Synergistic Integration of Agent Technologies for Military Simulation*

Paul Scerri, Sean Owens, Robin Glinton and Katia Sycara
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{pscerri, owens, rglinton, byu, katia}@cs.cmu.edu

Abstract

To perform large-scale coordination in real-world environments requires that many individually complex technologies come together to form integrated solutions. In this paper, we present an application where several key technologies are integrated into a unified system via a multiagent infrastructure. We show how the synergistic behavior among heterogeneous technologies results in a significant improvement over the performance of the individual technologies acting alone. Critical extensions were required to the language describing required behavior to allow the pieces to work together. Initial experimental results show system performance on a task of coordinating a military convoy in an adversarial environment was significantly improved when all technologies worked together. However, experiments with a human user in the loop showed that significant advances must still be made before such systems can be fielded in the real-world.

Introduction

Autonomous coordination has rapidly progressed over recent years, with significant advances in both specific algorithms(Modi *et al.* 2003; Jeffrey Cox & Bartold 2005) and complete approaches to coordination being successfully demonstrated(Goodrich *et al.* 2001; Schurr *et al.* 2005a). High bandwidth, widely available, low cost communcation and improvements in algorithm scalability are allowing significant advances in the number of agents able to be efficiently coordinated(Ortiz, Vincent, & Morisset 2005; Scerri *et al.* 2004). Moreover, advances in artificial intelligence(Nair, Tambe, & Marsella 2003) and robotics mean that the robots and agents being coordinated are significantly more individually intelligent than in most previous multiagent systems. This progress is quickly putting important, exciting applications of multiagent technology within reach. However, because there have been relatively few demonstrations of large scale, highly complex multi-agent systems, it is unclear whether existing automated coordination techniques are sufficient for these very complex systems or whether key challenges remain.

Many exciting multi-agent and multi-robot applications have been demonstrated in recent years. The RoboCup Initiative has shown some of the most dramatic progress, both within the soccer leagues(Stone, Balch, & Kraetszchmar 2001) and in the disaster response league(Kitano *et al.* 1999). Recently, SRI's Centibot project demonstrated effective behavior of a very large scale robot team(Ortiz, Vincent, & Morisset 2005). However, often these robot systems coordinate relatively homogeneous robots to execute reasonably simple plans. Other systems, including RETSINA-OOA(Giampapa & Sycara 2002) and the Electric Elves(Chalupsky *et al.* 2002), have demonstrated coordination of a more heterogeneous set of entities to achieve complex objectives. However, in many of these systems, much of the intelligence can be attributed to the coordination, with the individual agents being relatively simple, e.g., web services. Some military simulation environments involve both very complex individual technologies and large numbers of heterogeneous entities, but do not typically make extensive use of state-of-the-art automated coordination technology(Aircraft 1995). Thus, despite the obvious potential for automated coordination of large numbers of intelligent heterogeneous entities, there is a lack of sufficiently high fidelity systems that can expose key remaining challenges. Moreover, in most of these systems effective user control was not required, yet many real-world applications cannot be deployed without such control.

To better understand the challenges of complex coordination of intelligent entities, we have developed a simulation environment, called Sanjaya, where successful performance requires effectively dealing with these type of challenges. Sanjaya is a large scale military simulation where many realistic issues including uncertain sensing, diverse terrain, an intelligent adversary, large numbers of units and complex plans, are modeled. A sophisticated terrain analysis agent provides the team with expected locations of opposing forces and safe paths for travel. A sensor fusion process is available to take sensor readings from multiple sources and reduce uncertainty when distinguishing between opponent and civilian vehicles. Importantly, a human commander has high level control and makes decisions to strike identified opponents. Each of the system components is able to use input from the other components and provide input to those components. Intuitively, synergies should occur in Sanjaya, since the various agents should be able to mutually provide each other with input to improve performance. Thus, evi-

Acknowledgements: This research has been sponsored in part by AFRL/MNK Grant F08630-03-1-0005 and AFOSR Grant F49620-01-1-0542

dence of synergies would both show that the coordination was effective and provide support to Xu's result. Our results showed that such synergies were observed. When all of the technologies were brought to bear friendly asset survivability exceeded cases where subsets of, or none of the available technologies were used.

A variety of distinctly different approaches to coordination exist, developed to meet specific goals and have specific properties. Due to the dynamic nature of the fundamental task of moving and protecting convoys of ground vehicles, we choose to apply *teamwork* based coordination, since it is designed to be flexible and robust, key requirements for this domain(Cohen & Levesque 1991). The specific proxy-based implementation of teamwork that we use executes user designed *team oriented plans* instantiated at runtime from templates(Pynadath *et al.* 1999). While team oriented plans have been used extensively in the past, when attempting to apply them to this domain, we encountered two key problems, both related to the production and use of information during coordination. First, the plans did not have sufficient semantic content to allow the proxy to determine where information produced during plan execution should be directed. For example, when the terrain analysis agent generated a least resistance path, it could not determine what to do with the path. Second, some tasks within a plan could not be initiated until other tasks within the plan produced information to completely specify the task, but there was no way of specifying this. For example, a task for protecting a convoy could not be allocated until the path of the convoy was determined. Extensions adding additional semantics, but not changing the declarative nature of the TOP's were implemented.

While automated coordination and intelligent robot and agent technology is rapidly advancing, human intelligence is staying constant. Several previous investigations have shown that human interaction with intelligent distributed systems is extremely difficult, sometimes even leading to poorer performance than the system on its own(Schurr *et al.* 2005b; Kortenkamp, Schreckenghost, & Martin 2002; Goodrich *et al.* 2001). A key interface design issue is how much of the underlying intelligence to make opaque to the user and how much to make transparent. In Sanjaya, we took the approach that when the details of the intelligence reasoning were not critical to the human's reasoning, they were made completely opaque. We performed an initial set of user tests with some users being in control of a system with all the intelligent agents performing and another set of users using a less intelligent system with the same interface. Our results showed that users actually performed relatively better with a less intelligent system, indicating that significant advances need to be made before users can exert effective control over very complex multiagent systems.

Problem Description

Complex military operations are an instance of a general coordination problem that occurs in a variety of domains, including commerce, sports and homeland defense. Issues inherent to the coordination problem are well understood and include task and resource allocation, communicating in key situations, planning and dealing with failure(Cohen & Levesque 1991; Cockburn & Jennings 1996; Kinny 1993; Lesser *et al.* 1999).

In real-world domains various analysis tools or niche assets can potentially be brought to bear. For example, planning and execution of a real military operation will involve expert input on everything from terrain, to adversary tactics, to weather, to culture and may involve thousands of different types of physical systems including many humans in a broad range of roles. Information links in real-world systems such as these are by necessity in constant flux in an attempt to adapt to the uncertainty of the world. Specifically, it is impossible to know a-priori which part of the system the results of a certain analysis or a piece of information will be relevant to. Simply sharing all information by broadcast is not appropriate because at times irrelevant information will overwhelm a receiver by sheer volume or obscure the information that is relevant. Intelligent actors in real-world systems are deluged with noisy, uncertain information and must bring their expertise to bear to analyze this information and draw useful conclusions that facilitate system goals. This requires an actor to reason about what information is relevant to the analysis task, which colleagues to seek supporting information from, who to share analysis products with, and how to rectify conflicts when they occur. Each of these problems is challenging in isolation. The difficulty that these problems pose in the coordination of a large scale system is evident in failures during recent efforts to coordinate relief efforts after natural disasters. In the case of hurricane Katrina, which recently devastated the city of New Orleans, vehicles were available to evacuate citizens from the disaster area but officials were not aware of their existence. There are many similar examples in the military and disaster response domains of the confusion that can occur when attempting to coordinate at such a large scale and its serious consequences.

Sanjaya

Sanjaya is a constructive simulation supporting simulation of ground, air and unmanned aerial vehicles. The scenario being tested consists of five convoys of ground vehicles each supported by a group of unmanned aerial vehicles (UAVs), traveling across a 50km by 50km map defended by 20 opposing tanks. This is a challenging scenario for a number of reasons.

The UAVs have a limited sensing range, and there are not enough of them in the scenario to patrol the entire map. Obviously intelligent coordination is necessary to use the UAVs efficiently. When an opponent has been found the UAVs may be used to attack the oppoent, but in doing so the UAV destroys itself. As we have a limited number of UAVs in the scenario, each UAV used to attack further reduces our ability to sense, and to destroy any further opponents found.

Sanjaya simulates an uncertain sensing environment. Sensor readings return a list of possible classifications of sensed objects, with a probability associated with each classification. In the scenario being simulated, sixty trucks, representing civilian noncombatants, are placed on the map and may be confused with enemy forces. UAVs, are a limited resource and must be conserved to attack those entities that we are reasonably certain are actual opposing forces. This means that we must have a reasonably high confidence classification of a sensed entity as a military, non civilian entity before we issue a command to attack it.

Unlike other currently available simulation environments, Sanjaya simulates the kind of uncertainty and confusion that can occur in large scale coordination in real-world environments. We find this type of simulator necessary to study the challenges that large scale real-world coordination poses. Furthermore, this type of simulator is necessary to determine potential synergies that can occur while coordinating complex entities and technologies.

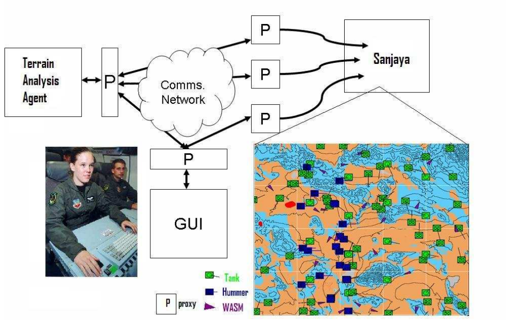

Figure 1: The system architecture. The Machinetta proxies provide the general purpose coordination infrastructure that connects the assets in the environment, the Commander and the terrain analysis agent.

Proxy based Integration

Coordination via teamwork has been shown to be applicable to a variety of interesting domains. Because of the reusability of teamwork, software modules encapsulating the key algorithms have been developed for reuse across domains. The *proxies* as they are commonly known, have become a standard mechanism for building heterogeneous teams(Jennings 1995; Tambe *et al.* 1999; Pynadath & Tambe 2002; Scerri *et al.* 2003). Each team member works closely with a single proxy that coordinates with the other proxies to implement the teamwork. Specifically, proxies manage team plans, which includes task allocation and reallocation, and coordinate information sharing. In addition, when the team-member the proxy represents is a human, proxies are equipped to reason about when to act autonomously, and when to defer to the team-member. The overall architecture is shown in Figure 1. The proxy communicates via a high level, domain specific protocol with the robot, agent or person it is representing in the team. This interface can vary in complexity from a simple communication layer when the RAP is a UAV to a complex Graphical User Interface that intelligently filters incoming information and interprets the user's intent when the RAP is a human Commander. Most of the proxy code is domain independent and can be readily used in a variety of domains requiring distributed control.

Team Oriented Plans

Team Oriented Plans (TOPs) are the abstraction that define required team behavior for proxies(Pynadath & Tambe 2002). The TOPs provide the mapping from team level goals to individual *roles* that are performed by individual team members. Roles are lowest level of abstraction in a team plan and are goals, activities or responsibilities that will most often be performed by a single team member. The TOP may describe various constraints between roles, including temporal constraints or AND constraints, which mean the plan should not proceed unless all roles are currently filled. Importantly, while TOPs describe the breakdown of team activities into individual roles, they do not describe what coordination is required to execute the plan. A TOP does not describe which team member will perform which role or exactly how the role should be performed. Role allocation is handled by the proxies. At any one time, the team may be simultaneously executing many team plans with allocation and reallocation of team members to tasks within the plans determined autonomously by the proxies.

Information Requirements and TOPs When we attempted to design TOPs to coordinate the individually very complex entities, two short-comings were revealed. First, there are insufficient semantics encapsulated in a TOP for a proxy to determine what information should be communicated to which other team mate. Particularly in the case of team members that might perform analysis for any member of the team, information must be added to the TOP to tell the analyst agent's proxy whom to send its analysis to. Second, some roles cannot be initiated or allocated until other team members have produced information as a part of their roles. For example, UAVs cannot be allocated to protect vulnerable locations on a convoy's path before a safe path has been computed and the vulnerable locations identified, but determining the safe path and identifying vulnerable locations is also part of the plan. Two separate extensions were made to the TOP specification language, with care taken to adhere to the basic principle that a TOP should provide a declarative specification of what should be done and the proxies be given the maximum latitude possible to execute it.

The first extension added to the TOP specification language was an optional parameter on each role called a *Directed Information Requirement*(DIR). The purpose of this addition was to give the proxies guidance in how to share specific types of information. These requirements each have the form $Send(X) \rightarrow [SpecificProxy|ProxyPerformingRole]$, where X is either a generic type of information or a specific piece of information. If a specific proxy is given on the right hand side of the requirement, information will be sent to that particular proxy. This form is used, for example, to require that while performing a particular role the agent keep the commander informed of certain events. On the other hand, if the $ProxyPerformingRole$ version is used, the proxy will route the information to the agent/proxy performing the specified role. This form of Directed Information Requirement is used, for example, to inform an agent performing a role that its team mate will require a particular type of information. The Directed Information Requirements are a powerful tool for specifying how information should be shared and were extensively used (see below.)

The second extension made to the TOP specification language for use on this application was to add an additional type of constraint on role execution. *Data Constraints* specify that a role should not be instantiated until a certain piece of data becomes available. These Data Constraints work like any other constraint on role execution and, thus, fit easily into the TOP framework. It is important to distinguish Data Constraints from simpler message passing or other fixed protocols. At the time the TOP is designed the agent producing the data for the Data Constraint and the agent that will be constrained are not known, these will be allocated at runtime. In fact, even when the data producing agent is producing the information, the eventual destination of that information will not be known because the role will not have been instantiated and thus, not allocated.

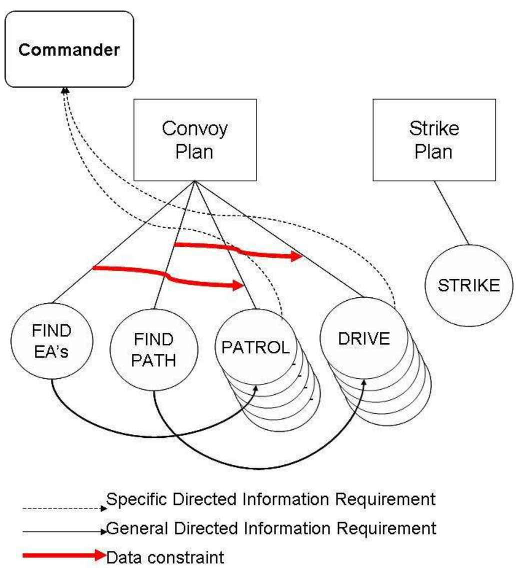

Figure 2: Team Oriented Plans for escorting convoys. The plan on the right is the Strike Plan which directs a UAV to prosecute a target. The plan on the left is the *Convoy Plan* which directs and protects a convoy of ground vehicles along a path of least resistance to its destination.

Figure 2 shows the two TOPs currently used in the system. In the diagram proxies are represented as rectangles with rounded edges, plan operators with rectangles and roles as circles. Dashed arrows represent general *DIRs*, thin solid arrows represent specific *DIRs*, and thick solid arrows represent *Data Constraints*. The plan on the right is the Strike Plan which directs a UAV to prosecute a target. This plan has only a single role, the Strike role. The plan on the left is the *Convoy Plan*. The *Convoy Plan* coordinates a group of vehicles to drive between the extreme East and West ends of the map, an expert Terrain Analysis agent to identify safe routes for the convoys and potential ambush spots along these routes, and a group of UAVs to scout the potential ambush spots. This plan has 12 roles. The first two, the *find EAs* role and the *find Path* role require finding potential ambush points and safe travel paths respectively. The next five roles are *Patrol* roles that can be filled by any UAV. The last five roles are *Drive* roles that are filled by Hummer convoys.

There is a specific *Directed Information Requirement* between the *Drive* and *Patrol* roles and the Commander proxy. The purpose of this DIR is to ensure that the friendly asset proxies relay any potential Tank sightings to the Commander proxy. This allows the fusion to be performed. The general DIR between the *find EAs* role and the *Patrol* role is necessary to allow the TAA to refine the UAVs search area before the UAVs begin to search. The general DIR between the *find Path* role and the *Drive* roles give the TAA the opportunity to calculate safe travel routes for the Hummer convoys.

Algorithms

In this section, we briefly describe the main components that work together to make the integrated system. Figure 1 shows the overall system architecture. The Machinetta proxies provide the general purpose coordination infrastructure that connects the assets in the environment, the commander and the terrain analysis agent. Control agents connected to the assets encapsulate their local intelligence and communicate via a high level interface to their own proxy. The terrain analysis agent is a separate agent with its own proxy that provides information about likely engagement areas and safest paths when a convoy of Humvees needs to move. The Commander interface is connected to its own proxy and only gets information when the proxies of the assets or terrain analysis agent explicitly provide it. Since the Commander receives updates from all the proxies, the information fusion process takes place there and its results are immediately made available to the Commander (as well as the team). In the following, we briefly describe the key features of each of the components, focussing on the aspects that were adapted for the overall intergration.

Terrain Analysis Agent

Human intelligence analysts combine heterogeneous sources of information throughout the course of a military operation to interpret and predict adversarial actions. One of the most important of these fusion products is information about the terrain encompassed by the area of operations. Consequently, we have developed a terrain analysis agent (TAA) to act as a member of the team and provide a tactical analysis of the map to the rest of the team. The TAA models the terrain using a network of electrical resistors and uses current flow through the network to infer tactical characteristics of the terrain. The TAA takes two categories of input. The first input category includes low-level terrain information such as soiltype, vegetation, and elevation. The second category includes sensor reports providing information about the placement of enemy obstacles and weapon systems. In the current system configuration only the first category of input is used.

Two key outputs from terrain analysis are likely ambush areas called *Engagement Areas* and low resistance paths.

This information facilitates the execution of many of the other team-member's tasks. The TAA's output allows UAVs to refine their search areas and convoys to find safe travel routes. Perhaps most importantly, the TAA's assesment of Engagement Areas is fused with target identifications from Patrolling UAVs to increase confidence in these assesments before they are presented to the Commander. This reduces the burden on the Commander while critical target prosecution decisions are made.

The TAA is a full member of the team and has its own proxy which coordinates with the proxies of the other team-members when managing the *Convoy Plan*. The TAA is capable of the *find Path* and find EAs role in this plan and can produce a list of potential ambush points and safe travel routes for the convoys in under a minute. In user experiments a similar analysis by a human took 45 minutes.

Information Fusion

A distributed system acting in a complex environment must formulate actions based on large amounts of uncertain data arriving asynchronosly from multiple sources. In domains such as the military where high certainty is a prerequisite for action, it is necessary to fuse data to increase confidence levels beyond that which results from individual agent assesments. Data fusion techniques seek to fuse them together to provide increased confidences for each target. Sensor data fusion could happen from raw data level, e.g., images, to the decision level. In this paper we consider decision level fusion, where the sensor output for each target is a list of candidate target types (e.g., M1 tank, T80 Tank, etc.) with different confidence levels. Figure 3 describes two lists of candidate target types with different confidence levels from two SAR sensors for a $T80$ tank on the ground. The figure also gives the fused confidence levels for each candidate target type based on Dempster-Shafer theory. For more detail see (Yu *et al.* 2005).

Terrain Context for Sensor Fusion

Sensor fusion tasks attempt to raise the confidence levels in low level fusion assessments by combining results from multiple sensors. Terrain analysis products can be a powerful context for such a task. If a tactical analysis of terrain is done properly then the identification of a tactically significant area should lend credence to a sensor identification of an enemy unit in the vicinity. In the same way one might want to decrease the confidence in the identification of an enemy unit in a region not deemed tactically significant. Our model is designed so that the degree of current flow through a grid cell indicates tactical significance. If we normalize current flow across the grid to lie in the range (0,1) then current flow can be used directly as a multiplier for confidence levels in a sensor fusion system to indicate evidence due to terrain. Formally, let $V = \{v_1, v_2, \ldots, v_n\}$ be the set of possible vehicle types and $c(v_i)$ be the confidence level of v_i, where v_i, $1 \leq i \leq n$, is the possible type of vehicles a SAR sensor can recognize. If a target u is detected in a grid cell with the degree of current flow s, we can adjust the confidence level for each vehicle type as follows if the vehicle type with maximal confidence level v_i is a tank, e.g., T80 and M1 tanks.

$$\forall v_i \in V, c(v_i) = c(v_i) * (s + 0.5)$$

Moreover, we need to normalize the confidence levels for target u to $\sum_{v_i \in V} c(v_i) + c(clutter) = 1$.

Commander Interface

In some of the most advanced "autonomous" systems for space and the military, large teams of humans are required to oversee the "autonomous" activity. Thus, it is reasonable to expect that effective commander control in Sanjaya would be a very difficult problem.

The human commander is a member of the team represented by a proxy. The proxy handles the coordination for the teamwork but a domain specific interface is required between the commander and his proxy to ensure that the proxy represents the commanders intentions effectively during coordination of team activities. This interface comes in the form of a graphical user interface with facilities for the commander to instantiate team plans and to receive filtered information from other team-members. It is important to note that like any other team-member the commander provides information but does not dictate who should receive it. The proxies determine the flow of information between team-members.

The following is a description of some of the facilities provided to the commander by the interface and how they convey the commander's intent to the system.

At the start of the scenario, the commander issues orders to move across the map. In response to this the Machinetta coordination code generates a Team Oriented Plan and coordinates choosing which team members fill which roles in the plan. This coordination occurs without the commander's participation, being executed by his proxy and the other proxies representing members of the team.

The commander can select enemy units displayed on the map and examine their classifications, and confidence levels. These potential targets and their confidences are a result of the fusion of multiple UAV sightings with terrain analysis information. An interactive confidence level filter allows the commander to dynamically inform the team of the level of certainty of identifications the commander wishes before being alerted of a potential target. Based on these classifications the commander can then place 'strike' icons over enemy units, causing a team oriented plan to be created to attack that target, the proxies then coordinate plan execution.

Experiments

The scenario we tested on has the Blue Force with 25 ground vehicles, and 25 UAVs opposed by 20 tanks. The scenario also includes 60 civilian trucks, that are not on either side. All units are placed on the same 50 kilometer by 50 kilometer map. The scenario calls for five convoys, of 5 ground vehicles each, travel from starting positions on one side of the map to destinations on the other side. Each convoy is supported by a group of either 5 UAVs. 20 Opposing Force tanks are placed on the map to defend the area. The 60 civilian trucks are placed on the map to provide confusion and

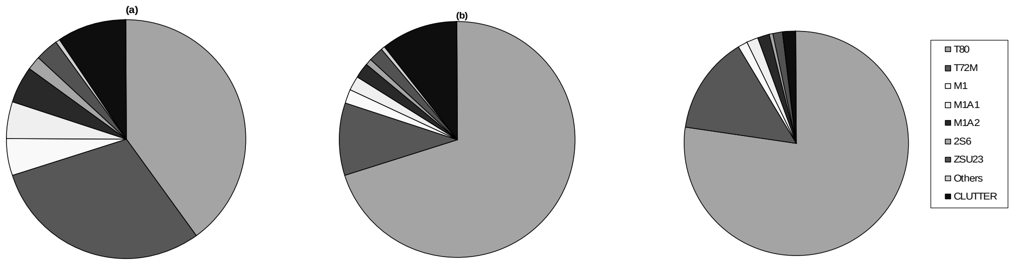

Figure 3: A low resolution sensor SAR_1 (a) and a high resolution sensor SAR_2 (b) return a list of candidate target types with different confidence levels for a ground target $T80$ tank. The right table shows the fused confidence levels for each candidate target type.

make the task of locating Opposing Force units more difficult.

Twenty sets of dispositions for all 20 Opposing Force tanks each, were generated and saved to a file. Each of these sets of dispositions were then reused for all of the simulation runs. Tanks were placed so they were more likely to be in locations predicted by the terrain analysis. This was not intended to make the terrain analysis perform well, instead the terrain analysis was simply saying where the best locations to be were. For each separate run of the simulation, each of the 60 civilian trucks are randomly placed. Trucks then move about randomly within a 5 kilometer by 5 kilometer box centered on their starting position.

The Commander Interface was provided with a simple 'auto strike' feature that creates an attack plan (only UAVs are capable of filling attack roles) when an Opposing Force tank was identified with a confidence of 0.6 or higher. When the intelligent information fusion is not being used, the Commander Interface instead simply used the most recent sensor reading. When terrain analysis was not used to provide paths for the convoy and patrol locations for UAVS, a straight line from start to destination was used as the path and the UAVs were tasked to randomly patrol the entire map.

Figure 4 shows the number of surviving ground vehicles for configurations which did not use the Terrain Analysis agent for planning safe travel routes and identifying potential ambush points. These configurations were fully automated and did not use a human commander. The Y axis gives the number of surviving ground vehicles. The bars on the X axis represent different system configuration groups. The first bar represents the average number of survivors for trials where no sensor fusion features are used to increase confidence of sensor readings. The second bar shows the average number of survivors for trials where output from the Terrain Analysis agent was used to adjust confidence levels during sensor fusion. The third bar shows the survivor average for trials where Dempster-Shafer theory was used to perform sensor fusion on multiple sensor readings. And the fourth bar shows the survivor average where both Dempster-Shafer and Terrain Analysis output were used to adjust confidence levels during sensor fusion.

Figure 5 shows the number of surviving ground vehicles for configurations which did use the Terrain Analysis agent for planning safe travel routes and identifying potential ambush points. These configurations were fully automated and did not use a human commander. The Y axis gives the number of surviving ground vehicles. The bars on the X axis represent different system configuration groups. The first bar represents the average number of survivors for trials where no sensor fusion features are used to increase confidence of sensor readings. The second bar shows the average number of survivors for trials where output from the Terrain Analysis agent was used to adjust confidence levels during sensor fusion. The third bar shows the survivor average for trials where Dempster-Shafer theory was used to perform sensor fusion on multiple sensor readings. And the fourth bar shows the survivor average where both Dempster-Shafer and Terrain Analysis output were used to adjust confidence levels during sensor fusion.

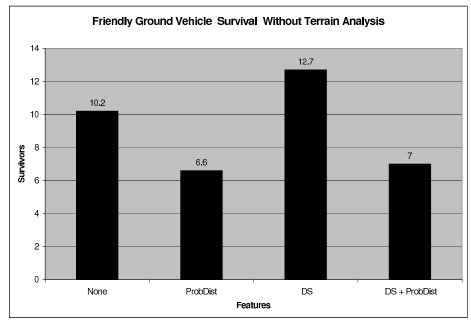

Figure 4: Surviving ground vehicles for trials without Terrain Analysis for safe travel routes and identifying ambush points.

Figure 6 shows the number of surviving ground vehicles for human commanders, for configurations which used the Terrain Analysis agent as well as those that didn't. The Y axis gives the number of surviving ground vehicles. The two bars represent the two sets of live trials which were run. In the first live trial group, none of the features of sensor fusion or terrain analysis were available to the commanders. In the second group, all of the sensor fusion and terrain analysis features were made available.

Figure 7 shows the number of messages exchanged between Proxies for configurations which did not use the Terrain Analysis agent for planning safe travel routes and identifying potential ambush points. The Y axis gives the av-

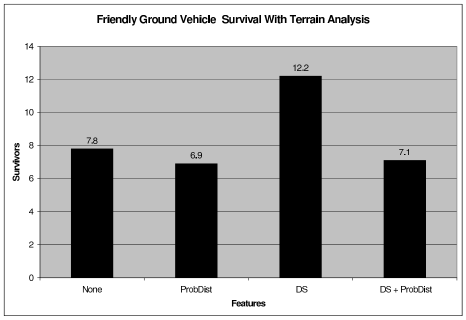

Figure 5: Surviving ground vehicles for trials with Terrain Analysis for safe travel routes and identifying ambush points.

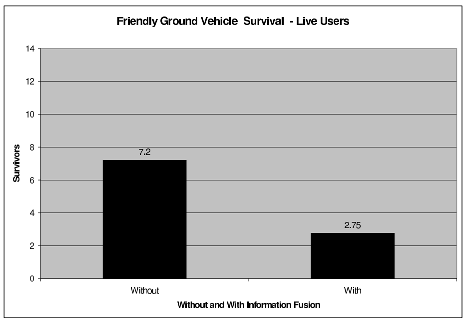

Figure 6: Surviving ground vehicles for human commander trials

erage number of messages exchanged. The bars on the X axis represent different system configuration groups. The first bar shows the average number of Proxy messages exchanged for trials where no sensor fusion features are used to increase confidence of sensor readings. The second bar shows the average number of Proxy messages exchanged for trials where output from the Terrain Analysis agent was used to adjust confidence levels during sensor fusion. The third bar shows the average number of Proxy messages for trials where Dempster-Shafer theory was used to perform sensor fusion on multiple sensor readings. And the fourth bar shows the average number of Proxy messages exchanged where both Dempster-Shafer and Terrain Analysis output were used to adjust confidence levels during sensor fusion.

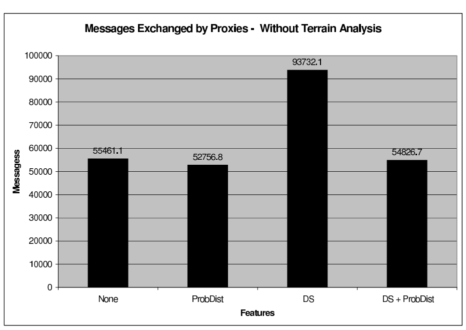

Figure 7: Messages exchanged between Proxies for trials without Terrain Analysis for safe travel routes and identifying ambush points.

Figure 8 shows the number of messages exchanged between Proxies for configurations which did use the Terrain Analysis agent for planning safe travel routes and identifying potential ambush points. The Y axis gives the average number of messages exchanged. The bars on the X axis represent different system configuration groups. The first bar shows the average number of Proxy messages exchanged for trials where no sensor fusion features are used to increase confidence of sensor readings. The second bar shows the average number of Proxy messages exchanged for trials where output from the Terrain Analysis agent was used to adjust confidence levels during sensor fusion. The third bar shows the average number of Proxy messages for trials where Dempster-Shafer theory was used to perform sensor fusion on multiple sensor readings. And the fourth bar shows the average number of Proxy messages exchanged where both Dempster-Shafer and Terrain Analysis output were used to adjust confidence levels during sensor fusion.

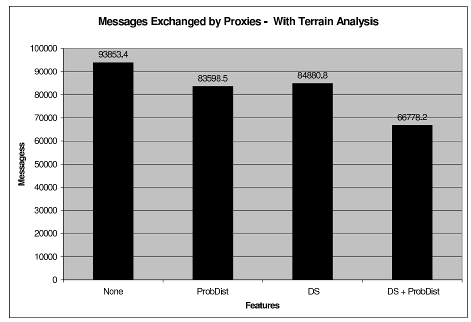

Figure 8: Messages exchanged between Proxies for trials with Terrain Analysis for safe travel routes and identifying ambush points.

Figure 9 shows the number of messages exchanged between Proxies for human commanders, for configurations which used the Terrain Analysis agent as well as those that didn't. The Y axis gives the average number of messages exchanged. The two bars represent the two sets of live trials which were run. In the first live trial group, none of the features of sensor fusion or terrain analysis were available to the commanders. In the second group, all of the sensor fusion and terrain analysis features were made available.

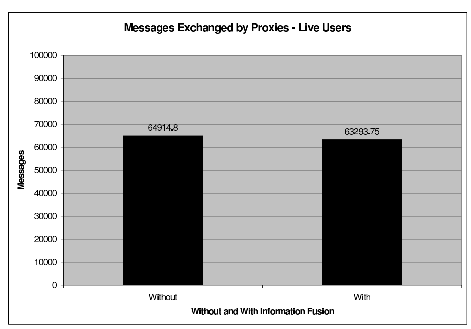

Figure 9: Messages exchanged between Proxies for human commander trials

Analysis For the survival of ground vehicles metric there is an obvious pattern reflected in both sets of bar graphs representing trials without a human commander. Trials that used only Dempster-Shafer for sensor fusion clearly performed best in both cases. Configurations where no sensor fusion was used came in a close second in the trials that did not use the Terrain Analysis agent for paths and ambush points. However, in those trials that did use the Terrain Analysis agent for paths and ambush points, the configuration with no sensor fusion performed much poorer. Trials using both Dempster-Shafer and Terrain Analysis output to adjust confidence levels performed almost as well as those using only the Terrain Analysis output, but both were worse than using no sensor fusion at all. Notice, that when all technologies were used a high number of ground vehicles survived, although relatively few UAVs were required to hit targets. This indicates that the technologies were effectively brought together.

Related Work

Multiagent coordination is an extensively studied area of multiagent systems. However, some of them, including distributed constraint-based algorithms(Mailler & Lesser 2004; Modi *et al.* 2003), combinatorial auctions(Hunsberger & Grosz 2000) do not scale well to very large teams. Recent work on scalable coordination illustrates that exponential search spaces, excessive communication demands, localized views, and incomplete information pose major problems for large scale systems. Initial work on token-based approaches promises a way to address these challenges(Paul Scerri & Mailler 2004). Large scale coordination in the GPGP/TAEMS framework was demonstrated using a token-based algorithm(Wagner, Guralnik, & Phelps 2003). Ortiz et al. study distributed task management in robotic systems(Ortiz, Vincent, & Morisset 2005). However, most of the existing approaches study team coordination as a separate problem and do not consider the uncertainty of sensor data and environments and their effects on task allocation.

Conclusions

In this paper we presented a teamwork based approach to intergrated complex technologies into a flexible, cohesive intergrated system. Results in a complex military simulation environment illustrated that the infrastructure was able to produce synergies between the individual technologies. However, individual technologies did not always improve overall performance. Importantly for the future deployment of complex multiagent systems, initial user testing showed that users were not able to take full advantage of available technology and often made system performance worse.

This work has exposed more problems to solve than it has solved. Clearly, there are key issues in giving a human effective control, but a range of other issues must be addressed as well. Specifically, we believe a key issue is to develop more sophisticated TOP languages to allow specification of the types of complex behavior we require of the systems. More practical issues, such as development environments and debuggers will also be critical for future progress.

References

Aircraft, S. M. 1995. The TACSI users guide. technical report GDIO-MI-98:356. Technical report, Saab Military Aircraft. Edition 5.2.

Chalupsky, H.; Gil, Y.; Knoblock, C. A.; Lerman, K.; Oh, J.; Pynadath, D. V.; Russ, T. A.; and Tambe, M. 2002. Electric Elves: Agent technology for supporting human organizations. *AI Magazine* 23(2):11–24.

Cockburn, D., and Jennings, N. 1996. *Foundations of Distributed Artificial Intelligence*. Wiley. chapter ARCHON: A Distributed Artificial Intelligence System For Industrial Applications, 319–344.

Cohen, P. R., and Levesque, H. J. 1991. Teamwork. *Nous* 25(4):487–512.

Giampapa, J., and Sycara, K. 2002. Team oriented agent coordination in the RETSINA multi-agent system. In *Proceedings of Agents02*.

Goodrich, M.; Olsen, D.; Crandall, J.; and Palmer, T. 2001. Experiments in adjustable autonomy. In Hexmoor, H.; Castelfranchi, C.; Falcone, R.; and Cox, M., eds., *Proceedings of IJCAI Workshop on Autonomy, Delegation and Control: Interacting with Intelligent Agents*.

Hunsberger, L., and Grosz, B. 2000. A combinatorial auction for collaborative planning.

Jeffrey Cox, E. D., and Bartold, T. 2005. A distributed framework for solving the multiagent plan coordination problem. In *AAMAS'05*.

Jennings, N. 1995. The archon systems and its applications. Project Report.

Kinny, D. 1993. The distributed multi-agent reasoning system architecture and language specification. Technical report, Australian Artificial intelligence institute, Melbourne, Australia.

Kitano, H.; Tadokoro, S.; Noda, I.; Matsubara, H.; Takahashi, T.; Shinjoh, A.; and Shimada, S. 1999. Robocup rescue: Searh and rescue in large-scale disasters as a domain for autonomous agents research. In *Proc. 1999 IEEE Intl. Conf. on Systems, Man and Cybernetics*, volume VI, 739–743.

Kortenkamp, D.; Schreckenghost, D.; and Martin, C. 2002. User interaction with multi-robot systems. In *Proceedings of Workshop on Multi-Robot Systems*.

Lesser, V.; Atighetchi, M.; Benyo, B.; Horling, B.; Raja, A.; Vincent, R.; Wagner, T.; Xuan, P.; and Zhang, S. 1999. The UMASS intelligent home project. In *Proceedings of the Third Annual Conference on Autonomous Agents*, 291–298.

Mailler, R., and Lesser, V. 2004. A cooperative mediation-based protocol for dynamic, distributed resource allocation.

Modi, P. J.; Shen, W.-M.; Tambe, M.; and Yokoo, M. 2003. An asynchronous complete method for distributed constraint optimization. In *Proceedings of Autonomous Agents and Multi-Agent Systems*.

Nair, R.; Tambe, M.; and Marsella, S. 2003. Role allocation and reallocation in multiagent teams: Towards a practical analysis. In *Proceedings of the second International Joint conference on agents and multiagent systems (AAMAS)*.

Ortiz, C. L.; Vincent, R.; and Morisset, B. 2005. Task inference and distributed task management in centibots robotic systems. In *AAMAS*.

Paul Scerri, R. V., and Mailler, R. 2004. Comparing three approaches to large scale coordination. In *Proceedings of AAMAS'04 Workshop on Challenges in the Coordination of Large Scale MultiAgent Systems*.

Pynadath, D., and Tambe, M. 2002. Multiagent teamwork: Analyzing the optimality and complexity of key theories and models. In *First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02)*.

Pynadath, D.; Tambe, M.; Chauvat, N.; and Cavedon, L. 1999. Toward team-oriented programming. In *Intelligent Agents VI: Agent Theories, Architectures, and Languages*, 233–247.

Scerri, P.; Pynadath, D. V.; Johnson, L.; Rosenbloom, P.; Schurr, N.; Si, M.; and Tambe, M. 2003. A prototype infrastructure for distributed robot-agent-person teams. In *The Second International Joint Conference on Autonomous Agents and Multiagent Systems*.

Scerri, P.; Xu, Y.; Liao, E.; Lai, J.; and Sycara, K. 2004. Scaling teamwork to very large teams. In *Proceedings of AAMAS'04*.

Schurr, N.; Marecki, J.; Lewis, J.; Tambe, M.; and P.Scerri. 2005a. The DEFACTO system: Training tool for incident commanders. In *IAAI'05*.

Schurr, N.; Marecki, J.; Tambe, M.; Scerri, P.; Levis, J.; and Kasinadhuni, N. 2005b. The future of disaster response: Humans working with multiagent teams using DEFACTO. In *AAAI Spring Symposium on Homeland Security*.

Stone, P.; Balch, T.; and Kraetszchmar, G., eds. 2001. *RoboCup-2000: Robot Soccer World Cup IV*. Berlin: Springer Verlag. To appear.

Tambe, M.; Shen, W.-M.; Mataric, M.; Pynadath, D.; Goldberg, D.; Modi, P. J.; Qiu, Z.; and Salemi, B. 1999. Teamwork in cyberspace: using TEAMCORE to make agents team-ready. In *AAAI Spring Symposium on agents in cyberspace*.

Wagner, T.; Guralnik, V.; and Phelps, J. 2003. A key-based coordination algorithm for dynamic readiness and repair service coordination. In *AAMAS*.

Yu, B.; Giampapa, J.; Owens, S.; and Sycara, K. 2005. An evidential model of multisensor decision fusion for force aggregation and classification. In *Proceedings of Eighth International Conference on Information Fusion*.