

Experiments in Implicit Control

Katia Sycara
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
katia@cs.cmu.edu

Michael Lewis
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA 15260
ml@sis.pitt.edu

Abstract

Human agent cooperation can run the gamut from a priori division of tasks, to explicit delegation of responsibility, to tightly coupled behavior without any apparent division of tasks or responsibilities. This paper examines the potential of tightly coupled behavior drawing examples from collision handling in virtual environments, path planning for military teams, and robot control for urban search and rescue. In each case an implicit aiding scheme in which an agent shadowed the human's actions proved more effective than explicit divisions of labor or responsibility.

1 Introduction

In most cases the limiting factor in interacting with automation lies not in any lack of capability by the machine but in the difficulty of communicating our intentions to it. While the computer offers the ultimate in flexible automation, instructing it do what we wish may be arbitrarily hard for humans as demonstrated by the difficulty experienced in using traditional programming and scripting languages. It is much simpler for us to drive a car or set a table than to instruct a robot to do so, yet we would rather adjust a thermostat or program a milling machine once, than repeatedly perform these actions by hand. One facet of understanding mixed initiative systems therefore lies in understanding what makes some tasks so difficult to communicate and others so easy.

Lewis (1998) proposed a model for human-agent interaction (HAI) based on Norman's (1986) model of "seven stages of action" shown in Figure 1. The Norman model (labeled reference) depicts interaction as a loop in which a series of perceptual stages are followed by action stages

that in turn give rise to new rounds of perception and action. According to the model, errors may happen at any of the stages so an error at the "interpret" stage might occur if someone trying to program a vcr did not realize that the machine was actually in the "time setting" mode and therefore inappropriately advanced the time when they actually intended to advance the channel. The HAI model categorizes agent roles as automating either the perceptual side of the cycle (filtering), the action side of the cycle (adapting/anticipating), or closing the cycle (supervised autonomy).

Equating "software agents" with programs which automate aspects of interaction agrees with many popular definitions. Agents that gather, filter, or summarize information for a user automate the perceptual cycle. Those that perform a prespecified asynchronous task such as an eSnipe bidder (www.esnipe.com) or similar auction bot are examples of supervised autonomy. Agents that learn or adapt to user preferences automate the action chain by inferring or predicting the user intent and then performing the "intended" action. Communication bottlenecks can be found for each of these forms of automation but appear particularly acute for automating actions which depends on recognizing goals and intentions.

Just as friends and spouses cannot always guess what we want unless we tell them, it is unreasonable to presume that primitive inductive learning methods will empower agents to do so. Many of the routinized cognitive tasks we would like to delegate to agents involve relatively complex sequences of behaviors, such as extracting particular information from a site on the web, monitoring a news group for a recent topic or performing some set of conditional actions. What is significant about this list is that these requests all involve sequences of actions which although individually constrained combine to form

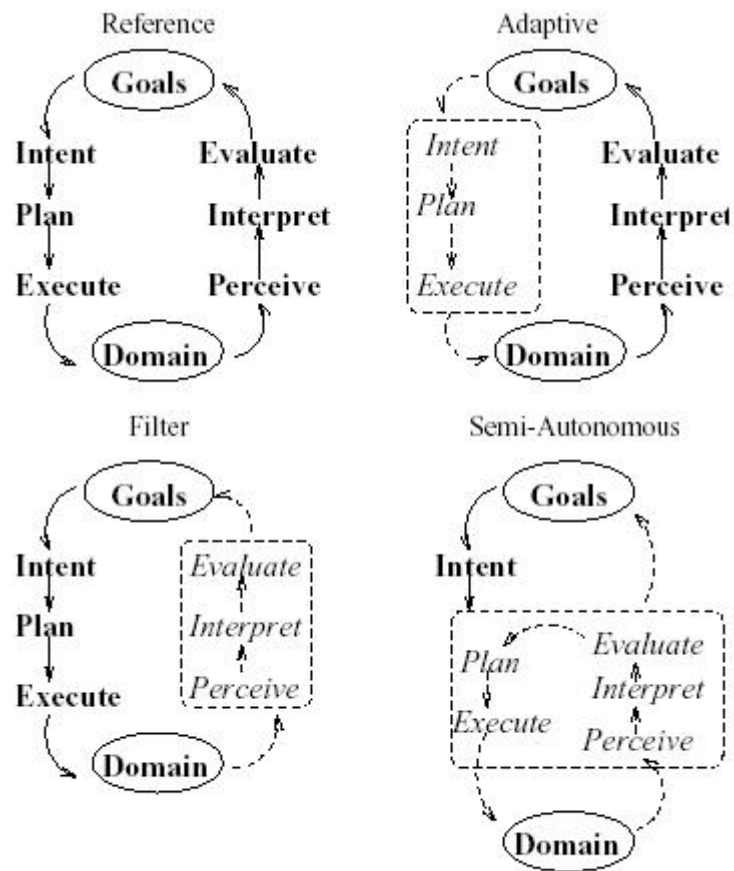


Figure 1. HAI Reference Model

unlearnable wholes. The problem of communicating these desires directly, however, is precisely the problem of end user programming that intelligent agents were supposed to resolve. So the plan recognition problem must be solved before the action cycle can be automated in any general way.

Against this discouraging backdrop there appears to be a class of problems for which automating action is not only feasible but human-agent communication can be made apparently effortless. For these problems the match between the user's goals and the automation's performance can be made so close that awareness of the automation fades away and the user feels himself to be interacting directly with the domain (Norman 1998). Under these conditions the penalty for high bandwidth communication is eliminated and even relatively complex autonomous actions are no longer disruptive.

Defining characteristics for such design solutions are:

- direct correspondence between interaction with automation and interaction with the domain
- Restrictive, recognizable and repeatable automation subtasks

- Means for unambiguous communication of intent

The resulting interactions more resemble the close following and complex behaviors found in dance than the explicit effort and expressiveness of verbal communications. As the dance metaphor implies this kind of synchronization is particularly appropriate for spatial movement but can fit other cases where the automated response is uniquely determined.

2 Collision Handling

To navigate a cluttered environment, one must be able to plan a route without considering every step. In the real world, we use low-level psychomotor behaviors to get around objects. [Turvey and Kugler, 1987] For example, suppose someone is in a room and moving toward the door, but there is a chair in the way. Rather than plan some perfect route around the chair, she will most likely move straight toward the door, and step around the chair when she gets to it; all without much thought. Unfortunately, most virtual environment (VE) interfaces available today provide neither the range of perception nor the body control for the user to

employ these reactive behaviors. The problem is especially acute with the most commonly used physical computer interface: the screen-and-mouse.

Collision handling strategies generally fall into the three categories, which we term, “ghost”, “clunk” and “slip”:

Ghost Mode

Ghost mode is actually the absence of collision handling—one simply floats through objects and walls. In a VE, ghost mode reduces or eliminates the need to avoid objects, but limits the types of applications one can sensibly build. For example, ghost mode might be perfect for exploring some otherwise closed structure, like the human body. However, it would be inappropriate for many military training applications.

Operationally, the ghost mode user can also get lost in places where the visual field is cluttered, as with most locations in the maze. Even under the best of circumstances, ghost users would have to learn both sides of every wall to fully understand the maze. Added baffles between the walls make it more confusing, (fig 1) but this is representative of the topologies one might find in simulated office buildings, nuclear power plants or complex machinery.

Clunk Mode

In clunk mode, the user simply stops when he encounters an object. In uncluttered environments [Sperger, 1996] this works reasonably well, but not in spaces that are narrow or cluttered. There, the merest brush with any object causes the user to come to a full stop. Before he can start moving again, he must back up and reorient. Sometimes, objects are out of the user’s field of view, which in most applications is only about 100 degrees, compared to almost 200 in the real world. In the act of moving to free himself, the user may touch some other objects and be stopped again. Clunk mode is impractical for many situations; even a virtual living room can become unmanageable.

Slip Mode

Upon collision, the user’s movement is deflected, effectively making him slide around the object he ran into. Slip mode appears to be the best compromise for “realistic” situations, preserving the solidity of objects while allowing comfortable movement. Slip mode imitates the overall effect of the reactive behaviors people normally use to get past obstacles. Although Slip mode may be the most “natural” solution to collision handling it is also the most complex, involving multiple computations and redirection of motion in ways that may sometimes be inconsistent with physics, the user’s expectations, or both. Figure 2 shows our implementation of Slip mode.

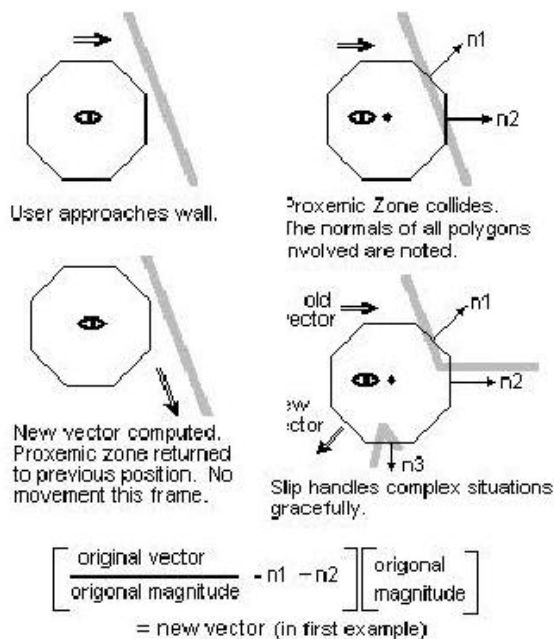


Figure 2. Implementation of Slip mode

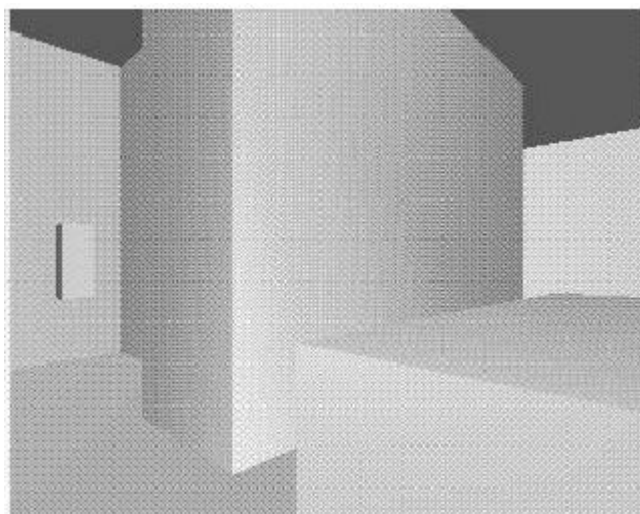


Figure 3. Baffles Maze – floating cube is a treasure

These three approaches to collision handling were tested [Jacobson and Lewis, 1997a,b] in a complex VE, shown in Figure 3, using a “treasure hunt” task in which participants explored to find as many “treasures” as possible in an experimental session. Because of ceiling effects for participants who found all the treasures yet continued to search, we have used inter-treasure search time as a primary performance measure. Figure 4 shows the performance of the three groups which were found significantly different (p < .01) using Analysis of Variance. As our results indicate, the

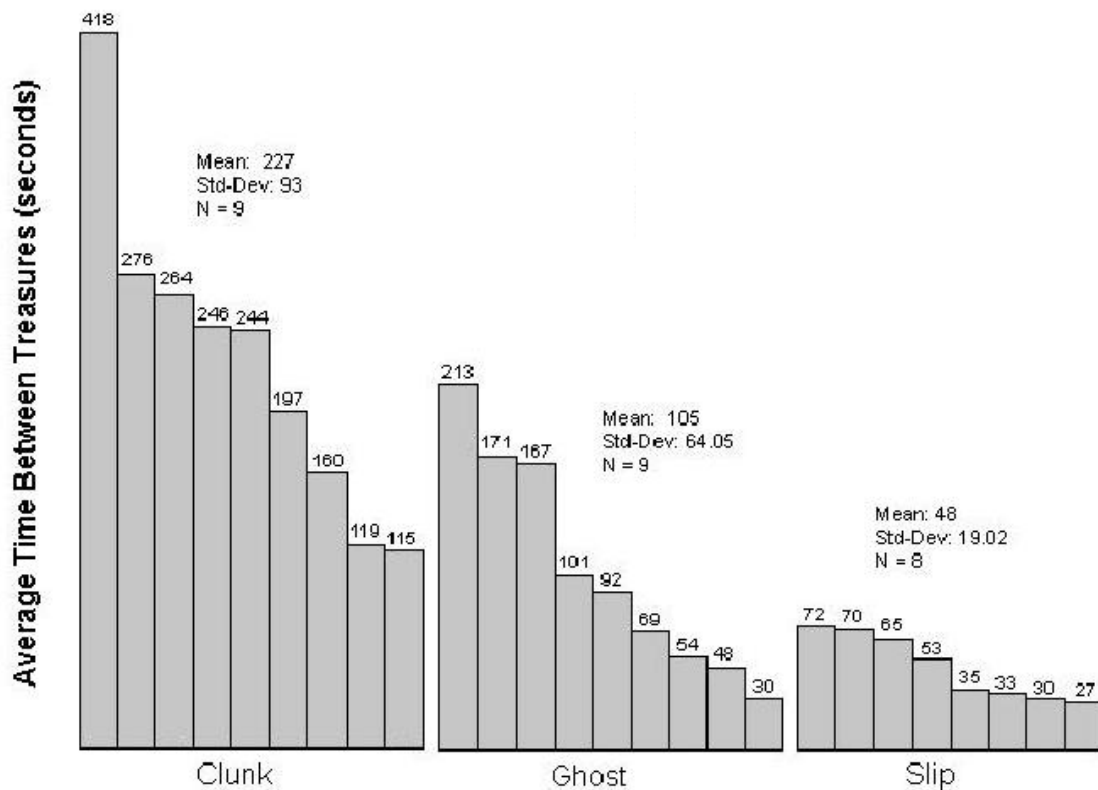


Figure 4. Inter-Treasure Seek Times

collision handling scheme most consonant with user intent was superior to both simpler and more physically accurate schemes.

3 Path Planning

Typically, human decision-makers, particularly military commanders, face time pressures and an environment where changes may occur in the task, division of labor, and allocation of resources. Information such as terrain characteristics, location and capabilities of enemy forces, direct objectives and doctrinal constraints must all play a part in the commander's decisions. Software agents have privileged access to the masses of information in the digital infosphere and can plan, criticize, and predict consequences from this sea of information with greater accuracy and finer granularity than a human commander could. There is also, however, information that may not be explicitly represented electronically and is therefore inaccessible to software agents. Such information includes intangible or multiple objectives involving morale, the political impact of actions (or inaction), intangible constraints, and the symbolic importance of different actions or objectives. Before agents can consider information that is outside their infosphere, this information must be

re-expressed in agent-accessible terms.. Military commanders, like other professional decision-makers, have vast experiential information that is not easily quantifiable. Commanders must deal with idiosyncratic and situation-specific factors such as non-quantified information, complex or vaguely specified mission objectives and dynamically changing situations (e.g., incomplete/changing/ new information, obstacles, and enemy actions). In order to cooperate with software agents in planning tasks commanders must find ways to translate these intangible constraints into tangible ones their agents can understand. The issue therefore becomes how should software agents interact with human teams to assist with problems which may be vague, ill-specified, with multi-attribute goals.

3.1 MokSAF: A Team Planning Environment:

We have developed a computer-based simulation called MokSAF to evaluate how humans can interact and obtain assistance from agents within a team environment [Payne et al., 2000, Lewis et al., 2001]. MokSAF is a simplified version of a virtual battlefield simulation called ModSAF (modular semi-automated forces). MokSAF allows two or more commanders to

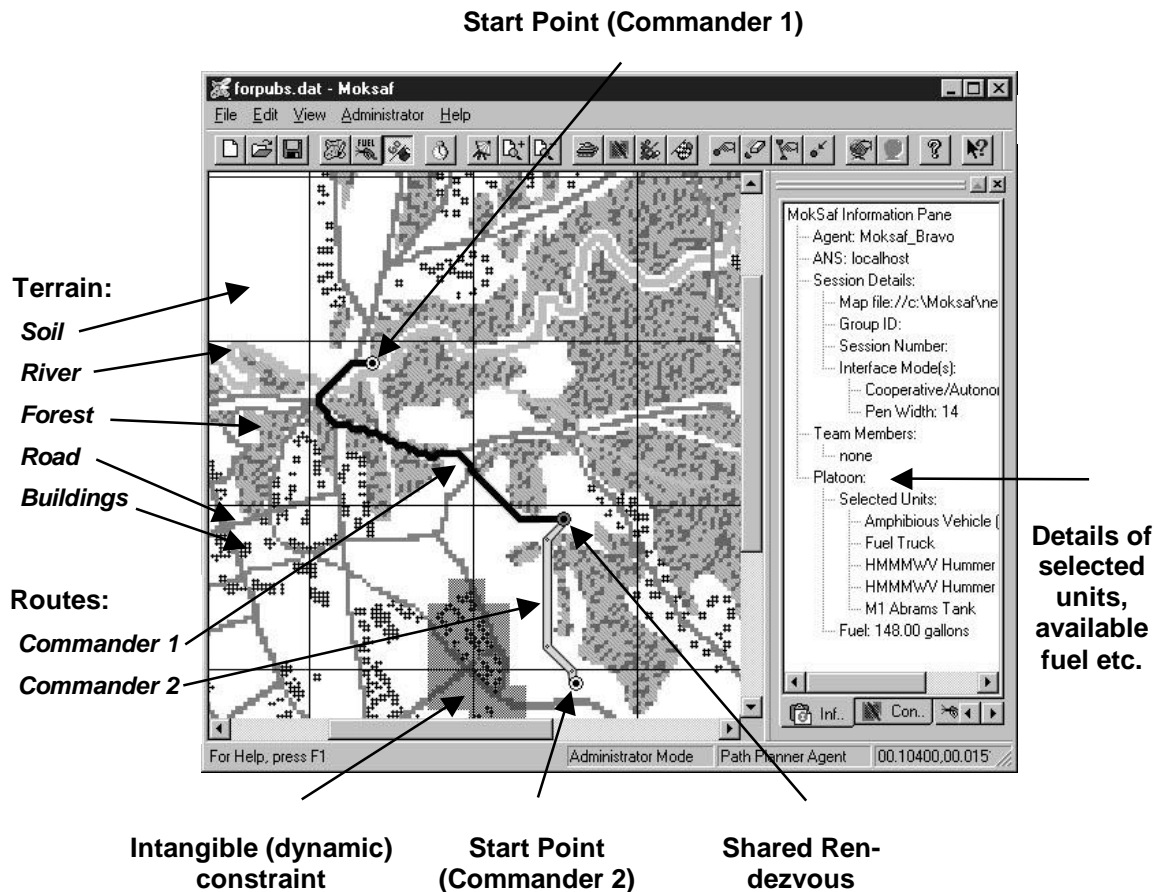


Figure 5. MokSAF Display

interact with one another to plan routes over a particular terrain. Each commander is tasked with planning a route from a starting point to a rendezvous point by a certain time. The individual commanders must then evaluate their plans from a team perspective and iteratively modify these plans until an acceptable team solution is developed.

One of the interface agents used within the *MokSAF* Environment is illustrated in Figure 5. This agent presents a terrain map, a toolbar, and details of the team plan. The terrains displayed on the map include soil (plain areas), roads (solid lines), freeways (thicker lines), buildings (black dots), rivers and forests. The rendezvous point is represented as a red circle and the start point as a yellow circle on the terrain map. As participants create routes with the help of a *route-planning agent* (see below), the routes are shown in bright green. The second route shown is from another *MokSAF* commander who has agreed to share his planned route. The partially transparent rectangles represent intangible constraints that the user has drawn on the terrain map. These indicate which areas should be avoided when determining a route.

3.2 Route-Planning Agents

Three different *route-planning agents* (RPA) have been developed to interact with the human team members in the planning task. The first agent, the *Autonomous RPA*, performs much of the task itself. This agent acts like a “black box.” The agent creates the route using its knowledge of the physical terrain and an artificial intelligence planning algorithm that seeks to find the shortest path. The agent is only aware of physical constraints, which are defined by the terrain map and the platoon composition, and intangible constraints, which are graphically specified by the commanders by drawing exclusionary regions on the map.

The second agent, the *Cooperative RPA*, analyzes routes through a corridor drawn by the human team members, selects the optimal route and helps them to refine their plans. In this mode, the human and agent work jointly to solve the

problem (e.g. plan a route to a rendezvous point). The workload should be distributed such that each component matched to its strengths. Thus, the commander, who has a privileged understanding of the intangible constraints and utilities associated with the mission, can direct the route around these constraints as desired. However, the commander may not have detailed knowledge about the terrain, and so the agent can indicate where the path is sub-optimal due to violations of local physical constraints such as traversing swamp or wooded areas.

The third condition, the *Naïve RPA* (or control), provides minimal assistance to the human commanders in their task of drawing and refining routes. Using this RPA, the commander draws a route that the agent then critiques for constraint violations such as impassible terrain or insufficient fuel. The commander is allowed to iteratively alter his failed route until a plan is found which passes muster. All three RPAs are intended to be used for iterative cooperative refinement of routes and the task of coordinating with other commanders requires continuous replanning as the team searches for its own best solution..

3.3 Experimental Methodology

The *MokSAF* experiments examine a deliberative, iterative and flexible planning task. There are three commanders (Alpha, Bravo and Charlie), each with a different starting point but the same rendezvous point. Each commander selects units for his/her platoon from a list of available units. This list currently contains M60A3 tanks, M109A2 artillery units, M1 Abrams tanks, AAV-7 amphibious assault vehicles, HMMWVs (i.e., hummers), ambulances, combat engineer units, fuel trucks and dismounted infantry. With the help of an RPA, each commander plans a route from his starting point to the rendezvous point for the specified forces.

Once a commander is satisfied with the individual plan, she can share it with the other commanders and resolve any conflicts. Conflicts could arise due to several issues including shared routes and/or resources or the inability of a commander to reach the rendezvous point at the specified time. The commanders also must coordinate regarding the number and types of vehicles they can take to the rendezvous because their mission specifies the number and composition of forces needed at the rendezvous point. Commanders were additionally instructed not to plan routes that took them on the same paths as any other commander which required them to coordinate routes to avoid shared paths.

Twenty five teams consisting of three-persons were recruited (10 teams used the *Autonomous RPA*, 10 teams used the *Cooperative RPA* and five used the *Naïve RPA*) from the University of Pittsburgh and Carnegie Mellon University communities.

3.4 Results

Data were examined from two critical points in the session – the time that individuals first shared their individual routes (first share) and at the end of the 15 minute session (final). Overall, we found that the two aided conditions, *Autonomous RPA* and *Cooperative RPA* achieved lower cost paths, earlier rendezvous, and lower fuel usage.

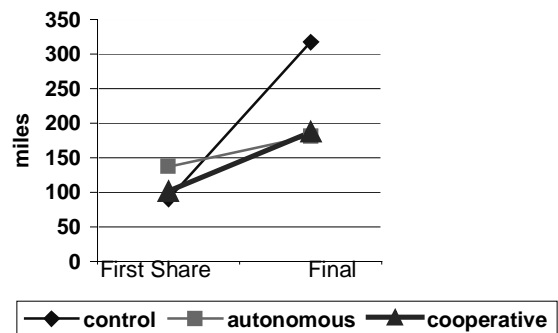


Figure 6. Path lengths were shorter for aided teams

These results held true both for the team as a whole and for individual participants. It was expected that path lengths between the first time a route was shared and at the end of a trial would vary due to issues related to conflict resolution among the teammates.

As shown in Figure 6, participants in the aided conditions managed to maintain the quality of their plans despite the modifications and replanning needed to coordinate with other team members. An ANOVA for total path length found a main effect for type of agent, $F(2,20)=67.975$, $p < .001$, (Naïve mean=552.5, $SD=41.45$; Autonomous mean=282.6, $SD=52.06$; Cooperative mean=260.22, $SD=31.25$) and post hoc tests using Tukey's HSD statistic showed both the Cooperative and Autonomous groups differed significantly ($p < .001$) from the Naïve control condition.

A main effect was also found for total route times, $F(2,20)=3.519$, $p = .049$, (Naïve mean=2617, $SD=358$; Autonomous mean=2170, $SD=275$; Cooperative mean=2192, $SD=215$). As figure 7 shows the conditions retain a Cooperative, Autonomous, Naïve ordering although post hoc tests show the only significant difference ($p=.04$) to be between the Cooperative and Naïve agents.

Although the RPA agents did not support teamwork directly, their assistance for the individual planning task allowed the commanders to find new routes as short as the ones they abandoned. Unaided commanders by contrast were forced to resort to longer paths in order to accommodate the requirements of coordinating with their team. Teams participated in three sessions; the first session was training. The second session involved a more challenging task to correctly find an appropriate route from the starting point to the rendezvous point for all three commanders.

They appeared to spend most of their time on this individual task and very little time on the team task of coordinating the selection of vehicles and meeting at the rendezvous point. On this more difficult Session 2 task, teams using the *Cooperative RPA* most closely approximated reference performance on the interdependent team

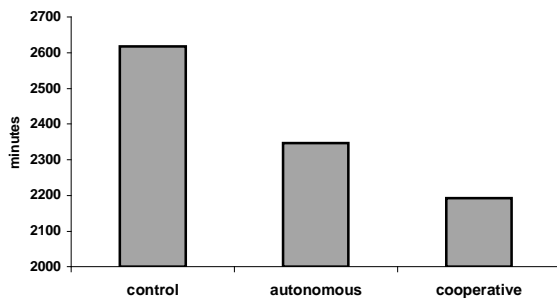


Figure 7. Travel times were reduced for aided commanders esp. Cooperative RPA users

task of selecting units. While the main effect for vehicle selection (sum of over and under represented unit types) only approaches significance, $F(2,20)=3.078$, $p=.068$; (Naïve mean=12.75, $SD=6.99$; Autonomous mean=9.1, $SD=2.56$; Cooperative mean=6.78, $SD=3.87$) as figure 12 shows the difference appears substantial.

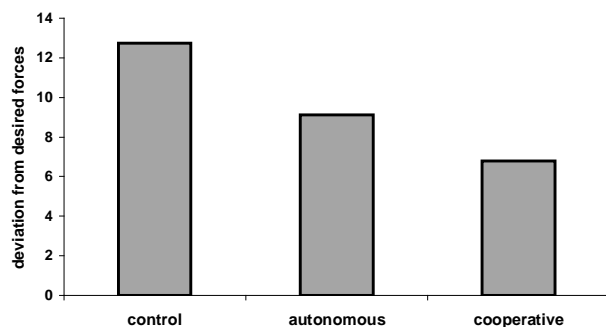


Figure 8. Agent help with route planning freed resources for vehicle selection

3.5 Discussion

The aided conditions, *Autonomous RPA* and *Cooperative RPA* were shown to provide a better interface for both individual route planning and team-based re-planning. Despite this clear superiority over the unaided condition (*Naïve RPA*), participants in the *Autonomous RPA* group frequently expressed frustration with the indirection required to arrange constraints in the ways needed to steer the agent's behavior

and often remarked that they wished they could “just draw the route by hand”.

Comments on the *Naïve RPA* focused more closely on the minutiae of interaction. In its current form, the user “draws” a route on the *interface agent* by specifying a sequence of points at the resolution of the terrain database. To do this, the user clicks to specify an initial or intermediate point in the path and then clicks again at a second point. A sequence of points is then drawn in a straight line between these locations. A route is built up incrementally by piecing together a long sequence of such segments. Although tools are provided for deleting unwanted points and moving control points, the process of manually constructing a long route is both tedious and error prone. While the *Cooperative RPA* automatically avoids local obstacles such as trees and closely follows curves in roads due to their less costly terrain weights, a user constructing a manual route is constantly fighting unseen obstacles which void her path or line segments which stray a point or two off a road into high penalty terrain. The anticipated advantages of heuristic planning and cooperation among human users were largely lost due to the necessity of focusing on local rather than global features of routes in the *Naïve RPA* condition. Rather than zooming in and out on the map to see the start and rendezvous points before beginning to draw, our subjects were forced to work from the first at the highest magnification in order to draw locally correct segments. The resulting problems of maintaining appropriate directions across scrolling segments of a map are not dissimilar to hiking with a compass. Although you can generally move in approximately the right direction you are unable to take advantage of features of the terrain you might exploit if a more global view were available.

While the Naïve RPA gratuitously forced the human to deal with physical constraints to which it already had direct access, the Autonomous RPA also diverted the user from the conceptual task of choosing and coordinating routes to that of representing intangible constraints. We believe this conceptual incongruity between the route planning and representation tasks left Autonomous RPA subjects unprimed for route planning related communications and coordination. Although the quality (path length/fuel usage) of routes between these two groups was very similar Cooperative RPA teams were better able to coordinate the composition of their forces and deal with deconflicting routes. These results emphasize the importance of designing human-agent interactions that promote direct interaction with the problem domain rather than focusing on information needs of the automation.

4 USAR Teleoperation

We have encountered a very similar problem in developing robots and simulations for urban search and rescue. Large-

scale coordination tasks in hazardous, uncertain, and time stressed environments are becoming increasingly important for fire, rescue, and military operations. Substituting robots for people in the most dangerous activities could greatly reduce the risk to human life and even allow new and more hazardous tasks to be undertaken. Because such emergencies are relatively rare and demand full focus on the immediate problems there is little opportunity to insert and experiment with robotic assistants.

The National Institute of Standards' Reference Test Facility for Autonomous Mobile Robots for Urban Search and Rescue [Jacoff, et al., 2001] is an attempt to replicate the challenges of such environments in a safe and reproducible way. The NIST USAR Test Facility is a standardized disaster environment consisting of three scenarios: Yellow, Orange, and Red physical arenas of progressing difficulty shown in Figure 9. The USAR task focuses on robot behaviors, and physical interaction with standardized but disorderly rubble filled environments. We have begun research on this problem [Lewis et al., 2003a] using real robots in a replica of the Orange Arena and in a simulation [Lewis et al, 2003b] of the same environment.



Figure 9. Orange (near) and Yellow Arenas-NIST photograph

The Orange Arena presents challenges to both perception and locomotion. It is constructed in two levels separated by difficult to navigate stairs and a ramp. Some of the floor is littered with paper while another area is strewn with dowels and small sections of pipe. Walls of some of the rooms are painted in Op art like stripes and patterns to confuse image processing and venetian blinds are used as apparent obstacles. A negative obstacle (drop off) is introduced on the platform in the form of an open HVAC shaft. (Negative obstacles present a significant problem in robotics because they are often not apparent from an image and are more difficult to sense than 'positive' obstacles that reflect signals.) Successfully navigating the Orange Arena requires both reasonably robust sensing and locomotion able to handle stairs and some surface irregularities.

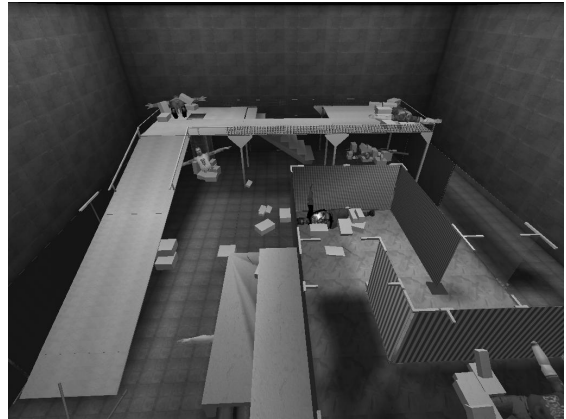


Figure 10. Simulation of Orange Arena

As a first step toward developing robotic teams we have been experimenting with simple teleoperated robots in the real and simulated orange arenas. Teleoperation has conventionally involved either direct control (similar to radio controlled cars) or mediated control through commands where the robot is given an intermediate waypoint as a goal to be reached. In developed applications, the robot frequently has a "safeguarded" interface. Safeguarding is a form of mixed initiative control in which the robot responds to control inputs as long as it can do so "safely". When its sensors detect a hazard such as a negative obstacle that might cause damage, the robot halts and informs its controller of the impasse. To proceed, the controller must explicitly override the safeguard.

So far we have tested unsafeguarded robots controlled through direct teleoperation and mediated command interfaces in the real arena and the simulation. Figure 11 shows the egocentric (through the robot's camera) control interface used for mediated robot control and the mediated and teleoperated robot simulations. The artificial horizon window at the top shows the robot's attitude (pitch and roll) while the camera icon to its right shows the camera position (pan and tilt). The teleoperated real robot was radio controlled using the operator's own exocentric view and did not relay any sensor information. The teleoperated simulation used a joystick for control. The mediated controller interface for both the simulation and the real robot consisted of concentric circles with the robot's initial position at its center. To move the robot, the operator clicked within the circles. The angular location of the selected point determined the robot's (terminal) bearing and its distance from the center the duration of the movement.

The three egocentrically (camera) controlled robots were extremely difficult to operate with many operators unable to move them beyond the small region of the arena where they started. Several things contributed to this difficulty including a narrow 40-degree field of view and difficulties in co-

ordinating camera movements with navigation. Radio controlled operation was somewhat easier because of a field of view almost five times wider and an unambiguous view of the robot's pose. All operators, however, had an extremely hard time maneuvering robots through the rubble. Time and again robots would get hung-up on invisible obstacles. When the operators tried to back them out they would run into something else. When they cleared the blocks that were stopping them they ran up on piles of pipes and rocked back and forth spinning their wheels sometimes for minutes before breaking free and returning to the floor.

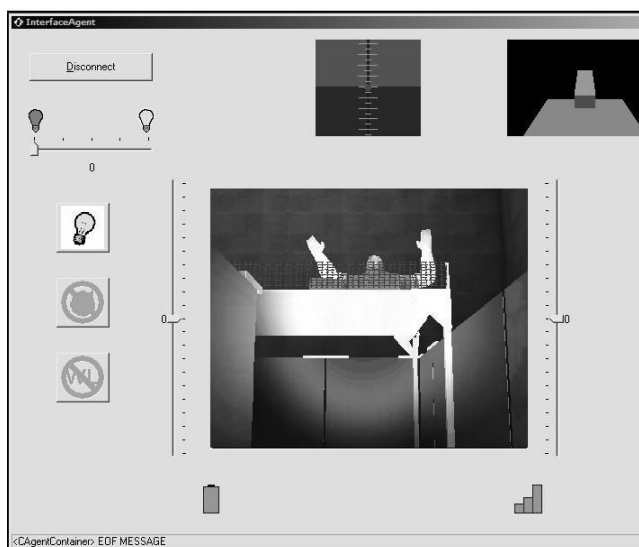


Figure 11. USAR Robot Control Interface

The USAR environment with its many obstacles to locomotion appears qualitatively different from the office and outdoor environments that make up most of mobile robotics experience. In the cluttered USAR environment safeguarding is almost moot. If the robot were safeguarded for collisions it would stay in an almost constant state of alarm and its operator would need to remain continuously in override mode to make any progress. It does, however, seem to be an excellent domain for the sort localized obstacle avoidance/resolution strategy that worked so well in virtual environments and path planning. We hope to implement and test this scheme in the USAR simulation as soon as adequate sensor models are developed.

5 Summary

In all three examples implicit control appeared superior to control based on explicit dialog, division of tasks or simplicity of interaction. The conditions making this possible were the human's (1) unambiguous expression of intent (2) through interactions with the domain (3) with highly restricted autonomous subtasks. While all three of our examples involved spatial movement we believe implicit control

can be effective for any task for which these conditions hold. In other studies, for example, we have found implicit control effective for automating message passing between radar operators [Sycara et al., 1998] and anticipatory information retrieval [Sycara and Lewis 2002]. While explicit control and delegation may be necessary for many human-agent interactions we may get the best cooperation when we can do without it.

Acknowledgments

This research was supported by NSF grant NSF-ITR-0205526, ONR grant N-00014-96-1-1222, and the Link Foundation. The collision handling experiments were conducted as part of Jeffrey Jacobson's thesis and the MokSAF experiments to which Terry Payne made major contributions as part of Terri Lenox's dissertation. Illah Nourbakhsh is a co-PI on the USAR research to which our students Jijun Wang, Mary Berna, Alexander Gutierrez, Terence Keegan, Kevin Oishi, Binoy Shah, Steven Shamlan, Mark Yong, and Josh Young have made significant contributions.

References

- [Jacoff et al., 2001] A. Jacoff, E. Messina, J. Evans, J. Experiences in deploying test arenas for autonomous mobile robots, Proceedings of the 2001 Performance Metrics for Intelligent Systems (PerMIS) Workshop, Mexico City, Mexico.
- [Jacobson and Lewis, 1997a] J. Jacobson and M. Lewis. An Experimental comparison of three methods for collision handling in virtual environments, *Proceedings of the 41st Annual Meeting of the Human Factors and Ergonomics Society*, Sept. 22-26, Albuquerque, NM
- [Jacobson and Lewis, 1997b] J. Jacobson and M. Lewis. Collision handling in virtual environments; facilitating natural user motion, *Proceeding of the 1997 IEEE International Conference on Systems, Man, and Cybernetics*, Oct. 12-15, Orlando, FL
- [Lewis et al. 2003a] M. Lewis, M. Berna, K. Sycara, K., and I. Nourbakhsh Robots, Agents, and People. IEEE Workshop on Safety, Security and Rescue Robotics, Tampa, FL, Feb 19-20.
- [Lewis et al., 2003b] M. Lewis, K. Sycara, K., and I. Nourbakhsh, I. Developing a Testbed for Studying Human-robot Interaction in Urban Search and Rescue *Proceedings of the 10th International Conference on Human Computer Interaction (HCI'03)*, Crete, Greece, June 22-27.

[Lewis et al., 2001] M. Lewis, T. Lenox, T. Payne, and K. Sycara, K. Spatial planning in teams of human and machine agents. Proceedings of the 2nd Conference on Information Technology and Spatial Planning, Isole Tremiti, It.

[Payne et al., 2000] T. Payne, K. Sycara, M. Lewis, T. Lenox, and S. Hahn, S. Varying the user interaction within multi-agent systems *Autonomous Agents 2000*.

[Sperger, 1996] Michael Sperger, Michael. Virtual Ancient Egypt, *SIMLAB Web Pages*,
<http://topaz.rec.ri.cmu.edu/files/egypt/>.

[Sycara and Lewis 2002] K. Sycara and M. Lewis .Integrating agents into human teams. *Proceedings of the Human Factors and Ergonomics Society's 46th Annual Meeting*, Baltimore MD.

[Sycara et al., 1998] K. Sycara, M. Lewis, T. Lenox, and L. Roberts, L. Calibrating trust to integrate intelligent agents into human teams, in *Proceeding of the 31st Annual Hawaii International Conference on System Sciences*, Jan 5-9,

[Turvey and Kugler, 1987] M.T.Turvey. and P.N. Kugler, N *Information, Natural Law, and the Self-assembly of Rhythmic Movement*, Lawrence EarlBaum Associates, 1987.