Mechanical Simulation

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

Technical Memo

carsim.com

CarSim 2016 Release Notes

Starting with this release, Mechanical Simulation is following a calendar year convention for numbering versions for BikeSim, CarSim, SuspensionSim, and TruckSim.

Model Features

CarSim 2016 includes new model features involving the vehicle, roads, paths, and controls.

Steering System Extensions

The underlying steering system model has been reworked to support more options, including improved support of external model components. Any axle on a vehicle may now be steered. Five top-level options exist to choose steering systems: the full internal model; no steering (bypasses steering code entirely); replace the steering rack or gear and any boost system; replace all but the tie rod / steer arm kinematics; or replace the entire system. All options may be used with open- or closed-loop control.

An option `OPT_DRIVER_ACTION` is available for the closed-loop steer controller (driver model) to calculate but not apply a steering wheel input. An output variable `STEER_DM` has been added to pass the calculated driver steer to an external model.

Connections Between Road Surfaces

The road model has been extended to support the connecting of multiple road surfaces. Each road surface now includes information for four boundaries involving the local S and L coordinates: one at the end (SMAX); one at the start (SMIN); one on the left (LMAX); and one on the right (LMIN). Each tire/road contact is tracked to automatically determine which surface is used to calculate the contact geometry and friction. The tracking is also applied to moving objects (traffic vehicles, lane edges, etc.).

Four new library screens were added to support Road Boundaries: the **Road Boundaries** screen, plus three Configurable Function screens that define boundary conditions (lateral distance, new road ID, and new station) as functions of station.

The single system parameter for tire/ground rolling resistance surface coefficient `RR_SURF` was replaced with an indexed parameter `RR_SURF` that is part of each road surface (along with friction and elevation).

ADAS Sensors and Moving Objects

Sensor-object detections have five more output variables: the X-Y-Z coordinates of the closest point of the target in the sensor coordinate system, and the derivatives of the X and Y components.

The recycling option for moving objects has been extended to work when the object is on a different path than the ego vehicle; this option was not supported in CarSim 9x.

Several minor improvements were made to VS moving objects. The speed control configurable function `SPEED_TARGET` was extended to support 100 independent target speed datasets.

Objects and sensors now have a NAME parameter that is assigned a user-defined description, to make the Echo file more helpful for simulations involving multiple sets of sensor or objects.

GUI Support for New Twist Beam Suspension

A new model was introduced in CarSim 9 that was based on the `i_i` math model, with VS Commands to set compliance coefficients to match K&C test data for twist beam suspensions. The CarSim 2016 version has been refined and is supported with six dedicated GUI screens (one kinematics and five compliance screens).

VS Standard Tire Interface

An interface called STI (Standard Tyre Interface) for connecting tire models to multibody vehicle models was published in 1996 by a German working group called TYDEX. The standard applied for the Fortran language and limited the options for communications between the tire models and multibody vehicle models, but has been modified and extended by third-party suppliers of tire models such as TASS (MF-Tyre/MF-Swift) and COSIN (F-Tire).

In CarSim 2016, the connections are unified between VS vehicle and external tire models with a set of functions called the *VS STI Interface* (VS version of Standard Tire Interface). Those VS STI interfaces are implemented as separate program modules (*VS STI module*). Sample C source code and Microsoft Visual Studio project files are provided to assist users who wish to make interfaces to support their own tire models.

The VS STI module can be compiled for a Windows DLL or for dSPACE DS1006 library file. Therefore, CarSim 2016 RT (DS1006) can support external tire models.

Please note that for the CarSim 2016 release, four-wheeled VS vehicle models with MF-Tyre v7 selected for each of the four tire positions consume nearly 50% of CPU time on a DS1006 system (i.e., consuming nearly 500 micro seconds per calculation step). Therefore, our work thus far has indicated that four-wheeled vehicle models with the MF-Tyre v7 model is guaranteed to run in real-time on a dSPACE DS1006 system. However, more computationally intensive models such as vehicles towing trailers are not guaranteed to run in real-time when both the tow vehicle and trailer are using the MF-Tyre v7 model.

Support for dSPACE RT TASS Tyre Model

In 2014 TNO moved the Delft-Tyre activities to the private company, TASS International. Within TASS, a complete MF-Tyre/MF-Swift product renewal was started with a specific focus on Real Time simulations.

The new VS STI interface supports the dSPACE version of the TASS MF-Tyre/MF-Swift model added in version 7.0. That is, the TASS MF-Tyre model works with CarSim RT (DS1006).

Miscellaneous Improvements

1. Past versions of CarSim have used two 2D Configurable Functions to define “ground” elevation and friction as functions of global X and Y coordinates, which served as a set of default conditions in the case where no road surface was defined. For CarSim 2016, the legacy “ground” Configurable Functions were removed from the VS Solvers. In the event that no road is added from input Parsfiles, the VS Solver automatically defines a straight road during initialization to ensure that the simulation always includes a road. Legacy GUI screens for “ground” elevation and friction were modified to make use of a road surface, maintaining backward compatibility.

2. Two commands were added to conditionally define new reference paths or roads: `SET_IPATH_FOR_ID` and `SET_IROAD_FOR_ID`. This eliminates the possibility of accidentally installing two paths or roads with the same ID, and simplifies the reuse of an existing path or road.

VehicleSim Architecture

As the vehicle models in VS Solver become more extensible, many minor improvements have been made in the architecture.

Functional Mockup Interface (FMI)

CarSim 2016 supports the Functional Mock-up Interface (FMI) in several combinations of options. CarSim can generate a Functional Mock-up Unit (FMU) as a slave in a co-simulation to run in the Windows environment. You can integrate this FMU into other simulation environments such as MATLAB/Simulink with the Pilot Support Package (PSP), starting with Matlab release R2015a; dSPACE VEOS; and others. The options for the CarSim FMU include:

1. Support of both FMI versions 1.0 and 2.0.
2. Support of both “flat” and “structured” naming conventions for import and export variables.
3. Support for self-contained FMU modules that include all input Parsfile data, as well as FMU modules that access external Parsfiles and can write output VS / ERD files.

VS Commands

Several minor improvements were made in the handling of VS Commands and VS assignment statements that occur when scanning inputs from a Parsfile.

1. All self-contained lines can be split using a continuation delimiter of three dots '…' as is done in MATLAB. This is helpful for VS Commands with long formulas, or any lines in which alternate text descriptions are provided for installed variables.
2. A new VS Command `SET_DESCRIPTION` is provided to set the descriptive text associated with a variable, written in the Echo files generated with each simulation. This is intended for use with new variables added with VS Commands or with parameters associated with parts such as sensors and moving objects, to document the use of the model extensions.
3. Assignment statements and VS Commands that define new variables have been extended to include a description for the variable being defined or assigned. In past versions, the variable could be assigned using a number or formula, and the units could optionally be specified if the statement included a semicolon delimiter. These commands have been extended to include descriptive text, specified if a second semicolon is included in the statement. This is a compact way of applying the new `SET_DESCRIPTION` command.
4. Equations added with VS Commands such as `EQ_IN` can be used to assign new values to integer parameters. In such cases, the floating-point value from the formula is rounded to

the nearest integer value. (In past versions, equations from VS Commands such as `EQ_IN` could only be used to set new values for floating-point parameters and variables.)

5. The error handling of VS Commands has been made more aggressive, such that a diagnostic error message is always produced if a command is recognized but has missing information.

Echo File Organization

The Echo file lists values for all parameters that were used in a simulation. Some parameters are grouped into sections, such as the Tire Model, Motion Sensors, Payloads, Roads, etc. Parameters for some model parts are created only if specified with commands, such as `DEFINE_ROADS`, `DEFINE_PAYLOADS`, etc.

The organization of the Echo file has been extended such that all parameters of the same module appear together, regardless of the sequence in which the associated `DEFINE_` commands were encountered.

For most modules, a new parameter has been added to provide a description of the component. For example, the parameter `PAYLOAD_NAME` has a description that is automatically set to the title of the dataset in the CarSim GUI.

Applying VS Commands and Statements with an API function

A new API function was added: `vs_statement`. This has just two string arguments: an uppercase keyword and the text that would follow the keyword in a Parsfile. This will then be processed using the same internal input handling that is applied for lines of text read from a Parsfile. It should handle any self-contained statement that can currently be provided in a Parsfile. (The only type of data not supported is tabular data for a Configurable Function, which requires multiple lines of text.)

This adds support for options such as setting up Import and Export variables programmatically, rather than through the Parsfile.

A new location was also added for callback functions in support of the use of the `vs_command` function during the same stage of the simulation setup as when the Parsfile is read; this is just after the VS Solver has finished reading the Parsfile but before the internal initialization starts.

Miscellaneous Improvements

More Support for Single-Step Integration

An error was discovered in the coding for the single-step Adams-Bashforth 2nd Order Method (AB-2). The effect was that the calculation method was actually the single-step Euler integration method. This bug was introduced years ago and has been present at least since 2008.

The error was fixed, and an option was added to use the Euler method in addition to the AB-2 method.

Recording Time Included as an Output Variable

When viewing plots or animation, the variable shown as “Time” is calculated by VS Visualizer with the formula:

Time = `TSTEP_WRITE` * (i – 1) + `TSTART_WRITE`

where `TSTART_WRITE` and `TSTEP_WRITE` are parameter values written in a header file (VS or ERD) and i is the sample number from the file. An output variable named `T_Record` is now available that can be written into the VS or ERD file. This supports advanced users who wish to transfer simulation results from CarSim into other software.

RT Support for VS File Format

VS Solvers have an option to defer writing to file as the simulation runs. Outputs are saved to an array in memory and written to file at the end of the run. This is required for most for the RT applications. The option is controlled with parameters `OPT_BUFFER_WRITE` and `NSAMP_BUFFER`. This option is now supported when the output file is written with the VS format (both 32-bit and 64-bit versions).

Graphical User Interface (GUI)

The CarSim browser program `carsim.exe` includes new screens to support the model options described earlier. In addition, many improvements were made that involve existing math model capabilities.

Building Paths

Two new GUI screens were added: the **Path: Segment Builder** and the **Path X-Y Coordinates for Segment**.

The new **Path: Segment Builder** screen supports the use of spline tables as segments in a path, along with straight lines and circular arcs. The screen also supports clothoids (Euler spirals) as connectors between segments, in which case it automatically generates an X-Y table to represent each clothoid. Please note that the VS Solver currently does not include native support for clothoids.

The new **Path X-Y Coordinates for Segment** library is used to provide X-Y tables that can be used as segments within a longer path.

The Calculator Tool (available in the lower right corner of the X-Y Coordinate screens for paths and segments) includes an option to create a custom clothoid. Also, the **Calculator: Symbolic** library screen supports this capability.

GPS Import

Three screens that show X-Y coordinates for paths now include a button “Import GPS Coordinates” that allows you to import decimal degree geodetic coordinates from a comma-separated-variable (CSV) file. Each line on the CSV file will contain one “decimal degree tuple.”

This format is supported by several on-line tools, including one developed and maintained by Mechanical Simulation.

The Mechanical Simulation service is found at http://atlas.carsim.com; a link to this web page is also on the screens that support GPS coordinates as input data.

VS/ERD File Utility

CarSim has included an interactive tool used to convert the output files generated by VS Solvers to formats that can be used in Excel and MATLAB. It also supports the conversion of Excel files to ERD format for use with VS tools.

The old tool did not support the new VS file format introduced with Version 9. The tool has been updated to support the VS file formats (32-bit and 64-bit) and provide better browsing of files using the new longer machine-generated names in the CarSim database.

I/O Channel Library Screens

CarSim includes four library screens where lists are specified for Import or Output variables. (These are the I/O Channel libraries for **Import**, **Export**, and **Write**, and the **Plot Setup** library.)

All four of these library screens include browser controls for selecting variables to Import, Export, Plot, or Write to File. The lists of variables are obtained from tabbed text files that are generated automatically based on a selected **Run Control** dataset.

The process of scanning necessary files and refreshing the screen display was improved and made automatic, so the browse information is always shown instantly when viewing a dataset from any of these libraries when a run was selected.

Three of the libraries show output variables, which have multiple labels for each variable. The displays were improved to always show both the short unique name along with a longer, more descriptive name, as well as the user-units of the selected variable. The names are sorted alphabetically by the short names, which are created using consistent conventions (X coordinates begin with ‘X’, Forces begin with ‘F’, etc.).

The screen for the **I/O Channels: Write** library now provides full support for automatic generation of output files formatted for use in Excel (CSV files) and MATLAB (binary MAT files). It also allows for adding a filename suffix if desired.

VS Browser API (Windows COM)

Commands have been added to retrieve and set data on the MixTable user interface control type, used in the new **Path: Segment Builder** screen. See the **VS COM Interface** reference manual for detailed information.

Miscellaneous

1. The option to search for text in the database (**Tools** > **Find Text in the Database**) now has the keyboard shortcut Ctrl+F, and supports two more options: **Find whole words only** and **Current Library only**. The new options are in addition to the previously available options **Allow wildcard** and **Case sensitive**.

2. Screens for adding optional components (paths, payloads, sensors, etc.) write the dataset title into the Parsfile using the new `SET_DESCRIPTION` command to assign the title to the first parameter associated with each component. The effect is that the dataset titles set

in the GUI now appear in Echo files for Payloads, Reference Points, Roads, Motion Sensors, Range and Tracking Sensors, and Moving Objects.

3. The Calculator Tool in the GUI now includes the option to generate a X-Y coordinates for a custom clothoid (spiral) for paths. The **Calculator: Symbolic** GUI screen also has this capability.
4. Symbol Stack support was added for range and tracking sensors and for moving objects. The current sensor number is available with the symbol `<<s>>` and the current object is `<<o>>`. Advanced users can specify output variables for VS Commands and export purposes using these symbolic variables. An immediate effect is that VS Commands for sensors can be used without concern for when the sensor was defined; for example, they can be set to work the same for sensor 5 as for sensor 1.
5. The screens for **Brakes With Boost and Thermal Effects** have been brought up to date by adding a selector for separate left/right properties.
6. The **Multiple Moving Objects** library (renamed from the **Traffic Motion** library) has more fields and controls for supporting multiple paths and roads.
7. Blue links were added to the **Animator: Reference Frame** screen to allow single animator shapes to be specified on the same screen. This allows a single dataset to be used to view or animate simple objects.
8. The **Road: 3D Surface (All Properties)** screen has an additional blue link for animator information.
9. A miscellaneous yellow field was added to the **Sensors for Range and Tracking** screen for advanced users.
10. Splitter controls were added to Generic screens and other screens with miscellaneous yellow fields, to help view the contents when they contain long lines of text.
11. **Calculator: Symbolic** datasets can be linked to other datasets for any link that supports Generic libraries. It can be handy to have a link to the **Calculator** dataset that was used to calculate the numbers in the table. Further, any exported CPAR files involving the dataset with a link to the **Calculator** dataset will include that dataset, in case it is of interest when the CPAR is used in the future.
12. Screens that set values for legacy features in the WinEP or SurfAnim programs that are not relevant for VS Visualizer identify those features with red text. They will be removed in 2017.
13. Legacy screens developed for older versions of the math models that are redundant or do not fully support the current model features have red text indicating that they will be removed in 2017. Examples include the Simple Steering and Suspension screens, and the original Twist Beam Suspension Kinematics screen.

VS Visualizer

VS Visualizer adds features to share data and provides a few capabilities from the legacy tool WinEP that were not previously available.

VSRAP File Support for Viewing Results Without CarSim

A VehicleSim Results Analysis Package (VSRAP) is a file that contains all of the necessary data to animate and/or plot the results of one or more simulation runs. VS Visualizer now supports both creating and viewing VSRAP files. A VSRAP file can be created by VS Visualizer on one computer, and then transferred to another computer where it can be opened with VS Visualizer. The second computer does not need to have CarSim installed, but does have to have the VS Visualizer installed.

An installer for VS Visualizer as a stand-alone application is available from www.carsim.com.

Copying Data Channels to Other Software

Data Channels can now be copied to other software using Drag-and-Drop or the Windows system clipboard. To use this feature, select one or more Data Channels in the Data Manager window and either drag them to the other software package, or press Ctrl-C to copy them to the system clipboard. The Data Channels are copied in CSV text format. The Data Channels can be copied to a text editor, spreadsheet, MATLAB, etc.

Overall, there are now three methods for getting simulation outputs into Excel or MATLAB:

1. Drag them or copy them from VS Visualizer (good for occasional transfers).
2. Use the VS/ERD File Utility to convert a complete VS or ERD file (good to get occasional files or files generated from past work).
3. Link to an **I/O Channels: Write** dataset when making new runs, to generate the Excel or MATLAB files as each new simulation is run (easiest for new runs).

Plotter Improvements

1. Font type, size, color can be specified for Legend, Axis Labels, Tick Numbers.
2. A Fine grid is now supported in addition to course grid.
3. Improved interactive mouse control: middle mouse button and middle mouse button emulation improved.
4. Added ability to copy a bitmap image of the plot to the Windows clipboard, and/or save the plot as a bitmap file.
5. Support auto scaling and manual scaling on each axis independently (e.g. automatically scale the X axis, but specify range for Y).

Camera-Based Sensors (Beta)

VS Visualizer now contains preliminary/experimental features for simulating some types of sensor data, and providing this data to other applications, including MATLAB/Simulink. More information, example programs, MEX functions and a sample Simulink model can be found in the CarSim 2016 database in the folder “Extensions\VSV_Sensor_Beta”. Please view the README file within that folder for more information on what is provided.

Documentation Updates

CarSim has over 50 reference documents containing over 1900 pages. Many were revised for CarSim 2016. Updates of note are listed below.

GUI Documentation

1. The **Paths and Road Surfaces** document was rewritten and extended.
2. The **External Models and RT Systems** document was extended to include FMU/FMI support.
3. The **ADAS Sensors and Moving Objects** document, previously called **Traffic, Target Objects, and Sensors**, was updated and extended.
4. The User Manual for the ERD Converter was completely replaced with new documentation for the replacement **VS/ERD File Utility** tool.
5. The document **Setting up Import and Output Variables** was updated to describe the new library GUI for I/O screens, with automatic browsing of Import and Output variables, and new interface to generate Excel and MATLAB files.
6. The **Calculator Tool for Tables** document was updated to include details on Clothoid generation.

Reference Manuals

7. The **VS Solver Programs** Reference Manual was updated to add a new numerical integration option; the new three-dot continuation option supported for reading text from Parsfiles; and the new in-line format for setting value, units, and description for parameters.
8. The **VS Commands** Reference Manual was updated to add options for setting descriptions for new and existing variables; details on setting units for Configurable Functions; and new VS Commands for paths, roads, and tables, and managing user-defined ID numbers for paths, roads, and tables. The sections with road and path functions were reorganized.
9. The **VS API** Reference Manual was updated to add documentation for the `vs_statement` function and a new callback location `VS_EXT_AFTER_READ`. The organization was updated to have more information about steps in the simulation in Chapter 2, and more information about MATLAB usage.
10. The **VS COM Interface: API of the VS Browser** manual was updated to include new commands.

Tech Memos

11. The tech memo **VS Numerical Integration Methods** was updated with new information about single-step options.
12. The tech memo **Automating Runs with the VS API** was updated to show newer examples.
13. The tech memo **Example: Extending a Model with VS Commands and API** was rewritten to give more information about MATLAB and to use the new `vs_statement` API function.

New Examples

Many of the existing datasets were modified slightly to provide better example parameters or settings. New example simulations that showcase features in CarSim 2016 are organized in categories with names that begin with "* CS 2016" (see the **Datasets** menu from the **Run Control** screen). These categories are:

1. * CS 2016 - ADAS. Simulations involving moving objects and sensors that intervene and/or show alerts during the simulations.
2. * CS 2016 - Euro NCAP Autonomous Braking. Simulations involving Euro NCAP test procedures for vehicles with Autonomous Emergency Braking (AEB), as well as encountering a towed Euro Vehicle Target (EVT).
3. * CS 2016 - External Steer w/ Closed Loop. These simulations show variations of a vehicle with parts of the steering system modeled externally, running in closed-loop (path following) conditions.
4. * CS 2016 - External Steer w/ Open Loop. These simulations show variations of a vehicle with parts of the steering system modeled externally, running open-loop.
5. * CS 2016 – FMU/FMI. These examples demonstrate how to generate FMUs for either FMI version 1.0 or 2.0, as well as Flat or Structured Variable Naming Conventions.
6. * CS 2016 - MF-Tyre/MF-Swift v7 (Extra License). These simulations use the new (November 2015) version 7 of the MF-Tyre/MF-Swift model from TASS.
7. * CS 2016 - Restoring to a Previous State. These simulations demonstrate the "restore state" capability of CarSim.
8. * CS 2016 - Road Options. These examples show some of the new capabilities of CarSim 2016 for managing multiple roads, importing GPS coordinates, and defining paths with segments that include X-Y tables and clothoid spirals.
9. * CS 2016 - Twist Beam. This example has a vehicle with a new version of the twist-beam rear suspension, defined using new GUI screens.
10. * CS 2016 - User Tire Model/STI. The new VS STI interface is used to link to two external tire model (C source code is in the `Extensions\User_Tire` folder).

Backward Compatibility

Most of the new features involve new parameters and output variables. Old datasets do not use these features, so backward compatibility is maintained. However, some of the new features replace older capabilities, making some differences inevitable.

CarSim 2016 will automatically convert databases going back to 7.0 (2007). If you need to work with pre-2007 datasets (version 6), import into a CarSim version between 7.0 and 8.1.1 and then import from there into CarSim 2016.

Simulations with No Road

Prior versions of CarSim allowed a simulation to be run without defining a road surface. Starting with version 2016, all simulations include at least one road surface.

To support existing databases, the old ground libraries are still included, with these changes:

1. The legacy library **Ground Elevation, X-Y Grid** is now named **Road Elevation Map (Legacy)**.
    a. Each Parsfile from this library adds a Road to the model with the `DEFINE_ROADS` command, and a straight-line path added with the `DEFINE_PATHS` command. This ensures that the new road has compatible coordinates: S = X and L = Y.
    b. The Parsfile has data for Z written for the Configurable function `ROAD_DZ`, with `IROAD` set to the number of the new road and `IDZ_ROAD` = 1.
2. The legacy library **Ground Friction, X-Y Grid** is now named **Road Friction Map (Legacy)**. The friction data are written for the Configurable function `MU_ROAD`.

These legacy libraries have not been used for examples shipped with CarSim for years; they may be removed in the 2017 version.

Static Readme Files

Prior versions of CarSim included a set of static text readme files to support browsing for the **Import** and **Export** library screens, and to provide some information about the math model. These files do not contain any information about variables added at runtime. Starting with version 9, dynamic machine-generated text and spreadsheet files are generated as needed from the **Run Control**, **Import**, **Export**, **Write**, and **Plot Setup** screens. The dynamically generated files include all variables of a given type (Import, Output, State Variable), including those added with optional modules and even VS Commands.

Many variables that used to be fixed in the VS Solver are now added at runtime (with many more options), such that the static text readme files are too limited.

Datasets imported from older versions of CarSim for the **Import**, **Export**, and **Write** libraries might have yellow fields that specify old readme files that no longer exist. This does not affect the dataset, but does limit the capability for browsing. In case you want to modify an imported dataset that references an old readme file, you should use the link on the screen to an existing Run

Control dataset. When you do this, documentation files are automatically generated that are current and complete.

Library Titles and Menu Item Names

If you are a long-time user of CarSim, you might notice that some items on the **Libraries** menu have been renamed, as have some of the libraries.

As the number of libraries in CarSim has increased, some of the names have been changed to provide better consistency. For example, CarSim includes sensors for motion detection (e.g., accelerometers), sensors for internal controllers (ABS, ESC, etc.), and ADAS sensors (radar, video, etc.). The Libraries menu has a submenu **ADAS Sensors and Moving Objects** that was renamed from **Traffic, Objects, and Sensors** in earlier versions.

All libraries from version 9 still exist in version 2016, and names in old databases are automatically converted to the new names.

The Original Calculator Tool

Prior versions have included two Calculator Library tools: **Calculator: Original** and **Calculator: Symbolic**. The **Calculator: Original** tool has been obsolete since the introduction of the **Calculator: Symbolic** tool. It has been removed from CarSim 2016.

Bug Fixes and Errata

The following bugs were identified and corrected.

1. The VS Browser often flickered or flashed when processing COM commands in minimized mode. This has been cleaned up.
2. A VS Browser startup crash has been fixed when the initial license screen is forced up as a reminder for renewal when processing COM commands.
3. Past versions of CarSim supported five methods of numerical integration. The Adams-Bashforth 2nd Order Method (AB-2) differs from the others by only doing one calculation of the derivatives of state variables at the designated time step. An error in the code was identified that caused the calculation to use a less accurate Euler integration method. The error was fixed, and an option was added to use the Euler method. Thus, there are now six supported methods of numerical integration.
4. The Road X-Y-Z Coordinates of Edges screen did not always write the Parsfile correctly if the user switched to use the lower table as the reference.
5. VS Solvers did not correctly maintain heading continuity for X-Y Table segments embedded in multi-segment reference paths. The option to include X-Y tables in a multi-segment path was not supported in the GUI, so this bug was not relevant unless advanced users generated datasets outside the GUI. (The bug did not affect single-segment paths.)
6. Sorting of detections for each sensor did not exclude objects with magnitudes of zero unless they were occluded.

7. The option to specify speed of multiple moving objects with target speed vs. station on the **Traffic Motion** screen (now named the **Multiple Moving Objects** screen) generated a VS Command that incorrectly used time instead of station.

8. Preview Points for External Driver Control. In CarSim version 9.0, the number of supported points was extended from 5 to 10. However, the output variable `Lx_Sen_10` contained 9 characters, one more than the maximum allowed. For CarSim 2016, the 10th preview point has been removed.

9. CarSim 9.0 introduced three options in a ring control in the upper right corner of the Road: 3D Surface (All Properties) screen: New road with automatic ID; New road with custom ID; and Replace data for existing road. The third option was intended to allow advanced users to override road properties using Events. In CarSim 2016, this option has been removed, and the recommended method is to refer to the road ID directly in VS Event datasets.

10. Several steering system variables were removed that were obsolete and unused.

11. A review of the tire data used for the Internal Table Look-up tire model indicated that some of the Aligning Moment data was too large. New Aligning Moment data has been created for the following tires: 175/65 R14; 185/65 R15; 205/55 R16; 215/55 R17; and 235/65 R17.

Known Issues

1. The internal algorithm for handling occlusion of moving objects in ADAS Sensor detections is based on several detection points (left edge, closest, right edge) and an implicit assumption that objects are similar in size. When one object is significantly larger than others (e.g., a 50-m wall behind 5-m vehicles), then the smaller objects might be occluded incorrectly. The workaround is to replace the large object with multiple connected smaller objects.