Using a Qualitative Sketch to Control a Team of Robots

Marjorie Skubic, Derek Anderson, Samuel Blisard

Electrical and Computer Engineering Department
University of Missouri-Columbia
Columbia, MO

Dennis Perzanowski, Alan Schultz

Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory
Washington, DC

***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 sketches a qualitative 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.**

Index Terms – human-robot interaction, sketch-based navigation, qualitative map.

I. 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. 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 recent 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 sketches a qualitative 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.

A sketch-based interface has been proposed previously. Perzanowski et al. [1] 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 to indicate target positions of the robot. Lundberg et al. [2] 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 [3] 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 the PDA interface also supports multi-robot control and sketching waypoints on top of a map as well as an image [4].

Other work has included the use of a qualitative map. Chronis et al. [5] 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 using landmark states. Kawamura et al. [6] 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. Freksa et al. [7] 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. [8] use a scanned, hand-drawn map of an indoor scene (with walls and corridors) and extract a topological map for controlling a robot.

Fig. 1. The team of robots included in the usability study

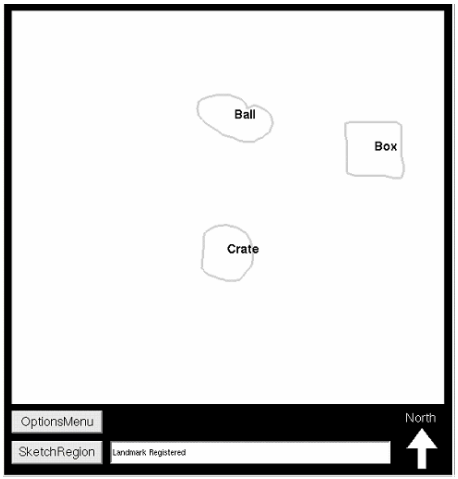


Fig. 2.
Sketching landmarks

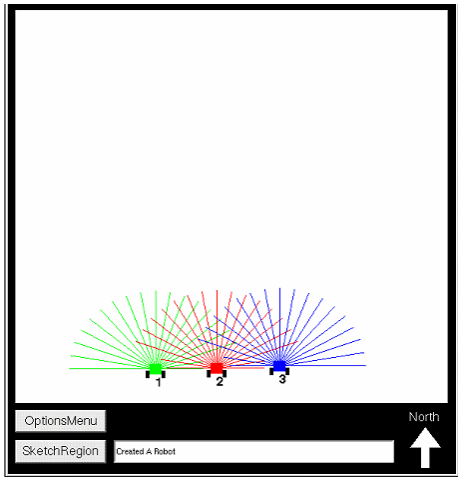


Fig. 3
Sketching robots

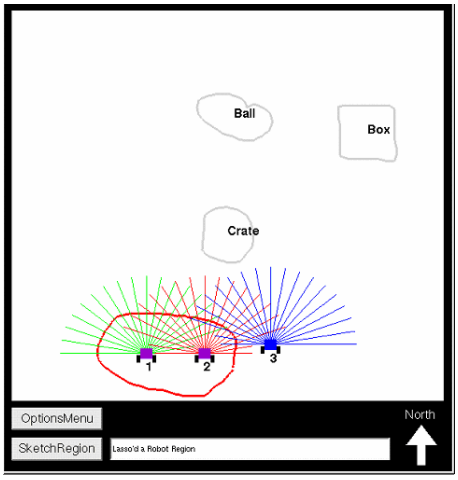


Fig. 4
Lassoing a group of robots

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. The interface has been investigated with a usability study in which 23 users were asked to perform a series of tasks. The robot team is shown in Fig. 1.

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 issues that provide feedback from the robot to the sketch platform. A usability study and results are also included.

II. 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.

Users add environment landmarks by sketching a closed polygon anywhere on the screen (shown in Fig. 2). 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 evidence grid map. In the prototype interface, this correlation between sketch and robot objects is manually handled by the user providing the identifiers.

To create a robot, the user sketches a small concentrated stroke anywhere on the screen and labels the robot with a name. A robot icon is displayed in place of the stroke and, if communications can be established with the real robot, then sensor feedback is shown from the range sensors. Fig. 3 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. Fig. 4 shows two robots being selected; their color changes to purple to indicate selection.

Identifying the robots in a lasso is done using the Bresenham line algorithm [9] 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 location is checked to see if it was a flood filled or boundary point.

Feedback from 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.

Navigational commands may be issued to robots after one or more landmarks are sketched. Because we use qualitative and not quantitative information, navigational commands are issued relative to landmarks. Sketching an "X", which is two intersecting short lines, issues a Go-to command for all selected robots. If a user wants the robots to follow a route, he 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.

Fig. 5 shows a scenario in which both path and Go-to commands are issued. The landmark that is closest to the last sketched goal point changes color to indicate its use as a reference object. The segmented path is shown as a sequence of gray triangles. All target locations are drawn the same color as the corresponding robot for clarity. The center of each robot changes color to yellow to indicate its motion.

In Fig. 5, 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 navigational commands (e.g., for synchronization of robots), a menu option allows simultaneous execution of robot commands after an arrow is sketched. The symbol recognition method used to classify the arrow is based on Hidden Markov Models [10].

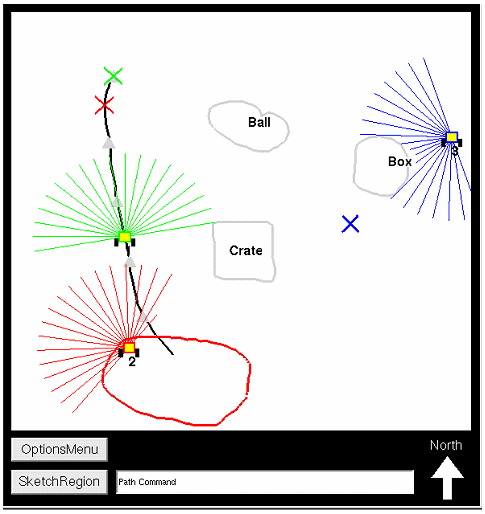
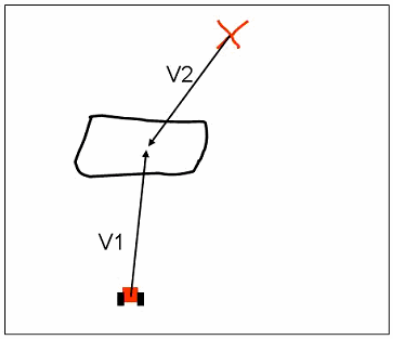


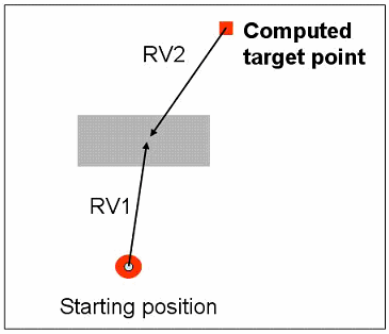
Fig. 5. Robots 1 and 2 are instructed to follow a path while robot 3 is directed to a target location. The path has been segmented into a sequence of intermediate points, shown as gray triangles along the path. The yellow center of the robot indicates motion. Each robot displays its laser readings in its corresponding color.

III. 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 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. Fig. 6 shows how these two vector quantities are computed in the sketch and for the robot.



(a)



(b)

$$\| \mathbf{RV2} \| = (\|\mathbf{V2}\| / \|\mathbf{V1}\|) * \| \mathbf{RV1} \|$$
$$\mathbf{RV2} = (\mathbf{V2} / \|\mathbf{V2}\|) * \| \mathbf{RV2} \|$$

(c)

Fig. 6. Conversion of a Go-to command from the sketchpad to world coordinates in the robot scene. (a) Sketchpad. (b) Robot scene. (c) Equations. "X" marks the goal location sketched by the user. 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.

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. Fig. 7 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. Fig. 8 illustrates this procedure.

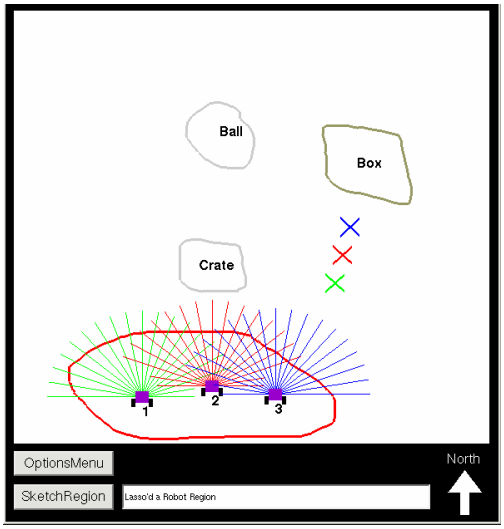


Fig. 7. For robot group commands, target points are computed according to the distance of each robot to the sketched goal point.

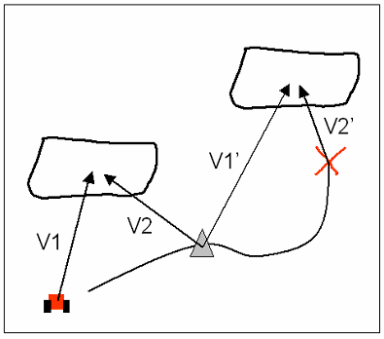
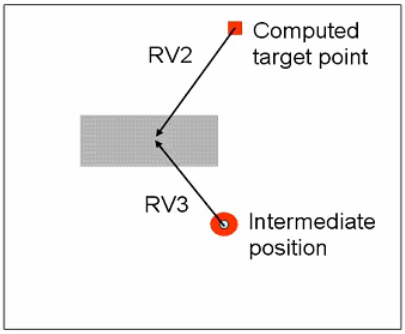


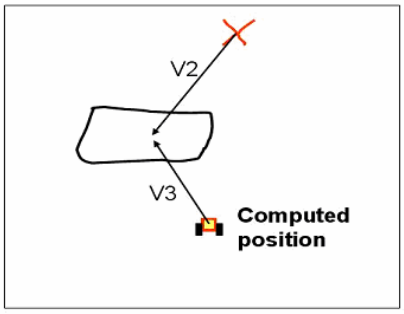
Fig. 8. The segmentation of a sketched path and the sequence of computed Go-to commands. The path originates at the robot and is drawn up to the point where the "X" is displayed. Intermediate points are calculated and shown as gray triangles that appear along the path. Path navigation is performed by 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.

IV. 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. 9.



(a)



(b)

$$\| \mathbf{V3} \| = (\|\mathbf{RV3}\| / \|\mathbf{RV2}\|) * \| \mathbf{V2} \|$$
$$\mathbf{V3} = (\mathbf{RV3} / \|\mathbf{RV3}\|) * \| \mathbf{V3} \|$$

(c)

Fig. 9. 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.

There is a final, subjective matter about how to display the stopping location on the interface after a robot makes it to the goal. If a robot completed the command and moved to the desired position in the real world, then the robot is translated on the sketchpad to the goal location that the user sketched. Another option, which can be enabled through the options menu, involves keeping the robot at its last updated location. However, 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. Our default mode is to move the robot icon to the sketched target position.

V. 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 that 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.

A. *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, 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 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 ball;
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.

Fig. 10. The environment of the experiment.

B. *Participants*

The average age of participants was 33.5 years; 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.

Each participant was randomly assigned to one of two groups: one group with an unaltered environment and one group with a slightly altered environment from the one shown. 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 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.

C. *Robot Implementation*

The robots used for this experiment were commercially available, four-wheeled, slip-steer robots equipped with laser rangefinders and internal gyroscopes (Fig. 1). The robots were controlled with software developed through the Player/Stage project [11]. The robots used wireless access bridges to communicate with the controlling computer through the use of the IEEE 802.11b protocol. In order to provide a consistent experimental environment, participants interacted with the simulator, and the robots were directed by manually issuing waypoints from the controlling computer.

D. *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 Fig. 11 and 12. Five additional test subjects were excluded due to incomplete data collection (i.e., problems in video taping). We report results on 16 participants (8 in each group).

Typical sketches collected from the participants are shown in Fig 11 and 12. 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. We did not find statistically significant differences in 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. In general the standard deviations tended to be large for each group.

Task times for the two experimental groups are summarized in Fig. 13 and 14. 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, task 4 took the most time.

If the subjects in either group correctly labelled the environment, they had a very 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.

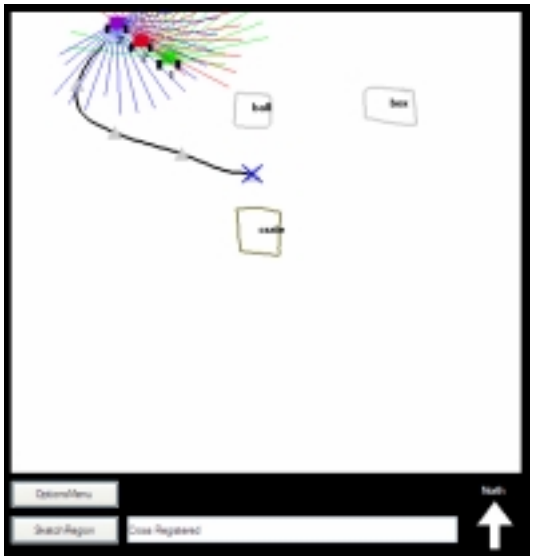


Fig. 11. A participant uses a Path command to move robot 3 to a position south of the ball.

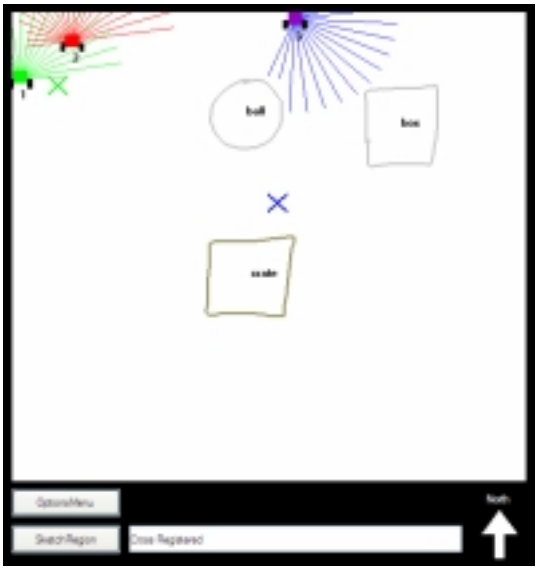


Fig. 12. Another sketch from a different user, directing robot 3 to go south of the ball.

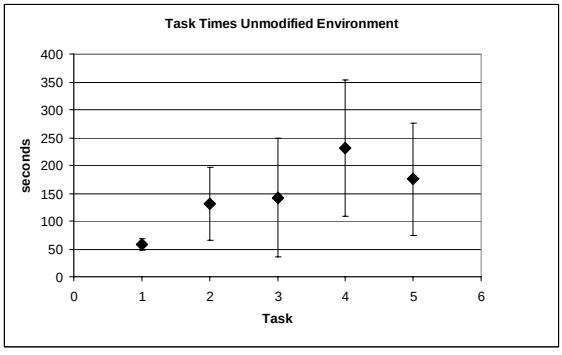


Fig. 13. . Task Times for the Unmodified Environment with Error Bars at One Standard Deviation.

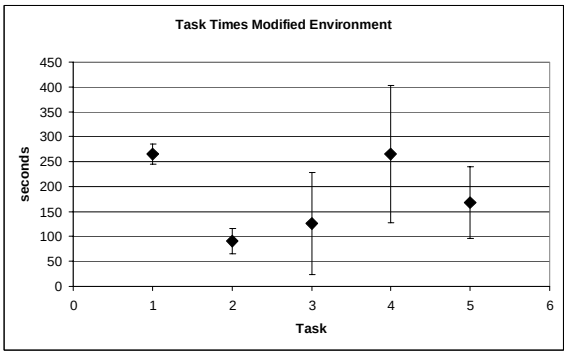


Fig. 14. Task Times for the Modified Environment with Error Bars at One Standard Deviation.

E. 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 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 the ability to verbally command robots to make minor adjustments, e.g. "Move slightly more to the left," the sketch interface would be particularly useful in similar scenarios.

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.

Users had the ability to "tweak" their sketches, i.e. move objects if they thought they were positioned incorrectly based on sensor feedback. There were very few participants who used this feature to make major moves of objects, where the move was more than the size of the object being moved. Most object 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.

There is room for improvement. One usability problem resulted from the small space in which the study was conducted (6.7 x 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.

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 was artificially constrained and will be increased in the future.

We conjecture that, another reason why the reaction times and coping strategies between the two groups were not statistically significant is that humans are talented at coping with a dynamic environment. In the study, there was not enough of a change to cause a significant burden for the participants.

V. Concluding Remarks

As robotics research matures, it 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 designed a sketchpad 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 sketchpad 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.

The results of the usability study validate the concept of a sketchpad interface for controlling a team of robots. In future work, we will extend 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 will also drive the next iteration of the sketchpad interface.

Acknowledgements

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

[1] D. Perzanowski, A.C. Schultz, W. Adams, E. Marsh, M. Bugajska, "Building a multimodal human-robot interface," *IEEE Intelligent Systems*, pp. 16-20, Jan/Feb, 2001.

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

[3] T.W. Fong, C. Thorpe, and B. Glass, "PdaDriver: A Handheld System for Remote Driving," *IEEE Intl. Conf. on Advanced Robotics 2003*, July, 2003.

[4] T. Fong, C. Thorpe, and C. Baur, "Multi-Robot Remote Driving with Collaborative Control," *IEEE Transactions on Industrial Electronics*, vol. 50, no. 4, pp. 699-704, 2003.

[5] G. Chronis and M. Skubic, "Robot Navigation Using Qualitative Landmark States from Sketched Route Maps," in *Proc. 2004 IEEE Intl. Conf. on Robotics and Automation*, New Orleans, LA, April, 2004, pp. 1530-1535.

[6] K. Kawamura, A.B. Koku, D.M. Wilkes, R.A. Peters II and A. Sekmen, "Toward Egocentric Navigation", *Intl. Journal of Robotics and Automation*, vol. 17, no. 4, 2002, pp. 135-145.

[7] C. Freksa, R. Moratz, and T. Barkowsky, "Schematic Maps for Robot Navigation", *in Spatial Cognition II: Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, C. Freksa, W. Brauer, C. Habel, K. Wender (ed.), Berlin: Springer, 2000, pp. 100-114.

[8] V. Setalaphruk, A. Ueno, I. Kume, and Y. Kono, "Robot Navigation in Corridor Environments using a Sketch Floor Map," in *Proc. 2003 IEEE Intl. Symp. On Computation Intelligence in Robotics and Automation*, July, 2003, Kobe, Japan, pp. 552-557.

[9] Jack E. Bresenham**,** *Algorithm for Computer Control of a Digital Plotter*, IBM Systems Journal, 4(1):25-30, 1965

[10] Anderson, D., Bailey, C., and Skubic, M. 2004. Hidden Markov Model Symbol Recognition for Sketch Based Interfaces. *AAAI Fall Workshop on Making Pen-Based Interaction Intelligent and Natural*. Washington, DC. October 2004.

[11] http://playerstage.sourceforge.net