# Mobile Robot Planning to Seek Help with Spatially-Situated Tasks

**Stephanie Rosenthal and Manuela Veloso**
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

## Abstract

Indoor autonomous mobile service robots can overcome their hardware and potential algorithmic limitations by asking humans for help. In this work, we focus on mobile robots that need human assistance at specific spatially-situated locations (e.g., to push buttons in an elevator or to make coffee in the kitchen). We address the problem of what the robot should do when there are no humans present at such help locations. As the robots are mobile, we argue that they should plan to proactively seek help and travel to offices or occupied locations to bring people to the help locations. Such planning involves many trade-offs, including the wait time at the help location before seeking help, and the time and potential interruption to find and displace someone in an office. In order to choose appropriate parameters to represent such decisions, we first conduct a survey to understand potential helpers' travel preferences in terms of distance, interruptibility, and frequency of providing help. We then use these results to contribute a decision-theoretic algorithm to evaluate the possible choices in offices and plan where to proactively seek help. We demonstrate that our algorithm aims to minimize the number of office interruptions as well as task completion time.

## Introduction

Mobile robots have the ability to perform a variety of tasks for us today including giving visitors directions in malls (Shiomi et al. 2008) and tours in museums (Nourbakhsh et al. 2005), and acting as companions for individual users (Rosenthal, Biswas, and Veloso 2010). However, robots are limited by their sensing and actuation capabilities and state-action policies. While it is sometimes possible for robots to overcome their limitations through learning better policies (Argall et al. 2009) or asking for human help to reduce uncertainty (Fong, Thorpe, and Baur 2003; Nicolescu 2003; Rosenthal, Biswas, and Veloso 2010) or to take control (Shiomi et al. 2008), neither an autonomous nor human-controlled robot with actuation limitations could ever perform their limited actions without new hardware. A robot without manipulators will never be able to pick up objects, and a robot without legs will never be able to walk up stairs.




Figure 1: Our CoBot robots are capable of autonomous localization and navigation, but cannot manipulate objects.

If tasks require these actions, we argue that robots should plan to complete the task by seeking help a human in the environment (Rosenthal, Veloso, and Dey 2012).

Our CoBot robots (Figure 1), for example, are capable of autonomous localization and navigation (Biswas and Veloso 2010) and can perform tasks such as delivering messages to building occupants and transporting objects from one location to another. However, they do not have manipulators to be able to pick up objects or push elevator buttons to travel between floors of our building. In order to overcome their limitations, we have them plan to request help from people in the physical environment in order to complete their tasks.

We address the problem of determining where robots can proactively find people to help them with their tasks. Many of the actions that our CoBots need help with are spatially-situated actions - those that must be performed in a particular location or set of locations in the environment (*e.g.,* at the elevator or in the kitchen). People in the environment visit these locations at different frequencies. When they are there, the potential cost of helping the robot is low. However, the robot may have to wait a long period of time for someone to arrive. Alternatively, we propose that because the robot is mobile, it could travel to offices in our building to find immediate help at the higher cost of interrupting the office worker. Identifying an optimal help policy hinges on evaluating this tradeoff between interruption costs to the people in the environment and task completion time.

We first illustrate the process of seeking spatially-situated help to use the elevator and highlight the challenges of planning to request such help. Then, we present the results of a survey in which participants were asked to rate their preferences about where our robots navigated to find help, in order to understand the potential tradeoffs the robot will need to make when planning who to ask. Based on the results of the study, we contribute a decision-theoretic algorithm that takes into account both the robot's costs and humans' preferences when planning to ask. Importantly, the algorithm first waits at the help location for some time and then replans in case someone is not willing to help the robot or is not available. Finally, we demonstrate in simulation and on our real robot that our algorithm, which includes the result of our study, limits the number of occupants that are interrupted in their offices while still completing tasks faster than waiting indefinitely at the help location.

## Related Work

Much recent work has focused on different techniques to allow robots to reason about their own limitations and capabilities to proactively ask for help from users (Lee et al. 2010; Rosenthal, Biswas, and Veloso 2010), supervisors (Fong, Thorpe, and Baur 2003; Shiomi et al. 2008; Yanco, Drury, and Scholtz 2004), teachers and demonstrators (Argall et al. 2009; Hayes and Demiris 1994; Lockerd and Breazeal 2004; Nicolescu 2003), and passers-by in the environment (Asoh et al. 1997; Hüttenrauch and Eklundh 2006; Michalowski et al. 2007; Weiss et al. 2010). However, this work largely assumes 1) that help is always available when the robot needs it or the robot can wait indefinitely until it arrives, and 2) the robot can receive the help from its current location.

Few planning algorithms for robots have included plans for the possibility of needing help while executing tasks (Armstrong-Crews and Veloso 2007; Rosenthal, Biswas, and Veloso 2010; Rosenthal, Veloso, and Dey 2011) or the possibility modifying the questions based on the human helper (Fong, Thorpe, and Baur 2003; Shiomi et al. 2008). However, work in modeling multi-robot teams has included plans for helping other robots by communicating new observations to peers (*e.g.,* (Roth, Simmons, and Veloso 2006)). In robots that do ask while executing tasks (*e.g.,* (Asoh et al. 1997; Chernova and Veloso 2008; Fong, Thorpe, and Baur 2003; Grollman and Jenkins 2007; Katagami and Yamada 2001; Weiss et al. 2010)), the planning algorithms assume that they will not need to ask, and only react by asking when they determine they need more information. In this work, our planning algorithm takes into account the possible need for help at planning time and determines all actions based on where it will need help and who is available to help.

Many asking policies also assume that the robot needs help at its current location. Robots typically have contacted humans through user interfaces on computers (*e.g.,* (Shiomi et al. 2008; Yanco, Drury, and Scholtz 2004)) or mobile devices (*e.g.,* (Fong, Thorpe, and Baur 2003)), or in person (*e.g.,*(Lee et al. 2010; Weiss et al. 2010)). A notable exception is when a robot must follow a person in order to learn routes to a particular location (*e.g.,* (Asoh et al. 1997)). However, we recently showed that office workers were willing to leave their office to perform tasks such as moving chairs out of the way for a mobile robot (Rosenthal, Veloso, and Dey 2012; Rosenthal 2012). In this work, our robots require humans to help at spatially-situated locations, and as a result they ask humans to travel with them to the help locations.

Next, we illustrate the need for spatially-situated help for our CoBot robots.

## Example of Spatially-Situated Help

We aim for our robots to take into account helper preferences to determine who to ask for help in order to reduce task completion time while maintaining high usability for helpers. In order to illustrate the many decisions and challenges that must be considered, we illustrate the possible scenarios that CoBot finds itself in when it must use the elevator and tradeoffs the robot must make in determining where to find help (Figure 2, subfigures referenced below).

**(a) Waiting at the Help Location.** People arrive in help locations at varying frequencies. If CoBot navigates to the elevator, it may or may not find a person who is also trying to use the elevator and who could help it get to the correct floor. The elevators are less frequently used during class time, for example, so the robot could be waiting a long time, delaying its task completion. The benefit of asking the person already at the help location is that they are already performing the action themselves and should have little cost to helping the robot. If CoBot waits at the help location for a long time, it may instead be beneficial to proactively navigate to find a person in an office who can help immediately. In this work, we contribute an algorithm to determine how long to wait before navigating away from the help location.

**(b) Deciding Where to Proactively Travel.** If CoBot travels to find a person in an office who could help, they must travel together back to the elevator. Determining who to ask for help is important in maintaining robot usability over long term deployments. In particular, we identify several possible factors of the decision of who to ask. First, the distance between CoBot's current location and the location of the office helpers may be a factor. Once CoBot arrives at an office, other factors may include the potential helper's availability, meeting schedule, or unwillingness to respond due to too many questions from the robot. If the helpers are willing to help, another factor is the travel distance to the help location with CoBot. When the robot and helper arrive at the help location, there may be a new possible helper at the help location which may factor in to the decision about whether to help again later. Finally, if the helper does not know how to help (*e.g.,* use the coffee maker), it may impact their willingness to help and success of actually helping.

We performed a study to understand which factors are important for robots to take into account. We then contribute a tradeoff to determine who to ask for help based on these costs of asking each potential helper in different office locations.

**(c) Requesting Spatially-Situated Help.** Once a person is at the help location, CoBot also must plan its task questions. To use the elevator, CoBot asks the helper to press up/down button and notify it when they have done so. It also tells the person to hold the doors open so that it does not get

(a) (b) (c)

Figure 2: (a) CoBot autonomously navigates to the elevator that it must use to reach a destination on a different floor. It waits at the elevator for someone to arrive and help. (b) If no one arrives at the elevator, CoBot replans to find someone in an office rather than waiting longer, determines the best office to ask, and proactively navigates there. At the office, CoBot asks a person in the office if they are willing to help. If not, CoBot replans to find someone else. (c) When CoBot and a helper arrive at the elevator, it can ask for the help it needs. CoBot asks them to press the elevator buttons as well as to hold the doors open.

stuck. Then, CoBot waits for the elevator doors to open and navigates autonomously inside. After stopping inside the elevator, it asks the person to press the appropriate floor number and to hold the doors open when they get to that door. to again assist in keeping the doors open when the elevator gets to the correct floor. Upon reaching the correct floor, CoBot navigates out of the elevator and then continues navigating to its destination.

## Study of Tradeoffs when Providing Spatially-Situated Action Help

Because our CoBot robots are deployed in the environment long term, we would like them to improve their functionality by asking for help but also ask in a usable way. In order to understand the tradeoffs people would make in determining where a robot should travel for spatially-situated help, we conducted a web survey about preferences for when, under what conditions, and how frequently they would be willing to help a robot. In the first half of the survey, subjects were shown a partial map of our building with different configurations of people in offices who could be available, different locations of the robot, and different locations for receiving help - the elevator or the kitchen to make coffee (Figure 3). They were asked which person the robot should choose to ask for help. In the second half of the survey, participants were told to suppose that they were the one being asked for help and answered questions about their willingness to help under different conditions of interruptibility, recency of the last time they could be asked, and frequency of questions they could be asked per week.

Fifty participants were recruited through a Carnegie Mellon University website that hosts advertisements for human-subject studies. The survey contained 100 questions and took about 45 minutes for participants to complete. We will use our study results to contribute an our algorithm which determines where to navigate and who to ask for help.

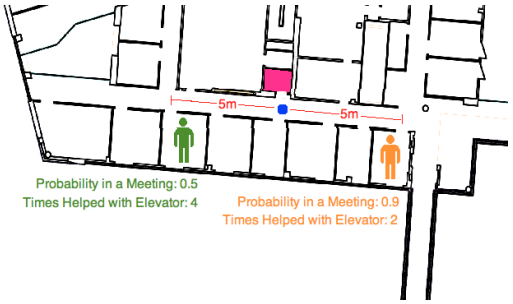


Figure 3: Participants were asked to judge which of the two available people the robot (blue dot) should ask for help in a location (pink) given their probabilities of being in meetings and the number of times they'd helped the robot before.

### Results

We evaluate the participant responses to understand how CoBot should ask for spatially-situated help and what to model about office helpers.

**First Location to Find Help.** We found that in the scenarios where there was one person in an office near the help location and there was a chance of another person at the help location, participants indicated that the robot should check if there was a person at the help location 60% of the time when asking for help to use the elevator and 80% of the time for making coffee (Figure 4). As expected, participants noted that they chose the help location because the helper would already be performing the actions and it would not be much harder to help the robot (the helpers at the help location have a lower cost of helping). We also asked participants whether they would be more or less likely to help the robot from their office if it had already checked the help location, and 80% of participants said that they would be. This indicates that our algorithm should first always check the help location before navigating away to choose an office helper.

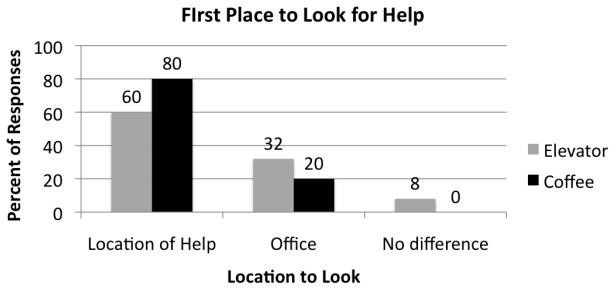


Figure 4: A majority of participants specified that they thought the robot should look in the location of help before asking in offices.

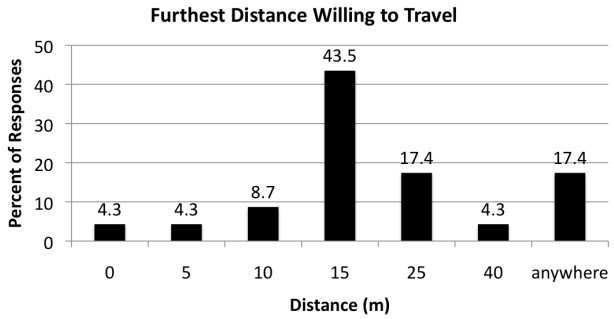


Figure 5: Most participants were willing to travel up to 15 meters to help the robot, but 17.4% responded that they were willing to help from anywhere in the building.

**Travel Distance.** We tested whether the robot should ask the closer person to the elevator and how far people were willing to travel to help the robot (Figure 5). When shown different scenarios of the robot and available people in offices, surprisingly only 75% participants responded that the robot should choose the closer person, irrespective of whether the person would be helping with the elevator or coffee. Participants said the robot could ask someone further away if it would pass the further person first. This indicates that while the travel distance of a helper to the help location is important, the distance of the robot to the helper should also be taken into account when determining who to ask.

**Question Timing and History.** While other algorithms have taken into account some human state such as availability or interruption (whether someone is in a meeting), we hypothesized that the frequency and recency of questions would also significantly affect a person's likelihood to want to help the robot. These two parameters reflect the fact that the robot is available long term and has a history with these helpers. We asked participants to predict whether they would be likely to help the robot depending on whether they were in a meeting, how many times they had been asked in the last week, and the last time they had been asked. We specifically test the statistical significance of each of these three parameters to validate their necessity in our algorithm using a $\chi^2$ statistic and found that they are all statistically significant ($p < 0.05$). These results confirm our hypothesis that recency and frequency of help do play a significant role in potential helpers' willingness to help the robot.

**Unnecessary Help.** Because we were specifically concerned about the possibility that office helpers would feel unnecessary if they found another person already at the help location, we told participants to suppose this situation happened and asked when they would be willing to help the robot again. Our results show that participants wanted the robot to ask them less frequently after helping in this scenario compared to when there was no one else present at the help location. In particular, 69% of office helpers were willing to help the robot within 8 hours if there was no one at the help location compared to 47% when they were told that someone had arrived. Despite feeling unnecessary, 83% participants said they would be willing to help the robot again.

To summarize, robots should consider the spatial locations of the potential helpers and the help location to first try to find someone at the help location and then, if necessary, to find someone in an office close to it. Additionally, robots should model their human helpers to maintain usability with requests for help over time by limiting the possibility that an office helper will find a possible helper at the help location and by limiting the frequency of requests for help and increasing the time between requests for help to individuals in offices. Next, we use these results to contribute algorithms for planning where to travel to proactively request spatially-situated help during execution.

## Spatially-Situated Help Algorithms

We present our spatially-situated action help algorithm for the robot to execute when it requires actuation help. Based on our survey findings, our algorithm always navigates to the help location first to wait for someone there to request help from. Then, only after waiting without someone agreeing to help, the algorithm employs our proactive travel tradeoff to determine who to seek help from in offices also based on our survey findings. We will demonstrate that our robots can receive help faster by proactively navigating to find help compared an algorithm that only wait at the help location. Additionally, we show that our algorithm is more usable than an algorithm that always proactively navigates because it asks at the help location first which our survey shows is preferable to our participants.

### Spatially-Situated Action Help (SSAH)

We contribute our Spatially-Situated Action Help Algorithm to plan (and replan) to proactively seek help to overcome its actuation limitations while completing tasks in the environment. When CoBot reaches an action in its plan that it cannot complete autonomously, it calls SSAH(help,$l_{help}$) with the type of help it needs and the location $l_{help}$ where it should be receiving help (Algorithm 1).

The algorithm first initializes the taskSuccess indicator variable, list of offices, sets a waitThreshold to indicate how long to wait for a person to answer in offices, always chooses to travel to the help location $l_{help}$ first (line 1). In order to ensure that the robot does not wait too long at the help location, we set the taskThreshold for the max time that the robot should wait before proactively navigating to offices (line 2) (See Setting Wait Threshold for details).

**Algorithm 1** SSAH(help,$l_{help}$)

```
1: taskSuccess ← false, waitThreshold ← waitTime, offices ← getAllOffices(), travelLoc ← l_help
2: (taskThreshold,loc) ← PTT(l_help,l_help,offices)
3: while ¬taskSuccess AND |offices| > 0 do
4:    Navigate(loc)
5:    Ask(helptype)
6:    if loc = l_help then
7:       willing ← WaitForResponse(taskThreshold)
8:    else
9:       willing ← WaitForResponse(waitThreshold)
10:   end if
11:   if willing then
12:      taskSuccess ← ExecuteWithHuman(help)
13:      updateHelper(loc)
14:   end if
15:   if ¬taskSuccess then
16:      (time,travelLoc) ← PTT(travelLoc,l_help,offices)
17:      offices ← offices - travelLoc
18:   end if
19: end while
20: return taskSuccess
```

After setting these values, the robot navigates to find a helper at the help location (line 3). Then, it asks for help (line 5) and waits for a response depending on where it is (lines 6-10). If someone is willing to help, it tries to execute with them and updates its information about the helper at location lot (line 11-14). If it was not successful or could not find a person, it picks a new lowest cost location using our Proactive Travel Tradeoff (PTT) (line 15-18), subtracts the office from its list so that it doesn't revisit it (line 16). The PTT tradeoff uses the updated location information to accurately determine the cost of asking each office for help. The algorithm returns when it either has been helped successfully or there are no more offices to visit (line 20).

### Proactive Travel Tradeoff (PTT)

In order to determine who to ask for help, our Proactive Travel Tradeoff ($PTT$) computes the expected cost to complete the help with proactive navigation to each possible office and chooses the minimum cost office. The costs are computed based on our survey findings. This tradeoff is decision-theoretic (Lehmann 1950; Schoemaker 1982) in that it computes the best action with the lowest expected cost by taking into account helpers':

- availability $\alpha$ or probability the person is in their office,
- interruptibility $\iota$ or the probability the person is not busy
- expertise $e$ or the probability of successful help,
- location $l$ of offices,
- recency of help $r$ or the time since person's last help,
- frequency of help $f$ per week, and
- willingness to answer $w(\iota, r, f)$ based on past experiences and current interruptibility.

Using this model of helpers, the $PTT$ tradeoff computes the cost of asking at each office $o$ including:

- $COT$ cost traveling to the office $o$,
- the probability of a person being available $\alpha_o$ and the cost of asking them to help $COA$,
- the willingness of them to help $w(\iota_o, r_o, f_o)$ and the cost of the helper traveling back to the help location $COTH$,
- the probability of being unnecessary $\alpha_{help}$ and the corresponding cost $COU$,
- the probability of failing due to expertise $e_o$ and the cost of failure $COF$,
- the cost of replanning from the current location with $PTT$ in case a person is not available or willing to help.

Formally, $PTT(l_{start},l_{help},\textit{offices}) = \min_{o \in \textit{offices}}$

$$
\begin{aligned}
&COT(l_{start}, l_o) \\
&\quad + \alpha_o \Big[ COA(\iota_o, r_o, f_o) \\
&\qquad + w(\iota_o, r_o, f_o) \big[ COTH(l_o, l_{help}) \\
&\qquad\quad + \alpha_{help} * COU(l_{help}) \\
&\qquad\quad + (1 - e_o)[COF(l_o, l_{help}) \\
&\qquad\qquad + PTT(l_{help}, l_{help}, \textit{offices} - o)] \big] \\
&\qquad + (1 - w(\iota_o, r_o, f_o)) * PTT(l_o, l_{help}, \textit{offices} - o) \Big] \\
&\quad + (1 - \alpha_o) * PTT(l_o, l_{help}, \textit{offices} - o)
\end{aligned}
$$

While this tradeoff finds the optimal office, it is intractable to compute for any large number of offices given the recursion for failure and the branching factor equal to the number of offices in the building. For our implementation, we compute the greedy best office that does not recurse on $PTT$ and instead uses a constant FAIL_COST that is greater than the cost of successfully asking any office.

### Setting Task Thresholds

The SSAH algorithm also required a task threshold to wait at the help location before starting to proactively ask for help. We explored two methods for setting the threshold. First, we looked to queueing theory to model the arrival of a person at the help location using a Poisson Process (Arlitt and Williamson 1997). If the robot were to model the likelihood of finding help using this distribution, however, we found that the probability that a person will arrive increases over time so the robot never proactively navigates away from the help location. Second, we used the Buy or Rent problem (Ski Rental problem) as an analogy for setting this threshold, in which there is a small cost to continuing to wait and a large cost to finding a person elsewhere (Karlin et al. 1994). The solution to this problem is not optimal in hindsight but ensures that the robot will take no longer than twice the expected time to proactively find a person to complete the task. We choose this solution because not only does threshold task completion time but also has the property that it will wait at the help location.

## Experiments

Our SSAH algorithm combines waiting at the location of help with proactive navigation in order to speed task completion while limiting the number of questions asked at offices. In order to characterize the performance of our greedy $PTT$

algorithm, we performed real-world and simulated experiments. These results are meant to be examples of how the algorithm behaves and not a complete analysis of behavior over all possible environments.

## Simulated Experiments

We simulated four hallways of our building, the elevator nearby, and the occupants in their offices. The hallways contain 28 offices in an elongated rectangle shape with the elevator in the middle of one of the long hallways. Our real building contains occupancy sensors in each office, so we define $\alpha_o \in \{0, 1\}$ randomly. Additionally, we generate random probabilities for willingness to respond $w \in [0, 1]$. Finally, we use real world data to compute the frequency that a person appears at the elevator ($\alpha_{help}$). Through an observational study, we found that on one floor $\alpha_{help} = 5$min and on another floor $\alpha_{help} = 10$min.

We test our SSAH algorithm against two other algorithms. In the Wait Only algorithm, the robot travels to the help location and waits indefinitely until someone helps there. This algorithm is guaranteed to find a low cost helper at the help location but may result in long task completion times. In the Proactive Only algorithm, the robot uses the $PTT$ tradeoff to immediately find a helper in an office. This algorithm is guaranteed to find help quickly but at the cost of always interrupting an office worker.

**Time to Find Help** Our SSAH algorithm finds help faster than Wait Only but slower than Proactive Only (Figure 7). The Wait Only algorithm that only waits at the elevator has high variance and takes on average 5 minutes and 10 minutes respectively on the two different floors of the building. The Proactive Only algorithm which goes directly to find help in an office rather than waiting at the elevator almost always finds help in under three minutes. The SSAH algorithm is in between the two algorithms as it waits first and then navigates away.

**Number of Offices Asked** While the Proactive Only algorithm found a helper faster, it also interrupted people in offices every time it needed help. This is costly according to our survey results. Instead, our SSAH algorithm was able to cut the number of offices visited in half for the 5 minute floor and by 20% for the 10 minute floor compared to the Proactive Only algorithm. While we would like the robot to complete tasks quickly, we also want to make sure that people are willing to help the robot months and even years. The less frequently the robot actually interrupts people in offices and can instead ask people who are already using the elevator, the more usable and deployable we expect the robot and algorithm to be for our building in the long run.

We conclude that while SSAH takes longer to find help than Proactive Only, the reduction in office help requests is significant for the future usability of our robot.

## Real Robot Deployment

In order to understand how our algorithm performs in practice, we conducted an experiment in which the robot performed multi-floor tasks which required help using the elevator. In the first phase of the experiment, we deployed

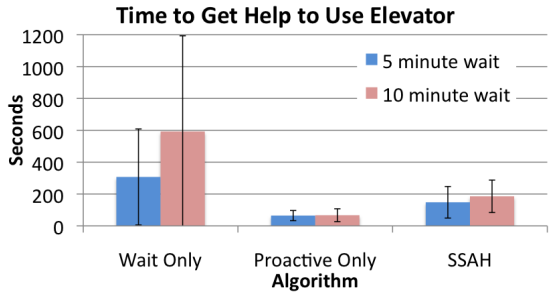


Figure 6: Our SSAH algorithm finds help use the elevator faster than the Wait Only, but not as fast as Proactive Only which always goes directly to find someone in an office.

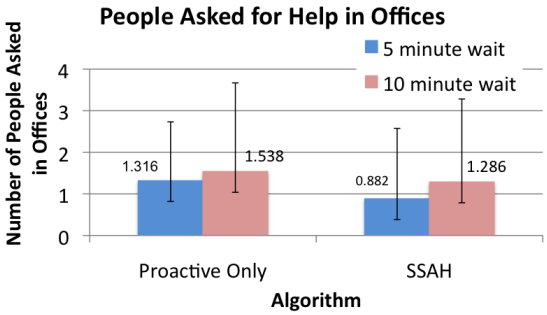


Figure 7: Our SSAH algorithm requires less help from offices, making it more usable for our occupants long-term.

CoBot with the Wait Only algorithm. With this algorithm, the robot was able to accomplish 66 multi-floor tasks which it could not have completed without help, waiting on average 1 minute to find help. Interestingly, this result conflicts with our previous finding that people use the elevator once every 5-10 minutes depending on the floor. We found that people were often following the robot or interested in its deployment and therefore helped more often and faster than what one can expect on a more normal day.

In the second phase, we conducted a preliminary experiment of CoBot using our SSAH algorithm to find help. After waiting for help at the elevator, CoBot contacted a server to access the occupancy sensors in each office of the building and then estimated the interruptibility of those who were available. It then chose the best office using our $PTT$ algorithm, navigated there, and asked for help. If the person refused to help or ignored the robot's question, CoBot would replan using the $PTT$ tradeoff to find another office.

Because the novelty had worn off by the time we deployed this phase, we found the robot waiting times were much more of what we would have expected. CoBot waited an average of 190 seconds for a person to arrive at the elevator before proactively navigating. This result is much shorter than the 5-10 minutes for Wait Only and results in more satisfaction for the people who request the tasks as well as those located around the elevator.

From these results, we conclude that our SSAH algorithm is expected to find help faster than Wait Only without asking as many people in offices as the Proactive Only algorithm.

## Conclusion

Our CoBot robots are capable of autonomous localization and navigation, but have actuation limitations that prevent them from performing some actions such as pushing buttons to use the elevator and making coffee in the kitchen. Interestingly, these limitations require humans to be spatially-situated in the help location in order to help the robots perform these actions. We proposed to take advantage of the fact that our robots are mobile and have them proactively seek humans in offices to travel to the help location. We first conducted a survey to understand potential helpers' preferences about where to navigate and who to ask in the environment based on helper interruptibility, and how recently and frequently the robots might ask them for help. We showed that many helpers thought that the robot should check at the help location to see if anyone was there before navigating to offices, but that many potential helpers were willing to travel when necessary. We used these results to contribute our SSAH algorithm to find spatially-situated action help and our $PTT$ decision-theoretic tradeoff to determine which office to proactively navigate to. Finally, we demonstrated in simulation and in a real-world deployment to that our algorithm balances the time waiting at the elevator with the expected interruption of proactively finding helpers.

## Acknowledgments

This work was partially supported by a Fellowship from the National Science Foundation, and by the National Science Foundation award number NSF IIS-1012733. The views and conclusions contained in this document are those of the authors only.

## References

Argall, B.; Chernova, S.; Veloso, M.; and Browning, B. 2009. A survey of robot learning from demonstration. *Robotics and Autonomous Systems* 57(5):469–483.

Arlitt, M. F., and Williamson, C. L. 1997. Internet web servers: Workload characterization and performance implications. *IEEE/ACM Transactions on Networking* 5(5):631.

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

Asoh, H.; Hayamizu, S.; Hara, I.; Motomura, Y.; Akaho, S.; and Matsui, T. 1997. Socially embedded learning of the office-conversant mobile robot jijo-2. In *IJCAI-97*, 880–885.

Biswas, J., and Veloso, M. 2010. Wifi localization and navigation for autonomous indoor mobile robots. In *ICRA 2010*.

Chernova, S., and Veloso, M. 2008. Multi-thresholded approach to demonstration selection for interactive robot learning. In *HRI '08*, 225–232.

Fong, T. W.; Thorpe, C.; and Baur, C. 2003. Robot, asker of questions. In *Robotics and Autonomous Systems*, volume 42, No. 3-4, 235–243.

Grollman, D. H., and Jenkins, O. C. 2007. Learning robot soccer skills from demonstration. In *International Conference on Development and Learning*, 276–281.

Hayes, G., and Demiris, J. 1994. A robot controller using learning by imitation. In *2nd International Symposium on Intelligent Robotic Systems*, 198–204.

Hüttenrauch, H., and Eklundh, S. 2006. To help or not to help a service robot: Bystander intervention as a resource in human-robot collaboration. *Interaction Studies* 7(3):455–477.

Karlin, A. R.; Manasse, M. S.; McGeoch, L. A.; and Owicki, S. 1994. Competitive randomized algorithms for non-uniform problems. *Algorithmica* 11(6):542–571.

Katagami, D., and Yamada, S. 2001. Real robot learning with human teaching. In *4th Japan-Australia Joint Workshop on Intelligent and Evolutionary Systems*, 263–270.

Lee, M. K.; Kielser, S.; Forlizzi, J.; Srinivasa, S.; and Rybski, P. 2010. Gracefully mitigating breakdowns in robotic services. In *HRI '10: 5th ACM/IEEE International Conference on Human Robot Interaction*, 203–210.

Lehmann, E. L. 1950. Some principles of the theory of testing hypotheses. *Annals of Mathematical Statistics* 21(1):1–26.

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

Michalowski, M.; Sabanovic, S.; DiSalvo, C.; Busquets, D.; Hiatt, L.; Melchior, N.; and Simmons, R. 2007. Socially distributed perception: Grace plays social tag at aaai 2005. *Autonomous Robots* 22(4):385–397.

Nicolescu, M. N. 2003. *A framework for learning from demonstration, generalization and practice in human-robot domains*. Ph.D. Dissertation, University of Southern California.

Nourbakhsh, I.; Hamner, E.; Porter, E.; Dunlavey, B.; Ayoob, E. M.; Hsiu, T.; Lotter, M.; and Shelly, S. 2005. The design of a highly reliable robot for unmediated museum interaction. In *International Conference on Robotics and Automation (ICRA '05)*, 3225 – 3231.

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.; Veloso, M.; and Dey, A. K. 2011. Learning accuracy and availability of humans who help mobile robots. In *AAAI 2011*, 1501–1506.

Rosenthal, S.; Veloso, M.; and Dey, A. K. 2012. Is someone in this office available to help? proactively seeking help from spatially-situated humans. *Journal of Intelligence and Robotic Systems* 66(1-2):205–221.

Rosenthal, S. 2012. *Human-Centered Planning for Effective Task Autonomy*. Ph.D. Dissertation, Carnegie Mellon University.

Roth, M.; Simmons, R.; and Veloso, M. M. 2006. What to communicate? execution-time decision in multi-agent pomdps. In *The 8th International Symposium on Distributed Autonomous Robotic Systems (DARS)*.

Schoemaker, P. J. H. 1982. The expected utility model: Its variants, purposes, evidence and limitations. *Journal of Economic Literature* 20:529–563.

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.

Weiss, A.; Igelsböck, J.; Tscheligi, M.; Bauer, A.; Kühnlenz, K.; Wollherr, D.; and Buss, M. 2010. Robots asking for directions: the willingness of passers-by to support robots. In *HRI '10*, 23–30.

Yanco, H.; Drury, J. L.; and Scholtz, J. 2004. Beyond usability evaluation: analysis of human-robot interaction at a major robotics competition. *Human-Computer Interaction* 19(1):117–149.