Coordinated Multi-Agent Teams and Sliding Autonomy for Large-Scale Assembly

Brennan Sellner[†], Frederik W. Heger[†], Laura M. Hiatt[‡], Reid Simmons[†‡], Sanjiv Singh[†]

[†]*The Robotics Institute,* [‡]*Computer Science Department*

School of Computer Science

Carnegie Mellon University

5000 Forbes Avenue, Pittsburgh, PA 15213

Abstract

Recent research in human-robot interaction has investigated the idea of Sliding, or Adjustable, Autonomy, a mode of operation bridging the gap between complete robot autonomy and full teleoperation. This work, by and large, has been in single-agent domains – involving only one human and one robot – and has not examined the issues that arise when moving to multi-agent domains. Here, we discuss the issues involved when adapting Sliding Autonomy concepts to coordinated multi-agent teams. In our system, remote human operators have the ability to join, or leave, the team at will, to assist the autonomous agents with their tasks while not disrupting the team's coordination. We employ user modeling in order to allow agents to request help when appropriate, regardless of whether human operators are actively monitoring their progress. To validate our approach, we present the results of two experiments. The first evaluates the human–multi-robot team's performance under four different collaboration strategies including complete teleoperation, pure autonomy, and two distinct versions of Sliding Autonomy. The second experiment compares a variety of user interface configurations, to investigate how quickly a human operator can attain situational awareness when asked to help. The results of these studies support our belief that by incorporating a remote human operator into multi-agent teams, the team as a whole becomes more robust and efficient.

I. Introduction

As expectations for robotic systems increase, it becomes harder and harder to meet them with the capabilities of a single robot. One approach is to use multiple simple robots to perform tasks that would

This work has been supported by NASA under grant number NNA04CK90A.

require a very complex single mechanism. These teams not only bring a much broader spectrum of potential capabilities to a task, but also may be more robust in the face of errors and uncertainty.

While it is envisioned that robot teams eventually will be able to perform complex tasks such as large-scale assembly autonomously in remote hostile environments, the current state of the art falls short of the necessary capabilities. In particular, the number of contingencies that must be considered and provided for to make robots fully autonomous in open and dynamic environments is prohibitively large. On the other hand, pure teleoperated control of such robots is unlikely to be very efficient because of the communication delays involved, the large number of human operators required, and the sensing and visualization problems inherent in any teleoperation domain. Our goal is to develop a framework within which a single human operator can oversee and intervene in the operation of a team of largely autonomous robots.

One scenario exemplifying this approach is the assembly of large structures in hazardous environments, such as orbital solar power arrays or Mars habitats. In such environments, in-place human labor is either unfeasible or scarce and expensive, making less fragile robots an attractive option. We have been examining how robots and ground-based humans could work together while assembling large scale orbital structures, such as kilometers-wide solar power arrays. We envision many teams of robots working independently on different portions of the structure, overseen by a small number of human operators on Earth.

In order to develop the architectures, software capabilities, and models needed for such an endeavor, we have assembled a heterogeneous team of Earth-bound robots. Our experimental task is the construction of a square from four beams and four compliant nodes (Figure 1). Before each end of a beam can be inserted into a node, that node must be braced against the insertion forces. After each side of the square is completed, the next beam must be procured and brought to the worksite. Heterogeneous teams are well-suited to such construction scenarios, where many different skills, and correspondingly different hardware, are required. Our team consists of an imprecise heavy-lift robot, a weaker but more precise mobile manipulator, and a dedicated sensing robot (Figure 2).

Human operators bring their own unique set of skills to the team. To make the most efficient use of the flexible problem-solving skills of humans, a robot team should operate mostly autonomously, getting help from a human operator only when a problem arises that it cannot resolve by itself or when human control provides significant benefits. Sliding, or Adjustable, Autonomy is an approach that has been developed to address this, yielding more robust and efficient systems. Most work in Sliding Autonomy, however, has been limited to control of single robots [FIXME:REFS; MAYBE KORTENKAMP? ANY

MULTI-AGENT SA SYSTEMS OUT THERE?]. In this work, we extend the concept to heterogeneous multi-robot teams.

One fundamental difference in applying Sliding Autonomy to multi-agent teams is that the human will not always be aware of everything that is happening to all of the agents, whereas a human who must monitor only a single agent is able to keep abreast of all relevant developments. This gives rise to three major issues that need to be addressed for multi-agent Sliding Autonomy:

1) **Requesting help:** Since the human cannot keep track of all robots at all times, the ability of an agent to ask the human for help is critical. By using performance models of the autonomous system and the human operator that incorporate empirical performance data for the current operator, knowledge of typical human learning curves, and information about the team's state, the system can make reasonably accurate performance predictions, which in turn allows it to make principled decisions about when to ask the human for help.
2) **Gaining situational awareness:** Since the human will not be able to monitor all robots at once, or may be called away to attend to another team, it is important to ensure that the operator can quickly gain situational awareness of a robot's workspace when help is requested. It is also important that the autonomous system model how long this will take, in order to make principled decisions about when to ask for help.
3) **Maintaining coordination:** During and after human intervention, the components of the system that remain under autonomous control must continue to operate and maintain the inter-agent coordination necessary for task completion. We address this issue by enabling agents to monitor themselves and other agents as appropriate for their current task, allowing them to maintain coordination with another agent even when the operator is in control of either agent.

To validate our approach to Sliding Autonomy in the multi-agent domain, we have conducted an experiment that varied the degree of human involvement in the task. The experiment consisted of performing our construction task under four different human/robot cooperation strategies: pure autonomy, System-Initiative Sliding Autonomy (SISA), Mixed-Initiative Sliding Autonomy (MISA), and teleoperation. Pure autonomy clearly does not involve the human, consisting solely of autonomous behaviors and recovery actions. During teleoperation, the human is in complete control of all aspects of all the robots. Bridging the gap between these two extremes, SISA allows the operator to intervene only when asked to do so by the autonomous system, while in MISA the human can also intervene at any time of his own volition. SISA is designed to approximate situations where the operator is a scarce resource, and must multi-task

in attending to different robot teams. MISA, on the other hand, captures situations where humans can be more dedicated to observing the team's activities.

In this experiment, we examined time to complete the task, robustness, and the human's perceived workload. The results indicate that the autonomous system is consistently faster, but less reliable, than a purely teleoperated approach. In both SISA and MISA, the speed of assembly approaches that of the autonomous system. While operator workload results showed that the preference for either SISA or MISA is very task- and user-dependent, the workload for the human is clearly less than during pure teleoperation. We conclude that our adaptation of Sliding Autonomy improves the multi-agent team's reliability without compromising efficiency, and can be easily reformulated to meet the differing constraints of a variety of domains.

We also conducted a study examining how human operators can best attain situational awareness in our scenario. The experiment examined the types and amount of information that should be maintained in order to minimize time to achieve situational awareness, given that the operator was attending a different task prior to the request for help. The results show that the interface designer can make clear trade offs between time to achieve situational awareness and the quality of the resulting understanding. In general, accuracy (i.e. quality of understanding) increases as more data is available to the subject. However, the time needed to attain situational awareness is not a monotonic function of the amount, nor type, of information available. Instead, there is a clear point at which the time taken to absorb additional information outweighs the corresponding decrease in response time. The results from these two experiments bolster our contention that Sliding Autonomy can be an effective approach to robust control of multi-robot teams.

II. Related Work

A. Multi-Agent Robotic Assembly

The most common multi-agent assembly systems are those in factory settings, where multiple industrial robots are involved in the assembly of a product. To allow for maximum flexibility in production, such systems are designed to be easily reconfigurable; this allows them to adapt to different variants of the same product, or even entirely different products. A system of four industrial robots arranged around a conveyor network for material handling is described in [1]. The setup is typical of industrial applications, with stationary robots and interactions limited to scheduling of shared resources (such as the conveyor system or temporary storage areas). Since time is of significant concern in factory settings, such industrial applications typically are managed by a central controller. In contrast, our system is

comprised of multiple mobile and stationary robots. They must flexibly coordinate their motions in order to complete the assembly task, and adapt to a dynamic, uncertain environment; this requires close coordination between various combinations of heterogeneous robots, often involving more than one robot simultaneously manipulating the structure.

Coordinated assembly performed by teams of mobile robots is of prime interest to the space community. Stroupe et al. [2] use the CAMPOUT architecture to coordinate robots with purely behavior-based strategies to perform very tightly coupled tasks, similar to ours. Two heterogeneous robots collaboratively carry a beam and position it with respect to an existing structure with sub-centimeter accuracy.

B. Human-Robot Interaction

Recently, there has been significant interest in allowing human collaboration with robots in assembly scenarios. The COBOT project seeks to make manually operated machines more intelligent by providing guidance so that the operator does not have to provide fine guidance control. Typically, the human provides the force input, while the system steers the mechanism into the right place [3] [4]. The roles of the human operator and the system are clear and unvarying and both human and the system must operate simultaneously.

NASA's ASRO project [5] developed a mobile robot to assist a space-suited human by carrying tools, helping to manipulate objects, and providing sensor information. While the robot was physically working alongside the astronaut, it was teleoperated by another person in communication with the astronaut from a remote site. Unlike the complete teleoperation used in ASRO, our system allows the remote user to take control of parts of the assembly task, while leaving the remainder active under robotic control. The human and robots in our scenario cannot directly interact physically since, unlike [3], [4], and [5], they are not collocated.

A system closely related to our approach to human-robot interaction is presented by Fong et al. [6], in which the robot and the user participate in a dialogue. The robot can ask the operator to help with localization or to clarify sensor readings. The operator also can make queries of the robot. This framework assumes that the robot is capable of performing all tasks as long as it has full state information, whereas our approach allows the human to assume control if necessary.

C. Sliding, or Adjustable, Autonomy

Our use of the term Sliding Autonomy corresponds with the term Adjustable Autonomy as presented by Dorais et al. [7]. That work provides several future examples in which Sliding Autonomy will be

essential for space operations where demands on the operator must be focused and minimized, such as [FIXME: INSERT EXAMPLES FROM DORAIS PAPER].

Using a roving eye and a (fixed) manipulator similar to ours, Kortenkamp et al. [8] developed and tested a software infrastructure that allows for sliding control of a robot manipulator. The task involved a pick-and-place operation during which Sliding Autonomy allowed the operator to recover from visual servoing errors, participate in high-level planning, and teleoperate the manipulator to complete tasks beyond its autonomous capabilities. Our work extends this with a more complex assembly task that involves a team of robots and a finer granularity of Sliding Autonomy.

Scerri has proposed an architecture for Sliding Autonomy applied to a daily scheduler [9]. This autonomous system attempts to resolve timing conflicts (missed meetings, group discussions, personal conflicts, etc.) among some set of team members. Members are able to adjust the autonomy of the system by indicating their intent to attend gatherings or willingness to perform tasks.

Maheswaran et al. [10] describe a system of personal assistant agents that can operate under either user-based or agent-based autonomy. The entity in control (either an agent or a human) explicitly reasons whether and when to transfer decision making control to another entity (another agent or human). While our system currently does not allow control to be transferred from one robot to another, our approach to Sliding Autonomy operates at a much finer level of granularity.

D. Situational Awareness

Over the years there has been a large body of research on helping human operators maintain situational awareness. A significant amount of the initial research in this area focused on helping pilots maintain situational awareness while flying [11] [12]. The focus has shifted in more recent research to studying how human operators can maintain situational awareness while teleoperating robots, such as those in the search and rescue domain [11]. This work is most relevant to systems where the operator is in constant contact with the system. In our domain, however, the operator may have periods where he is not in contact with the system; thus, we are also interested in helping the operator repeatedly attain situational awareness after being out of contact with the system.

Goodrich et. al. [13] also study situational awareness in situations where the operator may not be in constant contact with the system. That work examines the effectiveness of an operator when controlling a robot at different levels of autonomy given increasing inattention to the robot. It attempts to facilitate such awareness by designing a usable interface for the operator. This is similar to our system, where the operator sometimes multi-tasks between interaction episodes, effectively ignoring the robots for that

period of time.

III. Context: Scenario, Hardware, and Architecture

A. Scenario

One significant area of robotics research is large-scale robotic assembly. Robots are especially useful in areas that humans are not well adapted to, particularly hazardous environments, such as space, the Moon, or Mars. Building one robot that can handle the entire construction task on its own is often either very difficult or impossible, especially for larger-scale assemblies. Our approach is to use multiple, heterogeneous robots that coordinate with one another to complete the task. While this increases complexity due to the necessary coordination [14], it also allows for more flexibility during task execution.

The construction task used in our experiments involves four beams and four planarly compliant nodes that are assembled together into a square structure (Figure 1, left). In a rough attempt at simulating conditions in space, the nodes are supported by casters that roll easily along the floor. A node must be braced before the end of a beam can be inserted into it; otherwise, the insertion forces can cause the node to roll away.

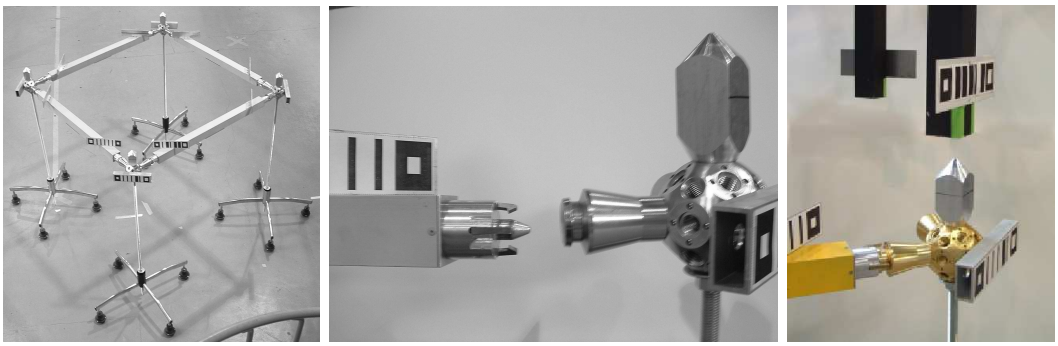

Fig. 1. The fully assembled four-beam structure (left), a view of a beam being inserted into a node (center), and a close-up of a node about to be braced by the Crane (right).

This task decomposes naturally into subtasks that can be completed by heterogeneous agents filling three different roles: an agent that provides information about the state of the world (the Roving Eye; Figure 2, left), an agent that braces the nodes during docking (the Crane; Figure 2, center), and an agent that does the actual manipulation and insertion of the beams into the nodes (the Mobile Manipulator; Figure 2, right).

To assemble one side of the square, the Crane first braces a node so that the Mobile Manipulator can insert one end of a beam into it. The Crane and Roving Eye then reposition to the other end of the beam, where the Crane braces another node while the Mobile Manipulator completes the node-beam-node subassembly. Once the beam is securely docked, the Mobile Manipulator releases it and moves to the next side of the square in order to receive the next beam and continue. This process repeats four times until the complete square is assembled. Running autonomously, the three robots can complete each side of the structure in 7.3 minutes, on average. The success rate per side for completely autonomous assembly is 75%.

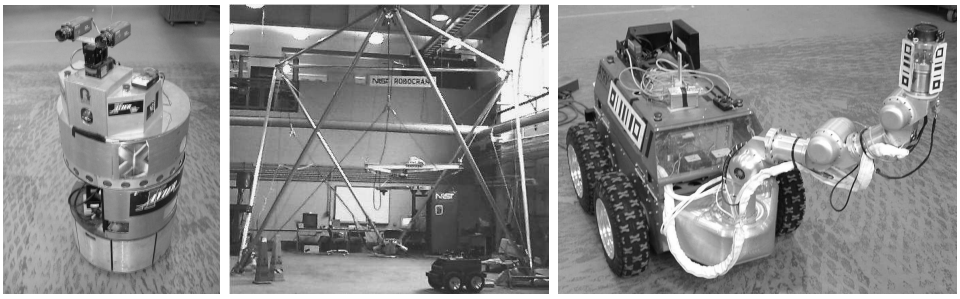

Fig. 2. The three robots used to build our square structure: the Roving Eye (left), the Crane (center) and the Mobile Manipulator (right).

B. Hardware

Our Roving Eye (Figure 2, left) is Xavier [15], a synchro-drive robot built on a RWI B24 base and equipped with a stereo camera pair mounted on a pan-tilt unit. The Roving Eye's cameras are the team's only extrinsic sensors - the Crane and the Mobile Manipulator rely on the camera data in order to complete their tasks. By using an independent sensing agent such as this, we avoid the conflicts of interest that arise when an agent has multiple duties. For instance, cameras mounted on the Mobile Manipulator might become obscured when carrying a beam, forcing the robot to compromise between manipulation and sensing, inevitably resulting in suboptimal performance.

In this scenario, we simplify the vision problem by attaching fiducials to all important objects. The Roving Eye uses the fiducials to identify and locate the objects in the workspace relevant to the task at hand. Sample fiducials can be seen on the side and wrist of the Mobile Manipulator (Figure 2, right).

The Crane (Figure 2, center) is a NIST-built RoboCrane [16], and consists of a 6-DOF inverted Stewart platform carrying a simple mechanism to allow the bracing of our nodes. The bracing mechanism is a

hollow square that can be lowered onto the top of a node, effectively preventing the node from moving (Figure 1, right). We chose the RoboCrane because it is strong enough to immobilize the nodes against sizable insertion forces by grasping them from above without impugning upon the workspace of either the Mobile Manipulator or Roving Eye. This vertical bracing lessens the interference problems that often occur when multiple robots are moving in a tight, shared workspace.

The Mobile Manipulator (Figure 2, right) consists of a Metrica/TRACLabs 5-DOF anthropomorphic arm [FIXME: CITE?] mounted on the front of an RWI ATRV-2 skid-steered base. The arm has an electromagnet on its end effector, which attaches to a metal plate fastened to the underside of each beam, allowing the Mobile Manipulator to grasp the beam. This is an ideal selection for the Mobile Manipulator, as it allows very precise manipulation of the beams while remaining low enough so as not to avoid interfering with the Roving Eye's view of the structure.

C. Architecture

In order to support the closely coupled coordination required between the robotic agents to complete the assembly scenario, we developed the Syndicate architecture. One of Syndicate's core features that enables this coordination is its three-layered approach. Each layer is associated with a different task granularity and level of abstraction about the world, and may communicate with the layers immediately above and below it. In general, higher and more abstract layers command lower, more reactive, layers, while the lower layers provide data to inform those commands (Figure 3).

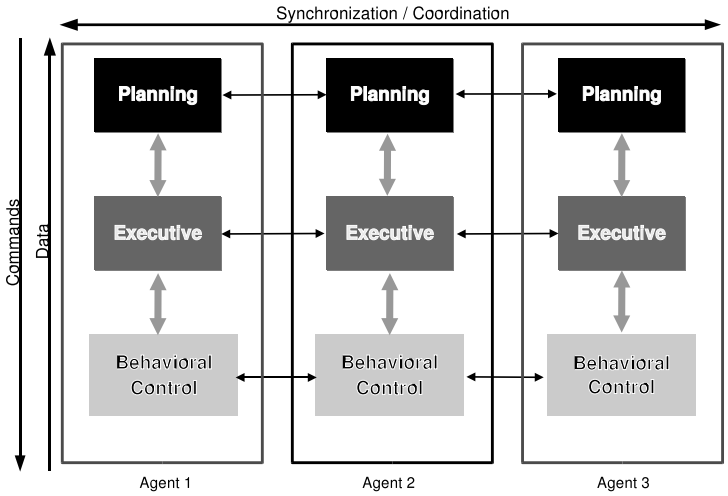

Fig. 3. The Syndicate architecture. Note that each layer may communicate directly with *all* other layers of the same type; all of these links are not depicted for reasons of clarity.

The Syndicate architecture has three layers of abstraction. The bottom-most is the behavioral layer, which deals with fine-grained, stateless control of the robots. The behavioral layer, which is based on the Skill Manager of [17], acts as the interface between the controllers/hardware and the executive layer. Thus, it views the world at a very fine temporal granularity, concerning itself with the hardware and environmental details that higher layers abstract away. By doing so, it is able to react very quickly to changes in the world. The executive layer is responsible for building and maintaining the agents' hierarchical task trees, thus managing all of the stateful task-level aspects of the system [18]. A task is loosely defined as an abstraction of one element of the scenario that requires state and/or may be decomposed into atomic behaviors in order to satisfy a goal. Many tasks rely on data passed up from the behavioral layer to decide when to change state. This relationship illustrates a secondary objective of the executive layer: configuring the behavioral layer based on the current state of the task tree. The top-level planning layer is not yet implemented. Instead, the executive layer contains a fixed task ordering for a given scenario in order to obviate the need for a planner. We are currently investigating different approaches to building a planning layer for Syndicate, which will likely include both planning and scheduling aspects, such as used in [19].

In addition to communication between the layers of a single agent, Syndicate supports communication amongst agents at each abstraction layer. This gives agents the ability to coordinate directly with other agents at all levels of the hierarchy. For example, in the executive layer, we want to ensure that the Roving Eye does not move to the next corner until the Mobile Manipulator is finished with the current beam-node docking operation, since the Mobile Manipulator depends on the raw position data provided by the Roving Eye to complete the docking. To accomplish this, Syndicate enables the executive layer on one agent to constrain its task execution with respect to a task on the other agent. Coordination at the behavioral level is also required by many tasks. For instance, to dock a beam into a node, the Mobile Manipulator's behavioral layer must communicate with the Roving Eye's in order to receive the raw position information generated by the Roving Eye's "watch" behavior, forming a distributed visual servoing loop [20]. Although our planning layer is not yet implemented, an example of coordination at that level can be seen in the FIRE project [19], where communication between peer planning layers is used for auction-based role assignment.

IV. Sliding Autonomy for Multi-Agent Teams

Even with closely coupled coordination, it is nearly impossible to prevent errors from occuring in large-scale construction tasks, such as our motivating scenario. Because of the complexity and uncertainty

involved in such domains, even a highly-specific, well-programmed robotic team can sometimes fail. One common way to compensate for this possibility is to add a teleoperation mode to the system. In this mode, instead of the robots operating autonomously, a human operator controls each of the robots during all of its tasks. Both teleoperation and pure autonomy have been shown to have distinct strengths and weaknesses. In general, teleoperation is slower, but more reliable, while full autonomy is faster, but less robust [21]. Additionally, communication lags, prevalent in space-based applications, make total teleoperation difficult and tedious. The goal of Sliding Autonomy is to allow human-robot teams to move smoothly along this autonomy spectrum, making appropriate use of the differing capabilities of the team members in order to outperform both pure teleoperation and pure autonomy.

Sliding Autonomy for multi-robot teams, however, has complications not found in single-robot systems. In general, we have found that there are three ways in which multi-agent Sliding Autonomy is more demanding than the single-agent version: (1) deciding when to ask for help, as the human is not guaranteed to be monitoring any one robot at any given time; (2) assisting the human in gaining situational awareness of the requesting robot's workspace when he is asked for help; and (3) maintaining coordination of the team as a whole when the human is controlling one of the robotic agents.

A. Our Approach to Sliding Autonomy

Our approach to Sliding Autonomy uses mixed-initiative interactions to orchestrate the collaboration between the human operator and the robotic agents. These interactions consist of either the human or the autonomous system varying who is in control of different aspects of the task; the autonomous system's motivation is to optimize metrics such as efficiency and robustness, measured by time elapsed and likelihood of success, respectively.

To implement these types of interaction, we introduce the notion of "task switching". Nearly every task in the executive layer decomposes into "monitor" and "action" components, each of which is itself a task, and may be controlled by either the human or the autonomous system. The monitoring component of a task is responsible for detecting failures and determining when the task is completed. The action subtask interacts with the robot controllers via the behavioral layer to perform the desired actions. While the majority of tasks are split in this fashion, some non-leaf nodes of the task tree that are merely responsible for creating other tasks, and thus have no obvious action/monitor split, are not divided. This methodology yields significant flexibility. For example, the autonomous system often is able to perform a task, but its sensing capabilities are not always sufficient to reliably determine when it is finished. In such a case, the human can be assigned the monitoring portion of the task, in order to ensure it is accurately

executed. A concrete example of this is the docking task carried out by the Mobile Manipulator. While the autonomous system is quite proficient at completing the task, the final tolerance is on the same order of magnitude as the noise in the Roving Eye's sensing system, on occasion making it quite difficult for the autonomous system to determine when the docking is complete. The human operator can look for additional clues in the video feed, and often can make a more accurate determination of when the docking has successfully completed (or when it is horribly stuck).

On the other hand, the action subtask may be given to the human if the autonomous system detects that it is having problems or believes the operator will be able to perform it more efficiently. In such a case, the human could be asked to complete execution, while the system monitors his progress. A prime example of this is the search task carried out by the Roving Eye when it loses sight of a fiducial of interest. Humans are better suited to the action component of this task, as they can decide where to move the camera based on their understanding of the workspace, as opposed to the blind grid-based search pattern the autonomous system uses. However, it is not obvious to the human when the system has detected a fiducial, since the human's vision is so much different from that used by the Roving Eye. Thus, in this instance, the autonomous monitoring component can work side-by-side with the human to complete the search task.

Whether a task is switched as a result of operator intervention or as a result of a system request for help, it interacts as needed with the behavioral layer to switch all necessary low-level components to the appropriate operational mode. This provides fine-grained control over operations, as the human can assume control of very specific portions of the system, while leaving the remainder under autonomous control. For instance, if the user were to take control of the search task on the Roving Eye responsible for visually searching for fiducials, the pan-tilt unit would be placed under human control, while control of the Roving Eye's base and responsibility for determining when the search was successful would remain under autonomous control. At the completion of a task, all affected behaviors are returned to the operational mode in which they were prior to the switch.

This approach to Sliding Autonomy still leaves open the questions of when and who decides to switch tasks. In order to investigate how best to structure these interactions, we have experimented with two different approaches, each of which spans the control spectrum, but varies in how tasks may be switched. System-Initiative Sliding Autonomy (SISA) enables the robot team to ask the human for assistance, but does not allow the human operator to interrupt the robots. SISA models situations where the human operator is multi-tasking and attending to other responsibilities while the robots handle the majority of the execution, so that the human has no ongoing knowledge of what is occurring in the robots' workspace.

The operator's attention is needed only when the team decides that the human is better suited for the execution of some task (see below) or when an individual robot has encountered an error condition from which it cannot recover on its own. While facilitating multi-tasking may raise the productivity of the overall system, the time it takes users to regain situational awareness of the robots' workspace when asked to help cannot be disregarded (Section IV-B). We hypothesize that SISA is the best fit when human resources are scarce and the autonomous system is reasonably proficient at error detection.

We also investigated Mixed-Initiative Sliding Autonomy (MISA). Here, while the robotic team members still can ask for help, the human operator also has the option of interrupting the system to take control of a task or subtask. While this has the potential to increase robustness, since the human operator can intervene before a robot makes a fatal error, such gains are only realized if the human actively monitors the progress of the robotic team.

The next three subsections describe our approaches to dealing with the three aspects that are endemic to multi-agent Sliding Autonomy: requesting help, gaining situational awareness, and maintaining coordination.

B. Requesting Help: System and User Modeling

In many Sliding Autonomy systems, only the human operator is able to change the control of tasks. In general, this is adequate for single-robot applications, since a human operator generally is more than capable of monitoring the status and progress of a single robotic agent, and can take over either if he feels he would do a better job than the robot or if the robot is entering into a dangerous situation. As the number of robots increases, however, it becomes harder and harder for a single operator to keep track of the status of each agent. One way of addressing this problem is to allow the robots to switch the control of tasks by asking for assistance as appropriate. Thus, a robot in trouble can request help from the human, instead of waiting for the operator to realize that there is a problem.

We have developed an approach that enables the autonomous system to make reasoned decisions about when to switch control of tasks based on current conditions and the specific operator available. These decisions are made when tasks are initially launched as well as when a failure occurs. If the operator has shown he is usually better than the autonomous system, the task will be assigned to him immediately. Alternatively, the system may try to perform the task itself, but later decide to hand control to the human if it is unable to complete the task. Such decisions need to be made in both SISA and MISA modes, as the system may request help with a task in either case.

While these task allocation decisions could be made via an arbitrary set of heuristics (such as "try

twice autonomously, then cede control to the human"), such a strategy has the potential to be extremely suboptimal, since corner cases exist for nearly every heuristic. Instead, by building empirical performance models of both the autonomous system and the human operators, we have developed a principled approach to making such decisions that allows the autonomous system to reason about expected task duration and agent reliability to decide who should perform what task.

The decision problem can be phrased as a comparison between the expected time to complete the task given that either the human or the autonomous system makes the next attempt. In order to make this comparison, we evaluate two decision trees – one where the human performs the task under consideration, the other where the autonomous system controls the task. A tree associated with the autonomous system making the next attempt is diagrammed in Figure 4.

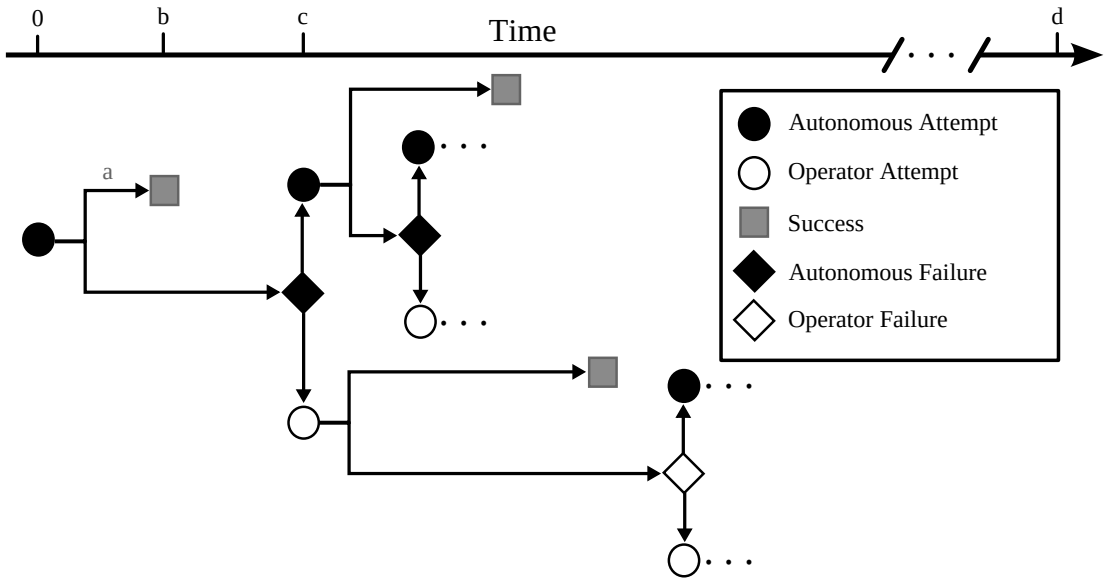

Fig. 4. An example of the decision trees that are evaluated by the user modeling system. (a) corresponds to the probability of success: $P(S_r|F_r = i)$ (Equation 2); (b) represents the expected time taken to succed: $E(t_r|S_r, F_r = i)$ (Equation 2); (c) is an example of the expected time to fail: $E(t_r|\neg S_r, F_r = i)$ (Equation 2); and (d) represents the exppected time of the entire decision tree: $E(t_s|F_r = i, F_h = j)$ (Equation 1)

There are three components to this recursive prediction: the probability of success for a given party's attempt, the expected time given success, and the expected time given failure. In Figure 4, these correspond to the probability of branch (a), timespan (b), and timespan (c), respectively. To estimate these values, we use prior observations of execution time, conditioned on the number of preceding failures by the controlling party that have occurred so far during this particular task. The number of preceding failures

is roughly equivalent to the current depth in the decision tree, and is used because a failure is empirically a good predictor of future failures (at least within our scenario). In addition, we condition our estimated time calculations on the outcome of the attempt, as failures often take significantly longer than successful attempts. Since our models are updated during task execution and are maintained on a per-operator basis, the system's decisions will dynamically change in response to the operator's current performance and will depend on the specific operator available.

By conditioning execution time on previous failures, we turn what was a multimodal distribution into a set of (more or less) unimodal distributions, which greatly eases the calculation of expected time. We estimate how long it will take to complete a task directly from these distributions:

$$E(t_s|F_r = i, F_h = j) = min \begin{pmatrix} E(t_r|F_r = i, F_h = j) \\ E(t_h|F_h = j, F_r = i) \end{pmatrix} \quad (1)$$

$$E(t_r|F_r = i, F_h = j) = P(S_r|F_r = i)E(t_r|S_r, F_r = i)+ \\ (1 - P(S_r|F_r = i)) \left(E(t_r|\neg S_r, F_r = i) + E(t_s|F_r = i + 1, F_h = j)\right) \quad (2)$$

$$P(S_r|F_r = f_r + 1) = 1.0 \quad (3)$$

$$E(t_r|S_r, F_r = f_r + 1) = E(t_r|S_r, F_r = f_r) \quad (4)$$

where:

$E(t_s|F_r = i, F_h = j)$: Expected time to complete the task, given i preceding autonomous failures and j preceding human failures. For $i = j = 0$, this corresponds to the expected time of the entire decision tree (timespan (d) in Figure 4).

$E(t_r|F_r = i, F_h = j)$: Expected time to complete the task if the autonomous system performs the next attempt, given i preceding failures. This is the expected time of one subtree of the decision tree, such as the expanded subtree depicted in Figure 4.

$P(S_r|F_r = i)$: Probability of the autonomous system successfully completing the task, given i preceding failures. This corresponds to the probability of branch (a) in Figure 4.

$E(t_r|S_r, F_r = i)$: Expected value of the distribution formed by all data points in which the task was successfully completed by the autonomous system with i preceding failures (timespan (b) in Figure 4).

f_r: Maximum number of preceding failures by the autonomous system that have occurred in practice. $f_r + f_h$ is the maximum depth of the decision tree.

The minimization in Equation 1 represents the decision about whether to assign the next attempt to the human or the autonomous system. Equation 2 weights the time taken to succeed and the time taken to fail plus the remainder of the decision tree by the probability of success or failure, respectively. While the decision tree could, in theory, continue indefinitely, our model contains a finite amount of data. Equations 3 and 4 represent the optimistic assumption that once we have passed beyond the boundaries of our data we will always succeed. This assumption serves to terminate Equation 2's recursion by setting the probability of failure to zero. In the presence of reasonably sized datasets, the effect of this assumption on the final predicted time is minimal, since the cumulative probability of reaching this base case is quite low. Also, note that the equations for $E(t_h)$ (the expected time given that the human performs the next attempt) are identical to Equations 2, 3, and 4; merely exchange the r and h subscripts.

The expected time of an attempt given its outcome ($E(t_r|S_r, F_r = i)$ or $E(t_r|\neg S_r, F_r = i)$) is treated as a sample from a static distribution when considering the autonomous system or an expert operator. This distribution is formed from all prior execution time observations that match this combination of success and preceding failures. Since it is nearly always unimodal, the expected value of such a sample is merely the mean of the component data points. However, this simple model does not apply in the case of a novice human. Since the operator is still learning, it is more appropriate to model $E(t_h)$ by predicting the next point on the operator's learning curve. We have previously conducted experiments

to determine a reasonable model for this curve and how best to use it as a predictor of the human's performance [22]. According to our data, a logarithmic curve fitted to the available data was a more accurate predictor of future performance than linear, exponential, or quadratic fits. The fit of this curve is updated, on a task-by-task basis, as more data is acquired about an operator's performance during a run. Unfortunately, it is not clear how to independently predict the time taken to succeed and the time taken to fail while learning, so we simply calculate $E(t_h|F_h = j)$ as the next point on the learning curve, as long as we believe the user is a novice. Once the operator's performance has leveled off (generally after 15-20 attempts on a particular task), we assume he is an expert and switch to using the static distribution assumption with the asymptoted data.

Branch points in the decision tree are caused by the start of the task or the failure of an attempt. There are two varieties of failures: physical and temporal. Physical failures are caused by erroneous states detected by the autonomous system or the human that force the controlling party to back off and try again. For instance, if the Roving Eye completes a visual search of its environment without finding all of its target fiducials, its search task fails. The servoing task responsible for docking a beam into a node fails if it manages to wedge the beam against the node at an angle such that docking cannot proceed without resetting. On the other hand, temporal failures occur when the human or autonomous system takes too long to accomplish a task. This threshold is generally $m + c * \sigma$, where m is the mean of the observed execution times, c is a tunable parameter and σ is the variance of the observations for the party in question. Because human operators rarely, if ever, give up on their own, temporal failures are the autonomous system's primary method for requesting the return of control.

There are currently three idiosyncracies with the way in which the user models are evaluated and interpreted by the system that curtail their usefulness. If the system's calculations show that the human is better at a certain subtask and assigns control to him, there is currently no way for the operator to hand control back to the system. Every attempt to do so simply results in the system reevaluating its decision, very likely coming to the same conclusion and passing control right back to the human. Unless the estimates of human and autonomous performance are very close, it will take a significant number of such exchanges to change the system's decision. We are considering a number of solutions to this problem, including allowing the operator to force the autonomous system to perform the next attempt, or allowing the operator to indicate he is entirely unwilling to perform the task. Secondly, giving up control over a subtask is currently counted as a failure for the party that was in control, while timing of the other party's attempt starts with the switch. Thus, failures close to the end of a subtask often lead to overly optimistic performance scores for the new controller because it/he can quickly complete the remainder of the task

and move on. One potential solution is to condition our expected time calculations on the failures of both parties (e.g. instead of $E(t_r|S_r, F_r = i)$, we would calculate $E(t_r|S_r, F_r = i, F_h = j)$). Finally, the autonomous system does not yet track the operator's current workload, and may request assistance with multiple simultaneous tasks from a single operator. Since the model does not incorporate the operator's inability to attend to more than one task at a time, this can introduce significant inefficiencies, as the autonomous system waits for the human to handle each of the tasks. The obvious solution to this problem is to track the operator's current task queue and include his expected time to service the queue when calculating the expected time to complete the task under consideration if the human performs the next attempt.

While currently we are simply comparing task execution times, we have considered using a full cost model in order to model such things as varying labor costs, the amortized cost of hardware, continuing expenses associated with teleoperation or autonomous control of the robots, the cost of repairs, etc. The specific cost function would be highly dependent on the particular domain in question, but would be parameterized at a minimum by $E(t_h)$ and $E(t_r)$, as well as domain-specific price or cost parameters. This is an area for future work.

C. Operator Situational Awareness

Another important issue in Sliding Autonomy is attaining operator situational awareness. This is particularly critical in multi-agent domains. In the single-agent domain, the operator needs to monitor only a single robot. Even if he does lose situational awareness, it is easier for the operator to remember the state of the system and use that to assist him in attaining situational awareness the next time he is asked for help. In the multi-agent domain, there are many robots with the ability to ask for help. Not only is there a potentially longer time before an operator assists a robot a second time, but the operator also has more than likely provided assistance to other robots in the interim, speeding the loss of situational awareness.

Our approach is to maintain a buffer of information of the state of the robots' workspace, including both synthesized and raw video views, and show these buffers to the operator when he is asked to assist the system. By viewing these buffers, the operator can more quickly gain situational awareness of the pertinent workspace, and can more efficiently assist the robots in their task, than he could from viewing just the current state of the system.

It remains an open question, however, what types of information and how much should be shown. There is an obvious tradeoff between performance and reliability. Having more information and a larger

buffer typically leads to a more accurate assessment, but it takes longer to attain situational awareness. Section VI desribes an experiment we performed to help quantify that tradeoff.

D. Maintaining Inter-Agent Coordination

The third issue that arises while adapting Sliding Autonomy to the multi-agent domain is how best to maintain inter-agent coordination of the robot team while one or more of its members is under operator control. We address this by enabling the robots to monitor their and others' task progress through monitoring subtasks, updating their own task execution accordingly. For example, consider a situation from our scenario in which the operator is asked to brace a node with the Crane. While the operator is performing this task, the Roving Eye continues to monitor the location of the Crane, and recognizes when the operator has finished. Then the system automatically continues with the tasks that depended on the completion of the bracing task: the Mobile Manipulator approaches the node with the beam, the Roving Eye changes its focus to the tip of the beam, and the Crane stays where it is in order to continue bracing the node.

This approach allows the team to remain coordinated during a human intervention, as long as the human does not deviate too greatly from the system's plan. The existing monitoring capabilities primarily monitor for task completion, so the human can take whatever course he wishes to complete the task at hand when he is in control of the action component of a task, and the agents remaining under autonomous control will remain coordinated. However, if the human decides to accomplish additional tasks first, undo previous work, or make arbitrary modifications to the plan, the current system will lose coordination. Open research questions in this area include the development of action recognition, plan prediction, and cooperative plan generation capabilities for the autonomous system.

E. Summary of Multi-Agent Extensions to Sliding Autonomy

To summarize, we have extended the traditional single-agent Sliding Autonomy approach to multi-agent teams. In the course of our work, we have discovered three primary differences between performing Sliding Autonomy with single agents and multiple agents: (1) Due to the human's inability to simultaneously monitor all of the agents in the team, the autonomous system must reason about when to request help, and cannot rely on the human to step in. This increases the importance of detecting that an error has occurred and makes user models a necessity. (2) In addition, the human's inability to monitor all of the agents simultaneously results in a loss of situational awareness between requests for help. Consequently, the human must first attain situational awareness before assisting the robotic team, a step that typically

does not need to occur in single-agent Sliding Autonomy. (3) Finally, the multi-agent autonomous system must maintain inter-agent coordination, even when one agent is under the control of the human.

We have conducted an experiment to compare the effects of teleoperation, MISA, SISA, and pure autonomy control strategies on robustness and efficiency, and report on the results next. In addition, we carried out an experiment to determine how different interfaces affect the human's ability to attain situational awareness, which helps encourage efficient interactions, and quantify how long he takes to do so, which can be used to refine the user model.

V. Multi-Agent Sliding Autonomy Experiment

This experiment investigated the effects of mixing full system autonomy with teleoperation on the overall efficiency and robustness of our multi-agent system.

A. Methodology

For this study, expert users of the system performed a number of trials for each mode of interaction with the system. We had initially hoped to use operators unfamiliar with the project, but found that the time needed for novices to become proficient with the system was excessive: after 14 hours of training, the novice users were still slower than experts by a factor of three. Since in order to obtain meaningful results, we needed our subjects to be roughly on par with the performance of the autohomous system, we changed the study to employ two project members.

During an earlier pilot study, we determined that the time taken for the system to complete a beam-node-beam subassembly was independent of the side of the structure by using a double-sided t-test with a 0.95 confidence threshold. Therefore, our unit of analysis was a one-beam subassembly, as detailed in Section III-A. Each subject performed this assembly task 12 times in teleoperation mode, 16 times in System-Initiative Sliding Autonomy (SISA) and 20 times in Mixed-Initiative Sliding Autonomy (MISA). The 48 trials were performed in random order. Together with 24 runs of the autonomous system,that gives a total of 120 datapoints.

In order to create a semi-realistic teleoperation experience, the subject sat at a workstation facing away from the robots and the workspace. She was able to see only the raw video output from one of the Roving Eye's cameras and the output of a visualization tool, which displays depth information relevant to the current task (see Figure 5), as provided by the Roving Eye. The robots were controlled via a 6-DOF "Space Mouse" [23].

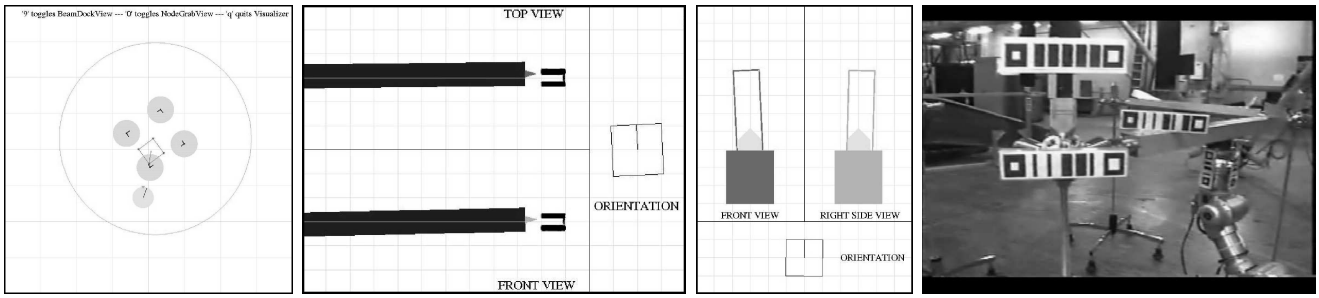

Fig. 5. Screen shots of the visualization tools used for controlling the robots. From left to right: a bird's eye overview of the workspace, a closeup used during beam docking, another closeup used during node bracing, and the raw video feed provided by the Roving Eye.

During teleoperation runs, subjects indicated which subtask of the overall assembly they were working on. This selection was used to extract timing information for each user to initialize her user model for the Sliding Autonomy trials. For Sliding and full Autonomy, the timing information was logged directly by the system. At the end of each run, the subjects completed a NASA-TLX survey [24] to assess her perceived workload while controlling the robots. This survey takes into account factors such as mental, temporal, and physical demand, as well as effort and frustration.

In order to emphasize the difference in initiative between the two Sliding Autonomy modes, the users performed a distractor task while using SISA. The screen of their workstation and the video feed from the Roving Eye were turned off. The only feedback available were audio cues asking for help. During MISA, the users actively followed the system's progress, and they were told to make use of their ability to actively take over control whenever they saw the opportunity to complete a subtask faster than the system.

B. Results

	Time to Completion	**Success Rate**	**Human Workload**
Teleoperation	worst	best	worst
Autonomous System	best	worst	best
System Initiative	medium	medium	good
Mixed Initiative	good	good	medium

TABLE I
EXPECTED RESULTS FOR SLIDING AUTONOMY PERFORMANCE.

Table I summarizes our hypotheses about the effects of introducing Sliding Autonomy into our system

on three relevant metrics. With the exception of several cells highlighted in bold in Tables II and III, the results we obtained agreed with our expectations. For instance, teleoperation (729–911 seconds) took 1.5–2 times as long as any of the modes involving autonomy (437–492 seconds). The times and TLX results of failed runs were not included in the results.

Both users generally followed the hypothesized trend that the success rate would increase as human involvement increased: pure autonomy was the worst at 75%, followed by SISA, MISA, and finally teleoperation, which had an average 96% success rate. Note that the users did not give up on the failed teleoperation runs; instead, the system reached a point of terminal failure.

Not surprisingly, due to the subjective nature of the NASA-TLX workload survey, the actual values varied strongly between users, although trends were consistent. Teleoperation had by far the highest workload, approximately twice that of the Sliding Autonomy cases (the rightmost column of Tables II and III).

User 1	**Mean Time to Completion [standard deviation]**	**Success Rate (# of trials)**	**TLX Workload [standard deviation]**
Teleoperation	729 [139] seconds	100% (12)	42 [10]
Autonomous System	437 [94] seconds	75% (24)	0
System Initiative	**462 [63] seconds**	**75% (20)**	16 [16]
Mixed Initiative	**492 [140] seconds**	81% (16)	17 [11]

TABLE II

RESULTS FOR USER 1. DISCREPANCIES COMPARED TO THE EXPECTED RESULTS ARE HIGHLIGHTED IN BOLD.

User 2	**Mean Time to Completion [standard deviation]**	**Success Rate (# of trials)**	**TLX Workload [standard deviation]**
Teleoperation	911 [193] seconds	92% (12)	71 [9]
Autonomous System	437 [94] seconds	75% (24)	0
System Initiative	458 [106] seconds	85% (20)	**50 [14]**
Mixed Initiative	445 [72] seconds	88% (16)	**34 [11]**

TABLE III

RESULTS FOR USER 2. DISCREPANCIES COMPARED TO THE EXPECTED RESULTS ARE HIGHLIGHTED IN BOLD.

Comparing our two users side by side with the autonomous system we note a completion time comparable with that of the autonomous system for both forms of Sliding Autonomy and a much longer completion time for teleoperation (Figure 6, left). The teleoperation time to completion also shows differences between the two users. In the center of Figure 6 we see an upwards trend of success rate proportional to the amount of human involvement during the assembly task. Finally, Figure 6 (right)

shows that the perceived workload measured by the TLX survey trends are not consistent between the two users - the second user found SISA to be mentally frustration, perhaps due to the difficulty of attaining situational awareness. The autonomous system is omitted from this chart for the obvious reason that there was no human operator involved and consequently no reported workload.

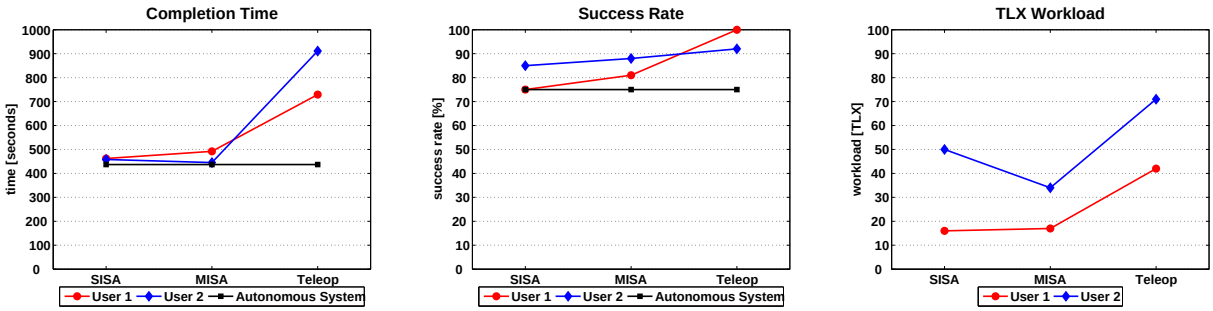

Fig. 6. Comparison of our two subjects and the fully autonomous system. Completion time and success rate generally follow our expectations, but perception of workload is very task and user dependent.

The histograms of User 1's performance (Figure 7) show again that the autonomus modes are clearly faster than teleoperation. The fastest run time was recorded under autonomous operation at just over 300 seconds. At the same time, a large portion of the teleoperation runs took almost 900 seconds, much longer than the slowest run during an autonomy trial.

C. Discussion

When comparing the two extremes of the autonomy spectrum (pure autonomy and complete teleoperation), it is obvious that there is an inherent trade-off of speed against robustness. If we were willing to allow an increase of 50-100% in the time needed to complete the structure, we could achieve near-perfect reliability by allowing the human to teleoperate everything. However, our operator workload data indicate that in addition to the significantly increased time to complete the assembly, operators would swiftly become mentally overloaded in addition to being unable to multi-task during assembly. Depending on the situation, this may or may not be an acceptable solution.

Our experimental results suggest that this dilemma may be resolved by employing some form of Sliding Autonomy. As is shown in Figure 6, adding any amount of autonomy reduces the required time to be comparable with the purely autonomous approach, while any amount of human involvement increases the reliability of the overall system. Sliding Autonomy sits in the intersection of these two trends. It

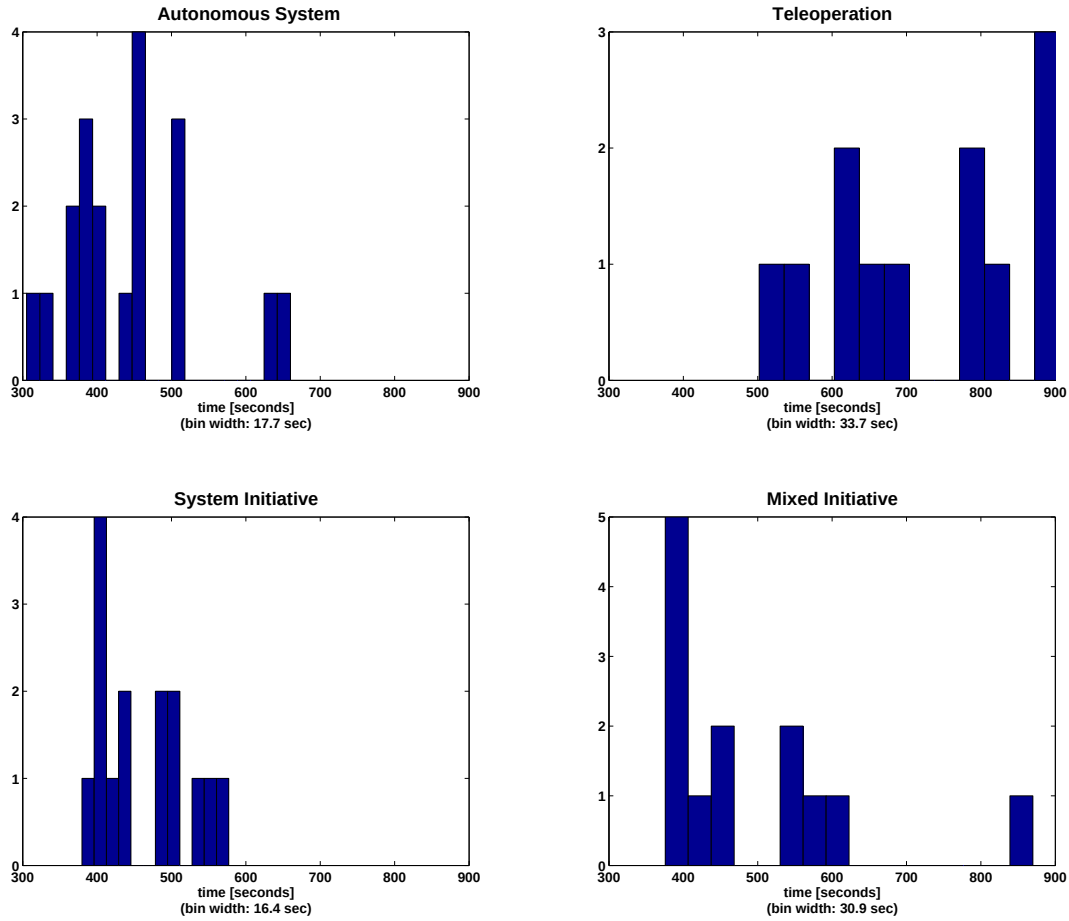

Fig. 7. Histograms comparing User 1's performance when using the the different Sliding Autonomy modes. The horizontal axis marks completion time in seconds, and the height of the bars shows the number of experimental runs for each time interval.

combines the low completion time advantage of autonomous operation with the increased success rate due to human involvement. The rise in system reliability during Sliding Autonomy operation can be attributed to the operator's intuition and ability to quickly understand problematic conditions and then initiate recovery measures to help the system avoid failure conditions. In addition, there is a clear benefit to introducing Sliding Autonomy from a workload perspective. Since the system still performs tedious tasks autonomously, the perceived workload reported by our users was significantly lower than during pure teleoperation.

The subjective nature of the TLX scores does not allow a direct user to user comparison, but we can make statements about trends in the data. We expected SISA to have the lowest workload because the

user has to work with the system onlywhen asked for help. However, there are at least two factors that are not captured well within the TLX framework: boredom and the frequency of interruption by the system. Both of these are directly linked to the specific user's ability to perform the task at hand. For example, a relatively strong user will likely be asked for help with many tasks, to the point where she is called back to help with the next task almost as soon as she returns to her distractor task from assisting the team. On the other hand, a relatively weak user will not be asked for help very much; she may be called to help only a few times during a 15-minute experimental run. Depending on the individual user, this could be interpreted either as a relaxing situation with low workload, or as a frustrating situation where the individual has to be on call for the system but is never asked to do anything. This frustration is likely responsible for high workload results such as the data shown for User 2 in Figure 6.

The multi-modal grouping of the completion time results shown in the histograms in Figure 7 corresponds to assembly attempts with varying degrees of success. The left-most group represents smooth runs without any failures, and subsequent groups indicate increasing amounts of difficulty or near-failure conditions.

There are three discrepancies between our hypotheses and our actual results: User 2's workload, User 1's mean time to completion in the MISA and SISA cases, and User 1's success rate uder SISA (the bolded entries in Figures II and III). The high workload of User 2 can be explained by the subjective nature of the TLX survey, as discussed above. User 1 took longer on average to complete the task under MISA than SISA (although there is not a statistically significant difference between the means). This may be due to an overeager operator intervening in cases where the autonomous system is in fact more efficient, resulting in an overall decrease in efficiency. Finally, User 1's success rate under SISA is not superior to the autonomous system's. Success rate improvement between pure autonomy and SISA is dependent upon the autonomous system's ability to discern failures with which it requires assistance; if few discernable failures occur, little gain will be realized by applying SISA.

Our experiment shows that adding a human agent to a multi-robot team via a Sliding Autonomy framework combines the advantages of autonomous robot operation with the reliability of teleoperation at a mental workload level tolerable by the operator. The amount of attention the operator pays to our task has no measurable effect on the time to task completion compared to fully autonomous operation, but it does manifest itself in overall system robustness: the greater the operator's involvement in the team's operation, the higher the success rate.

Purely from a system performance point of view, which form of Sliding Autonomy (SISA or MISA) is the better choice depends heavily on the system as a whole. If the available operators are comparable in

skill level with the autonomous system, and the system is able to perform significant portions of the task on its own, then the humans can productively multi-task when operating under SISA. With the system able to reliably detect failure conditions, the slightly lower success rate compared to MISA is outweighed by the fact that a few operators can oversee several different teams at the same time. The additional time required to attain situational awareness can be minimized by selecting the most effective user interface (see Section VI).

For comparatively weak human operators, a similar multi-tasking argument holds, but a higher degree of autonomy is required of the system since the operator's ability to help can be rather limited. For very skilled humans, however, their abilities often lead to them being continously asked for help. Switching between different teams performing different tasks very often and rapidly is more confusing and stressful to the operator than helpful to any team. Instead, for strong operators, a MISA setting is often the more appropriate one, since it allows the human to focus her attention on a specific task so that the team can benefit maximally from the available resource. Clearly, if the autonomous system were unable to detect most failures, MISA would be the preferred method in all cases in order to compensate for the autonomous system's lack of reliability.

VI. Situational Awareness Experiment

In addition to experimentally evaluating the differences between our different approaches to multi-agent Sliding Autonomy, we also conducted an experiment to test how well users attain situational awareness of our robotic system. In order to help the operator gain situational awareness more quickly when asked for help, we maintain a buffer containing the state of the system over the last n seconds, which can be replayed to the operator when the system asks for help. Two natural questions arise: "What kinds of information should be included in the buffer?", and "How much data should it save?". To help answer these questions, we tested how quickly users attain situational awareness of our system when using various combinations of displays and data buffer lengths. In addition to informing the design of operator interfaces, this experiment provides the information needed to account for the time necessary to gain situational awareness in the system's model of the human (Section IV-B). If the types of information available to the teleoperator were to change (e.g. if a sensor fails), the system could use the data from this experiment to dynamically update its user model appropriately.

A. Methodology

The experiment tested four different combinations of information streams. The first was simply a video feed from one of the Roving Eye's cameras (Figure 8.2). The second was the Roving Eye video along with the synthesized "technical drawing"-style visualizer (Figures 5 and 8.1), showing the relative positions of the beam and node from above and in front of the beam. The third was the Roving Eye video along with two other video feeds - one from a fisheye camera placed on top of the Crane, looking down (Figure 8.4), and one from an external camera placed outside the robot workspace, looking towards the structure (Figure 8.3). Finally, the fourth combination included all of the above elements.

We also varied the length of the data feed that was presented to the subjects. The four possible lengths were 0 (still shot), 5, 10, and 20 seconds. This, in combination with the four different displays, yielded a total of 16 different test conditions.

During each trial, each of the 32 subjects was shown the data buffer from an attempted docking, and was asked to identify through a dialog (Figure 8.5) why the robot requested help. The experimental procedure was a combination of training and testing. The subject's training began with reading a written overview of the task and hardware at hand, with the experimenter answering any questions. The subject was then shown one example of each of the seven types of errors via the graphical interface (Figure 8), using the maximal data and 20-second playback condition.

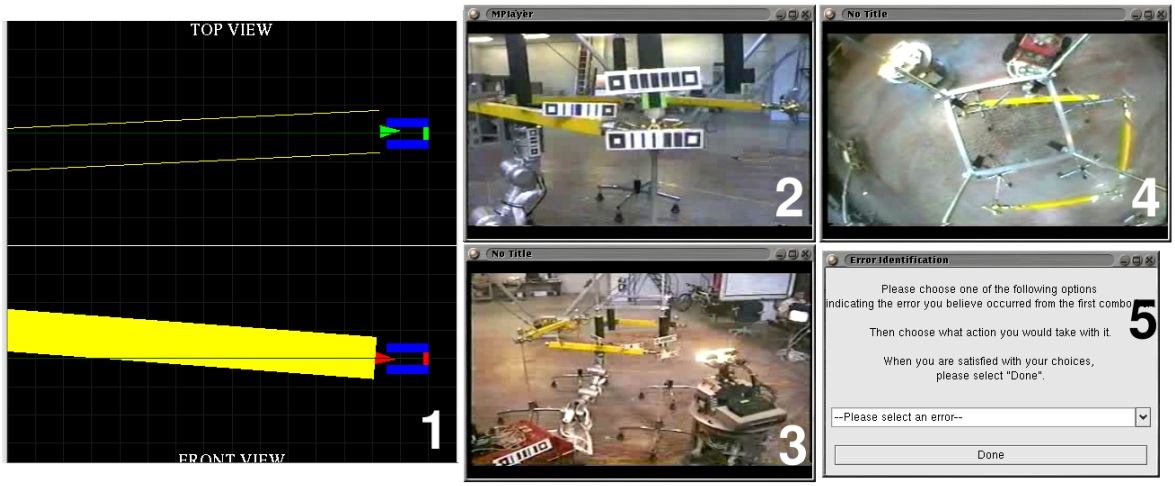

Fig. 8. The subject interface for the situational awareness experiment, including three video streams (the Roving Eye's cameras (2), an external camera (3), and a Crane-mounted camera (4)), a synthesized view of the beam and node (1), and the error categorization input dialog (5).

After training, the subjects began the actual experiment, playing Crack Attack (a Tetris-like game [25])

in between trials to simulate multi-tasking. Each user was tested on four of the 16 conditions, performing six trials per condition. We applied Latin squares to both the data feed and length effects to account for ordering effects. A Latin square is a statistical technique that allows experimenters to test effects while controlling for two other known sources of variation (here, inter-subject variability and ordering effects).

B. Results

During this experiment, we recorded the time that elapsed between the beginning of playback and when the user chose which error she believed had occurred. We also recorded her response, to allow analysis of user accuracy. Users were not allowed to choose a response until they had watched the entire playback clip. Also, although users were able to change their answer, we only considered the data collected from their final responses. We used a univariate ANOVA test to analyze this data.

The data suggest that with respect to response time, the best display and data feed length combination is the Roving Eye video plus visualizer, viewed for between 5 - 10 seconds (Figure 9, left), as this combination had the shortest average response time. Considering solely data feed length, significance was found between 0 and 5, 0 and 10, 5 and 20, and 10 and 20 second playbacks. If instead we consider the composition of the display, significance was found between the Roving Eye and the Roving Eye plus other videos displays, the Roving Eye and all displays, the Roving Eye plus videos and Roving Eye plus visualizer displays, and the Roving Eye plus visualizer and all displays.

With respect to accuracy, the best data feed condition is that with the longest (20 second) data feed length (Figure 9, right). Similarly, the display that had the highest accuracy was the Roving Eye video plus other videos (Figure 9, right), although it is not significantly different from the all displays condition. Considering data feed length, significance was found between 0 and 10, 0 and 20, and 5 and 20 second playbacks. Significance in display composition was found between the Roving Eye and Roving Eye plus videos displays, as well as the Roving Eye and all displays.

C. Discussion

If response time is the only metric under consideration, the results are clear - the best conditions were the Roving Eye video alone or the Roving Eye Video plus visualizer, with 5 - 10 seconds of playback. We believe that this is because as more raw videos are added, the mental overhead required to process, interpret and merge the available information increases, resulting in slower response times. The more abstract visualizer view, however, requires less mental overhead, and thus its inclusion does not significantly increase user response time. Similarly, we believe that a 5 - 10 second playback was

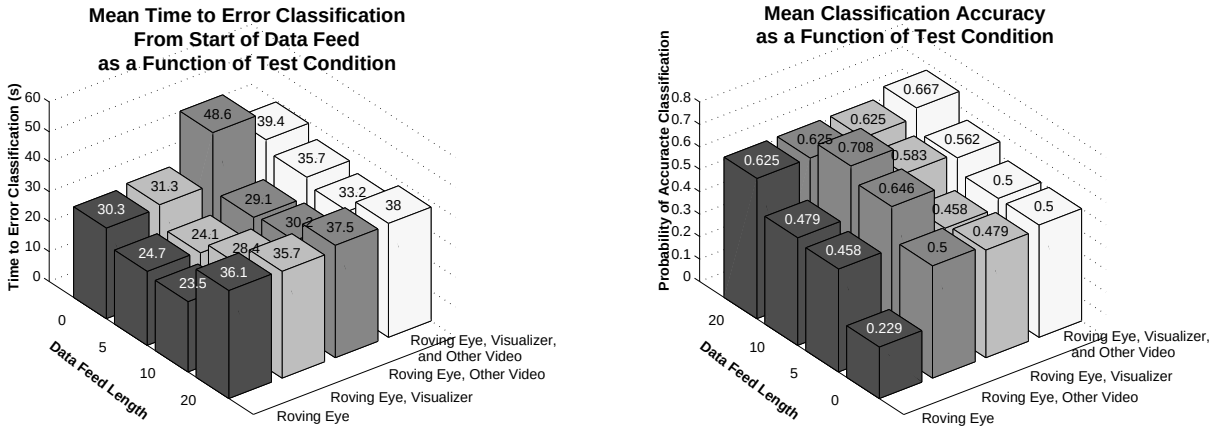

Fig. 9. The effects of available information and data feed length on how long subjects took to choose an error (left) and how accurate their decisions were (right). Note that the ordering differs between the two figures, although the shading is consistent.

the best range because with longer data feed lengths the amount of meaningful information the user can process and remember plateaus, making the extra information relatively less useful (when only considering response time) in the decision making process.

The accuracy measures, however, suggest a slightly different story. When considering this metric, the longer the data feed playback, the more accurate the users' answers become. This is most likely because users can make a more informed decision, even though they take longer to respond in order to process the additional information. The same argument applies to the display condition. Although the extra videos require more time to process, they allow users to make more informed decisions, leading to higher accuracy.

From this experiment, we can make some recommendations about attaining situational awareness in systems such as ours that involve a remote human as part of a multi-agent team. There is obviously a trade-off between accuracy and response time: in general, the most accurate conditions were those that had the longest response time. This effect, however, may decrease as users become more expert: the information they derive from the data feed may plateau earlier. Thus, a follow-up to this experiment should use expert users as subjects, and examine whether the accuracy trends still hold or if they begin to correlate with the response time results. Either way, a choice needs to be made between efficiency and accuracy. If timing is critical, then it might be worthwhile to accept a lower accuracy in order to

encourage the operator to respond more quickly; similarly, if accuracy is the more important metric, then users should be given as much information as possible, and sufficient time to process and merge the information, in order to allow them to make more informed decisions. It is important to keep in mind, however, that our measures of time did not include time for task completion. Because a misdiagnosis may lead to wasted recovery effort, it is quite possible that classification errors can affect the speed of the system to a greater extent than these results indicate.

VII. Future Work

Determining how to proceed when the human relinquishes control of a task remains an open research question. If the human returns control after completing the assigned task (and only the assigned task), it is straightforward: our current system is able to monitor the task's completion and continue with the tasks that were dependent upon it. However, the human may instead perform additional or different tasks than those that the system had planned on. In addition to deducing the human's goals and tracking the arbitrary effects her actions may have on the structure, the system needs to be able to replan to accomplish the scenario's goals from whatever state in which the human relinquishes control.

Our current scenario encompasses a relatively complex task, but contains minimal coordinated manipulation of a single object by multiple manipulator robots. Our earlier work involved a much simpler task, but required extensive coordinated manipulation of this type [26]. We are now moving to a new scenario that combines a complex three-dimensional assembly with a need for this type of coordinated manipulation. Addressing both these issues while using Sliding Autonomy in a multi-agent setting should uncover many new issues; for example, deciding how best to coordinate tightly coupled manipulation between a human and robot, instead of between two robots. Additionally, we are looking into adding additional robots with overlapping capabilities to the team. With these additional robots, the planning problem will become much more interesting, as a wider variety of solutions to the scenario will be possible. Their overlapping capabilities will also allow the robots to ask each other for help in addition to, or instead of, asking the human. This additional step will allow the human to be treated as simply another agent in the system, just one with a slightly different skill set.

VIII. Conclusions

We have presented various issues involved with extending Sliding Autonomy into a multi-agent domain. One of the main challenges is allowing the robots to ask for help, since the human is not always guaranteed to be paying attention to each robot at any given time. We make this possible by incorporating user models

into our system, which allow the robotic agents to make informed and reasonable decisions about when to request the operator's assistance. Experimental data showed that this way of incorporating a human into a multi-robot team combines the advantages of autonomous robot operation with the reliability of teleoperation, resulting in a more efficient and robust system as a whole.

By allowing the system to ask the operator for help, we also introduce the question of how to best enable operators to quickly gain situational awareness of the robot's workspace and state, so that they can provide assistance more quickly and effectively. To this end, we conducted an experiment testing different operator displays and playback times to find out which combination resulted in the most efficient interaction. Based on the results, we feel that a human operator in our system will be able to quickly gain situational awareness and assist any robotic team-member that asks for help, whether or not the operator had previously been paying attention to the situation.

In order to allow inter-agent coordination even when a human is controlling some task, we allow the robotic team members to monitor their and other team members' progress as related to their current task. This allows the robots to continue with their respective tasks even when their team members are being controlled by an operator or are waiting for an operator response. In other words, a team member can use its knowledge of the other agents' actions to ensure that its tasks remain coordinated. Our experiments have shown that this is an effective way of maintaining task coordination while using Sliding Autonomy.

Overall, we have shown some shortcomings of controlling large-scale construction systems through both full autonomy and complete teleoperation. We have demonstrated that by allowing humans to work as team members via Sliding Autonomy in a multi-agent system, a synergy of complementary skills and abilities emerges that increases the system's overall robustness and efficiency.

ACKNOWLEDGMENTS

The authors would like to thank the many individuals who contributed to the DIRA and Trestle projects over the years: Rob Ambrose, David Apfelbaum, Jon Brookshire, Rob Burridge, Brad Hamner, Dave Hershberger, Myung Hwangbo, David Kortenkamp, Simon Mehalek, Metrica/TRACLabs, Josue Ramos, Trey Smith, Pete Staritz, and Paul Tompkins.

REFERENCES

[1] P. Valckenaers, H. V. Brussel, L. Bongaerts, and F. Bonneville, "Programming, Scheduling, and Control of Flexible Assembly Systems," in *Computers in Industry*, vol. 26, no. 3, Aug. 1995, pp. 209–218.

[2] A. Stroupe, T. Huntsberger, A. Okon, and H. Aghazarian, "Precision Manipulation with Cooperative Robots," in *Multi-Robot Systems: From Swarms to Intelligent Automata*, L. Parker, F. Schneider, and A. Schultz, Eds. Springer, 2005.

[3] R. B. Gillespie, J. E. Colgate, and M. Peshkin, "A General Framework for Cobot Control," in *International Conference on Robotics and Automation*, Detroit, MI, 1999.

[4] W. Wannasuphoprasit, P. Akella, M. Peshkin, and J. E. Colgate, "Cobots: A Novel Material Handling Technology (best paper award)," International Mechanical Engineering Congress and Exposition, Anaheim, ASME 98-WA/MH-2, 1998.

[5] R. R. Burridge, J. Graham, K. Shillcutt, R. Hirsh, and D. Kortenkamp, "Experiments with an EVA Assistant Robot," in *Proceedings of the 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*, Nara, Japan, 2003.

[6] T. Fong, C. Thorpe, and C. Baur, "Robot, Asker of Questions," *Robotics and Autonomous Systems*, vol. 42, 2003.

[7] G. Dorais, R. Banasso, D. Kortenkamp, P. Pell, and D. Schreckenghost, "Adjustable Autonomy for Human-Centered Autonomous Systems on Mars," Presented at the Mars Society Conference, 1998.

[8] D. Kortenkamp, R. Burridge, P. Bonasso, D. Schrenkenghost, and M. Hudson, "An Intelligent Software Architecture for Semiautonomous Robot Control," in *Autonomy Control Software Workshop, Autonomous Agents 99*, 1999.

[9] P. Scerri, D. Pynadath, and M. Tambe, "Towards Adjustable Autonomy for the Real World," *Journal of Artificial Intelligence Research*, vol. 17, pp. 171–228, 2002.

[10] R. T. Maheswaran, M. Tambe, P. Varakantham, and K. Myers, *Lecture Notes in Computer Science*. Springer-Verlag GmbH, 2004, ch. Adjustable Autonomy Challenges in Personal Assistant Agents: A Position Paper.

[11] H. A. Yanco and J. Drury, ""where Am I?" Acquiring Situation Awareness Using a Remote Robot Platform," in *IEEE Conference on Systems, Man and Cybernetics*, October 2004.

[12] M. R. Endsley, S. J. Selcon, T. D. Hardiman, and D. G. Croft, "A Comparative Analysis of SAGAT and SART for Evaluations of Situation Awareness," in *42nd annual meeting of the Human Factors and Ergonomics Society*, Chicago, October 1998.

[13] M. A. Goodrich, D. R. Olsen, J. W. Crandall, and T. J. Palmer, "Experiments in Adjustable Autonomy," in *Proceedings of the IJCAI Workshop on Autonomy, Delegation and Control: Interacting with Intelligent Agents*, 2001.

[14] D. Hershberger, R. Simmons, S. Singh, J. Ramos, and T. Smith, *Robot Teams: From Diversity to Polymorphism*. AK Peters, 2002, ch. Coordination of Heterogeneous Robots for Large-Scale Assembly.

[15] R. Simmons, J. Fernandez, R. Goodwin, S. Koenig, and J. O'Sullivan, "Lessons Learned From Xavier," *IEEE Robotics and Automation Magazine*, vol. 7, no. 2, pp. 33–39, June 2000.

[16] R. Bostelman, J. Albus, N. Dagalakis, and A. Jacoff, "RoboCrane Project: An Advanced Concept for Large Scale Manufacturing," in *Proceedings of the AUVSI Conference*, Orlando, FL, July 1996.

[17] R. Bonasso, R. Firby, E. Gat, D. Kortenkamp, D. Miller, and M. Slack, "Experiences with an Architecture for Intelligent, Reactive Agents," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 9, no. 2-3, pp. 237–256, 1997.

[18] R. Simmons and D. Apfelbaum, "A Task Description Language for Robot Control," in *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, Victoria, Canada, 1998.

[19] D. Goldberg, V. Cicirello, M. B. Dias, R. Simmons, S. Smith, and A. Stentz, "Market-Based Multi-Robot Planning in a Distributed Layered Architecture," in *Multi-Robot Systems: From Swarms to Intelligent Automata: Proceedings from the 2003 International Workshop on Multi-Robot Systems*, vol. 2. Kluwer Academic Publishers, 2003, pp. 27–38.

[20] D. Hershberger, R. Burridge, D. Kortenkamp, and R. Simmons, "Distributed Visual Servoing with a Roving Eye," in *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, Takamatsu Japan, October 2000.

[21] F. W. Heger, L. M. Hiatt, B. Sellner, R. Simmons, and S. Singh, "Results in Sliding Autonomy for Multi-Robot Spatial

Assembly," in *8th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*, Munich, Germany, September 5-8, 2005.

[22] B. Sellner, R. Simmons, and S. Singh, "User Modelling for Principled Sliding Autonomy in Human-Robot Teams," in *Multi-Robot Systems: From Swarms to Intelligent Automata*, L. Parker, F. Schneider, and A. Schultz, Eds. Springer, 2005.

[23] "SpaceMouse Manual," Company website. [Online]. Available: "http://www.3dconnexion.com/spacemouseplus.htm"

[24] S. G. Hart and L. E. Staveland, *Human Mental Workload*. Amsterdam: North-Holland, 1988, ch. Development of the NASA-TLX (task load index): Results of Empirical and Theoretical Research, pp. 139–183.

[25] D. R. Nelson and A. Sayman, "Crack Attack! Manual," Game's home page: http://www.nongnu.org/crack-attack/, http://aluminumangel.org.

[26] J. Brookshire, S. Singh, and R. Simmons, "Preliminary Results in Sliding Autonomy for Assembly by Coordinated Teams," in *Proceedings of the Conference on Intelligent Robots and systems (IROS)*, Sendai, Japan, September 28 - October 2, 2004.