Collaborative Control: A Robot-Centric Model for Vehicle Teleoperation

Terrence Fong and Charles Thorpe
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania USA
{terry, cet}@ri.cmu.edu

Charles Baur
Institut de Systèms Robotiques
L'Ecole Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
charles.baur@epfl.ch

Abstract

Telerobotic systems have traditionally been designed and operated from a human point of view. Though this approach suffices for some domains, it is sub-optimal for tasks such as operating multiple vehicles or controlling planetary rovers. Thus, we believe it is worthwhile to examine a new approach: *collaborative control*. In this robot-centric teleoperation model, instead of the human always being "in charge", the robot works as a peer and makes requests of the human. In other words, the human is treated as an imprecise, limited source of information, planning and capability, just as other noisy system modules. To examine the numerous human-machine interaction and design issues raised by this new approach, we are building a vehicle teleoperation system using collaborative control. In this paper, we present our current design and implementation.

Introduction

Human as Controller

Telerobotics has traditionally been human-centric. Since telerobotics evolved directly from other human controlled devices, this approach seems only natural. Whatever the system and regardless the operational model the paradigm has always been *human-as-controller*: the human receives information, processes it, and selects an action. The action then becomes the control input to the system. For telerobotics, however, this human-machine relationship often proves to be inefficient and ineffective.

The first problem with human-as-controller is that it consumes valuable human resources and may awkwardly bind the system's capability to the operator's skill. These difficulties are particularly acute with direct teleoperation. Some of the common problems are: operator handicap (skill, knowledge, training), sensorimotor limits (reaction time, decision speed), cognitive and perceptual errors (judgement, misclassification), and physical difficulties (nausea, fatigue) (Ferrel 1967, Sanders 1993, Sheridan 1992).

The second problem with human-as-controller is that the quality of the human-machine connection significantly impacts performance. An effective *operator interface* (or *control station*) is critical for conveying information and feedback to the operator. Inadequate displays, inappropriate modeling, and inefficient control inputs contribute to operator error (Murphy 1996). Additionally, if the operator and robot are widely separated, communications may be affected by noise or signal transmission delays. Delay is particularly insidious because it can make direct teleoperation impractical or impossible (Sheridan 1993).

The third manner in which human-as-controller causes problems is the imbalance in roles (human as supervisor/master, robot as subordinate/slave). Whenever the human is "in-the-loop", he has reduced capacity for performing other tasks. Additionally, since the robot is under human control, the system halts whenever the robot waits for direction. Finally, the imbalance in operational dialogue (human commands, robot responds) means the relationship between human and robot is forever static.

Vehicle Teleoperation

Consider the task of remotely driving a robotic vehicle. The basic problems are: figuring out where the vehicle is, determining where it should go, and getting it there. These problems can be difficult to solve, particularly if the vehicle operates in a hazardous environment with poor communications. This is common with exploration robots (Mishkin 1997, Hine 1995) and unmanned ground vehicles (Shoemaker 1990). The difficulty increases when we add additional constraints such as operator variation, multiple vehicles, high delay, and moving (or malignant) hazards.

It has been shown that humans in continuous, direct control limit vehicle teleoperation performance. (McGovern 1988) conducted a study of rate-controlled ground vehicles and reported operator problems including slow driving, imprecise control, loss of situational awareness, poor attitude and depth judgement, and failure to detect obstacles. McGovern concluded that many vehicle failures (collision, roll over, etc.) were traceable to these operator problems.

Yet, even if a telerobotic vehicle has autonomous capabilities (route following, obstacle avoidance, etc.) and can be operated in a supervisory mode, fixed control flow may still restrict system efficiency. The Sojourner rover on Mars, for example, was operated via high-level command scripts received from earth-based operators (Mishkin 1997). Since these scripts were manually created and uploaded once per day, the overall system throughput (science return) was severely limited.

Collaborative Control

As we have seen, there are numerous problems and limitations arising from the conventional human-as-controller model. Since we would like to construct teleoperation systems which are able to operate flexibly and robustly in difficult environments, in spite of poor communications, and with high performance regardless of variations between operators, we need a new approach. Therefore, instead of human-as-controller, we propose the following:

> Teleoperation can be significantly improved by modeling the human as *collaborator* rather than *controller.*

In this new *collaborative control* model a human operator and robot are peers who work together, collaborating to perform tasks and to achieve common goals. Instead of a supervisor dictating to a subordinate, the human and the robot engage in constructive *dialogue* to exchange their ideas and resolve their differences. Instead of the human being completely "in control", the robot is more equal and can treat the human as an imprecise, limited source of planning and information, just like other noisy system modules.

An important consequence of collaborative control is that the robot can decide how to use human advice: to follow it when available and relevant; to modify (or ignore) it when inappropriate or unsafe. For example, if the robot is operating autonomously and has problems, it can ask the operator "what should I do?" If the human is capable of responding (and can do so in a timely fashion), then the advice will be used. However, if the advice is late (e.g,. due to communication delay) or unsafe (e.g., "drive off that cliff"), then the robot may view the advice with skepticism and disagree.

In short, when we construct a teleoperation system, rather than designing only from a human-centric viewpoint (human-as-controller), we also consider issues from a robot-centric perspective (human-as-collaborator). This is not to say that the robot becomes "master": it remains a subordinate following higher-level strategy (goals and tasks) set by the human. However, with collaborative control, the robot has more freedom in execution and is able to better function if the operator is distracted, inattentive, making errors, etc. As a result, teleoperation becomes more adaptable, more flexible and better able to accommodate varying levels of autonomy and interaction.

The term *collaborative control* is quite apt. We use it because it is analogous to the interaction between human collaborators. Specifically, when we engage in collaboration, we encourage each collaborator to work with others towards a common goal. We also allow each collaborator to take self-initiative and to contribute as best he can. At the same time, however, we leave room for discussion and negotiation, so that potential solutions are not missed.

Collaborative control raises many human-machine interaction and system design issues. In particular, when we build a collaborative control system, we must consider how to support human-machine dialogue, how to make the robot more aware, how to handle human-machine interaction, how to design the user interface, and how to handle dynamic control and data flow.

Related Research

Supervisory Control

Supervisory control emerged from research on earth-based teleoperation of lunar vehicles (Ferrel 1967). The term *supervisory control* is from the analogy between a supervisor's interaction with subordinate staff and an operator's interaction with a robot (Sheridan 1992). With supervisory control, an operator divides a problem into a sequence of tasks which a robot can achieve on its own. Supervisory control is used most often for telemanipulation (e.g., Blackmon 1996), though some work in vehicle teleoperation has been done (Wettergreen 1995, Lin 1995, and Stone 1996).

In a sense, supervisory control models military structure: hierarchical, rigid control flow, supervisor "in charge" and subordinates restricted in what they can do. Collaborative control more closely resembles a research group. Although collaborative control has hierarchy, it's control is more flexible and dynamic. Furthermore, each collaborator has greater freedom to take the initiative and to lead.

Multi-operator and Cooperative Teleoperation

In multi-operator teleoperation, multiple operators share or trade control. (Cannon 1997) describes the use of "virtual tools" for telemanipulation. In his system, operators use these tools to define key actions at a supervisory level. A networked interface allows multiple operators to share control. Cannon refers to this interaction as "collaborative control" since multiple operators collaborate to effect control.

Cooperative teleoperation, also known as *teleassistance*, tries to improve teleoperation by supplying aid (data filtering, decision-making tools, etc.) to the operator in the same manner an expert renders assistance. For example, (Murphy 1996) describes a teleassistance system which combines a limited autonomy robot architecture with a knowledge-based operator assistant. During teleoperation, this system provides "strategic assistance" so that the operator and robot can cooperate in cognitively demanding tasks.

Human-Robot Control Architectures

Although most robot control architectures are designed for autonomy, some have addressed the problem of mixing humans with robots. One approach is to directly incorporate humans into the design, i.e., as a system element. DAMN, for example, is a behavior-based architecture in which individual modules vote on possible actions (Rosenblatt 1995). Command arbitration allows modules as disparate as autonomous safety behaviors and teleoperation to coexist.

Another approach is the use of prioritized control, in which operator commands may be overridden by autonomous modules. The best-known example of this is NASREM, which explicitly incorporated an operator interface into a layered, hierarchical control system (Albus 1987). More recently, the concept of *safeguarded teleoperation* has been used to enable novices to teleoperate a planetary rover (Krotkov 1996).

Vehicle Teleoperation Systems

During the past twenty years, the majority of work in vehicle teleoperation has centered on rate-controlled systems for use in hazardous environments. These remote vehicles (e.g., underwater ROV's) are typically operated with single-mode "inside-out" control: the operator controls the vehicle's rotation and translation rates via hand-controllers and receives feedback from on-board video cameras and sensors. (McGovern 1988) reports on work with a fleet of ground vehicles, ranging from small indoor robots to large outdoor military automobiles.

More recently, vehicle teleoperation systems have emphasized the use of multi-modal operator interfaces and supervisory control. Multi-modal interfaces provide the operator with a variety of control modes (individual actuator, coordinated motion, etc.) and displays (text, visual, haptic). Supervisory control is often used to compensate for a variety of problems, most often poor communications. Recent systems include: Dante II (Fong 1995), STRIPE (Kay 1997), VEVI (Hine 1995), Nomad (Wettergreen 1996), and Sojourner (Mishkin 1997).

Research Issues

The collaborative control model creates a number of fundamental research issues. In particular, it significantly changes how we design and operate telerobotic systems. Collaborative control forces us to reexamine teleoperation: to approach it not only from human viewpoint, but also from a robotic perspective.

Dialogue

In order to use collaborative control, we must build a system with the capacity for dialogue. That is, we need to enable the operator and the robot to converse, to exchange opinions, and to negotiate. Each one should be able to say "here is what I think", to ask "what do you think?" and to interpret the response. For example:

robot:	Based on *a priori* map data, I think I should go to point 'B' directly.
operator:	No, it is better to go to point 'A' first and then to point 'B'.
robot:	I think the path to 'A' is blocked. Look at this image and tell me if you agree.
operator:	Sorry, I'm too busy to look at it.
robot:	In that case, I'm going to wander until I find an unobstructed path to 'B'. When you have time, I'll tell you what happens.

Though simple, there is a strong dynamic at work here. Initially, the robot seeks advice and the human provides a response. Since the response is unhelpful, the robot offers additional detail and asks for clarification. By this point, the human has become unavailable. So, the robot takes the initiative and suspends the dialogue until a later point in time. In this scenario, we see the human and the robot working together as peers, not supervisor and subordinate.

In short, good dialogue is two-way: it requires each party to understand to what the other is saying and to speak so the other can understand. To an extent, traditional teleoperation has dialogue (i.e., the feedback loop), but the conversation is limited. Dialogue offers the potential for richer, more flexible teleoperation. However, it creates questions such as: "When should the robot 'speak'?", "How does the robot format its queries?" and "What language features are required for effective communication?"

Awareness

Under collaborative control, the robot is free to use the human such that its needs are best satisfied (e.g., making queries in different ways and frequencies). But, as a consequence, the robot needs to have *awareness*: it must be aware of its capabilities as well as those of the human. Specifically, the robot has to be able to adapt to different operators and to adjust the dialogue as needed. For example, it should ask questions based on the operator's capacity to answer (i.e., a geologist and a roboticist differ in expertise). Similarly, it should handle information received from a novice differently than that received from an expert.

However, awareness does not imply that the robot needs to be fully sentient. It merely means that the robot be capable of detecting limitations (in what it can do and what the human can do), judging the quality of information it receives, and recognizing when it has to solve problems on its own. The research questions are: "At what level does the robot need to model the human?", "How does robot adapt to different operators?", and "How does the robot handle conflicting or unreliable information?".

Human-Machine Interaction

When we build a collaborative control system, the traditional roles of operator and robot change. Instead of a subordinate awaiting direction, the robot is a co-worker seeking dialogue. Though the human may make requests, there is no need for the robot to strictly obey them. This frees the human from performing continuous control or supervision. If the human is available, he can provide direction. But, if he is not, the system can still function. This allows use of human perception and cognition without requiring time-critical response.

Collaborative control also changes the way we view telerobotics. In conventional systems, there is an underlying notion of *robot as tool*: the robot extends human sensing and acting. With collaborative control, however, the robot is more equal, more *robot as partner*. Though the robot may ask for approval, it is not required to do so. Thus, to understand human-machine interaction with collaborative control we must answer "How do we decide who is in charge at a given moment?", "How does the robot recognize when the human is unavailable or unhelpful?" and "How does the human-robot relationship change over time?"

User Interface Design

In traditional teleoperation, the user interface serves only the operator: displays provide information for human decision making, mode changes are user triggered, etc. In a collaborative control system, however, the user interface also has to support dialogue and to serve the robot.

Most modern user interfaces are designed with user-centered methods. In user-centered design, the basic goal is to support human activity: to enable humans to do things faster, with fewer errors, and with greater quality (Newman 1995). A variety of human performance or usability metrics (speed of performance, error rate, etc.) are typically used to guide the design process.

We can use this approach to develop a collaborative control interface. For example, we can design dialogue support to maximize usability (e.g., allow the user to respond by drawing on maps or images). It is clear, however, that a strictly user-centered approach has limits. If we focus on the user, the interface will not support the robot. Thus, collaborative control raises the questions: "How useful is user-centered design?", "How do we consider the robot's needs?" and "Should the robot control the user interface?"

Control and Data Flow

Collaborative control adds new constraints to system design. In traditional teleoperation, the flow of control is clear: the operator controls the robot's actions. Though he may share or trade control, the operator retains ultimate authority. Collaborative control, however, allows control to be negotiated. It also allows the robot to consider commands as approximate or noisy. Thus, a collaborative control system must have *command arbitration*: a means for deciding which actions to take over the short and long term.

Another issue concerns the handling of robot questions. Under collaborative control, robot modules may ask multiple questions of the human at the same time. These questions may have different forms (text, image, etc.), priority, validity (temporal, spatial) and difficulty. Thus, a collaborative control system must have *query arbitration*: a mechanism for choosing which questions to ask based on both immediate (local) needs and overall (global) strategy.

A related issue is what to do with invalid advice. Consider the situation in which the human answers an outdated query (i.e., the robot has already made a decision by the time the human responds). Should the robot ignore the answer or should it reconsider its action? The problem is that outdated advice may be hard to distinguish from unsafe advice. Thus, if we allow a range of users, how do we cope with the varying speed and quality of information?

Lastly, collaborative control requires flexible data handling. Since humans and robots operate differently, a collaborative control system must provide data in a variety of ways. For example, the human may decide that the terrain is flat by looking at an image; the robot may decide it is rough using proprioception. To have meaningful dialogue ("why do you say it's rough when it's flat?"), both need to be able to exchange and present their data in a coherent manner.

System Design

To examine the numerous human-machine interaction and system design issues raised by this new approach, we are building a vehicle teleoperation system based on collaborative control. The following sections describe our current system design and implementation.

Architecture

Our current collaborative control architecture is shown in Figure 1. We use a message-based framework to connect system modules and to distribute information. Each module is designed to perform a specific high-level function. In the style of DAMN (Rosenblatt 1995), any task-achieving system element is considered a *behavior*, regardless of complexity or time constant. Thus, a module may be a behavior or may contain multiple behaviors.

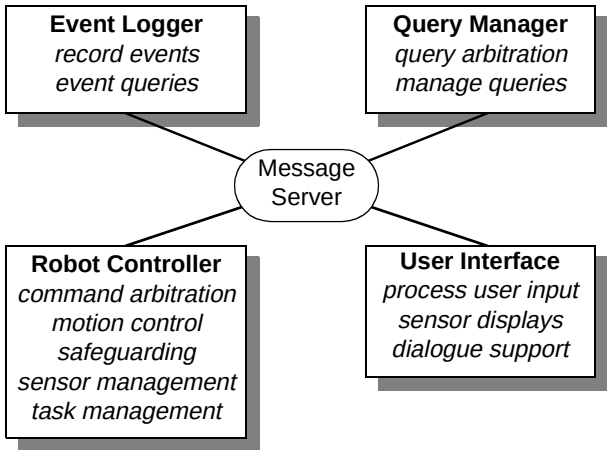

Figure 1. Collaborative control architecture

The primary modules and their respective functions are:

Event Logger. The *Event Logger* allows the operator to review what has happened and keep track of the dialogue. These functions enable an operator who is not performing continuous control to understand what has transpired in his absence and to maintain situational awareness.

Query Manager. Decides when and what queries to ask the operator. All modules send their questions to the *Query Manager*, which performs query arbitration to decide which (and in what order) are forwarded to the operator.

Robot Controller. Performs all robot control functions including motion control, sensor management, command arbitration, localization and low-level behaviors (e.g., reactive safeguarding). The *Robot Controller* is responsible for coordination and monitoring of robot tasks, environment modeling and simple navigation (e.g., path tracking).

User Interface. A non-intrusive interface which enables the operator to efficiently control the robot and which supports dialogue. The *User Interface* is described in greater detail in a following section.

Dialogue

As we discussed, dialogue offers the potential for richer, more flexible teleoperation. Effective dialogue, however, does not require a full and varied language, merely one which is suited to the task and which precisely communicates information. Thus, in our work, we are using dialogue only to address vehicle mobility issues (navigation, obstacle avoidance, etc.) and we avoid natural language (which is often ambiguous and imprecise).

In our collaborative control system, dialogue results from messages exchanged between the user and the robot. We classify messages as shown in Table 1. *Robot commands* and *user statements* are uni-directional (i.e., no acknowledgment is expected). Query and response messages are paired: a query is expected to elicit a subsequent response (though the response is not required or guaranteed).

Table 1. Dialogue message classes

User → Robot	Robot → User
robot command (command for the robot)	user statement (information for the user)
query-to-robot (question from the user)	query-to-user (question from the robot)
response-from-user (query-to-user response)	response-from-robot (query-to-robot response)

Our current system design contains approximately thirty dialogue messages, which are intended to promote a limited, but interesting human-machine conversation. A selection of these messages is given in Table 2. Note that Table 2 only describes the content of messages (what is said), and not the expression (how it is conveyed). Message expression is described in the *User Interface* section.

All dialogue messages share common properties. Each message (whether sent by the human or a robot) is marked with a *priority* level and a *validity* period (expiration time). The *Query Manager* uses these properties to perform query arbitration. Additionally, each message contains content-specific data (text, image, etc.). Each system module (e.g., *User Interface*) interprets this data as appropriate.

Table 2. Example vehicle mobility dialogue messages

Category	Message
query-to-robot	How are you? Where are you?
response-from-robot	bar graphs (How are you?) map (Where are you?)
user query	How dangerous is it this (image)? Where do you think I am (map)?
response-from-user	“8” (How dangerous is this?) position (Where do you think I am?)
robot command	rotate to X (deg), translate at Y (m/s) execute this path (set of waypoints)
user statement	I think I’m stuck because my wheels spin Could not complete task N due to M

User Interface

We designed our user interface to be non-intrusive: to enable efficient human-machine interaction and minimize use of human resources (attention, cognition, etc.) Our interface emphasizes usability (to support a wide range of users), sensor-fusion based displays (to facilitate comprehension), and rapid user input (to minimize response time).

Our interface layout is shown in Figure 2: it has three modes (*control*, *query*, and *messages*) and a mode-specific interaction area. Each mode supports two dialogue message classes. Partitioning the dialogue clarifies human-machine interaction, allowing the user to focus on a specific aspect. We chose this design to emphasize mode changes (i.e., to ensure that user is cognizant of the mode) and to facilitate mode customization (layout, interaction style, etc.).

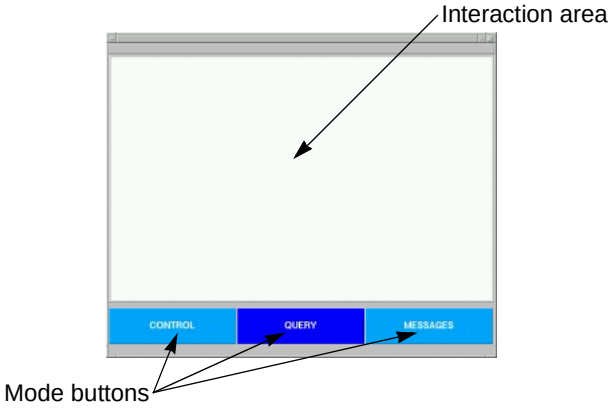

Figure 2. User interface layout

Proper expression of a message is extremely important: if a message is not presented clearly, dialogue will be inefficient. Thus, each mode is designed to express messages differently, using a variety of displays and interaction styles.

Control Mode. The *Control Mode* allows the user to send *robot commands* while receiving *user statements.*The interaction area is split: the right half supports command generation and the left half provides displays. We designed each sub-mode to be highly accessible: the user can rapidly input

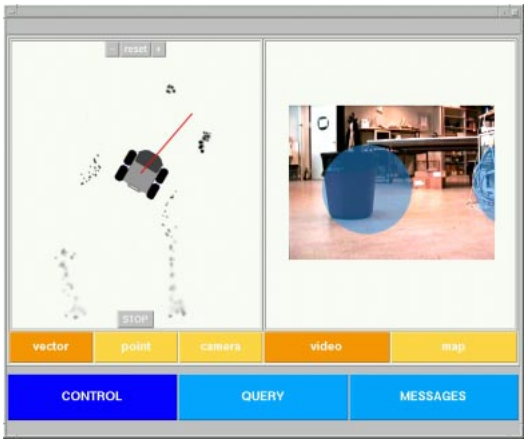

Figure 3. Control mode: vector input and video with overlay

commands and interpret display information. For example, in Figure 3, the user controls robot motion by directly specifying a velocity vector on a moving map display and observes video containing sensor data overlays.

Query Mode. In *Query Mode*, the user asks questions (*query-to-robot*) and receives answers (*response-from-robot*). Whenever Query Mode is active, a set of query-to-robot messages appear in the left half of the interaction area. The robot's response (if any) is displayed to the right.

The manner in which each robot response is shown depends on its content. In Figure 4, for example, the response to "How are you?" appears as a set of bar graphs which display the robot's current state of health: power level, roll over danger, and collision danger.

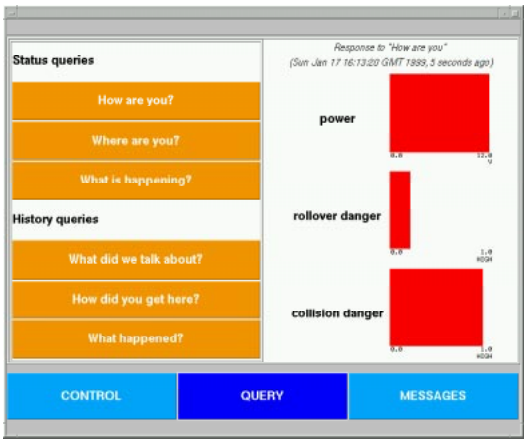

Figure 4. Query mode.

Messages Mode. The *Messages Mode* lets the user respond to questions (*query-to-user*) by sending answers to the robot (*response-from-user*). When Messages Mode is active, it receives any pending query-to-user (selected by the Query Manager via query arbitration) and presents them to the user, one at a time. For example, the query-to-user "How dangerous is this object?" is displayed to the user with an image for the user to evaluate (see Figure 5). The user's response (when given) is then sent to the robot.

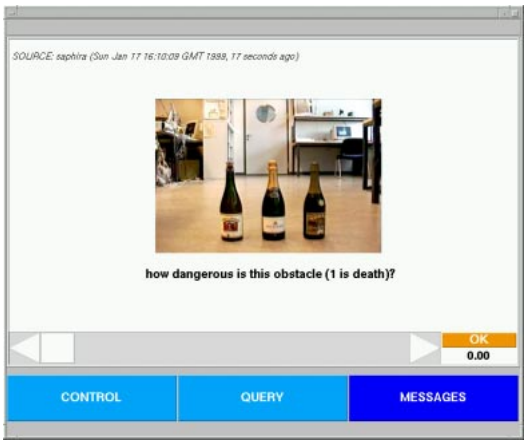

Figure 5. Message mode.

Implementation

We have implemented our collaborative control design using a PioneerAT mobile robot, wireless communications, and distributed message-based computing. The system is shown in Figure 6 below:

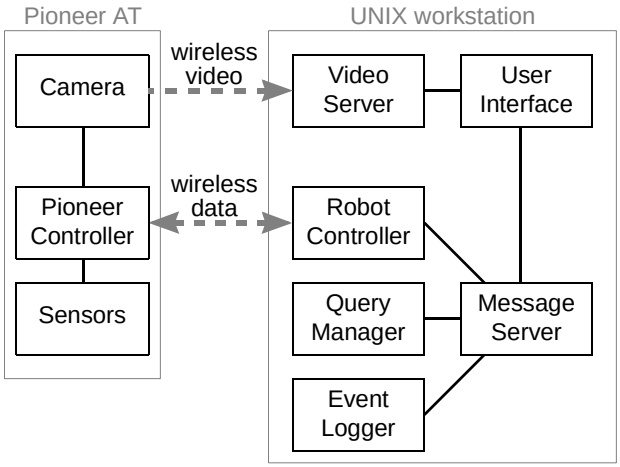

Figure 6. Collaborative control implementation

The PioneerAT is a skid-steered wheeled vehicle which is capable of traversing moderately rough natural terrain. The PioneerAT is equipped with a ring of ultrasonic sonars, power monitoring, and drive encoders (see Figure 7). A pan/tilt/zoom color CCD camera provides on-board video. An analog video transmitter and a RF modem are used for robot/control-station communications. A microprocessor-based *Pioneer Controller* manages on-board sensors and controls vehicle motion.

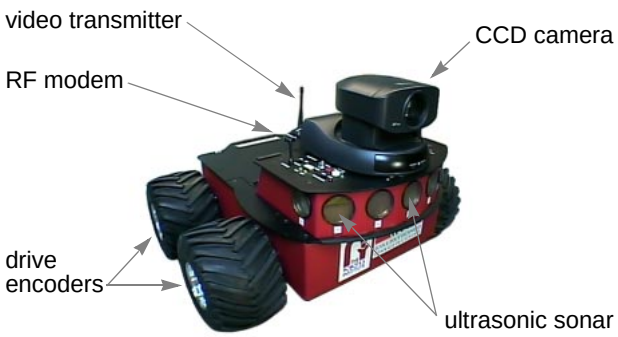

Figure 7. PioneerAT configuration

All collaborative control modules are run off-board the robot as independent, distributed processes. A centralized server-based message system is used to distribute information between processes and for inter-module synchronization. Although Figure 6 shows all collaborative control modules contained on a single workstation, they are not restricted as such and may be distributed and executed on multiple machines.

The *Event Logger* stores event records in a simple, time-indexed database. The *Query Manager* performs query arbitration using a priority and time validity based algorithm. The *Robot Controller* is based on Saphira (Konolige 1996) and performs command arbitration and motion control with fuzzy behaviors.

The *User Interface* is optimized for "single-click" interaction (i.e,. it is well suited for touchscreens) and incorporates video and sound. Since the User Interface executes within a Web browser (e.g., Netscape Navigator), our collaborative control system can be used by multiple users worldwide and does not require special-purpose control station hardware.

Evaluation

Perhaps the most fundamental vehicle teleoperation task is "A to B". That is, if the robot is initially located at position A, and if we know how to get to position B (e.g., we have a map, a world model, or directions), our objective is simply to control the robot's actions so that it moves from A to B. As simple as this task may seem, successful execution is critical to many vehicle teleoperation applications. In reconnaissance, for example, mission performance is directly related to moving quickly and accurately from point to point. Thus it is important to make execution of "A to B" as efficient and as likely of success as possible.

If we attempt "A to B" using only direct teleoperation (i.e., with a minimum of robot autonomy), we find that task performance can be impacted by a wide range of factors: operator limitations (cognition, perception, sensorimotor) communication delay (control instability), poor displays (misleads or confuses the operator), etc. Alternatively, if we rely on some level of autonomy to perform "A to B", unforeseen events, dynamic hazards, and inaccurate (or inadequate) planning may prevent the task from being achieved.

We have recently begun studying how collaborative control influences performance of "A to B". We use the scenario shown in Figure 8: the robot is operating in an unknown environment and is instructed to make a relative change in pose (i.e., from A to B). A mixture of dynamic (moving) and static obstacles prevent the robot from executing a move directly from A to B as well as providing opportunities for perception and decision making.

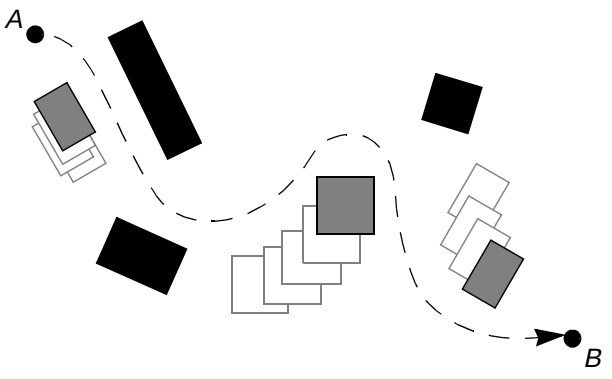

Figure 8. "A to B" test scenario
(static obstacles shown in black, dynamic in grey)

The question we would like to answer is: how does performance (completion, execution speed, situational awareness, etc.) change as the dialogue is varied? Specifically, what effects can we observe as the teleoperation is varied from direct control (operator in continuous control, does not answer queries from the robot) to full collaboration (high-level of operator-robot dialogue and interaction) to autonomy (operator gives command, robot executes without further interaction).

Additionally, we are interested in studying what is the difference between simple interaction and collaboration. In other words, at what point does human-machine interaction allow the user and the robot to truly work together, to jointly participate in decision making, and to collaborate towards task achievement?

Future Work

Currently, the *Query Manager* only considers message priority and time validity when it performs query arbitration. Consequently, query selection is not very discerning and few queries are pruned before presentation to the user. To improve the Query Manager's performance, we need to use a more sophisticated method which also takes into consideration query difficulty and the capability (skill, availability, etc.) of the user to respond.

To do this, we plan to develop a *User Modeler* for evaluating the operator. The User Modeler will monitor the interaction between the user and the robot, and produce a metric of the user's capability based on responses given over time. This metric will also provide an additional benefit by allowing us to improve command arbitration, i.e., better weighting of human commands.

We are also planning to add several high-level behavior modules to increase the robot's autonomous capabilities. In particular, we will be integrating image-based control modes such as a STRIPE (Kay 1997) style path designator and single-camera visual servoing (e.g,. object following). These behavior modules will enrich the dialogue by allowing the user to converse at higher levels of abstraction.

Conclusion

We believe that collaborative control will allow us to solve many of the problems associated with conventional human-as-controller teleoperation. Collaborative control lessens the impact of operator differences and handicaps on system performance. It reduces the need for continuous human involvement while enabling interaction appropriate for a given situation. Collaborative control helps balance the roles of operator and robot, giving the robot more freedom in execution and allowing it to better function if the operator is inattentive or making errors.

By treating the operator as an limited, imprecise, and noisy source of information, collaborative control allows

use of human perception and cognition without requiring continuous or time-critical response. Through the use of dialogue, it improves human/robot interaction and enables control flow to be dynamic, flexible, and adaptable. In short, we believe that collaborative control provides an effective framework for humans and robots to work efficiently together, to jointly solve problems, and to create robust teleoperation systems.

Acknowledgments

We would like to thank the Institut de Systèms Robotiques (Département de Microtechnique, EPFL) for providing research facilities and infrastructure. This work is partially supported by the DARPA TTO "Tactical Mobile Robots" program (NASA Jet Propulsion Laboratory 1200008).

References

Albus, J., et al. 1987. NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM), Technical Note 1235, NIST, Gaithersburg, MD.

Blackmon, T. and Stark, L. 1996. Model-Based Supervisory Control in Telerobotics, *Presence* 5(2).

Cannon, D. and Thomas, G. 1997. Virtual Tools for Supervisory and Collaborative Control of Robots, *Presence* 6(1).

Ferrel, W. and Sheridan, T. 1967. Supervisory Control of Remote Manipulation, *IEEE Spectrum* 4(10).

Fong, T., et al. 1995. Operator Interfaces and Network-Based Participation for Dante II, SAE 25th International Conference on Environmental Systems, San Diego, CA.

Hine, B., et al. 1995. VEVI: A Virtual Environment Teleoperations Interface for Planetary Exploration, SAE 25th International Conference on Environmental Systems, San Diego, CA.

Kay, J. 1997. STRIPE: Remote Driving Using Limited Image Data, Ph.D. Thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Konolige, K. and Myers, K. 1997. The Saphira Architecture for Autonomous Mobile Robots, in *Artificial Intelligence and Mobile Robots* (Bonasso, R. and Murphy, R., eds.), MIT Press, Cambridge, MA.

Krotkov, E., et. al. 1996. Safeguarded Teleoperation for Lunar Rovers, SAE 26th International Conference on Environmental Systems, Monterey, CA.

Lin, I. et. al. 1995. An Advanced Telerobotic Control System for a Mobile Robot with Multisensor Feedback, in *Intelligent Autonomous System IAS-4* (Rembold, U. and Dillmann, R. eds.), IOS Press.

McGovern, D. 1988. Human Interfaces in Remote Driving, Technical Report SAND88-0562, Sandia National Laboratory, Albuquerque, NM.

Mishkin, A. 1997. Operations and Autonomy for the Mars Pathfinder Microrover, briefing, NASA Jet Propulsion Laboratory, Pasadena, CA.

Murphy, R. and Rogers, E. 1996. Cooperative Assistance for Remote Robot Supervision, *Presence* 5(2).

Newman, W. and Lamming, M. 1995. *Interactive System Design*, Addison-Wesley, Reading, MA.

Rosenblatt, J. 1995. DAMN: A Distributed Architecture for Mobile Navigation, in *Proceedings of the 1995 AAAI Spring Symposium on Lessons Learned from Implemented Software Architectures for Physical Agents*, Stanford, CA.

Sanders, M. and McCormick, E. 1993. *Human Factors in Engineering and Design*, McGraw-Hill, New York, NY.

Sheridan, T. 1992. *Telerobotics, Automation, and Human Supervisory Control*, MIT Press, Cambridge, MA.

Sheridan, T. 1993. Space Teleoperation Through Time Delay: Review and Prognosis, *IEEE Transactions on Robotics and Automation* 9(5).

Shoemaker, C. 1990, Low Data Rate Control of Unmanned Ground Systems, in *Proceedings: Association for Unmanned Vehicle Systems (AUVS-90)*, Dayton, OH.

Stone, H. W. 1996. Mars Pathfinder Microrover A Low-Cost, Low-Power Spacecraft, in *Proceedings of the 1996 AIAA Forum on Advanced Developments in Space Robotics*, Madison, WI.

Wettergreen, D. et. al. 1995. Gait Execution for the Dante II Walking Robot, In *Proceedings of the 1995 IEEE Conference on Intelligent Robots and Systems*.

Wettergreen, D., et. al. 1996. NOMAD Operator Interface Development Plan, Intelligent Mechanisms Group, NASA Ames Research Center, Moffett Field, CA.