Presented at BRIMS 2011, Sundance, UT

Formative Evaluation of an IUI for Supervisory Control of CGFs

Glenn Taylor
Brian Stensrud, PhD
Soar Technology, Inc.
3600 Green Court Suite 600
Ann Arbor, MI 48105
734-887-7620, 407-207-2237
{glenn, stensrud}@soartech.com

Jeffrey Maddox
Harold E Aycock
U.S. Army Aviation and Missile Command
Research, Development & Engineering Center
Redstone Arsenal, AL 35898
256-876-7716, 256-842-6775
{jeffrey.a.maddox, harold.aycock}@us.army.mil

Keywords:
CGF Operator Interface, Intelligent User Interface, Supervisory Control

ABSTRACT: *Computer Generated Forces (CGFs) tend to require a great deal of user effort to keep the CGFs tasked and keep the user aware of what the CGFs are doing. Furthermore, every CGF system is different, with different user interfaces requiring significant learning time for operators to be effective. Users must conform to the idiosyncrasies of each CGF rather than the system conforming to user. We present an intelligent user interface (IUI) that raises the level of interaction to one of supervisory control, allowing a user to issue commands in more natural terms, and getting feedback from the CGFs in natural terms. We present some formative results that show the potential for significantly reducing the workload of CGF operators.*

1. Overview and Motivation

Computer Generated Forces (CGFs) tend to require a great deal of user effort to keep the CGFs tasked and keep the user aware of what the CGFs are doing. Furthermore, every CGF system is different and requires specialized training to use. CGFs also have different capabilities; for example some can be given "move along route" commands and can report when they finish the task; others can be given only fine-grained waypoint-by-waypoint tasking; others still can be told to perform whole missions with little supervision. With each of these different capabilities come different user interfaces that require significant learning time for operators to be effective. New simulation environments continue to be built; however, their user interfaces do not seem to be getting easier to use. Users must conform to the idiosyncrasies of each CGF, rather than the system conforming to the way users would like to interact with them.

In an effort to improve usability in this domain, we present an intelligent user interface (IUI) called the Smart Interaction Device (SID) that tries to raise the level of user interaction with CGFs to one of supervisory control, allowing a user to issue commands in more natural terms, and getting feedback from the CGFs in similar natural terms. In doing so, it allows a user to give directives in a single uniform way, without needing to know the particular capabilities of the CGF or the nuances of the CGF's user interface. The user can expect the CGF to perform the task as desired and to provide feedback along the way. For example, even if the CGF can accept only waypoint-by-waypoint commands, the user still would like to say simply, "follow that route and tell me when you're done."

In the rest of this paper, we describe our technical approach, the system architecture, a formative experiment, and results that show the potential for reducing the workload of CGF operators.

2. Related Work

There have been prior efforts related to building speech-based interfaces to CGF systems; for example SRI's CommandTalk system (Stent et al. 1999) provided a speech interface to creating and tasking ModSAF entities, though CommandTalk was more a direct extension of the UI rather than an attempt at doctrinal communication. Other efforts have been focused on doctrinal speech for interacting with CGFs as if they were humans, including TacAir-Soar (Jones et al. 1999) and its family of behavior models. Other systems have examined using high-level tasking to direct the behavior of teams of entities (DeKoven and

Murphy 2006; Colonna-Romano et al. 2009), though typically through point-and-click style interfaces. Another approach has been user interfaces that take a "playbook" metaphor to tasking entities (Miller et al. 2005), though this particular system focuses on the user interface rather than the full control loop.

In contrast to many of these systems, SID integrates natural interaction, task monitoring, and control into a framework that enables task-oriented dialogue between a user and a range of CGF systems in different simulation environments. We have focused explicitly on the goal of reducing user workload by raising the level of interaction that these CGFs are typically capable of, and making the CGFs appear more capable in the tasks they can accomplish.

3. Technical Approach

The specific domain of interest here is airspace management, in which an operator's job is to direct one or more aircraft through an airspace (which may contain potential hazards such as other aircraft), maintain situational awareness, and respond to changes in the environment (e.g., new no-fly zones) such as by redirecting aircraft on the fly.

Our general approach to this task has been to look to human communication as a metaphor for human-system interaction. Air traffic controllers interact with pilots using natural language over radios. Each participant has some autonomy, though the controller exerts supervisory control over the aircraft. Because the controller must deal with multiple aircraft simultaneously, he or she cannot attend to a single aircraft the whole time, so must trust the pilot to fly correctly according to directives, standard procedures, and other information provided by the controller, and to provide timely and useful feedback to the controller. A controller may issue a directive to one aircraft, then switch attention to another as it enters the airspace, and rely on verbal updates from the first aircraft to maintain situational awareness. (This notion of supervisory control is not limited to controller-pilot interactions; rather it is pervasive in hierarchical organizations.)

As an intelligent user interface (IUI), the Smart Interaction Device (SID) mediates the interaction between the user and the CGFs. It manages the dialogue between the participants, understanding user commands and translating them into potentially multiple CGF-level commands. Like communication between an airspace controller and a pilot, operator-to-CGF communication in our system is based on verbal dialogue between the participants that can occur over time. For example:

Controller: *"Joker1 this is TOC1, divert to Route RED at ACP40 and report at ACP42, over"*

CGF: *"Roger, Joker1 diverting to Route RED at ACP40, will report ACP42"*

… (time passes) …

CGF: *"TOC1 this is Joker1, at ACP42, over."*

These commands must be understood within the context of the current situation including the entire conversation up to that point. The level to which the commands must be translated into will vary depending on the particular capabilities of the CGF: for example, a route command may have to be translated to individual waypoint commands, which need to be doled out over time based on CGF progress through the route. Furthermore, some CGFs may not be able to execute certain commands, for example to report at a specific waypoint, so SID must monitor the CGF's progress and generate a report to the user, using standard reporting formats, on behalf of the CGF.

The core of SID consists of three components: 1) the **Facilitator** that manages the dialogue with the user, including understanding user utterances and engaging with a user to clarify where user statements are unclear; 2) the **Tasker** that translates the understanding of the user's commands into CGF-level commands or tasks for SID to execute on the CGF's behalf; and 3) the **Monitor** that tracks CGF progress and, in general, builds situational awareness that is used to understand user utterances and generate tasks. SID can manage multiple dialogues simultaneously between a user and multiple CGFs, and tracks obligations (e.g., which CGFs owe responses to the user). These three components are implemented in the Soar Cognitive Architecture (Wray and Jones 2005), which enables the fast application of different kinds of knowledge (e.g., domain knowledge, how to communicate, how to interpret user inputs in context) to accomplish the system's goals of user communication, CGF tasking, and constantly maintained situational awareness. More detail on the system design is presented in an earlier paper (Stensrud et al. 2008).

Around these core components are two outward facing components. First is a speech interface to do the raw speech recognition and generate speech output. Second is a network-level plug-in interface to issue commands and get updates from CGFs. The network interface can connect to a range of simulation protocols, including HLA, DIS, OneSAF's SORD, and simple XML over TCP/IP. To date, we have used SID to interact with a range of CGF systems, including OneSAF, Dynetics'

Aviation Mobility Server (AMS), SAIC's Interactive Distributed Engineering Evaluation and Analysis System (IDEEAS), SoarTech's HeloSoar rotary-wing behavior model (Jones et al. 2004), and aircraft in SoarTech's open-source simulation environment called SimJr (http://code.google.com/p/simjr/). SID also includes a graphics and text-based display interface that presents SID's understanding of the situation, including the current tasking from the user, task breakdown for individual CGFs, progress in task execution, and a log of the radio interaction between participants. Figure 1 illustrates the major components of SID.

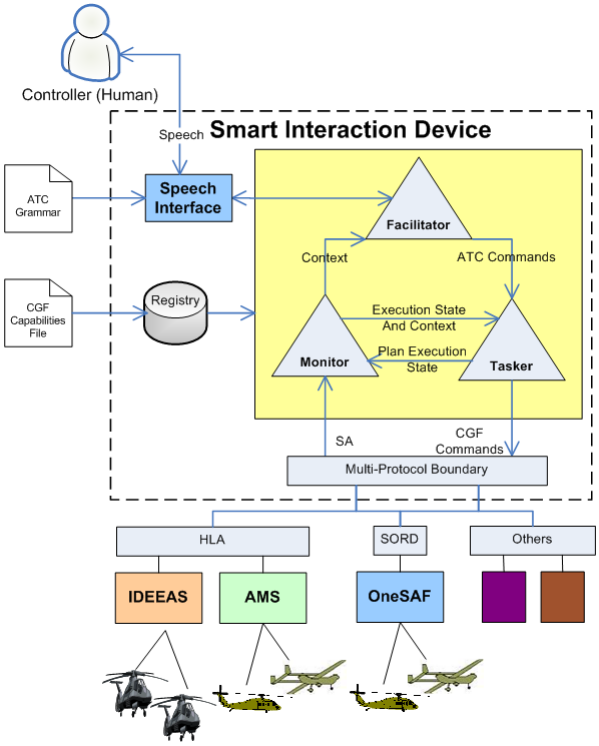

Figure 1: The Smart Interaction Device (SID) Architecture

4. Formative Evaluation

Our overall goal of building SID was to make the user's job easier in terms of CGF interaction and make the differences between CGF capabilities invisible to the user. For this evaluation, we used four different CGF systems, each with different capabilities:

- **IDEEAS** – a SAF environment developed by SAIC that allows entities to be tasked via HLA messages; could take
- **Air Mobility Server (AMS)** – a high-fidelity model of UAVs developed by Dynetics that can be directed in simulation via HLA messages
- **Helo-Soar** – a Soar-based autonomous helicopter human behavior model built on a prior project
- **SimJr helicopters** – a lightweight simulation that allows for simple scripted CGFs

(This evaluation occurred before we developed an interface to OneSAF, so results with OneSAF are not included here.)

One goal of the evaluation was to use each CGF as-is, without adding or removing behaviors, and tasking the CGFs through their standard network interfaces. The motivation here is that we want to connect to legacy systems to essentially make them appear "smarter" by adding a smart interface between them and the user.

For the evaluation, we needed to be able to repeat the same inputs across a range of CGFs to test the breadth of SID. Because of this, we chose to implement this evaluation without human operators in the loop to control for the variations in speakers, or even variation of one speaker from one performance to the next. To accomplish this, we created an evaluation script that walked the system through a set of text-based human-level commands. We were then able to use this script across each of the four CGF test cases.

We identified two factors to evaluate: Tasking Equivalence and Reduced Workload. The idea behind Tasking Equivalence is the idea that SID can be used to make very different CGFs appear equally capable by effectively shielding the user from those differences. SID essentially enhances the apparent capabilities of native CGF platforms so that CGFs of varying capabilities will equivalently respond to human-level ATC commands. In evaluating Tasking Equivalence, we are essentially verifying the idea that SID can create a uniform level at which users can interact with CGFs, despite underlying individual differences in CGF capabilities. Human operators would rather think of them as all simply entities that can be tasked in natural ways, and let the system deal with the idiosyncrasies.

In evaluating the Reduced Workload aspect of SID, we seek to assess the value of SID in terms of its ability to raise the level of interaction between a user and a CGF to one in which the user can essentially manage by supervisory control – issuing fewer commands to accomplish more, and having to work less hard to gather information about the CGF's progress.

Figure 2 illustrates the evaluation task and the scripted user commands issued to the CGFs. It includes a primary ingress, diversion around a popup restricted operating zone (ROZ), flying a holding pattern, landing at a designated area, and then egressing. In this rest of this section, we detail our evaluation setups for both tests and the results of the evaluations.

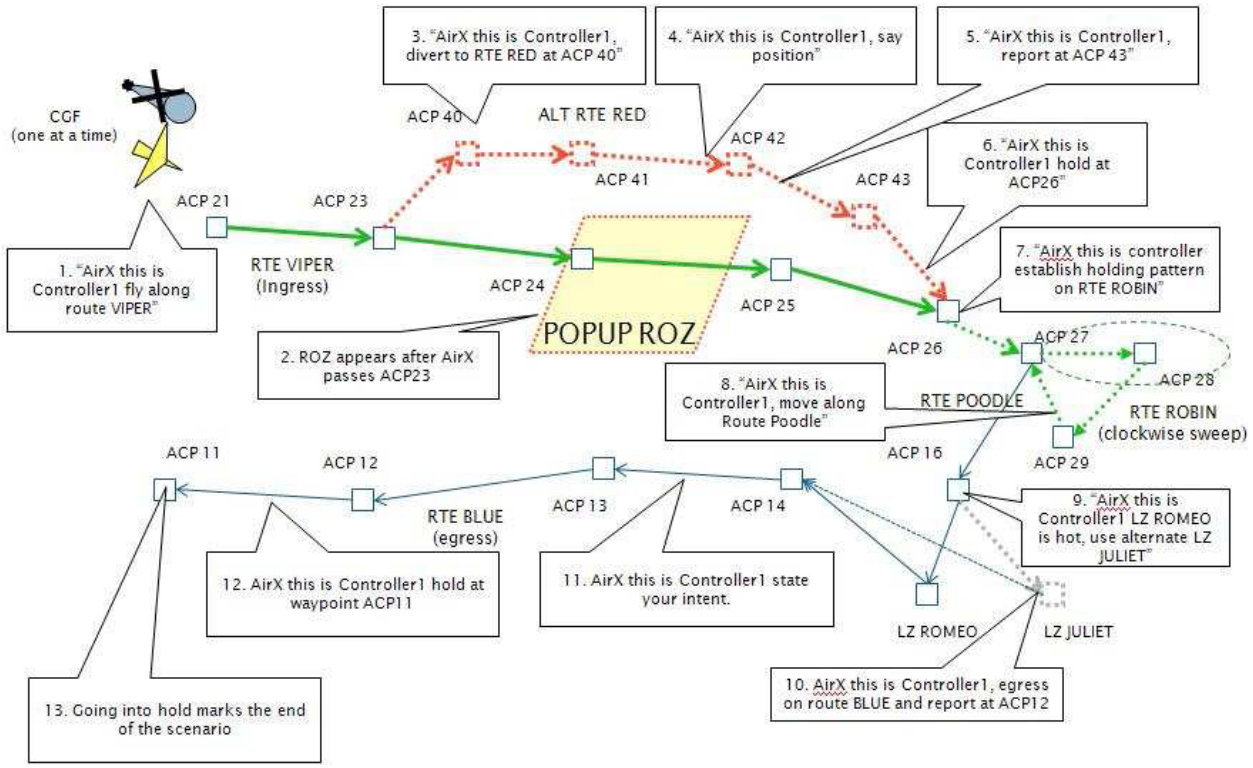

Figure 2: Evaluation Scenario

4.1 Tasking Equivalence Setup and Results

In this phase of the evaluation, we wanted to test the idea that SID could allow each supported CGF to both verbally respond and correctly execute the commands in the evaluation scenario. In order to execute this verification without the use of human experts, we developed an automated verification tool to generate metrics for a qualitative and quantitative analysis. As each CGF is directed through the scenario, a log of scenario events and situational awareness data is generated. In addition, the tool generates a sequence of 'checks' regarding behaviors and responses expected for each controller command, which are then analyzed automatically. Discrepancies between the expected event sequence and the log sequence indicate problems with the generated CGF behavior.

In this verification test, each CGF connected to SID performed the tasks 100% correctly in terms of meeting each of the expected verification checks, which covered both mobility behavior (e.g., moving to waypoints) and communication behavior (e.g., responding to requests). The performance of each CGF was not equivalent, however. The auto-verifier also logged a number of metrics that gives us some deeper insight into how the systems varied in their execution, as shown in Table 1. While the CGFs each performed the tasks properly, there were some noticeable variations when inspected by human observers. For example, the AMS-based CGF (a ducted-fan model) had a minimum airspeed of 20m/s, so commands to travel slower than that speed were translated to the platform's minimum speed.

Table 1: Detailed CGF Performance Metrics.

Metric	AMS	Helo-Soar	Sim Jr	IDEEAS
Total distance(m)	4785	4409	4785	4785
Total Time (seconds)	429	374	357	373
Ave speed (m/s)	11.2	11.8	13.4	12.8
Assigned Speed(m/s)	20.0	12.0	12.0	12.0
Max route deviation(m)	68.12	58.39	79.41	73.96
Ave route deviation(m)	11.18	11.40	9.05	6.24
Ave max route dev. (m)	32.64	30.49	27.99	26.97
Ave min route deviation (m)	0.03	0.22	3.31	0.13

These variations are not unexpected, since we did not change the baseline simulation platform behavior envelopes for the evaluation, and the platforms are each implemented differently in each of their respective simulation environments.

4.2 Reduced Workload Setup and Results

This evaluation focuses on the amount of work that an operator must do to task an entity and maintain situational awareness about that entity's behavior to ensure it is making progress. For this evaluation, we measure workload in terms of the number of commands that a user would need to issue to a CGF to complete a task, including steps to ensure that the task has been completed (e.g., monitoring for reaching a waypoint). We expect that that SID should reduce the number of commands that a user would need to issue.

To obtain a result, we compare the number of commands issued at the user level against the number of commands issued at the CGF level. The CGF-level commands are counted from the number of messages sent over the network to control the CGFs. We treat this count as the total amount of work that a human operator would have needed to do if tasking the CGFs directly. However, even some primitive commands could not be executed by some of the CGFs (e.g., some could not report their own position on demand). In these cases, SID itself performed the monitoring to generate a report back to the user.

We again ran this evaluation across a range of CGFs because each CGF is different in terms of the primitive tasks that it can accept, the amount of reporting it can do, etc., and therefore the amount of user effort will vary across CGF systems. The results across all four CGF systems are shown in Table 2. As with the previous test, the user commands were delivered through SID via the evaluation scenario script. Each CGF received the same user-level commands in the same order. The table includes a breakdown of how much work each CGF was doing (given as *Task steps handled by CGF*) and how much work SID was doing on behalf of the CGF (given as *Task steps handled by SID*). The final metric is Work Savings, which we defined to be the rough measure of how much a user gains by using SID over tasking the CGF directly using its native interface. We compute Work Savings as:

$$\text{Work Savings} = 1 - \left(\frac{\text{\# User Commands Given}}{\text{\# Primitive Derived Actions}} \right)$$

Table 2: Workload Comparison per CGF

	AMS	IDEEAS	Helo-Soar	SimJr
Total # of user commands given	11	11	11	11
Task steps handled by CGF	23	16	10	23
Task steps handled by SID	42	21	27	42
Total # of primitive task steps	65	37	37	65
Rough Work Savings	**83%**	**70%**	**70%**	**83%**

Across these four different kinds of CGF systems, SID contributed on average 76.5% workload savings in terms of the number of commands that an operator would have had to issue to a CGF manually to accomplish the same task. This is actually a somewhat conservative measure, since it does not include the time and attention that an operator would have to spend deliberately watching the simulation display to monitor CGF progress, since SID provides verbal reports either by doctrine or by request.

5. Summary and Future Work

In this paper, we describe an evaluation of an intelligent user interface called the Smart Interaction Device (SID) that we have developed to help simulation operators interact with CGFs at a supervisory control level, as compared to performing low-level manipulations on CGFs. Part of this project included the development of an objective, automated verification tool that logs the behavior of the CGFs, and allows for collection of quantitative metrics (e.g., execution time) and qualitative metrics (e.g., completion of milestones) for use in an evaluation. Using the auto-verify tool, we have conducted a formative evaluation of SID to assess the potential workload savings for a user. We have tested features of SID that could positively affect the user's experience when directing CGFs: 1) that SID could allow the simulation operator to interact with categorically different CGFs in a uniform, high-level way, despite their underlying differences (tasking equivalence); 2) that SID could reduce the overall workload of an operator managing CGFs by providing a more natural interface and explicit verbal feedback about the CGFs.

The first evaluation demonstrated that all the CGFs in the experiment were able to complete the high-level tasks assigned to them, despite underlying differences

in how they could be natively tasked. However, there were some execution differences in timing and or how much deviation was seen in terms of ranges, speeds, etc. Generally, we feel the distinction between *task-level* behavior and *sub-task* behavior is a useful one for evaluating CGF behaviors. Other levels, such as *mission level* (was the mission completed successfully?), could also be introduced in the auto verification tool. Behavior verification and validation must be performed at an appropriate level: it may be perfectly valid for a CGF to be able to complete a task correctly while exhibiting minor sub-task variation.

The second evaluation demonstrated an average of 76.5% workload savings across a full mission while using SID (averaging across 4 different types of CGFs), where workload is measured in terms of the number of user-level commands issued to a CGF. SID can significantly reduce the amount of work that it takes to task a CGF directly in its native task language. The amount of work savings will vary per CGF. We anticipate that less capable CGFs (i.e., those that can only take very primitive commands) will see a greater relative effect in using SID compared to those CGFs that can already take higher-order commands and can do their own reporting. Even legacy CGFs that were not designed with user interaction in mind could, using SID, be made to appear interactive. In large-scale, multi-environment simulation exercises, another potential advantage of using SID is that it can provide a uniform interface to all the CGF systems involved, with the potential to reduce the training requirements across multiple CGF systems. SID's ability to reduce the workload of individual operators could help to lower the costs of running complex simulation exercises by reducing the number of simulation operators needed to manage multiple CGF systems.

This initial set of evaluations has given us some indication that SID can be beneficial for controlling CGF systems. The two features of SID we have demonstrated – high level interaction and reduced workload – are two aspects of supervisory control. However, more extensive human-in-the-loop studies are required to evaluate other facets of supervisory control. For example, including human operators in the evaluation would allow us to test hypotheses regarding span of control (i.e., how many CGFs an operator could effectively manage with and without SID) and operator situational awareness (with and without SID). More fine-grained measures of workload including time spent monitoring the situation would also give us better gauges on SID's contribution.

We are also working in other domains and other tasks to evaluate SID under different conditions. For example, this same core architecture is now being used to create IUIs to help human operators interact more effectively with autonomous robotic systems. This presents new challenges in terms of supervisory control, dialogue, situational awareness and task monitoring.

6. Acknowledgements

This work was funded under contract # W911W6-07-C-0052. Thanks to our teammates at Dynetics, Inc., and SAIC. Thanks to the US Army for selecting this project for a 2010 SBIR Achievement Award.

7. References

Colonna-Romano, J., Stacy, W., Weston, M., Roberts, T., Becker, M., Fox, S. and Puall, G. (2009). Virtual Puckster -- Behavior Generation for Army Small Team Training and Mission Rehearsal. Behavior Represenation in Modeling and Simulation (BRIMS), Sundance, UT.

DeKoven, E. and Murphy, A. K. G. (2006). A Framework for Supporting Teamwork between Humans and Autonomous Systems. Command and Control Research and Technology Symposium (CCRTS06), San Diego, CA.

Jones, R. M., Laird, J. E., Nielsen, P. E., Coulter, K. J., Kenny, P. and Koss, F. V. (1999). Automated Intelligent Pilots for Combat Flight Simulation. AI Magazine**:** 15.

Jones, R. M., Wallace, A. J. and Wessling, J. (2004). An Intelligent Synthetic Wingman for Army Rotary Wing Aircraft. IITSEC, Orlando, FL.

Miller, C., Funk, H., Wu, P., Goldman, R., Meisner, J. and Chapman, M. (2005). The Playbook Approach to Adaptive Automation. Human Factors and Ergonomics Society.

Stensrud, B., Taylor, G., Schricker, B., Montefusco, J. and Maddox, J. (2008). An Intelligent User Interface for Enhancing Computer Generated Forces. Simulation Interoperability Workshop, Orlando, FL.

Stent, A., Dowding, J., Gawron, J. M., Bratt, W. E. and Moore, R. (1999). The CommandTalk Spoken Dialogue System. Thirty-Seventh Annual Meeting of the ACL, University of Maryland, College Park, MD, Association for Computational Linguistics.

Wray, R. E. and Jones, R. M. (2005). An introduction to Soar as an agent architecture. Cognition and Multi-agent Interaction: From Cognitive Modeling to Social Simulation. Cambridge, UK, Cambridge University Press**:** 53-78.

Author Biographies

GLENN TAYLOR is a Senior Scientist at Soar Technology. His R&D activities include interactive cognitive, social, and cultural human behavior models, and natural interaction with autonomous systems. He received his BS and MS in computer science and engineering from the University of Michigan, and has been at Soar Technology since 1998.

BRIAN STENSRUD, Ph.D., is a Research Scientist at Soar Technology and lead behavior developer for the IUI described in this paper. Dr. Stensrud received his Ph.D. in Computer Engineering from the University of Central Florida in 2005. He also holds B.S. degrees in Computer Engineering, Electrical Engineering, and Mathematics from the University of Florida. Brian has over ten years experience in the areas of knowledge-based systems and artificial intelligence.

JEFFREY MADDOX is a System Engineer for the US Army RDECOM/AMRDEC at Redstone Arsenal, Alabama, in the Advanced Prototyping, Engineering and eXperimentation (APEX) Lab. He earned a B.S. in Electrical Engineering and a B.S. in Math and Physics Education, both from Auburn University.

HAL AYCOCK is a System Engineer for the US Army RDECOM/AMRDEC at Redstone Arsenal, Alabama, in the Advanced Prototyping, Engineering and eXperimentation (APEX) Lab. He earned a B.S.in Electrical Engineering from the University of Alabama.