Human-Robot Coordination Using Scripts

Laura E. Barnes, Robin R. Murphy, Jeffrey D. Craighead
University of South Florida, Tampa, Florida 33620, USA

ABSTRACT

This paper describes an extension of scripts, which have been used to control sequences of robot behavior, to facilitate human-robot coordination. The script mechanism permits the human to both conduct expected, complementary activities with the robot and to intervene opportunistically taking direct control. Scripts address the six major issues associated with human-robot coordination. They allow the human to visualize the robot's mental model of the situation and build a better overall understanding of the situation and what level of autonomy or intervention is needed. It also maintains synchronization of the world and robot models so that control can be seamlessly transferred between human and robot while eliminating "coordination surprise". The extended script mechanism and its implementation in Java on an Inuktun micro-VGTV robot for the technical search task in urban search and rescue is described.

Keywords: human-robot interaction, artificial intelligence, mobile robot, control architecture, teleoperation

1. INTRODUCTION

Challenging domains such as robot-assisted search and rescue, operations in space and demining require humans to interact with robots. These interactions may be in the form of supervisory control, which connotes a high human involvement with limited robot automation (e.g., [1]), semiautonomy, where the robot is truly autonomous for portions of the task (see [2]), or mixed-initiative systems, where the robot and human are largely interchangeable (most notably [2, 3]). The interactions between humans and robots are often interchangeably referred to as "coordination" or "collaboration". Coordination is defined for the purposes of this paper as *the joint efforts of interdependent communicating actors towards mutually defined goals*, following the National Science Foundation, 1989 definition, coordination is viewed as a stricter form of collaboration which is defined as *a human and a robot collaborate to perform tasks and to achieve common goals, instead of a supervisor dictating to a subordinate* [4]. The literature identifies at least six different issues associated with human-robot coordination, each of which is addressed in this paper. These issues are part of the larger question of how the robot and the human as a team maintain situational awareness and work together to accomplish a mission.

- *How to select the level of autonomy for a robot for a particular task or portion of a task?* [2] Typically the level of autonomy is hard-coded for an application. A better solution would be a structure that permits the insertion of techniques requiring different levels of autonomy and a way of representing the requirements.

- *How can the robot's mental model be displayed to the human?* [5] In order to determine the proper course of action in unpredictable domains, humans have to have some sort of way to see both the state of the world and the robot's *mental model*, e.g., what are its current goals, what is it perceiving, etc. Most operator workstations provide adequate displays of the world, but not of the robot internal workings.

- *How will the human know when to intervene (and how)?* [2] A control structure should permit the human to intervene as a part of a normal, expected sequence (e.g., the robot does X, and Y, then the human has to do Z) and to intervene opportunistically. In the first case, even with the most routine interventions, the human may need to be reminded that it is time to intervene and how to accomplish that part of the task. For example, in robot-assisted search and rescue, the search process is repetitious and the operator is working without sleep. As a result, the operator may forget to intervene or not perform all the steps, for example, to update the handwritten log and map associated with the task.

Unmanned Systems Technology VIII, edited by Grant R. Gerhart, Charles M. Shoemaker, Douglas W. Gage,
Proc. of SPIE Vol. 6230, 62300E, (2006) · 0277-786X/06/$15 · doi: 10.1117/12.663650

- *How can "coordination surprise" between agents (human and robot) be eliminated?* [5] "Coordination surprise" occurs when there is a lack of feedback between the agents about their current and future states (in this case, human and robot). If the human is aware of what the robot is doing and what its next expected actions should be during any given task, there is less chance of surprises.

- *How to smoothly transfer control from the human to the robot?* [2] When a human intervenes in supervisory control to accomplish a difficult part of a task or to perform an opportunistic action, the robot may not know how environment was changed or what parts of the task have been accomplished during human control. This can cause significant error because the robot may assume that the world is in the last known state.

- *How to synchronize the human and robot mental models, creating a shared perspective?* [5] While the current level of human-robot coordination is generally "either-or," either the robot does it or the human, as autonomy becomes more advanced and missions more challenging, the agents will act more as a true team. This will require more sophisticated methods for creating a true team perspective.

In addition to these issues, there is practical aspect of how to implement a mechanism for human-robot interaction within behavioral or hybrid deliberative/reactive software architectures. In these styles of architectures, a task is decomposed into sub-tasks which are accomplished through behaviors. Behaviors typically execute concurrently and independently, e.g., *avoid-obstacle* and *move-to-goal*, but also must be sequenced, e.g., navigate to a new location, then stop and look. The central tenet of this paper that the human-robot coordination structure should allow the activity to be represented as behaviors, regardless of which agent performs the activity. This permits behaviors concurrently executed by the human to be replaced with autonomous behaviors without changing the rest of the task code.

This paper addresses each of the theoretical and practical issues of facilitating human-robot coordination through the *scripts* knowledge structure from artificial intelligence [6]. It first presents the related work on human-robot coordination and the use of scripts as a behavioral control mechanism, followed by a discussion of how the script structure is expanded to include *teleoperated* and *coordinated behaviors* as well as an event display. The paper next describes the generic implementation in Java and its application to the technical search task for urban search and rescue [7] using a fieldable mobile robot.

2. RELATED WORK

Extensive research is being conducted in the area of human-robot coordination; see [2, 3, 8, 9, 10, 11, 12, 13, 14] for examples. Target tasks include such as telemanipulation [14], space navigation [13], teleoperation [3, 12] hospital activity [11], roboflag and marsupial docking [10]. The research approaches have not focused on programming structures but rather on agent roles in the system. The script mechanism or any other means of regulating sequences does not appear to have been applied. The work reported in this paper is most similar to the traded autonomy systems of [2, 10] where the robot can completely hand over control to a human operator when necessary, and resume autonomous activity when the operator permits. One disadvantage of [2] is the need to re-plan after control is traded from human to robot. In the search task used for this paper, re-planning is not necessary, unlike, in [10], thus the task can continue from the last state.

3. APPROACH

The approach taken in this paper is to extend the artificial intelligence *script* mechanism developed for sequencing robot behaviors to address the six problems with human-robot coordination identified in Section 1. The overall idea is that robot facilitates the human operator playing two roles– one is that of a direct controller and the other is that of a partner. The operator plays these roles by interacting with the robot through a very high level of supervisory control, where the task is performed by the mostly autonomous robot with human assistance at expected points in the task sequence (e.g., predictable intervention). The operator workstation for the robot displays the user interface which contains a panel with a script for a particular task, highlights the currently active behavior(s) within the script, and cues the operator for required coordinated actions at the correct point. However, the operator can also opportunistically take control of the robot (e.g., unpredictable intervention), and the robot remembers the state of progress and correctly resumes the task when it returns to autonomous control.

As described in [6], the script mechanism consists of *causal chain* of modular activities (behaviors) to be executed by a specific *actor* (in this case, a human or a robot) on the basis of *cues* (events in the world, changes in state, etc.).

The causal chain is equivalent to finite-state automata in expressiveness and power, but uses `case-statement`-like logic providing more readability. The template for a causal chain has slots for the *initialization*, *nominal* activity, and *termination* phases of the task performance. A script also supports sub-scripts. A nominal activity might be composed of several such sub-scripts depending on the cues being received. In addition, a little encountered situation might be captured by an exception sub-script.

In order to accommodate both predictable and unpredictable human intervention in robot tasks, the basic script mechanism in [6] was extended in three ways:

1) Two new types of behaviors were added to mitigate the problem of determining what level of automation to use [2] and to encapsulate predictable interventions. The new behaviors are modular placeholders for evolving future autonomous behavior, where the robot processes the sensory input and computes the appropriate action. *Teleoperation behaviors* are where the human has direct control of the robot, manually processing sensory input and motor output. An example is a human interrupting a preprogrammed search pattern to stop and examine an object of interest. *Coordinated behaviors* are where both actors must contribute to the completion of the behavior, in essence *traded control* [15] within a particular activity. An example is a *calibration* routine which requires the human to move the robot to a known configuration.

2) The explicit sequencing represented by the script now serves as the visible representation of the task in the robot user interface. The visualization of the script allows the human to observe the current task progress, and reminds the human where intervention is required, solving the problem noted by [2]. Explicit, visibly displayed sequencing also addresses the concerns of [5] by reducing “coordination surprises” and sets the foundation for future programs that will enable more feedback between humans and robots.

3) A special category of exception sub-scripts, “improvisation” sub-scripts, provide a mechanism for unexpected human intervention. Exception sub-scripts now store the current state of task progress and enforce synchronization of the robot’s model of the world with the actual environment when the human relinquishes control back to the robot. This is intended to eliminate the errors noted by [2].

4. IMPLEMENTATION

The script approach was implemented in two reusable parts: the actual script, detailing the task to be performed, and the graphical user interface (GUI) developed in Java. As shown in Figure 1, the code consists of two classes, a robot class and a scripts GUI class. The robot’s user interface, including the scripts portion follows the Distributed Robot Field Architecture (DFRA) [16] centered around Java and JINI. The user interface for the robot is depicted in Section 5.

The robot module consists of the script for a particular task and the behaviors. The script consists of a causal chain of behaviors that will run in sequence unless the human generates a subscript. The GUI module informs the human operator of their options, and tells them the current state of the task. The human communicates with the robot via the GUI. The two modules are independent to promote reusability across different task domains.

The human operator can coordinate with the robot module via the GUI. This communication occurs under two circumstances: (1) to send a notification that *teleoperation behavior* is complete, (2) the generation of a subscript. In the case of a *teleoperation behavior*, the human interacts with the GUI by means of a button click to notify the robot module that the next event in the task sequence can begin.

The system is event driven. One generic script event is created to handle both robot generated events and human generated events. The script event is a string consisting of the value of the current task state. The robot module implements script events, while the GUI implements, as well as listens for script events. For any human generated event, the button is equipped with a generic action listener to send a notification when clicked. The action listener will then trigger a script event.

In order for communication from the robot module to the GUI module, a script event is generated which in turn informs the GUI. Since the GUI module implements the script event listener, when there is a state change, the GUI is updated with the new state. Within a specific task, there is a finite set of states. Within each behavior for a given task, the state or states must be saved. When a new state occurs, the GUI is notified.

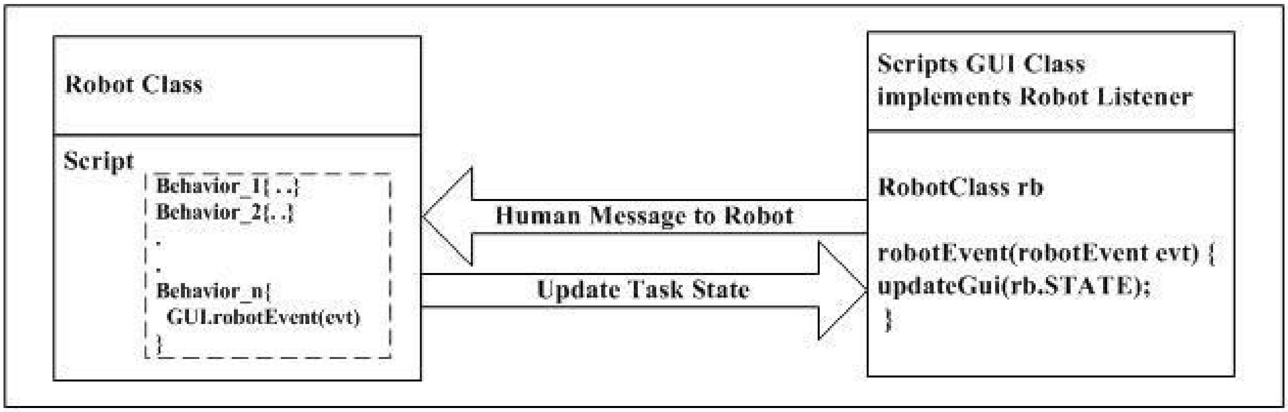

Figure 1: Software Module Diagram

Each task behavior must be modified to send script event notifications to the GUI in order to notify the human operator. These events will signify the action that needs to be taken (in the case of a *teleoperation behavior*) or the current state of the robot (in the case of a *robot behavior*).

5. APPLICATION TO SEARCH TASK

The script mechanism was applied to the search task in the domain of robot-assisted urban search and rescue using a fieldable mobile robot. This task requires a high degree of human-robot coordination because of the complexity and the general difficulty of automating the task with the settings and equipment. This section first describes the search task and the hardware, and then steps through the script for the search task in order to detail the implementation.

5.1 Search Task

The robot-assisted search task is described in [7], but can be summarized with the mnemonic LOVR. The operator typically moves the robot about 1-2 meters to the next interesting point and *L*ocalizes the robot visually, *Observes* the situation using the video camera, scans for *V*ictims using a thermal camera, and *R*eports the findings to his partner. Note that LOVR is an explicit sequence that can be captured in a causal chain. The sequence repeats until the entire void is searched. The “localize visually” and “reporting” activities require the operator and his/her partner to write down findings in their logbook, which they often forget to do.

LOVR can be decomposed into three broad behaviors. The localization activity forms one behavior, `localization`. Localization in a highly confined, unstructured void is very difficult to automate and so the current state-of-the-art is to teleoperate the robot, e.g., a *teleoperated behavior*. The observation and scanning processes are behaviorally identical; the only difference is in the sensor. This is a tedious up-down, right-left scan that lends itself to automation through the `search scan` robot behavior, which is instantiated and executed twice. In practice, however, there is a third behavior, `calibration`, where the parameters for the robot configuration and camera pan motors are reset at the beginning of LOVR. Due to the lack of proprioceptors, calibration requires the operator to put the robot and camera pan into a known position. From that point, the robot can complete the calibration. This is an instance of a *coordinated behavior*.

5.2 Hardware

The equipment used consisted of a customized Inuktun micro VGTV (Variable Geometry Tracked Vehicle) controlled through an operator-console unit (OCU) and a ruggedized 1 GHz Pentium III processor laptop. The system is shown in Figure 2. The robot itself is a shoebox sized, multi-tracked, tethered robot with a 53° field of view color CCD camera on a tilt unit and two-way audio. The video feed from the robot camera is displayed on the OCU. The robot is skid steered, with two tracks, and also polymorphic, able to change from perfectly flat to fully upright. The robot was chosen because it is a fieldable robot representing both the complexity of control and because it has so few proprioceptive sensors that many actions cannot be fully automated. Since the commercial version of the Inuktun micro-VGTV is teleoperated, an API was written in order for the Inuktun to permit access to every control available on the VGTV OCU as well as the return information. A serial cable from the OCU to the laptop allows for the robot to be controlled from the computer.

Figure 2: Full system including OCU, laptop, and robot.

5.3 Operation

All human interaction with the robot is controlled from the GUI Figure 3. The human operator has the control to begin the task, as well as control to pause or end the task at any time from the scripts panel as shown in Figure 4. When the human operator initiates the task, by clicking the start button, the *pause* and *end* options become available. Before task initiation, these buttons are inactive. If the operator clicks the pause button, all motor commands are halted and the human is free to manually operate the robot. Since the pause and resume are mutually exclusive, after the pause button is clicked, it becomes the resume button. The search script is iterative and can only be ended by the human operator. The active task is highlighted in the script and operator tasks are denoted by a red arrow with an *"O"* and automated tasks are denoted by a red arrow with an *"A"*. After an iteration of the search script ends, the task begins again with the `localize` behavior. Previous, current, and future task states are always visible to the operator.

The initialization phase, shown in Figure 4 and Figure 5, consists of *robot localization* and a coordinated behavior, *calibration*. The `localize` behavior consists of the human manually moving the robot to the desired point. This portion of the task is not automated because given a rough terrain, simply moving the motors at a constant speed may not be sufficient. The robot's height or geometry might be necessary to change in order for it to move on this terrain or the tether might need to be used. Once the `localize` behavior is complete, robot *calibration* occurs.

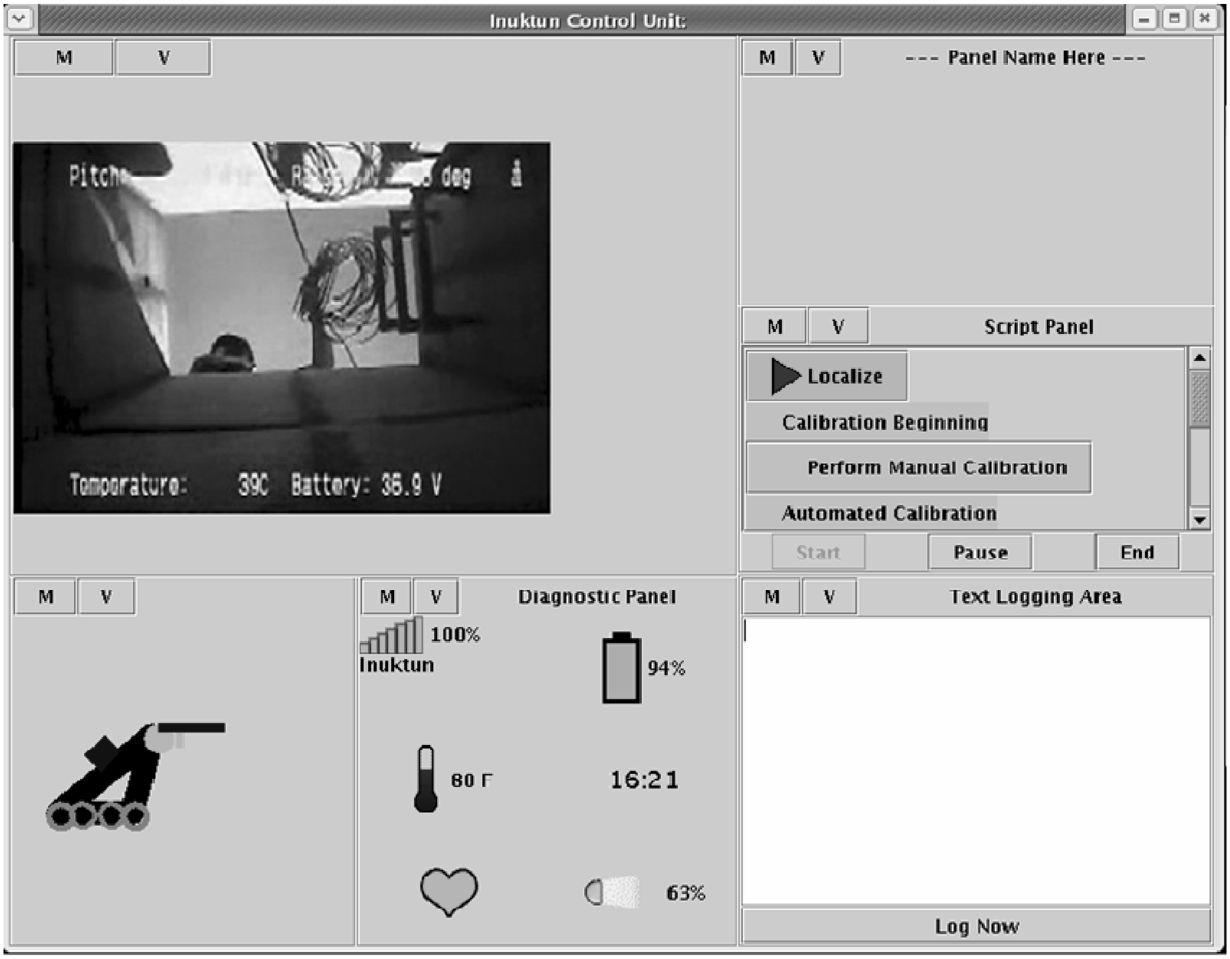

Figure 3: Screenshot of Inuktun graphical user interface.

The robot calibration is a coordinated behavior. The human portion of the calibration behavior consists of manually positioning the camera into the most upright position (53^o). This *calibration* is necessary because there are no sensors on the Inuktun that output camera position. The sensory input is the video, and the human gauges that the camera has been moved to the full upright position when the video is unchanging indicating that the camera has no more degrees of freedom to travel in that direction. Once the human portion of the calibration is complete, the automated portion of the calibration begins. The *automated calibration c*onsists of centralizing the camera position, and setting the raise of the robot appropriately. If on the first iteration of the search task, the raise will always be calibrated to 0^o. If returning from a *pause* state, and re-calibration is necessary, the robot will return to the height of the last state. The sensory inputs are time for the camera calibration. The camera motors are turned on for the time needed to put the camera in the central position. The sensory input for raising the height is the raise sensor on the robot. The robot is raised until the raise sensor for the robot reads the desired value.

The nominal phase of the search task consists of a *vertical search scan* and a *horizontal search scan*. Once the robot calibration is complete, the `scanning` behavior begins. Figure 6 and Figure 7 depict the GUI in the nominal behavior state. This behavior is a coordinated behavior because the rough surface makes it impossible to conduct the horizontal portion of the scan with only dead reckoning. The fully automated vertical scan occurs first. The vertical scan consists of a full camera rotation at 3 different raise values including 0^o, 45^o, and 90^o, and then returning to 0^o raise value. The sensory input for this behavior is again the raise sensor. Following the *vertical scan*, the human operator is asked to perform a manual *horizontal scan*. The *horizontal scan* consists of moving the robot approximately 45^o to the left and 45^o to the right of the current robot position.

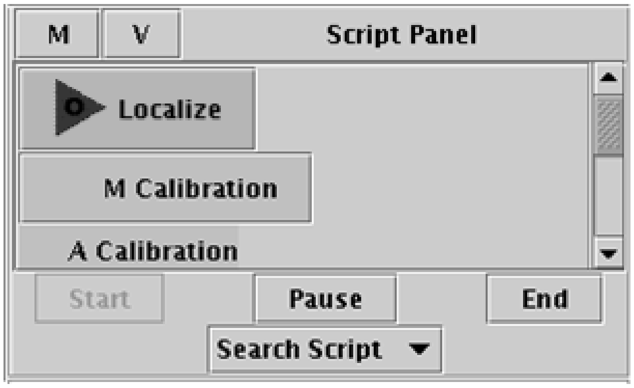

Figure 4: Screenshot of scripts panel in initialization task phase.

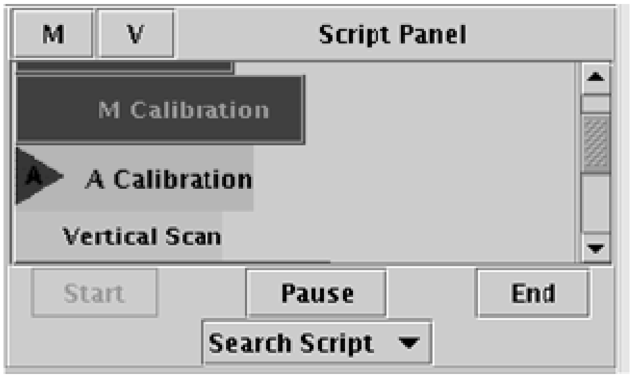

Figure 5: Coordinated behavior in initialization phase.

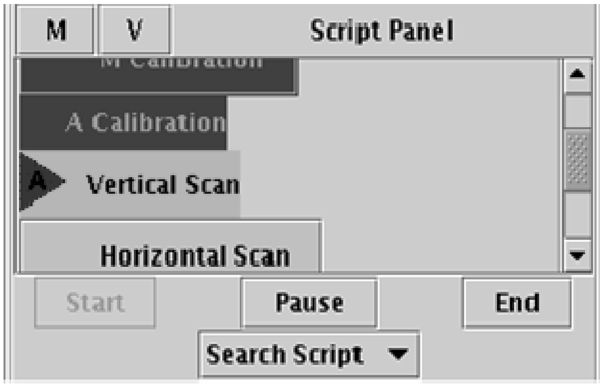

Figure 6: Screenshot of scripts panel in nominal task phase.

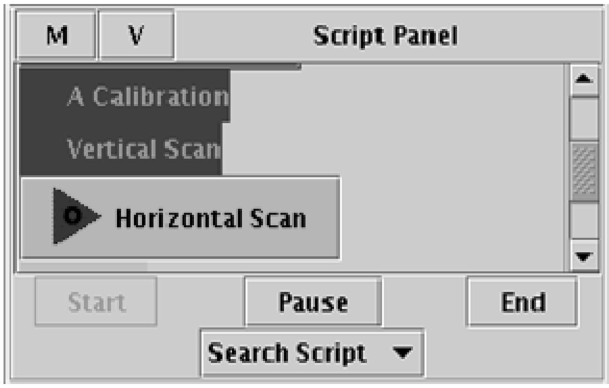

Figure 7: Final task in nominal phase.

At the end of the *vertical* and *horizontal scan*, the operator is prompted to *report* to the mission specialist and *navigate* the robot forward. This is depicted in Figure 8 and Figure 9 respectively which show a complete iteration of the search task finishes.

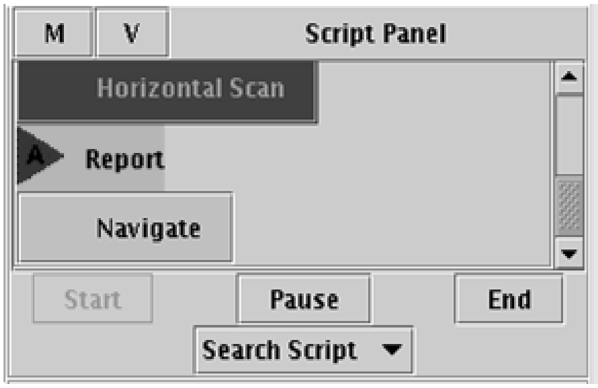

Figure 8: Screenshot of scripts panel in final task phase.

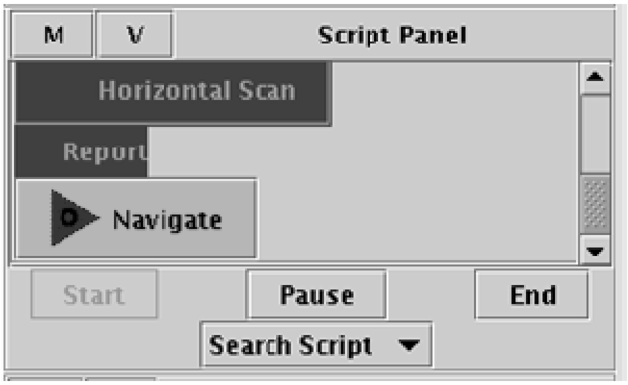

Figure 9: Screenshot of scripts panel after 1 iteration of script

The search script exits on human termination only. When the search task is completed, the script sequence is automatically reported for future analysis by operator and programmer and the task instance is killed, and the GUI screen is cleared. After *termination*, the search behavior can be restarted at anytime. The *pause* command permits the operator to interrupt execution of the script at any point. When this occurs, the task sequence is suspended, halting all autonomous motor function, until further human notification. The operator can now *teleoperate* the robot. This manual operation is treated as a generated subscript. During this time there is no communication between the behavior module and the GUI module. If the task is resumed, the operator is given a choice of whether to continue the search task without calibration, or to re-calibrate the robot. The need to re-calibrate depends on the motor actions if any during the time the behavior was suspended. If the robot needs recalibration, the search tasks resumes with the `calibrate` behavior followed by the last state the robot was in before the pause. If no recalibration is needed, the search task resumes with the last state before the *pause*.

6. CONCLUSIONS

This paper shows how the extended script mechanism resolves some of the issues of human-robot coordination discussed in Section 1. These issues include determining the level of autonomy [2], the appropriate time for human intervention [2], avoiding "coordination" surprise [5], and the smooth transfer of control between human and robot [2]. The level of autonomy and appropriate time for human intervention are solved through coordinated behaviors and the option for human opportunistic control. "Coordination surprise" is avoided through a user interface notifying the human of the robot state. The smooth transfer in control from robot back to human is accomplished through utilizing the human as a source to find out if the state has been significantly modified.

The application of a script to interleave human and robot coordination is both logical and natural. Scripts simplify the relationship between human and robot making the task comprehensible to both novice and expert system users. The ability to simplify a task as a series of simple steps is necessary for this comprehension. The available or possible actions for the human operator at any particular time are clear in the script because of the GUI. This approach can be applied to any task, with any level of human-robot coordination.

The approach is simple conceptually, and easy to implement and modify. More functionality can be added easily to the behavior set for a particular task redefining the event sequence and/or the relationship between the human and robot. The human will always have complete supervisory control, regardless of the autonomy level of the system. The system design simplifies the task of integrating human control into a robotic system. The behaviors fire events to the GUI while at the same time the human via the GUI can send messages to the motor controllers, in the case of a shared behavior or generation of a subscript. The design is easily transferred to other tasks, because the GUI just acts as a listener for any robot event updating it accordingly depending on the event.

This work is expected to be extended to a distributed architecture where multiple human-operators can view the script and intervene. Another possible expansion might be to create an XML script implementation that would easily facilitate adding new tasks and editing the behavior sequence.

ACKNOWLEDGEMENTS

This work was conducted under grants from the National Science Foundation CISE-RR program (*0224401*) and the Office of Naval Research (*N00014 - 03 - 1- 0786*).

REFERENCES

1. R. Ambrose, H. Aldridge, R. Askew, R. Burridge, W. Bluethmann, M. Diftler, C. Lovchik, D. Magruder, and F. Rehnmark. Robonaut: Nasa's space humanoid. *IEEE Intelligent Systems.*, 15(4):57–63, 2000.
2. D. Kortenkamp, R. P. Bonasso, D. Ryan, and D. Schreckenghost. Traded control with autonomous robots as mixed initiative interaction. In *AAAI Spring Symposium on Mixed Initiative Interaction*, 1997.

3. Y. Horiguchi, T. Sawaragi, and G. Akashi. Naturalistic human-robot collaboration based upon mixed initiative interactions in a teleoperating environment. In *IEEE International Conference on Systems, Man, and Cybernetics*, 2000.
4. T. Fong, C. Thorpe, and C. Baur. Collaborative control: A robot-centric model for vehicle teleoperation. In *AAAI Spring Symposium: Agents with Adjustable Autonomy*, 1999.
5. D. D. Woods, J. Tittle, M. Feil, and A. Roesler. Envisioning human-robot coordination in future operations. *IEEE Transactions on Systems, Man, and Cybernetics- Part C*, 34(2):210–218, May 2004.
6. Robin R. Murphy. *Introduction to AI Robotics*. MIT Press, 2000.
7. Robin R. Murphy. Human-robot interaction in rescue robotics. *IEEE Transactions on Systems, Man, and Cybernetics- Part C*, 34(2):138–153, May 2004.
8. M. A. Blake, G. A. Sorenson, J. K. Archibald, and R.W. Beard. Human assisted capture-the-flag in an urban environment. In *International Conference on Robotics and Automation*, 2004.
9. P. Zigoris, J. Siu, O. Wang, and A. Hayes. Balancing automated behavior and human control in multi-agent systems: a case study in roboflag. In *American Control Conference*, 2003.
10. A. Gage, R. R. Murphy, and B. Minten. Adjustable autonomy for marsupial docking. Technical Report CRASAR-TR-2002-9, University of South Florida, 2002.
11. N. Sarkar. Psychophysiological control architecture for human-robot coordination- concepts and initial experiments. In *International Conference on Robotics and Automation*, 2002.
12. A. Monferrer and D. Bonyuet. Cooperative robot teleoperation through virtual reality interfaces. In *Sixth International Conference on Information Visualisation*, 2002.
13. M. C. Nechyba and Y. Xu. Human-robot cooperation in space: sm2 for new space station structure. *IEEE Robotics and Automation Magazine*, pages 4–11, 1995.
14. Itoh T, K. Kosuge, and T. Fukuda. Human-machine cooperative telemanipulation with motion and force scaling using task-oriented virtual tool dynamics. *IEEE Transactions on Robotics and Automation*, 16(5):505–516, 2000.
15. Thomas Sheridan. *Telerobotics, Automation and Human Supervisory Control*. MIT Press, 1992.
16. Matt Long. Creating a distributed field robot architecture for multiple robots. Master's thesis, 2004.