Auton Robot
DOI 10.1007/s10514-007-9023-1

Using a hand-drawn sketch to control a team of robots

Marjorie Skubic · Derek Anderson · Samuel Blisard · Dennis Perzanowski · Alan Schultz

Received: 8 March 2006 / Revised: 22 November 2006 / Accepted: 2 January 2007
© Springer Science + Business Media, LLC 2007

Abstract In this paper, we describe a prototype interface that facilitates the control of a mobile robot team by a single operator, using a sketch interface on a Tablet PC. The user draws a sketch map of the scene and includes the robots in approximate starting positions. Both path and target position commands are supported as well as editing capabilities. Sensor feedback from the robots is included in the display such that the sketch interface acts as a two-way communication device between the user and the robots. The paper also includes results of a usability study, in which users were asked to perform a series of tasks.

Keywords Human-robot interaction · Sketch-based navigation · Qualitative map

1 Introduction

Currently, most of the mobile robots used in operational settings rely on teleoperated control using live video. This requires intensive interaction with a human operator (Burke and Murphy, 2004). Often, more than one person is required to deploy the robot. At best, one operator is required per robot, making control of a multi-robot team complicated and difficult to synchronize.

There is interest in moving towards an interface that allows one operator to manage a team of robots. Certainly, this would be advantageous for military applications such as surveillance and reconnaissance. It would also be helpful for many humanitarian efforts such as in the relief efforts for the hurricane disaster in New Orleans and the U.S. Gulf Coast. Robots could be helpful in search and rescue, as well as in assessing damage or the extent of hazardous conditions. Deploying a team of robots means a larger area can be covered more quickly, provided there is some method of coordinating their control.

In this paper, we describe a prototype interface in which a single operator can control a team of robots using a sketch-based interface on a Tablet PC. A precise map of the environment is not required. Rather, the user hand draws a map of a live scene and includes each robot in an approximate starting location. We assert that, in the cases mentioned, requiring a precise map of the environment may slow the efforts, as the landscape may have changed in hostile or natural disaster environments. Therefore, the ability to use an approximate, hand-drawn map is viewed as a matter of convenience and efficiency.

The proposed interface allows the user to sketch a route map for controlling a team of robots, as might be done in directing a team of people. In addition, the interactive sketch interface acts as a two-way communication device between the user and each of the robots. We assume that each robot has low level behaviors to handle obstacle avoidance. The sketch interface provides a mechanism for directing each robot according to task needs, where each directed move is viewed as a guarded move.

Several previous sketch-based systems have been proposed, which rely on a quantitative map. Perzanowski et al. (2001) have developed a multi-modal robot interface that includes a PDA in which a quantitative map is displayed based on the robot's sensors as it travels through an environment. The user can draw gestures on top of the map

M. Skubic (✉) · D. Anderson · S. Blisard
Electrical and Computer Engineering Department, University of Missouri-Columbia, Columbia, MO
e-mail: skubicm@missouri.edu

D. Perzanowski · A. Schultz
Navy Center for Applied Research in Artificial Intelligence, US Naval Research Laboratory, Washington, DC

to indicate target positions of the robot that are converted to go-to commands in a global reference frame. Lundberg et al. (2003) have developed a similar PDA interface, which supports the display of a map that can be used to designate a target location or a region to explore. Fong's PDA interface (Fong et al., 2003) includes the ability to sketch waypoints on top of a robot-sensed image, which allows live imagery to be used in the control. Another version of his PDA interface also supports multi-robot control and sketching waypoints on top of a map (an occupancy grid built as the robot explores) as well as an image (Fong et al., 2003).

Other work has included the use of a qualitative map. Chronis and Skubic (2004) have developed a PDA interface in which the user sketches a route map as a means of directing a single robot along a designated path. Navigation is done by sequencing turn commands based on landmark states. Kawamura et al. (2002) also use a landmark-based approach, where artificial landmarks are placed in the scene and on top of a sketched map drawn on a PDA screen. In their experiments, artificial landmarks were multi-colored cones that were recognized using vision. Freksa et al. (2000) have proposed the use of a schematic map which they describe as an abstraction between a sketch map and a topological map, e.g., a subway map. Finally, Setalaphruk et al. (2003) use a scanned, hand-drawn map of an indoor scene (with walls and corridors) and extract a topological map for controlling a robot.

Additional research has addressed the general area of sketch understanding at different levels of abstraction in non-robotic environments. For example, one approach seeks a high level sketch understanding for automatic generation of user interfaces from sketch examples specific to multiple domains (Alvarado et al., 2002; Hammond and Davis, 2004). Formal descriptions of the user interface are generated from example sketches, and top-down contextual recognition is coupled with bottom-up shape recognition. One component in this work is the sketch language LADDER, proposed by Hammond and Davis (2005), which is capable of describing how shapes and shape groups are drawn, edited, and displayed. At a lower level of abstraction, other sketch understanding work has incorporated HMM-based sketch recognition (Sezgin and Davis, 2005) and scale-space feature point detection (Sezgin and Davis, 2004).

With the exception of Fong's work, none of the related work has attempted to control multiple robots with one sketch-based interface. Here, we describe an interface that supports the control of multiple robots using a qualitative, hand-drawn map.

In the remaining paper, we describe components of the system: the algorithms used to process the sketch, the translation of sketch information into robot commands, and synchronization that provides feedback from the robot to the sketch platform. Integration is discussed for the robot system. Finally, we discuss a usability study in which the interface was investigated using 23 participants who were asked to perform a series of tasks.

2 Sketch understanding

Our sketch interface incorporates intuitive management of multiple robots simultaneously in combination with the display of sensor feedback and the synchronization of robot locations. Users sketch a qualitative map of the environment that describes the scene, and then sketch navigation gestures to direct the robots. Feedback from the robots' sensors is displayed on the sketch to help the user keep a current representation of a possibly dynamic environment, or to adjust an initial sketch that was not accurate enough. Various interactions produce color changes in the user interface, such as the center of robots changing color. These additional aids provided for users to understand their interactions with the interface are not reproduced here. However, we will verbally indicate color changes in this paper so that the reader has a sense of what users of the interface experience.

Users add environment landmarks by sketching a closed polygon anywhere on the screen (shown in Fig. 1). The user provides an identifier for each landmark, which is used to correlate objects in the sketch with objects in the real robot environment. Objects in the robots' environment correspond to what is observed and segmented from an occupancy grid map. In the prototype interface, this correlation between sketch and robot objects is manually handled by the user providing the identifiers. In the experiments, the map is built a priori, and an identifier is assigned to each landmark.

To create a robot, the user sketches a small concentrated stroke anywhere on the screen (i.e., a squiggle) and labels the robot with a name. A robot icon is displayed in place

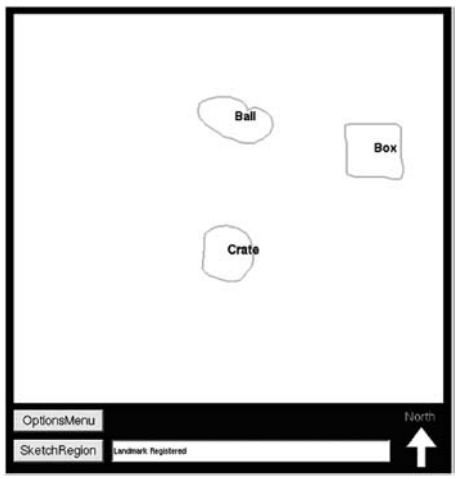

Fig. 1 Sketching landmarks

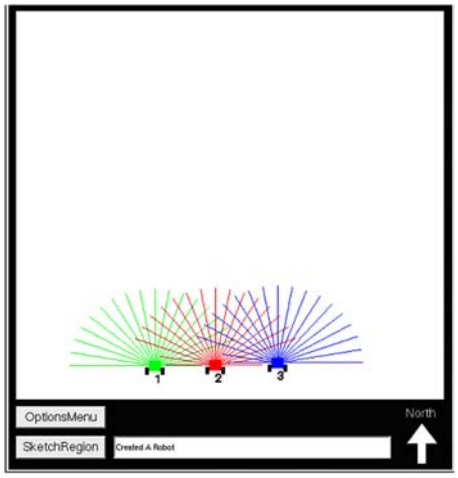

Fig. 2 Sketching robots

of the stroke and, if communications can be established with the real robot, then sensor feedback is shown from the range sensors. Figure 2 shows three connected robots with laser rangefinders that span the front 180 degrees of the robots.

Individual robots and landmarks can be selected by clicking on the robot or landmark. The user can then edit the sketch by dragging the selected entity to a new location. Such editing features allow the user to fine-tune the sketch without redrawing but do not result in robot commands. A group of robots can be selected by drawing a lasso around a subset of robots. Figure 3 shows two robots being selected; in the sketch-based interface, their color changes to purple to indicate selection and provides additional feedback to the user.

Identifying the robots in a lasso is done using the Bresenham line algorithm (Bresenham, 1965) on simple closed polygons, dilating each point on the lasso, and then picking a point inside the lasso and doing a flood fill. To determine which robots are in the lasso, the pixel at the robot's center

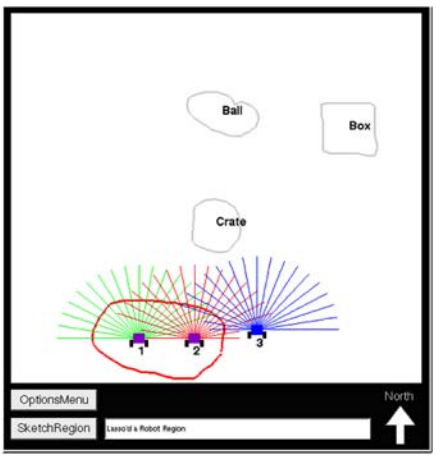

Fig. 3 Lassoing a group of robots

location is checked to see if it was a flood-filled or boundary point.

Feedback from the robot sensors can be used to detect the present environment configuration, which allows a user to adjust the current placement of landmarks and robots by dragging them. If the shape and size of a landmark does not match what is being detected from feedback, then a user can delete and redraw landmarks. Right clicking or holding the pen on a robot or landmark will delete it. If a robot encounters additional landmarks, if a landmark was moved, or if a landmark was removed, users can sense this from sensor feedback and edit the sketch to show a more accurate scene.

Navigation commands may be issued to robots after one or more landmarks are sketched. Because we use qualitative and not quantitative information, navigation commands are issued relative to landmarks. Sketching an "X", two short intersecting lines, issues a go-to command for all selected robots. If a user wants the robots to follow a route, he or she sketches a path that originates from a single robot or a location inside a lasso. Paths are segmented into a series of go-to commands and issued to all robots in a group.

Figure 4 shows a scenario in which path commands are issued, and Fig. 5 illustrates go-to commands. The landmark that is closest to the last sketched goal point changes color to indicate its use as a reference object (the ball in Fig. 4 and the box in Fig. 5). The segmented path is shown as a sequence of gray triangles. The center of each robot changes color to yellow to indicate its motion. All target locations (X's in Figs. 4 and 5) are displayed in the same color as the corresponding robot for clarity.

In Fig. 4, the sensor readings of robot 3 indicate the presence of an object. Note that the sensor readings match the position of the box. Inconsistencies in sensor readings and sketched landmarks can be used to adjust positions to match the sensor feedback or to inform the user of an unknown landmark that should be included in the sketch.

As a default mode, robots are automatically dispatched once a navigation command is registered. If a user wants to postpone navigation commands (e.g., for synchronization of robots), a menu option allows simultaneous execution of commands after an arrow is sketched. The symbol recognition method used to classify the arrow is based on Hidden Markov Models (Anderson et al., 2004).

3 Translating a sketch into robot commands

Go-to commands are computed for each robot by looking at the relative position of the robot to the landmark closest to the goal point and the relative position of the goal point to the same landmark. These two quantities are extracted

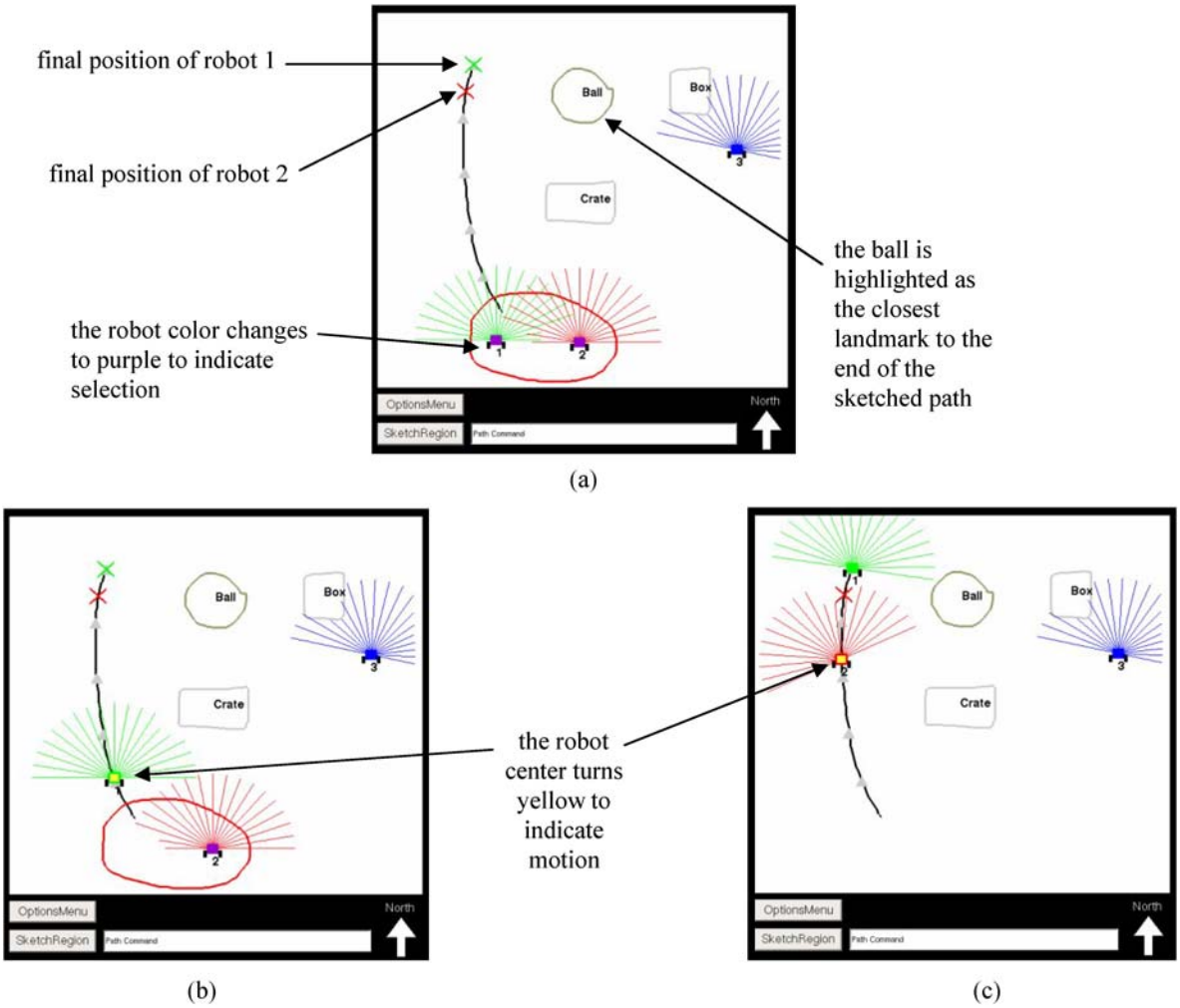

Fig. 4 Robots 1 and 2 are directed to follow a path. The path has been segmented into a sequence of intermediate points, shown as gray triangles along the path. The center of the robot which turns to yellow indicates motion as the robot moves along the path. Each robot displays its laser readings in its corresponding color. (a) A snapshot showing the selection of robots 1 and 2. Robot 1's final position is computed at the end of the path, as robot 1 is closer to the start of the path and will move first. (b) The synchronized motion of a group of robots. Robot 1 has started to move; robot 2 waits its turn. (c) Both robots move along the sketched path. Robot 1 has reached its final target destination

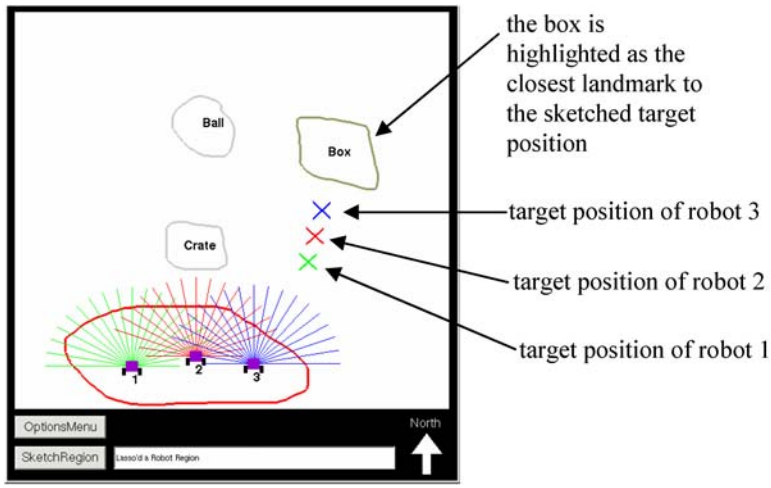

Fig. 5 Robots 1, 2 and 3 have been selected as a group. For robot groups, target points are computed according to the distance of each robot to the sketched goal point. Robot 3's position is closest to the box; robot 1's position is the farthest from the box

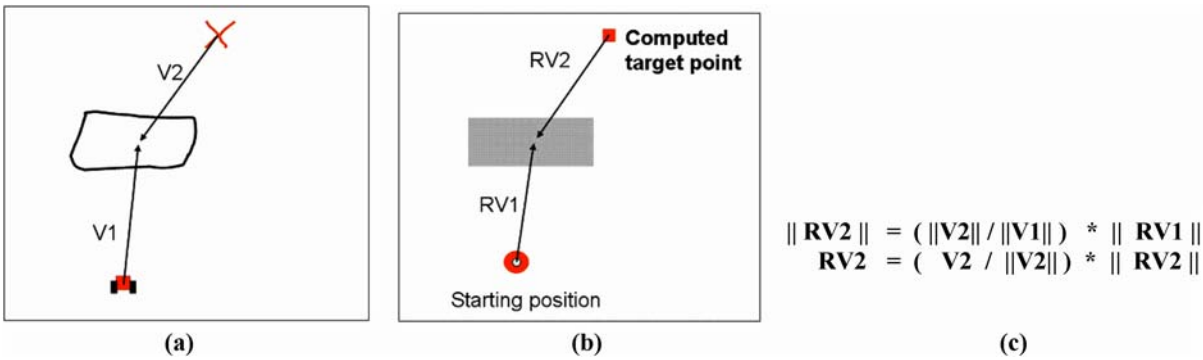

Fig. 6 Conversion of a go-to command from the sketchpad to world coordinates in the robot scene. (a) Sketchpad showing the robot icon and a sketched landmark. "X" marks the goal location sketched by the user. (b) Robot scene. (c) Equations. Vector V1 describes the relation between the robot and the landmark; vector V2 describes the relation between the goal and the landmark. The computed target location is identified by using V1 and V2 in combination with RV1 and RV2, from the real robot environment. RV2 is the only quantity that is not initially known

from the sketch as vectors and sent to the robot to be recomputed according to the relative positions of the robot and the landmark in the real environment. If, due to sketch inaccuracies, the computed point is inside a landmark or on top of another robot, the target point is shifted along the target vector. Figure 6 shows how these two vector quantities are computed in the sketch and for the robot.

If a single go-to command is issued for a group of robots, then the robot that is closest to the goal is given this location as its target. All other robots are ordered according to their respective distances to the goal point. Remaining robots are assigned different goals that are computed at different respective offset values along a line that originates at the goal location and is in the direction of a vector from the centroid of the landmark to the goal point. Figure 5 shows an example. Offset values can be changed via a menu option. The order of the robots is used to determine how long each should wait to begin moving in order to avoid congestion in navigation.

Path commands are computed by segmenting a stroke into a series of intermediate points based on a fixed interval length (set as a parameter in the options menu). Each consecutive pair of intermediate points is turned into a go-to command in the same fashion as described above. For each pair of intermediate points, the go-to command is computed with respect to the landmark that is closest to the ending intermediate point. Figure 7 illustrates this procedure.

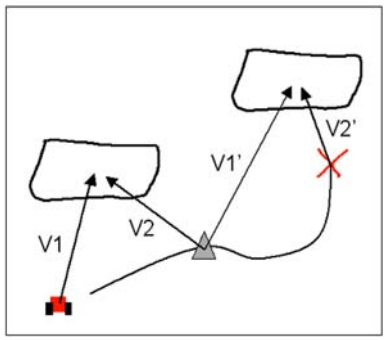

Fig. 7 The segmentation of a sketched path and the sequence of computed go-to commands. The path originates at the robot and is drawn up to the point where the "X" is displayed. Intermediate points are calculated and shown as gray triangles that appear along the path. Path navigation is performed by sending each robot to the sequence of computed intermediate points, and then to the goal location. Vectors V1 and V2 are the first to be extracted and sent to the robot for navigation. The robot is then sent V1' and V2', which are computed from the intermediate point to the goal, and are to be carried out after the intermediate point is reached

4 Synchronization of the sketch with the robots

To provide real time feedback of robot locations on the sketchpad, information about each robot relative to the landmarks in the real environment is extracted and sent to the interface. If a robot is not in motion, it sends back a command that tells the interface not to update. Moving robots send back their starting and ending vectors, along with a present vector that is computed from the robot's current location to the landmark closest to the goal. These vectors are used in combination with V1 and V2 to compute a new updated location. An example is shown in Fig. 8.

We investigated different strategies to display the final robot location after command completion. One option is to move the robot icon to the sketched goal location on the sketchpad. A second option is to leave the robot at its last updated location. In pilot trials, the second option appeared to be misleading and confusing to users. Depending on the quality of the sketch and where the robot stopped in the real environment, there can be a discrepancy in where the robot is displayed on the sketchpad and where the user expected to see the robot. Thus, our default mode is to move the robot icon to the sketched target position.

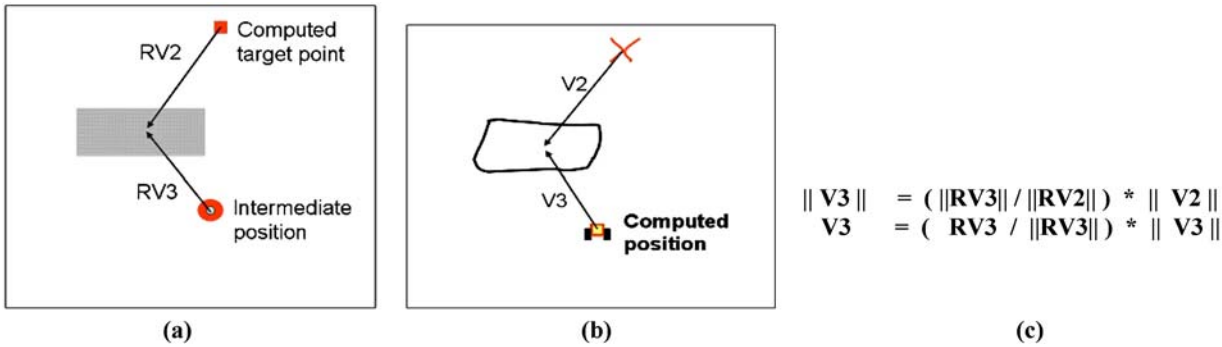

Fig. 8 Calculation of the robot's updated location on the sketchpad from the robot location in the real world. (a) Robot scene. (b) Sketchpad. (c) Equations. Vectors RV2 and RV3 convey the relationship between the robot and the real world landmark. The computed position on the sketchpad is identified by using RV2 and RV3 in combination with V2 and V3 from the sketch pad. V3 is the only quantity that is not initially known

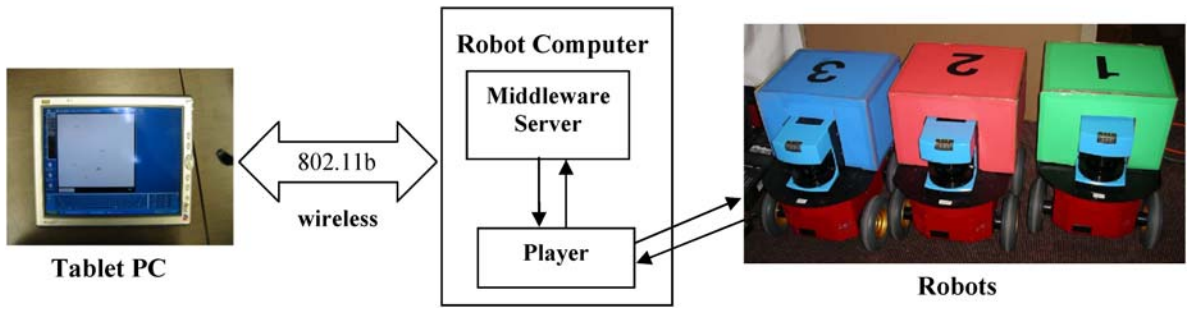

Fig. 9 The system architecture showing the Tablet PC with connections to the robot team. Each robot has a computer running the middleware server and Player

5 System integration

The robots used for this experiment were commercially available, four-wheeled, skid-steer Pioneer robots equipped with laser rangefinders and internal gyroscopes, as shown in Fig. 9. Each robot has an on-board PC which runs the Linux operating system. The robot control software was developed with the Player/Stage software suite (Gerkey et al., 2003), available from http://playerstage.sourceforge.net. A middleware server developed in-house provides a connection between Player and the Tablet PC sketchpad via a wireless access bridge. Figure 9 illustrates the system architecture.

The Tablet PC runs the sketch interface software under the Windows operating system. This software is responsible for recognizing the pen strokes and interpreting their meaning as described in the previous sections (e.g. creation/editing of robots and landmarks, robot move commands). These data are then passed to a middleware server running on the robot, which translates the commands from the sketch interface and issues commands to the robot. Odometry and sensor information is sent back from the robot through the middleware server to the sketch interface on the Tablet PC to update the display of the robot.

Wireless bridges are used to facilitate bidirectional communication over an 802.11b network between the robots and the Tablet PC. The commercially available bridges, powered by the robots' batteries, are used because the robots lack on-board wireless capabilities.

Each robot computer runs two main software components. The middleware server acts as the intermediary between the sketch interface and the robot control software. The underlying control software used in this project is the Player/Stage suite. It has native support for the robots we used, along with a simulated environment for rapid testing of the interface. By using this approach, we were able to switch from a simulated environment to the real environment rapidly. An instance of Player is run on each robot in order to open up the appropriate drivers, path planners, and obstacle avoidance on the robot. Currently, the AMCL (Adaptive Monte-Carlo Localization) and VFH (Borenstein and Koren, 1991) are used to update the robot's odometry and provide obstacle avoidance, respectively. These programs are implementations from the Player/Stage software suite.

6 Usability study

A usability study was conducted in conjunction with the Robotics Competition at the AAAI 2005 conference. The study was designed to test the sketch interface concept with a

group of users who are not necessarily robot experts. We also designed the study to investigate how users compensate for a change in the environment. As part of the study, we collected data on the participants' backgrounds and suggestions for improvements.

6.1 Experimental set-up

Participants were first acquainted with the sketch interface and allowed to use it until they felt comfortable. They were then shown the environment (Fig. 10) in which they were to perform the experiment. The environment consisted of the three robots named 1, 2, and 3, and three objects: a box, a crate, and a ball. The numeric robot names were chosen so that users could easily remember them. The sketch interface does not restrict the naming of robots.

The participant was then taken to an isolated area where he or she was unable to see the robots. Each participant was asked to perform the following five tasks:

1 Draw and label the robots and the objects;
2 Navigate the robots to a position to the northwest of the ball;
3 Navigate robot 3 to a position south of the box;
4 Navigate robot 1 to the north of the ball, robot 3 to the west of the ball but out of robot 1's sight, and robot 2 to the north of the box so that robot 1 can see robot 2 but robot 3 cannot;
5 Send the robots back to their starting positions.

To simplify the experiment, we fixed the menu options in the interface for a set of standard parameters. The arrow option for issuing robot commands was not used in the study. In order to provide a consistent experimental environment, i.e., to facilitate a controlled experiment, participants interacted with the robots within the Player/Stage simulator environment.

Fig. 10 The environment of the experiment, a space approximately 6.7 × 7 m

6.2 Participants

The average age of the 23 participants was 34.3 years with a standard deviation of 9.9; most held advanced degrees in computer-related fields. Participants were not paid. While most were very familiar with computers, few had experience using Tablet PCs. Several participants had extensive experience with video games. Only a few had experience with robots.

We are trying to get away from the notion of needing special robot operators for controlling robots. Instead, we view the role of the user as being a supervisor rather than just an operator. We foresee the users as being experts in the application domain who may or may not be experienced with robots. For the sketchpad interface, target users would ultimately be individuals with some sort of computer background; thus, we chose a sample of individuals from the selected population. Participants filled out questionnaires at the beginning and at the end of the experiment to provide feedback. This information was collected to help guide future improvements.

Each participant was randomly assigned to one of two groups: one group interacted with the robots in an unaltered environment, and one group had a slightly altered environment from the one shown in Fig. 10. In the altered environment, the box was moved to the west of the ball and shifted slightly south. This allowed us to see what kinds of coping strategies people use to compensate for the changed state of the environment.

Participants were told that the environment might change after they began using the sketch interface to control the robots; however, they were not told that there were two experimental conditions, nor in which condition they were participating. Participants were told that their interactions with the sketchpad would result in changes to the display. For example, if they moved an icon, the display would reflect that change. They were not told anything more about the potential changes. We did not inform users that updates might be correct or incorrect. We wanted to see their unbiased reactions in dealing with both correct and incorrect information on the sketchpad display.

6.3 Performance results

Most of the sketches drawn by the participants were an accurate qualitative representation of the environment. To be considered an accurate sketch, the participant had to correctly draw the three objects and the three robots and assign correct labels. Of the 23 subjects, only 2 had to be eliminated for incorrect sketches of the environment. The remaining sketches appeared qualitatively similar to those shown in Figs. 11 and 12, which are typical sketches collected from the participants. Five additional test subjects were excluded due to incomplete data collection (i.e., problems in video taping).

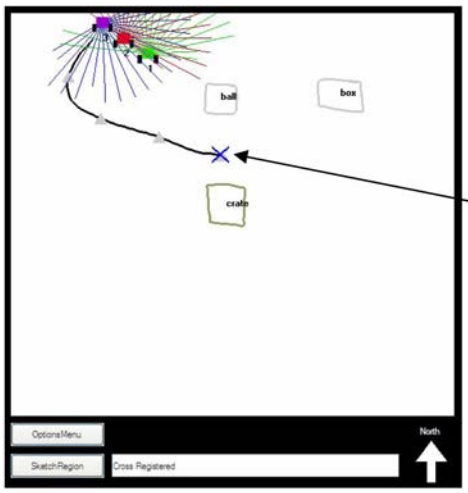

Fig. 11 A participant uses a path command to move robot 3 to a position south of the ball

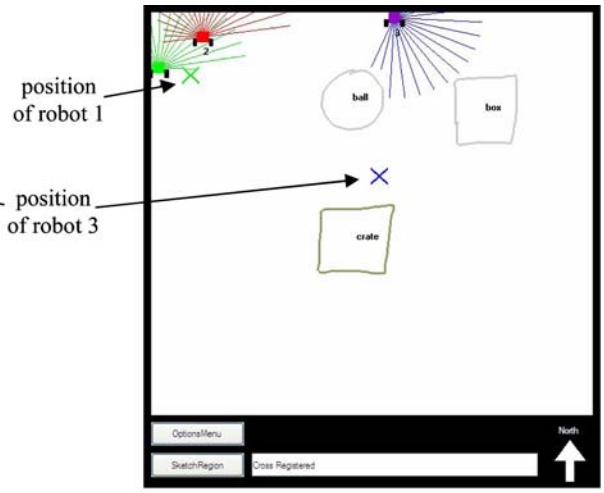

Fig. 12 Another sketch from a different user, directing robot 3 to go south of the ball

We report results on 16 participants (9 in the unmodified environment and 7 in the modified environment).

Generally, participants tended to favor one of the navigation commands (either path or go-to commands). However, no statistically significant difference was found in the performance of the two command types. In the unmodified environment, the 9 participants used a total of 106 path commands and 120 go-to commands, compared to 70 path commands and 89 go-to commands used by the 7 participants in the modified environment. Thus, the averages for the unmodified environment were 11.8 path commands (std. dev. = 3.3) and 13.3 go-to commands (std. dev. = 4.7), while the averages for the modified environment were 10.0 path commands (std. dev. = 2.9) and 12.7 go-to commands (std. dev. = 3.1).

We did not find statistically significant differences in command type used, navigation task time, or task completion for the two experimental conditions or for any other grouping, including those participants with some prior experience with robots or computer games. In general, the standard deviations tended to be large for each group so that no comparisons were statistically different within the sample sizes tested. Given this result, we intend to revisit the data at a later time in order to determine what might be responsible for this distribution.

Task times for the two experimental groups are summarized in Fig. 13. In the unmodified environment, participants took an average of 765 seconds to perform the experiment, while the participants in the changed environment took an average of 842 seconds (with standard deviations of 216 and 220 seconds, respectively). In both groups, as we expected, task 4, the most complex of the tasks, took the most time.

If the subjects in either group correctly labelled the environment, they had a high probability of successfully completing all of the tasks. All participants for the unmodified environment completed all tasks except for task 4; only 67% of these participants completed task 4. For participants with a modified world, 77% completed task 4 and all completed the remaining tasks.

Participants used the editing capabilities to move landmarks. Across the entire set of participants, only 5 major moves were made, where the move was more than the size of the object moved. The remaining 41 moves were minor tweaks. At the end, participants were mostly successful in producing an accurate final sketch with the robots in the designated locations. In the unmodified environment, seven participants scored 6 out of 6 points, where one point was awarded for each correct landmark position and one point for each correct robot position. The remaining participant scored 5 out of 6 points. In the modified environment, six participants scored 6 out of 7 points. Here, the seventh point was included for correctly adding an object in the new position west of the ball. The remaining participant scored 7 out of 7 points. These findings seem to indicate that there was a decreased situation awareness for participants in the modified environment. That is, participants did not have an accurate mental map of the modified scene, as most of them missed the placement of the new obstacle.

6.4 Discussion

Most users felt that the interface was highly applicable to the task of guiding mobile robots. The average rating given in the post-experiment survey using a 5-point Likert scale (Likert, 1932) was 4.2 with 1 being very negative and 5 being very positive. Participants indicated in the survey that the system was good enough to accomplish the tasks they were assigned; the average overall opinion of the interface was rated 3.5. Most also felt that, with some enhancements, such as the ability to hear audio output when errors had been committed, and

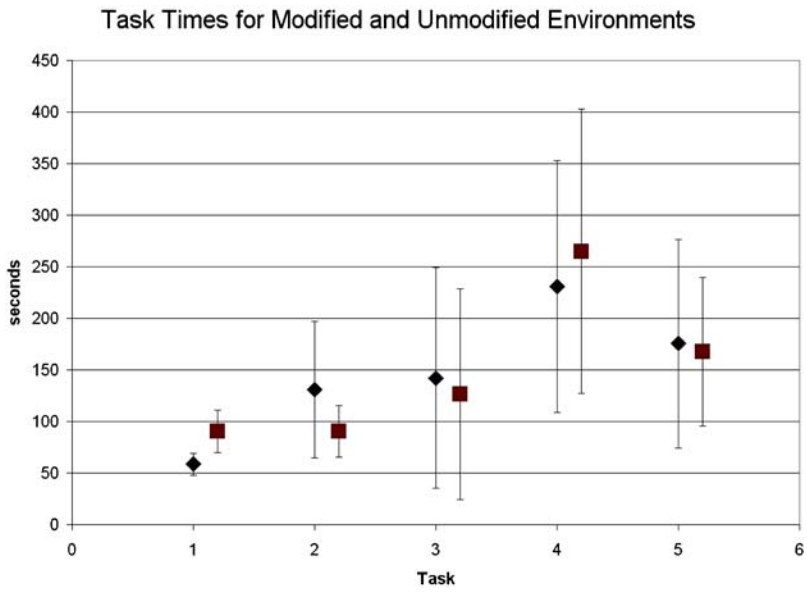

Fig. 13 Task times for the two experimental groups with error bars at one standard deviation. The diamond designates the unmodified environment, and the square designates the modified environment

the ability to verbally command robots to make adjustments, the sketch interface would be particularly useful in similar scenarios. Potential adjustments for verbal commands include issuing a Stop or a Turn-around command and minor nudge commands, e.g. "Move slightly more to the left."

The interface was apparently easy to learn. We did not time participants' training times, but it is our observation that all participants took a relatively short time in learning the environment and the interface.

Participants used the ability to edit their sketches, i.e. move landmarks if they thought they were positioned incorrectly based on sensor feedback. There were very few participants who used this feature to make major landmark moves; most moves were minor, consisting mainly of "tweaks". This shows that, for the most part, the sketches preserved the qualitative information of the environment and were "good enough" to accomplish the task at hand.

Although we did not find any statistically significant differences between the two experimental conditions, the study was helpful in identifying problem areas to address in future work. One usability problem resulted from the small space in which the study was conducted (6.7 $\times$ 7 m). When the robots were moved to the northwest of the ball, there was a tendency for them to get stuck in the corner. This was due to the robots using VFH for obstacle avoidance and being too close to each other. Also, there was a problem when the goal location was calculated very close to an obstacle (or another stationary robot), which caused it to be unrealizable. These problems were further complicated by the restriction on a robot turning around in place. The interface did not provide a way to change the robot orientation other than to turn through a wide expanse of space. The confined space made maneuvering even more difficult. These challenges in maneuverability of the robots within the confined space may have had a larger impact than the differences between the experimental groups and, thus, contributed to a finding of no statistically significant differences.

Some users noted that the behavior of the robot deviated from the sketched path, which was due to an obstacle (either known or unknown by the user). This problem could have been exacerbated by the relatively slow update rate (2 sec.), thereby causing all participants to react in similar ways, regardless of their experimental condition. The slow update rate decreased the situation awareness (SA) of users by reducing the time resolution of the range sensor readings. Participants were less likely to see the connection between the sensor readings and an obstacle. Participants also noted that the SA was affected by the consistent orientation of the robot icon, i.e., the robot icon did not rotate to indicate heading so it did not match the heading shown by the sensor range lines, which did rotate according to heading. Some users disliked the changing colors of the robots to indicate selection or motion and suggested that other techniques be used to show a status change. Changes have been made in a new version of the interface: increasing the update rate, changing the robot icon by rotating it according to the sensed heading, and animating the wheels to indicate motion instead of using a color change.

The study also gave us an opportunity to test the sketch understanding algorithms on a diverse set of users. In general, the system worked well. However, some participants had problems placing the robot in the sketch, which was done by drawing a squiggle. Some squiggles were interpreted as very small landmarks. This can be modified with a menu option to accommodate individual styles. In addition, there were some problems with lassoing a group of robots. The

selection algorithm sometimes missed one of the robots. This has since been corrected with a different algorithm.

In spite of these difficulties, participants were able to complete the tasks in most cases. However, we conjecture that another reason why the reaction times and coping strategies between the two groups were not statistically significant is that humans are adept at coping with a dynamic environment. In the study, there was not enough of a change to cause a significant burden for the participants.

7 Concluding remarks

As robotics research matures, the field is moving toward systems that support the management of multiple robots and teams of collaborative agents. To this end, and because exact representations of environments are not always available to human users of such systems, we have designed a sketch interface that handles qualitative input from human users rather than one that has to rely solely on quantitative information.

We conducted a usability study with the sketch interface to determine how people manage multiple robots simultaneously. Unbeknownst to the subjects, participants were randomly assigned to one of two groups. The first group controlled the robots in an unaltered environment. The second group controlled robots via the sketchpad in a slightly altered environment from the one they had been shown.

We found no significant differences in task time completion in either group, thereby suggesting that when slight changes are made in the environment from the one that is expected, humans are well-prepared to cope with those changes. From this, we conclude that our approach in designing an interface that tolerates the qualitative interchange of information can be useful in working with collaborative teams of robots.

At this point, it is unclear as to how well the approach will scale to more complex environments. Results in previous studies with a single robot have been promising; as many as 8 landmark objects have been used (Chronis and Skubic, 2004). Also, we are encouraged by the continued use of sketched maps as a communication tool in human-to-human knowledge transfer. As in other uses of sketches, we assume that the user is familiar with the environment or can see the environment well enough to sketch it. We also assume that the user knows the approximate starting position of the robots with respect to the landmarks. Certainly, the approach will benefit from landmark recognition and being able to automatically associate a sensed landmark in the physical environment with a sketched landmark.

The results of the usability study validate the concept of a sketch interface for controlling a team of robots. We are currently extending the interface to provide automated scene matching between the sketch and the physical world as sensed by the robots. Suggestions from the participants in the study are also being used to drive the next iteration of the sketch interface.

Acknowledgments Funding for the project was provided in part by the Naval Research Laboratory and the Office of Naval Research under work order N0001405WX20057. The authors also thank Scott Thomas and Greg Trafton from NRL as well as Vince Cross from Auburn University for their help in conducting the usability study and analysis.

References

Alvarado, C., Sezgin, T.M., Scott, D., Hammond, T., Kasheff, Z., Oltmans, M., and Davis, R. 2002. A framework for multi-domain sketch recognition. In *Papers from the 2002 AAAI Spring Symposium on Sketch Understanding, AAAI Technical Report SS-02-08*, Palo Alto, CA, March, pp. 1–8.

Anderson, D., Bailey, C., and Skubic, M. 2004. Hidden Markov model symbol recognition for sketch based interfaces. In *Papers from the 2004 AAAI Fall Symposium on Making Pen-Based Interaction Intelligent and Natural, AAAI Technical Report FS-04-06*, Washington, DC.

Borenstein, J. and Koren, Y. 1991. The vector field histogram—fast obstacle avoidance for mobile robots. *IEEE Trans. on Robotics and Automation*, 7(3):278–288.

Burke, J.L. and Murphy, R.R. 2004. Human-robot interaction in USAR technical search: Two heads are better than one. In *Proc. IEEE International Workshop on Robot and Human Interactive Communication*, Kurashiki, Okayama, Japan, pp. 307–312.

Bresenham, J.E. 1965. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30.

Chronis, G. and Skubic, M. 2004. Robot navigation using qualitative landmark states from sketched route maps. In *Proc. IEEE Intl. Conf. Robotics and Automation*, New Orleans, LA, pp. 1530–1535.

Fong, T.W., Thorpe, C., and Glass, B. 2003. PdaDriver: A handheld system for remote driving. In *Proc. IEEE Intl. Conf. Advanced Robotics*, Coimbra, Portugal.

Fong, T, Thorpe, C., and Baur, C. 2003. Multi-robot remote driving with collaborative control. *IEEE Trans. Industrial Electronics*, 50(4):699–704.

Freksa, C., Moratz, R., and Barkowsky, T. 2000. Schematic maps for robot navigation. In C. Freksa, W. Brauer, C. Habel, and K. Wender (eds.), *Spatial Cognition II: Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, Berlin: Springer, pp. 100–114.

Gerkey, B.P., Vaughan, R.T., and Howard, A. 2003. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proc. IEEE Intl. Conf. Advanced Robotics*, Coimbra, Portugal.

Hammond, T. and Davis, R. 2004. Automatically transforming symbolic shape descriptions for use in sketch recognition. In *The Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, July.

Hammond, T. and Davis, R. 2005. LADDER, a sketching language for user interface developers Elsevier. *Computers and Graphics*, 29:518–532.

Kawamura, K., Koku, A.B., Wilkes, D.M., Peters II, R.A., and Sekmen, A. 2002. Toward egocentric navigation. *Intl. Journal Robotics and Automation*, 17(4):135–145.

Likert, R. 1932. A technique for the measurement of attitudes. *Archives of Psychology*, 140:5–53.

Lundberg, C., Barck-Holst, C., Folkeson, J., and Christensen, H.I. 2003. PDA interface for a field robot. In *Proc. 2003 IEEE/RSJ Intl. Conf. Intelligent Robots and Systems*, Las Vegas, NV, pp. 2882–2887.

Perzanowski, D., Schultz, A.C., Adams, W., Marsh, E., and Bugajska, M. 2001. Building a multimodal human-robot interface. *IEEE Intelligent Systems*, 16(1):16–21.

Setalaphruk, V., Ueno, A., Kume, I., and Kono, Y. 2003. Robot navigation in corridor environments using a sketch floor map. In *Proc. IEEE Intl. Symp. Computation Intelligence in Robotics and Automation*, Kobe, Japan, pp. 552– 557.

Sezgin, T.M. and Davis, R. 2004. Scale-space based feature point detection for digital ink. In *Papers from the 2004 AAAI Fall Symposium on Making Pen-Based Interaction Intelligent and Natural, AAAI Technical Report FS-04-06*, Washington, DC.

Sezgin, T.M. and Davis, R. 2005. HMM-based efficient sketch recognition. In *Proceedings of the International Conference on Intelligent User Interfaces (IUI'05)*, New York, NY.

Marjorie Skubic received her Ph.D. in Computer Science from Texas A&M University, where she specialized in distributed telerobotics and robot programming by demonstration. She is currently an Associate Professor in the Electrical and Computer Engineering Department at the University of Missouri-Columbia with a joint appointment in Computer Science. Dr. Skubic has over 80 publications and has received funding by the National Science Foundation, the Naval Research Lab, the U.S. Army, and the Administration on Aging. In addition to her academic experience, she has spent 14 years working in industry on realtime applications such as data acquisition and automation. Her current research interests include sensory perception, computational intelligence, spatial referencing interfaces, human-robot interaction, and sensor networks for eldercare. Dr. Skubic has recently established the Center for Eldercare and Rehabilitation Technology at the University of Missouri and serves as the Center Director for this multidisciplinary team.

Derek T. Anderson received his M.S. in Computer Science in 2005 from the University of Missouri-Columbia. He was a research assistant at the University of Missouri-Columbia in the Computational Intelligence Research Laboratory from 2002 to 2006 and is currently pursuing a PhD in Electrical and Computer Engineering. He is presently a Pre-Doctoral fellow for the National Library of Medicine. Current research interests include real-time image processing for silhouette segmentation on graphics processing units and temporal pattern recognition for activity analysis and fall detection in an Eldercare setting.

Samuel Blisard received his M.S. in Computer Science from the University of Missouri-Columbia and is currently a Ph.D. student in Electrical and Computer Engineering at the University of Missouri-Columbia. He also works at the Navy Center for Applied Research in Artificial Intelligence at the Naval Research Laboratory, Washington DC. His research interests include human-robot interaction, spatial referencing language, and computational intelligence.

Dennis Perzanowski received his Ph.D. in Linguistics from New York University in 1982. His graduate training not only included theoretical syntax, but he also worked at the Linguistic String Project of NYU. Based on his work in computational linguistics, Dr. Perzanowski accepted a position in the Intelligent Multimodal/Multimedia Systems Section at the Naval Research Laboratory (NRL), Washington, DC, in 1984. His work in natural language led to work on human-robot interaction with the Intelligent Systems Section, also at NRL. His research interests are multimodal interfaces; spatial reasoning and language; and conversation and dialog management. He has authored over 30 journal articles and conference proceedings in the fields of natural language understanding, multimodal interfaces, and human-robot interaction. His most recent publication is Using Vision, Acoustics, and Natural Language for Disambiguation to appear in the Proceedings of the Second International Conference on Human-Robot Interaction.

Alan C. Schultz is Director of the Navy Center for Applied Research in Artificial Intelligence at the Naval Research Laboratory (NRL) in Washington D.C. He has twenty years experience and over 80 publications in robotics, human-robot interaction, and machine learning, and is responsible for establishing and running the robotics laboratory at NRL.

Mr. Schultz was selected to teach at the first IEEE/RAS Summer School on Human-Robot Interaction, has been editor of several collections in multi-robot systems, and has chaired many conferences and workshops in robotics and human-robot interaction. Mr. Schultz received his M.S. in Computer Science from George Mason University in 1988. He is the recipient of twelve Navy Special Achievement awards for significant contributions, and the Alan Berman Research Publication Award. His research is in the areas of human-robot interaction, machine learning, autonomous robotics, and adaptive systems.