A Bayesian Reinforcement Learning Approach for Customizing Human-Robot Interfaces

Amin Atrash
School of Computer Science
McGill University
Montreal, QC H3A 2A7
aatras@cs.mcgill.ca

Joelle Pineau
School of Computer Science
McGill University
Montreal, QC H3A 2A7
jpineau@cs.mcgill.ca

ABSTRACT
Personal robots are becoming increasingly prevalent, which raises a number of interesting issues regarding the design and customization of interfaces to such platforms. The particular problem addressed by this paper is the use of learning methods to improve the quality and effectiveness of human-machine interaction onboard a robotic wheelchair. In support of this, we present a method for learning and adapting probabilistic models with the aid of a human operator. We use a Bayesian reinforcement learning framework, that allows us to mix learning and execution, as well as take advantage of prior information about the world. We address the problems of learning, handling a partially observable environment, and limiting the number of action requests. We demonstrate empirical feasibility of our approach on an interface for an autonomous wheelchair.

Author Keywords
Activity & plan recognition, Intelligent assistants, Intelligent interfaces for ubiquitous computing

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation: Miscellaneous

INTRODUCTION
Robots are an example of personal agents which have become increasingly ubiquitious over the last ten years. That trend is unlikely to slow down. From robotic vehicles, to intelligent wheelchairs, to social and cognitive assistants, the opportunities are immense. Designing personal robots however requires a profound paradigm shift, compared to their industrial predecessors. In particular, it is imperative that these robots be able to learn and adapt to the environment and humans that surround them. Without the ability to learn, robots are condemned to use preset models of the environment and humans, which are invariably brittle, incomplete, and often inaccurate, especially when it comes to modelling the humans in the environment. This clearly suggests there are exciting opportunities for developing learning methods that can provide personal robots with the flexibility necessary to adapt to their domain, as well as extension of the lessons to other personal agents, such as software agents.

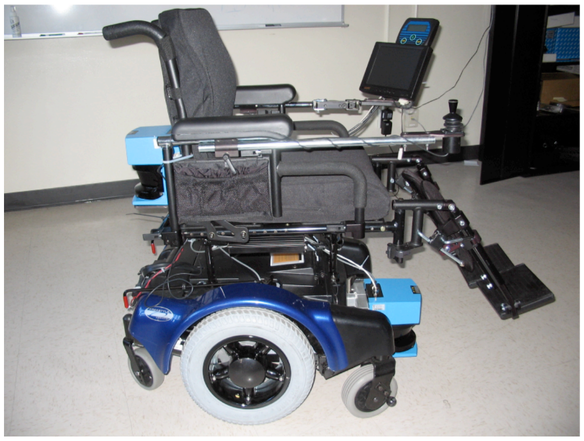

Figure 1. Autonomous Wheelchair

The particular problem addressed by this paper is the use of learning methods to improve the quality and effectiveness of human-machine interaction onboard a robotic wheelchair such as in Figure 1. One of the most interesting challenges raised by this problem is the customization of the interface to the preferences of the user. Most previous interface designs for intelligent wheelchair assumed a fixed model of the human's behavior, which was used in the process of selecting robot actions throughout the interaction.

In this paper, we provide a learning framework for inferring user preferences directly from a user's actions. This has some similarity to the preference elicitation paradigm [3], as well as imitation learning[1]. However the method we propose relies heavily on the Bayesian reinforcement learning framework [4, 6, 16, 5]. This framework has a few key advantages for our target problem. First, as is characteristic of all reinforcement learning methods, it allows us to interleave execution and learning, thus the user's preferences are assessed *in situ*. This is in contrast to the preference elicitation paradigm, which requires the user to state preferences about hypothetical situations. Second, the Bayesian formulation is useful to generalize preference information between users, thus the robot settings used for a new user need not be completely agnostic, but can rely on what was learned from previous users. This is useful for accelerating learning as well as incorporating domain knowledge.

The method we propose extends the standard Bayesian reinforcement learning framework in several directions. First, because our goal is to adapt the robot's behavior to match the user's preferences, the learning method focuses on inference over the reward function (in contrast to inference over the motion/observation models). Second, we must extend the framework to handle partially observable environments. Most Bayesian reinforcemet learning methods assume the state (X) is observable. Yet in human-robot interaction, this is rarely the case. We address this by changing the paradigm substantially and focus on learning by *example*. The assumption is that we can observe (upon request) a target action (provided by an oracle, most often the user), and we gradually learn a class of utility functions which are consistent with this behavior. The actual method is substantially more complicated, but described in full detail in the main technical section of the paper.

The main contribution of this paper is to present a new learning paradigm for adaptable human-robot interaction systems. The method provides scalability and generalizability (through the Bayesian formulation), as well as low cognitive burden on the user (through the use of *in situ* learning) and weak requirements on the robot (through the partial state observability assumption). Thus it is highly promising for deployment onboard complex interactive robot systems. We evaluate the method using test cases pertaining to the interaction interface for an autonomous wheelchair.

BACKGROUND

POMDPs

Partially Observable Markov Decision Processes (POMDPs) [7] are stochastic models used to model non-deterministic decision-making problems and have been shown as well suited for a variety of domains, particularly dialogue management[17]. POMDPs consist of a set of states, S, a set of actions, A, and a set of observations, Z. When an action, a, is execute in state s, the system transitions to state s' with probability $P(s'|s, a)$. The agent then receives a reward, $R(s, a)$ and an observation z is emitted with probability $P(z|s')$. The agent has an initial belief distribution across the states, $P(s_{t=0})$. The belief state, b, is updated using recursively as actions and observations are recieved.

Given a POMDP, an action-selection policy, π, can be determined which maps belief states to actions. Efficient approximate solution methods are available to solve this optimization problem, though details of these algorithms are beyond the scope of this paper. For our experiment, we use the Point-Base Value Iteration (PBVI) algorithm which approximates the policy by using stochastic trajectories to select belief points [18, 12]. This method allows us to solve relatively large POMDPs in a reasonable amount of time.

The algorithms presented here focus on learning the reward function $R(s, a)$. It is well-known that the optimal policy $\pi*$ is invariable to linear transformations in the reward function. Thus, without loss of generality, we can assume the target reward function $R(s, a) \rightarrow [0, 1]$. Furthermore, we can also assume (again, without loss of generality), that the reward function is in reality stochastic, and drawn from a Bernoulli distribution: $P(R(s, a) = 1) = p^{sa}$, where p^{sa} is the expected reward when applying action a in state s.

The Beta distribution is the conjugate prior of the Bernoulli distribution. This fact will be useful to maintain a Bayesian posterior over the parameters p^{sa}, as required by the Bayesian reinforcement learning framework.

Bayesian Reinforcement Learning

The aim of Bayesian reinforcement learning is to maintain a posterior distribution over possible model parameters, and to compute an action selection policy which is optimal with respect to this posterior.

A key step in all Bayesian RL methods is thus to compute the posterior over the transition and reward parameters that define an MDP model. This is usually done by maintaining Dirichlet distributions over possible models and updating the hyper-parameters of the Dirichlet as new events are experienced. A separate Dirichlet distribution is maintained for every (s, a) transition (and in the case of POMDPs, for every observation probability distribution). It is straight-forward to update the posterior over this distribution whenever new experience is acquired.

While updating the posterior can be done easily in closed-form, it is not so easy to compute an optimal policy with respect to this posterior. Existing methods take different approaches to this problem. Some of the most recent methods [6, 5] approximate the posterior by sampling a small set of candidate models, and solving those. The posterior over models continues to be updated, as new experience is acquired. Periodically, the set of sampled models can be resampled. Thus there are two mechanisms for learning: updating of the hyper-parameters, and re-sampling of models.

A BAYESIAN APPROACH FOR ONLINE LEARNING

The approach we propose follows the usual bayesian reinforcement learning paradigm. Recall that our primary motivation is to infer a reward function describing the user's preference, from direct interaction between the robot and user.

The overall approach

The overall structure of our algorithm is described in algorithm 1. There are four key steps. First, the system initializes a prior (i.e. Beta distributions) containing a rough model of the user's preferences. Next, a set of candidate n candidate reward functions are sampled and an optimal policy is obtained corresponding to each. During the learning phase, an oracle (most often the user, but this could also be an outside human operator) is queried to know the most desired action. This action is then executed, and a posterior over the reward function is computed. Periodically, models are removed and new models are resampled.

Algorithm 1 Algorithm

```
Initialize Beta Distribution
Sample $n$ POMDPs $P_1, P_2, ...P_n$ from Beta
Solve $P_1, P_2, ..., P_n$ using approximate POMDP method
**loop**
    Get action from oracle
    Execute Action
    Obtain Observation
    Update Belief States
    For each POMDP, $M_i$ whose policy agreed with the oracle action
    $\alpha_0(s,a) \leftarrow \alpha_0(s,a) + \lambda[b(s)(1 - p_M^{sa})]$
    $\alpha_1(s,a) \leftarrow \alpha_1(s,a) + \lambda[b(s)p_M^{sa}]$
    **if** resample_POMDP() **then**
        Remove least likely POMDP
        Sample new POMDP from Beta
        Find policy for new POMDP
    **end if**
**end loop**
```

We now discuss those sections of the approach which raise the most interesting technical issues.

Policy Oracle

We note that our approach relies on having an oracle which indicates the optimal action upon request. The *learning* component of our approach uses the information to update the posterior over reward functions.

Previous work on learning using an oracle [6] relied on the oracle for full *state* information. While this is sometimes feasible, in many scenarios it is not realistic to have an oracle to provide such detailed information. Especially in the case of human-robot interaction tasks, we believe it is much more reasonable to request that the user indicate a suitable action at any point in time. Consider the case where the task requires joint human-robot control of the navigation functions. In cases where the robot is unsure of the user's preferences, it is perfectly natural to ask the user to indicate the proper navigation action, yet it is much more difficult for the user to provide precise localization information. In contrast, in a task centered around verbal interaction between the robot and human user, it be quite natural to conceive of the oracle as a back-up operator (as many automated phone systems currently operate) which is available during an initial training period. With our particular wheelchair platform, there is always a human operator (in addition to the primary user) present during training phases, to monitor performance of the robot. It is much easier for this operator to substitute for the robot by providing optimal actions whenever needed, than it would be to reveal the state (which can be a very abstract notion in the case of dialogue interactions).

Note that we assume that the oracle and the agent have access to the same model of the world, thus sharing a common belief state. At any time, the agent can request an action from the oracle. The oracle will return the optimal action, a_o, for the current belief state. The agent will then execute the action and use the information to learn as described in the following section.

Representing and Learning Reward

Throughout this work, we focus on the case where the motion and sensor models are known, but information about the reward model is missing. Thus, $P(s'|s,a)$ and $P(z|s')$, but not $R(s,a)$. are known. The objective is for the agent to learn a reward function which results in a robot policy matching that of the oracle.

We first note that a linear transformation can be applied to the reward function of an MDP or POMDP without affecting the resulting policy. For our work, we will be assuming all rewards are within the range [0 1]. We will associate each $R(s,a)$ with a Beta distribution with two hyperparameters, α_0 and α_1, which will act as a prior over a Bernoulli distribution. When creating a new POMDP, M, we first sample a distribution for each $R_M(s,a)$ from the Beta distribution. This provides us with a Bernoulli distribution. We can then set $R_M(s,a)$ to be the expected value over that Bernoulli distribution, p_M^{sa}: $R_M(s,a) = p_M^{sa}$.

Because the environment is partially observable, and rewards are never truely observed by the agent, we must do a probabilistic update of the Beta parameters. To do this, we use the Bernoulli distribution sampled from the model as well as the belief state. After quering the oracle, for each model, M_i, which agreed with the oracle, we perform the following update:

$$\alpha_0(s,a) \leftarrow \alpha_0(s,a) + \lambda[b(s)(1 - p_M^{sa})] \quad (1)$$
$$\alpha_1(s,a) \leftarrow \alpha_1(s,a) + \lambda[b(s)p_M^{sa}] \quad (2)$$

where λ is the learning rate. This can also be seen as a gradient update over the class of models. The intutition behind this update is that models which agree with the oracle have captured some information about the world correctly and should have an influence on the direction of the learning.

WHEELCHAIR INTERACTION MANAGER

To validate our algorithm, experiments were conducted on a wheelchair interaction management system. Learning was focused on inferring the correct reward functions (or a linear transformation thereof). These interaction problems represent real challenging problems grounded in real human conversational data. We believe these are a powerful validation of our method.

The goal of the Smartwheeler[2] project is to develop an autonomous wheelchair to aid individuals with mobility impairments. This includes individuals with impairments such as spinal cord injuries or multiple sclerosis, as well as individuals who suffer from fatigue and sensory impairment which may limit their use of standard electric wheelchairs. Our aim is to develop a system which reduces the cognitive and physical load required to operate the wheelchair for these users. To achieve this, a standard electric wheelchair was outfitted with sensors and motor controls. Basic robot control such as navigation, obstacle avoidance, and map building are handled by the CARMEN robot control software[10].

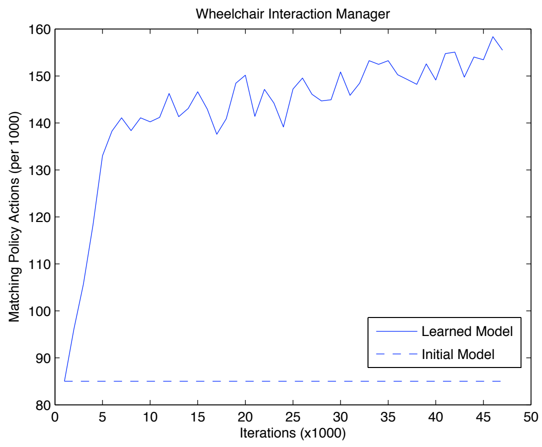

Figure 2. Wheelchair Interaction Manager

Our research focuses primarily on the higher-level interaction and decision-making elements of the system. For multimodal interaction, a touch screen display and microphone were mounted on the wheelchair to provide input in addition to the joystick.

This system poses many challenges. The interaction must be natural and comfortable for the user, requiring minimal technical expertise. The user should be able to communicate with the wheelchair as if he was talking to a human assistant. Any physical interaction through the touch screen or joystick should require minimal cognitive and physical effort. The user should not be required to physically guide the wheelchair, instead offering high level goals and intentions which are then executed by the wheelchair itself. This leads to issues such as determining the user's true intention in light of a very noisy speech recognition system, maintaining the context of the system over time, and making decisions given pieces of information from multiple sources.

To handle these problems, an interactive system was designed consisting of several components. A user speaks to a speech recognition system which attempts to transcribe the audio. This transcription is passed to a semantic parser which attaches semantic and grammar information if possible. This annotated information is now handled by the Interaction Manager, which is the decision-making component of the system and will be the focus of these experiments. Feedback is provided to the user through a mounted display. Details of this system are beyond the scope of this paper and can be found in [2].

As with the previous dialogue management system, the Interaction Manager is modelled using a POMDP. A set of 25 actions were selected for testing, including many basic wheelchair functions such as "move forward one meter," "turn the controller one," and "turn right ninety degrees." Each of these actions represents a user intention and defines the state space. Observations are handled dynamically based on the output from the semantic parser. If the phrase was successfully parsed, the belief state is updated using statistics based on the semantic assignments. If the parse failed, the phrase is treated as a bag-of-words. Statistics for both of these methods were obtained from a previously gathered data set. For each state, there is a corresponding "correct" action which can be executed by the wheelchair. In addition, there are four additional query actions, one general query action which requests a phrase to be repeated, and three action-specific queries which request parameter clarification for a class of action. For this set, there are classes of actions: movement actions, hardware actions (such as tilting the seat), and configuration actions (such as setting the drive mode).

Seven test subjects were asked to perform a series of tests. A test administrator presented them with a task from the Wheelchair Skills Test (WST)[8]. The subject was required to issue a verbal command to the system to complete the tasks. The command was transcribed, parsed, and an action was selected by a basic handcrafted POMDP acting as the Interaction Manager. Each subject was presented with 20-25 tasks. Transcripts of the audio and decision-making were recorded, resulting in 263 phrases and selected actions. Details and results of the UI experiments are documented elsewhere[2]. We will focus on the application of the collected data towards learning a model.

By considering the recorded transitions as observations and the selected actions as the action oracle, a set of experiments were run to attempt to learn the reward model of the POMDP. Learning was performing by iterating over the data set repeatedly. Because the size of this preliminary data set is rather small, a tied parameter approach was used for learning the model. It is assumed that the cost of an incorrect action and correct action are known, allowing learning to focus balancing the costs of the various queries. The cost of queries can be decomposed into three sets: the cost of the general query, the cost of making an appropriate action-specific query, and the cost of making an inappropriate action-specific query.

Throughout these experiments, 20 POMDPs are sampled and maintained from the Beta distribution. Planning on the sampled POMDPs is executed for 10 minutes. Every 1000 steps, the three POMDPs with the least likelihood are removed, and three new POMDPs are sampled from the current Beta. Experiments were repeated 5 times, with the average results reported. The initial set of experiments assume the oracle is queried at every step. This allows us to explore the effects of the learning on the system.

For these experiments, the ground truth POMDP acts as the oracle, the goaling being to determine if the ground truth model can be approaching starting with noisy models. The policy for the POMDP is determined, and the belief state maintained during execution. When the oracle is queried for the policy information, the action for the current belief state of the ground truth POMDP is returned. Initial Beta parameters are generated by taking the ground truth model, perturbing the values with Gaussian noise, and converting the results into corresponding α counts. This approximates a situation where a rough model of the world is known.

Because the reward functions cannot be directly compared, the policies must be judged based on their behavior. Periodically, a simple evaluation phase is run. Beginning with the initial belief state, at every step, an action is selected by the oracle, the action executed, an observation is returned, and the belief state is updated. The action which would have been selected at each step by the sampled POMDPs is compared to the true action. The number on which they agree is tallied. For our experiments, this simulation is run every 1000 iterations, and is run for 1000 steps.

Figure 2 shows the results of learning on the wheelchair interaction manager. The results show that learning is clearly occurring. As more data is gathering, the learning POMDP models behavior more like the ground truth model (ie. the oracle). However, the performance of the system suffers for the very limited amount of data. As more user experiments are conducted and data gathered, the performance overall is expected to improve. Additionally, a better informed prior would address some of these issues. These experiments used a relatively uninformed prior. Future experiments will take advantage of more prior information, possible estimated from an initial data set and refined through experience and deployment of the system. Overall, it is encouraging that learning is in fact occuring, even with such a limited data set for a very complex POMDP.

RELATED WORK

The goal of this work is to develop a method to learn the user's preferences during execution, by observing the target behavior. We adopt the Bayesian reinforcement learning framework [4, 6, 13, 16, 5]. Extensions to partially observable domains have been proposed, yet continue to rely on state information to ground learning [6]. Instead, our method focuses on learning a reward model that results in the same behavior as that the target policy. This does not necessarily have to be the same reward function, just one which results in the same policy.

This idea of imitation learning has been explored in recent literature [1], however the focus is often on matching the behavior, rather than learning a reward function. We belive there are many benefits to learning an actual value function, in particular for generalization (between similar users, or between similar tasks.) Inverse Reinforcement Learning (IRL)[11] also shares some similarity with our work. The maximum margin planning is particularly interesting [15]. However many of the above mentioned method assume full state observability and/or low stochasticity.

Other methods have been examined for learning in POMDPs. It is relatively straightforward to apply the Baum-Welch[14] training algorithm to learn the transition and observation parameters. A real-time memory-constrained version has been implement for robots navigation, as a robot learns parameters while traveling through an environment [9]. These systems require a significant amount of training data and learn the characteristics of the environment. Instead, our method focuses directly on learning the reward function, by matching the policy of the user.

The work on preference elicitation [3] is also closely related, although as pointed out in the introduction, it assumes that the user can directly state his/her preference (or reward function), including for states which are not experiences. There is substantial documentation in the psychology literature which suggests that data acquired in this way is very subjective, and inconsistent, and thus produces very unreliable reward functions.

REFERENCES

1. C. Atkeson and S. Schaal. Robot learning from demonstration. In *International Conference on Machine Learning*, pages 12–20, 1997.
2. A. Atrash, R. Kaplow, J. Villemure, R. West, H. Yamani, and J. Pineau. Towards the deployment of an intelligent wheelchair in a standardized rehabilitation environment. *Interaction Studies*, In Submission.
3. C. Boutilier. A POMDP formulation of preference elicitation problems. In *National Conference on Artificial Intelligence*, 2002.
4. R. Dearden, N. Friedman, and D. Andre. Model based Bayesian exploration. In *Uncertainty in Artifical Intelligence*, pages 150–159, 1999.
5. F. Doshi, J. Pineau, and N. Roy. Reinforcement learning with limited reinforcement: Using Bayes risk for active learning in POMDPs. In *International Symposium on Artificial Intelligence and Mathematics*, 2008.
6. R. Jaulmes, J. Pineau, and D. Precup. Active learning in partially observable Markov decision processes. In *European Conference on Machine Learning*, 2005.
7. L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. In *Artificial Intelligence*, pages 99–134, 1998.
8. R. L. Kirby, D. J. Dupuis, A. H. MacPhee, A. L. Coolen, C. Smith, K. L. Best, A. M. Newtown, A. D. Mountain, D. A. MacLeod, and J. P. Bonaparte. The wheelchair skills test (version 2.4). In *Arch. Phys. Med. Rehabil.*, volume 85, pages 795–804, 2004.
9. S. Koenig and R. Simmons. Unsupervised learning of probabilistic models for robot navigation. In *International Conference on Robotics and Automation*, 1996.
10. M. Montemerlo, N. Roy, and S. Thrun. Perspectives on standarization in mobile robot programming: The Carnegie Mellon Navigation (CARMEN) Toolkit. In *International Conference on Robotics and Systems*, pages 2436–2441, 2003.
11. A. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning*, pages 663–670, 2000.
12. J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*, pages 1025–1032, 2003.
13. P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete Bayesian reinforcement learning. *National Conference on Artificial Intelligence*, 1, 2006.
14. L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in Speech Recognition*, pages 267–296, 1990.
15. N. Ratliff, J. Bagnell, and M. Zinkevich. Maximum margin planning. In *International Conference on Machine Learning*, pages 729–736, 2006.
16. S. Ross, B. Chaib-draa, and J. Pineau. Bayes-adaptive POMDPs. In *Neural Information Processing Systems*, 2007.
17. N. Roy, J. Pineau, and S. Thrun. Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL2000)*, 2000.
18. M. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for POMDPs. In *Journal of Artificial Intelligence Research*, pages 195–220, 2005.