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

Initialize a Vehicle Using Imported Variables

The VS Math Models in CarSim, TruckSim, and BikeSim initialize state variables at the start of each simulation such that the vehicle is in a specified location, in approximate equilibrium, moving at a target speed. There are occasions where simulations are made under the control of other software, and there is a need to set the initial position and speed using external (imported) variables. This memo describes the initialization in a VS Math Model and provides an example for locating the vehicle with imported variables.

Normal Initialization

Initialization is normally performed for the case of a vehicle being controlled by the built-in driver model following a reference path whose PATH_ID matches the driver model parameter PATH_ID_DM, possibly adjusted laterally by a target specified with the Configurable Function LTARG for a dataset whose LTARG_ID matches the driver model parameter LTARG_ID_DM.

Note	For details on Reference Paths, please see the document: *Paths and Road Surfaces* from the submenu **Help** > **Paths, Road Surfaces, and Scenes**. For information about the controls for normal initialization, please see the document: *Procedures and Events* from the **Help** menu. For information about the initialization process in CarSim and TruckSim, please the **Help** > **Technical Memos** document: *Initialization in CarSim and TruckSim.*

The vehicle is located such that its station along the specified path matches the parameter SSTART, and the lateral coordinate L matches the value obtained by the specified LTARG datasets given station SSTART. The VS Math Model calculates the initial values of the state variables Xo and Yo (also identified by the names SV_XO and SV_YO) from the initial S coordinate SSTART and L calculated from LTARG. The initial yaw angle Yaw (also identified as SV_YAW) is obtained from the heading of the path and the parameter OPT_DIRECTION, which is 1 if the vehicle is heading in the direction of increasing S along the path, or –1 if heading in the opposite direction.

Even if the driver model is not used, there is still a reference path identified by the ID PATH_ID_DM. If there is no reference LTARG dataset, then the initial location is calculated using L = 0.

The tire and spring compressions are calculated based on the specified load conditions. They are combined with the underlying road/terrain 3D geometry to estimate the initial Zo coordinate, along

with Roll_E and Pitch. The calculation of Zo, Roll_E, Pitch is somewhat complicated, especially on a 3D ground surface, and also includes the calculation of all suspension deflection variables.

If the speed controller is engaged, then some of the speed-related state variables are set to match the target speed of the speed controller. SV_VXS (component of sprung mass velocity in the body-fixed X direction) is set to the target speed multiplied by cos(Θ_{rel}), where Θ_{rel} is the pitch of the road surface minus the pitch of the vehicle sprung mass, and SV_VZS (velocity component in the body-fixed Z direction) is set to the target speed multiplied by –sin(Θ_{rel}). The spin for each wheel is set to the target speed divided by the tire effective rolling radius. Powertrain spin variables are also set to match the wheel spins, where appropriate.

If the speed controller is not engaged, then the initialization is performed using the current value of SV_VXS as the target speed.

The overall initialization is enabled by three parameters: OPT_INIT_PATH, OPT_INIT_SPEED, and OPT_INIT_CONFIG. All have default values of 1.

The OPT_INIT_PATH parameter may be disabled by a checkbox (3) on the **Procedures** screen (Figure 1). If this checkbox is not checked, and the drop-down control (1) does not specify that the run is stopped when a station is reached, then the path station field (2) is not visible because a starting station is not used by the VSM Math Model. In this case, the Xo, Yo, and Yaw variables are not modified in the initialization.

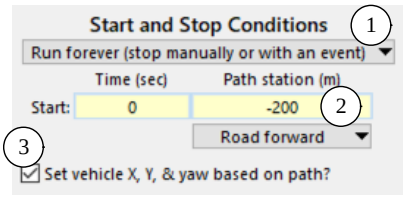

Figure 1. Checkbox on the Procedures screen.

Reset the Vehicle in an Event

The **Events** screen has four checkboxes to possibly reset the vehicle state when an **Events** dataset is read by the VS Math Model (Figure 2). If the checkbox **Specify initialization details** is not checked, the other checkboxes are not shown and the parameters OPT_INIT_PATH, OPT_INIT_SPEED, and OPT_INIT_CONFIG keep their values. If the Event dataset is read after the run is in progress, then the three parameters have values of 0 unless they are specifically modified in an Event.

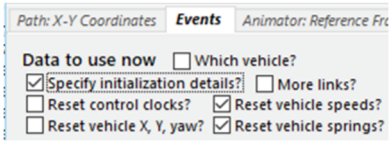

Figure 2. Checkboxes on the Events screen.

The reset options on the **Events** screen are used for simulations that connect a series of tests, with the vehicle being moved to a designated starting point for the next test. In some cases the speed is reset, in others it is left intact. If the ground surface is flat and level, it is not always necessary to reset the vehicle springs (and Zo, Pitch, and Roll). However, if the simulation takes place on a surface that is not flat and level, then the spring reset option is usually needed. If the location is not changed, or if the new location is specified with `Xo`, `Yo`, and `Yaw`, then the **Reset vehicle X, Y, Yaw** box should be unchecked. If checked, then the new location is calculated from `SSTART` and `OPT_DIRECTION` values, as described earlier.

Sequence of Operations with an External Tool

The sequence of operations in a simulation is described in the *VS Commands Reference Manual*, obtained from the submenu **Help > Reference Manuals**. The following description applies for the case where the VS Math Model is running under the control of an external software tool such as Simulink.

The first part of the simulation is called the initialization. The VS Math Model is assembled from modules based on information from input files, values are read for parameters, tables are read for Configurable Functions, additional parts are added with Install and Define commands, Import and Export variables are activated, and initial conditions are calculated. During this process, there has not yet been any exchange of information via import and export arrays. Therefore, it is not possible to use imported variables for the internal initializations done in this part of the simulation.

After the internal initialization, the simulation runs by performing calculations at discrete intervals of time, called time steps. At each step, the controlling tool applies the VS Math Model. Import and export arrays are used to exchange information between the controlling tool and the VS Math Model. The VS Math Model receives values in the import array, and provides values in the export array.

At the start of each time step, the VS Math Model has values for all state variables, and all variables imported from the external tool. The VS Math Model then proceeds to perform the following steps, simplified here to include only the actions that are relevant to this memo.

1. Calculate everything needed to provide derivatives for built-in ordinary differential equations and output variables that may be written to file and/or exported.
2. Write outputs to file or a buffer if the current time step is to be recorded.
3. Scan pending Events. If any event is triggered:
    a. Read the linked Parsfile. If the linked Parsfile includes links to more Parsfiles, read them also. Otherwise, stop the run.
    b. If any of the restart options were set in a Parsfile that is read (Figure 2), perform the associated initialization/reset calculations.
4. Copy selected output variables into the array of exports that shared with the external tool.
5. Numerically integrate all differential equations to obtain values of the state variables for the next time step.

Using Imported Variables for Initialization

As already noted, it is not possible to use imported variables for the initialization performed by a VS Math Model prior to the first time step; variables have not yet been imported.

However, it is possible to use imported variables for a reset performed at the end of the first time step. Consider a Simulink model that provides four constants: 10, 5, -15, and 80 (Figure 3).

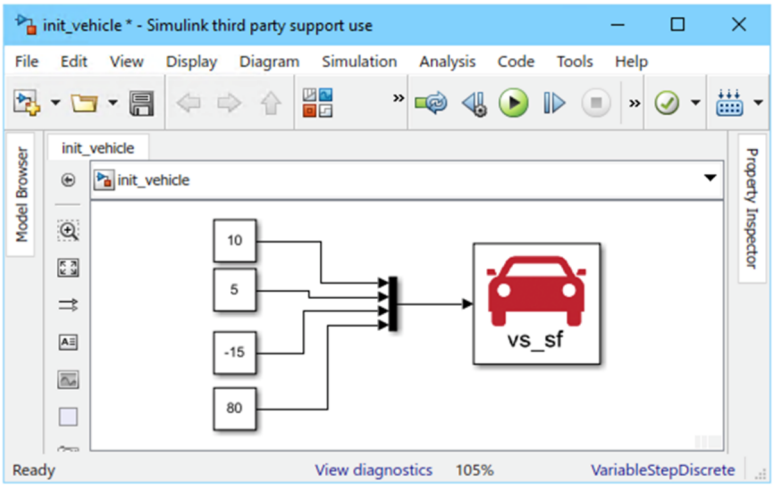

Figure 3. Simulink model that provides three constants to CarSim.

This model will be used with a simulation setup based on the CarSim Quick Start Guide double lane change (Figure 4). It was modified by linking to the Simulink model (1) and also an **Events** dataset (2). It has the output time step set to match the math model time step (3), to provide maximum detail in plots that will be shown later.

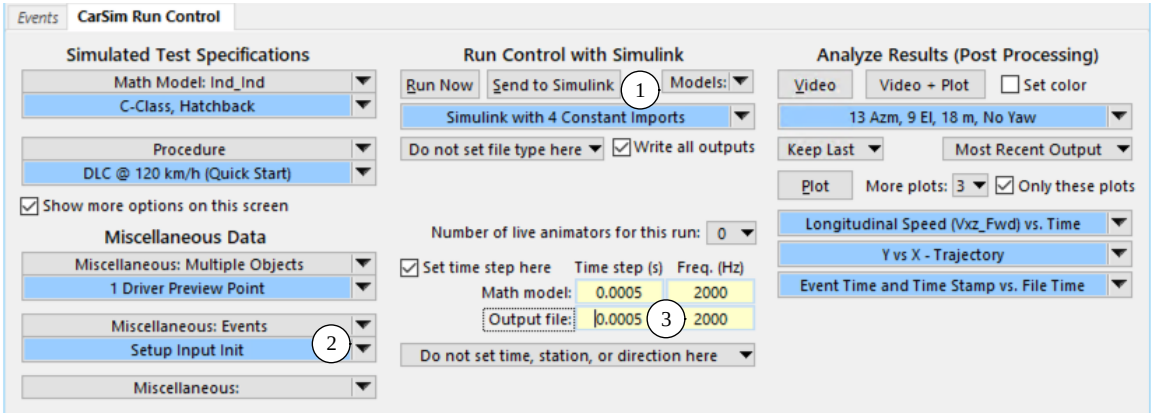

Figure 4. Run Control dataset that includes an Events dataset and a Simulink model.

Figure 5 shows the **I/O Channels Import** dataset used to bring the four variables from Simulink into the CarSim model. Three of these will be used to set the initial vehicle X, Y, and Yaw variables. Imports for these variables do not exist, so the VS Command `DEFINE_IMPORT` was used to create the three new variables (6), and also set their units. In order to use the browser capability of the screen, the **Run Control** dataset (Figure 4) was selected (1); the pathname is shown (2) with a

summary of the import variables. By selecting this run, the browser also includes the three new variables. They were listed in the category VS Commands (3), and were activated by double-clicking on their names. Another variable, `IMP_SPEED`, also exists, and will be used to set the target speed for the built-in speed controller. It was also activated by double-clicking on its name in the Available Variables list (4). Overall, there are four import variables activated, in the order shown (5).

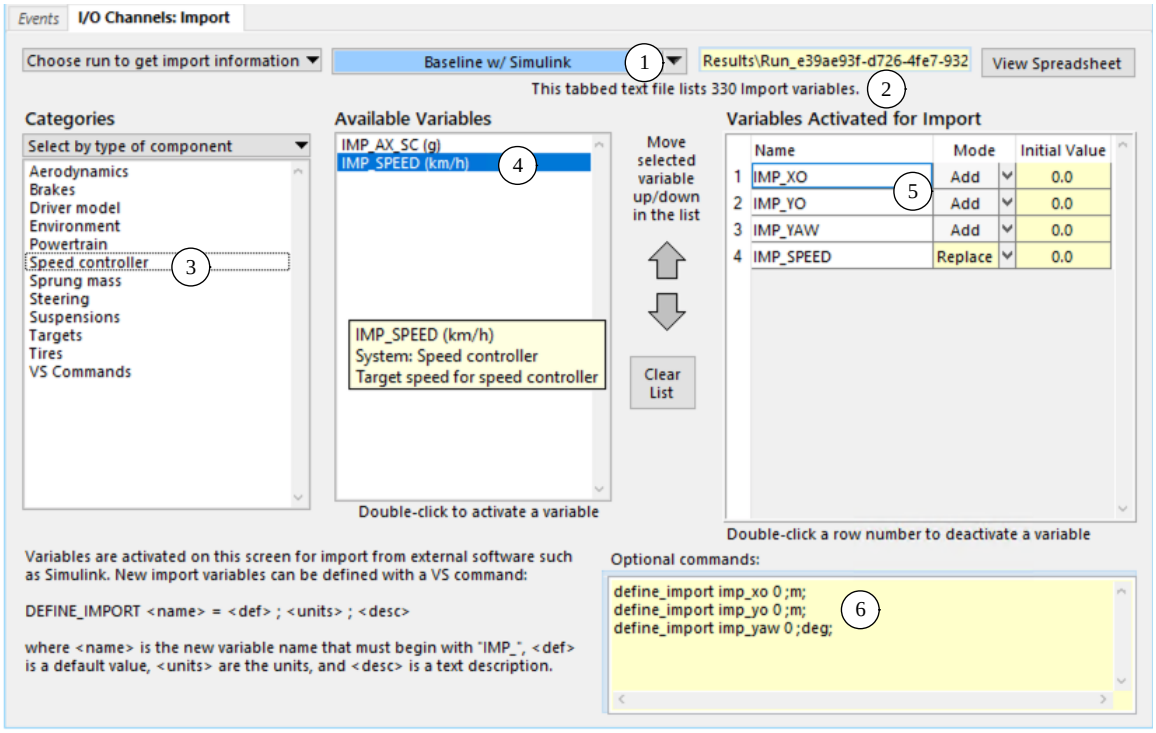

Figure 5. Three import variables created with VS Commands and an existing import for speed.

The speed import is set to **Replace** the target speed for the speed controller. The other three imports only have the option to **Add** to the model, as is the case for all imports defined with the `DEFINE_IMPORT` command.

Having four import variables and a link to the Simulink model (Figure 3) provide compatibility between the **Run Control** dataset (Figure 4) and the Simulink model. Clicking the **Send to Simulink** button sets up the Simulink model to make the run. If the Events are not used (2), the simulation results from Simulink would use a speed of 80 km/h (after the first time step), and start in the original location.

Figure 6 shows the **Events** dataset that is linked to the **Run Control** screen. This dataset is very simple: it does not specify initialization details (1); all it does is disable writing to file (2) and create a pending Event that will trigger if 1 is not equal to 0 (3). That is, it will trigger the first time it is tested, which will be near the end of the first time step. When triggered, the VS Math Model will read the linked dataset (4), which is shown in Figure 7.

The **Events** dataset shown in Figure 7 has the box checked to **Specify initialization details** (1), and also the boxes to reset speeds (3) and springs (4). However, the box to **Reset vehicle X, Y, and yaw** is not checked (2). Instead, that is done manually in the miscellaneous field (5).

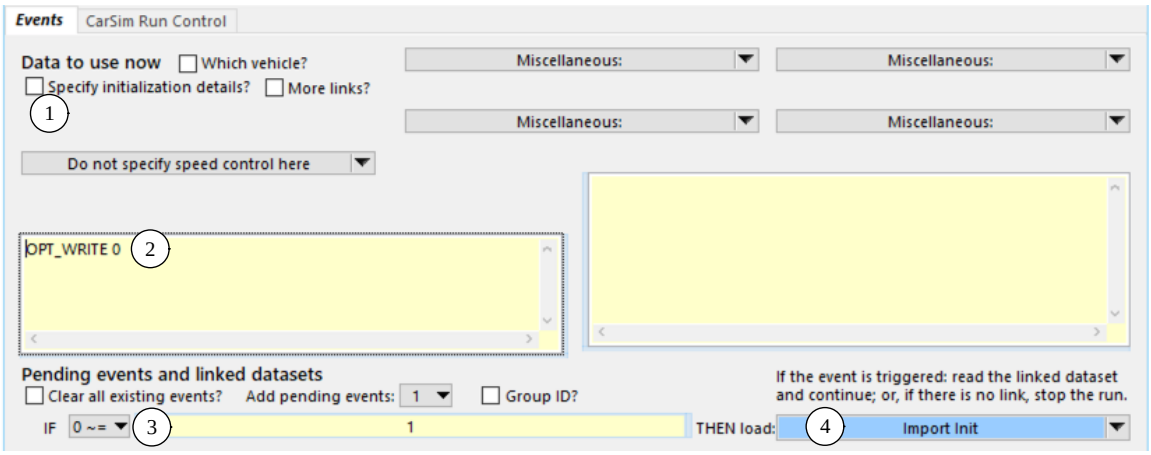

Figure 6. Setting up the Event that will trigger during the first time step.

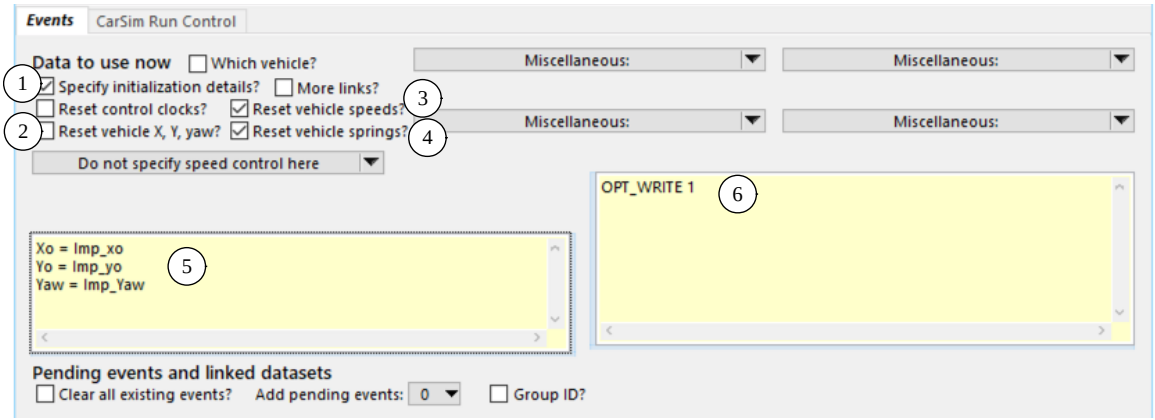

Figure 7. The dataset that will be read at the end of the first time step.

The X-Y coordinates `Xo` and `Yo` are set to `imp_xo` and `imp_yo`, respectively (5). The Yaw is set to `imp_yaw`. Recall from the Simulink model that these are constants of 10, 5, and -15, respectively. The speed target is automatically replaced with the specified import `imp_speed`, which has a value set in Simulink of 80.

Another setting in this dataset enables writing to file (6).

Results are shown for two runs. In the first (Figure 8), the statement `OPT_WRITE 0` in Figure 6 was commented out, to enable writing to file from the very start, when T = 0. Visually, the vehicle is located on the target path and aligned with it. The plot of longitudinal speed indicates that the target speed is initially 80km/h but the vehicle speed is 0. The plot of coordinates shows the X-Y coordinates of the vehicle to be 0,0. The time plot indicates that at the time shown as 0 in the lower-right corner of the VS Visualizer window, the simulation time stamp is also 0.

Recall from the earlier description of the initialization that import variables are not available during the internal initialization. At that time, `IMP_SPEED` had not yet been obtained from Simulink, and a value of 0 was used to initialize the vehicle speed. The X-Y coordinates and Yaw angle have their default values of 0.

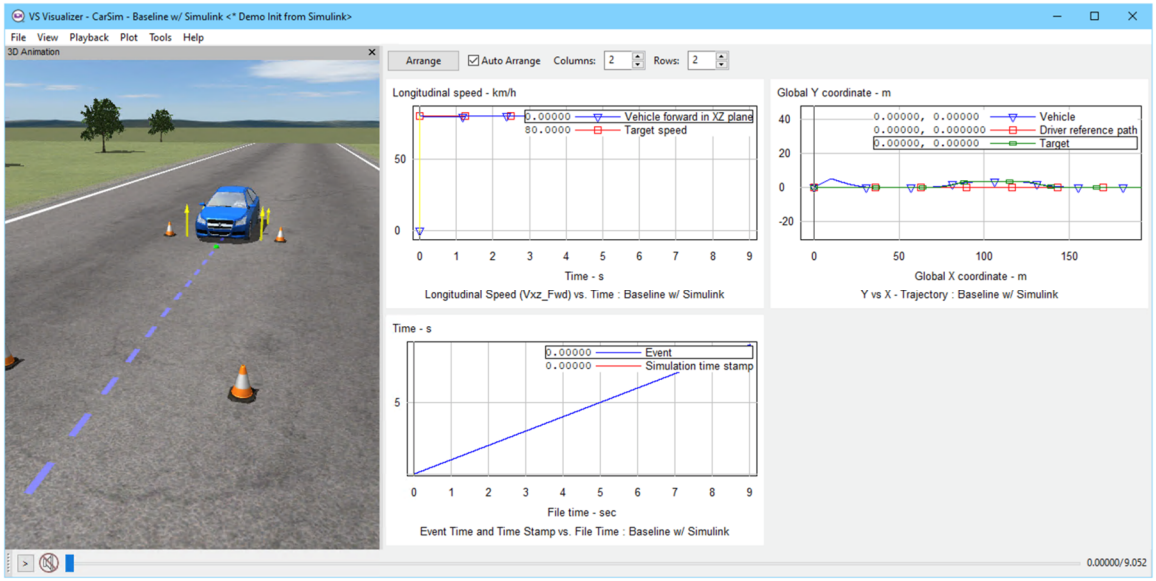

Figure 8. Video for T = 0 and plots starting at T = 0.

In the first time step, the target speed is set to 80 km/h from the import variable. The import variables `imp_xo`, `imp_yo`, and `imp_yaw` have the values set in Simulink, but they haven't been used yet when the VS Math Model write values into the output file at T = 0. After writing to file, the Event is triggered, and the dataset from Figure 7 is read. The model state variables for Xo, Yo, and Yaw are now set to the values imported from Simulink. They would potentially be updated by numerical integration, but because the vehicle is at rest, they will keep their values for the next time step.

To better see the behavior after T = 0, consider the plots shown in Figure 9. These result were obtained by restoring the setting statement `OPT_WRITE 0` in Figure 6. With that setting, results were not written to file until the completion of the second time step, when T = 0.0005s. Notice that the position at T = 0.0005s is X = 10, Y = 5. The yaw was set to -15°, as indicated visually in the video. The target speed is 80.0 km/h. The plot of time stamp vs. file time shows that the first data point (identified as T = 0 in the lower-right corner of the window) has the simulation time at 0.0005s.

Summary

When linking two or more software tools together, the timing for communication is important. In the case of a VS Math Model running under the control of another tool (e.g., Simulink), the first exchange of information between the tools is at the start of the run, when `T = TSTART` (typically 0). There are only two (theoretically) possible sequences here for the first communication:

1. The exports from the VS Math Model are known and provided to Simulink, but the imports received from Simulink were not known when the exports were calculated.
2. The imports from Simulink are known, but were generated without any knowledge of the state of the VS Math Model.

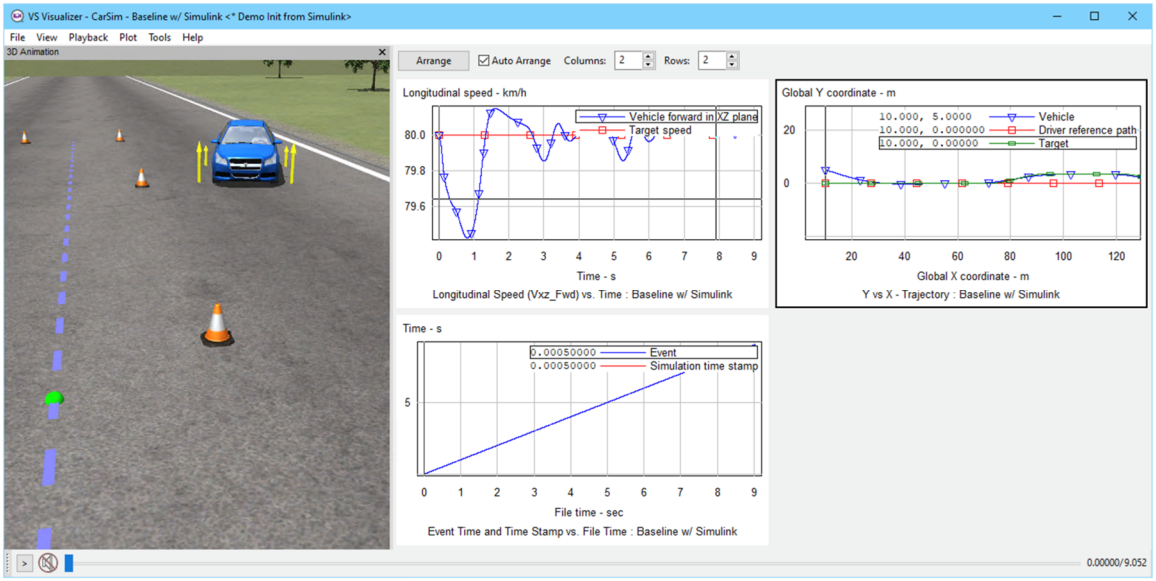

Figure 9. Video for T = 0.0005s and plots starting at T = 0.0005s.

In the vast majority of co-simulation setups, the behavior of the Simulink model depends strongly on information received from the VS Math Model. Therefore, the normal behavior is built on the first sequence, in which the VS Math Model has completed the internal initialization before any communication of import and export variables takes place.

However, the VS Event feature in VS Math Models provides a form of the second option, as shown by the example in this document. The Event supports a reset of the VS Math Model at any time during the simulation, allowing multiple test conditions to be run consecutively, with each initialized with specified conditions.