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 2017 Release Notes

VehicleSim Architecture

Modularization

CarSim 2017 represents the third year of a multi-year project to make the software more modular in many aspects. This effort began with CarSim 9.0 (2014), in which model features such as payloads, motion sensors, reference points, and roads were reworked to support multiple instances added with new VS Commands for installing and defining new instances.

Although many of the biggest improvements resulting from these changes will not be introduced until the next version (2018), some are already visible in the overall architecture and in new model features that were enabled.

Echo files give more information about modules.

1. Most model parameters are now listed in Echo file sections for subsystem modules, such as aerodynamics, sprung masses, suspension kinematics, springs and dampers, etc.
2. Most parameter sections of the Echo file include comments with more information about the model. The comments mention all of the relevant Configurable Functions by name. For example, the Powertrain section names all functions used for parts of the powertrain, so users may search (Ctrl+F) for tabular datasets without referring to the PDF Help file.

Functional Mockup Interface (FMI)

FMI support has had minor changes made to work with more external software. These changes typically involve details that are not covered by the official FMI documents and can be done several ways.

FMI support has been added for several RT operating systems, including dSPACE SCALEXIO, ETAS and Concurrent SIMulation Workbench.

New Real-Time Platforms

The VS solvers now support new real-time platforms:

1. National Instruments Linux-based hardware: cRIO and cDAQ. On the cRIO you can use tool LabVIEW or VeriStand. On the cDAQ you can use only LabVIEW because cDAQ does not support VeriStand.
2. Concurrent Redhawk 32bit and 64bit Real-Time system with SIMulation Workbench, SimWB. With SimWB you can work in Simulink or FMI interface.
3. ETAS LabCar on the RTPC 64bit system. The solvers can continue to run on the 32bit RTPC.

Linux versions of VS Solvers

The VS Solvers are now available for use running on the following Linux versions:

- Red Hat Enterprise Linux 7.2 64-bit

- CentOS 64-bit
- Ubuntu 16 LTS 64-bit (does not support USB dongles)

Output Files

The VS Solver can directly write comma-separated variable (CSV) text files as output options in addition to the existing ERD and VS binary outputs. This option was added for the Linux versions where VS Visualizer is not available. However, the Windows versions also supports the option, as does VS Visualizer.

Commands originally introduced in SuspensionSim `LOAD_TABLE_FILE` and `FILE_TO_TABLE` allow data from an ERD/VS file to be copied into a table for use with a Configurable Function. Both of these are now available in CarSim as VS Commands, along with `FILE_TO_CARPET` and `ECHO_LOADED_TABLE`. These are handy when data for a table is obtained from outputs generated from another simulation, or when very large datasets are used that might exist in compact binary form, rather than text form. For example, the surface information for the Mcity proving ground is handled this way.

Licensing

CarSim 2017 introduces some licensing changes.

License Manager Changes

The stand-alone license manager, VSLM.exe, has been separated into separate products for BikeSim, CarSim, TruckSim, and SuspensionSim. The executables are named BSLM.exe, CSLM.exe, TSLM.exe, and SSLM.exe, respectively.

Licensing Interface Changes

Users will notice some changes to licensing behavior with version 2017. Only the first VehicleSim Browser window opened will ask for license selections. Additional windows will not show the licensing dialog. If the original window closes, then a single remaining Browser window will show the license dialog, and that particular window will be in control of the license settings for your machine. To suppress licensing messages over long sessions, run the stand-alone license manager (BSLM.exe, CSLM.exe, TSLM.exe, or SSLM.exe).

Model Features

Paths and Roads

1. The maximum number of paths and roads that can be defined was increased from 50 to 100.
2. Reference paths now support a built-in clothoid option (i.e., the clothoid’s segment type is defined with an equation rather than an X-Y spline table).
3. The reference path used by the driver model has been promoted as the "official path" for use even if the driver model is not active. The path specified with the parameter `PATH_ID_DM` is used even if the driver model is disabled to set the initial vehicle

position when `OPT_INIT_PATH = 1`. This avoids some ambiguities that can occur in driving simulators and scenarios where open-loop steering is specified.

4. The number of incremental `ROAD_DZ` elevation layers for each road surface was changed to a user option, with new layers being added with the VS Commands `DEFINE_DZ_TABLE` and `SET_IROAD_DZ_FOR_ID`. In earlier versions, there were two layers always built in. Now, up to 100 layers can be added for a road. Each `ROAD_DZ` dataset now includes a user ID number, in support of reusing datasets multiple times.

5. GPS data from https://atlas.carsim.com/ is now provided by both Google and HERE Maps. These providers provide denser data when depending on global location. HERE Maps are available in more countries. The atlas user interface now allows for point and click route drawing.

Steering Systems

The steering system was partly re-written for CarSim 2016; the work has been largely completed for CarSim 2017.

1. The new solvers support more options for replacing portions of the system with external models and/or hardware-in-the-loop (HIL) environments.

2. The time constant for power steering boost was formerly applied to the table input (torsion bar torque); it is now applied to the output (boost force or torque). This improves model stability.

3. Power steering boost for the rear axle of Four-Wheel-Steer (4WS) vehicles has been removed. Active steer of the rear axle in CarSim has always operated as a servo-controlled system, so the notion of power steering boost shouldn't apply. Boost never had any effect on vehicle behavior, and was merely written to output. If a user wishes, this can be recreated using VS Commands.

4. New import variables were added: `Imp_F_TIEROD_`*si* and `Imp_M_TIEROD_`*si*, where *s* is L or R and *i* is the axle number (1 or 2). The import variable `Imp_F_TIEROD_`*si* is the force on the rack at the tie rod L1 (L2, R1, R2) calculated in, and coming from, an external model. The import variable `Imp_M_TIEROD_`*si* is the moment on the Pitman Arm due to a load in the tie rod, from an external model.

5. The import options (add/multiply/replace) were added to existing import variables.

Braking Systems

The braking system was completely rewritten to provide more model options. Most of the following options can be selected in arbitrary combinations.

1. Top-level control can be pressure from the master cylinder or pedal force.

2. If pedal force is used, power boost can be included.

3. There are three options for brake actuator dynamics: (a) a time constant for hydraulics; (b) a nonlinear function relating caliper volume and pressure, with transient response specified with a hydraulic resistance; or (c) no dynamics (instant response).
4. Torque at the wheel can be a nonlinear function of pressure with or without thermal effects.
5. Import and export variables have been added to support various places for inserting ABS controllers and other parts of the system that might be defined with external models or HIL.

Powertrain Systems

An ESC/TCS torque control has been added to the CarSim solver. The CarSim powertrain model modulates the internal throttle to meet the engine torque request by ESC/TCS which is fed through a new import variable (`Imp_MEngine_ESC_Request`). An importable control state (`Imp_Engine_Con_State`) determines either to reduce the engine torque to meet the request as maximum (limiting the torque by ESC/TCS) or to increase the engine toque to meet the request as minimum (e.g. autonomous throttle for hill-climbing systems). One new output variable (`M_Eng_Req`) represents the driver-requesting engine torque.

Support for Dual Tires

Dual tires are now supported in CarSim. A library has been added to specify tires for two or four wheels, where either of the two axles can have single or dual tires (i.e., front and rear axles on the vehicle). When using the dual tire option, the tires on each axle can either be the same tire dataset, or a mixed group of tires can be selected in which each tire dataset is different. The name of the new library is **Tires: 2 Axles with Support for Duals**. This screen is used for the lead unit, 1-axle trailer, and 2-axle trailer when dual tires are to be used.

If a link is made to a dataset from this library, then the VS Solver for CarSim is defined to support duals; if not, the VS Solver does not recognize parameters involving duals. Note that this applies to output variables as well.

For example, output variables for use with dual tires (i.e., those that contain inside and outside tire indices) will work for the dual tires, but these output variables will not work for the case in which dual tires are not selected (i.e., only single tires at each corner). In this case, the end-user will have to select the correct output variables for single tires vs. the dual tire option.

Initialization

An initialization typically includes an estimation of a quasi-static initial condition for the sprung mass pitch angle, sprung mass roll angle, suspension deflections, and tire deflections. The estimation method has been improved for 2017.

Compared to CarSim versions 2016.2 and earlier, the difference observed in the CarSim 2017 initialization is small when the road surface is flat and level, but can be noticeable when the surface has grades or 3D features. The effect can be more significant when there is a trailer.

ADAS Range and Detection Sensors
Three import variables were added to dynamically aim range and detection sensors. As with other imports, these can be controlled from external software or via VS Commands.

The import variables are:

- `Imp_Pitch_Aim_Sensor_i`
- `Imp_Roll_Aim_Sensor_i`
- `Imp_Yaw_Aim_Sensor_i`

where *i* represents the ADAS sensor (i.e., 1, 2, 3…20)

Support for TASS Rigid-Ring Tyre Model on dSPACE RT
CarSim 2017 supports MF-Tyre/MF-Swift version 7.1 which adds on the Rigid-Ring option for both Windows and dSPACE DS1006 systems.

> **Note** Rigid-Ring involves additional degrees-of-freedom with stiffness which may cause numerical instability as a combined effect with vehicle parameters. The numerical instability can be avoided using a smaller time step down to 0.5ms. In order to keep the calculation in real-time on DS1006 systems, consider alternate integration methods such as Adams-Bashforth 2nd order (AB-2) or Euler method.

Graphic User Interface (GUI) and Database

Visual Appearance
The sizing of text and windows has been redone to work with Windows OS settings for display.

1. The font used throughout the GUI is now Segoe UI, which is the standard font used in the US versions of Windows. This font is also available in international versions. Most screens and popup dialogs have been redone such that displays are easier to read regardless of the Windows preferences.
2. The sizing options for all GUI windows are now tied to the Windows display preference. Text shown in menus and the window title always matches the OS preference (e.g., 100%, 150%, 300%, etc.). An option has been added to the legacy Viewing choices to also use the Windows display preference to size GUI windows. This provides full compatibility with newer high-definition monitors and laptops with small pixels.

New Screen for HUD (Heads Up Display) Objects
A new GUI Screen has been created to facilitate the rapid creation of HUD components for the simulation of Console and ADAS output controls for the Visualizer. This makes it easier to create new ADAS videos for blind-spot detection, lane departure, crash warning, etc.

New Results Section in Database

A new `Results` folder in the database provides the location for all output files generated when running simulations. This provides several advantages:

1. Output file names can be given short default names that are easily exchanged and manipulated by external software.
2. The `Runs` folder now contains only database files (Parsfiles and some image files).
3. The new organization is well suited to support the new feature to make multiple simulations from a single **Run Control** dataset as described in the next subsection.

Multiple Simulations from a Single Run Control Dataset

The **Run Control** screen has been extended to support the reuse of a single **Run Control** dataset to make multiple simulations without overriding previous results. This feature greatly extends the options for users running the models from external programs such as Simulink, especially when conditions are varied outside the scope of the CarSim database. For example, multiple tests made with HIL systems and driving simulators.

Output Files can be included in CPAR

There is now an option to bundle run output files when creating a CPAR from the **Run Control** screen, or when adding runs from the **Library Tool**. This way a user importing the CPAR can immediately view results, even if they were made with advanced licenses not available for the person importing the data.

Cloning Options Added to Dataset Rename

Options to clone an existing dataset, as well as re-routing links to the new copy, and deleting the old original, have been added to the **Edit > Change Title or Category of This Dataset** command (**CTRL + T**).

Updated Screens

1. The **Plot Setup** and **Plot Format** screens were updated to match new capabilities of VS Visualizer.
2. The **Brake System** screens were redone to support new modeling options. Legacy “Dynamic Braking” screens are now unnecessary; they were modified to remove obsolete features, and have red text recommending that legacy datasets be copied to the updated libraries.
3. CarSim includes two screens for specifying the properties of a hitch, which include nonlinear torsional springs, linear damping, and torsional friction. Both screens were updated to provide all modeling options, with more detailed right-click information.

VS Visualizer

Camera Sensors

VS Visualizer now provides camera-based sensor information in shared memory buffers that can be used by other “client” programs. These are called Camera Sensors, and are intended for use in ADAS simulations. Any number of Camera Sensors can be created and attached to any Reference Frame (or Transform) in the animation.

Sensor data types available include color/visual, distance, and surface normal vectors. Camera data is computed per-pixel at arbitrary resolutions (independent of the main animation window).

ANSI-C and C++ interfaces are provided (via DLL) for creating client programs to analyze the camera-based sensor data. The lowest level ANSI-C API is referred to as the “VS Shared Buffer Core Library.” A utility library (VS Shared Buffer Utility Library), which is built upon the Core Library, adds ANSI-C convenience/utility functions and an object-oriented C++ “VsSensor” class that simplifies and automates low-level management of the shared buffer system resources. Header, LIB, and DLL files for these interfaces can be found in a folder named `vs_sb` within the `Programs` folder.

Two additional components are provided, which are built upon the Core and Utility libraries:

- SensorViewer — a program for viewing shared buffer data for the purpose of demonstration, development, and debugging. This program (`SensorViewer.exe`) can be found in the VS Visualizer program directory.
- Simulink S-function — a Simulink interface to access the camera data in the shared buffers.

CSV Input

VS Visualizer can now handle CSV files for making plots and generating animations. This supports the new capability of VS Solvers for generating CSV files as another alternative to ERD and VS files.

Be aware that less labeling information is available for plots based on CSV files. The convention for spreadsheets is to have the first row provide names of the variables, with the rest of each column being a time history of the named variable.

New S-Function for Live Video with Simulink: S-Function2v

A new Simulink S-Function has been added that allows a live video mode. The video runs at the simulation speed (usually faster than real-time) to enable rapid viewing of simulation results during a run. This capability is intended for lengthy simulations that might run for minutes or hours, in which case it is helpful to visually confirm that the simulation is running as intended (for example, that the vehicle has not run off a road and is driving in a circle).

Unlike the first- or second-generation S-Functions that are used in other CarSim/Simulink examples, only one instance of this S-Function block can be used in a Simulink model; attempting to use two will cause conflicts.

Note The new S-Function is not recommended for short-duration simulations (those taking less than a minute to finish). This is because it is far more efficient to view results after the simulation has finished, when controls can be used to pause, reverse, use slow-motion, etc.

Plotting

A number of improvements were made for the plotting within VS Visualizer:

1. Commands were added to manage the time using in zoomed plots and in the time control for the camera(s). Type ‘t’ to pan the active plot to the time used by the camera; type Shift-T to change the camera time to match the active plot.
2. The labels used to identify datasets in the legend are handled automatically to account for titles, categories, and file names. These support the new option for making multiple simulations from a single Run Control dataset.
3. Support was added for options available in the legacy WinEP program such as showing a frame around the plot area, not showing axes, and providing a fine grid.

Documentation Updates

Minor edits were made throughout the Help documents. Documents that were changed to better describe the new version are listed here.

The following Guides and Tutorials were updated:

1. CarSim Demo Tutorial
2. CarSim Quick Start Guide
3. Running a VS Math Model in Simulink (previously a technical memo)

The following Reference Manuals were updated:

4. VS API
5. VS Commands
6. VS COM Interface
7. VS Solver Programs
8. VS Visualizer

The following Screen documents were updated:

9. ADAS Sensors and Moving Objects
10. Animator HUD (new)
11. Animator Reference Frames
12. Animator Vehicles and Sensor Targets

13. Brake System
14. Driver Control Screens
15. Import and Export Variables
16. Paths and Road Surfaces
17. Payloads
18. Plot Screens
19. Powertrain System
20. Procedures and Events
21. Steering Systems
22. Suspension Systems
23. Tire Models
24. Tools > Calculator Tool for Tables
25. Tools > Batch Runs
26. Tools > Preferences
27. Trailer Hitches
28. Vehicles

The following Technical Memos were updated:

29. Camera Sensor Example (new)
30. Example: Running Multiple VS Vehicles in Simulink
31. VS Camera Sensor S-function (new)

The following Release Notes from Version 9 were updated:

32. Version 9 Backward Compatibility: Database and Automation
33. Version 9 Backward Compatibility: Paths and Roads
34. Version 9 Backward Compatibility: Payloads
35. Version 9 Backward Compatibility: Reference Points and Motion Sensors

New Examples

Categories and Titles in the Run Control Library

In the **Run Control** library, names and categories have been changed for existing simulation examples to reduce the size of the **Datasets** menu. Many of the steering system datasets and some tire datasets were updated for CarSim 2017 using improved data settings.

When changes were made to any dataset that existed in 2016.2, a duplicate was made and a different title was given to the new dataset. This was done in support of version control software that tracks changes to files based on file titles; any duplicate dataset was given a new unique name that cannot be mistaken for a file from another database or older version. Higher level datasets that make use of new component datasets (e.g., a vehicle assembly that uses an updated steering system) were also copied and given new names. In most cases, older datasets are deleted from the 2017 database.

Examples for New 2017 Model Features

Some of the new features apply for all examples (e.g., the new Echo file layout) and others can be easily set for any simulation (e.g., specifying a CSV output file rather than a binary VS/ERD output). New example simulations that showcase features in CarSim 2017 are organized in categories that begin with "* CS 2017" (see the **Datasets** menu from the **Run Control** screen). These categories are:

1. * CS 2017 – Dual Tires. A new example has been added to demonstrate dual rear tires on a full-size pickup truck.
2. * CS 2017 – Engine Torque Control. Two new import variables and one output variable were added that involve engine torque management and are intended to be used for traction control (TCS) and electronic stability control (ESC) systems.
3. * CS 2017 – External Powertrain Model. This example demonstrates a co-simulation with GT-SUITE full powertrain model (RWD) which is connected to CarSim through FMI (Functional Mock-up Interface). The GT-SUITE full powertrain model (`FullPowertrain.gtm`) is under folder `CarSim_Data\Extensions\GTI\Full_powertrain\`. The simulation should run on the GT-SUITE model and GT-SUITE software and its license are required to execute this example.
4. * CS 2017 – External Steering Models. Fifteen examples are given that show how to replace parts of the steering model and extended, such as replacing 1D tables with 2D tables.
5. * CS 2017 – Live Video w/ Simulink. One example demonstrates the capability of the new CarSim S-Function2v to show live video streaming when running with Simulink, another uses the new Camera Sensor capability of VS Visualizer from Simulink (the Computer Vision System Toolbox from The MathWorks is required for this example).
6. * CS 2017 – Loaded Tables. This category contains two examples that involve road roughness tables. One provides the data in the traditional method (a text Parsfile). The other access the data from a binary file using the new commands `LOAD_TABLE_FILE` and `FILE_TO_TABLE`.
7. * CS 2017 Mcity. This category contains a collection of datasets which also make use of the new `LOAD_TABLE_FILE` command to build a unique environment, Mcity. Along with a dense set of elevation information, Mcity includes extensive graphics based on the physical test area in Ann Arbor.
8. * CS 2017 – Multiple Simulation. This category contains three examples that demonstrate the capability of saving multiple sets of simulation results from a single Run

Control dataset. The first example has multiple runs made with no external software. The second example contains two ABS Simulink runs which have different brake release points. The third example were run using MATLAB simple steer control and two runs were produced by using different SWA control gain settings.

Python interface for the VS API

The VS API can be accessed using many programming languages, including Python. A folder was added to the database `Extensions\Custom_Py` with example Python code to access the VS API. Example run setups were added to the **Run Control** library in the category: **External Control: COM, FMU, MATLAB, Python, VB**.

Bug Fixes and Errata

The following bugs have been fixed.

1. In certain transient situations, the closed-loop speed controller could set the master cylinder pressure negative, attempting to generate positive wheel torque. The brake moment always resists the rolling direction, so the effect was that the controller generated brake torque. Although the physics remained valid, this bug sometimes affected the controlled behavior in an unintended manner.
2. References to road and path ID numbers in algebraic expressions could crash CarSim if the ID was specified with a number, e.g., the function call `ROAD_SSTART_ID(1212)`. The problem was that the ID (e.g., 1212) would not be valid until the model initialized, which is after the files are read. The error handling was changed for 2017, and some automatic preprocessing was added to allow the user of ID numbers in algebraic expressions.
3. The hitch model includes options for friction in the three hitch moments. The friction support variables were not updated properly, causing distortion in the friction part of the moment.
4. In versions prior to 9.0 (2014), tables made with the VS Command `DEFINE_TABLE` supported the option to use an algebraic formula to calculate the result from the Configurable Function. This capability was accidently removed in versions 9 and 2016; it is now restored.
5. When off-center payloads (those with non-zero Y coordinates) were added to the model, the XY and YZ products of inertia due to the location of the masses were calculated incorrectly.
6. The speed parameter passed to the Steering Parking Torque configurable functions was the signed value of speed. This could produce unintended results for a vehicle traveling in reverse (negative speed). The function now receives the absolute value of speed.
7. Setting tire relaxation length to zero could cause a divide by zero error at initialization.
8. A parameter `VLOW_DAMP_Y(`*iaxle, iside*`)` for advanced use (not supported by the GUI, set only by typing into yellow fields) had a default value of 0.5 km/h. This parameter causes tire lateral force damping to be increased below the specified speed. It can be

valuable in certain applications such as driving simulators. However, it can also contribute to instability in certain situations, notably in vehicles with large caster angles. The default value for this parameter has been changed to zero, but the parameter itself has been retained for advanced users to set by typing into a miscellaneous yellow field.

9. The limit on power steering boost was only applied when the sign of boost force or torque was positive. It now applies in both directions.

Backward Compatibility

CarSim 2017 will automatically convert databases going back to 8.0 (2009); this conversion takes care of the screen parsfiles and keywords such that the GUI will recognize old keywords and update them appropriately. Please note that if you need to work with pre-2007 datasets (version 6), you must import into CarSim version 8.0 and then import from there into a newer version such as CarSim 2017.

External Simulink ABS Controllers and the Dynamic Brake Option

The brake model in CarSim was redone for 2017 to merge options that were available in past versions as a “simple model” and a “dynamic model.” The dynamic model was introduced when options for activating Import variables were limited, so it included several options for using/ignoring sets of import variables. The parameter `OPT_BRAKE_MODEL` was used to choose between Basic, Dynamic, Dynamic + External “Type A” ABS Controller, and Dynamic + External “Type B” ABS Controller.

The Type A and Type B options are handled in version 2017 using ordinary Import and output variables. However, the names had to be changed because the older version used internal logic that was inconsistent with the methods used for the past 10 years for managing regular import and export variables.

Expect that the interfaces to Simulink models created for Type A and Type B ABS controllers will need to be adjusted for use in 2017. The new database includes examples showing how the legacy interface is handled.

Occlusion Calculation for ADAS Sensors and Sensor Targets

Updates have been made to the calculation of occlusion for sensor targets when used with the ADAS Range and Tracking Sensors. The internal algorithm for handling occlusion involves sorting the objects based on proximity. In CarSim 2016.2 and earlier, the sorting was based on the distance to the closest point for each object. An effect is that sometimes a large object such as a wall would be used incorrectly to occluding smaller objects.

The sorting order is changed for CarSim 2017 to use the furthest of the points found by checking both the left and right edges. This solves the original problem of having a false occlusion. However, a new effect (less common) is that sometimes small objects that should be occluded are not. For example, in the ADAS examples that use sensor targets for traffic vehicles and lane edge markers, the ADAS sensor may detect the lane marker if it is slightly behind the corner of a traffic vehicle. This is noted later in the Known Issues section.

Initialization using a Path

When the vehicle is initialized relative to a path, there were some ambiguities in past versions if the driver model was not enabled. In 2017, the parameter `PATH_ID_DM` always identifies the path used to set initial position and heading. `PATH_ID_DM` is always set by default to the user ID of the most recently defined path. If another path should be used, then `PATH_ID_DM` must be set explicitly after all path datasets are loaded. There may be a few datasets in past versions where this was not true.

Paths and Road Surfaces

1. The user ID numbers for paths and road surfaces (`PATH_ID` and `ROAD_ID`, respectively) are restricted in version 2017 to either match the internal ID (e.g., `PATH_ID(3)=3`) or they must be assigned values 999 or greater. This is done to avoid conflicts when the maximum number of paths and roads is increased in future versions beyond the current limit of 100. Past versions did not restrict the values that could be set to `PATH_ID` and `ROAD_ID`. If datasets from previous versions used ID numbers that did not match the internal ID numbers and were less than 999, then they must be renumbered to work in version 2017.

2. In past versions, the road surface model has included two incremental DZ layers for each road, provided with the Configurable Function `ROAD_DZ`. The 2017 version extends the model to include any number of layers ranging from 0 to 100. Links in old datasets to a flat `ROAD_DZ` dataset are unnecessary. The new examples from Mechanical Simulation do not link to a layer dataset unless it adds 3D information. However, links in old databases are not automatically removed. They do not cause any harm, but they do add some complexity that is not needed.

3. The indexing used to access a specified `ROAD_DZ` dataset was changed from two indices (road ID and 1 or 2) to one (`IROAD_DZ`). A user ID was added to the `ROAD_DZ` Configurable Function. When road surfaces are assembled from the GUI, the changes are handled automatically. However, advanced users who specify road information without going through the GUI might have to change some files.

4. Versions prior to 2001 used a rectangular grid to provide a 3D surface. Two legacy library screens existed for specifying elevation and friction using tables of X-Y coordinates. In version 2016, the grid option was removed, as it is the same as a straight road (S = X, L = Y). These X-Y grid screens had red text announcing they would be removed in version 2017, and they are now gone.

Import Variable Names

User-defined import variables should be named with a prefix `IMP_`, e.g., `IMP_CTRL_1`. If the prefix is not included (e.g., the import is named `CTRL_1`), then CarSim cannot properly scan the data when connecting to external models via Simulink, FMU, and other methods.

Past versions had a checkbox on the **Preferences** screen labeled “Scan for IMP_ and EXP_ commands and send last to math model.” If unchecked, the scanning was not performed, and newer connection methods could not be used. This checkbox was removed from the 2017 version, and an error message is generated if an import variable is defined that does not begin with `IMP_`.

The recommended syntax is:

```
define_import Imp_NewVar
```

```
import Imp_NewVar Add 0
```

where the number zero (0) represents the user-defined initial value of the import variable at the beginning of the simulation. Depending on the intended use of the new import variable, this initial value can be any number; supported options include integers (0, 1, 2, etc.), fractions (2/3), and symbols recognized by the GUI such as `pi`. If the new import variable is activated correctly, it is written at the end of the Echo File.

Retired and Renamed Variables

Older versions of CarSim included variables for supporting animation of dashboard dial gauges. VS Visualizer provides native support for HUD displays making the old method obsolete. The old variables were named `A_Comp`, `A_Speedo`, and `A_Tach`. If needed, they can be redefined with VS Commands.

Another set of output variables was provided to support animations of estimated lower-control arm motions. These were named `RolLcaL1`, `RolLcaL2`, `RolLcaR1`, and `RolLcaR2`. These variables were introduced before VS Commands existed, and can be replaced using VS Commands if needed.

CarSim includes two variables per wheel that indicate the steering angle and angular rate about the kingpin axis. In previous versions, these were named `AStKP_`*si* and `VStKP_`*si* where *s* is L or R and *i* is the axle number (1, 2, …). They are renamed in 2017 to be more consistent with other steer variables: `StrKp_`*si* and `AVzKp_`*si*.

Support for SurfAnim Animator and WinEP Plotter

The legacy animator SurfAnim and plotter WinEP are no longer supported, in order to provide full support for VS Visualizer. For example, the use of translucent 2D shapes to resemble shadows in SurfAnim is obsolete given the true shadows shown in VS Visualizer. GUI controls for workarounds needed for the legacy tools were identified in past versions with red text, announcing they would be removed in 2017. They are removed in 2017.

WinEP supported a moving average filter option, with a **Plot Transform** screen for settings. It has not been used in any examples for over ten years and is not supported in VS Visualizer. That screen has been removed.

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 smaller than others (0.2-m target used for lane detection vs a 5-m vehicle), then the smaller objects are sometimes not occluded as expected.
2. Eight variables that are sometimes used to locate a virtual video camera were removed from CarSim 2016.1 to avoid situations where they were not applicable and calculations

might fail. The variables can be installed with a new command `INSTALL_CAMERA_OUTPUTS`. Databases for previous versions do not include this command. Hence, the command must be added to older datasets in order to use the road-based camera tracking option.

3. If the extended table-lookup tire model is used to include nonlinear inclination effects, the influence of friction is not applied to inclination effects.

   Tire forces due to camber are not influenced by friction in the same way as tire forces due to slip. The issue is that camber-induced forces do not saturate in the absence of slip in the current model. (The similarity method used for handling friction uses combined lateral and longitudinal slip, but does not include large inclination effects.) The sensitivity to friction is still under investigation.

4. The Road: X-Y-Z Coordinates of Edges screen can crash when only two rows of data are used for each table. The requirement is that three or more rows of data must be used for each table when entering the edge coordinates. Work is on-going to improve this screen.