AN INTELLIGENT SUGGESTION SYSTEM FOR IMPROVED HUMAN-ROBOT INTERACTION

BY

MICHAEL BAKER
B.S. COMPUTER SCIENCE, UNIVERSITY OF LOWELL

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE
IN COMPUTER SCIENCE
UNIVERSITY OF MASSACHUSETTS LOWELL

Signature of Author: Michael Baker Date: 7/12/06

Signature of Thesis Supervisor: ____________________
Holly Yanco

Signatures of Other Thesis Committee Members:

____
Gary Livingston

____
Fred Martin

An Intelligent Suggestion System for Improved Human-Robot Interaction

Master's Thesis

Computer Science Department

University of Massachusetts Lowell

Abstract—In a remote robot system, a user navigates a robot using a graphical interface and a steering control. The graphical interface provides a video display showing the robot's camera view and various indicators showing the robot's status. The interface also provides autonomy modes, which specify how much control over navigation is split or shared between the robot and the human operator. Autonomy modes range from teleoperation (full human control) to fully autonomous (full robot control). To address the problems frequently experienced by users of remote robot systems (e.g. incorrect use of autonomy modes, inability to interpret sensor information, etc), we created an intelligent suggestion system. The suggestion system provides safety, autonomy mode, camera, lights, and battery suggestions. We present data from user tests describing the performance of the system; the suggestions in our system teach users how to operate the robot safely, enforce safety in the driving task, teach users how to understand and trust robot autonomy, teach users how to switch autonomy modes effectively, and improve situation awareness by providing status alerts.

Thesis Committee:

- Prof. Holly Yanco (advisor)
- Prof. Fred Martin
- Prof. Gary Livingston

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Task Description

In the urban search and rescue (USAR) task, an operator navigates a mobile robot through a remote environment that may be unsafe for humans. Using an interface that provides a video display of the robot's camera(s) and information from the robot's distance sensors, the operator must navigate the robot safely and efficiently to find victims who may be trapped and injured. USAR interfaces typically provide "autonomy modes" to aid the user with navigation and reduce the user's cognitive load.

1.2 Problem Description

Previous studies have illuminated a number of difficulties that users experience while operating a robot through a control interface (Section 1.3). In a remote robot task, the interface provides information about the robot and its environment. A user processes this information to gain an understanding of the robot's status and surroundings. We use the term situation awareness (SA) to describe this understanding. A user needs adequate SA to operate the robot safely and efficiently, but existing interface designs do not provide adequate situation awareness. A study of SA showed that users spend 30% of their run time on average solely trying to gain situation awareness, yet still express confusion and demonstrate a lack of SA. Poor situation awareness leads to operator confusion and frustration, and poor task performance [Yanco and Drury 2004].

In addition to the SA problem, we have observed a number of counterproductive user behaviors in our studies of human-robot interaction (HRI) in the USAR task. Prominent among these is poor autonomy mode selection. Despite receiving training on the autonomy modes and using them in a practice session before their test run, many users tend to stay in a direct control autonomy mode. Our robot system provides two direct control autonomy modes, Teleoperation and Safe (Sections 3.5.1 and 3.5.2). A study by Marble et al. found that some users prefer direct control because it requires less understanding or because they have previous teleoperation experience [Marble, Bruemmer and Few 2003]. We have also observed users forget what the modes are and how to use the controls to switch modes.

The direct control modes are not appropriate for all situations. In Teleoperation mode, the robot responds directly to joystick commands and can be driven into obstacles. Therefore, Teloperation mode is generally unsafe and should only be used in limited circumstances. In Safe mode, the robot uses its range sensors to protect itself and the environment by avoiding contact with obstacles. Safe mode is not the most efficient mode, however, when the remote environment is cluttered with obstacles.

In numerous studies, we have observed operators who focus on the video display to the exclusion of all other information on the interface [Baker et al. 2004]. Since the camera view provides a false sense of width and depth, users who navigate by video alone tend to bump the robot into obstacles, especially when going through doorways. (Marble et al. discuss this problem and how it also causes users to control the robot inefficiently [Marble, Bruemmer and Few 2003].) In addition to driving the robot unsafely, users who stay glued to the video may miss important indicators on other parts of the interface.

Since it is faster to turn the robot than to pan the camera, some users “pan” the robot to gain situation awareness [Yanco and Drury 2004]. Since our robot has a long wheelbase and skid steering, this is generally unsafe, especially for users who only pay attention to the video display. We have recorded many hits by users who turned our robot in place while not watching for obstacles near the robot. Other typical unsafe driving practices include backing the robot without looking and accidentally driving with the camera panned to the side. Finally, we have seen user driving styles range from very aggressive to very conservative. Aggressive drivers tend to operate the robot recklessly and cause damage to the test arena.

1.3 Motivation

This section provides a few examples of the types of errors, confusion and frustration that users experience while controlling a remote robot through an interface. This motivated us to create a suggestion system to assist users with the kinds of problems frequently encountered in remote robot tasks.

In a study of situation awareness in the USAR task, a test subject drove the robot into a tight area with walls blocking the front and sides of the robot. Since it was difficult for the operator to move the robot in this area, he spent several minutes trying to free the robot. This situation took time away from performing the task and caused the operator to become frustrated. Although the interface provides an autonomous Escape mode in which the robot drives itself out of tight spaces, the operator did not use it [Yanco and Drury 2004].

In a similar study, an operator became frustrated when the robot would not move forward in Safe mode. In Safe mode, the robot uses its range sensors to prevent collisions with obstacles in the environment. Although the interface indicated an obstruction, the operator relied on the video display for obstacle information and the video display showed a clear path. The operator switched the robot to direct control (Teleoperation) mode and drove the robot through a Plexiglas panel. If this has been a real search and rescue mission, this mistake could have had catastrophic consequences. A secondary collapse of a damaged structure could damage and trap the robot, and cause further harm to any victims inside [Yanco, Drury and Scholtz 2004].

Another operator in this study moved the robot's video camera to identify a mock victim and then checked the thermal camera display to see if the victim was alive. After the operator finished identifying the victim, the operator switched the navigation mode and proceeded to leave the area, but the operator did not realize that the robot's video camera remained turned to the left. When the operator switched back to Safe mode, the operator became very confused, began bumping the robot into walls, and even drove the robot out of the test arena. The operator finished the run without realizing that the video camera was panned off-center [Yanco, Drury and Scholtz 2004].

1.4 Suggestion System

The suggestion system described in this thesis addresses the problems described in Sections 1.2 and 1.3 by providing safety, autonomy mode, camera, lights and battery suggestions. Safety suggestions emphasize safety in the task, discourage unsafe driving habits, and teach the user how to operate the robot safely. Autonomy mode suggestions encourage effective autonomy mode switching by teaching the user when it makes sense to switch modes. Eventually, users learn how to make this judgment for themselves. Through accepting autonomy mode suggestions, users learn to understand and trust robot autonomy. In this regard, the suggestion system makes automatic adjustable autonomy possible since it can automate mode suggestions that a user always takes.

Users frequently forget about interface features and controls. Suggestions compensate for this by reminding operators about useful interface features and providing automatic services. For example, the suggestion system includes lights suggestions that detect the ambient light level at the robot's location and offer to turn the robot's lights on or off. A user who had forgotten about the robot's lights or how to turn them on would find this suggestion very helpful. We describe the suggestion system as "training wheels" for novice users and reminders for experienced users. Users may need some encouragement to experiment with a useful, but unfamiliar interface feature and suggestions provide this benefit as well (Section 4.1.5).

Finally, all of the suggestions improve situation awareness by alerting the user to the presence of obstacles near the robot and other important task information. Since the suggestion system delivers warnings and reminders when something becomes critical or important, the user is relieved from having to monitor and interpret status indicators on the interface. Since the user may be functioning under stress and therefore prone to mistakes, lowering the mental load provides a significant benefit. In short, the suggestion system helps users accomplish the USAR task (or any remote robot task) more safely, efficiently and effectively.

1.5 Contributions of this Work

This thesis presents an intelligent suggestion system designed to improve human-robot interaction (HRI) in a remote robot task. To the best of our knowledge, intelligent suggestions have not been incorporated into any other interfaces used to perform remote robot tasks. This research provides extensive qualitative results from a 16-subject study of the suggestion system in a simulated USAR task. The research provides the results of applying machine learning to the suggestion acceptance data for the purposes of analyzing and improving the suggestion system. Also, in this work we provide principles for designing suggestions and ideas for future work using this approach.

We characterize our suggestions as intelligent because they attempt to infer a situation for the purpose of providing automatic (with user consent) assistance. Furthermore, our suggestions behave intelligently by appearing and disappearing in a way that allows users to notice them, but not become annoyed by them.

The suggestion system is also sensitive to human factors. Suggestions are easy to notice and understand. Suggestions are easy to accept or dismiss. A user can respond to or ignore suggestions while continuing to work on another task. Since a user can only respond to a single suggestion at any instant, we devised a priority and preemption scheme to present only the best suggestion when multiple suggestions are possible. If a suggestion becomes repetitive or annoying, the user can disable it dynamically.

2 Related Work

2.1 Intelligent Interface Agents

An interface agent communicates with a person directly through the user interface. It can observe a user's manipulations of the interface, sense what the user sees on the screen, and invoke interface commands. This type of agent can add graphics or animation to the interface, use speech input or output, and even use sensors and effectors that sense and act directly with the real world. Lieberman and Selker use the term "intelligent interface agent" to describe an interface agent that shows intelligence in user interactions. According to this definition, our suggestion system qualifies as an intelligent interface agent [Lieberman and Selker 2003].

A proactive interface agent (like our suggestion system) anticipates the user's needs and may even take the initiative to interrupt the user when an important event occurs. The agent can alert the user to opportunities and information that would otherwise be missed. Interruption must be used carefully or the user will object to it.

An advisory agent only teaches or suggests. Since the person does all of the work, the person retains responsibility for the work. Agents that teach, tutor, suggest, or document can be even more productive than assistant agents. Advisory agents have been verified experimentally to improve user performance [Lieberman and Selker 2003].

Lieberman and Selker describe several characteristics that interface agents may or should exhibit. For an agent to be useful, the user must trust the agent. A trust relationship is built through successful interactions over time. An agent's authority can be increased gradually as the user gains confidence in the agent's abilities. Trust and confidence in an agent is developed when agents provide feedback and allow the user to influence its operation.

Every time a user responds to a suggestion, either implicitly or explicitly, this data is used to improve the suggestion system using machine learning (Chapter 6). In this way, every user response is a vote of confidence that influences how suggestions may behave in the future (after incorporating the learned result back into the suggestion system).

2.2 Interfaces that Give Advice

In [Lieberman 2001], Lieberman proposes building human-computer interaction around the notion of advice as the primary means of communication. The computer can give and take advice, and thereby act as both advisor and advisee. When the computer is an advisor, top-level control and critical decisions stay with the human. The user can benefit from the agent's work while retaining responsibility because it is up to the user to accept, reject or ignore the advice. In 1959, John McCarthy proposed advice as a stepping stone toward a fully autonomous artificial intelligence [McCarthy 1959].

The goal of advice is improvement of the advisee's behavior. Unlike commands, advice is flexible and supports a collaborative relationship. Advice need not be exact or sequential. It may not solve a whole problem or specify an action. It is conversational and ongoing. The advisor and advisee can operate with or without advice.

Systems need to consider a user's limitations when they are providing advice for that advice to be useful to the user. Systems that assume a user has infinite attention, complete knowledge of the system, and infinite patience quickly become annoying. Our suggestions are sensitive to a user's limitations because they are easy to ignore or dismiss while performing another task. They relieve the user from having to monitor interface indicators and compensate for a user's lack of system knowledge by providing reminders about interface features.

Letizia [Lieberman 2001] is an example of an interface that gives advice. Letizia assists a user browsing the World Wide Web and treats browsing as a cooperative activity between the user and agent. As the user browses Web pages, Letizia uses a keyword-frequency algorithm to extract topics and searches breadth-first out from the current link. Then the search results are filtered through a profile created from the user's browsing behavior. Letizia's recommendations are displayed continuously to the right of the user's browser window.

Since Letizia compiles an interest profile through watching a user's actions, its advice is personalized. Letizia runs continuously when the user is idle. Its advice is ongoing, always visible and reacting immediately to changes in the user's browsing behavior. Finally, Letizia's advice is noncoercive since the user can choose to ignore it. Letizia does not accept advice, but it could be made to support feedback and allow the user to advise and modify the profile it forms. Lieberman thinks speech would be useful for accepting advice since the user is occupied by the browsing process.

As previously mentioned in Section 2.1, suggestions could be personalized if we consider a suggestion system that uses machine learning to learn how to make suggestions for a class of users (Chapter 6). Like Letizia, our suggestion system is noncoercive and uses speech. It does not support explicit feedback, but the collected acceptance data can be viewed as feedback since we use it to improve the suggestion system.

2.3 Appropriate Behavior of Suggestions

The most well-known user interface agent is Microsoft's Office Assistant, popularly known as "Clippy". Bundled with Microsoft Office since 1997, Clippy has attracted widespread negative opinion. In [Swartz 2003], Swartz explores these negative user responses.

The design of the Office Assistant was partially inspired by Computers As Social Actors (CASA) theory, which states that users instinctively treat computers like people. Since CASA showed that computers behave like social actors, it was believed that an anthropomorphic character could make interaction with a program more natural.

The Microsoft Office Assistant is a partially autonomous, intelligent user interface agent that fulfills three major roles. As a proactive help system, it suggests ways to finish a task better or easier. The most infamous example of this is the proactive letter-writing help feature. If the user repeatedly refuses help—even if the user hides the agent—it continues to reappear whenever the user types a salutation.

Other proactive help features are triggered by user behaviors. For example, typing a line in all uppercase triggers a "Headings" tip, and repeated clicking in the margin triggers a "light bulb" tip that explains how to enter text there. If the Assistant is visible and the user clicks on it, the agent will suggest helpful tips based on what the user is working with.

The Office Assistant incorporates an Answer Wizard feature that uses basic Bayesian inference to guess a users' goal given a particular help query. Users can ask questions in relatively natural language and the Office Assistant provides a list of help topics that it thinks would be useful.

The Office Assistant acts as the locus of agenthood. For example, dialog boxes appear as voice bubbles whose text is "spoken" by the Assistant. The Assistant displays animations for commands like printing, saving, and sending an email, which makes it seem like the Assistant is responsible for those actions. Finally, the Assistant performs social animations like blinking when the user is idle. As of Office 2000, it is possible to turn off the Office Assistant altogether.

According to CASA theory, people (consciously or unconsciously) treat the Assistant like a person. Therefore, human rules apply to interactions with the Office Assistant. The Assistant breaks these rules by asking the same questions repeatedly, staring at the user, and monitoring the user's work. The Assistant ignores social conventions by persistently displaying its proactive letter-writing help feature. Dismissing this disturbance multiple times does not cause the Assistant to learn from its mistake. This kind of behavior would not be tolerated in a human assistant.

Like the Office Assistant, the suggestion system provides proactive help, but it does not use dialog boxes that must be dealt with explicitly, which necessarily interrupts and may annoy the user. By contrast, suggestions in our system are easy to accept or ignore without stopping to clear a dialog box from the screen. Suggestions behave according to several timing parameters that we tune to control how persistent and/or repetitive they may appear (Section 4.2.2). Also, in our suggestion system it is possible to disable unwanted suggestions dynamically.

2.3.1 Qualitative Study

Swartz performed a qualitative study of the Office Assistant in the form of in-depth, open-ended interviews and determined that one's *cognitive label* profoundly influences one's perception of the agent. For example, people who labeled the Assistant character as a "productivity tool" found its distracting animations counter-productive and annoying. People who labeled the character as an "office diversion" welcomed the distracting animations.

Next, Swartz designed a quantitative experiment to explore the effect of labels and appearance on user responses to interface agents. The task scenario involved buying items from a simulated e-commerce website. The independent variable *label* ("fun" or "useful") corresponded with how the online character was introduced to the user. The independent variable *character* (human or cartoon) corresponded with what the online character looked like.

Swartz found that labels not only influenced users' attitudes towards the system, but also their behavior. That is, what the participant bought and how long the participant stayed on the website. Swartz also found that consistency between label and appearance is an important consideration. For example, since "fun" is a person characteristic, it should label a person character.

In his second quantitative experiment, Swartz asked what would be different when no character accompanied the help agent and how would a "fun" agent influence

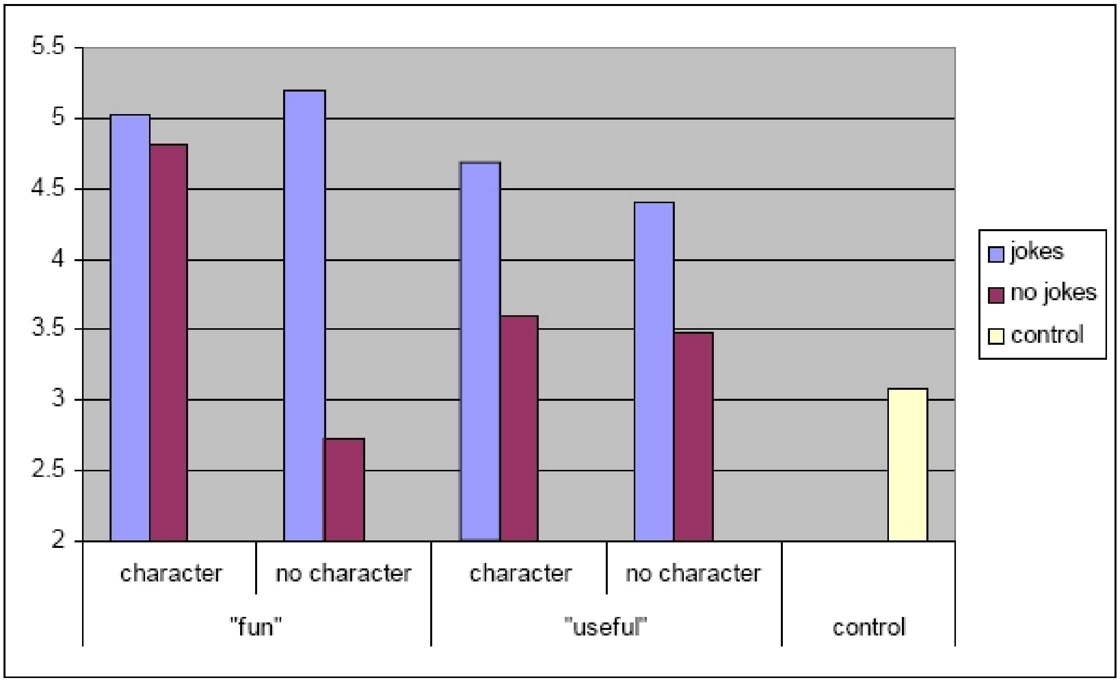


Figure 2-1. Positive Rating of the Website (from [Swartz 2003]).

the interaction. The first independent variable was *label* as in the first experiment. The second independent variable was *presence of character* (character or no character), and the third independent variable was *joking behavior* (jokes or no jokes).

As in the first experiment, labels and appearance significantly affected people's experience of the system. The "useful" label caused users to buy more items. The "fun" label caused users to expect a fun appearance and fun behavior. Agents with characters and agents with joking behavior elicited the most positive responses (see Figure 2-1). The no-agent condition received a low rating even though the agent added no features to the website.

Swartz concludes with some design guidelines (Table 2-1) and the lesson that designing effective user interface agents is hard because many factors including task, situation, behavior, appearance and label influence users' responses.

• Consider the agents' task in its social element (for example, beginners may want to rely on more experienced users for help and guidance—how can one facilitate this?).
• Agents should obey human rules of etiquette as much as possible (if one doesn't like a person who disobeys these rules, one will especially dislike a computer agent that disobeys them!).
• Explore ways to use the agent to teach users skills to make them more self-sufficient (thus allowing users to retain a sense of control over the program).
• Carefully introduce the agent so as to realistically showcase its best features—and be sure that the appearance and behavior are consistent with that introduction (for example, if one calls the agent "fun," there should be something fun about it!).
• Study whether it is beneficial to use characters or agents at all (in some cases, a less anthropomorphic agent, or no agent at all, may provide the same benefits with less costs).

Table 2-1. Guidelines for User Interface Agent Design (from [Swartz 2003]).

Since the suggestion system does not use a cognitive label or anthropomorphic character, it avoids the problem of how these attributes influence user attitudes. Regarding the guidelines in Table 2-1, the suggestion system is sensitive to human etiquette (Section 1.5) by maximizing noticeability and understandability while minimizing the disturbance to the user. Suggestions make the user self-sufficient by teaching the user how to operate the robot safely and effectively, even without suggestions.

2.4 Interpreting User Responses to Suggestions

[Xiao, Stasko and Catrambone 2004] studied the effect of interface agent competence on user performance and perception. The agent helped users to learn and use a text editor by responding to spoken questions and making proactive suggestions using a synthesized voice. According to the authors, the challenge of interface agent design is finding the delicate balance between providing help and interrupting users in an annoying way.

The authors cite competence as a fundamental issue regarding the quality of interface agents. Competence refers to the general quality, appropriateness, and timeliness of help provided by an interface agent. Perception of competence is subjective, but competence here is an objective term meaning the accuracy of an agent's answers and the relevance of an agent's suggestion to the user's situation.

Since current agents are far from competent, Xiao et al. think it is important to examine user reactions to less-than-competent agents. They also wanted to examine how different styles of assistance affect user performance and impression. Specifically, the authors wanted to explore the following questions about interface agent systems:

- How will degradation of interface agents' competence affect user performance and perceived usefulness of the agents?
- Will the competence of an interface agent and the performance of a user affect the likeability of the agent?
- Will user preferences, especially preference of assistance style, have an effect on user performance and subjective assessment? Given alternative choices, will people use an agent at all?

Xiao et al. built a simple text editor where the user has to type a specific control key combination for each editing command. The text editor included an interface agent (Section 2.1) that could answer questions and provide proactive suggestions. An experimenter introduces the experiment and guides the participant on how to use the interface, which includes a microphone and speaker. A second experimenter, the *wizard*, monitors the experiment from another room. The wizard controls the quality of replies and suggestions made by the interface agents.

Since people feel more comfortable talking to an interface agent with a life-like appearance, a realistic, animated, 3-D female was chosen. She blinked, moved her head occasionally, and moved her mouth in synchronicity with the synthesized voice. The user could not control the female agent, who had no emotion or personality traits, and whose responses always followed a preset form.

Proactive advice from the wizard was offered only when it was logically plausible for a computer to infer the situation. For example, if the user does something in many steps, the agent could suggest a shortcut command. If the user begins a predictable sequence of commands, such as a copy followed by a paste, the agent could tell the user the key combination for paste when the user copies some text.

Four agent competencies were tested; three conditions provided 100% relevant suggestions and one condition provided 50% relevant suggestions. User performance was measured as time and efficiency of task execution, and how many times the user received reactive and proactive help from the agent. A Likert scale questionnaire rated the user's subjective experience with the agent. Qualitative data consisted of an open-ended interview and observations made during the experiment.

Analysis of the data showed that the perceived utility of an agent varies with the types of errors it makes. People seem to be more forgiving of proactive errors than reactive errors because they do not expect proactive suggestions to always be right. As long as the agent did not make repeated errors, people would blame themselves rather than the agent. One participant suggested telling the user beforehand what types of questions the agent understands.

Regarding the proactive suggestions, one participant said that the shortcut suggestions were too late to be helpful since he had already done something the long way. Some participants found the proactive suggestions distracting, especially if they came while the user was in the middle of doing something. A few participants did not notice the timeliness and relevance of the suggestions at first. One user said the suggestions were annoying, but later realized their usefulness.

To summarize, the authors suggest the following about applications with interface agents:

- Avoid repeating errors to greatly improve people's perceptions of the quality of reactive agents.
- People's expectations and judgements of the usefulness of proactive agents are relatively low.
- Proactive suggestions are more readily accepted if they can be immediately applied and are easy to understand.

The authors further concluded that a person's subjective view of an agent has little to do with its utility. Most participants had positive reactions toward the proactive suggestions, but some who had received competent suggestions described the agent's proactive behavior as interfering and annoying. One user preferred to control everything and not bother with the suggestions, even if it would help in the long term.

The authors discovered, however, that participants' reactions to the agent related more to the perceived quality of the agent's face and voice. Reactions to the face and voice correlated with a user's rating of the agent as intrusive, friendly, intelligent, or annoying. Half of the participants reacted favorably to the face and voice, while half did not.

Concluding the interview, participants were asked their impressions of the Microsoft Office Assistant, "Clippy". The vast majority of the responses were negative, describing Clippy as annoying, hard to get rid of, and not looking real. Some participants said it gets in the way because its text dialog blocks the screen and needs attention, where the voice agent can be easily ignored. Some participants said they liked the Assistant after changing its appearance to a cat or a dog.

The authors list some design factors for applications with an embodied interface agent:

- The subjective appeal of a proactive interface agent has more to do with people's individual differences than with the competence of the agent.
- People's subjective experience with an interface agent relates to features of the agent, such as face and voice.
- It is unlikely that one can design a representation of an interface agent that would suit all users.

Careful inspection of the performance data with respect to the agent's competence condition revealed that users who could choose their preferred assistance style—the help screen, or the agent, or both—performed significantly better. Users who favored the help screen exclusively cited easy, direct, efficient access to the information. One user did not feel comfortable talking to a computer. Users who favored the agent exclusively made similar arguments the other way, saying that it was easier and more natural to ask the agent questions, and that switching to and from the help screen is inefficient.

Users who used both help screen and agent had mixed reactions. One participant used the agent a lot, but only after the agent spoke first and demonstrated its usefulness. This same user later voiced a negative opinion about Clippy and its tendency to jump in and sell itself. The other participants used the help screen or the agent depending on their individual perceptions of how quick or easy it is to obtain an answer by either means. One participant claimed to be a visual person who prefers to see things written down.

Finally, with respect to interaction styles, the authors concluded the following:

- It is crucial to match a user's preferred interaction style to the way help is provided.
- Provide alternative forms of help rather than just an agent acting as an assistant.
- It is important to build people's confidence by illustrating the utility of interface agents.

Several of the findings of the Xiao et al. study are consistent with the results of our study (chapter 5). In particular, we also found that different users have varying preferred interaction styles. As in the Xiao et al. study, some subjects in our study did not feel comfortable talking to a computer and some preferred to retain more control in the task. In both studies, repeated errors lowered user confidence in suggestions while high relevance and understandability increased user confidence.

Both the text editor and the suggestion system mitigate the problem of interrupting the user by using voice controls. As mentioned in Section 2.3, we avoid user reactions to an anthropomorphic character altogether by not including one in our design.

2.5 Mixed Initiative User Interfaces

The mixed-initiative approach to interface design assumes that intelligent services and users may collaborate efficiently to achieve the user's goals. Interface agents are a type of automated service that sense a user's activity and take automated actions. Problems with the use of agents in interfaces include poor guessing about the user's goals and needs, inadequate consideration of the costs and benefits of automated action, poor timing of action, etc.

The LookOut project, a mixed-initiative user interface, overlays automated scheduling services on Microsoft Outlook. LookOut parses the text in the body and subject of an email message in focus and attempts to identify the date and time of an event implied by the sender. Then the system brings up Outlook's calendar and the user's appointment book, filling in the relevant fields of an appointment record. Then the user can edit the system's guesses and save the result.

LookOut uses a probabilistic classification system to assign a probability that a user would like to view the calendar or schedule an appointment. The classification system is trained by watching the user working with email.

Although the system may be uncertain about a user's goals, it must decide whether to invoke an agent's services or not. The decision to act or not is directed by *expected utility* from decision theory [Horvitz 1999]. Depending on the inferred probability and the expected utility assessment, the system either waits for the user to manually invoke LookOut, starts a dialog with the user to offer a service, or goes ahead and provides its service. LookOut has behaviors for handling response delays and reacting to signs that service is being declined; it notices when the user is too busy to respond and gets out of the way.

LookOut provides multiple interaction modalities. In manual operation, the system will only take action if a user clicks on the LookOut icon. When the system displays an alert on the system tray icon, the user can place the mouse cursor over the system tray icon to see the intended action. In basic automated-assistance mode, LookOut launches and populates fields in Outlook windows. It may also use dialog boxes to request information from users when appropriate. In social-agent mode, one of the animated characters from the Microsoft Agent package issues queries and announces results. In hands-free mode, the system uses text-to-speech and automated speech recognition for a more natural interaction.

Service that comes too soon is distracting while late service is less valuable. To determine when a user is most ready for service, LookOut uses a model of attention that considers the temporal pattern of a user's focus of attention. The default model of attention in LookOut is based on user studies, but the system can learn a custom model by watching a user interact with email. Finally, LookOut supports continued learning where the user specifies a training schedule for the system to incrementally refine the probabilistic user model and time-based attention model.

The current suggestion system assumes novice users and therefore includes an implicit user model. By applying machine learning to the acceptance data for a class of users (chapter 6), the implicit user model can be adapted for users having different expectations and levels of experience. In the robot navigation task, modeling the user's attention and goal is much less necessary since we may assume the user is focused on the interface while operating the robot and we know the user's goal is to operate the robot safely and efficiently while searching for victims.

2.6 Mutual Initiative in the INEEL Robot System

As mentioned in Section 3.5, our robot system is a variant of the INEEL robot control architecture [Bruemmer, Dudenhoeffer and Marble 2002]. This section discusses the concept of mutual initiative in this design and implications for the suggestion system.

2.6.1 Dynamic Autonomy

Since the robot is often better at local navigation than the distant operator, INEEL's control modes allow the robot to infer when a human's commands need to be supplemented or overridden. INEEL's mode system also acknowledges the robot's limited perception, understanding, and decision-making ability by allowing the human to override and configure the robot's initiative through autonomy mode selection. Section 2.6.2 discusses how autonomy modes permit the human and robot to operate as a team.

The INEEL robot control architecture supports four levels of mutual initiative on the autonomy scale. In Teleoperation mode, the user has full control over the robot. In Safe mode, the robot takes the initiative to protect itself and the environment by refusing to execute motion commands that would cause a collision. In Shared mode, the robot navigates itself through open space while avoiding obstacles and the operator's joystick commands are blended with the robot's initiative to guide the robot in a general direction. In Full Autonomy mode, the robot does high-level tasks such as target tracking and area mapping without user intervention [Bruemmer, Dudenhoeffer and Marble 2002].

2.6.2 Concept of Teaming

Rather than view the robot as a tool, the INEEL robot control architecture is based on the human and robot working together as a team. Since the human and robot interact as peers, either team member can take initiative in the task. The goal of teaming is to augment the human's physical and mental abilities for the purpose of increasing task performance. Task performance measures include time to complete the task, ability to complete the task and overall efficiency.

Frequently the robot is able to make better judgments about its environment than humans. Consequently, the INEEL system allows the robot to modify or override dangerous user commands in its Safe and Shared autonomy modes. Robot perception is limited, however, and unable to understand obstacles like half-open doors and empty boxes. In these situations, the human can override the robot's self-protective behavior by switching to an autonomy mode (Teleoperation) that gives the human more control. In general, the user must be able to predict and understand robot behavior within each level of robot initiative. The developers of the INEEL system believe this understanding is acquired through (possibly extensive) training [Bruemmer, Marble and Dudenhoeffer 2002].

Robot-initiated mode switching is a possibility in the team concept, but the user must develop expectations about when and why the robot wants to switch modes. Essential to learning these expectations is feedback since failing to inform the user could lead to mode confusion, loss of situation awareness, loss of confidence in the system, and degraded performance. Feedback could be visual or audible and should include the mode change and an indication of the reason for the change.

Our suggestion system addresses the issues of training and feedback in a human-robot system with switchable autonomy modes. The suggestion system teaches the user how to switch autonomy modes effectively without suggestions, but also provides semi-automatic (by accepting suggestions) and possibly automatic mode switching (by learning which mode switches are likely to be accepted). Suggestions also provide feedback, communicated through the suggestion icon and text, about why a mode switch makes sense in a particular situation.

2.6.3 Experiment and Results

In a usability test of the INEEL system, participants were asked to search a building using a robot to locate and identify targets as quickly and safely as possible. INEEL ran their experiment with eleven participants; seven having little or no experience with remotely controlled robots, and four having extensive experience with teleoperation or master-slave systems. Before the search task, participants were given twenty minutes to become familiar with the interface and robot behavior by performing simple tasks using each level of autonomy [Marble, Bruemmer and Few 2003].

Inexperienced participants having no previous expectations of how the system might work were the best at learning how to interact with the robot. In particular, two participants having no previous experience with remote systems were the best at adapting to Shared mode. Participants having the most experience with teleoperated systems reported the most frustration in Shared mode. These participants attempted to give constant commands to the robot in Shared mode as if the robot were in Teleoperation mode. These participants experienced difficulty when the robot contradicted the user's commands while steering away from obstacles. Users could always complete the search task, but user feedback indicated confusion about robot autonomy.

The usability test revealed a wide disparity between users. Some users preferred to use and learn robot autonomy while others preferred to teleoperate the robot because it did not require learning or understanding robot autonomy. Participants who performed the best in Teleoperation and Safe mode were not the best at adapting to Shared mode. The test indicated that previous experience with remote systems made learning interaction with an autonomous system more difficult. The authors felt that more training is needed before an experiment to increase the participants' understanding of how robot initiative works and what triggers it [Marble, Bruemmer and Few 2003].

The suggestion system can reduce the need for extensive training by providing "just-in-time" training in the context of a real or simulated task. As mentioned in section 2.6.2, mode suggestions teach users about robot autonomy by pointing out the situations and conditions under which a mode switch makes sense. Furthermore, the suggestion system may provide useful refresher or reinforcement training for users who may be out of practice.

2.7 Collaborative Control

2.7.1 System Description

With collaborative control, the human functions as a (limited) resource for the robot, providing information and processing just like other system modules. The robot can ask the human questions to obtain assistance with perception and cognition, which allows the human to compensate for limitations of autonomy. In the collaborative control model, human and robot collaborate to solve problems.

Since the human can only attend to one robot at a time, the system arbitrates among requests to select one for presentation. This allows human attention to be directed where it is most needed, in terms of safety, priority, etc. Collaborative control incorporates a user model (stereotype) for dialogue management so it can accommodate users with varied backgrounds and capabilities.

The current collaborative control system has thirty messages: human-to-robot (commands, queries, responses) and robot-to-human messages (information statements, queries). The robot can ask two types of queries: s*afeguard* queries concerning safety issues and *task* queries about task-specific functions, such as “Can I drive through?” A robot query is described by a number of attributes used to select which queries will be asked. Some depend on the operator (required response accuracy, required expertise), and some do not (expiration, priority, etc).

2.7.2 User Study

HRI in collaborative control (CC) was investigated in an 8-subject Contextual Inquiry (CI) user study designed to understand users’ learning strategies, observe users’ reactions to robot dialogue, and evaluate system usability. At the start of a test, the subject was familiarized with CI and the CC system was configured with the user's stereotype. Then the user was given a remote driving task; novices were asked to explore an area and experts were asked to drive the robot through a narrow doorway and then explore the area. While the user worked, written notes were taken to record user behavior. To elicit information, questions such as "Why are you doing that?" were repeatedly asked. From time to time, robot questions were triggered by injecting artificial sensor data into the system (e.g., high temperature) or by simulating emergency conditions.

2.7.3 Results and Discussion

A small indicator on the interface blinks when the robot has a question. To respond, the user must notice the indicator, click to retrieve the question and input a response. Because users must stop what they are doing to respond, some users avoided answering questions for long periods of time. Other users immediately stopped what they were doing whenever a question appeared, viewing the robot's questions as more important than the driving task.

Occasionally, the robot would ask many questions in a short period of time. This happened when the robot was operating in a cluttered area, or when multiple safeguarding behaviors were triggered simultaneously. One user became annoyed, stated he was tired of answering questions and began ignoring the questions altogether. Some users commented that the robot asked the same question over and over. The battery voltage fluctuated significantly during high-torque, skid-steering maneuvers, causing the voltage to fall below the safe threshold. When this happened, the power question would repeat until the battery level stabilized. Some users became annoyed by this and started ignoring the questions, reasoning that it's probably the same question again.

Sometimes users were busy with the task and failed to notice the robot's questions. Since some questions have a time limit, occasionally questions would go completely unnoticed. The flashing indicator was not always sufficient to attract the user's attention. Adding more "passive" signals may not improve the recognition rate, but active signals could hurt performance since users have cognitive limitations and sometimes make mistakes when interrupted. When some users realized that questions were "disappearing", the experimenter reminded the user about the dialogue log. A user would look once, but not consult the log again, even when other questions disappeared.

During the assessment of how useful the questions were, one user said they needed to provide more detail. The developers assumed that the image accompanying a particular question made it clear what the robot was asking, but this was not true for at least one user. More context could improve the question such as annotating the image with text or graphically highlighting features of interest in the image. One user said the questions seemed very technical while another user said the temperature information would be better if displayed visually. Perhaps questions should be phrased differently to different users.

Dialogue affects how users perceive and relate to the robot, sometimes causing users to personify the robot and construct inaccurate cognitive models. One user felt the robot only asked questions when it urgently needed help, but some questions are triggered when the robot has difficulty assessing the situation. Another user felt like the robot was asking the same question again and again to cause the user to give in. The user's perception of stubbornness by the robot was actually due to the robot's conservative safeguards. One user thought the robot had priority over questions and decided how to use the human when, in reality, the robot only retains control of the dialogue when it is unable to function.

In general, users were only told that the robot might ask questions occasionally. As a result, some users made incorrect assumptions about the questions or how the robot works. Only users who asked for clarification were given detailed explanations. One user incorrectly assumed that questions were triggered by user commands only and became confused when a command did not clear old questions and interpreted this as the robot disobeying him. Some amount of training is needed to ensure that users understand how the questions work.

2.7.4 Summary of Results

- Different users may respond quite differently to the same question.
- Users may grow weary of answering questions.
- A question without adequate detail is hard to answer.
- Dialogue can make users personify the robot.
- Indicating the urgency of questions is important.

2.7.5 Collaborative Control Versus Suggestion System

In collaborative control, the robot solicits help from the human operator. By contrast, the suggestion system provides unsolicited help to the human operator. A major difference between systems is the method by which users respond to queries or suggestions. The suggestion system has a distinct advantage in that users do not have to stop what they are doing to respond.

There are, however, a number of similarities between the user studies and results of both systems that warrant discussion. Both user studies involved a remote driving and exploration task, but we did not question our subjects during the task. Rather, subjects were encouraged to "think aloud" during the task and the subjects' comments were tape recorded for analysis. There was an important similarity between methods, however, in that we did not provide a detailed explanation of the suggestion system or individual suggestions unless a subject asked for one. In our study, this was a conscious decision because we wanted to gauge the clarity of suggestions without special instruction or training. In both studies, this omission caused users to develop their own mental models, which were sometimes incorrect.

Also, in both studies, responses varied widely across users, showing once again that different users interact very differently with the same robot system [Marble, Bruemmer and Few 2003]. Both studies found that users would both miss and ignore questions (suggestions), but missing suggestions is much less of a problem in our system since suggestions are placed over the video display. Although CC provides a mechanism (log file) for looking at missed questions, users did not use it.

Although two users in our study recommended stacking preempted suggestions beside the video display, this feature is unnecessary because a missed suggestion will reappear if its triggering condition stays true. The wisdom of our approach is that users do not have to look at or think about multiple suggestions, which we think is unproductive since a user can only assimilate and respond to a single suggestion at one time. We believe that allowing multiple simultaneous suggestions could overwhelm the user and cause the user to ignore suggestions altogether.

Repetitious questions (suggestions) caused by fluctuating sensor values is a problem in both systems. Understandability of questions (suggestions) is another issue shared by both systems. Both of these problems can cause users to ignore questions (suggestions). Finally, both systems caused users to personify the robot. In our study, a few subjects interpreted suggestions as things the robot wanted or needed. One subject wondered aloud whether certain actions would bother the robot and another subject "obeyed" suggestions to keep the robot happy.

3 Robot System

This chapter describes our robot system, which consists of a hardware platform (Section 3.1), a graphical user interface (Section 3.2), user controls (Section 3.3), a communications link between the robot and the user interface (Section 3.4), and several autonomy and operational modes (Section 3.5).

3.1 Hardware Platform

Our robot, shown in Figure 3-1, is an iRobot ATRV-Jr robot. This robot has a symmetric 26-sonar ring with eight sonars on the left and right sides of the robot, and five sonars on the front and rear sides. The sonars are angled to detect obstacles in every direction around the robot. At the front of the robot is a SICK laser range finder, which provides high-resolution obstacle detection covering 180 degrees, but only at a single height just below the robot's front bumper. This platform also provides a number of odometry sensors including distance, position, tilt and direction. There are bump sensors in the rubber front and rear bumpers.

Figure 3-1. iRobot ATRV-Jr robot.

To support our research, we upgraded the internal computer from a Pentium III to a Pentium IV and installed a multiple frame grabber, which can capture video from up to eight camera sources. Then we installed Canon VC-C4 cameras facing forward and backward on the robot. These cameras have a maximum pan angle of 50 degrees to the left or right, a maximum tilt angle of 90 degrees up and 30 degrees down, and 16 zoom positions. They have a 47.5-degree field of view (65 degrees with a wide angle lens adapter) and a serial interface for software control.

To aid victim detection and identification we installed a Raytheon NIGHTDRIVER infrared camera and a carbon dioxide monitor, both facing forward on the robot. We mounted fluorescent lights on all sides of the robot, which can be turned on and off through a serial interface to a custom PIC-based controller board.

3.2 Graphical User Interface

Figure 3-2 shows the user interface for our robot system (described in [Casey et al. 2005], [Baker et al. 2004], and [Baker and Yanco 2004]). The map display on the left side of the interface is generated using pmap, open source laser-based mapping software [Howard 2006]. The large video display in the center of the interface shows the front camera view. If the user pushes the joystick forward, the robot will move into this view. The smaller video display at the upper right hand corner of the main video display shows the view behind the robot. The image in this display is mirrored to look like the rear view mirror in a car.

Just below the rear view mirror display is a carbon dioxide meter. When the carbon dioxide level rises above the threshold mark, it indicates the presence of human life.

Below the main video display is a range indicator that we call the "sonar ring." The sonar ring has a 3D perspective to give the user a realistic sense of the robot, to appear as if the user were physically sitting behind the robot while driving it. Since there are 26 sonars on the robot, some of the segments of the sonar ring represent multiple sonars. A sonar segment can have one of three colors. Gray indicates no obstacle, yellow indicates an obstacle, and red indicates a close obstacle (red ≤ .5m, yellow ≤ 1.5m, gray > 1.5m). Our scheme is intentionally simple to spare the user from having to remember the meanings of several colors or paying attention to raw sonar values.

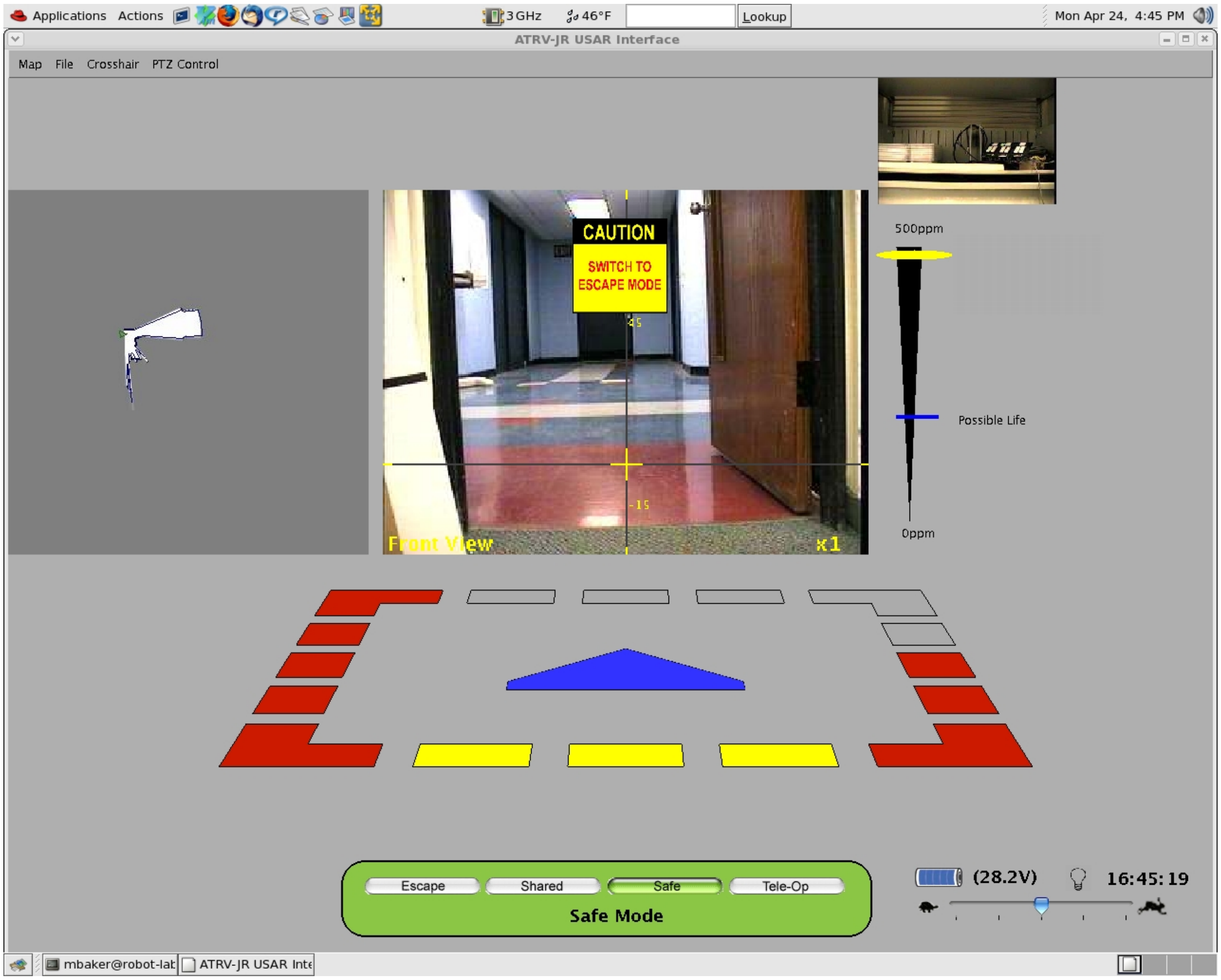


Figure 3-2. UMass Lowell Robotics Lab USAR interface.

The sonar ring rotates whenever the user pans the camera to stay aligned with the camera view. When the user pans the camera to the left, the sonar ring rotates to the right and vice versa. This is the same effect as turning your head to the left while keeping your body straight. Referring to Figure 3-2, the sonar ring indicates a close obstacle at the left front corner of the robot. If the user wants to have a look at this obstacle, the user can pan the camera to the left until the corner of the front left sonar segment lines up with the center of the video display. If the user wanted to have a look at the obstacle by the left rear corner, the user could switch camera views (Section 3.5.1) and pan the camera to the right.

Below the sonar ring is the autonomy mode indicator, which supports four modes in our system. Each mode is color coded to make is easy for the user to recognize which mode the robot is in. For example, switching to Escape mode would cause the Escape mode button and the autonomy mode indicator to turn blue, and change the text description from "Safe Mode" to "Escape Mode." To the right of the autonomy mode indicator is a speed indicator, a battery level indicator, a lights status icon, and a clock display. The user can switch autonomy modes and toggle the lights on and off by button press on the keyboard. The speed control is a dial on the joystick (Figure 3-3).

3.3 User Controls

The joystick control for our system is shown in Figure 3-3. It is a generic USB joystick with a "hat" on top of it (labeled "PAN / TILT" in Figure 3-3) that functions like a miniature joystick. The user manipulates the hat control using a thumb to pan and tilt the camera corresponding to the view in the main video display. The home button below the hat control returns the camera to its center position looking straight ahead. The zoom-in and zoom-out buttons control the camera's zoom position and the brake button turns the robot's brake on and off. As a safety feature, there is a trigger on the joystick that the user must hold down while operating the robot. This prevents the user from accidentally brushing the joystick and moving the robot unintentionally. The user controls also consist of a standard keyboard, which is used to switch autonomy modes (Section 3.5) and toggle the lights on and off.

Figure 3-3. Joystick control.

3.4 Communication

The communications link between the interface and the robot is wireless 802.11a. Since the wireless connection fades in and out occasionally, we use UDP instead of TCP sockets for all communication. TCP "backs off" the connection and sends less data when the connection times out, which would happen whenever the wireless connection drops out. This behavior is unsuitable for robot control. UDP is not as reliable as TCP, however, since UDP does not guarantee data delivery and can drop packets. To maintain consistency between the interface and the robot, we continually broadcast state information back and forth at varying frequencies depending upon the importance of the information. State information includes range sensor values, lights, brake and ADR mode status, autonomy mode, battery voltage, camera positions, and the CO_2 sensor value.

3.5 Autonomy Modes

Our autonomy mode system was adapted from the INEEL robot control architecture [Bruemmer, Dudenhoeffer and Marble 2002]. On the surface, our mode system resembles the INEEL system, but we have completely rewritten the underlying code, making changes to existing modes and adding modes of our own (Table 3-1). In addition, we have created a flexible behavior system that allows adding behaviors to existing modes and building new modes from blended combinations of behaviors [Casey et al. 2005].

Autonomy Mode	Description
ADR Mode	Automatic direction reversal; from the user's perspective, the robot spins 180 degrees in place.
Teleoperation Mode	The user controls the robot directly.
Safe Mode	The robot uses its sensors to prevent collisions with obstacles.
Shared Mode	The robot drives itself, but the user can influence the robot's direction with joystick commands.
Escape Mode	The robot drives itself out of tight spaces.
Goal Mode	The robot drives itself toward a goal.
Joystick-Goal Mode	The user's joystick command represents a goal to move the robot in that direction; the user's goal is modulated by obstacles in the robot's proximity.
FLIR-Goal Mode	The robot drives to a heat source autonomously.
Governor Mode	The robot prevents collisions with obstacles when driving autonomously.

Table 3-1. Robot Autonomy Modes (Summarized).

3.5.1 Teleoperation Mode

In Teleoperation mode, the user controls the robot directly using a joystick. The robot has no autonomy in this mode and since it responds directly to joystick commands, the user can drive the robot into obstacles. Therefore, this is the least safe mode of operation in terms of the risk it poses to the environment and the robot. Furthermore, this mode requires the most interaction from the user. This keeps the user engaged in the task, but it can also fatigue the user and cause the user to ignore everything in the interface except the video display. There are limited circumstances where Teleoperation mode is useful, however. For example, the user may want to use the robot to push a cardboard box, but in a higher autonomy mode the robot would see the box as an obstacle and avoid it.

3.5.2 Safe Mode

As in Teleoperation mode, in Safe mode the user controls the robot directly, but the robot protects itself when a joystick command would cause the robot to hit something. In its original implementation, the robot would simply stop when it came close to an obstacle in its path. This version of Safe mode caused a lot of user frustration, just as it did to the operator who drove through the Plexiglas panel (as described in Section 1.3) because the user sees no obstacles in the video, yet the robot seemingly refuses to obey joystick commands for no reason.

To mitigate this frustration, we changed the stopping behavior of the robot to a gradual slowing behavior as the robot nears an obstacle. Also, to show the user why the robot will not move, we flash the sonar segment (Figure 3-2) that senses an obstacle blocking the robot. We have received positive comments about the feedback from the controls and the interface, verifying our hypothesis that frustration would be less if the user understood why the robot was resisting against a pushed joystick. Our joystick is not a haptic device with actual force feedback, but the gradual slowing of the robot creates an impression of resistance to the user who is watching the video display.

3.5.3 Shared Mode

In Shared mode, the robot drives itself while avoiding obstacles. Also, the robot accepts user joystick commands and combines them with its own drive commands, which allows the user to influence the robot's direction. Although the robot drives itself in Shared mode from the INEEL system, it drives rather randomly in the direction of most open space. We came to the conclusion that this behavior did not make sense because the robot does not navigate toward a goal, but drives aimlessly. In the USAR task, we want to explore the search space methodically and quickly to locate victims and Shared more is not the most effective way to do this. As a result, we created Joystick-Goal mode (Section 3.5.7) to replace Shared mode.

3.5.4 ADR Mode

Since our robot has a rear camera and a symmetric sonar ring, we have created an interface feature called ADR mode, which stands for Automatic Direction Reversal. When the user invokes ADR by pressing the “CAM VIEW” button on our joystick, shown in Figure 3-3, three things happen. First, the video displays are swapped. For example, if the main video display showed the front camera view (as in Figure 3-2) then, after the switch to ADR mode, the main video display would show the rear camera view and the rear view mirror display would show the front camera view. Secondly, the sonar ring flips over to stay consistent with the direction the robot is facing after the switch. Finally, the joystick commands are inverted to complete the effect or illusion that the robot has spun around 180 degrees in place.

After the ADR switch, from the user’s perspective the “front” of the robot is now the “back” of the robot and vice versa. Using ADR, the user can designate which direction is forwards, regardless of which way the robot is physically facing. In the middle of the sonar ring on the interface (Figure 3-2) is a blue triangle that points to the physical front of the robot. The user needs to know this because our robot is not perfectly symmetric; the SICK laser, FLIR camera, and CO_2 monitor are on the front side of the robot. ADR virtually eliminates the need to drive the robot backwards, which is risky and difficult. If the user drives the robot down a narrow hallway, for instance, the user can use ADR to reverse direction and drive out normally.

3.5.5 Escape Mode

In Escape mode, the robot drives itself out of a tight space autonomously. While the robot senses that it is “stuck,” it will drive itself until it is no longer stuck and then stop. If the robot can not make any progress in one direction, it will try the opposite direction. Originally, this mode chose to escape in the direction of most open space, but users have commented that it should go in the direction that the user last commanded with the joystick. Some savvy users have used Escape mode to navigate the robot through narrow doorways and down narrow hallways.

3.5.6 Goal Mode

In Goal mode, the robot drives toward a goal. The default behavior for Goal mode is the Joystick-Goal behavior (Section 3.5.6.1), which treats the user's joystick command as a vector pointed toward the user's "goal." Since the user's cognition and perception are needed to manage the higher task, this makes more sense than letting the robot wander. We also use Goal mode as a placeholder for other goal-tracking behaviors. Goals in USAR correspond to victim cues such as heat, sound, or gas. We have implemented a FLIR tracking behavior in our system that causes the robot to drive itself to a heat source (Section 3.5.6.2).

3.5.6.1 Joystick-Goal Mode

This mode, created to replace Shared mode, is based on a shared-control teleoperation algorithm described in [Crandall and Goodrich 2002]. Our implementation is slightly different, but the behavior is the same. When the user issues a joystick command, it gets treated as a vector that represents the user's goal. Informally, the goal is, "I want the robot to go this way." The current sonar readings represent vectors that point to obstacles and open space around the robot. The algorithm combines the user's goal with the robot's goal (avoid obstacles) using vector summation to compute the actual speed and direction the robot will go.

Straightforward vector summation would not work because the sonar vectors outnumber the joystick vector 26 to 1. Instead, the joystick vector is given full weight and the sonar vectors are weighted from 0-1 according to their angular distance from the joystick vector. Sonar vectors whose angles are closer to the joystick vector are weighted more heavily. The result is a vector that is the user's goal *modulated* by obstacles near the robot. A depiction of the algorithm is shown in Figure 3-4. To the user, it feels like the robot is guiding itself around obstacles.

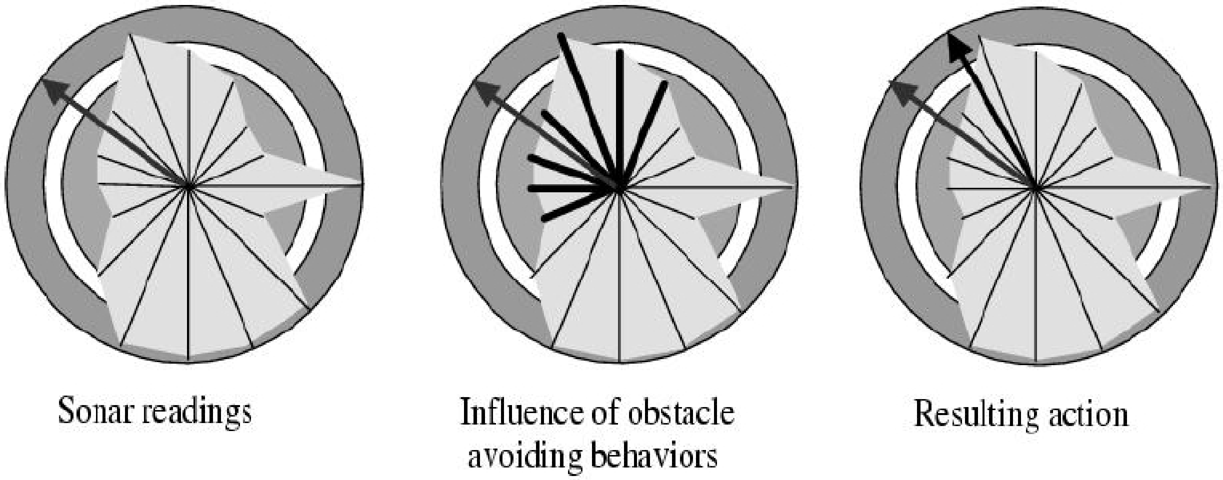


Figure 3-4. A graphical depiction of Shared Control teleoperation (from [Crandall and Goodrich 2002]).

In this algorithm, it is possible for the sonar vectors to *cancel out* or sum to zero. If this happens, the user could drive the robot into an obstacle. The Crandall and Goodrich implementation of Shared Control teleoperation uses a safe-guarding behavior which rejects commands that would take the robot out of its safe region. The safe region is the area within the polygon (Figure 3-4) created by connecting the endpoints of the sonar vectors. In our implementation, we filter the drive command generated by the algorithm through Governor mode (Section 3.5.6.3) to achieve the same safe-guarding effect.

3.5.6.2 FLIR-Goal Mode

A vision algorithm running on the robot detects heat "blobs" in the FLIR camera image and overlays them on the front camera video display using red pixels. A user can invoke the FLIR tracking behavior manually through a menu on the interface or automatically by accepting the Track FLIR suggestion (Section 4.2). The FLIR tracking behavior causes the robot to drive itself toward the largest heat blob in the image until the robot is within a few feet of the heat source with the blob centered in the video image. If the robot "loses" the heat source while tracking, it will steer back to the left or right depending on which direction it was steering when the heat blob disappeared.

3.5.6.3 Governor Mode

Safe mode modifies the user's joystick command to slow the robot in the presence of obstacles. Governor mode performs the same function on the robot's autonomous drive command with one difference. Governor mode slows the robot to a minimum speed where Safe mode eventually stops the robot altogether. This makes sense because the user can back away from obstacles in Safe mode, but when the robot is driving itself it needs to continue to make progress toward the goal. Since Joystick-Goal mode is based on vector summation, the robot can stall if the vectors cancel out. Governor mode keeps the robot moving slowly until the vectors have influence again.

4 Suggestion System

Figure 4-1 shows the current interface design. A suggestion appears as a large icon over the video display, but user tests have shown that it is possible to ignore suggestions while driving the robot (Section 5.4). In the figure, it is clear that Escape Mode is being suggested and it is also clear that it is being suggested for safety reasons.

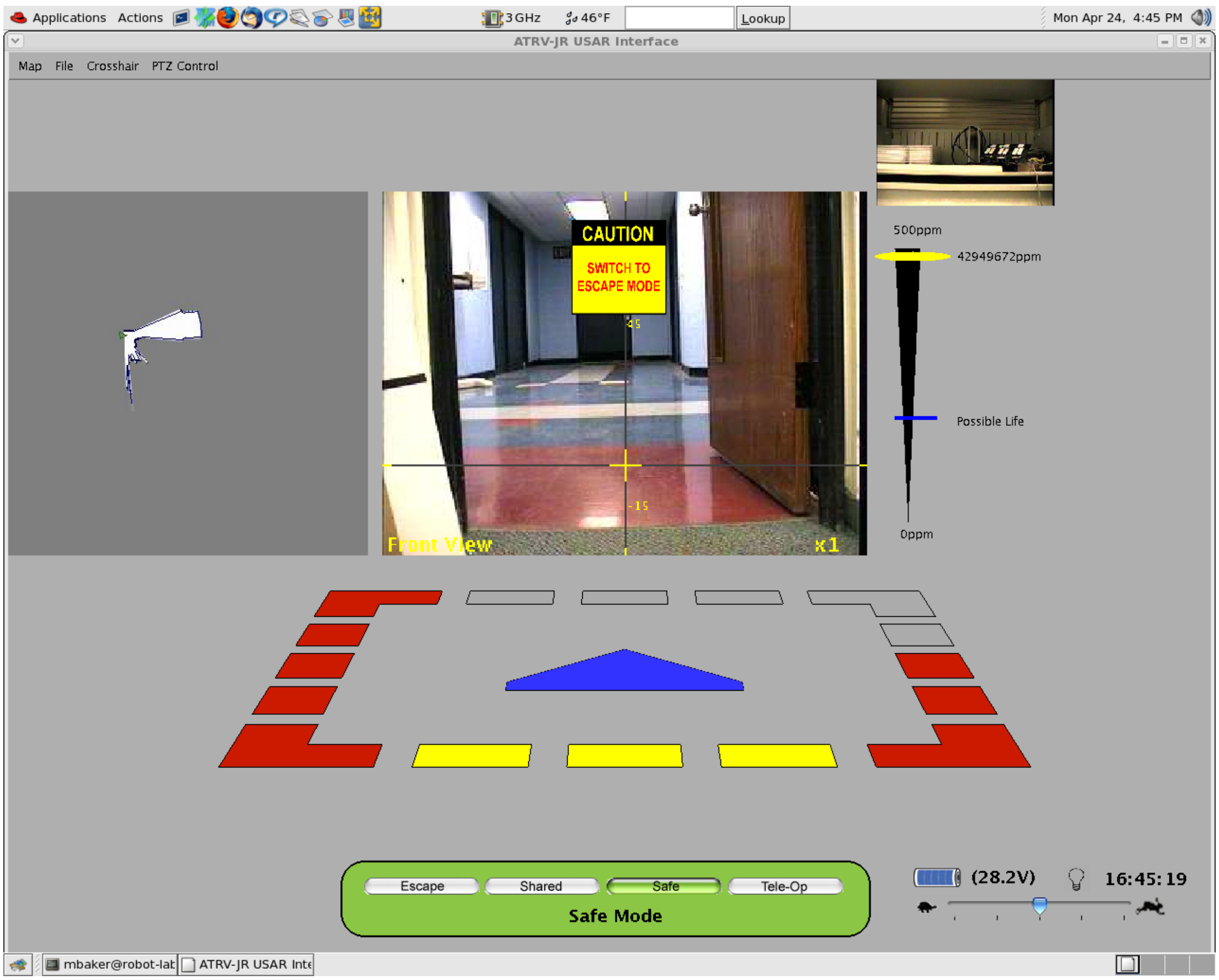


Figure 4-1. The current suggestion system uses large suggestion icons appearing over the main video display.

4.1 Implemented Suggestions

The current suggestion system includes the thirteen suggestions summarized in Table 4-1.

Suggestion	Description
Teleop-to-Safe Mode	Suggested when the user demonstrates unsafe driving in Teleoperation mode by coming near or bumping into an obstacle.
Reduce Speed	Suggested after the uses refuses the Teleop-to-Safe Mode suggestion and the speed is set high.
Turning In Place	Suggested when the user turns the robot in place.
Restore Cameras	Suggested after a Turning in Place maneuver; restores the cameras' PTZ settings.
Escape Mode	Suggested when the robot is "stuck" in a tight space.
Escape-to-Safe Mode	Suggested once the robot has freed itself in Escape mode.
Center Camera	Suggested when the user drives the robot with the camera off-center.
Switch Camera View	Suggested when the user drives the robot backwards.
Turn Lights On	Suggested when the robot's environment is dark and the lights are off.
Turn Lights Off	Suggested when the robot's environment is bright and the lights are on.
Battery Low	Suggested when battery power drops to 50% or less.
Battery Very Low	Suggested when battery power drops to 25% or less.
Track FLIR	Suggested when the FLIR camera detects a heat source; invokes the autonomous FLIR tracking behavior.

Table 4-1. Implemented Suggestions.

4.1.1 Teleop-to-Safe Mode and Reduce Speed

The rationale underlying this suggestion is that it is almost always safer to be in Safe mode than Teleoperation mode. In its original implementation, this suggestion would trigger whenever the user switched to Teleoperation mode. In user tests, users would become frustrated when the robot would not move in Safe mode, so the user would switch to Teleoperation mode. The Teleop-to-Safe mode suggestion would immediately suggest returning to the mode the user had just left out of frustration. This would happen frequently, which pointed out the need for a smarter suggestion.

Now, when a user switches to Teleoperation mode, the suggestion system puts the user "on notice." If the user bumps into or comes close to an obstacle, the Teleop-to-Safe mode suggestion triggers. If the user ignores or dismisses the suggestion, it will fire again the next time the robot bumps. This suggestion keeps track of safety violations in Teleoperation mode and although the current implementation is not very aggressive, it could be made to display a more urgent warning with each violation. In USAR and other safety-critical tasks, it might even make sense to switch to Safe mode automatically and have the suggestion notify the user of the switch.

The Reduce Speed suggestion is triggered when the user declines the Teleop-to-Safe mode suggestion and the speed control has a fast setting. Although users receive instruction on the need to operate the robot safely in a USAR environment, driving styles in user testing vary widely across users from overly conservative to reckless. This suggestion reinforces the idea of safety to users who drive the robot recklessly. It can also be viewed as an attempt by the robot to protect itself since bumping at a slower speed is less likely to cause damage.

4.1.2 Turning in Place and Restore Cameras

The Turning in Place suggestion was inspired by repeatedly observing users hit obstacles while turning the robot in place. Our large robot platform has skid steering, which causes the robot to rotate in place like a tank when the user leans the joystick to the left or right. Turning in place is a particularly risky maneuver since the robot is rectangular; most bumping of our robot platform results from it. To rotate the robot safely, the user has to be aware of obstacles on both sides of the robot. Novice users of our system do not have an appreciation for this problem, and since most users concentrate on the video display, they're not even aware of the obstacles they're directing the robot into.

To address this problem, the Turning in Place suggestion offers to aim the cameras to the sides so the user can watch for obstacles while turning. It would be tedious and time consuming for the user to aim the cameras each time the user wanted to rotate the robot, but Turning in Place provides this service automatically. If the user turns the robot counterclockwise, the Turning in Place suggestion will aim the front camera to the left and the rear camera to the right, and vice versa when the user turns the robot clockwise. Initially, this suggestion used reasonable default camera positions, but now the suggestion aims the cameras at the nearest obstacle on either side of the robot. Once the user has finished turning the robot, the Restore Cameras suggestion is triggered automatically.

When the user accepts a Turning in Place suggestion, it records the camera positions before moving the cameras so the previous camera positions can be restored automatically after the turning-in-place maneuver. The Turning in Place suggestion would be less useful if the user had to restore the positions of both cameras manually.

4.1.3 Escape Mode and Escape-to-Safe Mode

The Escape mode suggestion triggers when the robot senses that it is "stuck," which means there are close obstacles on three or more sides of the robot. Taking this suggestion causes the robot to switch to Escape mode and free itself. Once free, the robot stops and the Escape-to-Safe mode suggestion is triggered. Since Escape mode is a fully autonomous mode, the user must switch to another mode to resume controlling the robot and Safe mode is the default mode of operation in our system.

4.1.4 Center Camera

Users who have inadvertently driven the robot with the camera aimed sideways motivated this suggestion. The addition of a camera crosshair and reference lines to the video display has decreased the likelihood of this occurring, but it still continues to happen in user tests. Rather than triggering immediately upon moving the joystick, this suggestion delays briefly to give the user a chance to realize the camera is not centered. This suggestion also teaches inexperienced users, as evidenced in user tests, that driving with the camera off-center is a bad idea.

It could be the case that the user is driving with the camera panned off-center intentionally. For example, we have seen users follow along a wall with the camera aimed at the wall to avoid scraping it. Since the suggestion system can not possibly know this, it will offer the Center Camera suggestion, but it is easy enough for the user to ignore or dismiss the suggestion. Whether the suggestion is declined explicitly or it times out and expires on its own, it will not reappear for some time. If a suggestion were to reappear immediately after being declined, it would surely annoy the user.

4.1.5 Switch Camera View

The suggestion system detects when the user drives the robot backwards and then offers this suggestion, which switches the robot into ADR mode. Like the Center Camera suggestion, this suggestion delays to make sure the robot is really being driven backwards. Since backing up the robot is risky, we want to discourage backing up and encourage using ADR mode instead. Since ADR is an unfamiliar concept to users, this suggestion provides the incentive to experiment with ADR. Both the Center Camera and Switch Camera View suggestions have a strong teaching effect. Users typically encounter these suggestions a few times and then modify their behavior by always homing the camera before moving the robot and using ADR instead of backing up. These suggestions provide the benefit of teaching good robot driving practices.

4.1.6 Lights Suggestions

A light sensor on the robot triggers the Lights On and Lights Off suggestions. Since the analog light sensor value fluctuates, the suggestion has to make sure it is dark (or bright) in the environment to prevent the suggestion from flickering on and off with the changing sensor value. The suggestion uses separate thresholds for dark and bright, and waits for the light sensor value to stay above or below the threshold value continuously for some time before triggering. We do not want to leave the lights on when the robot is operating in a bright environment because USAR test arenas often have Plexiglas panels and mirrors in them. The glare of the lights on these surfaces and the auto gain feature of our cameras could cause the video to appear dark.

4.1.7 Battery Suggestions

The battery suggestions are the only suggestions that do not have an automatic action associated with them. In that sense, the battery suggestions are more like warnings than suggestions. Like the lights suggestions, the battery suggestions make sure the batteries are truly low by waiting for the battery voltage to stay below a threshold continuously for some period of time. Hard driving causes the battery voltage to dip significantly so this measure is necessary to prevent false battery suggestions. A battery suggestion tells the user that the robot should be returned to the starting position so it can be recharged. There is no conflict between battery suggestions because the Battery Very Low suggestion has a higher precedence in the suggestion system (Section 4.2.1).

The battery suggestions are useful, but they could be made smarter. For instance, the battery suggestions do not take into account how far the robot has traveled and hence do not really know how much battery power is needed for the return trip. Since robots operate in human hazardous environments, having enough battery power for the return trip is important. 50% is a reasonable default, but the robot may have traveled downhill and need more than that. In principle, we could use the robot's position within its dynamically generated map to determine the robot's distance from the starting position and keep track of the slope of the ground using the robot's pitch sensor. We could even have the robot drive itself back to the start position autonomously as the outcome of taking a battery suggestion.

Also, the battery suggestions follow a fixed schedule of appearing, disappearing and reappearing. They do not become more urgent over time. As the battery situation becomes more critical, the suggestion should display more frequently and forcefully to emphasize a sense of urgency. The notion of "progressive urgency" is not currently part of the suggestion system or user interface.

4.1.8 Track FLIR

A vision algorithm detects color blobs in the FLIR image corresponding to heat sources, which triggers the Track FLIR suggestion. Taking the suggestion causes the robot to navigate autonomously toward the heat source. When the robot reaches the heat source it stops. Suggestions of this type can be implemented for various environmental cues including sound and gas sources. Even if the robot lacks the sensing capability to track a source, alerting the user to its presence is useful. We did not simulate a heat source during user testing so this suggestion is not included in the results of chapters 5 and 6.

4.2 Implementation Details

The suggestion system is written in C++. A suggestion is an object and the suggestions are organized in an inheritance hierarchy. Since most of the implementation of a suggestion belongs to an abstract parent class, it is easy to add new suggestions to the system. Creating a new suggestion consists of specifying a priority (Section 4.2.1) and a few timing parameters that determine a suggestion's behavior (Section 4.2.2), and supplying a method that returns "true" whenever the suggestion's triggering condition is valid. Optionally, the developer can define the method that executes as the action of a suggestion when the user accepts it. In the current system, only the battery suggestions do not have actions associated with them.

4.2.1 Priority Scheme

An early version of the suggestion system allowed one mode suggestion and multiple non-mode suggestions to activate simultaneously. This design overwhelmed the users who tested it since the user can only assimilate and respond to one suggestion at a time. The current suggestion system allows only one suggestion to be active at any given time. Since it is possible for two or more triggering conditions for different suggestions to be true simultaneously, to achieve this design the suggestion system must arbitrate among the candidate suggestions and present the best one to the user.

The suggestion system described in [Baker and Yanco 2004] handled arbitration using complicated if-then-else logic. The current suggestion system uses a priority scheme, shown in Table 4-2, which is based on risk to safe navigation. Under this scheme, when multiple suggestions are possible, the highest priority suggestion is

Suggestion	Priority	Rationale
Escape Mode	0	High safety risk
Escape-to-Safe	0	Follow-on to Escape Mode
Teleop-to-Safe	1	High safety risk
Reduce Speed	1	Follow-on to Teleop-to-Safe
Turning In Place	2	High safety risk
Restore Cameras	2	Follow-on to Turning In Place
Turn Lights On	3	Medium safety risk
Turn Lights Off	3	Medium safety risk
Switch Camera View	4	Medium safety risk
Center Camera	4	Medium safety risk
Track FLIR	5	Task priority
Battery Low	6	Task priority
Battery Very Low	6	Task priority

Table 4-2. Suggestion Priorities. A lower number indicates a higher priority.

presented to the user. Also, a suggestion having higher priority can preempt (replace) an active suggestion having lower priority.

For example, referring to Table 4-2, the Escape mode suggestion has the highest priority because its triggering condition—the robot is stuck—represents an immediate risk to the environment and the robot. Since we want to enforce switching back to Safe mode after the robot frees itself in Escape mode, the Escape-to-Safe mode suggestion has the same priority. The Teleop-to-Safe mode suggestion has the next highest priority because, although the user has already committed an unsafe act by bumping the robot, it is safer for the robot to drive itself if it also happens to be stuck. Reduce Speed has the same priority as Teleop-to-Safe because it is the "follow-on" suggestion to Teleop-to-Safe, as previously described.

Turning In Place has a lower priority than Teleop-to-Safe so the system will suggest Teleop-to-Safe if the robot bumps while being turned in Teleoperation mode. Since Restore Cameras is the follow-on to Turning In Place, it inherits its priority from Turning in Place. We give the two lights suggestions priority over the camera suggestions because, if the video can't be seen, it doesn't matter where the camera is pointed. The risk to navigation represented by the lights and camera suggestions is less specific than the higher priority suggestions involving movement and obstacles. Finally, the Track FLIR and battery suggestions represent no risk to navigation, but they do represent task priorities. Track FLIR has priority over the battery suggestions because we may be willing to risk losing the robot to locate a victim.

4.2.2 Timing Parameters

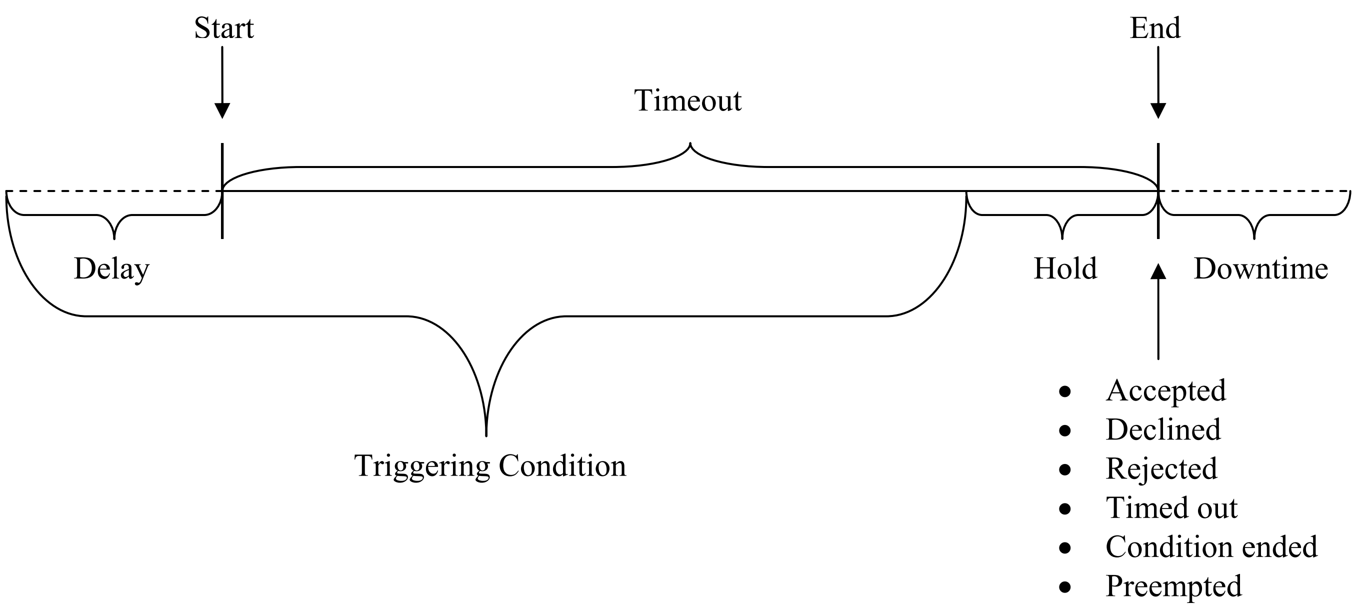


Figure 4-2. Life span of a suggestion. A suggestion is visible to the user between the Start and End markers on the timeline.

Figure 4-2 shows the life span of a suggestion. The extreme left of the timeline is the point at which a suggestion becomes active in the system. The Delay parameter specifies an amount of time to delay before displaying the suggestion to the user. As previously described, the system uses the delay to make sure the triggering condition is "really" true. A suggestion is visible to the user between the Start and End markers on the timeline. The Timeout parameter specifies maximally how long a suggestion can remain visible before deactivating and disappearing.

The Hold parameter specifies how long to keep a suggestion visible after its triggering condition has ended. Suggestions tied to the joystick, such as the camera suggestions, use this parameter to artificially prolong a suggestion. Otherwise, the user might stop moving the joystick to acknowledge a suggestion and cause the suggestion to disappear. Beyond the End marker, a suggestion is no longer active. The Downtime parameter is used to suppress a suggestion for some time so it does not reappear immediately after being declined by the user or ignored until it times out.

The bulleted list beneath the End marker shows all the ways a suggestion can terminate. The user can explicitly accept or decline a suggestion using “simulated voice recognition” (Section 5.2.2), or make a suggestion go away forever by rejecting it. Accepting a suggestion causes its action to take place and declining a suggestion causes it to be deferred for some time specified by the Downtime parameter. If a suggestion times out, then we assume the user ignored the suggestion and declined it *implicitly* so the suggestion is deferred in this case too.

If the condition that triggered a suggestion ends before the suggestion terminates in some other way, it is deferred to prevent suggestion “flicker”. The Downtime parameter can have a different value for each of these cases (declined, implicitly declined, condition ended). There are two ways in which the condition causing a suggestion can end. If a suggestion is based on sensing the environment, the environment can change. For example, the user can navigate the robot to a bright area and cause the Turn Lights On condition to end. Or, the user can turn the lights on using an interface control instead of taking the Turn Lights On suggestion. In the later case, the user took the suggestion *implicitly*. The suggestion system records statistics on implicitly and explicitly accepted and declined suggestions.

The final way an active suggestion can be terminated is by preemption (Section 4.2.1). To summarize, there are four timing parameters that determine how a suggestion behaves. There is a Timeout parameter, a Delay parameter, a Hold parameter, and a

Downtime parameter. The only parameter that strictly requires a nonzero value is the Timeout parameter, but the developer can tune all of the parameters to make a suggestion act intelligently. Appendices A and B show developer applications for visualizing and tuning the suggestion parameters.

4.3 Suggestion Icons

Through the iterative process of trying different icon designs, we have arrived at the current icon design (Figure 4-3). We have also developed design principles for suggestion icons that we use as a litmus test for icon design (Figure 4-4).

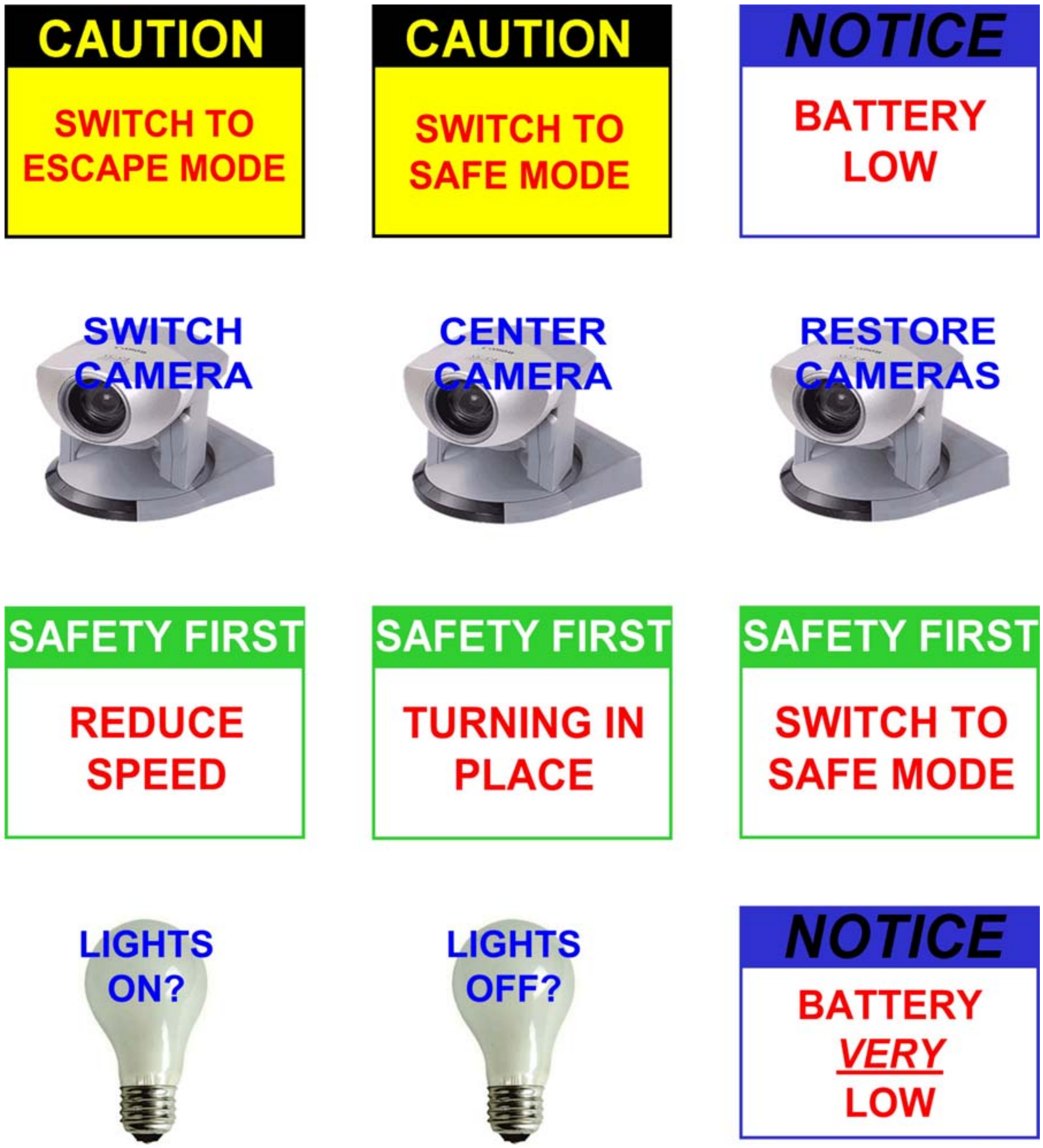


Figure 4-3. Current suggestion icons.

A suggestion icon should communicate the following to the user:

1. There is a problem.
2. This is the problem.
3. This is what I can do to fix the problem (or you can do it yourself).

Figure 4-4. Design principles for suggestion icons.

4.4 Evolution of Interface Designs

This section is included to show previous interface designs in which the suggestions were less noticeable than in the current version. In a study of a previous version of the suggestion system, a subject remarked that he was so focused on the video that he did not notice the suggestions at all (Section 5.1). In each revision, the suggestions migrated closer to (and eventually onto) the main video display to exploit users' natural locus of attention.

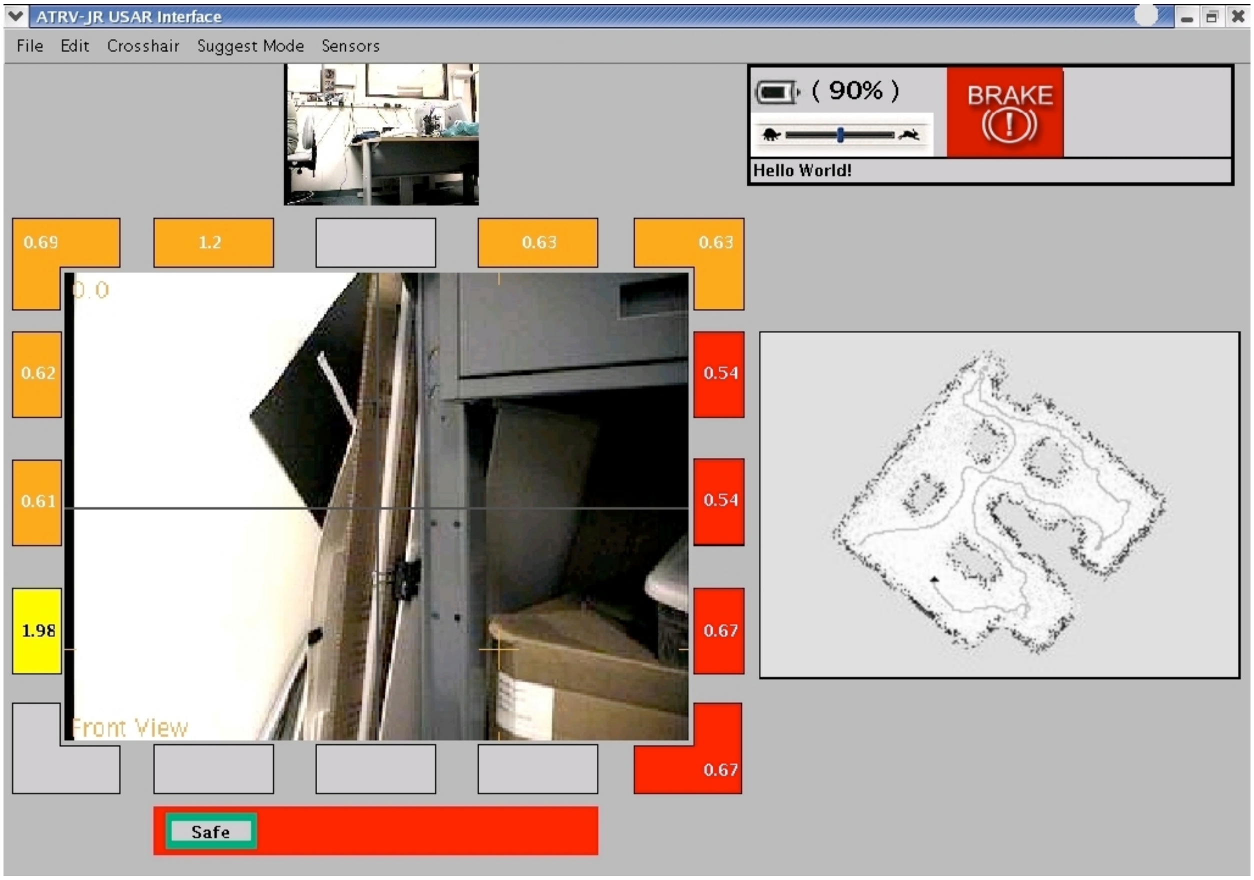


Figure 4-5. The earliest version of the interface that included a mode suggestion system.

Figure 4-5 shows the earliest version of our interface that included a mode suggestion system. Suggestions appear as buttons on the mode indicator bar beneath the main video display. The color of the mode indicator bar indicates which mode the robot is in. (As previously mentioned in Section 3.2, the autonomy modes are color coded to make is easy for the user to tell which mode the robot is in.) The button at the left side of the bar is a suggestion to switch to Safe mode. This design uses color, position and text to make it easy to notice which mode is being suggested. The text label is important because the user may have color blindness or forget the color correspondence.

Figure 4-6. The suggestion system "box" appears at the upper right corner of the interface.

Figure 4-6 shows a later version of the interface in which suggestions appeared in a "suggestion box." This version of the suggestion system added battery, camera and lights suggestions to the existing mode suggestions. This version included iconographic representations of suggestions to make the suggestions more noticeable and make it clear what type of suggestion is being offered. Textual descriptions were added to help the user understand what is being suggested and why it is being suggested. The F5-F8 labels indicate which function key the user must press to take a suggestion. The design of the suggestion box uses color and position to aid recognition of suggestions. The green robot at the left side of the suggestion box represents a suggestion to switch to Safe mode.

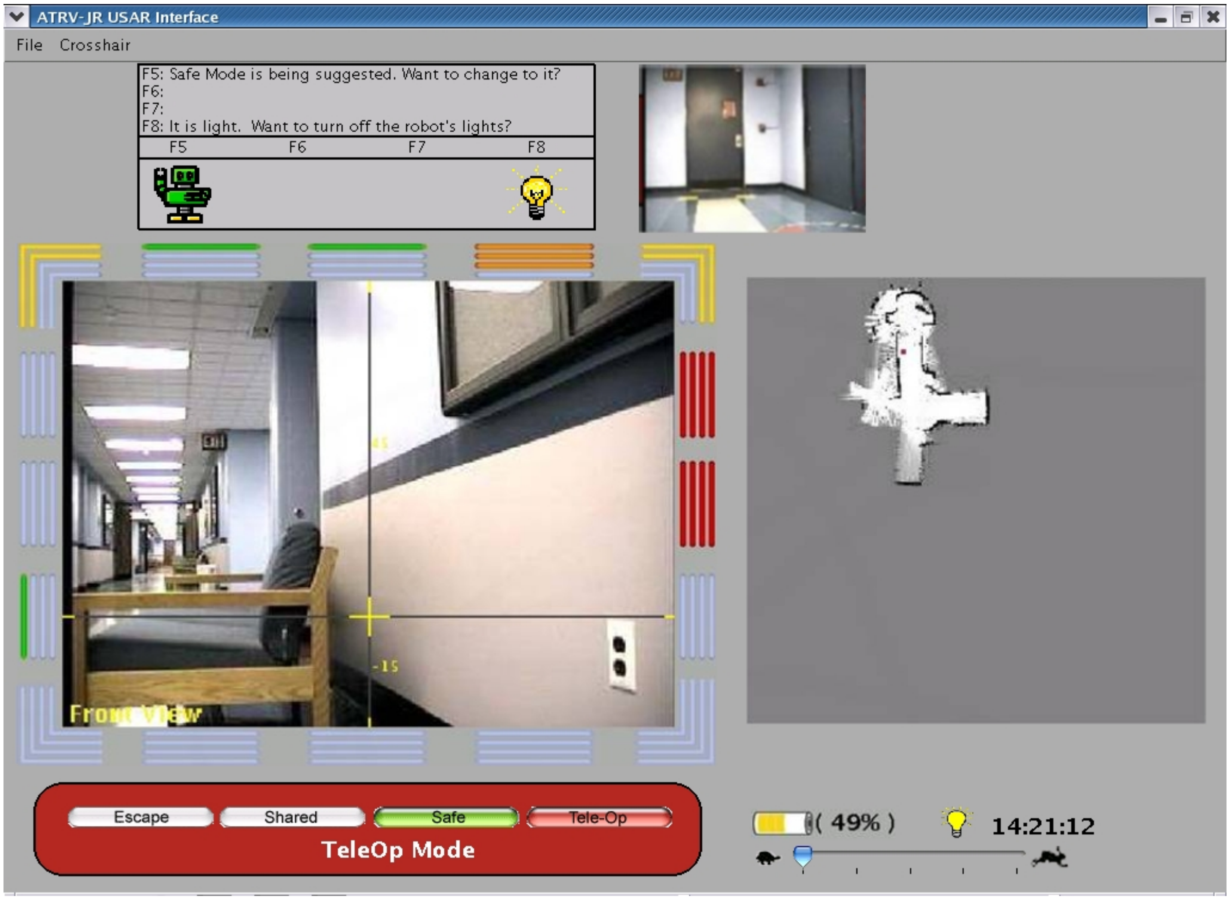


Figure 4-7. The suggestion system box appears directly above the main video display.

Figure 4-7 shows a redesign of the interface that brings the suggestion system box closer to the video to make the suggestions more noticeable. Notice the redesign of the suggestion box itself to bring the icons closer to the video. Testing of the previous version of the interface showed that users did not notice suggestions because they were intently focused on the video display while driving the robot.

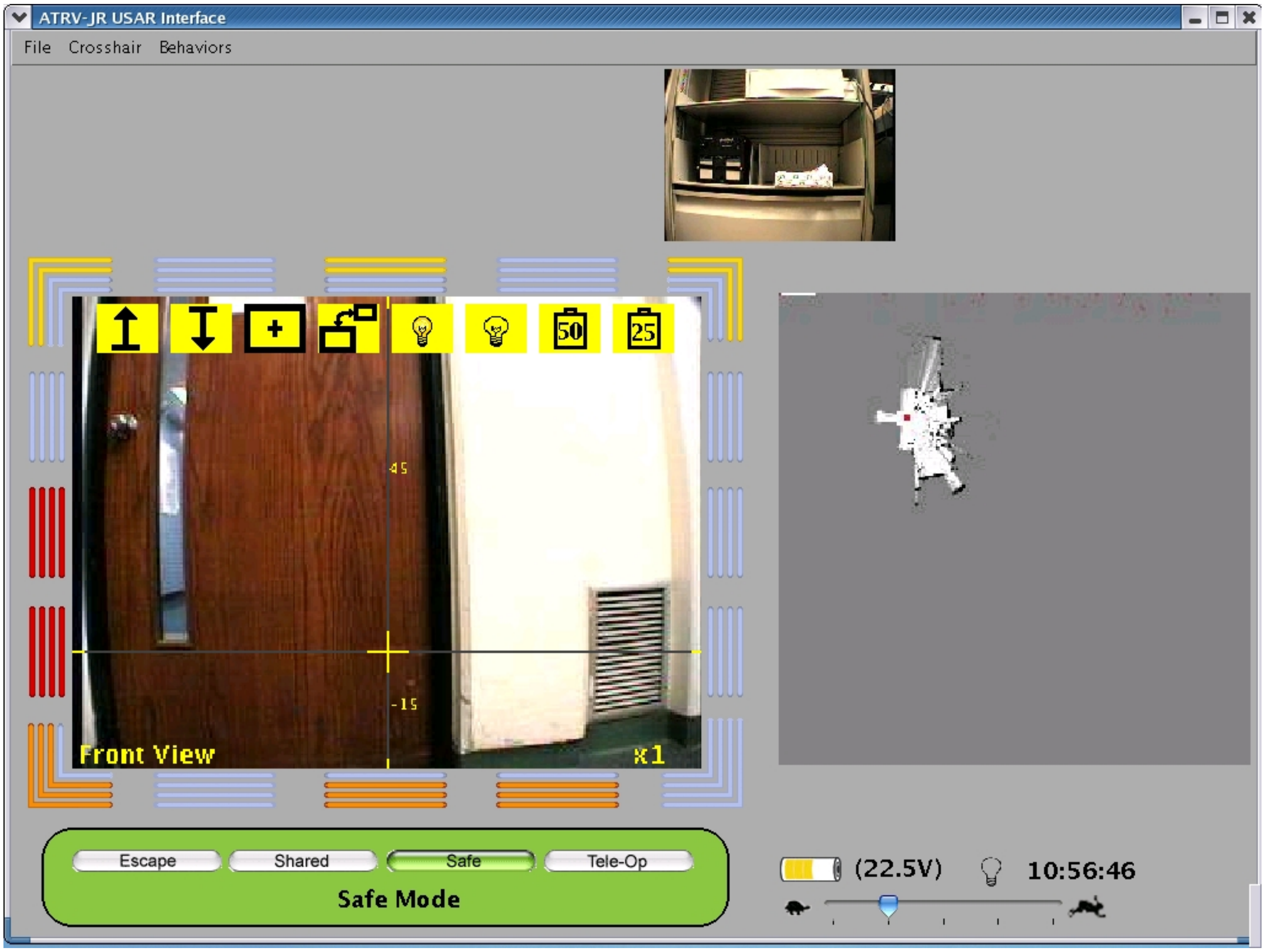


Figure 4-8. In this version of the interface, suggestion icons appear on top of the video display.

Figure 4-8 shows a version of the interface that eliminates the suggestion system box in favor of placing icons directly over the video image. This design is even more noticeable than the previous design without being obtrusive since it does not block the video. The user can choose to ignore a suggestion and continue operating the robot. The compact icons have symbolic representations of suggestions that do not provide enough meaningful description. For example, the leftmost suggestion icons are up and down arrows that signify an increase or a decrease in autonomy, respectively. From this description, it is not entirely clear which mode is being suggested and there is no indication why a mode switch is being suggested. This design, the immediate predecessor of the current design, was abandoned without ever being tested formally.

5 Experiments and Qualitative Results

5.1 Preliminary Experiment

Figure 5-1 shows a version of our interface that we evaluated in an early study of our robot system. The participants in this study responded to a video game survey that we used to recruit subjects because we wanted to investigate the hypothesis that video gamers would find our robot system intuitive and easy to use. There are a number of similarities between our control interface and video games including a first-person camera perspective, an overhead map, mode and camera view switching abilities, and various status indicators.

Figure 5-1. Suggestion system evaluated in a 4-subject study.

The subjects in this study were students at this university, three males and one female. Each subject was allowed fifteen minutes to train on the robot system and then asked to perform a timed task in a mock USAR environment, which involved finding and identifying six mock victims. The USAR environment was located in a poorly lit basement and consisted of wall segments arranged in a disjointed pattern to simulate an urban disaster site.

During his test run, the first subject recommended flashing the suggestion icons on the video screen to make them more noticeable, and then moving them to the suggestion box. The experimenter noted that this subject would turn the robot around in place rather than switch camera views. The second subject used a strategy of moving the robot to the center of the room and scanning (turning in place) to search the area. The experimenter also noted that this subject never switched camera views. In Teleoperation mode, the second subject asked, “Did I just hit the wall?” when he heard the sound of a wall being hit. The experimenter noted that the third subject would “pan” the robot instead of panning the camera. The fourth subject heard the robot hit something in Teleoperation mode and switched to Safe mode. Rather than use ADR mode, she spent a long time backing the robot out of an area.

Three of the subjects in this study tended to pan the robot instead of the cameras, which we have observed in other studies (Section 1.2). (This version of the suggestion system did not include a Turning In Place suggestion, which triggers whenever the robot is being rotated toward obstacles (Section 4.1.2).) We consider panning the robot instead of the cameras a dangerous driving practice, especially for novice users of the system. Some subjects appeared to not notice the suggestions and some did not take advantage of the useful ADR capability. Finally, subjects were unable to drive safely in Teleoperation mode.

These observations prompted us to move the suggestions closer to the video and eventually onto the video. The Turning in Place suggestion (Section 4.1.2) could have made these subjects aware of the danger of turning the robot in place. Had the suggestions been noticed in this experiment, the Switch Camera View suggestion (Section 4.1.5) might have encouraged users to use ADR mode instead of backing the robot. Finally, the Teleop-to-Safe and Reduce Speed suggestions could have emphasized the need to be especially careful while driving in Teleoperation mode.

5.2 Full System Evaluation

5.2.1 Subject Population

We conducted a 16-subject experiment for the purpose of evaluating the suggestion system qualitatively and collecting data for machine learning (see Chapter 6). Our subject population consisted of 12 males and 4 females whose ages ranged from 19 to 65 (average: 31.38, stddev: 15.17). Thirteen of our subjects were students, 6 were past or present computer science majors, one was a faculty member, and two were not affiliated with the university.

5.2.2 Experimental Setup

We constructed a test arena in the hallway near our laboratory to simulate a USAR task environment. The maze consisted of hallways, doorways, narrow passages, various obstacles, and a long unlit and narrow tunnel assembled using wood paneling. The tunnel was not designed to collapse easily, but it was possible for an operator to bump the robot into the sides of the tunnel and cause the whole structure to fall.

At the start of a run, users were seated at the user interface and operator controls, and interviewed by an experimenter to determine the subject's computer and robot experience. The experimenter then described the robot system using the script in Appendix C to ensure a consistent description to all subjects. Subjects were provided with the "cheat sheet" in Appendix D, which the user could consult during the test for the keyboard and suggestion system controls.

The suggestion system was described as a help feature of the interface and users were told how to respond to suggestions using simulated voice recognition (Section 5.2.3). No subject received a detailed description of any of the suggestions because we wanted to test the design of the suggestion icons according to the design principles in Figure 4-4. Subjects were told that they could ask questions about any of the interface features (including the suggestions) at any time during the run.

5.2.3 Simulated Voice Recognition

Previous versions of the suggestion system that allowed multiple suggestions used function keys for accepting the various suggestions. This design forced the user to stop driving the robot to find and press a key on the keyboard. Therefore, responding to a suggestion involved a mental "context switch" away from and back to the driving task. We hypothesized that users would find it easier to ignore the suggestions than be pulled away from the driving task momentarily.

When the suggestion system changed to a single-suggestion-at-one-time design, the interface needed to provide a single button to accept a suggestion. Using a button on the joystick would make it easier for users to take suggestions while continuing to drive the robot, but the suggestion system also allowed users to decline or reject suggestions. Interacting with three buttons on the joystick would be unwieldy for the user so we came up with an alternative solution called "simulated voice recognition." In principle, we could ask subjects to train voice recognition software before a user test, but this would take too long to be practical. Instead, an experimenter sitting near the user listens to voice commands and presses a corresponding key on a keyboard connected to the interface.

Users were permitted to speak commands informally to make the interaction as natural as possible since users may find talking to an interface unfamiliar and strange at first. For example, a user could say "yes" or "ok" to accept a suggestion, or "go away" to reject a suggestion. When it is not clear what the user intends, the experimenter would ask what the user means. Since this method of responding to suggestions has low mental and physical overhead, and permitted the user to interact with another control method simultaneously, we presumed that users would be more likely to interact with this suggestion system than previous versions that used more cumbersome input methods.

5.3 Data Collection

In addition to a number of log files that recorded data for machine learning (Chapter 6), we collected video, voice and written forms of data. An experimenter who followed the robot during a test run had three responsibilities. First, the experimenter recorded the robot throughout the run using a video camera. Second, the experimenter recorded critical events on a score sheet (Appendix F). Critical events included the robot hitting something, the discovery of a mock victim by the test subject, or a hardware (or software) failure of the robot (or interface) that forced a restart of the run. Finally, the experimenter was a safety monitor whose job was to press an emergency stop button on the robot whenever the robot was about to cause damage, either to itself or the environment.

Another experimenter, seated with the test subject, used a tape recorder to record the subject's comments during the test run. Also, this experimenter interviewed each subject before and after a test run and took notes about a subject during a run. Our pre- and post-run questionnaire is provided in Appendix E. In the earlier experiment (Section 5.1), subjects filled in forms themselves, but in this experiment the test administrator interviewed subjects and transcribed their responses. In this way, the test administrator could ask follow-on questions when appropriate.

5.4 Data Analysis

Figure 5-2 shows how users responded to suggestions. If a user performs the action of an active suggestion manually, we say the user accepted the suggestion implicitly. We also assume that timed-out suggestions were ignored by the user, and therefore declined implicitly (Section 4.2.2). Figure 5.3 provides a breakdown of the suggestions made during the user tests. According to the breakdown, the Escape Mode, Lights Off and Turning in Place suggestions were the most frequently triggered suggestions.

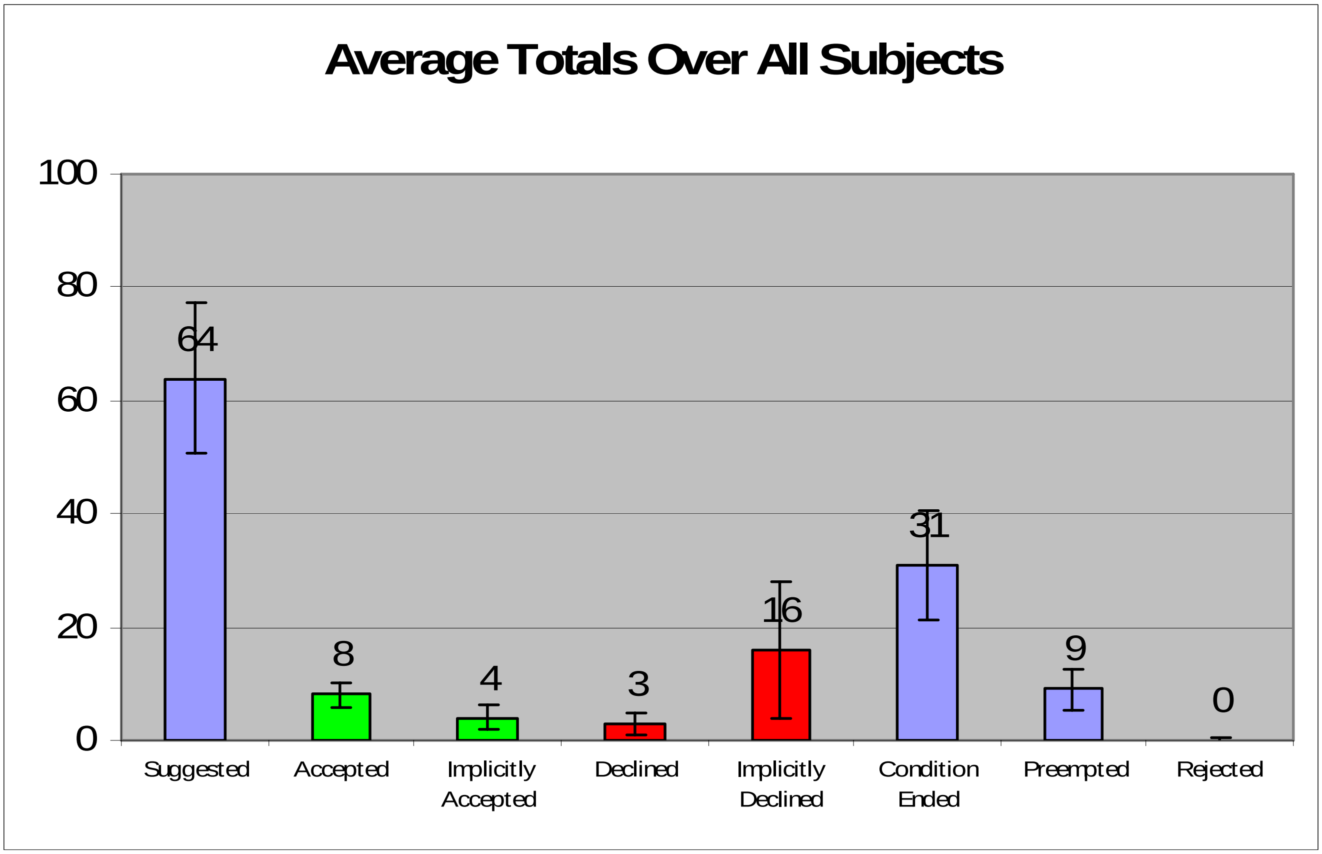


Figure 5-2. On average, 64 suggestions were made during a user run lasting approximately 45 minutes.

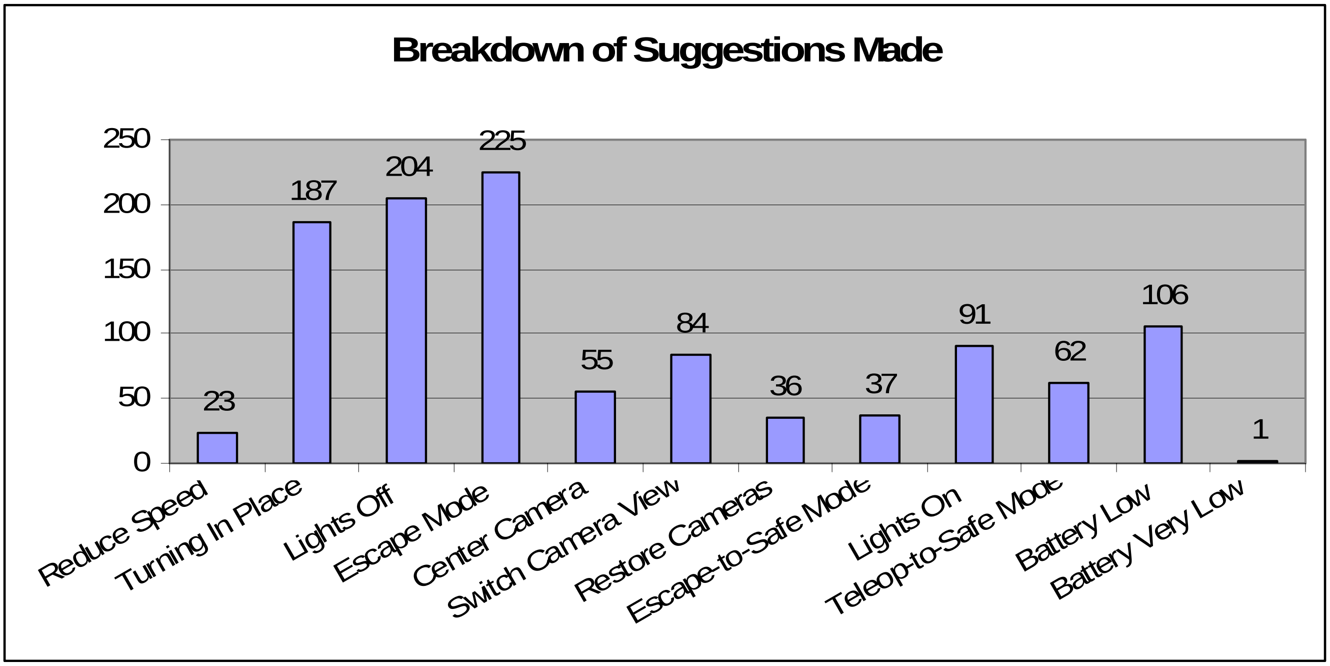


Figure 5-3. Breakdown of suggestions made for all subjects by type.

Suggestion	Accepted	Implicitly Accepted	Declined	Implicitly Declined	Condition Ended	Preempted	Rejected
Reduce Speed	4	2	0	3	11	3	0
Turning In Place	36	0	14	27	88	21	1
Lights Off	20	9	7	112	14	42	0
Escape Mode	12	7	19	37	150	0	0
Center Camera	1	16	5	1	30	2	0
Switch Camera View	10	17	10	0	39	7	1
Restore Cameras	7	6	1	0	16	6	0
Escape-to-Safe Mode	18	0	2	13	0	3	1
Lights On	11	5	1	32	4	38	0
Teleop-to-Safe Mode	6	2	1	12	29	11	1
Battery Low	0	0	0	0	105	1	0
Battery Very Low	0	0	0	0	1	0	0

Table 5-1. Breakdown of suggestions for all subjects by termination.

Table 5-1 provides a more detailed presentation of the suggestions showing the outcome of every suggestion made during the user tests. Sixty seven percent of the Escape Mode suggestions disappeared prematurely, indicating that the robot was not really stuck when Escape mode was suggested in those cases. Although the triggering condition for the Escape Mode suggestion tries to make sure the robot is actually stuck by polling the sonars for some time, the notion of "stuck" is based on raw sonar values which are notoriously noisy.

The high number of Lights Off suggestions made (204) was due to an implementation problem, which was only experienced by Subject 1 who received 50% of the total number of Lights Off suggestions. The fact that only 7 of the 204 Lights Off suggestions were declined explicitly shows that even "nagging" suggestions can easily be ignored and do not annoy users.

Finally, the Battery Low suggestion was made 106 times without ever timing out or soliciting a user response. This reveals another implementation problem and points out the difficulty of using unreliable sensor data to detect the true state of the robot and its environment. As explained in Sections 2.7.3 and 4.1.7, the battery voltage dips whenever the robot is being driven fast or encounters resistance. Since our robot has skid steering, it experiences significant friction whenever it is being rotated in place and the high number of Turning in Place suggestions (Figure 5.3) implies that the robot was rotated often.

Although the Escape Mode, lights, and battery suggestions all use a delay timer to ensure that the triggering condition is valid, these suggestions have to balance safety against making false suggestions occasionally. Especially in the case of the Escape Mode suggestion, waiting too long before making a suggestion could have very negative consequences. The high number of false Escape Mode, lights and battery suggestions, however, underscores the need for better sensors and detection algorithms in the suggestion system.

5.4.1 Statistical Comparison of Subject Responses to Suggestions

t-Tests were performed to examine the differences in subject responses to suggestions. We compared subjects by gender, age, video game experience, and computer experience. The t-Tests (two-tailed distribution, two-sample unequal variance) showed no statistical differences ($\alpha = 0.05$) between subject groups. The results of these tests are shown in Tables 5-2, 5-3, 5-4, and 5-5.

Gender	Average % Accepted	Average % Implicitly Accepted	Average % Declined	Average % Implicitly Declined
Female (4/16)	31.72 (13.57)	14.72 (17.66)	8.74 (10.46)	44.82 (30.11)
Male (12/16)	32.68 (19.89)	15.95 (17.05)	18.17 (19.43)	33.21 (28.80)
p-value	0.93	0.92	0.29	0.58

Table 5-2. Comparison of subject responses by gender.

Age	Average % Accepted	Average % Implicitly Accepted	Average % Declined	Average % Implicitly Declined
< 25 years old (10/16)	34.67 (19.93)	15.36 (16.89)	13.39 (18.89)	36.58 (28.29)
> 25 years old (6/16)	28.71 (15.16)	16.11 (17.73)	19.84 (15.85)	35.34 (31.56)
p-value	0.54	0.94	0.51	0.94

Table 5-3. Comparison of subject responses by age.

Video Gamer	Average % Accepted	Average % Implicitly Accepted	Average % Declined	Average % Implicitly Declined
Yes (6/16)	34.07 (14.31)	17.35 (15.19)	20.72 (21.65)	27.86 (21.08)
No (10/16)	31.46 (20.57)	14.61 (18.25)	12.87 (14.80)	41.06 (32.66)
p-value	0.79	0.77	0.49	0.38

Table 5-4. Comparison of subject responses by video game experience.

Computer Experience	Average % Accepted	Average % Implicitly Accepted	Average % Declined	Average % Implicitly Declined
Moderate (2/16)	26.01 (8.47)	26.80 (18.03)	4.33 (2.57)	42.86 (29.07)
Expert (7/16)	35.40 (23.84)	11.90 (14.23)	16.85 (15.13)	35.84 (29.22)
Guru (7/16)	31.31 (13.15)	16.19 (18.23)	18.05 (21.57)	34.45 (29.77)
Moderate vs. Guru p-value	0.65	0.66	0.18	0.82
Expert vs. Moderate p-value	0.51	0.56	0.10	0.85
Expert vs. Guru p-value	0.72	0.66	0.91	0.94

Table 5-5. Comparison of subject responses by computer experience.

5.5 Qualitative Results

5.5.1 Subject Responses to the Lights On and Lights Off Suggestions

Five subjects cited a lights suggestion as one of the most useful suggestions and five cited a lights suggestion as a least useful suggestion. One subject said the Lights On suggestion is unnecessary because “you know when you need them.” He said the Lights off suggestion is “good because you forget they’re on.” (There is a light bulb icon on the interface that shows when the light is on (lower right corner of Figure 4-1)). Another subject supposed that the Lights On suggestion is more helpful than Lights Off, but drains the battery more. One subject remarked that she didn’t need the lights on for this task while another subject said, “I can do it better.” This subject responded to a Lights Off suggestion by saying, “No, leave them on,” and to a Lights On suggestion by saying, “I’m ignoring it because I can see.” Finally, the fifth subject who found a lights suggestion least useful asked, “Why did it even suggest that?” during his run and later stated that having the lights on or off is “not a bad situation.”

One of the five subjects who found a lights suggestion most helpful did not want the lights off in the hallway while another subject appreciated the Lights Off suggestion because he had forgotten the keyboard control. (The keyboard control for the lights was included on the “cheat sheet” provided to every subject (Appendix D).) He also remarked that the Lights Off suggestion is good because it saves battery and that he had completely forgotten that the lights were on (despite the light bulb indicator on the interface). Interestingly, none of the subjects was told that the Lights Off suggestion had anything to do with saving battery power, but two subjects assumed that purpose.

In the earlier experiment (Section 5.1), the Lights On suggestion elicited positive comments from all of the subjects, but garnered a mixed reaction in this study. As with the Turning In Place suggestion, reactions varied widely across subjects. In the earlier study, the basement was quite dark and users could easily appreciate the helpfulness of the suggestion, but in this study the benefit of taking a lights suggestion was less clear (except within the tunnel obstacle). Most of the time the robot was in a hallway with ample fluorescent lighting, but subjects who tilted the camera up toward the ceiling lights (some did) saw the severe darkening effect caused by the camera's auto gain and auto white balance features. A smarter Lights Off suggestion could use a vision algorithm to detect this condition and offer to disable the camera's auto gain and auto white balance.

5.5.2 Subject Responses to Mode Suggestions

Subject responses to the mode suggestions were mostly positive with seven subjects considering Escape Mode a most helpful suggestion. One of these seven rated it as most useful even though he said it moved opposite to what he wanted and expected. He recommended changing it so you could tell it which way to go. Another of these subjects said it "worked well" and was a "good decision" during his run. Another subject discovered a new use for Escape mode and explained why it was useful: "That was a handy feature, it brought me back out into the clear and angled me exactly where I had to go. Previously I could not get through the door in Safe mode…that was a great help because it angled me to go through the door and then I didn't have to worry about anything." Later, he stated that Escape mode turned out to be very helpful because, "I was stuck in an area and it brought me back where I could see everything better and then go right through [the doorway]."

One subject who took an Escape Mode suggestion reacted with, "This is good. I really like Escape mode. I think it did a good job." One subject remarked "good option" when presented with the Escape-to-Safe Mode suggestion after the robot finished escaping in Escape mode. Another subject ranked the Escape-to-Safe Mode suggestion as the most useful suggestion and one subject ranked the Teleop-to-Safe suggestion as the most useful because it's "better than me." A different subject said the "Go to Safe mode" suggestion was one of the most useful because "I am not thinking about which mode I'm in." Among the less positive responses to the mode suggestions, one subject said the Escape Mode suggestion was not clear at first. One subject responded to an Escape Mode suggestion by saying, "No, I'd rather escape myself." Finally, one subject ranked the Escape Mode suggestion as least useful, but only because he did not use it.

5.5.3 Subject Responses to Camera Suggestions

The camera suggestions elicited 5 negative responses, 2 positive responses, and 3 neutral responses. One subject commented during his run that the camera suggestions are not helpful because, "If I wanted to do that, I would probably realize it and do it myself." At another point in his run, he said "good suggestion" in response to a Center Camera suggestion and later cited Restore Cameras as one of the least helpful suggestions because "you took the [Turning in Place] suggestion so you obviously wanted to move the camera." He further remarked that Restore Cameras is not helpful because "I can just do it myself." Another subject cited Restore Cameras as a negative suggestion and described it as "odd." It was not clear to him whether he could still take the suggestion after it disappeared and he misunderstood the suggestion to mean switch the cameras back after taking a Switch Camera View suggestion. Interestingly, this subject said that Switch Camera View was the most useful suggestion.

One subject said, "But I see it's clear," when presented with a Switch Camera View suggestion and two others reported Switch Camera View as a least helpful suggestion. One of the two subjects who found Switch Camera View least useful said she already knew she was going backwards and using the "rear view mirror." The other subject who found it least useful always switched the camera view herself whenever that suggestion came up. She also was confused by the Center Camera suggestion because she thought center was wherever the camera was pointed and that the robot would move in that direction. When she received an explanation of the Center Camera suggestion, she stated, “I didn’t catch on.”

One subject stated, “Very smart, I like that,” in response to a Center Camera suggestion and another subject said, “That’s good,” whenever he saw a Center Camera or Switch Camera View suggestion. Among the neutral responses, one subject said the Switch Camera View suggestion is “kind of a reminder actually,” and another subject who appeared to not understand the Switch Camera View suggestion did not ask for an explanation of it. Finally, one subject did not want to switch the camera view and preferred to use the “rear view mirror” instead. She asked whether “the robot cared” about being driven backwards and was it “bad for the robot.”

5.5.4 Subject Responses to the Turning in Place Suggestion

As described in Section 4.1.2, the Turning in Place suggestion was created to mitigate a problem that novice users frequently encounter while operating our robot platform. Because our robot is a large rectangular platform with skid steering, and since novice users tend to “pan” the robot to see the environment instead of panning the cameras, there is an increased chance of bumping the robot while turning it in place. If taken by the user, the Turning in Place suggestion automatically pans the camera(s) to show any obstacles that the robot will hit if the user continues turning in place. Originally, this suggestion would trigger whenever the user rotated the robot, but it triggered too frequently so we changed it to trigger only when it sensed nearby obstacles as well.

Six subjects responded positively to the Turning in Place suggestion, four responded negatively, 3 had a neutral response, 3 did not comment on it, and one subject provided several ambivalent comments. Among the negative responses, one subject remarked, “Now I feel lost, not what I expected,” after taking a Turning in Place suggestion. Later, this subject cited Turning in Place as the least useful suggestion and described it as a “strange experience.” Another subject, who also cited Turning in Place as the least useful suggestion, described it as annoying and too frequent. A third subject said that she “hated” the Turning in Place suggestion and rated it as both confusing and not understandable. When asked what she thought it meant, she described it as a warning that means, “You are not looking where you are going.”

Finally, the subject who had the most negative reaction to the Turning in Place suggestion made a number of comments about it. While interacting with the Turning in Place suggestion, he said it did not make sense and slowed him down. Initially, he understood the suggestion as asking him to do something. Later, he stated that he did not want an explanation of what the robot is doing and that he did not need to know this level of detail. He characterized the suggestion as “diagnostic” information from the “autonomous operator.” He stated that time is critical and reiterated “this slows me down.” During the post-run interview, he described the Turning in Place suggestion as “useless” and “interrupts what I’m doing and offers no help.” He rated the Turning in Place suggestion as confusing and not understandable in response to those questions on the questionnaire.

Among the positive responses to the Turning in Place suggestion, one subject rated it as the most useful suggestion because it provided positive feedback. Another subject said, “It showed me [the obstacle], that helped,” while interacting with the Turning in Place suggestion. During the post-run interview, this subject cited the Turning in Place suggestion as one of the most useful features of the interface overall. He described it as, “switching the cameras right to the general vicinity of what was blocking me.” He also stated that Turning in Place was the most helpful suggestion and referred to it as “very helpful.” When another subject first encountered the Turning in Place suggestion, he said, “Nice ability, I didn’t think it could do that.” Later, he said, “I like it to tell me,” and, “I like the Turning in Place feature; it’s not in video games.” During the post-run interview, he referred to the Turning in Place suggestion as “helpful” and “reinforcing.”

Subjects who exhibited a neutral response did not understand the Turning in Place suggestion. One subject stated that the Turning in Place suggestion is not clear and another subject couldn't determine whether the suggestion was "telling me or limiting me." He also stated that turning the robot and the cameras simultaneously was disorienting.

The single subject who responded ambivalently stated that he prefers to move the camera himself to understand what is happening. He said if the camera just starts moving, it takes a second to figure out in which direction it is moving. He went on to say, "I like to have control, I prefer a standard car to an automatic." Later, he described the Turning in Place suggestion as "irritating" and "a pain," and stated, "I'd rather do it myself."

During the post-run interview, however, this subject cited Turning in Place as the most helpful suggestion and said it was nice to have it pan the cameras. When asked if all of the suggestions were understandable, he responded that it was not obvious what Turning in Place would do, but "you learn right away." Later in the conversation, he said that the Turning in Place suggestion "pops too much" and keeps "telling me" that I'm turning and "no kidding." Finally, he recommended changing the Turning in Place icon because "Safety First" is an irritating phrase. The suggestion system made the Turning in Place suggestion 12 times on average and 19 times to this subject who accepted it 4 times, ignored it 11 times and declined it once.

The Turning in Place suggestion elicited a wide range of responses from our subjects. Subjects who disliked the Turning in Place suggestion did not understand the suggestion, but neither did some subjects who found it useful. Even though most subjects did not understand the Turning in Place suggestion, it was a successful suggestion because it taught novice users to exercise caution while turning the robot in place. With respect to the design principles in Figure 4-4, the Turning in Place suggestion does not communicate what it can do to fix the problem (the outcome of taking the suggestion). Section 7.5.1 discusses how to improve the Turning in Place suggestion.

5.5.5 Subject Responses to Simulated Voice Recognition

Three subjects responded negatively to simulated voice recognition, two responded favorably, and two exhibited a neutral response. One subject kept stating that he was ignoring suggestions and at one point said, "I'm getting a safety warning, I'm going to ignore it." When asked why he didn't use a voice command to decline it, he said, "It did not occur to me." When asked the same question later, he responded, "It never occurred to me to say 'not now'." He later admitted that talking and saying commands at the same time is hard and that he was not used to talking to a computer. Another subject made this point even more explicitly by stating, "My verbal channel is a comment channel, not a control channel."

A third subject who appeared reluctant to speak commands explained it as "one less control to think about" and "two different ways of doing the same thing." This subject was also the least talkative subject in terms of making comments during his run. Another subject made comments continuously throughout her run and also appeared comfortable speaking commands at the same time. One subject did not realize how simulated voice recognition worked until late in his run. (All subjects were given the same description of simulated voice recognition prior to their run.) One subject who liked simulated voice recognition said, "It's easier to do by voice command," and "It helps because you are doing too many things at once sensory-wise." Another subject said he "liked voice controls" and disliked keyboards and mice.

Reactions to simulated voice recognition varied widely between subjects. Using voice recognition may have been an unfamiliar experience for some users. Perhaps adding a "prop" microphone to the operator controls would make the interaction feel more natural. In any event, the suggestion system should provide an alternate means of responding to suggestions for users who do not want to use voice commands.

5.5.6 Subject Attitudes toward Audio Feedback

The subject questionnaire did not include a question about audio feedback from the robot, but five subjects were asked their opinion about audio feedback. We use Festival speech synthesis software to provide debugging output on the robot [Festival 2006]. For example, when the wireless connection drops out, our robot speaks the phrase "no heartbeat" until the connection restores. (When the robot stops receiving the "heartbeat" message from the interface, it stops moving until the interface regains control.) Sometimes subjects would ask what the robot was saying, but subjects could not hear the audio "feedback" when the robot was in the remote task environment.

One subject said that audio feedback is helpful while another subject said that she would only want audio feedback if it were "useful and important." A subject who said that audio feedback is not helpful explained his response by saying, "You are paying attention to too many things at once and all the info is there [interface]." He went on to say that it could be a useful reminder, but not once you are "adjusted and experienced." Another subject who said audio feedback is not helpful provided a different reason. He said that he was used to systems with multiple alarms and that because the sound channel has limited bandwidth, this creates a priority problem. He recommended putting all alarms in the visual field.

Finally, the subject who was the most enthusiastic about audio feedback said the robot should say "oops" when it bumps. She wanted the robot to talk to her and also provide warning beeps that would become faster and louder as the robot came nearer to obstacles. We can see that attitudes toward audio feedback may vary widely between subjects and that prior learned expectations can influence this opinion. If the user interface or suggestion system were to incorporate audio feedback, we would need to provide a mechanism for enabling and disabling that feature.

5.5.7 Overall Subject Responses to the Suggestion System

Table 5-6 shows a summary of subject responses to suggestions. Thirteen of the sixteen subjects replied "yes" to the question, "Did you find the suggestions helpful?"

Suggestion	Most Helpful?	Least Helpful?	Unclear?
Escape Mode	9	1	0
Escape-to-Safe Mode	4	0	0
Switch Camera View	4	4	0
Center Camera	3	0	0
Turning In Place	7	3	7
Restore Cameras	1	1	1
Lights On	6	4	0
Lights Off	7	2	0
Teleop-to-Safe	4	0	0
Reduce Speed	2	0	0
Battery Low	1	0	0
Battery Very Low	1	0	0

Table 5-6. Summary of subject responses to suggestions. Nine subjects said the Escape Mode suggestion was the most helpful suggestion and seven subjects did not understand the Turning in Place suggestion.

One of these subjects said he "grew to enjoy them [suggestions]." Of the three subjects who did not answer "yes" to this question, one said the suggestions were "occasionally" helpful and one said they were "marginally" helpful. The last of these three said the suggestions were "somewhat" helpful and provided "vaguely nice companionship."

During the post-run interview, three subjects referred to the suggestions as "reminders" and two referred to them as "popups." (The test administrator never described the suggestions as reminders or popups during any of the testing sessions.) One subject described the suggestions as "good reminders about things" and telling you "I shouldn't be doing that." When asked if he found anything negative about the suggestions, he said they were irritating when you were trying to do something intentionally. He described them as a "crutch for beginners" and said that they should be completely removed for experienced users. He made a reference to the Windows operating system saying that it was designed for beginners. When asked what he would change about any of the suggestions, he said that he wanted a button to disable all of the suggestions temporarily, but also admitted, "That would be dangerous."

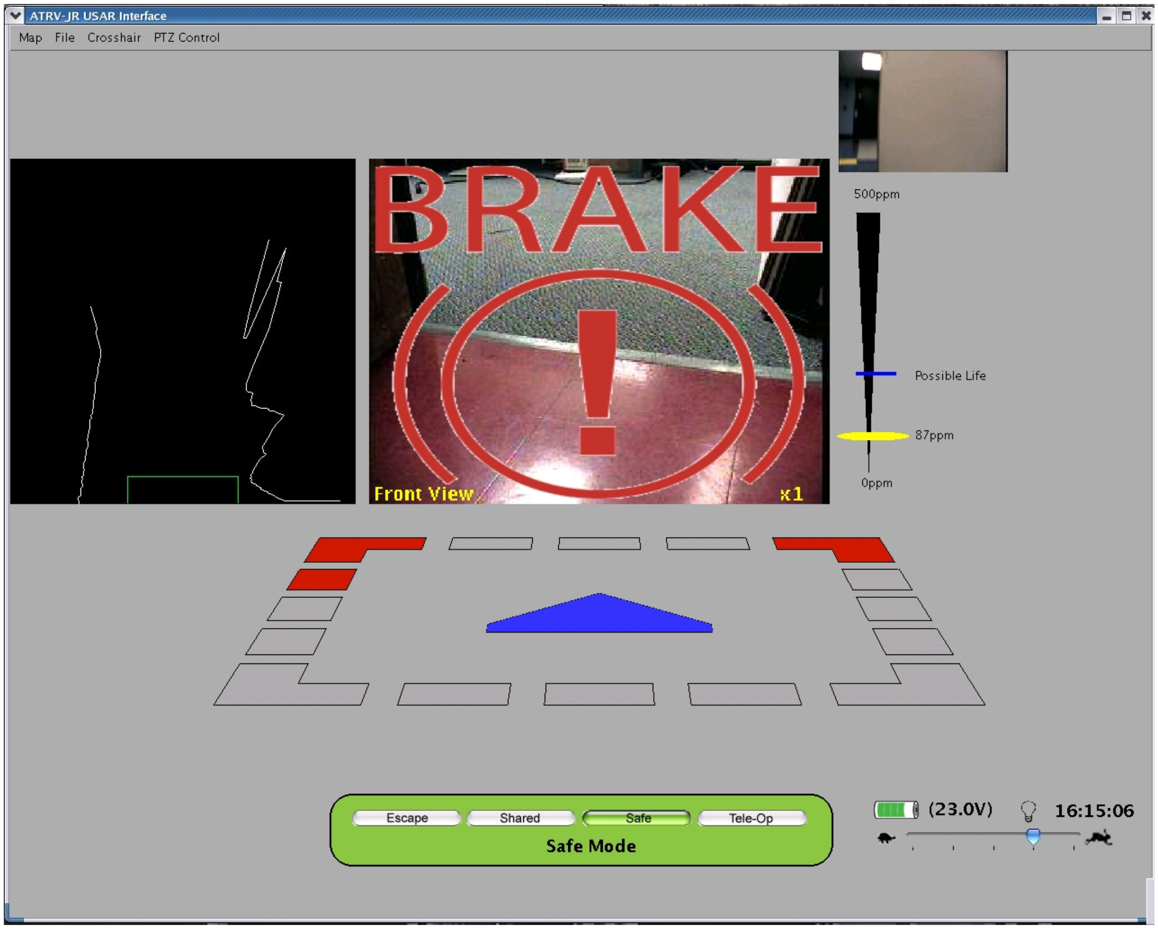


Figure 5-4. One subject expected suggestions to look like the brake indicator shown covering the video display.

Two subjects said the suggestions flashed too quickly and one subject remarked that the Escape Mode suggestion took over the lights suggestion too fast. Two subjects recommended having the suggestions display and then stack up off the video screen. One subject said they should display and then slide out of the picture so they won't "impede" you. One subject recommended taking the suggestions off the video altogether. On the other hand, one subject did not recognize the suggestions at first. He said he was expecting something that looked like the brake "reminder" (Figure 5-4) with red blinking text. He recommended making the suggestions more "flashy" and "prominent." He said he liked the suggestions because they told him what the robot is doing and he described that as "reinforcing."

Another subject described the suggestions this way: "Some are asking, some are telling, some conveying more information, some asking for further input." One subject recommended having the interface speak the text of a suggestion as well as display it. Finally, two subjects commented that the suggestions are a good idea that needs improvement.

5.5.8 Subject Ideas for Additional Suggestions

During the post-run interview subjects were asked for ideas for additional suggestions. In particular, subjects were asked if any suggestions or information was missing that could have helped them in the task. Four subjects wanted additional help with navigation. One subject wanted a suggestion to tell him what to do when "I am stuck in Safe mode." Another subject wanted a suggestion "to stop you when you are going to hit something." A third subject wanted a suggestion to help him go into the tunnel and the fourth subject wanted a "camera popup" to show the obstacle when blocked in Safe mode. Since the sonars blink normally, he said that flashing the sonar ring was not "registering." He said "reinforcing" it with a suggestion "would help a lot."

Three subjects recommended a "weak wireless signal" suggestion. (Subjects ran the robot to the limit of our wireless signal during their runs.) Finally, one subject wanted a suggestion to tell him when "someone is approaching my right rear."

5.6 Correlating Subject Groups and Responses

We looked for a correlation between the subject responses summarized in Table 5-6 and the subject groups we identified in Section 5.4.1 (gender, age, video game experience and computer experience). This analysis did not reveal a correlation between subject groups and their responses. The results are shown in Tables 5-7, 5-8, 5-9 and 5-10.

Age	Sex	Video Gamer	Computer Experience
< 25	M	N	Expert
< 25	M	Y	Guru
< 25	M	Y	Expert
< 25	M	N	Guru
> 25	M	Y	Expert
> 25	M	N	Guru
< 25	F	N	Moderate

Table 5-7. Subjects who found the Turning In Place suggestion helpful.

Age	Sex	Video Gamer	Computer Experience
< 25	M	N	Expert
> 25	M	N	Guru
> 25	M	N	Expert
< 25	M	Y	Expert
< 25	M	N	Guru
> 25	F	N	Expert
< 25	F	N	Moderate

Table 5-8. Subjects who found the Turning In Place suggestion unclear.

Age	Sex	Video Gamer	Computer Experience
< 25	M	Y	Guru
> 25	M	N	Expert
< 25	M	Y	Expert
< 25	M	N	Guru
> 25	M	Y	Expert
> 25	F	N	Expert
< 25	F	N	Moderate
< 25	F	N	Moderate
< 25	M	Y	Guru

Table 5-9. Subjects who found the Escape Mode suggestion helpful.

Age	Sex	Video Gamer	Computer Experience
< 25	M	N	Guru
> 25	M	Y	Expert
> 25	F	N	Expert
< 25	F	N	Moderate

Table 5-10. Subjects who did not find the Switch Camera View suggestion helpful.

6 Machine Learning Analysis

6.1 Machine Learning Problem

The suggestion system was originally created as an autonomy mode suggestion system (Section 4.4). We conceived of autonomy mode suggestions as a way to achieve automatic mode switches and thereby improving the mode selection problem described in Section 1.2. By tracking acceptance of suggestions, we could identify suggestions that are always taken, those that are never taken, and those that are taken under certain conditions. Then we could improve the suggestion system by automatically performing suggestions that are likely to be taken and withholding suggestions that are not likely to be taken. To preserve situation awareness, we could also use an "alert" suggestion to notify the user of the automatic action taken by the suggestion system.

The robot, interface and suggestion systems produce a large quantity of data stored in log files. This data includes sensor and status information, user commands, and suggestion information including acceptance data. We applied a machine learning algorithm to the suggestion acceptance data and learned a classifier that predicts whether a user will take a suggestion. We plan to incorporate the learned classifier into the suggestion system and use it to automatically perform suggestions (see Future Work).

We experimented with neural network and naïve Bayes learning, but decision tree learning provided the highest accuracy results. The decision tree results may be transformed into simple rules that are easy to read, understand and implement in software.

6.2 Decision Tree Learning Using Weka

We used the J48 decision tree learner provided in the Weka learning platform [Witten and Frank 2005]. J48 is Weka's implementation of Ross Quinlan's C4.5 decision tree algorithm. The interested reader is referred to [Mitchell 1997] for a full description of decision tree learning.

Attribute	Range of Values	Suggestion(s)
Sonars (26)	0.0–22.66 (meters)	EM, TP, TS
Volts	0.0-30.0V	BL, BV
Lights State	On or Off	L0, L1
Light Sensor	0-255	L0, L1
Autonomy Mode	0-3	TS, EM, TP, RS, ES, SV
ADR Mode Status	On or Off	SV, CC, TP
Front Camera Pan & Tilt	-50˚-50˚ (pan), -30˚-90˚ (tilt)	CC, TP
Rear Camera Pan & Tilt	-50˚-50˚ (pan), -30˚-90˚ (tilt)	CC, TP
Joystick Translate & Rotate	-100-100	SV, CC, TP, RS, RC
Speed Multiplier	0.0-1.0	RS
Translate Velocity	-5.0-5.0	None

Table 6-1. Attributes used for machine learning.

2-letter code	Suggestion
RS	Reduce Speed
TP	Turning In Place
L0	Lights Off
EM	Escape Mode
CC	Center Camera
SV	Switch Camera View
RC	Restore Cameras
ES	Escape-to-Safe Mode
L1	Lights Off
TS	Teleop-to-Safe Mode
BL	Battery Low
BV	Battery Very Low

Table 6-2. Suggestion codes for Table 6-1.

Table 6-1 shows the attributes that we chose for machine learning. The codes used in Table 6-1 are explained in Table 6-2. The rightmost column of Table 6-1 lists the suggestions that use the corresponding attributes to decide whether to make a suggestion. For example, the Volts attribute is used to decide whether to activate a battery suggestion. None of the suggestions use translate velocity explicitly, but five of the suggestions are triggered by joystick commands so they use translate velocity implicitly. The effect of translate velocity on the learning result is discussed in Section 6.4.

6.3 Experiments and Results

6.3.1 Learning All Suggestions

All of the learning experiments we conducted used a percentage split of 66%, which causes the learning algorithm to use 66% of the total data for training and the remainder for testing. Initially, we used the Weka defaults for J48 learning, which induces conservative post-pruning of the decision tree. This experiment produced a large, complicated tree having 83 total nodes and 42 leaf nodes. This tree had an accuracy of 76%, which means it correctly classified 76% of the testing examples.

Changing two of the default parameters for J48 to induce stronger pruning produced the tree shown in Figure 6-1. This tree is much smaller and simpler, and only 4% less accurate than the tree produced by the initial experiment. The J48 confidence factor was changed from its default value of 0.25 to 0.001 to incur more pruning and the minimum number of instances per leaf was increased from 2 to 11 to shorten the tree further.

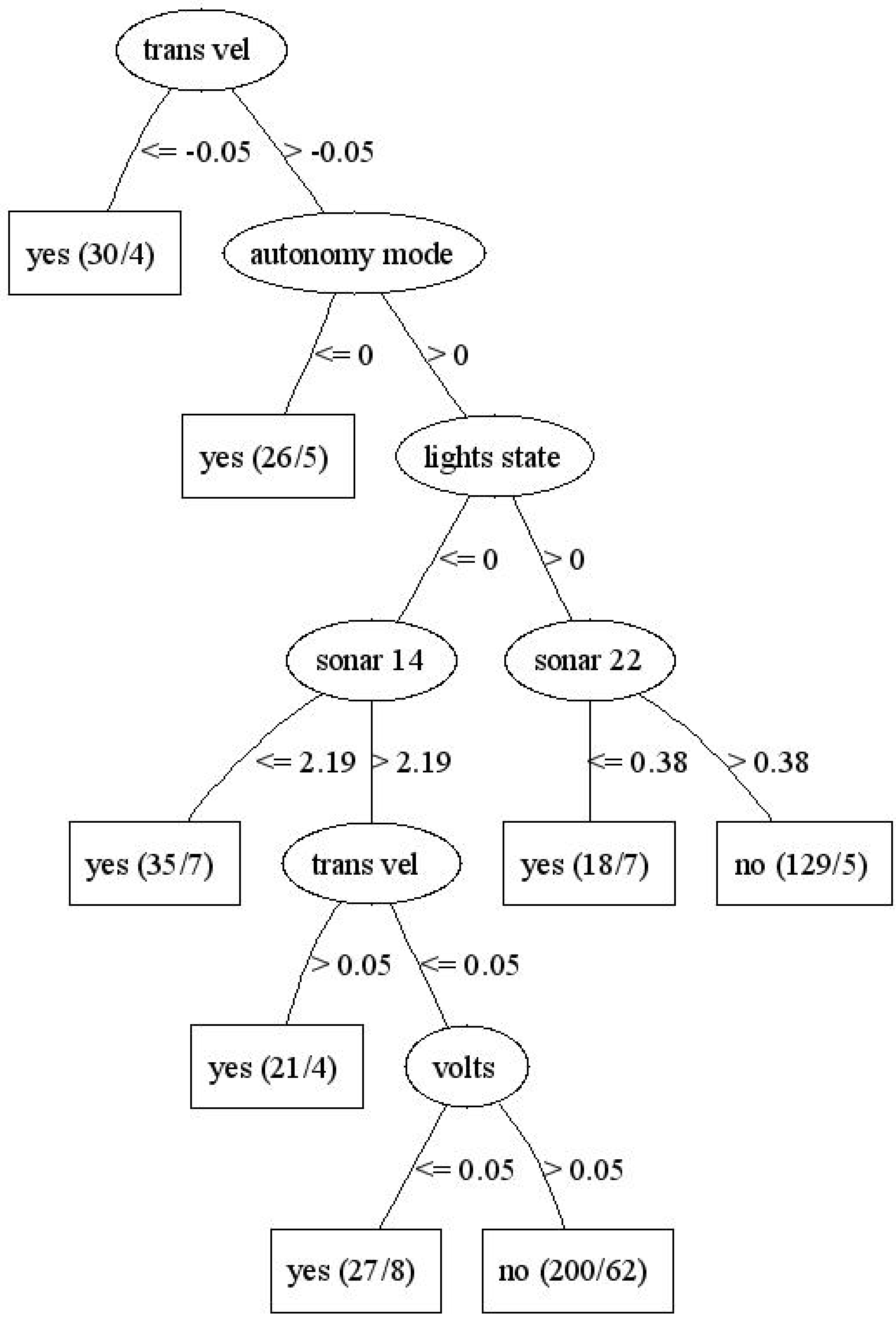


Figure 6-1. A simple decision tree that correctly classified 72% of the testing examples.

Node	0	1	2	3	4	5	6	7	Total
Reduce Speed	1	0	1	0	6	1	0	0	9
Turning in Place	0	0	5	11	46	2	4	9	77
Lights Off	4	4	10	4	13	5	1	107	148
Escape Mode	0	3	5	4	48	0	11	4	75
Center Camera	8	0	3	2	7	1	0	2	23
Switch Camera View	11	0	2	0	14	8	0	2	37
Restore Cameras	0	0	3	1	9	0	0	1	14
Escape-to-Safe Mode	5	18	0	0	5	1	0	4	33
Lights On	1	1	4	4	37	2	0	0	49
Teleop-to-Safe Mode	0	0	2	1	15	1	2	0	21
Total	30	26	35	27	200	21	18	129	486

Table 6-3. Suggestions classified by the decision tree in Figure 6-2. The leaf nodes of the decision tree are numbered from 0 to 7 according to an in-order traversal of the tree.

Table 6-3 shows how the decision tree classified our examples. Since several suggestions use the autonomy mode attribute to trigger suggestions (Table 6-1) and these suggestions represent a large number of the examples used for learning (52%), it is not surprising to see the autonomy mode attribute near the root of the decision tree (Figure 6-1). Similarly, given the large number of lights suggestion examples (31%), it is not surprising to see the lights state attribute high in the decision tree.

A large number of the examples (41%) were classified to node 4. Referring to the “translate velocity” nodes in Figure 6-1, this result implies that users are less likely to take a suggestion when the robot is not moving forward or backward. Some of this can be explained by the Turning in Place suggestion (16% of the learning examples), which is triggered by pure rotation.

Figure 6-2 shows the effect of using the same parameter settings for J48 as for the tree in Figure 6-1, but removing the translate velocity attribute. The accuracy of the predictions for the tree in Figure 6-2 drops to 68%. Substituting the rotate velocity attribute for the translate velocity attribute produces the same result as removing the

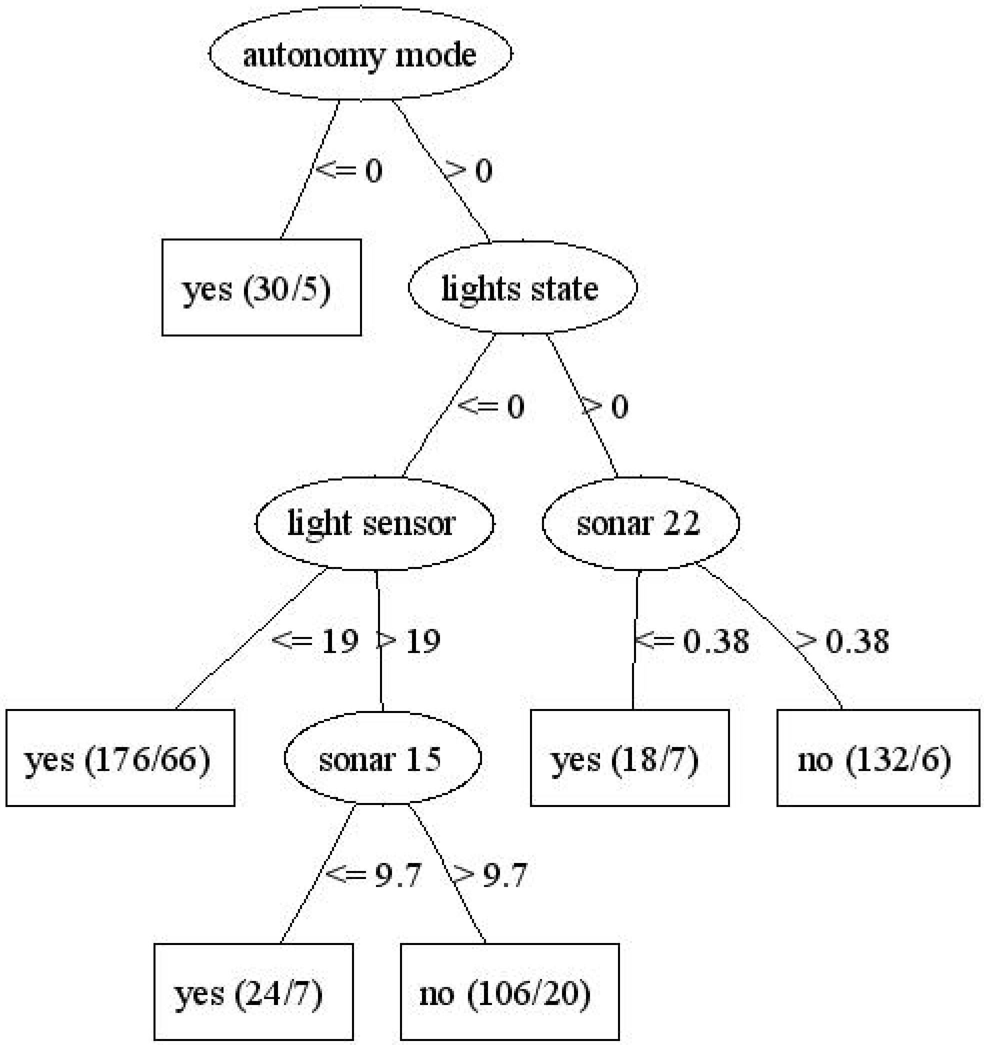


Figure 6-2. Decision tree learned after removing the translate velocity attribute.

translate velocity attribute, which shows that rotate velocity has no effect on learning in this J48 experiment.

6.3.2 Learning Individual Suggestions

Table 6-4 summarizes the learning results for individual suggestions. The table is sorted by accuracy of the result and the "Instances" column does show some correlation between accuracy and the number of examples available for learning. The "Tree" column shows our subjective rating of the learned decision tree; a "Good" tree chose attributes that made sense for that suggestion and a "Bad" tree chose attributes that did

Suggestion	% Accuracy	Instances	Tree
Reduce Speed	25	9	Bad
Center Camera	50	23	Bad
Restore Cameras	80	14	Bad
Escape-to-Safe Mode	83	33	Bad
Switch Camera View	85	37	O.k.
Turning In Place	85	77	Bad
Teleop-to-Safe Mode	87.5	21	Good
Escape Mode	88	75	Good
Lights On	88	49	Bad
Lights Off	92	148	Good

Table 6-4. Summary of learning results for individual suggestions.

not make sense. When the tree does not make sense, we presume the decision tree algorithm learned coincidences in the data. Figures 6-3, 6-4, 6-5 and 6-6 show the trees corresponding to the highest accuracy results.

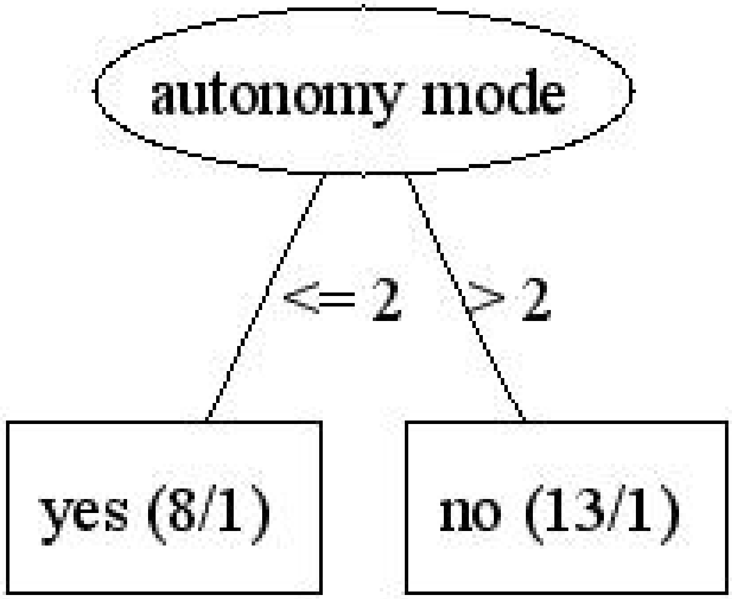


Figure 6-3. Decision tree for the Teleop-to-Safe Mode suggestion.

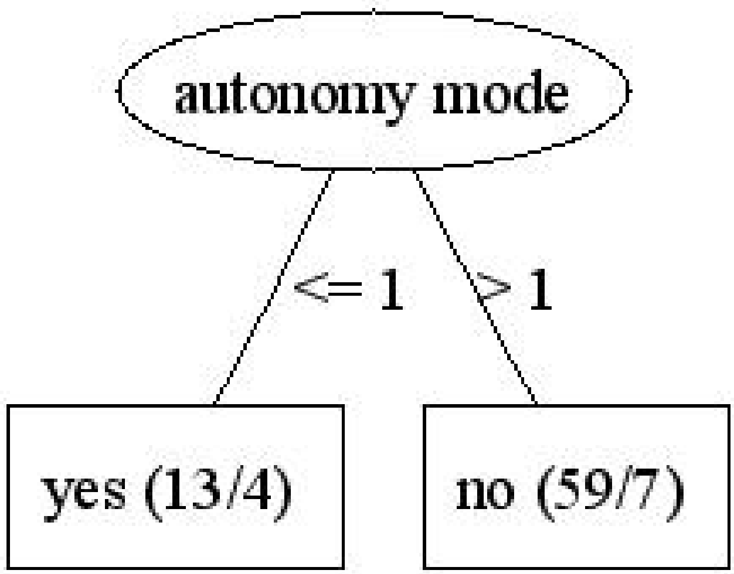


Figure 6-4. Decision tree for the Escape Mode suggestion.

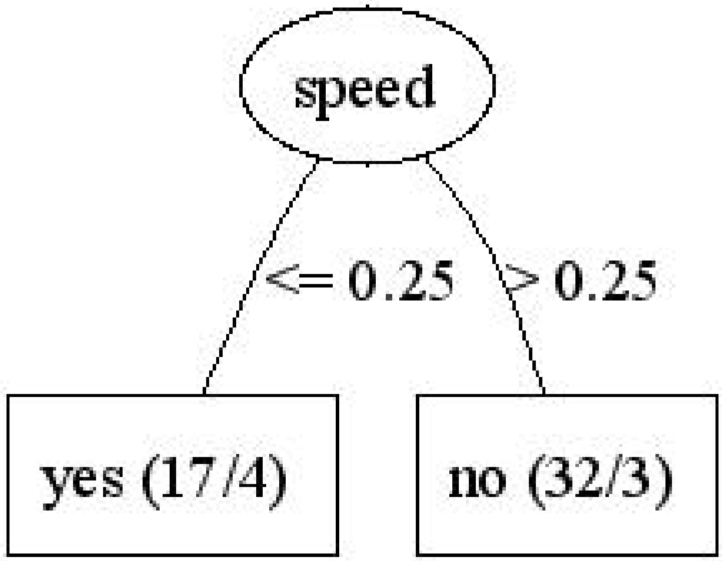


Figure 6-5. Decision tree for the Lights On suggestion. The fact that this tree uses speed to predict whether a Lights On suggestion will be taken appears to be a coincidence in the data.

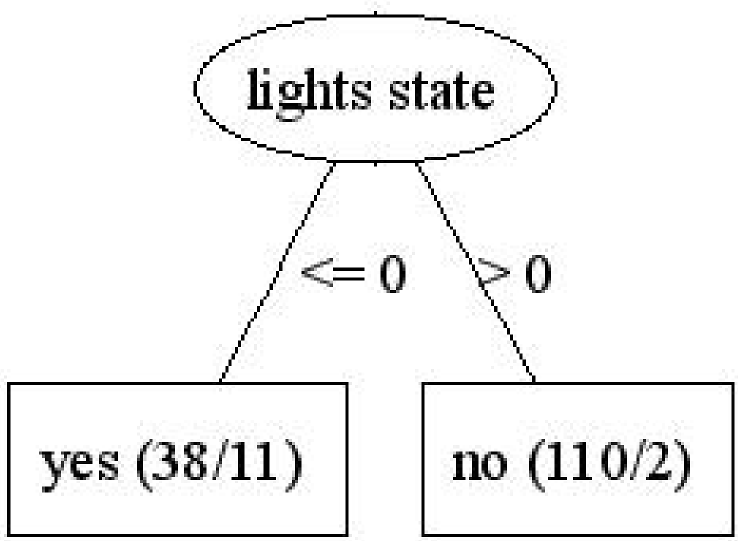


Figure 6-6. Decision tree for the Lights Off suggestion. Although it makes sense that lights state predicts whether this suggestion will be taken, this particular tree is incorrect (predicts “no” when the lights are on) because it is based on a large number of false suggestions that were ignored (implicitly declined) by Subject 1.

6.4 Discussion of Results

In our experiments, we had 189 “yes” examples and 297 “no” examples. Our learned decision tree classified 68% of the examples as “no,” which reflects the uneven split of examples.

Regarding the effect of translate velocity on the learning result, the tree in Figure 6-1 implies that users are more likely to take a suggestion when the robot is moving forward or backward. It could be that users look at the top half of the video image to see the path ahead of the robot. In doing so, users would be more likely to notice the suggestions. Also, users may be more aware of bumping the robot while moving forward and backward, but these hypotheses would need to be verified experimentally.

7 Conclusions and Future Work

7.1 Conclusions about Related Work

Although it does not support immediate or explicit user feedback, we apply machine learning to the acceptance data to analyze and improve the suggestion system. It is even possible using this approach to customize the suggestion system for a class of users.

Like the Microsoft Office Assistant our system provides proactive help, but our system was designed to minimize interrupting and annoying the user, a common complaint about the Office Assistant. In our system, it is possible to continue working on another task while responding to or ignoring suggestions. In our design, we use timing parameters that control the behavior of suggestions. By tuning these parameters, we can make suggestions appear and disappear in a way that makes them noticeable and timely without being too persistent or repetitive.

Our suggestion system is also sensitive to other human factors related to the limited attention of users, an issue mentioned several times throughout Chapter 2. We use a priority and preemption mechanism to present the user with the single best suggestion for the current situation. Furthermore, we deliver the suggestion to the video display (without blocking it) to take advantage of the user's natural locus of attention while performing a remote robot task.

In Section 1.4, we described the suggestion system as "training wheels" for the novice user. In our user study, we observed users who quickly adapted their behavior to match what the safety and camera suggestions were teaching them. If the user continues to operate the robot safely, then those suggestions will not trigger, but if the user becomes careless or forgetful, the suggestion system can provide important safety reminders.

7.2 Conclusions about Qualitative Results

A common theme in the discussion of user comments in Section 5.5 is that opinions regarding individual suggestions (and other features like voice controls and audio feedback) varied widely across users. This disparity is consistent with the finding of a related study (Section 2.7.4). It was also clear in our study that users have different interaction styles and preferences when working through an interface, which is consistent with the studies discussed in Sections 2.4 and 2.6.3. We should also keep in mind that user expectations and behavior are influenced by previous experience with similar applications.

Therefore, in the design of the suggestion system, we need to take different users varying working styles into account. Applying machine learning to the acceptance data can help to adapt the suggestion system for a class of users, but we still need to provide other customizations for the individual user, particularly with regard to preferred input style (voice or key press) and audio outputs (sounds and audible speech).

Several user comments about individual suggestions were confused and contradictory. None of our users had ever operated a robot and some users developed inaccurate mental models of how robot intelligence, the interface and the suggestions worked. When considering user feedback to improve the suggestion system, we need to take these observations into account, not just the number of users who liked or disliked a particular suggestion.

7.3 Conclusions about Machine Learning Results

Despite an imbalanced dataset consisting of more negative than positive examples, we were able to learn a simple decision tree with reasonable predictive power (72% correctly classified instances). We believe that reducing the number of false suggestions by making their triggering conditions smarter and increasing the number of explicit responses by not requiring users to speak both comments and commands (a problem discussed in Section 5.5.5) would produce higher quality data for machine learning. Future experiments to evaluate the machine learning results are proposed in Section 7.5.4.1.

7.4 Design Principles for Suggestions

To summarize our findings, we provide some design principles for suggestions in an HRI task (Figure 7-1). Referring to Figure 7-1, suggestions should be both noticeable and understandable. At the same time, suggestions should be easy to ignore. Suggestions should not force the user to respond explicitly, thereby interrupting the user and violating human etiquette rules for assistants (Section 2.3). Suggestions should not use complicated schemes that require the user to remember the meanings of multiple colors and/or icons (Section 4.4).

- Suggestions should be noticeable
- Suggestions should be understandable
- Suggestions should not interrupt the user's current task
    - Suggestions should be easy to ignore
    - Suggestions should not require an explicit user response
- Suggestions should not use complicated colors and icons
    - Too much for the user to remember and assimilate
    - The user may be color-blind
- Suggestions should include brief text labels
- Suggestions should communicate their meaning well
    - This is the problem
    - This is the solution
- Suggestions should minimize false positives
- Suggestions should act conservatively where safety is concerned
- Suggestions should match user expectations
- Suggestions may incorporate ***customizable*** audio signals
- Provide alternate forms of input for users with special needs or preferences
- Account for differences in user experience
    - Use machine learning to learn how to make suggestions for a class of users

Figure 7-1. Design principles for suggestions in HRI user interfaces.

Suggestions should provide concise textual descriptions that succinctly communicate both the problems and the solutions (Figure 4-4). For example, the battery suggestions in our system may or may not communicate the need to return to the starting position. We should make this idea more explicit in the battery suggestion icons.

The triggering conditions for suggestions should minimize the number of unaccepted suggestions or subjects will begin to ignore suggestions. On the other hand, a suggestion system should err on the side of caution when making suggestions that promote safe navigation. If withholding a suggestion causes the robot to bump into something, the subject may lose confidence in the suggestions. Suggestions should match a user's expectation about its outcome. An example of this is the Escape Mode suggestion, which may or may not cause the robot to escape in the direction wanted or expected by the user.

Suggestions may incorporate audio cues to make suggestions more noticeable and to reinforce a suggestion's graphical representation (Section 7.5.3). Using voice input for responses to suggestions frees the user to do other tasks simultaneously, but some users may not feel comfortable using voice controls so we should provide an alternate means of input for those users. Suggestions should account for differences between users in experience, background, expectations and working styles. It is unlikely, however, that we will be able to create a suggestion system that pleases all users (Section 2.4).

Finally, we recommend using machine learning to predict user responses to suggestions. If a suggestion system makes too many suggestions that a user is likely to ignore, then the user may grow accustomed to ignoring suggestions and consequently miss helpful suggestions.

7.5 Future Work

7.5.1 Improving Existing Suggestions

The most likely candidate for improvement is the Turning in Place suggestion, both in the design of its icon and of its behavior. Based on the experimental results and user feedback about the Turning in Place suggestion, we have decided to completely redesign the Turning in Place suggestion and make it more generally useful by triggering whenever the robot is being steered toward an obstacle, not just when it is being rotated in place. In addition, changing the "Safety First" icon to the camera icon and changing the text to "Show Obstacle?" will clarify that this suggestion is offering to aim the camera(s) at an obstacle. To make this suggestion even more helpful, we could add IR distance sensors at various angles around the robot to determine more precisely where to position the camera(s). The current implementation uses an approximate pan angle because sonars detect obstacles within a 30-degree cone.

The icons for a few other suggestions could be clearer in terms of communicating why the suggestion makes sense and what is being offered to correct some problem. The Lights Off suggestion could benefit from a more explicit icon and making its triggering condition smarter (Section 1.2). Also, referring to the discussion in Section 1.2, it may make sense to reduce the frequency of some suggestions once the user has seen them a few times. On the other hand, it may make sense to increase the frequency of suggestions that a user continues to ignore. The suggestion system includes an unused "urgency" parameter that could be used to make suggestions become more forceful over time. More forceful could include blinking or flashing the suggestion, using an audio alarm, and appearing more and more frequently. Care must be taken to only use "progressive urgency" in critical situations.

A suggestion could also be designed to execute an action automatically for users who repeatedly commit safety violations. For example, if the user bumps the robot into obstacles too many times in Teleoperation mode, the Teleop-to-Safe suggestion could switch to Safe mode automatically and notify the user of the switch. Ultimately, however, the user has control over the system and could switch back to Teleoperation mode.

7.5.2 Additional Suggestions

On our subject questionnaire (Appendix E), we solicited ideas for additional suggestions. Four subjects asked for more specific help with navigation (a mode switch is a more general type of help) and three subjects asked for a "weak wireless signal" warning. One subject asked for help going into the tunnel and several subjects spent a lot of their run time trying to move the robot through a doorway. We could add a wireless strength meter to the interface similar to the CO_2 meter, but the user would have to pay attention to the meter. A suggestion would relieve the user from having to pay much attention to the wireless signal strength and the suggestion would be a more robust indicator because it could use a delay timer to make sure the signal is truly weak and not just oscillating momentarily, making this suggestion similar to the battery suggestions (Section 4.1.7).

Another student in our laboratory created a door detection algorithm, which could be combined with a directional Escape mode to create a "Go Through Door" suggestion. Regarding specific help with navigation, first we want to detect that a user really needs help; otherwise we run the risk of offering help too frequently. We might be able to detect user frustration by noticing that the user continues to issue joystick commands continuously or frequently and that the robot doesn't move or hardly moves within the same period of time, indicating that the user is fighting against the robot's obstacle avoiding behavior. Figuring out how to provide specific steering commands to navigate around an arbitrary configuration of obstacles could be too difficult, but we could always suggest a higher autonomy mode in which the robot performs more of the navigation task.

7.5.3 Incorporating Audio

There are a few opportunities for incorporating audio into the suggestion system. The current suggestion system plays a sound from a popular instant messenger program when a suggestion appears. It is not clear from the experiment whether this additional cue made suggestions more noticeable or whether subjects were able to tune out that sensory channel while interacting with the interface. One possibility is playing a different type of sound for different types of suggestions. For example, one sound could signal a camera suggestion and another could signal a mode suggestion.

One subject recommended using beeps to warn of obstacles, perhaps something like a metal detector that beeps faster when moved closer to metallic objects. One type of audio signal could indicate obstacles and another could indicate other types of warnings such as battery level and wireless signal strength. All of the different combinations of audio signals and graphics that indicate increasing urgency would have to be tested experimentally. It may turn out that remembering different audio signals is too difficult for most users. It is likely that some users will prefer some types of cues and find other types annoying so it is important to provide customizations (Sections 2.2 and 2.4). Finally, the subject who wanted audio signals also wanted spoken suggestions. Based on our experiments, not all users would appreciate this feature.

7.5.4 Further Experiments

7.5.4.1 Machine Learning Experiments

To test the learning results, a future experiment could include two conditions: one suggestion system that uses the learned prediction classifier to make suggestions and one that doesn't. If the machine learning-enhanced suggestion system performs better in terms of suggestions accepted, then we could take this as evidence in support of this approach.

Another machine learning experiment could use prediction classifiers learned for two different classes of users (e.g., experienced and inexperienced users) and testing both on a population of users from one class. For example, both suggestion systems could be tested on a population of experienced users. In this experiment, we would expect the "experienced users" suggestion system to perform better.

7.5.4.2 Quantitative Experiments

We could perform an experiment with two conditions: with the suggestion system and without the suggestion system. In this experiment, we would want to test the hypothesis that task performance is improved in the with-suggestion system condition. Typical quantitative measures of task performance include obstacle hits, number of victims found, and percentage of search area covered.

7.5.4.3 Voice and Audio Experiments

An experiment could be devised to investigate the usefulness of simulated voice recognition using three conditions: voice controls only, voice and another method, and the other method only. The hypothesis in this experiment might be that users respond to more suggestions when voice controls are available. For an audio experiment, the conditions could be: audio only, audio and graphics, graphics only. For this experiment, the hypothesis might be that audio reinforces the graphical cue and makes suggestions more noticeable. An option for spoken suggestions could be provided, but it should be easy to turn on and off dynamically.

8 References

Michael Baker, Robert Casey, Brenden Keyes and Holly A. Yanco. Improved Interfaces for Human-Robot Interaction in Urban Search and Rescue. *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, October 2004.

Michael Baker and Holly A. Yanco. Autonomy Mode Suggestions for Improving Human-Robot Interaction. *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, October 2004.

David Bruemmer, Donald Dudenhoeffer, and Julie Marble. Dynamic autonomy for urban search and rescue. *AAAI Mobile Robot Workshop*, Edmonton, Canada, August 2002.

David Bruemmer, Julie Marble, and Donald Dudenhoeffer. Mutual initiative in human machine teams. *IEEE Conference on Human Factors and Power Plants*, Scottsdale, AZ, September 2002.

Jill L. Drury, Jean Scholtz, and Holly A. Yanco. Awareness in Human-Robot Interactions. In *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, Washington, DC, October 2003.

Festival Speech Synthesis System. <http://www.cstr.ed.ac.uk/projects/festival/>. Accessed on June 5, 2006.

Terrence Fong, Charles Thorpe, and Charles Baur. Robot, asker of questions. *Robotics and Autonomous Systems 42(3-4)*, March 2003.

Terrence Fong, Charles Thorpe, and Charles Baur. Robot as partner: vehicle teleoperation with collaborative control. *Proceeedings Workshop on Multi-Robot Systems*, Naval Research Laboratory, Washington, D.C. March 2002.

Eric Horvitz. Principles of mixed-initiative user interfaces. *Proceedings CHI'99, ACM SIGCHI Conference on Human Factors in Computing Systems*, May 1999.

Andrew Howard. <http://robotics.usc.edu/~ahoward/pmap/index.html>. Accessed on June 5, 2006.

Henry Lieberman. Interfaces that Give and Take Advice. In *Human-Computer Interaction for the New Millenium*, John Carroll, ed., ACM Press/Addison-Wesley, pp. 475-485, 2001.

Henry Lieberman and Ted Selker. Agents for the User Interface. In *Handbook of Agent Technology*, Jeffrey Bradshaw, ed., MIT Press, 2003.

Julie Marble, David Bruemmer, Douglas Few. Lessons learned from usability tests with a collaborative cognitive workspace for human-robot teams. In *Proceedings 2003 IEEE International Conference on Systems, Man and Cybernetics*, Washington, D.C., October 2003.

John McCarthy. Programs with common sense. In V. Lifschitz, *Formalization of Common Sense, Papers by John McCarthy.* Ablex, 1959.

Tom Mitchell. Machine Learning. McGraw Hill, 1997.

Jean Scholtz, Jeff Young, Jill L. Drury, and Holly A. Yanco. Evaluation of Human-Robot Interaction Awareness in Search and Rescue. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans, April 2004.

Luke Swartz. Why People Hate the Paperclip: Labels, Appearance, Behavior, and Social Responses to User Interface Agents. *Honors Thesis for Symbolic Systems Program*, Stanford University, Professor Clifford Nass, Advisor. June 12, 2003.

Ian H. Witten and Eibe Frank. Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

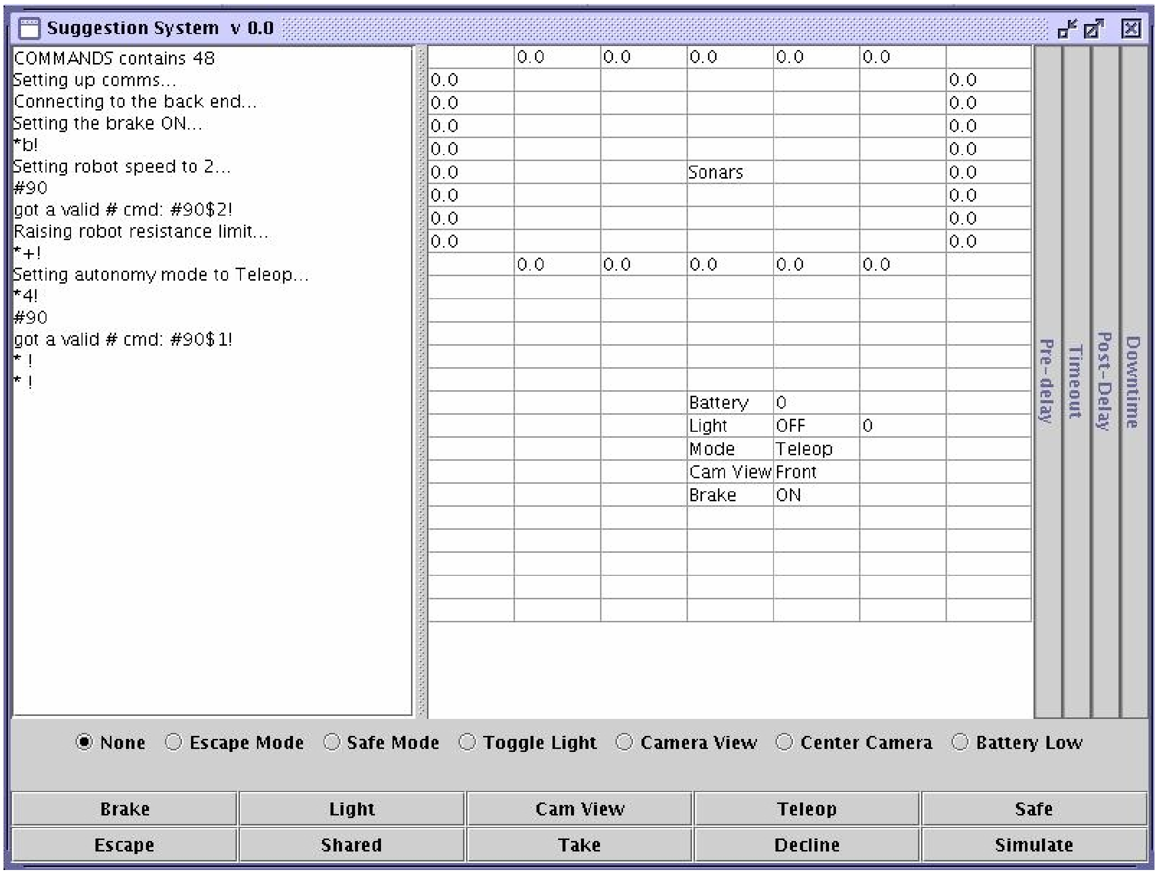
Jun Xiao, John Stasko and Richard Catrambone. An Empirical Study of the Effects of Agent Competence on User Performance and Perception. *Proceedings of AAMAS '04*, New York, NY, July 2004, pp. 178-185.

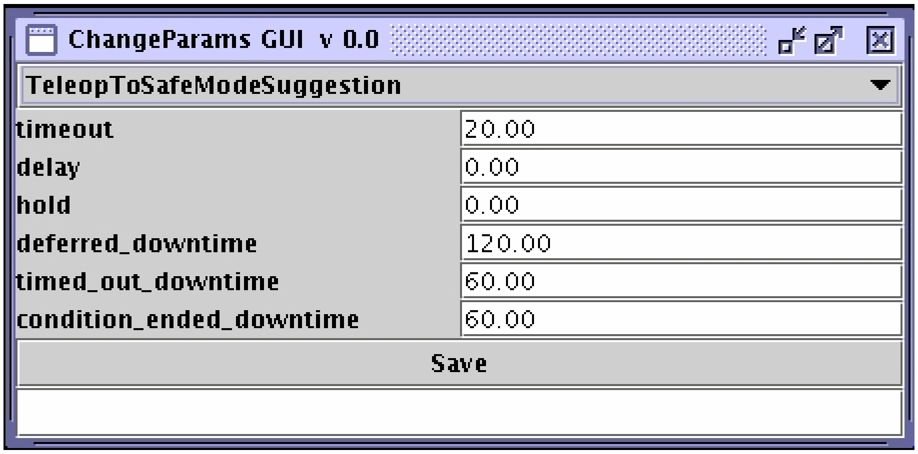
Holly A. Yanco and Jill Drury. "Where Am I?" Acquiring Situation Awareness Using a Remote Robot Platform. *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, October 2004.

Holly A. Yanco, Jill L. Drury, and Jean Scholtz. Beyond Usability Evaluation: Analysis of Human-Robot Interaction at a Major Robotics Competition. *Journal of Human-Computer Interaction*, Volume 19, Numbers 1 and 2, pp. 117-149, 2004.

A Timer Visualizations

This application was created for visualizing the timers that determine how a suggestion behaves. At the right side of the interface are four vertical progress bars that allow a developer to watch the timers in real time and verify that a suggestion is operating correctly.





This application was created for adjusting the timing parameters for the suggestions dynamically. The empirically determined values are saved to a file, which the suggestion system uses to initialize the suggestion parameters.

B Testing Materials

Robot Testing Script

Created 12-3-2005 by Michael Baker

Robot Features:

- distance (obstacle) sensors
- lights
- front and rear cameras
- how it steers

Interface Features:

- front and rear camera displays
- what the camera crosshair means
- explain ADR mode
- how to turn lights on/off
- explain sonar ring and color meanings
- how to switch modes
- what the modes mean (show cheat sheet)
- show speed, lights, battery level indicators

Joystick Controls:

- how to drive using trigger safety
- how to pan/tilt/zoom camera
- how to home camera
- how to switch camera view
- how to adjust speed
- how to turn brake on/off

Suggestion System:

- what is a suggestion
- what a suggestion looks like
- how to accept, reject or defer a suggestion by simulated voice recognition

Suggestion System Controls:

Say **"TAKE IT"** to take a suggestion.
Say **"NOT NOW"** to make a suggestion go away for awhile.
Say **"GO AWAY"** to make a suggestion go away forever.

Keyboard Controls:

F1 = Escape mode
F2 = Goal mode
F3 = Safe mode
F4 = Teleop mode

Caps Lock = Lights on/off

Participant No. ____ Updated 5 November 2005

Human-Robot Interaction Experiment: Pre-Experiment Questionnaire

Thank you for participating in our experiment. Please answer the following questions to help us better understand some aspects of your background that may be relevant to our experiment.

1. Please tell us your age. _____

2. Have you worked in search and rescue? If "yes," what was your role and how many years have you been involved in search and rescue?

3. What types of computers do you normally use? Check all that apply, and state whether you've used any of these for over 5 years.

___ Macintosh	Used over 5 years? Yes ____ No ____
___ PC	Used over 5 years? Yes ____ No ____
___ UNIX or LINUX	Used over 5 years? Yes ____ No ____
___ Other: _________________	Used over 5 years? Yes ____ No ____

4. How would you rate your level of computer expertise? Please choose the description that represents the closest match.

   ___ Casual: primarily occasional email and web-browsing
   ___ Moderate: I do a lot of my regular work or leisure activities on a computer
   ___ Expert: I troubleshoot and upgrade applications or operating systems
   ___ Guru: others come to me for solving their computer problems

5. Have you ever used robots?

   ___ No ___ Yes If yes, please describe.

6. Have you used remote control cars or aircraft?

   ___ No ___ Yes If yes, please describe.

7. Do you play video games?

   ___ No ___ Yes If yes, please describe.

8. How would you describe your attitude towards using robots in search and rescue?

Participant No. ____
Robot Code ____
Run No. ____

Post-Run Questions

1. For this robot, please circle the number that most closely expresses your opinion about the controls:

Hindered me	1	2	3	4	5	Helped me tremendously
Extremely difficult to use	1	2	3	4	5	Very easy to use
Irritating to use	1	2	3	4	5	Pleasant to use

2. What interface features were most useful for you in performing this task?

3. What interface features were least useful?

4. Would you like to make any suggestions to the developers of the interface software?

Participant No. ____
Robot Code ____
Run No. ____

Suggestion System Questions

(Tailor conversation to answers given by subject.)

1. Did you find the suggestions helpful?

2. Which suggestions were most helpful?

3. Which suggestions were least helpful?

4. Did find anything negative about (any of) the suggestions? Please explain.

5. Were all of the suggestions understandable?

6. If not, which suggestions did you find confusing?

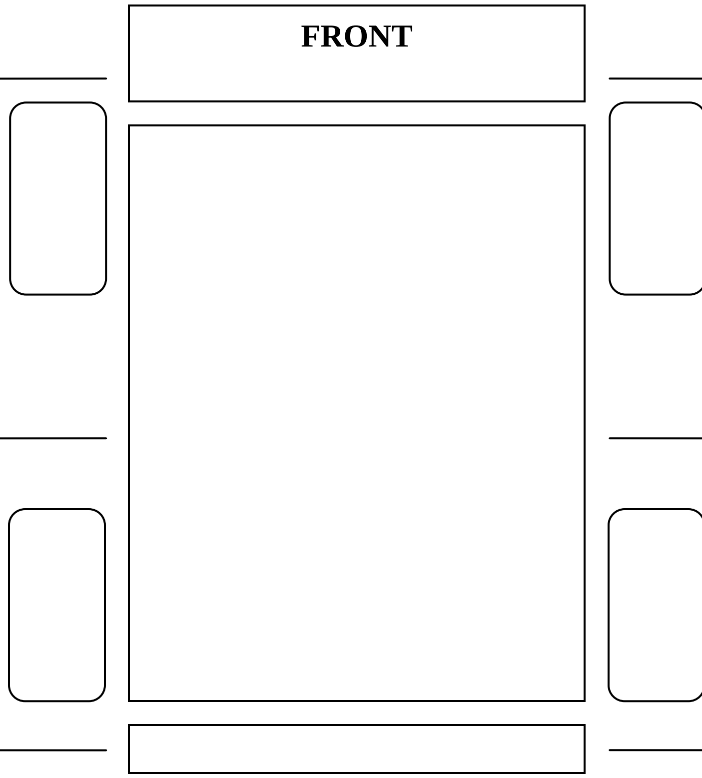
7. If you could, what would you change about (any of) the suggestions?

8. Do you have any other comments about (any of) the suggestions?

9. Ideas for additional suggestions?

Critical Events Record Sheet
Updated 8/10/2005

Subject # **Start Time:**
Run # **End Time:**



Event	Time	Type	Direction	Comment
1		H V	F B ADR TIP	
2		H V	F B ADR TIP	
3		H V	F B ADR TIP	
4		H V	F B ADR TIP	
5		H V	F B ADR TIP	
6		H V	F B ADR TIP	
7		H V	F B ADR TIP	
8		H V	F B ADR TIP	
9		H V	F B ADR TIP	
10		H V	F B ADR TIP	
11		H V	F B ADR TIP	
12		H V	F B ADR TIP	