S34-63
60055
207657
P-10

AIAA-94-1209-CP
N94-30560

Object-Based Task-Level Control: A Hierarchical Control Architecture for Remote Operation of Space Robots

H.D. Stevens * E.S. Miles † S. J. Rock ‡ R. H. Cannon §
Stanford Aerospace Robotics Laboratory
Stanford, California 94305

Abstract

*† ‡§

Expanding man's presence in space requires capable, dexterous robots capable of being controlled from the Earth. Traditional "hand-in-glove" control paradigms require the human operator to directly control virtually every aspect of the robot's operation. While the human provides excellent judgment and perception, human interaction is limited by low bandwidth, delayed communications. These delays make "hand-in-glove" operation from Earth impractical.

In order to alleviate many of the problems inherent to remote operation, Stanford University's Aerospace Robotics Laboratory (ARL) has developed the Object-Based Task-Level Control architecture. Object-Based Task-Level Control (OBTLC) removes the burden of teleoperation from the human operator and enables execution of tasks not possible with current techniques. OBTLC is a hierarchical approach to control where the human operator is able to specify high-level, object-related tasks through an intuitive graphical user interface. Infrequent task-level commands replace constant joystick operations, eliminating communications bandwidth and time delay problems. The details of robot control and task execution are handled entirely by the robot and computer control system.

The ARL has implemented the OBTLC architecture on a set of Free-Flying Space Robots. The capability of the OBTLC architecture has been demonstrated by controlling the ARL Free-Flying Space Robots from NASA Ames Research Center.

1.0 Introduction

As NASA expands America's presence in space, on-orbit assembly, maintenance, and servicing must become routine operations. The extreme cost and risk of astronaut EVA dictate that automation and robotics must play a key role in providing such services in any viable long-duration human-in-space future. The enormous number of EVA hours currently required to perform these operations can be significantly reduced by the timely provision of effective human/robot teams. Such a team would consist of a human in a safe haven, such as on Earth or inside a space vehicle, indicating at a high level the tasks to be done, while robots in the space environment execute the tasks with quick proficiency.

To date, the operation of space robots requires the user to manually control the robot's actions *directly* by a "hand-in-glove" method (i.e. teleoperation). Robot performance is consequently characterized by the *fundamental* limitations of any system where human control is intricately involved -- namely,

* Ph.D. Candidate, Department of Aeronautics and Astronautics. Member AIAA. hdsteven@sun-valley.stanford.edu
† Ph.D. Candidate, Department of Aeronautics and Astronautics. esm@sun-valley.stanford.edu
‡ Associate Professor, Department of Aeronautics and Astronautics. Member AIAA. rock@sun-valley.stanford.edu
§ Charles Lee Powell Professor, Department of Aeronautics and Astronautics. cannon@sun-valley.stanford.edu

Copyright © 1993 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

time delay between the human and robot due to long distance communications, low bandwidth performance due to slow human response characteristics, and intense operator tedium and fatigue due to the complexity of teleoperating complex dynamic systems. Clearly, these limitations call into question the viability of teleoperated systems for the extended, sophisticated on-orbit operations for which they are intended.

Object-Based Task-Level Control (OBTLC), an architecture developed by Stanford University's Aerospace Robotics Laboratory (ARL), removes the burden of teleoperation from the human operator, enabling execution of tasks not possible with current teleoperation techniques. OBTLC is a hierarchical approach to control where the human operator is able to specify high-level, object-related tasks through an intuitive graphical user interface. Occasional task-level commands replace constant joystick operations, eliminating communications bandwidth and time delay problems. The details of robot control and task execution are handled entirely by the robot and computer control system.

2.0 THE OBTLC ARCHITECTURE

In order to fully comprehend the OBTLC architecture, it is first necessary to have a clear understanding of the terms "object" and "task", as they are used in this paper.

The notion of an object is fundamental to the OBTLC architecture. An object is any physical entity that the operator wishes to manipulate and/or to which a specific relationship with the environment or other objects is desired. An object might be something independent of the robot, such as an Orbital Replacement Unit (ORU), a space truss member, a tool or a bolt; or it might be a significant part of the robot, such as a manipulator end-effector or perhaps the entire robot.

A task is integrally related to this notion of an object. Specifically, a task is a manipulation of objects in the environment (including robots) to match a desired configuration of, or relationship between, objects. Examples of tasks include: "replace that ORU with this ORU", "join these two truss members together", "extract that bolt with this wrench", and commanding a free-flying space robot to "move from point A to point B." In all of the above task examples, one theme is constant: task specifications directly correspond to high-level desired **object** behavior, not low-level details of robot manipulation and control to achieve these tasks. This approach to control is therefore referred to as **object-based control**, and the tasks performed are **object-based tasks**.

The objective of the Object-Based Task-Level Control (OBTLC) architecture is to provide the human operator with the ability to specify directly, and in a simple way, the object-based tasks he or she wishes to execute. The details of how these tasks are carried out are handled autonomously by the robot, and therefore do not burden the operator. Thus, the human is free to concentrate on high-level issues, such as devising strategies and solving problems, while the robot's computers perform the fast calculations necessary to close control loops precisely and autonomously. This novel approach exploits the complementary capabilities of robotic control and human decision-making to construct a powerful human/robot team.

Implementation of the OBTLC architecture provides numerous advantages over lower-level remote teleoperation. First, the detrimental effects of time delay are minimized because the human operator is eliminated from the low-level control of the robot. Task level commands from the human and responses from the robot need only occur at infrequent intervals.

Second, operator fatigue is significantly reduced because the human is no longer burdened with the low-level details of teleoperating a sophisticated dynamic system. Finally, this complementary division of labor between human and robot enables the human/robot team to perform more complicated tasks than is possible with traditional teleoperation approaches.

THE HIERARCHICAL NATURE OF OBTLC

OBTLC involves the management of three different kinds of information:
1) Infrequent communication between human and robot(s) about tasks to be performed.
2) Strategic information used by a robot or shared between several robots to break complicated tasks into smaller sub tasks.
3) Low-level dynamic control information used to close high-speed control loops on each robot.

The OBTLC control architecture is correspondingly divided into three layers-- the User Interface, the Strategic Controller, and the low-level Dynamic Controller.

The **USER INTERFACE** maintains and displays a world model, and receives desired changes to the state of the world from the operator. By manipulating iconic images of objects in this world model, the operator simply and intuitively instructs the robot to perform complex tasks. For example, insertion of the icon of one part into another is all that is necessary to instruct the robot system to perform all actions necessary to complete the insertion task.

The second layer, the **STRATEGIC CONTROLLER,** is based upon a finite state machine structure and embodies the logic and decision-making capabilities necessary for the robot to operate autonomously. Examples include path-planning, advanced manipulation and assembly of objects, and multiple-robot coordination. The Strategic Controller monitors changes in the state of the world, new commands from the human operator, and low-level sensor information, and uses this information to devise and execute new plans and to dictate changes in low-level control behavior. It is also this layer that identifies and sends to the user interface indications of events or problems that may require closer operator attention.

The third layer, the **DYNAMIC CONTROLLER**, incorporates high-bandwidth, sensor-based feedback control to achieve precise, high-speed dynamic performance of the robot system. This layer renders all details of robot control (i.e. position and force regulation, coordination of dynamic coupling, use of redundancy, control optimization, disturbance rejection, etc.) completely transparent to the human operator.

3.0 RELATED WORK

There are several control Architectures designed for space operation. Lumia and Albus proposed the NASA/NBS Standard Reference Model for the Telerobot Control System Architecture (NASREM)[1]. NASREM is made up of three six-level hierarchies for task decomposition, world modeling, and sensory processing. In the NASREM system, the concept of an object at a high level is lost. The architecture is focused on controlling and coordinating manipulators. Strategic control, as defined in the previous section, is not incorporated into the NASREM architecture.

The Modular Telerobot Task Execution System (MOTES) [2], developed at JPL, is another type of hierarchical robot controller. The MOTES system is based on a command interpreter similar to that used in spacecraft. This approach differs from OBTLC in that it only generates plans that sequence pre-programmed,

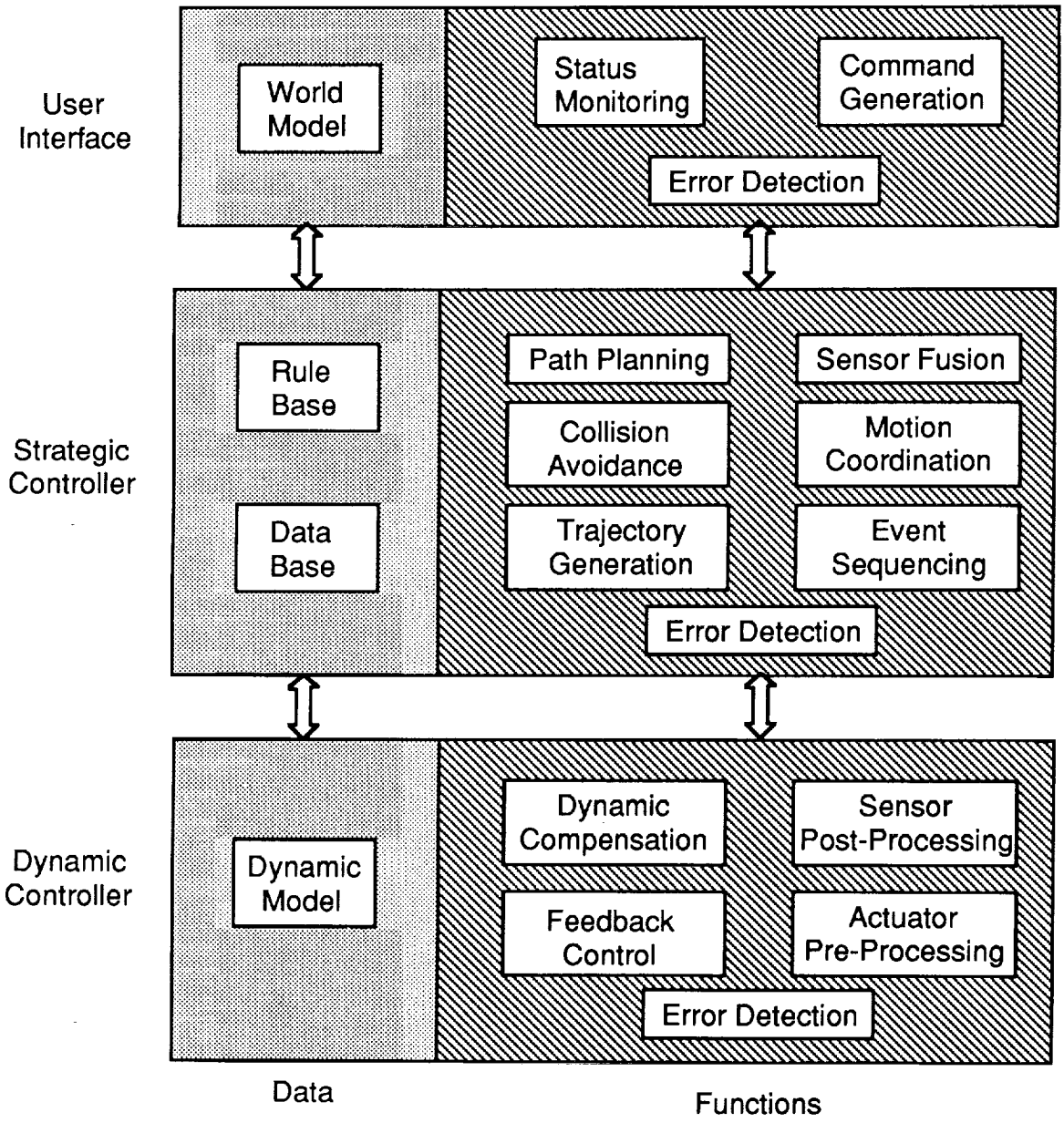


Figure 1: The Object-Based Task-Level Control Architecture. The architecture consists of a user interface, a strategic controller, and a dynamic controller. Occasional task-level commands from the user interface to the strategic controller create a system that is unaffected by communication delay.

open-looped macros and does not incorporate any sensor based decision making.

Another architecture which bears greater similarity to OBTLC is Sheridan's concept of supervisory control [3]. Indeed, at their most simplified level, both supervisory control and OBTLC involve human instructions to complex systems, which are than translated into actuator commands. In practice, however, most researchers interpret supervisory control to mean computer-augmentation of human teleoperation (i.e. incorporating control loops and

compensators in the system to make teleoperation more tractable). OBTLC differs from this interpretation in that the human input to the system is at a much higher level; in fact, human input is absent at the lowest level. OBTLC therefore represents an exploration of Sheridan's concept in a novel direction.

4.0 IMPLEMENTATION OF OBTLC ON A FREE-FLYING SPACE ROBOT

OBTLC has been implemented on several experimental systems at Stanford ARL, including several mobile and stationary robots with cooperating manipulators [4,5,7,8,10], and an underwater vehicle [6]. The application of OBTLC to a free-flying space robot prototype [7,8] is particularly interesting because of the complexity of the system.

ARL's space robotics facility features three autonomous self-contained free-flying space robots. A space environment is simulated in two dimensions using an air bearing over a flat granite surface plate. In this environment, the robots float on a cushion of air approximately 0.003 inches thick, and they propel themselves using on/off compressed air thrusters. The space robot is equipped with an on-board compressed gas supply, two two-link SCARA configuration manipulators, an on-board power supply, on-board computing, wireless ethernet communications, and local vision-sensing capability.

These space robots are capable of a variety of tasks including: capturing a translating, spinning object, adaptively identifying an objects mass and inertia properties, cooperatively maneuvering large objects, and assembling multiple objects. All of the space robots are based on the Object-Based Task-Level Control paradigm, although each implementation is slightly different. In this manner, the OBTLC architecture continues to evolve in response to new requirements.

EXAMPLE TASK: CAPTURE THAT OBJECT

To fully explore the concepts involved in OBTLC, one should examine, in detail, what is involved in carrying out a specific task. The task of capturing a translating, spinning object with a free-flying space robot is a particularly good example. An object, called Scooter, floats on the same granite table as the robot and is not within the initial workspace of the robot's manipulators. The operator wishes to capture Scooter, necessitating that the robot rendezvous with and grasp Scooter. Figure 2 shows the robot and object.

A global sensing system provides position and orientation information for the objects on the table (i.e. the robot and Scooter) in real-time. This information is used by both the user interface and the strategic controller to update the world model.

USER INTERFACE

One implementation of the user interface uses the Virtual Environment Vehicle Interface (VEVI) developed by the Intelligent Mechanisms Group at NASA Ames Research Center. The VEVI is an interactive virtual reality user interface which utilizes real-time interactive 3D graphics and position/orientation sensing to produce a range of interface modalities from flat-panel (windowed or stereoscopic) screen displays to head mounted/head-tracking stereo displays [9].

The VEVI displays the virtual reality model of the world (robot, Scooter, and table) with the position and orientation of the objects updated at about 1 hz from the global sensing system. The operator simply manipulates the objects by controlling a virtual hand icon with a 3D mouse. To command a capture, the operator places the hand in select mode (using a button on the mouse) and

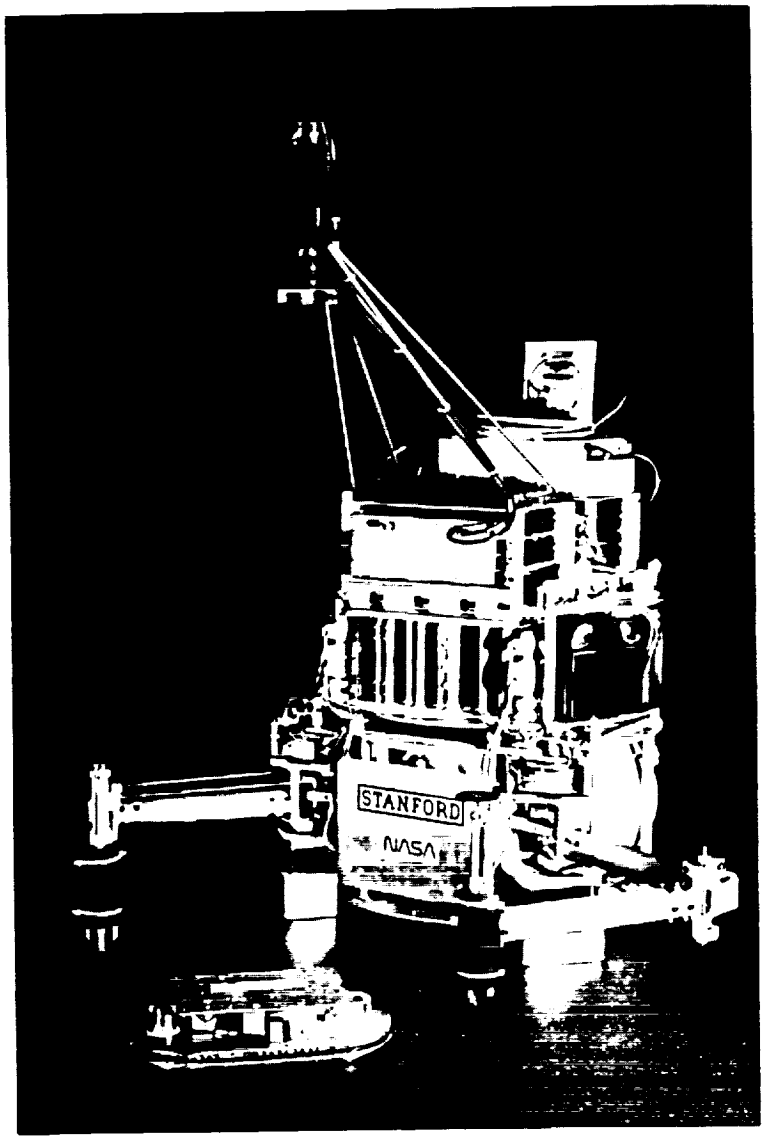
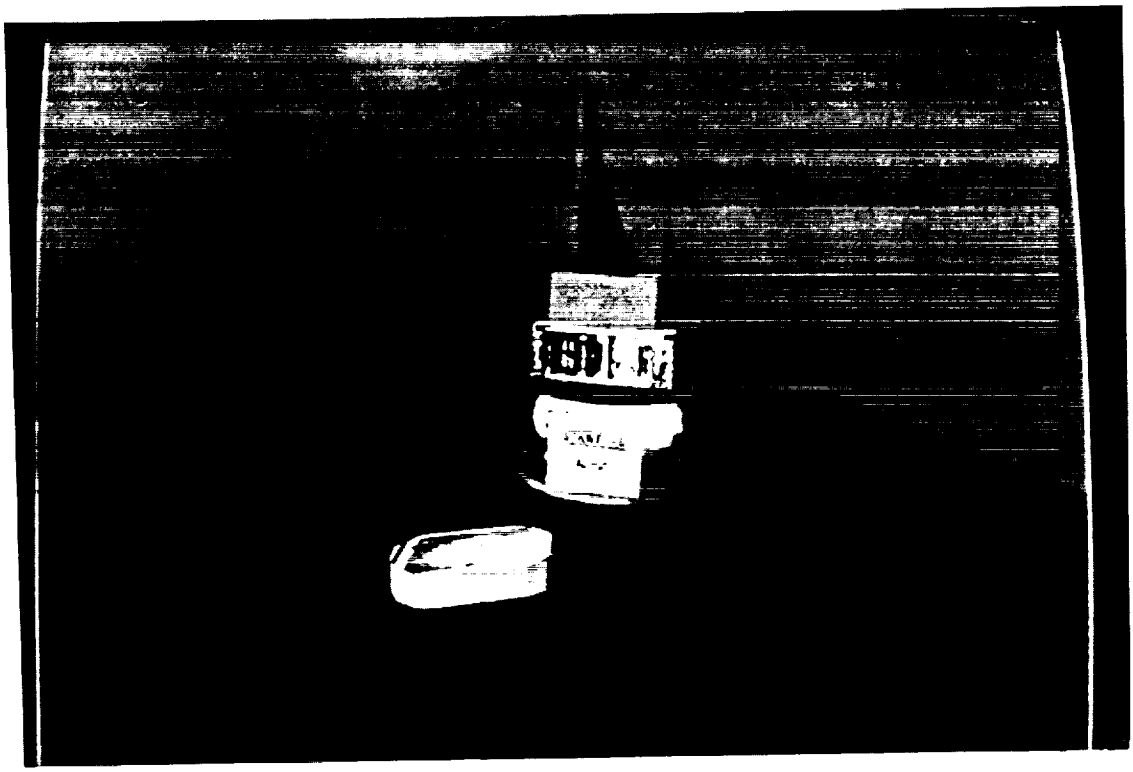


Figure 2: A Free-Flying Space Robot and an object. The space robot uses the Object-Based Task-Level Control architecture. ARL's Free-Flying Space Robots are capable of a variety of tasks including: capturing a translating, spinning object, adaptively identifying an objects mass and inertia, cooperatively maneuvering large objects, and assembling multiple objects.

Figure 3: The Virtual Environment Vehicle Interface. The VEVI, developed at NASA Ames Research Center, provides a simple, intuitive operator interface. By manipulating iconic images of the objects, the operator simply and intuitively instructs the robot to perform complex tasks.



Operation	Control Mode	Trajectory	Error Law
System Initialization	Joint	Fifth Order	PD
Rendezvousing with Object	Endpoint (Base Relative)	Set Point	PD
Intercepting Object	Endpoint (Inertial)	Fifth Order	PD
Tracking Object	Endpoint (Inertial)	Tracking	PID
Stopping Object	Object-Based (Base Relative)	Fifth Order	PD
Holding Object	Object-Based (Base Relative)	Set Point	PD
Placing Object	Object-Based (Inertial)	Fifth Order	PD

Table 1: Control Modes Required for Object Capture.
This table lists the set of controller configurations that the strategic controller takes the system through in the process of rendezvousing with and capturing a free-flying object. In all of these configurations, the base motion is controlled in the inertial reference frame using bang-off-bang trajectories and PD error laws.

touches the object. The VEVI then transmits the capture command, which requires the object name, Scooter in this case, as the only parameter. Figure 3 shows an operator's view of the VEVI.

It is this high-level of interaction that enables low-bandwidth communication and eliminates the effect of time delay. The operator is now free to plan the next task, contemplate the strategy, or just watch the task execution.

ON THE ROBOT: STRATEGIC CONTROLLER and DYNAMIC CONTROLLER

Upon receipt of the capture command the strategic controller begins a multi-step process of intelligently carrying out the capture task. The strategic controller is implemented using a finite state machine. As new events or stimuli occur, the finite-state machine reacts, depending on the current state of the system, by either progressing to the next phase of a multi-step procedure or by initiating a new course of action.

A major portion of the strategic controller's coordination involves the switching of control modes in the dynamic controller. There are seven different control modes required to complete the capture task. These seven are listed in Table 1. All of these are implemented in the dynamic controller. A complete discussion of these low-level control modes can be found in [8]. One control mode of interest is the object-based control mode. This control mode is based on the theory of Object Impedance Control [4,10]. This control methodology carries the concept of object down to the lowest levels of the control architecture.

The capture command sets in motion the finite state machine (FSM) to capture the object. The topology of the FSM is depicted in Figure 4. In the figure an ellipse signifies a state in the finite state machine, a rectangle signifies a state transition procedure, and a phrase over a line indicates the stimulus that causes the transition from one state to another. State transition procedures are similar to subroutines that return a stimulus. Thus each procedure completes some actions and returns the appropriate stimulus.

To complete the capture, the strategic controller determines if the object requested is either in view (i.e. within the range of the local sensing system) or found (i.e. on the table, but not within view of the local sensing system). In the example, the object is found. The object trajectory and robot base intercept trajectory are computed and the proper dynamic controllers switched in. The dynamic controller is provided with the proper intercept trajectory to follow. At

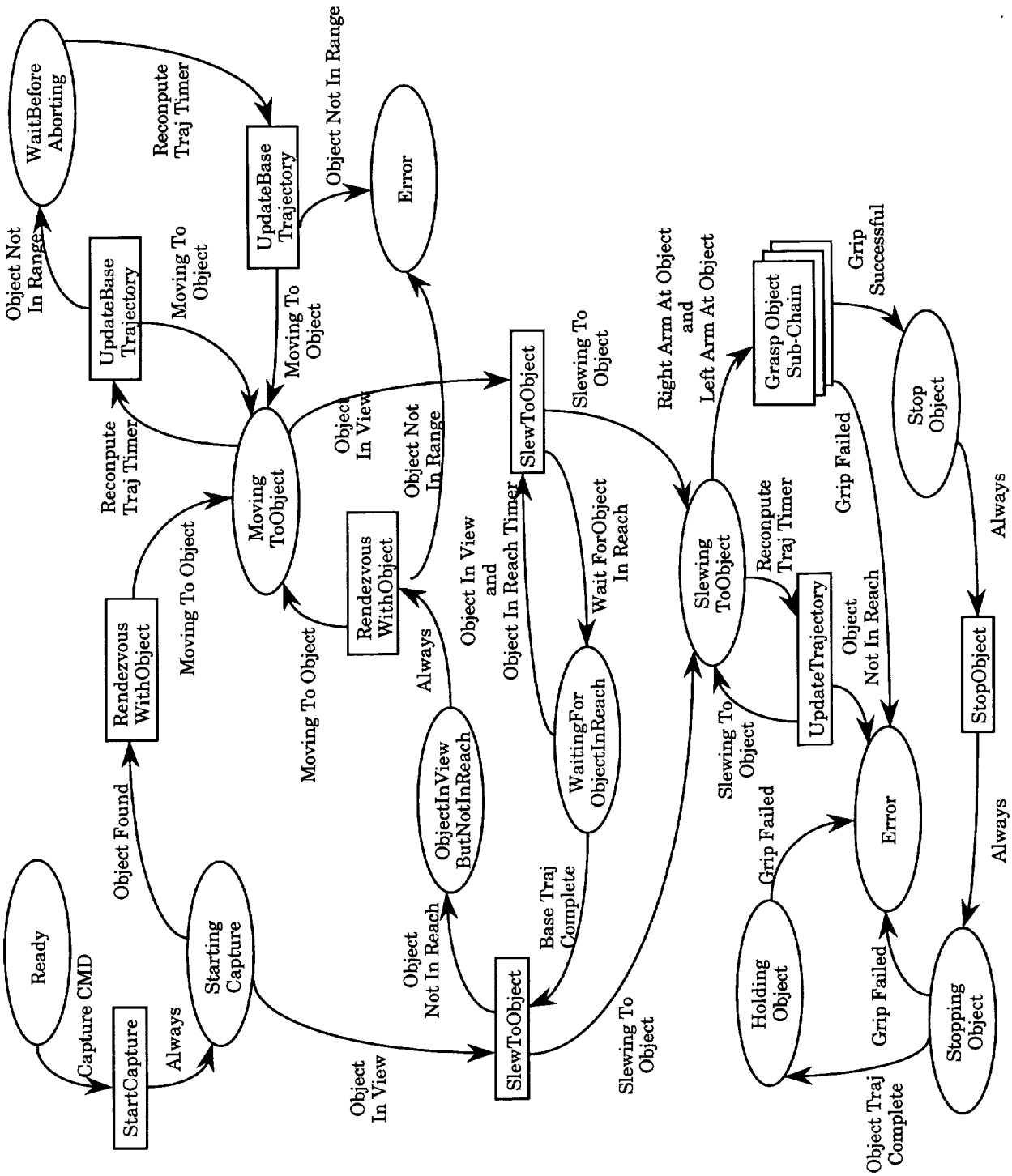


Figure 4: Object Rendezvous Finite-State Machine Graph.
This is the portion of the Strategic Controller which is executed when a Capture command is issued by the operator. Using the Finite-State Machine, the Strategic Controller is able to react, intelligently, to new sensor information. This sensor based decision ability is the unique feature of OBTLC.

regular intervals, the intercept trajectory is recalculated to allow for new information to enter the system. The base motion and trajectory recalculation continue until the object comes into view of the local sensing system.

With the object in view of the local sensing system, the robots manipulators are commanded to begin slewing to the object. Trajectories for each of the two manipulators are computed, checked for collisions, and executed as the object comes within the workspace of the manipulators. The trajectories place the endpoints over the grip points for grasping. The object is grasped, and the motion of the object stopped using the manipulators. Scooter has been successfully captured.

The entire sequence of events described above is initiated with a simple "capture that object" command issued by the operator. The operator has been completely removed from the details of robot motion and control modes required to complete this capture. It is apparent that the details of this operation, and the speed at which they must be accomplished, are daunting for the human operator alone. It is quite possible that a human operator with no help could not even accomplish this task.

5.0 CONCLUSIONS

The OBTLC architecture is a powerful new paradigm in the remote control of robot systems where the operator interacts with the system via an intuitive interface. The system is commanded at the task level, allowing the human operator to focus on the strategic issues, such as what to do next, while the robot system carries out the desired tasks quickly and deftly. This paradigm raises the human/robot team to a level never before possible.

Development of the OBTLC architecture has been guided by the principles of systems engineering and the desire to enable humans to interact with a robotic system at an intuitive level. This architecture has evolved to the current point only by the strict adherence to these principles. As with any architecture, OBTLC continues to evolve enabling its application to a broad range of problems.

6.0 ACKNOWLEDGMENTS

This research is supported by NASA Grant NCC 2-333. The authors are grateful to NASA for their continued support of this research.

7.0 REFERENCES

[1] Ronald Lumia and James S. Albus. Teleoperation and Autonomy for Space Robotics. *Robotics,* 4(1):27-33, 1988.

[2] Paul G. Backes, Mark K. Long, and Robert D. Steele. The Modular Telerobot Task Execution System for Space Telerobotics. In *Proceedings of the IEEE International Conference of Robotics and Automation*, pages 524-530, Atlanta, GA, May 1993. IEEE Robotics and Automation Society.

[3] Thomas B. Sheridan. Telerobotics, Automation, and Human Supervisory Control. Cambridge, Massachusetts: The MIT Press, 1992.

[4] Stanley A. Schneider. *Experiments in the Dynamic and Strategic Control of Cooperating Manipulators*. PhD thesis, Stanford University, Department of Electrical Engineering, Stanford, CA 94305, September 1989. Also Published as SUDAAR 586.

[5] Lawrence E. Pfeffer. *The Design and Control of a Two-Armed, Cooperating, Flexible-Drivetrain Robot System.* PhD thesis,

Stanford University, Department of Aeronautics and Astronautics, Stanford, CA 94305, December 1993. To Be published.

[6] Howard H. Wang, et all. Task-Based Control Architecture for an Untethered, Unmanned Submersible. In Proceedings of the 8th Annual Symposium of Unmanned Untethered Submersible Technology, September 1993.

[7] Marc A. Ullman. *Experiments in Autonomous Navigation and Control of Multi-Manipulator, Free-Flying Space Robots*. PhD thesis, Stanford University, Department of Aeronautics and Astronautics, Stanford, CA 94305, March 1993. Also published as SUDAAR 630.

[8] William C. Dickson. *Experiments in Cooperative Manipulation of Objects by Free-Flying Robot Teams*. PhD thesis, Stanford University, Department of Aeronautics and Astronautics, Stanford, CA 94305, December 1993. To be published.

[9] T. W. Fong. A Computational Architecture for Semi-autonomous Robotic Vehicles. In *Proceedings of the AIAA Computing in Aerospace 9 Conference*, San Diego, CA, October 1993. AIAA.

[10] Stanley A. Schneider and Robert H. Cannon, Jr. Object impedance control for cooperative manipulation: Theory and experimental results. *IEEE Journal of Robotics and Automation*, 8(3). June 1992.