# Dynamic User Task Scheduling for Mobile Robots

**Brian Coltin and Manuela Veloso**
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
{bcoltin, veloso}@cs.cmu.edu

**Rodrigo Ventura**
Visiting Scholar
Institute for Systems and Robotics
Instituto Superior Técnico
Lisboa, Portugal
yoda@isr.ist.utl.pt

**Abstract**

We present our efforts to deploy mobile robots in office environments, focusing in particular on the challenge of planning a schedule for a robot to accomplish user-requested actions. We concretely aim to make our CoBot mobile robots available to execute navigational-based tasks requested by users, such as telepresence, and picking up and delivering messages or objects at different locations. We contribute an efficient web-based approach in which users can request and schedule the execution of specific tasks. The scheduling problem is converted to a mixed integer programming problem. The robot executes the scheduled tasks using a synthetic speech and touch-screen interface to interact with users, while allowing users to follow the task execution online. Our robot uses a robust Kinect-based safe navigation algorithm, moves fully autonomously without the need to be chaperoned by anyone, and is robust to the presence of moving humans, as well as non-trivial obstacles, such as legged chairs and tables. Our robots have already performed 15km of autonomous service tasks.

## Introduction and Related Work

We envision a system in which autonomous mobile robots robustly perform service tasks in indoor environments. The robots perform tasks which are requested by building residents over the web, such as delivering mail, fetching coffee, or guiding visitors. To fulfill the users' requests, we must plan a schedule of when the robot will execute each task in accordance with the constraints specified by the users.

Many efforts have used the web to access robots, including the early examples of the teleoperation of a robotic arm (Goldberg et al. 1995; Taylor and Trevelyan 1995) and interfacing with a mobile robot (e.g, (Simmons et al. 1997; Siegwart and Saucy 1999; Saucy and Mondada 2000; Schulz et al. 2000)), among others. The robot Xavier (Simmons et al. 1997; 2000) allowed users to make requests over the web for the robot to go to specific places, and other mobile robots soon followed (Siegwart and Saucy 1999; Grange, Fong, and Baur 2000; Saucy and Mondada 2000; Schulz et al. 2000). The RoboCup@Home initiative (Visser and Burkhard 2007) provides competition setups for indoor service autonomous robots, with an increasingly wide scope of challenges focusing on robot autonomy and verbal interaction with users.

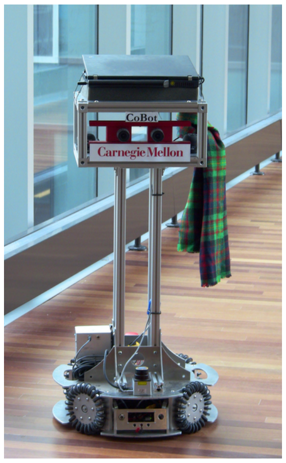


Figure 1: CoBot-2, an omnidirectional mobile robot for indoor users.

In this work, we present our architecture to effectively make a fully autonomous indoor service robot available to general users. We focus on the problem of planning a schedule for the robot, and present a mixed integer linear programming solution for planning a schedule. We ground our work on the CoBot-2 platform [1], shown in Figure 1. CoBot-2 autonomously localizes and navigates in a multi-floor office environment while effectively avoiding obstacles (Biswas and Veloso 2010). The robot carries a variety of sensing and computing devices, including a camera, a Kinect depth-camera, a Hokuyo LIDAR, a touch-screen tablet, microphones, speakers, and wireless communication.

CoBot-2 executes tasks sent by users over the web, and we have devised a user-friendly web interface that allows users to specify tasks. Currently, the robot executes three types of tasks: a *GoToRoom* task where the robot visits a location, a *Telepresence* task where the robot goes to a location

[1] CoBot-2 was designed and built by Michael Licitra, mlicitra@cmu.edu, as a scaled-up version of the CMDragons small-size soccer robots, also designed and built by him.

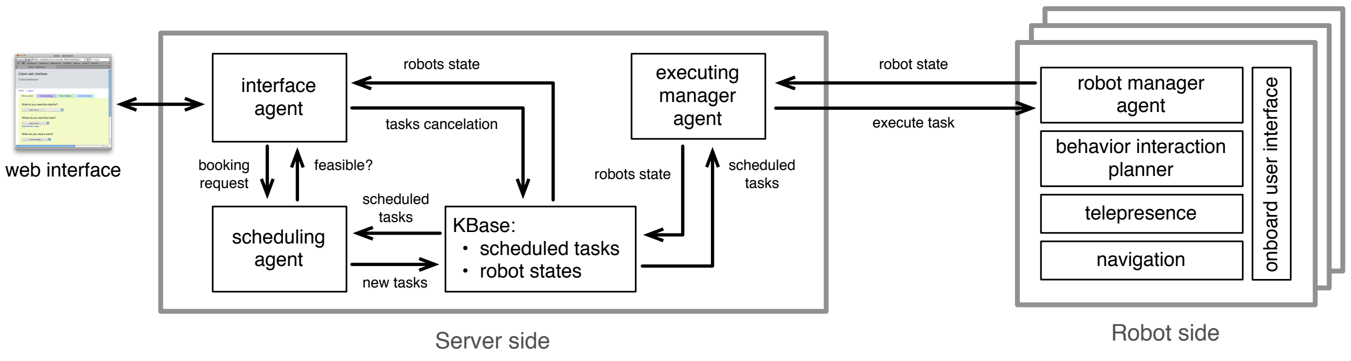


Figure 2: Users to Mobile Robots (UMR) architecture.

and allows someone to control it remotely, and a *Transport* task where the robot picks up an object at one location and brings it to another. When placing a reservation for a task, the user selects when the task should be completed. Tasks are scheduled taking into account feasibility constraints involving other previously scheduled tasks as well as the estimates of both the navigation time and of the interaction time. If the task cannot be completed, the user may be asked to loosen the constraints or is told that the task cannot be completed.

Once tasks are scheduled, a behavior plan is executed on the robot to interact with the users at the task locations and to incidentally interact with other humans in the environment (Fong, Nourbakhsh, and Dautenhahn 2003). The plan may need to be adjusted when tasks do not complete in the expected time.

In the remainder of this paper, we will present our complete users to mobile robots architecture. We will then discuss task scheduling and the behavior planner. We present the web-based interface for users to interact with the robot, and show illustrative examples of the effectiveness of the system.

## Users to Mobile Robots Architecture

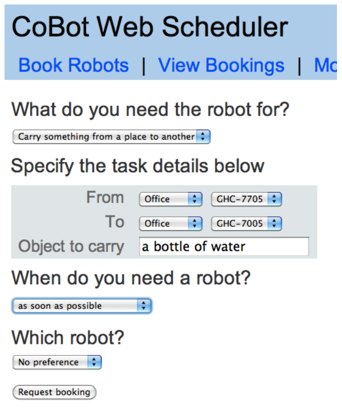
Deploying a mobile robot to general users poses several challenges: (1) the task-request interface must be easily accessible and user-friendly, (2) the scheduling algorithm must plan feasible schedules, (3) navigation must be safe and reliable in office environments, and (4) the robot must interact intuitively with humans.

To address these challenges, we contribute a *Users to Mobile Robots* (UMR) architecture (see Figure 2), with different modules to connect the user requests to the executor robot, namely:

- **interface agent** — manages the interaction with users using the web interface, serving two main purposes: managing bookings, including placing new ones and canceling previously booked tasks, and showing the robot's location in real-time;
- **scheduling agent** — receives booking requests from the *interface agent*, checks whether they are feasible with respect to previously scheduled tasks, stores them if feasible, or proposes an alternative otherwise;
- **KBase** — stores all scheduled tasks, including the ones already executed, the one being executed, and the ones scheduled in the future, as well as the robot's state (such as location and battery level);
- **executing manager agent** — handles the communication with the robots, sending them tasks to be executed, and receiving their current state;
- **robot manager agent** — handles the communication of the robot with the server;
- **behavior interaction planner** — executes tasks by breaking them down into sub-tasks (e.g., the Transport tasks comprises a sequence of interleaved navigation and interaction sub-tasks);
- **navigation** — handles the navigation of the robot to a location given by the *behavior interaction planner*, using the robot sensors for localization, navigation, and obstacle avoidance (Biswas and Veloso 2011);
- **onboard user interface** — manages human-robot interaction, using synthetic speech and touch-screen display.

This architecture interacts with users in two distinct ways: through the web interface, for managing bookings and following the robot's state, and directly with the robot through its onboard user interface.

The web-based booking interface addresses challenge (1), to the extent that web-based booking systems are a widespread and familiar scheme for reserving services, being found in numerous services such as in hotel reservations, car rental, and more recently in ZipCar™. Challenge (2) is addressed by the scheduling agent presented in section . This agent plans a feasible schedule of when the robot can complete all of the tasks, or informs the interface agent that it cannot. A robust navigation method based on the Kinect depth camera and capable of effectively navigating in an office environment, while avoiding moving and fixed obstacles addresses challenge (3) (Biswas and Veloso 2011). The robot's face-to-face interaction with its users is based on a joint synthetic-voice and touch-screen *onboard user interface*. Messages are both spoken by a synthetic voice and displayed on the touch-screen, while the user can respond to these messages using buttons displayed on the touch-screen. This interface is simple and easy to use, thus addressing challenge (4).



What do you need the robot for?

Carry something from a place to another

Specify the task details below

| | | |
|---|---|---|
| From | Office | GHC-7705 |
| To | Office | GHC-7005 |
| Object to carry | a bottle of water | |

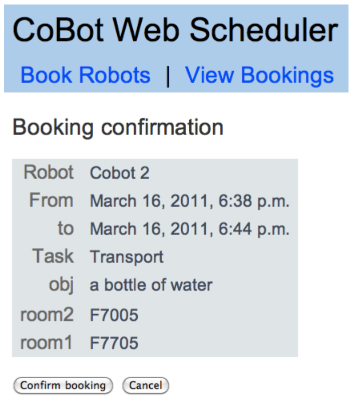
When do you need a robot?

as soon as possible

Which robot?

No preference

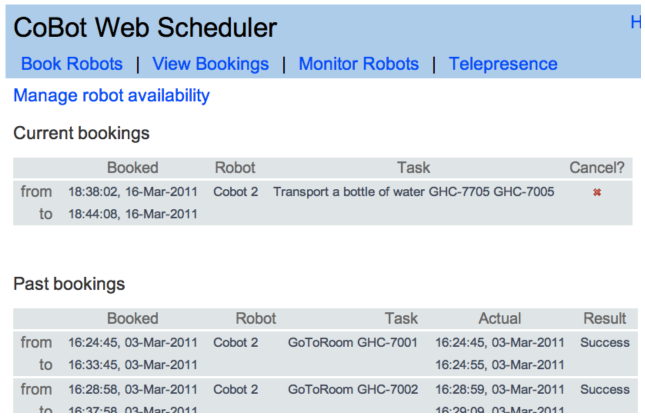
(a) book a robot



Booking confirmation

| | |
|---|---|
| Robot | Cobot 2 |
| From | March 16, 2011, 6:38 p.m. |
| to | March 16, 2011, 6:44 p.m. |
| Task | Transport |
| obj | a bottle of water |
| room2 | F7005 |
| room1 | F7705 |

(b) confirmation


CoBot Web Scheduler
H
Book Robots | View Bookings | Monitor Robots | Telepresence
Manage robot availability

Current bookings

| | Booked | Robot | Task | Cancel? |
|---|---|---|---|---|
| from | 18:38:02, 16-Mar-2011 | Cobot 2 | Transport a bottle of water GHC-7705 GHC-7005 | ✖ |
| to | 18:44:08, 16-Mar-2011 | | | |

Past bookings

| | Booked | Robot | Task | Actual | Result |
|---|---|---|---|---|---|
| from | 16:24:45, 03-Mar-2011 | Cobot 2 | GoToRoom GHC-7001 | 16:24:45, 03-Mar-2011 | Success |
| to | 16:33:45, 03-Mar-2011 | | | 16:24:55, 03-Mar-2011 | |
| from | 16:28:58, 03-Mar-2011 | Cobot 2 | GoToRoom GHC-7002 | 16:28:59, 03-Mar-2011 | Success |
| to | 16:37:58, 03-Mar-2011 | | | 16:29:09, 03-Mar-2011 | |

(c) view bookings

Figure 3: Screenshots of the web interface, showing (a) the web interface to perform a booking, (b) the confirmation screen containing the start and (estimated) end times, and (c) the list of current and past bookings performed by the user.

To illustrate the functionality of the UMR architecture, we present next a running example of the booking and execution of a task requested by a user:

1. At 6:35PM, Alice uses a web browser to request a robot to transport a bottle of water, from room 7705 to room 7005, as soon as possible (Figure 3a);
2. The web interface passes this new request to the scheduling agent, which plans the robot's arrival at the pick up location (7705) at 6:38PM (Figure 3b);
3. After confirmation, CoBot-2 starts executing this task immediately: it navigates to room 7705, while displaying and speaking the message "Going to 7705 to pick up a bottle of water and bring it to 7005" on the *onboard user interface*;
4. Upon arrival to 7705, CoBot-2 displays and speaks the message "Please place a bottle of water on me to deliver", and awaits someone to click the 'Done' button displayed on the touch-screen;
5. Once this button is pressed, the robot starts navigating to room 7005;
6. upon arrival to 7005, CoBot-2 displays and speaks the message "Please press 'Done' to release me from my task", and awaits the user to press the 'Done' button;
7. Once this button is pressed, the task is marked as successfully executed, and the robot navigates back to its home location.

After the task has been booked, Alice can check the booking on the web (Figure 3c) and cancel it if necessary. During the task execution, Alice could follow the progress of the robot navigation, either on the map view (Figure 5a) or through the camera view (Figure 5b).

## Scheduling Agent

Once users have placed requests for the robot over the web, a schedule of tasks must be formed with times for the robot to complete them. Tasks that the robots could perform include coming to a specific location for a demonstration, guiding a visitor from one room in the building to another, fetching coffee or printouts, delivering mail, or delivering a spoken message. When the user makes a request, a time is specified for when the task is to be carried out. This may either be "as soon as possible", an exact time, or a window of time with a beginning and an ending time. Internally, all these representations are converted to windows of time. The server may choose to either accept, reject, or ask to refine a request.

We first look at the batch problem, where the scheduling agent is given a list of $n$ task requests, $T$, that the robots must fulfill. The goal is to find a starting time, $t_i$, for each task such that the tasks do not overlap and the requests are fulfilled as soon as possible. Each time $t_i$ must fall within a window of times $[s_i, e_i]$, and the task has a duration $d_i$. Furthermore, each task has a starting and ending location $l_i^s, l_i^e \in L$ (they will be different for the Transport task), and there is a function $dist : L \times L \to \mathcal{R}$ giving an estimate of the time to travel between two locations.

We solve this scheduling problem using mixed integer programming. We solve for the variables $t_i$, and also introduce some helper indicator variables $pre_{i,j}$ which indicate whether task $i$ precedes task $j$. Our first set of constraints is that each time $t_i$ must fall within the start and end times of the window.

$$\forall i \; s_i \leq t_i \leq e_i$$

Then, we constrain the times so that they do not overlap. First, we define the $pre_{i,j}$ to be indicator variables.

$$\forall i,j \; 0 \leq pre_{i,j} \leq 1 \text{ int}$$

Then, we add constraints which ensure that the execution times of two tasks do not overlap, handling both the case that task $i$ comes before task $j$ or the reverse order depending on the value of $pre_{i,j}$.

$$\forall i,j \; t_i + d_i + dist(l_i^e, l_j^s) - t_j \leq |e_j - s_i|(1 - pre_{i,j})$$
$$\forall i,j \; t_j + d_j + dist(l_j^e, l_i^s) - t_i \leq |e_i - s_j| pre_{i,j}$$

The objective that we choose to minimize is the total difference in time from the start of the scheduling window for all the tasks. This objective ensures that user tasks are completed as soon as possible.

$$\min \sum_i (t_i - s_i) = \min \sum_i t_i$$

Alternatively, we could minimize the total distance travelled by the robot to prolong battery life, or we could optimize a constant to more quickly find any feasible schedule. So we have $\binom{n}{2} + n$ variables, and a proportional number of constraints.

However, we can reduce the problem further. Some pairs $i$ and $j$ may never even have a possibility of overlapping depending on their time windows. If either $e_i + d_i + dist(l_i, l_j) \leq s_j$ or $e_j + d_j + dist(l_j, l_i) \leq s_i$, then there is no possibility of the two tasks conflicting and we can eliminate the constraint entirely. In practice, we expect that many constraints will not overlap at all.

Although solving the MIP is an NP-hard problem, we have found that in practice it can be solved quickly for certain problems. We generated a thousand random problem instances, each of 15 tasks with two minutes to half an hour durations, with time windows over the course of four hours. This is the type of input we expect the robot to receive. The scheduler solved (either found a schedule or found that no schedule existed) 99% of the problems in under two seconds. In the cases where a schedule is not found quickly, rather than waiting, we can declare that there is no valid schedule.

In practice, the task requests are not processed in a batch, but come in an online fashion over the web. We reschedule everything whenever a new request is made. If a schedule cannot be found, the user's request is rejected and the user has an opportunity to relax the constraints.

Since the tasks are running on actual robots, the actual time to complete a task will often differ from the expected time. In these cases, a new schedule is built, and some of the tasks may need to be abandoned.

After a task is scheduled, the executing manager agent sends the robot-specific scheduled task set to the corresponding robot manager agent to execute. The robot's Behavior Interaction Planner plans the sequence of action to complete each task.

## Behavior Interaction Planner

Typically, task planners only plan the autonomous actions to complete a task and a separate dialog manager interacts with humans to receive the task requests. However, a robot cannot always perform its actions autonomously and relies on humans in the environment to help it complete tasks. Additionally, as a robot performs actions, humans in the environment may want to know what the robot's goals are. Our Behavior Interaction Planner therefore reasons about a robot's incapabilities (Rosenthal, Biswas, and Veloso 2010) and human interest in the robot and plans for human interactions in addition to the autonomous actions. As the robot executes the plan, it reports a descriptive message back to the server for online users to follow the robot's progress in the web interface.

### Actions and Interactions

We define actions and interactions that are required to complete a task along with their preconditions and effects. For `ask` interactions, for example, there are no preconditions, the robot `speaks` the defined text, and the effect is the required human response (*e.g.* clicking a 'Done' button on CoBot's user interface). For `navigate` actions, the precondition is that the robot `speak` aloud its new goal to humans in the area, the robot then sends a desired location to the *navigation* module, and the effect is that the robot is in the location that it should navigate to. The separate *navigation* module controls the low level motor control and obstacle avoidance for navigation. Any other actions needed for a task can be defined similarly.

### Autonomous Planning and Execution

Given a new task, the robot plans the sequence of actions necessary to complete it. For example, in the $\text{Transport}(s, l_p, l_d, m)$ task, the Behavior Interaction Planner plans the following sequence of actions (illustrated in Figure 4): `navigate` to location $l_p$, `ask` for the object $m$, `navigate` to $l_d$, and `ask` for task completion confirmation.

The Behavior Interaction Planner can also plan for a robot's incapabilities. For example, if CoBot (with no arms) must navigate between different floors of the building, this requires not only `navigate` actions, but also human interaction to ask for help with pressing buttons and recognizing which floor the robot is on. In these cases, the Behavior Interaction Planner plans:

- `navigate` to elevator,
- `ask` for help pressing the up/down button,
- `navigate` into the elevators,
- `ask` for help pressing the floor number and recognizing that floor,
- `navigate` out of the elevator,
- `navigate` to goal

Upon arriving at goal locations, the robot may also need help picking up objects and plans for these additional `ask` interactions accordingly.

## Web-based User Interaction

The interface agent and the scheduling agent are implemented as a web application, running the web-based user interface, and connecting to the central scheduling database and the robot. The web interface is used for scheduling new tasks, examining and modifying existing tasks, and monitoring or controlling the robot through telepresence.

### Web Booking Interface

After successful authentication with a username and password, each user is presented with a web interface, composed of four sections.

- **Book a Robot.** A user specifies the task to book together with the desired time (Figure 3a). The booking interface

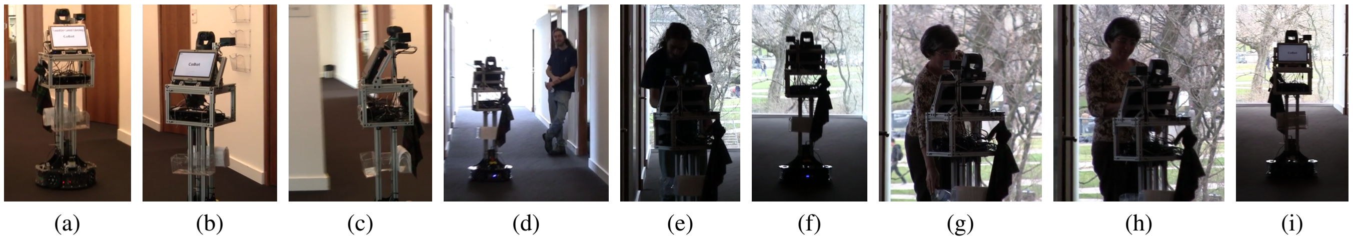


Figure 4: (a,b,c) After CoBot-2 receives a *Transport* task request, it autonomously navigates to the location $l_p$ to pick up a bottle of water. (d,e) Upon arriving to $l_p$, CoBot-2 asks the user to place the bottle of water and press ‘Done’. (f,g) Then, CoBot-2 navigates to location $l_d$ to deliver the bottle of water. (h,i) When the user presses ‘Done’ CoBot-2 navigates back to its home location. (The complete example sequence is submitted as a video with this paper.)

supports two ways of setting the start time of a task: users can request the task to be executed as soon as possible, for which the soonest feasible time slot is provided, or at a specific time, in 10 minutes intervals. The actual booking is preceded by a confirmation screen (Figure 3b).

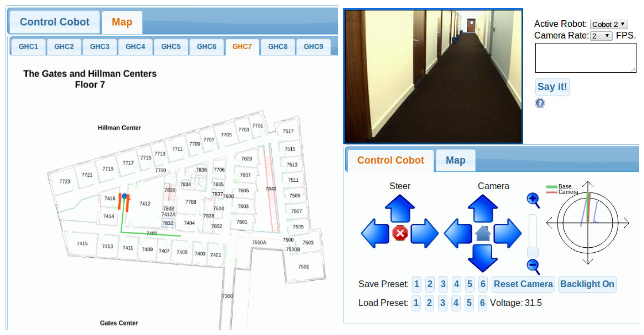
- **View Bookings.** This section displays both the current and the past bookings (Figure 3c). Current bookings can be canceled at any time. If the robot is already executing a task, it is immediately aborted. The past bookings list shows both the scheduled and the actual start and end times for each task, as well as whether its execution was successful or an unexpected event canceled it (e.g., low battery).
- **Monitor Robots.** Users can follow the location of the robot in real-time on a map. This interface also displays the task currently being executed by the robot, as well as its battery level.
- **Telepresence.** Users can view the state of the robot and operate it remotely if they have scheduled a telepresence task.

The web booking interface enables users to intuitively interact with the robot.

## Telepresence

In addition to performing tasks fully autonomously, one of the tasks users may request a robot for is to control CoBot-2 in a semi-autonomous “telepresence” mode in the *Telepresence* task. In telepresence mode, live sensory information and camera images are streamed and displayed directly in the user’s web browser on a page linked to from the web scheduler.

The telepresence interface, shown in Figure 5, displays the camera image, the text-to-speech interface, and the controls for both the robot navigation and camera pan-tilt-zoom settings. The telepresence interface provides three control modalities with increasing levels of autonomy. In all modalities, the robot autonomously avoids obstacles. In addition to controlling the robot with the interface buttons, users may click directly on the image to point the camera or to navigate the robot to the point clicked. The interface map displays the robot’s current location and orientation, and highlights detected obstacles to help the user to navigate safely. The user may click on the map to send the robot autonomously to a



(a) map view (b) telepresence view

Figure 5: Screenshots of the telepresence interface, showing (a) the map view of CoBot-2 location with its navigation path, and (b) the telepresence interface, showing the robot’s camera view, together with camera and robot motion controls.

location. We have found that users utilize all of these control modalities depending on the situation.

## Illustrative Results

In order to illustrate the effectiveness of our system we conducted a set of experiments comprising 41 tasks requested on the web interface (21 Transport, 20 Go-To-Room). The locations used were all existing 88 offices at the 7th floor of the Gate-Hillman Center in Carnegie Mellon University. The execution of these tasks resulted in 138 minutes of autonomous navigation, along a total distance of 4.9 km. The location(s) used in each task was randomly chosen from all of the 88 offices. Of the 41 tasks, 37 were successfully completed ($> 90\%$), three were manually aborted due to blocked paths, and one was automatically aborted due to a low battery condition.

Figure 6 shows all the trajectories traveled by CoBot-2 during these experiments, spanning almost all navigable space on the floor. The time taken by each navigation subtask as a function of traveled distance is plotted in Figure 7. The relation between distance traveled and time is roughly linear, except for a few outlier points. Some of these outliers correspond to cases when the task was aborted due to a blocked path, while others can be explained by the robot trying to navigate around people blocking its way, possibly

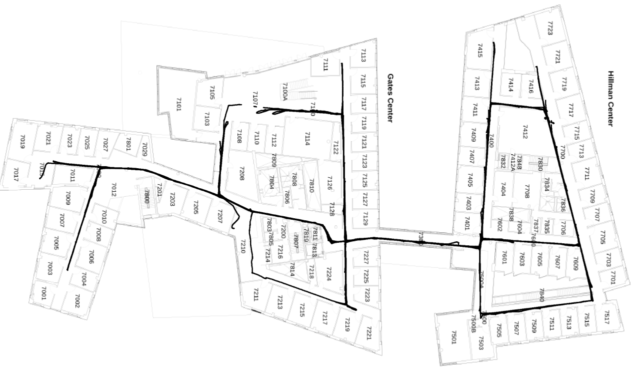


Figure 6: Union of all trajectories traveled by CoBot-2 on the 7th floor of the Gates-Hillman Center (CMU).

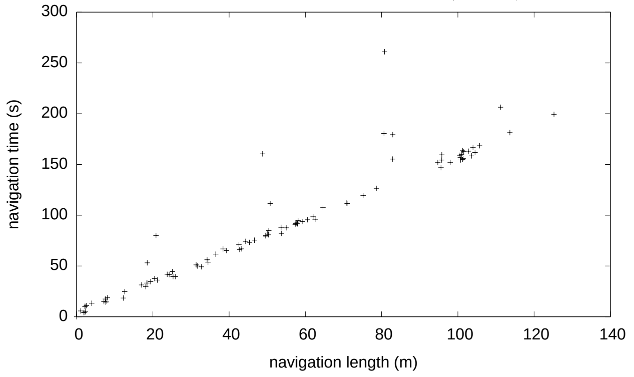


Figure 7: Time taken by each navigation sub-task.

driven by curiosity. This shows the robot safely navigates around walls, chairs, and people throughout the entire office environment.

## Conclusion

This paper presented the Users to Mobile Robots (UMR) architecture, addressing the goal to deploy mobile robots to general users. This architecture allows users to request tasks and monitor their execution, using an easy to use web interface. Our scheduling agent finds a feasible schedule, taking into account time constraints imposed by the mobile nature of robots. The Behavioral Interaction Planner autonomously plans and executes the scheduled tasks. We demonstrated that the navigation method is robust; CoBot-2 has traveled an accumulated distance of 15 kilometers without being chaperoned. The onboard user interface is intuitive, as any person was able to appropriately respond to it.

## References

Behnke, S.; Stückler, J.; and Schreiber, M. 2009. Nimbro @home 2009 team description paper. In *RoboCup 2009*.

Biswas, J., and Veloso, M. 2010. WiFi localization and navigation for autonomous indoor mobile robots. In *IEEE International Conference onRobotics and Automation (ICRA)*, 4379–4384.

Biswas, J., and Veloso, M. 2011. Depth camera based indoor mobile robot autonomy. In *submission*.

Breuer, T.; Giorgana, G.; Hegger, F.; Müller, C.; Jin, Z.; Reckhaus, M.; Paulus, J.; Hochgeschwender, N.; Awaad, I.; Hartanto, R.; Ploeger, P.; and Kraetzschmar, G. 2010. The b-it-bots robocup@home 2010 team description paper. In *RoboCup 2010*.

Fong, T.; Nourbakhsh, I.; and Dautenhahn, K. 2003. A survey of socially interactive robots. *Robotics and Autonomous Systems* 42:143–166.

Goldberg, K.; Mascha, M.; Gentner, S.; Rothenberg, N.; Sutter, C.; and Wiegley, J. 1995. Desktop teleoperation via the world wide web. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, 654–659.

Grange, S.; Fong, T.; and Baur, C. 2000. Effective vehicle teleoperation on the world wide web. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, 2007–2012.

Rosenthal, S.; Biswas, J.; and Veloso, M. 2010. An effective personal mobile robot agent through symbiotic human-robot interaction. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, volume 1, 915–922.

Saucy, P., and Mondada, F. 2000. KhepOnTheWeb: open access to a mobile robot on the internet. *IEEE Robotics Automation Magazine* 7(1):41–47.

Schulz, D.; Burgard, W.; Fox, D.; Thrun, S.; and Cremers, A. 2000. Web interfaces for mobile robots in public places. *IEEE Robotics Automation Magazine* 7(1):48–56.

Siegwart, R., and Saucy, P. 1999. Interacting mobile robots on the web. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Simmons, R.; Goodwin, R.; Haigh, K. Z.; Koenig, S.; and O'Sullivan, J. 1997. A layered architecture for office delivery robots. In *Proceedings of the first international conference on Autonomous agents (AGENTS'97)*, 245–252.

Simmons, R.; Fernandez, J.; Goodwin, R.; Koenig, S.; and O'Sullivan, J. 2000. Lessons learned from xavier. *IEEE Robotics Automation Magazine* 7(2):33–39.

Taylor, K., and Trevelyan, J. 1995. Australia's telerobot on the web. In *Proceedings of the International Symposium on Industrial Robots*, volume 26, 39–44.

Visser, U., and Burkhard, H.-D. 2007. RoboCup: 10 years of achievements and future challenges. *AI Magazine* 28(2):115–132.