Modeling Pilot Behavior For Assessing Integrated Alerting And Notification Systems On Flight Decks

Mathew Cover, Graduate Student; Thomas Schnell, Associate Professor
Operator Performance Laboratory, University of Iowa
mcover@engineering.uiowa.edu; tschnell@engineering.uiowa.edu

Abstract. Numerous new flight deck configurations for caution, warning, and alerts can be conceived; yet testing them with human-in-the-loop experiments to evaluate each one would not be practical. New sensors, instruments, and displays are being put into cockpits every day and this is particularly true as we enter the dawn of the Next Generation Air Transportation System (NextGen). By modeling pilot behavior in a computer simulation, an unlimited number of unique caution, warning, and alert configurations can be evaluated 24/7 by a computer. These computer simulations can then identify the most promising candidate formats to further evaluate in higher fidelity, but more costly, Human-in-the-loop (HITL) simulations. Evaluations using batch simulations with human performance models saves time, money, and enables a broader consideration of possible caution, warning, and alerting configurations for future flight decks.

1. INTRODUCTION

The aviation safety (AvSafe) program at NASA, is tasked with assuring that safety of current and future aircraft participating in the National Airspace System is always being evaluated and improved upon [7]. The Integrated, Intelligent Flight Deck (IIFD) program within Aviation Safety, has sponsored a research project at the University of Iowa's Operator Performance Laboratory to mitigate high crew workload and increase situational awareness in the operational NextGen environment. Specifically, the research project seeks to resolve conflicts in caution, warning, and alerts (CWAs) that may be presented to pilots. The mechanism in which this is done will be via a software solution called the integrated alerting and notification (IAN) function. This work is conducted in conjunction with Ohio University and is supported with efforts from Boeing, Rockwell Collins and Delft University of Technology (TUD).

2. ARCHITECTURE OVERVIEW

The architecture of our model is best introduced as a closed-loop control system where the aircraft state and IAN function are fed into a human model (Figure 1). The human model then analyzes the stimuli provided by the aircraft displays and sensors and responds accordingly, outputting feedback into the flight model.

The model needs to be run hundreds to thousands of times to test variations of the IAN function and displays on the human model. Using batch Monte Carlo simulations with a human model permits us to test out a wider variety of simulated avionics conditions and operational scenarios than could ever be feasible with HITL testing.

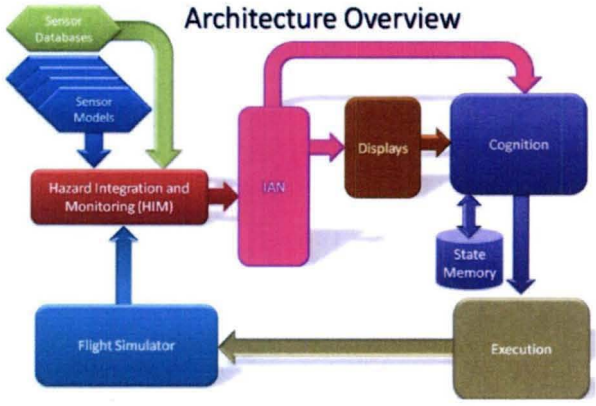

Figure 1: This figure outlines the top-level view of how the closed-loop model is constructed

A. Simulink

For the project model, Simulink provides a structure that many pieces of the model can plug into to complete the closed-loop architecture. Using some TCP/IP functions, programs that run outside of Simulink are able to communicate and participate in the model such as a flight simulator,

the human model, and display visualizations (Figure 2).

B. Flight Simulation Interface

The IAN project requires a realistic aircraft flight model for the human model to fly. Microsoft Flight Simulator X (FSX) has been chosen to run the flight model for this project. FSX provides *SimConnect*, a built-in interface that provides a standardized interface for add-on executable programs to communicate with and allows asynchronous communication over a network connection. These features make FSX ideal for multi-threaded applications and allow it to run out-of-process with the rest of the closed-loop model [6].

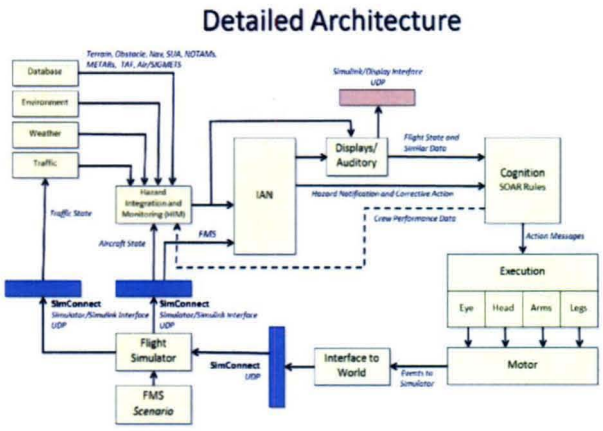

Figure 2: A more detailed view of the software architecture for the IAN / human model

C. Aircraft Sensors

As part of the aircraft model, additional sensors that make the aircraft NextGen-worthy are being added to the FSX flight model. Among the sensors that are being modeled and incorporated into the architecture are GNSS, ADS-B, TCAS, TAWS/EGPWS, WxR. Future work by Ohio University will include computer models of sensors such as FLIR, millimeter wave radar (MMWR), 3D imager, and Interferometer. All of these sensors are coordinated and filtered by the Hazard Integration and Monitoring (HIM) module. Relevant data is then passed along to the IAN function.

3. INTEGRATED ALERT AND NOTIFICATION (IAN)

The flight simulator is only a part of the closed-loop model. The human model will connect to a representation of a NextGen cockpit. This simulation models an important characteristic of the NextGen aircraft – an integrated alerting and notification system. One aspect of this functionality is the ability to sort through alerts and notification and resolve conflicting information prior to presenting it to the pilot. This system is named the integrated alert and notification (IAN) function, as mentioned previously.

This is critical as conflicts in the cockpit consume pilot's time and spare mental capacity to resolve and respond in what are usually challenging situations. For instance, an enhanced ground proximity warning system (EGPWS) may suggest increasing altitude to avoid a mountain while the traffic collision avoidance system (TCAS) may tell a pilot to lower altitude to avoid another aircraft nearby.

Work with Boeing and Rockwell Collins will be able to help provide a set of rules in the IAN function so that visual, auditory, and tactile cues do not conflict and convey information in a useful manner to the pilot. They will also be constructed so as not to fall out of line with standards and common practices followed today and envisioned for NextGen operations.

4. COGNITION

The following requirements were considered in the selection of a base modeling approach to incorporate into the IAN system modeling function for this research.

1) We wanted to make sure that the cognitive model could interact with an external environment in a software-feasible way. Having the best cognitive model in the world does us no favors if it is unable to communicate with the flight model and IAN function. This closed-loop architecture has been done with cognitive models previously with prime examples seen by ACT-R [1], Soar [5].

2) The cognitive model needs to demonstrate at least face validity; that is, it should accurately approximate how a pilot would behave in different

circumstances. It is felt that a rule-based system would best be suited for a model of an aviator.

Starting with a system that makes some simple assumptions about generic human cognitive performance allows for more time and effort to focus on tailoring the model to flying-related tasks.

3) In order to mimic a human interacting with an aircraft as much as possible, the interface needs to closely emulate the human body perceiving through the eyes and ears and manipulating with hands and feet. A software module/client, named the execution block, will be written that communicates with the cognitive model and creates a representation of arms, hands, legs, feet, head, and eyes. This helps unload the burden on the cognitive model of keeping track of the details regarding the sources of input and output.

4) We are concerned with efficiency. To this end, we aimed to select an approach that minimized development time, and leveraged existing work. In addition, the human model component is designed to run out-of-process, or asynchronously with the aircraft simulator to facilitate parallel development. Once a common interface is described, the human model should be able to fly many types of simulated aircraft that meet that software specification. This also permits development to not be dependent on the Simulink portions of the model in order to test and run it.

5) We are on a schedule and need to make sure that the cognitive model we select doesn't take an unnecessary amount of time to develop. It is undesirable to spend time re-inventing something that someone else has already done. Any existing models and architectures that exist out in the world should be considered as a potential baseline for our model and taken advantage of.

The following sections highlight some of the candidate cognitive models considered and some of the pros and cons of each with regard to their applicability toward our IAN model.

A. ACT-R/PM

ACT-R Perceptual-Motor (ACT-R/PM) is a set of extensions to ACT-R which provides perceptual-motor capabilities for ACT-R. The Perception-Motor layer is made up of modules that handle various aspects of perception and action. Among the modules covered with this extension to ACT-R are vision, motor, speech, and audition [2].

One of the difficulties of working with ACT-R is that it is written in Lisp, a powerful, yet older high-level programming language. While it is known for powering artificial intelligence research over the last several decades, it is not as prevalent as other high-level programming languages such as C, C++, and Java for general programming activities.

Another difficulty of ACT-R is that it traditionally is run as a stand-alone application where the cognitive model does not talk easily with any external application/devices/computers. That was resolved with release of the ACT-R/PM module which does incorporate the ability to interact with a simulated device easily. However, the simulated device must be a Lisp object which must have certain methods defined for it, which in turn, will be called automatically by ACT-R/PM at the appropriate times [2].

There also exists a version of ACT-R called jACT-R which is a java implementation of ACT-R. While not comprehensive of all features that ACT-R provides, it covers most of what one would expect of ACT-R, but written in Java rather than Lisp. jACT-R also provides some benefits to interact with external environments and control the models remotely [4].

B. Air MIDAS

Air Man-machine Integration Design and Analysis System (Air MIDAS) is a modeling and simulation tool designed to assess human-system integration in dynamic aviation-related environments. It is currently being used to analyze advanced air traffic management concepts at San Jose State University where it was originally developed by the Human Automation Integration Laboratory [3].

Like ACT-R, Air MIDAS is programmed in Lisp, although it interfaces with external simulators more natively. Air MIDAS also has the benefit that it was developed with an aviation emphasis and not just a generic cognitive model.

C. Soar

Soar is a theory of a cognitive model that is implemented as software architecture [10]. Soar research today tries to realize an approximation of human behavior and thought while minimizing the sets of mechanisms that are required. Soar memory is associative which means that the flow of control in Soar is not determined by a sequential, deterministic control structure that is used in most programming languages. In other words, evaluation of relevant knowledge can be done in parallel [10].

Another feature of Soar is the ability to automatically create sub goals to help resolve impasses in decision junctions. Soar also takes into account past experiences when adapting to unfamiliar situations and making decisions. This allows Soar to learn new conceptual knowledge, procedures, and even correct its knowledge as it gains feedback through experience in its environment [10].

The most recent release of Soar version 9.0.1 now includes a reinforcement learning (RL) module. Beta versions of Soar include episodic memory and semantic memory modules. These new memory and learning modules greatly enhanced Soar's ability to approximate human memory [9].

All of these positives, in terms of programming, come at the expense of under specifying the capabilities that must be built into intelligent agents. Most of the knowledge that a Soar agent has, stems from rules that have been programmed into it. For the agent to realize high-level intelligent behavior, the knowledge must be created. Soar also can make simplifications which leads to unrealistic behavior in the model [5].

Perhaps one of the greatest benefits Soar provides for a closed-loop model is a standardized way to connect to external simulators via a language called Soar Markup Language (SML). The method was debuted in Soar version 8.6, and has been supported since. The SML specification allows external programs to send and receive information from Soar which allows external simulations, such as a flight simulator and/or custom applications to interact with the human model easily [8].

D. Model Selection

Based upon the model considered, we have determined that Soar best meets our requirements to integrate into the IAN / human model. Soar provided the easiest method to integrate into the closed-loop architecture, provided models that already existed to build upon (Air-Soar, TacAir-Soar), and is a reasonable cognitive approximation with notable, but acceptable faults.

E. Model Extension for IAN

Our concept in the NextGen IAN functions will not focus on modeling perception as much as it will on comprehension and cognition. We are aware that perception of stimuli in the closed-loop simulation can be a factor in the evaluation of IAN functions; however, parameters such as font sizes, brightness, and contrast ratios are prescribed by detailed design specifications for flight decks. We assume that these same design specifications will be used in NextGen avionics. This minimizes the need to study perceptual parameters in the closed-loop simulation of the model.

However, we intend to use the closed-loop simulation, as described in this paper, to determine design specifications for the cognitive processing aspects of IAN functions. The number of simultaneous or near-simultaneous caution, warning, and alerts that may be presented to the pilot could be competing for scarce cognitive resources. Multiple stimuli could be subject to the psychological refractory period wherein the pilot may delay reaction to an important stimulus while attending another. It is these types of scenarios that we wish to use the human model to determine the best candidate IAN functions.

5. CREW PERFORMANCE DATA

The OPL has years of experience collecting data from human pilots in both aircraft and flight simulators to evaluate pilot performance and estimate workload. This capability is being developed under a separate NASA project entitled Operator State Sensor Investigations and Operator State Classification and Feedback Algorithms (NNL07AA00A). A significant piece of software has stemmed from this project called the Cognitive Avionics Toolset or CATS. This program

is used to provide real-time data exploration and analysis to support effective operator state feedback.

It is our hope that we may be able to also gain an understanding of what the workload is of the human model in the IAN simulation. The OPL has conducted several studies in airborne platforms and ground-based simulators that involved collecting workload measurements as well as physiological responses and eye tracking. Empirical data collected from such studies can help us fine-tune the human model for similar scenarios such as the standard terminal arrival route (STAR) approach and landing.

It is also of interest to allow the IAN function to be aware of pilot state during all phases of flight. Should, during long stretches of low workload, the pilot allow their attention to fall elsewhere (or nowhere at all, should they fall asleep), IAN will become aware of the pilot's inattention. How IAN presents information to an attentive versus an inattentive pilot should be quite different. For example, sounding audio cues in addition to presenting visual cues may be necessary to draw the pilot's attention to significant information.

6. FUTURE WORK

As part of the IAN / human model project, human-in-the-loop experiments will be conducted using the candidate formats down-selected by the IAN / human model. The CATS software developed by the OPL will be used as the primary collector of human performance data during these experiments. This allows us to estimate the workload of pilots in the simulator with the new IAN functions being tested. This provides a quantitative way of comparing the different IAN functions in terms of reducing workload for pilots and indicates desirable function and display formats.

After the IAN closed-loop model is able to successfully start testing candidate IAN functions and display properties, there will be a selection of the top four candidates. These top candidates will then be implemented in a flight simulator and flown as part of a pilot-in-the-loop experiment/study that will verify the properties and characteristics of the IAN system. The OPL houses and maintains several research aircraft and flight simulators, including a 737-800 fixed-based flight deck that we will use to conduct the human-in-the-loop study (Figure 3).

Figure 3: OPL's 737-800 Flight Simulation Facility

7. TEST SCENARIOS

Under consideration for scenarios to be tested with the human model are terminal approaches. In a NextGen aircraft, this would not typically involve much from the human model other than to act as a supervisor for the auto-pilot. In order to put the IAN system to the test, a circumstance that would cause pilot intervention needs be a part of the scenario. While these scenarios are still being finalized at the time of this writing, one scenario being developed is an aircraft receiving a STAR to O'Hare (ORD). This scenario would involve conflicting air traffic interfering with the intended route to the airport. Presentation of this information to the pilot(s) would progress from simple messages and advisories to full warnings (if action was not taken earlier).

8. ACKNOWLEDGEMENTS

We would like to acknowledge the National Aeronautics and Space Administration under grant number NNX08BA01A. We also would like to thank Dr. Maarten Uijt DeHaag and his great team at Ohio University. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NASA, the U.S. Government, or any other organization.

9. REFERENCES

1. Byrne, M.D.; Kirlik, A.; Fleetwood, M.D.; Huss, D.G.; Kosorukoff, A.; Lin, R.; Fick, C.S., 2004. "A closed-loop, ACT-R approach to modeling approach and landing with and without synthetic vision system (SVS) technology." *Proceedings of the Human Factors and Ergonomics Society 48th Annual Meeting.*
2. Byrne, Mike. ACT-R/PM Theoretical Notes. Computer-Human Interaction Lab at Rice University. 20 April 2009. http://chil.rice.edu/projects/RPM/theory_notes.html
3. Freund, Louis. San Jose State University. Telephone Interview. Conducted 15 July 2009.
4. Harrison, Anthony M. jACT-R. 20 July 2009. http://jactr.org
5. Jones, R.M. Laird, J. E., Nielsen, P.E., Coulter, K. J., Kenny, P., Koss, F. V. (1999). "Automated Intelligent Pilots for Combat Flight Simulation." *AI Magazine*. Spring, 1999.
6. Microsoft. "About SimConnect." 4 March 2009. http://www.fsinsider.com/developersPages/AboutSimConnect.aspx .
7. NASA, Aviation Safety Program. Aeronautics Research Mission Directorate. 20 July 2009. http://www.aeronautics.nasa.gov/programs_avsafe.htm .
8. Pearson, Douglas. "XML Interface to Soar (SML) Software Specification." August 28, 2008.
9. Soar Group. "Soar-EpMem Manual". 11 May 2009. http://sitemaker.umich.edu/soar/
10. Soar Technology. "Soar: A Functional Approach to General Intelligence." Ann Arbor: 2002. http://www.soartech.com .