Learning Accuracy and Availability of Humans who Help Mobile Robots

Stephanie Rosenthal, Manuela Veloso, and Anind K. Dey
School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213

Abstract

When mobile robots perform tasks in environments with humans, it seems appropriate for the robots to rely on such humans for help instead of dedicated human oracles or supervisors. However, these humans are not always available nor always accurate. In this work, we consider human help to a robot as concretely providing observations about the robot's state to reduce state uncertainty as it executes its policy autonomously. We model the probability of receiving an observation from a human in terms of their availability and accuracy by introducing Human Observation Providers POMDPs (HOP-POMDPs). We contribute an algorithm to learn human availability and accuracy online while the robot is executing its current task policy. We demonstrate that our algorithm is effective in approximating the true availability and accuracy of humans without depending on oracles to learn, thus increasing the tractability of deploying a robot that can occasionally ask for help.

Introduction

When navigating in complex environments, robots may become uncertain of their location due to imprecise sensors and other factors such as crowds that affect sensor readings. To complete tasks in uncertain environments, many robots have relied on supervisors who are always available and accurate to tell them which action to take (*e.g.,* teleoperators). As more robots are deployed in our environments, it will be infeasible to employ supervisors for each robot.

To reduce the dependence on supervisors during tasks, we propose that robots ask for help, when needed, from people already located in the environment - particularly those in known static locations such as offices. We view these humans as *observation providers* capable of assessing or observing the state of the robot but not directing the robot's actions given that state. For example, a human can indicate the robot's location, but not which direction to travel.

Compared to traditional supervisors, humans in the environment also have limitations:

- they are only accessible in their offices,
- they may be busy and not interruptible,
- they may have limited availability to provide help, and
- they may not always be accurate.

As a robot plans to navigate in the environment, we argue that it must not only consider the distance and expected uncertainty on its many possible paths, but also who is available to help and where, the cost of interrupting and asking them, and whether they will provide an accurate response. A robot that relies on humans in the environment but does not model those humans may navigate along shorter paths with no humans available or with humans who provide inaccurate help. As a result, a robot may not be able to receive the help it may need and may fail to complete tasks.

In this work, we represent robot navigation in an environment with human observation providers as a Human Observation Provider POMDP (HOP-POMDP). Optimal and approximate HOP-POMDP policies can be solved using POMDP solvers to determine not only the actions that the robot should take, but also who and where to ask for help. However, it may be infeasible to approximate the availability and accuracy of human helpers prior to the deployment of the robot in the environment.

We, then, introduce an algorithm to learn the availability and accuracy of human observation providers while the robot executes its current policy using an explore/exploit strategy. While other algorithms instantiate many hypothesis POMDPs with varying observation and transition functions and take advantage of an always-accurate and available human to reduce the hypothesis space and learn the correct functions, it is intractable to solve these hypothesis POMDP policies while a robot is executing and we cannot depend on humans to be available to help the robot. Our algorithm

1. instantiates only a single hypothesis HOP-POMDP,
2. recomputes the HOP-POMDP policy only when the learned accuracy/availability change significantly, and
3. does not depend on a human to always be accurate and available in order to learn.

In terms of the explore/exploit strategy, we show that our algorithm earns more reward and converges faster than either explore-only or exploit-only algorithms. Additionally, when comparing our learning algorithm to prior algorithms on similar-sized POMDPs, we demonstrate that our algorithm converges faster while significantly reducing the number of times a HOP-POMDP policy must be recomputed during learning and without requiring a human to reduce the POMDP hypothesis space.

Related Work

We are interested in creating a model of humans in the environment for a robot to use to determine who can be queried to provide observations during a task. Goal-directed human interaction has primarily been modeled using Partially Observable Markov Decision Processes (POMDPs) (Schmidt-Rohr et al. 2008; Karami, Jeanpierre, and Mouaddib 2009; Armstrong-Crews and Veloso 2007). To briefly review, POMDPs (Kaelbling, Littman, and Cassandra 1998) are represented as the tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{O}, \Omega, T, R\}$ of states $\mathcal{S}$, actions $\mathcal{A}$, observations $\mathcal{O}$ and the functions:

- $\Omega(o, s, a) : \mathcal{O} \times \mathcal{S} \times \mathcal{A}$ - observation function, likelihood of observation o in state s after taking action a
- $T(s, a, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ - transition function, likelihood of transition from state s with action a to new state s'
- $R(s, a, s', o) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \times \mathcal{O}$ - reward function, reward received for transitioning from s to s' with action a

There have been many proposed algorithms to solve the state-action policy for the POMDP (Aberdeen 2003), but it has been shown that solving them optimally is PSPACE-HARD (Papadimitriou and Tsisiklis 1987; Madani 2000).

POMDPs for Collaboration Multi-Agent POMDPs for HRI combine the possible states of the robot R, human H, and the environment E to form a new POMDP representing the task for both the human and robot (*e.g.,* (Schmidt-Rohr et al. 2008; Karami, Jeanpierre, and Mouaddib 2009)). These models represent the human as an agent in the robot's environment that it can interact with. However, multi-agent POMDPs have increased complexity in terms of their exponentially larger state spaces which are less tractable to solve.

POMDPs with Oracle Observations Information provider "oracles" who are always available and accurate have also been considered to reduce uncertainty in POMDPs. Oracular POMDPs (OPOMDPs) plan for needing help to reduce uncertainty, modeling the oracle states separately from the robot's states (Armstrong-Crews and Veloso 2007). OPOMDPs assume that there is an always-available oracle that can be queried for observations from any of the robot's states at a constant cost of asking, λ. The robot executes the best non-asking for help policy (the Q^{MDP} policy (Littman, Cassandra, and Kaelbling 1995)) unless the cost of asking is lower than the cost of executing under uncertainty. However, actual humans in the environment are not always available or interruptible (Fogarty et al. 2005; Shiomi et al. 2008), may not be accurate (Rosenthal, Dey, and Veloso 2009), and they may have variable costs of asking or interruption (Cohn, Atlas, and Ladner 1994; Rosenthal, Biswas, and Veloso 2010).

Learning POMDPs with Oracles Recent work has also focused on using oracles to learn the transition and observation probabilities of POMDPs when it is difficult to model a robot before it is deployed in the environment (Kearns and Singh 2002; Jaulmes, Pineau, and Precup 2005; Doshi, Pineau, and Roy 2008; Cai, Liao, and Carin 2009). In these algorithms, a robot instantiates hypothesis POMDPs that could possibly represent its transition and observation functions. The robot executions the action consensus from all of the hypotheses until there is disagreement between them of which action to take. The robot then asks an oracle to reveal the current state, the hypotheses which do not include the current state in the belief are removed, and new hypotheses are instantiated to replace them. In this way, the robot converges to choosing hypothesis POMDPs with the correct observation and transition functions. However, the robot must solve hypothesis POMDP policies $10^3 - 10^6$ times to learn the observation and transition functions for small problems like the Tiger Problem, which is intractable for robots in real time (Kearns and Singh 2002; Jaulmes, Pineau, and Precup 2005).

In this work, we differentiate traditional oracles from real humans in the environment. We will model locations, availability, cost of asking, and accuracy of humans. We define HOP-POMDPs to take into account the benefits of asking different humans for observations (who may not always be available or accurate) in addition to the distance to its goal in order to determine who to ask and where to navigate. Without a model of humans, the robot may choose a path that has no help available or one where humans often provide inaccurate help, causing the robot to execute with more uncertainty and to possibly fail to complete its task. We, then, introduce an algorithm to learn the availability and accuracy of humans (the HOP-POMDP observation function) without the instantiation and solution of many hypothesis POMDPs as prior work requires.

Humans as Observation Providers

We first formalize the limitations of humans. In particular, we will model the probability of a robot receiving an observation from a human in terms of the human's availability, accuracy, location, and cost of asking.

Location We assume that humans are located in a particular known location in the environment (*e.g.,* an office), and can only help the robot from that location. When the robot is in state s it can only ask for help from the human h_s in the same state. As a result of taking the ask action a_{ask}, the robot receives an observation o from the human.

Availability The availability of a human in the environment is related to their response rate (how often they provide an observation). Receiving o_{null} is equivalent to receiving no observation or timing out waiting for an answer. We define availability α_s as the probability that a human provides a non-null observation o in a particular state s: $0 \leq \alpha_s \leq 1$. If there is no human available in particular state, $\alpha_s = 0$. A human provides any observation other than o_{null} with probability:

$$\sum_{o \neq o_{null}} p(o|s, a_{ask}) = \alpha_s \tag{1}$$

and would provide no observation o_{null} otherwise

$$p(o_{null}|s, a_{ask}) = 1 - \alpha_s \tag{2}$$

This is to ensure that $\sum_o p(o|s, ask) = 1$.

Accuracy When human h_s responds ($o \neq o_{null}$), the probability that he correctly responds o_s depends on his accuracy η_s. The more accurate the human h_s, the more likely they are to provide a true observation o_s. Otherwise, h_s provides observations $o_{s'}$.

Formally, we define the accuracy η_s of h_s as the probability of providing o_s compared to the probability they provide any non-null observation $o \neq o_{null}$ (their availability α_s).

$$\eta_s = \frac{p(o_s|s, a_{ask})}{\sum_{o \neq o_{null}} p(o|s, a_{ask})} = \frac{p(o_s|s, a_{ask})}{\alpha_s} \tag{3}$$

Cost of Asking It is generally assumed that supervisors are willing to answer an unlimited number of questions as long as their responses help the robot. However, there may be a cost of asking in terms of the time it takes to answer the question and the cost of interruption, limiting the number of questions that should be asked (Armstrong-Crews and Veloso 2007).

Let λ_s denote the cost of asking for help from h_s. These costs vary for each person, but are assumed to be known before planning. The cost for querying the human if they answer with a non-null observation $o \neq o_{null}$ is

$$R(s, a_{ask}, s, o_s) = -\lambda_s \tag{4}$$

A robot receives no reward if the person does not respond, $R(s, a_{ask}, s, o_{null}) = 0$, because we assume that they were not bothered by a request that the do not answer. Because there is no negative reward for null observations, the policy can afford to be riskier in who it tries to ask rather than incurring a higher cost of asking someone who is more available.

HOP-POMDP Formalization

We define the HOP-POMDP for a robot moving in the environment with humans, and then discuss differences between humans as observation providers and noisy sensors.

Let HOP-POMDP be $\{\Lambda, \mathcal{S}, \alpha, \eta, \mathcal{A}, \mathcal{O}, \Omega, T, R\}$. where

- Λ - array of cost of asking each human
- α - array of availability for each human
- η - array of accuracy for each human
- $\mathcal{A} = A \cup \{a_{ask}\}$ - autonomous actions and a query action
- $\mathcal{O} = O \cup \{\forall s, o_s\} \cup o_{null}$ - autonomous observations, a human observation per state, and a null observation
- $T(s, a_{ask}, s) = 1$ - self-transition for asking actions

Specifically, let h_s be the human in state s with availability α_s, accuracy η_s, and cost of being asked λ_s. Our observation function Ω and reward function R reflect the limitations of humans defined in Equations 1-4. Remaining rewards, observations, and transitions are defined as in any POMDP.

Humans vs Noisy Sensors Unlike sensors, querying a human multiple times (as is common in POMDPs to overcome sensor noise) will not result in different observations. Thus, if the optimal policy requires a_{ask} at state s when h_s is not available during execution, the robot should instead execute a different action. In our work, we use the Q^{MDP} action (like OPOMDPs (Armstrong-Crews and Veloso 2007)) which chooses the best non-asking action (Kaelbling, Littman, and Cassandra 1998).

Learning Accuracy and Availability

Although the HOP-POMDP model includes availability and accuracy, it may be difficult to determine these values before the robot is deployed. We introduce an online algorithm to learn the availability and accuracy of humans in the environment (the HOP-POMDP observation function) while the robot executes the current optimal policy using an explore/exploit strategy. While prior work on learning observation functions would also learn our HOP-POMDP observation function, the work is not tractable to solve in real environments due to their instantiation of multiple hypothesis POMDPs and the requirement that an oracle provide accurate observations about the robot's state (Jaulmes, Pineau, and Precup 2005; Doshi, Pineau, and Roy 2008). Instead, our algorithm for Learning the Model of Humans as Observation Providers (LM-HOP):

1. requires only one HOP-POMDP to be executed at a time,
2. selectively recalculates the policy only when the observation probabilities have changed significantly, and
3. does not require an always-accurate and available oracle to provide accurate observations.

We detail the LM-HOP Algorithm (1) in terms of these three contributions.

Single HOP-POMDP Instantiation We only instantiate a single HOP-POMDP for learning rather than several hypothesis POMDPs. We maintain counts ($\#o_{s',s}$) for each observation $o_{s'}$ in each state s and for each null observation in each state, as well as the robot's belief $b(s)$, and the availability and accuracy of each human (Lines 1-5). Before each action, the robot chooses a random number ρ to determine if it should explore or exploit the current best policy π (Lines 8-12). Then, as usual, the belief is updated according to the current policy and observation (Line 14), rather than taking the consensus action of many hypothesis POMDPs.

Learning Human Accuracy and Availability In HOP-POMDPs, the robot only needs to learn after a_{ask} actions. Prior work assumes that a human will always answer accurately (o_s). However, in our algorithm, if o_{null} is received after asking, the robot does not know its current state. As a result, it must update the approximate availability of all possible states it could be in, weighted by the probability of being in each state $b(s)$ (Lines 17-18). If an observation o_s is received, its availability of each state is incremented by the belief $b(s)$, because we still do not know if the human answered accurately (Lines 19-22). In order to learn accuracy from observations o_s, each η_s is incremented by the belief $b(s)$ of the robot (Lines 23-24). The accuracy and availability are calculated as averages (over time t) of observations over all questions asked.

It should be noted that due to the limitations of humans in the environment, our algorithm may not converge to the true availability and accuracy. In particular, the robot attributes the unavailability o_{null} to all states weighted by $b(s)$. If one human is always available and another is never available, some unavailability will still be attributed to the always-available human because the robot is uncertain and does not know it is not asking the always-available human.

Selective Learning In order to reduce the number of times a HOP-POMDP policy must be recomputed, we se-

Algorithm 1 LM-HOP$(\pi, \tau, \alpha_{init}, \eta_{init}, b_{init})$

```
 1: // Initialize availability, accuracy and
    counts
 2: $\hat{\alpha_s} \leftarrow \alpha_{init,s}, \forall s, s', \#o_{s',s} = 0, \#o_{null,s} = 0$
 3: // Execution Loop
 4: for $t = 1$ to $\infty$ do
 5:   $b \leftarrow b_{init}$
 6:   loop
 7:     // Choose explore or exploit action
 8:     if $\rho > \frac{1}{t}$ then
 9:       $a \leftarrow$ random action
10:     else
11:       $a \leftarrow \pi(b)$
12:     end if
13:     // update belief using transitions $\tau$,
        receive observation $o$
14:     $b \leftarrow \tau(b, a), o \leftarrow \Omega(b, a)$
15:     // if a = ask, update availability
        based on $o_{null}$ and accuracy based on
        $o_{s'}$
16:     if $a = a_{ask}$ then
17:       if $o = o_{null}$ then
18:         $\forall s, \hat{\alpha_s} \leftarrow (1 - \frac{1}{t}b(s))\hat{\alpha_s}, \#o_{null,s} \leftarrow b(s)$
19:       else
20:         #observed $o_{s'}$
21:         $\forall s, \hat{\alpha_s} \leftarrow (1 - \frac{1}{t}b(s))\hat{\alpha_s} + \frac{1}{t}b(s)$
22:         $\forall s, \#o_{s',s} \leftarrow b(s)$
23:         for $s \neq s', \hat{\eta_s} \leftarrow (1 - \frac{1}{t}b(s))\hat{\eta_s}$
24:         $\hat{\eta_s} \leftarrow (1 - \frac{1}{t}b(s'))\hat{\eta_{s'}} + \frac{1}{t}b(s')$
25:       end if
26:     end if
27:     // is $\hat{\alpha}$ different than $\alpha_{init}$?
28:     if for any $s$, $\chi^2(s) > 3.84$ for $\alpha_s$ or $\eta_s$ then
29:       $\alpha_{init,s} \leftarrow \hat{\alpha_s}, \eta_{init,s} \leftarrow \hat{\eta_s}$
30:       $\pi \leftarrow$ SOLVE_POLICY$(\alpha_{init})$
31:     end if
32:   end loop
33: end for
```

lectively update the policy when the estimated availability or accuracy of a human has changed significantly from the current HOP-POMDP estimate. We determine if any availability or accuracy has significantly changed using Pearson χ^2 test which tests the difference between an observed set of data and the expected distribution of responses. For example, with availability $\alpha_s = 0.7$ we would expect that only about 30% of the received observations are o_{null}. To compute the statistic, we define the number of observations from state s as $n_s : n_s = \sum_{s'} \#o_{s',s} + \#o_{null,s}$

Then we define $\chi^2(s) =$

$$\frac{(\sum_s' \#o_{s',s} - n_s\alpha_s)^2}{n_s\alpha_s} + \frac{(\#o_{null} - n_s(1-\alpha_s))^2}{n_s(1-\alpha_s)} \quad (5)$$

to test whether the observed availability $\hat{\alpha_s}$ is different than the initialized $\alpha_{init,s}$ or the accuracy $\hat{\eta_s}$ is different than $\eta_{init,s}$ with 95% confidence ($\chi^2(s) > 3.84$, Lines 27-31). If so, then it is unlikely that our current HOP-POMDP represents the humans in the environment, the approximations of all accuracies and availabilities must be updated, and the HOP-POMDP policy must be recomputed. The confidence parameter can be adjusted to further reduce the number of policy recalculations (*e.g.,* 99% confidence would require $\chi^2(s) > 6.64$). We expect our algorithm to recompute few times in contrast to the prior algorithms which recalculate each time the hypothesized POMDP actions conflict.

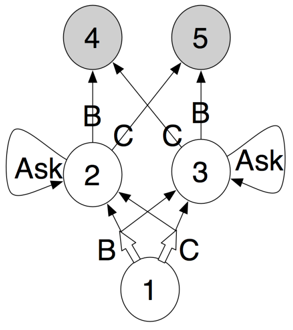

Figure 1: The robot starts at state 1 and can take actions to travel to states 4 (reward -10) or 5 (with reward 10). There are humans in states 2 and 3 that the robot can decide to ask so that it travels to state 5 to maximize its reward.

Experimental Results

In order to test our learning algorithm, we constructed a benchmark HOP-POMDP with two available humans. We show that our algorithm significantly reduces the number of HOP-POMDP policies that must be computed compared to prior work. Additionally, compared to these approaches and other explore/exploit learning algorithms for POMDPs (*e.g.,* (Kearns and Singh 2002; Cai, Liao, and Carin 2009)), we show that our algorithm converges towards the true accuracy and availability of human observation providers without requiring an additional oracle to provide true state observations. As a result, the algorithm is more tractable to execute in a real environment.

Benchmark HOP-POMDP

Our benchmark HOP-POMDP contains 5 states and 2 actions with two humans (states 2 and 3) and two final states (4 and 5) (Figure 1).The robot starts at state 1 and chooses to take action B or C, where

$$T(1,B,2) = 0.75 \qquad T(1,B,3) = 0.25$$
$$T(1,C,2) = 0.25 \qquad T(1,C,3) = 0.75$$

The robot can then execute B or C from 2 and 3 to 4 or 5:

$$T(2,B,4) = 1.0 \qquad T(2,C,5) = 1.0$$
$$T(3,B,5) = 1.0 \qquad T(3,C,4) = 1.0$$

However, the reward for state 4 is -10 and the reward for state 5 is +10. The robot has the opportunity to ask for help in states 2 and 3 to ensure it receives +10. The costs of asking when the humans respond are $R(2, a_{ask}, 2, o) = R(3, a_{ask}, 3, o) = -1$ and when they do not respond $R(2, a_{ask}, 2, o_{null}) = R(3, a_{ask}, 3, o_{null}) = 0$.

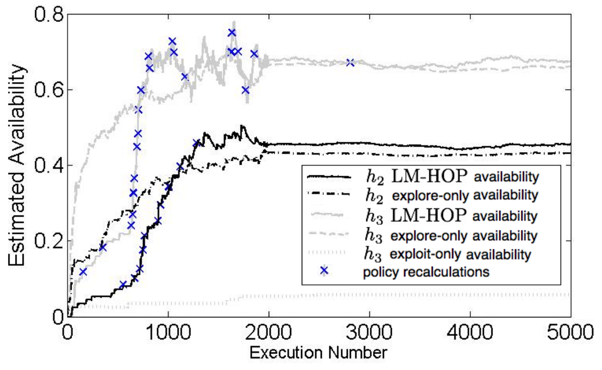

Figure 2: The estimated availability of h_2 (light grey) and h_3 (black) over 5000 executions of the HOP-POMDP with true availability $\alpha_2 = 0.7$ and $\alpha_3 = 0.4$.

Depending on the availability and accuracy of the humans h_2 and h_3, the optimal policy will determine whether the robot should take action B or C from state 1 and whether it should ask at the next location. This POMDP is similar in size and complexity to other POMDP benchmarks such as the Tiger Problem, and we will compare our results to those of other results on similar problem sizes.

Benchmark Results

We first tested the LM-HOP algorithm, assuming that the humans were 100% accurate. We initialized the availability $\alpha_{init,2} = \alpha_{init,3} = 0$ to understand how fast the algorithm would converge to true availabilities. As an example, Figure 2 shows the estimated availability of h_2 (light grey) and h_3 (black) over 5000 executions of the HOP-POMDP with true availability $\alpha_2 = 0.7$ and $\alpha_3 = 0.4$. We then tested the simultaneous learning of both the availability and accuracy of humans in the environment. We initialized the availability of both humans to $\alpha_{init,2} = \alpha_{init,3} = 0.0$ and the accuracy of both humans to $\eta_{init,2} = \eta_{init,3} = 1.0$. We then varied the true accuracy and availability of our humans to understand the learning curves. Figure 3 shows an example of the learning rates for the availability and accuracy of h_2 and h_3 when true accuracy of each humans is 0.5 and the true availabilities are $\alpha_2 = 0.7$ and $\alpha_3 = 0.4$.

Explore/Exploit Strategy We find that our LM-HOP algorithm and the explore-only algorithm closely approximate the true availability and accuracy. The approximate availabilities in Figure 2 are 67% (compared to 70%) and 41% (compared to 40%). Compared to the explore-only algorithm (Figure 2 dot-dash lines), our LM-HOP algorithm is slower to start converging because it tries to maintain high expected reward by exploiting the current best policy of not asking for help. If we modified the explore-exploit learning parameter ρ, our LM-HOP algorithm would spend more time exploring at first and would converge faster. Our algorithm does converge faster in the end because, after recalculating the policy, the policy includes asking. The exploit-only algorithm learns very slowly in our example because the initial optimal policy does not include ask actions.

We also compare the average reward (collected over

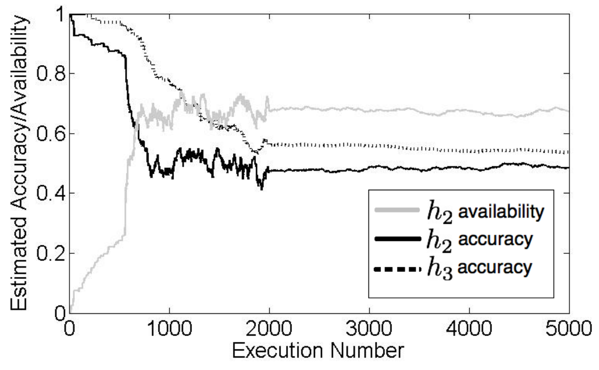

Figure 3: The estimated availability (light grey) is learned at the same time as the accuracy of the humans (black). h_2 is visited more often and his accuracy (0.5) is learned faster.

10000 executions) between the learning algorithms and the optimal policy reward if true accuracy and availability were known. Although the explore-only algorithm performs similarly to our LM-HOP algorithm in terms of number recalculations and convergence, it earn only an average -.215 reward compared to our algorithm which earns 3.021. The exploit-only algorithm earns 4.742 reward, and the optimal policy initialized to the true availability and accuracy earns 5.577. While the exploit-only algorithm earns more on average than our LM-HOP algorithm, we find that it earns very little reward when it chooses the path with lower availability first and high reward otherwise. Our algorithm does not have this dichotomy, and therefore we believe it performs better. We found no statistical difference between the average reward received when only learning availability and when learning availability and accuracy.

Comparison to Prior POMDP Learners The X's on Figure 2 show the number of policy recalculations while learning availability only. On average, our LM-HOP algorithm recalculated the policy 20-30 times when learning only availability. When learning both accuracy and availability, our algorithm recalculated the policy an average of 10 more times for a total of 30-40. Additionally, our algorithm converges to the true availability and accuracy within 1000-2000 executions of a single HOP-POMDP. Overall, the LM-HOP algorithm recalculates the policy significantly fewer times than the prior work's $10^3 - 10^6$ recalculations of 20 POMDPs of the similar-sized Tiger Problem (Jaulmes, Pineau, and Precup 2005; Doshi, Pineau, and Roy 2008).

Real-World Building Results

In order to understand how the HOP-POMDP policy differs from traditional POMDPs in a real-world environment, we model an indoor robot navigation problem in which the human observation providers are the occupants of the offices around the building. We gathered availability data through a study of 78 offices in our building (Rosenthal, Veloso, and Dey 2011). The availability of our office occupants is shown in Figure 4(a) where darker gray represents higher availability.

We tested the north portion of the building from the hall-

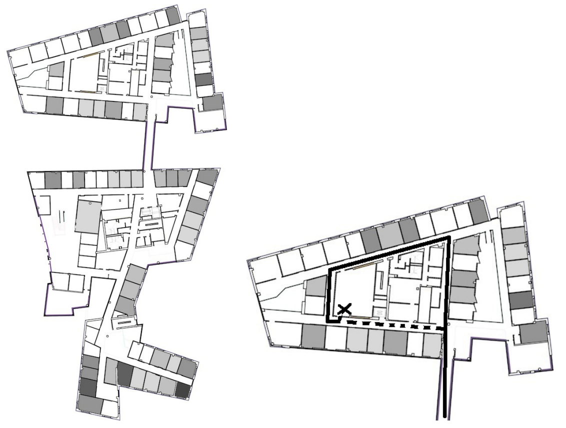
(a) Availability (b) Policies

Figure 4: (a) Availability of humans in our building - darker gray represents higher availability. (b) The HOP-POMDP takes a longer route with more available people.

way to the lab marked with an X, with a graph consisting of 60 nodes including 37 offices (Figure 4(b)). Taking an action between a current node s and a connected node s' on the graph had the following transition probabilities $T(s, a, s) = 0.1$ and $T(s, a, s') = 0.9$. We assigned a constant cost $\lambda = -1$ as the cost of asking each occupant and a reward $R(final, a) = 100.0$ for reaching its laboratory space. Our HOP-POMDP policy takes a longer route that has more available building occupants compared to the shortest path (Figure 4(b)). We conclude that our HOP-POMDP policy results in a path that increases the likelihood of finding an occupant to query which in turn increases the number of successful navigation tasks.

Conclusions

As robots become more ubiquitous in our environments, they will increasingly need to ask for help from humans located there rather than depending on supervisors or oracles. We introduce the Human Observation Provider POMDP (HOP-POMDP) to model human location, availability, accuracy, and cost of asking in the environment as they provide observations to the robot. In particular, the HOP-POMDP incorporates availability and accuracy into the observation function when the robot performs action a_{ask}. We then introduce our LM-HOP explore/exploit algorithm to learn the availability and accuracy of the humans while executing the current best policy. Unlike previous approaches to learning POMDPs, our LM-HOP algorithm

1. instantiates only a single hypothesis HOP-POMDP,
2. recomputes the HOP-POMDP policy only when the learned accuracy/availability change significantly, and
3. does not depend on a human to always be accurate and available in order to learn.

We demonstrate that in terms of the explore/exploit strategy, our algorithm converges faster and with consistently higher reward than the explore-only and exploit-only algorithms. Compared to prior learning algorithms, we demonstrated that our algorithm, with a single hypothesized HOP-POMDP, recomputes POMDP policies at least 2 orders of magnitude fewer times. Our LM-HOP algorithm is effective in approximating the true availability and accuracy of humans without depending on oracles to learn.

Acknowledgments This research was supported by the National Science Foundation award number IIS-1012733, and a scholarship from the National Physical Science Consortium. The views and conclusions contained in this document are those of the authors only.

References

Aberdeen, D. 2003. A (revised) survey of approximate methods for solving pomdps. *National ICT Australia, Technical Report*.

Armstrong-Crews, N., and Veloso, M. 2007. Oracular pomdps: A very special case. In *ICRA '07*, 2477–2482.

Cai, C.; Liao, X.; and Carin, L. 2009. Learning to explore and exploit in pomdps. In *NIPS*, 198–206.

Cohn, D.; Atlas, L.; and Ladner, R. 1994. Improving generalization with active learning. *Machine Learning* 15(2):201–221.

Doshi, F.; Pineau, J.; and Roy, N. 2008. Reinforcement learning with limited reinforcement: using bayes risk for active learning in pomdps. In *ICML '08*, 256–263.

Fogarty, J.; Hudson, S. E.; Atkeson, C. G.; Avrahami, D.; Forlizzi, J.; Kiesler, S.; Lee, J. C.; and Yang, J. 2005. Predicting human interruptibility with sensors. *ACM ToCHI* 12(1):119–146.

Jaulmes, R.; Pineau, J.; and Precup, D. 2005. Active learning in partially observable markov decision processes. In *ECML 2005*, volume 3720, 601–608.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101(1-2).

Karami, A.-B.; Jeanpierre, L.; and Mouaddib, A.-I. 2009. Partially observable markov decision process for managing robot collaboration with human. *Tools with Artificial Intelligence* 0:518–521.

Kearns, M., and Singh, S. 2002. Near-optimal reinforcement learning in polynomial time. *Machine Learning* 49:209–232.

Littman, M. L.; Cassandra, A. R.; and Kaelbling, L. P. 1995. Learning policies for partially observable environments: Scaling up. In *ICML*, 362–370.

Madani, O. 2000. Complexity results for infinite-horizon markov decision processes. *Ph.D. dissertation, University of Washington*.

Papadimitriou, C., and Tsisiklis, J. 1987. The complexity of markov decision processes. *Mathematics of Operations Research* 12(3):441450.

Rosenthal, S.; Biswas, J.; and Veloso, M. 2010. An effective personal mobile robot agent through a symbiotic human-robot interaction. In *AAMAS '10*, 915–922.

Rosenthal, S.; Dey, A. K.; and Veloso, M. 2009. How robots' questions affect the accuracy of the human responses. In *Ro-Man*, 1137–1142.

Rosenthal, S.; Veloso, M.; and Dey, A. K. 2011. Is someone in this office available to help? proactively seeking help from spatially-situated humans. *Journal of Intelligence and Robotic Systems*.

Schmidt-Rohr, S. R.; Knoop, S.; Lösch, M.; and Dillmann, R. 2008. Reasoning for a multi-modal service robot considering uncertainty in human-robot interaction. In *HRI '08*, 249–254.

Shiomi, M.; Sakamoto, D.; Takayuki, K.; Ishi, C. T.; Ishiguro, H.; and Hagita, N. 2008. A semi-autonomous communication robot: a field trial at a train station. In *HRI '08*, 303–310.