Discretizing and Managing the Task Environment

Braulio Coronado, Crisrael Lucero, Dr. Douglas S. Lange
Space and Naval Warfare Systems Center - Pacific
San Diego, California
UNITED STATES OF AMERICA

{braulio.coronado, crisrael.lucero, doug.lange}@navy.mil

ABSTRACT

Partners usually need to know what the other partners are doing. Players on football teams, executives in a company, pilot and co-pilot must know what tasks their teammates are taking responsibility for, so they can each do their part. Even when we say that two agents are working together on a task, one can decompose their roles. For instance, if two people are carrying a large plank, one has the job of carrying the right side of the plank and the other takes the task of carrying the left side of the plank. But the two know the other individual has the other task. As humans we know the task assignments through communication (e.g., "I've got it"), doctrine, visual cues, etc. We don't have a direct connection into another human's mind in order to see their task queue, but we might get to look at their to-do list on their office white board. We then keep a model of our partners' task lists in our mind, on a piece of paper, or by some other means. How will autonomous agents know what their human partners are doing, and how will we communicate task lists and negotiate divisions of labor with such agents? While there is research being conducted on activity recognition through robotic vision, this is not an approach that will work for all human-autonomy teaming (HAT) environments.

1.0 INTRODUCTION

The evolution of Command and Control (C2) systems to match a perpetually evolving battlespace provides several challenges for human operators. The introduction of Artificial Intelligence (AI) and autonomous unmanned vehicles increases C2 system capabilities by offloading work from operators, but it is difficult to have complete trust in fully autonomous systems within the context of military operations. Additionally, increased use of autonomy and AI can abstract too much information from human operators, leaving them less engaged in certain mission critical areas of the battlespace. However, without autonomy, C2 operators can experience stressful workloads of processing information, managing tasks, and accomplishing goals in a timely manner.

Successful HAT systems involve a level of interactivity and information-sharing to keep the operator engaged while also balancing workload and stress. Within the context of task management in a C2 environment, this would involve discretizing and digitizing the task environment to allow tasks to be understood by computers and not just humans. The computer-based agents would, in turn, model human performance, track urgency and importance, balance workloads, provide visibility and control, and have the ability to take control when permissible.

In this paper we discuss an approach for discretizing the task environment to be understood by humans and machine agents. We then demonstrate different task management techniques to autonomously aid the operator. Our system also practices human-in-the-loop and human-on-the-loop principles as to not remove the operator from the equation in order to build trust through human-autonomy teaming.

2.0 BACKGROUND AND RELATED WORK

Successful Intelligent Decision Support (IDS) systems contain high-level characteristics that make them useful for humans [1]. We intend to implement some principles of successful human-autonomy collaboration systems into our task management agent. *Interactivity* is important so that the system can work well with other databases, systems, and human users; preferably, the agent would explore the space of possibilities as opposed to just the optimal solution. In this fashion, our task management agent must be able to represent new tasks in a human-centered way while also having the capability of processing requests from other agents in a multitude of different structures. *Event and Change Detection* is necessary to effectively communicate important activities. The task management agent can respond to tasks immediately and switch from a human-in-the-loop to a human-on-the-loop approach in order to avoid potential mission failures. *Predictive Capabilities* are used for predicting the effect of actions on future performance. The task management agent will be able to predict both the workload balance of the user as well as urgency and importance of tasks.

Cognitive assistants in support of task management is being investigated for different military and administrative tasks [2]. This work describes implications for designing a task manager list. *Time constraints should be captured* in order to determine urgency, and provide a clear idea of the overall picture. A task manager should *offer more benefits than just task reminders*, which we expand upon by using an autonomous agent. *Entire tasks can be viewed, but with different perspectives for different kinds of planning.* Our task management agent is capable of showing tasks to be done, but also allows different approaches and plans to approach each task.

The Intelligent Multi-UxV Planner with Adaptive Collaborative Control Technologies (IMPACT) system is a prototype C2 platform for centralized supervised control of simulated autonomous unmanned vehicles [3, 4, 5]. The underlying goal of the IMPACT system is to allow an inversion of the unmanned vehicle staffing ratio and allow a small number of human operators to control a large number of unmanned heterogenous assets. IMPACT uses a "playbook" approach, in which an operator calls a "play" to task teams of vehicles with disparate mission objectives [3]. We implemented our agent, known as the Task Manager, into the IMPACT system to practice discretizing and managing the task environment.

3.0 APPROACH

Providing autonomous assistance for task management requires a few fundamental capabilities:

1. Digitizing the task environment. Tasks not held explicitly in digital form are difficult to manage by computer-based agents. Many tasks are now held only within an operator’s mind.

2. Modeling the performance of agents. This includes building models of human performance on tasks.

3. Tracking the urgency and importance of tasks as the operational situation changes.

4. Balancing workloads to prevent under- and over-loading of agents.

5. Providing visibility and control over the task management process to human operators to act as the ultimate authority. This is done through working agreements with the automation as well as through the user interface.

6. Links to tools for execution of tasks to allow operators to perform tasks from the task manager allowing it to better track completion and performance.

4.0 IMPLEMENTATION IN ALLIED IMPACT

A task manager that begins to address all of the key capabilities described under the approach above was developed and utilized as part of IMPACT. In this section, we discuss the features representing steps along the way to the key capabilities described above.

The Allied IMPACT (AIM) system is a joint international effort and the next evolution of IMPACT. The AIM system integrates core technologies from IMPACT with several autonomous systems from the United States, United Kingdom, Australia, and Canada. Some international systems working directly with our task manager are the COMPACT [6] policy checking system and the Narrative virtual human [7].

The allied capabilities were integrated in preparation for Autonomous Warrior 2018. The goal of the task manager is to aid the operator by accomplishing the fundamental capabilities addressed above within the context of a C2 application controlling several autonomous unmanned vehicles. The AIM system also has other agents working on route planning, resource allocation, constraint solving, and capability enhancements for the operator. The task manager aids with high level tasking, as the vehicles have platform autonomy and abstracts maneuvering from the operator. Due to a high volume of tasks, tasking can easily become overwhelming and disorganized, leading the user to become frustrated and less effective in completing assigned tasks. The task manager seeks to address these issues by determining the user tasks, dividing tasks into a hierarchy, presenting the tasks to the user, and providing a mechanism for their execution [8].

Figure 1: The Task Manager interface showcasing current supervisory tasks.

4.1 Digitizing the task environment

A human operator will have goals and objectives depending on the tactical situation while performing supervisory C2 in AIM. For instance, a scenario in AIM may involve an objective to provide base defense. In this case, the operator's goal is to manage the autonomous assets in order to protect the base. Tasks in the operator's mind may be to distribute assets throughout the environment in order to maintain coverage of the base as a whole, or to execute periodic patrols of specific critical areas of the base. These tasks are known only to the operator, and unless they are written down or recorded in some fashion there is no way of tracking their progress. AIM features a chat messaging system that allows operators to communicate with other operators or commanders. An operator controlling sensors may notice an important event that should be looked at more closely by the operator performing C2, and will send a chat message with information on the event. Similarly, a commander with information not available to the C2 operator may direct the operator to perform a high-level supervisory task. These messages are read, understood and added to a checklist in the operator's mind for a task to be performed later. AIM features agents that can evaluate the tactical situation in real time, through policy checks and environment monitoring to communicate important events that may require operator intervention. In general, supervisory tasking for the C2 operator is realized through these two methods; chat messages, and changes in the operational situation through events. We employ a method to discretize tasks into digital form that can be understood by both the human and the autonomy, and discuss the different ways that supervisory tasks are generated by the task manager.

4.1.1 Task structure and generation

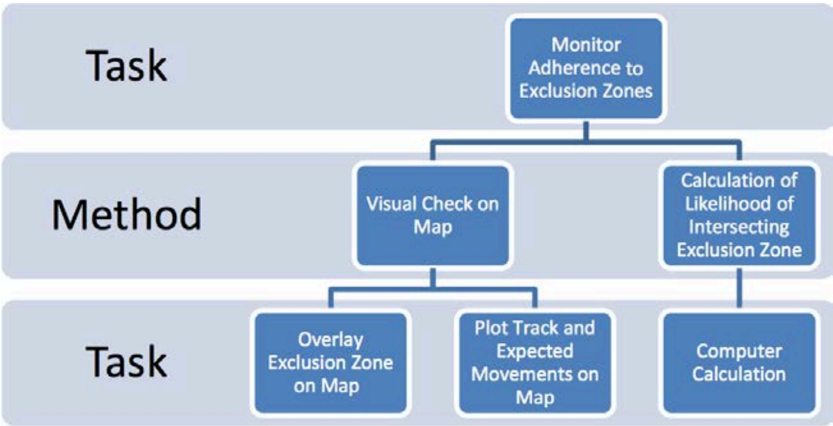

Figure 2: Directed Acyclic Graph (DAG) showcasing the bipartite nature of task method structure. From [8].

Our approach towards discretizing tasks uses a bipartite DAG to decompose tasks into methods, which are in turn composed of tasks, and can be applied recursively [8, 9]. This approach allows us to decompose tasks into a hierarchy of subtasks, where a subtask can be a simple action to perform; and to categorize tasks by the suitability for execution by either the human or the autonomy. Methods such as a visual check on a map are easily performed by a human, while data analytic methods like calculating the probability of a vehicle intersecting a restricted area is more suitable for the autonomy. By decomposing tasks in this way, we can track progress of a task in segments, with completion of each subtask essentially a check on a checklist towards completion of the task as a whole. There are two methods for generating tasks in the task manager; through chat analysis, and through event response.

4.1.1.1 Chat Analysis

An AIM setup will have a C2 operator manning a station that can communicate with other stations; e.g., a sensor operator (SO) station with an AIM configuration specifically tailored towards monitoring vehicle sensor feeds. A SO can control vehicle cameras, switch camera modes, and change camera zoom among other useful methods towards improving quality of surveillance. A SO can then relay information that may be of tactical use for the C2 operator. For example, a SO could notice an area that may be experiencing unexpected activity; and could communicate that information to the C2 operator for action. A commander can communicate with the C2 operator to provide critical data that is normally not obtainable through the operator's tool suite, or to provide supervisory tasking. In both cases, this communication can be done through the AIM chat system. A C2 operator can read the chat, determine if anything needs to be done, and execute or modify plays towards resolution of the purpose of the chat message. To that end, the task manager features a repository of sample chat text that is directly mapped to supervisory tasks. As chat messages arrive through subscriptions to AIM's central messaging hub, the task manager will compare a new chat message to its repository of sample text. Regular expressions are used to identify whether a chat message will invoke the instantiation of a supervisory task in the task manager.

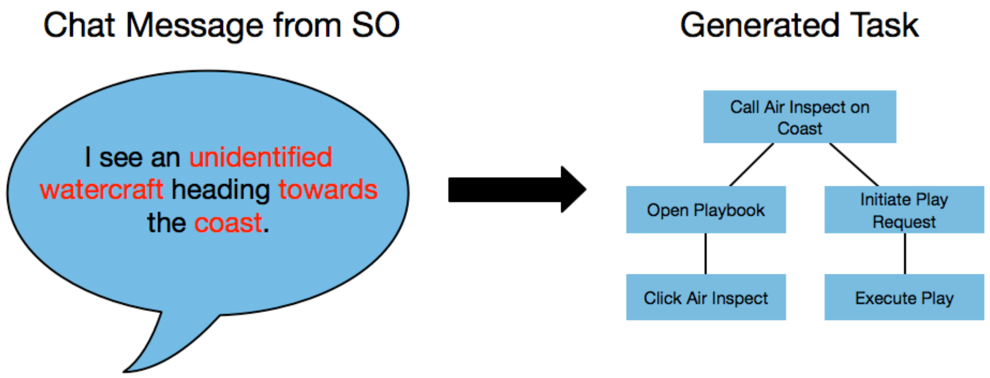

Figure 3: Generation of supervisory task from chat message.

A supervisory task in the task manager is a data structure that represents a high-level task, and is composed of actions towards accomplishing that task. In Figure 3, the generated supervisory task will have Call Air Inspect as the root task, and a subtask to call an air inspect by clicking a button, which the Open Playbook method dictates can be achieved through the playbook. The right side of the DAG is amenable to execution by autonomy as it involves dialogue through request protocols with the resource allocation and route-planning agents. The decision to select the appropriate method for completion of the task, that is, for the human to open the workbook, or for the autonomy to initiate an automated play will be discussed in the Working Agreements section (4.5) of this paper.

4.1.1.2 Events

Events in an AIM instantiation may involve changes in the environment that need to be addressed by the C2 operator. Some events are triggered through the policy-checker in AIM; the Configurable Operating Model Policy Automation for Control of Tasks (COMPACT) [6]. COMPACT provides AIM the ability to perform compliance-checking of policies as the operational situation evolves. Among the policies monitored by COMPACT are projected asset intersection with restricted areas, and ensuring asset communications. COMPACT may have knowledge of restricted areas that AIM does not have, and will communicate policy summaries announcing a projected infraction. The plan monitoring agent [10], which provides runtime mission plan and environmental monitoring, ingests these policy summaries and initiates a dialog with COMPACT to introduce new knowledge of restricted areas known only to COMPACT into AIM. The plan monitor then publishes information on the affected assets that the task manager uses to generate new supervisory tasks directing the C2 operator, or the automated assistant, to re-route affected plans around restricted areas. Similarly, a communications policy summary is processed by the plan monitor in order for the task manager to generate a task towards establishing a communications relay for an asset at risk of losing communications.

4.2 Modeling agent performance

The ability to track the performance of agents may help optimize task management. While some tasks are more suitable for human operators, e.g. those involving decision making, other tasks such as data collection or analysis are better suited for autonomous assistants [8]. Repetitive or tedious tasks may introduce the risk of lowering human operator attention but such tasks are perfectly suitable for execution by autonomy. Human operators are unique and can have preferences towards the types of tasks they prefer to prioritize. An operator's ability to perform tasks can change due to stress, fatigue, or through changes in the environment. Therefore, modeling the human's ability to perform tasks is an important step towards useful task management. Modeling the autonomous agent's task performance can help to compare and determine which tasks should be assigned to whom.

One approach to model agent performance is through autonomics. Autonomic computing aims to grant systems the ability to self-manage through high-level administrator goals. Inspired by the autonomic nervous system, autonomics-enabled systems exhibit self-healing properties through the Monitor, Analyze, Plan, Execute (MAPE) paradigm [11]. Key to autonomics is the ability to model the system in order to recognize when adaptation is needed. One such autonomics tool is the Rainbow Autonomics Framework, developed at Carnegie Mellon University [12]. Rainbow enables self-adaptation to implementing systems through special software. Rainbow probes sense the system and publish useful data to Rainbow gauges, which reflect the state of the system in models defined in an architectural description language, Acme [13]. An Acme model can be primed with design assumptions, in the form of rules which are analyzed and compared against the model at runtime by an Architecture Evaluator. Upon detection of a rule violation, an Adaptation Manager selects a strategy that is carried out by a Strategy Executor through effectors, which make changes in the target system to bring the system back to a desirable state. Although Rainbow is primarily used to manage computer networks, we have implemented a Rainbow instantiation in the task manager for the purpose of managing supervisory tasking in AIM. To build a model of a task-executing agent, we have developed probes to collect information on the AIM system, and gauges to build models at runtime.

4.2.1 Probes and Gauges

To begin building a performance profile for an agent, we must first be able to uniquely identify an agent; whether they are a human operator, or an autonomous assistant. Meta-data such as user name, login, timestamps, and operating system information can help determine the identity of a human operator. Similarly, a model of an autonomous assistant can be tied to system data. Such data is reachable by probes into operating system properties, and databases backing an AIM instantiation. Aside from agent data, the task manager needs information comprising the AIM environment where the agents perform supervisory tasking. This is done through probe subscriptions to AIM’s central messaging hub where entities such as vehicle states, vehicle tasks, and mission plans are published. These entities help provide context for the supervisory tasks to be performed by agents. Probes into the task generation described in the Task structure and generation section (4.1.1) complete the information required for agent modeling. Rainbow gauges process probe reports to maintain a model of the target system. In this case, models for agents, vehicles, tasks, and mission plans are used to represent the environment. Gauges update agent models as supervisory tasks are performed at runtime. Parameters towards measuring agent performance include timestamps when a task is initiated and completed, and its duration. Supervisory tasks can be categorized by the type of the task such as a task to call a play, or a task to respond to a query in chat. Therefore, performance metrics could be established per task type. Rate of cancellation per task type may provide insights for agent performance profiling, and since an operator can transfer tasks to the automated assistant; we can record such actions as parameters as well. The end goal is to be able to predict agent performance on a particular task based on past performance on similar tasks [14].

4.3 Tracking task urgency

Critical events can occur while the operator performs C2 operations in AIM. Such events may require immediate attention and should be addressed in a timely manner. Supervisory tasks generated by the task manager towards resolution of these events are configured with a higher priority than tasks that may not be as urgent. Other tasks have deadlines where failure to complete the task in the specified timeframe has a detrimental effect on the tactical situation. Within the task manager, tasks utilize a priority attribute that is configured by task type, and can be updated over time. For example, query tasks have a low priority by default, as they involve simply answering a query in the chat system. The identity or situation of the request source could affect its priority. While communicating information is an important aspect of C2, there may be more urgent tasks than responding to a chat message. A critical event such as detection of a gate-runner requires an immediate response by the C2 operator. Therefore, the intruder response supervisory task is by default a high priority task. Tasks generated through chat messages may contain time information. A chat message such as "Alpha company is sending out a foot patrol, with callsign Devonian12, at 1430Z" can generate a task with a priority that is initially low, but will increase as the time approaches 1430Z. The task manager UI could sort tasks in the task queue based on priority, so that the more urgent tasks appear at the top of the queue and elicit more of the operator's attention.

4.4 Balancing workload

The task manager is capable of balancing the workload of the operator by tasking an autonomous assistant with extraneous tasks. The assistant provides services to receive tasks and execute them, which alleviates operator tasking on lower priority or repetitive tasks. Tasking the automated assistant through Task Manager can be done in two ways: manually assigning tasks with the click of a button, and automatic assignment through load balancing. The operator has the authority to give or take tasks from the autonomous assistant if deemed necessary for control of mission parameters [5]. An approach for task assignment towards enhancing the performance of the human-autonomy team is to balance operator attention with risk [8]. If both the operator and autonomy are available to perform a task, then we want to assign the task to the agent that will complete the task with the least risk. During high workloads, the operator's available attention is reduced and there is a higher risk of tasks failing; therefore, we should offload more tasks to the autonomy. On the other hand, if the workload is low, we want to assign more tasks to the operator; otherwise we risk reducing the operator's situational awareness. This work is currently in progress in the task manager, and we are experimenting with ways of modeling operator attention in order to offer more sophisticated load balancing. Currently, we balance the workload based on how amenable a task is towards execution by the human or the autonomy; which was discussed in the Task structure and generation section (4.1.1) of this paper. Later we discuss future work towards load balancing when there are multiple human operators performing from a common task queue.

4.5 Working Agreements

A consideration in HAT is how to determine the tasks permissible for execution by the autonomy. In order to foster trust in the autonomy, and for the autonomy to be a dependable teammate, the human needs to understand what the autonomy can do. One approach towards enabling predictable behavior for the autonomy is through working agreements [15]. Through working agreements, human operators can configure the tasks or types of tasks permitted to the autonomy. In other words, the operator is the ultimate authority on task assignment and execution. Over time, the operator can choose to allow more work to be done by the autonomy to slowly build preferences as to the division of labor. A form of working agreements is in development in the task manager. In its current state, we have configured certain tasks to be executed by the autonomy. In Figure 3, we see a supervisory task to call a play that was generated from a chat message. The working agreement for this task is set to only be executed by the human operator. However, in a future version of the task manager, the operator will be free to change the configuration to a human-on-the-loop approach and always assign this task to the autonomy. In this case, the autonomy will immediately initiate dialogue with the AIM system to call a new play. The operator could instead choose a human-in-the-loop approach and assign this task to the autonomy but require the autonomy to request permission from the operator at certain steps along the play calling process.

4.6 Linked tools

An important milestone towards task management is helping to improve operator workflow. In complex situations, tasks may be streaming into the task manager and it may become difficult for the operator to keep up with the work. Productivity and performance can suffer as the backlog of tasks increases, which can ultimately lead to decreased supervisory C2 effectiveness. The task manager UI features a task queue with supervisory tasks to be completed by the operator, or the autonomy. Certain tasks can link to other UI elements in the AIM system. In Figure 4, we observe the task manager linking a call play task to AIM's playbook tool, which is a completely separate module from the task manager within AIM. The task manager forwards task meta-data to the playbook in order to pre-populate play request settings towards calling of the play and completion of the task. In a demanding environment, the operator would need to manually open the playbook, select the location where the play is to be called, input the specific parameters to call the play, and finally accept the play. These actions could distract from other potentially more urgent tasks, and since this involves manually copying data, is prone to mistakes through the operator misreading, or mistyping a piece of data. Finally, task meta-data can have timestamps, that can help us track when a task was initiated, and completed which can help with performance profiling.

Figure 4: The Task Manager facilitating the calling of a play by opening a playbook with pre-populated settings from a supervisory task.

5.0 EXPECTATIONS FOR TEAM PERFORMANCE

IMPACT has gone through several evaluations during development and experimentation. With each iteration, the scenarios have become more complicated and the operator task load has increased and become more complex. In IMPACT's Spiral 1 evaluation, Air Force personnel with experience piloting UAVs, and subject matter experts in base defense operations were tasked with conducting C2 in IMPACT [3]. In general, the IMPACT system was rated positively with high scores in potential for use in future C2 operations. The task manager was not included in this evaluation. However, in the Spiral 2 evaluation, the task manager was one of the autonomous agents enabling HAT. In this evaluation, a baseline IMPACT version with minimal autonomy was compared to a full version of IMPACT, with all autonomy enabled [4]. Results showed participants preferred the version of IMPACT with full autonomy enabled, and again positively rated the system for its potential in future C2 operations. The upcoming evaluation of Allied IMPACT in Autonomous Warrior 2018 will feature the latest evolution of the task manager, with new features. One such feature will be providing the human with explanations of AIM scenario or activities with a provenance service [16] through the Narrative virtual human [7] which we believe will be useful for situational awareness. Another feature is the automatic re-routing and play-calling of new plays through COMPACT policy checking [6], which should increase the performance of the human-autonomy team.

6.0 FUTURE WORK

We plan to enhance task generation through chat with Natural Language Processing (NLP) methods. Currently, our regular expression-based method has limitations, as we use direct mappings of text to tasks. This approach is somewhat brittle as a human is able to write a chat message in several ways, while keeping semantics of the message intact. NLP could help provide a more robust chat generation technique.

A future evolution of C2 operations could enable multi-human autonomy teams where multiple operators, aided by autonomy, work together to provide supervisory C2. Multi-Armed Bandit (MAB) problems are basic examples of sequential decision problems using reinforcement learning [17]. MAB is used for regret analysis and maximizing total payoff obtained in a sequence of actions. We intend to use treat task management problems as MAB problems for multi-operator task assignment. This will enhance our task manager's load balancing abilities.

Deep Q-Learning from Demonstration is a deep reinforcement learning approach that leverages human demonstration data to massively accelerate the training process [18]. We plan to train the Task Manager's autonomous agent using data from the operator in other C2 systems to learn how to accomplish complicated tasks and optimize with reinforcement learning. The Task Manager will then be able to recommend and execute more complex strategies learned through human demonstration.

7.0 SUMMARY

In this paper we approached task management in a complex C2 environment. The challenges that come with task management for humans are difficult in an ever-changing and evolving battlespace. Autonomy within C2 has offloaded some work to computers, but a human-in-the-loop or human-on-the-loop approach is necessary to maintain operator engagement. Our task management system follows principles of both intelligent decision support systems and classical task management. Our approach takes an abstract command and converts it to a machine- and human-readable task and goal for completion. Additionally, an autonomous agent exists within our task manager in order to accomplish mission critical tasks that are repetitive, trivial, or unfavorable to the current operator, or the operator may be too overworked to notice. These key capabilities enable a more efficient human-autonomy team, as the human is able to manage supervisory tasking with greater control. The benefits of the task manager will become more pronounced as the complexity in the C2 environment increases.

8.0 REFERENCES

[1] Guerlain, Stephanie, Donald E. Brown, and Christina Mastrangelo. "Intelligent decision support systems." In *Systems, man, and cybernetics, 2000 ieee international conference on*, vol. 3, pp. 1934-1938. IEEE, 2000.

[2] Bellotti, Victoria, Brinda Dalal, Nathaniel Good, Peter Flynn, Daniel G. Bobrow, and Nicolas Ducheneaut. "What a to-do: studies of task management towards the design of a personal task list manager." In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 735-742. ACM, 2004.

[3] Behymer, Kyle, Clayton Rothwell, Heath Ruff, Michael Patzek, Gloria Calhoun, Mark Draper, Scott Douglass, Derek Kingston, and Doug Lange. *Initial evaluation of the intelligent multi-uxv planner with adaptive collaborative/control technologies (IMPACT)*. Infoscitex Corp. Beavercreek, 2017.

[4] Draper, Mark, Allen Rowe, Scott Douglass, Gloria Calhoun, Sarah Spriggs, Derek Kingston, Laura Humphrey et al. Realizing Autonomy via Intelligent Hybrid Control: Adaptable Autonomy for Achieving UxV RSTA Team Decision Superiority (also known as Intelligent Multi-UxV Planner with Adaptive Collaborative/Control Technologies (IMPACT)). No. AFRL-RH-WP-TR-2018-0005. 711 Human Performance Wing Wright-Patterson Air Force Base United States, 2018.

[5] Dzieciuch, Iryna, John Reeder, Robert Gutzwiller, Eric Gustafson, Braulio Coronado, Luis Martinez, Bryan Croft, and Douglas S. Lange. "Amplifying human ability through autonomics and machine

learning in IMPACT." In *Micro-and Nanotechnology Sensors, Systems, and Applications IX*, vol. 10194, p. 101941Y. International Society for Optics and Photonics, 2017.

[6] Lucero, Crisrael, Braulio Coronado, Eric Gustafson, and Douglas S. Lange. "Tracking Provenance in Decision Making Between the Human and Autonomy." In *International Conference on Engineering Psychology and Cognitive Ergonomics*, pp. 171-180. Springer, Cham, 2018.

[7] Nowina-Krowicki, Marcin, Peter Argent, Crisrael Lucero, Douglas Lange, and Steven Wark. "Now This Changes Everything: Managing Provenance using Multimedia Narrative."

[8] Gutzwiller, Robert S., Douglas S. Lange, John Reeder, Rob L. Morris, and Olinda Rodas. "Human-computer collaboration in adaptive supervisory control and function allocation of autonomous system teams." In *International Conference on Virtual, Augmented and Mixed Reality*, pp. 447-456. Springer, Cham, 2015.

[9] Chandrasekaran, Balakrishnan, Todd R. Johnson, and Jack W. Smith. "Task-structure analysis for knowledge modeling." *Communications of the ACM* 35, no. 9 (1992): 124-137.

[10] Coronado, Braulio, Eric Gustafson, John Reeder, and Douglas S. Lange. "Mixing Formal Methods, Machine Learning, and Human Interaction Through an Autonomics Framework." In *2016 AAAI Fall Symposium Series*. 2016.

[11] Kephart, Jeffrey O., and David M. Chess. "The vision of autonomic computing." *Computer* 1 (2003): 41-50.

[12] Garlan, David, S-W. Cheng, A-C. Huang, Bradley Schmerl, and Peter Steenkiste. "Rainbow: Architecture-based self-adaptation with reusable infrastructure." *Computer* 37, no. 10 (2004): 46-54.

[13] Garlan, David, Robert Monroe, and David Wile. "Acme: An architecture description interchange language." In *Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research*, p. 7. IBM Press, 1997.

[14] Lange, Douglas S., Robert S. Gutzwiller, Phillip Verbancsics, and Terence Sin. "Task models for human-computer collaboration in supervisory control of teams of autonomous systems." In *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2014 IEEE International Inter-Disciplinary Conference on*, pp. 97-102. IEEE, 2014.

[15] Lange, Douglas S., and Robert S. Gutzwiller. "Human-autonomy teaming patterns in the command and control of teams of autonomous systems." In *International Conference on Engineering Psychology and Cognitive Ergonomics*, pp. 179-188. Springer, Cham, 2016.

[16] Crisrael Lucero, Braulio Coronado, Oliver Hui, and Douglas S. Lange. Exploring Explainable Artificial Intelligence and Autonomy through Provenance. *IJCAI-ECAI 2018 2nd Workshop on Explainable Artificial Intelligence*, 2018.

[17] Bubeck, Sébastien, and Nicolo Cesa-Bianchi. "Regret analysis of stochastic and nonstochastic multi-armed bandit problems." *Foundations and Trends® in Machine Learning* 5, no. 1 (2012): 1-122.

[18] Hester, Todd, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan et al. "Deep Q-learning from Demonstrations." *arXiv preprint arXiv:1704.03732*(2017).