755 Phoenix Drive, Ann Arbor MI, 48108, USA
Phone: 734 668-2930 • Fax: 734 668-2877 • Email: info@carsim.com
carsim.com

# Analyzing CarSim/TruckSim with Tensorflow

Using the shared buffer information from CarSim/TruckSim Live Animation, it is possible to conduct the image data analysis. This memo describes the way to connect CarSim with Tensorflow and analyze the image data. The same methods are applied in TruckSim.

The example and Python scripts are not included in the Windows version of our software but are available to download from www.carsim.com.

## Who Should Use this Manual

The information in this document is intended for programmers who wish to run CarSim and run image data analysis using Python programming. A basic knowledge of running Tensorflow and machine learning is required. This manual will NOT explain how to do data analysis/machine learning, how to use python, or how to install or run Tensorflow.

## Prerequisites

To run the simulation described in this documentation, a CarSim license with a live animation feature is required.

All the files/software to run Tensorflow Object Detection API need to be installed from the following link: https://github.com/tensorflow/models/tree/master/research/object_detection

Please consult the Installation section of the above site.

Note that since this site is contributed to by several parties and is often updated, there is no guarantee that the provided files with this documentation will work in all cases. The followings are conditions we tested with:

- The above Tensorflow Object Detection API was downloaded and last tested on June 28, 2019. Any updates after this date were not tested.
- NVIDIA CUDA toolkit 9.0 was installed. We tested with a machine with NVIDIA Quadro M2000 and another with GeForce GTX 1050 Ti.
- cuDNN v7.1.4 and cuDNN v7.3.1.
- Python 3.6.5 and 3.6.6.
- Protobuf 3.4.0 version.
- Tensorflow-gpu version 1.8.0 and 1.12.0

## Installation

To install, go to www.carsim.com and download a ZIP archive.

The following files are contained in a ZIP archive.

- This documentation.
- CPAR file for an example.
- Four Python files.
- Wrapper DLL files.
- Movie file.

The movie files are created to preview what sort of run you will expect to see. These files do not need to be installed anywhere.

The CPAR file can be imported and installed in the existing database or in the new database.

The two DLL files and four python files need to be stored where the Tensorflow Object Detection API is installed. Put the following six files in */tensorflow_models/research/object_detection* folder. The descriptions of these files are as follows:

| | |
|---|---|
| `vs_sb_64.dll` | The shared buffer DLL |
| `vs_lv_ds_x64.dll` | The solver/live animation DLL |
| `GetsharedBufferInfo.py` | A python wrapper to read information from the shared buffer using the shared buffer DLL. |
| `Simulation_with_LiveAnimation.py` | A python wrapper to run solver and live animation using vs_lv_ds_x64.dll |
| `ObjectDetection.py` | Run pre-analyzed Tensorflow. |
| `VS_Run_with_LiveAni_Object_Detection.py` | Import above three Python files, run solver with live animation, run pre-analyzed Tensorflow, and terminate the simulation. |

## How to run CarSim Example

After installing the CPAR file from the ZIP file, the example datasets are found in one of the categories.

Click the blue link under the **Generate For this Runs** button and go to **Models** dataset (Figure 1).

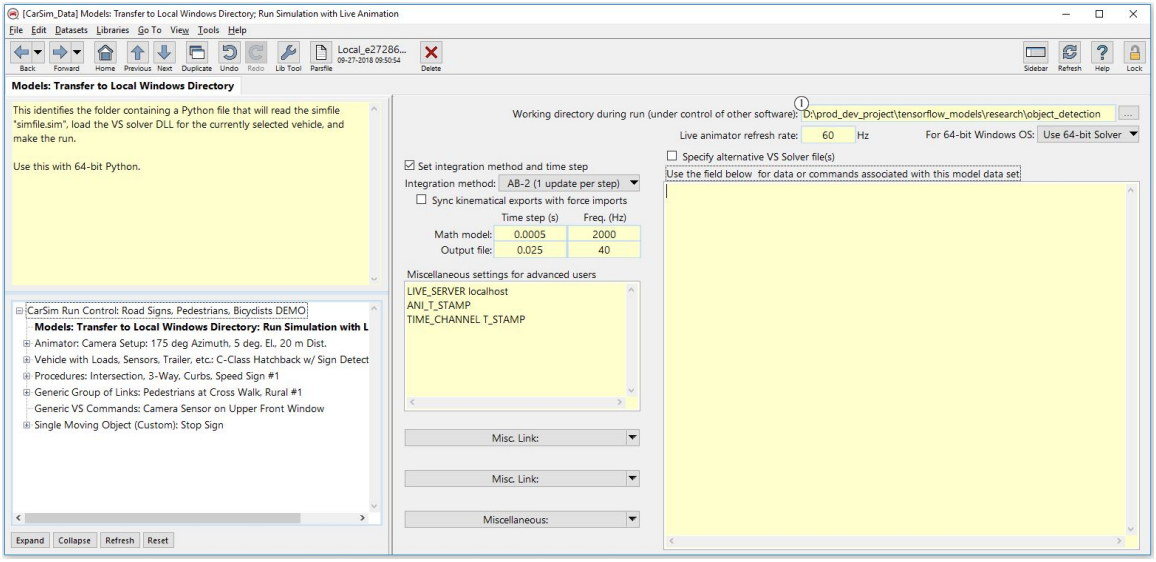


*Figure 1. Models: Transfer to Local Windows Directory*

Change the path of **Working directory during run (under control of other software)** to where the Tensorflow Object Detection API is installed; */tensorflow_models/research/object_detection* folder. Changing this path means creating a simfile.sim in this path. This is the same directory where the *.dll and four Python files are installed in the previous section.

Go back to the Run Control screen. Press the **Generate Files for this Run** button. Check to make sure simfile.sim is created inside the above path.

Press the **Start Live Video** button to open a live animation. Make sure '1' is selected for the number of live videos.

Run the VS_Run_with_LiveAni_Object_Detection.py that lives in the Tensorflow Object Detection API directory. One way to do this is to open a command window, navigate to this directory, and then run the python script by typing:

```
python VS_Run_with_LiveAni_Object_Detection.py
```

This should start the CarSim solver, and another window will open displaying the Tensorflow analysis. When finished running, press **q** to close the window.

# Editing Python Files

Depending on the CarSim example, it might be necessary to edit a few lines of some Python files which are included in the ZIP file.

## GetSharedBufferInfo.py

The shared buffer name can be set in CarSim. The example in the CPAR file specifies the shared buffer name as "EBS". This was set inside **Generic VS Commands; {Camera Sensors} Camera Sensor on Upper Front Window** dataset. If the shared buffer name is edited, the bufferName variable in Line 23 needs to be changed to the new name.

```
bufferName = 'EBS'
```

### ObjectDetection.py

Since the purpose of this example is to simply show the users how to connect CarSim with Tensorflow, the pretrained “SSD with Mobilenet” model is used. The users can edit this file to use their own trained model.

## Making an Own CarSim Example

This section discusses how to Make a CarSim example run with the pretrained Tensorflow.

We will use our **{ADAS and Active Safety} ACC, 4-Lane Road, Fwd and Opp. Traffic** example to explain how to set up.

Duplicate **{ADAS and Active Safety} ACC, 4-Lane Road, Fwd and Opp. Traffic** example. Leave the title the same but change the category to: **Tensorflow Example**.

[Copy and Link Dataset] and duplicate C-Class Hatchback w/ACC & Blind-Spot dataset (Figure 2). Provide the new name for this duplicated dataset. Here, the default title name, **C-Class Hatchback w/ ACC & Blind-Spot #1**, is used. Click this dataset to go to Vehicle with Loads, Sensors, Trailers, etc. dataset.

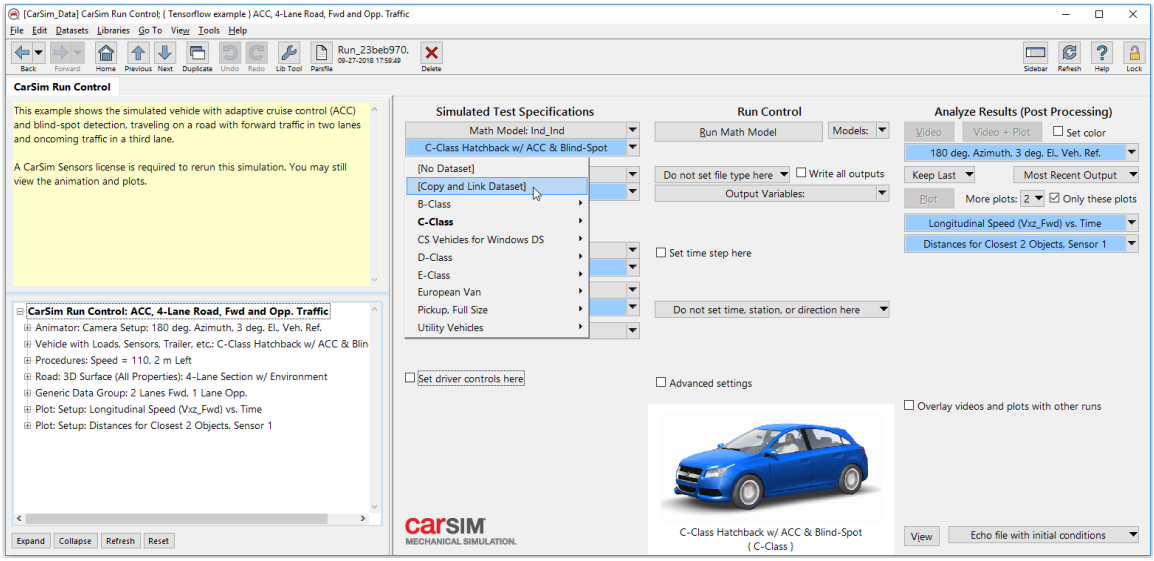


*Figure 2. Duplicating C-Class, Hatchback Vehicle Model.*

For **Lead Unit: Ind_Ind**, select **VS Visualizer Camera Sensor > C-Class, Hatchback: Camera** dataset (Figure 3).

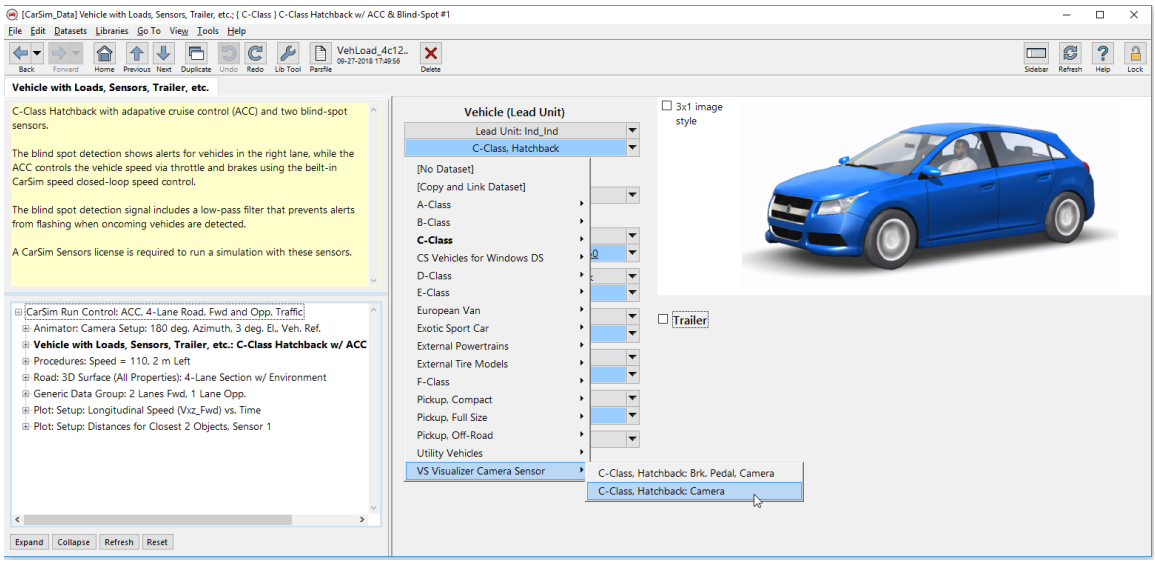


*Figure 3. Select C-Class, Hatchback: Camera Dataset.*

**C-Class, Hatchback: Camera** dataset already includes the camera sensor location information. Go back to the Run Control screen.

Select **Miscellaneous: > Generic > Generic VS** Commands (Figure 4), and then select Camera **Sensors** > **Camera Sensor on Upper Front Window** dataset (Figure 5).

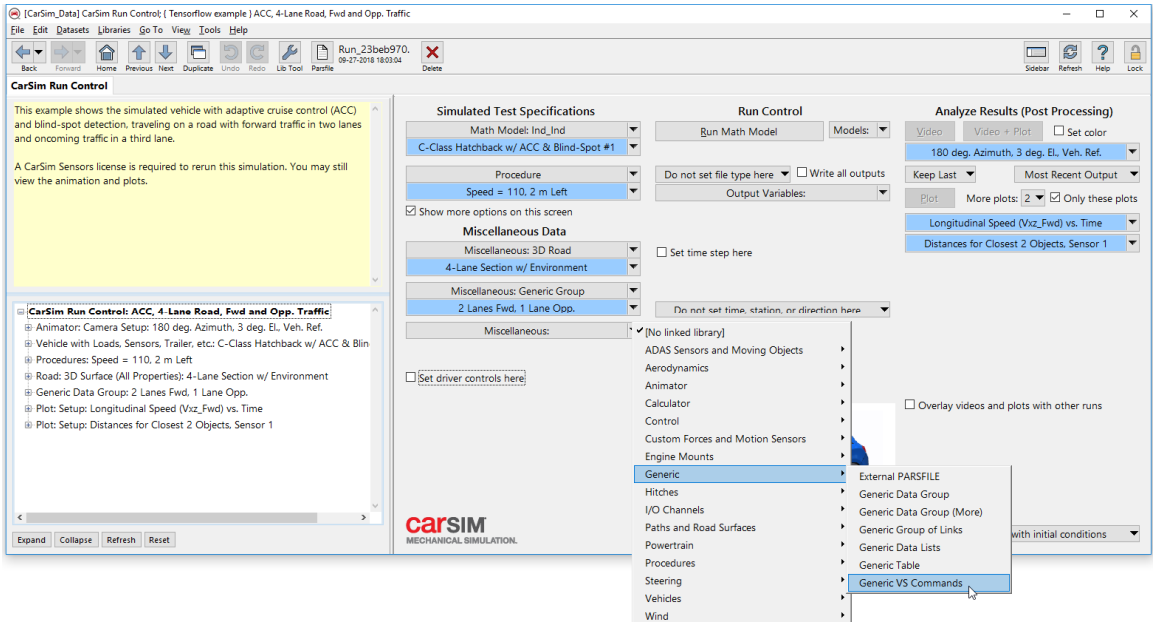


*Figure 4. Selecting Generic > Generic VS Commands in Run Control Screen.*

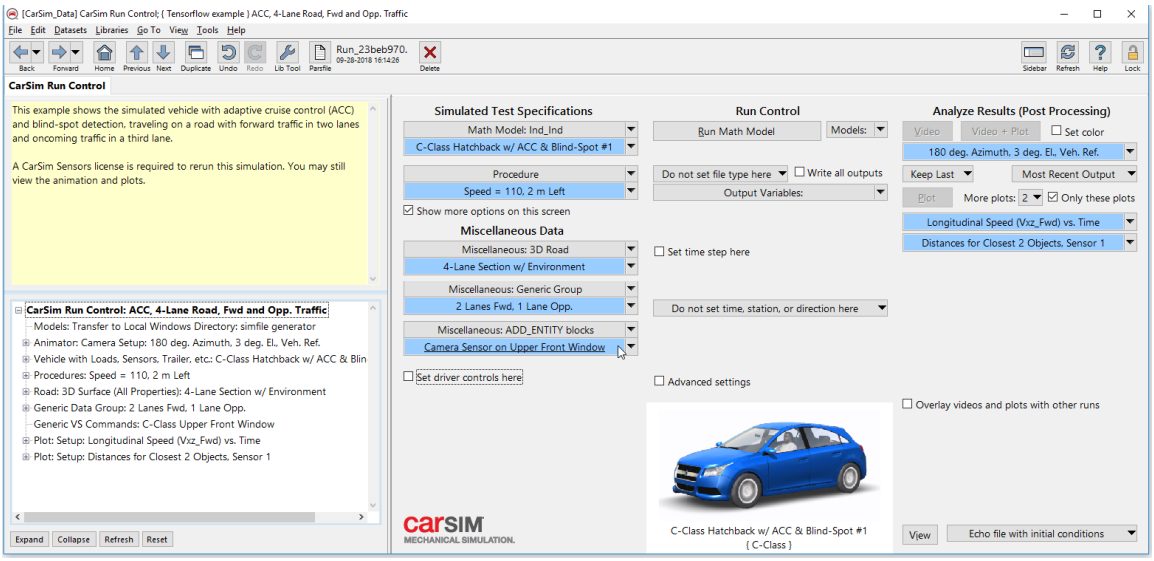


*Figure 5. Selecting Camera Sensor on Upper Front Window Dataset.*

Camera Sensor on Upper Front Window already has the camera sensor information such as the field of view, render size, and the shared buffer name. For more information on how to set a camera sensor using VS commands, please consult the Camera Sensors documentation.

Next, the location of where to save simfile.sim needs to be specified. Using the **Models:** pulldown menu, select **Models: Transfer to Local Windows Directory** (Figure 6).

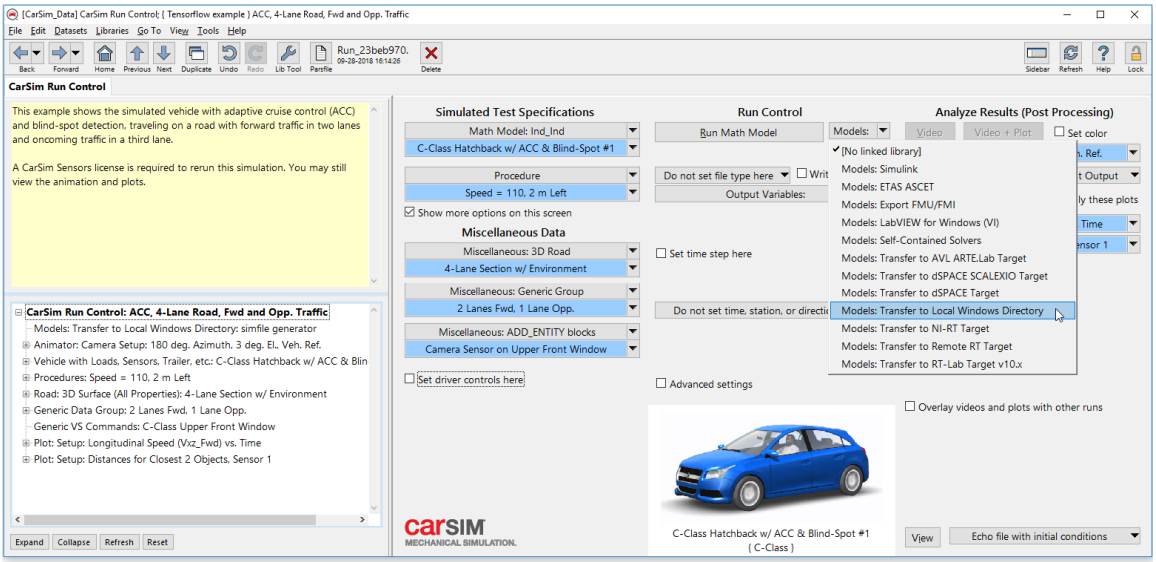


*Figure 6. Selecting Models: Transfer to Local Windows Directory.*

Use the pull down menu under Models: and select **[Link to New Dataset]** to create a new dataset. Name this dataset as: **simfile generator** (Figure 7).

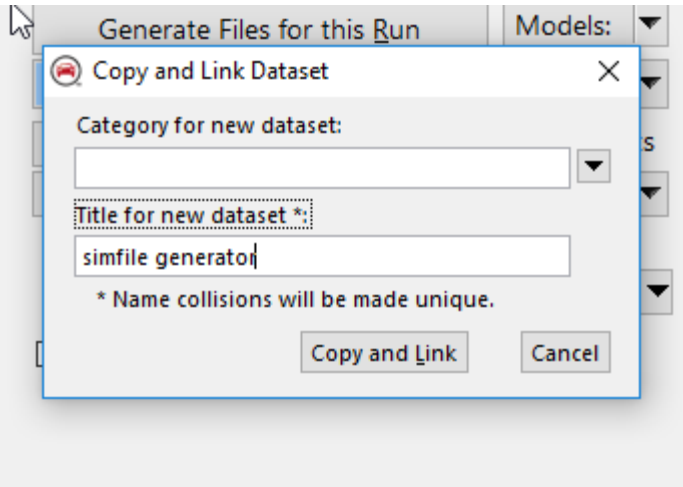


*Figure 7. Creating a New Dataset.*

Click this newly created dataset. Enter the path to the Tensorflow Object Detection API (*/tensorflow_models/research/object_detection*) in the field next to **Working directory during run (under control of other software)** (1) (Figure 8). Select **Use 64-bit Solver** (2). In the yellow field, enter the following lines. Adding these lines connect S-Function to Live Animator.

```
LIVE_SERVER localhost

ANI_T_STAMP

TIME_CHANNEL T_STAMP
```

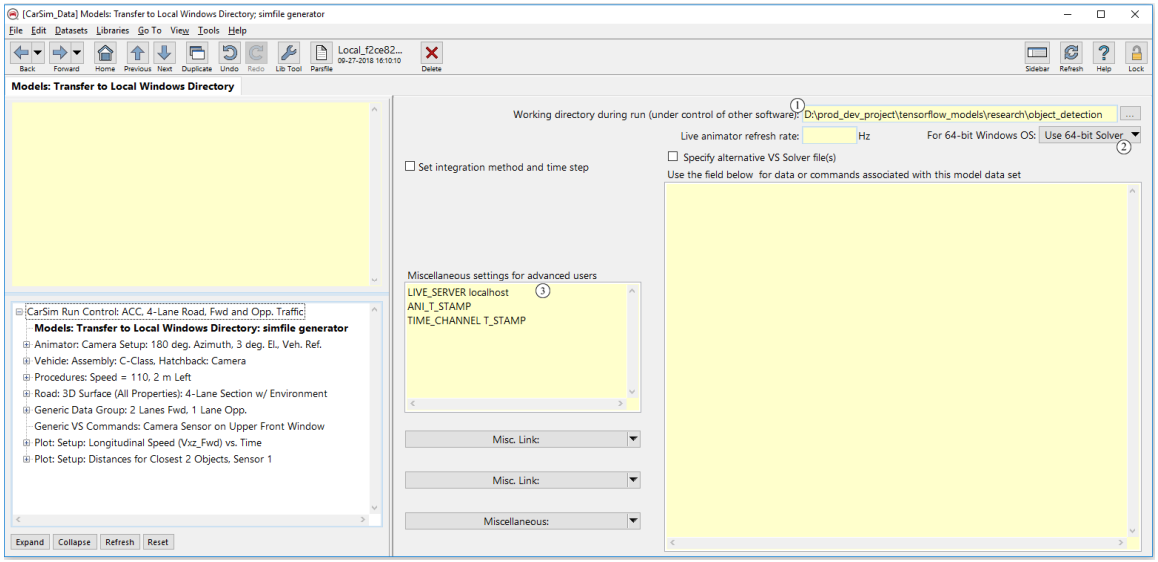


*Figure 8. Models: Transfer to Local Windows Directory Set-up.*

Go back to the Run Control screen. Select ‘1’ for **Number of Live Animators for this Run** (Figure 9).

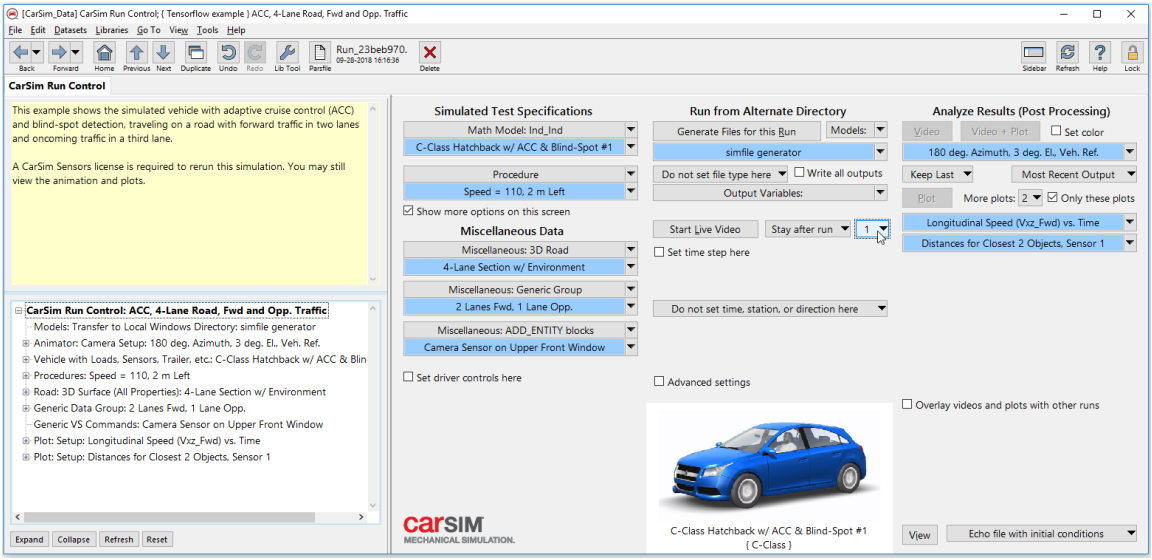


*Figure 9. Live Animator Set-up*

Press the **Generate Files for this Run** button. Check to make sure simfile.sim is created in the correct folder. Press the **Start Live Video** button to open the live animation.

After the live animation window opens, run VS_Run_with_LiveAni_Object_Detection.py either from a command window or from IDE if using. After starting this Python script, it will bring up another window for the object detections. When finished running, press **q** to close the window.