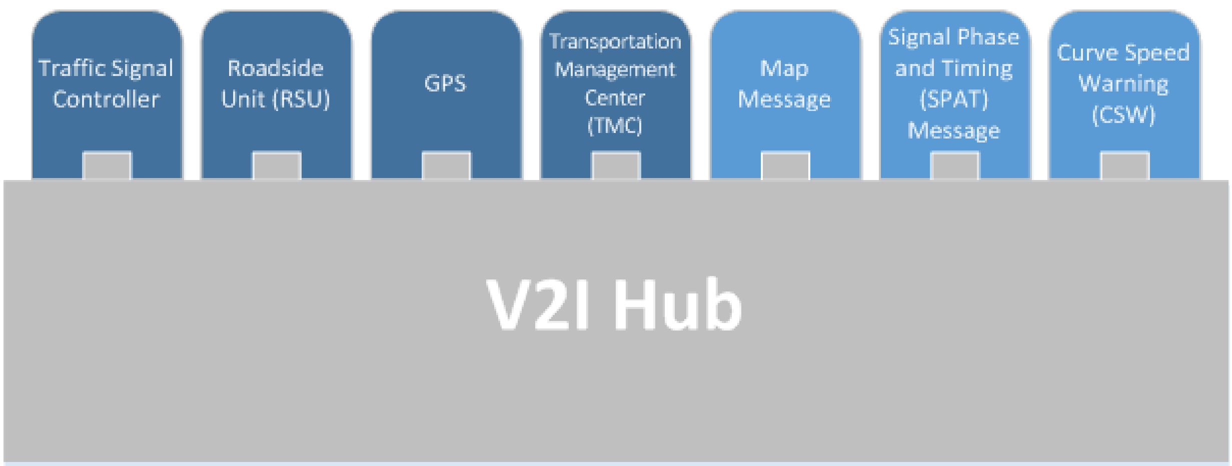
V2I Hub

Vehicle Simulation Tool User Guide

www.its.dot.gov/index.htm

Final Report – July 3, 2018
FHWA-JPO-XXX



U.S. Department of Transportation

Produced by Battelle Memorial Institute under DTFH61-12-D-00040
U.S. Department of Transportation
Office of the Assistant Secretary for Research and Technology
Joint Program Office

Notice

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA-JPO-XXX		

4. Title and Subtitle	5. Report Date
V2I Hub Vehicle Simulation Tool User Guide	July 3, 2018
	6. Performing Organization Code Battelle
7. Author(s) Anthony Polinori, Ben Paselsky	8. Performing Organization Report No.
9. Performing Organization Name and Address Battelle 505 King Ave Columbus, OH 43201-2693	10. Work Unit No. (TRAIS)
	11. Contract or Grant No. DTFH61-12-D-00040
12. Sponsoring Agency Name and Address Federal Highway Administration 1200 New Jersey Avenue, S.E. Washington, DC 20590	13. Type of Report and Period Covered Final
	14. Sponsoring Agency Code FHWA

15. Supplementary Notes

16. Abstract

Connected Vehicle technologies help reduce the number of driving related injuries and fatalities by allowing road users to be aware of potential dangerous situations on the road. There are two main types of Connected Vehicles communications, vehicle-to-infrastructure and vehicle-to-vehicle. Vehicle-to-infrastructure communication takes place between vehicles and deployed roadside communication devices, which capture vehicle generated data while providing information pertaining to safety, mobility, and environmental conditions.

This user guide is intended for Information Technology staff of State and local Departments of Transportation, Metropolitan Planning Organizations or other agencies testing or deploying V2I Hub infrastructure. This guide describes the hardware and software configurations required and provides step-by-step instructions for creating scenarios or visually playing back previously recorded trip data using the V2I Hub Vehicle Simulation Tool.

17. Keywords	18. Distribution Statement
V2I, V2I Reference Implementation, V2I Hub, testing, diagnostics, simulation, deployment, simulation, Connected Vehicle, CV	Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	45	

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

Table of Contents

List of Tables

List of Figures

Executive Summary

The Vehicle-to-Infrastructure (V2I) Hub Vehicle Simulation Tool User Guide was developed by Battelle on behalf of the Federal Highway Administration (FHWA) for information technology staff of State and local Departments of Transportation (DOT) Metropolitan Planning Organizations or other agencies testing or deploying V2I Hub infrastructure. The V2I Hub is a connectivity platform developed to be deployed at signalized intersections, and other infrastructure locations, with the intention of making roadways safer and smarter by reducing accidents and providing informational alerts to mobile users.

The V2I Hub Vehicle Simulation Tool is a diagnostic software program designed to assist with CV application testing and development, and the deployment of CV technology. This simulation tool operates in conjunction with the V2I Hub software. The tool allows simulated scenarios to be created with customizable parameters and features. It also enables previously recorded trip data to be played back in a visual format and for additional vehicles to be introduced to the previously recorded trip data, to create a new scenario.

This guide describes the hardware and software configurations required and provides step-by-step instructions for creating scenarios with the V2I Hub Vehicle Simulation software. The V2I Hub Vehicle Simulation Tool was created in parallel to support V2I Hub deployments as part of the V2I Reference Implementation project.

Chapter 1. Background

The V2I Hub is part of U.S. DOT's V2I Program, developed to support jurisdictions deploying vehicle to infrastructure (V2I) technology by reducing integration issues and enabling use of their existing transportation management hardware and systems. The V2I Hub is a Connected Vehicle (CV) software platform developed to be deployed at signalized intersections, and other infrastructure locations with the intention of making roadways safer and smarter by reducing accidents and providing informational alerts to drivers. The V2I Hub software uses plugins to translate messages between different devices and run CV applications on roadside equipment.

The V2I Hub software and V2I Hub Vehicle Simulation Tool can be found in the U.S. DOT's Open Source Application Development Portal (OSADP) repository along with supporting documentation and other reference materials at https://www.itsforge.net. It should be noted that the V2I Hub Vehicle Simulation Tool is not required for deploying the V2I Hub. It is a complementary tool for diagnostic testing and software development used to support the deployment of the V2I Hub.

This user guide was written for any engineer or individual with an Intelligent Transportation Systems or CV background help with testing or deploying the V2I Hub. This guide describes the hardware and software configurations required and provides step-by-step instructions for creating scenarios and displaying previously recorded trip data with the simulation software.

The V2I Hub Vehicle Simulation Tool is commonly referred to as V2I Hub Vehicle Simulation software simulation software, and simulation tool throughout this user guide.

Chapter 2. Vehicle Simulation Tool Overview

Simulation Tool Functions and Benefits

The V2I Hub Vehicle Simulation Tool is a diagnostic software program designed to assist with CV application testing and development, and the deployment of CV technology. This simulation tool operates in conjunction with the V2I Hub software described above in Chapter 1. The tool allows simulated scenarios to be created with customizable parameters and features. It also enables previously recorded trip data to be played back in a visual format and for additional vehicles to be introduced to the previously recorded trip data, to create a new scenario.

This simulation software serves as a development platform for software engineers to test CV system or subsystem, software code and updates in a controlled environment.

Software testing in a simulated environment offers several benefits:

- Testing can be performed more quickly and efficiently.
- It is a more cost-effective approach than having to outfit the infrastructure or a vehicle with equipment for testing.
- Simulation parameters and features can easily be changed, prior to running additional tests.
- Controlled environment testing allows more granular traceability with software changes.
- Less corrective maintenance and/or system modifications will be required in the field.
- Improved system reliability through additional testing that was not previously conducted due to testing inefficiencies, lack of resources, or schedule conflicts.

The simulation tool can be used to simulate and test an entire CV system including both roadside and in-vehicle subsystems shown in Figure 1. Simulation can also be performed at a more granular level where each subsystem or individual components of that subsystem can be simulated and tested. The various hardware configurations and software testing that can be performed is explained in the subsequent chapter.

Chapter 3. Hardware Setup and Configuration

This chapter details the hardware setup and configuration for the V2I Hub Vehicle Simulation Tool. As discussed in Chapter 1, the simulation software can be utilized for testing across an entire system or a specific subsystem, in this case the roadside and in-vehicle subsystems. Independent of either configuration, a Personal Computer (PC) with Windows 7, 8, or 10 with Microsoft .Net Framework 4.5.2 is required to run the V2I Hub Vehicle Simulation Tool software as shown in Figure 1.

System Level Simulation Testing

Figure 1 illustrates the hardware setup that is required to conduct system level simulation testing. The roadside and in-vehicle subsystems are required, and the V2I Hub Vehicle Simulation software must reside on a PC. CPUs for the roadside and in-vehicle subsystems utilize the V2I Hub software. After setup and configuration is complete, it is important to verify that Dedicated Short-Range Communication (DSRC) messages are being sent from the Roadside Unit (RSU) and received by On-Board Unit (OBU) in the in-vehicle subsystem prior to running the simulation software. A description of each subsystem is discussed in detail below.

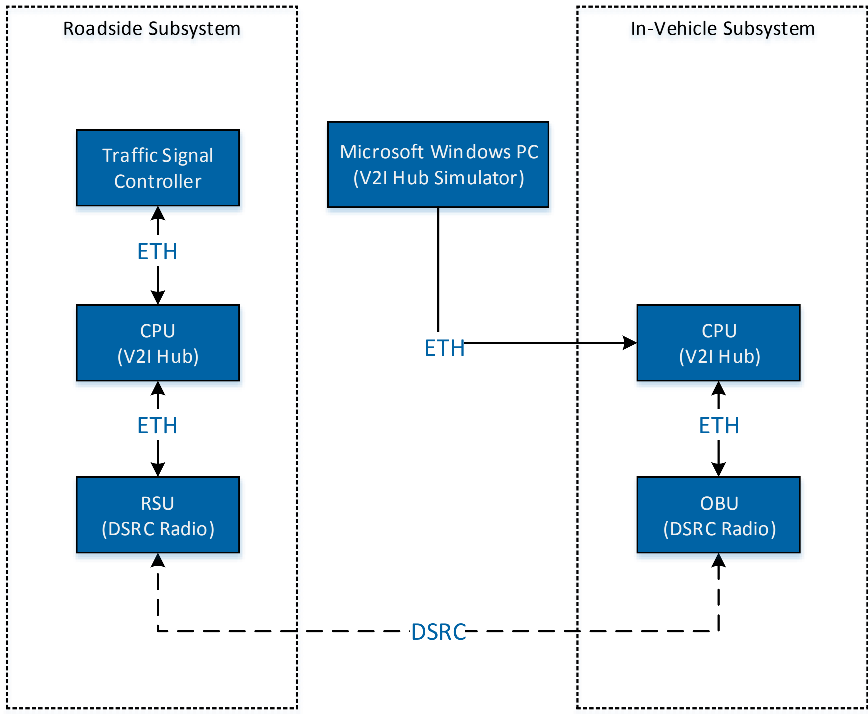


Figure 1. V2I Hub Vehicle Simulation Tool Hardware Setup for System Level Simulation Testing

Roadside Subsystem Description

The roadside subsystem is comprised of a Traffic Signal Controller, V2I Hub, and RSU as shown in Figure 2. The V2I Hub is connected to the Traffic Signal Controller via an Ethernet cable. This communication channel is bi-directional, which allows for information to be sent and received between each component. The Traffic Signal Controller transmits National Transportation Communications for Intelligent Transportation System Protocol (NTCIP) 1202 v03-based messages containing the Signal Phase and Timing (SPaT) stream via User Data Protocol (UDP) messages requested by the V2I Hub at the roadside. Refer to Chapter 3. Signal Phase and Timing in the V2I Hub Plugins document for additional information on SPaT configuration.

The V2I Hub then transmits the SPaT data to the RSU through its Ethernet connection. The V2I Hub also generates the MAP messages and transmits them to the RSU. The MAP messages contain physical intersection geometry, which is utilized by the in-vehicle V2I as a geographical point of reference. Once received by the RSU, the MAP and SPaT data is then encoded into a J2735 2016 message and transmitted to the OBU in the in-vehicle subsystem.

In-Vehicle Subsystem Description

Similar to the roadside subsystem, the in-vehicle subsystem is also made up of the V2I Hub and OBU. Although not specifically part of the in-vehicle subsystem, V2I Hub Vehicle Simulation Tool interfaces directly with the in-vehicle V2I Hub via an Ethernet connection. This communication channel is unidirectional with the simulation software transmitting but not receiving information. Depending on the scenario, the simulation software generates either Basic Safety Messages (BSMs) or simulated BSMs, and Signal Request Messages (SRMs), which are received by the Message Receiver Plugin in the V2I Hub. The J2735 message signing is performed by the OBU prior to being broadcast. The V2I Hub messages sent to the OBU for broadcast contain a flag that denotes whether the message should be signed. The V2I Hub plugins that send messages to the OBU or RSU for broadcast have a configuration setting that determines whether the signing flag should be set. The messages are then processed by the DSRC Message Manager Plugin also within the V2I Hub.

Likewise, in the in-vehicle subsystem, the communication channel between the V2I Hub and OBU is bi-directional. The V2I Hub transmits the simulated BSMs or BSMs and SRMs to the OBU, where the messages are encoded into a J2735 2016 message and transmitted to the RSU.

The roadside and in-vehicle subsystem simulation testing configurations are described below.

Roadside Subsystem Simulation Testing

Figure 2 is representative of how the roadside subsystem should be configured to test roadside functionality independently of a functioning in-vehicle subsystem. The V2I Hub Vehicle Simulation software can be used to directly inject BSM and SRM messages into the roadside V2I Hub to test the functionality of applications that process these messages. For example, a roadside application that generates warning on a dynamic message sign when an approaching vehicle is traveling over the speed limit could be tested this way.

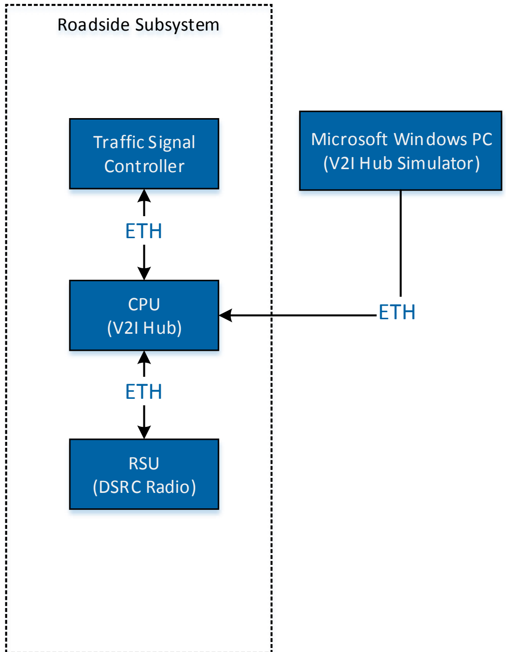


Figure 2. V2I Hub Hardware Setup Roadside Subsystem Simulation Testing

In-Vehicle Subsystem Simulation Testing

Figure 3 is representative of how the in-vehicle subsystem should be configured to test in-vehicle functionality independently of a functioning roadside subsystem. The V2I Hub Vehicle Simulation software can be used to directly inject BSM and SRM messages into the in-vehicle V2I Hub to test the functionality of applications that process these messages. For example, an in-vehicle application that warns the driver of an impending collision with another vehicle could be tested this way.

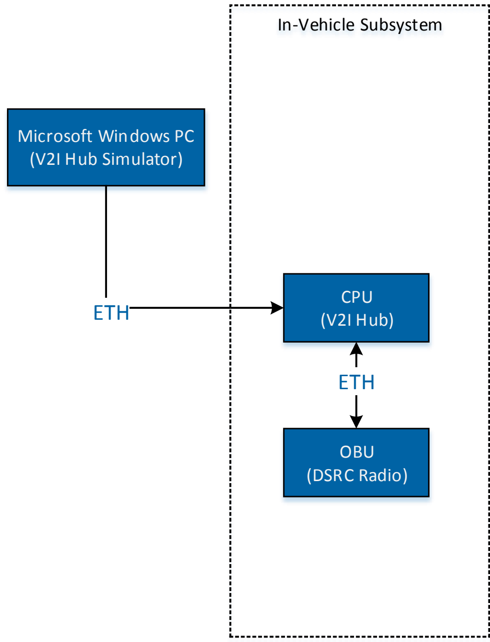


Figure 3. V2I Hub Hardware Setup In-Vehicle Subsystem Simulation Testing

Chapter 4. Plugin Configuration and File Formatting

The V2I Hub Vehicle Simulation Tool works in conjunction with the V2I Hub software as described in Chapter 1. In addition to the required plugins configured for the V2I Hub, the Message Receiver Plugin must be configured to meet the functional requirements of the simulation tool. Configuring this plugin is further discussed in detail in the subsequent section.

As mentioned in Chapter 1, the simulation tool can be utilized for two different purposes. The tool allows for custom simulations to be created from scratch without any previous data. It also enables previously recorded trip data to be visually displayed. To visually playback previously recorded trip data, this data must be formatted in a KML format recognizable by the simulation software. The KML File Formatting section below discusses this in further detail and provides an example of a KML format recognizable by the simulation software.

Message Receiver Plugin Configuration

The V2I Hub Vehicle Simulation software generates a stream of UDP messages in a custom format containing BSM and SRM simulated data, which is received by the Message Receiver Plugin. Once these messages are received, they are converted to J2735 2016 messages and injected into V2I Hub. The Message Receiver Plugin requires certain configuration parameters to be set to enable BSM, SRM, and/or location message injection into the V2I Hub. To configure these parameters, refer to the V2I Hub Plugin document found on the OSADP website.

KML File Formatting

For custom created scenarios using the V2I Hub Vehicle Simulation software, any file that is created will be saved in a custom JavaScript Object Notation (JSON) format. If the simulation tool is going to be utilized to playback a previously recorded trip, the file must be formatted in a recognizable Keyhole Markup Language (KML) format by the simulation tool. An example of the custom JSON file format is shown in Figure 4. The example demonstrates how the file would be written if there were two vehicles in the scenario with the first vehicle having two clones and the second vehicle not having a clone.

```
{
    "WaypointLists": {
        "56789": [{
                "VehicleID": 56789,
                "WaypointNumber": 0,
                "Speed_mph": 25.0,
                "Pause_seconds": 0.0,
                "Latitude": 39.987717836590868,
                "Longitude": -83.023282476598467,
                "Background": "#FF00FFFF"
            }, {
                "VehicleID": 56789,
                "WaypointNumber": 1,
                "Speed_mph": 25.0,
                "Pause_seconds": 0.0,
                "Latitude": 39.987673084219011,
                "Longitude": -83.021645239258291,
                "Background": "#FF00FFFF"
            }, {
                "VehicleID": 56789,
                "WaypointNumber": 2,
                "Speed_mph": 0.0,
                "Pause_seconds": 2.0,
                "Latitude": 39.987613414344267,
                "Longitude": -83.0201637607137,
                "Background": "#FF00FFFF"
            }
        ],
        "3456": [{
                "VehicleID": 3456,
                "WaypointNumber": 0,
                "Speed_mph": 25.0,
                "Pause_seconds": 0.0,
                "Latitude": 39.988634616012348,
                "Longitude": -83.0226946350576,
                "Background": "#FFFF7F50"
            }, {
                "VehicleID": 3456,
                "WaypointNumber": 1,
                "Speed_mph": 25.0,
                "Pause_seconds": 0.0,
                "Latitude": 39.988584512749306,
                "Longitude": -83.021115830842277,
                "Background": "#FFFF7F50"
            }, {
                "VehicleID": 3456,
                "WaypointNumber": 2,
                "Speed_mph": 25.0,
                "Pause_seconds": 0.0,
                "Latitude": 39.988140739386729,
                "Longitude": -83.021078462695186,
                "Background": "#FFFF7F50"
            }
        ]
    },
    "Colors": {
        "56789": "#FF00FFFF",
        "56790": "#FF00FFFF",
        "56791": "#FF00FFFF",
        "3456": "#FFFF7F50"
    },
    "Clones": {
        "56789": 2,
        "3456": 0
    },
    "CloneOffsets": {
        "56789": 5.0,
        "3456": 5.0
    },
    "Name": "test2",
    "Duration_seconds": 0.0,
    "NumberOfVehicles": 0
}
```

Figure 4. Custom JSON file format example

The KML file should contain a collection of Placemark elements in sequential order with respect to their TimeStamp element. The elements highlighted in red in Figure 5, are the minimum elements required. The elements in black are additional Placemark elements that are included in the KML file but are not required. Figure 5 is an example of a KML file for a single point. In any previously recorded trip, it is probable for many more points to be included; however, the emphasis on this example is on the format of the file. All required fields must be in the format as shown below. All floating-point numbers must contain a value before and after the decimal point. The KML file below is consistent with the Open Geospatial Consortium standards and is compliant with the KML 2.2.0 schema.

```
<?xml version="1.0" encoding="utf-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
        <Placemark>
                <name>Speed: 11.5 MPH</name>
                <description>
                        Time: 8/24/2017 10:24:58 AM
                        Heading: 87.780
                        VehicleId: 5
                        HDOP: 0.750
                        NumSatellites: 10
                        FixTypeId: 2
                        SignalQualityTypeId: 1
                </description>
                <TimeStamp>
                        <when>2017-08-24T10:24:58-
04:00</when>
                </TimeStamp>
                <Point>
                        <coordinates>-
81.7286376953125,41.491573333740234,0</coordinates>
                </Point>
        </Placemark>
</kml>
```

Figure 5. Example of a Formatted KML File Readable by the Simulation Software

The Placemark elements shown in Figure 5 are listed and defined below in Table 1.

Table 1. Placemark Element Descriptions

Placemark Element	Description
<name>	The name in the KML file is the travelling speed of the vehicle at the point where the vehicle transmits a BSM.
<description>	The Placemark description must include Heading and a VehicleId at minimum, but can also include the Time, HDOP, NumSatellites, FixTypeId, and SignalQualityTypeId.
<TimeStamp>	The timestamp includes the date and time of when the BSM was transmitted at that point.
<point>	The point value is the GPS coordinates of where the BSM was transmitted.

It is recommended that a program be used to translate the raw data collected from a previously recorded trip scenario to the format of the KML file shown in Figure 5.

Chapter 5. How to Create a Simulated Vehicle Scenario

This chapter describes how a simulated vehicle scenario can be created using the V2I Hub Vehicle Simulation Tool. As mentioned in Chapter 1, the tool allows simulated scenarios to be created and for previously recorded trip data to be played back in a visual format. Instructions for creating and modifying scenarios and playing back previously recorded trips are detailed below.

Setting up the Simulation

Creating a simulated vehicle scenario will require enabling and disabling features and defining scenario parameters. This section describes each of the features and parameters and their respective default values.

After the hardware has been set up and the Message Receiver Plugin has been configured, a simulation scenario can be created. Within the simulation tool, the scenario parameters must be defined, and the features should be enabled or disabled to fulfill needs of the scenario.

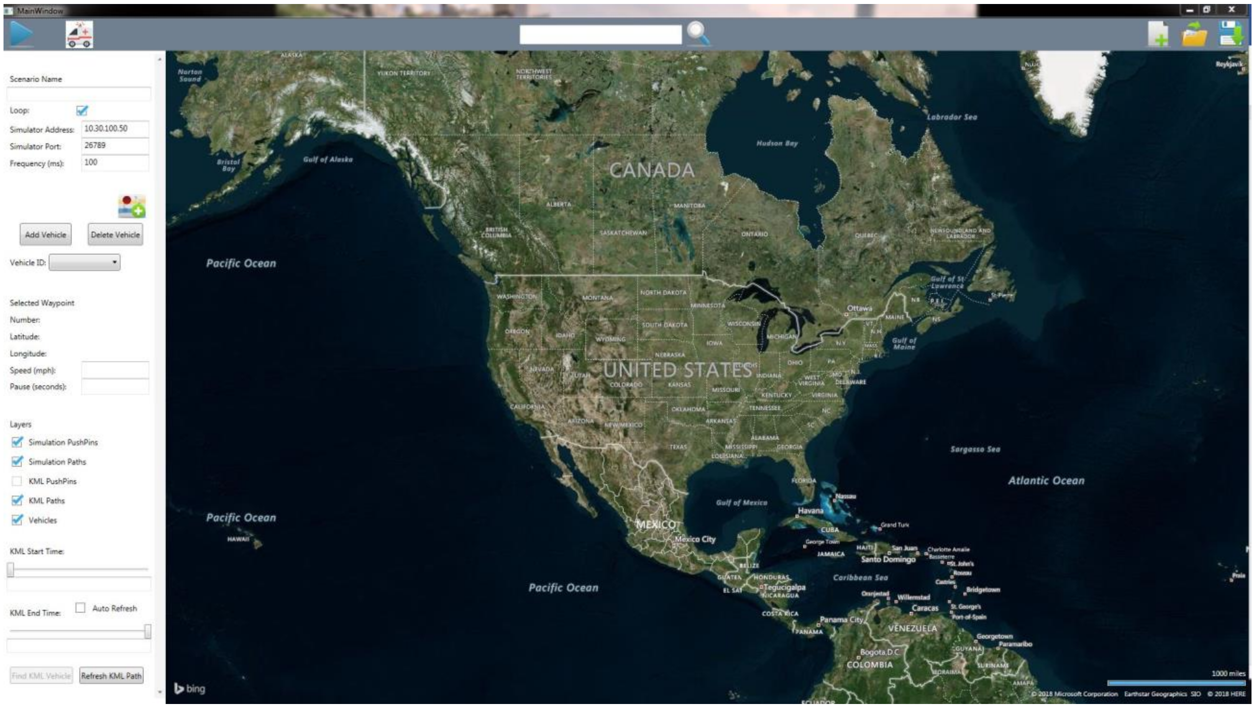


Figure 6. V2I Hub Vehicle Simulation Tool Homepage

Parameter Configuration

Constructing a new scenario requires several parameters to be defined as these values will be inputs into the scenario's design. A combination of simulation parameters and features are shown below in Figure 7. Tables 2 and 3 help to define each parameter and feature.

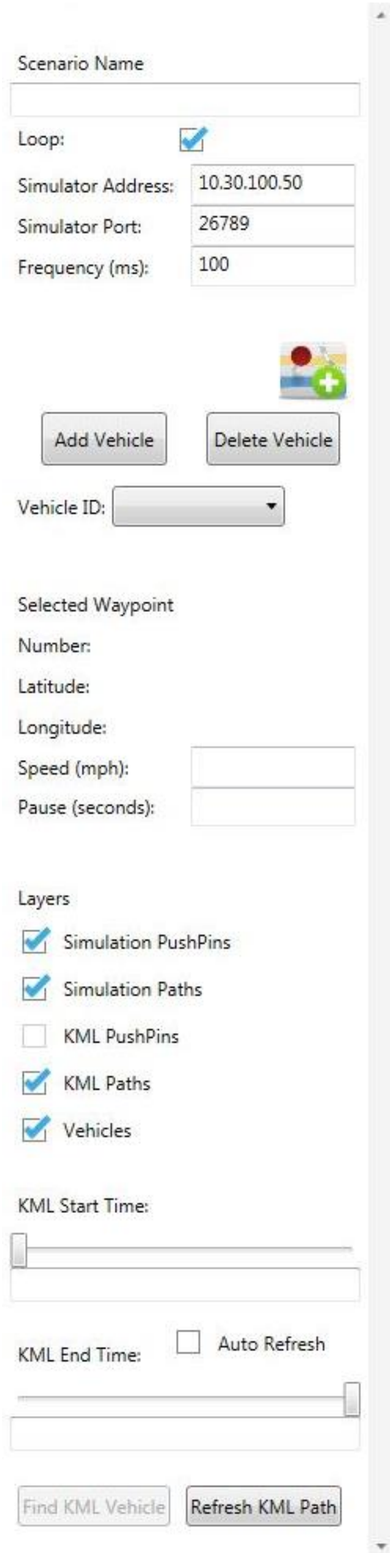


Figure 7. Parameters and Features Toolbar

Table 2. Parameter Descriptions

Parameter	Default Value	Description
Scenario Name	N/A	Name of scenario
Simulator Address	10.30.100.50	Internet Protocol (IP) address of Central Processing Unit (CPU) receiving the simulated BSMs
Simulator Port	26789	Port number from the CPU receiving the simulated BSMs
Frequency (ms)	100	Frequency in milliseconds of BSMs being transmitted from the vehicle
Vehicle ID	N/A	ID assigned to the vehicle, must be a number
Number of Clones	0	Number of additional vehicles travelling the same path
Clone Offset (seconds)	5.0	Duration of time that the new vehicle will trail the vehicle ahead
Speed (mph)	N/A	Vehicle travelling speed at the selected point
Pause (seconds)	N/A	Duration of time vehicle stops at the selected point
KML Start Time	N/A	Previously recorded trip start time, only enabled for KML based replay
KML End Time	N/A	Previously recorded trip end time, only enabled for KML based replay

Table 3 describes each of the features and their default values that are shown in Figure 7. The first three features including the Search, Play, and SRM are not shown in Figure 7, but are illustrated below in Figure 8. The enabling and disabling of the subsequent features are explained in Creating a New Scenario.

Figure 8. Search, Play, and Signal Request Message (SRM) features

Table 3. Feature Descriptions and Default Values

Feature	Default Value	Description
Search	Disabled	Searches for location entered
Play	Disabled	Begins playing the scenario
Signal Request Message	Disabled	Enables all vehicles in the scenario to transmit SRMs
Loop	Enabled	Enables the scenario to be played continuously
Add Vehicle	Disabled	Adds a vehicle to the scenario
Delete Vehicle	Disabled	Removes a vehicle from the scenario
Simulation PushPin(s) Layer	Enabled	Displays points where the simulated path was adjusted
Simulation Path(s) Layer	Enabled	Displays the simulated path of the vehicle(s)
KML PushPins	Disabled	Displays every point where a vehicle transmitted a BSM or SRM, it is recommended this feature is never enabled due to the amount of data displayed
KML Path(s) Layer	Enabled	Displays the path the vehicle travelled
Vehicle(s) Layer	Enabled	Displays the vehicle(s) or simulated vehicle(s)
Auto Refresh	Disabled	Enables the path to update in real-time as the KML Start Time and End Time are adjusted
Find KML Vehicle	Disabled	Locates the KML Vehicle when the vehicle cannot be seen on the map
Refresh KML Path	Disabled	Refreshes the KML Path after changes have been made to the KML Start Time and End Time

Creating a New Scenario

This section details how to create a new scenario from scratch using the tool, and how to load a KML file to visually playback previously recorded trip data.

To begin creating a new scenario, open the simulation software from the PC as shown in Figure 6. Assign the scenario a Scenario Name and save the file. To save the file, click the “floppy disk” save icon in the upper right-hand corner of the window. See Figure 9 below.

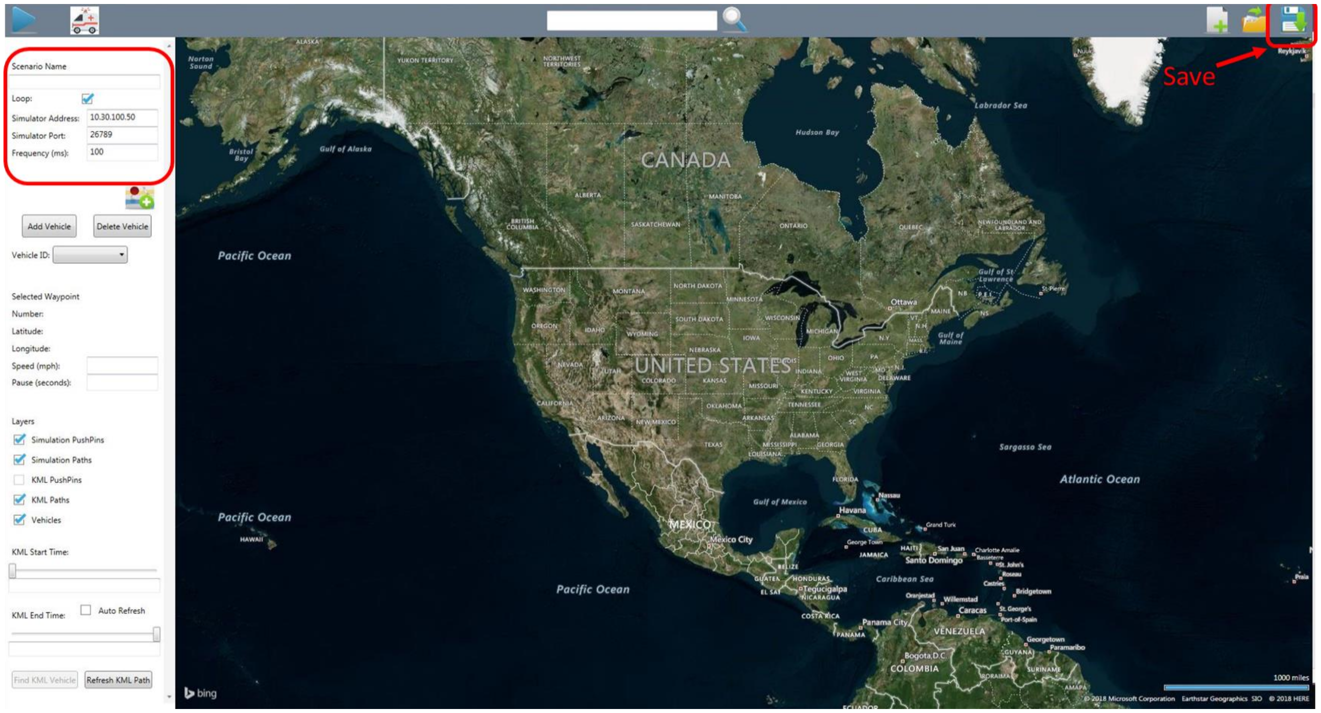


Figure 9. Assigning a Scenario Name and Saving the File

After the file has been saved, type the Simulator Address, Simulator Port, and Frequency in their respective fields (see Figure 9 above). The Simulator Address is the IP address of the machine transmitting the simulated BSMs and/or SRMs to the RSU, which is the OBU's IP address from the in-vehicle subsystem. The Simulator Port is the port of the in-vehicle subsystem's OBU that the BSMs and/or SRMs are being forwarded to. The frequency is referring to the frequency at which the DSRC messages will be broadcast. The default value is set to 100 ms, which is compliant with the IEEE 1609 standard for Wireless Access in Vehicular Environments.

The Loop feature's default state is set to be enabled, which will continuously repeat the scenario being played. This feature can be disabled by clicking the checkbox. See Figure 10 below.

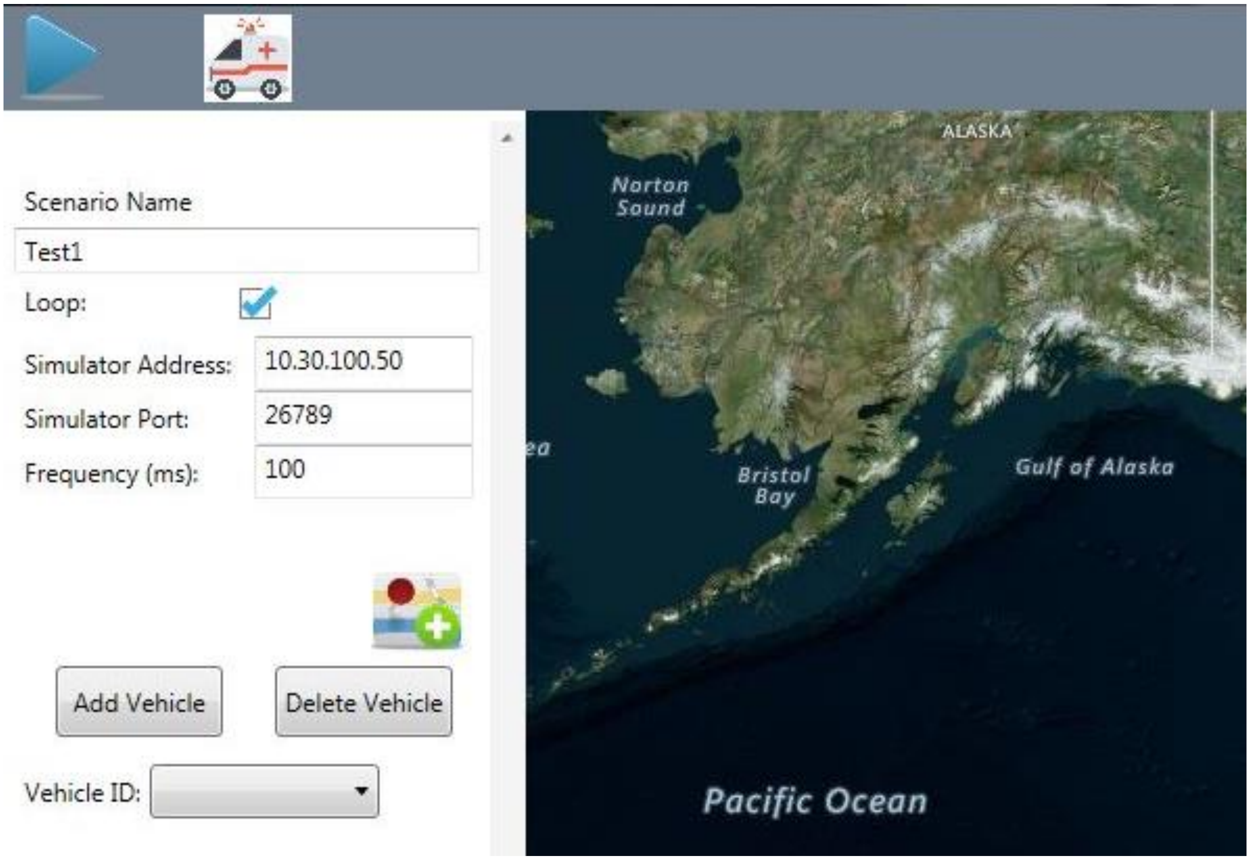


Figure 10. Connecting the PC with the In-Vehicle OBU

Choose a location where this scenario will take place. This can be done manually by selecting and dragging the map to the location of interest and scrolling up or down to zoom in and out of the map, or by using the search bar at the top of the window. Acceptable search terms include: addresses, GPS coordinates, cities, counties, states, and countries. See Figure 11 below.

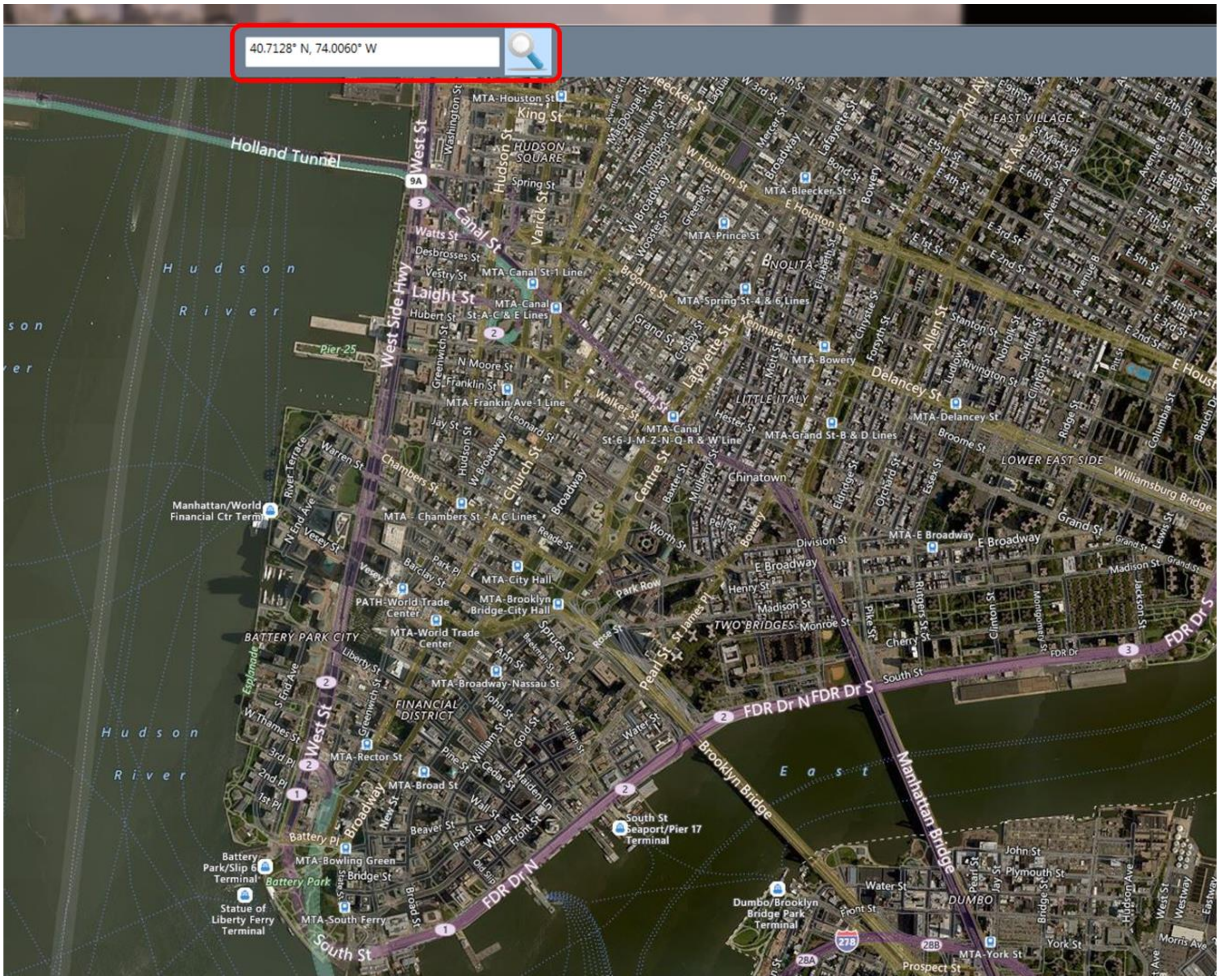


Figure 11. Search Results Using GPS Coordinates

If a new scenario is being created from a previously recorded trip, open the KML file by clicking the folder icon and selecting the file. This will display the path recorded on the map within the simulation tool. Prior to opening the KML file, it must be converted to the KML format defined in Chapter 4.

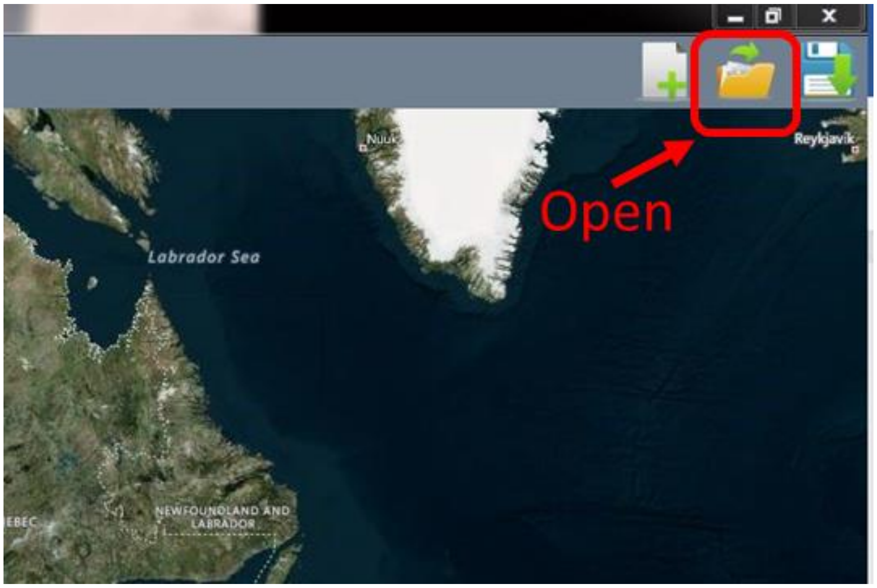


Figure 12. Loading a Previously Recorded Trip

After loading a previously recorded trip, the point of origin is indicated by the green marker, while the trip's destination point is designated by the red marker. An example of a previously recorded trip shown in the simulation tool is illustrated in Figure 13 below.

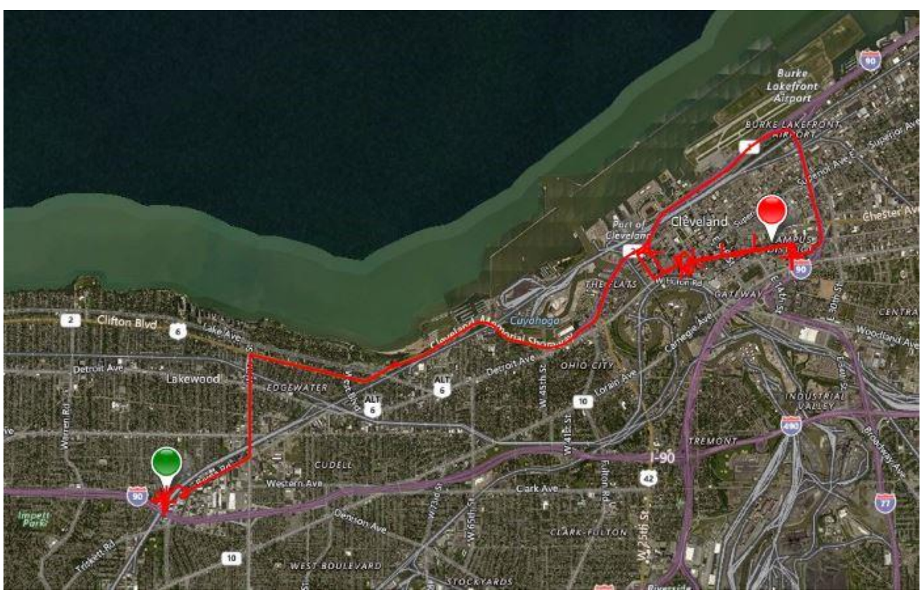


Figure 13. Previously Recorded Trip Data

After the general location has been selected for a scenario not using any previously recorded data, add a vehicle to the scenario by clicking the Add Vehicle button. Enter a Vehicle ID. This ID will be the unique identifier for the vehicle being added to the scenario. The ID is not visually displayed on the vehicle;

however, this ID is included in the BSMs or SRMs being transmitted from that vehicle. Enter the number of clones desired. Clones are replicas of the vehicle being added to the scenario. Each clone will have a different ID that is included in the BSMs it transmits. The clone vehicles will travel the same path drawn for the first vehicle created in the scenario. If a different path is desired, a new vehicle must be added to the scenario. Enter a value for the Clone Offset, which is the duration of time from which the cloned vehicle will enter the scenario after the first vehicle. See Figure 14 below.

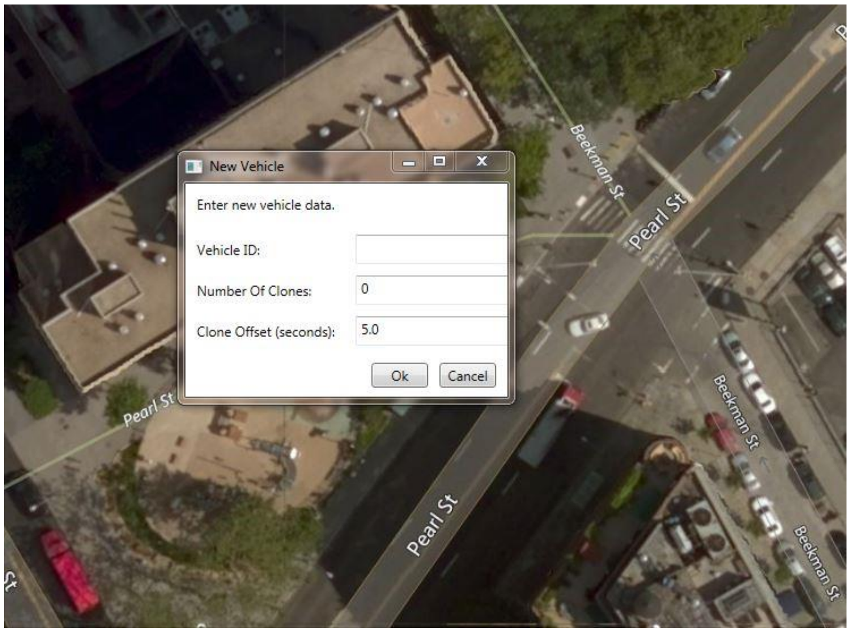


Figure 14. Adding a Vehicle to the Scenario

Once the vehicle parameters have been set, click Ok. The Vehicle ID dropdown box will be populated with the Vehicle ID it was assigned. A colored box will appear beside the Vehicle ID, which will represent the color of the vehicle, its path, and its waypoints.

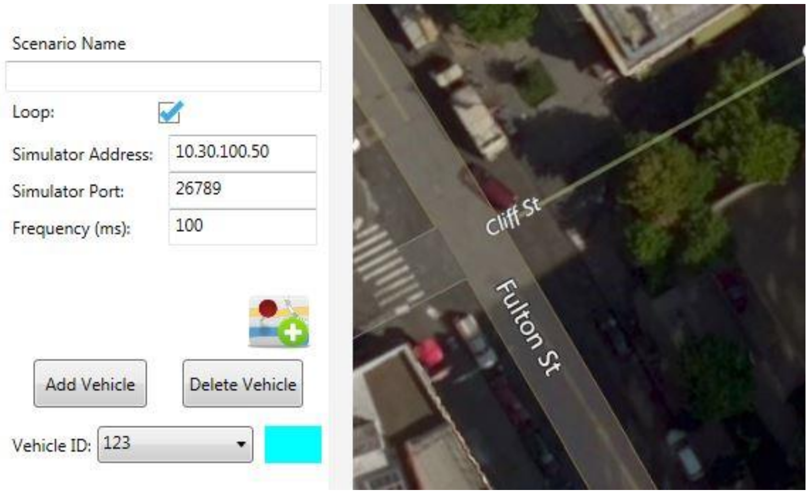


Figure 15. Vehicle ID and Color

Up to ten vehicles and their clones can be added to any scenario or previously recorded trip data following the same procedure described above. The Vehicle ID of the clones increases sequentially by one from the original Vehicle's ID. More than one vehicle can be added at this phase in creating the scenario or after creating the vehicle's path. After adding a vehicle, it will not appear until the path is drawn and the play icon is clicked.

To create the vehicle's path, first select the Add Points icon. See Figure 16 below.

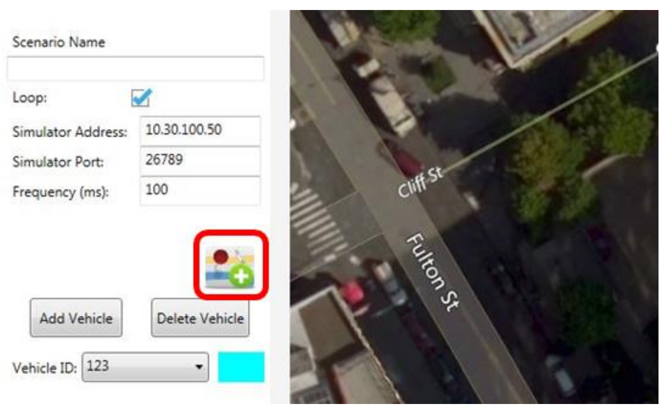


Figure 16. Creating the Path's Point of Origin

After selecting the Add Points icon, select the path's point of origin by double-clicking the desired location. After this point is established, create a path by adding waypoints in the path by double-clicking each location. Any number of waypoints can be included as part of the path. Each waypoint will be shown by a marker that is the same color of the vehicle that will be travelling that path. This is illustrated in Figure 17 below.

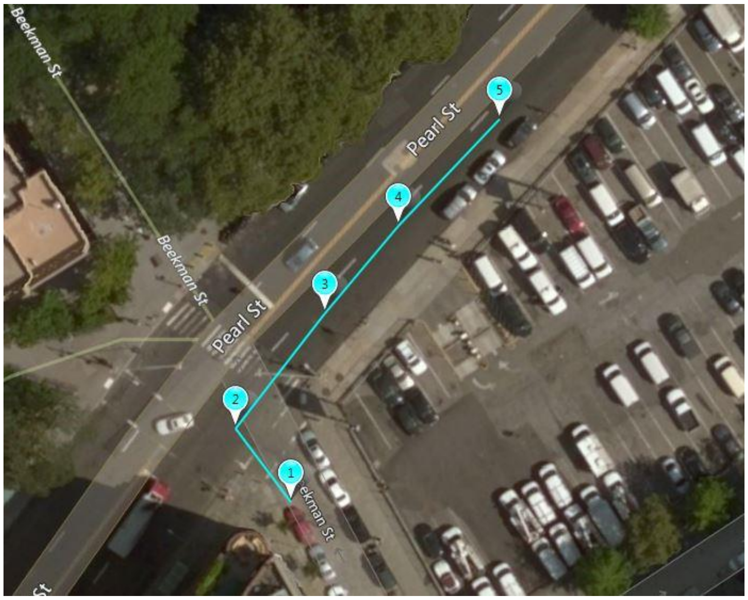


Figure 17. Vehicle Path Drawn with Markers

Once the path is finished being drawn, click the Add Points icon again to complete the path. This is shown in Figure 18. Paths and individual markers cannot be edited once created.

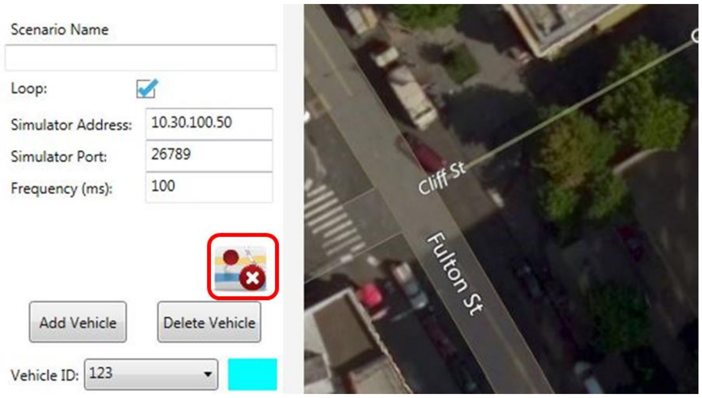


Figure 18. Finish Creating the Path

The vehicle can be deleted while waypoints are being added to the vehicle's path or after the path has been created. To delete the vehicle while adding waypoints to its path, click the Delete Vehicle button shown in Figure 19. To delete the vehicle after its' path has been drawn, select the Vehicle ID of the vehicle that is desired to be deleted, and then click Delete Vehicle. When a vehicle has been deleted, the vehicle's path will also be deleted.

Figure 19. Delete Vehicle Feature

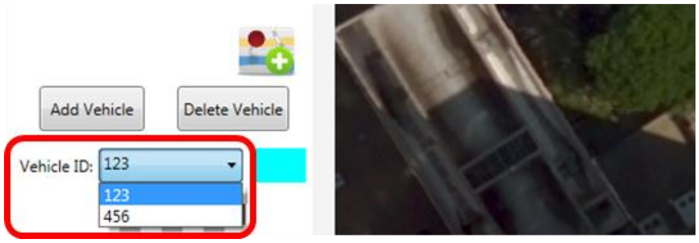


Figure 20. Deleting a Vehicle after its' Path as been Drawn

After the path has been drawn, the path parameters must be defined for each waypoint as shown in Figure 21. By selecting each waypoint, the properties of that waypoint including the waypoint number, latitude, longitude, speed, and pause will all be populated. The waypoint number, latitude, and longitude are non-configurable parameters. The speed and pause parameters default to 25mph and zero seconds respectively but are configurable to suit the needs of the scenario. This is shown in Figure 22. These parameters should be set for each waypoint in the path. The speed parameter is the speed at which the vehicle will be travelling when it reaches that waypoint. For example, if the speed for waypoint one is 25mph and the speed for waypoint two is set to 35mph, the vehicle's initial travelling speed will be 25mph and the vehicle will accelerate to reach 35mph by the time it arrives at waypoint two. The pause parameter is the duration of time that the vehicle pauses at the respective waypoint. For example, if the pause parameter is set for five seconds at waypoint two, when the vehicle reaches waypoint two, it will pause for five seconds and continue traversing its path.

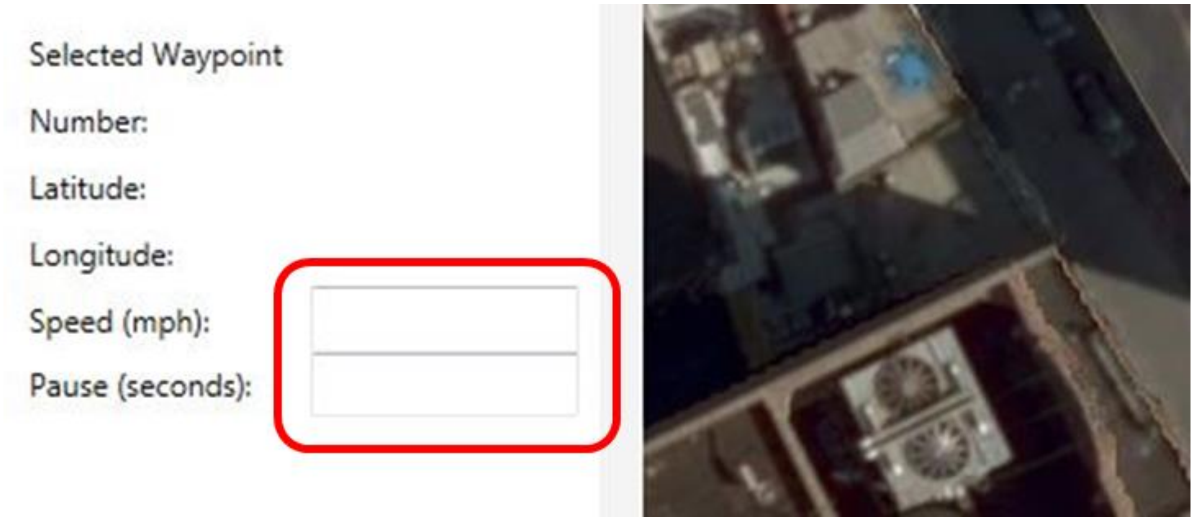


Figure 21. Defining the Waypoint Parameters

Figure 22. Latitude and Longitude Parameters Automatically Populated

Once the waypoint parameters are defined, the scenario can be played back by clicking the play icon in the upper-left corner of the window. At any instance in the scenario, the emergency vehicle icon can be clicked, which will enable all vehicles in the scenario to transmit SRMs. See Figure 23 below.

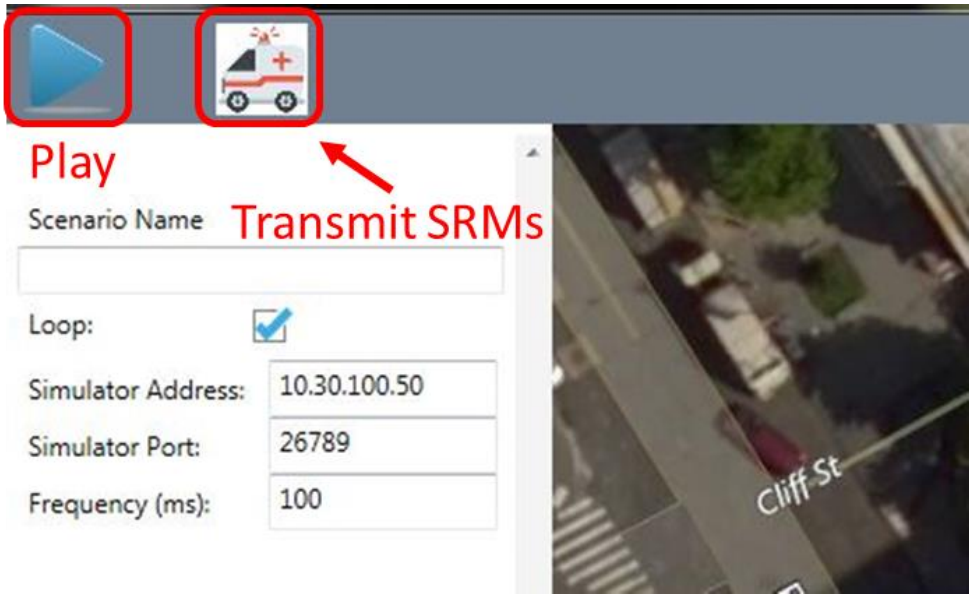


Figure 23. Play and Transmit SRM icons

If the vehicle(s) are enabled to transmit SRMs, the vehicle in the scenario will blink red rather than appearing the solid color the vehicle was assigned. This is illustrated in Figure 24.

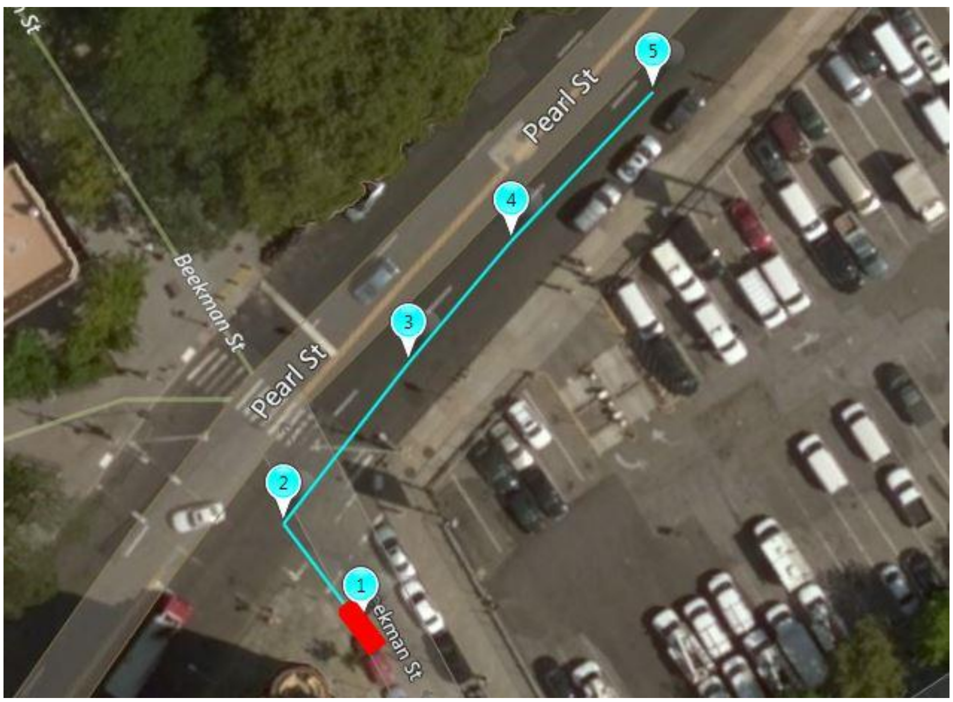


Figure 24. Transmit SRMs Enabled

The scenario can be stopped by clicking the stop button at any moment while the scenario is being played. The play button icon will transition to a stop button, while the scenario is being played. If the vehicles were enabled to transmit SRMs, this feature can be disabled by clicking the emergency vehicle icon. This is shown below in Figure 25.

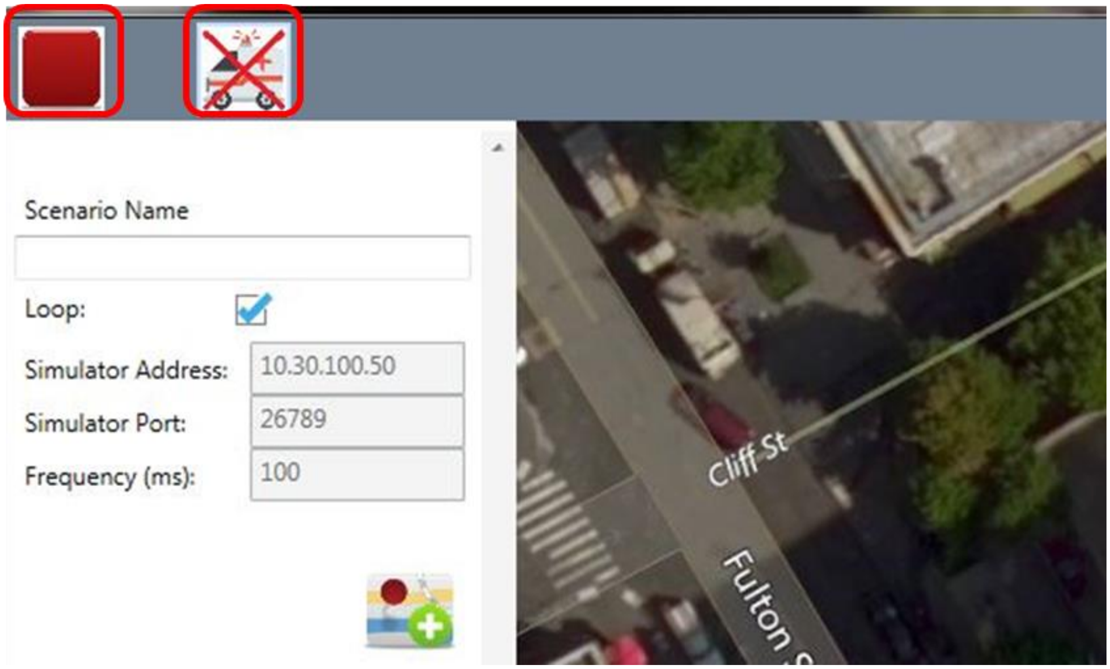


Figure 25. Pausing the Scenario and Disabling SRMs Being Transmitted

The layer features can be enabled or disabled throughout the time the scenario is being played or when the scenario is stopped. Simulation PushPins are the waypoints generated when creating the vehicle's path. The Simulation Path is the path that is drawn for the simulation vehicle(s) to follow. These two layers can only be enabled or disabled when a custom path is being drawn for the scenario. These features do not apply when a KML file is loaded into the simulation tool.

KML PushPins are the points at which the vehicle transmitted BSMs or SRMs in a previously recorded trip. This feature is currently disabled, but may be enabled in future versions. The KML Path is the layer that displays the previously recorded path of the vehicle. The Vehicles layer displays the vehicle traveling the custom path created or the path that was previously recorded. These three layers are only applicable when a previously recorded scenario is being played.

Figure 26 through Figure 29 illustrate the different feature layers that can be enabled and displayed in the simulation.

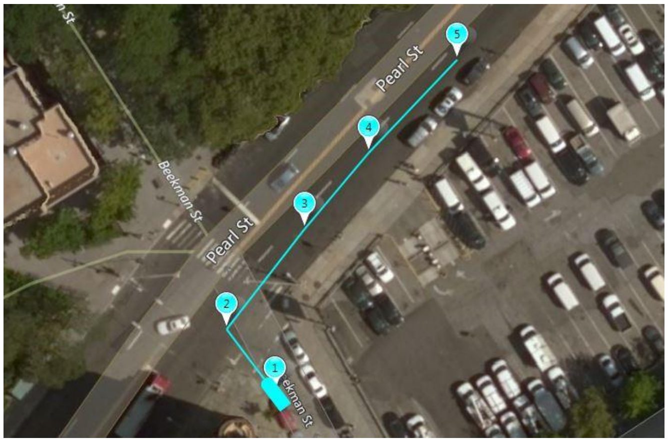


Figure 26. Vehicle, Simulation Path, and Simulation PushPin Layers Enabled

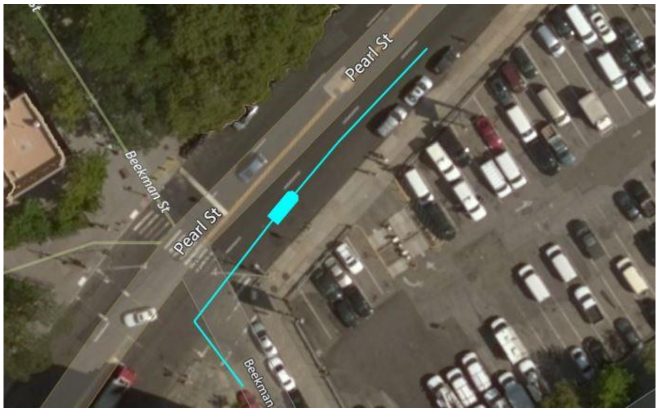


Figure 27. Vehicle and Simulation Path Layers Enabled, Simulation PushPin Layer Disabled

Figure 28. Vehicle Layer Enabled, Simulation Path and Simulation PushPin Layers Disabled

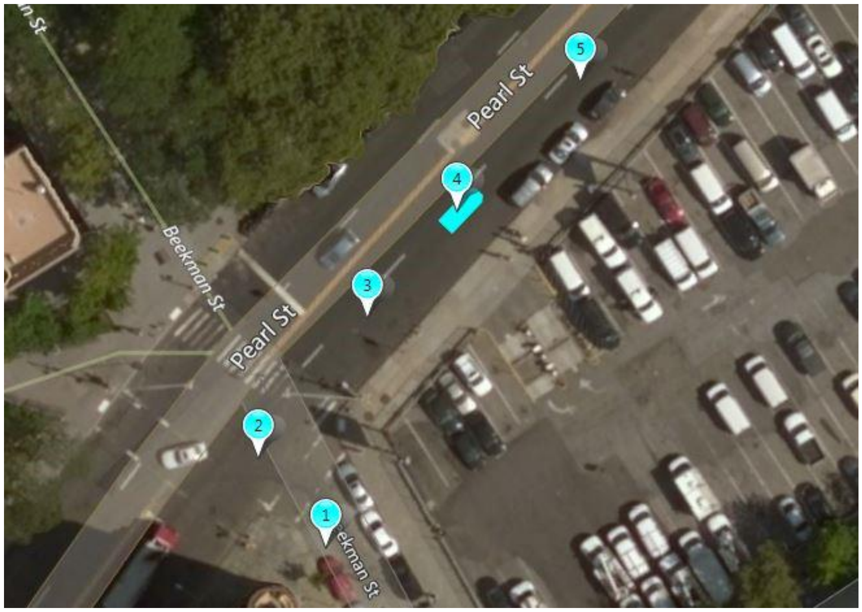


Figure 29. Vehicle and Simulation PushPin Layers Enabled, Simulation Path Layer Disabled

The KML Start Time and End Time slide bars shown in Figure 30 can be used to play a previously recorded trip from a specific start and end time within the scenario. These parameters must be set prior to playing the scenario or the scenario must be stopped in order for the Start or End Times to be adjusted. For example, the KML Start Time could be set to play from the middle of the scenario rather than from the beginning or these parameters may be adjusted to play a short duration of the scenario instead. The Find KML Vehicle button can be used to locate the vehicle, while playing a previously recorded scenario if the vehicle cannot be found within the map area displayed in the simulation window.

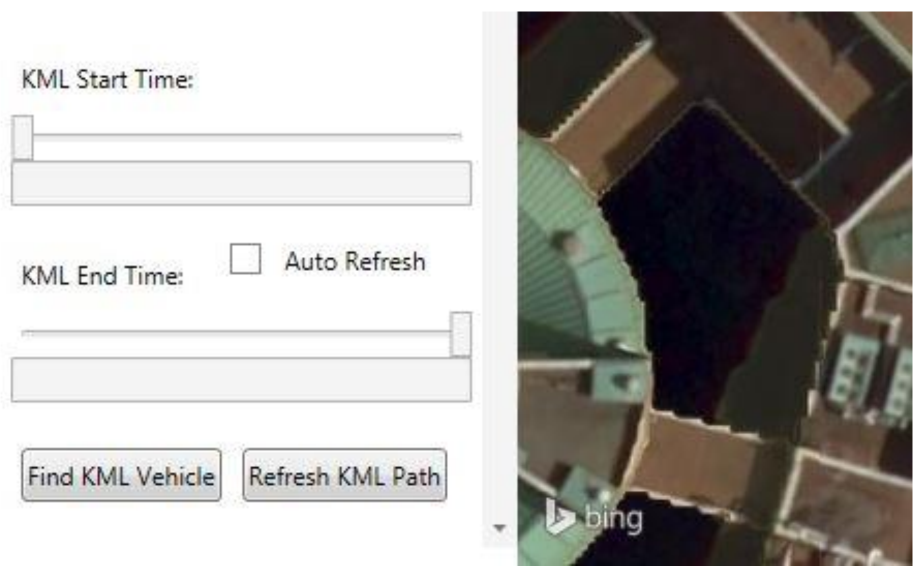


Figure 30. KML Start and End Time Parameters and Refresh Features

The Auto Refresh feature can be enabled by clicking the Auto Refresh checkbox. This will enable the KML Path to automatically update, while the KML Start and End Times are being adjusted. The Refresh KML Path button is the manual option to refresh the KML Path visually displayed, while the KML Start and End Times are being adjusted. Both refresh features can only be enabled or disabled while a previously recorded trip is being played. The refresh features are not available for custom scenarios being created.

Modifying a Scenario

Custom created, or previously recorded trips can both be modified; however, each type of scenario has limitations on the parameters and features that can be changed. These limitations are summarized in Table 4 through Table 7.

To modify a custom created scenario or previously recorded trip, begin by opening the KML file containing the previously recorded GPS data by clicking the folder icon in the upper-right hand corner of the window. Select the file and click the Open button. Once this file is opened, modifications to the scenario can be made by adjusting the parameters and features in the toolbar on the left-hand side of the window. This toolbar is shown in Figure 7. Opening a previously recorded trip is illustrated in Figure 31 below.

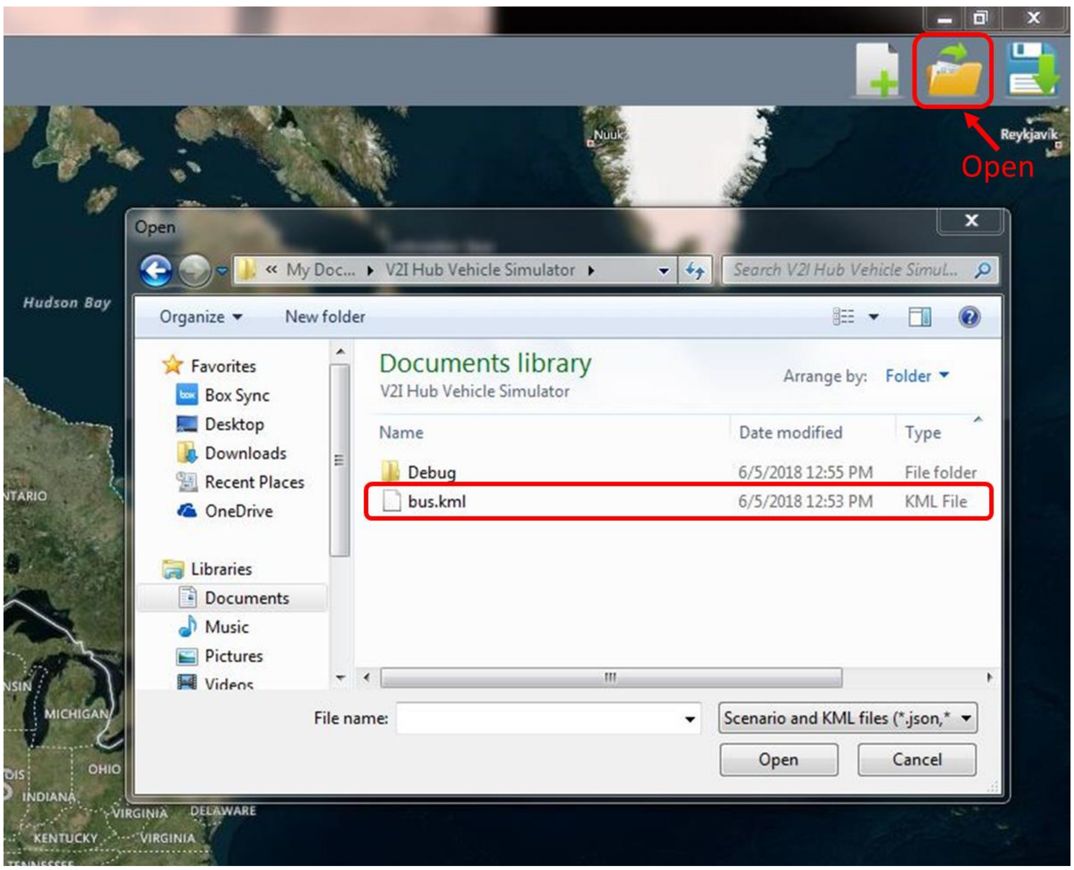


Figure 31. Opening a Previously Recorded Trip

The following tables summarize the modification limitations for custom created scenario and previously recorded trips.

Table 4. Parameter Modification Limitations of a Custom Created Scenario

Parameter	Description
Scenario Name	This parameter cannot be changed during playback.
Simulator Address	This parameter cannot be changed during playback.
Simulator Port	This parameter cannot be changed during playback.
Frequency (ms)	This parameter can be changed during playback.
Vehicle ID	This parameter cannot be changed at all once configured.
Number of Clones	This parameter cannot be changed during playback.
Clone Offset (seconds)	This parameter cannot be changed during playback.
Speed (mph)	This parameter can be changed during playback, while the scenario is stopped.
Pause (seconds)	This parameter can be changed during playback, while the scenario is stopped.
KML Start Time	This parameter can be changed during playback, while the scenario is stopped.
KML End Time	This parameter can be changed during playback, while the scenario is stopped.

Table 5. Parameter Modification Limitations of a Previously Recorded Trip

Parameter	Description
Scenario Name	This parameter cannot be changed during playback.
Simulator Address	This parameter cannot be changed during playback.
Simulator Port	This parameter cannot be changed during playback.
Frequency (ms)	This parameter cannot be changed during playback.
Vehicle ID	This parameter cannot be changed during playback.
Number of Clones	This parameter cannot be changed during playback.
Clone Offset (seconds)	This parameter cannot be changed during playback.
Speed (mph)	This parameter cannot be changed during playback.
Pause (seconds)	This parameter cannot be changed during playback.
KML Start Time	This parameter can be changed during playback, while the scenario is stopped.
KML End Time	This parameter can be changed during playback, while the scenario is stopped.

Table 6. Feature Modification Limitations of a Custom Created Scenario

Feature	Description
Search	Locations can be searched for, while playing the scenario
Play	The scenario can be stopped at any moment during playback.
SRM	Vehicles can be enabled to transit SRMs at any point during playback.
Loop	This feature can be enabled or disabled at any point during playback.
Add Vehicle	This feature can be enabled when the scenario is stopped.
Delete Vehicle	This feature can be enabled when the scenario is stopped.
Simulation PushPin Layer	This feature can be enabled or disabled at any point during playback.
Simulation Path Layer	This feature can be enabled or disabled at any point during playback.
KML PushPins	Displays every point where a vehicle transmitted a BSM or SRM. This feature is currently disabled but may be enabled in future versions.
KML Path Layer	This feature is disabled for custom created scenarios.
Vehicle Layer	This feature can be enabled or disabled at any point during playback.
Auto Refresh	This feature is disabled for custom created scenarios.
Find KML Vehicle	This feature can be enabled at any point during playback.
Refresh KML Path	This feature is disabled for custom created scenarios.

Table 7. Feature Modification Limitations of a Previously Recorded Trip

Feature	Description
Search	Locations can be searched for, while playing the scenario
Play	The scenario can be stopped at any moment during playback.
SRM	Vehicles cannot be enabled to transit SRMs. A new vehicle must be added to the previously recorded trip and enabled to transmit SRMs.
Loop	This feature can be enabled or disabled at any point during playback.
Add Vehicle	This feature can be enabled when the scenario is stopped.
Delete Vehicle	This feature can be enabled when the scenario is stopped.
Simulation PushPin Layer	This feature can be enabled or disabled at any point during playback.
Simulation Path Layer	This feature can be enabled or disabled at any point during playback.
KML PushPins	Displays every point where a vehicle transmitted a BSM or SRM. This feature is currently disabled but may be enabled in future versions.
KML Path Layer	This feature can be enabled or disabled at any point during playback.
Vehicle Layer	This feature can be enabled or disabled at any point during playback.
Auto Refresh	Automatically updates the path when the KML Start or End Times are adjusted.
Find KML Vehicle	This feature can be enabled at any point during playback to center focus on the vehicle(s) in the scenario.
Refresh KML Path	This feature is disabled during playback.

Appendix A. Acronyms

BSM	Basic Safety Message
CPU	Central Processing Unit
CV	Connected Vehicle
DOT	Department of Transportation
DSRC	Dedicated Short-Range Communications
FHWA	Federal Highway Administration
IP	Internet Protocol
JSON	JavaScript Object Notation
KML	Keyhole Markup Language
MAP	Roadway Geometry and Attribute Data (a.k.a. GID)
NTCIP	National Transportation Communications for Intelligent Transportation System Protocol
OBU	On-Board Unit
OSADP	Open-Source Application Development Platform
PC	Personal Computer
RSU	Roadside Unit
SPaT	Signal, Phase, and Timing
SRM	Signal Request Message
UDP	User Data Protocol
V2I	Vehicle-to-Infrastructure

U.S. Department of Transportation
ITS Joint Program Office – HOIT
1200 New Jersey Avenue, SE
Washington, DC 20590

Toll-Free “Help Line” 866-367-7487

www.its.dot.gov

FHWA-JPO-XXX

U.S. Department of Transportation