Enabling Dynamic Delegation Interactions with Multiple Unmanned Vehicles; Flexibility from Top to Bottom

Christopher A. Miller[1], Mark Draper[2], Joshua D. Hamell[1],
Gloria Calhoun[2], Timothy Barry[3], and Heath Ruff[3]

[1] Smart Information Flow Technologies (SIFT), 211 First St. N. #300,
Minneapolis, MN USA 55401
[2] Supervisory Control and Cognition Branch, 711th Human Performance Wing,
Air Force Research Laboratory, 2255 H Street,
Wright-Patterson Air Force Base, Ohio 45433-7022, USA
[3] Ball Aerospace and Technologies Corporation, Fairborn, OH 45324, USA
{cmiller,jhamell}@sift.net,
{mark.draper,gloria.calhoun,tim.barry.ctr,
heath.ruff.ctr}@wpafb.af.mil

Abstract. A "delegation approach" to human interaction with automation should strive to achieve all of the flexibility that a human supervisor has in instructing, managing, redirecting and overriding well-trained human subordinates. But in the absence of human-like, natural-language understanding "androids", what would such interaction look like? This multi-year design and evaluation project explores such interaction concepts for pilot control of multiple remotely piloted systems. This paper details the underlying philosophy of delegation and presents many design innovations developed to date.

Keywords: flexible automation, adaptive/adaptable automation, playbook, delegation, UAVs, UASs, remotely piloted aircraft, multi-modal interaction.

1 Introduction

Humans do not "control" other humans in the sense that we control aircraft, automobiles, or power plants. Instead, we "delegate" actions, tasks, goals or responsibilities to "subordinates", and we "supervise" their performance. We have been advocating a "delegation approach" to human-automation interaction [1] that strives for all the flexibility that a human supervisor has in instructing, managing, and overriding well-trained human subordinates. But machines are not human, and at least most proposed automation for interaction with pilots does not have a human-like appearance, much less share natural language understanding or apply "common sense". So, what should delegation interactions look like with pilot-centric automation and how should it behave? A multi-year design project is exploring this question for pilot control of multiple semi-autonomous systems with the U.S. Air Force Research Laboratory (AFRL).

The application domain is control of multiple next-generation Air Force Remotely Piloted Aircraft (RPAs) in likely missions. This poses a challenging range of

D. Harris (Ed.): EPCE/HCII 2013, Part II, LNAI 8020, pp. 282–291, 2013.

interaction needs from precision or emergency reaction manual flight control to aggregate, multiple vehicle participation in precise integrated activities such as coordinated attack. Future needs require that a single operator be able to manage multiple, concurrent vehicle missions with no RPAs behaving in an unproductive or problematic fashion. The human pilots must remain in full charge of all capabilities of the vehicles they manage, even when they are focused on, perhaps even manually flying, one of them. In practice, this has meant that the operator must be able to intervene and demand any activity (including manually-controlled flight) at any time without impinging on the vehicle automation's ability to take control back when it is delegated. Pilots must be able to adapt automation's preplanned missions and behaviors to the uncertainties that will inevitably exist in military operations, or override them *effectively* to ensure that human intent is accomplished even in the "fog of war."

The overall AFRL program, called FLEX-IT (for Flexible Levels of Execution-Interface Technologies) has developed a series of designs and demonstrations to illustrate these goals. Below, we first describe the work domain, then the FLEX-IT control and delegation methods along with the delegation rationale for their selection and implementation. Next we present some specific design innovations, and summarize pilot feedback received to date. Finally, we discuss directions for future development.

2 The Multiple RPA Control Domain

Many current RPAs require at least 2-3 individuals during operations. A widespread goal is to at least reverse that ratio—enabling a single pilot to productively operate 3 or more RPAs [2]. Such missions would certainly include intelligence, surveillance and reconnaissance and might involve force protection, resupply, and/or attack and combat roles. Missions might be independent (searching for different, unrelated targets), or coordinated and loosely coupled (e.g., searching portions of a broad area) or cooperative and tightly coupled (e.g., coordinated attack). Present and likely future Air Force missions will require the ability to for pilots to exert precise control over flight paths, sensor operations, etc. Automation plays a key role in achieving the desired operator-to-vehicle ratio, but precise, expert human input will still be required.

This necessitates highly flexible human-automation interaction which spans, supports and integrates levels and styles of control ranging from full manual flight to rapid and comprehensive full-mission automation. It also mandates a wide variety of command input methods from the familiar throttle and stick, through keyboard and mouse to potentially speech, touch and sketch inputs—all to facilitate easy, natural yet efficient and flexible RPA control. System information output is similarly diverse, demanding use of the traditional map- and sensor-view screens along novel modalities and configurations to provide rapid but comprehensive awareness.

Our overall goal is to create designs that enable the flexible delegation a human supervisor has with intelligent subordinates in a well-understood operational domain. Thus, our designs take a "delegation approach" to multi-RPA control—preserving the functionality, if not always the modality, of instructions that a pilot commanding a multiple aircraft mission would employ if interacting with human subordinates.

3 FLEX-IT Control and Interaction Methods

In human organizations, supervisors can communicate with trained subordinates with a presumption of shared understanding of the common plans, goals, and methods in the work domain, of the common, culturally-determined priorities and limitations on appropriate behavior, etc. They can talk about activities in detailed natural language, using all modalities for nuanced intent, urgency, uncertainty, etc. that humans share. Moreover, they will likely share specialized vocabulary—a jargon or prescribed protocol to reduce errors and improve speed. These factors enable delegation with great flexibility. Supervisors can activate complex, multi-agent behaviors with a word, but they can also shape and refine those behaviors, create new ones, can task at various time points before and during behavior execution, and can even jump in and physically perform or illustrate behaviors for subordinates if desired. This is the flexibility we are striving to achieve in human-automation interaction.

3.1 Delegation within a Hierarchical Task Decomposition

We have argued [1, 3] that delegation can be viewed as taking place within a "space" of possible domain tasks and this space can be characterized using a hierarchical decomposition [as in 4] where alternate performance methods are represented as "or" branches, and sequential dependencies, looping, conditional branching, etc. are captured as a directed graph. Automation changes the nature of tasks to be performed [5,6], but these alternatives can be regarded as alternate branches. The selection of a branch to take is, in essence, a function allocation decision—which is to say, a delegation decision. It can be made either at design time or left to execution time. If left for execution, this enables adaptive function allocation, but leaves open the question of who gets to make the allocation decision. If automation makes it, then the system can be called adaptive; if the decision is left to a human supervisor, then the system is adaptable [7]. Adaptable systems, especially those that offer flexibility in how delegation actions are made and communicated, have demonstrated advantages in terms of workload, predictability and user acceptance [1,8].

Figure 1 shows a decomposition for a single task (monitoring a target). Resource allocation alternatives have been suppressed (assuming a single RPA equipped with some kind of sensor) for ease of presentation; including them would entail many alternate branches. The parent task (Monitor Target) is decomposed into sequentially ordered subtasks (Ingress, Approach, Inspect, etc.). Most of these can be skipped (as depicted by the dotted forward arrows), but the Inspect subtask is required for a valid instance of Monitor Target—though one can iterate through Inspect multiple times (note the dotted backward arrow). Each subtask is further decomposed into one or more "Fly Maneuver" subsubtasks and "Inspect" also requires a concurrent "Maintain Sensor Focus" subsubtask. A range of possible maneuvers is shown as the next de composition of the first "Fly Maneuver"--the branching lines convey that any maneuver is potentially viable. Each maneuver is decomposed into specific control actions.

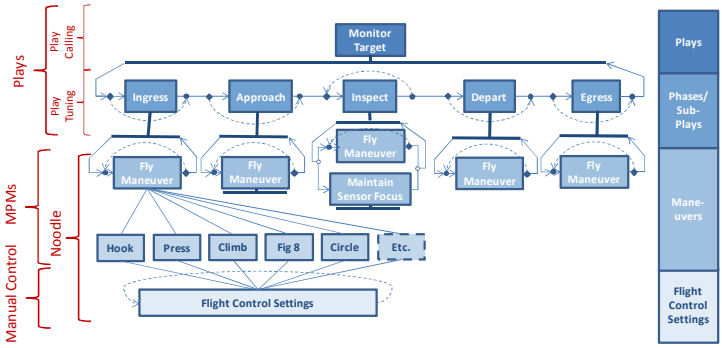

Fig. 1. Hierarchical decomposition of a single task along with illustrations of how FLEX-IT's control modes cover the range of delegation interactions within this hierarchy

A supervisor interacting with human subordinates could choose to do this entire task alone or could delegate it to a subordinate to plan and execute simply by invoking the name and providing a target. S/he could also choose to delegate parts (e.g., the Ingress portion) but retain control of the rest, and could constrain or stipulate how tasks are to be performed (e.g., specific routes to take or avoid, etc.). S/he could offer instructions either holistically (as a full mission plan) or piecemeal, and could jump in to revise or override as needed. S/he could also assemble lower-level tasks to achieve a new and different goal—something that would not be a Monitor Target mission, but would nevertheless use some of the same behaviors. Delegation need not occur at any fixed level within the hierarchy—an entire mission might be assigned to one subordinate, while only a maneuver flying task to another. Decisions are revisable: after initially delegating full mission execution, the supervisor might later decide to constrain a subtask—or even to fly portions personally. It is precisely this level of flexibility and ease of control input that we are striving to provide in FLEX-IT.

3.2 FLEX-IT Control Interaction Methods and Modalities

Figure 2 illustrates a current FLEX-IT workstation. The different control approaches described above are implemented and available in a variety of supporting input and output modalities including speech, touch, Hands On Throttle And Stick (HOTAS), and keyboard and mouse controls. This also parallels the flexibility of human-human delegation since humans communicate in multiple modalities—usually concurrently, but sometimes preferring one or more due to contextual factors.

FLEX-IT's control modalities and pilot interactions are described in more detail in [9] but will be summarized here. The left screen shows a map with RPAs, targets, routes, etc. with the bottom pane showing auxiliary details about plays and vehicles. The right screen is devoted to sensor feeds from each RPA. Besides HOTAS (manual control) and keyboard/mouse (most other inputs), FLEX-IT has also used a touchscreen to duplicate or extend some input capabilities. Headphones and a boom microphone enable auditory output and speech input.

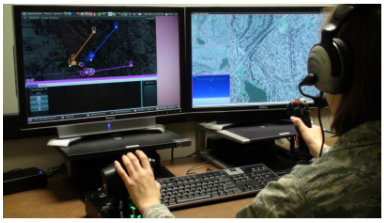

Fig. 2. The FLEX-IT workstation

Fig. 3. A chain of noodles representing a complex 4D flight path achievable by this RPA

FLEX-IT's Manual Control is available through vehicle selection and HOTAS activation that controls flight and also sensor movements (as well as "noodle" inputs as described below). This requires a sophisticated HOTAS controller with multiple auxiliary buttons. HOTAS control has evolved to be the most convenient and natural method for controlling the immediate flight of a vehicle. As a design principle, pilots should be able to manually control any vehicle at any time regardless of other previously delegated activities. This is akin to a supervisor saying "Stop what you're doing and do this instead."

The "Noodle" (Figure 3) is a unique control approach in FLEX-IT. It provides a convenient and natural way for pilots to tactically "pre-fly" the airplane from its current position. Using the current flight profile (position, altitude, airspeed, heading, etc.) it takes inputs of hypothetical flight control movements, via HOTAS, and then shows the results of sustaining those control actions for a variable time. This "noodle" (named for its appearance on the map) is pushed in front of the RPA icon and constantly updated until accepted for execution. Then, the RPA flies the noodle to its end. Subsequent flight control actions can be appended to initial ones to "chain noodles" as much as 15 minutes into the future.

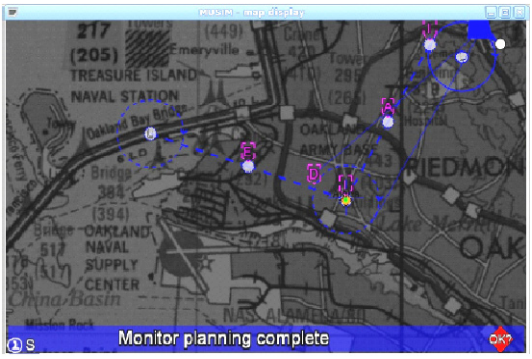

Fig. 4. A completed Monitor Target plan, ready for pilot review, editing and/or acceptance

Noodle control differs from a waypoint-based, autopilot-flown route in two important ways. First, it is generally much faster and more natural to command since it uses the same modality used for flight—the HOTAS—rather than the typical keypad and

cursor for inputs. Second, most existing autopilots treat waypoints as "goal points" and do not necessarily show the route used to achieve them. Especially for points very near a current position or placed very close together, the aircraft may not fly a straight line to achieve the waypoints. This is potentially dangerous if the actual route enters a restricted zone or crosses a national border. Since the noodle projects flight control inputs, it shows the vehicle's precise flight path barring disturbances.

Noodle control is similar to waypoint control in one important aspect: it allows delegating a flight path ahead of time. This can save the pilot time and/or enable offsetting when precise directions are given. If a pilot needs to steer one RPA through restricted zones and concurrently use HOTAS sensor controls to lock onto a ground target from another RPA, the noodle can dictate the flight path, allowing attention to be directed to the sensor task while the first RPA flies it.

Micro-Play Maneuvers (MPMs) are limited duration, fine-grained tasks. MPMs are primarily intended for quick verbal interactions, though we also provide methods for them to be "called" by mouse. Similar flight behaviors could be achieved by a small set of flight control inputs. Examples include: climb or descend to an altitude or at a specified rate, change to a specified heading or at a specified bank rate, circle and figure eight loiter patterns, hook (left or right), and press (that is, fly at) or separate from a specified target. The voice inputs to activate an MPM are defined from Air Force usage and radio practices—so while the pilot must remember specific verbal strings, these are natural, expected ways of discussing these behaviors. Feedback is provided on the map to show what the system has heard and will execute.

MPMs are suited for quick commands or corrections. Their verbal modality is ideal for use when controlling another vehicle—akin to a supervisor engaged in other work who can nevertheless tell a subordinate "Do this!" to maintain productivity over short periods of reduced attention. There are relatively minor consequences to invoking an incorrect MPM as long as it is noticed and overridden quickly—especially easy via the verbal input modality. Thus, MPMs are initiated immediately upon speech recognition with no further authorization from the pilot. They also persist for a limited time before either completing and entering into a default holding pattern or persist indefinitely with occasional queries as to whether this behavior should continue.

Plays are more complex means of commanding RPA(s) over longer time spans. FLEX-IT makes use of SIFT's Playbook® approach to adaptable automation, described elsewhere [1]. Plays are templates representing an agreed-upon goal and range of acceptable methods of achieving it. For example, in our Monitor Target play (Figure 1), the goal is achieving sensor images of a stationary target and acceptable methods involve getting a sensor-equipped RPA into position to provide that imagery and then depart. Other methods (such as satellite imagery, or even imagery provided by multiple vehicles) are not acceptable exemplars of *this* play. This is strictly a definitional convention, but it is precisely the kind of conventions human organizations evolve to facilitate rapid communication and shared understanding.

Play calling is flexible as well. Plays can be called (verbally or by touch/mouse) by invoking the play name along with required parameters—e.g., saying "Monitor Target Alpha" (a pre-designated target), or "Monitor Target here" with a pointing gesture. This represents maximal delegation since detailed planning (what route, inspect for-

mation, etc.) is left to automation using pre-stipulated defaults and preferences. The user can command more specific methods, though, by issuing verbal commands either at initial play call (e.g., "Monitor Target Alpha, altitude, 14000" to stipulate an altitude of 14000' throughout the play), or subsequently (e.g., "Change Inspect altitude, 15000" to revise only the Inspect phase).

Since plays generally involve more substantial planning that may be left to automation, the plan for a given play is reviewed by the pilot prior to acceptance and execution. A tentative plan is shown with a dashed line route plan in Figure 4. Sub-play phases are indicated by chevrons denoting their start points ("I" for Ingress, etc.) Sub-plays can be independently edited, as can the play as a whole, either verbally or via touch or mouse-controlled menus or by dragging and dropping waypoints. Once the play is accepted, FLEX-IT executes it with ongoing automated monitoring and revisions within the constraints of the play. The pilot can continue to revise it by manipulating the route or other details manually or through subsequent verbal "change" commands.

4 Design Challenges and Innovations

The governing heuristics we have used in our design may be stated as follows:

1. The pilot should interact with the FLEX-IT automation, as much as possible analogous to how s/he would interact with an intelligent subordinate.
2. Everything must be doable via multiple modalities
3. Pilot must be able to interact via any of the control approaches at any time
4. Pilot can seize control at any point with automation "backing off"
5. Detailed, longer duration plans must be preserved and intelligently resumable
6. Pilot must approve detailed and/or risky automation behaviors before enacting
7. Pilot must not be forced to approve predictable, limited, recoverable and/or non-risky behaviors before enacting
8. Predictability should generally be preferred over presumed workload saving

Achieving flexible interactions has led to several innovations. Our ultimate goal in FLEX-IT is both "top down" and "bottom up" play construction. We have largely achieved top down play calling and refinement—using a simple label or action to call a complex, high level play and having automation interactively develop a plan in context to achieve it. We have not yet fully achieved "bottom up" construction — using any and all of the control approaches, at any time and via multiple convenient modalities, to construct and revise innovative sequences actions as desired. We are working toward it, however, with innovations such as:

Play Suspension, Resumption and Staleness: Plays generally represent longer term intentions and will take longer to develop and refine. Allowing the pilot to seize the controls at any point could negate the effort spent in laying out an initial plan for automation. By contrast, a human supervisor could indicate that an interruption was temporary and or, alternatively, that a whole new plan was needed. We provide this

flexibility by not merely enabling pilots to interrupt a play to take manual control, but also to "pause" that play and make it available for later resumption.

When the pilot grabs the HOTAS or calls an MPM disrupting an ongoing play, the vehicle responds, but four changes appear on the display (as shown collectively in Figure 5e): (1) the leading "whisker" off the nose of the RPA icon shows a symbol indicating the vehicle's type of control. During play (or waypoint following) control, the whisker has a white circle (see Figure 5a), but under manual control, this icon is removed, (2) a pause icon is shown on the RPA symbol (see Figure 5b) indicating a paused play is available, (3) a moving white diamond appears on the route indicating where the RPA will resume unless the pilot dictates otherwise (Figure 5c), and (4) a "staleness" meter conveys information about how "out of date" the paused play is becoming (Figure 5d). The pilot can issue a verbal or menu-based mouse command to "Resume" at any point. The resumption point (cf. Figure 5c) is generally not where the vehicle left the route, but rather a point on the route as near as possible to the vehicle's current location without passing a subplay phase boundary. This provides a conservative, yet more intelligent resumption behavior. Mouse or touch gestures can also be used to say "Resume here" at any desired point along the planned path.

Fig. 5. Symbology for a paused play for an RPA off plan

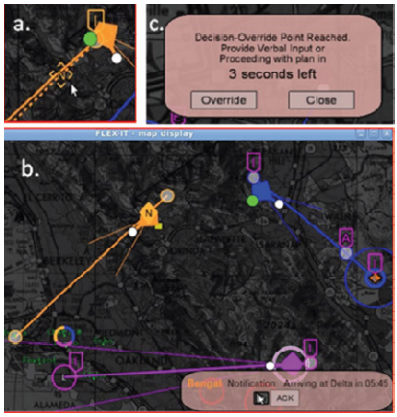

Fig. 6. Notification and Decision point behaviors showing (a) insertion of a notification point, (b) activation of that notification point and the resulting message, and (c) a similar message for a decision point

Our current staleness indicator simply sums the time since pause with that required to return to the computed resumption point. The expanded meter (shown in Figure 5d) shows these details, but this would normally be collapsed to show just the upper portion: a count-up timer that gradually shades from green to brown over 15 minutes. The pilot's true need is for knowledge of whether the vehicle's time away from plan threatens its ability to accomplish goals, requiring extraordinarily complex reasoning. This representative meter is a stand in for a more complex decision aid, but near term re-designs may go a step further to provide, for time-based plays, an indication as to

how much, if any "slack time" exists before the initial play can no longer be resumed and still meet initial deadlines.

Notification and Decision Points: Another form of flexibility that human teams have is the ability to issue contingent instructions and/or instructions to report and/or ask for guidance. We have provided these in FLEX-IT by use of Decision and Notification Points. These are simple points (currently tied to route locations, but also linkable to times or events) that, when reached, trigger an interface action such as a popup notification or window providing a selectable choice. Both decision and notification points come "pre-packaged" with some plays (e.g., a "Transit" play moves a vehicle from its current location to an endpoint and notifies the pilot when almost there) but pilot would have the ability to lay these down at any point. Figure 6a illustrates a notification point being placed along a route, while Figure 6b shows that point, having been reached, triggering a message ("voice notes" in the form of audio recorded messages can also be provided). Figure 6c shows a decision point message provided upon reaching a number of orbits in the Inspection phase of Monitor Target. This point took the form of an "Override or Proceed" decision; if the pilot does nothing, the system will proceed with the plan. A complex taxonomy of decision points and resulting actions is viable, but we have only illustrated the concept to date.

These are only a few examples of innovations that have been developed during FLEX-IT within the overall objective of attempting to replicate human delegation flexibility.

5 Feedback and Future Directions

FLEX-IT has been reviewed with 19 active duty RPA pilots to date. Extensive results are reported in [10] but will be summarized here. Ratings of the overall concept have generally been very high. In our most formal review to date, six RPA pilots saw the demonstration described above, and all rated the overall flexible interaction design as either a 4 or 5 (where 5 corresponded to "Great Aid" and 1 to "No Aid"; mean rating=4.7). All six pilots gave the high-level play calling implementation a rating of 5, and all raters said the system would be a "Great Aid" to reducing workload and increasing situation awareness in future, multi-RPA operations. All other design concepts (noodle, MPMs, tailoring plays, transition between control methods, decision points, etc.) had mean ratings ranging from 4.2 to 4.7 with no rating lower than 3. Specific feedback from this and other exercises has helped us tune many aspects of the system including the format of speech commands, the review and approval process for MPMs and plays and the addition of notification points.

Work on FLEX-IT is ongoing. We are currently transitioning from our initial implementation in a low-fidelity simulation to a higher fidelity control environment used for simulation, user testing, and for control of live RPAs (AFRL's Vigilant Spirit Control Station). Core current design challenges include:

- Maintaining user awareness and expectations about the vehicle designated to receive commands in the various modalities available (a non-trivial issue in a multi-actor, multi-modal control system striving for human-level naturalness),

- Migrating to a play- or activity-centered awareness of the current situation rather than (or in addition to) a vehicle-centered awareness,
- Maintaining robust speech interaction vocabulary as our suite of possible verbal interactions is beginning to increase exponentially, and
- Achieving completely flexible assembly of interaction modes at the user's initiation—including intermixing of control actions at all levels and time frames to construct unique activities according to the pilot's needs.

FLEX-IT's powerful yet integrated and natural suite of interaction modes have clearly indicated that there is promise to using human-like flexible and adaptable delegation as a guide to interactions with complex automation. Flexible human-centered delegation will continue to serve as a guide as we proceed.

Acknowledgments. This research was supported by AF/A2Q ISR Innovations Division and managed by AFRL/RHCI at Wright-Patterson Air Force Base. Support from Ball Aerospace & Technologies Corporation and Smart Information Flow Technologies personnel was provided under Air Force Contract FA8650-08-D-6801.

References

1. Miller, C.A., Parasuraman, R.: Designing for Flexible Interaction Between Humans and Automation: Delegation Interfaces for Supervisory Control. Human Factors 40(1), 57–75 (2007)
2. Eggers, J.W., Draper, M.H.: Multi-UAV Control for Tactical Reconnaissance and Close Air Support Missions: Operator Perspectives and Design Challenges, In: Proc. NATO RTO Human Factors and Medicine Symp. HFM-135. NATO TRO, Neuilly-sur-Siene, CEDEX, Biarritz, France (2006)
3. Miller, C., Parasuraman, R.: Beyond levels of automation: An architecture for more flexible human-automation collaboration. In: Proc. 47th Human Factors and Ergonomics Society, pp. 182–186. HFES, Santa Monica (2003)
4. Diaper, D., Stanton, N. (eds.): The handbook of task analysis for human-computer interaction. Erlbaum, Mahweh (2004)
5. Billings, C.: Aviation automation. Erlbaum, Mahweh (1997)
6. Woods, D.: Decomposing automation: Apparent simplicity, real complexity. In: Parasuraman, R., Mouloua, M. (eds.) Automation and Human Performance: Theory and Applications, pp. 1–16. Erlbaum, Mahweh (1996)
7. Opperman, R.: Adaptive User Support. Erlbaum, Hillsdale (1994)
8. Layton, C., Smith, P., McCoy, C.: Design of a Cooperative Problem-solving System for Enroute Flight Planning. Human Factors 36(1), 94–119 (1994)
9. Miller, C., Hamell, J., Barry, T., Ruff, H., Draper, M., Calhoun, G.: Adaptable operator-automation interface for future unmanned aerial systems control: Development of a highly flexible delegation concept demonstration. In: Proc. Infotech@Aerospace Conf., AIAA, Reston, VA (2012)
10. Calhoun, G.L., Draper, M., Ruff, H., Miller, C., Hamell, J.: Future unmanned aerial systems control: Feedback on highly flexible operator-automation delegation interface concept. In: Proc. Infotech@Aerospace Conf., AIAA, Reston, VA (2012)