University of Pennsylvania
ScholarlyCommons

Technical Reports (CIS)

Department of Computer & Information Science

January 1995

Human Management of the Hierarchical System for the Control of Multiple Mobile Robots

Julie A. Adams
University of Pennsylvania

Follow this and additional works at: https://repository.upenn.edu/cis_reports

Recommended Citation
Julie A. Adams, "Human Management of the Hierarchical System for the Control of Multiple Mobile Robots", . January 1995.

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-95-17.

This paper is posted at ScholarlyCommons. https://repository.upenn.edu/cis_reports/213
For more information, please contact repository@pobox.upenn.edu.

Human Management of the Hierarchical System for the Control of Multiple Mobile Robots

Abstract

In order to take advantage of autonomous robotic systems, and yet ensure successful completion of all feasible tasks, we propose a mediation hierarchy in which an operator can interact at all system levels. Robotic systems are not robust in handling un-modeled events. Reactive behaviors may be able to guide the robot back into a modeled state and to continue. Reasoning systems may simply fail. Once a system has failed it is difficult to re-start the task from the failed state. Rather, the rule base is revised, programs altered, and the task re-tried from the beginning.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-95-17.

This technical report is available at ScholarlyCommons: https://repository.upenn.edu/cis_reports/213

Human Management of a Hierarchical System for the Control of Multiple Mobile Robots (Dissertation Proposal)

MS-CIS-95-17
GRASP Lab 392

Julie A. Adams

University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

1995

HUMAN MANAGEMENT OF A HIERARCHICAL SYSTEM FOR THE CONTROL OF MULTIPLE MOBILE ROBOTS

JULIE A. ADAMS

A DISSERTATION PROPOSAL

in

COMPUTER AND INFORMATION SCIENCE

Presented to the Faculties of the University of Pennsylvania in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy.

1995

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem Statement

In order to take advantage of autonomous robotic systems, and yet ensure successful completion of all feasible tasks, we propose a mediation hierarchy in which an operator can interact at all system levels. Robotic systems are not robust in handling un-modeled events. Reactive behaviors may be able to guide the robot back into a modeled state and to continue. Reasoning systems may simply fail. Once a system has failed it is difficult to re-start the task from the failed state. Rather, the rule base is revised, programs altered, and the task re-tried from the beginning.

Human-machine interfaces have been developed for applications in the areas of nuclear power plants, aviation, and telerobotics [Christensen, 1993, Hancke and Braune, 1993, Sheridan, 1992], however, these systems are generally not considered autonomous with the operator providing a "supervisory" role. Typically, the human operator controls the entire task execution.

One aspect of the system we are developing permits the human operator, when necessary, to interact with all levels of a system to assist with process errors. This interaction encompasses all areas of a semi-autonomous system from the processes which would be considered fully-autonomous to those considered telerobotic.

Our system, MASC - Multiple Agent Supervisory Control system, permits the agents to work autonomously until the human supervisor is requested to take control or detects a problem. Our design strategy is to develop a general system which is applicable to various robotic systems. We combine the advantages of autonomous systems with the human's ability to control a system through a human-machine interface. MASC provides the supervisor with tools to interact with all the robotic system processing levels. These interactions may correct corrupted data or process decisions which would typically cause an autonomous system to enter an unstable state. We desire to create a more comprehensive semi-autonomous system based on this interaction which will successfully complete the execution of task assignments.

We have defined four hierarchical levels of supervisory interaction with the various robotic system levels. MASC permits the supervisor to specify task assignments, teleoperate agents, display sensory data, override process conclusions and reconfigure the system during sundry sensory and agent failures.

1.2 Scope and Outline of Document

This proposal describes the mediation hierarchy and the process integrations which we will employ to verify our hypothesis. The remainder of this chapter will provide a related research review. Chapter Two briefly describes the test bed for the interface as well as the MASC interface aesthetics. Chapter Three provides the motivation for the development of this theory, a formal definition of the mediation hierarchy and its levels. A description of the implementations of the processes which we are integrating into the MASC interface is provided in Chapter Four. We explain the planned experiments to prove the hypothesis in Chapter Five. Finally, Chapter Six states the contributions and conclusions of this research.

1.3 Literature Review

1.3.1 Control Methods

Kelley [Kelley, 1968] states:

> Control involves a choice or selection among possible future states, the chosen state comprising the chooser's goal. This choice is implicit in every control activity, whether action to achieve it is carried out by living individuals, by an automatic device or control system, or by some complex arrangement of men and equipment. And the choice itself is always made by man.

When exercising control over an object we are changing that object's course of events. Many "choices" were necessary when developing our system. One "choice" was the control method we would employ through our human-machine interface. During the course of determining this "choice" we reviewed behavior based and supervisory control methods.

This subsection is a review of these control methods and an explanation of the "choice" of supervisory control.

Behavior Based Control

Prior to Brooks development of the subsumption architecture, [Brooks, 1986, Brooks, 1987], most control architectures were organized as horizontal subsystems; broken into sub-subsystems, sub-sub-subsystems, etc; such that systems were built as a chain of causes and effects. An example of this as demonstrated by Kelley in [Kelley, 1968]:

> ... consider the control of an environmental variable X, where X is brought about by Y, and Y is brought about by Z. The energy of control is applied to Z in the innermost loop. Z is varied to bring about a desired change in Y, which will in turn bring about the desired change in X in the outer loop.

This example is shown in Figure 1.1. Generally, the inner loops, denoted by Z and Y, are stronger and of a higher frequency than the outer loops, denoted by X and Y. These inner loops may also create rate changes or accelerations in the variables of their respective outer loops.

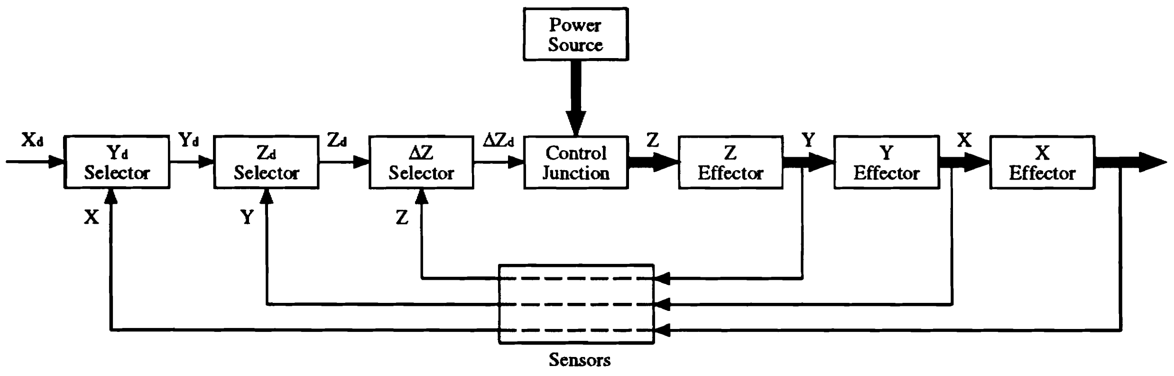


Figure 1.1: The horizontal control structure as described in [Kelley, 1968].

Living beings control their surrounding environments as do control systems [Kelley, 1968]. The outer loops represent a being's increased control over its environment, but as each outer loop is developed, it becomes increasingly dependent upon the previously developed inner loops. Therefore, when the inner loops fail, the outer loops also fail and are unable to recover. This as well as the ability to distribute a representation amongst individual behaviors and the ability to create reasoning from various behaviors led Brooks [Brooks, 1986, Brooks, 1987] to develop the subsumption architecture. His concept was to

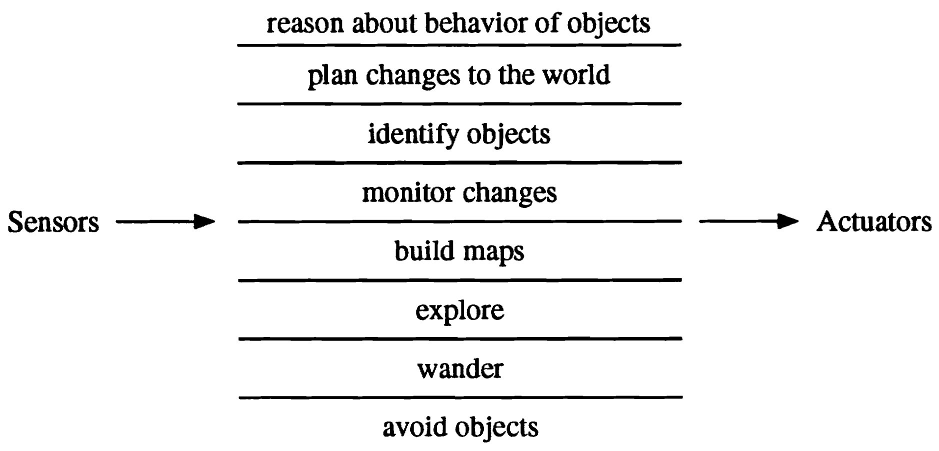


Figure 1.2: Brook's levels of competence.

construct a system bottom-up which could exhibit intelligence and navigate in an unstructured environment. Brooks decomposed the problem into levels of competence, as shown in Figure 1.2. Brooks believes this architecture permits a complete control system to be built and tested and then allows the addition of higher level control systems.

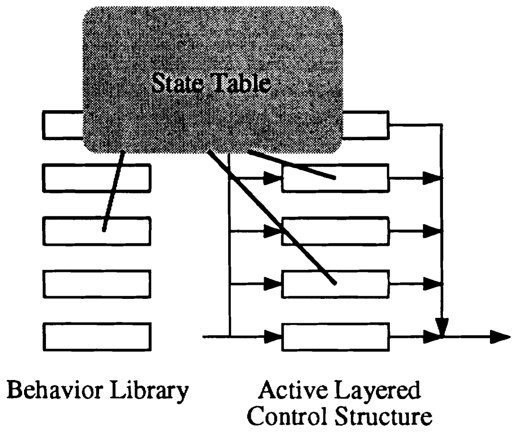


Figure 1.3: Bellingham and Consi's state configured layered control architecture.

Bellingham and Consi expanded the basic subsumption architecture into the state configured layered control [Bellingham and Consi, 1990]. This control method was developed to address the complexities which develop when employing the subsumption architecture. While employing Bellingham and Consi's method, only the layers relevant to the current portion of the mission are active, thus reducing the complexity. See Figure 1.3. The "Behavior Library" is composed of all possible inactive behaviors while the "Active Layered Control Structure" houses only those behaviors pertinent to the current goal or sub-goal. A "State Table" is employed as an external structure to ensure behaviors are activated at the proper moment and with the correct priority. The objective is to minimize the number of active behaviors at any given time frame.

Taipale and Hirai, in [Taipale and Hirai, 1993], extend Brooks' subsumption architecture to a multiple robots domain. Their master robot can subsume the slave robot's behaviors. Figure 1.4 displays this control scheme. They employ master-slave control such

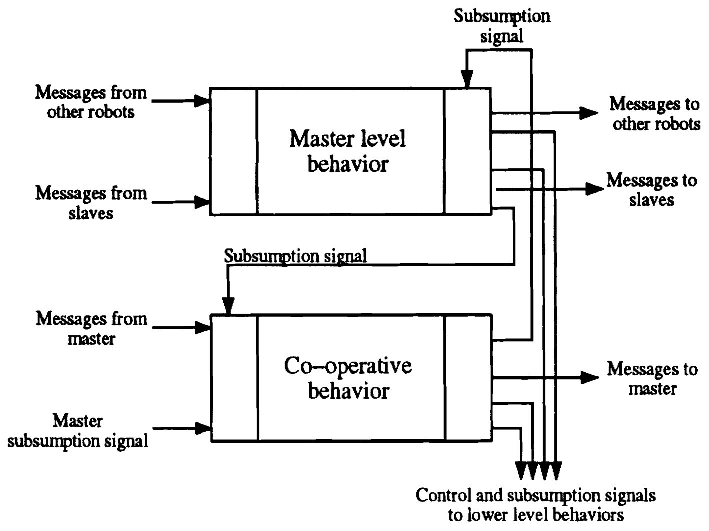


Figure 1.4: Taipale and Hirai's control level scheme of one robot.

that the master's signals subsume the slave's normal behavior. If a robot is the master, it employs the upper "Master level behavior" for control while the slave actions are controlled by the "Co-operative behavior". If the master fails then the "Master level behavior" of a slave assumes control. The agent which requires assistance becomes the master. While this approach extends the layered control to include master-slave control it also raises issues when determining which robot should be the master, and permits the possibility of deadlock when help is needed by numerous masters and there are an insufficient number of slaves to assist.

There are many problems associated with the use of the subsumption architecture described in [Bogoni, 1994] and similarly in [Hartley and Pipitone, 1991]. Bogoni observes, it is not clear that a strict hierarchical relation between the various system modules is sufficient or possible. It is possible that mutual exclusion may be necessary such that only one behavior is active at a time. Bogoni notes that Brook's model employs the world as a means of communication which may create numerous problems when dealing with reactive behaviors that are initiated by "preconditions matched in the environment". With complex systems it is possible many of these behaviors may be initiated at any single time instance. Also as Bogoni observes, for complex systems with numerous behaviors, "the original schema of control and passing of simple message scheme is not sufficient when attempting to carry out a more interesting task." There are also problems associated with the need for redundant code when addressing the the behavior's parameterization and instances where a behavior is a subset of another behavior at a different level. Many others, [Fleury et al., 1994, Hartley and Pipitone, 1991, Martinengo et al., 1994, Mataric, 1992, Stein, 1994], have developed systems based upon behavior based control. While some of these systems have attempted to implement versions of this architecture, they were unable to solve all the associated problems. We believe, as does Sheridan [Sheridan, 1992], that a more robust system can be created employing human supervisory interaction.

Supervisory Control

Almost 20 years prior to Sheridan's definition of supervisory control, in [Sheridan, 1986], Kelley [Kelley, 1968] concluded that automatic control devices could not "approach the flexibility and versatility of man." He explained that a human could act as an adaptive controller[1] since the human was able to foresee the possible system alternatives and then invoke the proper procedures to obtain the desired goal. He believed that the human's ability to "understand and evaluate complex criteria" and to appropriately modify the control behavior were a virtue for the human operator's existence. In essence, this description is a high level idealism of supervisory control as defined by Sheridan.

Sheridan defines supervisory control in two "senses" [Sheridan, 1992]:

- The stricter sense, one or more human operators are intermittently programming and continually receiving information from a computer that itself closes an autonomous control loop through artificial effectors and sensors to the controlled process or task environment;
- the less strict sense, one or more human operators are continually programming and receiving information from a computer that interconnects through artificial effectors and sensors to the controlled process or task environment.

[1]As defined in [Stramler Jr., 1993], adaptive control is a form of automated control equipped with a self-contained decision-making capability for modifying its own operation based on previous experience.

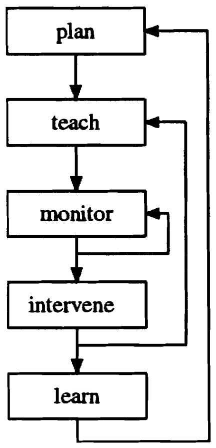


Figure 1.5: The five supervisory functions as nested control loops as presented by Sheridan.

The five basic human supervisory functions, as displayed in Figure 1.5, include:

- the task planning which entails learning about the process and how it is carried out, setting goals and objectives which the computer can understand and then formulating the plan to move from the initial state to the goal state;
- teaching the computer by translating goals and objectives into detailed instructions such that the computer can automatically perform portions of the task;
- the monitoring the autonomous execution either via direct viewing or remote sensing instruments to ensure proper performance;
- the ability to intervene through updating instructions or direct manual control if a problem exists during execution; and
- the ability to learning from the experience by reviewing recorded data and models and then applying what was learned to the above phases in the future.

The need for these basic function can be observed for tasks which encompass the general characteristics of applications for human supervisory control as described in [Baron, 1984]:

1. large scale, technological systems involving high economic value and, often, significant risk.
2. Complex and dynamic processes with many outputs to be "controlled" and many potential inputs for achieving that control.
3. A structure in which there are many sub-systems, with the coupling between variables in different sub-systems much looser than that among variables in a given sub-system.

4. A significant degree of automation, both in system monitoring and control.
5. Relatively slow response of the variables to be controlled (with rapidly changing variables controlled automatically).
6. Event driven demands.
7. The need to interact and coordinate with other operators and/or external entities.
8. The requirement to follow specific procedures in defined situations (available in procedures "manuals" or residing in the operator's memory).

While our situation does not encompass all of these characteristics, it is composed of many of them. Our problem is rather large, although, it is not as large as controlling a nuclear power plant or involve the same high economic value. While the multiagent problem does not entail the second characteristic group, it is a complex group of agents with dynamic processes with many outputs and inputs. The multiagent problem is composed of multiple robots, two of which include manipulator robots, therefore, the individual agents are composed of many sub-systems such as described in characterization three. We are proposing a semi-autonomous system. This implies the multiple agents will work autonomously until they require assistance. Also, we monitor the agent's actions, thus encompassing characteristic four. As our system is not extremely large, we do not encounter characteristic five. Many processes in our system are event driven therefore we must consider the sixth characteristic. Characterization seven is met by our system. We may need to interact with objects in the environment through the mobile agents such as moving a large object using two agents. We also have specific procedures which are defined for particular situations, hence, the characteristic eight is relevant. Pooling these characteristics with the multiagent problem characteristics implies the use of the supervisory control method.

Rasmussen [Rasmussen, 1984] defines a supervisory control system as:

> a feedback system with the task to monitor the actual operating state of the system, and to keep it within the specified target domain.

He believes the system should be developed iteratively based upon the properties of his abstraction hierarchy shown in Figure 1.6. The lowest level of the abstraction hierarchy is the "Physical Form" level which comprises the system's physical appearance and configuration. The "Physical Functions" level represents the physical system processes. This is the level which detects the physically limiting properties and malfunctions. The "Generalized Function" level typically represents the functional system structure in a form which

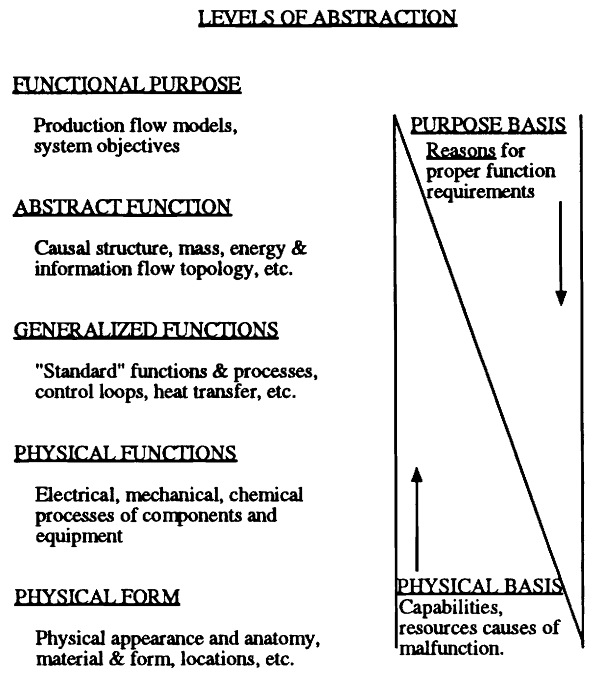

LEVELS OF ABSTRACTION
FUNCTIONAL PURPOSE
Production flow models, system objectives
ABSTRACT FUNCTION
Causal structure, mass, energy & information flow topology, etc.
GENERALIZED FUNCTIONS
"Standard" functions & processes, control loops, heat transfer, etc.
PHYSICAL FUNCTIONS
Electrical, mechanical, chemical processes of components and equipment
PHYSICAL FORM
Physical appearance and anatomy, material & form, locations, etc.
PURPOSE BASIS
Reasons for proper function requirements
PHYSICAL BASIS
Capabilities, resources causes of malfunction.

Figure 1.6: Rasmussen's abstraction hierarchy.

is above the levels of standard components. The "Abstract Function" level represents the overall system function utilizing a generalized causal network. The "Functional Purpose" level describes the system's purpose. This hierarchy can be employed to determine, bottom-up, the system's components and functions utilization, and top-down, to define how the proposed system may be implemented as the functions and components. This manner of development permits identification of the necessary control constraints as each level of abstraction is developed.

Figure 1.7 from [Sheridan, 1992] presents the basic model of our multiagent system. Sheridan utilizes this example to display supervisory control for multiple computers and tasks[2]. The Human-Interactive System is the human operator interface employed to command, control and monitor the Task-Interactive System. The Task-Interactive System is composed of multiple computers and/or robots which individually execute a task to reach an overall goal.

Sheridan also proposes a supervisory control model in which the supervisor controls process information similarly to our proposed supervisory control method. This is displayed in Figure 1.8. This particular model passes information through the physical system external to the human via various information channels. The human may receive three types of

[2]Essentially this model is the same for our multiple agents which may also be composed of multiple computers for example the manipulation robots.

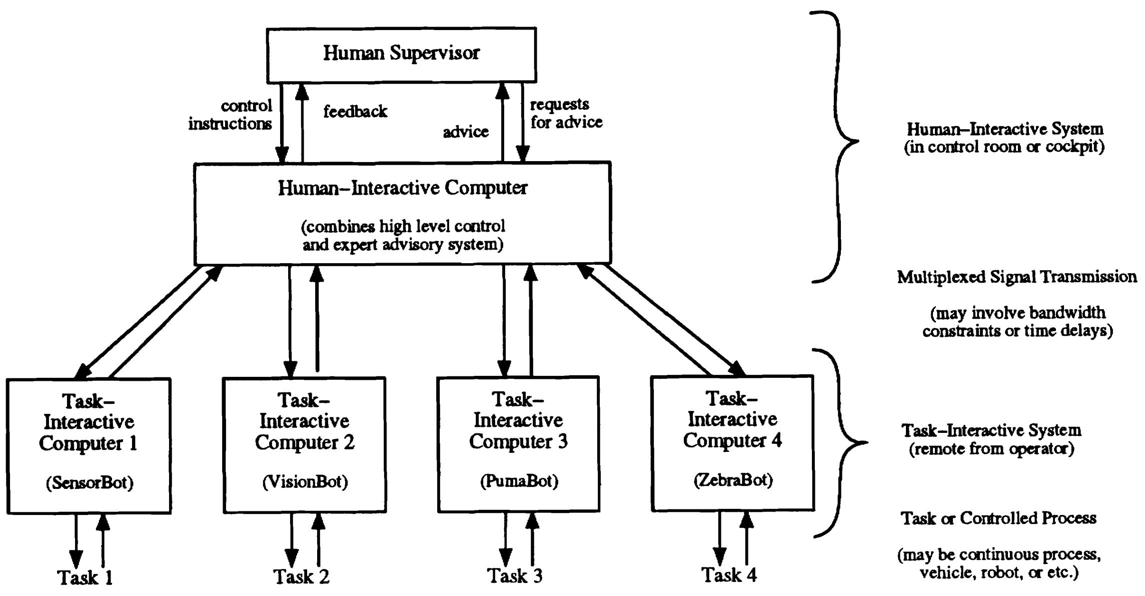

Human Supervisor
control instructions
feedback
advice
requests for advice
Human–Interactive Computer
(combines high level control and expert advisory system)
Task–Interactive Computer 1
(SensorBot)
Task–Interactive Computer 2
(VisionBot)
Task–Interactive Computer 3
(PumaBot)
Task–Interactive Computer 4
(ZebraBot)
Task 1
Task 2
Task 3
Task 4
Human–Interactive System (in control room or cockpit)
Multiplexed Signal Transmission
(may involve bandwidth constraints or time delays)
Task–Interactive System (remote from operator)
Task or Controlled Process
(may be continuous process, vehicle, robot, or etc.)

Figure 1.7: The supervision model of multiple computers and tasks (in our case robotic agents).

system input, as defined by Sheridan:

1. Those that arrive via loop 1 directly from the task (direct seeing, hearing, or touching).
2. Those that arrive via loops 2, 3 and 8 through the artificial display and are mediated by the computer.
3. Those that arrive via loops 9 and 10 from the display or manual controls without going through the computer. (... display itself such as brightness or format, or present position of manual controls, ...)

There also exist three output forms from the human operator, as defined by Sheridan:

1. Those sent via loop 6 directly to the task (the human operator bypasses the manual controls and the computer and directly manipulates the task, makes repairs, etc.).
2. Those that communicate instructions via loops 3, 7 and 8.
3. Those that modify the display or manual control parameters via loops 9 and 10 without affecting the computer (i.e., change the location, forces, labels or other properties of the display or manual control devices).

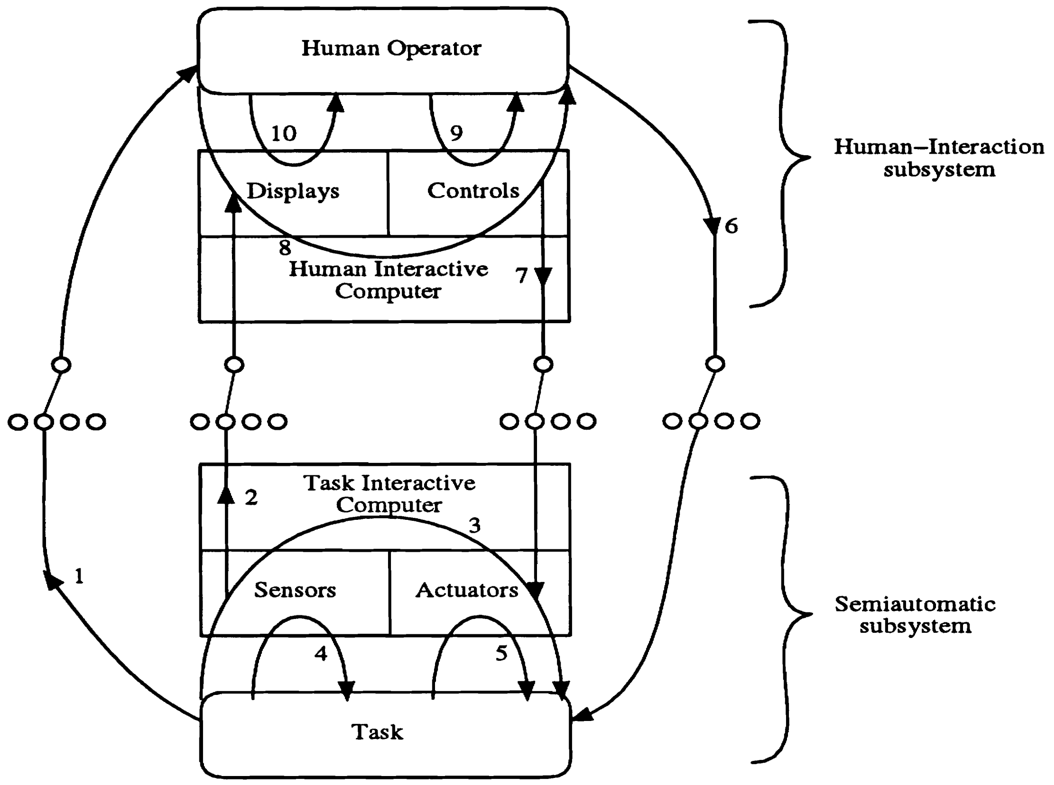


Figure 1.8: As presented in [Sheridan, 1992], supervisory control as multiple and mirrored loops through the physical system.
(1) Task is observed directly by human operator's own senses.
(2) Task is observed indirectly through artificial sensors, computers, and displays. This task-interactive-system (TIS) feedback interacts with that from within human-interactive system (HIS) and is filtered or modified.
(3) Task is controlled within TIS automatic mode.
(4) Task is affected by the process of being sensed.
(5) Task affects actuators, and in turn is affected.
(6) Human operator directly affects task by manipulation.
(7) Human operator affects task indirectly through controls interface, TIS/HIS computers and actuators. This control interacts with that from within TIS and is filtered or modified.
(8) Human operator gets feedback from within HIS in editing a program, running a planning model, or etc.
(9) Human operator orients herself relative to control or adjusts control parameters.
(10) Human operator orients herself relative to display or adjusts display parameters.

Sheridan proposes this model will permit the human operator to intervene at the various physical system levels. Thus permitting the system to be directly controlled or controlled at a higher level by the supervisory control system. This idea is similar to ours. Although, our model does not permit the human operator to directly observe execution via their own physical sensors or to manipulate a task with direct physical manipulation of the robotic system. In our method, the human operator must always observe and manipulate the environment through the human operator interface. Therefore, communication channels 1 and 6 of Figure 1.8 are nonexistent in our system.

Another essential supervisory control consideration is how the human and computer should share and/or trade control. Sheridan states:

> The computer can *extend* the human's capabilities beyond what she can achieve alone, it can partially *relieve* the human, making her job easier; it can *back up* the operator in cases where she falters; and it can *replace* her completely.

We consider this tradeoff in our system development. We permit the interface to extend the human's capabilities and to relieve the human but we also allow the human to take control of any system level. This permits the human to assist the system when difficulties occur. In essence, sharing control indicates the human and computer operate on aspects for which each are better suited and trading control implies the human can take control of the system.

Hirai et al. proposed the cooperative control system for telerobotics in [Hirai and Sato, 1989, Hirai et al., 1992, Sato and Hirai, 1987]. This approach permits the human operator to superpose various control schemes onto the direct maneuvering. The superposed control schemes include: the rate control scheme; the incremental control scheme; the indexing scheme; the software jigs and the programmed control scheme. While these schemes permit the operator to modify the slave actions, the ability to work within and control all system levels is not shown. Their teleoperation intelligence includes: "planning intelligence" which plans the task cooperation between the human operator and the robot; "execution intelligence" which realizes skill cooperation; and "evaluation intelligence" which maintains the environmental changes during task execution. Their proposal is a form of supervisory control but we believe our method elevates supervisory control to a higher level.

Nakamura developed a human-supervised control system for a flexible manufacturing system (FMS). His system design is based upon the following principles as defined in [Nakamura, 1991]:

1. A human in the control loop must be responsible for supervising the automation, monitoring material flows and outputs, and intervening to diagnose and either correct or compensate for machine failures and other unanticipated events [Ammons, 1985].
2. A human can make more effective solutions to the complex problem by modifying the computer solutions [Nakamura, 1990]. For the sake of it, the knowledge-based system and the intelligent human-computer interface are required to aid the human's decision-making.

His system is composed of three components: the human supervisor; the knowledge-based system; and the intelligent interface shown in Figure 1.9. The human supervisor's responsibilities include monitoring the system and system interventions. The knowledge-based system's purpose is to aid decision making. The intelligent interface is composed of a FMS screen for monitoring purposes and a Gantt chart for scheduling. This system requires the

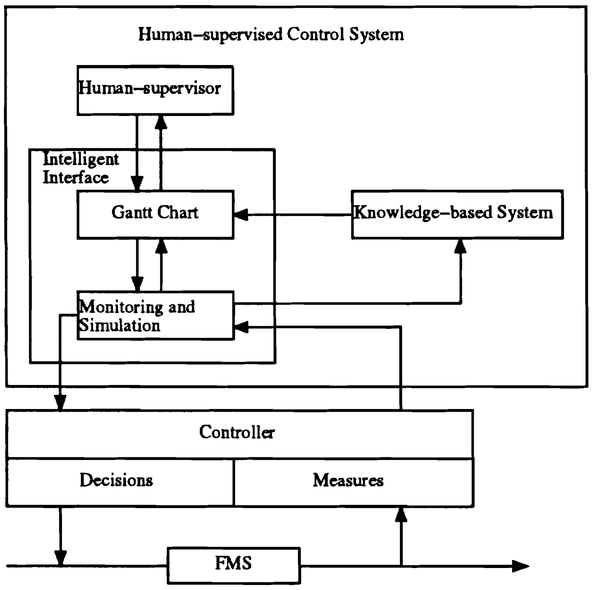


Figure 1.9: Nakamura's structure of a human-supervised control system.

knowledge-base to determine a solution for all instances and then the human must determine if this is the correct solution. This differs from our approach as we do not employ a knowledge-base and decisions are either directly determined by a process or the human. The human may override a decision but the human's verification is not necessary.

Bellingham and Humphrey [Bellingham and Humphrey, 1990] have incorporated supervisory control into the behavior based layered control described in the behavior based control section. They consider only the behavior based control layers in which the human and computer cooperatively control the system. Their "user override mode" permits the operator to create vehicle control commands while also allowing the system's avoidance behaviors to override the operator's commands. The "behavior modification mode" permits the operator to modify the internal vehicle behaviors settings. The "architecture modification mode" permits the operator to turn behaviors on and off or assign a new priority to the overall layered control structure. They have shown that supervisory control may be integrated into the subsumption architecture. While this is an interesting approach, it still does not resolve all the difficulties associated with the subsumption architecture.

1.3.2 Global Planning Methods

There has been considerable research in the field of global artificial Intelligence planning methods[3]. The purpose of planning as stated by Georgeff [Georgeff, 1987]: "is to find out of all the possible actions that an agent can perform, which, if any, will result in the world's behaving as specified by the goal conditions, and in what order these actions should occur." This section will review some of these methods.

One of the most famous classical planners is STRIPS (STanford Reseach Institute Problem Solver) developed by [Fikes and Nilsson, 1971]. It was introduced as a "general-purpose problem solver for robot tasks" and was intended to overcome the computational difficulties associated with situational calculus plan construction. This planner creates plans which are linear and does not employ abstraction methods [Wilkins, 1988].

Hierarchical planners were proposed to introduce abstraction into the planning methodologies. Previous planning methods did not permit the planner to distinguish the actions which were critical for success and those that were not. Sacerdoti [Sacerdoti, 1973] extended the STRIPS planner into ABSTRIPS (Abstraction-Based STRIPS). This method expands the STRIPS problem domain to comprise an abstraction space hierarchy. The method first constructs the abstraction hierarchy and then determines an abstract plan for only the preconditions of the highest criticality level. The plan is then refined while considering the lower criticality levels. One problem associated with the planner is the assumption that preconditions and/or goal conditions will not interact [Knoblock, 1992].

Non-linear planners permit the creation of plans in which actions are unordered with respect to other actions[4]. Sacerdoti [Sacerdoti, 1975] introduced NOAH (Nets of Action Hierarchies) a non-linear, hierarchical planner. His approach considers the plan actions as a partial ordering with respect to time thus relinquishing the need for backtracking.

[3]Good overview papers are [Georgeff, 1987, Tate et al., 1990]

[4]Good overviews of non-linear planning methods can be found in [McAllester and Rosenblitt, 1991, Weld, 1994]

The method creates plans utilizing the procedural net data structure. This data structure contains procedural and declarative representation characteristics and is essentially composed of nodes placed into a hierarchy. The nodes are then expanded in "the order of their position in the time sequence". As this planner does not retain all possible information, it restricts the search space which may lead to an incomplete search.

Tate [Tate, 1977] introduced NONLIN a non-linear planner which generates plans from task descriptions. He implies the method will construct plans with increasing detail at each level while considering the sub-plan interactions to construct the actions into a partially-ordered network. This planner is an extension of NOAH and permits two orderings of subplans. NOAH is unable to "distinguish between important affects at nodes which achieved a condition on some later node and unimportant side-effects". This difficulty may lead to incorrect plan orderings upon reaching the goal node. NONLIN is able to distinguish these effects and construct a proper ordering.

Wilkins [Wilkins, 1984, Wilkins, 1988] introduced the SIPE (System for Interactive Planning and Execution Monitoring) planner which creates non-linear, hierarchical plans. A prominent feature of this planner is the user interface which allows easy access to alternative plans. SIPE plans are represented as procedural nets composed of a partial ordering of goals and actions. SIPE employs causal theories and permits general constraints and replanning of invalid plans. It delays decisions until it has accumulated as much useful information as possible. Figure 1.10 compares SIPE to the classical planners described above

Planner	NonL	Hier	Const	Repln	DI
STRIPS					•
ABSTRIPS		•			•
NOAH	•	•			•
NONLIN	•	•			•
SIPE	•	•	•	•	•

Figure 1.10: Features of Classical Planners.

based upon the capability features Wilkins defines in [Wilkins, 1988][5]. The columns of the figure correspond to non-linear planning (NonL), hierarchical planning (Hier), constraints (Const), replanning (Repln), and domain-independence (DI). In this comparison, SIPE is the only method which combines all of the desired features.

[5]This figure is a subset of the relevant features Wilkin's considers.

Currie and Tate [Currie and Tate, 1990, Currie and Tate, 1991] introduced O-PLAN (Open Planning Architecture) which is an extension of NONLIN. This planner is "intended to be part of a system for command, planning and control". It is a domain independent planner composed of functionality layers which concisely support the task control requirements. The plan representation is similar to NOAH's procedural network. This planner does not require the expansion of all nodes at one level in any particular order. This concept permits the definition of an abstraction level to be "dynamic or opportunistic" dependent upon the current problem. This planner may be unable to determine the best solution for all problems. Its input is a flawed partial plan with a set of action descriptions. It continually modifies the flawed plan until a proper plan is created. It is also equipped with a user interface.

Geib designed the Intentional Planning System (ItPlanS) [Geib, 1994]. This method expands a set of intentions in an "incremental, left-to-right, depth-first" manner. Once a partial plan is created, the planner simulates the plan execution and then employs planning experts for plan refinement. These steps are repeated until all the desired intentions are completed. This method does not require complete a priori knowledge of the environment and is sensitive to changes in the environment.

This review presented some of the basic planning methods. There exist many applications for this research. [Hahndel and Levi, 1994] and [Rocha and Ramos, 1994] employ task planning in flexible manufacturing systems. Applications within the general field of robotics include [Aylett et al., 1991, Dunias, 1993, Rondeau and ElMaraghy, 1990]. [Gerstenfeld, 1988] and [Tarn et al., 1994] have worked in the telerobotic areas. [Ephrati and Rosenschein, 1994, Ntuen et al., 1992, Smith et al., 1992] have explored multiagent planning. These are just a few application examples for global artificial intelligence planning methods.

A planning method which envelopes the incremental, hierarchical and non-linear planning features is desired for integration with our system. The planners which we considered to best fulfill these requirements were O-PLAN, SIPE and ItPlanS. We also require the planner to be developed in a common language and not require special hardware. As SIPE is written in Zetalisp and runs on a Symbolics 3600 machine, it is infeasible given our hardware constraints. O-PLAN is written in common lisp but is so sizable it is also infeasible for integration into the MASC system. Therefore, we are integrating the ItPlanS planner. ItPlanS is written in Lucid Common LISP and is small enough to be considered feasible for interpretation and integration.

1.3.3 General Human-Machine Interface Review

Human-machine interfaces have been developed by numerous researchers. These interfaces range from the early versions composed of dials and mechanisms to today's virtual reality systems [Ellis, 1991, Stansfield, 1994, Hodges et al., 1993, Zyda et al., 1993]. As this dissertation research is concentrated on the development of a human-machine interface, it is necessary to review the various aspects incorporated into human-machine interface development. The remainder of this subsection focuses upon principles which apply to all human-machine interfaces and the following subsections concentrate upon other areas which are applicable to our system.

Traditionally, the human operator was provided direct control into a dynamic system. As Rouse [Rouse, 1981] observes, the human operator's responsibilities are quickly changing. The human operator's task is becoming one of monitoring and supervising a self controlled system. While the human operator's role is changing, Hancke and Braune [Hancke and Braune, 1993] purport that humans are capable of handling the uncertainties through advanced technology. They view automation as an assistant to the human in abnormal situations and also as a provider of various information and control tools. Durand [Durand, 1993] supports this belief and explains that the human will remain a necessary system element. It remains to determine the proper task allocation trade off between the human and the machine.

As Weir observes in [Weir, 1991],

> The human-machine interface provides a means whereby users (operator) can affect the course or operation of the machine or underlying process. Control flow dialogue includes all operator actions on the application and all status data, including results of operator actions. Such actions are the basis for intelligent control.

The primary effectiveness of an interactive system is the human operator's ability to control key system factors.

What constitutes a good interface?

There exist many beliefs as to which characteristics compose a "good" interface. Bodker [Bodker, 1991] quantifies a "good" interface as one which "allows the user to focus on the objects or subjects that the user intends to work with". She believes a bad user interface is one which "forces the user to focus on other objects or subjects than the intended". More specifically, she contends that a "good" interface should:

- permit the user to conduct activities through various actions and operations dependent upon the user's operation skills and the actual material conditions.
- permit all actions to be directed toward their appropriate objects and subjects rather than toward the artifact[6]

While Bodker's view is fairly general, Cox and Walker's [Cox and Walker, 1993] view is much more specific. They contend a "good" interface is designed considering the following four characteristics:

- User Control: the user has ultimate control at all times and determines the task to be performed, not vise versa.
- Transparency: the interface should provide the user with the ability to clearly and completely monitor the task.
- Flexibility: the interface can be used for various tasks including those which the designer may not have considered.
- Learnability: The interface must be easy to use and provide the capability for the users to improve their skills with its use.

As can be observed from these two views, there are many aspects one should consider when developing a "good" human-machine interface. The remainder of this section will review proposed design schemes, interaction mechanisms and data display methods.

Design Approaches

There exist many ideas, methods and theories for human-machine interface design. For instance, Kirlik et al. [Kirlik et al., 1993], believe a human-environment interaction theory must consider both the human and the environment. The environmental component specifies those world features which are psychologically relevant to the desired behavior and provides a descriptive representative language.

Edmonds [Edmonds, 1992] illustrates that batch process designers are typically concerned with the input and output designs and then determine how processes will achieve the defined output. On the other hand, when designing an interactive system a similar technique would entail first designing the interface. This is considered much more difficult. Therefore, designers usually are more concerned with the processor dynamics. Some

[6]As defined by Bodker in [Bodker, 1991], Artifacts are things that mediate the actions of the human being toward another subject or toward an object.

designers believe the design process entails the analysis of both human-machine interactions and communication with complex systems. Others, such as [Grant and Mayes, 1991] believe in a cognitive approach, or a knowledge based approach [Johannsen, 1993].

Shneiderman [Shneiderman, 1984] illustrates five specific interface design issues:

- command language vs. menu selection,
- response time and display rates,
- wording of system messages,
- on-line tutorials, explanations, and help messages,
- hardware devices.

The trade off between command language and menu selection will be discussed in more detail in the following subsection. The response time is considered as the time required for the results of actions to appear on the monitor, such as characters or images. If there is a large delay in this measurement the user will become aggravated and displeased with the system. Shneiderman also found when displaying familiar information, the user prefers it displayed faster while unfamiliar information should be displayed at a slower rate. The manner in which the system words messages also plays an important role in the system usability. Cryptic messages and those which provide little information are found to frustrate users. The designer should consider this fact when creating all system messages including input prompts, menu selection choices, and/or help messages. When users must disrupt their activity to locate a manual they may easily loose track of their current efforts. Therefore, it has become necessary to provide on-line tutorials, help and explanations as the provision of such materials is found to be less disruptive to the user's task. When determining the hardware devices of the system, the designer must consider the hardware complexity. While devices may be interesting to incorporate, the user may find them difficult to use.

An interesting design approach presented in [Brandt, 1993], is the dual design system development approach. This approach, shown in Figure 1.11, is based upon a set of principles which are employed to ensure the proper development of the technical and human aspects of an interface. The technology-based design model is employed to create fully automated systems in which the human is not considered until late in the design process. The working-process based design approach attempts to solve the problem with less automation while considering the human interactions sooner. The optimal approach is to combine both models when designing a system as it introduces the human aspects early into the design process while also considering the technological concerns.

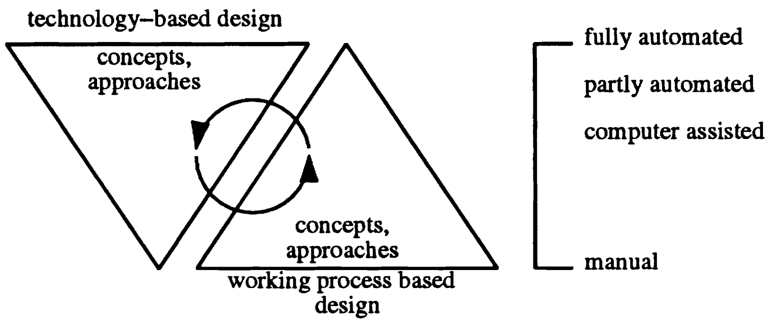


Figure 1.11: The dual design approach to human-machine interface development.

Interface Interaction Methods

In general, there are three distinct techniques of human-machine interface interaction. They are command, menu and direct manipulation which are defined in [Whiteside et al., 1988, Shneiderman, 1984, Jacob, 1989]. This subsection describes these interaction methods and also displays some experimental data to demonstrate which method is considered desirable. As our interface employs direct manipulation this is the method we primarily discuss.

Command Driven Interaction Method: The first mechanism for human-machine interaction was the command driven system. When employing this method, the user communicates via a specialized language which requires keyboard entry of commands. As Shneiderman observes in [Shneiderman, 1984], the use of a command language protocol requires the user to memorize possible commands and their combinations. One option of combining complex command sequences is to develop macro commands.

As a user becomes more skilled with a system, it is necessary to provide abbreviations as the users desire faster and simpler entry techniques. This method reduces the production time while retaining relevant information. Maher and Bell [Maher and Bell, 1992] designed a man-machine interface for abbreviation oriented interaction. They chose "meaningful" abbreviations for their identifiers. They also permitted multiline commands which they found were frequently employed.

Menu Driven Interaction Method: The menu-oriented interaction interface presents the user with various options from which she or he may choose. It eliminates the user's need to memorize commands, while providing a clearer understanding of the command options.

Widdel and Kaster's [Widdel and Kaster, 1991] users determined pull-down menus were faster than command driven input. Although, their study found a higher degree of error with menu use. This degree of error was concluded to be insignificant when compared to the cost of correcting errors in command input interactions. Shneiderman [Shneiderman, 1984] also studied menu driven interfaces. He observed that they eliminate the user's need to memorize commands as they provide an explicit list of the possible commands. He found a difficulty associated with this method to be the menu display rates. If the display rate is slow, it hinders performance and the users become aggravated. This annoyance is especially prevalent if the user must wait until the entire menu is displayed before they may enter their choice or other short cuts (such as key commands) are not provided.

Direct Manipulation Interaction Method: Stramler [Stramler Jr., 1993] defines direct manipulation as:

> A user-computer interface in which the entity being worked is continuously displayed, the communication involves button clicks and movements instead of test-like commands, and changes are quickly represented and reversible.

Our interface and those of [Bach, 1991, Keil-Slawik et al., 1991] are examples of interfaces which employ direct manipulation.

Hollan et al. [Hollan et al., 1987] view an interface built with this interaction method as one designed for communication considering the cognitive task the system supports. They believe the displays support the system but do not guarantee directness. This is a result of the interface language matching the manner in which the user contemplates the task. They state:

> Directness is thus not a property of interfaces but involves the relationship between the task a user has in mind and the way in which the task can be accomplished via the interface.

In [Shneiderman, 1983], he presents the characteristics of a direct manipulation interface which are also described by Jacob [Jacob, 1986, Jacob, 1989]. They are:

- Continuous representation of the object of interest.
- Physical actions (movement and selection by mouse, joystick, touch screen, etc.) or labeled button presses instead of complex systems.
- Rapid, incremental, reversible operations whose impact on the object of interest is immediately visible.
- Layered or spiral approach to learning that permits usage with minimal knowledge.

Shneiderman views the advantages of direct manipulation to be: novice's ability to quickly learn interaction; expert's ability to rapidly interact with the system; infrequent user's ability to retain operation use; the relative infrequent need for error messages; the user's ability to observe immediate action feedback; and the user's ability to gain system mastery as they are able to initiate actions, feel in control and are able to predict system responses.

Jacob [Jacob, 1989] believes the primary advantages of this interaction method are psychological as there is a decreased demand on the user's short and long-term memory. Long-term retention is reduced as the user must only remember a few commands. Short-term memory is reduced as changes to objects are immediately available. Memory is also affected by the reduced number of states and modes the user must execute. In general, as most people find recognition memory easier than recall memory, presumably direct manipulation (and menu-driven) interfaces are superior as they present the alternative to the operator. Motor operations should be minimized, such as typing commands, which implies direct manipulation (and menu-driven) interfaces are favored.

Direct manipulation interfaces are easiest to apply to domains which permit concrete graphical representations. They are more difficult for abstract domains as the object representation may not facilitate user visualization. Another difficulty associated with direct manipulation interfaces is the rigidly fixed one level of abstraction. This constrains the user in a situation where a command driven interface would be more flexible. The users of direct manipulation interfaces are generally not provided methods to create new commands as may be feasible in command systems.

Both Shneiderman and Jacob have significantly studied this area. Shneiderman [Shneiderman, 1984] found when employing this method the user reported:

- System mastery.
- Competence in task performance.
- Ease in learning original system as well as new features.
- Confidence in their capacity to retain mastery over time.
- Enjoyment in using the system.
- Eagerness to show it off to novices.
- Desire to explore more powerful system aspects.

He found that the system aspects which provided the users with these feelings were related to their ability to observe the object they were interested in, their abilities to rapidly reverse actions, and the disappearance of the complex command language syntax.

Jacob discusses his findings in [Jacob, 1986]. He found direct manipulation permits the user to directly operate upon the interface objects as opposed to carrying on a "dialogue about them". A measure of successful direct manipulation implementations is a low level of cognitive distance[7] He also found:

- the direct manipulation interface to comprise a collection of many relatively simple individual dialogues.
- The individual dialogues of a direct manipulation interface to be related to each other as a set of coroutines.
- The dialogue should be specified as a sequence of abstract input and output events, with layout and graphic details given separately.
- Direct manipulation interfaces have definite modes or states, despite their modeless appearance.

This permits the direct manipulation interface construction to be a combination of "individual, possibly mutually interacting interaction objects, organized around the manipulable objects and the loci of remembered state in the dialogue". In other studies conducted on all three interaction methods, Shneiderman established experienced users believe command driven interaction is faster than menu driven interaction, while novices were found to prefer menu driven interaction.

Whiteside et al. tested various system interaction methods described in [Whiteside et al., 1988]. The following questions motivated their experiments:

- Are there large and uniform differences in the usability and evaluation of systems?
- Are any differences which exist related to the class of interfaces (command, menu, iconic)?
- Are some types of systems more suitable for certain types of users?

They tested 165 users and classified them into novice, transfer and system groups. Novice users were those with minimal or no computer experience. Transfer users were those who used interactive computers daily but never the test system. System users were those who previously used the test system. They tested six command driven systems, three menu driven systems and two direct driven systems. They found a significant degree

[7]As defined by Hutchins et al. [Hutchins et al., 1986], cognitive distance is the mental effort required to translate from the input actions and output representations to the operations and objects of the problem domain itself.

of variance in usability and participant systems evaluations. This included the levels of success novices had with various systems. Some systems were easily used while others were very difficult. They also found the interaction type made little difference in the system usability. Systems which should have been easier to use were misleading. They found the systems considered easy to use by experts were also considered such by novices. They determined not only are command driven interfaces difficult to use, but some users also found the complex interaction of mouse button presses for direct manipulation difficult. They also found menu choices and icons were easily misinterpreted. The most important finding was that the interface interaction style can not solve usability problems. Another interesting determination was that more mature systems were considered to have a higher usability.

As Schneider observes [Schneider, 1984], when developing the interface system the designer must realize as users become more skilled with the interaction mode, the mode must change with their abilities. As the user becomes more experienced, she or he will desire a more concise interaction while a novice user will prefer a more detailed interaction.

Display Methods

There exist many varied representations for display methods. Information should be displayed in a useful manner for interpretation but not in a manner which overloads the operator's abilities to understand the displays [Woods, 1991]. It is necessary to display not only raw data readings but also levels of the abstract/processed data [Frey et al., 1993]. The availability of such data levels is useful for different tasks. There exist tasks for which the raw data may be the most informative display, while for other tasks, a higher level of data abstraction is more meaningful. There are also considerations of color use [Kraupner-Stadler, 1991], textual displays [Cox and Walker, 1993] and pictorial displays [Jacob, 1989, Shneiderman, 1984].

Whalley [Whalley, 1992] describes three considerations for display design. First, the information should be easily viewable and readable. Secondly, the information layout for a task should consider the user's ability to check static, dynamic, interactive, and abnormal information types. Finally, the integrated system should provide the user with a complete and accurate understanding of the system.

Misue and Sugiyama [Misue and Sugiyama, 1991] believe users have specific display requirements. They feel users desire displays which enable the viewing of necessary system details, display the entire display objects, permit the simultaneous display of complete information while allowing the display of only one image at a time and finally the consideration of the hardware display capabilities.

Rouse [Rouse, 1981] discusses another perspective of display design, termed display scanning. The amount of time the operator spends viewing data displays and the likely transition of displays should be considered during display development. For instance, he suggest displays which require a significant amount of attention from the operator should be centrally located. Those displays between which many transitions occur should be located in close proximity to one another or merged into one display. As our model has one main window which permits various data to be displayed simultaneously, we have essentially merged many of our displays into one main display.

Traditionally, as Woods observes [Woods, 1991], in industrial control rooms the data displays are dominated by the one display element philosophy which focuses upon the availability and accessibility of raw data (called base data units). He cites many issues which must be considered when designing an interface for data availability. Generally, base data units are assigned to one display mode in one location and are not readily integrated with other data units. He believes the data overload problem is more prevalent in systems designed for data availability. It is more difficult to extract the significance and meaning of the data in relation to other data. The operator must remember and mentally extract such meaning. They fear that an interface designed strictly for the purpose of data availability will not assist the operator when making judgments which should be based upon an entire data set. Thus the operator will base his or her decisions on partial data information.

Barfield and Kim [Barfield and Kim, 1991a] observed it is difficult and tiring for the human operator to visualize the data in three dimensions when observing three-dimensional information on a two dimensional screen because of "the relationship between the perspective geometry parameters used to design such displays and the accuracy with which observers can reconstruct the spatial information contained within the perspective projection". Graphical images provide the operator with a data image. This information is easier to understand and to remember [Herot, 1984]. Graphical user interfaces may be classified into two categories, those which are two-dimensional and those which are three-dimensional representations of the environment or system information. The use of three-dimensions provides the operator with a sense of reality [Regan and Pose, 1993]. [Borys, 1991] have also considered ergonomic issues with graphical display methods.

Hwang and Wang conducted experiments to study the effects of display format types, volumes of data and the layout of data in the human-machine interface. This experiment is described in [Hwang and Wang, 1991]. They found format type had a significant effect upon the operators task executions and that graphical formats were better than digital formats. They felt the graphical format reduced the mental overload of the operator. They also found when the volume of data increased the operator's performance was increased by utilizing a proper layout method.

1.3.4 Graphical User Interfaces

A Graphical user interface as defined by Stramler [Stramler Jr., 1993] is an interface in which the human employs:

> the use of direct manipulation and icons or other graphical symbols on a display to interact with a computer.

While we question the classification of iconic user interfaces as "graphical" user interfaces, they are generally accepted as such. There has been considerable research in the area of graphical user interfaces: [Pejtersen and Nielsen, 1991] employ iconic interfaces, [Eichelberg and Ackermann, 1993] cover general topics for object oriented systems; [Thomas and Goss, 1993] review general topics for three-dimensional graphics; [Heger et al., 1992] study decision support systems; [Chen and Trivedi, 1994] addresses multiple sensor based robots; [Askew and Diftler, 1993] work within an outer space domain; etc. As our interface incorporates a three-dimensional graphical user interface this will be the focus of this subsection.

Foley and Van Dam [Foley and Van Dam, 1982] classify interactive graphics as:

> a form of man-machine interaction which combines the best features of the *interactiveness* of textual (alphanumeric) communication via online keyboard terminals with the *graphical communication* of two-dimensional plotting. With interactive graphics, we are largely liberated from the tedium and frustration of looking for patterns and trends by scanning many pages of linear text on line printer listings or alphanumeric terminals.

There have been many applications combining graphics and raw images. Bejczy et al. [Bejczy et al., 1990] incorporate a "phantom" robot into their time-delayed teleoperation system. The purpose of this "phantom" is to permit the operator to view both "a real-time simulated display of the manipulator and an accurate display of static objects from the delayed video." They overlay real time graphics onto the video camera image. The "phantom" provides the operator with an indication of the actual manipulator location and the delayed position of the manipulator via the image providing the operator with predictive displays.

Matsui and Tsukamoto [Matsui and Tsukamoto, 1990] discuss the development of a Multi-Media Display employed in teleoperation tasks. The multi-media display permits the superimposition of a model onto the real images. This allows the operator to determine the difference between the model and images. The display incorporates multiple visual medias, multiple windows, real-time graphics and stereoscope. It permits the display of real images, graphical models and text on one screen. They incorporate video images and CAD models to learn the environment in which they will perform teleoperation tasks. This is further discussed in [Hasegawa et al., 1990]. The environmental modeling system displays an image on a monitor. When the operator recognizes an object she or he teaches this object's name to the system and then designates points on the object. These points are used to determine the proper three dimensional shape from a CAD database.

Sayers and Paul [Sayers and Paul, 1994] employ a three-dimensional graphical user interface as part of their master station in their time-delayed teleprogramming system. This interface employs a virtual reality display. They developed synthetic fixtures which permit very precise slave manipulator control. They also employ overlays of their working environment images to calibrate their master station.

At Sandia National Laboratories there is an initiative to employ intelligent robotic control for waste remediation [Drotning et al., 1992, Christensen et al., 1991, Christensen, 1993]. Their proposed system includes a three-dimensional graphical user interface which models the environment and controls the robots. The initial model incorporates the known information about the environment and unknown information is detected utilizing; an ultrasonic proximity sensor system, a metal detection system, a ground penetrating radar system, structured lighting range mapping and ancillary systems. Cooke and Stansfield [Cooke and Stansfield, 1994] are developing a method of interactively building graphical models employing telepresence and virtual reality. Their method integrates live video with the incomplete graphical model to permit the operator to incorporate previously unknown objects into the graphical model.

One aspect of graphical user interfaces concerns the user's interaction with the graphical programs. Dai [Dai, 1993] suggests the system's operator be provided with various levels of data. The interface should permit the operator to "modify some parameters and redo part of, not all of, the work according to the user's guidance in real time." He suggest the interaction functions should be centralized as this would simplify the structure of the program and separate the graphical system and the application functions which he calls *centralized, application-oriented interaction control.* He defines the interaction rules with graphical elements such as menus, and callback functions. He proposes the interactions for all system components should be handled via a centralized process associated with the graphical user interface. He proposes that this model increases work efficiency and creates a system with simpler structure which is more device independent. Utilizing this model, the user is able to obtain increased support via directly manipulating the graphics.

Barfield and Lim conducted a study, described in [Barfield and Kim, 1991b], to investigate the user's feeling of realism upon viewing images created with computer graphics rendering techniques and computer synthesized images. They created an image of an apple and designed the experiment such that they varied; the number of lighting sources, the number of specular highlights, the number of shadows, the presence or absence of a color map, and various shading techniques. They created pictures of the apple with changes amongst the various variables. The subjects then viewed these pictures. They found:

> there are diminishing returns in terms of computational resources required to render realistic three-dimensional images versus subjective ratings of image realism. The additional computing resources used to render multiple specular highlights and shadows were not effective in producing higher ratings of subjective realism given the images and rendering techniques evaluated in this study.

As is observed from this subsection, there are many approaches to incorporating graphics into the human-machine interface. There also exist issues which are strictly graphics oriented, such as how realistic the human perceives the graphical image. We employ a three-dimensional graphical model and permit the overlay of real-time system images. We do not build intricate models of the environmental objects. The image overlays provide the supervisor with a realistic environmental perspective.

1.3.5 Human Factors Considerations

There are numerous human factors considerations beyond basic interface design. They relate to the operator's optimal workload, the operator's desire to use the system and the operator's ability to correctly perform the fault diagnosis task. This subsection discusses these issues.

An interesting observation by Edmonds [Edmonds, 1992] which epitomizes the requirement for human-machine interface developers to consider human factors issues is:

The more interactive a system is, and the more inventive or unpredictable the human's part in it is, the less we can discover from task analysis, etc. and the more we must rely on evaluation of the performance of the system in use.

Workload

The human's workload level is an predominant factor when developing a human-machine interface. A common question as posed by Rouse [Rouse, 1981] is:

> What fraction of the task responsibilities should be allocated to the human (at a particular instant in time) in order to keep him sufficiently involved and motivated to perform acceptably over weeks, years, or a whole career?

Whalley [Whalley, 1992] purports the average operator workload should be contained within the 50 and 75 percent range of mental capability. She defines workload as "the time required to complete actions against the time available with the estimated workload assessed against the ergonomic recommendation of between 50 and 75 percent".

Rouse [Rouse, 1981] observes that situations of short term mental workload stress are tolerable though they are not tolerable over the long term and may lead to human error. He has studied the dynamic division of tasks and believes tasks should not be strictly divided between the human and the computer. Instead he suggests that in situations where the task may be performed either by the human or the computer, the task should be allocated to the one with the lowest current work load. He proposes this method will better utilize the system's resources as well as create less variability in the human's workload.

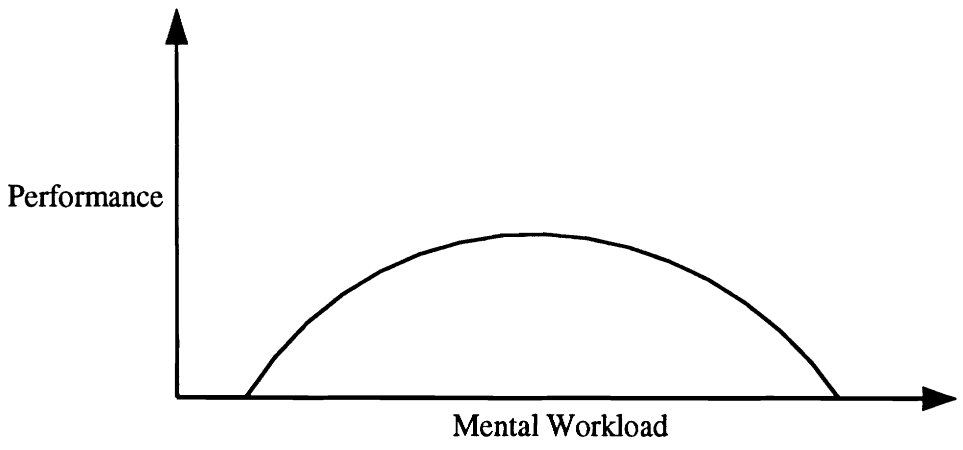


Figure 1.12: The inverted "U" hypothesis for performance vs. mental workload.

Workload may also be defined as a function of factors [Sheridan, 1992]. These factors are composed of those dependent and independent of the operator. Operator dependent factors are; the operator's perception of the task demands, her or his qualifications, capacities and motivations, as well as the operator's behavior. The operator independent factors are composed of; the objectives of the task, the hardware and software resources being utilized, and the environmental conditions such as lighting. Figure 1.12 displays the "U" hypothesis of a user's mental workload capabilities as related to the user's performance. When the mental workload is small, the operator tends to have a lower performance level as there is not enough to hold her or his attention. The operator's performance will also degrade when her or his mental workload is greatly increased. The optimal workload level is the peak of the inverted "U". Thus optimal performance is related to the optimal mental workload but this level is yet to be fully defined as it varies amongst operators.

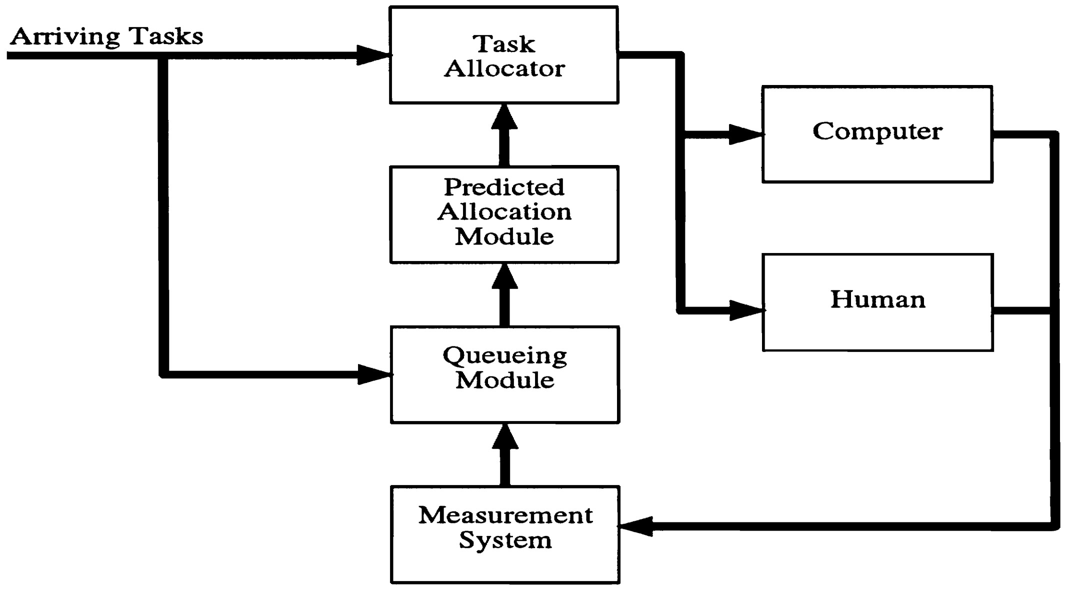


Figure 1.13: The adaptive task allocation human-computer interface.

Rencken and Durrant-Whyte [Rencken and Durrant-Whyte, 1993] propose an adaptive task allocation approach to deal with operator information overload. Their approach permits the operator and the computer to independently determine decisions. The computer is employed for backup decision making purposes when the operator appears unable to cope with all currently required decisions. Their proposed model is shown in Figure 1.13. The "measurement system" determines the operator's performance level and attempts to determine future performance. The "queueing module" describes the interactions amongst the operator and computer servers and the remaining portions of the system. The "predicted allocation module" first determines if the human requires assistance with decision making tasks, then it determines the optimal allocation of tasks between the operator and the computer. The "task allocator" assigns the tasks to the operator or the computer. They found when the operator was assisted by the computer the overall system performance improved while the operator's performance level remained high.

The operator's workload is an important aspect in the system design as all actions the operator is expected to perform will be affected. This review demonstrates the difficulty associated with determining the proper workload level as it varies for each individual.

Usability

A primary concern in interface development is whether the user will like and want to use the interface. Bevan et al. [Bevan et al., 1991] pose the question: "What is usability?". There is no single definition therefore they describe their various views which compose usability:

- The product-oriented view that usability can be measured in terms of the product's ergonomic attributes.
- The user-oriented view that usability can be measured in terms of the user's mental effort and attitude.
- The user performance view that usability can be measured by examining how the user interacts with the product, with particular emphasis on either
  - ease-of-use[8]: how easy the product is to use, or
  - acceptability: whether the product will be used in the real world.
- The contextually-oriented view that product's usability is a function of the particular user or class of users being studied, the task they perform, and environment in which they work.

The elements which they feel compose the determinants of usability are displayed in Figure 1.14. These determinants include those relevant to the user, the task and the environment. The product attributes include the general interface interface and its properties. Finally, they believe the product itself is not usable or unusable but is composed of the attributes which will determine its usability particular to the user, task and environment.

Cox and Walker [Cox and Walker, 1993] define a usable interface as one which is transparent, controllable and flexible. They believe the user must conclude the interface is satisfactory for the designed task. They propose a combination of the following considerations to determine the product usability:

[8]Bevan et al. [Bevan et al., 1991] define ease-of-use as how well the product can be used, whether the operator will use the product and how the user will employ the product.

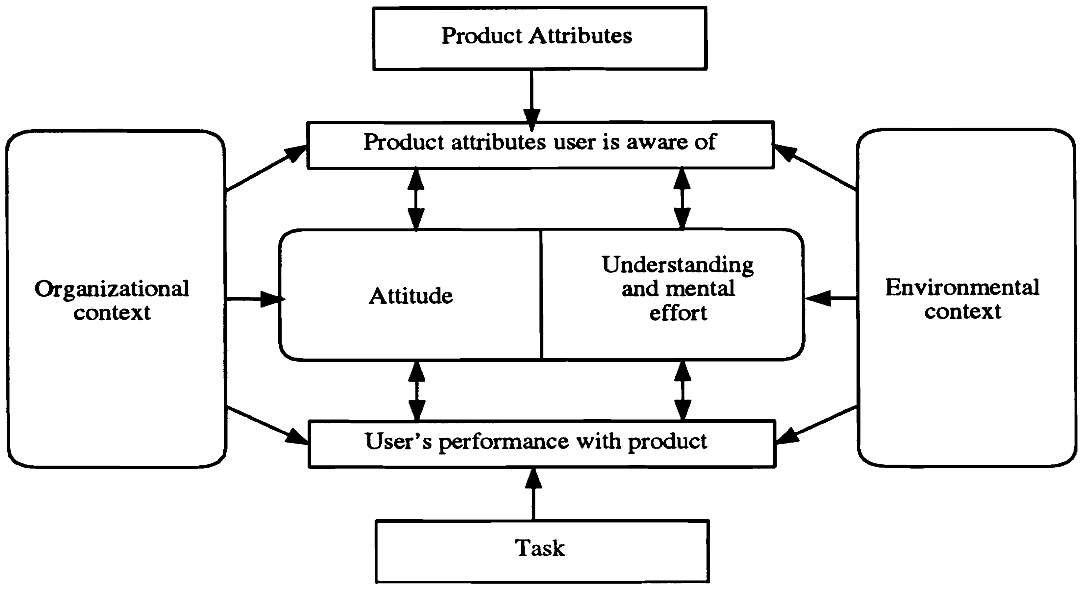

Product Attributes
Product attributes user is aware of
Organizational context
Attitude
Understanding and mental effort
Environmental context
User's performance with product
Task

Figure 1.14: The determinants of usability.

- Functionality: Is the user able to complete the required tasks?
- Understanding: Can the interface be understood by the user?
- Timing: Can the user complete the tasks within a reasonable time frame?
- Environment: Do the required tasks conform to the user's environment?
- Safety: Will the system harm the user?
- Errors: Does the user make many errors during use?
- Comparisons: How does this interface compare to other manners in which the user would carry out the task?
- Standards: Is this interface similar to other interfaces the user may utilize?

Rengger [Rengger, 1991] discusses the review of ten years of published materials on usability conducted in conjunction with the ESPRIT MUSiC project. This review attempted to create generic classes and types of usability measures based upon performance. He determined four classes of usability measures based on the literature review.

- *Goal achievement indicators*
  Indications of the level of success with which user's attained their goals where effectiveness is an indicator of goal achievement. The indicators he

describes are; Success rate, Success ratio, failure rate, failure ratio, success to failure ratio, accuracy, and effectiveness.

- *Work rate indicators*
  Indications of the rate at which users worked or attained their objectives. Terms such as efficiency and productivity being indicators of work rate. The indicators he describe are; speed, completion rate, completion ratio, efficiency, productivity, productivity period, and productivity gain.
- *Operability indicators*
  Indications of the user's ability to make use of the system's features and the level of problems encountered while doing so. The indicators he discusses are; error rate, error density, problem rate, problem density, operability, function usage, and interactive density.
- *Knowledge acquisition indicators*
  Indications of the user's ability and effort to learn, understand and remember how to use a system. The indicators he discusses are; learnability, learning period, and learning rate.

[Heinecke, 1993] and [Prothero, 1994] have also conducted similar studies for measuring usability. [Wiethoff et al., 1991] approached the problem from a biological angle, measuring the user's heart rate etc. [Gunsthövel and Bösser, 1991] employs the SANE (Skill Acquisition NEtwork) model of cognitive skills in their studies and [Gimnich et al., 1991] conducted his studies on a direct manipulation graphical interface. As can be observed, there are many approaches to determining this important measure. The only true measure for each individual system is to test the actual users and ask their opinions.

Fault Diagnosis

Another human factors concern is the human operator's ability to correctly diagnose problems. Rouse and Hunt [Rouse and Hunt, 1984] conducted numerous experiments to test the human fault diagnosis task. This data was utilized to create various models to predict the human's performance for such a task. Based upon these models they deduced:

- Humans are not optimal problem solvers, although they are rational and usually systematic.
- Human problem solving tends to be context-dominated with familiar, or even marginally familiar, patterns of contextual cues prevailing in most problem solving.

- Human's cognitive abilities for problem solving are definitely limited. However, humans are exquisite pattern recognizers and can cope reasonable well with ill-defined and ambiguous problem solving situations.

They found when the human performed suboptimally it was contributed primarily to the human's lack of problem understanding. They also found humans could successfully deal with unfamiliar problems. They concluded that in order to take advantage of the human's cognitive abilities one should develop methods to overcome the human's cognitive limitations.

Sheridan [Sheridan, 1992] observed the need for operator interfaces to be more "transparent" to the actual system. This would facilitate the operator's ability to "see through" the displays and observe the system's actions. He defines human error as "an action that fails to meet some implicit or explicit standard of the actor or of an observer". The following are his proposals for reducing human errors.

1. Design the interface to prevent error, this includes providing proper feedback and redundant information.

2. Properly train the system operators, specifically for emergency situations.

3. Restrict exposure such that actor opportunity is limited.

4. Warn or alarm the operator while not overloading the operator's mental capabilities and creating so many warnings they begin to ignore them.

5. Permit the operator to correct human errors when they occur.

Morris and Rouse study human error and the situations which promote it in [Morris and Rouse, 1993]. They relate that human "slips" typically occur during the automatic execution of routine tasks, when the operator is distracted or preoccupied, is working in environments which are familiar and there are few unexpected events. They also relate human mistakes occur more frequently under the following conditions: when making a decision which requires the simultaneous consideration of numerous variables; prominent environmental cues lead the human to incorrect solutions; when a solution is used which is incorrect for the current failures but was sufficient for previous similar failures; and if the solution must be approached in a new manner. The determination of generalizations which quantify why human operators make errors is very difficult.

The detection of failures is extremely important in any system. The determination of why a human misinterprets failure messages or does not employ the proper methods for failure recover is of grave importance to system designers. This subsection reviewed the methods which may assist the operator during failure situations and demonstrated the difficulty of determining the best methods to assist the operator during a failure.

1.3.6 Human-machine System Mediation/Intervention

A major feature of our human-machine interface is the ability of the operator to intervene into all system levels. Traditionally, most human-machine interfaces permit the operator to act as a monitor or supervisor without permitting significant interaction into the systems processes. This section discusses the work of others who have employed intervention/mediation into their systems and states how ours is different.

Sheridan [Sheridan, 1992] states:

> the supervisor intervenes when the system state has reached the designated goal and the computer must be retaught, or when the computer decides the state is sufficiently abnormal and asks the supervisor what to do, or when the supervisor decides to stop the automatic action because the system state is not satisfactory.

The intervention point may be influenced by: the criterion which define an abnormal state; the tools which the supervisor can employ for intervention; the criteria for risk-taking; the decision on whether to wait until more information is collected or to intervene immediately; or the supervisor's mental workload. He notes that the intervention stage is the most error sensitive, as human error is more apparent at that time.

Ammons [Ammons, 1985] suggest models for aiding real time flexible manufacturing systems. She believes the control loop must permit the operator to supervise the automation, to monitor the system and intervene to diagnose failures and either correct or compensate for failures. She proposes three interaction levels: production planning; release scheduling and item movement. This hierarchy is very manufacturing systems specific but does permit the operator to interact with the system levels. This is similar in principle to our model but ours is generally applicable to various systems and permits interaction into the entire system.

McKee and Wolfsberger [McKee and Wolfsberger, 1988] proposed a graphical human-machine interface system. This interface was proposed to permit the human operator to take control from the robotic system while also allowing it to work autonomously. Their ideas are very similar to our underlying principle but this system was never developed.

Hasegawa et al. [Hasegawa et al., 1990] incorporate a high level of intervention into their telerobotic system. When they encounter a system failure during autonomous execution the operator issues a command which switches the system to master-slave control. After the operator recovers from failure, the system switches back to autonomous mode. Their employment of master control is similar in principle to our system. Although, we permit the operator to take control away from the system or to employ intervention into the individual processes and then allow the system to continue its task autonomously.

Hirai et al. [Hirai et al., 1992] have incorporated multi-level human interaction into their MEISTER (Model Enhanced Intelligent and Skillful TEleoperational Robot) system. They permit the operator to teach the system information via their teaching-executing method. Tasks are simulated and the operator either approves the task or manually controls the task execution. They permit the operator to superpose control schemes onto the system. The *rate control scheme* permits the operator to create precise linear motions in the joint angle or Cartesian coordinate spaces. The *incremental control scheme* permits the operator to increment the slave manipulator position to an amount specified by the operator. The *indexing scheme* permits the master manipulator to be posed at a position which is comfortable for the operator while the slave manipulator is posed for task execution. The *Programmed Control Scheme* permits the operator to create commands utilizing the master manipulator and while the slave has been working autonomously it will then execute the operator created commands. While they superpose these methods onto the slave manipulator they are not interacting to all levels of the system in the manner we propose.

The human machine interface applied to multiple robots is currently being developed at the Chemical Engineering Laboratory RIKEN [Yokota et al., 1994, Suzuki et al., 1994]. While they report the interface will permit the operator to interact with the system when necessary they do not indicate this has actually been implemented. They have developed a sophisticated communications architecture for their system and a prototype of a two-dimensional human-computer interface.

Chapter 2

Multiple Agent Supervisory Control System (MASC) and Application Description

We developed a human-machine interface to interact with multiple mobile agents. The interface has been developed for the University of Pennsylvania General Robotics and Active Perception (GRASP) Laboratory's multiagent project in conjunction with four other graduate student project members. Therefore, this Chapter provides a brief overview of the multiagents project and then the Multiple Agent Supervisory Control System (MASC). As this interface has been developed in conjunction with the multiagents project, most of the software processes which we have integrated into the MASC system were developed for the multiagents project.

2.1 Multiagents Project

The purpose of the University of Pennsylvania General Robotics and Active Sensory Perception Laboratory's multiagents project has been to investigate the coordination and monitoring of multiagent systems for intelligent material handling and is described in [Adams et al., 1995, Bajcsy et al., 1992]. The contributions of this work have been an improved understanding of the fundamental problems underlying the multiple agent control and coordination as well as the development of algorithms for intelligent exploration, organization and coordination of multiple agents.

2.1.1 Multiagents architecture

The multiagents system is composed of four mobile platforms and the human-machine interface. The mobile agent bases are TRC Labmates. A six degree-of-freedom manipulator is mounted on each of the manipulatory agent bases. Since the platforms have three degrees-of-freedom, there exists an extra degree-of-freedom in the three-dimensional Cartesian space. Therefore, they are defined as redundant manipulators, for more information see [Wang, 1995, Wang and Kumar, 1995]. The observation agents are equipped with various sensing modalities and a general purpose workstation (SPARC2) and utilize *WINDATA*™ radio ether net communications. The human-machine interface has been developed on a Silicon Graphics Inc. *Indigo*^{2}. The low level software interface, PENGUIN [Sayers, 1993], was developed by Craig Sayers. It is written in C++ and employs X/Motif windows and the Silicon Graphics Inc. Graphics Library. The higher level interface is also written in C++ and employs the Silicon Graphics Inc. ImageVision Library as well as the TCP/IP communications software.

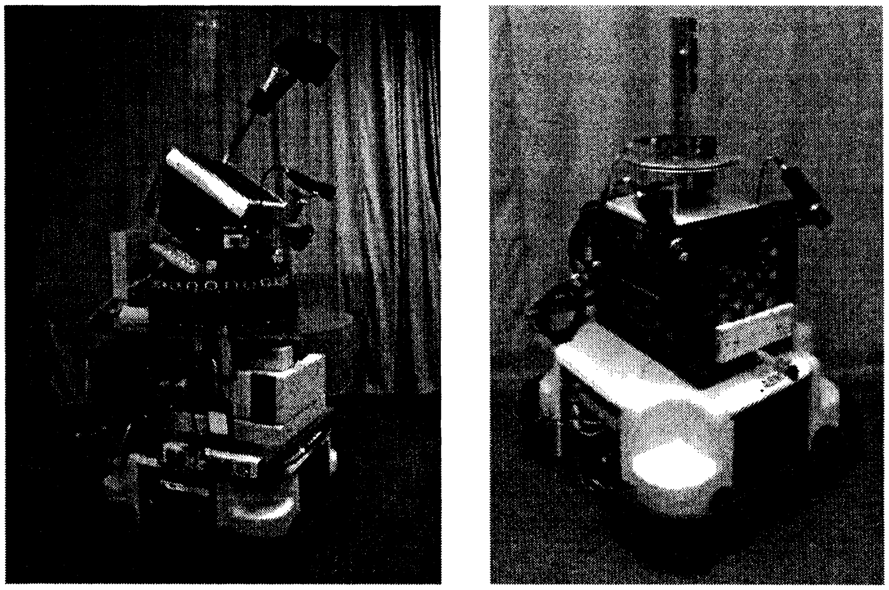


Figure 2.1: The Observation Agents: (a) SensorBot, and (b) VisionBot.

The two observation agents are portrayed in Figure 2.1. The sensorBot, (a), is equipped with the following sensing modalities:

- a partial ring of sixteen POLAROID™ ultrasound and infrared sensors,
- a stereo camera pair,
- a light-striping device which projects three planes of light in front of the agent on

the ground and a camera offset vertically which uses elementary projective geometry to detect when an object intersects any of the light planes.

The ultrasound sensors are used to detect objects in the environment. As this sensing modality may be unreliable, [Mandelbaum, 1995], the infrared sensors are used to verify the ultrasonic readings. This information is currently employed to detect features in the environment such as wall like objects and corners. This information may also be utilized to localize the SensorBot relative to these features. A stereo algorithm computes a localized correlation and extracts three-dimensional information about the environment. The stereo pair may also be employed for visually guided obstacle avoidance described below. The light-striping mechanism is employed to detect objects and then obtain the two-dimensional object information. It may also be utilized to localize the other agents relative to the localized SensorBot. The odometry readings are utilized to monitor the agent.

The VisionBot, Figure 2.1(b), is equipped with a stereo pair as well as a pan platform. The stereo pair are employed for visually guided obstacle avoidance which detects obstacles in the environment and then avoids them. The obstacles are detected via the difference between the stereo images after applying the inverse perspective mapping. This mapping is used to construct a free space map for the common field of view from each camera, see [Kosecka, 1995] for further details. The pan platform is composed of a camera and a turn table. This mechanism is used to track objects or other agents in the environment, further details can be found in [Kosecka, 1995].

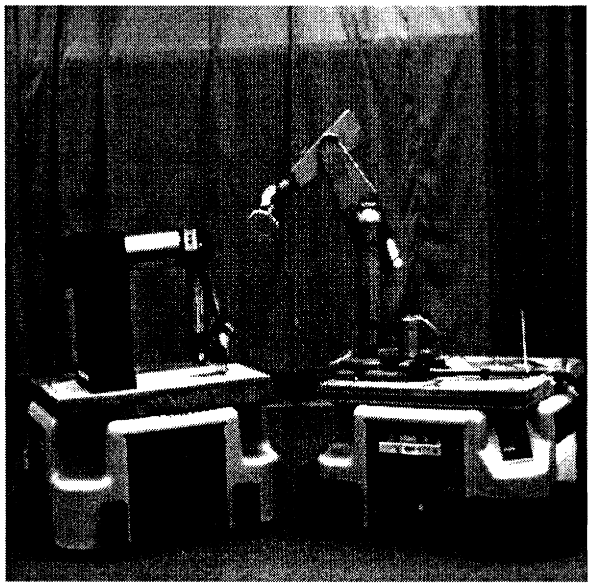


Figure 2.2: The Manipulatory Agents: ZebraBot (left) and PumaBot (right).

The PumaBot, Figure 2.2 (right), is equipped with a Puma 260 Manipulator and the ZebraBot (left in the figure) is equipped with a Zebra-ZERO Manipulator. As mentioned these are six degree-of-freedom manipulators. These agents have no sensors other than force feedback and are therefore "blind". They are primarily employed for the manipulation and relocalization of objects in the environment. The primary algorithms developed with the agents have been for testing the coordination of manipulation and locomotion as well as methods for redundant robots. The coordination of manipulation and locomotion considers the best configuration to carry objects and how to reconfigure the platforms to avoid obstacles or to pass through small passage ways, while maintaining the carrying task. Full details of this work can be found in [Yamamoto, 1994, Yamamoto and Yun, 1992, Yamamoto and Yun, 1994]. The redundant robot research focuses upon "the determination of joint motions for a given end effector displacement in kinematically redundant manipulators" and is described in [Wang and Kumar, 1995, Wang, 1995].

2.1.2 Multiagents experiments

The observation agents are employed to lead the "blind" manipulatory agents. This will occur via communications received from the SensorBot and VisionBot. Thus when executing tasks the observation agents not only gather sensory data for themselves but also for the manipulatory agents. The observation agents are required to sense whether passage ways are wide enough for the manipulatory agents to pass through when carrying objects in the side by side configuration or instruct them to change to a follow the leader configuration. As the observation agents are not equipped to physically monitor the actions of the manipulatory agents this task will be assigned to the human supervisor. One of the observation agents will be positioned such that optimal images and other available sensory information can be provided to the human supervisor for the monitoring of the manipulatory agents.

The current experiments employ the manipulatory agents to carry an object while the SensorBot explores the path specified by the human supervisor. The SensorBot's task is to determine if a wide enough path exist for the manipulatory agents to pass through to the goal as well as to advise them of the existence of all obstacles or passageways. The VisionBot will be employed to remain in a position from which images of the manipulatory agents may be returned to the human supervisor. Thus the VisionBot will follow the manipulatory agents to the goal point.

2.2 MASC System Overview

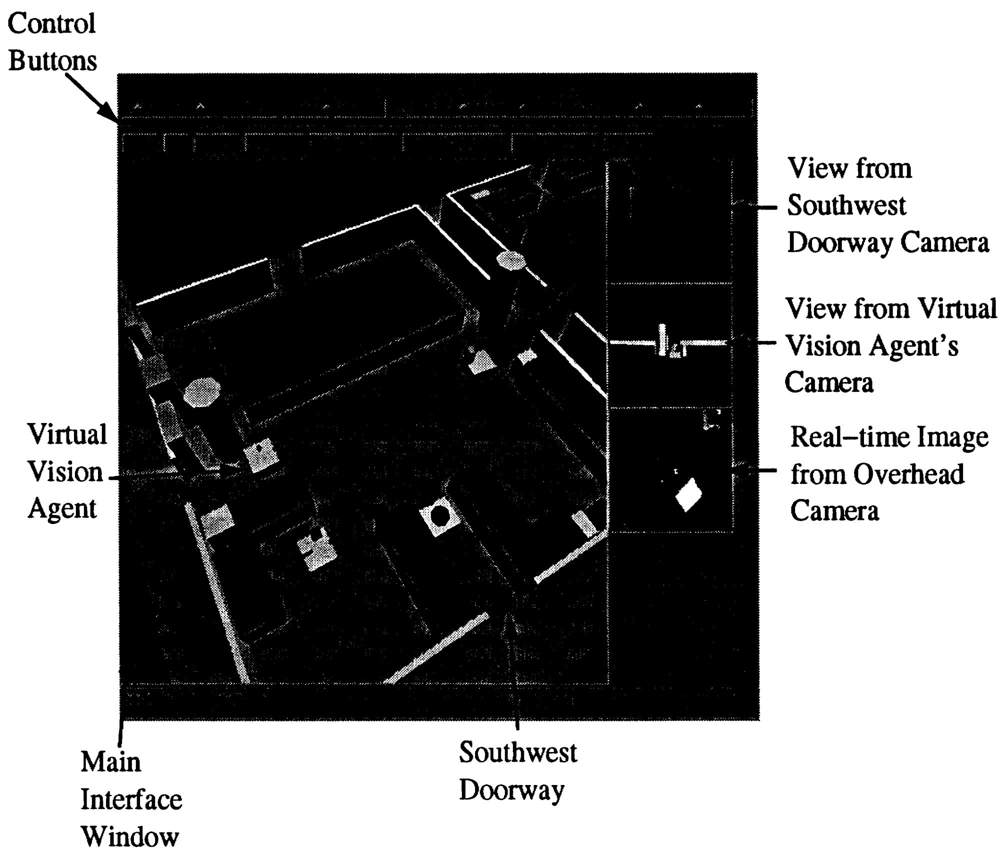

Control Buttons
View from Southwest Doorway Camera
View from Virtual Vision Agent's Camera
Real–time Image from Overhead Camera
Virtual Vision Agent
Main Interface Window
Southwest Doorway

Figure 2.3: The MASC system interface.

The Multiple Agent Supervisory Control System (MASC) is a human-machine interface. It has been designed in such a manner that it may be applied to any number or type of robotic agents, shown in Figure 2.3. The individual robotic agents, their associated manipulators and processes may be controlled by the supervisor through MASC. Our objective is to create a semi-autonomous system which successfully completes assigned tasks.

The human's primary task is to "supervise" the agent's actions during execution [Sheridan, 1992]. Through MASC, the human supervises the system while observing sensory data and images. The supervisor is permitted to assist the agents when requested and may assume control of an agent when necessary. Each agent is composed of multiple control and processing levels. MASC must permit the supervisor to interact with these levels for the successful semi-autonomous execution of feasible tasks. This interaction will permit the supervisor to revise incorrect agent decisions and reconfigure the system after partial system failures.

2.2.1 MASC System Layout

The MASC interface provides the supervisor with a three-dimensional environmental view. The main working window is the large window in Figure 2.3. The supervisor specifies necessary information on the model within this window. The default view is a birds eye view. The supervisor may rotate, zoom or translate the view to accommodate her or his current requirements, as shown in Figure 2.3. The right portion of the interface is employed to display two images of the graphical model, as well as images and process data (such as state diagrams). The full interface window will permit the display of up to eight such windows. The top window in this area displays the view of the virtual model from a camera placed in the lower doorway of the model, the "Southwest Doorway" in this figure. The second virtual view is that of the camera positioned on top of the VisionBot, which is marked the "Virtual Vision Agent". The third window displays an overhead camera image of the environment. The interface also provides a set of control buttons displayed on the top of the interface which are marked "Control Buttons" in the figure.

2.2.2 MASC System Control Buttons

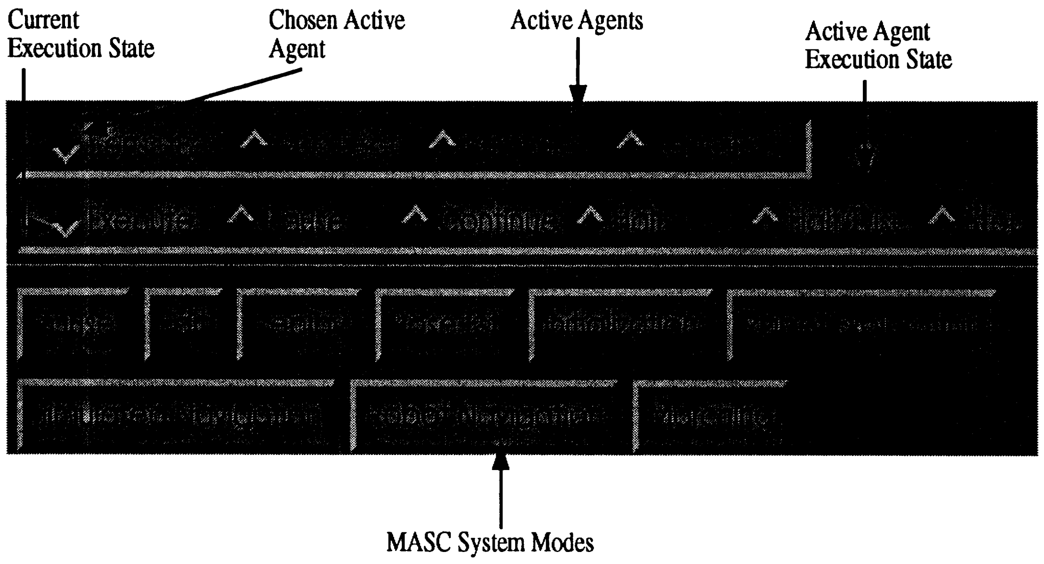


Figure 2.4: The MASC system Control Buttons.

The supervisor communicates with the agents through the MASC interface. We have provided display push buttons, termed control buttons. The control buttons are displayed in Figure 2.4. They allow the supervisor to specify system information. The top two sets of control buttons permit the supervisor to define which agent she or he wishes to control at a specific instance. There are two sets of buttons on this level. The top set lists the agents which are active in the system and are termed the "Active Agent" buttons. An agent which is active in the system is an agent expected to partake in the given task execution. The second set of control buttons specify the chosen agent's current execution state in the top set and are termed the "Active Agent Execution State" buttons. The options include pausing an agent's current task execution, continuing a paused task execution, halting an agent's task execution and removing all further commands from its command array as well as the issuance of an emergency stop command. The bottom set of control buttons permit the supervisor to alter the current system state and execute various processes. These buttons are termed the "MASC System Modes" buttons. The supervisor may choose from initialization, exploration, navigation or replay modes.

2.2.3 MASC System Modes

The initialization mode permits the human supervisor to run any processes which may be required prior to task execution, such as the task planning and assignment to the active agents. The Supervisor may edit the graphical model or add objects to the world model from the overhead camera image overlay. The purpose of this system mode is to prepare the MASC system for task execution.

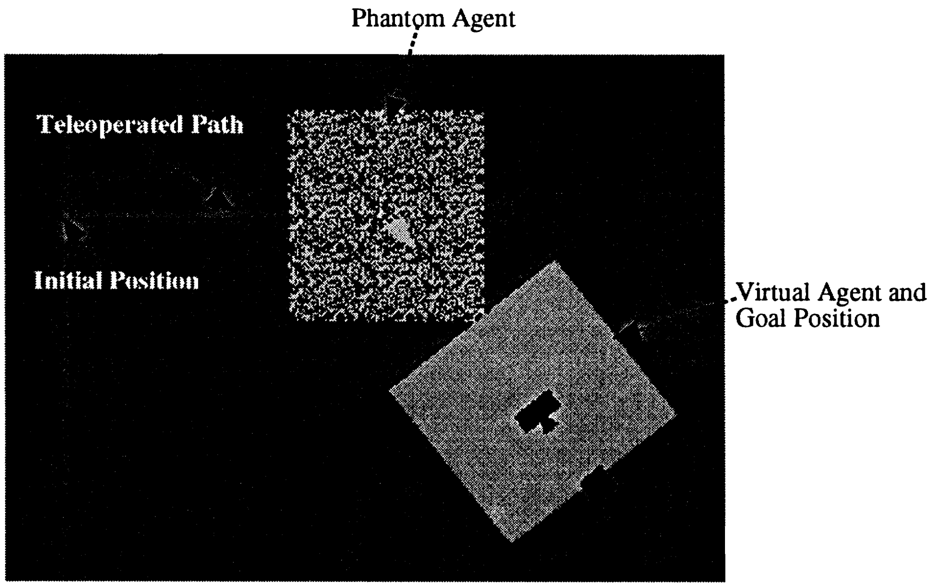


Figure 2.5: The "phantom agent" during teleoperation.

The exploration mode is a teleoperation mode. In this mode the human may teleoperate an agent to create locomotion commands. While working within this system mode, there exist a "phantom" agent which is similar in principle to Bejczy et al. "phantom" agent described in [Bejczy et al., 1990]. The purpose of this agent is to inform the supervisor of the real agent's actual position. As the supervisor teleoperates the virtual agent to create commands, its position no longer corresponds to the actual agent's position. Thus, the "phantom" agent, as shown in Figure 2.5, is updated with the actual agent's heading and odometry readings. As can be seen in the figure, the solid black lines and white triangle (underneath the "phantom" agent) represent the creation of a move command, a rotation command and a final move command. The virtual agent's position corresponds to the actual agent's goal position. The "phantom" agent is shown as a white semi-transparent replica of the virtual agent. Its position corresponds to the actual agent's position while executing the three commands. In this figure, it has completed the first move command and is about to begin the rotation and final move commands.

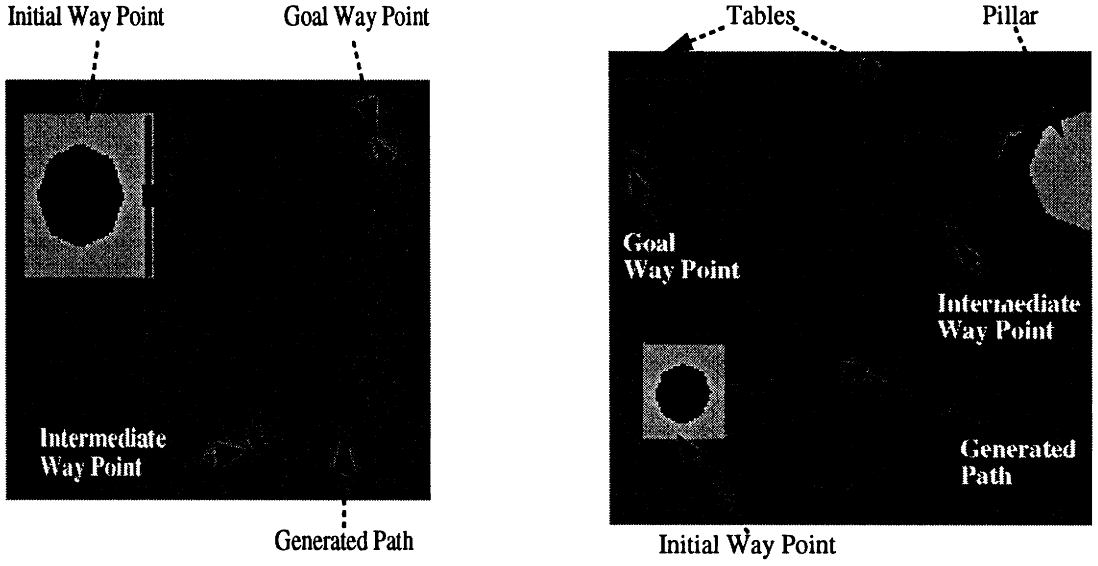


Figure 2.6: (a) The path created by the local R-geodesic path generator, and (b) The path created using the global path planning server.

The navigational mode is an autonomous mode. It permits the supervisor to drive the agents based on path plans created by one of the two MASC system path planners. The local path planner does not consider environmental information and is an R-geodesic path generator described in [Adams et al., 1995, Wang, 1993]. This planner is employed to plan short paths in well-known environments. Figure 2.6(a) displays a path planned with this planner. The agent's current position is considered the initial starting way point. There is an intermediary way point along with the goal way point. The black triangles represent the way points. The direction in which the triangles face determine the actual agent's heading. The solid line represents the path returned by the planner. The set of global path planners are managed as a path plan server at Stanford University. This server is described in [Becker, 1994] and is composed of the potential fields and cell decomposition planning methods as described in [Latombe, 1991]. This planner considers all objects in the environment. The path planned employing the global path server resolution potential fields method is displayed in Figure 2.6(b). Again, the robot's current position is considered the initial way point specification. There exist intermediary and goal way point specifications, represented as black triangles. The returned path is displayed as a solid line passing through the defined way points. From this figure, it can be seen that the planning method considers the objects in the environment when computing the path as the computed path avoids the tables and pillar in the environment.

The replay mode permits the supervisor to replay the task execution within the last five minutes. This mode is very helpful when a problem arises and the supervisor does not recall what actions occurred. While this option does not permit raw image data replay, it replays all other sensing modalities data displays. It replays a single virtual (and "phantom" if in exploration mode) agent's actions as well as any combination of the active agents. The replay begins after the supervisor has specified the agents and the replay time frame. Once the replay has completed the supervisor may respecify the agents and/or the time frame and replay again. If a particular agent was inactive during the specified time frame the supervisor is notified. If only an inactive agent is chosen for replay, no replay is provided. If other agents are also specified, the human is notified of a particular agent's inactivity and the replay continues with the other specified agents. This particular option is helpful for diagnosing uncertain situations.

Chapter 3

Mediation Hierarchy

3.1 Motivation for Development

Many human-machine interfaces interact at a high level but do not permit the human to interact with the various low levels of the system. When creating such an interface for a semi-autonomous robotic system, this may be a desirable feature. The robotic system will eventually find itself in a situation where it is unable to correct for an error and/or to right itself to its original goal. Interaction to all system levels will permit the human supervisor to interact with the processes and correct such situations. This interaction may be requested by the agents or the human may determine there is a problem and intervene. This aspect will allow the human to correct the system and then permit it to autonomously continue its original task. This approach is feasible for such a robotic system as the one described in Section 2.1.1. The idea is to build a more robust system which can accommodate problems through the human-machine interface rather than direct physical interaction with the system. Our hypothesis:

> With the addition of the supervisor's ability to interact with all levels of a semi-autonomous system, the supervisor will be capable of correcting problem situations and the system will successfully complete assigned tasks.

3.2 Mediation Hierarchy Description

A mediation hierarchy consisting of four levels has been formulated [Adams and Paul, 1994a, Adams and Paul, 1994b]. These levels define the various intervention types into the differing robotic system levels. The hierarchy furnishes the supervisor with the capabilities to interact with all levels. This interaction should permit the supervisor to correct situations which would cause a fully autonomous system to become unstable and possibly fail its task execution. It is important to note the supervisor only interacts with the agents when assistance is requested by the agents or when the supervisor detects a situation where she or he deems it is necessary to intervene on an agent's behalf.

3.3 Level Descriptions

3.3.1 Task Level

There are numerous tasks which one would propose to assign a robotic system. One manner by which to break up a task and assign the proper action set to each agent is to "hard code" the tasks and actions into the system. Unfortunately, this approach inevitably limits the number of tasks the system can execute and does not create a general system. In order to create a general system which executes various tasks, the supervisor, or a task planner, must derive the proper assignments. Since the system will not execute a task until these actions are taken, the *task level* resides atop our mediation hierarchy, see Figure 3.1.

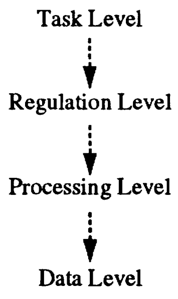


Figure 3.1: Hierarchical levels of human interaction.

The *task level* permits the supervisor to specify the actions an agent, or a group of agents, are to execute to complete an assigned task. Tasks may include environmental exploration to assist with model building, following an assigned path to a goal, observing the task execution assigned to another agent, moving in a configuration, carrying items such as pallets, and the navigation necessary to transport items from one location to another.

3.3.2 Regulation Level

There exist minimal interactions which are necessary between a human-machine interface and a robotic system. If an agent is on the verge of colliding either with another agent or an obstacle, the supervisor should be able to prevent such a collision. If it is necessary for one agent to complete a task before another agent begins its task, the second agent may need to be informed to wait while the first agent completes its execution. The supervisor possesses a means of monitoring an agent's actions. This may include video images, sensory data or positional update displays. It is essential that the interface permit the supervisor to choose such information for monitoring purposes. Also, in such a system, the agents processes may require information from the supervisor in order to begin processing. The interface must facilitate the means of providing this information. The *regulation level* couples these interactions into one mediation level as displayed in Figure 3.2. We have developed three interaction types on this level, *control interaction*, *request interaction* and *specification interaction*, which we define below.

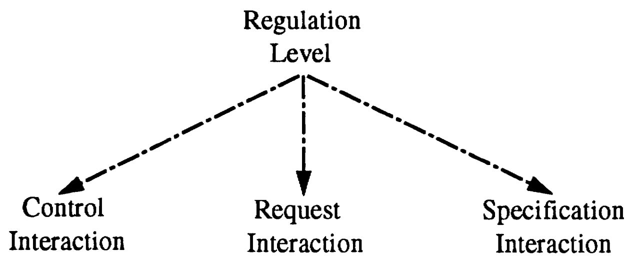


Figure 3.2: The interaction of the *regulation level.*

Control Interaction

MASC provides the supervisor with the capabilities to cope when an impending collision must be avoided by issuing an emergency stop command via the control buttons described in section 2.2.2. The supervisor may pause one agent's task execution if it must wait for another agent to first complete its execution. These interaction types are created via the *control interaction.* Also included in this interaction type are situations when the supervisor may teleoperate the agent to create locomotion commands. The supervisor may employ teleoperation as an alternative to the path planning methods or to assist the agent. The supervisor would assist the agent when the agent finds itself in a situation where it is unable to determine its next action. Such a situation may be a dead end passageway. The supervisor would teleoperate the agent to a location where it could then autonomously continue its task execution. Formally, *control interaction* provides the supervisor with the ability to control the agent's progress while executing a task either for the purpose of deterring or assisting progress.

Request Interactions

Systems contain various information which may be useful to the supervisor at different times throughout the task execution. The objective is to avoid overloading the supervisor with too much information [Sheridan, 1992, Whalley, 1992]. *Request interaction* permits the supervisor to request the sensory data and processed information from the agent's only when it is needed for error detection and/or monitoring purposes. When the supervisor no longer requires this information, she or he can inform the agent's processes to cease transmission.

Formally, the *request interaction* permits the supervisor to request information directly related to the current task. This information is then employed by the supervisor to monitor the task execution. If the supervisor believes a process is making incorrect decisions, she or he may request more information to assist with the problem detection. This information may include images, ultrasound sensors or vehicle position. The supervisor then reviews this information and draws conclusions as to process' behavior.

Specification Interaction

Various processes require information from the supervisor prior to the commencement of processing. Such a process may be a path planning process for which the supervisor must specify the desired path's starting, intermediary and goal way points. The process will then utilize this information and return a path for the supervisor to review, modify and approve. The *specification interaction* permits this interaction between a process and the supervisor. Formally, *specification interaction* provides the supervisor with the means to interactively specify information pertinent for a process' execution.

3.3.3 Processing Level

There exist instances when a process may be incapable of determining a correct decision based upon ambiguous information and must therefore request supervisory assistance. There also exist situations when a process will formulate a correct decision in a local context but the decision will not be correct in the global scheme, therefore the supervisor should either assist with the decision making process or override the decision formulated by the process with a correct decision.

While observing an agent's actions based on a particular process, the supervisor may determine the process is formulating its calculations based upon an incorrect interpretation. The supervisor may then intervene in the process to clarify the information, override a decision or allow it to continue with its processing. The supervisor should be capable of supplying variables, data and various processing decisions through this intervention level to properly direct the process with the task at hand. For instance, assume an agent is employing visually guided obstacle avoidance and another agent momentarily passes within its viewing field, in this case the visual agent would interpret the moving agent as an obstacle and begin the obstacle avoidance task. The *processing level* permits the supervisor to override the decision to avoid the "obstacle" and instruct the agent to continue with its original assignment. Formally, the *processing level* permits the supervisor to aid a process when it is unable arrive at a decision and to rectify incorrect decisions deduced by a process either upon the process' request for assistance or as determined by the supervisor through monitoring. This level of interaction will protect the agents from entering unstable states.

3.3.4 Data Level

It is known that from time to time mechanical devices fail, and that the automatic reconfigurations for such failures are not always successful, therefore, the supervisor should be provided with the means to reconfigure the system. The mediation hierarchy's *data level* will permit the supervisor to reconfigure the system when an automatic reconfiguration has failed.

The outcomes determined by the higher-level processes are dependent upon correct input data. If this data is not correct, the processes will likely formulate incorrect decisions and commands which may force the agent into an unstable state. The *data level* will also supply the supervisor with the ability to ensure processes receive correct data for interpretations. For instance, as mobile agents move throughout the environment executing an assigned task, they accumulate errors in their positional and heading readings due to wheel slippage. If an automatic reset fails, it may become necessary for the supervisor to reset the readings based upon localization information. Alternatively, assume the camera focus from a camera pair has been corrupted, this may hinder the information retrieval for the process using these images. The supervisor should be able to inform the process to stop image processing and instruct the agent to rely on another sensing modality to complete its task assignment.

Formally, the *data level* permits the supervisor to ensure correct data is passed up through the system for interpretations and processing. It also allows the supervisor to reconfigure the system during a hardware failure. This interaction type implies that as data flows upward through the system, the system will correctly interpret the data implying correct actions will be executed which in turn imply the successful task assignment execution.

Chapter 4

Multiagent's Process Integrations into the Mediation Hierarchy

4.1 Task Level

The main purpose of this level is to create plans which the agents employ for task execution. Previously, it was the human supervisor's task to determine the task plan. Currently we are incorporating the global task planner, ItPlanS, to assist the supervisor.

The original ItPlanS planner does not incorporate human interaction, therefore, we are currently designing the interactions we will create with the planner. The initial interactions will automate the initial planner specifications. We will also permit the human supervisor to redefine the plan search ordering. These interactions will permit the supervisor to assist the planner while not degrading its processing. We are also creating the knowledge base required for the tasks we will execute with the multiagents system.

4.2 Regulation Level

4.2.1 Control Interaction

At this level of interaction we have provided the human with the ability to teleoperate the vehicles using the mouse to create control commands. The human may create a move command by clicking the mouse button and moving the virtual agent, a rotation about the zero radius by clicking the mouse button and rotating the virtual agent or a combination of move and rotation about various radii (similar to a draw type mode) by clicking the mouse button and moving the virtual agent about the model.

Also, control buttons are created for all active system agents. These buttons are described in Section 2.2.2 and in [Adams and Paul, 1994a, Adams and Paul, 1994b]. They permit the supervisor to take control of a specific agent. Currently, when the supervisor desires to change control from the current agent she or he must first instruct this agent to pause, stop, halt or continue with its task execution. Once the supervisor provides such instruction, the top row of control buttons (in Figure 2.4) become active and permit the supervisor to choose another agent to directly control. As soon as the supervisor chooses an agent the top row of control buttons becomes inactive until the "active agent execution state" is changed from "execute". The human may then create a new path plan, teleoperate or issue a pause, halt or emergency stop command for the specified agent.

4.2.2 Request Interaction

The MASC interface data displays permit the supervisor to request data from any available sensing modality. MASC does not automatically display sensory data or attempt to decide which data should be displayed at a particular instance. The supervisor is responsible for requesting the pertinent data. The supervisor may request all sensory data from any system mode.

The supervisor may request any raw system camera images. This information is displayed to the right of the main window as in Figure 2.3. Certain images are also overlaid onto the MASC model in a manner similar to the free space map overlay in Figure 4.4 (bottom). These images are of particular importance when monitoring other agent's task executions. They also assist the supervisor in verifying sensing modality information, such as the detection of an obstacle or passageway.

All agents record their odometry and heading readings during task execution. This information is employed to update the position of the respective virtual or "phantom" agent. It is well known as mobile agents move throughout their environment they accumulate errors in their odometry and heading readings due to slippage. This slippage can not be detected by the human supervisor through MASC. For instance, if someone or something came into the environment and picked up an agent while it was executing a command and then placed it back on the ground at completion, the virtual agent would appear as if the actual agent had successfully completed the command. Thus, it is necessary to employ a localization procedure. Once the agents localize themselves using another sensing modality, this information can be employed to update the virtual agent's position.

The SensorBot's raw ultrasound and infrared sensors may be displayed directly upon the model as in Figure 4.1. As explained in Section 2.1.1, the ultrasonic readings may be

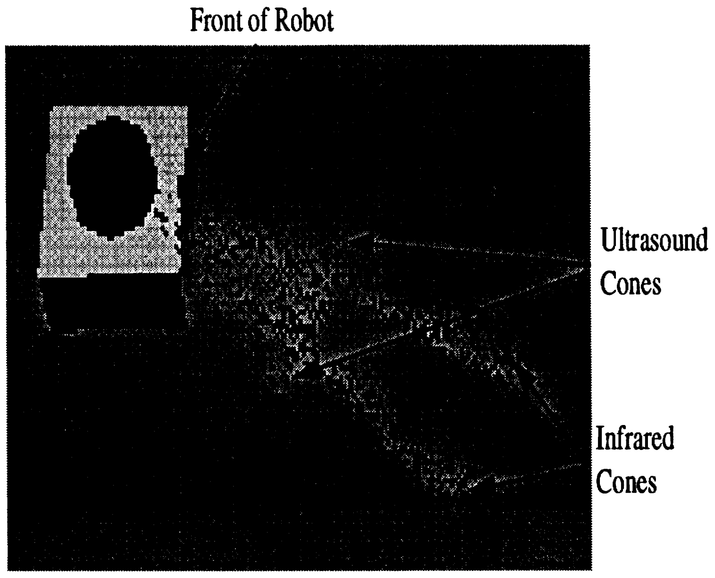
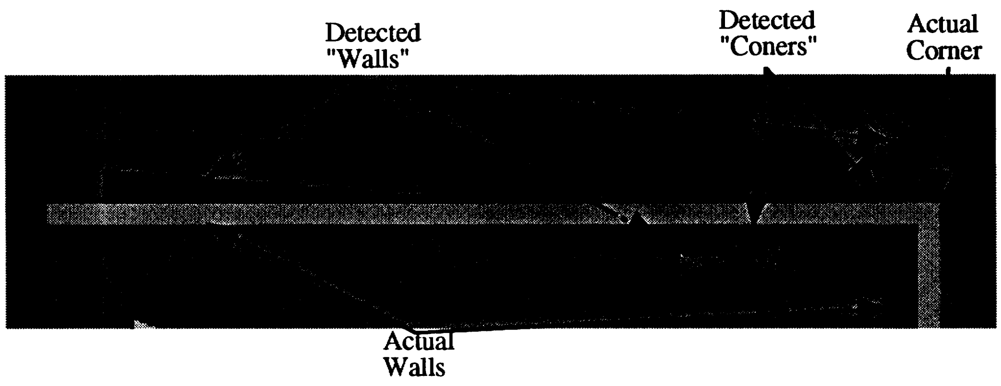


Figure 4.1: The display of the raw ultrasonic and infrared sensors.

unreliable therefore, the infrared sensors are employed to verify these readings. When there are no reflections or the reflection is beyond what is considered an "accurate" reading, no sensor information is displayed. Since there is a belt of sixteen sensors for each modality, we display the readings from the corresponding actual sensory position on the virtual agent. As can be viewed in Figure 4.1, the ultrasonic readings are portrayed as cones displayed to the actual reading distance (which have a 30 degree arch at full length of an "accurate" reading). Since one is unable to detect where along the arch the reflection occurs, the virtual displays are created to match the possible reflection area. The cones are transparent so the supervisor may view other information in the model which coincides with the sensor displays. When utilizing infrared sensors, one is unable to determine the distance at which a reading is reflected, thus the infrared sensors are either on or off. When the infrared detects a reading, the respective cones are displayed as smaller cones to the entire predetermined distance as it is unknown.

The ultrasound sensing modality is also employed to detect features or objects in the environment. When sufficient data has been collected and the ultrasound process determines an object exist based upon the number and confidence of the readings, this information is passed to MASC. For instance, the process can detect walls and corners. This information is displayed on the model as in Figure 4.2. The readings are clustered into two data groups, "tangent-segment clusters" which represent objects similar to walls and "corner clusters" which may be corners. Wall like "clusters" are displayed as wide lines at the



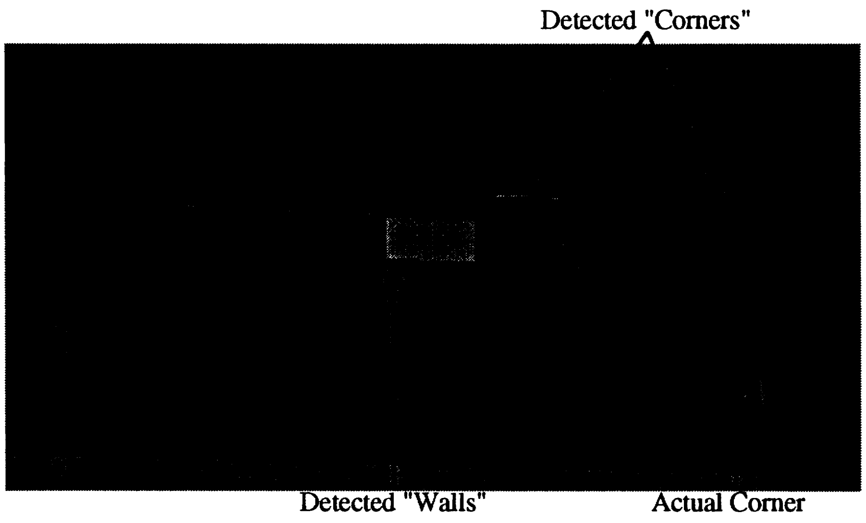


Figure 4.2: (top) The display of the detected features from a bird's eye view, and (bottom) The display of the detected features from the view of the agent.

actual sensor's height. These "clusters" are displayed in various shades of red (shades of grey in the figure) which portray the ultrasound process' confidence that this object actually exist. The lighter the shade of red the less confidence and the darker the shade the more confidence as to the object's existence. This shading is also utilized to represent the confidence of a corner's existence. The "corners clusters" are displayed as cylinders, also at the actual sensor's height as in Figure 4.2. All "clusters" below a predetermined value are not displayed. The "clusters" can then be utilized by the supervisor to verify the existence of such features or objects in the model. This process may also be used to localize the SensorBot relative to the objects it detects. We will also display this data. The current integration state is high level as we only receive information from the process.

The SensorBot's light-striping modality offers images and a two dimensional "footprint". The image can be displayed on the right of the model, as in Figure 2.3 and will be overlaid onto the model to verify the data produced by the process. We will include the protocol development to request process' information. The data display will include the two-dimensional polygons on the model. The overlaid image will be employed to verify the location information for the polygon. An object's "footprint" will either be classified as a new object, verification of an existing modeled object or to determine an agent's location.

The SensorBot and VisionBot are both equipped with a stereo camera pair. As discussed in 2.1.1, this hardware may be employed for visually guided obstacle avoidance. We have integrated a version of this process at a high level. Therefore, the human supervisor is capable of process monitoring but is unable to interactively instruct the process in problem situations. This process is actually composed of three processes, the obstacle avoidance, a path follower and supervisor processes. The obstacle avoidance detects the object and creates the commands to avoid it. The path following process

- monitors the given path execution either determined by teleoperation or a path planning methods;
- sends the commands generated by the obstacle avoidance process or the human supervisor to the robot control process; and
- determines how to return to the previously defined path.

The supervisor process is a Discrete-event system (DES)[1] supervisor who's task is to monitor the communications between the processes.

[1]As defined by Ramadge in [Ramadge and Wonham, 1989]. A Discrete-event system (DES) is a dynamic system in which state changes occur at discrete points in time in response to the occurrence of certain isolated events.

Aside from the availability of the camera's images, this process produces state diagrams (such as the one in Figure 4.3), and a free space map. The obstacle avoidance and path following processes supply MASC with the current processes' state information. The state digram is updated with the process' current state (this figure displays the state diagram for the obstacle avoidance process). For instance, the state diagram in the figure informs the supervisor the robot is "avoiding to the right" of the obstacle, the highlighted box in Figure 4.3. The previous state would have informed the supervisor that an obstacle was

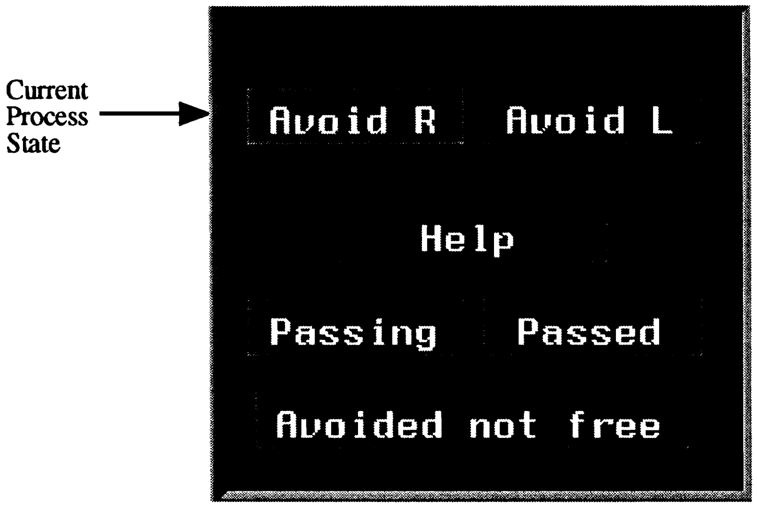
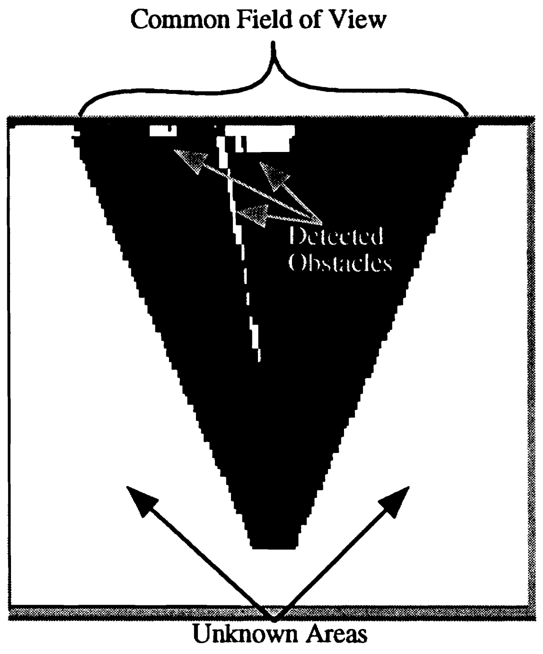


Figure 4.3: The state diagram display updated by the visually guided obstacle avoidance process.

detected. Once an obstacle is detected the process decides from which side to avoid the obstacle based upon the free space it detects surrounding the obstacle. As the obstacle is avoided the process will update the state diagram with states as follows: "passing obstacle" then "passed obstacle" and then it will return to "free space" as it no longer detects the obstacle. When the supervisor has assigned the agent a task with a specified path to follow and the agent begins to deter from the assigned path, the supervisor must diagnosis why. The state diagram provides this information.

The free space map may be either displayed in a window to the right of the interface, as in Figure 4.4 top or may be overlaid onto the model displayed in at the bottom of the figure. The overlay moves with the virtual or "phantom" agent as it is updated with the current odometry and heading readings. Occurring simultaneously, the free space map is updated with its latest version. In both figures, the dark "V" portion represents the common field of view for the stereo camera pair while the lighter area surrounding the "V" is unknown and uncommon to the stereo camera pair. Obstacles are represented as



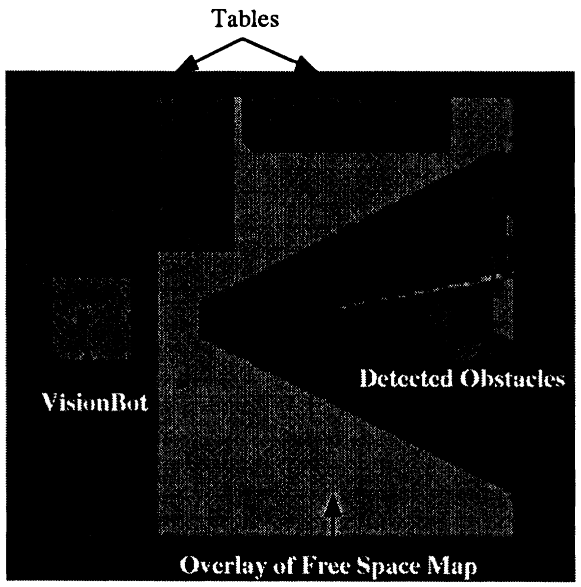


Figure 4.4: The free space map as displayed in its own window (top) and as overlaid onto the model (bottom).

white (or lighter color) areas within the common field of view. In the figures, there are three such areas with the remainder of the common field of view pertaining to free space. This process has been developed to run in real-time, therefore, it does not extract exact obstacle information. The free space map overlay provides the operator with an idea of the obstacle's location, but since the inverse prospective mapping projects behind the object, it is difficult to determine exactly where the object actually ends and free space begins. The overlay is useful when teleoperating a vehicle since it provides the supervisor a "general" idea of the obstacle's size.

The stereo camera pairs on the SensorBot and VisionBots may also be employed by the stereo processing. This process produces two dimensional polygons of objects it detects as well as their distance from the current agent position as shown in Figure 4.5. The figure

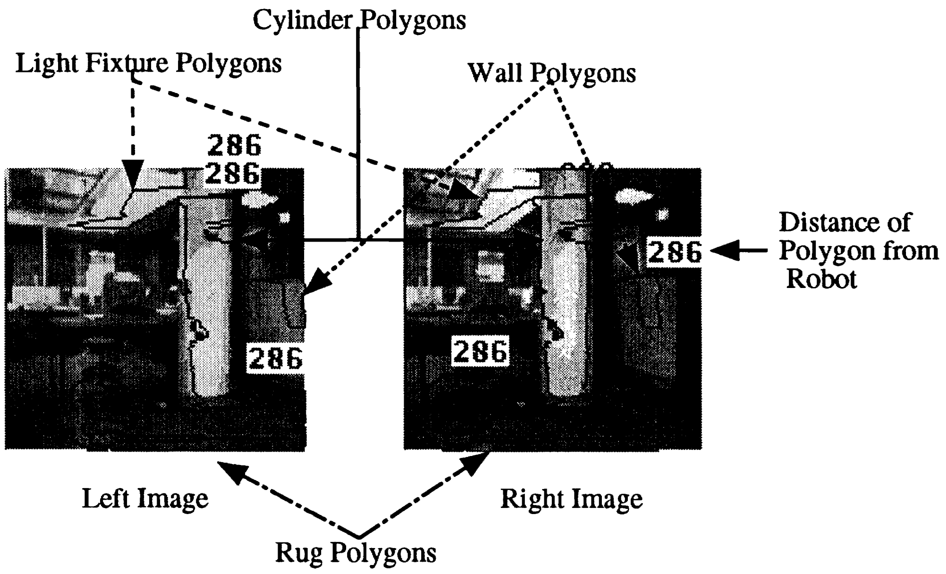


Figure 4.5: The images and polygons created by the stereo process.

displays both the stereo camera pair's left and right images. Objects which the process detects are represented as black outlined polygons. In this figure there are four objects the process detects: the column, a wall, an overhead light fixture and a portion of the rug. The corresponding numbers represent the object's distance from the robot in centimeters. We will develop a protocol to request and display this information. These data displays will include displaying the polygons in three-dimensions on the MASC model. We will also overlay the images from the stereo pair to verify the object's location and existence at the polygon locations. When the supervisor determines that a number of polygons constitute a single object, she or he will classify them as such. The overlaid images will assist the supervisor with this determination of which polygons belong to the same object. While the polygons are actually two dimensional when they are combined they will create a three dimensional object.

4.2.3 Specification Interaction

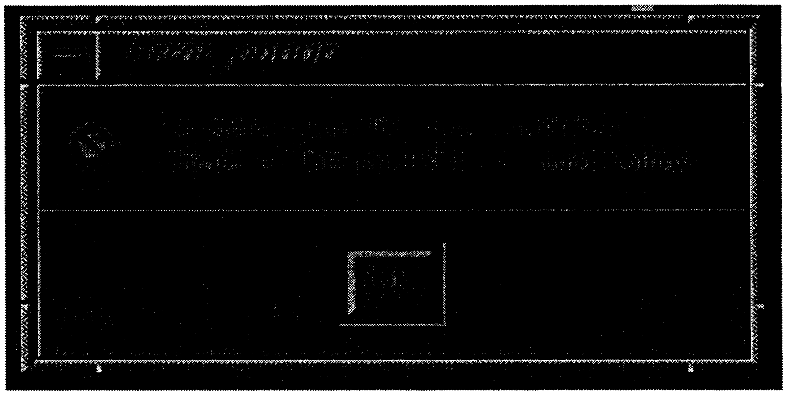


Figure 4.6: The error message generated when improperly adding way points.

As mentioned in Section 2.2.3 the MASC system includes two path planning mechanisms. Both planners require the human supervisor to prespecify data before a path can be planned. The path planning mode control buttons permit the supervisor to "add", "edit", or "remove" way point specifications as well as "disregard" or "display" the computed path. When adding way points, the supervisor points the mouse and clicks. A triangle representing the way point with zero heading appears. The supervisor may then rotate this point to specify the agent's heading angle. When adding additional way points the supervisor points the mouse at the way point the new one should follow in the path and then moves the mouse to the desired position of this point. If a previous way point is not selected, no way point will be added and an error message will appear as in Figure 4.6. When editing a way point the supervisor is permitted to only modify that point's heading. The point is chosen by clicking on it which then permits the supervisor to change the heading. When removing a point from the way points list, the supervisor clicks on the desired point. These options are available to the supervisor during path planning mode, so they are employed to determine the initial points as well as modify a computed path. At any time all way points and the corresponding path (if computed) may be disregarded. The supervisor may then start from scratch or abandon the path planning process. The agent is instructed to execute a path (only when one exist) upon exiting the path planning mode.

The local path planner requires the human to specify the way points and their respective headings as described above. This information is sent to the planner and the path is returned and displayed on the model, as in Figure 2.6(a). Before the agent is permitted to execute the path the human must verify it. At this point the path may be interactively modified and a new path returned or executed by the robot.

The global path plan server also requires the way point specifications which are conducted as described above. This planner server must also be sent information regarding the objects in the world, the types of robots used and their non-holonomic constraints, the desired planning algorithm such as potential fields or cell decomposition and the required variables for the planning algorithm such as requesting a safest or shortest paths and the path smoothness level. The path is then returned and displayed for confirmation and then may be executed.

4.3 Processing Level

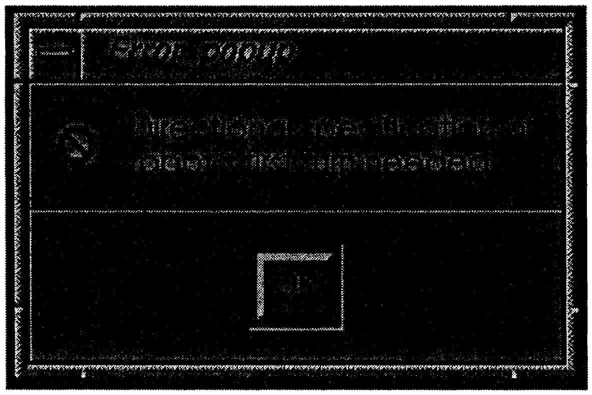


Figure 4.7: The error message generated for the local path planner singularity case.

The local path planning algorithm contains a singularity case. A normal path consist of three segments: a beginning arch, a straight line and an ending arch. The case appears when a way point is chosen directly behind the previous way point. In this instance, the planner is unable to determine which direction to turn for the first turn, either clockwise or counter-clockwise. When this occurs, the planner sends a message to MASC which is displayed as in Figure 4.7. The human must acknowledge this message and then specify the turn direction. Once the process receives this information it returns the desired path. In this instance the process request supervisory assistance.

The global path plan server algorithms may be unable to determine a path if a way point has been chosen too close to or inside an object. When this occurs, the planner sends a message to MASC. This message is displayed on the model and the human must then revise the selected way points.

The visually guided obstacle avoidance process sends the human information concerning the current process' state. The human can then instruct the system to stop its actions. The process is unable to request assistance from the human but the human can take control away from the process and directly control the agent. The supervisor may tell the agent to stop and do another task, teleoperate the agent around the obstacle, etc.

We are currently working to fully integrate the ultrasound process. As mentioned in Section 4.2.2. This process includes a localization function which attempts to localize the agent in conjunction with the "clusters" it detects. This function currently only monitors how far the actual agent is from the localized position. We are creating interactions with which localize the actual agent based upon this localization information.

This process is also unable to determine features such as walls and corners from the "clusters". We will provide the human supervisor with the ability to interactively instruct the process as to which "clusters" compose a feature. This feature determination function will be completed either when requested by the ultrasound process or when the supervisor deems it necessary. This information may be used by the localization process. Also, this process detects corners where there none exist (as displayed in Figure 4.2), the supervisor will instruct the agent as to a detected corner's existence. If we believe that the process has found a new feature or object we will then instruct the agent to employ the other sensing modalities to compile a more detailed object description.

We are also developing a protocol to instruct the agent when the process is unable to find previously known objects in the environment due to slippage. This protocol will also be utilized when the agent detects a new object which will then be added to the MASC model.

We will develop interactions with the stereo process to instruct or assist it when necessary. This will include instructing the process to stop processing and to rely upon another sensing modality. We will develop the ability to modify the algorithm as it runs so that we may change the size of the objects it attempts to match.

As described in Section 4.2.2, this process creates two dimensional polygons and a distance measure from the robot to the object location which will be drawn on the model in three-dimensions. It is likely that numerous polygons will actually belong to one object. We will interactively cluster polygons to create three-dimensional objects which will be added to the MASC model. For instance, a chair may be composed of a vertical polygon for the back of the chair, a horizontal polygon for the chair seat and some smaller polygons which correspond to the chair legs. The human supervisor will be able to combine these polygons into one three-dimensional object. Verification of the object's existence will be created by overlaying the image onto the model.

We will develop an interaction protocol with the basic light striping process to verify the footprints it detects. We will also develop interactions with the process to localize the other system agents based upon the SensorBot's localized position. As this process is not yet fully developed we are unable to determine what other types of interactions (low level) we will create with this process.

4.4 Data Level

As the agents are currently unable to automatically reset their own odometry and heading readings we are creating a protocol to permit the human supervisor to reset these readings. This reset information will be obtained from the localization information available from the ultrasound process and/or the light-striping footprint process. The ability to reset the readings will ensure that the information received from the agents will include less error.

We are working to formulate a method to determine if a sensor has entered a state in which it is sending incorrect readings. It is possible for the sensor boards to enter a state from which they send incorrect data or some object may be placed in front of a sensor. Once this is determined the ultrasound process will be instructed to ignore the sensor.

We will detect when the cameras of the stereo and visually guided obstacle avoidance processes are out of focus or have different apertures. Since this directly effects the correspondence in the stereo algorithm and the inverse perspective projection of the visually guided obstacle avoidance algorithm, we will instruct the agent it should no longer use the image processing mode. These two instances may occur by chance, someone changing the cameras, or it is possible they could run into some object and become unfocused as the cameras extend out beyond the vehicle front. In particular, if the stereo process suddenly is sending no data the supervisor may examine the images and then instruct the agent to rely upon another sensing modality.

Chapter 5

Experimental proof of Hypothesis

The motivation for the mediation hierarchy creation was to create a semi-autonomous multiple robot system which can always complete feasible tasks. Therefore, proof of the mediation hierarchy theory will entail executing various tasks until the agents require supervisory assistance then demonstrating the supervisor's ability to assist and correct the problem through the MASC system interface followed by the agent's ability to continue with the task execution to completion.

As we prefer to permit the agent's to work autonomously, we will have to measure:

1. the difficulty of the tasks involved for execution,
2. how often the agents request assistance,
3. how often the supervisor detects a problem which requires her or his taking control from the system,
4. how often the supervisor detects a problem which the system is able to correct without the supervisor's assistance,
5. the automation level which the system can routinely handle while executing tasks.

We will begin the experiments with relatively simple tasks such as moving the agents in a line ahead or box formation through an environment containing no obstacles. Presumably this type of task should require no human interaction other than the human specifying the task and the path the agents are to follow. We will then increase the task's difficulty, first by adding known obstacles to the environment, then unknown obstacles to the environment. We will move onto tasks which require the agents to move objects from one location to another. After this task, We will position the agents at unknown locations in the environment. They will be required to localize themselves and then execute various tasks. It is in this manner we will measure the difficulty of the tasks.

The measure of how often the agents request assistance will require the interface to count these requests. This number will be classified as to the task difficulty as it is expected that difficult tasks will encounter more request for assistance.

Another interesting measure entails how often the supervisor detects problem situations. This measure should then be classified into the number of times the agents request assistance before the operator takes control from the system, how often after a problem has been detected by the supervisor does she or he have to take control from the system and finally how often the system is able to correct itself.

Finally, we will measure the level of automation the system can routinely handle. This measure is important since we prefer the system to work autonomously thus reducing the supervisor's direct control of the system. Presumably, this measure will be derived from the above measures. While conducting the experiments it is suspected that we will encounter interactions which may require less control by the supervisor thus led to their further automation.

We will also investigate the possibility of automating some data display request. We may find that particular displays are required continuously and therefore should be automatically established upon entering the system.

Chapter 6

Summary

6.1 Future work for this dissertation

The future work of this thesis includes the completion of the process integrations. We will complete the ItPlanS planner integration on the *Task Level* as described in Section 4.1. While we have a high level of integration completed for the ultrasonic process, we will complete the integrations into the process described in Section 4.3. We will integrate the stereo process as described in Sections 4.2.2 and 4.3. We also hope to integrate the light striping process as described in Sections 4.2.2 and 4.3. All work described for the *Data Level* in Section 4.4 should be completed. Upon implementation completion we will experimentally test the hypothesis as described in Chapter Five.

6.2 Contributions

Many supervisory control systems built with human-machine interfaces do not permit the human supervisor to interact with all system levels. Most interfaces permit only high level interactions which pertain to sensory data and system state monitoring. They do not permit the supervisor to override or assist with low level decisions. For instance, when dealing with robots working in a hazardous waste environment, it is difficult and unsafe for a human to physically enter the environment to assist the robots. While teleoperation has been employed in these environments, there are numerous tasks which could be automated if the human-machine interface permitted the human supervisor to take control of the system, interact with the individual processes or if the system was able to interactively request assistance.

The major research contribution is the mediation hierarchy theory development. It should enable the multiagent system to successfully complete all feasible tasks. This theory combined with a working experimental system will demonstrate the ability of the hierarchy for our test bed. We feel the mediation theory can then be applied to other research areas including robotics, air traffic control, command and control systems, etc. This will improve the ability of both autonomous and teleoperated systems. By employing this theory new versions of autonomous systems may be developed which succeed when the strictly autonomous versions would fail. Also, new versions of teleoperated systems may be developed to include more autonomous functions which under current system definitions would be infeasible. This theory may also be applied to control systems at remote sites. For instance, there may exist a specialist for a specific robotic workcell. If the company has many of these workcells located throughout the country the specialist would normally have to travel to the workcell location to determine the problem and repair it. The mediation hierarchy would permit the specialist to remotely monitor the workcell in question and interactively determine the problem and correct it. This would reduce the workcell down time as well as reduce the cost of transporting the specialist. Another research contribution is the system's ability to request assistance from the human supervisor. In the scenario described above, this ability could keep the workcell from failing. The workcell could request assistance from the operator before the problem becomes too large.

The supervisor's ability to work within all system levels will elevate supervisory control to a higher level. As the supervisor is able to interact with all levels, the supervisor can obtain a better system view and therefore better perform the monitoring task. This permits the supervisor to obtain information concerning each process' state and to interact with these states. While all these interaction levels may not be used continuously, the fact they exist and that the supervisor may effect the process' outcome through the interactions is of higher importance. If a process is producing incorrect information it is likely the system will fail. In this situation the supervisor will be able to interact with the process to assist it or if absolutely necessary, take control away from the process and either teleoperate the process or give control to another process.

The supervisor's ability to request only the data required for the current task is another contribution. This ability reduces the chances that the human's sensory abilities will be exceeded since it is likely the human supervisor will not request unnecessary system information. It also implies the human supervisor will be able to request relevant data as opposed to miscellaneous data of little use for the current task diagnosis. This contribution also potentially reduces the communication load between the agents and the human-machine interface. When data is not required, the corresponding information channel can remain empty, thus reducing the number of channels the interface must monitor.

The supervisor's ability to display processed as well as raw data increases the system and the supervisor's capabilities. The supervisor can combine these two information forms to verify the processed data, verify the object's existence in the environment, localize the agents in the environment as well as to add newly detected objects to the world model. The supervisor may also employ the raw data to handle tasks for which there exist no processes. Such as the monitoring of "blind" agents through the viewing of images from a properly positioned observation agent. This extends the system's abilities by permitting the human supervisor to create a new "process".

6.3 Conclusions

We have presented the Multiple Agent Supervisory Control (MASC) system which has been developed in conjunction with the University of Pennsylvania General Robotics and Active Sensory Perception Laboratory's multiagent project. The goal is to create a semi-autonomous system which will successfully execute tasks. We have also defined the mediation hierarchy. This hierarchy is the basis for the MASC system development. The mediation hierarchy provides the supervisor with the ability to interact with all multiagent system processing levels. It permits the system's agents to work autonomously until they request supervisory assistance or the supervisor detects a problem in the system and takes control of the process in question.

We have also presented our test bed description and the available processes developed within the test bed as well as the general MASC human-machine interface. We described the processes integrations as well as the external processes' implementations such as the global task planner. Finally we presented a proposed hypothesis proof and the research contributions.

We feel that the mediation hierarchy theory will improve the supervisor's abilities and create a more robust system.

Bibliography

[Adams et al., 1995] Adams, J., Bajcsy, R., Kosecka, J., Kumar, V., Mandelbaum, R., Mintz, M., Paul, R., Wang, C., Yamamoto, Y., and Yun, X. (1995). Cooperative material handling by human and robotic agents: Module development and system synthesis. In *Submitted to: IEEE/RJS International Conference on Intelligent Robots and Systems.* IEEE/RJS.

[Adams and Paul, 1994a] Adams, J. A. and Paul, R. (1994a). Human-managed, hierarchical control of multiple mobile agents. In *Proceedings of the 33rd IEEE Conference on Decision and Control*, pages 3524 – 3529. IEEE.

[Adams and Paul, 1994b] Adams, J. A. and Paul, R. (1994b). Human management of a hierarchical control system for multiple mobile robots. In *Proceedings of the 1994 IEEE International Conference on Systems, Man and Cybernetics*, pages 2780 – 2785. IEEE.

[Ammons, 1985] Ammons, J. (1985). Scheduling models for aiding real time FMS control. In *Proceedings of the International Conference on Cybernetics and Society*, pages 185 – 189. IEEE.

[Askew and Diftler, 1993] Askew, R. and Diftler, M. (1993). Ground control testbed for space station freedom robot manipulation. In *Proceedings of the IEEE Annual Virtual Reality International Symposium*, pages 69 –75. IEEE.

[Aylett et al., 1991] Aylett, R., Fish, A., and Bartum, S. (1991). Task planning in an uncertain world. In *IEE Conference Publication*, pages 801 – 806. v.2 no. 332.

[Bach, 1991] Bach, C. (1991). A customizable direct manipulation user interface with automatic generation of help information. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Information Management*, pages 920 – 924. Elsevier, New York. v. 18B.

[Bajcsy et al., 1992] Bajcsy, R., Kumar, V., Mintz, M., Paul, R., and Yun, X. (1992). A small-team architecture for multiagent robotic systems. In *Workshop on Intelligent Robotic Systems: Design and Applications, SPIE's Intelligent Robotics Symposium*, Boston, MA.

[Barfield and Kim, 1991a] Barfield, W. and Kim, Y. (1991a). Computer graphics programming principals as factors in the design of perspective displays. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 93 – 97. Elsevier, New York. v. 18A.

[Barfield and Kim, 1991b] Barfield, W. and Kim, Y. (1991b). Evaluation of computer graphics techniques for the design of images for human-computer interaction. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 83 – 87. Elsevier, New York. v. 18A.

[Baron, 1984] Baron, S. (1984). A control theoretic approach to modeling human supervisory control of dynamic systems. In Rouse, W., editor, *Advances in Man-Machine Systems Research*, chapter 1, pages 1 – 47. JAI Press Inc., London, England. v. 1.

[Becker, 1994] Becker, C. (1994). *Internet Path Planner Server Protocol.* Stanford University.

[Bejczy et al., 1990] Bejczy, A., Kim, W., and Venema, S. (1990). The phantom robot: Predictive displays for teleoperation with time delay. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 546 – 551. IEEE.

[Bellingham and Consi, 1990] Bellingham, J. G. and Consi, T. R. (1990). State configured layered control. In *Proceedings of the International Advanced Robotics Program 1st Workshop on Mobile Robots for Subsea Environments*, pages 75 – 80.

[Bellingham and Humphrey, 1990] Bellingham, J. G. and Humphrey, D. (1990). Using layered control for supervisory control of underwater vehicles. In *Proceedings of the Conference on Remotely Operated Vehicles*, pages 175 – 181.

[Bevan et al., 1991] Bevan, N., Kirakowski, J., and Maissel, J. (1991). What is usability? In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 651 – 655. Elsevier, New York. v. 18A.

[Bodker, 1991] Bodker, S. (1991). *Through the Interface A Human Activity Approach to User Interface Design.* Lawrence Erlbaum Associates, Publishers, Hillsdale,N.J.

[Bogoni, 1994] Bogoni, L. (1994). Subsumption architecture and discrete event systems: A comparison. Technical Report MS-CIS-94-09, Grasp LAB 370, University of Pennsylvania.

[Borys, 1991] Borys, B.-B. (1991). Ways of supporting ergonomically and technically correct display design. In Weir, G. R. S. and Alty, J. L., editors, *Computers and People Series: Human-Computer Interaction and Complex Systems*, chapter 9, pages 211 – 222. Academic Press, New York.

[Brandt, 1993] Brandt, D. (1993). Research strategies for human-centered design of human-machine systems. In *Proceedings of the 12th World Congress, International Federation of Automatic Control*, pages 105 – 108. v. 8.

[Brooks, 1986] Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14 – 23.

[Brooks, 1987] Brooks, R. A. (1987). A hardware retargetable distributed layered architecture for mobile robot control. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 106 – 110.

[Chen and Trivedi, 1994] Chen, C. and Trivedi, M. (1994). Simulation and animation of sensor-driven robots. *IEEE Transactions of Robotics and Automation*, 10(5):684 – 704.

[Christensen, 1993] Christensen, B. (1993). Virtual environments for telerobotic shared control. In *Proceedings SPIE - The International Society for Optical Engineering*, pages 74 – 83. SPIE. v. 2057.

[Christensen et al., 1991] Christensen, B., Greibenow, B., and Burks, B. (1991). Graphic model based control of robotic systems for waste remediation. *Transactions of the American Nuclear Society*, (64):609.

[Cooke and Stansfield, 1994] Cooke, C. and Stansfield, S. (1994). Interactive graphical model building using telepresence and virtual reality. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1436 – 1440.

[Cox and Walker, 1993] Cox, K. and Walker, D. (1993). *User Interface Design.* Prentice Hall, New York, second edition.

[Currie and Tate, 1990] Currie, K. and Tate, A. (1990). O-PLAN: Control in the open planning architecture. In Allen, J., Hendler, J., and Tate, A., editors, *Readings in Planning*, chapter 5, pages 361 – 368. Morgan Kaufmann Publishers Inc., San Mateo, CA.

[Currie and Tate, 1991] Currie, K. and Tate, A. (1991). O-PLAN: the open planning architecture. *Artificial Intelligence*, 52:49 – 86.

[Dai, 1993] Dai, F. (1993). Centralized, application-oriented graphical interaction control using an example of planning robotic tasks. *Computers and Graphics*, 17(2):155 – 163.

[Drotning et al., 1992] Drotning, W., Christensen, B., and Thunborg, S. (1992). Graphical model based control of intelligent robot systems. *IEEE Control Systems Magazine*, 12(4):13 – 18.

[Dunias, 1993] Dunias, P. (1993). Robot task planning generalization. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 578 – 582.

[Durand, 1993] Durand, J.-P. (1993). Automated informational flows: Why human intervention is still necessary. In *Proceedings of the 12th World Congress, International Federation of Automatic Control*, pages 117 – 128. v. 8.

[Edmonds, 1992] Edmonds, E. (1992). The man-computer interface: A note on concepts and design (1982). In Edmonds, E. A., editor, *Computers and People Series: The Separable User Interface*, chapter 5, pages 185 – 194. Academic Press, New York.

[Eichelberg and Ackermann, 1993] Eichelberg, D. and Ackermann, P. (1993). Integrating interactive 3D-graphics into an object-oriented application framework. In *Proceedings of the Vienna Conference on Human Computer Interaction*, pages 3 – 12.

[Ellis, 1991] Ellis, S. R. (1991). Nature and origins of virtual environments: A bibliographical essay. *Computing Systems in Engineering*, 2(4):321 – 347.

[Ephrati and Rosenschein, 1994] Ephrati, E. and Rosenschein, J. (1994). Divide and conquer in multi-agent planning. In *Proceedings of the 12th National Conference on Artificial Intelligence*, pages 375 – 380.

[Fikes and Nilsson, 1971] Fikes, R. E. and Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189 – 208.

[Fleury et al., 1994] Fleury, S., Herrb, M., and Chatila, R. (1994). Design on a modular architecture for autonomous robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3508 – 3513.

[Foley and Van Dam, 1982] Foley, J. and Van Dam, A. (1982). *The Systems Programming Series: Fundamentals of Interactive Computer Graphics.* Addison-Wesley Publishing Co., London.

[Frey et al., 1993] Frey, P. R., Rouse, W., and Garris, R. D. (1993). Big graphics and little screens: Model-based design of large-scale information displays. In Rouse, W., editor, *Human/Technology Interaction in Complex Systems*, pages 1 – 58. JAI Press Inc., London.

[Geib, 1994] Geib, C. W. (1994). *Representing Actions for Planning.* PhD thesis, University of Pennsylvania, Philadelphia, PA.

[Georgeff, 1987] Georgeff, M. P. (1987). Planning. *Annual Review Computer Science*, 2:359 – 400.

[Gerstenfeld, 1988] Gerstenfeld, A. (1988). Integration of task level planning and diagnosis for an intelligent robot. In *Proceedings of the 4th Conference on Artificial Intelligence for Space Applications, NASA Conf. Pub. #3013*, pages 75 – 84.

[Gimnich et al., 1991] Gimnich, R., Kunkel, K., and Reichert, L. (1991). A usability engineering approach to the development of graphical user interfaces. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 673 – 677. Elsevier, New York. v. 18A.

[Grant and Mayes, 1991] Grant, S. and Mayes, T. (1991). Cognitive task analysis? In Weir, G. R. S. and Alty, J. L., editors, *Computers and People Series: Human-Computer Interaction and Complex Systems*, chapter 6, pages 147 – 168. Academic Press, New York.

[Gunsthövel and Bösser, 1991] Gunsthövel, D. and Bösser, T. (1991). Predictive metrics for usability. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 666 – 670. Elsevier, New York. v. 18A.

[Hahndel and Levi, 1994] Hahndel, S. and Levi, P. (1994). A distributed task planning method for autonomous agents in a FMS. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1285 – 1292.

[Hancke and Braune, 1993] Hancke, T. and Braune, R. (1993). Human-centered design of human-machine systems and examples from air transport. In *Proceedings of the 12th World Congress, International Federation of Automatic Control*, pages 343 – 346. v. 7.

[Hartley and Pipitone, 1991] Hartley, R. and Pipitone, F. (1991). Experiments with the subsumption architecture. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1652 – 1658.

[Hasegawa et al., 1990] Hasegawa, T., Suehiro, T., Ogasawara, T., Matsui, T., Kitagaki, K., and Takase, K. (1990). An integrated tele-robotics system with a geometric environment model and manipulation skills. In *Proceedings of the IEEE International Workshop on Intelligent Robots and Systems*, pages 335 – 341. IEEE.

[Heger et al., 1992] Heger, A. S., Duran, F., Frysinger, S., and Cox, R. (1992). Treatment of human-computer interface in a decision support system. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 837 – 841. IEEE.

[Heinecke, 1993] Heinecke, A. M. (1993). Software ergonomics for real-time systems. In *Proceedings of the Vienna Conference on Human Computer Interaction*, pages 379 – 390.

[Herot, 1984] Herot, C. F. (1984). Graphical user interfaces. In Vassiliou, Y., editor, *Human Factors and Interactive Computer Systems*, chapter 4, pages 83 – 104. Ablex Publishing Company, Norwood, N.J.

[Hirai and Sato, 1989] Hirai, S. and Sato, T. (1989). Multi-level manual and autonomous control superposition for intelligent telerobot. In *Proceedings of the NASA Conference on Space Telerobotics*, pages 131 – 140.

[Hirai et al., 1992] Hirai, S., Sato, T., and Matsui, T. (1992). Intelligent and cooperative control of telerobot tasks. *IEEE Control Systems*, 12(1):51 – 56.

[Hodges et al., 1993] Hodges, L. F., Bolter, J., Mynatt, E., Ribarsky, W., and van Teylingen, R. (1993). Lab review: Virtual environments research at the georgia tech gvu center. *Presence*, 2(3):234 – 243.

[Hollan et al., 1987] Hollan, J. D., Hutchins, E. L., McCandless, T. P., Rosenstein, M., and Weitzman, L. (1987). Graphic interfaces for simulation. In Rouse, W. B., editor, *Advances in Man-Machine Systems Research*, pages 129 – 163. JAI Press Inc., London. v. 3.

[Hutchins et al., 1986] Hutchins, E., Hollan, J., and Norman, D. (1986). Direct manipulation interfaces. In Norman, D. and Draper, S., editors, *User Centered System Design: New Perspectives in Human-Computer Interaction.* Erlbaum, Hillsdale, N.J.

[Hwang and Wang, 1991] Hwang, S.-L. and Wang, Y.-S. (1991). An experimental study of CRT graphical display in process control systems. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 200 – 204. Elsevier, New York. v. 18A.

[Jacob, 1986] Jacob, R. J. (1986). A specification language for direct-manipulation user interfaces. *ACM Transactions on Graphics*, 5(4):283 – 317.

[Jacob, 1989] Jacob, R. J. (1989). Direct manipulation in the intelligent interface. In Hancock, P. and Chignell, M., editors, *Intelligent Interfaces: Theory, Research and Design*, pages 165 – 212. Elsevier, New York.

[Johannsen, 1993] Johannsen, G. (1993). Knowledge based design of human-machine interfaces. In *Proceedings of the 12th World Congress International Federation of Automatic Control.* IFAC.

[Keil-Slawik et al., 1991] Keil-Slawik, R., Plaisant, C., and Shneiderman, B. (1991). Remote direct manipulation: a case study of a telemedicine workstation. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Information Management*, pages 1006 – 1011. Elsevier, New York. v. 18B.

[Kelley, 1968] Kelley, C. R. (1968). *Manual and Automatic Control - A Theory of Manual Control and its Applications to Manual and to Automatic Control.* John Wiley and Sons, Inc., New York.

[Kirlik et al., 1993] Kirlik, A., Miller, R. A., and Jagacinski, R. (1993). Supervisory control in a dynamic and uncertain environment: A process model of skilled human-environment interaction. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(4):929 – 952.

[Knoblock, 1992] Knoblock, C. A. (1992). An analysis of ABSTRIPS. In *Proceedings of the Conference on Artificial Intelligence Planning Systems*, pages 126 – 135.

[Kosecka, 1995] Kosecka, J. (In Progress, 1995). *Discrete Event Systems for Autonomous Mobile Agents.* PhD thesis, University of Pennsylvania.

[Kraupner-Stadler, 1991] Kraupner-Stadler, H.-C. (1991). Fundamentals of the use of colors in user interfaces. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 98 – 102. Elsevier, New York. v. 18A.

[Latombe, 1991] Latombe, J.-C. (1991). *Robot Motion Planning.* Kluwer Academic Publishers, Boston.

[Maher and Bell, 1992] Maher, P. K. C. and Bell, H. V. (1992). The man-machine interface - a new approach (1977). In Edmonds, E. A., editor, *Computers and People Series: The Separable User Interface*, chapter 3, pages 87 – 96. Academic Press, New York.

[Mandelbaum, 1995] Mandelbaum, R. (In Progress, 1995). *Sensor Fusion for Mobile Robot Localization, Exploration and Navigation.* PhD thesis, University of Pennsylvania.

[Martinengo et al., 1994] Martinengo, A., Campani, M., and Torre, V. (1994). Complex tasks and control strategies of robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 861 – 866.

[Mataric, 1992] Mataric, M. J. (1992). Integration of representation into goal-driven behavior-based robots. *IEEE Transactions on Robotics and Automation*, 8(3):304 – 312.

[Matsui and Tsukamoto, 1990] Matsui, T. and Tsukamoto, M. (1990). Integrated robot teleoperation method using multi-media display. In *Proceedings of the 5th International Symposium on Robotics Research*, pages 145 – 152.

[McAllester and Rosenblitt, 1991] McAllester, D. and Rosenblitt, D. (1991). Systematic nonlinear planning. In *Proceedings of the 9th National Conference on Artificial Intelligence*, pages 634 – 639.

[McKee and Wolfsberger, 1988] McKee, J. W. and Wolfsberger, J. (1988). A graphical, rule based robotic interface system. In *Proceedings of the Fourth Conference on Artificial Intelligence for Space Applications*, pages 85 – 91. NASA Conference Publication number 3013.

[Misue and Sugiyama, 1991] Misue, K. and Sugiyama, K. (1991). Multi-viewpoint perspective display methods: Formulation and application to compound graphs. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Information Management*, pages 834 – 838. Elsevier, New York. v. 18B.

[Morris and Rouse, 1993] Morris, N. M. and Rouse, W. B. (1993). Human operator response to error-likely situations in complex engineering systems. In Rouse, W. B., editor, *Human/Technology Interaction in Complex Systems*, pages 59 – 104. JAI Press Inc., London. v. 6.

[Nakamura, 1990] Nakamura, N. (1990). Human modifying knowledge acquisition for FMS scheduling. In Karwowsky, W. and Rahimi, M., editors, *Ergonomics of Hybrid Automated Systems II*, page 331. Elsevier, New York.

[Nakamura, 1991] Nakamura, N. (1991). A human-supervised control architecture for a flexible manufacturing system. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Information Management*, pages 1046 – 1050. Elsevier, New York. v. 18B.

[Ntuen et al., 1992] Ntuen, C. A., Park, E., Wang, Y.-M., and Byrd, W. P. (1992). The TOP architecture for multiagent task planning and scheduling. *Computers and Industrial Engineering*, 23(1-4):153 – 156.

[Pejtersen and Nielsen, 1991] Pejtersen, A. and Nielsen, F. (1991). Iconic interface for interactive fiction retrieval in libraries based on a cognitive task analysis. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 753 – 762. Elsevier, New York. v. 18A.

[Prothero, 1994] Prothero, J. (1994). A survey of interface goodness measures. Technical Report HITL R-94-1, Human Interface Technology Laboratory, University of Washington.

[Ramadge and Wonham, 1989] Ramadge, P. J. and Wonham, W. M. (1989). The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81 – 97.

[Rasmussen, 1984] Rasmussen, J. (1984). Strategies for state identification and diagnosis in supervisory control tasks, and design of computer-based support systems. In Rouse,

W. B., editor, *Advances in Man-Machine Systems Research*, chapter 3, pages 139 – 193. JAI Press Inc., London, England. v. 1.

[Regan and Pose, 1993] Regan, M. and Pose, R. (1993). An interactive graphics display architecture. In *Proceedings of IEEE Annual Virtual Reality International Symposium*, pages 293 – 299. IEEE.

[Rencken and Durrant-Whyte, 1993] Rencken, W. and Durrant-Whyte, H. (1993). A quantitative model for adaptive task allocation in human-computer interfaces. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(4):1072 – 1090.

[Rengger, 1991] Rengger, R. (1991). Indicators of usability based on performance. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 656 – 660. Elsevier, New York. v. 18A.

[Rocha and Ramos, 1994] Rocha, J. and Ramos, C. (1994). Task planning for flexible and agile manufacturing systems. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 105 – 112.

[Rondeau and ElMaraghy, 1990] Rondeau, J. and ElMaraghy, H. (1990). Robot programming and task planning. *Manufacturing Review*, 3(4):245 – 251.

[Rouse, 1981] Rouse, W. B. (1981). Human-computer interaction in the control of dynamic systems. *ACM Computing Surveys*, 13(1):71 – 99.

[Rouse and Hunt, 1984] Rouse, W. B. and Hunt, R. M. (1984). Human problem solving in fault diagnosis tasks. In Rouse, W. B., editor, *Advances in Man-Machine Systems Research*, chapter 4, pages 195 – 222. JAI Press Inc., London, England. v. 1.

[Sacerdoti, 1973] Sacerdoti, E. D. (1973). Planning in a hierarchy of abstraction spaces. In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 412 – 430.

[Sacerdoti, 1975] Sacerdoti, E. D. (1975). The nonlinear nature of plans. In *Proceedings of the fourth International Joint Conference on Artificial Intelligence*, pages 206 – 214.

[Sato and Hirai, 1987] Sato, T. and Hirai, S. (1987). MEISTER: A model enhanced intelligent and skillful teleoperational robot system. In Bolles, R. and Roth, B., editors, *Proceedings of the fourth International Symposium on Robotics Research*, pages 155 – 162.

[Sayers, 1993] Sayers, C. (1993). PENGUIN: PENN Graphical User Interface. University of Pennsylvania, Personal Communication.

[Sayers and Paul, 1994] Sayers, C. and Paul, R. (1994). An operator interface for teleprogramming employing synthetic fixtures. *Presence*, 3(4).

[Schneider, 1984] Schneider, M. L. (1984). Ergonomic considerations in the design of command languages. In Vassiliou, Y., editor, *Human Factors and Interactive Computer Systems*, chapter 7, pages 141 – 162. Ablex Publishing Corp., Norwood, N.J.

[Sheridan, 1986] Sheridan, T. (1986). Human supervisory control of robot systems. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 808 – 812.

[Sheridan, 1992] Sheridan, T. (1992). *Telerobotics, Automation, and Human Supervisory Control.* MIT Press, Cambridge, Mass.

[Shneiderman, 1983] Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16(8):57 – 69.

[Shneiderman, 1984] Shneiderman, B. (1984). The future of interactive systems and the emergence of direct manipulation. In Vassiliou, Y., editor, *Human Factors and Interactive Computer Systems*, chapter 1, pages 1 – 28. Ablex Publishing Corp., Norwood, N.J.

[Smith et al., 1992] Smith, S. J., Nau, D. S., and Throop, T. (1992). A hierarchical approach to strategic planning with non-cooperating agents under conditions of uncertainty. In *Proceedings of the Conference on Artificial Intelligence Planning Systems*, pages 299 – 300.

[Stansfield, 1994] Stansfield, S. (1994). Lab review: A distributed virtual reality simulation system for situational training. *Presence*, 3(4):360 – 366.

[Stein, 1994] Stein, M. (1994). *Behavior-Based Control for Time-Delayed Teleoperation.* PhD thesis, University of Pennsylvania, GRASP Laboratory.

[Stramler Jr., 1993] Stramler Jr., J. H. (1993). *The Dictionary for Human Factors/Ergonomics.* CRC Press, London.

[Suzuki et al., 1994] Suzuki, T., Yokota, K., Asama, H., Matsumoto, A., Ishida, Y., Kaetsu, H., and Endo, I. (1994). A human interface for interacting with and monitoring the multi agent robotic system. In *Proceedings of the 2nd International Symposium on Distributed Autonomous Robotic Systems*, pages 119 – 122. IEEE/RSJ/JSME/SICE.

[Taipale and Hirai, 1993] Taipale, T. and Hirai, S. (1993). A behavior-based control system applied over multi-robot system. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1941 – 1943.

[Tarn et al., 1994] Tarn, T.-J., Bejczy, A. K., Guo, C., and Xi, N. (1994). Intelligent planning and control for telerobotic operations. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 389 – 396.

[Tate, 1977] Tate, A. (1977). Generating project networks. In *Proceedings of the fifth International Joint Conference on Artificial Intelligence*, pages 888 – 893.

[Tate et al., 1990] Tate, A., Hendler, J., and Drummond, M. (1990). A review of AI planning techniques. In Allen, J., Hendler, J., and Tate, A., editors, *Readings in Planning*, chapter 1, pages 26 – 49. Morgan Kaufmann Publishers Inc., San Mateo, CA.

[Thomas and Goss, 1993] Thomas, P. and Goss, K. (1993). 3-dimensional visual representations for graphical user interfaces: The art of user-involvement. In *Proceedings of the Vienna Conference on Human Computer Interaction*, pages 429 – 430.

[Wang, 1993] Wang, C.-C. (1993). Local path planner. University of Pennsylvania. Personal Communication.

[Wang, 1995] Wang, C.-C. (In Progress, 1995). *The Performance of Repeatable Control Schemes for Redundant Manipulators.* PhD thesis, University of Pennsylvania, GRASP Laboratory.

[Wang and Kumar, 1995] Wang, C.-C. and Kumar, V. (1995). The performance of repeatable control schemes for redundant robots. *Submitted to: IEEE Transactions on Robotics and Automation.*

[Weir, 1991] Weir, G. R. S. (1991). Living with complex interactive systems. In Weir, G. R. S. and Alty, J. L., editors, *Computers and People Series: Human-Computer Interaction and Complex Systems*, chapter 1, pages 1 – 21. Academic Press, New York.

[Weld, 1994] Weld, D. S. (1994). An introduction to least commitment planning. *AI Magazine*, 15(4):27 – 61.

[Whalley, 1992] Whalley, S. (1992). The human computer interface - designing for process control. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 827 – 832. IEEE.

[Whiteside et al., 1988] Whiteside, J., Wixon, D., and Jones, S. (1988). User performance with command, menu, and iconic interfaces. In Hartson, H. R. and Hix, D., editors, *Advances in Human-Computer Interaction*, chapter 8, pages 287 – 315. Ablex Publishing Corporation, Norwood, N.J. v. 2.

[Widdel and Kaster, 1991] Widdel, H. and Kaster, J. (1991). Comparison of interaction techniques. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 640 – 645. Elsevier, New York. v. 18A.

[Wiethoff et al., 1991] Wiethoff, M., Arnold, A., and Houwing, E. (1991). The value of psychophysiological measures in human-computer interaction. In Bullinger, H.-J., editor, *Advances in Human Factors/Ergonomics: Human Aspects in Computing - Design and Use of Interactive Systems and Work with Terminals.*, pages 661 – 665. Elsevier, New York. v. 18A.

[Wilkins, 1984] Wilkins, D. E. (1984). Domain-independent planning: Representation and plan generation. *Artificial Intelligence*, 22:319 – 335.

[Wilkins, 1988] Wilkins, D. E. (1988). *Practical Planning.* Morgan Kaufmann Publishers Inc., San Mateo, CA.

[Woods, 1991] Woods, D. D. (1991). The cognitive engineering of problem representations. In Weir, G. R. S. and Alty, J. L., editors, *Computers and People Series: Human-Computer Interaction and Complex Systems*, chapter 7, pages 169 – 188. Academic Press, New York.

[Yamamoto, 1994] Yamamoto, Y. (1994). *Control and Coordination of Locomotion and Manipulation for a Wheeled Mobile Manipulator.* PhD thesis, University of Pennsylvania, GRASP Laboratory.

[Yamamoto and Yun, 1992] Yamamoto, Y. and Yun, X. (1992). Coordinating locomotion and manipulation of a mobile manipulator. In *Proceedings of the 31st Conference on Decision and Control*, pages 2643 – 2648. IEEE.

[Yamamoto and Yun, 1994] Yamamoto, Y. and Yun, X. (1994). Coordinating locomotion and manipulation of a mobile manipulator. *IEEE Transactions on Automatic Control*, 39(6):1326 – 1332.

[Yokota et al., 1994] Yokota, K., Suzuki, T., Asama, H., Matsumoto, A., and Endo, I. (1994). A human interface system for the multi-agent robotic system. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1039 – 1044.

[Zyda et al., 1993] Zyda, M., Pratt, D., Falby, J., Barham, P., and Kelleher, K. (1993). Lab review: NPSNET and the naval postgraduate school graphics and video laboratory. *Presence*, 2(3):244 – 258.