Human-Agent Collaboration for Time Stressed Multi-Context Decision Making

Xiaocong Fan, *Senior Member, IEEE,* Michael McNeese, Bingjun Sun, Timothy Hanratty, Laurel Allender, and John Yen, *Fellow, IEEE,*

Abstract— **Multi-context team decision making under time stress is an extremely challenging issue faced by various real world application domains. In this study we employ an experience-based cognitive agent architecture (R-CAST) to address the informational challenges associated with military command and control (C^2) decision making teams, the performance of which can be significantly affected by dynamic context switching and tasking complexities. Using context switching frequency and task complexity as two factors, we conducted an experiment to evaluate whether the use of R-CAST agents as teammates and decision aids can benefit C^2 decision making teams. Members from a US Army ROTC (Reserve Officer Training Corps) organization were randomly recruited as human participants. They were grouped into ten Human-Human teams each composed of two participants and ten Human-Agent teams each composed of one participant and two R-CAST agents as teammates and decision aids. Statistical inference of the experiment results indicates that R-CAST agents can significantly improve the performance of C^2 teams in multi-context decision making under varying time-stressed situations.**

Index Terms— **Human-Centered Computing, Cognitive Agents, Context Switching, Naturalistic Decision Making.**

I. Introduction

TEAM decision making involving multiple contexts is an extremely challenging issue faced by various real world application domains [1], [2], [3], [4]. Military command and control (C^2) in complex urban terrain is one of such domains, where C^2 teams have to frequently confront the so-called three-block challenge [5]—conducting humanitarian, peace-making, and combat missions in close proximity (i.e., involving three contexts that overlap in time). Multi-context team decision making is challenging because it requires effective team collaborations in rapidly gathering dynamic information from multiple sources (collateral space), in proactively sharing relevant information for establishing situation awareness, in managing and reasoning across multiple decision spaces for different contexts (areas of interest), and in choosing optimal courses of action.

In the Multi-Agent Systems (MAS) research area, there has been an increasing interest in empowering agents with naturalistic decision making models to better support *human-agent collaboration* in making decisions under time stress [6], [7], [8], [9], [10]. One particular model is Klein's Recognition-Primed Decision framework (RPD) [11]. The RPD model is based on the supposition that in complex situations human experts usually make decisions based on the recognition of similarities between the current decision situation and previous decision experiences [12]. Cognitive studies have shown that over 95% of human decisions conform to the RPD model in various time-stressed situations [12]. Norling, Sonenberg, and Ronnquist [7] have examined the integration of RPD model into a BDI agent framework. Fan, Sun, McNeese, and Yen have implemented an RPD-enabled cognitive agent architecture (R-CAST) [6] and evaluated the performance gains when human members were assisted by R-CAST agents in their time-stressed decision making.

However, multi-context decision making has attracted little attention from the MAS field especially when considered from the human-centered teamwork perspective. Norling et al.'s work [7] only explored ways of using reinforcement learning to enhance situational recognition. While Fan et al.'s attempt [6] centered on human-agent collaboration in the decision making process, they only investigated the impact of an adaptive decision making mechanism, with the issue of multi-context decision making left open. Although RPD-enabled agents were employed to start addressing the issue of multiple contexts [5], experimental studies are still needed for fully understanding the nature of multi-context decision making, which is critical for further development of agent technologies to enhance the performance of human-centered teamwork.

This research couples human-centered teamwork and cognitive agent technology to address team decision making challenges. *Human-centered teamwork* can be viewed as a sub-area of Multi-agent teamwork [13], which is concerned with joint commitments and joint responsibility. Human-centered teamwork argues for stronger interaction between software agents and their human peers. Within teamwork, both humans and agents are jointly responsible for establishing mutual situation awareness [14], developing shared mental models as situations evolve, and adapting to mixed-initiative activities. If agents could effectively collaborate with human peers, then they potentially offer humans an opportunity to pay attention to more important activities [15], and to make better decisions that take advantage of information with greater accuracy and finer granularity [16].

Previous studies on human-centered agent-based teamwork

Manuscript received January 20, 2007. This work was supported as an FY06 Research Task under the Army Research Laboratory's Advanced Decision Architectures Collaborative Technology Alliance (ARL ADA CTA).

X. Fan is with the Pennsylvania State University, email: xfan@psu.edu. M. McNeese is with the Pennsylvania State University, email: mmcneese@ist.psu.edu. B. Sun is with the Pennsylvania State University, email: bsun@cse.psu.edu. T. Hanratty is with US Army Research Lab, Aberdeen Proving Ground, email: hanratty@arl.army.mil. Laurel Allender is with US Army Research Lab, Aberdeen Proving Ground, email: lallende@arl.army.mil. J. Yen is with the Pennsylvania State University, email: jyen@ist.psu.edu.

include KAoS [15] and MokSAF [16]. KAoS captures a collection of agent services in the form of adjustable policies, which allow the renegotiation of roles and tasks among humans and agents when new opportunities arise or when breakdowns occur. MokSAF is a computer-based simulation system developed to evaluate how humans can interact and obtain assistance from agents within a team environment. However, neither directly touched the issue of multi-context decision making under time stress.

Case-based reasoning (CBR) [17], [18], [19] is another psychological theory of human cognition, focusing on the process of reminding (experience-guided reasoning) and learning. While there is no clear line between RPD and CBR as far as their process models are concerned, RPD focuses more on time-sensitive recognition refinement.

For this study we employ an experience-based cognitive agent architecture (R-CAST) to address the challenge of multi-context decision making associated with C^2 teams. In particular, we examine the potential impacts of R-CAST agents, when acting as human's teammates and decision aids, on the performance of C^2 teams in making decisions involving multiple contexts and significant time stress. The remainder of this paper is organized as follows. The background of R-CAST agent architecture and its features pertinent to this study are given in Section II. The problem domain, experiment design, and tasks are described in Section III. The details of experimental apparatus are presented in Section IV. Section V reports major findings from the study, and Section VI concludes the paper.

II. The R-CAST Agent Architecture

The R-CAST agent architecture [6] is built on top of the concept of shared mental models [20], the theory of proactive information delivery [21], and Klein's Recognition-Primed Decision (RPD) Model [11].

The major components of R-CAST is depicted in Figure 1. The "RPD Engine" module implements a recognition cycle as prescribed in the RPD theory. The cycle starts with "Situation Awareness", where an R-CAST agent utilizes the inference knowledge and past experience knowledge to generate/synthesize a recognition of the current decision situation. A recognition has four constructs: relevant cues (what to pay attention to), plausible goals (which goals make sense), expectancy (what will happen next), and courses of action (what actions worked before). The invalidation of an expectancy may indicate that the once workable course of action (COA) may become no longer applicable to the changing situation. Thus, an R-CAST agent keeps monitoring the status of the expectancy so that it can further adapt the recognition or, if necessary, start a new recognition cycle. The execution of a COA can involve inter-agent (taskwork) and intra-agent (teamwork) activities, which are coordinated by the "Process Manager". The experience knowledge can be gradually expanded with experience newly synthesized by an agent during the recognition cycle (Experience Adaptation).

One of the biggest challenges to develop cognitive agents for supporting human-centered computing is how to represent

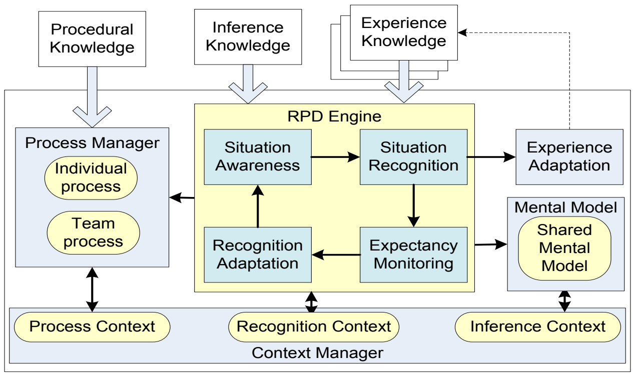


Fig. 1. Major components of R-CAST.

and reason about the *contextual needs of collaboration*. R-CAST implements a context manager that distinguishes three types of context: process context, recognition context, and inference context. These three types of context together enable an R-CAST agent to effectively identify information needs of other team members, to quickly adapt decisions to a dynamic environment, and to facilitate the reuse of inference knowledge in various means-ends reasoning activities. We next describe each of the three contexts.

A. Decision Process Context

A process context refers to the progress of the on-going processes, each of which may involve an individual agent or a team of agents. It is argued that teamwork progress can be employed as part of a shared mental model for an agent to progressively infer other's information needs [22].

The RPD process has been considered as an internal process of an individual decision maker. In order to enable close human-agent collaboration during the decision process, R-CAST implements a 'collaborative-RPD process'. Managing the RPD process at the team level enables an R-CAST agent to establish a shared mental model with its human peer about the dynamic progress of the decision process being pursued. Equipped with such a shared mental model, the agent is able to offer context-aware computing to, and reduce its cognitive gap with, its human peer.

The decision process context (a component of the process context as shown in Fig. 1) captures the progress of the current decision making task along the RPD process, including information collection, situation recognition (feature matching and story building), recognition consolidation, expectancy monitoring, recognition adaptation and COA evaluation. The decision process context is implemented as a view on the GUI, and it is directly manipulable to the human peer. In such a sense, it is at a higher level than the other two types of context, which are used in an agent's internal reasoning only.

Knowing of the progress of his/her agent encourages the human peer to focus his/her cognitive attention on the activity that the agent is currently working on so that the human could adjust the agent's behavior at the earliest opportunity; this is a step towards 'just-in-time' adjustable autonomy. For instance, anomalous situations can happen frequently in a dynamic environment. During a decision process, both a human decision maker and his/her agent need to monitor the expectancy associated with the current recognition, and to re-assess the recognition to determine how to best respond to an anomaly when it emerges. For example, suppose for a given situation it is expected that two crowds, G_1 and G_2, should be active in two isolated regions. When G_1 and G_2 start to move closer and merge, an R-CAST agent detects the anomaly and decides to adapt the chosen COA. However, the human peer may override the agent's decision, choosing to collect more situational information and start a new recognition cycle.

B. Recognition Context

R-CAST uses the concept of "decision space" (DS) to organize experiences by decision themes: experiences related to one decision theme are maintained in one decision space. Each experience is of form (S, E, COA), where S is a collection of cues describing a past situation, E if a collection of expectancy, and COA is the name of a course of action worked before. Cues and expectancy are represented by predicates, while COAs are written in a variant of the process language MALLET [23], where each process can be associated with preconditions and escape conditions (first-order expressions).

The following is an example experience:

```
(Experience e26
(Cue        (Close_enough ?crowd1 ?crowd2))
(Expectancy (Move_closer ?crowd1 ?crowd2))
(Anomaly    (Move_away ?crowd1 ?crowd2))
(Goal       (peacekeeping yes))
(Action     (Alert_converged_threat)))
```

It says that experience "e26" applies to the cases where two crowds are coming close enough, and it is expected that they are moving closer; if this is the case, the process "Alert_converged _threat" is triggered; if it turned out that the crowds moved away, the process would be terminated.

Experiences in a decision space are organized hierarchically in a "tree-like" structure (experience tree): an experience at a higher level node is refined by its children at a lower level (i.e., experiences at a lower-level consider more relevant cues and expectancy). Given a decision task, an R-CAST agent first starts with the root (most abstract experience) of the current experience tree. As more and more information becomes available (e.g., some expectancy is confirmed as the situation evolves), potential patterns for further situation evolution become more predictable, and the agent's recognition of a workable experience becomes more and more fine-grained (reaching the lowest possible level of the hierarchy).

An R-CAST agent uses "recognition anchors" to mark the nodes that it believes are closest in similarity to the current situation. A recognition anchor helps an agent to determine the collection of cues and expectancy that needs to considered in the recognition cycle. In addition, when an anomaly occurs, a recognition anchor allows an agent to easily backtrack to an upper level of the experience tree to seek a better solution without necessarily invoking the time-consuming feature-matching computation.

A *recognition context* refers to the collection of all the recognition anchors being considered by an agent. Note that an agent can manage and work with multiple decision spaces simultaneously, each of which may be a forest of experience trees, and moreover, it can place more than one recognition anchor on multiple branches of an experience tree at different levels. Each experience node has a flag indicating whether the node is recognition-anchored by the agent or not. When multiple R-CAST agents form a decision-making team, it is beneficial to allow the agents to exchange their respective recognition anchors so that they could proactively help each other with missing information pertinent to the current context. A simple way to encode the recognition anchors of an experience tree is to traverse the tree in certain order (e.g. preorder) to generate a bit string where 1 (0) indicates the corresponding node is (not) anchored. An agent may choose to inform other agents of the bit string of the experience tree being worked on, or to share the whole recognition context (bit strings of all the experience trees).

An agent (human) working in high demand situations often needs to switch attention among multiple active decision tasks. For instance, as the situation changes, an agent may need to switch from handling battlefield tasks to handling firefighting tasks (between-domain context switching), or simply switch from peacekeeping to humanitarian operations (within-domain context switching). For example, a C^2 team may decide to adjust the military forces allocated to a humanitarian mission to a peacekeeping mission prompted in a nearby location, as more and more field reports indicate that the situation there might evolve in an undesirable direction. Even for this within-domain context switching, the agent has to keep track of the evolution of both the humanitarian mission and the peacekeeping mission so that it can coordinate activities such as resource allocation and information sharing between the two missions and among other members of the C^2 team.

The notion of recognition anchor facilitates an agent to save and restore recognition contexts. For between-domain context switching, an agent needs to freeze the recognition context of the current decision space then defreeze the recognition context of another decision space, while for within-domain context switching, an agent only needs to freeze the recognition context of the current experience tree then defreeze the recognition context of another experience tree. This feature was exploited in our experiment (cf. Section IV) to configure decision aids that can dynamically switch among multiple within-domain tasks: humanitarian, peacemaking, and combat contexts (here we reuse the word 'context' to refer to tasking contexts, which should not be confused with the three agent contexts as shown in Figure 1).

C. Inference Context

The inference context captures inference knowledge that links high-level information needs to lower-level information that is directly obtainable from a wide variety of sources. For example, a cue used to describe a situation is a high-level information need, which can be the root of an inference tree consisting of many other intermediate or directly-observable types of information. This is typical in real-world domains and it is exactly why collaborative computing and distributed cognition (the kind of cognition teams engage in when the task is complex and constraining) play a key role here, because one single agent/human simply cannot handle the complexity due to lack of sufficient expertise, inference knowledge, etc.

The inference context is used in two ways in R-CAST. First, it is used to relate high-level information needs, once they are identified, to low-level information, and to identify missing relevant information. Second, the inference context is used to aggregate lower level information, once it is available, to higher level cues. Separating the inference context from the decision process context and the recognition context allows a piece of inference knowledge to be used in multiple phases of a process, and for multiple experience trees. For instance, inference knowledge that relates shipping density in an area to geopolitical cost can be used to evaluate a COA in responding to a threat. The same knowledge can also be used for feature-matching, if the cue of an experience involves geopolitical cost, and for expectancy monitoring.

In sum, to better support human teams to make decisions involving multiple types of tasks, R-CAST has been developed with a context manager that reflects how a human adapts his/her decisions in a dynamic environment, as well as supports the identification and satisfaction of other team members' information needs pertinent to the current context.

III. Task Description and Experiment Design

Our problem domain involves C^2 teams reacting to potential threats that emerge unexpectedly in an urban area. It imposes challenging information demands associated with the command and control of urban operations, including humanitarian, peacekeeping, and combat operations.

The simulation environment contains three types of threats: Improvised Explosive Devices (IEDs), crowds, and insurgents, which represent the targets of humanitarian, peacekeeping, and combat operations, respectively. *IEDs* are motionless targets, and if exploded, can cause damage to the nearby objects. A *crowd* represents a group of people which may contain activists that can be friends or foes. A crowd can be of medium (M) or large (L) size, and the group size of a crowd can change over time. Two crowds can merge together if they move close enough. Another type of movable targets is *insurgents*, each is associated with a threat level that can be L (low), M (medium), or H (high).

Other objects of interest in the environment are main supply routes (MSRs) and three types of key buildings: religious buildings, schools, and hospitals. There are also limited number of friendly units, squads and Explosive Ordnance Disposal (EOD) teams, under the control of a C^2 team.

We next present the task description, knowledge acquisition for R-CAST agents, and the design of a human-in-the-loop experiment with R-CAST agents as teammates and decision aids.

A. Task Description

In this study, a C^2 team consists of an S2 suite (intelligence cell) and an S3 suite (operations cell). Each human operator of a C^2 team has equipment with two monitors: a map display for tracking situation development, and a graphical user interface (GUI) for collaborating with teammates to handle threats.

The roles of C^2 operators have been simplified. S2 is responsible for processing incoming reports, called Spot reports; collecting relevant information from another source– a simulated Military Intelligence Database (MIDB); and alerting S3 of potential threats. S3 needs to process alerts from S2, and make decisions on which target to handle next and which resources (friendly units) to allocate toward that target.

In particular, the tasks of S2 suite include:

1. Focus attention on an active target pertinent to the current context (humanitarian, peacekeeping, or combat);
2. If the target is a crowd *c*:
   i. If *c* is unknown to S3, then alert S3;
   ii. If the size of *c* changes, then alert S3;
   iii. If there are unknown activists associated with *c*, then query MIDB to figure out whether the activists are friends or foes, alert S3 afterward;
   iiii. If *c* is moving closer to a key building or a supply route, query MIDB to check whether *c* is close enough to impose a threat. If so, alert S3;
3. If the target is a key insurgent *k*:
   i. If *k* is unknown to S3, then alert S3;
   ii. If the threat level of *k* is unknown, then query MIDB to figure out the threat level, alert S3 afterward;
   iii. If *k* is moving closer to a key building or a supply route, query MIDB to check whether *k* is close enough to impose a threat. If so, alert S3;
4. If the target is an IED *i*:
   i. If *i* is unknown to S3, then alert S3;
   ii. Query MIDB to check whether *i* is close enough to a key building or a supply route to impose a threat. If so, alert S3;
5. Repeat [1–4] for another active target, if any changes happened.

The tasks of S3 suite include:

1. Select an active target to handle. It is critical to prioritize the order of operations on the active targets. It is preferable to first handle targets with a higher level threat.
2. Allocate resources (friendly units) to the selected target. Free resources that are closer to the selected target are preferred. Resources already engaged with other targets can be reallocated if the currently selected target has a higher level of threat;
3. Monitor the map display to see whether the selected target, if it is movable, is getting close to or away from certain key buildings. Mark the nearby buildings, if any, through S3 GUI;
4. Issue the command to handle the selected target;
5. Repeat [1–4] to handle another active target.

In this multi-tasking environment, resource allocation is a constraint-satisfaction problem. The S3 suit needs to consider (a) resource constraints—balancing requirements (resource type, amount) among multiple targets; (b) utility constraints—maximizing the number of targets successfully handled; (c)

TABLE I
REQUIREMENTS ON HANDLING TARGETS

Targets			*Value*	*Res. req.*	*Action*
Crowd	M	w/o foe	20	1U	monitor
	M	w/ foe	40 (+10)*	2U	disperse
	L	w/o foe	40 (+10)*	2U	disperse
	L	w/ foe	50 (+10)*	3U	disperse
Insurgent			n=1,2,3 for L,M,H		
(3 threat levels: L, M, H)			50+50n	(n+1)U	capture
IED			0, 60, 80*	1U + 1E	remove

'U' refers to "squad unit", 'E' refers to EOD team.
*60 if the IED is close to buildings only or MSRs only;
80 if close to both; 0 otherwise.

spatial constraints—others being equal, considering first the resources closest to a target; and (d) timing constraints—issuing tasks earlier has a better chance to succeed, due to the uncertain nature of a target's lifespan.

B. Expert Knowledge for Agents

As indicated in Figure 1, R-CAST needs three types of domain knowledge from external: procedural knowledge (COAs), inference knowledge (rules), and experience knowledge. To conduct this experiment (i.e., using R-CAST agents as teammates and decision aids), we need to elicit knowledge from domain experts and empower an R-CAST agent with the domain knowledge.

We worked with three domain experts on this C^2 scenario and termed it as 'Three-Block Challenge': within three-block area in a city officers in command must react to a constant flow of intelligence reports and make timely decisions for combat, peacekeeping and humanitarian operations. We produced high-level COAs (e.g., QueryMIDB, AlertS3, AlertS4, CancelAlert, MonitorCrowd, DisperseCrowd, CaptureInsurgent, and RemoveIED) that were considered to be critical to the S2 and S2 roles. Each of these COAs is a process being composed of lower-level actions like MoveTo, LockTarget, and Fire. The domain experts also helped with target prioritization, which allowed us to generate rules, say, for an agent to reason about the threat level of a target.

Guided by the domain experts, we generated a decision space composed of three experience trees, one for each of the three tasking context: combat, peacekeeping and humanitarian operations. Each experience is of the form of the experience 'e26' described in Section II-B, where the Action field takes values from the high-level COAs mentioned above.

To relate to the real-world demands, we also worked with the domain experts on how to handle threats. Table I lists for each type of target the credit value, the number of resources required to handle a target, and what COA S3 should take. Typically insurgents impose more threats than IEDs, which impose more threats than crowds. A credit value is assigned to a target such that it proportionally reflects how much effort to take (or reward to claim) for handling the target successfully.

The allocation of friendly units depends on the threat level and type of the target being considered. For example, the second entry says dispersion of a medium-sized crowd with a foe needs two squad units, and 40 points can be credited if the crowd is dispersed successfully. The last entry says that one squad unit and one EOD team are required to remove an IED. If successful, 60 points can be credited if the IED is close to buildings only or MSRs only, 80 points if it is close to both.

TABLE II
FACTORIAL TREATMENT DESIGN

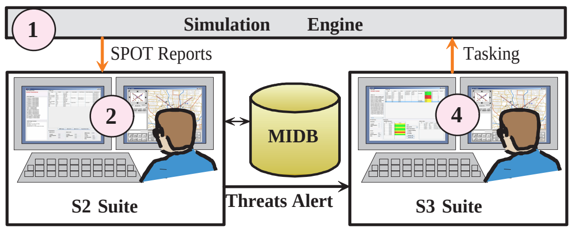
Context	*HH*			*HA*		
Switching	*Task Complexity*			*Task Complexity*		
Frequency	L	M	H	L	M	H
L	10	10	10	10	10	10
M	10	10	10	10	10	10
H	10	10	10	10	10	10

C. Experiment Design

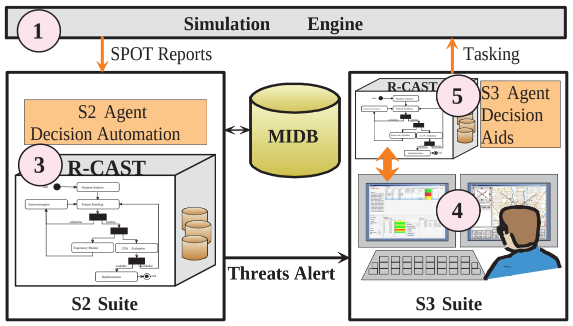
For the domain problem described above, we view activities related to IEDs, crowds, and key insurgents as humanitarian context, peacekeeping context, and combat context, respectively. A *macro* context-switching is imposed (to human operators) when Spot reports from two consecutive cycles are about different types of targets. In addition, when the S2 (or S3) suite gets Spot reports (or alerts) about multiple targets simultaneously, even though they are of the same type, human operators have to switch their attention back and forth to handle individual targets. We refer to this as *micro* context-switching. Because task difficulty correlates with time demands [24], varying the number of active targets (i.e., controlling *micro* context-switching) changes how much time a human operator can take on each target, consequently varies task complexity. We thus have two factors to control: context switching frequency (CSF) and task complexity (TC), which respectively denote how fast a C^2 team has to switch among contexts, and how many active targets a C^2 team has to handle under time stress.

Moreover, our main objective is to understand how R-CAST agents, acting as teammates and decision aids, may affect C^2 performance in multi-context decision making under stress. We want to distinguish C^2 teams with pure human operators from C^2 teams with R-CAST agents acting as decision aids. This factor is denoted as team type (TType).

In sum, this study examines the effects of three factors (team type TType, context switching frequency CSF, and task complexity TC) on the performance of participants in simulated C^2 operations. The TType variable has two levels, human-human (HH) teams and human-agent (HA) teams, where an HH team is composed of two human participants playing the roles of S2 and S3 respectively, and an HA team is composed of an R-CAST agent playing the role of S2 and one human participant aided by another R-CAST agent playing the role of S3. We treat the CSF and TC variables each as having three levels, namely low, medium, and high. For the CSF variable, low, medium, and high correspond to updating the dynamic target information every 15, 10, and 5 seconds, respectively. For the task complexity variable, low, medium, and high correspond to each type of threats having 2, 3, or 4 active targets to be handled. For example, when TC=H, the C^2 team needs to face 12 active targets all the time, 4 for each type (a new target of the same type will pop up when a target is removed by S3 or disappears by itself).



(a)



(b)

Fig. 2. (a) Human-Human (HH) teams; (b) Human-Agent (HA) teams.

In total we designed 9 scenarios reflecting the nine different CSF-TC combinations. For example, one scenario was designed such that the context is switched every 10 seconds, and has 3 active targets for each type (9 active targets in total) at any one time. The scenarios also differ in the settings of initial locations (targets, MSRs, key buildings, IEDs), itineraries and velocities of movable targets, sizing of crowds, threat levels of insurgents, targets' appearance time, etc. Each scenario lasted 10 minutes. In all the scenarios, S3 suite only had limited resources under control: 1 EOD team and 9 squad units (they all moved at the same speeds in the simulation). The goal of a C^2 team is to remove as many threats as possible with limited resources and varying timing constraints.

As shown in Table II, this is a $2\times3\times3$ factorial treatment design, and 10 replications are to be collected for each treatment. Our research problems are (1) Whether the performance of HA teams is significantly ($\alpha = 0.05$) different from HH teams; (2) Whether the CSF and TC factors have significant effects on the performance of C^2 teams; and (3) Whether different level combinations of CSF and TC have significant effects on C^2 team performance. For the first problem, we can formulate a hypothesis test:

$$H_0 : \mu_{HH} = \mu_{HA},$$
$$H_a : \mu_{HH} \neq \mu_{HA}.$$

Hypothesis tests can be formulated for the other two problems similarly.

IV. Experiment

In this section, we detail the apparatus, the human subjects, and the dependent measures used in the experiment.

A. Apparatus

In the order as indicated in Fig. 2, below we describe the five computational components employed in the experiment.

1) Simulation Engine: The development of simulated field situations is controlled by the Simulation Engine. The Simulation Engine module has three components: scenario generator, tasking simulator, and performance evaluator.

The scenario generator accepts script-based description of target dynamics. At every cycle, the *scenario generator* creates a Spot report for each active target based on the script of its dynamics. Here is an example script of a crowd:

```
((crowd C221 40.01 -82.06 0.8 10)
 (report_key A18 foe 7)(successor C212)
 (m 40.04 -84.965 1.0)(m 41.95 -82.975 0.6)
 (s 27 20) (s 49 90))
```

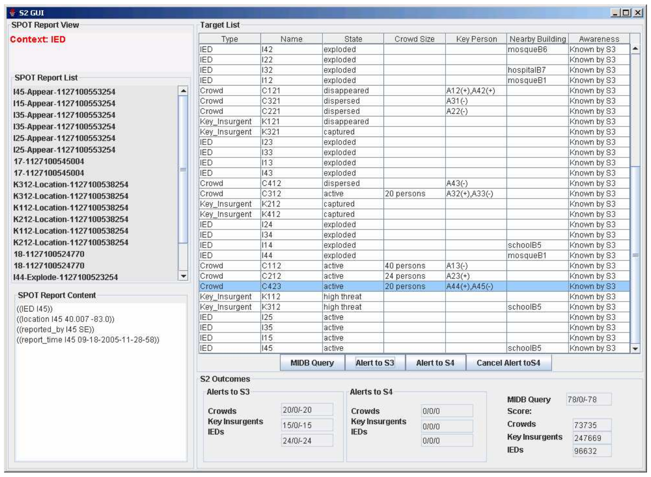
This defines a crowd named C221, that appears at location (40.01, -82.06) 10 seconds after the system starts, and can move at full speed of 0.8 u/s. An associated foe activist with ID A18 shows up 7 seconds later. The crowd moves to (40.04, -84.965) at its full speed, then moves to (41.95, -82.975) at 0.6 times of its full speed. The crowd size changes to 27 after 20 seconds and to 49 after 90 seconds. The disappearance of this crowd will trigger another target named C212 to show up. In general, each target has a lifespan, which depends on whether a movable target moves to the end of its pre-specified itinerary, whether it comes to the expiration time of a motionless target, or whether the target has been successfully handled by the S3 suite, whichever comes first.

Spot reports about situational changes are dynamically generated and fed to the S2 suite following certain communication patterns (e.g., sending reports every n seconds; sending reports of different target types interleavingly in fixed or random order). The communication pattern can be set prior to each run, which is flexible, so that it allows, for example, the setting where reports about crowds are being sent every other cycle. Upon getting new Spot reports, the S2 suite recognizes potential threats and decides whether and when to alert the S3 suite. Thus, although each suite has a map display, the S3 suite may only have a partial view of the current situation.

The *tasking simulator* manages the tasks issued by the S3 suite by monitoring the engagement (e.g., taskload) of each squad/EOD unit, tracking the progress of each ongoing task, and reporting to the scenario generator about any situational changes effected by the completion of a task.

The *performance evaluator* records the number of targets successfully handled; whether the S3 suite paid attention to the nearby buildings when issuing tasks; how many times all the resources were fully engaged; how many times S3 canceled a task that was no longer appropriate as the situation evolved; how many times resources were wasted on non-threatening targets (i.e., small crowds, IEDs far away from key buildings or MSRs); and how many times resources were re-allocated to another target that could offer a better chance to succeed. The *performance evaluator* uses these values to evaluate final team performance; it does not offer on-line performance feedbacks to human users.

2) S2 Human-Computer Interface: S2 GUI is used for HH teams only. The design of S2 GUI was largely influenced by the domain experts we consulted. A screen shot of S2 GUI is shown in Figure 3(a). Displayed on the left of S2 GUI are a


S2 GUI

SPOT Report View

Context: IED

SPOT Report List

I45-Appear-1127100553254
I15-Appear-1127100553254
I35-Appear-1127100553254
I35-Appear-1127100553254
I25-Appear-1127100553254
I25-Appear-1127100553254
I7-1127100545004
I7-1127100545004
K312-Location-1127100538254
K312-Location-1127100538254
K112-Location-1127100538254
K212-Location-1127100538254
K112-Location-1127100538254
K212-Location-1127100538254
I8-1127100524770
I8-1127100524770
I44-Explode-1127100523254

SPOT Report Content

((IED I45))
((location I45 40.007 -83.0))
((reported_by I45 SE))
((report_time I45 09-18-2005-11-28-58))

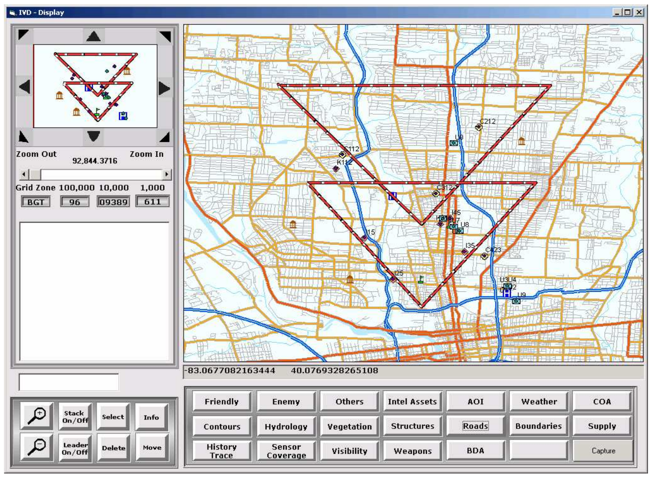
Target List

Type	Name	State	Crowd Size	Key Person	Nearby Building	Awareness
IED	I42	exploded			mosqueB6	Known by S3
IED	I22	exploded				Known by S3
IED	I32	exploded			hospitalB7	Known by S3
IED	I12	exploded			mosqueB1	Known by S3
Crowd	C121	disappeared		A12(+),A42(+)		Known by S3
Crowd	C321	dispersed		A31(-)		Known by S3
Crowd	C221	dispersed		A22(-)		Known by S3
Key_Insurgent	K121	disappeared				Known by S3
Key_Insurgent	K321	captured				Known by S3
IED	I23	exploded				Known by S3
IED	I33	exploded				Known by S3
IED	I13	exploded				Known by S3
IED	I43	exploded				Known by S3
Crowd	C412	dispersed		A43(-)		Known by S3
Crowd	C312	active	20 persons	A32(+),A33(-)		Known by S3
Key_Insurgent	K212	captured				Known by S3
Key_Insurgent	K412	captured				Known by S3
IED	I24	exploded				Known by S3
IED	I34	exploded				Known by S3
IED	I14	exploded			schoolB5	Known by S3
IED	I44	exploded			mosqueB1	Known by S3
Crowd	C112	active	40 persons	A13(-)		Known by S3
Crowd	C212	active	24 persons	A23(+)		Known by S3
Crowd	C423	active	20 persons	A44(+),A45(-)		Known by S3
Key_Insurgent	K112	high threat				Known by S3
Key_Insurgent	K312	high threat			schoolB5	Known by S3
IED	I25	active				Known by S3
IED	I35	active				Known by S3
IED	I15	active				Known by S3
IED	I45	active			schoolB5	Known by S3

S2 Outcomes

Alerts to S3		Alerts to S4			
Crowds	20/0/-20	Crowds	0/0/0	MIDB Query	78/0/-78
Key Insurgents	15/0/-15	Key Insurgents	0/0/0	Score:	
IEDs	24/0/-24	IEDs	0/0/0	Crowds	73735
				Key Insurgents	247669
				IEDs	96632

(a) S2 human-computer interface.

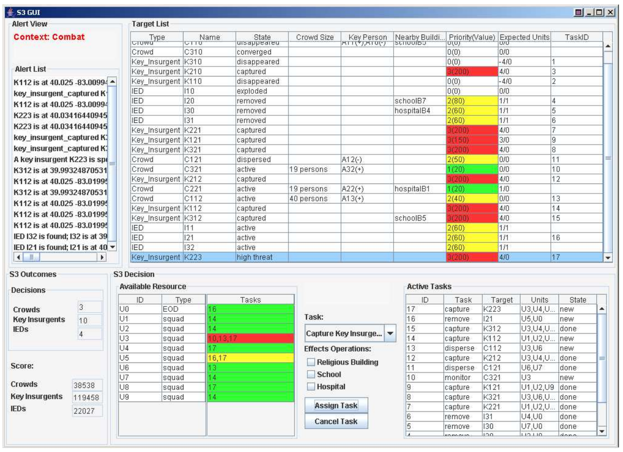


(b) Map Display (one for each operator).

Fig. 3. S2 GUI and Map Display.

list of incoming Spot reports and the detail of a report once it is selected. The upper-left area shows the current *macro* context. In the experiment, S2 human participants were asked to perform communication actions by following the current macro context as far as they could.

Shown in the list on the upper-right pane is fused information about targets, which are categorized into target type, name, status, crowd size, activists associated with crowds, nearby buildings, and whether the S3 suite has been ever alerted. S2 humans can perform two kinds of communication actions. After selecting an active target from the target list, an S2 human can (a) press the button "MIDB Query" to query the simulated Military Intelligence DataBase for additional information about the target being highlighted. Depending on the type of the selected target, from MIDB S2 humans can gather threat level information for an insurgent, recognize whether an activist with a crowd is a friend or foe, and update information about buildings nearby an active target; (b) press the button "Alert to S3" to update S3 suite with new changes regarding the selected target. (The other two buttons "Alert to S4" and "Cancel Alert to S4" were not used in this study; they are reserved for future studies when the S4 suite, called logistics cell, is considered). The bottom portion of the interface shows S2's communication statistics.


S3 GUI

Alert View

Context: Combat

Alert List

K112 is at 40.025 -83.0099
key_insurgent_captured K
K112 is at 40.025 -83.0099
K223 is at 40.03416440945
K223 is at 40.03416440945
key_insurgent_captured K
key_insurgent_captured K
A key insurgent K223 is sp
K312 is at 39.99324870531
K112 is at 40.025 -83.0199
K312 is at 39.99324870531
K112 is at 40.025 -83.0199
K112 is at 40.025 -83.0199
K112 is at 40.025 -83.0199
IED I32 is found; I32 is at 39
IED I21 is found; I21 is at 40

Target List

Type	Name	State	Crowd Size	Key Person	Nearby Buildi...	Priority(Value)	Expected Units	TaskID
Crowd	C110	disappeared		A11(+),A10(-)	schoolB5	0(0)	0/0	
Crowd	C310	converged				0(0)	0/0	
Key_Insurgent	K310	disappeared				0(0)	-4/0	1
Key_Insurgent	K210	captured				🟥 3(200)	4/0	3
Key_Insurgent	K110	disappeared				0(0)	-4/0	2
IED	I10	exploded				0(0)	0/0	
IED	I20	removed			schoolB7	🟨 2(80)	1/1	4
IED	I30	removed			hospitalB4	🟨 2(60)	1/1	5
IED	I31	removed				🟨 2(60)	1/1	6
Key_Insurgent	K221	captured				🟥 3(200)	4/0	7
Key_Insurgent	K121	captured				🟥 3(150)	3/0	9
Key_Insurgent	K321	captured				🟥 3(200)	4/0	8
Crowd	C121	dispersed		A12(-)		🟨 2(50)	0/0	11
Crowd	C321	active	19 persons	A32(+)		🟩 1(20)	0/0	10
Key_Insurgent	K212	captured				🟥 3(200)	4/0	12
Crowd	C221	active	19 persons	A22(+)	hospitalB1	🟩 1(20)	1/0	
Crowd	C112	active	40 persons	A13(+)		🟨 2(40)	0/0	13
Key_Insurgent	K112	captured				🟥 3(200)	4/0	14
Key_Insurgent	K312	captured			schoolB5	🟥 3(200)	4/0	15
IED	I11	active				🟨 2(60)	1/1	
IED	I21	active				🟨 2(60)	1/1	16
IED	I32	active				🟨 2(60)	1/1	
Key_Insurgent	K223	high threat				🟥 3(200)	4/0	17

S3 Outcomes

Decisions

Crowds	3
Key Insurgents	10
IEDs	4

Score:

Crowds	38538
Key Insurgents	119458
IEDs	22027

S3 Decision

Available Resource

ID	Type	Tasks
U0	EOD	🟩 16
U1	squad	🟩 14
U2	squad	🟩 14
U3	squad	🟥 10,13,17
U4	squad	🟩 17
U5	squad	🟨 16,17
U6	squad	🟩 13
U7	squad	🟩 14
U8	squad	🟩 17
U9	squad	🟩 14

Task: Capture Key Insurge...

Effects Operations:
☐ Religious Building
☐ School
☐ Hospital

Active Tasks

ID	Task	Target	Units	State
17	capture	K223	U3,U4,U...	new
16	remove	I21	U5,U0	new
15	capture	K312	U3,U4,U...	done
14	capture	K112	U1,U2,U...	new
13	disperse	C112	U3,U6	new
12	capture	K212	U3,U4,U...	done
11	disperse	C121	U6,U7	done
10	monitor	C321	U3	new
9	capture	K121	U1,U2,U9	done
8	capture	K321	U3,U6,U...	done
7	capture	K221	U1,U2,U...	done
6	remove	I31	U4,U0	done
5	remove	I30	U7,U0	done
4	remove	I20	U2,U0	done

Fig. 4. S3 human-computer interface.

Once the S2 operator has informed a target to the S3 suite, the S3 suite will keep being updated of the dynamic *location* information of that target afterward. However, to secure better team performance, the S2 operator has to update the S3 suite with situational changes other than locations (e.g., crowd sizes, nearby buildings) in a timely manner.

3) S2 Agent Decision Automation: For HA teams, an R-CAST agent automates solely the role of S2 suite.

As depicted in Figure 1, an R-CAST agent has a mental model, which captures the state of the current situation. For this experiment, the mental model of the S2 agent includes information about all the active targets (e.g., threat type, threat level), and the information needs of the S3 suite.

The S2 agent reacts to an incoming Spot report in the following manner.

1. The agent updates its mental model about the active targets accordingly;
2. As described in Section III-B, the S2 agent manages a decision space with three experience trees corresponding to the humanitarian, peacekeeping, and combat operations, respectively. If the new tasking context is different from the previous tasking context (e.g., from crowds to insurgents), the agent switches to the corresponding decision tree (i.e., switching *recognition context*);
3. The agent uses its *inference context* to check whether the state change of a target can impact the S3 suite's decision making on target selection and resource allocation. A significant impact can trigger the agent to take actions to share the relevant information with S3 (i.e., anchor an experience and execute the corresponding COA);
4. The agent uses its *inference context* to check whether there are any information about the active targets that is needed by S3 but is still missing (with unknown value). If so, the agent will take actions to query the MIDB and share the collected information with S3.

4) S3 Human-Computer Interface: S3 GUI is used for both HH teams and HA teams. Figure 4 shows a screenshot of S3 GUI. Shown on the left are the context indicator, a list of alerts from the S2 suite, and the S3 suite's current performance.

The bottom-right is a list of tasks issued by the S3 operator. This allows the human operator to track which unit was allocated to handle which target, and the task status (newly issued, done, impossible).

The upper-right is a target list, showing information similar to the target list on S2 GUI but with three new columns: 'Priority(value)', 'ExpectedUnits', and 'TaskID'. The TaskID column indicates whether a task is already issued for a target; more information about the task can be obtained by referring to the task list at the bottom-right. The Priority column displays the priorities of the active targets (1: low, 2: medium, 3: high) and the credit values if the corresponding threats can be cleared. The ExpectedUnits column gives the number of squad/EOD units required to handle each target. For HA teams, an S3 agent, acting as a decision aid to the S3 operator, will use color codings to highlight target priorities, and recommend the number of squad/EOD units required for each active target. For HH teams, the S3 operator has to make self judgments regarding the priorities of active targets (no color coding) and resource requirements (the 'ExpectedUnits' column is blank). The 'Available Resource' table gives all the allocable resources and their taskloads. An S3 operator can issue 4 types of operations: monitor/disperse a crowd, capture an insurgent, and remove an IED.

The S3 human can easily get overloaded when conducting resource allocation operation under time stress. We thus designed a secondary task for the S3 suite, the performance of which would be an indicator of how demanding a scenario is. In the real-world case, when a commander issues a task, he/she ought to be aware of the potential effects of the committed operations. In this experiment, when issuing a task to remove a threat, an S3 operator needs to click the checkboxes that correspond to the key buildings nearby the target being considered, if applicable.

5) S3 Agent Decision Recommendation: For HA teams, an S3 operator together with an R-CAST agent (S3 agent) plays the role of the S3 suite. To offer highly acceptable aids for S3 operators to make decisions and decision adaptation, the S3 agent is equipped with an expert recommendation model.

The recommendation model has four components. First, the S3 agent can recommend *target priorities* to the S3 human operator using color codings: red for targets with a high priority, yellow for targets with a medium priority, and green for targets with a low priority. However, the decision making task is still high demanding even with the help of color codings. For example, there may exist multiple targets with a high priority at the same time. An S3 operator still needs to make decisions on which target to handle first; this can largely impact how the situation evolves and the overall team performance. Moreover, the best target selection may not be the one with the highest priority due to the uncertainty and dynamic nature of the problem. For example, the appearance time of a target is unknown to S3 operators (the longer time a target has been there, the more likely it can disappear from the simulation); The locations of targets and friendly units keep changing as the situation evolves; There are insufficient available resources for handling a high-priority target. S3 operators need to take all such factors into consideration when deciding which target to handle.

Second, the S3 agent can remind the S3 operator about the current resource requirements of each active target (i.e., how many more units are needed). Third, the S3 agent can leverage its *inference context* to mark the appropriate "effects of operations" (i.e., key buildings nearby the selected target).

Lastly, based on the following heuristic model, the S3 agent can recommend resources to the target being highlighted to support S3 human's *resource allocation behavior*.

We use $1, 2, \cdots, n$ to denote resource types; R_i^o to denote the set of resources of type i; R_i to denote the set of *free* resources of type i; and R_i' to denote the set of *re-assignable* resources of type i. *Re-assignable* resources are those that have already been engaged in a task but can be preempted by another task. The current domain has 2 resource types: squads and EOD.

Each target places a resource requirement (cf. Table I). The resource requirement of target t is denoted by $\overline{r_t} = (r_1(t), r_2(t), \cdots, r_n(t))$, where $r_i(t)$ is the number of needs on resources of type i. For example, $\overline{r_{c121}} = (2, 0)$.

Given a target t, if $r_i(t) > |R_i|$, we need to find re-assignable resources engaged in the on-going tasks. The on-going tasks can be ordered by *priority* including tasks with priorities no more than the new task being considered. Let $target(T)$ be the target of task T, $R(T)$ be the set of resources engaged in T. The priority of a task T with respect to a target t, is determined by its *TValue*: $TValue(t, T) =$

$$\frac{TargetValue(t)}{\sum_{r_i \in \overline{r_t}} w_i \cdot r_i(t)} \times \frac{TargetRemainTime(t)}{max\{distance(t, \gamma) | \gamma \in R(T)\}}$$

where w_i is the weight of resources of type i, representing the cost of using a resource of type i, and the second factor on the right represents the chance of success when using resources engaged in T to handle target t. Here, $TargetRemainTime(t)$ represents the *estimated* measure of how long t will remain before disappearing from the simulation.

Resources engaged in task T will be considered re-assignable for t if $TValue(t, T) > \delta * TValue(target(T), T)$, where a relatively good value for the adjustable variable δ can be obtained by running pre-experiments.

Let $(T_1, T_2, \cdots, T_k)$ be a list of tasks determined in the above way relative to target t, and $(y_i^{T_1}, y_i^{T_2}, \cdots, y_i^{T_k})$ be a list, where $y_i^{T_j}$ is the set of resources of type i from task T_j. Then, resources of type i re-assignable for t is $R_i'(t) =$

$$\begin{cases} y_i^{T_1} & (r_i(t) - |R_i|) \leq |y_i^{T_1}| \\ \bigcup_{j:1..m} y_i^{T_j} & |\bigcup_{j:1..m-1} y_i^{T_j}| < (r_i(t) - |R_i|) \leq |\bigcup_{j:1..m} y_i^{T_j}| \end{cases}$$

Next, given the set $R_i(t)$ of resources of type i available for target t, resources closest to the target t ought to be selected first. Let $D = (D_1, D_2, \cdots, D_x)$, where $\{D_i : (1 \leq i \leq x)\}$ is the partition of $R_i(t)$ by the distance of resources to target t, and D_i in D are ordered by distance from near to far. That is, resources in D_i have the same distance to t and they are closer to t than those in D_{i+1}. Then, $closest(R_i(t), x) =$

$$\begin{cases} select(D_1, x) & x \leq |D_1| \\ select(\bigcup_{j:1..m} D_j, x) & |\bigcup_{j:1..m-1} D_j| < x \leq |\bigcup_{j:1..m} D_j| \end{cases}$$

which returns x number of resources closest to t, where $select(A, z)$ returns any one of A's subset of size z. Here, it doesn't matter whether to consider resources in $\bigcup_{j:1..m-1} D_j$ first, because at least one will come from D_m, which dominates the time it takes for the resources to reach t's location (we assume all the resources move at the same speed).

Now, given a target t and its resource requirement $\overline{r_t} = (r_1(t), r_2(t), \cdots, r_n(t))$, S3 agent follows the criteria below to allocate resources for target t. For each resource type i, (1) if $r_i(t) \leq |R_i|$, return $closest(R_i, r_i(t))$; (2) if $|R_i| < r_i(t) \leq |R_i \cup R'_i(t)|$, return $R_i \cup closest(R'_i(t), r_i(t) - |R_i|)$.

Actually, in cases when $r_i(t) > |R_i \cup R'_i(t)|$, two tasks can be issued separately to meet the resource requirement of t: one task employs resources in $R_i \cup R'_i(t)$, then issue another task later when more resources become available. However, this might confuse matters such that human participants may easily forget to return to finish those partially committed operations. We thus left this open: human participants were allowed to split tasks if they could manage their attention.

However, it is worth noting that the S3 agent's recommendations are relative to its knowledge of the current situation. The S2 (S3) suite updates its knowledge of the dynamic situation once it gets Spot reports from the Simulation Engine. Because in this study the Simulation Engine is controlled such that the Spot reports are sent out every 5s (10s, or 15s), there is inevitably a difference between the situational information known by the S3 agent and the real situation known by the Simulation Engine (which evaluates C^2 performance based on its knowledge). Consequently, the S3 agent's recommendations can be inappropriate or incorrect. For example, the resource allocation to a target t recommended by the S3 agent could be improved if the S3 agent had predicted that a friendly unit which is even closer to t will be free in a second. The S3 agent might make a different recommendation about the effects of operations should the agent be able to predict that the target under consideration is moving closer to a key building. The S3 agent also could make a wrong recommendation about the effects of operations if the S2 suite does not update the S3 suite with the latest nearby building information. R-CAST does not support such prediction capability; thus it is still the S3 operator's responsibility to judge whether to accept or override the recommendations from the S3 agent.

We here comment that it is not that such a prediction capability cannot be implemented, it is that we have to face the reality that artificial intelligence (agent) is not a panacea; it can fail in this way or another. This is exactly one of the benefits of human-agent teamwork: leveraging the strengths of both to achieve the best possible performance. In this experiment, being equipped with a map display, an S3 operator has a better position to predict the trajectory of a moving target; and being a team member, he/she has the responsibility of correcting the S3 agent's recommendations whenever necessary.

B. Human Subjects

In order to recruit participants with a fair degree of military C^2 knowledge, we chose members of US Army ROTC (Reserve Officer Training Corps) organization at Penn State University as the human participants.

We randomly recruited 26 ROTC students and 4 ROTC Sergeants (Sergeants were used to avoid biased results due to different levels of military C^2 knowledge. The Sergeants datapoints were further analyzed and were found to be within the range of students datapoints. Further details appear in the results section). 10 of the ROTC students were used to form 10 HA teams; 16 of the ROTC students were used to form 8 HH teams, and the 4 Sergeants were used to form another 2 HH teams. Each team was tested using the 9 scenarios reflecting the nine factorial level combinations of CSF and TC as described in Section III-C.

C. Dependent Measures

On the 180 runs of the experiment (90 runs for Human-Human teams and 90 runs for Human-Agent teams), we measured a number of response variables. For each experiment run i, we recorded n_{aKi}, n_{bKi}, and n_{cKi}— the numbers of key-insurgents captured with high, medium, and low threats respectively; n_{aDi}, n_{bDi}, and n_{cDi}—the numbers of IEDs removed with high, medium, and no threats respectively; and n_{aCi}, n_{bCi}, n_{cCi}, and n_{dCi}— the numbers of crowds dispersed with high, slightly high, medium, and low threats respectively. Let $n_{Ki} = n_{aKi} + n_{bKi} + n_{cKi}$, $n_{Di} = n_{aDi} + n_{bDi} + n_{cDi}$, $n_{Ci} = n_{aCi} + n_{bCi} + n_{cCi} + n_{dCi}$.
The Average Performance Index (API) is defined as:

$$API_i = (\sum_{X \in \{K,D,C\}} \frac{\theta_{Xi}}{n_{Xi}})/(n_{Ki} + n_{Di} + n_{Ci}),$$

where $\theta_{Ki} = 200n_{aKi} + 150n_{bKi} + 100n_{cKi}$,
$\theta_{Di} = 80n_{aDi} + 60n_{bDi} + 0n_{cDi}$, and
$\theta_{Ci} = 60n_{aCi} + 50n_{bCi} + 40n_{cCi} + 20n_{dCi}$,
where the weights in computing $\theta_{Ki}, \theta_{Di}, \theta_{Ci}$ are the credit values of targets, which are proportional to their degree of devastation and were suggested by domain experts as explained in Section III-B. The API measure reflects a team's average performance across three context (overall competency).

Another performance measure is MAR (Missed-Attention Rate), which reflects S2 and S3's joint efforts in paying attention to the effects of operations.

The summary statistics of the data collected (sample treatment means and standard deviations) is presented in Table III. The table also provides the summary statistics for variables KP, DP, and CP, which represent the ratios of key-insurgents, IEDs, and crowds successfully handled relative to the total number of key-insurgents, IEDs, and crowds that appeared, respectively. The scatterplot matrix with LOWESS lines is given in Figure 5. It clearly indicates that API is highly correlated with variables KP and CP, which confirmed our choice of API as the main performance measure in the analysis of the results.

V. Result Analysis

From Table III, we can see that the HH teams suffered from time stresses (CSF): the mean API measure dropped from 4.748, to 3.208, then to 2.015 as the CSF changed from L, to M, to H (the measure of MAR has the opposite pattern). This supports other cognitive studies [24], [25] in general, because

TABLE III
SUMMARY STATISTICS: MEANS AND STANDARD DEVIATIONS OF RESPONSES FOR EACH TREATMENT

					HH					*HA*		
TC	*CSF*		KP	DP	CP	API	MAR	KP	DP	CP	API	MAR
L	H	Mean	0.555	0.3003	0.1333	3.324	0.6441	0.7538	0.4184	0.1833	4.460	0.4859
		StDev	0.324	0.1190	0.1315	1.565	0.1753	0.1975	0.1191	0.0861	1.022	0.0630
M	H	Mean	0.4508	0.1704	0.0500	1.500	0.571	0.6991	0.3529	0.0944	2.523	0.2539
		StDev	0.2339	0.0904	0.0611	0.771	0.321	0.1673	0.1291	0.0743	0.611	0.1623
H	H	Mean	0.4545	0.1907	0.0458	1.222	0.6067	0.5661	0.2389	0.0886	1.5173	0.3786
		StDev	0.2111	0.0611	0.0604	0.470	0.2002	0.1283	0.0634	0.0746	0.2176	0.1320
	Team mean (H)					*2.015*	*0.6074*				*2.833*	*0.3728*
L	M	Mean	0.6553	0.2456	0.2206	4.521	0.477	0.8260	0.4546	0.1886	5.480	0.2067
		StDev	0.2514	0.1394	0.1989	1.305	0.401	0.1445	0.0997	0.1005	0.844	0.2477
M	M	Mean	0.6097	0.1567	0.1378	3.163	0.5330	0.7730	0.1817	0.1488	3.779	0.2322
		StDev	0.1965	0.0755	0.1087	0.940	0.3008	0.1065	0.0902	0.1203	0.666	0.1747
H	M	Mean	0.5483	0.1090	0.0547	1.940	0.542	0.6948	0.1984	0.0810	2.329	0.4404
		StDev	0.2690	0.0746	0.0649	0.828	0.344	0.1362	0.0752	0.0792	0.421	0.0973
	Team mean (M)					*3.208*	*0.5172*				*3.863*	*0.2931*
L	L	Mean	0.7310	0.1289	0.2685	6.893	0.333	0.8917	0.1787	0.3324	8.689	0.2517
		StDev	0.3003	0.1294	0.1947	1.977	0.333	0.1006	0.1204	0.1232	1.308	0.1976
M	L	Mean	0.6339	0.0671	0.2022	4.136	0.7369	0.7506	0.1747	0.2564	5.017	0.2952
		StDev	0.2451	0.0523	0.1792	1.464	0.2630	0.1011	0.0656	0.1123	0.732	0.1725
H	L	Mean	0.6500	0.0708	0.1960	3.215	0.471	0.7624	0.0896	0.1682	3.470	0.326
		StDev	0.2104	0.0658	0.1206	0.968	0.351	0.1229	0.0743	0.1454	0.800	0.317
	Team mean (L)					*4.748*	*0.5137*				*5.725*	*0.2911*

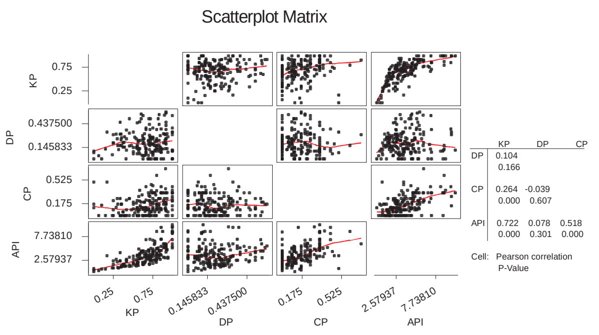


Fig. 5. Scatterplot Matrix: API is highly correlated with KP and CP.

both S2 and S3 participants have limits to their cognitive capacities. While monitoring the situation development from the Map Display, at the same time S2 participants need to process Spot reports under time stress, to recognize/ gather missing information, and to constantly make decisions on *when* to share information with *whom* about *what*. Similarly, S3 participants, while being limited by resources and multi-tasking capacity, need to constantly monitor the status of active targets, to decide *when* and *how* to handle *which* target (prioritization, resource (re-)allocation), and to judge whether to cancel an ongoing task if a situation changes.

From the data collected, we also found that the performance of the two ROTC-Sergeant teams were not the best among the 10 HH teams. This removed our concern that the results could be biased if the ROTC students were not representatives of the population with military C^2 experiences. This provides a level of confidence that at least as far as this particular experiment is concerned, ROTC students can be taken as having a fair degree of military C^2 knowledge.

From Table III, the mean API measure for HA teams also dropped from 5.725, to 3.863, then to 2.833 as the CSF changed from L, to M, to H (the measure of MAR has the opposite pattern). It also clearly indicates that, compared with HH C^2 teams, the average performance of HA C^2 teams has been improved at each CSF level.

We next examine whether the performance improvement is statistically significant and assess interaction effects of treatment factors, if any.

A. Statistical Inference: Model Selection

To conduct statistical inference, one key issue focuses around the proper selection of statistical models.

Model selection for this study is complicated for two reasons. First, due to the restricted availability of the pool of participants, the experiment design resulted in nested sampling of experiment units and repeated measures of performance. The ten HH teams used in the experiments are different from the ten HA teams (we had 9 sets of measures for each team); hence in the ANOVA model we should have a "team" variable nested within the team type (TType) variable.

Second, we had repeated measures because each team performed in the experiment for all the nine level combinations of CSF-TC. Typically, there should be a "washout" time between repeated measures to avoid carryover effects. However, due to the participants' limited schedules the teams had to finish all 9 treatments one after another. Thus, we have to consider such anomaly of repeated measures in model selection. However, the complexity of dealing with repeated measures can be avoided if it can be shown that there is no presence of autocorrelation in adjacent observations. The Durbin-Watson statistic [26] can be used to test for the presence of autocorrelation in residuals of time series data. To be sure of the randomness of the observations from the same team, we also use the runs test to check whether the residuals are independent. If both the Durbin-Watson test and the runs test confirm that the residuals are statistically independent, we can proceed as though there is no repeated measure in the observations.

Furthermore, we used ten randomly formed HH teams and ten randomly formed HA teams. However, the specific teams that participated in the experiment are not of interest to us. We hope the inference from the data are not only for the teams used, but for all the possible teams in the populations. So, below we will use mixed-effects model, with factors TType, CSF, and TC having fixed levels, and the factor 'team' having randomly selected levels.

B. The Primary Performance Measure: API

We first use the Durbin-Watson test and the runs test to check whether the residuals of the API measures in time series are statistically independent. Table IV gives the results of the Durbin-Watson test and the runs test, where both tests suggest that there exists no serial correlation except for HH team 2. We thus can assume the residuals are statistically independent and build our model as though there are no repeated measures. We use the following ANOVA model:

$$\begin{aligned} y_{ijmnk} = \mu + \alpha_i + \beta_j + \gamma_m + \delta_{n(i)} + \alpha\beta_{ij} + \alpha\gamma_{im} + \\ \beta\gamma_{jm} + \beta\delta_{jn(i)} + \gamma\delta_{mn(i)} + \alpha\beta\gamma_{ijm} + \varepsilon_{ijmnk}, \end{aligned} \quad (1)$$

where, y_{ijmnk}: the response,
μ: Overall mean, an unknown constant,
α_i: An effect due to the ith level of factor TType,
β_j: An effect due to the jth level of factor CSF,
γ_m: An effect due to the mth level of factor TC,
$\delta_{n(j)}$: The nth level of factor team is nested in the ith level of TType,
$\alpha\beta_{ij}$: An interaction effect of the ith level of TType with the jth level of CSF,
$\alpha\gamma_{im}$: An interaction effect of the ith level of TType with the mth level of TC,
$\beta\gamma_{jm}$: An interaction effect of the jth level of CSF with the mth level of TC,
$\beta\delta_{jn(i)}$: An interaction effect of the jth level of CSF with the nth level of team nested in the ith level of TType,
$\gamma\delta_{mn(i)}$: An interaction effect of the mth level of TC with the nth level of team nested in the ith level of TType,
$\alpha\beta\gamma_{ijm}$: Three-way interactions,
ε_{ijmnk}: A random error associated with the response from the kth replication receiving the ith, jth, mth, and nth level of TType, CSF, TC, and team.

We next need to verify the assumptions of ANOVA are at least roughly satisfied in order to avoid misleading results. In particular, we should check (1) We have independent and random samples from all groups, (2) The samples are taken from Normal distributions, and (3) The variances of the different groups are all equal. Assumption (1) is met because participants were randomly assigned to either be in HH or HA teams, and team members are randomly assigned. Assumption (2) can be checked using normal probability plot of residuals, and assumption (3) can be checked by plotting the residual errors verse their fitted values. Figure 6(a,b) gives the plots after fitting the API measures to Model (1). The normal probability plot of residuals of API is roughly a line, and the plot of the residual errors versus their fitted values seems to have a constant vertical spread. We thus can proceed as if the assumptions are all met. Table V gives the ANOVA output for the API response provided by Minitab.

TABLE V
ANOVA TABLE FOR API (AVERAGE PERFORMANCE INDEX)

Source	**DF**	**SS**	**MS**	**F**	**P**
TType	1	30.0227	30.0227	5.99	0.025
CSF	2	240.7284	120.3642	169.44	0.000
TC	2	335.4676	167.7338	247.78	0.000
Team(TType)	18	90.1490	5.0083	N/A	N/A
*TType*CSF*	2	0.7792	0.3896	0.55	0.583
*TType*TC*	2	7.2640	3.6320	5.37	0.009
*CSF*TC*	4	26.4677	6.6169	14.41	0.000
*CSF*Team(TType)*	36	25.5736	0.7104	1.55	0.058
*TC*Team(TType)*	36	24.3696	0.6769	1.47	0.081
*TType*CSF*TC*	4	1.6401	0.4100	0.89	0.473
Error	72	33.0507	0.4590		
Total	179	815.5125			

TABLE VI
MEANS OF API PERFORMANCE BY TC×CSF

4H	3H	4M	4L	3M	2H	3L	2M	2L
1.37	2.01	2.13	3.34	3.47	3.89	4.58	5.00	7.79

Significant difference exists when difference $\geq 4.47\sqrt{\frac{1.075}{20}} \approx 1.036$.

The ANOVA output indicates that there are two-way interactions between CSF levels and TC levels, and between TType levels and TC levels. Figures 7(a-c) plot the interaction effects of TType*CSF, TType*TC, and CSF*TC, respectively. There is no significant TType*CSF interactions, as can be confirmed by the roughly parallel lines in Fig. 7(a). However, it does indicate that HA teams performed significantly better than HH teams at each CSF level. This suggests that cognitive agents can play a critical role in alleviating the impact of human's cognitive capacity on the performance of decision making involving multiple contexts. Fig. 7(a) also shows a performance drop for both HH and HA teams as the context switching frequency increased: both suffered more under more time-stressed situations. Fig. 7(b) shows that the performance of both HH teams and HA teams were affected considerably by task complexity, but the slope for HH teams is less than the HA teams. It suggests that HA teams are more sensitive to the changes of tasking situations.

The Tukey's Honest Significant Difference (HSD) procedure [26] can be employed to answer the third research problem (patterns of difference of C^2 team performance at different level combinations). Because there are insignificant terms in Table V, by dropping insignificant terms and pooling, we can perform a conservative analysis using a new model:

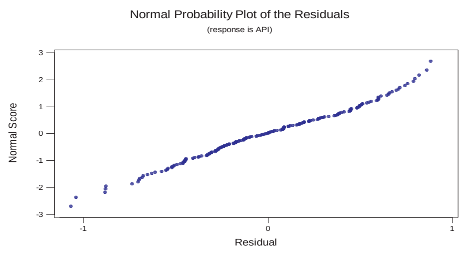
$$y_{ijk} = \mu + \alpha_i + \pi_j + \varepsilon_{ijk}, \quad (2)$$

where α_i is the effect due to the ith level of TType, and π_j is the effect due to the jth level of Scenario (a variable that has 9 levels, corresponding to the 9 combinations of CSF-TC).

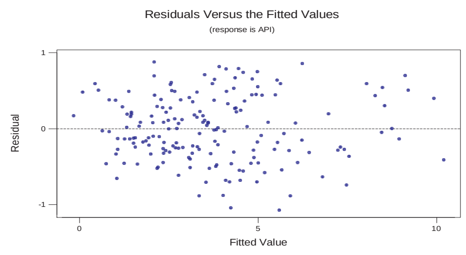
It is checked that the assumptions of ANOVA were not violated for this new model. The ANOVA output shows that the Scenario variable is highly significant ($n = 20, MSE = 1.075$, p-value < 0.001). The means of the nine levels are given ascendingly in Table VI. If the mean difference of two

TABLE IV
SERIAL CORRELATION TEST FOR API: NO APPARENT SERIAL CORRELATION IF $1.5 \leq d \leq 2.5$ OR $p < 0.05$

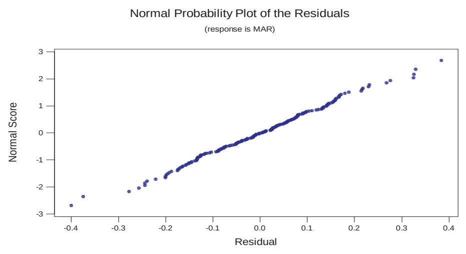
Statistic	*TType*	*Team1*	*Team2*	*Team3*	*Team4*	*Team5*	*Team6*	*Team7*	*Team8*	*Team9*	*Team10*	*Average*
Durbin-Watson test d	HH	3.14	3.12	2.21	1.85	2.32	2.06	2.17	2.38	1.64	2.35	2.33
	HA	2.19	2.49	2.34	2.17	2.01	2.26	2.74	1.83	2.02	1.90	2.11
Runs test (p-value)	HH	0.4142	0.0128	0.0128	0.688	0.688	0.4142	0.0128	0.688	0.688	0.688	0.688
	HA	0.688	0.4142	0.688	0.688	0.688	0.0128	0.688	0.688	0.4142	0.4142	0.688



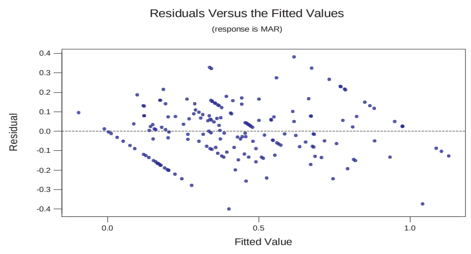
(a)



(b)

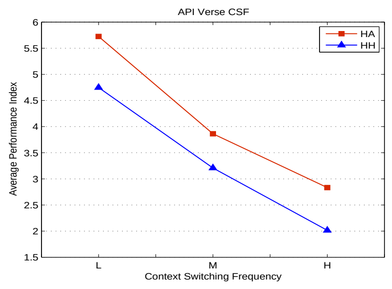


(c)

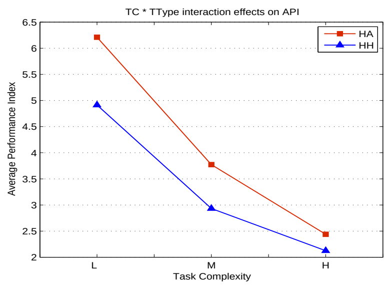


(d)

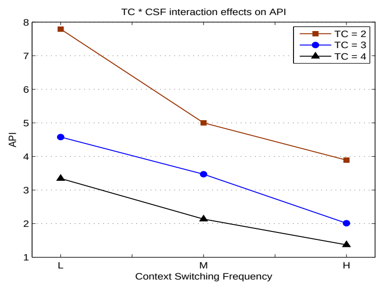
Fig. 6. (a) Residuals of API follow a normal distribution; (b) Residuals of API vs. fitted values have a constant vertical spread; (c) Residuals of MAR follow a normal distribution; and, (d) Residuals of MAR vs. fitted values have a constant vertical spread.



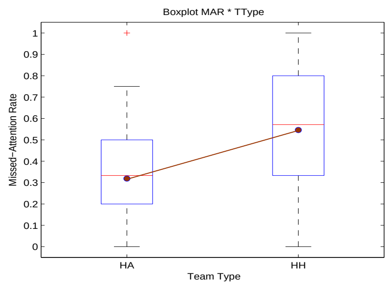
(a)



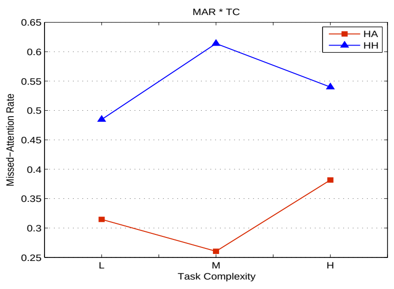
(b)



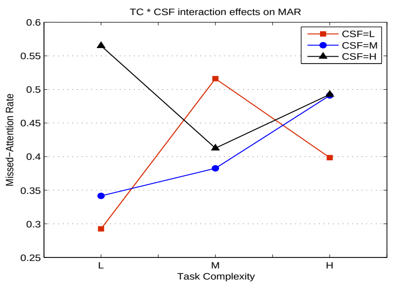
(c)



(d)



(e)



(f)

Fig. 7. (a) API verse CSF; (b) Interaction effects of TC*TType on API; (c) Interaction effects of TC*CSF; (d) Boxplot of MAR by TType; (e) Interaction effects of TC*TType on MAR; and, (f) Interaction effects of TC*CSF on MAR.

TABLE VII
ANOVA TABLE FOR MAR (MISSED-ATTENTION RATE)

Source	DF	SS	MS	F	P
TType	1	2.32135	2.32135	24.05	0.000
CSF	2	0.29849	0.14925	1.45	0.249
TC	2	0.11370	0.05685	1.15	0.329
Team(TType)	18	1.73753	0.09653	N/A	N/A
*TType*CSF*	2	0.00128	0.00064	0.01	0.994
*TType*TC*	2	0.35968	0.17984	3.63	0.037
*CSF*TC*	4	0.85755	0.21439	4.75	0.002
*CSF*Team(TType)*	36	3.71780	0.10327	2.29	0.001
*TC*Team(TType)*	36	1.78252	0.04951	1.10	0.362
*TType*CSF*TC*	4	0.18898	0.04724	1.05	0.389
Error	72	3.24932	0.04513		
Total	179	14.62819			

levels is greater than 1.036, we can say the corresponding population means are significantly different. In conclusion, we have that the performance on (4H, 3H, 4M) is significantly worse than (4L, 3M, 2H, 3L, 2M, 2L). Significant difference also includes (4L, 3M) <3L<2L, and 2H <(2M, 2L).

C. Performance of the Secondary Task: MAR

Using similar inference procedures, we analyze the performance measures on Missed-Attention Rate. We expect to get results consistent to the analysis of API, because the two measures are negatively correlated (the Pearson correlation of API and MAR is -0.233 with P-Value = 0.002).

The ANOVA output for MAR is given in Table VII. Figures 7(d-f) plot the team type difference, the interaction effects of TType*TC, and CSF*TC, respectively. Clearly, HA teams missed significant less attentions than HH teams. The interaction effect of TType*TC in Fig. 7(e) indicates that when task complexity switched from L to M, the contribution of agents (HA teams) in reducing MAR became more salient. In other words, C^2 teams could benefit more from RPD-agents under medium task complexity. The potential reason is that S3 participants in HH teams have to balance their attention. As the situations became more demanding (with more active targets), less attention was paid to the effects of operations, which led to increased MAR.

However, it becomes more interesting when task complexity changed from M to H: HH teams' MAR decreased while HA teams' MAR increased. The main reason, we believe, is that S3 human participants relied too much on the recommendations from agents rather than on their own judgment of the potential evolvement of the situation. Because of the nature of the simulation, Spot reports were available every certain time interval (15, 10, 5 sec). Thus, the S3 suite had to consider the latency of target location information in determining the appropriate effects of operations. S3 humans in HH teams relied on themselves to judge the movement of a target, which largely compensated for the latency of information and improved their performance in identifying the effects of operations. However, S3 humans in HA teams tended to rely too much on the recommendations from agents, which could be wrong because S3 agents in HA teams, unknown to their human peers, made recommendations without considering the latency of information. The issue manifested itself when there were too many active targets (task complexity is H). This to some extent indicates that our task design is useful for investigating human-agent collaboration issues, and suggests that the best possible human-agent team performance can only be achieved when both human and agents contribute their best.

VI. CONCLUSION

Multi-context real-time decision making is an extremely challenging problem faced by various real-world application domains. It can be very complex especially when human factors and computing technologies are mutually constraining in their interaction.

We have started to address this challenge associated with C^2 teams operating in complex urban environments, using the R-CAST cognitive agent architecture as human operators' teammates and decision aids. This paper, focused on the coupling of cognitive agent technology and human-centered teamwork, reported our experimental studies about the impact of RPD-enabled agents on C^2 teams' performance in multi-context decision making under time stress. It has been shown in the experiment that the performance of Human-Agent teams was significantly better than pure human teams, and different level combinations of CSF and TC had significant effects on C2 team performance. Also, as far as the average performance index (API) was concerned, the CSF and TC factors had significant effects on the performance of C2 teams.

The experiment represents an important step forward in uncovering the nature of real-world problems, because the environment simulated the real domain and the human participants were recruited from Army ROTC students. This study suggests that C^2 team performance, while still limited by human cognitive capacity, could be largely improved when human operators are assisted by cognitive agents empowered with expert experiences and proactive information gathering/sharing capabilities.

Intelligence analysts need tools and techniques to help protect themselves from avoidable errors [27], [28]. Some human "errors" rest on their fundamental cognitive capacities. Their performance cannot be improved by simply providing a better human user interface without employing appropriate technologies as cognitive aids. Our experiment demonstrated that RPD-enabled agents, to some extent, can serve as one such tool to achieve reduced cognitive load, enhanced situation awareness, and positive human-agent collaboration.

In real-world applications where information availability and credibility is concerned, it is expected that C^2 teams with both the S2 and the S3 operators assisted by an R-CAST agent could outperform either of the two team structures considered in this paper, because S2 human's ability to reason about imperfect information can be fully exploited. This is worthwhile to consider in future studies.

ACKNOWLEDGMENT

The authors would like to thank John Dumer, Holly Ingham and Mark Mittrick from US Army Research Lab at Aberdeen Proving Ground for their help in designing C2 tasks and the supports on SA Map Displays. We are also grateful to the anonymous reviewers for their constructive comments.

References

[1] E. Horvitz and M. Barry, “Display of information for time-critical decision making,” in *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1995, pp. 296–305.

[2] J. A. Cannon-Bowers and E. E. Salas, “Individual and team decision making under stress: Theoretical underpinnings,” in *Making decisions under stress: Implications for individual and team training*, J. A. Cannon-Bowers and E. Salas, Eds. APA Press, 1998, pp. 17–38.

[3] S. Noh and P. Gmytrasiewicz, “Flexible multi-agent decision making under time pressure,” *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 5, pp. 697–707, 2005.

[4] D. Georgiev, P. T. Kabamba, and D. M. Tilbury, “A new model for team optimization: The effects of uncertainty on interaction,” *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 38, no. 6, pp. 1234–1247, 2008.

[5] X. Fan, S. Sun, B. Sun, G. Airy, M. McNeese, and J. Yen, “Collaborative RPD-enabled agents assisting the three-block challenge in C2CUT,” in *BRIMS’05: Proceedings of the 2005 Conference on Behavior Representation in Modeling and Simulation*, 2005, pp. 113–123.

[6] X. Fan, S. Sun, M. McNeese, and J. Yen, “Extending the recognition-primed decision model to support human-agent collaboration,” in *AAMAS ’05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. ACM Press, 2005, pp. 945–952.

[7] E. Norling, “Folk psychology for human modelling: Extending the BDI paradigm,” in *AAMAS ’04: International Conference on Autonomous Agents and Multi Agent Systems*, 2004, pp. 202–209.

[8] E. Norling, L. Sonenberg, and R. Ronnquist, “Enhancing multi-agent based simulation with human-like decision making strategies,” in *Proceedings of the Second International Workshop on Multi-Agent Based Simulation*, S. Moss and P. Davidsson, Eds., 2000, pp. 214–228.

[9] J. Sokolowski, “Enhanced military decision modeling using a multiagent system approach,” in *BRIMS’03: Proceedings of the Twelfth Conference on Behavior Representation in Modeling and Simulation*, 2003, pp. 179–186.

[10] W. Warwick, S. McIlwaine, R. Hutton, and P. McDermott, “Developing computational models of recognition-primed decision making,” in *Proceedings of the tenth conference on Computer Generated Forces*, 2001, pp. 232–331.

[11] G. A. Klein, “Recognition-primed decisions,” in *Advances in man-machine systems research*, W. B. Rouse, Ed. Greenwich, CT: JAI Press, 1989, vol. 5, pp. 47–92.

[12] ——, “Recognition-primed decision making,” in *Sources of power: How people make decisions*. MIT Press, 1998, pp. 15–30.

[13] P. R. Cohen and H. J. Levesque, “Teamwork,” *Nous*, vol. 25, no. 4, pp. 487–512, 1991.

[14] M. Endsley, “Toward a theory of situation awareness in dynamic systems,” *Human Factors*, vol. 37, pp. 32–64, 1995.

[15] J. M. Bradshaw, M. Sierhuis, A. Acquisti, P. Feltovich, R. Hoffman, R. Jeffers, D. Prescott, N. Suri, A. Uszok, and R. Van Hoof, “Adjustable autonomy and human-agent teamwork in practice: An interim report on space applications,” in *Agent Autonomy*, H. Hexmoor, R. Falcone, and C. Castelfranchi, Eds. Kluwer, 2003, pp. 243–280.

[16] T. L. Lennox, T. Payne, S. K. Hahn, M. Lewis, and K. Sycara, “MokSAF: How should we support teamwork in human-agent teams?” Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-99-31, September 1999.

[17] S. Slade, “Case-based reasoning: A research paradigm,” *AI Magazine*, vol. 12, no. 1, pp. 42–55, 1991.

[18] A. Aamodt and E. Plaza, “Case-based reasoning: Foundational issues, methodological variations, and system approaches,” *AI Communications*, vol. 7, no. 1, pp. 39–59, 1994.

[19] A. Krause, “Learning and herding using case-based decisions with local interactions,” *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 39, no. 3, pp. 662–669, 2009.

[20] J. A. Cannon-Bowers, E. Salas, and S. Converse, “Cognitive psychology and team training: Training shared mental models and complex systems,” *Human Factors Society Bulletin*, vol. 33, pp. 1–4, 1990.

[21] X. Fan, J. Yen, and R. A. Volz, “A theoretical framework on proactive information exchange in agent teamwork,” *Artificial Intelligence*, vol. 169, pp. 23–97, 2005.

[22] X. Fan, R. Wang, S. Sun, J. Yen, and R. A. Volz, “Context-centric needs anticipation using information needs graphs,” *Applied Intelligence*, vol. 24, pp. 75–89, 2006.

[23] X. Fan, J. Yen, M. Miller, T. Ioerger, and R. A. Volz, “MALLET:a multi-agent logic language for encoding teamwork,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 1, pp. 123–138, 2006.

[24] C. D. Wickens, “Multiple resources and performance prediction,” *Theoretical Issues in Ergonomics Science*, vol. 3, no. 2, pp. 159–177, 2002.

[25] G. Kuk, M. Arnold, and F. Ritter, “Effects of light and heavy workload on air traffic tactical operations: a hazard rate model,” *Ergonomics*, vol. 42, no. 9, pp. 1133–1148, 1999.

[26] R. L. Ott and M. T. Longnecker, *An Introduction to Statistical Methods and Data Analysis*. Duxbury Press, 2001.

[27] A. Tversky and D. Kahneman, “Judgement under uncertainty: Heuristics and biases,” *Science*, vol. 185, pp. 1124–1131, September 1974.

[28] R. J. Heuer, *Psychology of Intelligence Analysis*. Center for the Study of Intelligence, 1999.

Xiaocong Fan is currently an Assistant Professor of Electrical and Computer Engineering at the Behrend College, The Pennsylvania State University. He received his Ph.D. degree in Software Engineering from Nanjing University, China in 1999. He previously conducted his Post-Doctoral research at the Pennsylvania State University (2002-2007), and at the Abo Akademi University, Finland (2000-2002). He recently focuses his research on Human-Centered Computing, Multi-agent systems, Cognitive modeling, Knowledge Management, and Service-oriented architectures. He is a Senior Member of IEEE.

Machael McNeese is currently a Professor of Information Sciences and Technology at the Pennsylvania State University. He received his Ph.D. in Cognitive Science from Vanderbilt University. McNeese studies human interaction with information technology in complex environments, particularly collaborative systems that bring together the confluences of cognition, computation, collaboration, and context for given fields of practice - deriving advanced human-computer interfaces through innovations of affective computing, artificial intelligence, and computer-supported cooperative work perspectives.

Bingjun Sun received his Ph.D. in Computer Science and Engineering from the Pennsylvania State University in 2008. His main research interests include data mining, search engines, and graph information retrieval.

Timothy Hanratty is a senior computer scientist at the US Army Research Laboratory. His professional interest focuses on applied research and development in the areas of knowledge engineering and advance decision and collaborative technologies. Hanratty received his BS in Computer Science from Towson University and his MS from Johns Hopkins University.

Laurel Allender is currently the Director of the U.S. Army Research Laboratory’s Human Research and Engineering Directorate (ARL-HRED). She received her Ph.D. in psychology from Rice University in 1987. Her recent research thrusts include research and development of human behavior representations, cognitive neuroscience methods, human-robot-system-team interaction, and secure mobility issues for vehicle-mounted crews. She is a member of the Human Factors and Ergonomics Society (HFES), currently is Co-Chair of the Conference on Behavior Representation in Modeling and Simulation (BRIMS).

John Yen is currently the University Professor of Information Sciences and Technology at the Pennsylvania State University. He received his Ph.D. in Computer Science from the University of California, Berkeley in 1986. Before joining Penn State in 2001, he was a Professor of Computer Science at Texas A&M University and the Director of Center for Fuzzy Logic and Intelligent Systems Research. His research has focused on developing theories and architectures for multi-agent systems to support information sharing and decision makings of human teams. He is a member of Editorial Board of several international journals on intelligent systems. He received an NSF Young Investigator Award in 1992 and he is a Fellow of IEEE.