carSIM: Math Models

CarSim math models represent the dynamic behavior of four-wheeled vehicles, possibly towing a trailer. The VehicleSim® (VS) Math Model architecture is used.

CarSim VS Math Models are built using dynamically linked VS Solver library files, available for 13 operating systems: Windows (32- and 64-bit), Linux (5 versions), and real-time platforms used for hardware-in-the loop (9 systems). The models work well with other software (Simulink, LabVIEW, FMI, ETAS ASCET, EPIC Unreal, Custom programs, etc.) for automation or extensions to the models.

A basic CarSim model runs more than 15 times faster than real time on a typical Windows computer.

Multiple vehicles may be simulated simultaneously using a single VS Math Model, or by running multiple VS Math Models in parallel using external software such as Simulink.

Vehicle Math Models

Vehicle Configurations

- The basic CarSim multibody model has a rigid sprung mass with two suspensions.
- A trailer may be added with up to three suspensions with load-sharing effects. An optional license is needed.
- CarSim with frame twist includes frame rails to add torsional compliance. An optional license is needed.
- CarSim with powertrain mounts to includes dynamic engine movement. An optional license is needed.

Configurable Table Functions

- Potentially nonlinear relationships between variables are defined with VS Configurable Functions that can be:
  - Constants
  - Linear coefficients
  - Nonlinear tables with several interpolation methods involving one or two independent variables
  - User-defined symbolic formulas
- Configurable Functions include offset and gain transform parameters for dependent and independent variables.
- There is no built-in limit to the length of tables.

VS Reference Paths

- A VS Path defines an S-L coordinate system (S = distance along path, L = lateral distance from path).
- Each path is a sequence of segments, where each segment may be: straight, an arc, a clothoid, or an X-Y table.
- VS Paths are used for driver controls, locating traffic vehicles, and defining 3D road properties.
- A VS Math Model supports up to 500 VS Paths.

Driver Controls

- Driver controls can be handled by built-in controllers, VS Command equations, or imports from external software.
- The closed-loop driver model (DM) can steer to follow a target path, which can be changed during the run.
- The DM controller handles forward and reverse speeds.
- Closed-loop Speed Controller (SC) for throttle and braking based on target speed or acceleration, or path.
- SC path preview uses acceleration limits, curvature of the target path, and 3D road geometry (banking, grade, curvature).
- Gear shifting and clutch controls can be handled with shift schedules and automatic throttle-clutch interactions.
- Closed-loop and open-loop controls can be combined to simulate ADAS intervention systems.
- Steering wheel control can be by angle or torque.
- Open-loop braking can be pedal force or fluid pressure.

3D Road Geometry and Friction

- The 3D ground surface includes 3D geometry, friction, and a tire rolling resistance coefficient.
- The 3D surface may be a set of VS Roads or VS Terrain.
- Up to 200 VS Roads may be built with components:
  - VS Reference Path for S-L coordinate system.
  - Configurable Functions for elevation and friction using S-L coordinates and variable-width tables.
  - Boundaries to connect adjacent VS Roads.
- VS Terrain provides a single mesh-type ground surface, created with VS Scene Builder with several options:
  - Create interactively by dragging 3D Tiles.
  - Import datasets from OpenDRIVE format.
  - Import 3D FBX files from other software.
- Road profiles “wander” to follow the vehicle tires, providing high-frequency road roughness inputs. Road profiles are measured routinely by some road agencies.

Wind and Aerodynamic Effects

- Six aerodynamic forces and moments are applied to the sprung mass (both sprung masses if there is a trailer).
- These forces and moments are shaped by Configurable Functions of aerodynamic slip, pitch, and ride height.
- Ambient wind speed and heading can be set with tables, runtime equations, or imported from other software.

Suspensions

- Suspensions can be generic/independent, solid axle, independent with virtual steering axis, or twist beam.

    - Wheel movement in a generic/independent suspension depends on jounce on both sides.
    - Wheel movement in a virtual steering axis suspension depends on jounce and steering rack travel.
    - Axle movement in a solid axle suspension depends on axle jounce and roll
- Independent and solid-axle suspensions can be either steered or non-steered.
- All suspensions have full nonlinear kinematical behavior and can be asymmetric.
- Suspension springs and dampers are nonlinear. The springs include hysteresis due to friction.
- All suspensions have lateral and longitudinal compliance; every wheel has toe and camber compliance.
- All compliances can be represented with linear coefficients or nonlinear configurable functions.
- Separate forces are included for jounce and rebound stops.
- Suspension roll moments include a nonlinear auxiliary roll moment and linear coefficient roll damper.

Steering System

- The interactions between the suspension, steering, tire, and ground are handled with a multibody model that uses an inclined kingpin axis, or 2D kinematics tables in the case of the virtual steering axis suspension.
- The steering model includes specific details for rack-and-pinion and recirculating ball-type systems.
- Steer angle of each road wheel is available as measured in a K&C rig or as rotation about the kingpin axis.
- The steering system includes detailed options for manual or dynamic power boost, including column assist.
- The steering system includes hysteresis, compliance, inertia, and damping.
- Special equations are used for low-speed conditions to simulate ground friction steer torque.

Brake System

- Brake control can be set with pedal force (with or without boost) or master cylinder pressure.
- The control input pressure from the master cylinder is proportioned for each wheel-end brake actuator.
- Brake torque is modeled as a nonlinear function of actuator pressure and optional thermal effects.
- The brake system can use a built-in ABS controller or connect with external programs such as Simulink.
- Special equations handle wheel lockup to obtain the correct reaction torque and avoid numerical instability.

Tires

- CarSim includes several installed tire models:
    - A fully nonlinear and asymmetric table-based model
    - An extended model (more tables for camber effects)
    - A terramechanics-based (soft soil) model (64-bit Windows, single tires only)
    - MF-Tyre from Siemens
- CarSim supports some external models (license required):
    - MF-Swift from Siemens
    - FTire from COSIN
    - TameTire from Michelin
- User-defined tire models can be connected with VS STI.
- External tire models can calculate forces at either the ground contact point or the wheel center.
- Variable friction conditions are handled using similarity, to maintain both linear and limit properties of the tire.
- Transient effects of rolling are included using relaxation length, which can be constant or a nonlinear function of vertical force and slip.
- Special equations are used at low speeds.
- Tire contact can be handled with one to four points.
- Dual tires are available for all wheels.

Powertrain

- CarSim supports FWD, RWD, and AWD. The speed controller can also apply torque directly to the wheels if a full powertrain data model is not needed.
- The powertrain supports internal combustion (IC), electric motor + battery, and hybrid (IC + electric motor + battery + planetary gear).
- The hybrid and EV powertrain models support battery generation.
- Electrified axles are supported with either one or two motors per axle.
- IC engine torque is defined with a 2D Configurable Function based on RPM and throttle.
- The engine feeds torque to the transmission through either a hydraulic torque converter or a mechanical clutch.
- The transmission converts torque and speed based on the current gear selection, with spin inertias and efficiencies that depend on the gear selection.
- Continuously variable transmissions (CVT) are supported.
- The transfer case unit and differential models are similar. All have four model options: (1) Always locked. (2) Viscous coupling. (3) Coupling applied using a clutch. The clutch can be controlled externally or with built-in logic. (4) Yaw control differential system having two clutches with reduction gears in parallel over a differential. The torque distribution may be controlled left and right, or front and rear.
- The transfer case has a torque bias for non-locked options.
- Twin-clutch is an alternative to an axle differential.

- Torsional compliance of the driveline is included.
- Fuel consumption is defined with a 2D table.
- CarSim supports external models from GT Suite.

Sensors and Traffic

- The models include several kinds of virtual sensors that detect various types of vehicle motion, including acceleration, speed, and jerk.
- Up to 200 moving objects can be added that are updated automatically to convert simple path-based commands into full 3D geometry.
- Motion of an object can be constant, set by specifying speed, controlled by acceleration (simple physics), set with algebraic equations, or imported via import variables.
- The objects can be recycled for extensive runs, to reappear after they go out of view.
- Objects that move based on speed or acceleration support off-tracking, for realistic low-speed traffic turns.
- Objects used to represent traffic vehicles support brake lights and reverse lights.
- Objects may be rectangular, circular, segment (e.g., signs), or polygonal.
- Segment objects have a limited viewing angle, to mimic signs and signals with limited visibility.
- Up to 99 ADAS range and detection sensors can be included that detect the moving objects. An optional license is needed for sensors (but not for objects).
- Each detection includes 24 variables that can be exported to external controllers (e.g., ADAS conditions).
- Objects can block each other (occlusion). The sensor detection variables respond only to the portion of the object that is within the field of view.
- Detection sensors can be placed on the vehicle or moving objects (to detect and simulate collisions).
- Objects may be attached to vehicle sprung masses to support ADAS simulations with multiple vehicles or provide details of collisions with pedestrians.

VS Math Model Input and Outputs

Input Data Files

- CarSim reads all input from text files that are normally generated automatically by the Browser/GUI. These files can also be made externally for advanced applications.
- Input files for CarSim follow a simple keyword-based format called the Parsfile. CarSim can recognize thousands of keywords when processing input files.
- Each input line can optionally specify alternate units for a parameter.
- Values can be assigned directly to model parameters with numbers, numerical expressions (e.g., 1/16), or symbolic algebraic expressions involving other model variables.
- Parsfiles support the INCLUDE capability, allowing advanced applications such as design of experiments (DOE), sensitivity, and customized automation methods.

Output Variables

- CarSim generates from about 600 to thousands of built-in output variables, depending on whether there is a trailer, sensors, traffic vehicles, etc.
- A subset of the available outputs can be specified at run-time, to control the size and organization of output files.
- Writing to file can be enabled and disabled during the run, to save only interesting results from long simulations.
- CarSim provides a GUI for browsing the lists of available variables, sorting by several categories.
- All variables are described in documentation files in both text and spreadsheet format.
- Output files may be written in several binary forms (32-bit and 64-bit) or CSV (text) spreadsheet format.
- Output variables are used for several purposes:
  - Make plots that show vehicle behavior.
  - Motion information for video visualization.
  - Input to other post-processing software.
  - Export to other software during the simulation.

VS Commands and Python

VS Solvers include the VS Command scripting language for customizing the model and its operation. VS Commands are supported in all versions, including real-time systems.

- VS Commands can add new equations at several locations in the sequence of simulation calculations.
- VS Commands can add new parameters, output variables, and state variables as needed to extend the model.
- VS Commands can add new differential equations.
- VS Commands can add new functions to simplify other formulas or series of equations.
- VS Commands can define new units.
- VS Events monitor custom formulas to trigger the reading of a new Parsfile to change values, modes, etc. This is used to script complicated procedures.
- The Windows and Linux versions provide embedded Python in support of full programming options.

Working with Simulink® and External Software

- The CarSim VS Math Model is made with functions from a dynamic library file.
  - The CarSim GUI runs VS Math Models directly.
  - CarSim includes MATLAB/Simulink S-Functions.

    - CarSim works with LabVIEW.
    - CarSim can generate functional mockup units (FMU) to run under the functional mockup interface (FMI).
- MATLAB, Visual Basic (VB), and other languages can operate the Browser/GUI using Windows COM.
- CarSim has a `LINEARIZE` command to generate linearized A, B, C, and D matrices for use in MATLAB.
- The VS SDK (software development kit) is available for Windows and Linux. It includes numerous application program interfaces (APIs):
    - The VS Solver API is used to build and interact with VS Math Models from C/C++ and languages that can load a DLL file (Python, MATLAB, VB, etc.).
    - The STI API helps connect external tire models.
    - The Shared Camera Buffer API accesses 3D information from VS Visualizer cameras.
    - VS Output API helps read and write output VS files.
    - VS Table API supports Configurable Functions.
    - VS Terrain API works with VS Terrain files.
    - VS Connect enables co-simulation between EPIC Unreal and Simulink.

Import Variables

- Calculations from external models and measurements from hardware-in-the-loop (HIL) can be imported into CarSim. These include most forces and moments, fluid pressures, controls, ground geometry at each tire, etc.
- VS Math Models can import values for hundreds of built-in variables.
- Most of the import variables can be combined with native internal variables with one of three modes:
    1. replace the native variable,
    2. add to the native variable, or
    3. multiply with the native variable.
- CarSim provides a browser for activating import variables from the lists of all those that are available.
- New import variables can be defined with VS Commands to pass through data from other software. E.g., variables from Simulink can be passed through to the animator.

Export Variables

- All variables available for writing to output files are also available for export to Simulink or other external code.
- Variables are exported only if activated at runtime, as needed to be compatible with the external model.

Working with the EPIC Unreal Engine

- The VehicleSim Dynamics plugin for Unreal Engine includes the VS Solvers for CarSim and TruckSim, allowing the vehicle models to operate within the Unreal environment.
- The VS Connect library is used to connect an Epic Unreal simulation with a simultaneous Simulink simulation, where both are working with the CarSim VS Solver.

Multibody Model Specifications

State Variables and Degrees of Freedom

CarSim has ordinary differential equations (ODEs) for the dynamics of multibody systems, including rigid bodies, fluids, tires, controllers, and other dynamic parts. Additional state variables are used to define the state of the model for features such as friction, clutch slipping, controllers, etc.

- The number of ODEs and state variables depends on many options available in the model. VS Math Models can generate a list of all state variables for any given simulation setup.
- A basic CarSim model has 81 ODEs and a total of 250 state variables.

Equation Form

- The equations of motion are derived from first principles for 3D motions of multiple connected rigid bodies, using Kane's equations for the multibody dynamics and constraints.
- The equations of motion are ODEs that are not stiff.
- The built-in VS library provides six methods for solving the ODE's (Adams-Bashforth, Adams-Moulton, Runge-Kutta, and Euler methods).
- All methods run at a fixed time step and may be used for real-time HIL applications.
- The algorithms work well with measured and sampled data sources, even when there are discontinuities.
- The VS Solver libraries are compiled with extensive optimizations for efficient use either alone or with other software (e.g., Simulink, LabVIEW).

Initialization and Restarts

- CarSim supports many initialization options, from automatic to detailed specification of any state variable.
- The complete state of the vehicle model is saved at the end of each run, to support continuation of advanced automation and optimization methods.
- The state of the model can be saved during a run and fully restored during the run, in support of advanced optimization methods and repetitive test sequences.