T Only	S Only	Neither	Lights (S)	Collisions
human	92%	4%	2%	1%	1%	39%	1%
time POMDP	88%	5%	0%	5%	2%	35%	3%
full time MDP	85%	10%	2%	3%	0%	35%	2%
no time POMDP	80%	0%	0%	20%	0%	2%	0%

The results for the trials in which the robot intended to take the Pittsburgh left and the human driver intended to yield are shown in Table 8.12 . When the intentions of the human driver and the robot were in conflict, the time-dependent MDP and POMDP policies yielded to the human driver far more frequently than a human turner yielded to an oncoming car. But high number of collisions for the MDP policies suggest that this is more appropriate behavior given the controllers' capabilities. Once again, the non-time-dependent POMDP policy succeeded in going first in the majority of cases by running the red light.

Table 8.12. Summary of trial outcomes for Pittsburgh Left, No Yield interactions

Pittsburgh Left, No Yield (conflicting intentions)							
Model	Outcome					Event	
	T First	S First	T Only	S Only	Neither	Lights (S)	Collisions
human	59%	40%	0%	1%	0%	11%	10%
time POMDP	13%	78%	0%	8%	0%	0%	0%
full time MDP	25%	72%	2%	2%	0%	5%	32%
no time POMDP	88%	0%	0%	12%	0%	0%	2%

The results for the trials in which the robot intended to turn left after the human driver and the human driver did not intend to yield are shown in Table 8.13. In this case, all the models performed relatively similarly to the human drivers. All of the models were able to coordinate their behavior relatively successfully because the human's and the robot's intentions were not in conflict. Additionally, because the event of the human driver crossing through the intersection first could be observed and then responded to (rather than needing to predict likely future actions by the human), the less expressive representation of the non-time-dependent POMDP also performed this interaction in an appropriate manner in the majority of cases.

Table 8.13. Summary of trial outcomes for Regular Left, No Yield interactions

Regular Left, No Yield (non-conflicting intentions)							
Model	Outcome					Event	
	T First	S First	T Only	S Only	Neither	Lights (S)	Collisions
human	2%	97%	0%	1%	0%	6%	1%
time POMDP	2%	88%	3%	6%	0%	2%	2%
full time MDP	0%	97%	2%	2%	0%	0%	0%
no time POMDP	10%	77%	0%	12%	2%	2%	0%

The results for the trials in which the robot intended to turn left after the human driver and the human driver intended to yield are shown in Table 8.14. All of the policy groups had a large number of interactions where both of the cars failed to make it through the intersection before the red light. While humans were better at performing this interaction, it is worth noting that this case was the one that was most difficult for them as well. As in the other case where the robot and the human's goals were in conflict, both the time-dependent POMDP and MDP models' policies took the Pittsburgh left less frequently than human drivers. In fact, the MDP policies seemed to be quite inflexible, taking the left before the other car only in a very small number of cases. This seems to support the idea that the prior model may have overestimated the probability that the straight driver would eventually go first, allowing the turning car to achieve its most preferred outcome. The time-dependent POMDP policy chose to make the turn before the straight car more frequently than the MDP policy. This more flexible range of responses may have been one of the contributing factors in forming people's impressions that its behavior was more natural and socially appropriate than the behavior of the policies for the MDP model, as will be discussed in the next section. The non-time-dependent POMDP was most willing to take the left before the other car, most likely because of its inability to accurately estimate how much time remained to act. Because the model overestimated the possibility that the episode could end soon from many states, it to chose to achieve a less desirable outcome that it had direct control over rather than wait for the other car to cross the intersection first.

8.5.1. Summary of Results

The distributions of outcomes reached during interaction by each of the model variants show that the differences between their rewards were caused by significant differences

Table 8.14. Summary of trial outcomes for Regular Left, Yield Interactions

Regular Left, Yield (conflicting intentions)							
Model	Outcome					Event	
	T First	S First	T Only	S Only	Neither	Lights (S)	Collisions
human	51%	29%	11%	2%	7%	74%	0%
time POMDP	16%	27%	12%	16%	28%	51%	0%
full time MDP	0%	45%	5%	28%	22%	43%	0%
no time POMDP	48%	35%	6%	6%	3%	50%	0%

in the way they engaged in the interaction. All of the policy groups were conservative drivers overall when compared to human drivers performing the same task, with a few notable exceptions. In the non-time dependent POMDP case, the desire to avoid collisions, combined with a model that lacked the time representation to accurately represent the interaction, caused the policy to choose to take penalized actions to run the red light in cases when the robot's intention was to make a Pittsburgh Left. The behavior of the MDP policies was less responsive to the other driver than the other model variants. When intending to take the Pittsburgh left against humans who did not intend to yield, inaccuracies in control capabilities caused by the human task performance data in the model caused the MDP controller to choose actions that resulted in collision far more often than the actions of the other model variants. The time-dependent POMDP model produced behavior that was the most socially correct (not running the red light or hitting other drivers) and was responsive to the behavior of the human driver. This is probably the reason that the experiment participants found its behavior to be significantly more natural and socially appropriate than the other model variants, as will be shown by the survey results.

8.6. Reasoning About Beliefs

The differences in the behavior of the time-dependent POMDP and MDP policies can be seen clearly in the (Pittsburgh Left, No Yield) interaction, based on the reward received. But there are also more subtle differences between the behaviors of these policies that may have different effects on people's impressions of the driver they interact with, even when they achieve similar levels of reward. The MDP policies act based on an assumed intention for the human. This assumed intention, chosen randomly, could either be correct or incorrect for each interaction. If it is incorrect, the burden is on the human to figure out what belief the agent holds about them and possibly adjust their behavior. Because each MDP policy contains only a model of human behavior for a fixed intention, the robot has no "belief" about multiple possible intentions a person may hold. In contrast, the POMDP policies include both the Yield and No Yield intention for the human driver. Based on the observations received, the robot's belief about the human's intention may change. The ability to act based on beliefs about intentions produces behavior that is more responsive to human action.

Consider the graph shown in Figure 8.5. The belief associated with the intention assigned to the two most probable states is graphed at each timestep over the course of an example interaction for which the human had the intention to yield to the robot. The traffic light doesn't change to green until the 30th timestep. For the early timesteps it is very uncertain which intention the human holds. At the 22nd timestep, the human flashes their headlights. After this point, the uncertainty in the human's intention is greatly reduced because this action is strongly associated with the intention to yield. In this case, the robot is able to disambiguate the human's intention well before the green light. After the light changes, the human makes no actions to contest the right-of-way when the robot moves into the intersection, so the robot's belief in correct human intention remains high. The drop in the last few timesteps corresponds to a flattening of the belief when the policy stops receiving new observations just before the end of the episode, as discussed in Section 8.1.2.

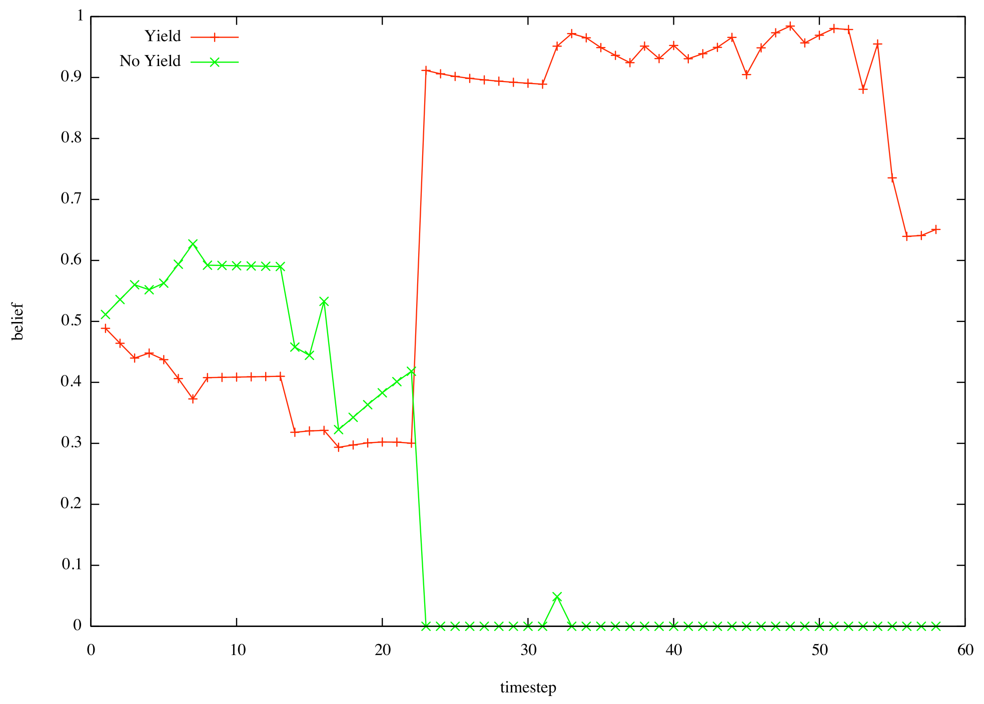


Figure 8.5. A belief trace for the intentions associated with the 2 most probable states during an interaction in which the straight car flashed their headlights.

The graph in Figure 8.6 shows an example of the beliefs over the course of another interaction for which the human intended to yield. In this case, the human did not at any point flash their lights at the robot to indicate that they were yielding. The robot's beliefs indicate that it recognized the human's intention to yield over most of the course of the interaction. The brief period of uncertainty between the 30th and 40th timesteps was in response to the human inching forward towards the intersection before stopping and waiting again.

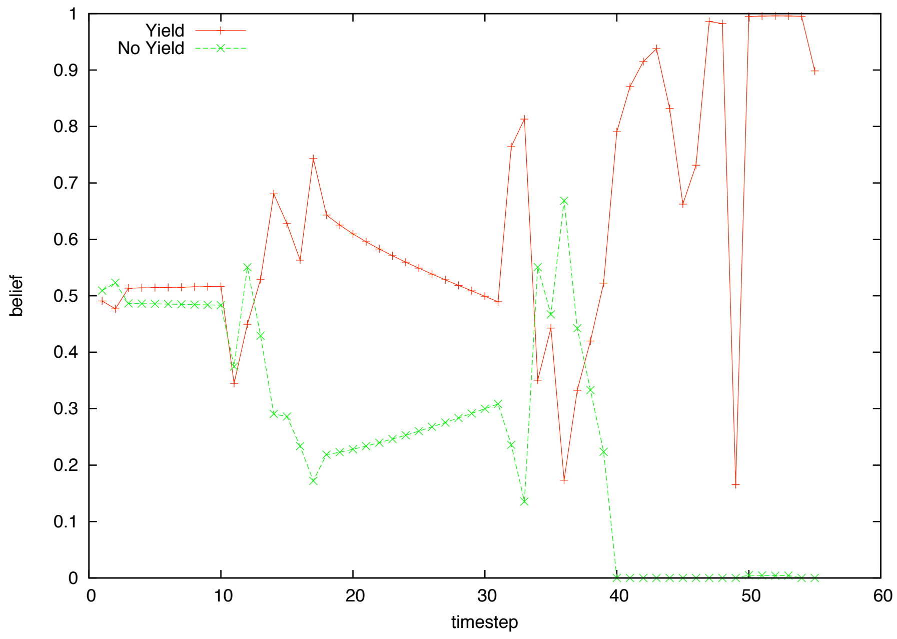


Figure 8.6. A belief trace for the intentions associated with the 2 most probable states during an interaction in which the straight car did not flash their headlights.

8.7. Survey Results

The paper survey given to experiment participants after each trial was identical to the survey administered to participants in the data collection experiment in Chapter 5. The survey was made up of four statements with a 5-point Likert-type scale ranging from ”strongly disagree” to ”strongly agree”. The survey form used in the experiments is included in Appendix B. The responses to the survey from the data collection experiment are included in the bar graphs in Figures 8.7 through 8.10. But caution should be exercised in directly comparing the responses between the two separate experiments, or inferring that the time-dependent POMDP policies were "human-like" because of the similarity of the responses they received to the responses for the human drivers. There are factors influencing people's impressions in the second experiment relative to the first that are impossible to measure. The most important of these is the lack of a human opponent driver. While participants in the data collection experiment were separated from one another by a screen, they saw their driving opponent before and after the interaction and knew that another human was controlling the car. For the experiment comparing the controllers, there was obviously no other driver present. The briefing for the experiment told participants that they would be driving against a different "opponent driver" each trial, and it was not specified whether the opponent was human or a computer-controlled agent. There was no attempt made to measure whether the participants believed they were driving against a human.

It would have been possible to have human actors pretend to be driving the car each trial rather than the controllers, but this design would be undesirable for a number of reasons. The purpose of this experiment was to compare the performance of the controllers, not to perform a Turing test, and to attempt to do both in one experiment could confound the measurements of the differences between the controllers. If people were told that they were driving against a human opponent but became suspicious of this fact because of unnatural behavior by one of the controllers, it could negatively impact their judgement for the entire experiment. They may put their effort into figuring out whether the other driver was human rather than performing the interaction in a natural manner. Also, the feeling of being deceived about the way the experiment operates might reduce their willingness to honestly and attentively participate in it.

8.7.1. Statement 1: Human Driver Control

The first survey statement given was, "I felt that I was able to control the car in the simulation to do what I wanted it to do." As in the data collection experiment, this question was given as a control to ensure that difficulties with low level control of the car were unlikely to have a significant impact on people's performance of the task. As in the data collection experiment (where this statement received a median response of 4), the vast majority of participants agreed that they could control the car. In fact, as can be observed in Figure 8.7, all of the controller distributions were more heavily skewed to the right ("agree") side of the scale than the human distribution. This is most likely because the

Table 8.15. Median values for the response to survey statement 1 with 95% bootstrap confidence intervals and results of matched pairs Wilcoxon signed rank test.

Model	Response	Test Result
time POMDP	4 [4, 5]	
full time MDP	4 [4, 5]	
no time POMDP	5 [4, 5]	T=-12.5, p=0.2734

drivers in the controller comparison experiment only had to drive straight through the intersection rather than making a turn, which some people had difficulties with due to the narrowness of the two-lane road. The medians for each model variant are presented in Table 8.15. While the median response for the other two variants was 4 as in the human experiment, the median response for the non-time-dependent POMDP was 5. However, a comparison between its distribution and the distribution for the time-dependent POMDP failed to find a statistically significant difference. Any effect that was not due to chance may be explained by the human driver often having easier trials in terms not having to take any driving maneuvers to coordinate behavior because the other car had run the red light.

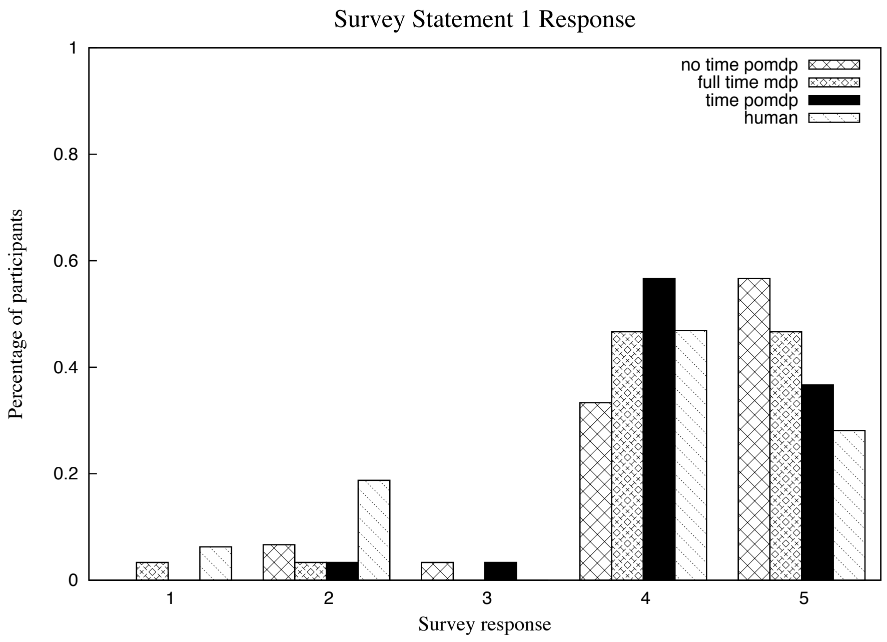


Figure 8.7. Responses to the survey statement, "I felt that I was able to control the car in the simulation to do what I wanted it to."

Table 8.16. Median values for the response to survey statement 2 with 95% bootstrap confidence intervals and results of matched pairs Wilcoxon signed rank test.

Model	Response	Test Result
time POMDP	4 [3, 4]	
full time MDP	2 [2, 4]	T = 50.5, p = 0.041* ($< p_2$)
no time POMDP	2 [1, 2]	T = 130.5, p $<$ 0.001* ($< p_1$)

8.7.2. Statement 2: Perceived ”Naturalness” of Controller

The second survey statement was, ”The actions of the other car seemed natural to me”. For this statement, there were significant differences found between the responses for the time-dependent POMDPs and both of the other two model variants, as summarized in Table 8.16. The time-indexed MDP and the non-time dependent POMDP both received a median response of 2 (”somewhat disagree”) while the median response for the time-dependent POMDP was 4 (”somewhat agree”). The median response for the human drivers in the data collection experiment was also 4, but an examination of the distribution of the responses in Figure 8.8 reveals that people appeared to find the time-dependent POMDP controller at least somewhat less natural than a human driver. This is most likely due to the inflexibility of that controller in the cases where both drivers intended to yield. The difference for the time-indexed MDP was more pronounced, as there is a bimodal split in the responses which indicate that while some people agreed its behavior was natural, a greater proportion disagree. This result may be due to some people finding the ”aggressive” driving style of MDP policies realistic. The distribution for the non-time-indexed POMDP is skewed to the left (”disagree”) side of the scale, probably due to the fact that a person running a red light when making a left turn is an extremely rare occurrence.

8.7.3. Statement 3: Similarity of Interaction to ”Real World”

The third survey statement was, ”The interactions we engaged in were similar to the way I interact with other drivers taking the Pittsburgh left in real life.” As shown in Table 8.17, the median response for all of the controllers was 4 (”somewhat agree”), which was the same median response to this statement for the human drivers in the data collection experiment. Because the medians did not differ, no statistical tests were performed. However, the confidence intervals for the medians suggest that there were potentially meaningful differences between the distributions of responses. Figure 8.9 shows that all of the model variants’ distributions differed from one another and from the human drivers’ distribution in interesting ways. The non-time-dependent POMDP controllers’ responses followed a bimodal and relatively flat distribution, with more weight falling on the left

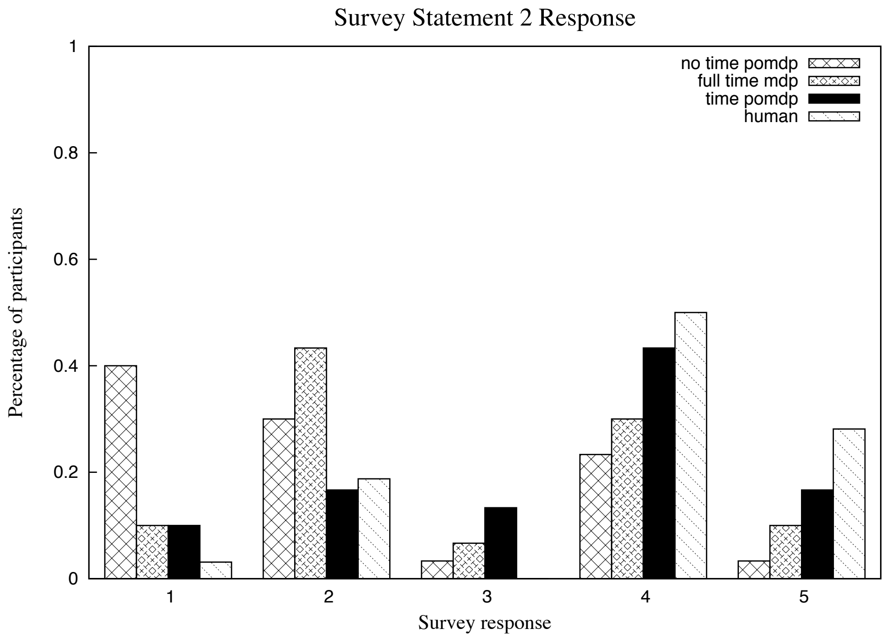


Figure 8.8. Responses to the survey statement, "The actions of the other car seemed natural to me."

("disagree") side of the scale than any of the other drivers. It seems that there was little consensus among drivers as to how closely their behavior mimicked real world interactions. This might be due to differences in subjects' opinions on how negatively they perceived the red light running behavior and how strongly it influenced their opinion versus other cases where the behavior of those controllers were more similar to the other models. Despite having the same median, it seems that people's opinion of these controllers was very different from their opinion of the other model variants and of human drivers.

The responses for the time-indexed MDP and the time-dependent POMDP controllers were more similar both to one another and to the human driver responses. Of the two distributions, the one for the POMDP controllers is more heavily skewed to the right ("agree") side of the scale, indicating that people may have found the POMDP controllers slightly more realistic. However, both model variants clearly produced behavior that people thought resulted in realistic interactions. This might seem surprising, considering that the MDP sometimes caused a collision. But as noted before, this did not happen to every driver, and when it did it was typically one interaction out of eight. People might have viewed the collision as an anomaly or as an event for which they shared responsibility.

Table 8.17. Median values for the response to survey statement 3 with 95% bootstrap confidence intervals and results of matched pairs Wilcoxon signed rank test.

Model	Response
time POMDP	4 [4, 4]
full time MDP	4 [3.5, 4]
no time POMDP	4 [2, 4]

These results indicate that people found the MDP controllers' relatively aggressive driving style to be similar to driving behavior that they'd experienced in the real world, which is not so surprising after all, given that drivers vary a great deal in their level of aggressiveness. Both of the controllers did seem to be viewed as producing slightly less realistic interactions than interacting with a human driver. While the medians were the same, the mode of the responses for the human drivers was 5 ("strongly agree") while the modes for the two model variants were the same as their medians.

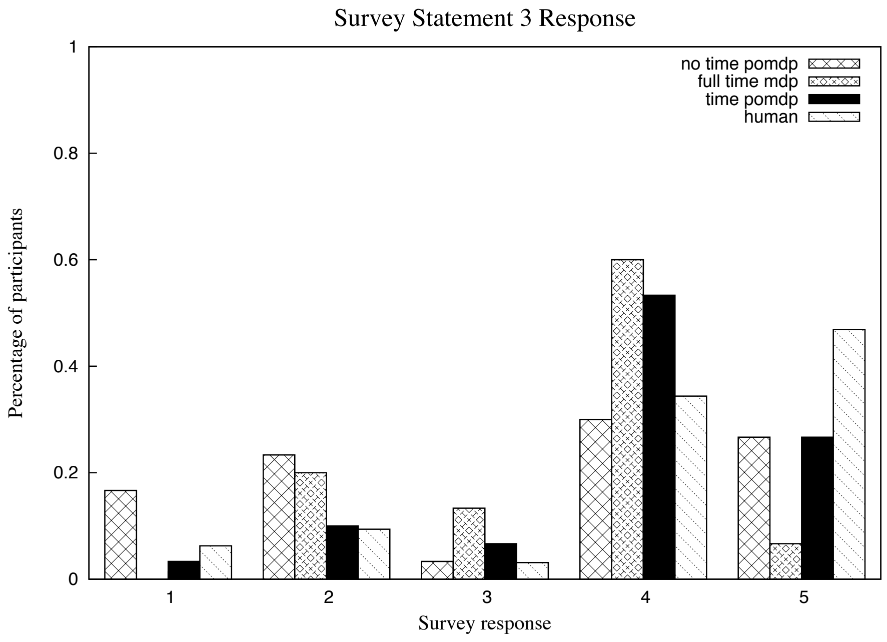


Figure 8.9. Responses to the survey statement, "The interactions we engaged in were similar to the way I interact with other drivers taking the Pittsburgh left in real life."

8.7.4. Statement 4: Socially Appropriate Interaction

The final question of the survey, and the one most significant to accessing the relative performance of the controllers for social interaction, was, "I felt that the other driver followed proper driving etiquette for taking and yielding to the Pittsburgh left." The median response for the time-dependent POMDP controllers was 4 ("somewhat agree"), which was the same median response as for human drivers in the data collection experiment. Both other model variants were accessed less favorably, with the non-time-dependent POMDP receiving a median response of 2 ("somewhat disagree") and the time-indexed MDP a median response of 2.5 (just between "somewhat disagree" and "no opinion"). Table 8.18 shows that these differences were found to be statistically significant.

Figure 8.10 gives a more detailed view of the differences in people's impressions of the model variants. The distribution for the non-time-dependent POMDP is strongly skewed to the left ("disagree"). This strong negative response is most likely due to the fact that this controller ran the red light during multiple trials of its interaction with every experiment subject. No matter how it may have behaved in other cases, people's impressions were undoubtedly effected by its violation of this traffic law. It is not unusual to see drivers move through the intersection briefly during red lights at boundary cases (turning from red to yellow or just before the turn to green as judged by watching the perpendicular light change). There is little social stigma to such an action, particularly if there is no visible cross traffic that could be endangered. But to run through a red that has not recently changed or is on the verge of changing is seen as a breach of social propriety as well as traffic law.

The reactions to the behavior of the time-indexed MDP controller were more mixed. The responses had a bimodal distribution, with the majority of the weight split between "somewhat disagree" and "somewhat agree". One might wonder if these modes correspond to the drivers that were and were not hit by a car controlled by the MDP. But a closer examination of the data reveals a more complex picture. A collision occurred for 17 out of the 30 drivers during a trial with the MDP controller. But 6 of these 17 gave the controller a response of 3 or 4. Of the 13 drivers that did not experience a collision, 4 gave the controller a score of 1 or 2. While the collisions probably made a very strong impact on people's opinion of whether the MDP controllers' behavior was socially appropriate, it was also probably not the only factor. The MDP controller generally appeared less responsive to the actions of the other driver than the POMDP controllers. When taking the Pittsburgh left, it would start moving immediately in most cases, as opposed to

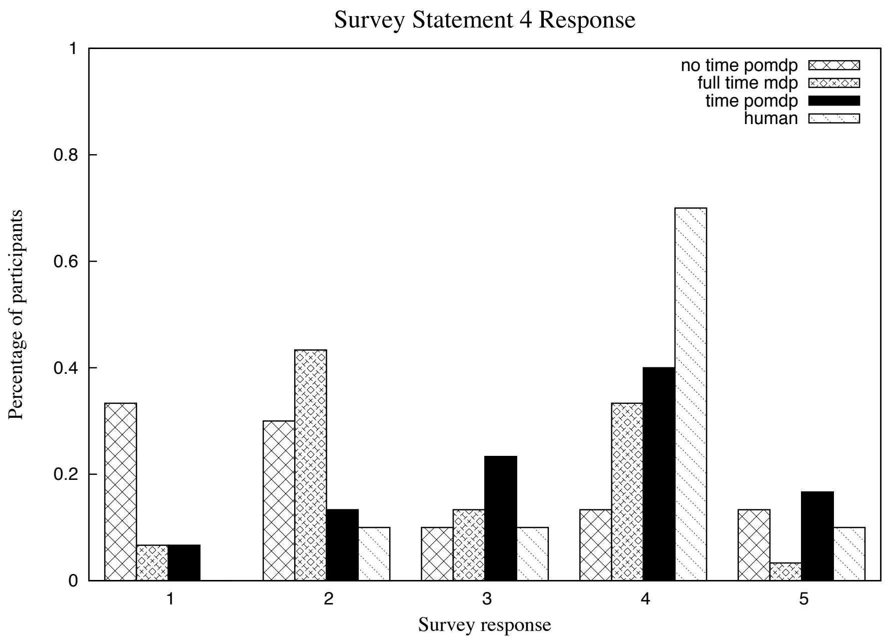


Figure 8.10. Responses to the survey statement, "I felt that the other driver followed proper driving etiquette for taking and yielding to the Pittsburgh left."

the POMDP controllers, which often would hesitate until the human driver flashed their lights.

Of all the controllers, the time-dependent POMDP model variant received the most positive response. The distribution is clearly skewed to the right, though there are a not insignificant number of responses that judged the controller a 3 ("no opinion") or worse. This may be due to the controller's unwillingness to relent and make the Pittsburgh left in cases where both the robot and the human were attempting to yield. But it seems unlikely, given the available data, that even humans would be significantly more positively evaluated. It should be cautioned that one shouldn't read too much into the form of the distribution for the human driver responses. As mentioned in Section 5.3.2, this data only represents the responses of a third of the subjects in the data collection experiment. But given how peaked the data is, it is likely that this small sample gives a reliable estimate of the median. Even without considering the data collection experiment responses, the time-dependent POMDP controllers are clearly more successful than the other model variants at following proper etiquette for the Pittsburgh left, therefore experimentally demonstrating the validity of the representation used.

Table 8.18. Median values for the response to survey statement 4 with 95% bootstrap confidence intervals and results of matched pairs Wilcoxon signed rank test.

Model	Response	Test Result
time POMDP	4 [3, 4]	
full time MDP	2.5 [2, 4]	T = 60.0, p = 0.019* ($< p_1$)
no time POMDP	2 [1.5, 3]	T=102.0, p = 0.001* ($< p_1$)

8.8. Conclusion

Analysis of the rewards achieved by the model variants during interaction supports the experimental hypothesis that the time-dependent POMDP model performs better than either the time-indexed MDP or the non-time-dependent POMDP model. Model inaccuracies caused the MDP policy to cause collisions with human drivers noticeably more frequently than the other model variants, and its behavior was less responsive to the human participants overall. The non-time-dependent POMDP was unable to accurately represent the interactions in which the robot intended to take the Pittsburgh left, and its policies chose to run the red light rather than risk collisions with the human driver. The time-dependent POMDP policies exhibited the most consistently successful behavior across all combinations of interactions and were responsive to the actions of human drivers, most likely contributing to their preference for that model variant.

According to statistical testing of the survey results, the controllers from time-dependent POMDP model variant were judged as producing more natural actions than either the time-indexed POMDP or the non-time-dependent POMDP model variants. More importantly, they were also found to be better at following the proper etiquette for the Pittsburgh left than either of the other model variants. The time-dependent POMDP models were also the only model variant to have the same median response that was given when evaluating human drivers during the data collection experiment for every survey statement. In light of these results, the time-dependent POMDP clearly outperformed the other model variants in terms of people's subjective impressions of their behavior. It can also be concluded that this representation was highly successful at performing its role in the interaction overall.

CHAPTER 9

CONCLUSION

THIS thesis investigated the question of representation in planning for socially situated tasks in human-robot interaction. The motivation for this work was the insight that interaction in these types of domains were dependent on characteristics of the problem that are not commonly modeled. When humans interact, they follow patterns of behavior based on their intended goals, and they understand the actions of others in terms of the intentions they believe them to hold. People's actions and changes in the dynamic environments in which many of these tasks are carried out are dependent on time. It was hypothesized that in order to act successfully and in a socially appropriate manner, it is necessary that the models used to develop policies for action must allow for reasoning about both time and human intention.

Because a person's intention is part of their internal mental state, it cannot be directly observed by a robot. But a model that describes observable behaviors in terms of these internal states can be used to reason about a human's intention and respond appropriately. POMDP models were chosen to represent these planning problems because of their ability to represent observable characteristics of the environment in terms of partially observable state. The class of probabilistic graphical models of which POMDPs are a part have also proved to be very successful at modeling the uncertainty present in a wide variety of robotics domains. A model description language was designed that created models based on weighted action rules that describe the effects of: the robot's actions, the behaviors exhibited by the human, changes in the environment, and interactions between any of these sources of change in the state. These model descriptions allow large, complex models to be created from relatively small sets of rules that describe the relationship between low-level action and the robot and human's intentions.

In order to represent time-dependent action outcomes in these models, a time-index was added to the state space. This representation of time is simple and flexible, though at the cost of dramatically increasing the size of a model's state space. However, many of these time-states are likely to be redundant or irrelevant to the creation of a good policy. Acting on this insight, a state aggregation method was developed that combined physically identical time-states based on an estimate of their future reachable rewards. The amount of aggregation performed by this method can be adjusted by changing the threshold value for aggregation. Time-state aggregation proved to produce policies that perform comparably to those of the original time-indexed model and take far less time to solve in both of the experimental domains tested.

The appropriateness of this approach to modeling social interaction can only truly be verified by evaluating the policies produced in real interactions with human participants. These experiments were performed in the Pittsburgh left domain, a driving task based on a local driving convention governed by social rules. In an initial data collection experiment, humans performed the interaction in pairs in a driving simulator. This data was combined with the rule based prior model of the task to produce models that were solved to create policies to control a robot driver in a second experiment. In this experiment, the robot played the role of the turning driver in repeated interactions with human drivers taking the role of a car driving straight through the intersection. The robot driver was controlled by three different sets of policies designed to test the effects of different representations of the problem on policy performance. In addition to the time-state aggregated POMDP models, two less expressive sets of models were used. In one set, time-indexed MDP models for each combination of robot and human intention were produced to assess the effect of acting according to an assumed intention rather than reasoning about the human's intention. For the other set, POMDP models were created without a time-index in the state space to assess the effect of not representing the time-dependent effects of actions. Each human participant in the experiment interacted with all three sets of policies. After interacting with each policy set, the participant filled out a brief survey about their subjective impressions of the controller's behavior.

As hypothesized, the policies from time-dependent POMDP models achieved more reward, on average, during interaction than either of the other representations. Though the less expressive representations performed similarly for some of the combinations of robot and human intention, they both performed significantly worse for at least one combination. Additionally, the time-dependent POMDP policies were also viewed as more natural and socially appropriate than the other representations according to the survey results. These results suggest that even in cases where the different representations received similar reward, the time-dependent POMDP policies' behaviors were preferred by human subjects.

9.1. Modeling Socially Situated Tasks

The type of human-robot social interactions focused on are the class of *socially situated tasks*. The goal-oriented nature of these tasks makes decision-theoretic planning an appropriate approach to producing policies for action. Decision-theoretic planning is attractive because it allows behavior to arise from a prior model of the task that may combine domain knowledge specified by an expert with whatever task performance data is available. This approach is more flexible and powerful than hand-coding a policy for a specific domain. It can can perform well (given a good model) in situations where limited data availability would make it difficult for a learning-based approach to perform adequately. This is especially important for human interaction domains, as getting quality task performance data from people may be time-consuming and costly. The social nature of these tasks means that people follow commonly known social conventions to achieve their goals. The existence of these conventions for behavior make it possible to express the human's actions in terms of action rules relating their behavior to their intentions.

For the purpose of this thesis, an agent's *intention* with respect to a task is expressed as a preference ordering over the possible goals an agent may attempt to achieve in a given role. By making the unobservable human intention part of the state space of the POMDP model, policies can be created that act based on beliefs about the human's intention given their observed behavior. The relationship between human intention and behavior is encoded into the state transition structure of the POMDP.

The robot's intentions are represented in the reward structure of the POMDPs. In addition to receiving reward for the achievement of its own goals, the robot receives reward when the human achieves their goals as well. The human's contribution to the reward structure is chosen in such a way as to produce a level of interest in the common good of all interacting agents on the part of the robot that is appropriate for the task domain. At one extreme (no reward for human goal achievement), the robot will act in a purely self-interested manner. At the other extreme (reward only for human goal achievement), the robot will behave in a purely altruistic manner.

9.2. Time-State Aggregation

The modeling of time-dependent action outcomes is an issue that is commonly ignored in decision-theoretic planning with probabilistic graphical models. These models rely on the Markov property, which assumes a model is memoryless with respect to a given state, in order to create computationally efficient representations of problems. Semi-Markov extensions, which are capable of more accurately modeling the time spent in a state and time-dependent action outcomes, require the form of the time-distributions to be specified *a priori*, do not represent time-dependent action outcomes, and cannot be solved using common solution techniques designed for fully Markov models.

In order to avoid these limitations, the time representation chosen was the addition of a time-index variable to the state space of the model. This representation is simple, flexibly models a wide variety of time-dependence, and produces a fully Markov model. The downside of this representation is that it can dramatically increase the size of the state space of the model. This is of particular concern for the practical use of POMDPs, as large models may be prohibitively difficult to solve even using state-of-the-art approximate solution techniques. However, many of these time-states are likely to contain redundant information or be irrelevant to the optimal policy for action according to the model. Acting on this intuition, a method was developed to aggregate time-states in a way that would produce a smaller model while preserving the important time-dependent characteristics of the time-indexed model.

States are selected for aggregation based on an estimate of their value. These values represent the *reachable future reward* from that state for a fixed policy (for the models in this thesis, a random policy was used). The state values are represented either by a distribution over reward values or by the expected reward. The values of *physically identical* states at different timesteps are compared, and they are combined if the differences between values are smaller than a pre-determined threshold. The size of the resulting aggregated models can be adjusted by changing this threshold value. While there are no theoretical guarantees for the performance of these aggregated models, a threshold that yields good performance can be identified by comparing the reward obtained by the aggregated model to loose performance bounds obtained from non-time-indexed and fully observable versions of the models. In both of the experimental domains tested, a wide range of threshold values produced models with performance close to that of the original time-indexed models with a significant speed up in the solution time. For the Pittsburgh left models, the aggregated models could be solved in less than a day, while the time-indexed models took weeks to converge to a solution.

9.3. Evaluating Representations

In the second experiment in the driving simulator, humans performed the Pittsburgh left driving interaction with a robot car taking the role of the turning driver. The robot was controlled by three sets of policies created from three different model representations: time-state aggregated POMDP models, POMDP models without a time-index in the state space, and time-indexed MDP models. Each human participant drove against the policies for all three model variants. The performance of the policies was evaluated both in terms of objective measures of success at the task and subjective measures of how the humans viewed the robot's behavior. The purpose of this experiment was to test this thesis's central hypothesis that both reasoning about human intention and representing time-dependent action outcomes is critical to planning to interact with people in socially situated tasks in a successful and socially appropriate manner.

The time-dependent POMDP policies achieved greater average reward during interaction than either of the other representations. By examining the trials for each combination of robot and human intention, the source of these differences in performance can be explained. The models without a time index performed significantly worse than the time-dependent POMDP policies for the cases where the robot's intention was to take the Pittsburgh left. Because the non-time-indexed model could not represent the problem accurately, its policies chose to take the penalized action of running the red light. This behavior was chosen to avoid an exaggerated risk of collision if the robot were to attempt the Pittsburgh left during the green light when the human might enter the intersection. The MDP polices performed significantly worse than the time-dependent POMDP in the case where the robot intended to take the Pittsburgh left and the human did not intend to yield. In this case, the MDP policies frequently took actions that led to a collision between the cars. This occurred because the human task performance data in the model caused the robot to overestimate its ability to take aggressive driving actions without causing a collision. Though this data was also used to construct the POMDP model, the POMDP policies did not take these aggressive actions, probably because actions were chosen based on a distribution of belief across the human's possible intentions. Additionally, even in the cases where the reward achieved by the MDP and POMDP policies were similar, there were qualitative differences in the way the reward was achieved. Because the MDP policies assumed an intention for the human, the robot acted according to this assumption, leaving it up to the human to recognize the robot's intention and compensate. The POMDP policies were more responsive to the actions of the human driver. For example, when the robot intended to take the Pittsburgh left, it would often wait for a person to signal their willingness to yield by flashing their headlights.

After interacting with the policies of each model variant, the human subjects filled out a short survey about their impressions of the interactions and the other driver. This survey was the same one given to the people who drove with other human drivers in the data collection experiment. The time-dependent MDP models also outperformed the other model variants according to these subjective measures. The human subjects reported the behavior of the time-dependent POMDP policies to be the more natural than the other policies. Most importantly, people agreed that the time-dependent POMDP policies followed the proper etiquette for the Pittsburgh left. The median responses were in disagreement with this statement for the other two model variants. The median responses for the time-dependent POMDP policies for all the survey statements were the same as the median responses that humans reported for their interaction with other people in the data collection experiment. The proposed modeling approach clearly captures aspects of the problem necessary to perform the task successfully and in a socially appropriate manner that the less expressive representations fail to capture.

9.4. Future Work

This work addresses a previously largely unexplored planning domain with methods for creating practically useful policies, and there are a great number of future directions to explore. The experimental results emphasize the importance of accurate models for planning (and the difficulty of acquiring them). While the domain description language makes expressing the model much more concise than it would be to hand-code a policy, a large amount of work and trial-and-error testing is required on the part of the domain expert to produce a good prior model. It would be interesting to explore ways in which available data could be used to help create the structure of the model itself, while still leveraging a human expert's ability to recognize the characteristics of the state structure that are important for social interaction.

Another area of model specification where learning techniques could potentially be used to improve model accuracy is in the design of the reward structure. In the proposed approach, the value of different rewards given the robot's intention and the value of the human's goals are chosen as part of the model design. It might be possible to find the numeric values that best explain the human data observed by using inverse reinforcement learning (Ng and Russell, 2000). By learning the values from the data (perhaps given the preference ordering over the rewards), the level of altruism exhibited by the robot could be fit to the level that humans exhibit while performing the task.

A person's intention has an obvious impact on their behavior during an interaction, and specific intentions can be associated with data traces using the data collection experiment design demonstrated in the Pittsburgh left driving task. But there are other unobservable mental characteristics of people that are likely to have an impact on how they engage in interactions. Personality traits such as aggressiveness will also effect the ways in which people act to achieve their goals. In many interactions, it may improve the quality of the model to include these personality traits in the hidden state of the POMDP models. How these personality traits can be measured and the way their relationship to behavior should be expressed is a possible direction for future work on more complete and accurate models of human interaction behavior.

There are also interesting issues in applying time-state aggregation to a wider range of domains. The state value estimates for time-state aggregation are parameterized by the policy used to calculate the reward distributions or expected reward. For the domains explored in this thesis, a random policy proved to provide useful estimates. But for certain types of models, such as ones where reaching a state with high reward is a rare occurrence that may only happen very late in an episode of interaction, a random policy may not propagate this reward back to earlier states strongly enough to distinguish important differences between the states. What other policies may produce useful state value estimates and which policies are likely to work best with certain reward structures are questions that may require experimental investigation when applying time-state aggregation to new classes of problems with time-dependent action outcomes.

9.5. Conclusion

In summary, this thesis demonstrates a novel approach to developing policies for human-robot interaction. Human-robot interaction in socially situated domains is framed as a planning problem, and model-based decision-theoretic planning with time-indexed POMDP models is proposed as the solution. A method for designing such models based on domain-specific action rules and human-task performance data is presented. A time-state aggregation algorithm is designed to reduce the models to a tractable size for efficient solution times. In experiments with human subjects, models designed in this manner produced policies which achieved more reward and were viewed as more natural and socially acceptable by people than less expressive representations. It is the hope that this work demonstrates the effectiveness of this planning approach for human-robot interaction and calls attention to social tasks in human-robot interaction as a potential source of rich and interesting planning domains.

BIBLIOGRAPHY

Abele, S., Bless, H., and Ehrhart, K.-M. (2004). Social information processing in strategic decision-making: why timing matters. *Organizational Behavior and Human Decision Processes*, *93*(1), 28–46.

Alami, R., Clodic, A., Montreuil, V., Sisbot, E. A., and Chatila, R. (2005). Task planning for human-robot interaction. In *sOc-EUSAI '05: Proceedings of the 2005 joint conference on Smart objects and ambient intelligence*, pages 81–85, New York, NY, USA. ACM Press.

Allouche, M., Boissier, O., and Sayettat, C. (2000). Temporal social reasoning in dynamic multi-agent systems. In *ICMAS '00: Proceedings of the Fourth International Conference on MultiAgent Systems (ICMAS-2000)*, page 23, Washington, DC, USA. IEEE Computer Society.

Aumann, R. J. (1976). Agreeing to disagree. *Annals of Statistics*, *4*, 1236–1239.

Baldwin, D. A. and Baird, J. A. (2001). Discerning intentions in dynamic human action. *Trends in Cognitive Sciences*, *5*(4), 171–178.

Bengio, Y. and Frasconi, P. (1996). Input/output hmms for sequence processing. *IEEE Transactions on Neural Networks*, *7*, 1231–1249.

Berninghaus, S. K. and Ehrhart, K.-M. (1998). Time horizon and equilibrium selection in tacit coordination games: Experimental results. *Journal of Economic Behavior & Organization*, *37*(2), 231–248.

Boella, G. and van der Torre, L. (2003). Local policies for the control of virtual communities. In *IEEE/WIC WI'03*.

Bosch, L. t., Oostdijk, N., and Ruiter, J. P. d. (2004). Durational aspects of turn-taking in spontaneous face-to-face and telephone dialogues. In *Text, Speech, and Dialogue, 7th International ConferenceTSD 2004*, pages 563–570.

Boutilier, C., Dearden, R., and Goldszmidt, M. (2000). Stochastic dynamic programming with factored representations. *Artificial Intelligence*, *121*(1-2), 49–107.

Boyan, J. A. and Littman, M. L. (2000). Exact solutions to time-dependent MDPs. In *Neural Information Processing Systems*, pages 1026–1032.

Bratman, M. E. (1990). What is intention? In Cohen, P. R., Morgan, J., and Pollack, M. E. (Eds.), *Intentions in Communication*, pages 15–31. Cambridge, MA: MIT Press.

Breazeal, C. and Berlin, M. (2008). Spatial scaffolding for sociable robot learning. In *AAAI*, pages 1268–1273.

Breazeal, C. and Scassellati, B. (2000). Infant-like social interactions between a robot and a human caregiver. *Adapt. Behav.*, *8*(1), 49–74.

Cantaluppi, L. (1984). Optimality of piecewise-constant policies in semi-markov decision chains. *SIAM Journal on Control and Optimization*, *22*(5), 723–739.

Chitgopekar, S. S. (1969). Continuous time markovian sequential control processes. *SIAM Journal on Control*, *7*(3), 367–389.

Christensen, H. I. and Pacchierotti, E. (2005). Embodied social interaction for robots. In Dautenhahn, K. (Ed.), *AISB-05*, pages 40–45, Hertsfordshire.

Colman, A. M. (2003). Cooperation, psychological game theory, and limitations of rationality in social interaction. *Behavioral and Brain Sciences*, *26*, 139–153.

Dautenhahn, K. and Nehaniv, C. L. (2002). The agent-based perspective on imitation. In *Imitation in animals and artifacts*, pages 1–40. Cambridge, MA, USA: MIT Press.

Dean, T. and Givan, R. (1997). Model minimization in markov decision processes. In *AAAI/IAAI*, pages 106–111.

Delgado, J. (2002). Emergence of social conventions in complex networks. *Artif. Intell.*, *141*(1), 171–185.

Demiris, Y. (2007). Prediction of intent in robotics and multi-agent systems. *Cognitive Processing*, *8*(3), 151–158.

Dietterich, T. G. and Flann, N. S. (1995). Explanation-based learning and reinforcement learning: A unified view. In *International Conference on Machine Learning*, pages 176–184.

Duong, T. V., Bui, H. H., Phung, D. Q., and Venkatesh, S. (2005). Activity recognition and abnormality detection with the switching hidden semi-markov model. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 838–845, Washington, DC, USA. IEEE Computer Society.

Felicissimo, C., Lucena, C., Briot, J.-P., and Choren, R. (2006). An approach for contextual regulations in open mas. In Garcia, A., Ghose, A., and Kolp, M. (Eds.), *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'2006) International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS'06)*, pages

25–32, Hakodate, Japan. AAMAS.

Fern, A., Natarajan, S., Judah, K., and Tadepalli, P. (2007). A decision-theoretic model of assistance. In *IJCAI*, pages 1879–1884.

Fong, T. W., Nourbakhsh, I., and Dautenhahn, K. (2003). A survey of socially interactive robots. *Robotics and Autonomous Systems*, *0*.

Frith, U. and Frith, C. (2001). The biological basis of social interaction. *Current Directions in Psychological Science*, *10*, 151–155(5).

Geanakoplos, J. (1992). Common knowledge. In *TARK '92: Proceedings of the 4th conference on Theoretical aspects of reasoning about knowledge*, pages 254–315, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Geanakoplos, J., Pearce, D., and Stacchetti, E. (1989). Psychological games and sequential rationality. *Games and Economic Behavior*, *1*(1), 60–79.

Gockley, R., Forlizzi, J., and Simmons, R. (2007). Natural person-following behavior for social robots. In *Human-Robot Interaction*, pages 17–24.

Gold, K. and Scassellati, B. (2006). Learning acceptable windows of contingency. *Connection Science*, *18*(2).

Good, J. M. M. (2007). The affordances for social psychology of the ecological approach to social knowing. *Theory & Psychology*, *17*(2), 265–295.

Hall, E. T. (1968). Proxemics. *Current Anthropology*, *9*(2-3), 83–108.

Hansen, E. and Zhou, R. (2003). Synthesis of hierarchical finite-state controllers for pomdps. In *Thirteenth International Conference on Automated Planning and Scheduling (ICAPS)*.

Hirschaur, S. (2005). On doing being a stranger. *Journal for the Theory of Social Behaviour*, *35*(1), 41–67.

Hoey, J., Bertoldi, A. v., Poupart, P., and Mihailidis, A. (2007). Assisting persons with dementia during handwashing using a partially observable markov decision process. In *5th International Conference on Computer Vision Systems ICCVS07*.

Isbell, C., Shelton, C. R., Kearns, M., Singh, S., and Stone, P. (2001). A social reinforcement learning agent. In Müller, J. P., Andre, E., Sen, S., and Frasson, C. (Eds.), *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 377–384, Montreal, Canada. ACM Press.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artif. Intell.*, *101*(1-2), 99–134.

Lewis, D. K. (1969). *Convention: A Philosophical Study*. Cambridge, MA: Harvard University Press.

Li, L., Walsh, T. J., and Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. In *Symposium on Artificial Intelligence and Mathematics*.

Limnios, N. and Oprisan, G. (2001). *Semi-Markov Processes and Reliability*. Birkhauser: Boston, MA.

Lockerd, A. and Breazeal, C. (2004). Tutelage and socially guided robot learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 4, pages 3475–3480.

Madani, O., Hanks, S., and Gordon, A. (1999). On the undecidability of probabilistic planning and infinite-horizon partially observable markov decision processes. In *Sixteenth National Conference in Artificial Intelligence*, pages 406–416.

Mahadevan, S., Ghavamzadeh, M., Rohanimanesh, K., and Theocharous, G. (2003). Hierarchical approaches to concurrency, multiagency, and partial observability. In Si, J., Barto, A., Powell, W., and Wunsch, D. (Eds.), *Learning and Approximate Dynamic Programming: Scaling up to the Real World*. John Wiley & Sons, New York.

Marhasev, E., Hadad, M., and Kaminka, G. A. (2006). Non-stationary hidden semi markov models in activity recognition. In *AAAI Workshop on Modeling Others from Observations MOO-06*.

Mataric, M. J. (1997). Learning social behavior. *Robotics and Autonomous Systems*, *20*, 191–204.

Maynard Smith, J. (1998). The origin of altruism. *Nature*, *393*, 639–+.

McMillen, C. and Veloso, M. (2007). Thresholded rewards: Acting optimally in timed, zero-sum games. In *AAAI '07*.

Meltzoff, A. N. (1995). Understanding the intentions of others: Re-enactment of intended actions by 18-month-old children. *Developmental Psychology*, *31*(5), 838–850.

Michalowski, M. P. and Kozima, H. (2007). Methodological issues in facilitating rhythmic play with robots. In *IEEE International Symposium on Robot and Human Interactive Communication(RO-MAN)*.

Mui, L., Mohtashemi, M., and Halberstadt, A. (2002). Notions of reputation in multi-agents systems: a review. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 280–287, New York, NY, USA. ACM Press.

Nakauchi, Y. and Simmons, R. (2002). A social robot that stands in line. *Autonomous Robots*, *12*(3), 313–324.

Ng, A. Y. and Russell, S. (2000). Algorithms for inverse reinforcement learning. In *in Proc. 17th International Conf. on Machine Learning*, pages 663–670. Morgan Kaufmann.

Nourbakhsh, I., Powers, R., and Birchfield, S. (1995). Dervish an office navigating robot. *AI MAGAZINE*, *16*(2), 53–60.

Papadimitriou, C. and Tsitsiklis, J. (1987). The complexity of markov decision processes. *Mathematics of Operations Research*, *12*(3).

Pelphrey, K. A., Morris, J. P., and Mccarthy, G. (2004). Grasping the intentions of others: The perceived intentionality of an action influences activity in the superior temporal sulcus during social perception. *J. Cognitive Neuroscience*, *16*(10), 1706–1716.

Pineau, J., Gordon, G., and Thrun, S. (2003a). Point-based value iteration: An anytime algorithm for pomdps. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Pineau, J., Gordon, G., and Thrun, S. (2003b). Policy-contingent abstraction for robust robot control. In *Uncertainty in Artificial Intelligence (UAI)*.

Pineau, J., Montemerlo, M., Roy, N., Thrun, S., and Pollack., M. (2003). Towards robotic assistants in nursing homes: challenges and results. *Robotics and Autonomous Systems*, *42*(3-4), 271–281.

Poupart, P. and Boutilier, C. (2004). Vdcbpi: an approximate scalable algorithm for large scale pomdps. In *Advances in Neural Information Processing Systems 17 (NIPS)*, pages 1081–1088.

Puterman, L. M. (1994). *Markov Decision Processes*. Wiley, New York.

Pynadath, D. V. and Marsella, S. C. (2005). Psychsim: Modeling theory of mind with decision-theoretic agents. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1181–1186.

Rosemary Emery-Montemerlo, Geoff Gordon, J. S. and Thrun, S. (2004). Approximate solutions for partially observable stochastic games with common payoffs. In *International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*.

Roy, N. and Gordon, G. (2002). Exponential family pca for belief compression in pomdps. In *Advances in Neural Information Processing Systems*.

Schultz, J., Imamizu, H., Kawato, M., and Frith, C. D. (2004). Activation of the human superior temporal gyrus during observation of goal attribution by intentional objects. *J. Cognitive Neuroscience*, *16*(10), 1695–1705.

Sebanz, N., Bekkering, H., and Knoblich, G. (2006). Joint action: bodies and minds moving together. *Trends in Cognitive Sciences*, *10*(2), 70–76.

Shoham, Y. and Tennenholtz, M. (1997). On the emergence of social conventions: modeling, analysis, and simulations. *Artif. Intell.*, *94*(1-2), 139–166.

Si, M., Marsella, S., and Pynadath, D. V. (2006). Thespian: Modeling socially normative behavior in a decision-theoretic framework. In *IVA*, pages 369–382.

Simmons, R. and Koenig, S. (1995). Probabilistic robot navigation in partially observable environments. In *International Joint Conference on Artificial Intelligence*, pages 1080–1087.

Sing, C. (1980). Equivalent rate approach to semi-markov processes. *IEEE Trans. Reliability*, *29*(3), 273–274.

Smart, W. D. and Kaelbling, L. P. (2002). Effective reinforcement learning for mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3404–3410.

Smith, T. (2007a). *Probabilistic Planning for Robotic Exploration*. PhD thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Smith, T. (2007b). ZMDP software for POMDP and MDP planning. http://www.contrib.andrew.cmu.edu/ trey/zmdp/.

Smith, T. and Simmons, R. (2004). Heuristic search value iteration for pomdps. In *Uncertainty in Artificial Intelligence (UAI)*.

Smith, T. and Simmons, R. G. (2006). Focused real-time dynamic programming for MDPs: Squeezing more out of a heuristic. In *Proc. Nat. Conf. on Artificial Intelligence (AAAI)*.

Smith, T., Thompson, D. R., and Wettergreen, D. S. (2007). Generating exponentially smaller POMDP models using conditionally irrelevant variable abstraction. In *Proc. Int. Conf. on Applied Planning and Scheduling (ICAPS)*.

Stone, L. D. (1973). Necessary and sufficient conditions for optimal control of semi-markov jump processes. *SIAM Journal on Control*, *11*(2), 187–201.

Striano, T., Henning, A., and Stahl, D. (2006). Sensitivity to interpersonal timing at 3 and 6 months of age. *Interaction Studies*, *7*(2), 251–271.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Theocharous, G. and Kaelbling, L. P. (2004). Approximate planning in pomdps with macro-actions. In Thrun, S., Saul, L., and Schölkopf, B. (Eds.), *Advances in Neural Information Processing Systems 16*. Cambridge, MA: MIT Press.

Thomaz, A., Hoffman, G., and Breazeal, C. (2006). Reinforcement learning with human teachers: Understanding how people want to teach robots. *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, 352–357.

Tomasello, M., Carpenter, M., Call, J., Behne, T., and Moll, H. (2005). Understanding and sharing intentions: The origins of social cognition. *Behavioral and Brain Sciences*, *28*, 675–735.

TORCS (2005). TORCS, the open racing car simulator, version 1.2.4. http://torcs.sourceforge.net/.

Walter, H., Adenzato, M., Ciaramidaro, A., Enrici, I., Pia, L., and Bara, B. G. (2004). Understanding intentions in social interaction: The role of the anterior paracingulate cortex. *J. Cognitive Neuroscience*, *16*(10), 1854–1863.

Wereschagin, M. (2006). Pittsburgh left seen by many as a local right. *Pittsburgh Tribune-Review*.

Wolfe, A. P. (2006). Pomdp homomorphisms. In *NIPS-2006 Workshop on Grounding Perception, Knowledge, and Cognition in Sensory-Motor Experience*.

Younes, H. L. S. and Simmons, R. G. (2004). Solving generalized semi-markov decision processes using continuous phase-type distributions. In *Nineteenth National Conference on Artificial Intelligence*, pages 742–747, San Jose, CA. AAAI Press.

APPENDIX A

Domain Descriptions

A.1. Elevator Domain POMDP Model

A.1.1. Model Description

Each elevator model is parameterized by a number of characteristics that may be varied to produce models of different sizes with different properties.

- people - the number of human passengers inside the elevator
- resolution - the number of grid squares that make up the path from the beginning of the lobby to the back of the elevator
- time - the number of timesteps in the problem before the elevator door closes
- first hesitate - the number of timesteps that the first person exiting the elevator (if any intend to exit) may wait before beginning to move
- follower hestitate - the number of timesteps that any people exiting after the first person starts moving (if any more intend to exit) may wait before beginning to move

The resolution can be thought of as adjusting either the size of the space of the speed at which the robot and people may move. The resolution is always an even number. Because the grid divides the space along lines parallel to the elevator doorway, it is assumed that the grid squares are large enough to contain the robot as well as one or more people. The exception to this is passage through the doorway itself. The grid squares $\frac{resolution}{2}$ and $\frac{resolution}{2}+1$ are special. If the robot is occupying the grid square immediately outside the doorway and a person is occupying the grid square immediately inside it, they are unable to pass each other. They will be stuck in these positions until the door closes and the episode ends.

The movement of the people follows simple rules based on their intentions and the actions of the robot. If a person does not intend to exit, they do not move. If a person does intend to exit, they move every timestep with a few exceptions. Before they move for the first time, they have only a 50% chance of moving for the number of timesteps indicated by the hesitate parameter. After that, they will move every timestep unless they are moving into one of the two doorway grid squares and the person is within a grid square of the square they would be moving into. In this case, there is only a 50% chance they will move at that timestep. This is meant to simulate a person possibly hesitating in confusion if they don't think that the robot is yielding to let them off the elevator.

A.1.2. State Features

The state space for the POMDP model for the elevator riding task is defined as follows:

- robot location - the currently occupied grid square
- person location - the currently occupied grid square
- person intention - stay on at floor or exit at floor
- current timestep

The starting state distribution is a uniform probability distribution over all the states of the first timestep, which produces an equal probability that the humans may hold any possible combination of intentions for a trial. The robot always starts at the grid square of the lobby furthest from the elevator door, and the people always start at the grid square in the elevator furthest from the doorway.

There are 16 possible combinations of these binary valued state features. However, only 12 combinations correspond to states of the world that will actually occur according to our model. In order to create a model in which the person's intention is not deterministic from trial to trial, we create an additional start state with transitions to both of the beginning world states corresponding to the person's possible intentions (to exit the elevator or stay on).

A.1.3. Actions

In each state, the robot has two actions available to it: move or wait. The outcome of these actions is uncertain. When the robot chooses the "wait" action, there is no chance that it's own location will change. The allowable state transitions will update the locations of the people according to the rules governing their behavior.

The possible outcomes of the "move" action are more complicated. In states where the robot is moving into a grid square that is not part of the doorway, the move action will advance the robot by one grid square. If the robot tries to move into an occupied doorway grid square, the move action will fail. If the robot tries to move into a doorway grid square that an exiting person is within a grid square's distance from, the action has a 50% chance of failure. This models the uncertain outcome of the two agents competing to move into a space that only one can occupy.

A.1.4. Observations

During the interaction, the robot receives observations about whether the closest person immediately in front of it is currently moving (px) or standing still (ps). This is a noisy observation with a 20% chance of being incorrect. If there are no a people in the elevator (because all of the passengers have already exited), the robot perceives an elevator empty (ee) observation.

A.1.5. Rewards

We assign reward to states where the robot achieves its own goal (to get onto the elevator before the door closes) and where a person achieves their goal (either to remain on the elevator, or to exit the elevator before the doors close, depending). However, we want the policy the robot learns to favor self-interested action, so we assign greater reward to states in which the robot's goals are achieved. Reward for goal achievement is additive and assigned in the end states (the states in which the door is closed).

- robot reward/penalty: +60/-60
- human reward/penalty: +40/(number of people) / -40/(number of people)

A.2. Pittsburgh Left Domain

A.2.1. Model Description

The pittsburgh Left interaction is modeled from the point of view of the robot performing the role of the car taking a left turn. The human driver takes on the role of the car driving straight. The 30 second interaction takes place over 60 half-second timesteps. One model is produced for each of the two possible intentions for the robot in this role, to take the Pittsburgh left or to yield and take a regular left. The full domain description file used to produce the models is given in Section A.2.6.

A.2.2. State Features

The state space for the POMDP model for the Pittsburgh left driving task is defined as follows:

- human intention (2) - go first, yield
- robot position (5) - the region currently occupied by the robot
- human position (5) - the region currently occupied by the human
- robot angle (2) - whether the car body is turned more than 45 degrees
- robot speed (3) - stopped, rolling/creeping, fast
- human speed (3) - stopped, rolling/creeping, fast
- human headlights (2) - whether the human has flashed their headlights
- collision (3) - no collision, collision at this timestep, collision has occurred
- traffic light (3) - the color of the traffic light facing both the drivers
- time (60) - the current timestep

The intersection and the surrounding roads are partitioned into rectangular regions relative to each agent's starting position that are significant to the interaction. These regions are: more than a car length from the start of the intersection, a car length from the start of the intersection, the half of the intersection closest to its start, the half of the intersection closest to its end, and the road on the other side of the intersection. There is an additional position value that corresponds to the event of an agent entering the region of the road beyond the intersection for the first timestep. This value is used to assign the one-time reward for an agent crossing the intersection.

A.2.3. Actions

Four actions are available to the robot to control its motion. These are high level actions that must be converted into control commands for the car in the driving simulator by a low level controller.

- set speed (3) - stop, go slow, go fast
- turn (1) - turn car body to the left (perpendicular to current orientation)

It is assumed that the actions can produce their immediate effect as a change in state within the timestep they are applied. For example, if the car is going fast and the robot chooses the stop action, the car will be stopped by the next timestep. Other action effects, such as whether moving at a certain speed will cause the robot to move into the next region, are probabilistic.

Further probabilistic action outcomes are created by applying the action rules that represent the human agent and the environment to the states resulting from the robot's action rules. The action rules for the "Other" action describe the possible behavior of the human. Many of the rules have a specific intention value as a precondition because the action represented only occurs or is more likely for one intention or the other. The action rules for the "Env" action have preconditions based on the time variable and control the changing of the traffic light. The action rules for the "Side Effects" action detect combinations of state variables where the robot and human have driven into the same region during a timestep in a way that may result in a collision and update the collision variable.

A.2.4. Observations

Observations are created by defining a subset of the state variables that are directly observable. The observable state variables for this domain are as follows:

- robot position
- human position
- human headlights
- human speed
- collision
- traffic light

Because the robot's current speed and angle should always be known based on the effects of the set speed and turn actions, these variables are not included in the observation space. The value of each observation variable is the same as the value of the variable in the underlying state. As a result, observations for this model design are aliased rather than noisy, meaning that only one observation is possible for any state but the same observation may arise from multiple different states.

A.2.5. Rewards

The robot receives reward for states in which either its car or the car of the human driver exits the intersection. The values of the rewards are chosen to create an ordering over the values of the possible outcomes of an interaction that matches the preference ordering over outcomes for the robot's intention.

Rewards for goal achievement:

- Turning car (robot) first 15
- Turning car (robot) second
  - Pittsburgh left intention 15
  - Regular left intention 25
- Straight car (human) first 10
- Straight car (human) second
  - Pittsburgh left intention 20
  - Regular left intention 10

The rewards obtained for the possible outcomes of an interaction (ignoring the discounting factor and any penalties that may be incurred) are a combination of the reward for the robot's goal achievement and the human's. These combined values are shown in Table A.1.

Table A.1. Rewards for possible interaction outcomes in Pittsburgh left domain

Intention	Outcome				
	T First	S First	T Only	S Only	Neither
Pgh left	15 + 20 = 35	15 + 10 = 25	15	10	0
Reg left	15 + 10 = 25	25 + 10 = 35	15	10	0

The robot is penalized for taking actions that result in damage to the cars or are socially undesirable. A collision between cars receives a large negative reward. Occupying the intersection during a red light receives a small negative reward. Entering the fail state (which occurs when the robot takes an action which is illegal according to the domain description) receives a very large negative reward, which prevents the policies from selecting actions that lead to the fail state. An example of an illegal action would be to turn while over a car length from the intersection, which would result in the car driving off of the road.

Penalties:

- Robot in intersection when traffic light is red -5
- Collision -100
- Fail state -1000

A.2.6. Domain Description File

```
TIMESTEPS
60

STATES
Goal_T 0 0
Goal_S 0 1
Pos_T 0 5
Pos_S 0 5
Ang_T 0 1
Vel_T 0 2
Vel_S 0 2
Light_S 0 1
Penalty_T 0 2
Trafficlight_T 0 2

ACTIONS
Ang_T 1 1
Vel_T 0 2
Other 0 0
Env 0 0
SideEffect 0 0

OBSERVATIONS
Light_S
Pos_T
Pos_S
Vel_S
Penalty_T
Trafficlight_T

#note that you can't turn if you're not moving

RULE
Ang_T 1 a
EFFECTS
Ang_T REL 1
CONDITIONS
Pos_T 2 3 5
Vel_T 1 2

RULE
Ang_T 1 b
EFFECTS
Ang_T REL 1 Pos_T REL 1
CONDITIONS
Pos_T 2 3 5
Vel_T 2

RULE
Ang_T 1 c
EFFECTS
Ang_T REL 1 Pos_T REL 1
CONDITIONS
Pos_T 2 3 5
Vel_T 1
WEIGHTS
4

RULE
Vel_T 0 a
EFFECTS
Vel_T ABS 0
CONDITIONS
Pos_T 1 2 3

RULE
Vel_T 0 b
EFFECTS
Vel_T ABS 0
CONDITIONS
Pos_T 0
time 0 10

RULE
Vel_T 1 a
EFFECTS
```

```
Vel_T ABS 1
CONDITIONS
Pos_T 0 1 2

RULE
Vel_T 1 b
EFFECTS
Vel_T ABS 1
CONDITIONS
Pos_T 3
Ang_T 1

RULE
Vel_T 1 d
EFFECTS
Vel_T ABS 1 Pos_T REL 1
CONDITIONS
Pos_T 0
WEIGHTS
8

RULE
Vel_T 1 e
EFFECTS
Vel_T ABS 1 Pos_T REL 1
CONDITIONS
Pos_T 1
WEIGHTS
4

RULE
Vel_T 1 f
EFFECTS
Vel_T ABS 1 Pos_T REL 1
CONDITIONS
Pos_T 2 3 4
Ang_T 1
WEIGHTS
4

RULE
Vel_T 1 g
EFFECTS
fail
CONDITIONS
Pos_T 2
Ang_T 0
WEIGHTS
4

RULE
Vel_T 2 a
EFFECTS
Vel_T ABS 2
CONDITIONS
Pos_T 0 1 2

RULE
Vel_T 2 b
EFFECTS
Vel_T ABS 2
CONDITIONS
Pos_T 3 5
Ang_T 1

RULE
Vel_T 2 d
EFFECTS
Vel_T ABS 2 Pos_T REL 1
CONDITIONS
Pos_T 0
WEIGHTS
4

RULE
Vel_T 2 e
EFFECTS
```

```
Vel_T ABS 2 Pos_T REL 1
CONDITIONS
Pos_T 1

RULE
Vel_T 2 f
EFFECTS
Vel_T ABS 2 Pos_T REL 1
CONDITIONS
Pos_T 2 3 4 5
Ang_T 1

RULE
Vel_T 2 g
EFFECTS
fail
CONDITIONS
Pos_T 2
Ang_T 0

#Rules for straight car position 0

RULE
Other 0 a0
EFFECTS
Vel_S REL 1
CONDITIONS
Vel_S 0
Pos_S 0

RULE
Other 0 b0
EFFECTS
Vel_S REL 0
CONDITIONS
Vel_S 0
Pos_S 0
time 0 10

Other 0 c0
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Pos_S 0
WEIGHTS
8

RULE
Other 0 d0
EFFECTS
Vel_S REL 0
Vel_S REL 1
CONDITIONS
Vel_S 1
Pos_S 0

RULE
Other 0 e0
EFFECTS
Vel_S REL 0 Pos_S REL 1
CONDITIONS
Vel_S 1
Pos_S 0
WEIGHTS
8

RULE
Other 0 f0
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Pos_S 0
WEIGHTS
4
```

```
RULE
Other 0 h0
EFFECTS
Vel_S REL -1
Vel_S REL 0
CONDITIONS
Vel_S 2
Pos_S 0

RULE
Other 0 i0
EFFECTS
Vel_S REL 0 Pos_S REL 1
CONDITIONS
Vel_S 2
Pos_S 0
WEIGHTS
4

RULE
Other 0 j0
EFFECTS
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Pos_S 0
WEIGHTS
8

#Rules for straight car velocity 0

RULE
Other 0 a1
EFFECTS
Vel_S REL 0
CONDITIONS
Vel_S 0
Pos_S 1 2 3 5

#Straight car always changes out of position 4 after 1 timestep

RULE
Other 0 b1
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Pos_S 4

#car is unlikely to drive into the other car in the intersection

RULE
Other 0 c1
EFFECTS
Vel_S REL 1
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 1
Pos_T 3
WEIGHTS
50

#Straight car with intention to not yield will start moving unless turning car
#is blocking it (pos 3)

RULE
Other 0 d1
EFFECTS
Vel_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 1 2 3 5
Pos_T 0 1 2 4 5
Goal_S 1
```

```
RULE
Other 0 e1
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 1 2 3 5
Pos_T 0 1 2 4 5
Goal_S 1
WEIGHTS
8

RULE
Other 0 h1
EFFECTS
Vel_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 3 5
Pos_T 3
Goal_S 1

RULE
Other 0 i1
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 3 5
Pos_T 3
Goal_S 1
WEIGHTS
4

#Straight car with intention to yield less likely to move into intersection
#ahead of the turning car

RULE
Other 0 j1
EFFECTS
Vel_S REL 1
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 1
Pos_T 0 1 2
Goal_S 0
WEIGHTS
10

#Straight car with intention to yield moves into intersection after
#turning car clears it

RULE
Other 0 k1
EFFECTS
Vel_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 1
Pos_T 4 5
Goal_S 0

RULE
Other 0 l1
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 1
```

```
Pos_T 4 5
Goal_S 0
WEIGHTS
4

#Straight car in or past intersection with intention to yield speeds up from stop

RULE
Other 0 m1
EFFECTS
Vel_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 2 3 5
Goal_S 0

RULE
Other 0 o1
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 0
Trafficlight_T 1 2
Pos_S 2 3 5
Goal_S 0
WEIGHTS
4

#Straight car travelling at vel 1 may slow down to stop at any position

RULE
Other 0 a2
EFFECTS
Vel_S REL -1
CONDITIONS
Vel_S 1
Pos_S 1 2 3 5

#Straight car travelling at vel 1 must change out of pos 4 after 1 timestep

RULE
Other 0 b2
EFFECTS
Vel_S REL 0 Pos_S REL 1
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Pos_S 4

#Straight car with intention to not yield will go if other car not
#blocking path (in pos 3)

RULE
Other 0 c2
EFFECTS
Vel_S REL 0
Vel_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1 2 3 5
Pos_T 0 1 2 4 5
Goal_S 1

RULE
Other 0 d2
EFFECTS
Vel_S REL 0 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1 2 3 5
Pos_T 0 1 2 4 5
Goal_S 1
WEIGHTS
4
```

```
RULE
Other 0 e2
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1 2 3 5
Pos_T 0 1 2 4 5
Goal_S 1

RULE
Other 0 f2
EFFECTS
Vel_S REL 0
Vel_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 3 5
Pos_T 3
Goal_S 1

RULE
Other 0 g2
EFFECTS
Vel_S REL 0 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 3 5
Pos_T 3
Goal_S 1
WEIGHTS
4

RULE
Other 0 h2
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 3 5
Pos_T 3
Goal_S 1

#car is unlikely to drive into the other car in the intersection

RULE
Other 0 i2
EFFECTS
Vel_S REL 0
Vel_S REL 1
Vel_S REL 0 Pos_S REL 1
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1
Pos_T 3
WEIGHTS
50

RULE
Other 0 l2
EFFECTS
Vel_S REL 0
Vel_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 0
Pos_S 1
Pos_T 3
WEIGHTS
50
```

```
#Straight car with intention to yield less likely to speed up into intersection
#before the turning car reaches the intersection

RULE
Other 0 m2
EFFECTS
Vel_S REL 0
Vel_S REL 1
Vel_S REL 0 Pos_S REL 1
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1
Pos_T 0 1 2
Goal_S 0
WEIGHTS
10

#Straight car with intention to yield will speed up after turning car
#clears intersection

RULE
Other 0 n2
EFFECTS
Vel_S REL 0
Vel_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1
Pos_T 4 5
Goal_S 0

RULE
Other 0 o2
EFFECTS
Vel_S REL 0 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1
Pos_T 4 5
Goal_S 0
WEIGHTS
4

RULE
Other 0 p2
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 1
Pos_T 4 5
Goal_S 0

#Once in or past intersection, turning car w/ intention to yield will speed up

RULE
Other 0 q2
EFFECTS
Vel_S REL 0
Vel_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 2 3 5
Goal_S 0

RULE
Other 0 r2
EFFECTS
Vel_S REL 0 Pos_S REL 1
CONDITIONS
```

```
Vel_S 1
Trafficlight_T 1 2
Pos_S 2 3 5
Goal_S 0
WEIGHTS
4

RULE
Other 0 s2
EFFECTS
Vel_S REL 1 Pos_S REL 1
CONDITIONS
Vel_S 1
Trafficlight_T 1 2
Pos_S 2 3 5
Goal_S 0

#Straight car at vel 2 may stop at any position (but unlikely)
#or slow down
RULE
Other 0 a3
EFFECTS
Vel_S REL -2
Vel_S REL -1
CONDITIONS
Vel_S 2
Pos_S 1 2 3 5
WEIGHTS
8

#Straight car must change position out of pos 4 after 1 timestep

RULE
Other 0 b3
EFFECTS
Vel_S REL 0 Pos_S REL 1
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Pos_S 4

#Straight car w/ intention to not yield will keep moving at same or lesser speed
#as long as turning car not blocking it (pos 3)

RULE
Other 0 c3
EFFECTS
Vel_S REL 0
Vel_S REL -1
Vel_S REL 0 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 1 2 3 5
Pos_T 0 1 2 4 5
Goal_S 1

RULE
Other 0 d3
EFFECTS
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 1 2 3 5
Pos_T 0 1 2 4 5
Goal_S 1
WEIGHTS
4

RULE
Other 0 e3
EFFECTS
Vel_S REL 0
Vel_S REL -1
Vel_S REL 0 Pos_S REL 1
CONDITIONS
```

```
Vel_S 2
Trafficlight_T 1 2
Pos_S 3 5
Pos_T 3
Goal_S 1

RULE
Other 0 f3
EFFECTS
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 3 5
Pos_T 3
Goal_S 1
WEIGHTS
4

#car is unlikely to drive into the other car in the intersection

RULE
Other 0 g3
EFFECTS
Vel_S REL 0
Vel_S REL -1
Vel_S REL 0 Pos_S REL 1
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T  1 2
Pos_S 1
Pos_T 3
WEIGHTS
50

RULE
Other 0 i3
EFFECTS
Vel_S REL 0
Vel_S REL -1
CONDITIONS
Vel_S 2
Trafficlight_T  0
Pos_S 1
Pos_T 3
WEIGHTS
50

#Straight car with intention to yield unlikely to move into intersection
#before the turning car

RULE
Other 0 j3
EFFECTS
Vel_S REL 0
Vel_S REL -1
Vel_S REL 0 Pos_S REL 1
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 1
Pos_T 0 1 2
Goal_S 0
WEIGHTS
10

#Straight car with intention to yield will move once other car has
#cleared intersection

RULE
Other 0 k3
EFFECTS
Vel_S REL 0
Vel_S REL -1
Vel_S REL 0 Pos_S REL 1
```

```
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 1
Pos_T 4 5
Goal_S 0

RULE
Other l3
EFFECTS
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 1
Pos_T 4 5
Goal_S 0
WEIGHTS
4

#Once in or past intersection, turning car w/ intention to yield will keep going

RULE
Other 0 m3
EFFECTS
Vel_S REL 0
Vel_S REL -1
Vel_S REL 0 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 2 3 5
Goal_S 0


RULE
Other 0 n3
EFFECTS
Vel_S REL -1 Pos_S REL 1
CONDITIONS
Vel_S 2
Trafficlight_T 1 2
Pos_S 2 3 5
Goal_S 0
WEIGHTS
4

#the outcome of this rule always turns on lights

RULE
Other 0 a4
EFFECTS
Light_S ABS 1
CONDITIONS
Light_S 0
Pos_S 0 1
Goal_S 0
Vel_S 0
WEIGHTS
10

#the outcome of this rule always turns on lights

RULE
Other 0 b4
EFFECTS
Light_S ABS 1
CONDITIONS
Light_S 0
Pos_S 0 1
Goal_S 1
Vel_S 0
WEIGHTS
100

#this section is for side effects of both cars’ actions (like collision)
```

```
RULE
SideEffect 0 a
EFFECTS
Penalty_T ABS 1
CONDITIONS
Penalty_T 0
Pos_T 3
Pos_S 2

RULE
SideEffect 0 b
EFFECTS
Penalty_T ABS 0
CONDITIONS
Penalty_T 0
Pos_T 3
Pos_S 2
WEIGHTS
50

RULE
SideEffect 0 c
EFFECTS
Penalty_T ABS 1
CONDITIONS
Penalty_T 0
Pos_T 3
Pos_S 3
WEIGHTS
100

RULE
SideEffect 0 d
EFFECTS
Penalty_T ABS 1
CONDITIONS
Penalty_T 0
Pos_T 4
Pos_S 2 3
WEIGHTS
100

RULE
SideEffect 0 e
EFFECTS
Penalty_T ABS 1
CONDITIONS
Penalty_T 0
Pos_T 1 2
Pos_S 3
WEIGHTS
1000

RULE
SideEffect 0 f
EFFECTS
Penalty_T ABS 0
CONDITIONS
Penalty_T 0
Pos_T 0 1 2 4 5
Pos_S 0 1 3 4 5

RULE
SideEffect 0 g
EFFECTS
Penalty_T ABS 0
CONDITIONS
Penalty_T 0
Pos_T 3
Pos_S 0 1 3 4 5

RULE
SideEffect 0 h
EFFECTS
Penalty_T ABS 0
CONDITIONS
Penalty_T 0
```

```
Pos_T 0 1 2 4 5
Pos_S 2

RULE
SideEffect 0 i
EFFECTS
Penalty_T ABS 2
CONDITIONS
Penalty_T 1 2

#Environment rules (governing the lights) start here

RULE
Env 0 a
EFFECTS
Trafficlight_T ABS 0
CONDITIONS
Trafficlight_T 0
time 0 28

RULE
Env 0 b
EFFECTS
Trafficlight_T ABS 2
Trafficlight_T ABS 0
CONDITIONS
Trafficlight_T 0
time 29 31

RULE
Env 0 c
EFFECTS
Trafficlight_T ABS 2
CONDITIONS
Trafficlight_T 0
time 32 32

RULE
Env 0 d
EFFECTS
Trafficlight_T ABS 2
CONDITIONS
Trafficlight_T 2
time 29 51

RULE
Env 0 e
EFFECTS
Trafficlight_T ABS 1
Trafficlight_T ABS 2
CONDITIONS
Trafficlight_T 2
time  52 54

RULE
Env 0 f
EFFECTS
Trafficlight_T ABS 1
CONDITIONS
Trafficlight_T 2
time 55 55

RULE
Env 0 g
EFFECTS
Trafficlight_T ABS 1
CONDITIONS
Trafficlight_T 1
time 52 59

#rewards

REWARD
-5
CONDITIONS
Pos_T 2 3
Trafficlight_T 0
```

```
REWARD
-100
CONDITIONS
Penalty_T 1

#rewards for turning car

REWARD
15
CONDITIONS
Pos_T 4
Pos_S 0 1 2 3 4

REWARD
15
CONDITIONS
Pos_T 4
Pos_S  5
Goal_T 0

REWARD
25
CONDITIONS
Pos_T 4
Pos_S 5
Goal_T 1

#rewards for straight car

REWARD
10
CONDITIONS
Pos_S 4
Pos_T 0 1 2 3 4

REWARD
20
CONDITIONS
Pos_S 4
Pos_T  5
Goal_T 0

REWARD
10
CONDITIONS
Pos_S 4
Pos_T 5
Goal_T 1


START
Goal_T 0 Goal_S 0 Pos_T 0 Pos_S 0 Ang_T 0 Vel_T 0 Vel_S 0 Light_T 0 Light_S 0
 Penalty_T 0 Trafficlight_T 0
Goal_T 0 Goal_S 1 Pos_T 0 Pos_S 0 Ang_T 0 Vel_T 0 Vel_S 0 Light_T 0 Light_S 0
Penalty_T 0 Trafficlight_T 0
```

APPENDIX B

Pittsburgh Left Experiment Materials

B.1. Pittsburgh Left Experiment Survey

1. I felt that I was able to control the car in the simulation to do what I wanted it to.

   strongly disagree   somewhat disagree   no opinion   somewhat agree   strongly agree

2. The actions of the other car seemed natural to me.

   strongly disagree   somewhat disagree   no opinion   somewhat agree   strongly agree

3. The interactions we engaged in were similar to the way I interact with other drivers taking the Pittsburgh left in real life.

   strongly disagree   somewhat disagree   no opinion   somewhat agree   strongly agree

4. I felt that the other driver followed proper driving etiquette for taking and yielding to the Pittsburgh left.

   strongly disagree   somewhat disagree   no opinion   somewhat agree   strongly agree

Document Log:

Manuscript Version 2 — December 30, 2008

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX — 13 May 2010

FRANK BROZ

ROBOTICS INSTITUTE, CARNEGIE MELLON UNIVERSITY, 5000 FORBES AVE., PITTSBURGH, PA 15213, USA
E-mail address: fbroz@cmu.edu

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX