Trust in Sparse Supervisory Control

Douglas S. Lange[1], Phillip Verbancsics[1], Robert S. Gutzwiller[2], and John Reeder[3]

[1]Space and Naval Warfare Systems Center Pacific
[2]Colorado State University
[3]University of Central Florida
doug.lange@navy.mil

Abstract

Command and Control (C2) is the practice of directing teams of autonomous units, regardless if the units are directed by human decision making or are unmanned. Humans are adaptive and their behaviors are recognizable to their superiors who were educated in a similar manner. This paper describes the sparse supervisory control that must be exercised over teams of highly autonomous units, and considers what it means for a commander to supervise autonomous un-manned systems (AUS) that employ machine learning and cooperative autonomy. Commanders must decide whether to trust behaviors they have never seen before, and developing that trust may require several strategies. This paper describes some possible strategies in an effort to explain the challenges that must be solved.

Sparse Supervisory Control

In this paper we envision the future of unmanned vehicle command and control (C2). Our assumption is that vehicle autonomy in a basic sense will be successfully implemented for our domain and that vehicles will conduct the basic elements of completing a task without specific human direction. As autonomy matures, vehicles will become purposeful members of a team (Schuster et al. 2011) and begin to execute stages of information processing on their own.

However, complex strategic decisions involving teams of autonomous vehicles will still be desired, and are not currently provided in most conceptualizations of multi-vehicle C2. Human commanders will, as they have always done with team of autonomous units (albeit human populated), direct the employment of assets and determine the objectives for a team. Additionally, human commanders will monitor team performance to determine if adjustments are necessary. Higher levels of autonomy will allow controllers to 'invert the ratio' of operators to units and exert control over larger groups of autonomous unmanned systems (AUS) with fewer humans. The term adopted is sparse supervisory control, but we view it as virtually synonymous with command and control.

The control exercised by a military commander over forces is described as "…guiding the operation" (Willard 2002). The presumption is that there is a mission statement, a set of assets with which to perform the mission, and an environment to operate in that may include an opposing force. According to Willard, there are several ways in which a commander guides an operation.

- *Maintain alignment*: The commander must ensure that all decisions remain aligned with the operation's mission and the commander's intent.
- *Provide situational awareness*: The commander must assess the status of plan execution constantly, utilizing a common operational picture (COP).
- *Advance the plan*: The commander must monitor the status of plan execution against the plan's timeline.
- *Comply with procedure*: The commander oversees compliance with warfighting procedures to avoid mistakes (e.g., friendly fire engagements or collateral damage) and achieve efficiencies.
- *Counter the enemy*: The commander must be responsive to emerging intelligence, surveillance, and reconnaissance information that differ significantly from expectations.
- *Adjust apportionment*: Changes to asset availability or changes to requirements and priorities may require reapportionment of assets.

Human Management of Automation

Automation has also been shown to result in two highly related phenomena. These are *complacency,* in which operators may fail to detect an automation malfunction, and *automation bias*, which consists of operators blindly following automation recommendations or failing to act unless the automation requests the human action within decision making systems (Parsasuraman and Manzey 2010). The information provided to the operator, and the method by which it is provided matter greatly to the success or failure of human management of automation.

It is possible that it is specifically the integration that plays a large role in determining the effectiveness of a system outcome. In (Leli and Filskov 1984), automated diagnostic systems consistently outperformed clinicians when in isolation; however decision accuracy decreased as a direct result of integrated clinician application of the aid to diagnose psychological conditions. This result suggests that perhaps the most critical aspect of automation is not the engineering behind the automation itself, but the interaction between any automation and the operator who is expected to work together with it.

Issues have also been identified related to the ability of human operators to understand the actions of any automation (Parasuraman and Wickens 2008). The operator's trust and ability to evaluate the performance of autonomous systems comes, in part, from an ability to recognize behaviors as correct or incorrect given contextual constraints. AUS that have been programmed to perform in a particular fashion may or may not exhibit recognizably correct behaviors that are, however, optimal for the given situation.

Achieving Sparse Supervisory Control

Where we previously had one operator (or more) manually controlling only one AUS, we plan to have one operator now oversee the workings of several semi or fully autonomous vehicles, changing the role to one of supervision and overseeing automated functions. This role shift is where the control of automation merges back with command and control in terms of goals and where what has been learned about the management of automation must be applied to succeed in exercising sparse supervisory control of AUS.

For command and control to function there are critical requirements. One such requirement is that commanders must be able to communicate intent to AUS teams, and commanders must trust that the teams understand this intent and must be able to recognize when decisions made by the team are out of alignment with that intent. This is true whether the units in the team are autonomous due to human intelligence, or through computer automation.

Trust between human controllers and human crews of ships and aircraft is enhanced due to a common language and education. Computer-based autonomous systems do not have those features in common with their human controller. While it is possible to create a standard encoding of commands and status information that is both human and machine readable (e.g., some Extensible Markup Language based grammar), there is little likelihood of the human having a clear understanding of the processing of a command from a connectionist vehicle controller (e.g., one utilizing artificial neural networks).

The basic opportunity for developing the necessary insight and resulting trust is likely to come from repeated practice. Simulation-based and live training of the human-AUS team is a natural approach. In the next section of this paper, we introduce added complexity by arguing that AUS will necessarily become adaptive systems. This adaptation is likely to occur at a rapid pace during team training, and indeed we suggest that the need for simulation systems that will support this type of training and adaptation will be essential. But adaptation is likely to occur after deployment. Certainly human-occupied systems (the human components in particular) adapt while deployed. AUS will necessarily adapt to avoid being returned to a laboratory environment at every significant change by either the environment or an opposition. Therefore trust will have to be developed in other ways as well.

Adaptive Behaviors

A common approach to the design of autonomous systems is to design with fixed policies in place to guide behaviors. Such systems face many challenges. One difficulty is that unexpected situations can cause the autonomy to not work. Another is that opposing forces can take advantage of predictable behaviors. This lack of robustness is due to the expansive state space that exists in the real world. Human designers will not be able to test or even anticipate every situation the autonomous system will be exposed to. Additionally, fixed systems often lack scalability. In particular, the designs will be tied to a particular number of autonomous agents, or a particular team composition, meaning each time the number or types of unmanned vehicles change, the autonomy must be redesigned.

In a proof of concept experiment, such a fixed system was compared to a multiagent learning method. Multiagent Hypercube-based NeuroEvolution of Augmenting Topologies (HyperNEAT) approaches the problem of multiagent learning by focusing on the geometric relationships among agent policies (D'Ambrosio and Stanley 2008). The policy geometry is the relationship among policies located at particular positions and the team behavior.

Whether due to malfunctions or due to changing command decisions, team composition will change. The multiagent HyperNEAT approach allows such scaling because it represents team policies indirectly as a function of team geometry. Thus new agents can be added by simply generating the policy for their assigned team position.

Overall, the results from our experiments showed that policies created by multiagent learning approaches are more robust to change (Calinescu and Garlan 2012). The scripted parallel search and learned multiagent HyperNEAT policies were compared on three variations of a threat detection task.

The first variation is the training task for multiagent HyperNEAT, in which there were seven simulated unmanned vehicles patrolling and threats could randomly appear along any of the four edges of the square operational area. In this task, the learned policy had statistically the same performance as the scripted policy.

In the second variation, the tactics employed by the threats were altered such that they now appeared from just two of the four sides at random, thus increasing the density of the attacks along that vector and testing the robustness of the approaches. The learned policy significantly outperforms the scripted policy. In the third variation, the number of simulated unmanned vehicles in the team was increased from seven to eleven. The learned policy exploited the increased number of vehicles, decreasing the missed threats to zero. However, the scripted policy was unable to take advantage of the new vehicles and did not significantly change in performance.

Trusting Adaptive Behaviors

Using HyperNEAT to develop team tactics will create more robust and scalable policies and behaviors. However, we must also be concerned with whether or not the human controller will recognize the behaviors as being safe and correct. As the HyperNEAT approach produces artificial neural networks (ANNs), we can only look at the team tactics as black boxes, and even within the proof of concept experiment, interpretation of behaviors was difficult. A human controller in such a system however, must be able to decide if the tactics being employed are aligned to the mission and whether or not they are properly countering the enemy or handling arising complications.

One of the primary drawbacks to learning behaviors is that in the search for optimal actions the agents can behave in ways that seem foreign and unintelligible to the human operators. It may be the case, that agents that behave in a more humanlike fashion are more easily trusted by human observers. The development of humanlike agents is possible through hand coding and expert systems, but it is a tedious and complicated process. However, learning humanlike behavior is possible through observation. Such behavior can then be improved with practice and feedback.

FALCONET is a method of agent training that follows the biologically inspired cycle of observation and experiential learning. It was designed to enable the creation of high performing, humanlike agents for real time reactionary control systems (Stein 2009).

The training in FALCONET follows such a two phase training approach. First, a supervised observational phase is conducted in which the objective of the learning is to be similar to the actions of a human trainer. Human trainers run through selected tasks starting from many different scenarios to generate an observational training set. The agents are then trained on this data set while being graded on how closely they mimic the decisions of the human. In the second, experiential phase, the agents are trained further using a measure of performance on the task. In FALCONET all training is done by a hybrid genetic algorithm (GA) particle swarm optimization (PSO) algorithm called PIDGION-alternate. This is an ANN optimization technique that generates efficient ANN controls from simple environmental feedback. FALCONET has shown that it can produce agents that perform as well or better than experiential training alone while incorporating humanlike behaviors. The results from FALCONET also show that unique human operator traits can be incorporated and can be behaviorally evident in the final highest performance controls, that is to say, agents sourced from different human trainers have slightly different, and importantly interpretable, behavioral 'quirks'.

Layered Learning

The potential state space of system containing a large number of AUS acting in a noisy, real-time, cooperative, and adversarial environment is staggering. Utilizing a single learning method or utilizing a single adaptive controller for a team or each member of a team does not appear practical. With so many decisions being made from a small set of intelligent components, the likelihood of gaining trust in the decision making seems small.

One approach that can mitigate this problem is layered learning (López de Mántaras and Plaza 2000). Layered learning is a technique that decomposes a task into separate layers and uses learning algorithms tailored to the needs of each layer. Some possible layers in naval AUS system might include the classification of reconnaissance targets, the planning and scheduling of patrols, and the reactive controls of the individual AUS.

Layered learning is meant to facilitate learning in problems where it is intractable with current algorithms to learn a direct mapping from environmental inputs to system outputs. There are four major principles of layered learning:

- A mapping directly from inputs to outputs is not tractable when the state space is large, continuous, noisy, and contains hidden states. Layered learning uses a bottom up approach to incrementally learn a solution from low level tasks to high level strategic behaviors.
- The layers of the system are a function of the domain to be learned. The layers are defined a priori by the machine learning opportunities in the domain. However, it could be possible to combine layered learning with an algorithm that learns abstraction levels.
- Learning is done at each layer, and can be done off-line or on-line. The type of learning is dependent on the subtask being learned.
- The learning at each layer affects the next layer in the chain. A learned sub-task can affect the next layer by

providing a portion of the behavior to be learned, defining the features that are learned, or by pruning the output set.

The original application of layered learning was in the robo-soccer domain (Stone and Veloso 2000). In this case the layered learning system had three layers; the first layer was an interception behavior learned by a neural network off-line. An agent was set up in the simulation to learn how to intercept a ball from varying initial states.

The second layer was a decision tree to learn when it was safe to pass the ball to a team mate, and in this layer the agents in the simulation used the previously learned interception behavior as the decision tree was being trained. In this way, the first layer provided a portion of the behavior of the agents and made the training data for the decision tree more robust.

The final layer was a reinforcement learning technique. In this layer the output of the decision tree from the previous layer was used as inputs to the reinforcement learning providing confidence levels for passing to each of the agent's teammates. In this manner the second layer was used as a state generalization algorithm for the third layer.

Robust layered learning is an adaptation aimed at making the system robust to failure and change (Richert and Kleinjohann 2007). Instead of having each layer feed directly into the next layer, layer interfaces are defined which restrict how the layers communicate. By doing this it becomes possible to switch out the learners at each layer without significantly changing the other layers. These layer interfaces are usually in the state language of the lower layer, meaning the higher layer in effect tells the lower layer which state it would like to be in, then the lower layer attempts to achieve that state.

Each higher layer is a more abstract representation of the environmental state space. By isolating each layer through the layer interfaces it makes it easier to try different learning techniques at each layer, or to have a system with redundant layer techniques. For instance, one layer could be a rule-based system that performs well in known situations, but might degrade when faced with novel situations. Once the rule based system starts to perform poorly, a more general learning system could be swapped in. The cost of using robust layered learning comes at the expense of defining the layer interfaces as well as the layers as part of the design. On the other hand, having these interfaces in place ensures that each layer encapsulates knowledge such that it can be transferred to other systems.

From the standpoint of trust, robust layered learning appears to have significant benefits. Once the levels and interfaces have been designed it would be possible to pick and choose where learning takes place and where hard coded deterministic behaviors are in control. Moreover, it creates a system that could be upgraded over time with more learning techniques as the system progresses and the need for them becomes more obvious.

A similar strategy that provides this feature without layering the learning is to utilize unlike redundancy of complete subsystems (Calinescu and Jackson 2011). Layered learning accomplishes this same capability and provides support for learning at various levels of abstraction.

Conclusions

When fielding a team of autonomous vehicles, one expects emergent behaviors. In fact, one fields a team of AUS in order to get emergent behaviors; and a single AUS that is allowed to adapt by definition will result in emergence. A team of AUS that includes interaction and cooperative autonomy is likely to exhibit emergence even without adaptive techniques.

Novel but correct behaviors and intentions may be misinterpreted by the operator and interrupted. For robust use, we will likely require teams of AUS to exhibit adaptive behaviors. To trust those behaviors, human controllers will need to be able to recognize those behaviors as correct. To prevent catastrophe, controllers will also need to be able to recognize emergent aberrant behaviors; we suggest that a combination of observational and experiential learning will lead to behaviors that can adapt but yet also be utilized within a command and control framework.

Layered learning provides an approach that will aid in the acquisition of complex behaviors, and does it in a manner where confidence and trust can be gained at each layer. Unlike redundancy, it can also be developed within the layered learning structure.

We also suggest that the level of abstraction and the manner in which information is provided to the human controller is critical. Studies of complacency and automation bias show that humans often misinterpret the state or intentions of complex automated systems. Heterogeneous teams of AUS that employ adaptation will result in the need for situational awareness much like military commanders must have for complex battle spaces. The key difference will be that the unmanned systems will speak a different language and do not share a common educational background as the controller.

References

Calinescu, R., and Garlan, D., eds. 2012. *Large-Scale Complex IT Systems. Development, Operation and Management.* 81-93. Berlin: Springer.

Calinescu, R., and Jackson, E., eds. 2011. *Foundations of Computer Software: Modeling, Development, and Verification of Adaptive Systems*. 193-205. Berlin: Springer.

D'Ambrosio, D.B., and Stanley, K.O. 2008. Generative Encoding for Multiagent Learning. In Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation. 819-826. Atlanta, GA, Association for Computing Machinery.

Leli, D., and Filskov, S. 1984. Clinical detection of intellectual deterioration associated with brain damage. *Journal of Clinical Psychology*. 40(6). 1435-1441

López de Mántaras, R., and Plaza, E., eds. 2000. *Machine Learning: ECML 2000,* 369-381, Barcelona, Springer Verlag

Parsasuraman, R., and Manzey, D.H. 2010. Complacency and Bias in Human Use of Automation: An Attentional Integration. *Human Factors*. 52(3) 381-410.

Parasuraman, R., Wickens, C.D. 2008. Humans: Still vital after all these years of automation. *Human Factors*. 50(3). 511-520.

Richert, W., and Kleinjohann, B. 2007. Towards robust layered learning. In Proceedings of the Third International Conference on Autonomic and Autonomous Systems. 34-41. Athens, Institute of Electrical and Electronics Engineers.

Schuster, D.; Ososky, S.; Phillips, E.; Lebiere, C.; and Evans, A.W. 2011. A Research Approach to Shared Mental Models and Situation Assessment in Future Robot Teams. In Proceedings of the Human Factors and Ergonomics Society 55th Annual Meeting, 456-460. Las Vegas, NV, Human Factors and Ergonomics Society.

Stein, G. 2009. Force-feedback Approach for Learning From Coaching and Observation Using Natural and Experiential Training. Ph.D. diss., Department of Computer Science, University of Central Florida, Orlando, FL.

Stone, P., and Veloso, M. 2000. Layered learning and flexible teamwork in robocup simulation agents. *RoboCup-99: Robot soccer world cup III*, 1856: 65-72.

Willard, R. F. 2002. Rediscovering the Art of Command & Control. *Proceedings of the United States Naval Institute*, 128(10): 52–54.