

# A HUMAN FACTORS TESTBED FOR COMMAND AND CONTROL OF UNMANNED AIR VEHICLES

*Kam S. Tso, Gregory K. Tharp, Ann T. Tai, IA Tech, Inc., Los Angeles, CA  
Mark H. Draper, Gloria L. Calhoun, Heath A. Ruff, AFRL/HECP, WPAFB, OH*

## 1 Introduction

As technological advances allow unmanned air vehicles (UAVs) to operate more autonomously, human performance in supervisory control of UAVs becomes a critical issue. In particular, experience shows that human performance in supervisory control is a function of extent of automation, levels of system fidelity, and rates of information update. In order to explore this issue, we developed a testbed that supports the planning and conducting of UAV human factors experiments. The testbed is built upon the Multi-Modal Immersive Intelligent Interface for Remote Operation (MIIRO) which was designed to support UAV control [1].

The testbed implements a client/server architecture in which UAV operations are simulated on a server that maintains the states of the UAVs and their payloads. The operator control interface is implemented as a client Java applet which can be run via a web browser or as a standalone application. This configuration supports the collaboration of a crew of operators in control of multiple UAVs. The clients communicate with the server via message passing with messages that are time stamped and placed in a message queue so that their delivery can be delayed arbitrarily to simulate various link delays.

An assortment of facilities have been developed in the testbed client to support both the planning and execution of human factors experiments. An experiment designed by the Air Force Research Laboratory (AFRL) Crew System Interface Division concerns a multiple UAV mission to locate and identify ground targets. This experiment was designed to investigate several human factors issues raised in earlier work [2], including operator interaction with automation levels and decision-aid fidelity. The testbed client enables the experiment planner to design routes for the UAVs by designating waypoints in the Tactical Situation Display (TSD). The client also automates the generation of images of ground targets based on

the specified parameters, such as the number of targets in each image, type of targets, and fidelity level of the automatic target recognition (ATR). Another planning facility supported by the client is event generation. It allows the planner to add events such as pop-up threats, ad-hoc targets, unidentified aircraft, and mission mode indicators during the experiments. During an experimental trial, as the UAV flies over a target area, the captured images are entered in the Image Queue. The image at the top of the queue is displayed with boxes surrounding the ATR identified targets. The experiment subject must then verify the ATR and redesignate the boxes (if necessary) optionally within a specified time limit. Events also occur at random times and the subject is required to respond to them. The subject's responses and their timings are recorded by the client for later analysis.

## 2 Operational Architecture

MIIRO is designed to perform or simulate the operations of a number of autonomous UAVs. A detailed description of its architecture and operational capabilities can be found elsewhere [1,3], but a brief summary is provided below highlighting features of importance for use as a human factors testbed.

### 2.1 Client/Server Architecture

MIIRO is based upon a client/server architecture with the server node either being at the location of the communications links with actual UAVs or the location of the simulation UAVs. The client provides an operational interface for command/control and monitoring of UAV activities. The Web-based architecture allows the Java client software running on the MIIRO stations to be deployed over the network and executed via the ubiquitous Web browser [4]. The distributed architecture provides flexible and secure communication between the MIIRO stations and their server for information collection and

distribution. It also allows users separated by large physical geographical distances (e.g., Commanders at the Pentagon, Field Commanders at the mission center, and crew members in the battlefield) to participate interactively in situation assessment and collaborative planning.

The MIIRO software architecture is highly modularized so that it can be easily reconfigured to meet new requirements and expanded for future applications. The intent of the design is to not only support real vs simulated operations, but also to support various UAVs, various UAV configurations (e.g. instrumentation/weaponry), and various missions (e.g. reconnaissance/attack).

## **2.2 Multi-Modal Interface**

An important factor in MIIRO's utility both in envisioned operations and as a human factors testbed is its multi-modal interfaces. The system allows operators to control the UAVs not only in either autonomous or manual modes, but also in a shared control scheme [5]. Further, it supports various input methods to supplement the graphical user interface including joystick, motion tracker, and voice. Finally, it offers several visualization modes to allow the operator to maintain situational/mission awareness including a mission plan view, a 3D view (that can also simulate video) and various instrumentation interfaces.

### **2.2.1 Control**

A standard scenario for the control of a UAV mission includes the creation of a nominal preplanned mission that is executed autonomously by the UAV. The operator then monitors the progress of the mission and can either update the plan as contingencies arise or can take shared or full manual control of the UAV if necessary.

For autonomous mission control the operator creates a sequence of navigation waypoints through which the UAVs will fly. The placement of these waypoints is driven by a set of target locations, which represent the mission goals, and threats, which represent one form of mission constraint. The operator must also provide commands at the relevant waypoints that produce the desired method of prosecution of the target locations. This sequence of waypoints and commands is referred to as the

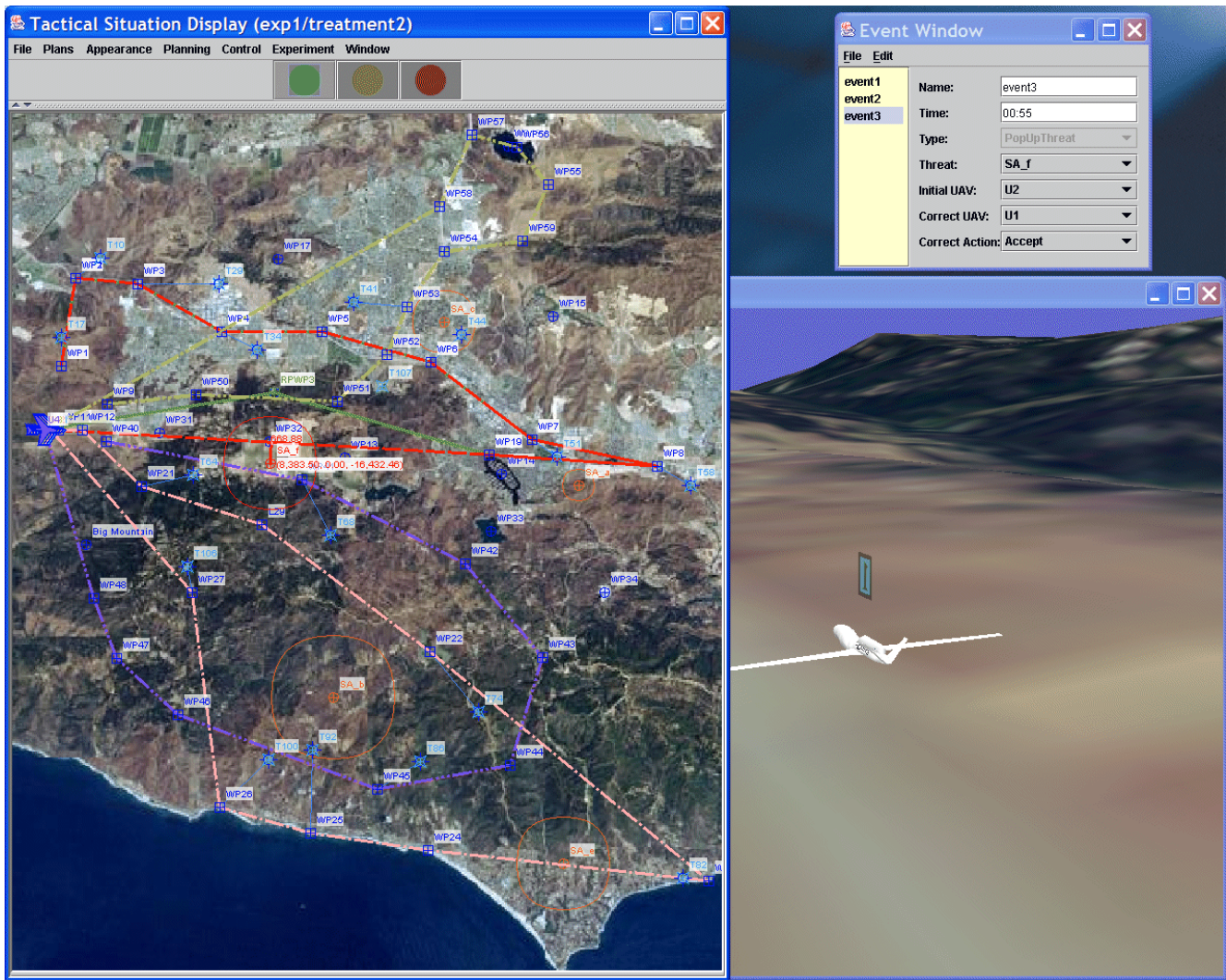
mission plan. The system also provides a method for the addition of goals and constraints during mission execution. We call these ad-hoc targets and pop-up threats. If possible, the operator would deal with these items by updating the mission plan. However, sometimes it is necessary for the operator to take manual control of the UAV. This can be done with a joystick, motion tracker, or GUI panel. Additionally the system supports other mission constraints, such as time and fuel, by providing a resource estimator for those parameters. If the mission contains time and/or fuel constraints, the operator can view the usage of these resources either on a segment by segment basis or cumulatively for the entire plan. We have also implemented a shared control scheme where the operator can take control of some aspect of the navigation (e.g. altitude, speed) and leave the remaining control to the autonomous system.

### **2.2.2 Situation Awareness**

Situation awareness is imperative for the operator to effectively control the UAVs and payloads. MIIRO has been designed to maximize the situation awareness by providing the operator with multiple views of the engaging area simultaneously, and a virtual environment to induce a sense of presence.

The primary graphical interface for the system is the Tactical Situation Display, which is shown in Figure 1. The TSD provides a plan view of the mission environment with an overlay of the waypoints (squares), flight segments (connecting lines), targets (stars), and threats (circles) in that environment. It also contains icons for each of the UAVs showing their positions and status. The TSD is used to create and edit mission plans and is also used during operations to monitor the progress of missions. Lastly, the TSD is one means of alerting the operator to contingencies.

MIIRO provides for a 3D model of the UAVs and their environment, also shown in Figure 1. This virtual model can be used in a number of ways. First, it can be used to provide an external overview of the environment to provide context to the UAV operations. Second, it can be used to provide a pilot's eye view from the UAV. MIIRO uses the virtual window concept to provide the virtual environment for inducing a sense of presence in the



**Figure 1. Experiment Planning**

engaging area [6]. It can also be used as a tool for creating generated still or motion imagery as from cameras mounted on the UAV thus providing instrument simulation.

MIIRO also provides interfaces for UAV instrumentation. For example, when the UAV camera captures an image of a target area, the target icon in the TSD changes from a star to a rectangle. In addition a number displayed which indicates the number of target objects the ATR system has detected. Imagery can be returned automatically, or the operator can use information such as the ATR results to decide when to request images and in what order. The later, a low-bandwidth mode, for situations where transmitting imagery may overload the telemetry.

Imagery sent back from the UAV for the operator to evaluate is placed in another interface called the Image Queue. The Image Queue consists of a panel which displays the current image, and a list of all the imagery currently in the queue. The list contains contextual information about the images as well as interfaces to allow the operator to handle the images. One important feature of the interface is the ability to designate target objects within the imagery. When displayed, the ATR results are shown as boxes around the objects. The operator must then verify and, if necessary, correct the target object designations.

Each instrument can also have its own custom interface. These interfaces can be used to monitor the status of the instrument itself, set global

instrument parameters, or for the manual operation of the instrument independent of the mission plan. We have implemented a camera control interface for these purposes.

### **2.2.3 Levels of Automation**

An important feature built into MIIRO is the ability to change the level of automation of operations. (This is distinct from the control modes of the UAV which relate to the autonomy of the UAV). Automation of operations refers to automated assistance for the tasks assigned to the operator including automation for mission replanning and imagery evaluation. Further, the fidelity of the information presented to the operator by this automated system can be adjusted. The system can present information or suggestions to the operator that are in error with respect to the underlying data. For example we can control the fidelity of a simulated automatic target recognition system to control the number and types of errors that are presented to the operator. This feature is discussed in greater detail in section 5.2.

## **3 Multi-UAV Target Location and Verification Experiment**

The Air Force is currently performing an initial study to explore the human factors of multi-UAV operations using MIIRO as a testbed [7]. It is a goal to empirically determine how levels of automation, levels of system fidelity, and levels of information update rate affect human performance in supervisory control of multiple, hypothesized, future UAVs. This control activity involves prosecuting existing and ad-hoc ground-based targets, avoiding existing and pop-up threats, identifying unknown aircraft, and monitoring a mission mode indicator. The hypothesis is that decreased system fidelity and update rate adversely affect operator performance. Increasing automation will encourage complacency and lack of system event understanding, especially during lower fidelity and update rate. Increasing the number of vehicles to supervise will multiply the information content by a factor, increasing operator requirements (e.g., increase of rate of images entering the image queue) will lead to adverse performance.

## **4 Experiment Planning**

### **4.1 Experiment Manager**

In order to set and organize the many parameters that are necessary to specify the conditions of a particular trial and set of trials for an experiment we developed an experiment manager tool. This tool is used by the experimenter during the planning process to set the values of the parameters for a particular trial, to control the experiment trials and to view raw preliminary results to ensure that a trial was run and recorded successfully.

#### **4.1.1 Experiment**

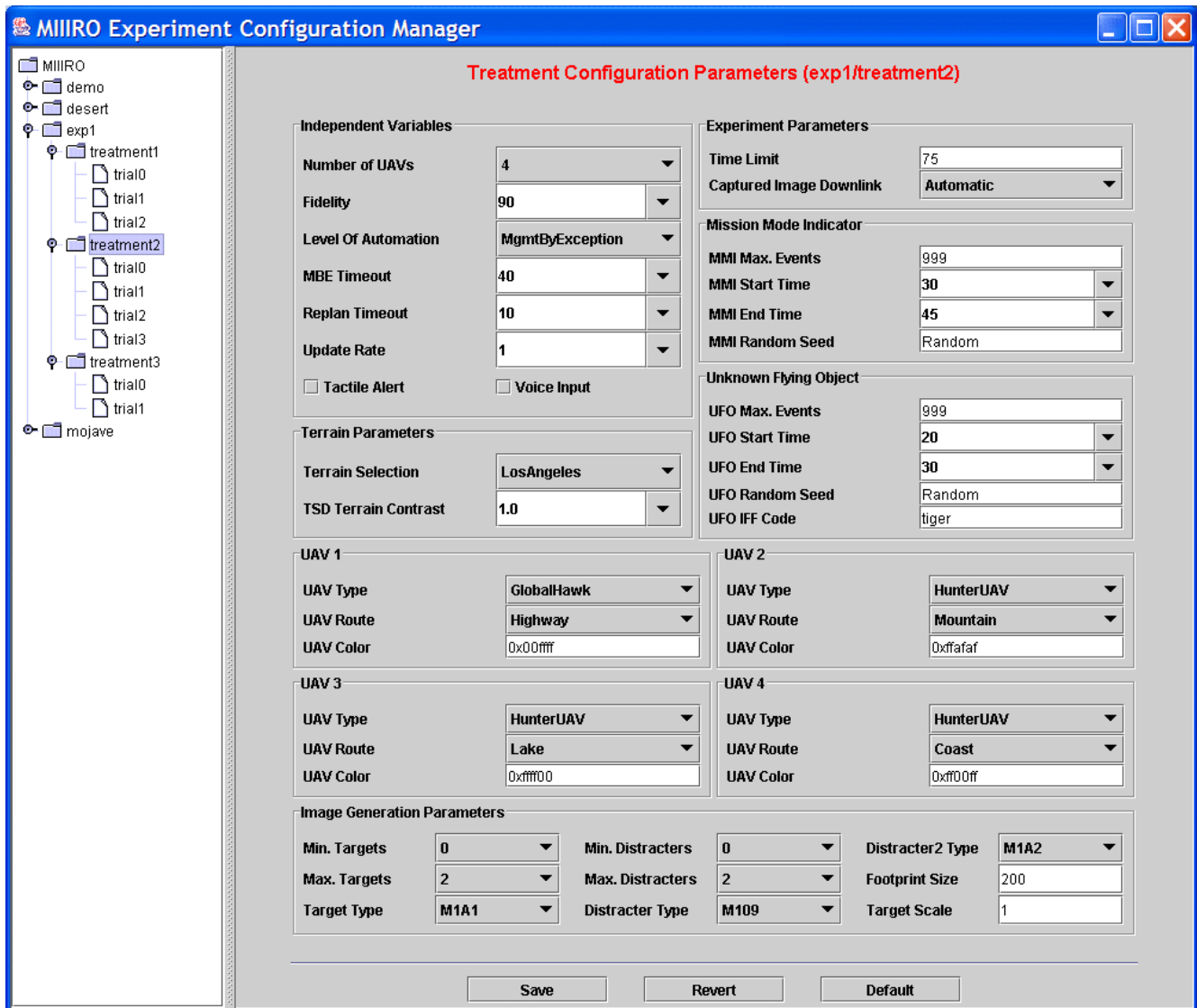
There are a number of panels in the Experiment Manager that control or display various aspects of an experiment (see Figure 2). These panels are accessed from a hierarchical tree listing of the experiments, treatments, and trials. At the top level of this tree is the system itself. By selecting the root of the tree a panel is displayed that allows the experimenter to select the mode to run in, and other basic parameters that affect the running of the system. The most important of these is the selection of the experiment and treatment (or condition of the experiment) to run. The next level of the tree defines a set of experiments. There are no parameters that are controlled at the experiment level; they are collections of treatments which hold the varying conditions that are presented to subjects at a particular trial.

#### **4.1.2 Treatment**

The next level of the tree is the treatment level. This is where all the parameters of the system are specified that create the set of conditions for an individual trial. This includes such items as: the number of UAVs, fidelity of automation, level of automation, standard timeouts for the various subsystems, and the system state update rate.

### **4.2 Route Planning**

In order to use MIIRO to run an experiment, the experiment planner must provide a set of mission plans that will be presented to the subjects during the course of the experiment. The TSD is used to specify the flight path of the UAV for a mission. In MIIRO, mission plans are represented by waypoints and targets.



**Figure 2. Treatment Parameters**

The waypoints define the flight path that the UAV will fly. The TSD allows the operator to visualize the mission area and enter waypoints at the locations directly on the terrain image. Each pixel in the terrain image corresponds to the actual location in the mission area. This relieves the operator from knowing the detailed coordinates and doing calculations. The planner will also have to specify a number of targets and threats for each mission plan scenario. The targets define the locations and contain task commands to be executed at those locations. Some target task commands include taking closeup images, launching weapons, and operating payloads.

Threats are represented by circles indicating areas of the terrain where the UAVs would be subject to enemy fire. Finally, there is a planning window which shows the mission plan as a list of navigation commands. This interface can be used to add commands to the mission plan for the UAV to prosecute targets after reaching particular waypoints.

### 4.3 Event Planning

Ad-hoc targets and pop-up threats are not visible at the beginning of an experiment but appear during the experiment. When these items appear, it is called a replan event because the mission plan of

a UAV may need to be replanned to prosecute the ad-hoc target or avoid the pop-up threat.

Ad-hoc targets and pop-up threats can be added to an existing plan using the popup menus in the TSD. For these items the planner must specify when they will appear and which UAV's plan will be affected.

#### 4.3.1 Timing Considerations

When creating an ad-hoc target or pop-up threat there are some considerations that need to be taken into account when specifying the timing of the events. The most important of these is that the event must occur "in front of" the affected UAV; in other words, the event must occur on a segment of the plan over which the UAV has not yet flown. The Resource Estimator calculates the approximate amount of time it will take a UAV to fly particular flight segments. By viewing these estimates in cumulative mode the event planner can make a prediction as to where the UAV will be when an event is scheduled to occur. In addition the planner must account for an event handling time because the plan is not changed until the subject accepts the proposed replan. The event handling time is constrained by a settable parameter that limits the amount of time a subject has to respond to an event. Finally, the current system has a limitation that there can be only one active event at a time. Events should be scheduled such that all previous events have had sufficient time to be resolved.

#### 4.3.2 Auto-Replanner

When an ad-hoc target or pop-up threat is added, the auto-replanner is run on the UAV plan specified in the event's parameters. The auto-replanner creates a modification of the plan that involves either inserting a single new waypoint or moving a single existing waypoint (single waypoint solution) to account for the new item. During planning when a particular event is selected or during a trial when the event occurs, the replan result is displayed in the TSD as a red overlay of the existing plan. The display also shows a line that extends from the new or modified waypoint to the closest point on the original plan to show the amount of adjustment that the replan creates. The algorithm does not guarantee an acceptable solution especially in the case of threat avoidance, but it works well most of the time and usually an acceptable solution can be found by adjusting the

existing plan waypoints to avoid extreme geometric configurations. The auto-replanner also provides for a "bad" replan based upon the idea that if the item is too far away from the existing plan it should be ignored. So that subjects can evaluate this situation but always see some kind of replan, the replanner will create a "bad" replan in these cases.

#### 4.3.3 Replan Algorithm

When replanning for an ad-hoc target the algorithm adds a waypoint near the target. The distance between the target and waypoint is controlled by a parameter called the TargetReplanDistance ( $D_{tar}$ ). For targets that are farther away from the existing plan than  $D_{tar}$  a waypoint is added to the plan  $D_{tar}$  away from the new target. In order that there always be some deflection of the original plan, for targets that are closer than the  $D_{tar}$  a waypoint is added to the plan that is 80 percent of the distance between the plan and the target. If the target is farther away than another parameter called the BadReplanDistance, the algorithm proceeds as explained above, however, a flag is set that indicates that the target should be ignored.

When replanning for a pop-up threat the algorithm adds a margin called the ThreatReplanDistance to the threat radius. If the closest waypoint is within this margin, it is moved directly away from the threat until it is exactly on the margin boundary. Because the replanner only supports a single waypoint solution, it is conceivable that more than one waypoint is within the threatened area. In this case a solution must be found by modifying the existing waypoint positioning. If no waypoints are within the margin but a plan segment is, then a new waypoint is inserted in that segment and placed so that it is just on the margin boundary.

If no part of the plan intersects the threat margin, a "bad" replan is generated. In this case a waypoint is added at the margin boundary but on the opposite side of the threat from the plan. This usually produces plan segments that actually fly through the threatened area, but may just produce an unnecessary deviation of the flight path. In either case the flag is set that indicates that the threat should be ignored.

## **4.4 Image Generation**

In the course of an experiment there may be many target locations that will be prosecuted to search for target objects. In order to facilitate the creation of the target imagery, we have implemented an automated image generation capability. Once the mission plans for a set of UAVs have been finalized, including the scheduling of ad-hoc targets, the planner can generate an image for each target location along the mission plan. The image generation uses the virtual models available in the 3D window to produce a virtual image of the target location. It does this by moving the UAV model to the waypoint associated with the target location and then pointing the UAV camera model at that location. The field of view of the simulated camera is adjusted to maintain an approximately constant image footprint and then target and distracter objects are placed on the terrain within the image. Once all of the objects have been placed on the terrain, the virtual rendering of the environment is saved as an image. In addition, data are saved that record the actual positions of the target and distracter objects in that image.

### **4.4.1 Target Objects**

Many different items can be used as target and/or distracter objects. As with the terrain and UAV models, any object represented by a VRML or OpenFLY format file can be used as a target object. A typical usage might be something like a tank as a target and an armored personnel carrier or artillery piece as a distracter, but we have used different types of tanks or aircraft also. Once a model is specified as a target to the system, it is available as a selection for use in the automated image generation.

### **4.4.2 Image Generation Interface**

The parameters for automated image generation are set in a section of the manager panel for a particular treatment. This means that the parameters are used for all the images in a particular experimental condition. The first parameters are the minimum and maximum number of target objects. The system will position a uniformly random number of target objects between these minimum and maximum values inclusive in each image. The next parameter is the type of object that represents the actual targets for this condition which is chosen from a list of objects that have been

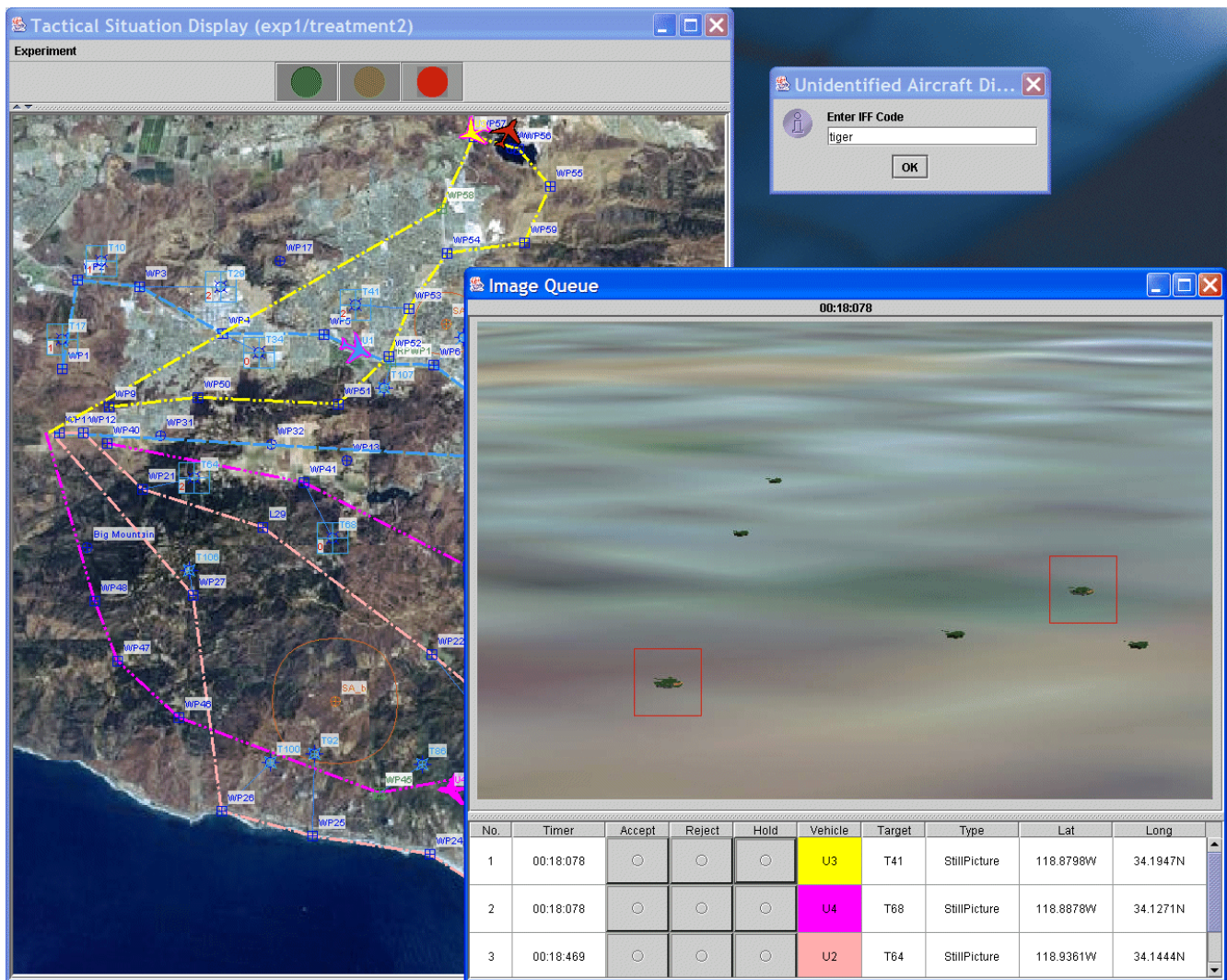
registered with the system as targets. Next, the minimum and maximum number of distracters along with one or two types of objects that will be the distracters are specified. If two distracter types are specified the system will randomly choose the type assignment of each distracter as it is placed in the image. Then a footprint size is specified which is used to adjust the field of view of the simulated camera so that the width of the terrain area captured by the image is approximately this value. Finally, a target scale can be specified that is used to make the target/distracter objects appear larger/smaller than their underlying models. This is mainly used to feature match the target models with the underlying terrain model which may in fact be very low resolution at the scale of the target objects.

## **5 Experiment Trials**

Once the planning has been done for all the conditions of an experiment, the system is ready to run subjects through a set of trials. Trials are initiated by the subject clicking an icon indicating that they are ready. At this point the interface windows are displayed, and after a settable delay the UAVs begin to fly along their mission plans, as shown in Figure 3. During actual trials a dual-monitor system is used so that the windows will be larger and will not be overlapped. The subjects monitor the target prosecution, needed navigation modifications, and other system status information. The speed and accuracy of the subjects responses are recorded during the trial. Trials end either when some preplanned trial duration has been reached, or when the subject indicates that they believe that all necessary actions have been accomplished.

### **5.1 Image Queue**

The primary operational task that subjects are asked to supervise is the acquisition of target imagery and the designation of target objects within that imagery. The interface used to accomplish this task is the Image Queue. Target imagery are placed in the Image Queue as the UAVs prosecute targets in the order in which they occur. The image queue displays the target image that is at the top of queue with an overlay of the currently designated target objects. In addition there is a list of all the images that are currently in the queue. The list contains information such as which UAV captured the image, the target location, and how long the image



**Figure 3. Experiment Trial**

has been in the queue. Subjects use this interface to monitor the target image acquisition and target object designation process.

### 5.1.1 Image Handling

When a target image comes to the top of the queue, it is displayed along with its current target object designations that have been identified by automatic target recognition. Target object designations appear as red boxes overlaid on the image around items within the imagery. The subject can make new target object designations by clicking on an item in the image or clear any existing designation by clicking within an existing box. The subject is supposed to make sure that only actual target items are designated and nothing else.

When the subject believes that all the target items and only the target items have been designated, he/she can accept the imagery and the image will be removed from the queue. An image should be rejected if the subject believes that there are no target items in the imagery. If the subject wants to postpone final determination of the target designations, the image can be placed on hold, moving it to the end of the current queue of images.

If the images in the queue are subject to a time limit, then if that time expires before the subject acts, the images are automatically accepted or rejected based upon their current target object designations. If targets have been designated the image is automatically accepted, but if there are no designations it is automatically rejected. When an

image is held the time limit is reset; this means that an image always has as much or more time than images above it in the queue.

### **5.1.2 Timing**

A set of timing information is recorded as images are handled in the Image Queue for assessing the performance of the trial subjects. The first time recorded is the time of capture and entry into the Image Queue. Images are entered into the queue in the order they are captured regardless of which UAV captures the image. The next time that is recorded is the time that the image is first displayed in the queue. If there are no images in the queue at the time of capture this will be the same as the time of capture/entry. Often there are already images in the queue when an image is captured and in this case the first display time is recorded when the image makes its way to the top of the queue. Every time an image is held, where it is placed at the end of the queue from the top, the hold time is recorded. Images are removed from the queue when they are accepted or rejected and this time is the final time recorded.

Another aspect of image queue timing is that, depending on the level of automation support, images may only stay in the queue for a limited time without subject action. In this case the time shown in the queue information counts down from this time limit instead of showing the actual amount of time that the image has been in the queue. When the time limit is reached the image is automatically accepted or rejected and that time is recorded as if the subject had performed the action.

## **5.2 Human Factors Issues**

While the primary task of the operator is to monitor and evaluate the target imagery, there are other more secondary tasks which the subjects must also supervise. These tasks may interfere with the responsiveness and accuracy of the primary task. In addition, there may be automation techniques that can assist the subject to accomplish various tasks thereby reducing the task load burden. The testbed supports some of this automation and has a means for controlling the level of automation assistance. Finally, because the timeliness of information may affect the subject performance, the testbed supports a means to adjust parameters to influence this aspect of an experiment.

### **5.2.1 Popups**

The first secondary task that subjects must monitor and respond to is the idea of an unidentified aircraft (UA) entering the workspace of the UAVs. The experimental planner specifies a window in time relative to the starting point of the experiment. A UA appears in the workspace sometime during this window. This is displayed to the subject with a red aircraft icon appearing somewhere in the TSD. As soon as the subject notices the UA, they are asked to click on its icon. At this point a dialog box appears into which the subject must type an IFF (Identify Friend/Foe) code that they have been given before the trial. When the correct code is entered, the UA icon will be cleared. The number of UA events that occur is specified by the experimenter and subsequent events occur within a window relative to the completion of the previous UA event. Times are recorded when the UA appears, when the subject clicks on the icon and when the correct IFF code is entered. The number of incorrect IFF code entry attempts is also recorded.

A second secondary task is the monitoring of the Mission Mode Indicator (MMI). The MMI is a series of lights (green, yellow, red) displayed at the top of the TSD. Nominally the mission mode has the green light displayed. Within a window defined similarly to the UA, the mission mode will change to yellow and then later, after a settable time delay, to red. At the point where the subject realizes that the mission mode is no longer green, they should click on any of the lights. When the lights are clicked, a dialog box is displayed that contains a random number which the subject must type into a field provided. As soon as the number is entered correctly the dialog is removed and the mission mode returns to green. As with the UA, the experimenter can specify the number of MMI events to occur and the timing window is reset relative to the previous MMI event. The time that the MMI changes from green, the subject clicks the lights and the subject enters the random number correctly are recorded along with the number of incorrect entry attempts.

### **5.2.2 Management by Exception or by Consent**

The testbed currently supports two levels of automation support called management by consent

(MBC) and management by exception (MBE). Management by consent is ostensibly a manual mode of operation which requires subject action to handle the experimental events. However, it is unworkable to leave some events pending indefinitely awaiting subject input, so in this condition the most rudimentary form of automation is implemented where the handling of events can time out. In these cases the automated response is the one most closely resembling no subject input.

Management by exception on the other hand implements a kind of intelligent automation. It imposes time limits on those aspects of the system that could be left to the subject indefinitely, and it attempts to correctly handle events given the information that is available. The most prominent example of this is the image queue. Under MBE the images in the queue have a time limit for handling, and when that time expires they are either accepted or rejected based upon the existing target designations. With regards to replan events (ad-hoc target, pop-up threats) the system will automatically accept the suggestion of the replanner after the time period expires.

### **5.2.3 Update Rate/Communications Latency**

One final aspect that can be explored to examine its affects on subject performance is the performance of the underlying systems themselves. To do this, the testbed supports the ability to change the update rate at which new information is displayed to the subject. In a real system this would be determined by the underlying communications bandwidth. Here we can set the rate at which updates are sent to the display to reflect the progress of the UAVs.

## **5.3 Trial Results**

Preliminary trial results can be viewed in the Experiment Manager using the lowest level of its tree listing, as shown in Figure 4. It shows the raw results that have been recorded for each trial of a particular treatment. The experimenter can use this information to verify that a particular trial was run and recorded correctly.

### **5.3.1 Timelines**

The main section of the trial result panel shows the timelines of the experimental trial. At the top it

shows when images were entered into the image queue with various information such as how long it took for them to be handled and whether they were handled correctly. Below that are the timelines of the replan events which also show when they occurred and how they were handled. Below that is the timelines of the MMI and UA events and handling if they are active in the trial. Finally, at the bottom of the panel is a trace for each UAV showing when it reached each of its mission plan waypoints. This display is particularly useful to examine when events occurred in relation to each other and if they may have had some interaction.

### **5.3.2 Statistics**

The trial results panel can optionally display the statistics of the performance in the tasks the subjects were asked to perform. In addition to raw counts of things like the number of images taken and various event counts, this panel shows things like the number of images correctly accepted/rejected, number of targets missed, and the time it took for the subject to respond to various events. It can be used by the experimenter to make a first approximation as to how well the subject performed during a trial.

## **6 Conclusions**

We have extended the MIIRO system as a testbed for human factors experiments. We did this with the addition of capabilities to plan and manage data sets for use in experiment trials as well as the capabilities for recording operator performance. We also added tasks (MMI/UA) for the testbed that have analogies in real operational scenarios. With the completion of the initial experiment being performed by the Air Force we will have demonstrated the efficacy of embedding the testbed capabilities into the MIIRO system.

A major focus of future work on the testbed will be to extend the support for levels of automation. Beyond just having the ability to support multiple levels of automation, we will be making the support independent so that each subsystem can operate at its own level of automation. Additionally, we also plan to support the modification of the automation level during operations.

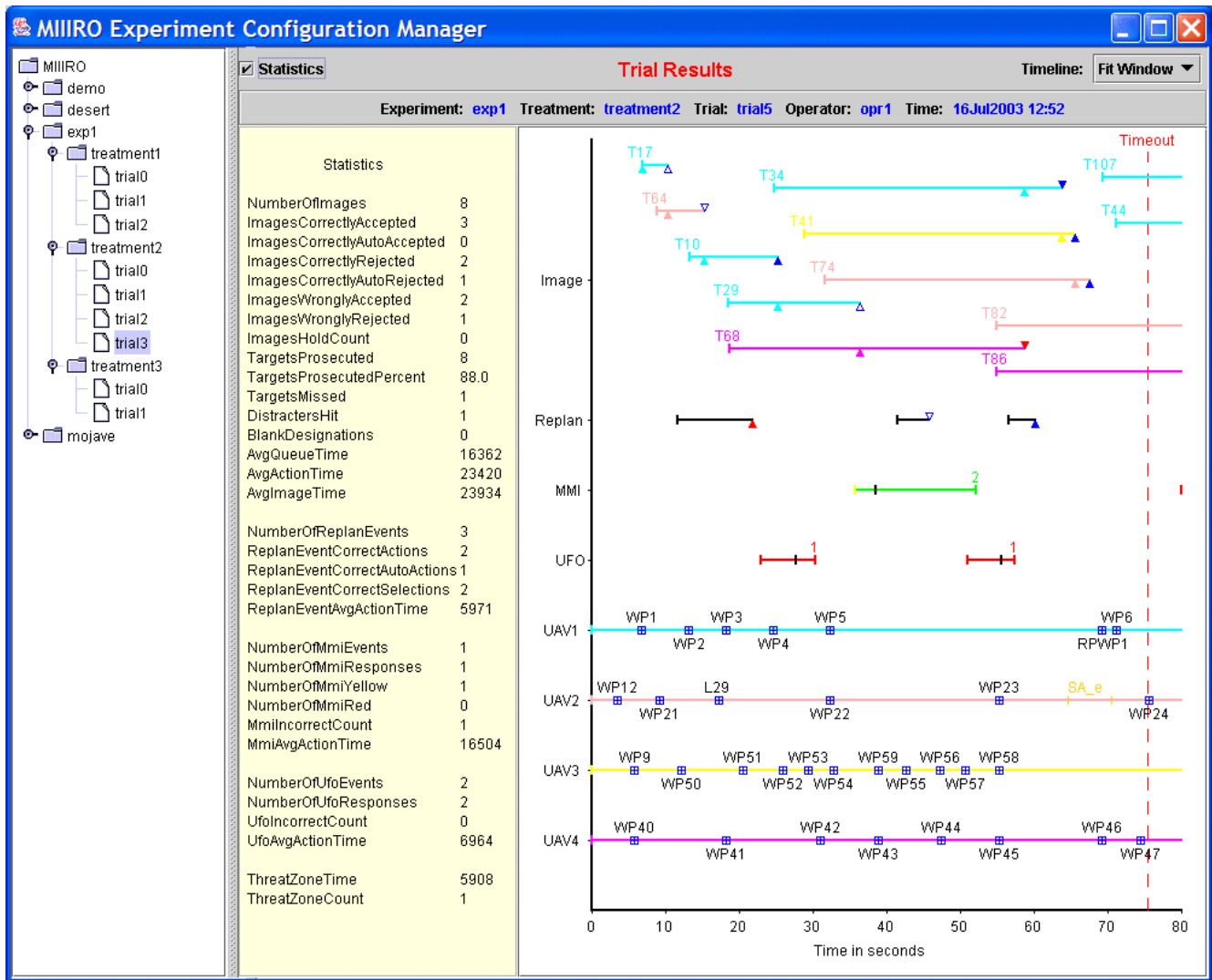


Figure 4. Trial Timelines and Statistics

## Acknowledgments

The development of MIIRO was supported by the Air Force Research Laboratory Crew System Interface Division (AFRL/HEC) under a Small Business Innovation Research (SBIR) contract F41624-98-C-6006. The authors wish to express our appreciation for the interest and support of Dr. Robert Eggleston, Project Manager of the contract. The testbed extension was supported by an SBIR Phase III effort sponsored by AFRL/HEC under a Sytronics subcontract 10678-9082-0202.

## References

- [1] K. S. Tso, G. K. Tharp, W. Zhang, and A. T. Tai, "A multi-agent operator interface for unmanned aerial vehicles," in *Proceedings of the 18th Digital Avionics Systems Conference*, (St. Louis, MO), pp. 6.A.4.1–6.A.4.8, Oct. 1999.
- [2] H. A. Ruff, S. Narayanan, and M. H. Draper, "Human interaction with levels of automation and decision-aid fidelity in the supervisory control of multiple simulated unmanned air vehicles," Presence: Teleoperators & Virtual Environments, Special Issue on Virtual Environments and Mobile

Robots: Control, Simulation, and Robot Pilot Training, vol. 11, pp. 335–351, Aug. 2002.

[3] K. S. Tso, G. K. Tharp, A. T. Tai, and W. W. Zhang, “Multi-modal immersive intelligent interface for remote operation,” SBIR Phase II Final Report AFRL-HE-WP-TR-2002-0069, IA Tech, Inc., Los Angeles, CA, Dec. 2001.

[4] M. A. Hamilton, “Java and the shift to net-centric computing,” *IEEE Computer*, vol. 29, pp. 31–39, Aug. 1996.

[5] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. Cambridge, MA: MIT Press, 1992.

[6] K. W. Arthur, K. S. Booth, and C. Ware, “Evaluating 3D task performance for fish tank virtual worlds,” *ACM Transactions on Information Systems*, vol. 11, no. 3, pp. 239–265, 1993.

[7] J. R. Wilson, “UAVs and the human factor,” *AIAA Aerospace America*, vol. 40, pp. 54–57, July 2002.