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

CarSim models are contained in dynamically linked library files called VS Solvers. VS Solvers are available for 12 operating systems: Windows (32- and 64-bit), Linux (3 versions), and real-time platforms used for hardware in the loop (8 systems). The models work well with other software (Simulink, LabVIEW, 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 and communicate with each other using external software.

**Vehicle Math Models**

**Vehicle Configurations**
- The basic CarSim 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 includes an option to include extra multibody degrees of freedom (DOF) for engine movement due to mount compliances. 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 formulas
- When linear coefficients or linear interpolation methods are selected, the simulations can run even faster.
- There is no built-in limit to the length of tables.
- Independent and dependent variables can be transformed in support of reusable table datasets.

**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.
- Up to 500 VS Paths may be defined.

**Driver Controls**
- All driver controls can be handled by built-in controllers, defined by equations added with VS Commands, or imported 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.
- The closed-loop Speed Controller (SC) controls throttle and braking based on target speed or target acceleration or path preview.
- The SC path preview mode 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 intervention systems.
- Steering wheel control can be by angle or by torque.
- Open-loop braking can be pedal force or fluid pressure.

**3D Road Geometry and Friction**
- The 3D surface may be VS Terrain or a set of VS Roads. The surface includes 3D geometry, friction, and a coefficient contributing to tire rolling resistance
- 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
  - Automatically convert 3D FBX files from other software
- 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 automatically to adjacent VS Roads
- Road profiles “wander” to follow the vehicle wherever it goes, 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 or solid axle.
- In generic suspensions, the wheel movements depend on jounce on both sides, possibly with coupling as measured for twist-beam suspensions.
- Suspensions can be either steered or un-steered.
- All suspensions have full nonlinear kinematical behavior and can be asymmetric.
- Suspension springs and dampers are nonlinear and 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 bump 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 detailed multibody model that uses an inclined kingpin axis.
- The steering model includes specific details for rack-and-pinion and recirculating ball-type systems.
- Steer angle of each road wheel is available as rotation as measured in a K&C rig (in the sprung mass coordinate system), 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 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)
  - The Pacejka 5.2 version of the magic formula
  - MF-Tyre from Siemens
- CarSim has built-in support for some external models (license required):
  - MF-Swift from Siemens
  - Fıre from COSIN
  - TameTire from Michelin
- Different models can be applied to different wheels of the same vehicle.
- Friction changes are handled with “similarity.”
- Transient effects are included with relaxation length, which may be constant or variable.
- Special equations are used to keep realistic tire transient behavior at very low speeds and very high speeds.
- External tire models can apply forces at either the ground contact point or the wheel center.
- Tire contact can be handled with one to four points.
- Dual tires are available for all wheels.

Powertrain
- CarSim has detailed powertrain models for front-wheel drive, rear-wheel drive, and four-wheel drive (all-wheel drive).
- The powertrain supports internal combustion (IC), electric motor + battery, and hybrid (IC + electric motor + battery + planetary gear)
- IC engine torque is a function of throttle and RPM.
- Fuel consumption is defined with a 2D table.
- The engine feeds torque to the transmission either through 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 center 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 have two clutches with reduction gears in parallel over a differential. The torque distribution may be controlled left and right, or front and rear.
• The center unit has a torque bias for non-locked options.
• Twin-clutch is an alternative to an axle differential.
• Torsional compliance of the driveline is included.
• CarSim supports powertrain from AVL Cruise. An optional license is needed.

**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. The objects can be recycled for extensive runs, to reappear after they go out of view.
- Motion of an object can be constant, set by specifying speed, controlled by acceleration (simple physics), set with algebraic equations, or imported from third-party software.
- Objects that move based on speed or acceleration support offtracking, to produce realistic low-speed traffic turns at intersections.
- Objects used to represent traffic vehicle support brake lights when controlled by speed or acceleration.
- 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 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).
- 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.

**VS Solver Program Input and Outputs**
CarSim uses standard VS library routines for processing input files, performing standard calculations, and generating output files.

**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 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 runtime, 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 Events monitor custom formulas to trigger the reading of a new Parsfile to change values, modes, etc. This is used to script complicated procedures.
- VS Commands can add new functions to simplify other formulas or series of equations.
- VS Commands can define new units.
- The Windows and Linux versions provide embedded Python in support of full programming options.

**Working with Simulink® and External Software**
- On Windows machines, the CarSim math model is a DLL file that runs in many environments:
  - The CarSim GUI runs the model with no additional software.
Models run as blocks in MATLAB/Simulink, LabVIEW, and other simulation environments.

- Models can run as functional mockup units (FMU) using the functional mockup interface (FMI).
- CarSim includes both a 32-bit and 64-bit DLL.

- The CarSim VS Solver is provided on Linux systems as a shared object file (.so).
- 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.

- Libraries, examples, and programming documentation are available in the VS SDK (software development kit) for Windows and Linux. It includes numerous application program interfaces (APIs):
  - The VS Solver API is used to run and interact with the VS Solver 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 helps interact with Configurable Functions.
  - VS Terrain API works with VS Terrain files.
  - VS Connect supports co-simulation between EPIC Unreal and Simulink.

### Input 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 under each tire, etc.
- The vehicle 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, with both 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. The VS Solver is used to generate a list of all state variables for any given simulation setup.
- The basic CarSim model has 89 ODEs and a total of 266 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 Solvers 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.