

CarSim 8.1 New Features

- Introducing the VehicleSim Visualizer: VsV..... 1
- Improvements in the Math Models..... 2
 - Continuously Variable Transmission (CVT) 2
 - Engine Mounts 2
 - AVL Cruise Interface 2
 - Driver Model Improvements..... 3
 - Reference Paths for Road Geometry and Controllers..... 3
 - Import and Output Variables..... 4
 - Configurable Table Functions..... 4
 - Tire Model Improvement..... 5
 - Miscellaneous Architecture Improvements 5
- Improvements in the Database 6
 - Organization of Example Datasets 6
 - Procedures..... 6
 - Vehicles and Components 7
 - Animation Data 9
 - Handling Course..... 9
- Improvements in the GUI 9
- Compatibility with Older CarSim Versions 10
 - Auxiliary Parameters and Variables Removed 10
 - Generic Tables Removed..... 11
 - Import Variable IMP_PWR_EXT_ENGING_AV Renamed 11
 - Y vs. Axle Roll Solid-Axle Kinematics (Keyword and Definition)..... 11
 - Reference Paths for Road Geometry and Controllers..... 11
 - Position of Yellow Field Changed in Procedures Screen 13
 - Road-Based Variables Removed..... 14
 - Hidden Parameter IROAD renamed as IDZ_ROAD..... 14
 - TSTART_BRAKES Parameter Redefined 14
 - Camber Output Variables 15
 - Steering Output Variables..... 15
- Bug Fixes and Errata 15

CarSim 8.1 provides major new features in the math models, an improved database, and introduces major improvements in visualization for all users.

Introducing the VehicleSim Visualizer: VsV

Animation in CarSim has been provided for over a decade by the program SurfAnim. In support of advanced driving simulators, Mechanical Simulation is pleased to introduce a new program called VehicleSim Visualizer (VsV). VsV makes use of modern graphical processing units (GPU) and shape file formats. It provides advanced lighting (true shadows, multiple light sources, reflected highlights), highly efficient rendering, rear-view mirrors, and advanced techniques taken from video game technology.

VsV includes plotting that is synchronized with animation. It supports commands similar to VS commands for defining new variables with algebraic equations that can be plotted or used to control motions of animated shapes. The user interface supports many interactive features to support artists working with new 3D shapes.

Future development of VsV includes more work in automating plots and spectral analysis, with the intent of replacing the CarSim plotter WinEP and the spectrum analyzer tool. In this initial release, the priority in VsV development has been to support driving simulators. Although the default animator is still SurfAnim, VsV is provided as an optional animation tool that can be used for evaluation.

To try VsV, go to the **Preferences** screen, view the dataset named **Defaults Settings VsV**, then return to the Home **Run Control** screen. To go back to SurfAnim, go to the **Preferences** screen and view the dataset named **Defaults Settings**.

VsV is backward compatible with file formats and datasets used in CarSim. Some of the shapes provided in past versions of CarSim do not display well in VsV; they have been replaced with cleaned-up shapes that work well in both VsV and SurfAnim.

The first thing to know when using VsV interactively is that you use the right mouse button to control the virtual camera. Hit the F1 key to bring up a summary of the keyword and mouse commands used to control the view in VsV.

Improvements in the Math Models

Several major new options have been added to the math models, plus there are many minor improvements in the architecture.

Continuously Variable Transmission (CVT)

The CarSim powertrain now includes an option for CVT, with corresponding improvements in the GUI (two new screens) and updated powertrain documentation.

Engine Mounts

CarSim 8.1 includes additional solver programs with extra multibody degrees of freedom (DOF) to include engine movements allowed by engine mount compliances. The extended models include ride vibrations involving engine movements, and torque reactions from the driveline that affect load transfers when power is applied. Solvers with the engine-mount capabilities are provided with DLL files with the prefix "e_" in the file name, such as e_i_s_ss. There are 12 new solvers, making a total of 36: 12 basic car/trailer combinations, 12 with frame twist, and 12 with engine mounts. Two copies of each are provided for use with 32-bit and 64-bit external software.

An optional license is needed to use the engine-mount models.

AVL Cruise Interface

The CarSim 8.1 models provide low-level integration with the commercial powertrain simulation software Cruise from AVL. Integration can be difficult in older versions using external software such as Simulink to connect the models. However, the new release provides tight integration with

minimal simulation time delays between the models, with automatic connections supported by the GUI in CarSim and AVL-Cruise.

An optional license is needed to use the AVL-Cruise interface. (And of course, the Cruise software and associated licenses from AVL must be installed.)

Driver Model Improvements

Several new options have been added to the closed-loop controllers for speed and direction.

- A path preview speed control option is available where the controller looks ahead at the driver reference line and defines a target speed as a function of curvature, 3D geometry (banking angle, grade angle, vertical curvature), aggressiveness (lateral and longitudinal acceleration limits), and skill level. A new screen was added to support this option.
- The closed-loop speed controller was extended to include nonlinear cubic feedback and to take engine braking into account when setting low throttle levels.
- There is a tight connection between the driver model, the road geometry (road reference line), and the vehicle initialization. The initialization options have been extended and clarified to handle ambiguities that could occur with looped paths and roads.

Reference Paths for Road Geometry and Controllers

CarSim has supported two reference paths that are used to initialize the vehicle position and orientation, provide a target for the closed-loop steering, and provide motion information for moving objects such as traffic vehicles. One of these paths—the road reference line—is also used to link geometric and friction properties together to define a 3D road surface. CarSim 8.1 extends these capabilities in several ways:

- Some ambiguities have been clarified involving the two potential paths. One path is always defined as the road reference line and the other is always defined as a driver reference line. Both have the same conventions for defining longitudinal position (station), setting the initial vehicle position and orientation, and handling of looped paths (test tracks and racetracks).
- Both reference paths can now be used at the same time. In this case, the road reference line is used to specify the 3D road surface properties and the driver reference line is used by the controllers and to optionally initialize the vehicle position and orientation.
- More information is available for looped paths. The starting station for each path is set with a parameter. When looped, the maximum station for each type of path is available from a parameter whose value is automatically calculated internally. A state variable, `SV_N_START_CROSS`, is updated every time the vehicle crosses the start of the loop (corresponding to the start of the table of X-Y coordinates used to define the path). This can be used to count the number of laps made.

- In past versions of CarSim, the initial vehicle direction along a path was determined by comparing two parameters `SSTART` and `SSTOP`. This method had limits on looped paths (where `SSTOP` could be reached traveling in either direction), and in complicated procedures where the vehicle changes direction during the run. A new parameter `OPT_DIRECTION` is now available that makes control more obvious and simplifies the creation of scenarios in which the vehicle might change direction in the middle of a run.

Import and Output Variables

Many minor changes were made to import and output variables involving name consistencies. Also, the following major changes were made:

- A road axis system was added to support road-based vehicle variables such as accelerations, roll angle, and pitch angle. These are more convenient in some applications involving roads with hills and/or banked turns. The documentation for roads was expanded to include more discussion in the concepts underlying the model. It also describes details of the road axis system.
- Older versions of CarSim included auxiliary import and output variables that were not part of the vehicle physics model, but were included to support user-defined model extensions. With the capabilities now provided in VS commands and the VS API for adding new variables, these old “built in” auxiliary variables are not needed and have been removed to avoid potential name conflicts with user-defined variables. The variables that were removed are listed in Table 1 on page 11. An example dataset is included in the CarSim database that will restore these variables if needed (page 11).

Configurable Table Functions

CarSim math models make extensive use of configurable functions that can be set at runtime to represent nonlinear relationships using various table-lookup methods, linear coefficients, or constants. New options are available in CarSim 8.1 to make it easier to extend the models.

- All of the configurable functions used to specify open-loop driver controls (steering, braking, etc.) allow scaling and offset of both the output variable and time. This simplifies the description of complicated tests that use variations of a waveform such as sine-with-dwell steering—the same waveform is simply rescaled for repeated tests with different amplitudes.
- Some of the configurable functions that calculate a dependent variable from two independent variables now include parameters to scale and offset both independent variables. For example, tire force tables now allow scaling of output force (dependent variable), vertical load (independent variable), and the other independent variable (lateral slip, longitudinal slip, or inclination angle). This provides support for advanced users to apply normalized datasets.
- A new option for the function type is `_EQUATION`. With this option, you provide an equation that is used in place of the function. The equation can be based on the built-in independent variable(s) and can make use of other variables in the model. This

allows functions to be extended to provide sensitivity to other variables beyond the original model. Support for this option is provided in the relevant GUI screens and is covered in the documentation.

- The VS API was extended to include functions that define new configurable table functions and apply them in external custom programs.

Tire Model Improvement

CarSim has two built-in tire models that take data in tabular form and apply combined slip theory to generate forces and moments. The original equations for combined slip were taken from a technical paper published in 1991 by Pacejka and Sharp, which summarized some work published in 1989 by Bakker, Pacejka, and Lidner that had additional detail to handle a transition between two types of combined slip (“normalized” and “theoretical”). With some sets of tire data, the transition option provides more realistic behavior for some extreme conditions.

The CarSim models now include additional tire parameters for advanced users who can evaluate friction ellipses using the CarSim tire testers and make adjustments.

The tire table-lookup functions have been extended to include scale factors for the two independent variables (instant vertical load and lateral slip, longitudinal slip, or inclination), allowing advanced users to work with normalized tables that can be transformed to define tires with different load ratings and stiffness properties.

Miscellaneous Architecture Improvements

Improvements were made in the simulation process that are not specific to vehicle features.

1. Processing of Parsfiles during events has been improved for RT systems.
2. Support has been added for 64-bit versions of TNO and COSIN tire models, allowing them to be used with CarSim when running with other 64-bit software (e.g., 64-bit Simulink).
3. New parameters were added—`ID_Run` and `ID_Event`—to identify the database parsfile ID associated with run and event condition. These are also available as output variables for plotting or exporting to external software such as Simulink.
4. Symbolic equations provided to set values or equations in VS commands can reference a constant with the symbol `G` (9.80655), along with existing constants `PI` (3.14159265...) and `DR` (57.2957795...).
5. Multiple instances of the VS commands `DEFINE_IMPORT`, `DEFINE_PARAMETER`, and `DEFINE_VARIABLE` applied with a variable that has already been defined can now be used to update the value assigned to the variable.
6. More information is provided in the log text file generated with each run.
7. Reading of parsfiles is more forgiving of the use of spaces (including lack of spaces) in expressions.
8. The VS API has functions for defining moving objects and sensors under full control of external software.

9. If a run is made with no outputs, existing ERD and BIN files with the same root file name are deleted, to avoid confusing old results with new PAR, ECHO.PAR, etc. files.

Improvements in the Database

In past versions of CarSim, the database has served to provide examples for using the software. Most users would then create new datasets for specific vehicles, subsystems, procedures, etc. This is still the case for experienced users working for OEMs and suppliers.

As usage of CarSim has extended to many new applications, some users rely on CarSim not only for the software tools, but also for representative datasets for tires, engines, entire vehicles, and advanced procedures. The database in CarSim 8.1 covers the same sorts of examples and previous versions. However, the example vehicles and procedures have been reviewed and tested to ensure that when used in simulations, they produce reasonable behavior for conventional tests covering full ranges of acceleration, braking, steering, handling, and stability.

Organization of Example Datasets

Existing users of CarSim can continue to work with existing databases with vehicles and procedures of interest. However, the examples in CarSim 8.1 have been reworked to provide more immediate help to new users. The example runs are organized into submenus for three general purposes:

1. Examples that show how to apply common test procedures. The submenus have names such as **Brake Tests**, **Handling and Stability Tests**, etc. Most of the runs show how a test procedure provided in CarSim is applied to some vehicle. Changing to a different vehicle is so easy that usually there is only one example.
2. Examples that show alternative modeling or data options in CarSim. These submenus have names such as **Alternate Vehicle Data Measurement**, **Roads**, **Tire Models**, etc. Example runs make use of some vehicle dataset using an alternate option that might be more convenient, depending on available sources of data.
3. Examples that show software features of CarSim. New features have CS 8.1 in the title (e.g., **CS 8.1 CVT Powertrain**). Others have titles related to the features of interest, such as **Aero Effects**, **Extended Models**, **External Control of Runs**, etc.

Features that require an extra license option (frame twist, AVL Cruise interface, engine mounts, etc.) are demonstrated with one run per option in the submenu **More Examples (Extra License Options)**. More examples for each option are available, but are not part of the standard CarSim database that is intended as a foundation for basic usage.

Procedures

The general method for making a new run is to select a vehicle, select a procedure, click the **Run Math Model** button, and view results with the **Animate** and **Plot** buttons. With this in mind, procedures are included that are fairly independent of the vehicle being run. There has been great interest recently about using CarSim to comply with regulations such as the sine with dwell ESC test (ECE R13H, FMVSS 126), and validating CarSim as a simulation tool using other standard tests such as ISO 4138 (e.g., constant-radius understeer test). Example procedures are included

that apply a complete test sequence specified in regulations and standard tests. Some test procedures included in CarSim 8.1 apply the scripting capabilities of CarSim to provide a complete test sequence with one click:

1. The sine with dwell test (ECE R13H, FMVSS 126) starts at zero speed, increases to 80 km/h for a slowly increasing steer test to the left, then recovers, then does a slowly increasing steer test to the right, then comes up to 82 km/h, then coasts down to 80 km/h, then performs a sine with dwell maneuver, then comes back to 82 km/h to coast down and do another sine with dwell test with a different steering amplitude, and so on for 30 to 50 sine with dwell repeats.
 - a. One option in this sine with dwell sequence is to disable writing to file during transitions, such that animations and plots only show the portions of interest (the increasing steer part and the sine with dwell maneuver), without the coasting down and recovery parts. This reduces the saved data by about half.
 - b. Another option is to use the capability of CarSim to save the state of the math model just as the sine with dwell starts, and eliminate transitions by jumping back to that start condition for the next iteration. This works for software-only simulations and reduces the simulation time by about half. (However, it cannot be used with HIL systems.)
2. The constant-radius understeer test (ISO 4138) starts at low speed on a constant-radius path and runs long enough for the vehicle to stabilize and determine the low-speed (Ackermann) angles for the specific vehicle and radius combination at the steering wheel and front road wheels. Then, the speed is increased at a rate such that the lateral acceleration increases by 0.1 m/s^2 and the steering relative to the Ackermann angle (at the steering wheel and the front wheels) is plotted against lateral acceleration.
3. The fishhook rollover resistance test (NHTSA 2001-9663) starts with the same process as the sign with dwell series, to determine a reference steering wheel angle using a slowly increasing steer test, and then applies two or more rapid fishhook steering inputs to determine the maximum speed (up to 50 mi/h) that the vehicle can handle without lifting wheels.

These procedures use VS commands to introduce new parameters and variables specific to the test, and switch from one stage to the next using vehicle responses specific to each test.

Vehicles and Components

The main vehicle examples in CarSim (hatchbacks and sedans in Classes A-F with some SUV, pickup trucks, and utility vehicles) have been reviewed and modified to provide realistic behavior over an extensive range of maneuvers.

Although the datasets for tires, engines, suspensions, and vehicle assemblies are not representative of specific components and vehicles, they do provide descriptions such that the behavior of the component or vehicle is something that could exist, and will generate vehicle motions that are representative of its class.

Tires

Most of the tire datasets in CarSim 8.1 were generated for general-purpose simulations. Tires in the data sets are identified by the Tire and Rim Association size designation which determines the radius and width, and an associated load rating. Tires should have a load rating at least equal to the maximum static load of the vehicle on which it is being used.

For purposes of creating the generic tires, force and moment properties from a number of passenger car tires were analyzed to discern trends in their properties. Performance was normalized by tire size as reflected in their load rating and the test load as a proportion of the rated load. The other primary variable in normalizing performance is the aspect ratio (ratio of section width to section height) that influences cornering performance.

The generic tire data sets in the CarSim examples were created by applying the Pacejka magic formula to create well-behaved table functions that resemble measured datasets that have been observed in use with CarSim. Longitudinal and lateral force behavior was characterized to match the stiffness (initial slope of force with slip), the peak friction coefficient, the slip at which the peak occurs, and the sliding coefficient of friction.

Tire lateral stiffness also varies with aspect ratio. Low aspect ratio tires are stiffer than high aspect ratio tires. High aspect ratio tires in the range of 60 to 80 are considered to be touring tires, meaning they are used on normal passenger cars where ride is emphasized over handling. Low aspect ratio tires in the range of 35 to 55 are considered sports tires and have higher cornering stiffness values in the data sets.

Powertrain

Engines in CarSim are characterized by the torque versus speed curves for different throttle openings. Data sets are provided for gasoline engines ranging from 75 to 300 kW. All use similar torque maps from an actual engine scaled and “cleaned up” to yield the power level reflected in the name for the data set. Changes were made in the data sets to yield torque properties below idle speed more typical of modern engines for the purpose of improving launch drivability in driving simulator applications. The closed throttle curves were also tuned to produce idle speeds when the transmission is in neutral that match the value shown on the engine screen.

CarSim has one diesel engine dataset example. The torque map has been modified from earlier versions by adjusting values below idle speed to create behavior that is more realistic regarding stall characteristics during launch.

Brakes

Most vehicles sold in developed countries have ABS systems and do not use proportioning valves. Some example systems are still included without ABS, however, to simulate systems used in low-cost vehicles. CarSim vehicles without ABS have proportioning valves; systems with ABS do not, with the exception of a full-sized pickup truck, which has both ABS and proportioning.

All brake actuators in the CarSim examples use linear coefficients to link fluid pressure to brake torque. The coefficients were selected from a library of rounded values (100, 200, 300, etc. N-m/MPa) such that front wheels typically have higher adhesion utilization than rear wheels, and will lock or engage the ABS with line pressures of about 6-7 MPa.

Animation Data

CarSim driving simulators are now being developed using the VehicleSim Visualizer (VsV) that is also provided in CarSim 8.1. VsV has many visualization capabilities that do not exist in the older SurfAnim program in CarSim. It is also more stringent about descriptions of 3D shapes in OBJ and STL files; datasets that violate some conventions display OK in SurfAnim, but not in VsV (and other advanced rendering programs).

All of the existing OBJ and STL files in CarSim were reviewed and modified to ensure that they display properly in both SurfAnim and VsV. Further, many of the files were modified to provide improved appearance, especially in VsV where more lighting and texture options are supported.

Handling Course

Recent versions of CarSim have included a looped racecourse called **Road Course** in the category **Handling Roads**. This was based loosely on a racecourse near the Ann Arbor office of Mechanical Simulation, and was included in the virtual proving grounds supplied with the CarSim DS (driving simulator) packages.

CarSim 8.1 includes a complete replacement in the 3D road library called **Handling Course** in the category **Handling Roads**. (This road is also part of the database to be provided in CarSim DS updates.) The new dataset is based on the same racecourse near Ann Arbor. In this case, the dataset was intended from the start to be used in driving simulators. It includes more comprehensive road data (taken from GPS measurements), curbs, and significantly more advanced 3D shapes for use in driving simulators.

The new handling road is used to showcase the new path preview option for the driver speed controller. It is a good resource for evaluating high-performance vehicles, and is useful for presenting animations with graphic details of a 3D road surface.

Improvements in the GUI

This update mainly introduces new model features and the new Visualizer. The GUI was updated to provide support for these new features, along with some other notable improvements.

1. New screens were added to support the AVL Cruise interface, engine mount model option, CVT option, and path preview option in the speed controller.
2. New Windows COM functions were added to access the VS browser remotely: `GetBlueLink` (get information about a blue link on the current screen) and `ExportParsfile` (export a parsfile from the current screen plus all linked data screens).
3. The **Libraries** menu was reorganized to simplify navigation to the main screens in systems such as **Powertrain** where there are many data screens.
4. The **Libraries** menu was also modified to merge the libraries for 3D ground and road properties into a single category **Ground and Roads**. (The old **Roads** category no longer exists.)

5. When making a simulation run using a built-in model, the progress status includes the current simulation time along with the status bar. The status bar is not shown if neither a stop time nor a stop station is specified.
6. Screens for configurable functions were modified to support the new `_EQUATION` option when possible. (Not all configurable functions support the new option, as described in the documentation.)
7. Screens for configurable functions include more information about scaling options for independent variables.
8. Screens for tire tables always show plots that go down to zero slip or inclination. (The values at zero slip or inclination are added by the tire model and are not part of the normal tabular data.)
9. The Calculator Tool, available from most screens with configurable functions, has settings for the number of significant digits in X and Y values that are calculated.
10. The **Preferences** screen was extended to include a miscellaneous field for setting global preferences for advanced system-level parameters in the math models.

Compatibility with Older CarSim Versions

Most of the new features in the math models involve adding new parameters and output variables. Old datasets do not use these features, so the lack of parameter settings is not a problem. However, a few variables used in old models were removed or renamed. These changes will not affect most users, because the affected variables were either hidden or only used for advanced applications.

Auxiliary Parameters and Variables Removed

Older versions of CarSim included auxiliary variables that were not part of the vehicle physics model, but were included to support user-defined model extensions. With the introduction of VS commands such as `define_variable`, `define_output`, and `define_parameter`, the old built-in variables are no longer necessary.

Table 1 lists variables in earlier versions of CarSim that were removed from CarSim 8.1. If they are referenced in existing datasets, they can be restored using VS commands. In fact, the database installed with CarSim 8.1 includes a VS command dataset in the library **Generic VS Commands** named **Install Obsolete CarSim 8.03 Generic Variables** that restores the variables from Table 1, along with the 12 generic tables mentioned in the next subsection.

Table 1. Discontinued auxiliary parameters and variables.

Name	Indexing	Description
FLAG_ <i>i</i>	<i>i</i> =1 - 1	Auxiliary flag variables (parameters and outputs)
IMP_AUX <i>i</i>	<i>i</i> =1 - 20	Auxiliary import variables
IMP_DXEXT_ <i>i</i>	<i>i</i> =1 - 10	Derivatives of auxiliary state variables (defined externally)
SV_XEXT_ <i>i</i>	<i>i</i> =1 - 10	Auxiliary state variables (defined externally)

Generic Tables Removed

Recent versions of CarSim included twelve generic tables (keyword = GENERIC) for use in VS commands. The VS command DEFINE_TABLE makes these built-in functions unnecessary. If you have existing datasets that use the GENERIC function, it can be restored with the DEFINE_TABLE command. These tables can be restored using the VS commands dataset in the library **Generic VS Commands** named **Install Obsolete CarSim 8.03 Generic Variables**.

Import Variable IMP_PWR_EXT_ENGING_AV Renamed

The variable IMP_PWR_EXT_ENGING_AV (power available from external engine model, needed by built-in speed controller) was misspelled. It has been renamed IMP_PWR_ENGINE_AV, and is consistent with a configurable function used to connect with external models from AVL Cruise (PWR_ENGINE_AV).

Y vs. Axle Roll Solid-Axle Kinematics (Keyword and Definition)

The sign convention for axle roll in the configurable function used to define lateral movement of the wheel centers used the opposite sign convention for roll that is used in all other screens. This was inconvenient and could lead to error if the documentation was not checked for this particular screen. The sign convention was changed to match the other screens: positive roll of the suspensions and axles occur when the sprung mass has positive roll (i.e., the sprung mass rolls to the right).

To avoid ambiguity about the sign convention in new and old datasets, the old function keyword SUSP_LAT_AXLE_ROLL has been replaced with SUSP_Y_AXLE_ROLL. When datasets from previous versions of CarSim are loaded into CarSim 8.1, the roll values are multiplied by -1 and the table is flipped (to provide ascending values of roll). Therefore, when you look at the data in CarSim 8.1, the plots will appear flipped horizontally from their appearance in earlier version.

For data handled through the browser (carsim.exe), conversion is completely automatic. However, if you are using advanced simulation methods where suspension data is being provided from external files, it will be necessary to change the data files to use the new keyword and sign convention.

Reference Paths for Road Geometry and Controllers

CarSim 8.1 extends the use of the road reference line and the driver reference path to provide new capabilities and fix some ambiguities that existed in past versions. The VS browser has been updated, such that datasets for basic runs that run in past versions should provide the same results

with no modifications required of the user. However, old datasets that pushed the limits or used tricks to accomplish complicated maneuvers might require modifications.

YIN_TABLE Used for Both a Target Path and the Road Reference Line

CarSim has supported two models for defining the 3D ground surface, selected using the parameter `OPT_ROAD`. When `OPT_ROAD` is zero, the ground geometry is defined with a 3D table that calculates elevation `Z` based on values of `X` and `Y`. When `OPT_ROAD` is not zero, the ground geometry is defined using the VS road model that uses coordinates of `S` (station) and `L` (lateral position). The road model is built with a road reference line, defined with a table of `X-Y` coordinates identified by the keyword `YIN_TABLE`.

If the closed-loop steering controller is enabled, a target path is defined with `S` and `L` coordinates along a reference path. With past versions, if the road model and closed-loop steering were both enabled, then the driver model always used the road reference line. However, if the road model was not enabled, then a target path was used that was represented with a table of `X-Y` coordinates, also identified by the keyword `YIN_TABLE`.

In CarSim 8.1, both paths can be used at the same time. The table of `X-Y` values used to define a road reference line is now identified with the keyword `ROAD_REF_XY_TABLE` and the table used for a driver path has the keyword `DRIVER_REF_XY_TABLE`. The keyword `YIN_TABLE` is still recognized internally by the VS solvers as being equivalent to `ROAD_REF_XY_TABLE`, allowing old datasets to work if used to describe road reference lines. However, old datasets that had the combination of no road and the closed-loop steering controller must be modified to use the new keyword `DRIVER_REF_XY_TABLE`. (Revisions are needed only if the data parsfiles are provided under the control of external software such as MATLAB. If the CarSim GUI and database are used, then everything is automatic.)

Station in Looped Paths

The CarSim VS solvers have a state variable `SV_STATION` and associated output variable `Station` for the distance along the reference path used by the driver controllers and the 3D road model. The VS road model has supported loops to represent test tracks and racecourses. With a looped path, the station is forced into the range covered by the loop in order for the road model to calculate the ground geometry (elevation and slope). Although the station was usually forced to the range of the looped path internally, the state variable and associated output variable was not always modified.

In CarSim 8.1, there are two potential station variables, one for the road reference line, and one for the driver reference path. In both cases, if the path is looped, the station is forced to be within the range of the loop. The new behavior is documented and more consistent.

For most scenarios, differences in how station is calculated do not affect results. However, old datasets that define complicated scenarios in which station is monitored in VS commands might fail on looped roads now that station is forced to stay within the range of the loop. If this is the case, the datasets will have to be modified.

Setting Direction of Travel

In past versions of CarSim, the direction of travel along a target path was determined by comparing values of the parameters SSTART and SSTOP. The parameter SSTOP had two roles: it was used as a condition for stopping the run if another parameter OPT_SSTOP was greater than zero. If OPT_SSTOP was not greater than zero, it was still used to define the direction of the vehicle for initialization and controller settings.

When working with looped paths, the old method could be confusing and sometimes required elaborate sequences of events.

CarSim 8.1 simplifies the setting of direction with a new parameter OPT_DIRECTION. During initialization, OPT_DIRECTION is set to one of two values: +1 or -1. For backward compatibility with old datasets, it will be set based on a comparison of SSTART and SSTOP during initialization if it has not been assigned a nonzero value in a parsfile.

The CarSim GUI sets up the initial direction automatically, providing backward compatibility for most simulations. However, in some complicated scenarios, events are used to change the direction during a run. Old datasets required tricks where SSTOP is given a new value based on the current position of the vehicle. Some of the tricks probably won't work anymore unless OPT_DIRECTION is reset. (It is usually much simpler using OPT_DIRECTION directly.)

Position of Yellow Field Changed in Procedures Screen

The **Procedures** screen has two sections with titles **Additional Data**, with nine potential data links and two miscellaneous scrollable yellow fields (Figure 1a). In the figure, the blue arrow zigzag line shows the sequence in which data from these controls are written into the parsfile associated with the display. One of the miscellaneous fields appears before any of the links (①), such that advanced users can use it to define parameters or variables that are referenced in linked datasets that follow. Another miscellaneous yellow field (⑨) appears after most of the links, but before that last two. It can be used to set values to override settings made in the linked datasets.

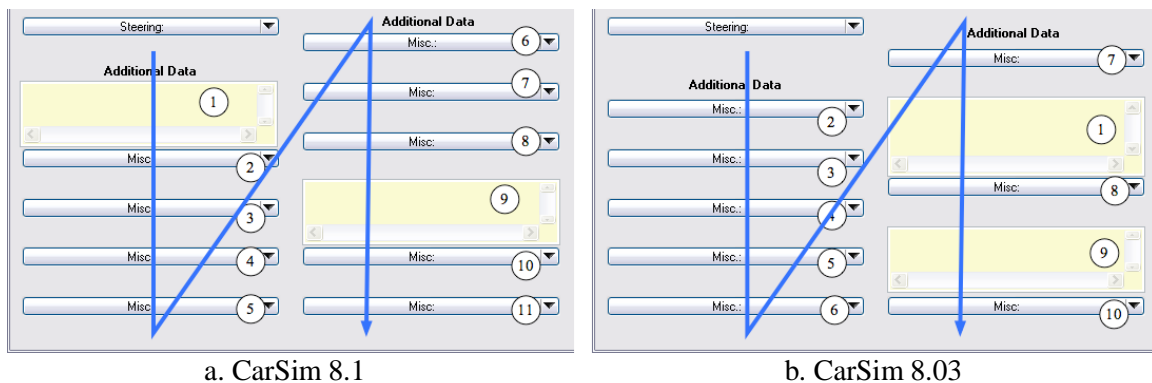


Figure 1. Layouts of bottom-left part of **Procedures** screen.

Past versions of CarSim 8 have the same two miscellaneous fields and eight data links with a different layout (Figure 1b). In this case, the first miscellaneous field (①) has only two potential links that follow (⑧ and ⑩). When data from CarSim 8.0x is loaded into CarSim 8.1, all of the linked datasets and contents of the yellow fields are retained. However, the sequence that they are

written into the parsfile is changed, as indicated by comparing Figure 1a and Figure 1b. In most cases, this is OK.

However, some advanced users might have made datasets in which settings in the first yellow field ① are not valid when read by the VS solver unless data from one or more of the links ② - ⑦ have already been read by the VS solver. If this is the case, the contents from the first field must be moved manually.

Road-Based Variables Removed

The CarSim solvers had built-in output variables for the station and lateral coordinate of the axle centers (Sta_A1, Lat_A1, Sta_A2, Lat_A2). These were removed because the calculation can sometimes cause numerical problems when using the option to set the driver path with X-Y coordinates that describe a tight turn. They can be defined easily using VS commands if needed. (Use the functions ROAD_S and ROAD_L, using X and Y coordinates for the axle centers as arguments.)

Hidden Parameter IROAD renamed as IDZ_ROAD

The 3D road model in CarSim has off-center elevation changes defined with a configurable function ROAD_DZ that supports two datasets. The first dataset is intended for design features such as cross-slopes, ditches, and curbs. Starting with the CarSim 8.1 release, it is also used to define a road reference axis system to support road-based vehicle variables such as roll and pitch. The second dataset can be used for other features such as roughness or special events. In past versions, the two datasets have an associated hidden parameter named IROAD. This parameter has been renamed IDZ_ROAD for consistency, allowing IROAD to be used for future improvements supporting multiple roads. This change is made in both the VS solvers and the CarSim GUI. It will only affect users who are using automation methods involving Parsfiles made with older versions that involve the second DZ component and have lines that read "IROAD 1" and "IROAD 2". If you have any such files, the lines should be changed to use the keyword IDZ_ROAD.

TSTART_BRAKES Parameter Redefined

CarSim uses configurable functions to define open-loop braking with either master cylinder pressure or brake pedal force. The pressure and pedal force configurable functions now have options to provide starting points and scale factors for time, as do all other open-loop controls. The parameter for brake pressure function PBK_CON is TSTART_PBK_CON; the parameter for the pedal force function F_PEDAL_FORCE is TSTART_F_PEDAL_FORCE.

Older versions of CarSim did not have the built-in option to offset the independent variable in table functions. Instead, a parameter TSTART_BRAKES was included to provide a time offset for both kinds of brake input.

In order to provide backward compatibility with datasets that set a time offset with the old keyword, CarSim 8.1 is programmed to handle lines of input that begin with the keyword TSTART_BRAKES followed by a number or symbolic expression. The expression is evaluated to obtain a number, which is then assigned to both the TSTART_PBK_CON and

TSTART_F_PEDAL_FORCE parameters. This fully supports parsfiles used during events to reset the starting time of a brake input.

The keyword TSTART_BRAKES is not directly associated with an internal variable. Therefore, the name cannot be used within symbolic expressions. This is mentioned here because an example brake test **Handbrake Turn** made use of the old TSTART_BRAKES parameter in past releases of CarSim. The example was redone in CarSim 8.1 using a new parameter defined during the run using the VS command DEFINE_PARAMETER.

Camber Output Variables

Camber output variables for independent and solid axle suspensions did not include compliance effects. The internal variables used in the multibody equations were correct; the output variables provided for plotting and export were inconsistent. The camber output variables now match the internal variables.

Steering Output Variables

Nonlinear steering ratios are used to determine measured road-wheel steering angle as a function of either rack displacement or a pitman arm angle. Two options are available for defining the road-steer angle (set with the parameter OPT_STEER_DEF): the angle as defined in the sprung-mass X-Y plane (typically as measured in the lab), or as a rotation about the kingpin steer axis. The two angles are identical if the steer axis is vertical, but they differ when the steer axis is inclined. In past versions of CarSim, the definitions of output variables Steer_L1, Steer_R1, Steer_L2, and Steer_R2 depended on the value of OPT_STEER_DEF. Another set of output variables ASt_KP_L1, ASt_KP_R1, etc., were always the angles about the kingpin axis.

In CarSim 8.1, the definitions are consistent regardless of the value of the parameter OPT_STEER_DEF: the steer as measured in the vehicle X-Y plane is available with the output variables Steer_L1, Steer_R1, etc., and the steer about the kingpin axis remains available via the output variables ASt_KP_L1, ASt_KP_R1, etc.

Bug Fixes and Errata

The following bugs were identified and corrected.

1. A bug in the steering geometry involved the calculation of the second Euler rotation of the body in the math model that orients the direction of the wheel spin axis. The result of the bug was the addition of a small steer effect on steered wheels with inclined steering axes (kingpin inclination) and suspension dive (rotation about the sprung mass Y direction). The differences between results of simulations produced with CarSim 8.1 compared to previous versions are small, but noticeable, for the typical range of kingpin angles and dive angles.
2. The parameter OPT_STEER enables torque steering input when operating in open-loop mode. A bug in the steering system caused it to interfere with closed-loop steering if set.
3. The open-loop steering configurable functions (both angle and torque) were always listed in the echo file, even if disabled. They are now hidden if not active.

4. The spring design deflection was added to the suspension deflection before considering mechanical advantage. This has been corrected so the design deflection is now added to the spring deflection as calculated using the mechanical advantage configurable function. The design deflection option is not used in any examples supplied with CarSim; they all use the reference jounce parameter option (OPT_JNC_DESIGN).
5. Past versions allowed import of camber angle for each wheel. This ignored an associated jacking effect that occurs on the road. The import variables related to camber of the road wheel were removed. Camber (and other suspension kinematical variables) can instead be imported by using configurable functions and VS commands, which can be set up to handling jacking effects correctly.
6. The aligning moment calculated by the built-in extended tire model (OPT_TIRE_MODEL = 6) sometimes missed the contribution due to F_x and the Y displacement of the contact patch center implied by the overturning moment.
7. The closed-loop speed controller would get confused if the vehicle rolled backward (e.g., sitting in neutral on a hill).
8. A limited simulation program is used to separate auxiliary roll moment from total roll moment measured at the ground when using the **Suspension: Measured Roll Stiffness** screen. The simulation program did not recognize some of the enhancements made for configurable functions in recent versions of CarSim. It has been replaced with a new VS solver that supports the newer options.
9. The closed-loop speed controller didn't consider negative engine torques when interpolating to calculate throttle.