

BikeSim 3.1 New Features

- Introducing the VehicleSim Visualizer: VsV 1
- Improvements in the Math Models..... 2
 - Continuously Variable Transmission (CVT) 2
 - Rider Model Improvements 2
 - 3D Ground and Road Profile 3
 - Reference Paths for Road Geometry and Controllers 3
 - Import and Output Variables 4
 - Configurable Table Functions..... 4
 - Tire Model Improvement..... 4
 - Miscellaneous Architecture Improvements 5
- Improvements in the Database 5
 - Nonlinear Suspension Kinematics Data..... 5
 - Animation Data..... 6
 - Handling Course 6
- Improvements in the GUI..... 6
- Compatibility with Older BikeSim Versions 7
 - Auxiliary Parameters and Variables Removed 7
 - Import Variable IMP_PWR_EXT_ENGING_AV Renamed 8
 - Reference Paths for Road Geometry and Controllers 8
 - Position of Yellow Field Changed in Procedures Screen 9
 - Hidden Parameter IROAD renamed as IDZ_ROAD 10
- Bug Fixes and Errata 10

BikeSim 3.1 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 BikeSim 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 VehicleSim 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 BikeSim. Some of the shapes provided in past versions of BikeSim 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 BikeSim powertrain now includes an option for CVT, with corresponding improvements in the GUI (two new screens and two updated existing screens) and updated powertrain documentation.

Rider 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 rider 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 rider 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.
- Improvements were made in the definitions of coordinate systems used by the rider model with respect to interactions between the steering axis, the main frame, and the ground surface. The rider model longitudinal X axis used to be fixed on the motorcycle main frame, but is now always within the inertial X-Y plane. This removes unwanted influences motorcycle pitch and roll, leading more stable steer behavior while cornering.

- The closed-loop rider model (steer controller for path following) was extended to include path curvature feedback to predict the target path better.

3D Ground and Road Profile

Two options were added for defining ground surface shapes.

- In addition to the 3D ground geometry that defined using the VS road model with coordinates of S (station) and L (lateral position), BikeSim 3.1, adds support for an alternative definition that uses a simple 3D table to calculate elevation Z based on values of X and Y.
- A “wandering road roughness profile” is available that follows the vehicle and provides incremental ground elevation at the wheel locations. This allows use of measured road profiles (obtained by highway agencies) to provide high-frequency roughness inputs without the need to define a complete 3D surface. (The profile is applied only where the bike travels.) The roughness is defined on the **Surface: Roughness Profile** screen, and is explained in the document from **Help -> 3D Ground and Roads**.

Reference Paths for Road Geometry and Controllers

BikeSim 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. BikeSim 3.1 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 (rider) 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 BikeSim, 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.

Older versions of BikeSim 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 8. An example dataset is included in the BikeSim database that will restore these variables if needed (page 7).

Configurable Table Functions

BikeSim 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 BikeSim 3.1 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 a lane change—the same waveform is simply rescaled for tests with different conditions.
- 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

BikeSim 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 BikeSim models now include additional tire parameters for advanced users who can evaluate friction ellipses using the BikeSim 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 BikeSim 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` (“degrees to radians,” 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.

Improvements in the Database

In past versions of BikeSim, 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 BikeSim has extended to many new applications, some users rely on BikeSim not only for the software tools, but also for representative datasets for tires, engines, entire vehicles, and advanced procedures. The database in BikeSim 3.1 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.

Nonlinear Suspension Kinematics Data

Recent versions of BikeSim have included example motorcycles with nonlinear suspension kinematics called **Sport Touring: Wishbone ...** in the category **Touring Bikes**. These were

based on multi-link suspensions, namely double wishbone front suspension and parallel linkage rear suspension.

More recently, those examples in BikeSim were used in a technical paper published in April 2011 by Watanabe and Sayers (SAE: 2011-01-0960), and those suspension kinematics data has been reviewed and refined based on more realistic kinematics design by hand drawing and VehicleSim LISP kinematical analysis. BikeSim 3.1 involves those refined kinematics data.

Animation Data

BikeSim driving simulators are now being developed using the VehicleSim Visualizer (VsV) that is also provided in BikeSim 3.1. VsV has many visualization capabilities that do not exist in the older SurfAnim program in BikeSim. 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 BikeSim 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 VehicleSim products 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/TruckSim DS (driving simulator) packages.

BikeSim 3.1 includes a complete replacement in the 3D road library called **Handling Course** in the category **Handling Roads**. 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 CVT, 3D ground, road roughness profiles, 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 BikeSim 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 BikeSim 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 BikeSim that were removed from BikeSim 3.1. If they are referenced in existing datasets, they can be restored using VS commands. In fact, the database installed with BikeSim 3.1 includes a VS command dataset in the library **Generic VS Commands** named **Install Obsolete BikeSim 3.0 Generic Variables** that restores the variables from Table 1.

Table 1. Discontinued auxiliary parameters and variables.

Name	Indexing	Description
IMP_AUX i	$i=1 - 20$	Auxiliary import variables
IMP_DXEXT_ i	$i=1 - 10$	Derivatives of auxiliary state variables (defined externally)
SV_XEXT_ i	$i=1 - 10$	Auxiliary state variables (defined externally)

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).

Reference Paths for Road Geometry and Controllers

BikeSim 3.1 extends the use of the road reference line and the driver (rider) 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

In BikeSim 3.1, the 3D ground surface can be defined by one of two different ways as specified with 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).

If the closed-loop steering controller is enabled, a target path is defined with S and L coordinates along a reference path.

In past versions, a single path served both purposes, and was identified with the keyword YIN_TABLE.

In BikeSim 3.1, the paths can differ. 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.

Station in Looped Paths

The BikeSim 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 BikeSim 3.1, 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 BikeSim, 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.

BikeSim 3.1 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 BikeSim 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.

Past versions of BikeSim 3 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 BikeSim 3.0 is loaded into BikeSim 3.1, 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.

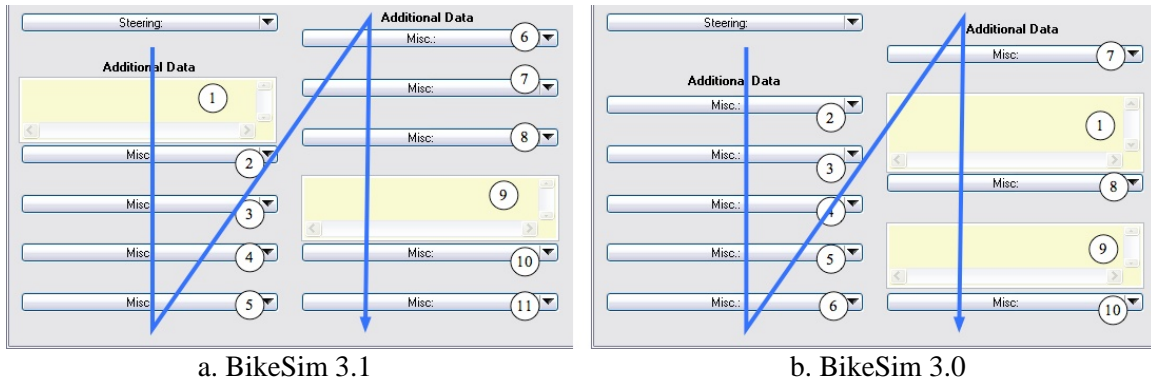


Figure 1. Layouts of bottom-left part of **Procedures** screen.

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.

Hidden Parameter IROAD renamed as IDZ_ROAD

The 3D road model in BikeSim 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 BikeSim 3.1 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 BikeSim 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.

Bug Fixes and Errata

The following bugs were identified and corrected.

1. In prior versions of BikeSim spring deflection was calculated by the suspension deflection multiplied by a mechanical advantage (i.e. lever ratio). The lever ratio was specified as a function of the wheel jounce. BikeSim 3.1 instead takes the spring compression directly as a function of wheel jounce. The new version involves a much simpler measurement, and is more accurate. In BikeSim 3.1, the new screen, **Suspension: Lever Compression**, can detect the old table keyword and automatically convert to the new format by integration of the old data. See more details in the tech memo **Converting Geometric Data from BikeSim 2.0/3.0 to BikeSim 3.1**, available from the **Help** menu.
2. The calculation of the spring design load of the front suspension is corrected. Front and rear spring design loads maintain the body height with static equilibrium without any

payload (without rider weight.) On the front suspension, the spring design load has been calculated based on the mechanical advantage which is mainly the caster angle. However, the wheel travel direction is not necessary as same as caster direction, and the initial wheel travel direction was not taken account for the design load calculation by the previous version. Now, the initial wheel travel direction is calculated by a small perturbation and it is taken into account for the design load.

3. The definition of the longitudinal and lateral axes of the intermediate axis system has been extended to use an axis system that involves the road surface orientation. In the earlier versions, the intermediate axis system is defined as a right-handed orthogonal axis system whose **Z** axis is parallel to Z_E , and whose **X** axis is perpendicular to both Y_W and Z_E . This definition is standard in vehicle dynamics terminology for four-wheel vehicles on a flat surface. However, it leads to some unwanted effects when the vehicle has a large roll angle and the ground is not level, as often occurs with motorcycles turning on a banked road. Table 2 compares the old and new axis definitions.

The change in the intermediate axis system influences several output variables when the ground surface is not flat. Those output changes do not directly affect the vehicle dynamics, but the changes in definitions can change the appearances of plots involving the variables (Table 3). The changes can influence the behavior of the rider model, leading to more realistic behavior (see **Rider Model Improvements** on page 2).

4. The brake system performance parameter (BK_PERF_SC) was internally treated as $(m/s^2)/MPa$ although it had the label (g/MPa) on the screen. Old data values should be divided by 9.80665 to obtain the same performance.

Table 2. Old and new expressions of the intermediate axis system

Name	X Direction	Y Direction	Z Direction
Intermediate (OLD) X, Y, Z	$\frac{Y_W \times Z_E}{ Y_W \times Z_E }$	$\frac{Z_E \times (Y_W \times Z_E)}{ Z_E \times (Y_W \times Z_E) }$	Z_E
Intermediate (NEW) X, Y, Z	$\frac{\{Z_E \times (Y_W \times Z_G)\} \times Z_E}{ \{Z_E \times (Y_W \times Z_G)\} \times Z_E }$	$\frac{Z_E \times (Y_W \times Z_G)}{ Z_E \times (Y_W \times Z_G) }$	Z_E

Y_W denotes spin axle of the wheel, Z_E is the vertical axis of the global coordinate system, and Z_G is the ground normal at the tire/ground contact point.

Table 3. Output variables influenced by the intermediate axis correction

Variable Name	Description
AX_SM	Sprung mass CG longitudinal acceleration
Ax_WC1, 2	Front and rear wheel longitudinal acceleration, respectively
Ay_SM	Sprung mass CG lateral acceleration
Ay_WC1, 2	Front and rear wheel lateral acceleration, respectively
Vx	Sprung mass CG longitudinal speed
Vxo	Sprung mass origin longitudinal speed
Vy	Sprung mass CG lateral speed
Vyo	Sprung mass origin lateral speed
R_Sys	Turn radius based on Vx and Ay