REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
12-09-2012	Final	June 15, 2010 - December 14, 2011

4. TITLE AND SUBTITLE
Process Integrated Mechanism for Human-Computer Collaboration and Coordination

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-10-1-0302

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
James Allen

5d. PROJECT NUMBER
F667100

5e. TASK NUMBER
TR0NL

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The Florida Institute for Human and Machine Cognition (IHMC)
40 South Alcaniz Street
Pensacola, FL 32502

8. PERFORMING ORGANIZATION REPORT NUMBER
Final

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
USAF, AFRL
AF Office of Scientific Research
875 N. Randolph St. Room 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
USAF, AFRL

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
ONR AFRL-OSR-VA-TR-2012-1071

12. DISTRIBUTION / AVAILABILITY STATEMENT
A - Unlimited
Distribution A: Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This project explored a novel approach to coordination called a Process Integrated Mechanism (PIM) that overcomes many of the problems of prior approaches. The PIM approach enables a simplicity of programing by using a single coordinating authority while avoiding the structural difficulties that have traditionally led to rejection of this approach in realistic settings. The components in the PIM architecture are conceived as parts of a single mechanism, even when they are physically separated and operate asynchronously. A PIM is a mechanism integrated at the software level rather than by physical connection. It maintains a single unified world-view, and behavior is controlled by a single coordinating process.

In this report, we describe our results in several key areas: 1) development of a new runtime system that implements the PIM model efficiency, 2) testing the PIM model in UCAV coordination problems using a simulated environment, 3) extending the PIM model that enables human participation in a human-robotic agent team, and 4) developing a novel framework for mixed teams of humans, robots and software agent based on Co-active design.

15. SUBJECT TERMS
distributed coordination, human machine teams, PIMs

16. SECURITY CLASSIFICATION OF: U			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	None	15	Robert Bonneau
U	U	U			**19b. TELEPHONE NUMBER** *(include area code)* 703-696+9545

Process Integrated Mechanism for Human-Computer Collaboration and Coordination

Final Report: AFOSR Award FA9550-10-1-0302

James F. Allen, PI   Florida Institute for Human and Machine Cognition

Background

Many military and civilian scenarios, including combat operations, surveillance, reconnaissance and search and rescue can benefit significantly from coordinated teams of humans, robots, and other computational devices that together to solve the problem at hand. Effective coordination mechanisms can greatly improve safety, robustness, quality and efficiency of the overall human-machine operation, but designing and building systems of humans and computers that coordinate well is a challenging problem, both in terms of the complexity of developing and verifying the algorithms as well as the runtime efficiency.

This project explored a novel approach to coordination called a Process Integrated Mechanism (PIM) that overcomes many of the problems of prior approaches. The PIM approach enables a simplicity of programing by using a single coordinating authority while avoiding the structural difficulties that have traditionally led to rejection of this approach in realistic settings. The components in the PIM architecture are conceived as parts of a single mechanism, even when they are physically separated and operate asynchronously. A PIM is a mechanism integrated at the *software level* rather than by physical connection. It maintains a single unified world-view, and behavior is controlled by a single coordinating process.

In this report, we briefly describe our results in several key areas: 1) development of a new runtime system that implements the PIM model efficiency, 2) testing the PIM model in UCAV coordination problems using a simulated environment, 3) extending the PIM model that enables effective control of human-robotic agent teams, and 4) developing a novel framework for mixed teams of humans, robots and software agent based on Co-active design.

Improvements to the PIM Runtime

The PIM Runtime supports the discovery of component parts of the PIM, the migration of the Coordinating Process (CP), and the interface to the underlying hardware on each part. During this period, we have extended the runtime to also support network sensitivity, which implies that the runtime now handles migration of the CP without requiring complete, direct connectivity between all nodes in the PIM. Therefore, the runtime now supports ad-hoc network topologies transparently. We have also added a fault-tolerance mechanism, which allows the PIM runtime to recover when the CP is lost when the node executing the CP fails.

Autonomous UCAV Coordination in Dynamic Search and Destroy Missions

Military operations of the last decades are no longer conflicts of only two participants. Coalitions of countries are participating on one or both sides of the conflict. These conflicts are composed of a number of missions of different nature. This work focused on the search and destroy missions which form an irreplaceable part of most of the conflicts since the Vietnam War. With the introduction of Unmanned Combat Air Vehicles (UCAVs) the importance of these missions increases even more since it allows to further decrease the number of causalities among the allies. Using the PIM integration with the Tactical AgentFly simulation, we were able to explore two different approaches to UCAVs in search and destroy missions: a multi-agent negotiation approach, and Process Integrated Mechanism (PIM). We found that both approaches allowed a high degree of autonomy of the UCAVs, promising to decrease the operator load and the base–UCAV communication. We proposed several different quality metrics and used them to evaluate and compare both approaches. We also proposed an interesting strategy that used both approaches to create a coalition of autonomous UCAVs, taking advantage of the strengths of a multi-agent approach and PIM. These results are explored in depth in Toziˇka et al (2011).

Our most recent task has been the extension of the UAV coordination implementation to include an operator interface – a Graphical User Interface (GUI) that allows a human operator to monitor the existing set of UAVs being controlled by the PIM, as well as to assign high-priority areas for surveillance. The Operator GUI has been incorporated as just another type of node that is part of the PIM, thereby keeping with the overall philosophy of the PIM approach. More details on these efforts are provided in the following sections.

Integration of PIM and Tactical AgentFly Simulation Environment

In conjunction with our team member – the Czech Technical University in Prague, we have developed a simulation of a persistent surveillance scenario. The scenario utilizes a simulator previously developed by the Czech Technical University and involves a simulated Iraqi town with buildings and moving targets. We have also completed integration of the PIM runtime with the Persistent Surveillance simulation environment. The integration allows any number of nodes (with each node representing a UAV) to be configured in the PIM, with a corresponding number of entities being created in the simulation environment. At the PIM level, each node is a separate computer that is executing the PIM runtime. These nodes are connected via network links in the NOMADS testbed, allowing us to emulate various types of networks as well as communication effects such as packet loss (due to unreliability), changes in latency, and limits on capacity. Therefore, the simulation environment, combined with the NOMADS testbed, provides a reasonably high-fidelity testbed for the overall evaluation of the PIM. The testbed is also instrumented to measure network traffic. After completing the PIM testbed setup and integration, we began development of PIM-based UAV coordination algorithms. This included a set of primitive UAV navigation commands, as well as higher-level allocation and path planning. The current algorithm is flexible enough to accommodate a varying number of UAVs, to adapt to the dynamic addition or deletion of UAVs (e.g., when a UAV has to return to base to refuel), as well as assignment of high-priority areas.

The PIM Runtime can load external libraries and expose an API to the CP to provide control over the specific hardware of the device. Different devices can be integrated in PIM by implementing a library to be loaded by the Runtime, providing an API for the CP. For the integration of PIM in the Tactical AgentFly system we implemented the TAFLib library that provides the communication with TAF. The data received from the TAF server is collected in a data structure at the C++ level. When the CP visits the device, it will pull the data through the Runtime thereby enriching its knowledge of the world. The additional knowledge allows the CP to take the appropriate decisions to continue its tasks. The CP will again use the API provided by the TAFLib library to send new commands and flight plans for the UAVs to the TAF server.

Test scenarios

Several scenarios have been implemented to test and prove our effort.

Basic Surveillance

In this testcase the world is divided in rectangular areas of the same size, there is one area per UAV. Upon startup the UAVs fly to the beginning of the area (lower left corner) and fly upward until they reach the margin of the area. They then turn for a semi-circumference and continue flying downward. The turn radius is such that the field of view of the camera mounted on the UAV covers up to the border of the area the UAV has just flown over.

If the area is “thin” when the UAV reaches the lower margin it turns back to the original upward trajectory, otherwise it turns to the right and continues patrolling. When the UAV reaches the right margin of its area it will fly back to the starting point.

This scenario was implemented to support the dynamic addition and removal of UAVs. PIM seamlessly provides discovery of new PIM parts and is resilient to component loss. When the CP is notified of a new component (or of the loss of a component) the number of areas increases (or decreases) the size of the areas is recalculated and the UAVs are reassigned, possibly allocating them to the area they are currently flying over. A larger number of UAVs implies smaller area size and therefore a better coverage of the ground to patrol.

The pictures below show screenshots from a demo. The demo started with two UAVs (U28 and U29), the area to cover is split in two areas each assigned to a UAV. A third UAV (U21) is then added to the scenario, the area to patrol is divided in three subareas, U29 repositions itself inside the newly assigned area leaving the area to the right to the new UAV. The third picture show the patrolling after the area adjustment is completed.

Figure 1: Two UAVs are patrolling the world. Two areas are highlighted, each UAV is assigned to one area.

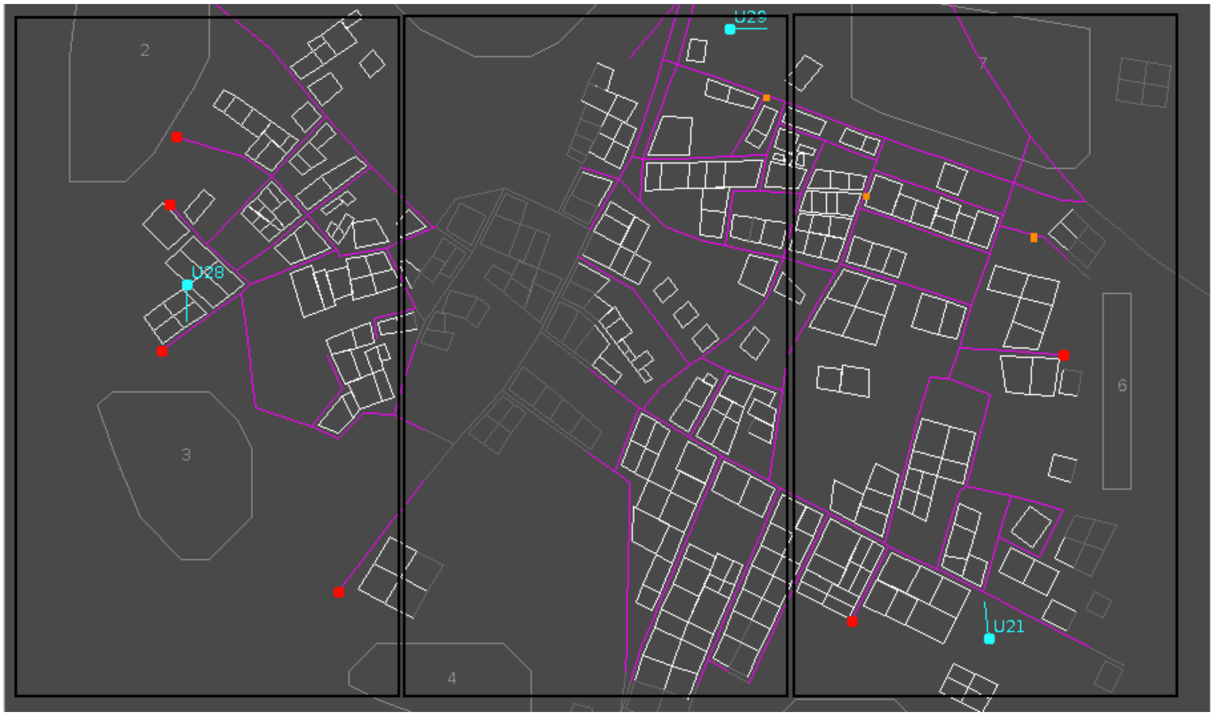

Figure 2: A third UAV joins the scouting team. A new area is created and the size of the areas and area assignments are recalculated. U29 is adjusting its position to patrol the newly assigned area.

Figure 3: The area adjustment is complete, the UAVs proceed patrolling their areas.

Shooting Enemies

The basic scenario proved the successful integration of PIM and the TAF simulation environment. Subsequently we improved the CP and tested the shooting capability.

The UAVs, or rather UCAVs (Unmanned Combat Air Vehicles) in this case, split the area to patrol among them and start scouting. When the camera sensor spots a target, the UCAV is relieved from patrolling the area and is tasked with tracking and eliminating the target. The target/enemy is shot after the UCAV has monitored it closely for 60 seconds (configurable time).

When a UCAV is tasked with tracking and shooting its area remains uncovered therefore the CP will recalculate the areas and perform new area assignments among the remaining patrolling UCAVs.

The UCAV waits after the target has been shot until it can positively recognize the enemy as destroyed (via additional tracking of the shooting area) the UCAV is then reassigned to area patrolling, the areas are recalculated and reassigned.

When a UCAV tasked with tracking and destroying recognizes another enemy in addition to the one it is currently tracking, it will request the intervention of another UCAV to track and destroy the new enemy.

This testcase was extensively tested and data was collected to compare the PIM approach with a multi-agent approach running natively on the Tactical AgentFly system. The main scenario for the comparison proceeded as follows:

1) There are four enemies on the ground

2) Four UCAVs are instantiated and they start patrolling the area

3) When a UCAV detects an enemy it starts tracking it and then shoots it (after the enemy has been in the field of view of the UCAV for 60 seconds)

Every minute (configurable) a new enemy appears on the map, ten enemies are created in total.

A similar scenario also tested the addition of supplementary UCAVs:

1) There are four enemies on the ground

2) Two UCAVs are instantiated and they start patrolling the area

3) When a UCAV detects an enemy it starts tracking it and then shoots it (after the enemy has been in the field of view of the UCAV for 60 seconds)

Every minute (configurable) a new enemy appears on the map, ten enemies are created in total.

After five minutes two additional UCAVs join the team.

The screenshots below show in 3D and in 2D the sequence of actions performed by a UAV that discovers a new target, than tracks it for some time, shoots and proceeds to scout the area for additional targets.

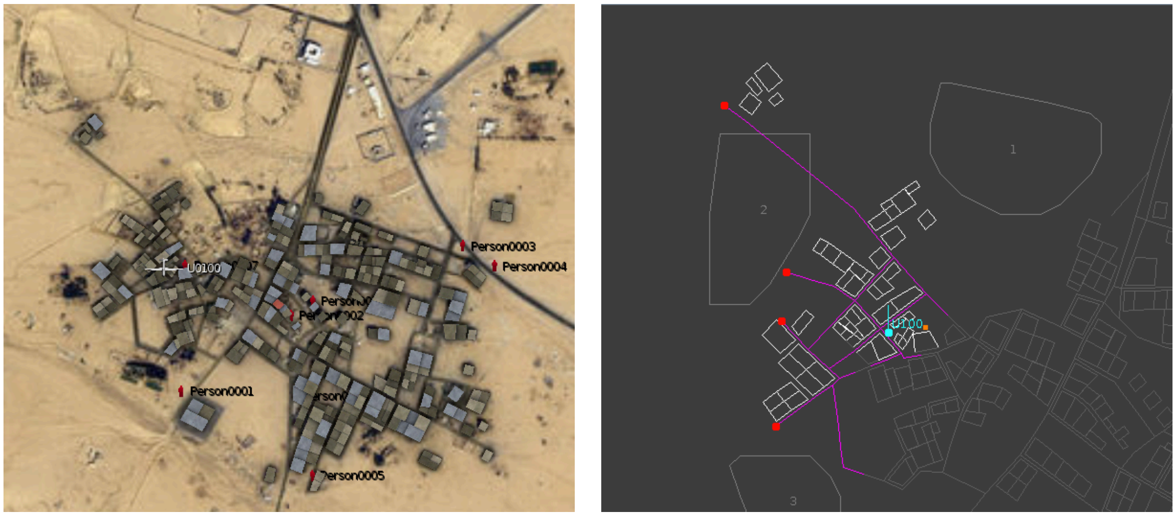

Figure 4: New target identified.

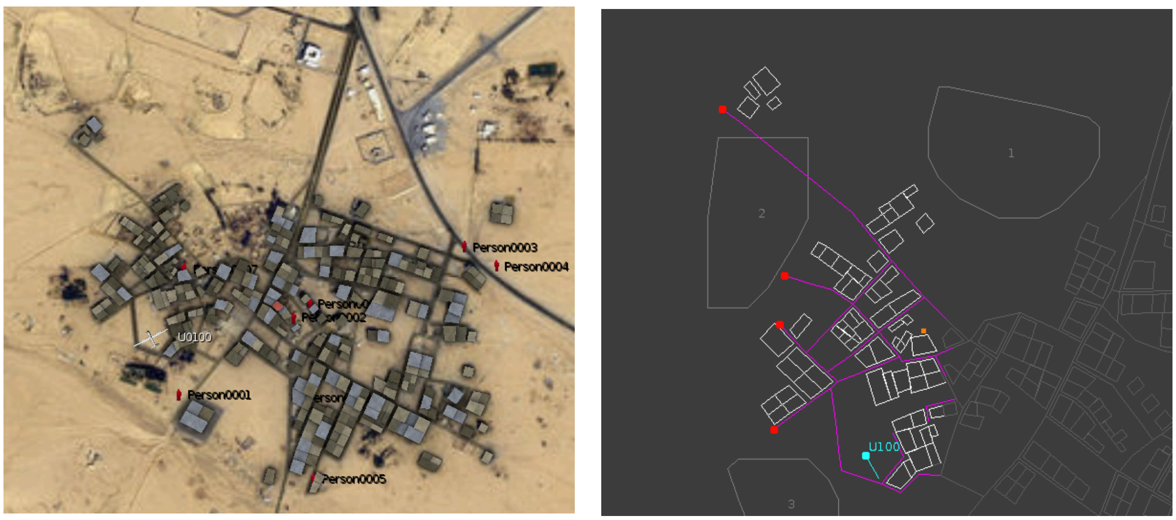

Figure 5: Fly to identified target.

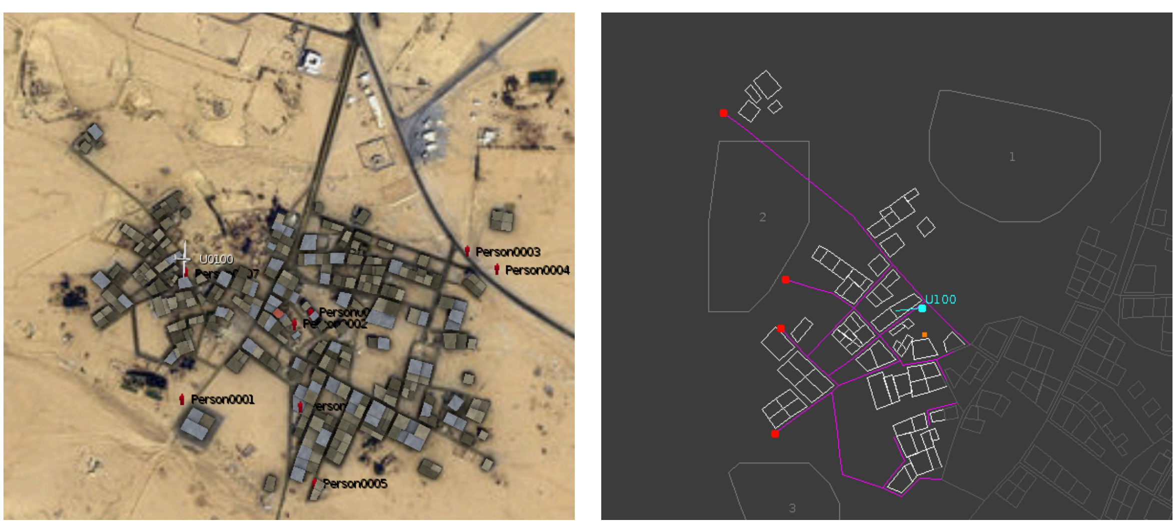

Figure 6: Tracking target.

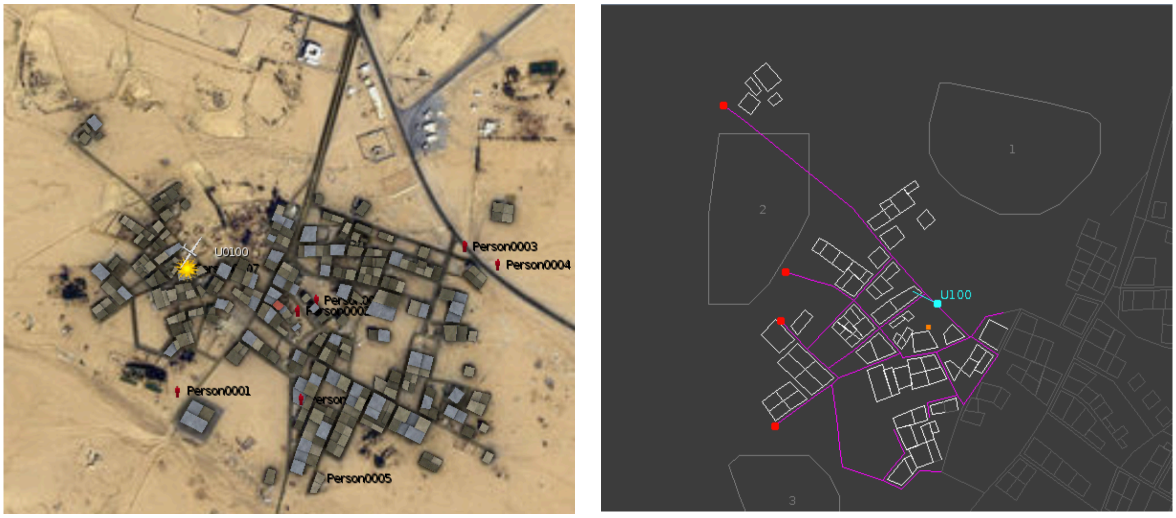

Figure 7: Missile launched.

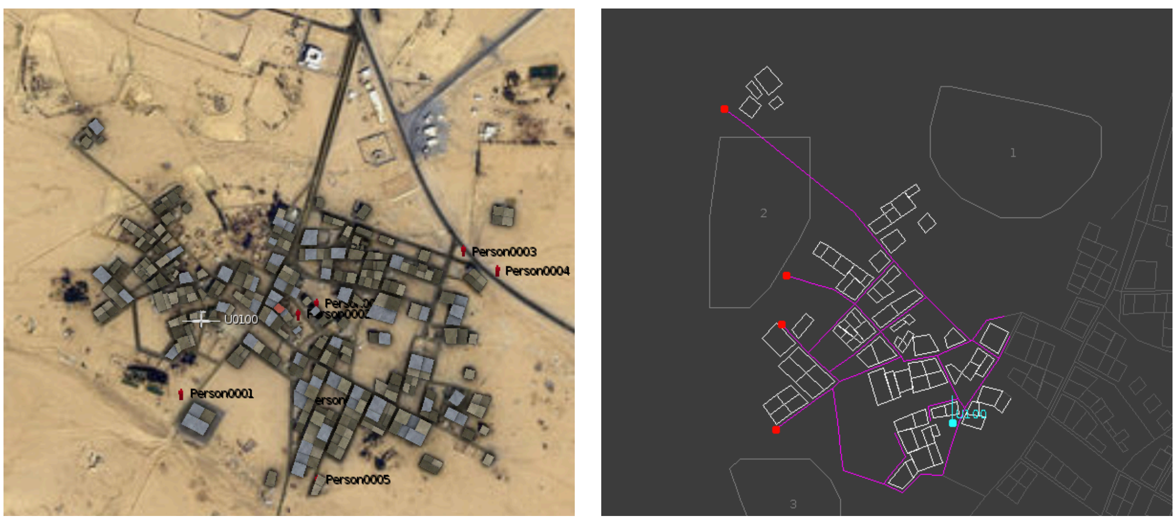

Figure 8: Target destroyed, continue scouting the area.

Two additional shooting testcases where created: Synchronous Shooting and Coordinated Fire.

Synchronous Shooting

In this scenario the UCAVs are performing a search and track mission. The area is split among the UCAVs as in the previous cases and when a UCAV discovers and enemy it is tasked to track it. When all the UCAVs are tracking an enemy (we assume there are at least as many enemies as UCAVs) they synchronously shoot the enemies, to achieve surprise. The efficient migration of the CP among the nodes allows the synchronous shooting to take effect with little to no delay.

Coordinated Fire

In the case of the coordinated fire we assume the UCAVs only carry one missile but more than one missile is needed to effectively destroy the target. The target could be for example a building identified with some intelligence as hosting hostiles. The CP proceeds to task the UCAVs to scout the area, when one of the UCAVs identifies the target, all the UCAVs (or as many as are

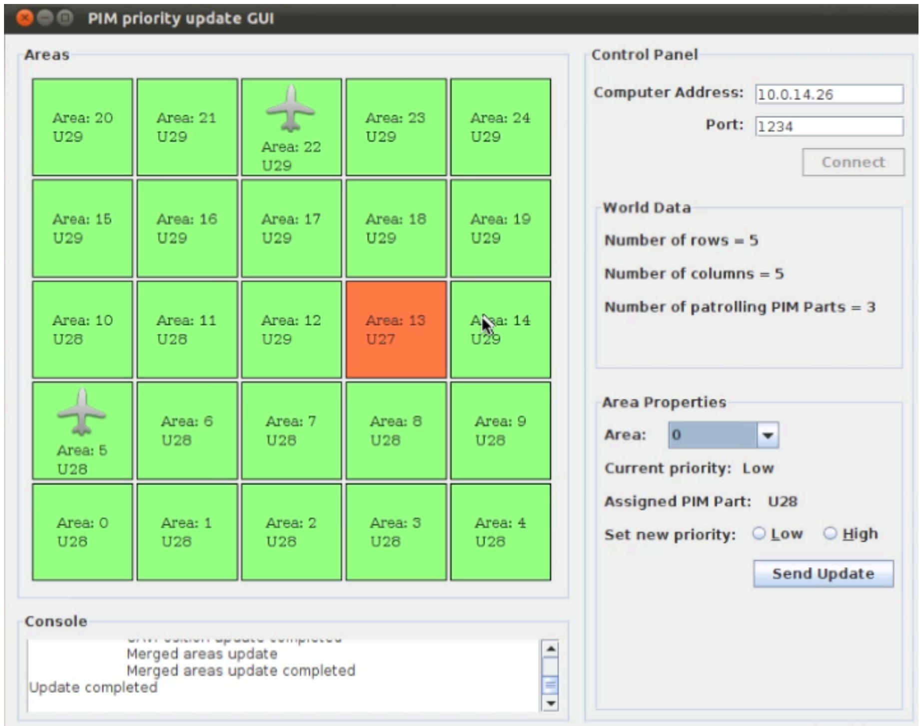

Figure 9. The PIM priority update GUI

needed depending on the size of the target) are tasked to fire in such a way that the missiles will reach the target at the same time no matter the current distance from the UCAVs to the target. Given that the time for the missile to reach the target depends on the distance from the UCAV to the target and the speed of the missile, and the speed of the missile cannot be changed, to achieve the goal of reaching the target at the same time the UCAV currently flying farther from the target should be the first to shoot. Subsequently all the other UCAVs will shoot at the appropriate time. PIM can effectively be employed to achieve the desired coordinated fire.

Dialogue-based interaction between humans and robotic agents in PIM models

In this part of the project we explored using the PIM architecture as the underlying framework for a coordination task involving both human and robotic agents. A human operator interacts with robotic agents via the TRIPS dialogue system. Both the TRIPS system and the robots are configured as components of the PIM. This set-up makes it possible for state information from the robots to be immediately available to the operator, and for the operator's commands to be immediately available to the robots.

The Graphical User Interface

A Graphical User Interface provides the human operator with situation awareness, including a detailed view of the region as shown in Figures 3-8. The human operator has an additional GUI at their disposal that shows the current area assignments and area priorities; it also allows the operator to update area priorities. Figure 9 shows a screenshot of this GUI during the execution of a surveillance mission. Three UAVs are assigned to the mission; two are patrolling their areas, while the third is on its way to Area 13, which had just been upgraded to high-priority status.

The GUI displays are provided by a dedicated PIM node with the sole purpose of facilitating communication to and from the human operator.

The TRIPS system

The TRIPS multi-modal is a general-purpose framework for developing speech- and text-based dialogue systems. Its core components include: (i) a toolkit for rapid development of language models for the Sphinx-3 speech recognition system, (ii) a robust parsing system that uses a broad coverage grammar and lexicon of spoken language, (iii) an interpretation manager (IM) that provides contextual interpretation based on the current discourse context, including reference resolution, ellipsis processing and the generation of intended speech act hypotheses, (iv) an ontology manager (OM) that translates between representations, and (v) generation manager (GM) and surface generator that generate system utterances from the domain-independent logical form.

Since the TRIPS system provides a spoken language interface, it is attached to the PIM node that also handles the graphical user interface. Thus, commands to the individual UAVs, as well as updates from them are available almost immediately, within one cycle of the Coordinating Process.

The dialogue interface

By using spoken commands, the human operator doesn't need to use the PIM GUI to change the area priorities (the GUI itself may still be useful for situational awareness, since aerial views don't show UAV assignments). In fact, the spoken interface allows for many more commands, including some that are difficult to implement in GUIs, but fairly easy to implement in sophisticated dialogue systems like TRIPS, which interpret the user's spoken command into a high-level logical representation of the meaning of the user's utterance.

Although the TRIPS system supports a more conversational, mixed-initiative interaction, due to the nature of the surveillance scenario, in this domain we have used a fairly conventional command and control dialogue strategy. Accordingly, to execute a mission, the operator would speak their commands, and the system would interpret them and convert them to one or more PIM-level actions. However, in contrast to most spoken language systems, the operator is not restricted to the use of specific wordings for the commands. Rather, they can speak freely, and the sophisticated natural language processing machinery in TRIPS tries to make sense of the command in terms of the current task domain. Language is, of course, a very efficient way of communicating complex content, and TRIPS can use inference to translate such content into lower level system actions. For the current stage of this project we have not implemented a domain-specific reasoner and the knowledge required for such a reasoner to work. Nevertheless, simple inference is available to ground linguistic expressions using conjunctions and/or quantifiers to sets of objects (UAVs or areas), and commands involving such complex expressions to sequences of actions, one for each referenced object.

Updates from the UAVs -- in particular information about what targets were found and where -- are communicated to the operator in natural language. For this instance of the system we used a simple template-based natural language generation technique. In addition to updates, the system will also generate acknowledgements, in accordance to its conversational policies.

To better illustrate the use of dialogue during the execution of the surveillance task, let us detail some of the control mechanisms and the way they can be achieved using dialogue:

Starting the mission: At start-up, the TRIPS node receives information about the UAVs available for the mission, as well as the number of areas that the region of interest is divided into. Each UAV is represented as a PIM node, and therefore the information about it, in particular its name/id, is available as soon as it connects to PIM.

Allocating/deallocating resources to the mission: Although multiple UAVs may be available, the human operator may choose to allocate all or only some of them to the mission. Also, during the execution of the surveillance task, the operator may decide that not all UAVs are needed (perhaps because of a lack of interesting targets to track), and therefore return some UAVs to the base. The operator may refer to UAVs by name, or by using quantifiers to specify (sub-)sets of UAVs. For example, *Start U29*, *Stop U29 and U27*, *Stop all UAVs*, or even *Start all the other UAVs* are all interpreted correctly. As explained above, when the operator refers to sets of UAVs by using conjunctions or (restricted) universal quantifiers, TRIPS will translate the command into a set of individual commands, one for each UAV in the set.

Updating area priorities: Area priorities can be set via a variety of commands. For example, this can be accomplished with a direct command such as *Set the priority for Area 12 to HIGH* or *Make Area 14 high-priority*. Because priority values are binary, a command such as *Change the priority for Area 12* will flip the priority back to “low”. Also, “low” and “high” are naturally ordered, which allows the use of commands such as *Increase the priority for Area 3* or *Lower the priority for Area 9*. Again, the operator may use conjunctions and quantifiers; for example, *Reset the priority for all areas* will set all areas’ priority to “low”. As explained above, the Coordinating Process will re-assign the UAVs automatically upon each change in the area priorities.

Stopping the mission: The operator may request the termination of the mission by saying, for example, something as direct as *Stop the mission* or something more indirect, such as *We’re done*. When the operator requests that the mission be stopped, TRIPS issues commands to all the UAVs still active to abort their current plan and return to the base.

Figure 10 shows a screenshot of the integrated TRIPS/PIM-TAF system. The PIM GUI is visible, including (i) the aerial view displaying the current features being scanned by the active UAVs, and locations of the discovered targets, and (ii) the priority display, showing area assignments, area locations of the active UAVs and the priority status of all areas (green means low-priority and orange is high-priority). At the bottom left, part of a TRIPS window showing the dialogue history is visible. It shows that the system had issued a notification: *People have been identified in Area 13 and Area 11*. The user had just assigned a new UAV to the mission, and the command to do so is visible in the bottom right part of the screen: *Start U27*. The UAV has already reached Area 2, which had just had its priority level increased.

A video of an end-to-end session is available at http://www.ihmc.us/groups/lgalescu/wiki/ca832/PIM2012.html.

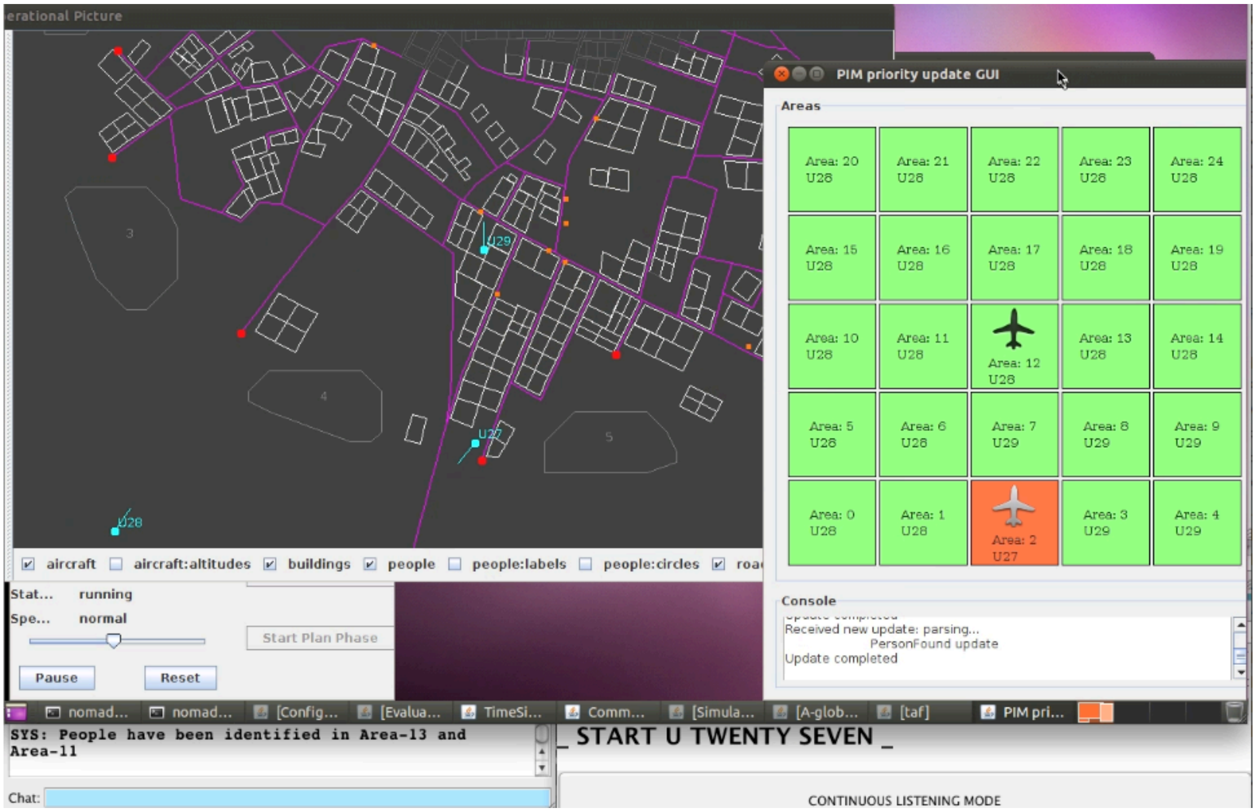

Figure 10. Screenshot of the integrated TRIPS/PIM-TAF system.

Addressing the human side of the equation through Coactive Design

As part of the PIM effort we performed a small study to help understanding interdependence and its impact on design of unmanned systems. The study highlighted that in complex joint activity involving mixed teams of humans, software agents, and robots, increases in autonomy may eventually lead to degradations in performance when the conditions that enable effective management of interdependence among the team members are neglected. The sophisticated robots envisioned for the future will be increasingly collaborative in nature, not merely doing things for people, but also working together with people and intelligent systems. Though continuing research is needed to make agents and robots more independent during times when unsupervised activity is desirable or necessary (i.e., autonomy), they must also be more capable of sophisticated interdependent joint activity when such is required (i.e., coactivity). Developing an understanding of the relationship between autonomy and interdependence is a first step toward this goal.

In this study, we explore how changes in autonomy can affect various dimensions of performance when interdependence is neglected. Although our experimental results stem from a simple task domain performed in a simulation environment, both our findings in the literature on human teamwork and our experience in a variety of human-agent-robot teamwork experiments and field exercises give us reason to believe that these results eventually can be generalized.

The Experiment

Our goal was to demonstrate that in human-agent-robot systems engaged in joint activity, increasing autonomy without addressing interdependence may lead to suboptimal performance.

Figure 1 (B) illustrates the general trends we expected to find in our results. We predicted that the highest level of autonomy would not demonstrate the highest level of team performance, consistent with the general shape of the notional bar graph shown in Figure 1 (C).

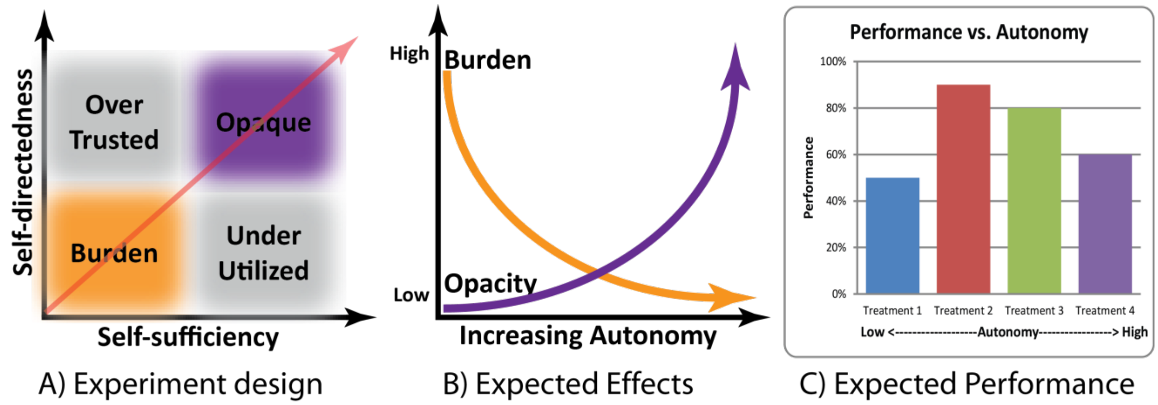

Figure 11 A) Illustration of our experimental design approach. B) Expected effects of increasing autonomy on the burden of managing the agent and the opacity of the agent to other task participants. C) Expected performance under treatment conditions of increasing autonomy, due to the competing factors of agent management burden and agent opacity.

Our domain was a simulation environment called Blocks World for Teams (BW4T). For this experiment, teams were composed of two players—a human and a software agent. The two players work toward the shared team goal, which is to deliver the colored blocks to the drop zone in a specified order.

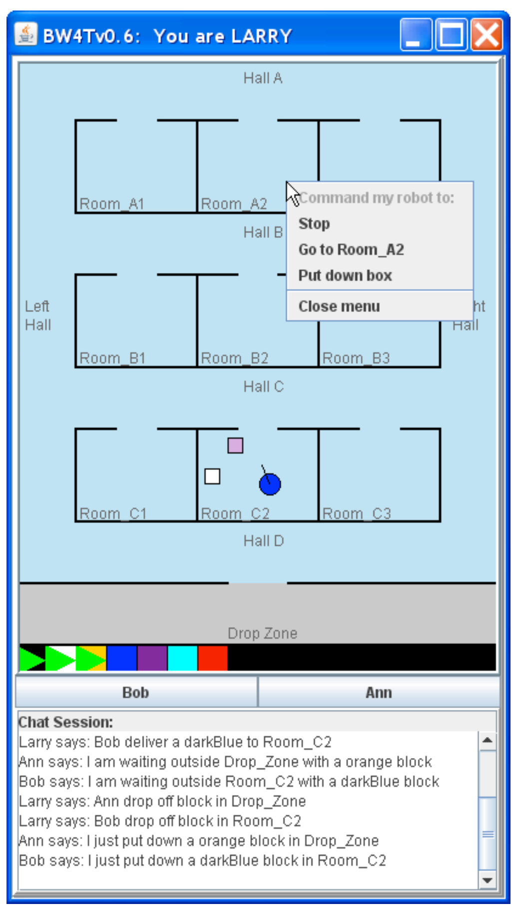

Figure 12 Example Blocks World for Teams (BW4T) interface

In order to compare the effects of changing autonomy, we defined different experimental conditions or "autonomy treatments." We intentionally left out any support for managing interdependence, except for communicating task completion status. By this means, we hoped to explore the relationship between autonomy and interdependence. We ran 24 participants through a series of trials and evaluated team burden, opacity, performance, and preference in each treatment.

The Results

Our results, shown in Figure 3 (A), indicate a very clear decrease in burden as autonomy increased. The results in Figure 3 (C) show opacity increasing with increasing autonomy as predicted. This validates the general expectation illustrated in Figure 1 (B).

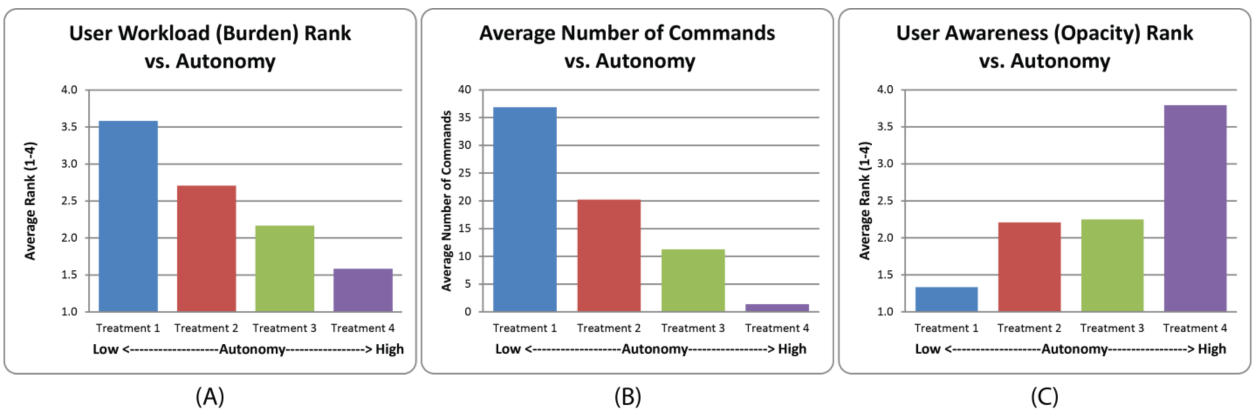

Figure 13 (A) Subject ranking of agent management workload (burden) as autonomy increases across experimental treatments. (B) Average number of commands (Burden) as autonomy increases. (C) Average subjective rankings of awareness (Opacity) as autonomy increases.

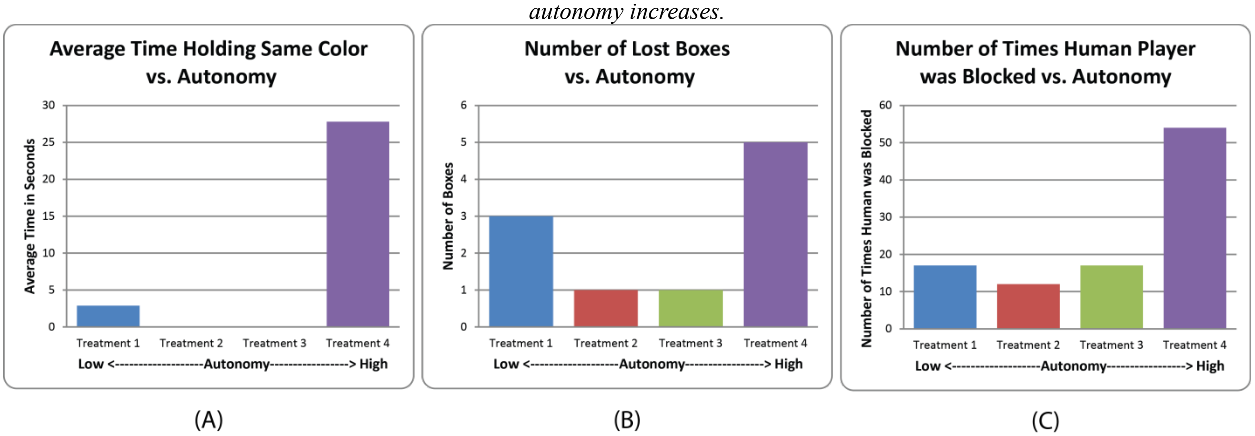

Figure 14 (A) Average time holding the same color (inefficiency) (B) Number of lost boxes (C) Number of times a human player was blocked by their agent partner while trying to enter a room

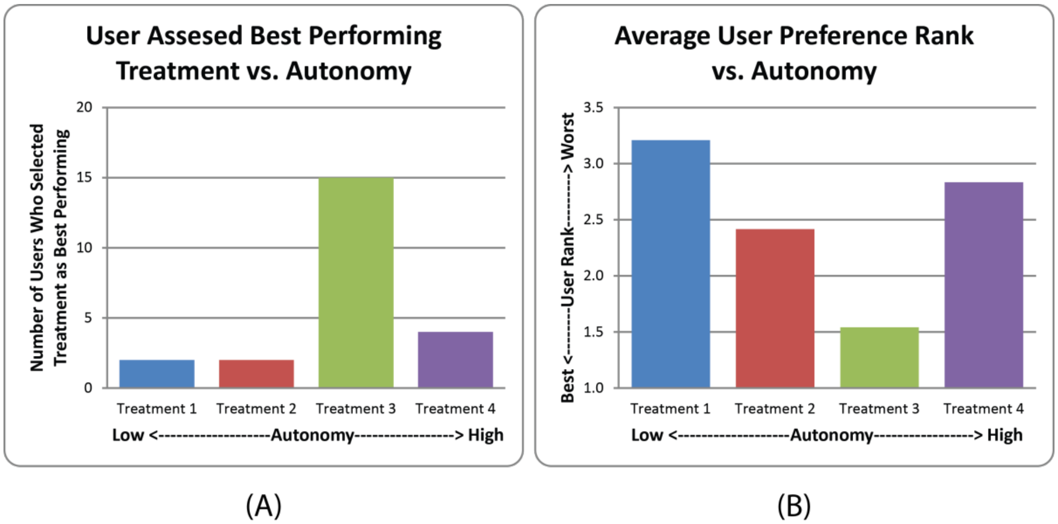

Figure 15 (A) User Assessment of Performance vs. Autonomy (B) User Preference vs. Autonomy

The results of our initial limited evaluation support our claim that increasing autonomy does not always improve performance of the human-machine system. In the BW4T domain, this was principally due to opacity in the system, derived from increasing autonomy without accounting for the interdependence of the actions and decisions of the players and the coordination challenges this creates. Additionally, we showed how keeping an agent busy does not equate to improved performance, how human error rates are not only due to workload but can also be affected by opacity (Figure 4), and how user preference is not necessarily driven by reduced burden when other factors such as transparency, predictability and directability are relevant to the task (Figure 5). A key point to take away is that the ability to work with others becomes increasingly important as interdependence in the joint activity grows. It is possible that in complex and uncertain domains, this may be more valuable than the ability to work independently.

Unmanned systems can be extremely useful tools, but we cannot simply offload tasks to unmanned systems without incurring some coordination penalty. Understanding how the unmanned system changes the nature of the task and addressing this relationship while designing the unmanned system is how we can help to ensure the advantages of unmanned systems are not outweighed by the incurred costs. Understanding the relationship of autonomy to interdependence is one step toward addressing the challenges facing future systems. We believe that consideration for interdependence while designing the autonomous capabilities of an agent can mitigate the effects demonstrated and will enable future systems to achieve greater potential. More details can be found in Johnson et al (2012).

Publications

Benvegnu, E , N. Suri, K. Ford, J. Allen, G. Finestrali (2011). PIM: A Unique Framework for Sensor Fusion in the Tactical Environment, MILCOM 2011.

Ford, K.M., J. Allen, N. Suri, P.J. Hayes, and R. Morris (2012) PIM - A Novel Architecture for Coordinating Behavior of Distributed Systems, *AI Magazine*, 31 (2).

Johnson, M., J. M. Bradshaw, et al. (2012). Analyzing Autonomy and its relation to Interdependence in Human-Machine Systems. *IEEE Intelligent Systems* Jan/Feb.

Sislák, D., Volf, P., Pechoucek, M., Suri, N., Nicholson, D., and Woodhouse, D. (to appear) Automated Conflict Resolution Using Probability Collectives Optimizer. *To Appear in IEEE Systems, Man, and Cybernetics*.

Tozicka, Jan, Erika Benvegnu, C. Stefanelli, N. Suri, K. Ford, M. Linderman (2012) Multiple-UAV Coordination and Communications in Tactical Edge Networks, *IEEE Communications Magazine*, October.

David Sislák, Michal Pechoucek, Niranjan Suri. (2012) Autonomous UCAV Coordination in Dynamic Search and Destroy Missions, In *Knowledge Systems for Coalition Operations,* 2012