MIXED INITIATIVE EXPERIMENTAL (MIX) TESTBED

HOW-TO GUIDE FOR USERS TO EASILY DEVELOP EXPERIMENTS

1 TABLE OF CONTENTS

2 INTRODUCTION

MIX generates simulations of unmanned vehicles, which work with the Operator Control Unit (OCU) software, generating an interface/menu to control any unmanned system compliant with the Joint Architecture for Unmanned Systems (JAUS), whether an actual device or simulation.

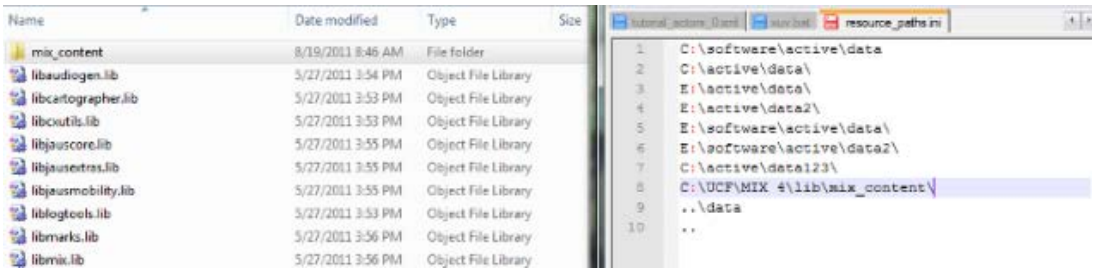
2.1 INSTALLING THE SOFTWARE

If MIX is not already installed:

- Go to http://sourceforge.net/projects/active-ist/files/MIX/
- Click the folder for MIX Installer and download the latest version (e.g. Mixed Initiative Experimental Testbed-X.XXXXX-win32.exe)
- Click Parent Folder, then MIX Content, and download the latest models, terrains, and audio .zip files (e.g. terrains- X.XXXXX.zip, models- X.XXXXX.zip, and audio- X.XXXXX.zip), then unzip them
    - You can unzip these content files to any location on your system, just remember where you extracted them to as you will need this information later
    - If the version number is part of the extracted folder filename, rename them to terrains, models, audio, respectively
    - Make sure you've extracted all content to the same directory, your content folder should look something like the following
        - <YOUR MIX CONTENT FOLDER>/audio
            - <YOUR MIX CONTENT FOLDER>/audio/clips
            - <YOUR MIX CONTENT FOLDER>/audio/questions
            - …
        - <YOUR MIX CONTENT FOLDER>/models
            - <YOUR MIX CONTENT FOLDER>/models/effects
            - <YOUR MIX CONTENT FOLDER>/icons
            - …
        - <YOUR MIX CONTENT FOLDER>/terrains
            - <YOUR MIX CONTENT FOLDER>/terrinas/gmet
    -

- Install “Mixed Initiative Experimental Testbed- X.XXXXX -win32.exe”
- Then go to C:\UCF\MIX\bin\settings and open “resource_paths.ini” to add the path to the content folder you created previously
- Type at the bottom of that file the location of the folder, then press enter, then save.

Example: If the folder is C:\UCF\MIX\lib\mix_content, then write at the bottom of the file C:\UCF\MIX\lib\mix_content.



NOTE: Be sure to **press enter** after typing in the file path, then save it.

- Finally, install OpenAL
  - Go to the MIX install path, then open the “ext” folder. Run OpenAL11CoreSDK.exe and follow the default installation instructions.
- Optional – Install Microsoft Redist Files
  - For some machines it may be necessary to install the Microsoft C/C++ Runtime Files. If you receive any missing DLL errors when running, download the latest Visual C++ Redistributable for Visual Studio, the current versions is for 2012 and can be located here:
    - http://www.microsoft.com/en-us/download/details.aspx?id=30679
    - If asked to choose a version, choose the x86 version of the file.

If MIX is already on your computer, and you are installing a newer version:

- Remove the Content folder if it is anywhere in the MIX folders (otherwise, the Content data would have to be downloaded again).
- Open Uninstall.exe in UCF/MIX and do a complete uninstall, you can also do this from Control Panel.
- Install the new version.

2.2 TESTING YOUR INSTALLATION

Once you have installed everything you can test your installation by running any of the tutorial or other example scenarios included with the installation. To do so, go to the installation directory for MIX, and open the folder named bin. From there you will see a folder named scenarios. The scenarios folder includes different project configurations for various experiments/training tasks that reuse common settings from the settings folder. Test your install by going to bin/scenarios/tutorial/tutorial_00/bats. Then double click on xuv_standard.bat, to run a simulated robot. Once that starts, double click on the ocu_standard.bat to run the OCU. You should see the following:

2.3 ABOUT THIS GUIDE

Preparing a complete scenario for an experiment requires three major steps

1. Determining the desired configuration of the OCU menu
2. Setting up the conditions of the experiment
3. Creating a scenario in the Scenario Editor, then testing the scenario programs

3 STEP 1: DETERMINE DESIRED CONFIGURATION OF OCU

The OCU determines the "look and feel" of the experiment as it runs. It is up to the experimenter to decide the desired elements and relay this information to the programmers. The OCU is either:

1. an existing one edited to fit the experiment
2. a new one created specifically for this experiment

3.1 EDIT EXISTING OCU

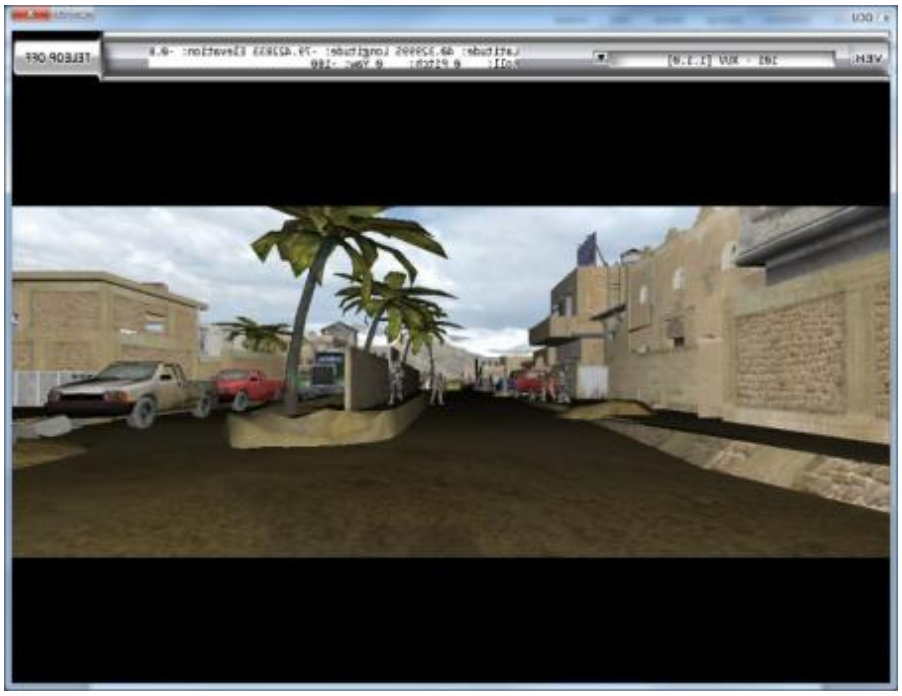
Usually, the OCU provided is a variant of an existing one, tweaked by the software developers by request. Many elements can be easily customized, added, or removed by the programmers, such as:

- Number of vehicles and cameras
- Maps types and icons on maps
- Number/Types of buttons
- Text prompts and gauges
- Position of any of the above components

3.1.1 STANDARD OCU

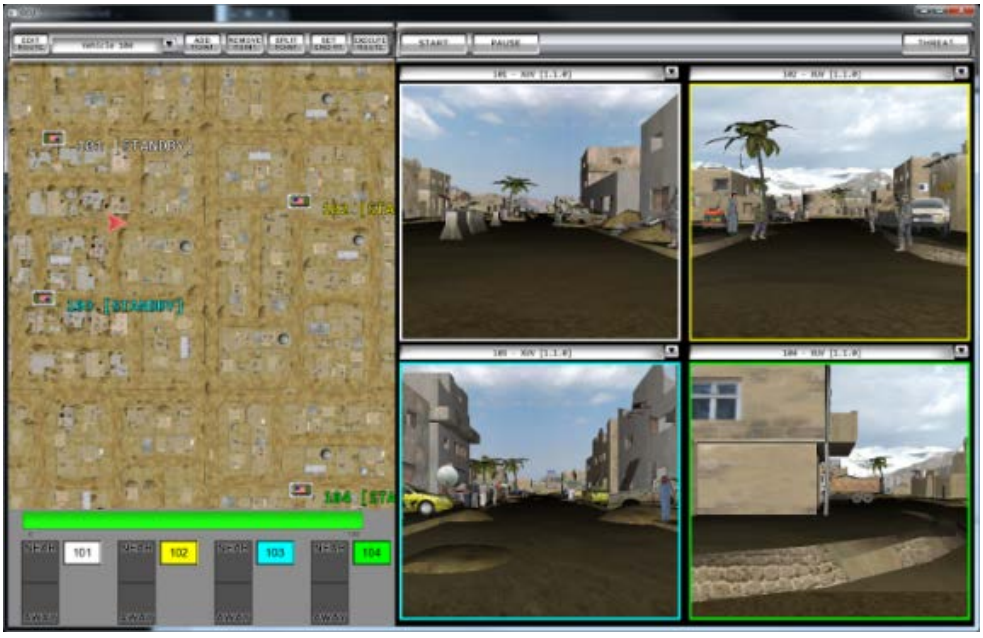
This is a standard layout of an OCU, mainly intended for simple observation of a single vehicle taking a route and the environment around it.

3.1.2 SMALL OCUS WITH SINGLE VIEWPOINT

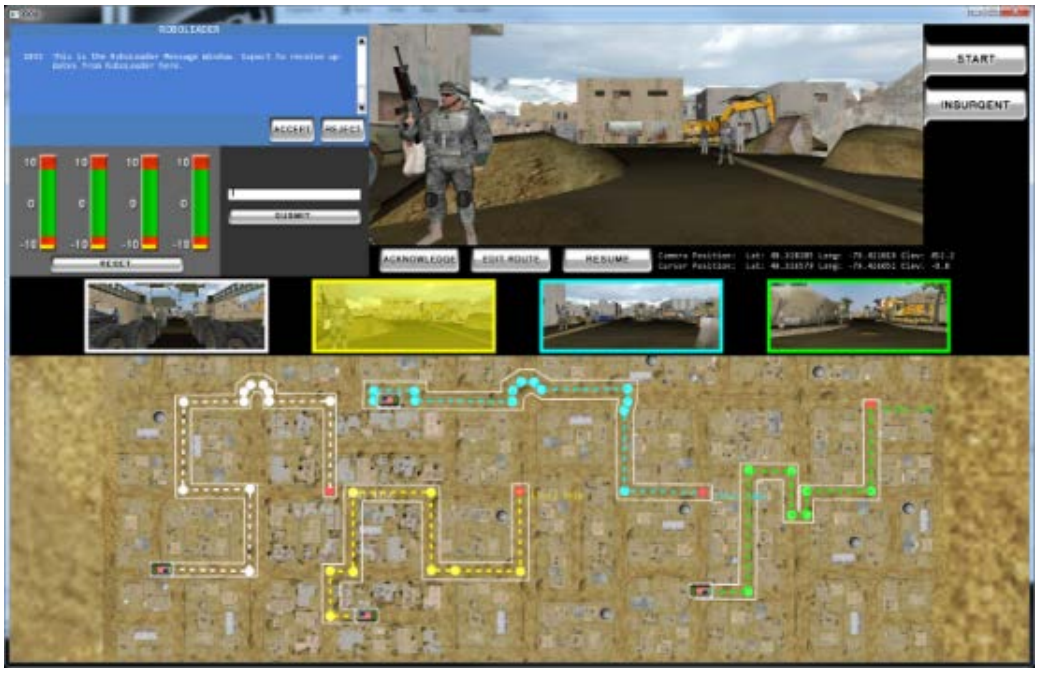


OCUs can be made with as many or few buttons and windows as necessary, such as the example of OCU mini above, which only has a single vehicle perspective, and a few buttons for identifying

3.1.3 MULTIPLE VEHICLE DISPLAYS



There are a few layouts available that accommodate multiple vehicle displays for judging how a subject multitasks. Another example of an OCU with multiple vehicles, with more information on display, can be seen below



3.2 CREATING NEW OCU IF NECESSARY

Rarely will a completely new OCU be needed since existing ones can be easily edited to fit many situations, but it is possible for one to be created if an experiment truly requires it.

4 STEP 2: SET UP CONDITIONS FOR PROGRAM

Experimenters are provided with a basic template for their scenario. It will then be up to the experimenters to edit the files and create the missions for that scenario. It is possible to skip this section for now and go to Step 3: Create & Test Mission, referring to this section as you go to understand each of the files and folders.

4.1 FOLDER STRUCTURE

Go to MIX 4/bin/scenarios/tutorial/tutorial_00. Note that some of the tutorial_00 folders are empty and are only there for demonstration, since some of them aren't always needed.

Folder	*Purpose*
actors	Contains the files describing placement of models in scenarios
analysis	Contains files for data processing for instances of experiments
areas	Contains files defining binding zones on map (i.e. hostile area, blocked area, high priority)
bats	Contains files that run configurations of the OCU and the MIX simulators
extras	Holds miscellaneous data, such as text files with additional info
missions	Contains files that hold the data for “missions”, the paths that vehicles travel
ocu	Contains files holding OCU config data
platforms	Contains files holding vehicle sim data for the scenario.
scripts	Contains folders with files that hold data on events/commands.
streets	Contains files that define the street layout of the terrain, which can be used as an overlay in the scenario editor; Intended to simplify creating mission routes

4.2 XML FILES

Almost all files that control settings are xml files. To open these, right click, then select “Edit with Notepad++”. Note that the lines in green are comments that don’t run as code, but are there to clarify what the settings do. Text can be commented out by surrounding it with “<!--" in the front and “-->” in the back.

```
<Component name="SituationalAwarenessMap">
    <!-- Sets the start position on the terrain of the camera for this map. -->
    <Position latitude="40.328482" longitude="-79.421902" elevation="250.0" rotation="-1.57079633"/>
</Component>
<Component name="MapNavigator">
    <Camera locked="false"/>
    <!-- Sets the start position on the terrain of the camera for this map. -->
    <Position latitude="40.328482" longitude="-79.421902" elevation="250.0" rotation="-1.57079633"/>
</Component>
```

This is the pertinent information on the xml files located in the previously mentioned folders.

4.2.1 ACTORS

Open tutorial_04/actors/tutorial_actors.xml.

```
<SimManager>
    <Actors>
        <Actor model="Insurgent 01" name="1" state="RelaxedB" gclamp="true">
            <Position latitude="40.328265" longitude="-79.421862" elevation="-0.807830" />
            <Attitude roll="0" pitch="0" yaw="-3" />
        </Actor>
        <Actor model="Havic Foreign Military 04 Sweat" name="2" state="Default" gclamp="true">
            <Position latitude="40.328322" longitude="-79.421862" elevation="-0.807830" />
            <Attitude roll="0" pitch="0" yaw="44" />
        </Actor>
        <Actor model="Generator" name="3" state="Default" gclamp="true">
            <Position latitude="40.328296" longitude="-79.421729" elevation="1000.000000" />
            <Attitude roll="0" pitch="0" yaw="-84" />
        </Actor>
        <Actor model="HUMV998" name="4" state="Default" gclamp="true">
            <Position latitude="40.328343" longitude="-79.421982" elevation="-0.807824" />
            <Attitude roll="0" pitch="0" yaw="44" />
        </Actor>
    </Actors>
</SimManager>
```

This is the general look of an actor file. Each block of "Actor" text represents a single actor's info. This is created and updated when you edit and save this actor file in the Scenario Editor, detailed later. Because of this, most of the time you won't have to edit the documents yourself.

The exceptions would be if:

- You want a model above ground: All models by default have the "gclamp", or ground clamp, option set to true. To have a specific elevation for the model, change elevation to a desired number, set gclamp to false, and then save.
- You place Fire or Flare Smoke, which cannot be deleted or moved in the editor: To change position, alter the latitude and longitude numbers, then save. If you want to delete these models, delete the actor text block (from "<Actor model=…" to "...</Actor>"), then save.

4.2.2 ANALYSIS

The analysis folder holds files that control how info from scenarios is processed by separate data processing tools. This includes how the participant identifies elements of the environment, at what times certain actions and events occur, where the vehicle has been and at what times, and what responses the participant types in when prompted.

This is detailed further in the Step 3: Scenario Editing, Testing, and Tutorials, under the Logging Options tutorials.

4.2.3 OCU

The many components of OCU files are detailed further in Section 5: Configuring OCU of "User Guide.docx", located in MIX 4/src/applications/mix/4.0/docs/

In general, the different fields control the look of the OCU, which terrain is assigned, and which scripts are associated with it. Look at tutorial_00/ocu/standard.xml, as an example of a file. Since most of what an experimenter has to alter is within the OCU, how and when to alter the OCU composes the majority of Step 3: Scenario Editing, Testing, and Tutorials.

4.2.4 PLATFORMS

Open tutorial_00/platforms/standard.xml.

```
<?xml version="1.0" standalone="yes" ?>
<USSIM>
    <!-- This tag specifices the 3D Simulation Environment Settings File. -->
    <SimManagerConfig>settings/terrains/gmet/xuv_simmanager_settings.xml</SimManagerConfig>
    <TerrainMap>../../../../../data/terrains/cartographer/gmet/terrain.xml</TerrainMap>

    <!-- Define a platform (unmanned vehicle) id is the Subsystem Number
         and identification is the type of vehicle. Other values will be provided by
         documentation. -->

    <!-- Testing/Demonstration Scenario.  -->
    <!-- Platform ID is the JAUS subsystem ID, identification is
         a string representing the type of vehicle (used to select physics model). -->
    <Platform id="101" identification="XUV">
        <!-- ActorsFile indicates a scenario file to load to put
             people and objects, etc. in the scene. If clear="true" than
             any other actors are cleared before loading. -->
        <ActorsFile clear="true">scenarios/tutorial/tutorial_0/actors/tutorial_actors.xml</ActorsFile>
        <!-- Starting position and orientation of vehicle in world. -->
        <InitialPose>
            <Latitude>40.329995</Latitude>
            <Longitude>-79.422833</Longitude>
            <Altitude>10</Altitude>
            <Roll unit="degrees">0</Roll>
            <Pitch unit="degrees">0</Pitch>
            <Yaw unit="degrees">-180</Yaw>
        </InitialPose>
        <!-- Physics update rate in Hz. -->
        <UpdateRate>30</UpdateRate>
        <!-- If 1, collisions are turned off when vehicle is driving (autonomous) itself. -->
        <DisableCollisionsOnAutonomous>1</DisableCollisionsOnAutonomous>
        <!-- Physics settings to use (sets max speed, etc.)-->
        <PhysicsModelConfig>settings/platforms/physics_xuv.xml</PhysicsModelConfig>
        <!-- Services information to use for this simulation.  The
             settings attribute is for the XML file with default values
             for the services beign loaded.. -->
        <Services settings="settings/jaus/services.xml">
            <VisualSensor>
                <!-- OSG Camera name, and model view to attach to.-->
                <CameraInfo name="Default" view="Front"/>
            </VisualSensor>
        </Services>
    </Platform>
</USSIM>
```

This is the general look of a MIX simulator file.

Xml Tags	*Purpose*
SimManagerConfig	Address for data for the vehicle and the terrain it will be in; only change if vehicle type changes
TerrainMap	Address for terrain map data; Only change if terrain changes.
Platform	Block of text with info for each specific vehicle ID and identification
ActorsFile	Address for associated actors file; If clear=true, then any other actors will be cleared before loading
InitialPose	Position of vehicle on terrain (latitude, longitude, altitude, roll, pitch, yaw); must be manually done, but can reference editor to find positions
UpdateRate	Physics update rate; no need to change this
DisableCollisionsOnAutonomous	If 1, collisions are off when vehicle drives itself
PhysicsModelConfig	Address for physics data for specific vehicle
Services	Address for specific JAUS services
VisualSensor	Contains camera information; which camera to use and its position

Users need mainly to be concerned with defining the start position of a vehicle in InitialPose. Otherwise, much of the document requires no editing by the user.

4.2.5 SCRIPTS

Open tutorial_00/scripts

Component	*Purpose*
1: adaptiveauto	Controls automatic toggling for teleop control (manual control of vehicle)
2: audio	Controls playback of audio files
3: blankscreen	Controls display of "blankscreens", pauses in scenarios where a blank screen, sometimes with text, is displayed
4: gauge	Controls the display and behavior of gauges
5: intel	Controls display of "intel", frequently updating info feed which responds to triggered events
6: mapicons	Controls display and changes in icons displayed on the map

The many components and what to do with them are detailed further in the tutorials in Step 3: Scenario Editing, Testing, and Tutorials.

4.3 BATCH (BAT) FILES

Open tutorial/tutorial_00/bats/ocu.bat with "Edit with Notepad++". These files must reference the correct xml file for OCUs and simulation programs to run. The example below must have the text "this_does_nothing.xml" replaced with "standard.xml" for ocu.exe use the standard.xml file located in tutorial_00/ocu.

```
cd ..
cd ..
cd ..
cd ..
ocu.exe scenarios/tutorial/tutorial_00/ocu/this_does_nothing.xml 0
```

5 STEP 3: SCENARIO EDITING, TESTING, AND TUTORIALS

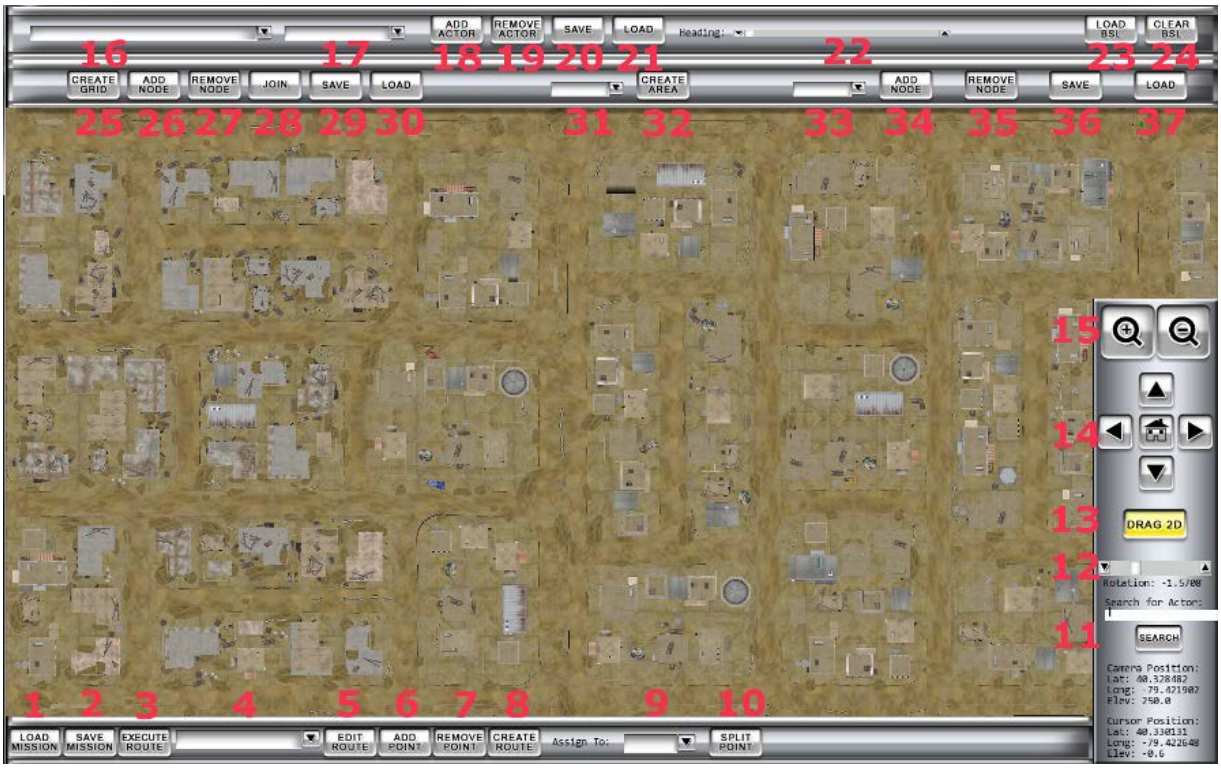
These tutorials will educate users on creating progressively complex experiments. Though there are other terrains that can be used, these examples will utilize the Geotypical Middle Eastern Terrain (GMET).

Note: The 29 Palms environment cannot be used for creating an experiment without explicit permission from the Army. 29 Palms is also not in any Content files.

5.1 THE EDITOR & OCU

5.1.1 COMPONENTS OF THE EDITOR

Each scenario will have a Scenario Editor of some kind. Open tutorial/tutorial_00/bats/tutorial_gmet_editor.bat.



Component	*Purpose*
1: Load Mission	Opens list of mission files to load
2: Save Mission	Saves current mission data to loaded file or new file name
3: Execute Route	Not used in editor
4: Route Dropdown	Shows which vehicle the selected route applies to
5: Edit Route	When clicked and highlighted, Add Point, Remove Point, and Split Point become active
6: Add Point	Click an area on the map to add an new point to a route; click button again to cancel
7: Remove Point	Click a point in route to remove it; click button again to cancel
8: Create Route	Create new route in mission

9: Dropdown (Assign to)	Assign a route to a vehicle ID
10: Split Point	Click a point in route to make a second point in between it and another point
11: Search for Actor	Write number ID for actor, then click search for camera to center on actor
12: Map Rotation Slider	Rotates map
13: Drag 2D	Toggle whether map can be dragged with mouse
14: Map Position Controls	Use arrows to shift camera over map in that direction; home button not used in editor
15: Zoom In/Out	Use buttons to zoom in and out of map
16: Actor Dropdown	Select actor type to add
17: Actor State Dropdown	Select different state of actor to be added, if available
18: Add Actor	Click on area on map to add actor in Actor Dropdown; click button to cancel
19: Remove Actor	Click on actor to remove; click button to cancel
20: Save (Actors)	Save current actor data to an Actors file
21: Load (Actors)	Load an Actors file to editor
22: Heading Slider	Rotate selected actor
23: Load BSL	Load Blank Screen Logs (BSL), which shows where all vehicles were located at a specific time a blank screen script was used in a scenario
24: Clear BSL	Clears the BSL data currently open from the editor
25: Create Grid	Clear any grid that is on the screen
26: Add Node (Grid)	Click on map to add a node that will join the grid
27: Remove Node (Grid)	Click on a grid node to remove
28: Join	Connects/disconnects the last two nodes clicked
29: Save (Grid)	Saves grid to file
30: Load (Grid)	Loads existing grid file to map
31: Area Type Dropdown	Select area type to create
32: Create Area	Creates area with its own number ID (starts with zero points)

33: Area ID Dropdown Menu	Select area number to add or remove nodes
34: Add Node (Area)	Add a node for assigned area; points connect in order placed
35: Remove Node (Area)	Remove a node for assigned area
36: Save (Area)	Save area data to file
37: Load(Area)	Load area data file

Other controls include:

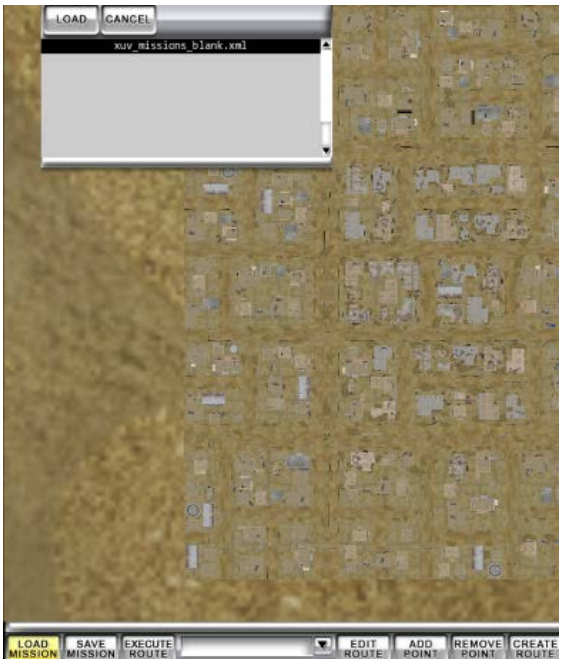
- Hold right click and drag to zoom in/out gradually
- Roll scroll wheel to zoom in/out drastically
- Hold click and drag to slide map in that direction
- Press Esc to close the program

These controls apply to other OCUs as well.

5.1.2 LOADING DATA

From the editor, existing files can be loaded, edited, and saved over. To load a mission:

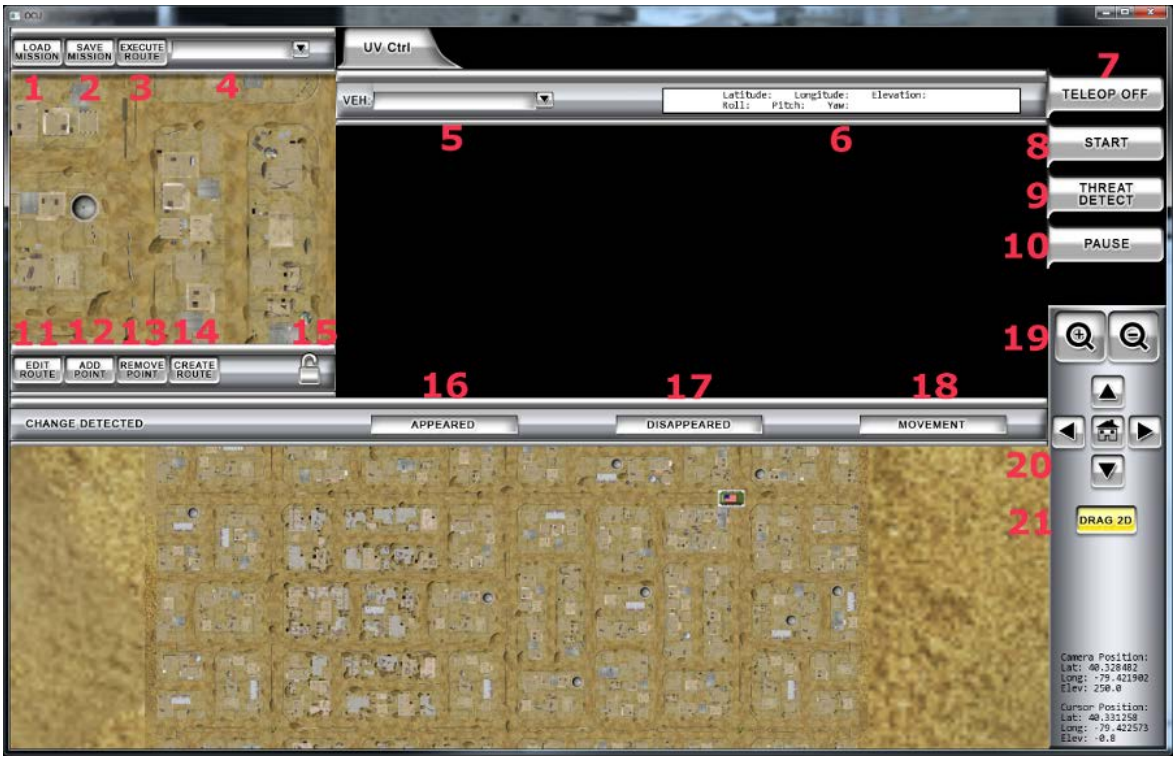
- Click the Load Mission button on the bottom left
- Click a file, such as “xuv_missions_blank.xml”, then click load



Since this specific file doesn't actually have any route information, no routes will appear on the map. It is a similar process to open an Actors file, Grid file, Area file, and BSL in both the editor and OCUs

5.1.3 COMPONENTS OF THE OCU

Open tutorial/tutorial_00/bats/ocu_standard.bat. OCUs can look different from this standard setup, but they all share many of the same components.



Component	*Purpose*
1: Load Mission	Opens list of mission files to load
2: Save Mission	Saves current mission data to loaded file or new file name
3: Execute Route	Starts route if mission paused for editing
4: Dropdown (Mission)	Shows which vehicle the selected route/minimap applies to
5: Vehicle Dropdown	Select from currently running vehicle sims to add to OCU
6: Vehicle Position	Stats on current vehicles position
7: Teleop Off	Toggles the ability to manually control the vehicle with a joystick
8: Start	Click to start mission
9: Threat Detect	When clicked, click on perceived threat record sighting; click again to cancel

10: Pause	Pauses commencing mission; click again to resume
11: Edit Route	When clicked and highlighted, Add Point and Remove Point become active, and the mission pauses
12: Add Point	Adds a new point for the mission route selected
13: Remove Point	Removes point from route
14: Create Route	Create new route in mission
15: Lock	Click to toggle whether camera can be moved or not on smaller map
16: Appeared	Click to record that an icon on the larger map appeared
17: Disappeared	Click to record that an icon on the larger map disappeared
18: Movement	Click to record that an icon on the larger map moved
19: Zoom In/Out	Zoom camera in and out
20: Map Position Controls	Use arrows to shift camera over map in that direction, and home button to shift map to default position
21: Drag 2D	Toggle whether map can be dragged with mouse

5.2 TUTORIAL 00: STARTING A MIX SIMULATION

To run a MIX sim program, the bare minimum necessary is a running OCU and MIX vehicle sim.

- Go to tutorial/tutorial_00/bats
- Open “ocu_standard.bat”, and “xuv_standard.bat”. Four windows will open: two command prompts, the OCU, and an XUV vehicle simulation. Do not close any of these, but the only thing of concern is the OCU
- Load the mission “xuv_mission_blank.xml”
- On the OCU, click the Vehicle Dropdown and click the vehicle “101-XUV [1.1.0]”.The camera feed on “xuv.bat” will now appear in the OCU. If it is not there, wait a moment and click the dropdown again.

This is the general process of opening an experiment scenario. Most of the buttons and components here will not do anything, but they are all active for demonstration purposes.

5.3 TUTORIAL 01: CREATING AND TESTING ROUTES

Warning: Every edit in the Scenario Editor must be saved before clicking the close button or pressing Esc. Otherwise, all changes will be lost. There is currently no prompt to save before closing. It is recommended to save each component after every change.

The vast majority of MIX programs have vehicles traveling a route created in the editor.

- Open tutorial/tutorial_01/bats/ gmet_editor/tutorial_gmet_editor.bat
- Load mission “standard_short.xml”; a route with two points will appear

Note that when a route is selected, the beginning point is always white (not just on white colored routes) and the ending point is always red (not just red routes). Become familiar with how routes work.

5.3.1 ROUTES

- Add a point to a route
    - Click the
    - Click Edit Route
    - Click Add Point
    - Click on the map; a new point will appear and connect to the end of the currently selected route
- Move a point on a route
    - Click Edit Route
    - Click and drag any point on the currently selected route to move it to the desired location
- Add a point between existing points (split a point)
    - Click Edit Route
    - Click on a point on the currently selected route; a new point will appear in between that point and the previous point it was connected to
- Remove a point
    - Click Edit Route
    - Click on a point on the currently selected route; the point will be removed and its previously connected points will connect to each other

All changes with missions must be saved with Save Mission.

5.3.2 TESTING THE ROUTE

- Open “ocu_standard.bat” and “xuv_standard.bat”, located in tutorial/tutorial_01/bats
- Load the mission “standard_short.xml” and add the vehicle “101-XUV [1.1.0]”.
- Click Start, then enter 1 in User ID (default), then click OK.

The vehicle will move along the route until it reaches the end point. Edit Route, Add Point, Remove Point, and (detailed later in the document) Create Route can be used in the same manner as in the editor, as long as the buttons are enabled.

5.4 TUTORIAL 02: PRELOADING A ROUTE AT OCU OPEN

In the last tutorial, the route had to be manually loaded at the start of the program. In experiments, the route is usually set up to be automatically loaded.

5.4.1 PRELOADING ROUTE

- Open tutorial_02/ocu/standard.xml
- Look at the section Options-> Component name= “MissionBuilder”-> LoadMission file= “”.
- Make Load Mission file equal “standard_short.xml”

```
</XmlFiles>
<Options>
    <Component name="MissionBuilder">
        <LoadMission file="standard_short.xml"/>
    </Component>
    <Component name="SituationalAwarenessMap">
```

Running the regular OCU will now automatically load the “standard_short.xml” route.

5.5 TUTORIAL 03: ADDING ACTORS

Open tutorial_03/bat/tutorial_gmet_editor.bat. Then load the actor file “tutorial_actors.xml”. Become familiar with the different ways to work with actors.

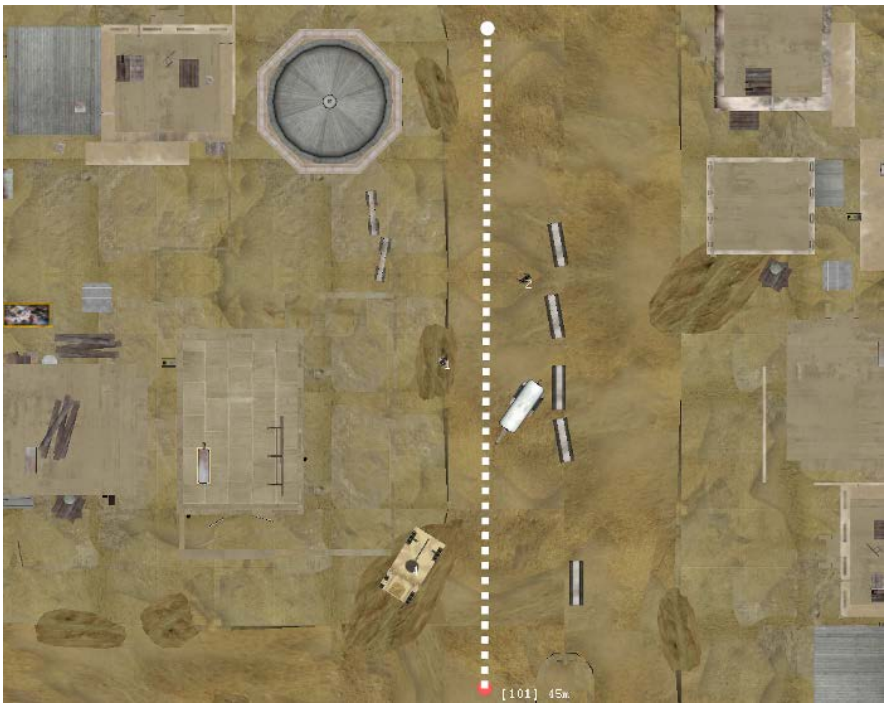
5.5.1 ACTORS

- Add actors
    - Navigate the Actor Dropdown and State Dropdown to find the desired actor
    - Click Add Actor
    - Click on a location on the map to place the actor
- Remove actors
    - Click Remove Actor
    - Click on the actor you want to remove
- Edit actors
    - Click and drag an actor to move it on the map
    - To rotate an actor, click on an actor, then shift the Heading slider

All changes with missions must be saved with Save Mission.

5.5.2 SEEING ACTORS IN OCU

Load the mission “standard_short.xml”. Then load the actors file “tutorial_actors.xml”. Place actors around the route to match this picture. The actor names and states, respectively, are listed below.



1. (figure on left) Insurgent 01, RelaxedB
2. (figure above) Havic Foreign Military 04 Sweat, Default
3. (white object) Generator, Default
4. (brown object) HumV998, Default

Use the Actor Type and Actor State Dropdown menus at the top of the editor.

Always save actor file when changes are made.

Now we make sure that the vehicle file will load this actor file when the vehicle sim runs. Open tutorial_03/platforms/standard.xml. Look at the section Platform id=”101”…-> ActorsFile. Make sure it specifies the filepath of the “tutorial_actors.xml” file, like the image below.

```
    a string representing the type of vehicle (used to select physics model). -->
<Platform id="101" identification="XUV">
    <!-- ActorsFile indicates a scenario file to load to put
         people and objects, etc. in the scene. If clear="true" than
         any other actors are cleared before loading. -->
    <ActorsFile clear="true">scenarios/tutorial/tutorial_02/actors/tutorial_actors.xml</ActorsFile>
    <!-- Starting position and orientation of vehicle in world. -->
    <InitialPose>
        <Latitude>40.328297</Latitude>
```

Actors will now be visible once you run the scenario (OCU and the vehicle).

5.6 TUTORIAL 04: LOGGING OPTIONS: THREAT DETECTION

Scenarios are usually made to record, or log, certain data for each run of a participant. These logs are made into Comma Separated Value lists, or Excel documents, and output in the logs folder in MIX. These logs include:

- GUI Logs: Record when certain buttons are clicked or when the vehicles view window is clicked
- Text Input Log: Records when and what text the participants input during their scenarios
- Vehicle Position Log: Records where the vehicle is on the map at certain time intervals
- Simulation Time Log: Records the start time and duration of the scenario
- Blank Screen Script Log: Records where vehicle was at specific blank screen times
- Target Log: records info on targets identified
- Threat Detection: Records info on the time and location of when a possible threat was identified, which participant located it, and how correct the choice was.

The Threat Detect Button in the standard OCU currently has no affect. This can be changed by editing the OCU file.

5.6.1 LOGGING SETUP

Two files in the tutorial_04 folder control the logging settings: the OCU file and a new file, an analysis file.

- Open tutorial_04/ocu/standard.xml
- Look in the section XML Files; notice a section at the bottom called Logging, which is commented out
- Uncomment this by deleting the “<!--" in the front and “-->” in the back

```
<XmlFiles>
    <!-- Specifies colors, buttons, graphics used by OCU. -->
    <Styles                    file="settings/styles/fcsx_style.xml" />
    <!-- Layouts specifies where visual elements and controls should appear on the window. -->
    <Layouts                    file="settings/layouts/standard_layout.xml" />
    <!-- Components defines individual components (e.g. Video Panel, Mission Building) settings. -->
    <Components                file="settings/components/standard_components.xml" />
    <!-- SimManager file specifies terrain, lighting, colors, 3D models, etc. to use in environment. -->
    <SimManager                  file="settings/terrains/gmet/ocu_simmanager_settings.xml" />
    <!-- NodeManager specifies JAUS specific settings for communication with unmanned systems. (very rare
    <JAUS                file="settings/jaus/ocu_services.xml" />
    <!-- Default Logger Settings. -->
    <Logging           file="settings/ocu_logger_tutorial_special.xml" />
```

The logging settings for this OCU come from the file bin/settings/ocu_logger_tutorial_special.xml. Open that file.

```
<?xml version="1.0" standalone="yes" ?>
<OcuLogger>
    <!-- Indicates how long to log data for if set. A value of 0.0
         will not log anything. Value is in minutes.seconds format (e.g. 1.30, 2.45).
         A value of -1 (default) will log indefinitely. -->
    <LoggerTimeLimit>-1</LoggerTimeLimit>
    <!-- Targets lists the model types to consider as threats.  This is used
         to evaluate if a user correctly identified a threat model in the
         3D scene. All other model types will be considered non-threats. -->
    <Targets>
        <TotalTargets>2</TotalTargets>
        <Target>Insurgent 01</Target>
        <Target>Generator</Target>
    </Targets>
</OcuLogger>
```

Compare with the file tutorial_04/analysis/example_threat_detection_logger.xml.

```
<?xml version="1.0" standalone="yes" ?>
<!-- Data analysis script for Tutorial 04 -->
<Analysis>
    <OutputDirectory>logs/Threat Playback Logs/</OutputDirectory>
    <Experiment name="tutorial_04" />
    <SimManager render_delay="5000" file="settings/terrains/gmet/xuv_simmanager_settings.xml"></SimManager>
    <!-- If 1, then go through every position record, otherwise skip the number listed (speeds things up). -->
    <RecordIncrement>10</RecordIncrement>
    <!-- List model types that are considered "Threats."-->
    <Reports>
        <Source name="Threat Detection Log">
            <Threat>Insurgent 01</Threat>
            <Threat>Generator</Threat>
        </Source>
    </Reports>
    <CameraVerticalOffset value="1.5"/>
    <!-- How high off the ground camera should be -->
    <ActorQualifiers dist="50"/>
    <!-- Actors must have minimum area on screen and be within distance of vehicle -->
    <NoRepeat value="1"/>
    <!-- If value == 1, then if an output file exists already new analysis is not done. -->
    <FoldersToIgnore>
        <Folder name="Pilot"/>
        <Folder name="Screen Capture"/>
        <Folder name="Audio"/>
        <Folder name="GUI Events"/>
        <Folder name="Pilot"/>
        <Folder name="NASA TLX"/>
        <Folder name="Spatial Orientation"/>
        <Folder name="Bin"/>
        <Folder name="VProRecovery"/>
        <Folder name="Guides"/>
        <Folder name="RECYCLER"/>
        <Folder name="Deffective"/>
    </FoldersToIgnore>
</Analysis>
```

This is the file that defines where the logs will actually go, and what will be recorded in those logs.

Actors can be classified as Targets or Threats, as Safe-Targets (non-threats), or not classified at all. In this example, a generator is set as a threat, which normally isn't the case. Settings file ocu_logger_default.xml, shows a more typical classification of threats.

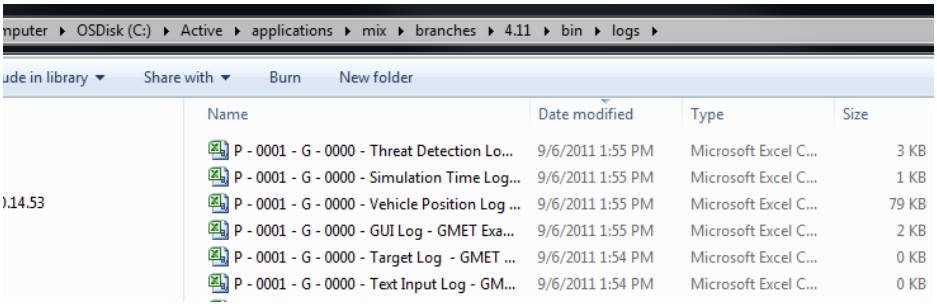
Refer to the Document "Threat Detection Task" located in///FILEPATH PENDING, WORK IN PROGRESS/// for more details on how to add threat detection to a program.

5.6.2 VIEWING LOGS

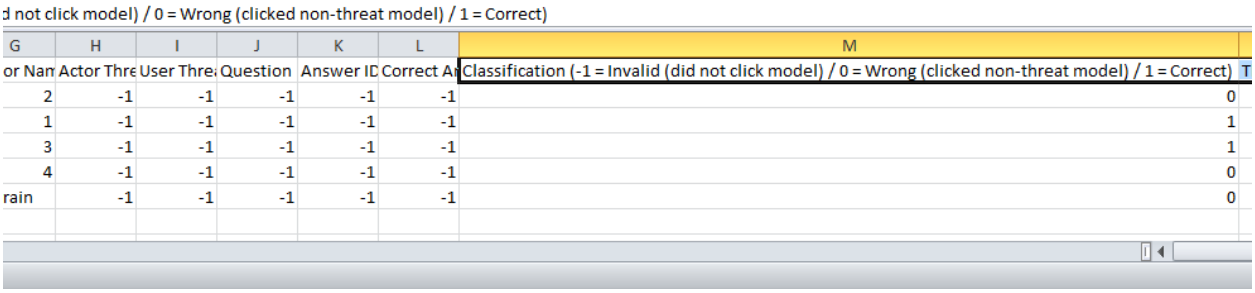
Run the scenario and start the vehicle. While the vehicle is following its route, click the Threat Detect button, and then on one of the four actors, such as the "Havic Foreign Military 04 Sweat" pointed at in the image below.

The Threat Detect button will now be unhighlighted. Do the same process with the Insurgent 01, Generator, HUMV998, and then once anywhere else on screen.

Once the program has been closed, go to the directory bin/logs. Note that there are six new logs, with "Date modified" tags close to the time the OCU and vehicle sim were shut down.



These logs record various kinds of information from a session. Open the Threat Detection Log file.



Look at the column M, Classification. Each row correlates with a clicked area, identified in column F, Model Type. For Insurgent 01 and Generator, Classification marked them as correctly identified, while the rest were listed as wrong. Clicking anything that isn't a model will output as a Low Terrain.

5.7 TUTORIAL 05: DISABLING/ENABLING BUTTONS

So far, all the buttons and other components on the OCU have been enabled. This isn't always the case in an experiment, especially with components that can alter the session in unwanted ways.

5.7.1 DISABLING LOADING/SAVING/EDITING

In this example, the basic ways routes and missions can be edited during the program will be disabled.

- Open tutorial_05/OCU/standard.xml
- Go to the section OCU-> Controls; note that there are several named controls, each with a flag called "enabled", which is either set to true or false
- Set LoadMission, SaveMission, ExecuteRoute, EditMission, AddPoint, RemovePoint, and CreateMission to "false"

```
        <Control name="UpArrowButton" enabled="true"/>
        <Control name="DownArrowButton" enabled="true"/>
        <Control name="UpArrowButton" enabled="true"/>
        <Control name="LeftArrowButton" enabled="true"/>
        <Control name="RightArrowButton" enabled="true"/>
        <Control name="HomeButton" enabled="true"/>
        <!-- Disable Mission Load/Save and Editing Options
             because missions are pre-loaded and sent automatically
             when Start is pressed because we enabled auto_send_rout
        <Control name="LoadMission" enabled="false"/>
        <Control name="SaveMission" enabled="false"/>
        <Control name="ExecuteRoute" enabled="false"/>
        <Control name="EditMission" enabled="false"/>
        <Control name="AddPoint" enabled="false"/>
        <Control name="RemovePoint" enabled="false"/>
        <Control name="CreateMission" enabled="false"/>
        <Control name="Track" enabled="false"/>
    </Controls>
```

- Open the scenario; note that missions cannot be loaded, saved or executed from the minimap area, nor can routes be altered

Now the OCU and Vehicle run normally, except the mission route cannot be changed

5.8 TUTORIAL 06: AUDIO SCRIPTS

Look at Audio Task Manual.docx for a more detailed look at how to create the necessary files associated with having auditory events presented to the user during OCU scenarios.

5.8.1 AUDIO XML FILE

Look at tutorial_06/scripts/audio/standard.xml

```
Cues>
   <Cue info="Cue 001" filename="../../../../data/audio/questions/call_signs/001.wav"  time="0.03" />
   <Cue info="Cue 002" filename="../../../../data/audio/questions/call_signs/009.wav"  time="0.13" />
   <Cue info="Cue 003" filename="../../../../data/audio/questions/call_signs/031.wav"  time="0.23" />
   <Cue info="Cue 004" filename="../../../../data/audio/questions/call_signs/015.wav"  time="0.33" />
/Cues>
!--
robes are triggered based on position
   filename - Name of the wav file to play when event triggered.
   info - Some identifying information for the event used in logging.
   radius - Radius around the point to be triggered in meters.
->
Probes>
   <Probe info="Audio 1" filename="../../../../data/audio/questions/practice_complete.wav" latitude="40.328280" longitude="-79.421448" radius="4
/Probes>
```

There are two kinds of audio triggers, cues and probes.

Cues are triggered at a specific time after startup. Note the section Cues in the file. Each Cue section has a name (info), an audio filename, file location, and time (in minutes.seconds) audio plays after startup.

Probes are similar to cues, but triggered when a vehicle enters a certain area, defined by a coordinate point and a radius around that point. Note that the latitude (40.328280) and longitude (-79.421448) in the probe were obtained by using the editor to mouse over the end of the scenario route.

Also note that the file directory has many "../" symbols, which means its backtracking from the root directory, in this case the bin folder, to access the folder which holds the subfolders containing the audio files.

If a probe plays over the time a cue is scheduled, the cue will play after the probe ends plus the time listed in CueDelayTimeSeconds. If an audio cue plays over the time another cue is scheduled, the following cue will play immediately after the previous one.

5.8.2 ADJUSTING OCU

Tutorial 06's OCU file will have to be adjusted to use this audio file. Uncomment the line <AudioScript…standard.xml" />. Note that this line is set to loop="0". If this was 1, the audio files from the specified audio .xml would repeat in order repeatedly until the OCU is shut down.

```
<Conditions>
    <Condition id="0" name="GMET Example Standard">
        <XmlFiles>
            <AudioScript    loop="0" file="scripts/audio/standard.xml" />
        </XmlFiles>
        <Options>
            <Component name="MissionBuilder">
                <LoadMission file="standard_short.xml"/>
            </Component>
            <Component name="SituationalAwarenessMap">
```

Now when the OCU and vehicle run, there will be four distinct audio cues along the path and an audio probe at the end.

5.9 TUTORIAL 07: BLANK SCREEN SCRIPTS

Blank screens can be triggered when a pause at a specific interval is desired, especially when the participant's observational skills should be tested.

5.9.1 BLANK SCREEN FILE

- Open tutorial_07/scripts/blankscreen/standard.xml
- Note that there are two Events, each with a Question (a message displayed at the blank screen) and a trigger time for how long after startup it occurs

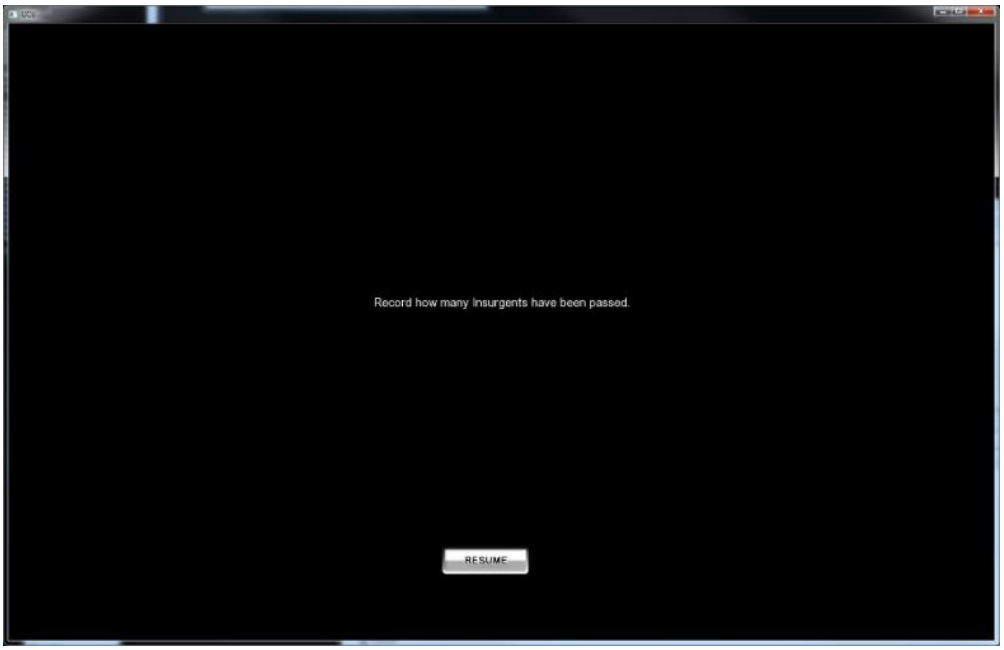
```
<BlankScreenScript>
    <!-- Events can be set using the following format:  -->
    <Events>
        <Event>
            <!-- Question to present to user. -->
            <Question>Record the color of the tank.</Question>
            <!-- When to trigger the event (minute.seconds)-->
            <TriggerTime>0.10</TriggerTime>
        </Event>
        <Event>
            <!-- Question to present to user. -->
            <Question>Record how many Insurgents have been passed.</Question>
            <!-- When to trigger the event (minute.seconds)-->
            <TriggerTime>0.30</TriggerTime>
        </Event>
    </Events>
</BlankScreenScript>
```

5.9.2 EDITING OCU FILE

- Open tutorial_07/ocu/standard.xml
- Go to the commented line in Condition id=”0” for BlankScreenScript.
- Note that looping is off, logging is on, and the file specified is the above blank screen file
- Uncomment the line

```
<Conditions>
    <Condition id="0" name="GMET Example Standard">
        <XmlFiles>
            <AudioScript    loop="0" file="scripts/audio/standard.xml" />
            <!--<BlankScreenScript    loop="0" logging="1" file="scripts/blankscreen/standard.xml" />-->
        </XmlFiles>
```

Now when the scenario runs, two blank screens prompts will appear, each with an example question.



5.10 TUTORIAL 08: ADAPTIVE AUTO SCRIPTS

With a script, teleop (manual control of the vehicle) can be automatically toggled on and off at set times.

5.10.1 ADAPTIVE AUTO FILE

- Open tutorial_08/scripts/adaptiveauto/adaptiveauto_example.xml
- Note three things
    - Two kinds of events can be triggered, “startteleop” and “stopteleop”, many times in a scenario
    - Each of these can be set to trigger at a certain time after startup (minutes.seconds)
    - An audio file is set for each switch on or off, with the below audio files being the defaults

```
<?xml version="1.0" standalone="yes" ?>
<AdaptiveAutoScript name="Adaptive Auto Script by Time">
    <TeleopOnNotification>../../../../data/audio/questions/question_099.wav</TeleopOnNotification>
    <TeleopOffNotification>../../../../data/audio/questions/question_100.wav</TeleopOffNotification>
    <!--
    Events can be set using the following format:

        <Event type=<EVENT_TYPE>>
            <TriggerTime<TIME_TO_TRIGGER></TriggerTime>
        </Event>

    Event Types available:
    type = "startteleop"    -  enables teleop mode.
    type = "stopteleop"    - disables teleop mode.

    Time is specified in the following format:
    "MM.SS"
    where MM is minutes, and SS seconds.
    -->
    <Events>
        <Event type="startteleop">
            <TriggerTime>0.00</TriggerTime>
        </Event>
        <Event type="stopteleop">
            <TriggerTime>.28</TriggerTime>
        </Event>
    </Events>
</AdaptiveAutoScript>
```

5.10.2 ADJUST OCU

- Open the OCU
- Note the AdaptiveAutomationScript line, which references the above file; uncomment it

```
<Conditions>
    <Condition id="0" name="GMET Example Standard">
        <XmlFiles>
            <AudioScript    loop="0" file="scripts/audio/standard.xml" />
            <BlankScreenScript    loop="0" logging="0" file="scripts/blankscreen/standard.xml" />
            <!--<AdaptiveAutomationScript file="scripts/adaptiveauto/adaptiveauto_example.xml" />-->
        </XmlFiles>
```

Now the scenario will start with the vehicle able to be controlled manually with a joystick, and then later switch it to automatically follow the path. Try experimenting with moving the vehicle.

5.11 TUTORIAL 09: LOGGING OPTIONS II: LOGGING SOFTWARE

Data created in logs is usually processed after a participant runs through a scenario. This is done with separate software from the OCU and MIX programs, including Vehicle Playback and Performance Analyzer.

5.11.1 VEHICLE PLAYBACK

Open tutorial_09/bats/vehicle_playback.bat with Notepad++

```
cd ..
cd ..
cd ..
cd ..
vehicle_playback.exe "logs/Tutorial 09 Logs/" scenarios/tutorial/tutorial_09/analysis/vehicle_playback.xml
```

Note that it refers to two filepaths. The one in quotes is the location where the logs will be taken from, while the second one is the xml file with vehicle playback settings. Open tutorial_09/analysis/vehicle_playback.xml

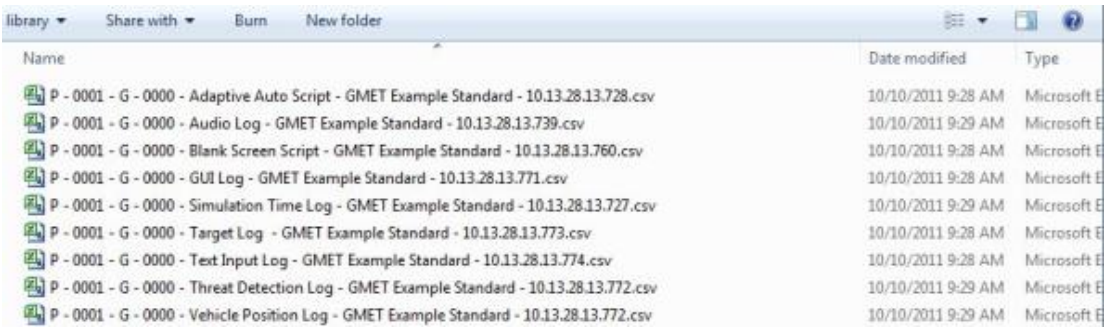
```
<Analysis>
    <OutputDirectory>logs/Tutorial 09 Logs/</OutputDirectory>
    <SimManager show_playback="0" render_delay="15000" file="settings/terrains/gmet/xuv_simmanager_settings.xml"></SimManager>
    <ProcessThreatDetect>1</ProcessThreatDetect>
    <Conditions>
        <Condition name="GMET Example Standard">
            <Vehicles>
                <Vehicle video="0" camera="0">101</Vehicle>
            </Vehicles>
            <PlatformsFile>scenarios/tutorial/tutorial_09/platforms/standard.xml</PlatformsFile>
        </Condition>
    </Conditions>
    <!-- List model types that are considered "Threats."-->
    <Targets>
        <Target>Insurgent 01</Target>
        <Target>Generator</Target>
        <Target>Havic Foreign Military 04 Sweat</Target>
        <Target>HUMV998</Target>
    </Targets>
    <!-- Actors must have minimum area on screen and be within distance of vehicle -->
    <ActorQualifiers dist="50"/>
    <!-- If value == 1, then if an output file exists already new analysis is not done. -->
    <NoRepeat value="1"/>
    <!-- Folders to ignore in scan. -->
    <FoldersToIgnore>
        <Folder name="1023"/>
        <Folder name="1024"/>
```

The sections of note are:

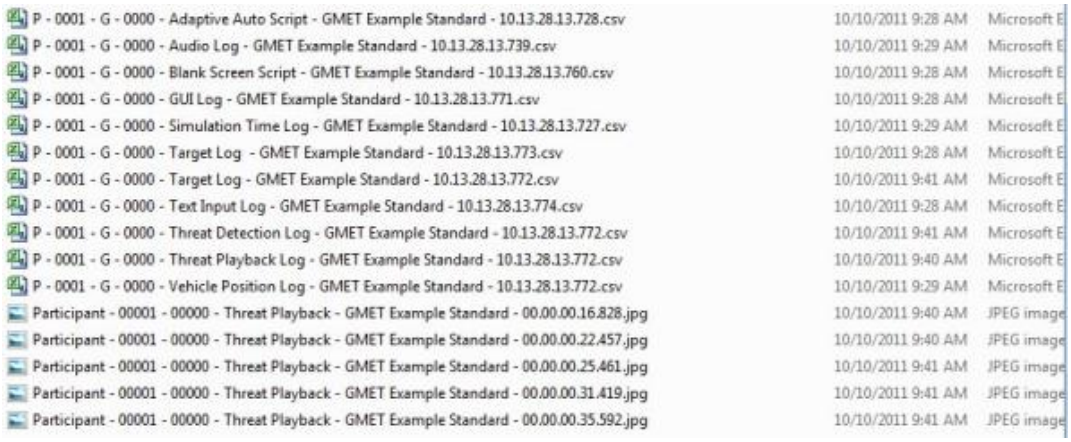
- Output Directory: location where files will be output
- Condition: holds vehicle and platform information for specific conditions
- Targets: Lists actor types in scenario

To properly run vehicle playback:

- Go to bin/logs. If you have been running the tutorials or other programs, there should be many logs in this folder and the subfolders. Delete any logs in the entire folder that have the string “GMET Example Standard” in them to make the following steps easier
- Run an instance of the scenario, preferably doing threat detection on the actors.
- Once that’s been done, go back to the logs folder. Create a folder called “Tutorial 09 Logs”, then place any logs with “GMET Example Standard” in the name into this folder.



- Run the vehicle playback program. This might take a while, since it runs a vehicle sim window recreating each instance of threat detection. Once it's done, check the Tutorial 09 Logs folder, where new logs for the vehicle playback will be, as well as screen captures of right when threat detection was made.



5.11.2 PERFORMANCE ANALYZER

Use Notepad++ to open the tutorial_09 bat file process_logs.bat

```
cd ..
cd ..
cd ..
cd ..
performance_analyzer.exe "logs/Tutorial 09 Logs/" scenarios/tutorial/tutorial_09/analysis/all_logs.xml
```

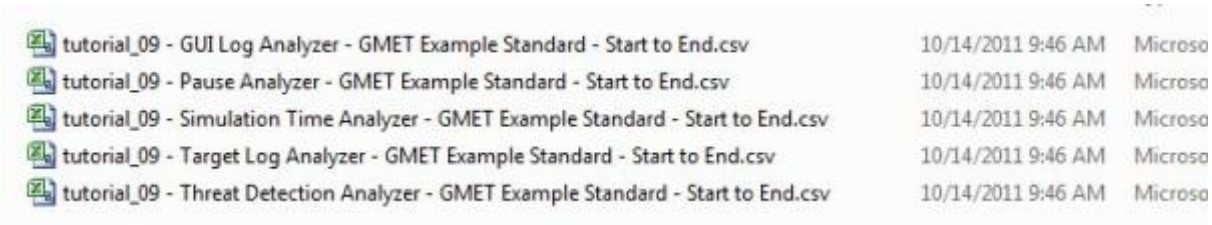
Just as with the vehicle playback program, this refers to both a log location and an xml file with performance analysis settings. Open tutorial_09/analysis/all_logs.xml

```
<Analysis>
    <Experiment name="tutorial_09" />
    <!-- Type of data this is for. -->
    <Reports>
        <Source name="Threat Detection Log">
            <Threat>Insurgent 01</Threat>
            <Threat>Generator</Threat>
        </Source>
        <Source name="Target Log">
            <TargetDistanceThreshold value="75"/>
            <Target model_name="Insurgent 01" icon_name="Threat ICON"/>
            <Target model_name="Generator" icon_name="Threat ICON"/>
            <Target model_name="HUMV998" icon_name="Neutral ICON"/>
            <Target model_name="Havic Foreign Military 04 Sweat" icon_name="Neutral ICON"/>
        </Source>
        <Source name="Threat Playback Log"></Source>
        <!--<Source name="Vehicle Position Log">
            <WaypointDistanceThresholdMeters>10</WaypointDistanceThresholdMeters>
        </Source>-->
        <Source name="GUI Log"></Source>
        <Source name="Simulation Time Log"></Source>
    </Reports>
    <Conditions>
        <Condition name="GMET Example Standard">
            <ScenarioFile>scenarios/tutorial/tutorial_09/actors/tutorial_actors.xml</ScenarioFile>
            <MissionFile>scenarios/tutorial/tutorial_09/mission/standard_short.xml</MissionFile>
        </Condition>
    </Conditions>
    <TimeBlocks>
        <!-- A block defines a period to perform analysis from a data log file.
             The resulting output will provide performance/simulation results for
             each time block (blocks can overlap).  Start and end times are in
             minute.seconds format. -->
        <Block start="0" end="0"/>
    </TimeBlocks>
    <!-- Lists folders to skip when doing analysis (speeds things up) -->
    <FoldersToIgnore>
        <Folder name="Screen Capture"/>
```

The sections of note are:

- Experiment: list name of experiment
- Reports: list several “sources”, which control log output, such as what is a threat or target
- Conditions: sets which mission and scenario files to look at for each condition name
- TimeBlocks: a period of time to perform analysis; when start and end equal zero, logging occurs until regular shutdown of the scenario occurred

If vehicle playback has been successfully run, run process_logs.bat. Then check bin/reports.



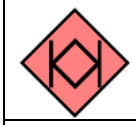
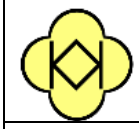
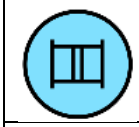
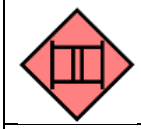
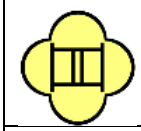
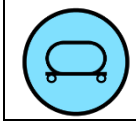
There are now logs here which have analysis of the performance in the scenario(s) by the subject(s).

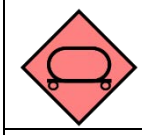
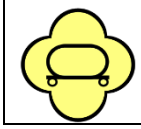
5.12 TUTORIAL 10: MAP ICONS & CHANGE DETECTION

Using Change Detection Scripts, map icons can be placed on the map to represent military units, which can appear, disappear, move, and change type/affiliation. Look at the file "Change Detection Task Manual.docx" in docs/Change Detection for more information on map icons.

5.12.1 ICON TYPES

Icons are classified into three types of units, and further classified by their affiliations.

Icon Graphic	ID Number	Description
	1	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.ARMINF **Affiliation:** Friend **General Description:** Armored Infantry
	2	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.ARMINF **Affiliation:** Hostile **General Description:** Armored Infantry
	3	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.ARMINF **Affiliation:** Neutral **General Description:** Armored Infantry
	4	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.ARMINF **Affiliation:** Unknown **General Description:** Armored Infantry
	5	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.TANK.MDM **Affiliation:** Friend **General Description:** Armored Tank
	6	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.TANK.MDM **Affiliation:** Hostile **General Description:** Armored Tank
	7	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.TANK.MDM **Affiliation:** Neutral **General Description:** Armored Tank
	8	**Model:** WAR.GRDTRK.EQT.GRDVEH.ARMD.TANK.MDM **Affiliation:** Unknown **General Description:** Armored Tank
	9	**Model:** WAR.GRDTRK.EQT.GRDVEH **Affiliation:** Friend **General Description:** Ground Vehicle

	10	**Model:** WAR.GRDTRK.EQT.GRDVEH **Affiliation:** Hostile **General Description:** Ground Vehicle
	11	**Model:** WAR.GRDTRK.EQT.GRDVEH **Affiliation:** Neutral **General Description:** Ground Vehicle
	12	**Model:** WAR.GRDTRK.EQT.GRDVEH **Affiliation:** Unknown **General Description:** Ground Vehicle

5.12.2 TYPES OF EVENTS

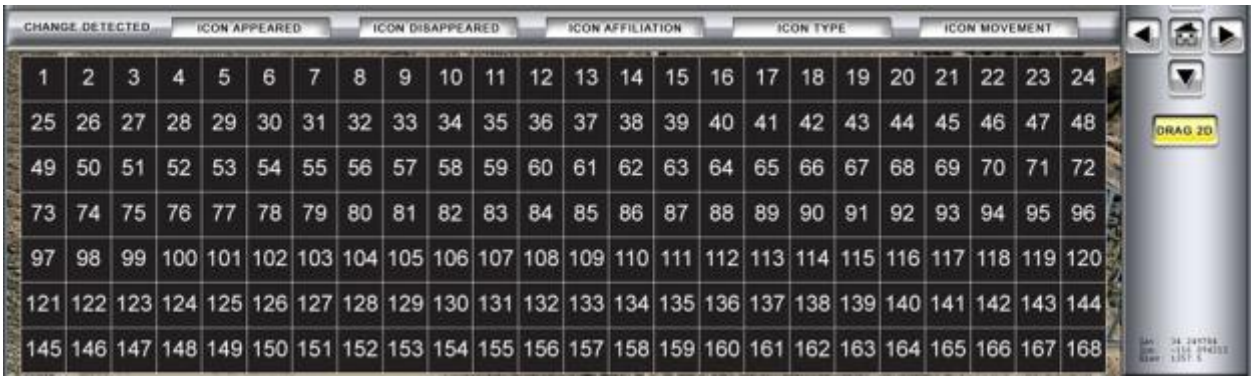
There are five event types:

- Move: move icon to a different position
- Unit: change icon into another icon (e.g. from Neutral Infantry to Friendly Tank)
- Affiliation: change affiliation of icon only (e.g. Unknown to Hostile
- Appear: icon appears at specified location
- Disappear: specified icon disappears

5.12.3 CHANGE DETECTION SCRIPTS: CSV

There are two kinds of Change Detection scripts: a CSV file (Comma Separated Value file, which can be generated by Excel or another spreadsheet program), or an XML file.

CSV scripts define the location of icons on the main map, based on a grid shown below.



This grid is persistent regardless of if the map is moved to a different position or not (anything assigned to 1 will always be in the upper left corner), so it is recommended that CSV scripts are used when movement of the main map can be disabled.

Open “tutorial_csv_mapscript.csv”, located in tutorial_10/scripts/mapicons/

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Example Change Detection																			
Event Number	Event Type	Original Location	New Location	Original Icon	New Icon	Simulation Time (min.second)	Timeout Time (min.second)	34	35	36	40	82	85	88	106	112	132	134	135
1	None	0	0	0	0	0	0	1	0	5	6	11	0	0	0	3	1	1	0
2	Move	0	1	1	1	0.05	0.05	0	1	5	6	11	0	0	0	3	1	1	0
3	Appear	106	106	0	2	0.1	0.05	0	1	5	6	11	0	0	2	3	1	1	0
4	Affiliation	106	106	2	3	0.15	0.05	0	1	5	6	11	0	0	3	3	1	1	0
5	Appear	135	135	0	10	0.2	0.05	0	1	5	6	11	0	0	3	3	1	1	10
6	Disappear	36	36	5	0	0.25	0.05	0	1	0	6	11	0	0	3	3	1	1	10
7	Unit	35	35	1	6	0.3	0.05	0	6	0	6	11	0	0	3	3	1	1	10
8	Move	35	36	6	6	0.35	0.05	0	0	6	6	11	0	0	3	3	1	1	10
9	Move	82	85	11	11	0.4	0.05	0	0	6	6	0	11	0	3	3	1	1	10
10	Appear	34	34	0	3	0.45	0.05	3	0	6	6	0	11	0	3	3	1	1	10

The sections of note are:

- Name: The first row lists a unique name for the script to be identified by
- Event Number: This column lists the events by number
- Event Type: what the icon should do at this event
- Original/New Location: Identifies where the icon is and where it should be moved if the event is a move type
- Original/New Icon: Identifies which type the icon is (based on numbering in above chart), and which Icon it should change to
- Simulation Time: Time event occurs
- Timeout Time: Time after Simulation Time that the participant can make a response to the event

Each number in the second row corresponds to which location on the grid the icons below should be.

5.12.4 ICON SCRIPT GENERATOR

CSV files can be generated randomly by running an xml file through a program. Open the file “iconscript-generator-tutorial.xml”, located in bin/settings/tools/

```
<?xml version="1.0" standalone="yes" ?>
<MapScripting>
    <!-- Define Unit types and icon files. -->
    <Icons>
        <Unit>
            <Image id="1" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.arminf/friend.png" />
            <Image id="2" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.arminf/hostile.png" />
            <Image id="3" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.arminf/neutral.png" />
            <Image id="4" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.arminf/unknown.png" />
        </Unit>
        <Unit>
            <Image id="5" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.tank.mdm/friend.png" />
            <Image id="6" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.tank.mdm/hostile.png" />
            <Image id="7" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.tank.mdm/neutral.png" />
            <Image id="8" file="../data/models/mil.std.2525b/war.grdtrk.eqt.grdveh.armd.tank.mdm/unknown.png" />
        </Unit>
        <Unit>
            <Image id="9" file="../data/models/mil.std.2525b/war.grdtrk.unt.cbt.arm.whd/friend.png" />
            <Image id="10" file="../data/models/mil.std.2525b/war.grdtrk.unt.cbt.arm.whd/hostile.png" />
            <Image id="11" file="../data/models/mil.std.2525b/war.grdtrk.unt.cbt.arm.whd/neutral.png" />
            <Image id="12" file="../data/models/mil.std.2525b/war.grdtrk.unt.cbt.arm.whd/unknown.png" />
        </Unit>
    </Icons>
    <!-- Define area of grid that can be used for script generation. -->
    <Cells start_col="6" end_col="19" start_row="1" end_row="7" />
    <!-- Number of blocks an icon can move. -->
    <MoveDistance>1</MoveDistance>
    <!-- How many icons should be on the screen at startup (initial state) -->
    <InitialIconCount>24</InitialIconCount>
    <!-- Define blocks of events, this allows you to
         segment different time periods with different event rates and
         event types. -->
    <EventBlock shuffle="true">
        <!-- An event has a type, and a delay since the previous (or until next
             event). The values for type are:
```

The icon types that will be used are defined at the top section Icons. The section Cells defines where on the map view icons will randomly appear, based on the grid above.

```
<EventBlock shuffle="true">
    <!-- An event has a type, and a delay since the previous (or until next
        event). The values for type are:
        move - Move an icon on the screen.
        appear - Have a new icon appear.
        disappear - Remove an icon from the map/screen.
        unit - Change the unit type from the map/screen.
        affiliation - Change the affiliation type of the icon (e.g. unknown->friendly)

        The delay value is in seconds. Some event types have extra parameters, see examples
        below for any additional options. -->
    <Event type="move" delay="10"/>
    <Event type="move" delay="10"/>
    <Event type="move" delay="15"/>
    <!-- If any is 1, then any affiliation type can appear, if 0, then only
        unknown affiliation will appear. -->
    <Event type="appear" delay="10" any="1"/>
    <Event type="appear" delay="10" any="1"/>
    <Event type="appear" delay="15" any="1"/>
    <!-- If any is 1, then any affiliation type can disappear, if 0, then
        unknown affiliation will have lowest probability of being removed. -->
    <Event type="disappear" delay="10" any="1"/>
    <Event type="disappear" delay="10" any="1"/>
    <Event type="disappear" delay="15" any="1"/>
</EventBlock>

<!-- This block, has 1 event, but will increase number of elements
    on the screen. -->
<EventBlock shuffle="false">
    <!-- If any is 1, then any affiliation type can appear, if 0, then only
        unknown affiliation will appear. -->
    <Event type="disappear" delay="10" any="1" num="17"/>
</EventBlock>
```

The most important segments of note are the Event Block sections, which contain define an order for both random and planned events to occur. Things of note are:

- The blocks themselves occur in order
- For each block labeled shuffle="true", the events contained will occur in a random order; otherwise the events occur in order
- For events where any="0", events will only occur on a random icon of "Unknown" affiliation (unless there are none); otherwise, event occurs on any random icon

Open "generate_scripts.bat", located in tutorial_10/bats, with Notepad++.

```
nerator.exe settings/tools/iconscript-generator-tutorial.xml "scenarios\tutorial\tutorial_10\scripts\mapicons\tutorial_csv_generated.csv"
```

The directory on the left is the location of the icon generator file to be used, and the right directory is the location where the resulting file will go (as well as the name of that file). Run the bat file. There will now be another csv file similar in structure to the other mapscript.

5.12.5 CHANGE DETECTION SCRIPTS: XML

The major difference between and xml script and a csv script is that the position of icons can be defined by latitude/longitude and remain at that position regardless of map movement. The only disadvantages are that they must be constructed manually, and the possibility that the icon won't be visible depending on the map position.

Open "tutorial_xml_mapscript.xml", located in tutorial_10/scripts/mapicons/

```
<?xml version="1.0" standalone="yes" ?>
<!-- This is an example XML Script file for generating Situational Awareness Map Events
<MapIconScript>
    <!-- File name will include participant number, date/time/scenario. and is unique.
    <!-- The following is a way to populate the SA Map with icons
         at the start of a scenario.-->
    <!-- See icons.xml models file for names of icons available. -->
    <InitialSetup>
        <Icon>
            <Model state="Neutral">WAR.GRDTRK.EQT.GRDVEH.ARMD.TANK.MDM ICON</Model>
            <Actor>A</Actor>
            <TriggerTime>0.0</TriggerTime>
            <TimeOutTime>.15</TimeOutTime>
            <Position latitude="40.328956" longitude="-79.422024"></Position>
        </Icon>
        <Icon>
            <Model state="Hostile">WAR.GRDTRK.EQT.GRDVEH.ARMD.TANK.MDM ICON</Model>
            <Actor>B</Actor>
            <Position latitude="40.325991" longitude="-79.420089"></Position>
        </Icon>
```

The script is divided into two sections:

- InitialSetup: Establishes where icons are at start
  - Model/state: Which type of icon and which state it should be in
  - Actor: an ID (usually a capital letter) given to the icon
  - TriggerTime (only in first): time all Icons appear on map
  - TimeOutTime (only in first): Time after the start of an event that the participant can make a response to the event
  - Position: latitude/longitude for position of icon
- Events (shown below): Lists events that will occur during scenario
  - Type: lists type of event to perform
  - TriggerTime: time after startup event
  - TimeOutTime: time after TriggerTime that participant can make a response to the event
  - Actor: which actor event is performed on
  - Position: latitude/longitude for position of icon after event

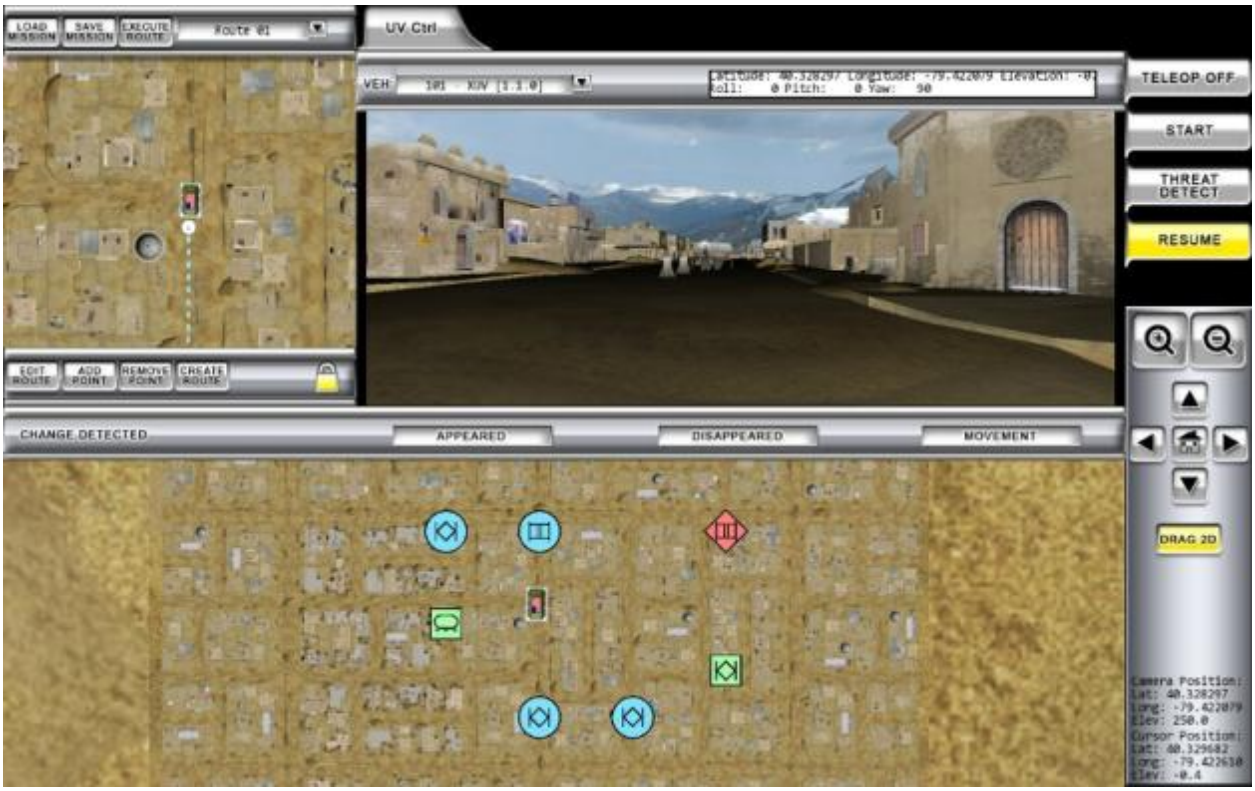
```
<Event type="move">
    <TriggerTime>.55</TriggerTime>
    <TimeOutTime>.10</TimeOutTime>
    <Actor>B</Actor>
    <Position latitude="40.325991" longitude="-79.420089"></Position>
</Event>
<Event type="disappeared">
    <TriggerTime>1.05</TriggerTime>
    <TimeOutTime>.10</TimeOutTime>
    <Actor>D</Actor>
</Event>
<Event type="appeared">
    <TriggerTime>1.15</TriggerTime>
    <TimeOutTime>.10</TimeOutTime>
    <Actor>D</Actor>
</Event>
<Event type="move">
    <TriggerTime>1.25</TriggerTime>
    <TimeOutTime>.10</TimeOutTime>
    <Actor>F</Actor>
    <Position latitude="440.325997" longitude="-79.423053"></Position
</Event>
```

5.12.6 RUNNING SCRIPT

Open Tutorial 10's OCU file.

```
<Conditions>
    <Condition id="0" name="GMET Example Standard">
        <XmlFiles>
            <AudioScript    loop="0" file="scripts/audio/standard.xml" />
            <BlankScreenScript    loop="0" logging="1" file="scripts/blankscreen/standard.xml" />
            <AdaptiveAutomationScript file="scripts/adaptiveauto/adaptiveauto_example.xml" />
            <MapIconScript  loop="0" logging="1" file="scripts/mapicons/tutorial_xml_mapscript.xml" />
        </XmlFiles>
```

Input a line similar to the MapIconScript above. The directory can be changed to any csv file or an xml file in the mapicons folder. Now, when the ocu and vehicle run, icons will appear on the map and events will occur involving the icons. Below is a screenshot of the "tutorial_csv_mapscript.csv" being utilized.

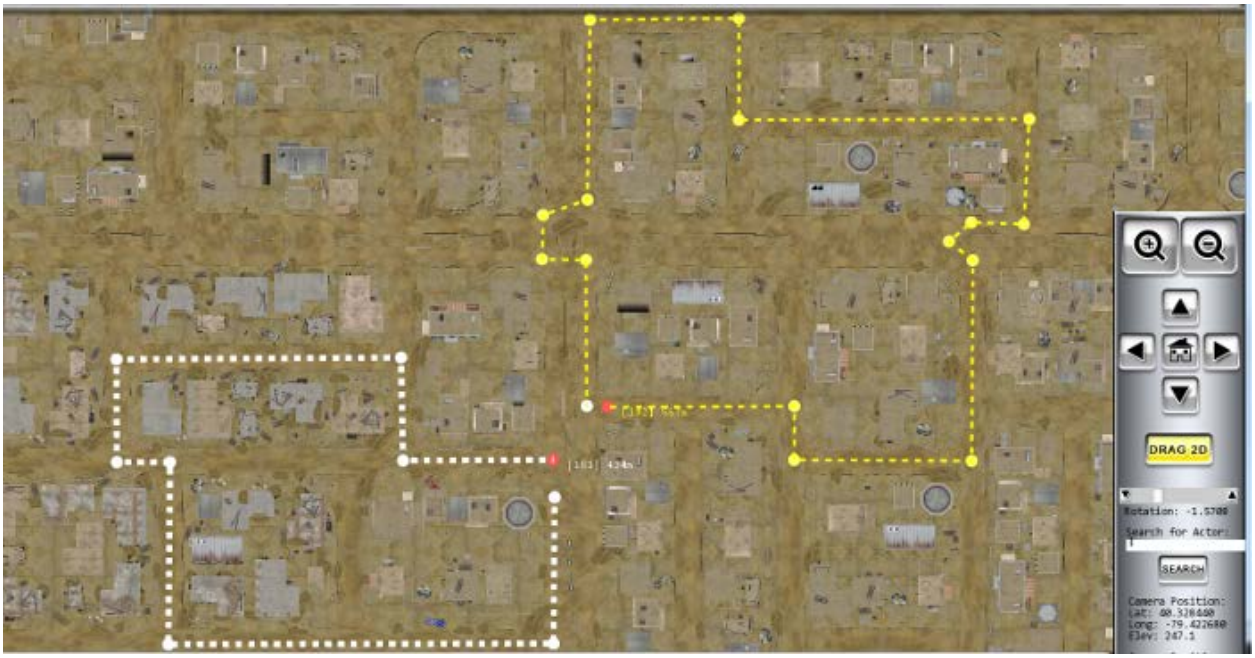


5.13 TUTORIAL 11: MULTIPLE ROUTES/VEHICLES IN A SCENARIO

Open the "tutorial_gmet_editor.bat" in tutorial_11/bats, and load the mission standard_short.xml. Add and shift points until the route looks like the image below.

Create New Route in loaded mission

- Click Create Route
- Name the mission (ex.: Route 02, since Route 01 is the name of the existing route), then click Create Route
- The Route Dropdown displays the current route being edited; points can now be added to the user's liking
- Create a Route that looks like the yellow one in the image below. Note the curved areas to maneuver around road medians
- Save the route as "standard_multi_route.xml"



5.13.1 ESTABLISHING A SECOND VEHICLE

Two files need to be changed to accommodate another vehicle: the platform file and the mission file. Look at tutorial_11/platforms/standard.xml.

- If there isn't already a block of "Platform" text for a second vehicle, it can be made by copying the text from "<Platform id=…" to "…</Platform>", then pasting it underneath.
- Change the Platform ID to a different number for the other vehicle, such as 102 instead of 101.
- The InitialPose data should be different for the vehicle, corresponding with the route's start
    - An easy way to determine these coordinates and the orientation of the vehicle is placing an actor unit at the location, saving it to the actor file, then looking at the actor xml file for the actor's coordinates
    - This actor text can then be deleted to remove the actor
- Make sure both blocks reference the proper file in the "ActorsFile" section, usually the same file

Open tutorial_11/missions/standard_multi_route.xml.

- Note that there are two "Mission" text blocks; the one with more points is the second route made
- Change this block's "assignedto" number to the second vehicle's number

Also, there must be a second .bat simulator file for the second vehicle, already provided in tutorial_11 as xuv_2.bat. Since this is also an XUV, this will be exactly the same as the xuv.bat, except it references "102" instead of "101".

5.13.2 AN OCU FOR MULTIPLE CAMERA VIEWS

- Open tutorial_11/bats/ocu_multi.bat; Note that this is a variant of the Thumbnail Display Gauges OCU, the second multiple vehicle display example in Section 3.1.3

- Open "xuv.bat" and "xuv_2.bat" in the same directory
- Note that there is a file "all_vehicles.bat", which opens both vehicle sims with one file; this can be opened instead of the above xuv files

- Also note the syntax of the file: each line (even with many vehicles) starts with “start cmd /c”, then the line normally found in the vehicles .bat file; the exception is the last line, which remains unchanged

```
1  cd ..
2  cd ..
3  cd ..
4  cd ..
5  start cmd /c xuv.exe scenarios/tutorial/tutorial_11/platforms/standard.xml 101
6  xuv.exe scenarios/tutorial/tutorial_11/platforms/standard.xml 102
7
```

- Look at the OCU again, and note that the two vehicle displays have appeared
- Click on one to expand it onto the larger window above; the currently selected display will be highlighted

- Start works the same as in the other OCU, except it starts both vehicles on their path