Command and Control of Robot Teams

Jill Drury, Laurel D. Riek, Alan D. Christiansen,
Zachary T. Eyler-Walker, Andrea J. Maggi, and David B. Smith

The MITRE Corporation
7515 Colshire Drive
McLean, VA 22102
(703) 883-6000 www.mitre.org
{jldrury, laurel, adc, zach, amaggi, daves}@mitre.org

Abstract

We describe our efforts to produce a command and control (C2) interface for a team of mobile robots. Specifically, we developed a robot interface with the eventual goal of facilitating operators as they perform search and rescue tasks in an indoor environment that includes simulated hazards and rubble. In this initial study, we focus on evaluating how well the interface supports an operator interacting with a single semi-autonomous robot. This paper describes our efforts to design an appropriate command interface and to measure the usability and effectiveness of that interface.

1. Introduction

We are concerned with the following problem: How can a human effectively control a team of robots in an urban-search-and-rescue (USAR) environment? Such a situation demands a well-designed human-robot interface that respects the cognitive and perceptual strengths and limitations of the human. Consequently, robot interface designers must be cognizant of what constitutes a "well-designed" interface.

This work was motivated by two factors. First, the MITRE robotics team plans to compete in the 2003 RoboCup Rescue Competition. To ensure a favorable outcome we needed to understand the usability of the MITRE human-robot interaction (HRI) design and how it could be improved prior to the competition. Second, we wanted to demonstrate the feasibility of applying human-computer interaction (HCI) techniques to

evaluate HRI. Scholtz (2002) maintains that techniques from HCI can be adapted for use in HRI evaluation as long as they take into account the complex, dynamic, and autonomous nature of robots. We used time-tested HCI evaluation techniques that were developed as a result of decades of empirical research to recommend interface improvements and provide independent validation of MITRE's basic design approach. Researchers in the robotics field often face pressure to perform demonstrations that limits the opportunity to validate their approaches (Harrigan 2002); we hope to begin the process of breaking this trend by providing an example of applying usability evaluation techniques to a robotic interface.

We performed two types of evaluations using HCI techniques, one of which is reported on in this paper. Prior to performing the evaluations, we defined goals and requirements so that we could better understand when improvements need to be made versus when the interface is "usable enough." We compared our findings with the requirements to determine the parts of the interface most in need of improvement.

Section 2 of this paper describes the MITRE robot's hardware and software, whereas section 3 contains a description of the human-robot interface. Section 4 discusses previous work related to the evaluation of HRI, while Section 5 focuses on the evaluation work. Section 6 provides a discussion prior to future work in Section 7.

2. Overview of Robotic Components

2.1. Hardware

The MITRE robot system is based on an ActivMedia Pioneer 2-AT robot platform. All interaction with the robot was mediated through an onboard computer running Redhat Linux with a 400MHz AMD K6-2 and 256MB RAM. The robot's onboard computer maintains an 802.11b wireless link to base station computers. The onboard computer is connected to an onboard microcontroller, which handles all low-level robot system tasks.

The robot's movement is generated through a four-wheel drive skid-steer system. In order to turn, the wheels on one side must be driven to turn more quickly than those on

the other. The set of wheels turning faster travel a longer distance than the slower set, and all four wheels skid sideways to effect the turn. In most wheeled robots, odometry (tracking of distance and direction) is accurate on smooth surfaces as long as the tires do not slip with respect to the supporting surface. In a skid-steer platform, slip cannot be avoided, making odometry inherently inaccurate.

The robot has a variety of additional sensors to provide information about internal state and its surroundings. A ring of Polaroid SONAR range finders returns information about the nearest detectable surface in each of sixteen directions. A SICK laser range finder returns very accurate ranges to diffusely reflective surfaces in a 180 degree swath in front of the robot, taking 361 samples per sweep in half-degree increments. Finally, there is a video camera that can be panned, tilted, and zoomed as necessary. Pyrosensors (heat detectors) were added to the robot after the evaluation took place.

2.2. Software

The software for the robots is written in C++ and makes extensive use of an application programming interface (API) called ARIA (ActivMedia Robotics Interface for Applications) provided by the manufacturer of the robot.

High level behaviors are created by composing a number of lower level actions. ARIA comes equipped with several simple actions already defined. For example, *avoid front* causes the robot to turn only if it perceives an obstacle near its front end; and *constant velocity* causes the robot to drive indefinitely at a given speed. Combining these actions, and others, in a prioritized fashion creates more complex behaviors. For example, composing actions *avoid front* and *constant velocity*, and specifying a higher priority for the former, will cause the robot to drive straight until it perceives a wall in front of it, and then it will turn to avoid the wall. In the MITRE robot, this combination of actions is called a wandering behavior.

Supervisory control of the MITRE robot team is made possible by a combination of autonomous behaviors. Autonomous behaviors that enable exploration without requiring continuous monitoring include:

- Wander (randomly explore a region)
- Go to (x, y), with destination selected by an operator gesture on the map display
- Seek certain types of regions (wander, but approach any region detected with certain defined characteristics)
- Seek with pyrosensor or video motion detection (implementation in progress)
- Return to home base (implementation in progress)

Additionally, MITRE roboticists have developed several behaviors that respond to direct tele-operation command by an operator. These commands specify robot velocity (forward-backward and turning), as well as pan-tilt-zoom controls for the onboard camera.

3. Interface Design

The MITRE command console interface includes several critical features that enable the operator to monitor and control a team of robots executing several behaviors. A single map pane indicates the locations of all robots and displays fused sensor data from all robots in a single world representation. The map also contains multiple layers, which can be displayed individually or simultaneously:

- *Obstacle layer:* Represents the probability that a location is occupied by an obstacle to the robots. Data comes from robot bumpers, wheel encoders, SONAR, LADAR, and operator input.
- *Victim layer:* Represents the probability that a region contains a victim. Data comes from pyrosensor and motion detection algorithms, as well as operator input.
- *Explore/avoid layer:* Guides the operation of autonomous behaviors by representing regions that robots should explore or avoid. Data comes from operator input.

Supervisory control of robot teams requires the operator to be provided with the capability to monitor the status of robots executing autonomous behaviors. For each robot in the team (up to a maximum of three robots), the interface provides a pane that includes a display of status messages, command history, and color video output, as well as a STOP control. Through status messages (and eventually through more noticeable visual cues), the robots can alert the user when they need assistance or confirmation of a possible victim.

A depiction of one of the interface screens can be seen in Figure 1.

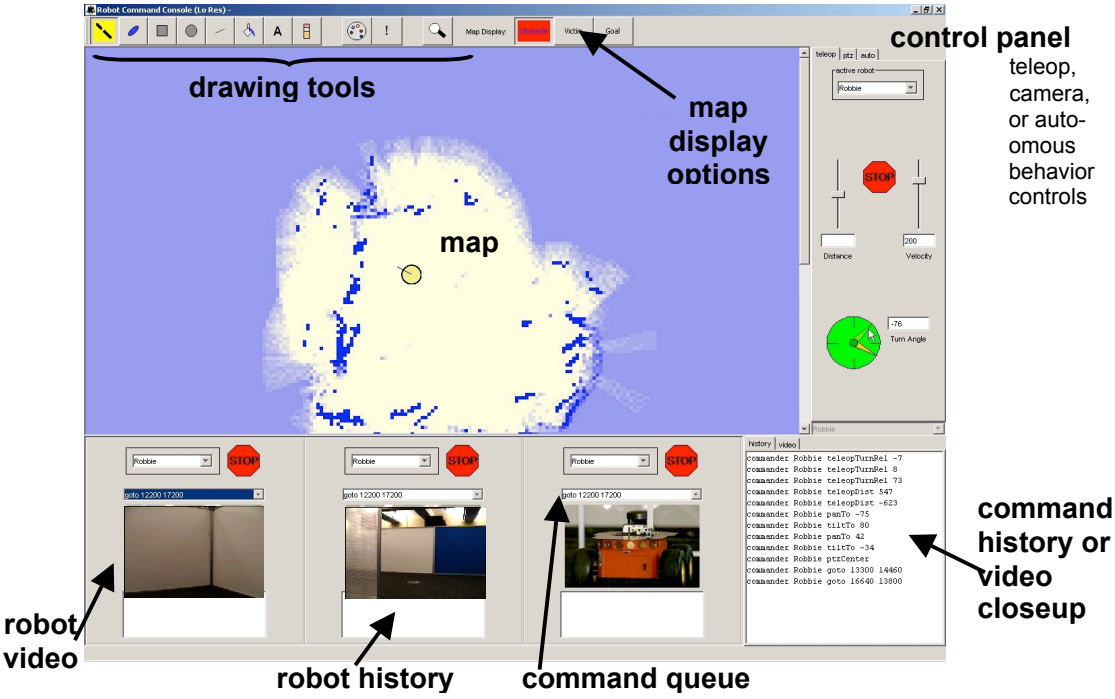

Figure 1: Command Console

A final interface feature that supports supervisory control is the ability to queue autonomous commands. This capability enables the operator to give a sequence of commands to one robot – such as a series of "Goto" waypoints, followed by a "Pyrosensor seek" – then attend to other robots for a longer period of time while the first robot carries out the sequence autonomously.

3.1. Map Display and Data Fusion

Robots often have a very difficult time understanding their environment. A critical part of search and rescue is to be able to communicate where the team has searched and where searchers are now. To help both the robot and the operator understand where a robot is and what kinds of environmental challenges it is facing, the MITRE robots' SONAR sensors and laser range finder data are fused to create a map.

Because of sensor error and a changing environment, there can never be 100% certainty regarding placing obstacles on a map. Instead, the map is generated in a probabilistic fashion so both the operator and the robot can estimate the likelihood that an obstacle exists at a given location. The robot starts by assuming that the whole region around it has about a 50% chance of containing obstacles. Sensors provide the robot with a list of angles and distances indicating the direction and location of nearby obstacles. At each map location corresponding to each distance and angle in the list, the obstacle probability is increased by a small amount. At the several map locations corresponding to a listed angle, but at distances between the robot and the returned distance, the probability is decreased by a small amount. (If there had been an obstacle at a closer distance, the sensors would have reported that closer distance.)

In general, the information from the laser is much more accurate than the SONAR sensors. The laser provides a 180 degree sweep in front of the robot at half degree increments and is accurate to within a few centimeters. However, the laser performs poorly when the obstacle is made of a translucent material or is very reflective, such as a mirror. The SONAR sensors, on the other hand, detect these surfaces when the surfaces are perpendicular to the sensor's view, but SONAR is easily fooled when the sensor's view is not square on to the obstacle. Because each device has different strengths and weaknesses the MITRE design uses a combination of the two to determine the locations of obstacles. In general the laser data is used, but in cases where the laser does not see an obstacle and the SONAR does, the SONAR data is used.

3.2. Communication and Networking Architecture

A variety of messages are sent over a wireless LAN to enable control of the robot team, fusion of robot sensor data, robot localization, and presentation of robot sensor data and status information to the operator. Control and status messages between the command console and the robots are sent as text in a simulated broadcast architecture. Messages are echoed by a communications relay subsystem of the command console to all connected robots via point-to-point TCP. Each robot sends raw or processed output from the color video camera to the command console in the form of a sequence of compressed frames in JPEG format.

This network architecture has several benefits. First, it enables operators to send high-level commands to the entire robot team, which may be interpreted differently by each robot. For example, if a command to explore a region were addressed to "any robot," the robots could negotiate or use a common distributed algorithm to determine which robot would perform the task. Second, it allows for alternative or secondary control consoles, which can relay commands to robots and receive status messages from the robots via the communications subsystem. Finally, even for commands or status messages pertaining to a specific robot, it allows each robot to be aware of the current status of the other robots.

The robot team architecture includes a mapping and localization server, which fuses operator inputs and robot sensor data (other than video) into a single representation of the environment. To support this functionality, the robots send raw sensor data to the map server, and the map server sends fused map data to the robots and the command console, and localization information to the robots. The operator can also modify the world representation via messages from the command console to the map server. This flow of messages can be seen in Figure 2.

4. Related Work for Evaluation of HRI

As can be seen by the previous two sections, the hardware, software, and interface designs developed for this project are unique. The MITRE robotics team needed a way

of determining whether the interface approach would support their goal of being able to efficiently find victims in a search-and-rescue environment.

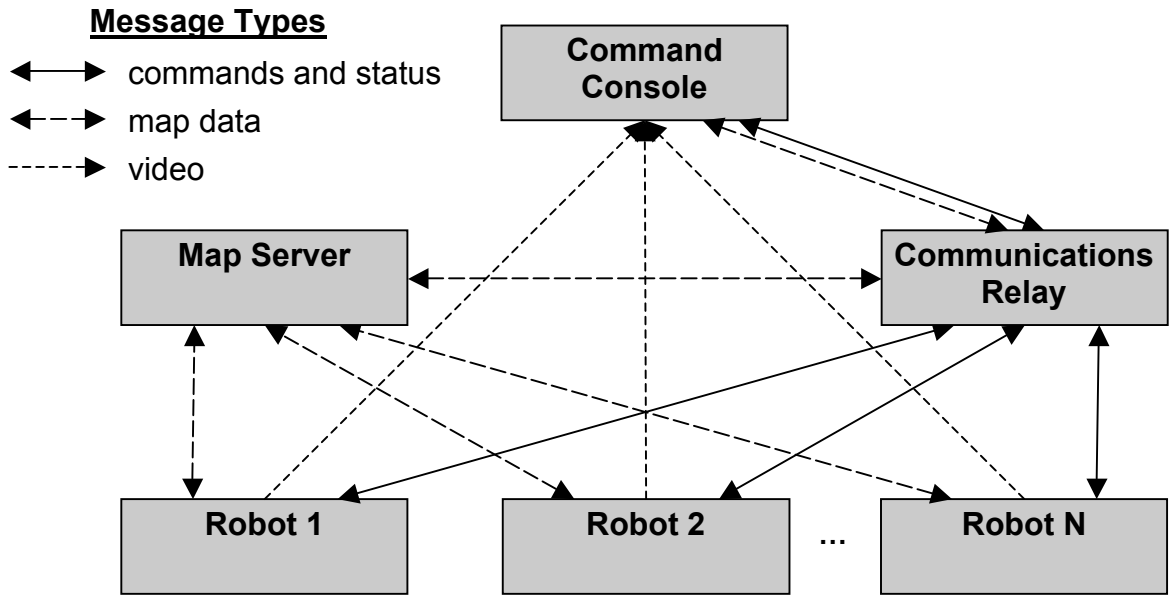

Figure 2: Communication Architecture

Evaluation of HRI can be viewed through the lenses of intelligent systems and HCI. Before any interface (robotic or otherwise) can be evaluated, it is necessary to understand the operators' relevant skills and mental models and develop evaluation criteria with those users in mind. There is no single, generally accepted set of evaluation criteria for HRI.

Some criteria were proposed by Messina et al. (2001) as part of the intelligent systems literature, but they are qualitative criteria that apply to the performance of the robot only, as opposed to the robot and the operator(s) acting as a cooperating system. An example criterion from Messina is: "The system … ought to have the capability to interpret incomplete commands, understand higher level, more abstract commands, and to supplement the given command with additional information that helps to generate more specific plans internally" (Messina et al. 2001). In contrast, an example of a qualitative criterion that addresses human-robot performance might be: "The HRI shall facilitate the

human's use of commands and his or her assessment of whether the robot has correctly interpreted and executed the commands."

Scholtz (2002) proposes six evaluation "issues" for intelligent systems. Scholtz raises these issues "to determine what information the user needs to understand what the intelligent system is doing and when intervention is necessary, and what information is needed to make any intervention as effective as possible" (Scholtz 2002). Examples of Scholtz' issues are "Is the interaction language efficient for both the human and the intelligent system?" and "Are interactions handled efficiently and effectively – both from the user and the system perspective?"

We defined evaluation criteria (stated as goals and requirements) as part of this study. They are informed by both Messina's criteria and Scholtz' issues, but are tailored for the specifics of the tasks expected to be performed by the MITRE robots and their operators.

We performed two types of evaluations whose origins spring from the HCI field: inspection evaluations (where experts examine an interface and compare it with known principles of human-computer interaction) and empirical evaluations (where users are involved, in this case to perform typical tasks in a realistic environment). The inspection evaluation is reported on in this paper.

A common inspection method is heuristic evaluation (Molich and Nielsen 1990), which involves multiple evaluators examining a system to see whether it complies with basic HCI principles (known as heuristics). We adapted principles from Nielsen (1994) to be more relevant to HRI evaluations. We do not know of any previous attempts to use heuristic evaluation for HRI.

5. Inspection Evaluation of Interface

Molich and Nielsen's heuristics consist of general principles that were derived based on looking at usability problem reports from numerous evaluations (Molich and Nielsen 1990). The causes of the problems were identified, clustered into groups, and related

groups were aggregated into supergroups. One primary cause of the problems was identified for each supergroup; these causes evolved into the ten heuristics in the first column of Table 1 (Nielsen 1994).

Table 1. Application of Nielsen's Heuristics to HRI

Nielsen's Heuristics	Nielsen's Heuristics Applied to HRI
Does the program speak the user's language?	Is the robot's information presented in a way that makes sense to human controllers?
Does the program minimize the user's memory load?	Can the human(s) control the robot(s) without having to remember information presented in various parts of the interface?
Is the program consistent?	Is the interface consistent? Is the resulting robot behavior consistent with what humans have been led to believe based on the interface?
Does the program provide feedback?	Does the interface provide feedback?
Does the program have aesthetic integrity (e.g., a simple design)?	Does the interface have a clear and simple design?
Does the program help prevent, and recover from, errors?	Does the interface help prevent, and recover from, errors made by the human or the robot?
Does the program follow real-world conventions?	Does the interface follow real-world conventions, e.g., for how error messages are presented in other applications?
Is the program forgiving; does it allow for reversible actions?	Is the interface forgiving; does it allow for reversible actions on the part of the human or the robot?
Does the program make the repertoire of available actions salient?	Does the interface make it obvious what actions are available at any given point?
Does the program provide shortcuts and accelerators?	Does the interface provide shortcuts and accelerators?

The problem reports that formed the basis of Nielsen's heuristics were obtained from evaluations of single-user computer applications such as word processors. The heuristics are stated at a high-enough level, however, that many are applicable to human-robot interaction. Few people would argue that a "clear and simple design," "consistency," and

"shortcuts" are undesirable for the interface of any computer-based system. The idea behind heuristic evaluation is that the heuristics can and should be tailored for the interface being examined (Nielsen 1993). Despite the differences between traditional single-user applications and robots, our tailoring was fairly minimal. Most tweaks to the heuristics were made to emphasize the semi-independent nature of the robots; e.g., recovery from errors may be needed due to either human or robot errors. Our tailored heuristics can be seen in the second column of Table 1.

An advantage of heuristic evaluations is that they can uncover many potential problems in a relatively short amount of time. Heuristic evaluations can be done without the extensive preparations often required of other evaluation methods, such as finding users willing to act as subjects in usability testing. The power of the method comes from having more than one inspector examine the interface since, while there are overlaps, different inspectors tend to see different problems.

A disadvantage of heuristic evaluations is that they can miss subtle problems that pertain to users' assumptions or specific methods of working that are unknown to evaluators. Thus, we followed the approach recommended by numerous HCI professionals and researchers of first performing a heuristic evaluation and subsequently performing a usability test with representative users engaged in typical tasks. (The results of the usability test are not included in this paper due to space limitations.)

After tailoring the heuristics, two evaluators examined the MITRE interface, looking at every screen and interaction dialog component to see how they compared against the heuristics. Upon completion of their examinations, the evaluators pooled their findings. A summary of these findings can be found in Table 2, which repeats the tailored heuristics in the first column.

Table 2. Summary of Heuristic Evaluation Findings

Nielsen's Heuristics Applied to HRI	Areas of Potential Non-Compliance
Is the robot's information presented in a way that makes sense to human controllers?	• The turn control did not make sense because it was implemented "backwards" (turning it to the right sent the robot to the left).
Can the human(s) control the robot(s) without having to remember information presented in various parts of the interface?	• Labels are missing on some controls. For example, operators must remember the units used in the distance slider.
Is the interface consistent? Is the resulting robot behavior consistent with what humans have been led to believe based on the interface?	• The robot's position is updated more quickly than the obstacle mapping portion of the display, leading to the case where the robot's position can be depicted in the middle an area that is shown as not having been explored yet (except the robot is exploring it). • The turn control is inconsistent with usual conventions (see above).
Does the interface provide feedback?	• Portions of the map update so slowly that feedback regarding the robots' position is not always received in time to prevent bumping or collisions. • Feedback is not quickly given when the user issues a "stop" command, so the user presses the command again.
Does the interface have a clear and simple design?	• The drawing tool palette contains some icons whose meanings are not clear to users. The drawing tools are in "prime" real estate (top left hand corner, where users' eyes first begin scanning the display) yet are not often used. • The design is not easily expandable for more than three robots.
Does the interface help prevent, and recover from, errors made by the human or the robot?	• It is possible to ask for a 'video closeup' with a small video window still open; when the same robot's video is requested this results in an error. • Insufficient feedback results in bumping errors. • Users who misremember the applicable distance units may undershoot or overshoot.

Table 2, concluded. Summary of Heuristic Evaluation Findings

Nielsen's Heuristics Applied to HRI	Areas of Potential Non-Compliance
Does the interface follow real-world conventions, e.g., for how error messages are presented in other applications?	• The turn control violates expectations for direct manipulation.
Does the interface make it obvious what actions are available at any given point?	• No non-compliance noted.
Is the interface forgiving; does it allow for reversible actions on the part of the human or the robot?	• Interface commands resulting in collisions may cause irreversible injury or damage.
Does the interface provide shortcuts and accelerators?	• Setting up the interface requires choosing the robot's name in three places; only one choice should be needed as a default. • The set-up default should automatically bring up the chosen robots' video and scroll the map window to the robots' location(s). • A user should not have to close a small video display to show the same video in the large display area.

6. Discussion and Recommendations

Since the goal of usability evaluations is to find problems, it is easy for the results presentation to sound negative. In fact, several aspects of the MITRE robot HRI design work very well. For example, the interface does not require extensive manipulation to see all the relevant information; thus the memory load on users is not very high. The "STOP" buttons are highly visible and available in all modes of operation. The data fusion implemented in the map overlays is promising.

To improve control of the robots in a search-and-rescue scenario, we recommended correcting the problems identified as non-compliance areas in the heuristic evaluation (Table 2). In particular, we feel the following modifications are critical:

- more timely map updates
- faster and more easily perceived feedback on acknowledgment and execution of commands
- additional context information regarding the robot's position
- more feedback concerning when a robot has made (or, better yet, is about to make) an error

Note that these recommended modifications all pertain to improving an operator's real-time awareness of a robot's position, immediate environment, and activities.

7. Future Work

We plan to perform future HRI evaluations after the aforementioned improvements have been made. Further evaluations will likely take the form of usability tests and may involve one user directing multiple robots simultaneously, and different user groups (e.g., search-and-rescue workers) under both single- and multiple-robot conditions.

We hope the methodologies presented in this paper will encourage other researchers to use HCI techniques to evaluate their HRI.

8. References

1. Harrigan, R. W. (2002). "Is it a wave or is it a particle? The synergy of theory and experimentation" Workshop on *Validation of Public Sector Robotic Systems: Moving from Demos to Experiments,* 2002 IEEE International Conference on Robotics and Automation, Crystal City, Virginia, May 12, 2002.
2. Messina, E., J. Evans and J. Albus. "Evaluating knowledge and representation for intelligent control". In *Proceedings of the 2001 Performance Metrics for Intelligent Systems (PerMIS) Workshop*, in association with IEEE CCA and ISIC, Mexico City, Mexico, Sept. 4, 2001.
3. Molich, R. and J. Nielsen (1990). "Improving a human-computer dialog." *Communications of the ACM*, 33(3): 338-348.
4. Nielsen, J. (1993). *Usability Engineering*. Chestnut Hill, MA: AP Professional Press.

5. Nielsen, J. (1994). “Enhancing the explanatory power of usability heuristics.” In *Proceedings of the CHI 94 Conference on Human Factors in Computing Systems*, Boston, MA, ACM.
6. Scholtz, J. (2002). “Evaluation methods for human-system performance of intelligent systems.” In *Proceedings of the 2002 Performance Metrics for Intelligent Systems (PerMIS) Workshop*.