A "Trust But Verify" Design for Course of Action Displays

Mark St. John, Daniel I. Manes, &
Pacific Science & Engineering Group, Inc
6310 Greenwich Dr., Ste 200
San Diego, CA 92122

Glenn A. Osga
Space and Naval Warfare System Center, San Diego
53560 Hull Street
San Diego, CA 92152

Abstract

Automation, particularly of complex cognitive tasks, is bound to be incomplete, simplistic, or otherwise less than completely reliable. Recently, we have begun developing "Trust but Verify" techniques for increasing the effectiveness of even unreliable automation. The user's trust should be conditioned on known situational factors that affect the reliability of the automation, and users should be able to verify the automation's results and operation to various qualitative degrees as the level of trust dictates. Here, we describe our preliminary work on these concepts in the domain of Course of Action (COA) selection for an Intruder Interception Task. This task involves deciding which of several available aircraft should be chosen to perform an interception of an unknown aircraft intruding into the air space. Based on repeated interviews with four subject matter experts, we identified and then distilled a set of factors essential to evaluating the optimal COA. We then designed a set of alternative displays to illustrate the factors based on the Trust but Verify concept and general human factors display guidance. Here we analyze the benefits and costs of two major design decisions: whether to display the COA factors using a tabular or graphic organization, and whether or how to integrate the COAs with the map or with each other in a common table.

Human Factors in Automation Design

Automation of many cognitive tasks, including command and control tasks, is becoming a reality. The trouble, of course, is that automation, particularly of complex cognitive tasks, is bound to be incomplete, simplistic, or otherwise less than completely reliable. One approach to resolving this trouble is to assign human operators to monitor the automation and quickly jump in and perform the task manually whenever the automation fails. Unfortunately, this approach has a number of drawbacks. First, it is tedious for operators while the automation is functioning correctly. Second, workload can become overwhelming when the automation fails because operators must quickly notice the failure, gain situation awareness, and make their own analyses and decisions. Numerous studies have confirmed the difficulty (see Parasuraman and Moulana, 1996).

A useful step toward understanding and resolving this difficulty is to define carefully the concept of automation for cognitive tasks and develop a taxonomy of its types. Sheridan (1987) has laid out a concept of supervisory control of automation that

*Acknowledgements***.** Many thanks to Harvey Smallman for his comments on a draft of this manuscript, and to Russell Frevele, Ron Moore, Gene Averett, and Marty Linville for their participation as Subject Matter Experts. This work was supported by the Office of Naval Research through the Space and Naval Warfare System Center, San Diego.

has several embedded "layers": plan, teach, monitor, intervene, and learn. Planning and learning represent the outer-most layers of supervision while teaching and intervening are nested within, and monitoring lies at the core. Within monitoring, there is a second taxonomic factor: the "level" of automation. The level of automation varies from complete manual control, to the automation offering a complete set of alternatives for the operator's consideration, to the automation offering a preferred set of alternatives, and so on, to the automation performing the task with little or no oversight by operators. The level of automation and operator supervision depends on factors such as the reliability of the automation and the consequences of automation failure. Lower reliability or graver consequences demand closer and more thorough supervision to maintain accuracy. On the other hand, more thorough supervision reduces efficiency by requiring more work and more monitoring from operators. A third taxonomic factor of automation is the "type" of task that is automated. Parasuraman, Sheridan, & Wickens (2000) classify tasks according to stages of information processing: information acquisition, information analysis, decision and action selection, and action implementation.

Most human factors research into automation and interface design tends to focus on the two more central types, information analysis and decision selection. The research asks how users respond when these types of automation fail, and how user interfaces should be designed so as to improve users' responses to these failures. In most laboratory studies, automation failures occur in one of two paradigms, signal detection errors and catastrophic breakdown. In the signal detection paradigm, automation failures occur as either misses or false alarms. Recent work on headway proximity detectors for automobiles is an example (e.g. Dingus et al., 1997; Wiese & Lee, 2001). Misses occur when the automation fails to detect a closing distance between cars, and false alarms occur when the automation signals a closing distance when the distance is not in fact closing. False alarms can be triggered by passing cars and even sign posts along the side of the highway.

Since misses can result in rear end collisions, the automation is biased toward making false alarms. Because of this bias, however, when an alarm does occur, the operator must first assess the situation to determine if a problem actually exists, and then work to resolve it. System efficiency comes from allowing the operator to pay relatively little attention to the task until an alarm sounds. Inefficiency occurs when false alarms force the operator to pay attention unnecessarily. Inefficiency also occurs when the operator pays attention in the absence of an alarm to check for misses. The goals for user interface design in this paradigm are to increase the efficiency of checking for misses, decrease the disruption caused by false alarms, and increase the efficiency and speed of gaining situation awareness to respond to alarms.

The second paradigm of automation failure involves the catastrophic breakdown of the automation. The automation suddenly and unpredictably stops functioning entirely as if it were unplugged (Parasuraman & Riley, 1997). If a failure occurs, then the user must notice it as quickly as possible, recover situation awareness, and take manual control over the system. Unfortunately, studies show that this does not work out very well: users may not notice the failure, may have difficulty recovering situation awareness quickly, and may have lost manual skill (Parasuraman and Riley, 1997). The increased efficiency of allowing operators to pay less attention, therefore, comes at the price of more errors and more time to regain situation awareness when it is needed. On the other

hand, increasing operator vigilance reduces the efficiency of the system. The goals for user interface design are to increase the efficiency of checking for breakdown and other errors and increase the ease of regaining situation awareness and manual control.

In both of these paradigms, automation failure is treated as an unpredictable and complete failure. There are no degrees or certainty factors for correct or incorrect performance. Because the failures are unpredictable, operators cannot be warned when to pay closer attention. They must pay attention all the time. Because the failures are complete, operators are redundant when the automation is working and entirely on their own when the automation fails.

There are many cases, however, in which automation error can be predicted, at least to some degree, and where the automation fails in only partial and predictable ways. Recent work on "fuzzy signal detection" (Parasuraman, Masalonis, & Hancock, 2000), for example, seeks to make automation failures in the signal detection paradigm more graded. Rather than labeling all signals below a certain cut-off as noise, and possibly missing some weak signals, the idea is to label signals in a graded fashion so that strong signals can be noticed and addressed quickly, and weaker signals can be examined at the user's discretion. Additionally, some automation failures may be predicted based on knowledge of prevailing conditions. When these conditions occur, users can increase their skepticism and verification procedures. For example, atmospheric conditions may corrupt a sensor and thereby reduce, but not eliminate, the automated detection of problem situations. An automated algorithm might also fail to account for a rare variable that affects a situation. For example, freshly oiled roads can increase stopping distance and thereby make normally safe headway distances unsafe. If the automation does not take this factor into account, it may fail to sound alarms appropriately. However, since this potential automation failure is predictable, savvy users could increase their own monitoring of headway when they detect freshly oiled roads.

The "Trust but Verify" Design Concept

The idea that automation failure can occur by varying degrees and in more or less predictable ways and at predictable times can be exploited in powerful ways. Recently, we have begun developing techniques to use this insight to improve the automation-user interface. The two techniques that are discussed here can be summed up by the design concept "Trust but Verify". Normally, work on automation reliability places the focus on the level of automation: what level of automation is appropriate for a task given the level of risk and automation reliability (Parasuraman, Sheridan, & Wickens, 2000). Our idea of Trust but Verify is to place the focus on the users and the users' tasks: the degree of trust, given the current conditions, and the degree of verification necessary to resolve the distrust. In other words, the focus is placed on the users and what they do *with* the automation.

The idea of the Trust but Verify design concept is based on the premise that the level of reliability of any automation is not context free. It varies depending on the conditions under which it is used. The supervisor's trust and use of the automation should be modulated by these conditions. We call this concept "Conditional Trust". Exercising this concept requires that the supervisor understand the automation's capabilities and limitations and how conditions are likely to affect it.

The use of automation is also modulated by a second process called "Qualitative Verification". The idea is that supervisors should be able to verify the automation's results and operation to various qualitative degrees as the circumstances dictate. Under trustworthy circumstances the supervisor would be able to perform cursory checks, and under less trustworthy circumstances, the supervisor would be able to perform more careful and thorough checks. Qualitative Verification can be implemented in numerous ways depending on the task. The key is to provide multiple levels of support that allow supervisors to scan the situation quickly and easily to find the most relevant aspects of a situation and to assess those aspects and the automation "at a glance" and also allow supervisors to drill down into the details for more thorough checking when it is appropriate.

In addition to supporting the modulation of use and trust of the automation, Qualitative Verification should help guard against complacency. The idea is to make at least cursory checking easy enough that it can be accomplished even by busy or overburdened supervisors. The most essential information ought to be available constantly on a display without the need for any display interaction. If qualitative checking is made easy enough, supervisors should not become complacent by failing to perform any checks.

We have been developing these concepts in several different domains: alert message management (St. John, Oonk, & Osga, 2000), target recognition, and threat assessment. Here, we describe our preliminary work in the domain of Course of Action (COA) selection.

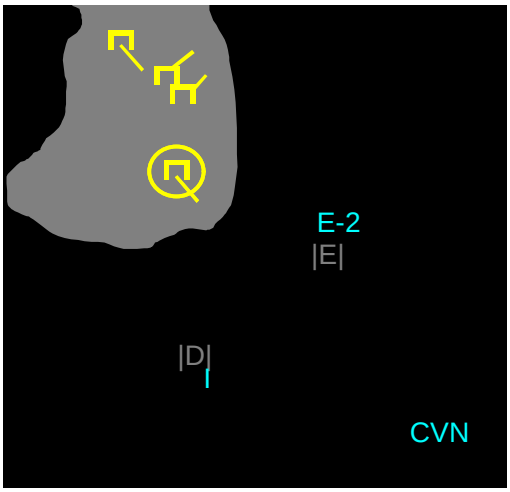


Figure 1. An intruder interception problem. The intruder (highlighted by the yellow circle) could be intercepted by the E-2 aircraft at station E, the interceptor aircraft at station D, or a relief interceptor aircraft of the carrier (CVN).

Trust But Verify Design in the Intruder Interception Task

The intruder interception task (See Figure 1) involves deciding which of several available aircraft should be chosen to perform an interception of an unknown aircraft intruding into the air space. This COA selection task is useful for studying automation and automation-user interface concepts because it is complex and time pressured. The task requires careful weighing and integration of many variables and making predictions

about future states of aircraft and schedules. For example, an intercept and escort implies a different fuel consumption than an intercept and Visual Identification (VID) only. Similarly, different aircraft and interception speeds imply different fuel consumption. And the most direct route to an interception might violate numerous airspace restrictions, while a longer route either takes longer or requires more fuel. The task grows even more difficult if the user attempts to consider the implications of delaying interceptions rather than ordering them immediately. For example, if an interceptor loiters for an additional 15 minutes and then performs the intercept, what will the interceptor's fuel levels be upon arrival at the interception point?

Our conception was that the automation would be used in the following way. The user would identify an intruding aircraft and initiate an automated process of identification and evaluation of alternative courses of action, namely, identification and evaluation of candidate interceptors and their routes to an interception point. The automation would then display the alternatives and their evaluation to the user, and the user would make the final selection.

Willis (2001) has recently been investigating the application of automation to the evaluation and visualization of different COAs for retargeting of tactical Tomahawk cruise missile while they are in flight. The users in that domain are faced with a comparably difficult set of mental computations to carry out on the fly and automation designers are faced with an equally tough set of design decisions.

Automation for the identification, evaluation, and visualization of multiple COA options could have a number of benefits. One benefit is the evaluation of a wider variety of options and a commensurate decrease in tunnel vision. Time pressured users might focus on one or a few options and fail to consider other possibilities that might ultimately prove to be better choices (c.f. Zsambok & Klein, 1997). The automation would make more alternatives available and visible to the user for consideration. A second benefit is the potential for fuller evaluation of more options. Time pressured users might not have the time or resources to consider more than a few options in any detail, but the automation would allow many more options to receive detailed consideration. A third benefit is the potential for the automation to identify problems with various options that might not be immediately clear to time pressured users.

Such automation, however, must be used with sophistication. Realistically, operational automation will always be fallible. To be used effectively, the user must guard against these failures. The difficulty for the user is to receive some advantage from the automation without becoming complacent on the one hand and without performing the task entirely manually on the other hand.

The Trust But Verify design concept suggests several design ideas to support sophisticated use of automation. The Conditional Trust half of the concept suggests that the user should be aware of the factors that the automation is using and how current conditions affect the automation's reliability. There are three categories of conditions that can modulate the level of reliability of the automation: data quality, algorithm quality, and user quality. Data quality is a straightforward matter of knowing whether sensors are operating correctly, are degraded by environmental factors, or data links are intermittent or down. Algorithm quality is the basic reliability of the automation algorithm even under the best of circumstances. Algorithm quality also involves knowing if any "out of bounds" conditions are in effect. Out of bounds conditions are

conditions that were not planned for during the development of the algorithm. An example might be an algorithm that was designed prior to the deployment of various aircraft or weapons, or an algorithm that was designed for the strategic conditions of one operating theater but then used in another theater. Algorithm quality also involves knowledge of any pertinent "extra-algorithmic factors" that may affect the evaluation of a course of action. Extra-algorithmic are factors that were not included in the algorithm, such as pilot experience or operational guidelines such as injunctions to minimize fuel wastage.

The third category of automation reliability factors is user quality. Trust and use of automation depends both on the reliability of the automation and the self-aware reliability, or quality, of the users (Lee & Moray, 1994). If users trust the automation to evaluate a situation or COA more than they trust themselves to do it, then they are more likely to use the automation. User quality depends on a number of factors including skill level with the manual task, current workload, and fatigue. For example, fatigued and overloaded users might choose to rely on even imperfect automation for a given task because the automation is likely to perform better than the "operationally degraded" users themselves.

Table 1
Categories of Condition Trust

Categories	**Factors**
Data Quality	Missing data (e.g., down links)
	Unreliable data (e.g., due to weather)
Algorithm Quality	Algorithm design/sophistication
	Out of bounds conditions
	Extra-algorithmic factors
User Quality	Skill Level
	Workload
	Fatigue

The Qualitative Verification half of the Trust But Verify equation suggests that methods should be available to the user for verifying or checking the automation and its results. The user should be able to see the potential options, see how the algorithm evaluated them on the known factors, and see how the options ranked against other options. The qualitative aspect of the verification implies that the user should be able to check the automation to varying degrees of depth. More and deeper verification can be used in less trustworthy conditions or when users have more opportunity or skill.

On the more qualitative end of the verification spectrum, users could view the computer-preferred option and verify its suitability. Or users could view a set of options and compare their suitability. Graphical portrayals of the automation's options and their evaluation would allow users to see, and possibly check or discount, the automation's evaluations. Users could verify the analyses or even verify the data on which the analyses are based. Greater user involvement could include having users generate options for the automation to evaluate. The automation could identify potential issues and problems and remind users of relevant factors. In this scenario, the automation, in effect, checks the users rather than the users checking the automation. Finally, the

automation could be used simply as a calculation tool and record keeper for comparing options and playing "what if?" games. These levels of verification roughly correspond with Parasuraman et al.'s (2000) levels of automation, but here, the emphasis is on what users do rather than on what the automation does.

This emphasis on users and their strategies makes clear the importance of user-automation interfaces that afford multiple levels of verification and that clearly indicate conditions of data and algorithm quality that lead to more or less trust and require more or less verification. The following is a list of design features suggested by the design concept of Trust but Verify.

Conditional Trust

- Data quality
  - Missing data: indicate data missing from the algorithm
  - Unreliable data: indicate information reliability
- Algorithm quality
  - Algorithm design: show factors included in the algorithm
  - Algorithm design: show the relative weighting of the factors (e.g. their rank order of important or influence)
  - Out of bounds conditions: indicate features of the current situation that violate assumptions of the algorithm (e.g. new operational areas, different steaming or alert conditions)
  - Extra-algorithmic factors: show factors and data known to be excluded from the algorithm
- User quality
  - User skill, workload, fatigue: it is not clear how or if these conditional trust factors should be displayed to users

Qualitative Verification

- Show multiple options
  - Display the relative merit of each option with scores on each factor
  - Display a visualization of each option on the geoplot
- Indicate top ranked option
- Identify secondary problems associated with each option
- Facilitate drill down capability for closer examination of data underlying options

Intruder Interception COA Design

Algorithm factors. The first step of the automation algorithm development was to identify the factors that are used to evaluate interception COAs and then distill them down to a small list of key factors to implement in the algorithm and display to users. This task analysis consisted of a series of knowledge elicitation sessions with four Subject Matter Experts (SMEs). All four SMEs were former Navy personnel with multiple years of experience in shipboard command and control environments and who were familiar with the tasks and duties involved in air warfare including Air Intercept Coordinator, Air Warfare Commander, and Red Crown. We relied on unstructured interviews which evolved over time, since this technique is widely considered to be a useful starting point in gaining familiarization with a previously unknown domain (Kirwan & Ainsworth, 1982). From the data collected via this process, we were able to

isolate 14 factors that decision makers would likely consider and utilize in ranking and selecting COAs. The more important factors are listed at the top.

1) platform of interceptor
2) weapons aboard interceptor
3) enemy Weapon Release Line (WRL)
4) angle of interception
5) estimated time-to-intercept
6) fuel adequacy for intercept only
7) fuel adequacy for intercept plus escort
8) availability of a tanker
9) exposure to enemy fire involved in approaching the target
10) presence of off-limit airspaces along approaching path to the target
11) pilot proficiency
12) aircraft reliability/status
13) scheduling conflicts
14) logistics

Through further interviews and careful analysis, we were able to distill and combine the initial set of factors into four factors that appeared to be the most appropriate for inclusion in the automation algorithm—range, weapons, fuel, and tanker availability. *Range* refers to the distance between the center of the battle group and the optimal point of interception. The optimal point of interception lies along the projected path of the target and is placed so that 1) the travel distance of the interceptor is minimized, 2) the interception occurs outside the WRL if possible, 3) off-limit airspaces are avoided, and 4) exposure to enemy fire is minimized. Since the identity of the target aircraft is frequently uncertain, it's weapons and capabilities would also be unknown. For these situations, a worst-case estimate of WRL, such as 100 nm would be used. Note that range encompasses several factors from the initial list (items 3, 4, 8, and 9).

The remaining three factors are more straightforward. *Weapons* is a numerical description of how many of three different types of weapons are aboard the interceptor. The weapons are 1) Sidewinders, 2) AMRAAMs (Advanced Medium-Range Air-to-Air Missiles), and 3) Phoenixes. The quantities of these weapons can be expressed as a three digit number (e.g., 000 for no weapons of any type or 123 for one Sidewinder, two AMRAAMs, and three Phoenixes). *Fuel* is a categorical description of what mission types are possible given current fuel levels and assuming minimal fuel efficiency throughout all flights. The possible values are Low (insufficient for an intercept), Icept (sufficient for an intercept only), and Escort (sufficient for an intercept and extended escort). If insufficient fuel remains for an intercept or escort, the final variable—tanker—would come into play. *Tanker* is also a categorical variable that can take on any of the following values: Far (too far away to provide refueling), In Use (already involved in another mission), Busy (available but currently in high demand), and Ready (available with minimal cost).

Interface design. With our newly acquired understanding of the interception task, we set out to design an automated interface that would plainly illustrate the relative

merits of each COA under consideration. An important goal was to ensure that the automation would not only instill trust in decision makers but also support and even encourage them to thoroughly verify the results of the automation.

The design of the display showing the COA factors and their evaluation by the automation algorithm began with several discussions and brainstorming sessions. Next, we began generating sketches that reflected the results of our discussions and our past experience with designing displays for similar tasks. At several junctures, we presented our display sketches to SMEs to gain their insight and feedback. This led to several revisions. Finally, we generated interactive storyboards of the more promising designs.

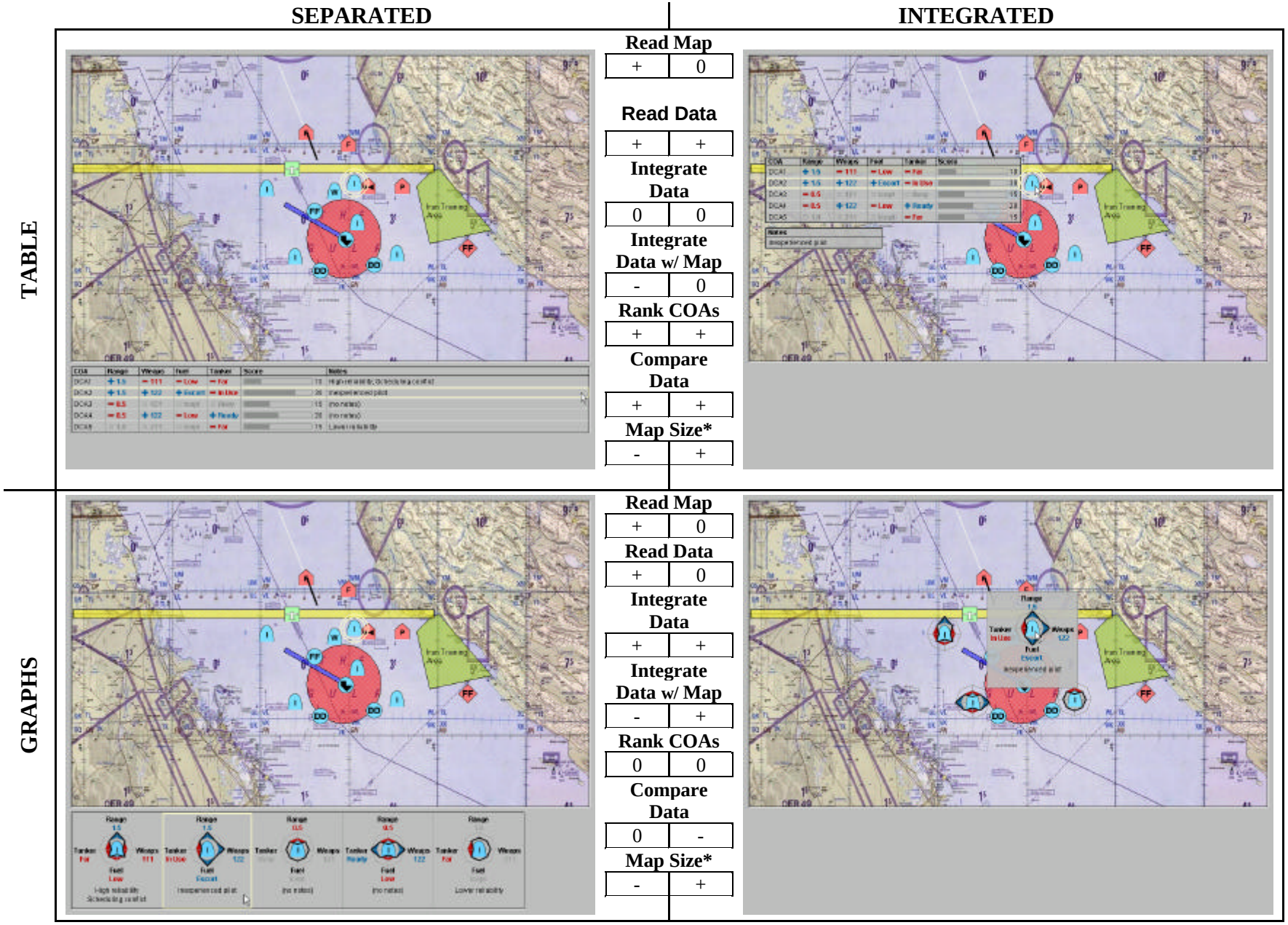
In the process of designing the display showing the four automation factors, we considered and debated a number of issues, including two major design choices, COA display format and degree of integration with a geographical display. Our initial conception of the tool consisted of a map as well as display elements supporting visualization of the current values of the four automation factors. It was also clear from the outset that we would need to build flexibility into the tool so that decision makers could easily remember and track extra-algorithmic factors. The following is a list of design issues addressed in this version 1 design.

- Whether and how to show a map? What scale? What information?
- How to display any idiosyncratic extra-algorithmic factors?
- How to display the COA factors? Tabular or graphic organization?
- How to integrate the COAs? With the map or with each other in a table or figure?

We deemed the map representation a necessity for its ability to support situation awareness, maintain continuity with other tasks and displays, and assist in the visualization of the tracks and route geometry involved in each proposed COA. We determined that the map would need to depict, at minimum, the region in question and the locations of the battle group and each track in the area. In addition, the map could also present air lanes, no-fly zones, and various other features that might impact the vectoring of interceptors.

While the four automation factors encompass the most significant items from the initial list of 14 factors, the interface would need to be flexible enough to account for the more idiosyncratic factors (items 11 through 14) and any additional factors not revealed by our interviews and analyses. Through brainstorming and discussion, we arrived at a "sticky note" concept as a potentially powerful catchall mechanism for idiosyncratic and extra-algorithmic factors. Specifically, the decision maker would be provided with a simple interface for making notes particular to each interceptor (e.g., "Inexperienced pilot" or "Needed for other mission") and for reviewing them later in context. This mechanism could potentially serve as a valuable decision and memory aid for decision makers working the current shift as well as decision makers working subsequent shifts.

For display format, we considered either presenting the COA data in a tabular or graphical format. For degree of integration, we weighed the merits of overlaying the COA data directly on the map (adjacent to the corresponding interceptor aircraft symbol) versus placing the COA display below the map. By crossing format and integration, we were able to create four alternative display configurations which are shown in Figure 2.



*Refers to potential map size (i.e., the map could be rendered taller for the integrated displays).

Figure 2. Ratings of the four display configurations for seven factors involved in choosing intercept COAs. The possible ratings were positive (+), negative (-), or indifferent/mixed (0).

Each configuration shows a map with an intruder aircraft, five own force aircraft that could potentially perform the intruder interception, and the five COAs, one for each aircraft. The Separated-Table configuration shows five potential COAs in a tabular format, separated from the map display. Each row of the table represents one COA, and each column represents a factor in the algorithm's analysis. The penultimate column shows a summary score and the final column is reserved for notes such as pilot experience. Selecting a COA row automatically selects the corresponding symbol on the map for that interceptor aircraft. This "dynamic visual linking" between the table and map provides a measure of integration. Nonetheless, the displays are separated in space, and they use very different formats.

The Integrated-Table configuration brings the table directly onto the map as a pop-up window that is displayed when an intruder interception task is triggered. This change increases the proximity of the COA data, which is known to be helpful (Wickens & Carswell, 1985), but does not increase the compatibility of the representations. On the other hand, placing the table on the map obscures sections of the map. A variant of the Integrated-Table scheme is to show only one COA in the table at a time. This variant obscures less map, but reduces users' ability to compare the COAs.

The Separated-Graph configuration shows the five potential COAs as "spider graphs", separated from the map. The concept of spider graphs is to integrate the evaluation factors into a single graphical figure rather than as separate columns of a table. "In addition to its objectness, a potential advantage of this display is that certain [situations], because of the nature of the parameter changes, cause a unique shape or configuration of the polygon to emerge" (Wickens, pp. 96-101). We chose spider graphs in lieu of other types of graphs, such as bar graphs, because the integrated spider graphs better support visualization of all dimensions at once, as prescribed by the Proximity Compatibility Principle (Wickens & Carswell, 1995). Since a major aspect of choosing the best course of action is visualizing the combination of variables at once, we believed that visually combining these variables would aid decision making.

Each spider graph begins as an octagon indicating normal or neutral values on the four factors. Deviations from neutral values are indicated by deviations from the octagon shape. Favorable values are represented by blue regions that extend beyond the neutral octagon, while unfavorable values are shown as red regions appearing within the inner octagon. The center of the graph shows a standard MIL-STD2525B military symbol for the type of aircraft involved in that interception (US DoD, 1999). Notes appear below the spider graph. Figure 3 shows a close up of a COA spider graph.

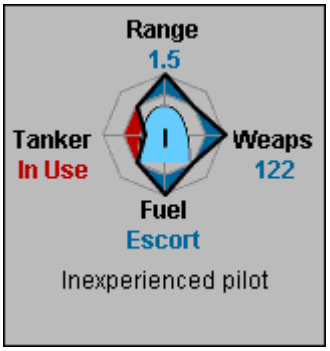


Figure 3. Close-up of a spider graph. See the text for details

The Integrated-Graph configuration shows the five COA spider graphs embedded into the map. Each COA graph is displayed directly on top of its corresponding aircraft. Normally, graphs on the map are shown in a reduced format in which the labels are removed, but the shape of the graph is still visible. This format reduces clutter on the map. When a graph is selected, it expands to full size and detail, revealing labels for each axis of the graph, the current values of each factor, and any notes.

We compared the four configurations on a number of criteria. For each criterion, we rated the four displays on whether they had a positive, negative, or indifferent/mixed effect on usability for the interception task. The ratings on the criteria are also shown in Figure 2. Since tactical decision makers tend to devote a large share of attention to maps (Kaempf, Wolf & Miller, 1993), the first criterion judged the four displays on how well they supported the task of map reading by minimizing clutter and the occlusion of map data. The separated displays received positive ratings because they provide an uncluttered view of all map elements and symbols, and vital map symbols are never occluded by data. The integrated displays received mixed ratings because they both involve overlaying information on top of map data. However, these overlays are presented only when a user selects an aircraft/COA. For the table configuration, if no COA is selected, the table disappears, leaving no extra clutter. For the graph configuration, if no COA is selected, the graph returns to its reduced size which is small and so tightly coupled to its aircraft symbol that it appears to generate little additional clutter.

We evaluated four additional criteria relating to the presentation of the COA factor data—how well the data for each factor could be read individually, integrated with each other, integrated with the symbols on the map, and compared between COAs. The table displays excel when it comes to reading the individual values and comparing values across COAs, while the graph displays provide better support for integrating the different factors within a COA. The Integrated-Graph display, in particular, provides superior integration of different data values within a COA as well as integration of the COA with other map-based data. However, it is at a slight disadvantage when it comes to reading actual data values because the user must first select the COA symbol, and it is at a sizeable disadvantage when it comes to comparing a single dimension across COAs because the graphs are not aligned with one another as they are in either Table display or the Separated-Graph display. On the other hand, tables are tailor-made for comparisons along particular factor dimensions across COAs.

The final criterion, map size, clearly favors the integrated displays. Map size is simply an issue of space utilization, as having the data separated and external from the map necessarily uses screen real estate that could otherwise be devoted to the map. In Figure 2, while the maps are currently rendered to be equivalent with the separated displays, the space freed up by overlaying the data directly on the map could be used to provide a larger, and, hence, more readable map.

As can be seen in Figure 2, the Integrated-Table configuration is favorable on four criteria and unfavorable on none. Each of the other three configurations is favorable on only three criteria and unfavorable on at least one other criterion. Of course, this simple analysis assumes equal weight for each criterion. If, for example, integrating the COA information with the map is very important, then the Integrated-Graph configuration might prove to be the best display. More discussion with our SMEs and more task analysis is required to resolve this issue.

Future Directions

Though substantial progress on the design of the automation algorithm and the interface has been made, much work remains to be done. The following is a partial list of issues for future research and development.

- Algorithm design
  - determine weights or rankings for factors in the algorithm
  - determine factor combination methods (e.g. linear combination)
  - evaluate algorithm reliability and robustness
- Interface design
  - Resolve issue of importance of integration between COA data and other map data
  - show route for each COA and allow adjustment of waypoints etc.
- Automation complacency and trust
  - Inclusion of more Trust but Verify features, such as indicators of missing and unreliable data
  - Empirical evaluation of display features for promoting trust and reducing complacency

Summary

Human factors research into the design of interfaces between automation and their human users tends to treat automation failure as unpredictable, context free, and catastrophic. Consequently, the user must either over-monitor to catch failures or under-monitor and miss mistakes. The happy medium is difficult to attain. The concept of Trust but Verify design is proposed as a way to ease this difficulty by recognizing two insights. First, automation failure may be at least partially predictable in that failures may be more likely to occur in some situations than in others. Savvy users may be able to take advantage of this context sensitivity by modulating their monitoring of the automation according to the situation and its likely effects on the automation's reliability and performance. We call this modulation of reliability and trust "Conditional Trust". A well designed automation-user interface would make the relevant conditions obvious to the user. The second insight is that modulating monitoring requires the ability to check or verify the automation's performance to greater or lesser degrees. We call this modulation of monitoring "Qualitative Verification". A well designed automation-user interface will support multiple levels verification from very qualitative to highly detailed.

Table 1 lays out a set of factors that could modulate the automation's reliability and potential for failure. These factors are divided into three categories: Data quality, algorithm quality, and user quality. These factors are developed into a set of design guidelines for an interface within the context of an automated decision support tool for course of action selection.

Next, we described our research into the design of an algorithm and an interface for a course of action selection support tool for the task of choosing an interceptor for an intruding aircraft. In developing the interface, we examined several additional design factors for how to visualize the automation's analyses and recommendations to the user. Two key issues were the format of the COA display, tabular or graphic, and where to place the COAs, integrated with the map or separated from the map but integrated with

each other. We crossed these alternatives to create four configurations, and we evaluated these configurations on a number of criteria. Finally, future developments for the interface were described.

References

Dingus, T. A., McGehee, D. V., Manakkal, N., Jahns, S. K., Carney, C., & Hankey, J. M. (1997). Human factors field evaluation of automotive headway maintenance/collision warning devices. *Human Factors, 39,* 216-229.

Kaempf, G.L. Wolf, S., Miller, T.E. (1993). Decision making in the Aegis combat information center. In *Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting.* Santa Monica, CA: Human Factors and Ergonomics Society. pp. 1107-1111.

Kirwan, B., & Ainsworth, L. K. (1992). *A guide to task analysis*. London: Taylor and Francis.

Lee, J. D. & Moray, N. (1994). Trust, self-confidence, and operators' adaptation to automation. *International Journal of Human-Computer Studies, 40,* 153-184.

Parasuraman, R. & Moulana, M. (1996) (Eds.) *Automation and human performance: theory and applications*. Mahwah, NJ: Lawrence Erlbaum.

Parasuraman, R., Masalonis, A. J., & Hancock, P. A. (2000). Fuzzy signal detection theory: Basic postulates and formulas for analyzing human and machine performance. *Human Factors, 42,* 636-659.

Parasuraman, R. & Riley, V. (1997). Humans and automation: Use, misuse, disuse, abuse. *Human Factors, 39,* 230-253.

Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans, 30,* 286-297.

Sheridan, T. B. (1987). Supervisory control. In G. Salvendy (Ed.) *Handbook of human factors*. New York: John Wiley & Sons.

St. John, M., Oonk, H. M., & Osga, G. A. (2000). Designing displays for command and control supervision: Contextualizing alerts and "trust-but-verify" automation. In *Proceedings of the Human Factors and Ergonomics Society 44th Annual Meeting*Santa Monica, CA: Human Factors and Ergonomics Society. pp. 646-649.

U.S. Department of Defense (1999). Department of Defense Interface Standard: Common Warfighting Symbology: MIL-STD-2525B. *Reston, VA: DISA/JIEO/CFS.*

Wickens, C. D. (1992). *Engineering Psychology and Human Performance* (2nd ed.). New York: Harper Collins.

Wickens, C. D., & Carswell, C. M. (1995). The proximity compatibility principle: its psychological foundation and relevance to display design. *Human Factors, 37*(3), 473-494.

Wiese, E. & Lee, J. D. (2001). Effects of multiple auditory alerts for in-vehicle information systems on driver attitudes and performance. *Proceedings of the Human Factors and Ergonomics Society 45th Annual Meeting.* Santa Monica, CA: HFES. pp. 1632-1636.

Willis, R.A. (2001) Effect of display design and situation complexity on operator performance. *Proceedings of the Human Factors and Ergonomics Society 45th Annual Meeting.* Santa Monica, CA: HFES. pp. 346-350.

Zsambok, C. E. and G. Klein (1997) Naturalistic Decision Making. Mahwah, NJ: Lawrence Erlbaum Associates.