Run Control Screen (Home)

VehicleSim products combine information from data screens with VS Math Models to simulate vehicle behavior. They also link the simulation results with VS Visualizer, as shown schematically in Figure 1. The main programs for the products are named `BikeSim.exe`, `CarSim.exe`, `SuspensionSim.exe`, and `TruckSim.exe`.

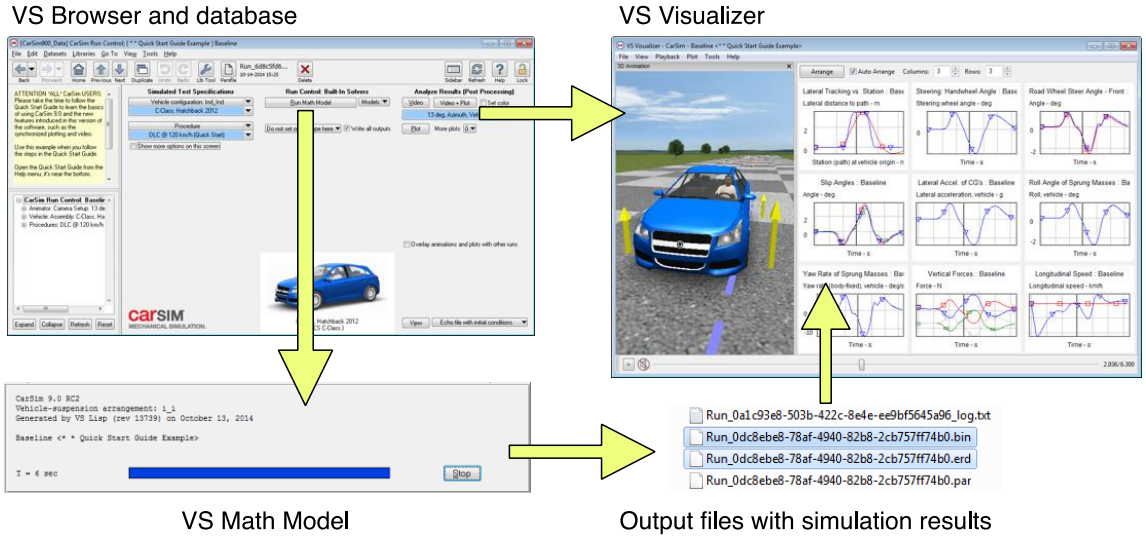

Figure 1. Main parts of a VehicleSim product.

This document applies to all products, and therefore the main program is called by the generic name *VS Browser*. The VS Browser provides the GUI for the products, it manages databases with simulation and animation data, and it coordinates the simulation calculations and visualization options.

The **Run Control** screen is the Home screen of the VS Browser.

Besides managing datasets in the **Run Control** library, it has buttons to construct and run VS Math Models and view results with VS Visualizer.

> **Note** Most figures in this document show datasets from CarSim. In most cases, the GUI controls and behavior described in this document also apply for BikeSim, SuspensionSim, and TruckSim.
>
> Settings that involve vehicle controls or position on a road are not included in the `SuspensionSim.exe` version because they are not applicable.

Basic Controls of the Run Control Screen

Each simulated test ("Run") in a database is usually configured in a unique dataset in the **Run Control** library and displayed on the **Run Control** screen. (An exception is that multiple models can run in parallel for co-simulation under the control of other software, as described in the Help document **Tools > Running with Parallel Solvers**.) Existing **Run Control** datasets can be selected from the list under the **Datasets** menu item. The **Run Control** datasets link a vehicle, a test procedure, and possibly other things such as a Simulink model or details that are unique for the run. A Run is created by making a new **Run Control** dataset and linking the elements to be used for the specific test.

The **Run Control** screen is divided into three regions (Figure 2).

1. **Simulated Test Specifications** — the left-hand region of the screen has links to the vehicle or system being simulated, test procedures, and optional datasets that may be used in the simulated run.
2. **Run Control** — the center region of the screen has a button to start a run and optional controls to set simulation parameters such as start time, time step, etc.
3. **Output & Post Processing** — the right-hand region of the screen has controls for viewing simulation results with VS Visualizer for 3D animation and plotting of simulation output variables.

Data in the left and center regions define the inputs and parameters that will be used in the computer model. Changes in these settings do not take effect until the next simulation is run.

The **Run Control** screen has several possible appearances depending on the settings you make. Figure 3 shows a typical display with only the basic controls visible.

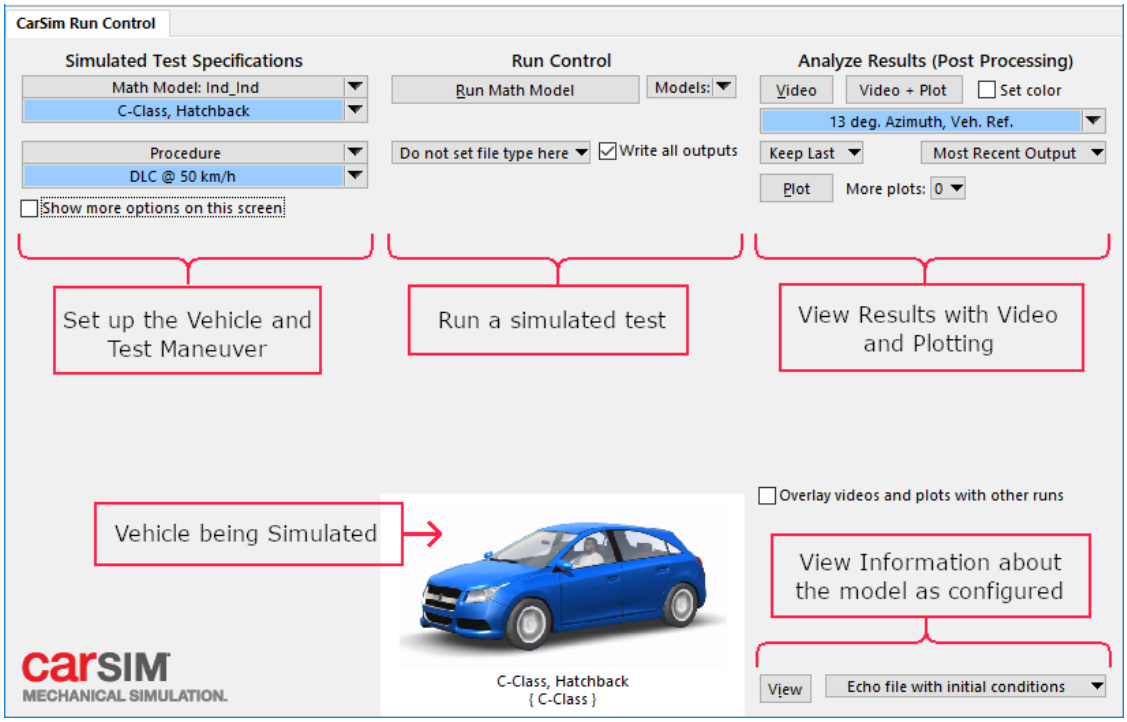

Figure 2. Layout of the Run Control screen.

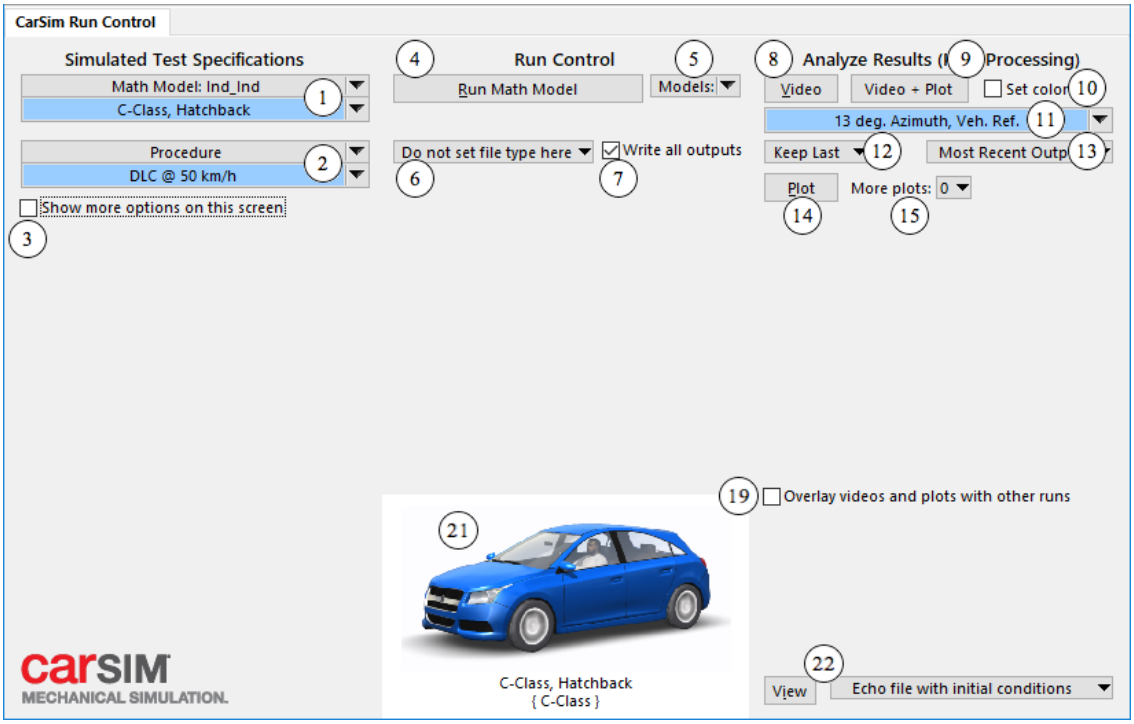

Figure 3. The Run Control screen (simplified view).

Vehicle and Procedure Data Links

(1) **Vehicle/System** link. This link normally goes to a dataset that describes the system being simulated. When a link is made to a dataset, it shows a code (e.g., `Ind_Ind`) used by the VS Browser to identify the math model that is matched to the selected vehicle data. (More information about the vehicle code is provided in the section **Vehicle Type** on page 27.)

In SuspensionSim, this link goes to a suspension model.

In the three vehicle simulation products, this link can also be set to a tire dataset, in which case the VS Tire Tester is run rather than a full vehicle simulation. (See **Help** > **Tire Models** for details on the tire tester.)

If the lower-middle part of the screen is not being used to show settings for advanced users, the graphic associated with the vehicle dataset is shown as a convenient indicator of which vehicle is being simulated (21). The image is set when the blue link (1) is made or modified.

(2) **Procedures** link. This is typically used for a link to a **Procedures** dataset. This type of dataset specifies details of a test procedure that includes driver controls, starting and stopping conditions, plots associated with the procedure, and possibly more information such as road and wind inputs, a sequence of events, and animation information.

(3) **Show more options on this screen**. Check this box to view more options for specifying data. When checked, more data links and several more checkboxes are displayed.

Run Control

(4) Run button(s). The appearance of this part of the screen depends on the type of **Models** link (5). There may be multiple buttons, and usually one of them will have the word "Run" in the name. For example, there is a single button **Run Math Model** shown if no link is made (Figure 3). Click it to run the simulated test.

When you click a run button, the VS Browser collects all of the specified data from linked datasets into a single file and uses a VS Solver library to construct a VS Math Model that runs the simulation. (More details about the process are provided in the section **Communication with the VS Math** Model on page 25.)

Different buttons might be shown if a **Models** link is made (5). For example, Figure 4 shows the buttons displayed when a link is made to a dataset from the **Models: Simulink** library.

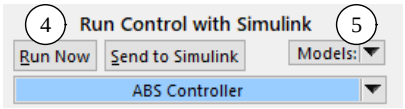

Figure 4. Run buttons when working with Simulink.

Details about alternative buttons requires some knowledge about how the other environments work and are not covered in this document. For information about extending the models or working with other simulation environments, please view the documentation accessed with the menu item: **Help** > **Model Extensions and RT > External Models and RT Systems**.

(5) **Models** drop-down control. The VS Browser includes a number of screens in the category **Models** that can be used to specify interfaces to RT HIL platforms, third-party software, and custom programs that provide model extensions. The drop-down list (Figure 5) is used to specify a **Models** library for the link.

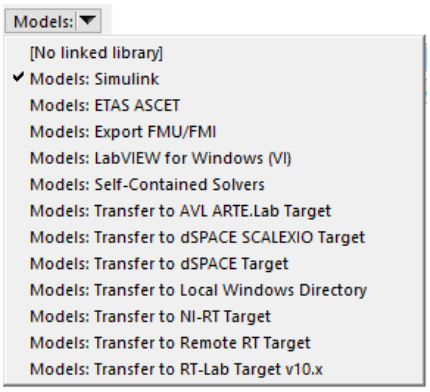

Figure 5. Drop-down list to choose a Model library option.

If a link is made, the linked dataset appears just under the run buttons (Figure 4). The run buttons are changed as needed to support the **Models** dataset, and the type of model is shown as a title above the run buttons. For example, when no link is made, the title is simply **Run Control** (Figure 3); when a link is made with a **Models: Simulink** dataset, the title is **Run Control with Simulink** (Figure 4).

Details about buttons that appear when a link is made to a **Models** dataset are contained in the documentation accessed with the menu item: **Help** > **Model Extensions and RT** > **External Models and RT Systems**.

> **Note** The list of model options depends on the product. For example, SuspensionSim is not used with real-time systems or Simulink, so the list is much shorter.

(6) Drop-down control to select format for the output file produced by the simulation. VS Math Models support four types of output files (Figure 6). The VS format supports both 32-bit and 64-bit data. The legacy ERD format is always 32-bit. The CSV format is pure text and can be read by Microsoft Excel and other spreadsheet programs.

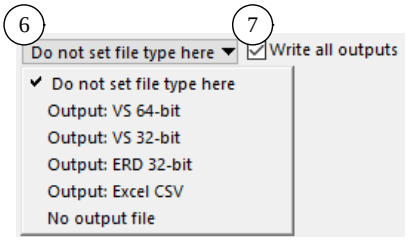

Figure 6. Drop-down menu of options for output file type.

This control allows you to set the file type for a single run (the default format is set in the **Preferences** library; see the **Tools** menu). It also provides an option to simulate without an

output file, supporting applications where post-processing is not planned such as long-duration real-time (RT) runs that may last several days.

(7) **Write all outputs** checkbox. VS Math Models can calculate thousands of output variables that are available for plotting and animation. This box is used to specify that all output variables are written to file. This is recommended if you are learning about the software or working with a new setup where you might experiment with alternate plots without needing to repeat the run.

If not checked, other methods are used to activate output variables such that they are written to file and available for post-processing visualization. All dataset files that will be provided to the VS Math Model are scanned for references to variables that are needed for plots or animations. These variables are automatically added to the output file. For routine work, this is the most efficient operation because the output file contains only the variables that will be plotted or used for animation.

When the box is unchecked, a library selection control is shown for advanced users. For routine work where all results are viewed in VS Visualizer, this link is not used. However, the control is available to create a blue link for a dataset that will customize the writing of outputs. When used, this will typically link to a dataset from the **I/O Channels: Write** (Figure 7). Datasets from the **I/O Channels: Write** library can specify other outputs of interest; they can also specify that the output file be automatically scanned to create additional files for use in Excel and/or MATLAB (e.g., CSV and MAT files). These options are described in the document accessed with the menu item: **Help > Model Extensions and RT > Import and Export Variables**.

Figure 7. Appearance when a link is made to a dataset specifying variables to write.

Note In SuspensionSim, all outputs are always written, so there is no checkbox for writing all outputs. The **Output Variables** dropdown is available for advanced users who are setting up automatic export to Excel and/or MATLAB.

Analyze Results (Post Processing)

The **Video** and **Plot** buttons on the right side of the screen show simulation results after a run is made, using results that have been saved in a simulation output file. If an output file does not exist (e.g., just after clicking the **Duplicate** button to create a new dataset, but before clicking the **Run** button), the buttons are dimmed. Once a simulation is made, the buttons are active and will remain so whenever the **Run Control** dataset is viewed.

(8) **Video** button. Click to view the vehicle's 3D motions as calculated by the VS Math Model with a simulated interactive video camera. This launches VS Visualizer with animation shape information associated with the selected vehicle (1) and environment (2), using time histories

written to an output file during the simulation, and video camera settings provided from the linked **Camera** dataset (11).

If the **Overlay video and plots with other runs** box is checked (19), VS Visualizer can overlay results from up to five additional simulations along with the simulation results associated with the dataset that is in view.

> **Notes** When using a driving simulator or real-time system, live animation during the run is obtained with different controls that only apply during the simulation, as described in the subsection **Live Animation** (page 16).

(9) **Video + Plot** button. Click to view plots and video associated with the run (Figure 8). This button sends all of the information to VS Visualizer associated with the **Video** button (8) plus all of the information associated with the **Plot** button (14).

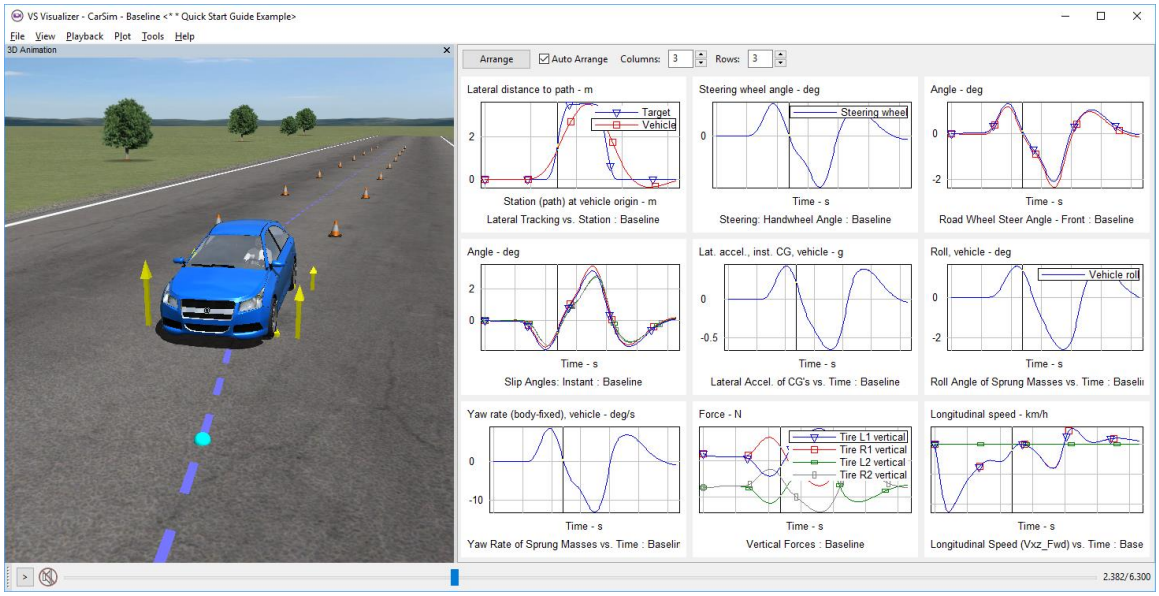

Figure 8. Analyze results after clicking the Video + Plot button.

In this mode, VS Visualizer has a video panel, a plot grid, and a time control panel. Working in this mode is explained in the Quick Start Guide for any VehicleSim product, and is covered in the VS Visualizer Reference Manual.

(10) **Set color** checkbox and color selector (Figure 9). Check this box to show a color selector. The color selector defines a *run color* that overrides the built-in colors of the vehicle and other 3D shapes such as tire force arrows in videos.

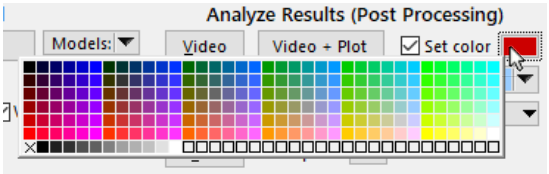

Figure 9. Color selector.

Note This option does not work in SuspensionSim. Colors are set for individual parts of the suspension.

The selected color does not affect the image shown on the run screen (21) because that image is part of the link to the vehicle dataset (1). The selected color will also not affect the animation unless the animator shapes for the vehicle have the corresponding box checked: **Use run color if there is one** (Figure 10).

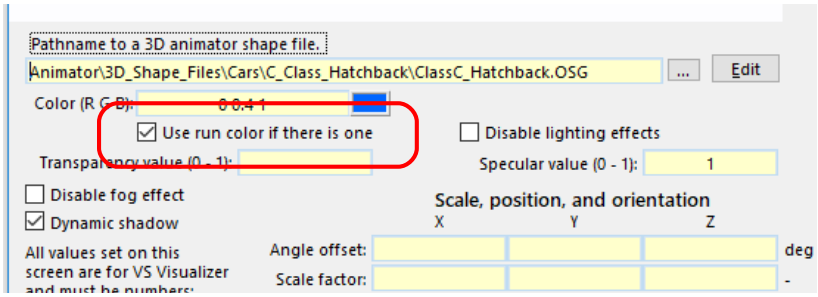

Figure 10. Checkbox on animation screens to "Use run color if there is one".

(11) Link to an **Animator: Camera Setup** dataset. The linked dataset sets the camera point of view used for the animation. If this link is not used, the **Video** (8) and **Video + Plot** (9) buttons will remain dimmed.

(12) Drop-down control to specify how any runs to keep. The normal setup when running on Windows without connecting to external tools is that when a new simulation is run, all files generated by the run overwrite any files that might already exist. In this case, only the last run is kept. This option is selected by the first item on the list: **Keep Last** (Figure 11).

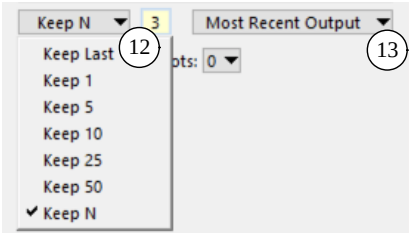

Figure 11. Options for keeping old runs.

All results based on a **Run Control** dataset are stored in a Results folder associated with the **Run Control** dataset. (Details of the locations of the files for a simulated test are provided in the section **Files for a Simulated Test**, page 25.)

The first option, **Keep Last**, uses the name `LastRun` for all files, with appropriate file extensions and suffixes (e.g., .vs, .par).

When working with driving simulators and setups where part of the model is defined outside the VS database, it may be convenient to make many simulations from the same **Run Control** setup. The other options in this drop-down control specify how many sets of simulation files will be kept. When the limit is reached, the oldest set of files will be deleted when a new run

is made. To specify a custom limit, choose **Keep N** and enter a number in the field to the right. E.g., the three most recent runs are saved for the setup shown in Figure 11.

(13) Drop-down control to choose which run to view. This control shows all of the runs that are currently available (Figure 12). If only the Last Run is being saved, then this control has only one item: **Most Recent Output**. Otherwise, it lists the runs using the root name for the output files.

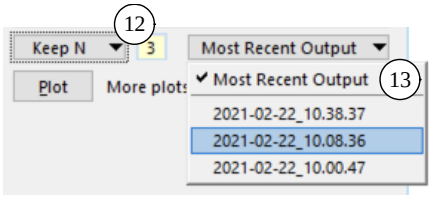

Figure 12. Select which run to view.

When results are retained for multiple simulations made from the same **Run Control** dataset, the files are named according to macros set in the Simfile. By default, the macros use the date and time, as shown in Figure 12. For information about custom naming conventions, please see the **Help** menu document **Reference Manuals > VS Math Models**.

(14) **Plot** button. Click to view plots of output variables calculated during the run. When you click the button, the browser scans all linked datasets for links to **Plot: Setup** datasets. These are typically specified in the **Procedures** dataset (2) and some other datasets that may specify variables to be plotted (e.g., sensor datasets, VS Command datasets). Plot setups can also be added on the **Run Control** screen if enabled by the drop-down list (15).

This information is provided to VS Visualizer which will show a view similar to that shown in Figure 8 (page 7), but without the video. When used in this manner, VS Visualizer does not load 3D animator asset files, reducing the time between clicking the button and showing the plots.

More Plots

(15) **More plots** drop-down control. Use this control to display up to eight **Plot: Setup** links on this screen (Figure 13).

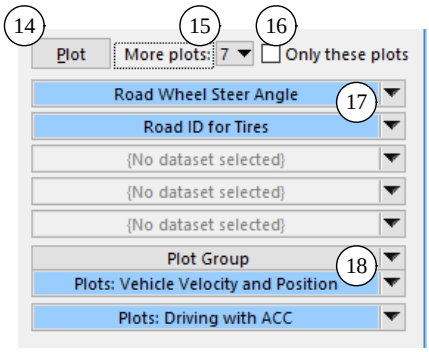

Figure 13. Additional controls shown for extra plots.

(16) **Only these plots** checkbox. When checked, only the plots specified on the **Run Control** screen will be shown by VS Visualizer when clicking the **Video + Plot** or **Plot** buttons. The option provides a quick means to show only plots of interest, reducing the time that might be spent zooming and moving plots in VS Visualizer.

When unchecked, all plot setups linked to the Procedures dataset and other linked datasets that might include plot links will also be included.

(17) The first five links always go to datasets in the **Plot: Setup** library. These are only visible if the **More plots** control (15) is set to a number greater than 0.

(18) Link number 6 includes a control to choose a library. It is provided to link to a Generic library dataset that might in turn include links to multiple **Plot: Setup** datasets. A popular choice is the **Generic VS Command** library, whose screen offers links to 16 **Plot: Setup** datasets. If visible, data links 7 and 8 use the same library that is chosen for link 6. These links are only visible if the **More Plots** control (15) is set to a number greater than 5.

Overlay Results from Other Runs

(19) **Overlay videos and plots with other runs** checkbox. VS Visualizer supports overlays from multiple runs for both video and plotting. If this box is checked (Figure 14), then five links are displayed that can be used to specify other runs (20).

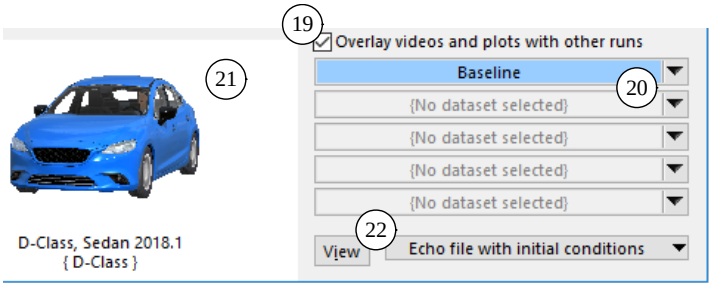

Figure 14. Overlaying results from multiple runs.

(20) Links to other **Run Control** datasets. If any other runs are specified, then videos and plots will show data from the simulation associated with the **Run Control** dataset in view, plus data from the other selected simulations.

If links are made and the box (19) is unchecked, the additional links are not cleared; they are hidden but not used. If the box is checked again, they reappear and will be fully active. This allows you to rapidly switch between overlay and non-overlay modes without needing to recreate the links each time.

The plotter will also handle output files that are not in the current VS database, such as files from other VS databases or test data that have been converted to the VS, ERD, or CSV format. The third item on the pull-down menu for each data link is the option **[Pick Output File]** (Figure 15); use this to select an output file that is not in the current database.

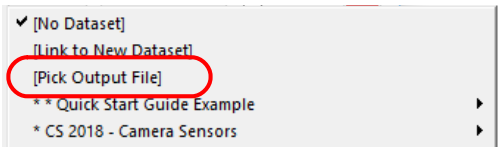

Figure 15. Top of menu to choose a dataset or output file for an overlay.

Model Information

21 Image of the simulated system taken from the link to the vehicle/system dataset 1. This image is shown if the selected vehicle/system has an associated image and if the lower-middle part of the screen is not being used to show settings for advanced users. The image is set when the blue link to the vehicle is made or modified.

22 **View** button and file type drop-down list. Click this button to view more information associated with the math model as configured for this simulation. The drop-down list has options to select different files that show how the model was set up for the simulation (Figure 16, for CarSim).

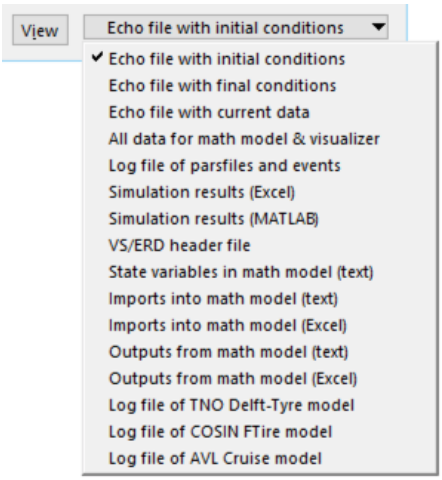

Figure 16. Drop-down list with files associated with current run that can be viewed.

These files provide extensive documentation about the model and simulation results (Table 1). The third item (**All data sent to math model & visualizer**) is a `Run_all.par` file based on the current dataset that is generated when a run is made or when the **Tools** menu command is selected: **Re-write every 'Run_all.par' file sent to the VS Math Model**. Most other files are generated by the VS Math Model either during the simulation, or when you click the **View** button.

These controls also give access to output files in Excel and/or MATLAB format if the conversion was specified in an output variables dataset 7 (Figure 7, page 6). (These files are created by VS Browser just after the run ends.)

Table 1. Files that can be viewed for a run dataset from CarSim or TruckSim.

File Type	Description
Echo file with initial conditions	Text files written by the VS Math Model at the start and end of a run listing all model parameters, tables, and VS Commands; End file also includes final values of all state variables.
Echo file with final conditions	
All data for math model & visualizer	Text file generated by VS Browser containing nearly all inputs for the VS Math Model or VS Visualizer, generated when clicking a **Run**, **Video**, or **Plot** button.
Log file of parsfiles and events	Text file generated by the VS Math Model as a run proceeds, reporting files read, Events, errors, etc.
Simulation results (Excel)	Spreadsheet made by the VS Math Model or by VS Browser immediately after the run ends.
Simulation results (MATLAB)	MATLAB arrays written to an M file by VS Browser immediately after the run ends.
VS/ERD header file	Text file written at start of run by the VS Math Model, listing all output variables; used by VS Visualizer to read the binary data file.
State variables in math model (text)	Text file written by the VS Math Model when the **View** button is clicked, listing all state variables for the model as configured, including those added with VS Commands.
Imports into math model (text)	File written by the VS Math Model when the **View** button is clicked, listing all import variables in the model as configured, including those added with VS Commands.
Imports into math model (Excel)	
Outputs from math model (text)	File written by the VS Math Model when the **View** button is clicked, listing all output variables in the model as configured, including those added with VS Commands .
Outputs from math model (Excel)	
Log file of TNO Delft-Tyre model	Text file generated by third-party extension to VS Math Model if the extension was active in the run.
Log file of COSIN Ftire model	
Log file of AVL Cruise model	

Notes The list of model-specific files depends on the product. Table 1 shows the options available in CarSim and TruckSim.

For BikeSim, the option for the Log file for AVL Cruise is not included.

For SuspensionSim, the options for state variables and Imports are not included, nor are the three Log files for third-party extensions. On the other hand, an additional item is available: **Parsfile written for VS math model**. This will select the Parsfile generated with the current simulation if that option was specified for the run.

More information is presented later about how the first four types files are created, including cases where the controls (12) and (13) are used to keep multiple simulation results for a single **Run Control** dataset (page 25)

Advanced Controls of the Run Screen

The **Run Control** screen has checkboxes that are used to display more controls. In addition, some controls are displayed if optional links are made. (E.g., linking to a Simulink dataset will display buttons for working with Simulink.)

Overriding Controls

The **Run Control** screen has several sections where linked datasets can override settings from the **Vehicle** and **Procedure** linked datasets.

Overriding Information in VS Math Models

The capability to override settings depends on the sequence in which pieces of information are provided to the VS Math Model. The VS Browser provides information shown on the **Run Control** screen as well as the information from linked datasets, in the order that the links appear in the Linked Library list in the sidebar on the left side of the screen. (If the sidebar is not in view, enable it with the **View** menu item **Show Sidebar**.)

Figure 17 shows an example dataset where the sidebar shows nine linked datasets. The nine blue fields are identified with circled numbers (1) - (9), with the number corresponding to the order in which the link appears in the sidebar. Note that the first (1) is a Simulink model dataset, and the last (9) is a dataset for output variables.

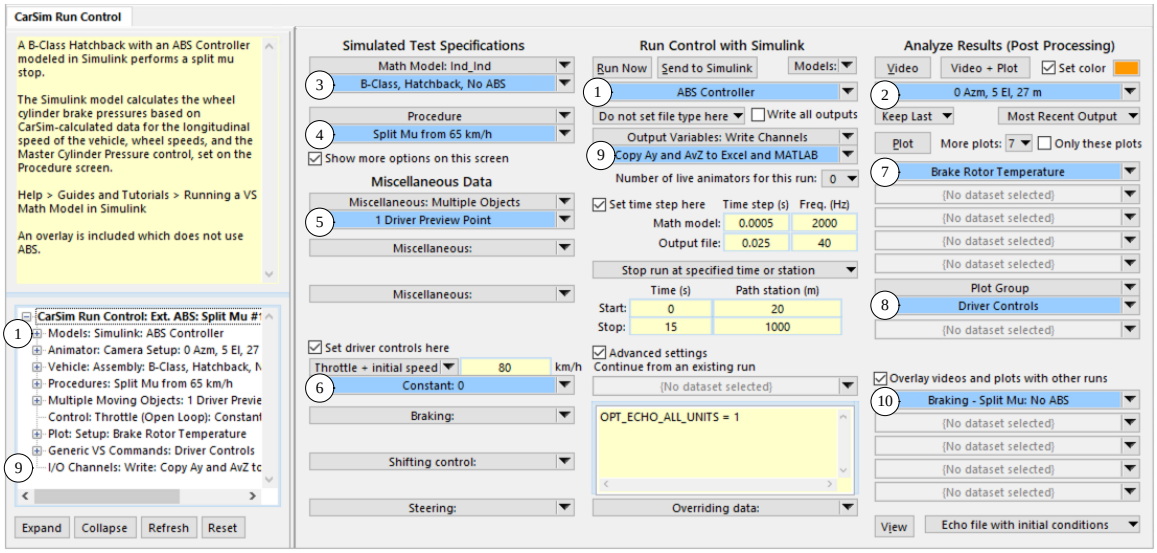

Figure 17. The sidebar shows the order in which the linked Parsfiles are read.

Another blue link identifies another **Run Control** dataset that will be used to generate overlay plots and/or videos (10). It is not an input to the VS Math Model, and does not appear in the sidebar.

When more than one setting or file exists for a given input, the last information that is read by the VS Math Model overrides any previous settings. For example, one speed control dataset can be specified in the linked **Procedures** library and another on the **Run Control** screen using the optional speed control link (6). You might use this feature to experiment with a change in speed without redefining an existing test from the **Procedures** library.

Adding Information for Visualization

The stream of data that is sent to the VS Math Model is sent to VS Visualizer when you click the **Video**, **Plot**, or **Video + Plot** button. In these cases, the effect is to *add* information rather than *replace* it. For example, animation information specified on the **Run Control** screen such as roads and vehicle shapes will not replace animation information from the **Procedures** screen; rather, information from both sources will be sent to VS Visualizer for visualization. Because road datasets often include related animation information, it is not always possible to override road geometry cleanly. If the intent is to use the same procedure with different roads, you should not specify any road information in the **Procedures** dataset. Rather, the road should be specified only from the **Run Control** screen.

Miscellaneous Data

Figure 3 (page 3) shows the screen with only basic controls; Figure 18 shows a more complex appearance when most of the advanced options are selected and the vehicle model is connected to a Simulink model.

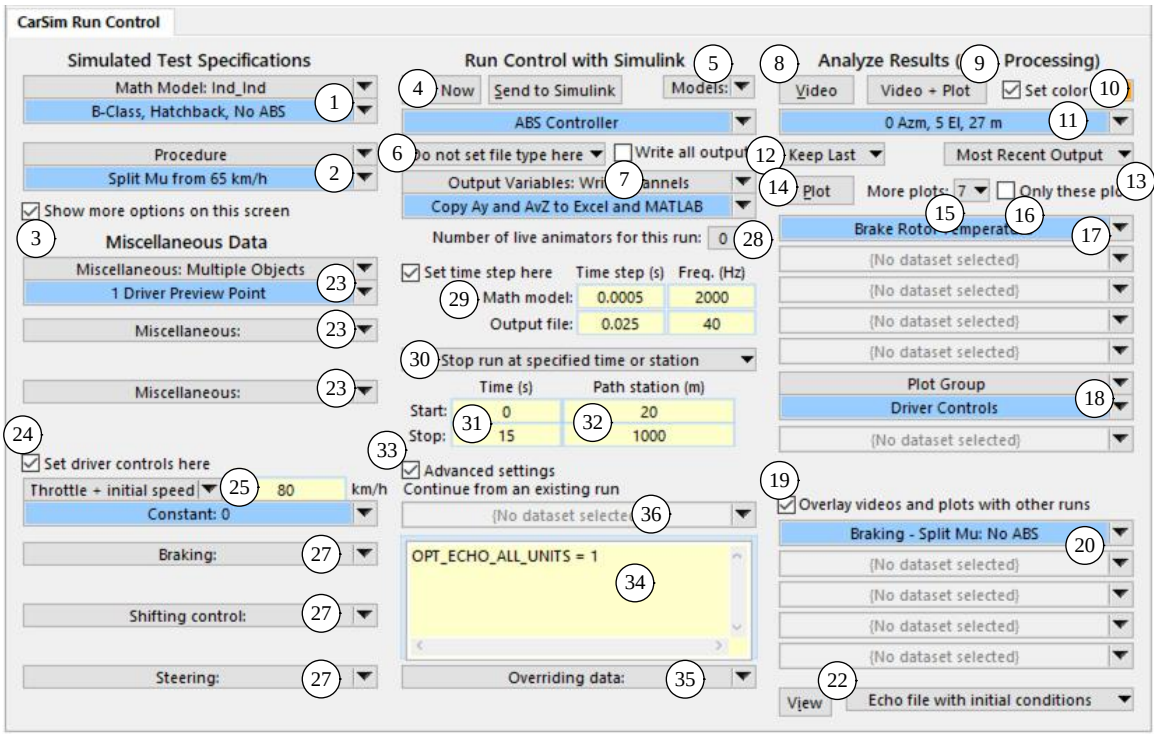

Figure 18. The Run Control screen with all controls visible.

All of the controls shown in Figure 3 are also present in Figure 18 and were described in the previous section. This remainder of this section (*Advanced Controls of the Run Screen*) describes the controls that were not visible in Figure 3.

(23) **Miscellaneous Data**. These links can be made to many of the VS Browser data screens. Some popular links are road conditions, VS Commands, and extra animation settings such as animated arrows that show tire forces.

These controls are visible only when **Show more options on this screen** (3) is checked.

Override Driver Controls

The following controls can override settings from the **Procedure** datasets in BikeSim, CarSim, and TruckSim. These options are not applicable in SuspensionSim.

(24) **Set driver controls here**. Check this box to view more options for specifying driver/rider control data for a run. The following controls are visible only when this box is checked. This control is visible only when **Show more options on this screen** (3) is checked.

(25) Speed / Acceleration option. This drop-down list has options for controlling the vehicle speed (Figure 19). If the selected option implies that more information is needed, then a yellow field and/or blue link are also shown.

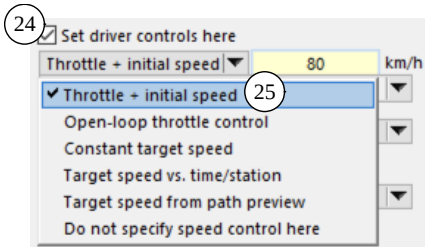

Figure 19. Options for speed control.

If the selected control involves target speed (i.e., a closed-loop controller), then a link is shown for connecting to a dataset in the appropriate library. If an open-loop control is selected, then a yellow field is used to set the initial speed and a link is shown for connecting to an open-loop throttle control dataset (Figure 19). If a constant target speed is selected, then a yellow field is shown for specifying that speed, along with a checkbox for accounting for engine braking in the controller (26) (Figure 20).

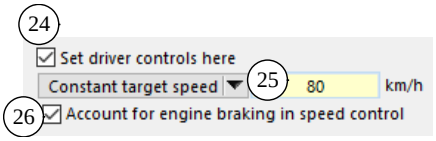

Figure 20. Extra controls shown for constant target speed setting.

Note The vehicle model has a closed-loop speed controller that can work with a target speed defined as a function of both time and station, or by previewing the target path and determining speed based on acceleration limits. The option shown in Figure 20 is a quick way to set the target speed to a constant value.

Be aware that the controller can be tuned using feedback parameters, and that the brake system can also be included (e.g., if trying to maintain a constant speed going down a hill). To access the full capabilities of the speed controller, please select one of the other speed control options.

(27) Optional links for braking, shifting, and steering. These links to various driver control options can be set to override any settings from the linked **Procedures** dataset.

Live Animation

VS Math Models can run in real time (RT) to test hardware in the loop (HIL), or to allow you to drive the vehicle in a driving simulator (DS) with appropriate control hardware (steering wheel, pedals, etc.).

Normally, the simulations run faster than real time. Synchronization between the simulation clock and real time is achieved by waiting each time step in the solver until the specified time has elapsed on a real-time clock (e.g., 0.0005s).

With Windows OS, the VS Math Model can synchronize the simulation time with real clock time (as provided by Windows) if you are running with an optional DS license. The VS Solver libraries are also available to run VS Math Models on RT systems such as dSPACE, Linux RT, QNX, and others if more precise synchronization is required.

If running with Simulink, you can view live animation by making use of the S-Function with video. Although it is live, the simulation is not slowed down to real-time. This means the vehicle behavior is displayed at the rapid simulation speed.

When running a DS or RT version of the software, the VS Math Models send motion information to one or more animator programs, which in turn provide 3D animation of vehicle motions "live" as the run proceeds.

> **Note** The **Video** button (8) provides post-processing visualization of simulated behavior; this button is not used for live animation.

(28) Pull-down control for selecting the number of live animators. If you want to disable live animation, then set the number to zero (Figure 18). If the selected number is greater than one, then the animators must be activated externally with display preferences for each. If the selected number is one, then two more controls appear to control the starting and stopping of the animator (Figure 21, Figure 22).

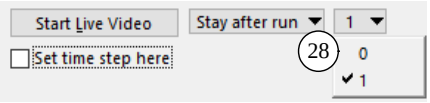

Figure 21. Pull-down control for selecting the number of live animators.

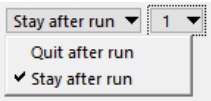

Figure 22. Controls when there is one live animator.

The **Start Live Video** button can be used to manually start VS Visualizer before starting the run. If not used, VS Visualizer will be launched automatically. The drop-down list has options for either quitting VS Visualizer at the end of the run or leaving it open for use in subsequent runs.

These controls are visible only when a link is made to a model dataset that also supports live animation. Examples datasets that support live animation are almost any model for an RT HIL systems (dSPACE, VeriStand, etc.), or a Simulink model that uses the live video S-Function.

Alert	When the **Run Control** screen is set up to support live animation (i.e. the number of animators is not zero), the Parsfile for the **Run Control** dataset sets the system parameter `OPT_BUFFER_WRITE` to 1 (the default is 0), specifying that all output variables that will be written to file are stored in a memory buffer, and are not written to file until the run ends. If there are many output variables (e.g., the **Write all outputs** box (7) is checked) and the output frequency is high, then the amount of memory needed to store all outputs can be large, and there may be a noticeable delay at the end of a run before control returns to the GUI.

Set Time Steps

The VS Math Model calculates variables as the run proceeds by using numerical integration (see *VS Math Models Manual* and the Technical Memo *Numerical Integration in VS Math Models* for details).

Outputs can be written at some multiple of the model time step to reduce the size of the output files. The output variables can be written into a binary file that is associated with a VS or ERD file that has header information needed to read the binary file. They can also be written into a CSV text file that is compatible with Excel and other external tools.

The default time steps for the numerical integration and output file can be specified in three places in the database; all have the same appearance as seen in Figure 18 (29).

1. Time step default values are set on the **Preferences** screen (to view this screen, select **Preferences** from the **Tools** menu). These will be used if no other values are specified in the datasets used to set up the simulation.
2. Time step values can be set on all **Models** screens such as the one used to specify a Simulink model. This can be necessary when using an external model that requires a specific time step that is not necessarily standard for other runs. The time steps specified in a linked **Models** dataset will override the values from the **Preferences** dataset.
3. Time step values can be set on the **Run Control** screen. The time steps specified here will override any other values.

(29) **Set time step here**. Check this box to view data fields for setting the time steps for the numerical integration and output file (`TSTEP` and `TSTEP_WRITE`, respectively).

This control and the adjacent yellow field are available only when **Show more options on this screen** (3) is checked.

The time steps can be specified using either seconds or frequency. The browser automatically maintains the correct inverse relationship between these two values. If you modify the time

step, the frequency is calculated and updated; if you modify the frequency, the time step is calculated and updated.

Note The time step `TSTEP` is passed to the math model, along with the integer ratio `IPRINT`. The output time step `TSTEP_WRITE` cannot be set directly; it is calculated by the VS Math Model as: `TSTEP_WRITE = IPRINT*TSTEP`.

As with the model time step, the browser automatically maintains the correct inverse relationship between time step and frequency for the output file. It also guarantees that these numbers are multiples of the internal time step used in the VS math model.

Alert Most parameters in the VS Math Model can be specified with numbers or formulas. However, the four fields (29) that specify the time step for the simulation and output file are also used to automatically calculate numbers, as described above. Because of the automatic calculations, each of these fields must contain a numerical value when they are visible. Blank fields and formulas are not supported.

Override Time and Distance Settings

Starting and stopping of the run is typically controlled by settings from the linked **Procedures** dataset. Some of the settings can be overridden from this screen.

The controls below are hidden unless the **Show more options on this screen** box (3) is checked.

Each simulation run starts at a specified time (typically zero) and runs until a stop condition is reached. The initial location of the vehicle is usually set using a location on a path (station). The direction of travel along the path is also set. The controls described here ((30) - (32)) set the parameters in the math model involving these start and stop conditions.

(30) Drop-down list of options for overriding time and station. The list has five options (Figure 23). Based on this selection, appropriate controls are displayed below ((31) and (32)).

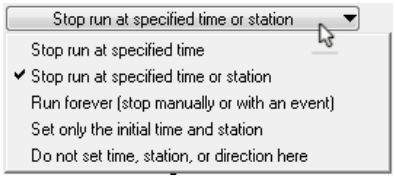

Figure 23. Built-in options for stopping a run.

The first three options in the list set a value for the system parameter `OPT_STOP`: a value of 0 (stop at specified time), +1 (stop at time or station), or -1 (run forever).

The fourth option (**Set only the initial time and station**) shows the start time and station fields only ((31) and (32)).

The last option (**Do not override time, station, or direction here**) hides the related settings (Figure 26, page 21). In this case, the settings from the linked **Procedures** dataset are used.

(31) Time starting and stopping values (keywords = `TSTART`, `TSTOP`). When a run is started, the time is set to the value specified here. It is normally zero, but other values can be used if desired.

> **Note** Some real-time systems require that the start time always be zero. If you are running with one of these systems, the option to set the start time is automatically disabled.

A stopping time is specified unless the option selected from the above drop-down list is **Run forever**. In this case the field for setting stop time is hidden, and a value of `TSTOP` = -1 is automatically written in support of external programs such as Simulink that use a negative value to indicate the simulation should run until stopped by some custom condition.

> **Note** The run will stop under several conditions, including user-defined conditions specified with *Events* or other VS Commands. More information about these options is available in the documentation for the **Events** screen and in *VS Commands Manual*.

(32) Location of the vehicle on a path, defined by station using the path identified by the parameter `PATH_ID_DM`. If you do not want to locate the vehicle using station on the designated path, use the option **Do not set time, station, or direction here**, and use the linked **Procedures** dataset to specify the initial location.

The starting location is shown for all options where settings can be made about the position (keyword = `SSTART`). If the choice made in the above control (30) is to stop at a specified station, then a field is shown for the stopping station (Figure 18, keyword = `SSTOP`). Otherwise, a drop-down control is shown to specify the direction of travel (Figure 24, keyword = `OPT_DIRECTION`).

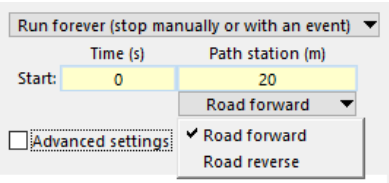

Figure 24. Specify the direction of travel if station is not used as a stop condition.

The direction **Road forward** means the vehicle is oriented in the direction of increasing station (`OPT_DIRECTION` = 1); the direction **Road reverse** means the vehicle travels in the direction of decreasing station (`OPT_DIRECTION` = -1).

> **Notes** If station is defined using a path that is looped, then the length of the loop limits the range of valid station values. The concept of using a value of station to stop the run does not always work with looped paths. In these

cases, *Events* or other VS Commands are typically used to stop the run. (For more information, please see *VS Commands Manual*.)

As noted earlier, controls involving road position and direction are not applicable in SuspensionSim.

If either the fourth or fifth options are selected (Figure 23), then both the field for stop station and the drop-down control for direction are hidden; in these two cases, nothing is written into the parsfile with the keyword `OPT_DIRECTION` or `SSTOP`.

Overriding Miscellaneous Data

The **Run Control** screen has two controls for specifying miscellaneous data that are read by the VS Math Model after everything else. If they involve properties that were set elsewhere, this data will override them because this field and any of its contents are read last.

These controls are hidden unless two checkboxes are checked: **Show more options on this screen** (3) is checked and **Advanced settings** (33).

(33) **Advanced settings** checkbox. This checkbox replaces the current vehicle graphic with three controls (34), (35), and (36) (Figure 18, Figure 25). This must be checked in order to override settings for quick studies or diagnoses, or to continue a run.

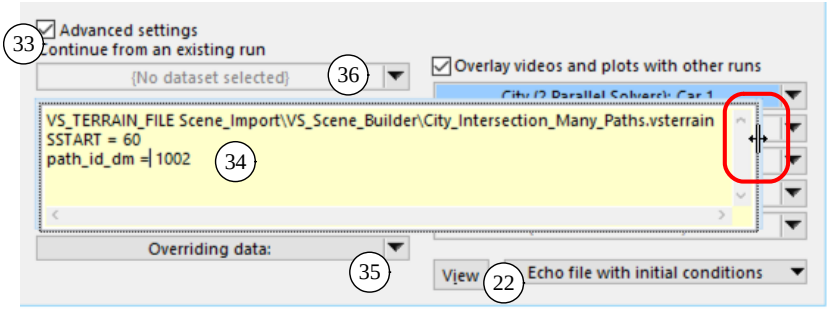

Figure 25. Use a splitter control to widen the miscellaneous yellow field.

(34) Overriding miscellaneous data field. This is a field where any parameter can be set. The format for each line of text should consist of a parameter name, then a blank space and/or '=' sign, and then the parameter value or formula. This field can contain any text that would be recognized by the VS Math Model. Use the **View** button (22) to view a file that lists of all the keywords and parameters that can be specified.

This field is also a convenient place to locate VS Commands to define simple model changes, such as creating a new output variable. See *VS Commands Manual* for more information about this capability. In case there are VS Commands that are too long to view, you can use the edge splitter controls to make the field wider and taller (Figure 25).

The contents of this field are sent to the math model just before the **Overriding Data** link (35). Therefore, it can also be used to specify a context for overriding data. For example, you might use it to specify the axle and side for a tire dataset using the parameters `IAXLE` and

ISIDE. (See *VS Math Models Manual* for details about indexed parameters used for repeated components such as axles and tires.)

(35) **Overriding data** link. This link can be used with most of the screens in the VS Browser. Linked data will be read after all other data on the screen and will override any previously specified values.

Continue from an Existing Run

The VS Math Model writes an Echo file before starting a simulation and it writes another — called an End file — when the simulation terminates. The End file contains all parameters and tables, plus the values of all state variables in the math model. The Echo and End files both are written in Parsfile format and can be used directly as input to a VS Math Model.

The **Run Control** screen has two controls that support the option of continuing an existing simulation run (Figure 26). These controls are invisible unless both the **Show more options on this screen** box (3) and the **Advanced settings** box (33) are checked.

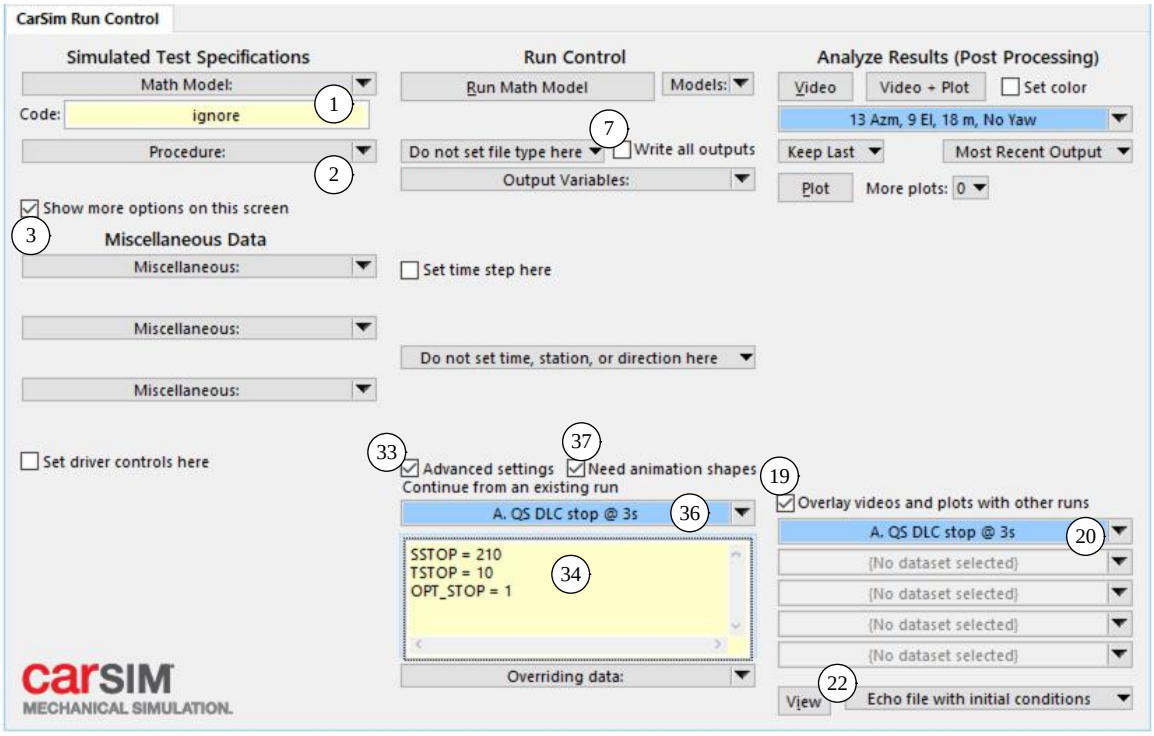

Figure 26. Run Control dataset for Part B of an example two-part simulation.

(36) **Continue from an existing run** link. This link can be used to start a new run that continues from the final conditions of an existing run. If a new run is made, all model parameters and inputs from the old run will be read by the VS Math Model before any of the inputs from this data screen are read, such that the same vehicle, procedure, and other conditions are repeated. The final conditions from the other run are also read to fully define the state of the math model at the start of the new run to match the state at the end of the previous run.

(37) **Need animation shapes** checkbox. This checkbox is only visible if a link is made to an existing **Run Control** dataset (36). When this box is checked, it will use all shapes used in the previous run, including the vehicle, road, and sky.

If the new run already has animation data via a vehicle link and road dataset, then this box should be left unchecked.

Typical setup for a continuation run

Consider the example for a continuation run shown in Figure 26.

There are many possible reasons that you might not want the VS Math Model to read the same input data twice. For this example, the links to the vehicle (1) and procedure (2) were broken, as shown. When the vehicle link is cleared, a yellow field is shown to specify the model code (1). In order to get the code from the existing reference run, put the keyword `IGNORE` in the yellow field.

If there is no **Vehicle** dataset, then there is also no animation information. When a link is made to an existing run, a checkbox is shown for the option to use animation data from the existing run (37).

Because the reference End file dataset is for a run that has already been completed, the original stop conditions will not be used. The VS Math Model puts a start time in the End file that matches the stop time, and estimates a new stop time based on the duration of the reference run. In this example, the stop conditions are set to match those of the original Quick Start Guide example: stop at T = 10s (`TSTOP` = 10) or Station = 210m (`SSTOP` = 210), whichever occur first (`OPT_STOP` = 1). These conditions are set in the miscellaneous field (34).

With a normal run setup, the VS Browser scans the input Parsfiles for plot setups and visualizer reference frames, and activates the necessary output variables for writing to output, using the `WRITE` command. The setup in Figure 26 does not include any plots or links for animation, so there is no automatic activation of output variables for writing to file.

The normal method for activating output variables for writing to file is to set up the reference run such that the End file activates the output variables. For this example, Figure 27 shows the reference run dataset.

> **Note** You can always view the **Run Control** dataset for a reference run by clicking on the blue link for the **Continue from an existing run** link (36) (Figure 26).

In the reference run, the parameter `OPT_ECHO_WRITE` is set to 1, which causes the VS Math Model to include `WRITE` commands for the output variables in the Echo and End files (Figure 28).

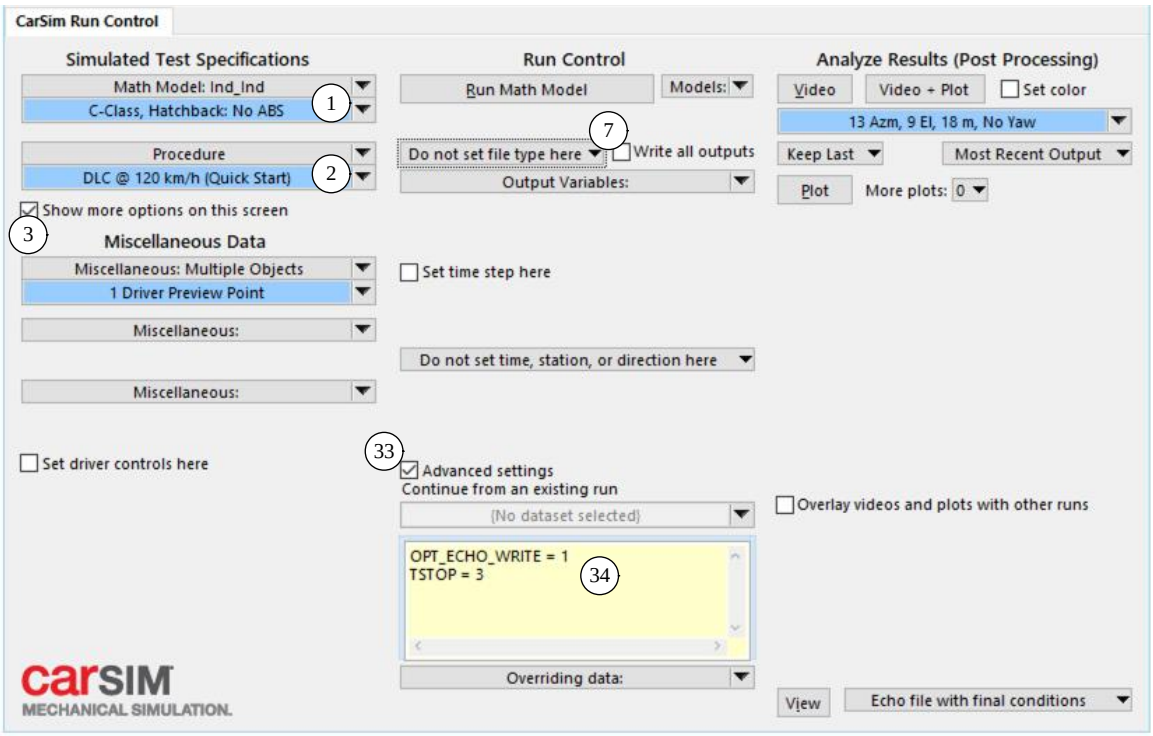

Figure 27. Run Control dataset for Part A of an example two-part simulation.

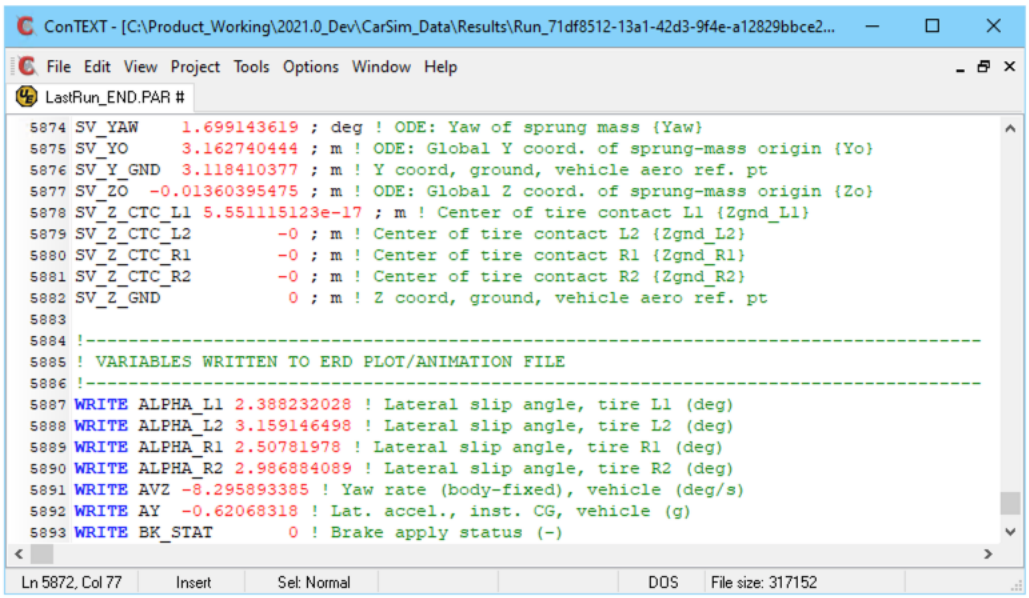

Figure 28. Part of End file with state variables and WRITE commands for output variables.

Another option is to check the box to **Write all outputs** (7). If this box is checked, the `OPT_ECHO_WRITE` parameter should not be specified (the default value is 0).

The miscellaneous field is used in this example to specify a stop time with `TSTOP` = 3.

Figure 29 shows plots defined in the original run for the examples used for Figure 26 and Figure 27. The original run ended at 3s; the continuation started at 3s. The two overlaid plots show perfect continuity, with the only visible indication of the switch being the colors of the lines in the plots.

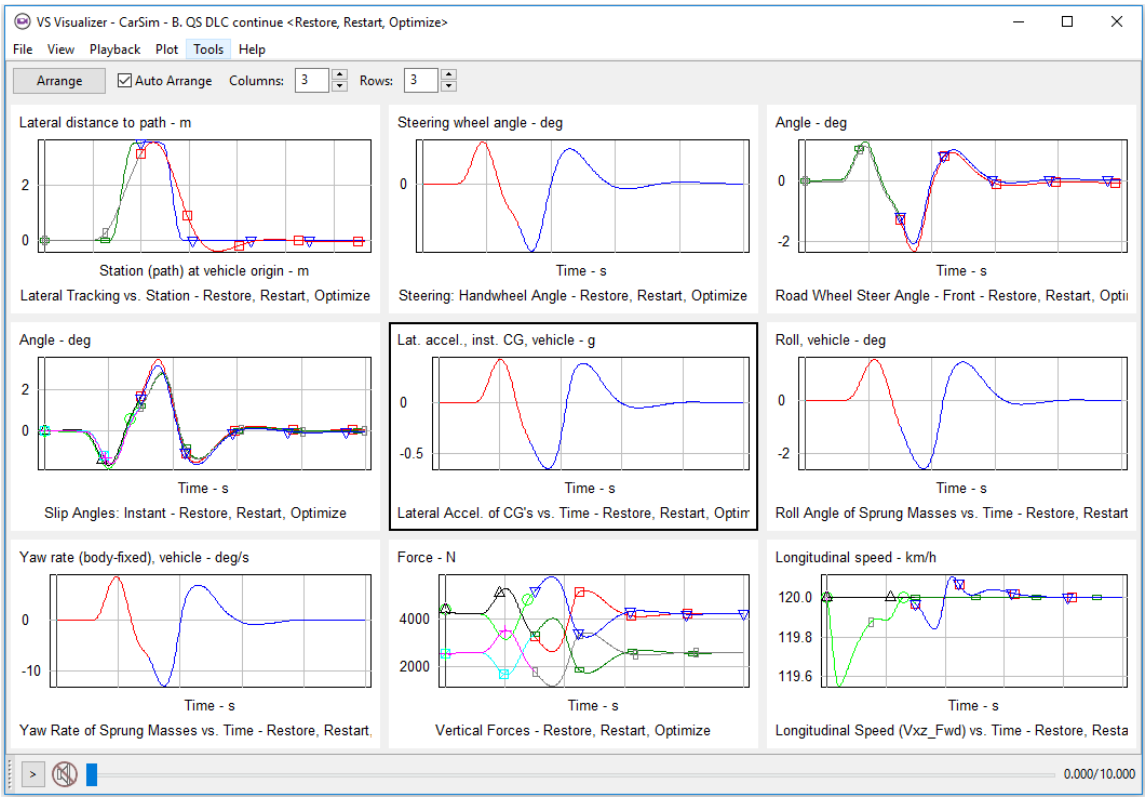

Figure 29. Plot for two runs involving a continuation.

Continuing a run using Simulink

If you are running a model from Simulink, then you need to keep a link to a dataset from the **Models: Simulink** library. The linked dataset specifies the Simulink project (MDL or SLX file), which is needed by the VS Browser to properly connect. The linked dataset will typically include links to I/O datasets that add variables to the Import and Export arrays that are shared with Simulink. The same information is also included in the End file from the original run. If the links are used for the continuation run, the variables used for Import and Export will be specified twice, causing a discrepancy in the sizes of the arrays between the VS Math Model and the Simulink model.

The solution is fairly simple. For the continuation run, use the Copy and Link option to duplicate the linked dataset from the **Models: Simulink** library. Go to that new dataset and break the links for the Import and Export datasets. The new **Models: Simulink** dataset will no longer conflict with the `IMPORT` and `EXPORT` specifications from the End file generated by the original simulation.

Communication with the VS Math Model

When you click a button to make a simulation run, the VS Browser uses a VS Solver library to construct a VS Math Model that runs a simulated test and generates output files with time histories of variables of interest. In doing so, the VS Browser takes into account the type of vehicle that will be simulated, along with the possible use of third-party simulation tools that might be used to provide automation or extend the vehicle math models.

Files for a Simulated Test

All libraries in the VS Browser use a custom screen to manage Parsfile datasets that are located in a folder associated with the library. For example, all **Run Control** datasets are located in a database folder named `Runs`. However, the **Run Control** screen is unique in the VS Browser by being the only screen that creates and manages another folder. Each dataset is linked to a folder that is contained in the `Results` folder for the database. The folder created for the run has the same name as the Parsfile, but without the extension `.par`, as will be shown below.

The VS Browser has a menu command to view the Results folder for the current **Run Control** dataset (Figure 30).

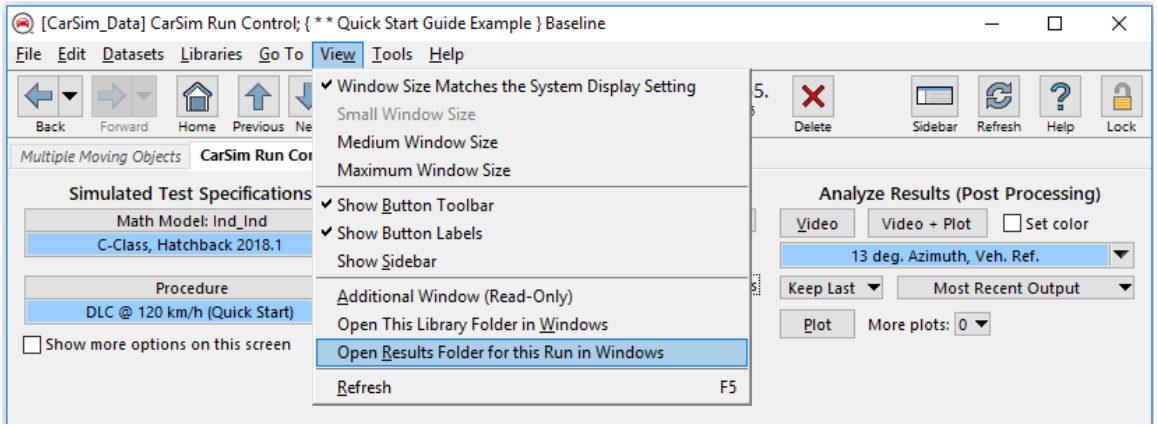

Figure 30. Menu command to view the Results folder for a Run Control dataset.

Figure 31 shows the Results folder for a typical run. The new folder is itself located in a folder named `Results`. It has the name `Run_`*uuid*, where *uuid* is the universally unique ID number used to name the **Run Control** dataset. For this example, the uuid for the dataset is 6cb07365-… (1), which is also part of the folder name (2).

When you click a button to run a simulation, several actions are taken:

1. If this is the first run made from a **Run Control** dataset, the VS Browser creates the new Results folder (2) that will contain files read and written by a VS Math Model.

2. When a simulation is run, the VS Browser creates two new files:

   a. It collects information from all linked datasets, located in the many folders in the database, and creates a single expanded Parsfile named `Run_all.par` with the information (4). This file is written into the folder created in step 1.

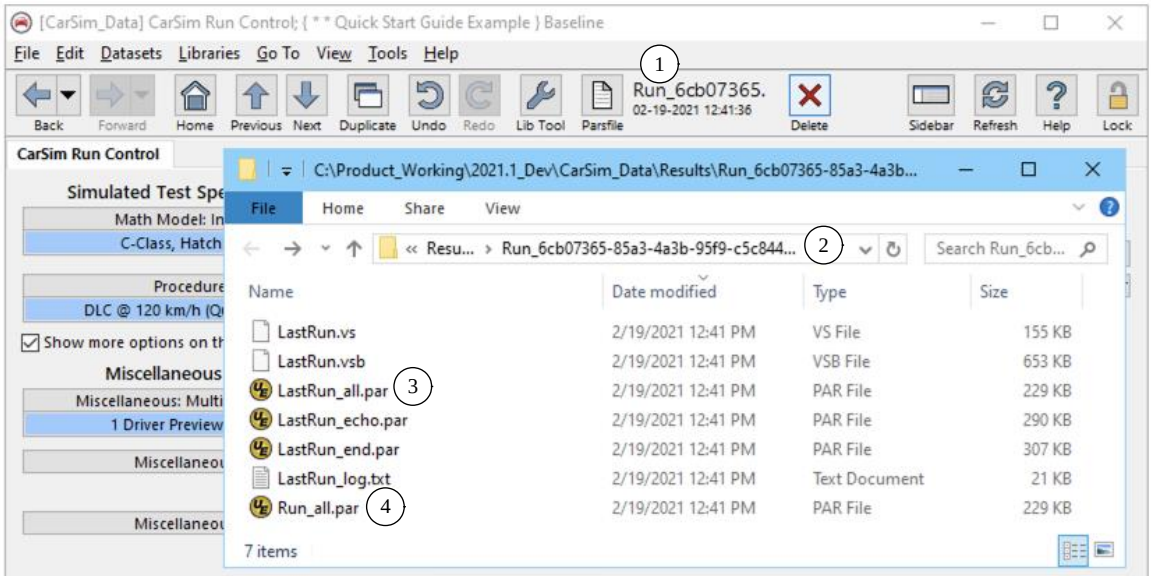

Figure 31. Each Run Control dataset is linked to a folder with simulation results.

b. It also writes a simulation control file (Simfile) in the root folder of the database. The simfile contains a vehicle code that identifies the model configuration, the pathname of the input file created in step 2a, and other information.

3. The VS Browser loads the VS Solver DLL, commands it to construct and run a VS Math Model using information obtained from the Simfile and the `Run_all.par` file, and when the run is complete, unloads the DLL file. All output files generated by the VS Math Model are located in the Results folder created in step 1. In this example, all of the new files have names that begin with `LastRun`.

4. A copy of the `Run_all.par` file is made with a name consistent with six other files that were generated by the VS Math Model, e.g., `LastRun_all.par` (3).

The **Run Control** screen has controls to keep results from recent simulations, as described earlier (Figure 11, page 8). When the option selected is to keep only the last run, the text `LastRun` is used as a file name or prefix, as shown in Figure 31.

When results are kept from multiple simulations, the files are named according to macros set in the Simfile. By default, the macros use the date and time. For example, Figure 32 shows the Results folder for the case of keeping the three most recent sets of simulation files (1).

There are always six files for each run, with names identifying the run ((3), (4), and (5)), plus the file `Run_all.par` that was made for the last run (6). The file `Run_all.par` is always created for a new run, and then copied with a name to match the other six files.

The `all.par` file whose name matches the other files is re-written when VS Visualizer is activated by clicking the **Video**, **Plot**, or **Video + Plot** button. This is done to support the choice among multiple runs of which to view (2) when more than one runs is kept (1).

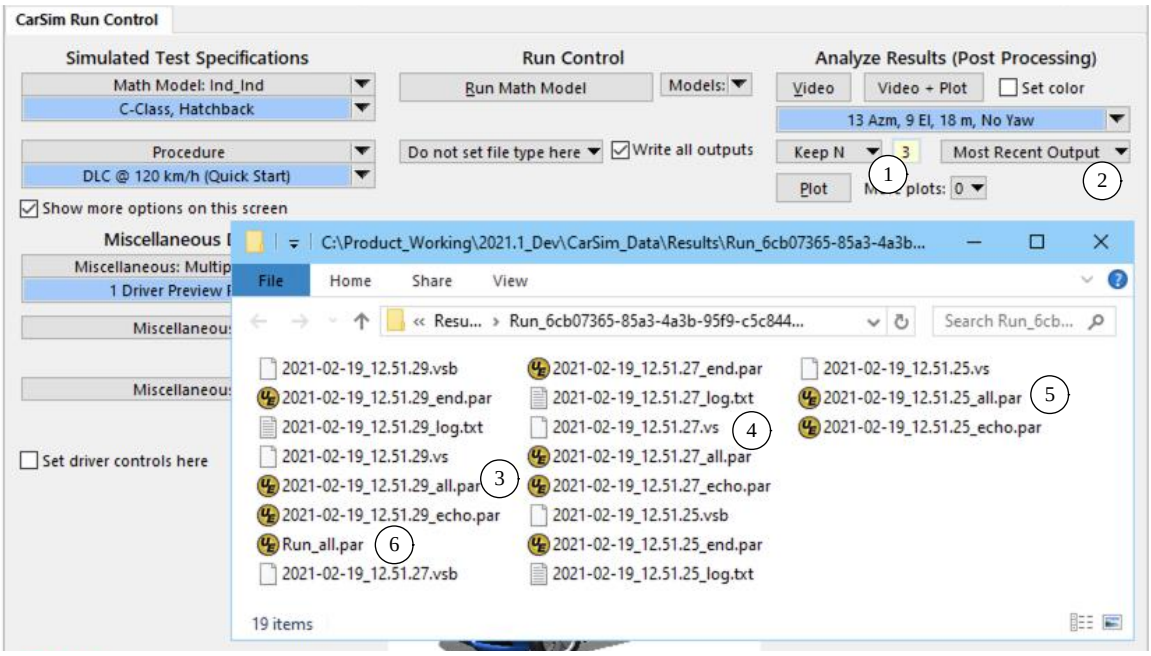

Figure 32. Results folder when three simulations are kept.

To summarize, all settings made on the **Run Control** screen are stored in a Parsfile in the `Runs` folder, in the same manner as all other screens in the VS Browser. All files used by the VS Math Model (both input and output) are located in a folder that is in turn located within the database `Results` folder, set to have a name that matches the **Run Control** Parsfile (`Run_`*uuid*).

Vehicle Type

A VS product includes VS Solver libraries with modules optimized for the 3D multibody model used to represent a range of vehicles. VS products also support a tire tester in which a simulated test uses a special dataset from the **Tire Tester** library. In this case, you link to a tire dataset rather than a vehicle dataset (please see the **Tire Models** documentation for details about this option).

The layout of the VS Math Model needed for the simulation run is normally obtained from the link made to a vehicle dataset. In addition to providing data that describe the vehicle or tire being simulated, this link has a second purpose—to identify the specific model to be used in the simulation. The vehicle type is shown in the right-hand part of the top link after the words "Vehicle configuration." For example, Figure 3 (page 3) shows a setup where the vehicle configuration is `Ind_Ind` (1). The vehicle model layouts are shown in the Echo files near the top, with the keyword `MODEL_LAYOUT`.

Note	CarSim and TruckSim share many suspension options. Because TruckSim models can involve many axles and trailers, the suspension types have always been represented with a single character. The `MODEL_LAYOUT` description uses single-character codes for both CarSim and TruckSim. For example, the configuration `Ind_Ind` shown for a CarSim vehicle is represented with the `MODEL_LAYOUT` description `I_I` in the Echo file.

In CarSim, TruckSim, and BikeSim, all vehicle configurations are handled with a single VS Solver: `carsim_32.dll`, `trucksim_32.dll`, or `bikesim_32.dll` when running with 32-bit applications, and `carsim_64.dll`, `trucksim_64.dll`, or `bikesim_64.dll` when running with 64-bit applications. Before the solver reads the `Run_all.par` file, it uses the vehicle code to determine the configuration of the vehicle in terms of what parameter and output variable names should be installed in its keyword database to read the Parsfile.

For more detail about how the VS Math Models are operated, please see *VS Math Models Manual*.