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

Custom Forces and Motion Sensors

VS Solvers support user-defined forces applied to up to 99 custom reference points that can be added to some of the model rigid bodies, including sprung masses, suspended cab (TruckSim), suspended engine (CarSim), motorcycle steer head (BikeSim), and knuckle (BikeSim). These forces can be imported from other software such as Simulink or adding equations with VS Commands. In addition, VS Math Models include some built-in forces and moments that are nominally zero, but which can be customized by importing values from external software or adding VS Command equations.

VS Solvers also support up to 99 custom motion sensors that provide up to 27 output variables for points of interest defined in various vehicle parts.

> **Note** BikeSim, CarSim, and TruckSim are intended for use by two levels of users. *Model Users* are engineers and designers who will run simulations in previously defined scenarios and conditions. New vehicles and test conditions are specified in the existing models by changing values of existing parameters or replacing tables.
>
> Another level of user is the *Model Builder*. A Model Builder is a more advanced user who will add new features to the model. The custom forces, reference points, and motion sensors are available for advanced users to extend the models.

> You will probably find this document to be more technical than most of the other documentation describing screens in your VehicleSim product. In order to add custom forces, reference points, or motion sensors to the model, you should be familiar with 3D vector concepts, including multiple coordinate systems and the vector dot-product and cross-product concepts.

Library Screens for Custom Forces and Sensors

The **Libraries** menu has a submenu **Custom Forces and Motion Sensors** with two (BikeSim) or three (CarSim and TruckSim) libraries:

1. Use the **Custom Forces and Reference Points** screen to define up to 99 forces (up to 18 per screen) imported from other software or defined with VS Commands, and apply them to a sprung mass, engine (if running CarSim with a mounted engine), cab (if running TruckSim with a suspended cab), steering head (if running BikeSim), or non-spinning wheel carrier (knuckle if running BikeSim).
2. Use the **Positions, Velocities, and Accelerations** screen to add up to 27 output variables each for up to 99 points (up to 18 per screen) attached to bodies in the model. Points can be located in a vehicle sprung mass, solid axle (if the model has solid axles), non-spinning wheel carrier (knuckle), engine (if running CarSim with a mounted engine), cab (if running TruckSim with a suspended cab), or steering head (if running BikeSim).
3. Use the **Preview Points for External Driver Control** screen (CarSim and TruckSim) to define points whose locations are used to generate tracking variables that can be provided to driver models implemented in external code (e.g., VS Commands, Simulink, etc.).

Custom Forces and Reference Points

VehicleSim math models include a command `DEFINE_REFERENCE_POINTS` with the form:

`DEFINE_REFERENCE_POINTS` *n*

where *n* is an integer or formula that is evaluated to provide an integer value. If *n* is less than 1, the command does nothing. If *n* is greater than 0, then the command adds reference points to the VS Math Model, up to a limit of 99 total. The existing number of points is provided with a calculated read-only parameter named `NREF_PT`.

Each point has an associated force vector applied to the body containing the point, reacted from the inertial reference (ground).

The VS Solver initially has zero custom forces and reference points. The easiest way to add custom forces is with the **Custom Forces and Reference Points** screen (Figure 1), which has settings to add up to 18 custom forces to the vehicle math model. Each force is applied to a reference point, defined by specifying the body in which it is located 3 and coordinates 6 in the coordinate system of the specified body 4.

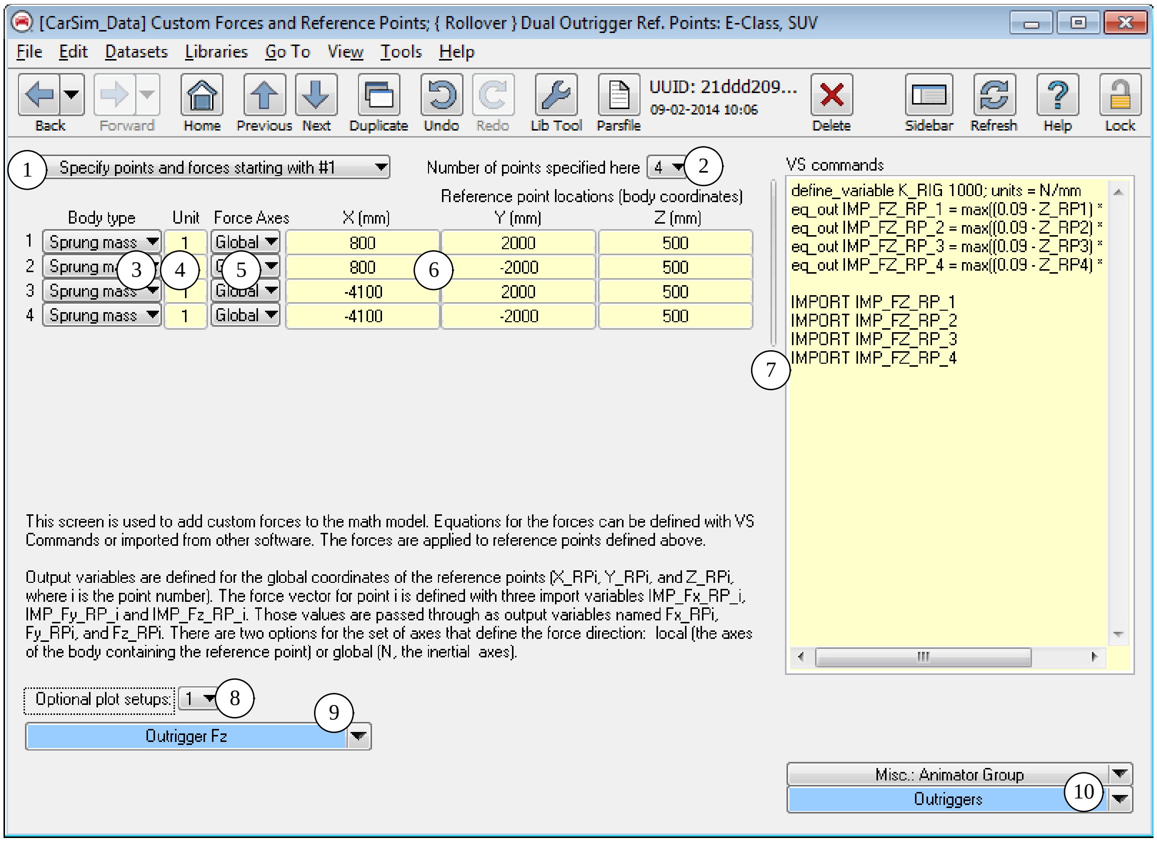

Figure 1. Custom Forces and Reference Points screen.

Note In addition to the forces that are added with this screen, VehicleSim Math Models include a number of built-in forces and moments that are nominally zero and are available for customizing, as described in the next section (page 6).

Overview of Screen

When this screen is used (Figure 1), the command `DEFINE_REFERENCE_POINTS` is automatically used as needed to define new points. The VS Math model supports up to 99 custom forces with associated reference points; multiple datasets from the library can be used if needed, adding up to 18 reference points per dataset.

The custom force added for each point is a vector defined with three imported scalar variables for the X, Y, and Z components of the vector, applied to the selected rigid body in the math model, and reacted by the inertial reference (ground).

The imported variables are named `IMP_FX_RP_`*i*, `IMP_FY_RP_`*i*, and `IMP_FZ_RP_`*i*, where *i* is the ID number of the force and reference point. The forces are nominally zero; they can be set by importing from Simulink or other software, or by adding equations with VS Commands as shown by example in the figure 7. Two conventions are available for orienting the force components: the local body axes, or the global inertial axes 5.

Three output variables are added to the model to provide the position of each point with global coordinates named `X_RP`*i*, `Y_RP`*i*, and `Z_RP`*i*. Another three outputs are added with the values of the associated force component, named `FX_RP`*i*, `FY_RP`*i*, and `FZ_RP`*i*. (The import and output variables are listed in Table 1 on page 6.)

The parameters on this screen are indexed to the reference point number as specified by the current value of the system parameter `IREF_PT`. This parameter is not shown on the screen, but is written in the dataset as needed to identify parameters. As noted earlier, the total number of installed points is calculated internally and shown in the Echo file with the keyword `NREF_PT`.

The VS Math Model sums all of the forces acting on each body, and accounts for the moment arm associated with the location of each reference point relative to the origin of the body containing the point.

> **Note** The coordinate systems for the sprung masses, suspended engines, and suspended cabs are shown on screens where the mass and inertia properties are set. The coordinate system for the steering head (BikeSim) is shown on **Steering System** screen.

User Settings

1 Drop-down list used to specify one of three options for creating new points (Figure 2):

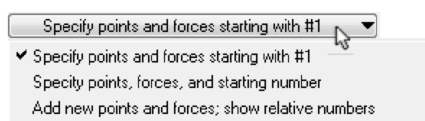

Figure 2. Options for creating new points.

1. The first option is to specify points starting with #1. The dataset is written to automatically add points if needed to obtain the specified number 2. If you are using 18 or fewer points in a simulation, then you should always use this option. If you are using multiple datasets from this library in a simulation, then you may use this option for one of the datasets; the others should use the second option to specify points with a different range of numbers.

2. The second option is provided in case the simulation will include links to multiple datasets defining reference points. If this option is selected, then a data field is available (Figure 3) to specify the number of the first point in this dataset. For example, if another dataset defined points one through four, then specify that the first number for this dataset is five.

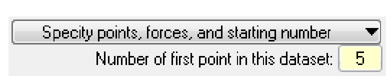

Figure 3. Setting the number of the first reference point.

3. The third option specifies that all of the points specified on the screen are new and to be added to the model, regardless of how many points have already been installed. In this case, the point numbers depend on how many datasets have already been linked for the simulation. Point numbers on the screen are therefore relative to this dataset only. This option is intended for quick tests by advanced users. If you use this option, you should check the Echo file to confirm that the numbers of the points are what you expected.

Advanced users might override parameter values for a reference point, or change the equations used to calculate the force vector. More detail on this topic is provided in a later subsection on page 6.

2 Drop-down control to specify the number of custom forces and associated reference points described in the dataset. The command `DEFINE_REFERENCE_POINTS` is written to define new reference points as needed, given the current number already installed, along with the option selected with the first drop-down list 1.

3 Drop-down control to specify the type of body (keyword = `OPT_RP_BODY_TYPE`). For CarSim, the options are sprung mass and suspended engine. For TruckSim, they are sprung mass and suspended cab. For BikeSim, they are sprung mass, steer head, and knuckle.

4 If the type of body is “sprung mass” (CarSim or TruckSim) or “knuckle” (BikeSim) then this field is shown to specify the part that contains the reference point (keyword = `OPT_RP_BODY_ID`).

5 Drop-down control to specify whether the force components are oriented with the local (body) axes or the global (inertial) axes (keyword = `OPT_RP_FORCE_N`).

6 X-Y-Z coordinates of the reference point within the coordinate system of the selected body (keywords = `X_RP`, `Y_RP`, `Z_RP`).

7 This field allows advanced users to provide VS Commands to calculate the forces associated with the reference points. For example, Figure 4 shows commands for setting the Z-direction forces to simulate an outrigger contacting the ground. A splitter control adjacent to the field can be used to widen the field to better view long commands.

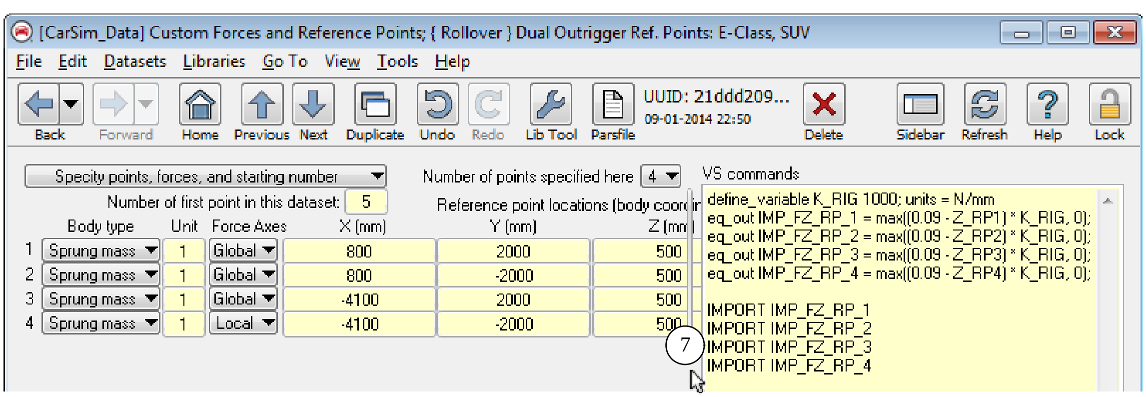

Figure 4. Using the splitter control to widen the VS Command field.

8. Number of optional plots setups. This screen can associate plots with the custom forces or point locations; use this control to specify the number of plot setup links (up to eight) to be used.
9. Links to **Plot: Setup** datasets that define sets of variables to be plotted.
10. Miscellaneous link for advanced users to include more data.

Names of Variables

The output variables that are added are the global X, Y, and Z coordinates of the reference points and the X, Y, and Z components of the force vector (Table 1).

Table 1. Import and output variables regarding the custom forces and reference points.

Imports	**Outputs**	**Description**
	`X_RP`*i* `Y_RP`*i* `Z_RP`*i*	X, Y, Z global coordinates of reference point *i* (1 to 99)
`IMP_FX_RP_`*i* `IMP_FY_RP_`*i* `IMP_FZ_RP_`*i*	`Fx_RP`*i* `Fy_RP`*i* `Fz_RP`*i*	X, Y, Z components of force vector *i* (1 to 99)

Changing Parameters or Forces for Existing Reference Points

The basic use of this library screen is to add a set of reference points to the model, define the locations of the points, define the axes used to define the custom force components, and possibly provide VS Commands for calculating the forces. As noted earlier, the reference points are installed with a VS Command `DEFINE_REFERENCE_POINTS`.

There are situations where advanced users might want to change the parameter values and/or force equations after the reference points have been installed. One way to do this is by putting parameter assignments or equations into miscellaneous yellow fields (e.g., use the **Generic VS Commands** library). Another is to use this library, with a dataset that overrides settings that might already exist. However, if a dataset from this library is used to override settings, care must be taken to ensure that new points are not added, and that the numbering is consistent.

When choosing the numbering option for the reference points via the drop-down control 1 (Figure 1, page 3), both of the first two options are valid for either the first encounter (when the reference points are added to the model) or for later links where the points already exist. The third option will always add new points and therefore should never be used when the intent is to change properties of existing points.

Built-In Customizable Forces and Moments

The BikeSim, CarSim, and TruckSim math models include six moments applied to each sprung mass, suspended engine, suspended cab, steering head, and non-spinning wheel carriers reacted from the inertial reference (Table 2). As indicated in the table, three of the moments are in the directions of the body axes, and three are in the directions of the global axes.

Table 2. Customizable moments acting on sprung masses, cab, and engine.

Imports	**Outputs**	**Body**	**Directions**
IMP_MX_EXT IMP_MY_EXT IMP_MZ_EXT	MX_EXT MY_EXT MZ_EXT	Sprung mass of lead unit	Body axes
IMP_MX_EXT*i* IMP_MY_EXT*i* IMP_MZ_EXT*i*	MX_EXT*i* MY_EXT*i* MZ_EXT*i*	Sprung mass of unit *i* (trailer, *i* > 1)	
IMP_MX_ENGINE IMP_MY_ENGINE IMP_MZ_ENGINE	MX_ENGINE MY_ENGINE MZ_ENGINE	Engine (CarSim only)	
IMP_MX_CAB IMP_MY_CAB IMP_MZ_CAB	MX_CAB MY_CAB MZ_CAB	Cab (TruckSim only)	
IMP_MX_EXT_STR IMP_MY_EXT_STR IMP_MZ_EXT_STR	MXEXTSTR MYEXTSTR MZEXTSTR	Steer head (BikeSim only)	Steering system axes
IMP_MX_EXT_W*i* IMP_MY_EXT_W*i* IMP_MZ_EXT_W*i*	MXEXTW*i* MYEXTW*i* MZEXTW*i*	Non-spinning wheel carriers (BikeSim only, front: *i* = 1; rear: *i* = 2)	Wheel carrier axes
IMP_MXGEXT IMP_MYGEXT IMP_MZGEXT	MXGEXT MYGEXT MZGEXT	Sprung mass of lead unit	Global axes
IMP_MXGEXT*i* IMP_MYGEXT*i* IMP_MZGEXT*i*	MXGEXT*i* MYGEXT*i* MZGEXT*i*	Sprung mass of unit *i* (trailer, *i* > 1)	
IMP_MXG_ENGINE IMP_MYG_ENGINE IMP_MZG_ENGINE	MXG_ENGINE MYG_ENGINE MZG_ENGINE	Engine (CarSim only)	
IMP_MXG_CAB IMP_MYG_CAB IMP_MZG_CAB	MXG_CAB MYG_CAB MZG_CAB	Cab (TruckSim only)	

The CarSim and TruckSim math models also include built-in forces (Table 3) and moments (Table 4) acting on the unsprung masses at the centers of the non-spinning wheel carriers. They are reacted either by the sprung mass or ground, and are oriented with the axes of the reaction body.

All of the forces and moments listed in Table 2, Table 3, and Table 4 are nominally zero. They have no effect unless nonzero values are imported from external software such as Simulink, or calculated with equations defined with VS Commands.

Table 3. Customizable forces acting on unsprung masses (CarSim and TruckSim).

Imports	Outputs	Body	Reaction	Directions
`IMP_FxExA`*a* `IMP_FyExA`*a* `IMP_FzExA`*a*	`FxExA`*a* `FyExA`*a* `FzExA`*a*	solid axle *a*	Sprung mass	Sprung mass axes
`IMP_FxEx`*sa* `IMP_FyEx`*sa* `IMP_FzEx`*sa*	`FxEx`*sa* `FyEx`*sa* `FzEx`*sa*	unsprung mass side *s* (L or R), axle *a*		
`IMP_FxGExA`*a* `IMP_FyGExA`*a* `IMP_FzGExA`*a*	`FxGExA`*a* `FyGExA`*a* `FzGExA`*a*	solid axle *a*	Inertial reference (ground)	Global axes
`IMP_FxGEx`*sa* `IMP_FyGEx`*sa* `IMP_FzGEx`*sa*	`FxGEx`*sa* `FyGEx`*sa* `FzGEx`*sa*	unsprung mass side *s* (L or R), axle *a*		

Table 4. Customizable moments acting on unsprung masses (CarSim and TruckSim).

Imports	Outputs	Body	Reaction	Directions
`IMP_MxExA`*a* `IMP_MyExA`*a* `IMP_MzExA`*a*	`MxExA`*a* `MyExA`*a* `MzExA`*a*	solid axle *a*	Sprung mass	Sprung mass axes
`IMP_MxEx`*sa* `IMP_MyEx`*sa* `IMP_MzEx`*sa*	`MxEx`*sa* `MyEx`*sa* `MzEx`*sa*	unsprung mass side *s* (L or R), axle *a*		
`IMP_MxGExA`*a* `IMP_MyGExA`*a* `IMP_MzGExA`*a*	`MxGExA`*a* `MyGExA`*a* `MzGExA`*a*	solid axle *a*	Inertial reference (ground)	Global axes
`IMP_MxGEx`*sa* `IMP_MyGEx`*sa* `IMP_MzGEx`*sa*	`MxGEx`*sa* `MyGEx`*sa* `MzGEx`*sa*	unsprung mass side *s* (L or R), axle *a*		

Positions, Velocities, and Accelerations

All VS vehicle models support the command `DEFINE_MOTION_SENSORS` with the form:

`DEFINE_MOTION_SENSORS` *n*

where *n* is an integer or formula that is evaluated to provide an integer value. If *n* is less than 1 the command does nothing. If *n* is greater than 0, then the command adds motion sensors to the VS Math Model, up to a limit of 99 total. The existing number of sensors is provided with a calculated (read-only) parameter named `NSENSOR_M`.

A single motion sensor can produce up to 27 output variables, with the exact number dependent on the value of a parameter `OPT_S_OUTPUT` that specifies which of five options are used. The command `MAKE_MOTION_SENSOR_OUTPUTS` causes the VS Solver to define the output variables for all motion sensors that are currently installed. This command is used to create the output variables so they can be used in VS Command equations, or be activated for export and writing to output files.

The VS Solver initially has zero motion sensors. The easiest way to add position, velocity, or acceleration outputs is with the **Positions, Velocities, and Accelerations** screen (Figure 5). This screen generates commands to the VS Solver to add output variables for points of interest.

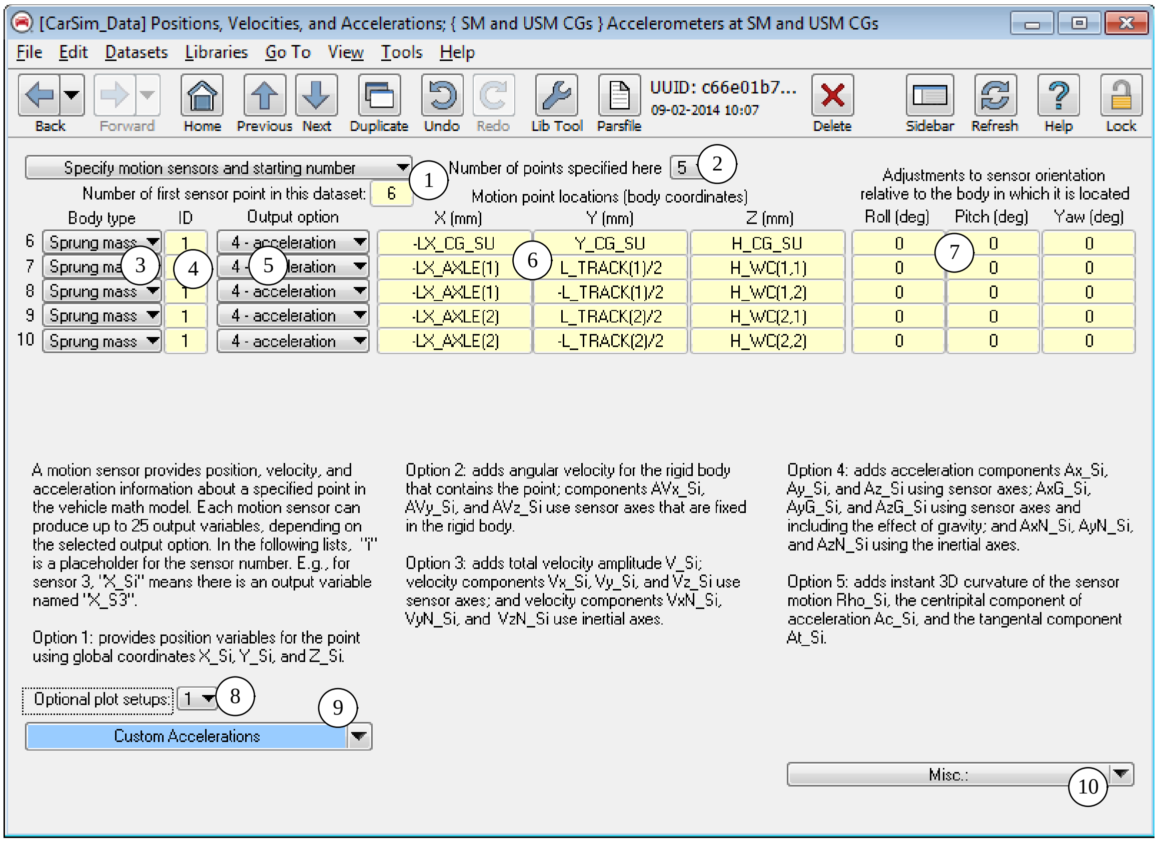

Figure 5. The Positions, Velocities, and Accelerations screen.

Overview of Screen

When this screen is used, the command `DEFINE_MOTION_SENSORS` is automatically written to define new sensors as needed. The VS Math model supports up to 99 motion sensors; multiple datasets from the library can be used if needed, adding up to 18 sensors per dataset.

The VS Solvers are set up to add motion outputs for points located in a sprung mass, solid axle (CarSim and TruckSim), knuckle (non-spinning wheel carrier), suspended engine body (CarSim), suspended cab (TruckSim), or steering head (BikeSim). Up to 27 output variables are added for each motion sensor. This screen is used to identify the type of body (3), the ID number of the body (4), the types of output associated with the point (5), the location of the point in the selected body (6), and an adjustment to the axis system of the body to mimic the orientation of a physical sensor (7).

The parameters on this screen are indexed to the sensor number as specified by the current value of the system parameter `ISENSOR_M`. This parameter is not shown on the screen, but is written in the dataset as needed to identify parameters. As noted earlier, the total number of installed sensors is calculated internally and shown in the Echo file with the keyword `NSENSOR_M`.

User Settings

(1) Drop-down list used to specify one of three options for creating new sensors (Figure 6):

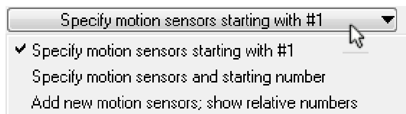

Figure 6. Options for creating new sensors.

1. The first option is to specify sensors starting with #1. The dataset is written to automatically add sensors if needed to obtain the specified number (2). If you are using 18 or fewer motion sensors in a simulation, then you should always use this option. If you are using multiple datasets from this library in a simulation, then you may use this option for one of the datasets; the others should use the second option to specify points with a different range of numbers.

2. The second option is provided in case the simulation run will include links to multiple datasets defining motion sensors. If this option is selected, then a data field is available (Figure 7) to specify the number of the first sensor in this dataset. For example, if you know another dataset defines sensors numbered one through five, then specify that the first number for this dataset is six.

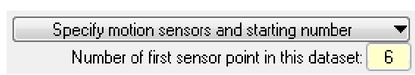

Figure 7. Setting the number of the first reference point.

3. The third option allows advanced users to specify that all of the sensors specified on the screen are new and to be added to the model, regardless of how many have already been installed. In this case, the sensor numbers depend on how many datasets have already been linked for the simulation. Numbers on the screen are therefore relative to this dataset only. This option is intended for quick tests by advanced users. If you use this option, you should check the Echo file to confirm that the numbers of the sensors are what you expected.

(2) Drop-down list to specify the number of motion sensor points. The command `DEFINE_MOTION_SENSORS` is written to define new sensors as needed, given the current number already installed, along with the option selected with the first drop-down list (1).

(3) Drop-down list to specify the type of body (keyword = `OPT_S_BODY_TYPE`). The options include sprung masses and knuckles (the non-spinning wheel carrier) for all products. CarSim also includes the suspended engine body and solid axles. TruckSim includes the suspended cab and solid axles. BikeSim includes the steer head.

4. If the type of body is “solid axle,” “knuckle,” or “sprung mass” (CarSim and TruckSim), then this field is shown to select which one contains the sensor (keyword = `OPT_S_BODY_ID`).

   For sprung masses in CarSim or TruckSim, the ID is the unit number (lead unit → 1, first trailer → 2, etc.).

   For solid axles or knuckles in BikeSim, the numbering is by suspension, starting at 1 and continuing for each suspension, including those in trailers (CarSim and TruckSim).

   For knuckles in CarSim and TruckSim, the numbering goes left to right, front to rear: L1 → 1, R1 → 2, L2 → 3, R2 → 4, L3 → 5, etc. for all wheels, including those on trailers

5. Drop-down list to specify a set of output variables based on the point (keyword = `OPT_S_OUTPUT`). The options and associated outputs are listed in Table 5.

6. X-Y-Z coordinates of the sensor location within the coordinate system of the selected body (keywords = `X_S`, `Y_S`, `Z_S`).

7. Orientation angles of the sensor relative to the X, Y, and Z axes of the body containing the sensor (keywords = `A_ROLL_S`, `A_PITCH_S`, `A_YAW_S`). Starting with the sensor axes parallel with those of the vehicle part, the adjustment is first made by the specified roll angle about the vehicle part X axis, then the pitch angle about the new Y axis of the sensor after the roll adjustment, then the yaw angle about the new sensor Z axis.

   > **Note** The Echo file shows the three rotation angle parameters, and also three calculated angles that would be observed for viewing sensor angles from the back (`ROLL_AXES_S`), side (`PITCH_AXES_S`) and top (`YAW_AXES`). These might be helpful when trying to simulate outputs from a physical sensor whose axes are not aligned with the axes of vehicle part in which the sensor is mounted.

8. Number of optional plots setups. This screen can associate plots with the sensors; use this control to specify the number of plot setup links (up to eight) to be used.

9. Links to **Plot: Setup** datasets that define sets of variables to be plotted.

10. Miscellaneous link for advanced users to include more data.

Names of Output Variables

Table 5 lists the 27 output variables that can be generated for motion sensor *i*.

Note that the number of output variables ranges from 3 (`OPT_S_OUTPUT` = 1) to 27 (`OPT_S_OUTPUT` = 5). Because the number of outputs depends on this parameter value, the outputs are not created when the command `DEFINE_MOTION_SENSORS` is used. Instead, a separate command is used to create outputs for all currently existing motion sensors: `MAKE_MOTION_SENSOR_OUTPUTS`. This command is automatically written at the end of each Parsfile written for this library, such that the outputs will all exist for use in VS Commands or other applications as soon as the dataset from this library is read by the VS Solver.

Table 5. Summary of output variables for sensor i.

Variable	Axes	Description	Options
X_S*i* Y_S*i* Z_S*i*	Global	Global XYZ coordinates	1 - 5
AVx_S*i* AVy_S*i* AVz_S*i*	Sensor	Angular velocity components	2 - 5
Vx_S*i* Vy_S*i* Vz_S*i*	Sensor	Translational velocity components	3 - 5
VxN_S*i* VyN_S*i* VzN_S*i*	Global		
V_S*i*		Translational velocity magnitude	
Ax_S*i* Ay_S*i* Az_S*i*	Sensor	Translational acceleration components	4 - 5
AxN_S*i* AyN_S*i* AzN_S*i*	Global		
AxG_S*i* AyG_S*i* AzG_S*i*	Sensor	Acceleration components with gravity effect	
Jx_S*i* Jy_S*i*	Sensor	Measures of longitudinal and lateral jerk	4-5
Ac_S*i*	Instant 3D motion	Centripetal component of acceleration	5
At_S*i*		Tangential component of acceleration	
Rho_Si		Instant 3D curvature (inverse of radius)	

Changing Parameters for Existing Motion Sensors

The basic use of this library screen is to add a set of motion sensors to the model, define the locations and orientations of the sensors, and specify which output variables should be created. As noted earlier, the sensors are installed with a VS Command `DEFINE_MOTION_SENSORS`, and the output variables are defined for all existing sensors with the command `MAKE_MOTION_SENSOR_OUTPUTS`.

There are situations where advanced users might want to change parameter values after the sensors have been installed. One way to do this is by putting parameter assignments or equations into miscellaneous yellow fields (e.g., use the **Generic VS Commands** library). Another is to use this library, with a dataset that overrides settings that might already exist. However, if a dataset from this library is used to override settings, care must be taken to ensure that new points are not added, and that the numbering is consistent.

When choosing the numbering option for the sensors via the drop-down control 1 (Figure 5), both of the first two options are valid for either the first encounter (when the sensors are added to the model) or for later links where the sensors already exist. The third option will always add new sensors and therefore should never be used when the intent is to change properties of existing sensors.

Be aware that the first encounter of the dataset should specify the output option. The output variables are defined only once for a given sensor via the command `MAKE_MOTION_SENSOR_OUTPUTS`.

Mathematical Definitions

The output variables listed in Table 5 for a single sensor are all based on a point fixed in a specified vehicle part (e.g. a sprung mass), with three orthogonal axes that are also fixed in the vehicle part. Position, velocity, and acceleration for the point are all motion vectors; the output variables listed in Table 5 are defined by taking vector dot products of the motion vectors with directions of interest.

Position Variables

The global position vector to this point may be written as the sum of two vectors:

$$\mathbf{p}_S = \mathbf{p}_O + \mathbf{p}_{S\_rel} \tag{1}$$

where $\mathbf{p}_S$ is the vector connecting the origin of the global coordinate system to the sensor point, $\mathbf{p}_O$ is the vector connecting the origin of the global coordinate system to the origin of the vehicle part (e.g., the sprung mass origin), and $\mathbf{p}_{S\_rel}$ is a relative vector connecting the origin of the vehicle part to the sensor point. The relative vector $\mathbf{p}_{S\_rel}$ is defined using the values specified on the screen 6 (Figure 5) that use the axis system of the vehicle component (e.g., sprung mass):

$$\mathbf{p}_{S\_rel} = \texttt{X\_S}\ \mathbf{c}_X + \texttt{Y\_S}\ \mathbf{c}_Y + \texttt{Z\_S}\ \mathbf{c}_Z \tag{2}$$

where $\mathbf{c}_X$, $\mathbf{c}_Y$, and $\mathbf{c}_Z$ are the unit-vector directions of the axes for the vehicle component.

The global coordinates for the point are:

$$\texttt{X\_S}i = \mathbf{p}_S \bullet \mathbf{n}_X \tag{3}$$

$$\texttt{Y\_S}i = \mathbf{p}_S \bullet \mathbf{n}_Y \tag{4}$$

$$\texttt{Z\_S}i = \mathbf{p}_S \bullet \mathbf{n}_Z \tag{5}$$

where $\mathbf{n}_X$, $\mathbf{n}_Y$, and $\mathbf{n}_Z$ are the unit-vector directions of the fixed global axes.

Angular Velocity Variables

The angular velocity of the vehicle part is defined within the math model with a vector $\boldsymbol{\omega}$. The sensor point, being fixed in the vehicle part, shares the same angular velocity vector. The components of angular velocity for the sensor are defined using the sensor axis directions:

$$\texttt{AVx\_S}i = \boldsymbol{\omega} \cdot \mathbf{s}_X \tag{6}$$

$$\texttt{AVy\_S}i = \boldsymbol{\omega} \cdot \mathbf{s}_Y \tag{7}$$

$$\texttt{AVz\_S}i = \boldsymbol{\omega} \cdot \mathbf{s}_Z \tag{8}$$

where $\mathbf{s}_X$, $\mathbf{s}_Y$, and $\mathbf{s}_Z$ are the unit-vector directions of the sensor axes.

Velocity Variables

The velocity of the sensor point is the derivative of the position vector:

$$\mathbf{v}_S = d(\mathbf{p}_S)/dt$$

$$\mathbf{v}_S = d(\mathbf{p}_O)/dt + d(\mathbf{p}_{S\_rel})/dt$$

$$\mathbf{v}_S = \mathbf{v}_O + \boldsymbol{\omega} \times \mathbf{p}_{S\_rel} \tag{9}$$

where $\mathbf{v}_S$ is the vector of the sensor point velocity and $\mathbf{v}_O$ is the vector of the velocity of the origin of the coordinate system for the vehicle part. Six output variables are calculated for this velocity vector using axes of interest:

$$\texttt{Vx\_S}i = \mathbf{v}_S \cdot \mathbf{s}_X \tag{10}$$

$$\texttt{Vy\_S}i = \mathbf{v}_S \cdot \mathbf{s}_Y \tag{11}$$

$$\texttt{Vz\_S}i = \mathbf{v}_S \cdot \mathbf{s}_Z \tag{12}$$

$$\texttt{VxN\_S}i = \mathbf{v}_S \cdot \mathbf{n}_X \tag{13}$$

$$\texttt{VyN\_S}i = \mathbf{v}_S \cdot \mathbf{n}_Y \tag{14}$$

$$\texttt{VzN\_S}i = \mathbf{v}_S \cdot \mathbf{n}_Z \tag{15}$$

The magnitude of the velocity $\texttt{V\_S}i$ is also calculated:

$$\texttt{V\_S}i = |\, \mathbf{v}_S \,| = [\mathbf{v}_S \cdot \mathbf{v}_S]^{1/2} \tag{16}$$

Acceleration Variables

The acceleration of the sensor point is the derivative of the velocity:

$$\mathbf{a}_S = d(\mathbf{v}_S)/dt$$

$$\mathbf{a}_S = d(\mathbf{v}_O)/dt + d(\boldsymbol{\omega} \times \mathbf{p}_{S\_rel})/dt$$

$$\mathbf{a}_S = \mathbf{a}_O + \boldsymbol{\alpha} \times \mathbf{p}_{S\_rel} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{p}_{S\_rel}) \tag{17}$$

where $\mathbf{a}_S$ is the vector of the sensor point acceleration, $\mathbf{a}_O$ is the vector of the acceleration of the origin of the coordinate system for the vehicle part, and $\boldsymbol{\alpha}$ is the angular acceleration of the vehicle part. Six output variables are calculated for this acceleration vector using axes of interest:

$$\texttt{Ax\_S}i = \mathbf{a}_S \cdot \mathbf{s}_X \tag{18}$$

$$\texttt{Ay\_S}i = \mathbf{a}_S \cdot \mathbf{s}_Y \tag{19}$$

$$\texttt{Az\_S}i = \mathbf{a}_S \bullet \mathbf{s}_Z \tag{20}$$

$$\texttt{AxN\_S}i = \mathbf{a}_S \bullet \mathbf{n}_X \tag{21}$$

$$\texttt{AyN\_S}i = \mathbf{a}_S \bullet \mathbf{n}_Y \tag{22}$$

$$\texttt{AzN\_S}i = \mathbf{a}_S \bullet \mathbf{n}_Z \tag{23}$$

Three output variables include the combination of acceleration and gravity, where gravity is defined with the vector –g $\mathbf{n}_Z$:

$$\texttt{AxG\_S}i = (\mathbf{a}_S - g\,\mathbf{n}_Z) \bullet \mathbf{s}_X \tag{24}$$

$$\texttt{AyG\_S}i = (\mathbf{a}_S - g\,\mathbf{n}_Z) \bullet \mathbf{s}_Y \tag{25}$$

$$\texttt{AzG\_S}i = (\mathbf{a}_S - g\,\mathbf{n}_Z) \bullet \mathbf{s}_Z \tag{26}$$

Jerk Variables

A measure of jerk in the sensor's longitudinal direction, `Jx_S`*i*, is given by the time-derivative of `Ax_S`*i*. This value is calculated numerically, because the VehicleSim math model is based on a Newtonian formulation of physics. Consequently, the derivatives of accelerations are, in general, not natively available for output.

Similarly, a measure of jerk in the sensor's lateral direction, `Jy_S`*i*, is given by the time-derivative of `Ay_S`*i*.

Instant 3D Curvature Variables

The instant direction vector (tangential direction) for the point velocity is:

$$\mathbf{e}_t = \mathbf{v}_S \,/\, |\, \mathbf{v}_S \,| \tag{27}$$

The tangential component of acceleration is:

$$\texttt{At\_S}i = \mathbf{e}_t \bullet \mathbf{a}_S \tag{28}$$

The normal acceleration is:

$$\texttt{Ac\_S}i = |\, \mathbf{a}_S - \mathbf{e}_t\, \texttt{At\_S}i \,| \tag{29}$$

and the instant curvature is:

$$\texttt{Rho\_Si} = \texttt{Ac\_S}i \,/\, \texttt{V\_S}i^2 \tag{30}$$

Note The rotation sequence from the vehicle's sprung mass coordinate system to the coordinate system of the motion sensor is roll-pitch-yaw.

Preview Points for External Driver Control

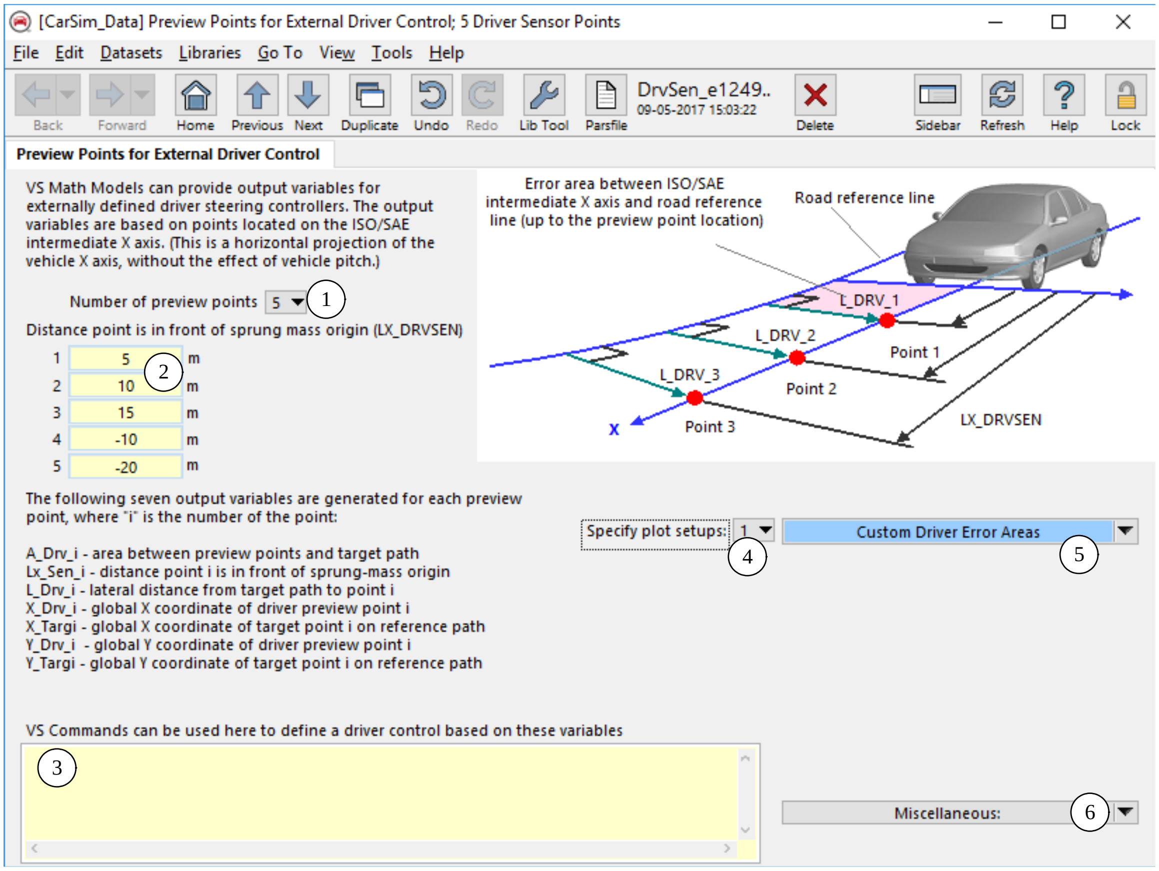

Figure 8 shows the **Preview Points for External Driver Control** screen used to define outputs that can be helpful for supporting user-defined driver models in CarSim or TruckSim (defined in Simulink, with VS Commands, etc.). This library does not exist in BikeSim.

The outputs are based on preview points located in front of the vehicle on an X (longitudinal) axis parallel to X axis of the ISO/SAE vehicle intermediate axis system, relative to the origin of the sprung mass coordinate system.

> **Note** ISO and SAE define major vehicle motions using an axis system whose X and Y axis are horizontal (that is, the Z axis is parallel with the Z axis of the inertial reference), and rotated relative to the inertial X and Y axes using the yaw angle.

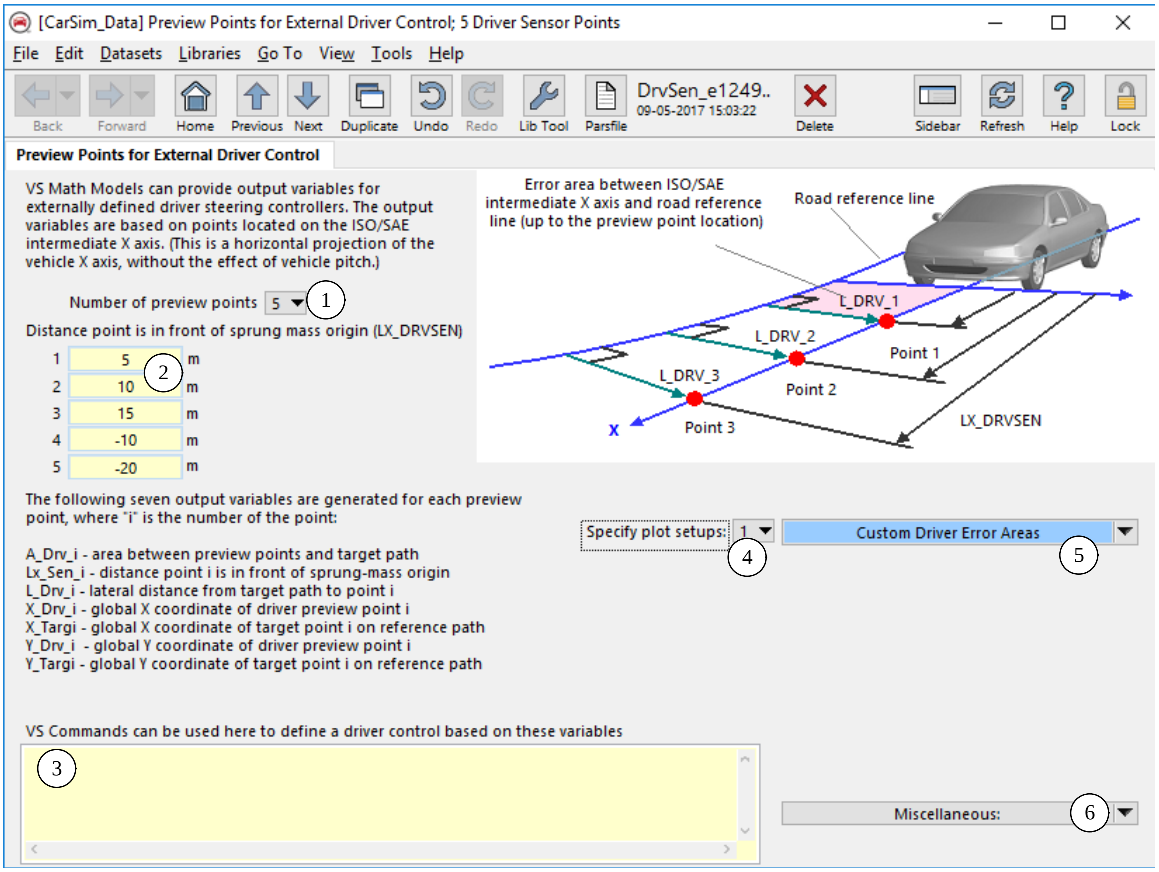

Figure 8. Preview Points for External Driver Control screen.

There can be up to nine driver preview points.

The VS math models calculate the global X and Y coordinates of each preview point, the lateral distance from the (horizontal) road reference line to the point, and the horizontal area between the ISO/SAE intermediate X axis and the road reference line (from the origin of the vehicle sprung-mass coordinate system up to the preview point). All calculations take place in the intermediate X-Y plane. That is, Z coordinates are not considered.

Driver preview point locations (longitudinal distance from the sprung-mass origin on the vehicle intermediate coordinate system) are set on this screen as parameters. However, the locations can also be varied during the simulation using import variables named `IMP_LX_SEN_`*i* (where *i* = the point number).

User Settings

(1) Drop-down list to specify the number of preview points. The associated keyword is `INSTALL_DRIVER_PREVIEW_SENSORS`, which is a command to the VS Solver to install the specified number of preview points and create the associated parameters, import variables, and output variables. The CarSim and TruckSim math models support up to nine preview points.

(2) Distance each preview point lies in front of the vehicle coordinate system origin (keyword = `LX_DRVSEN`). As shown in

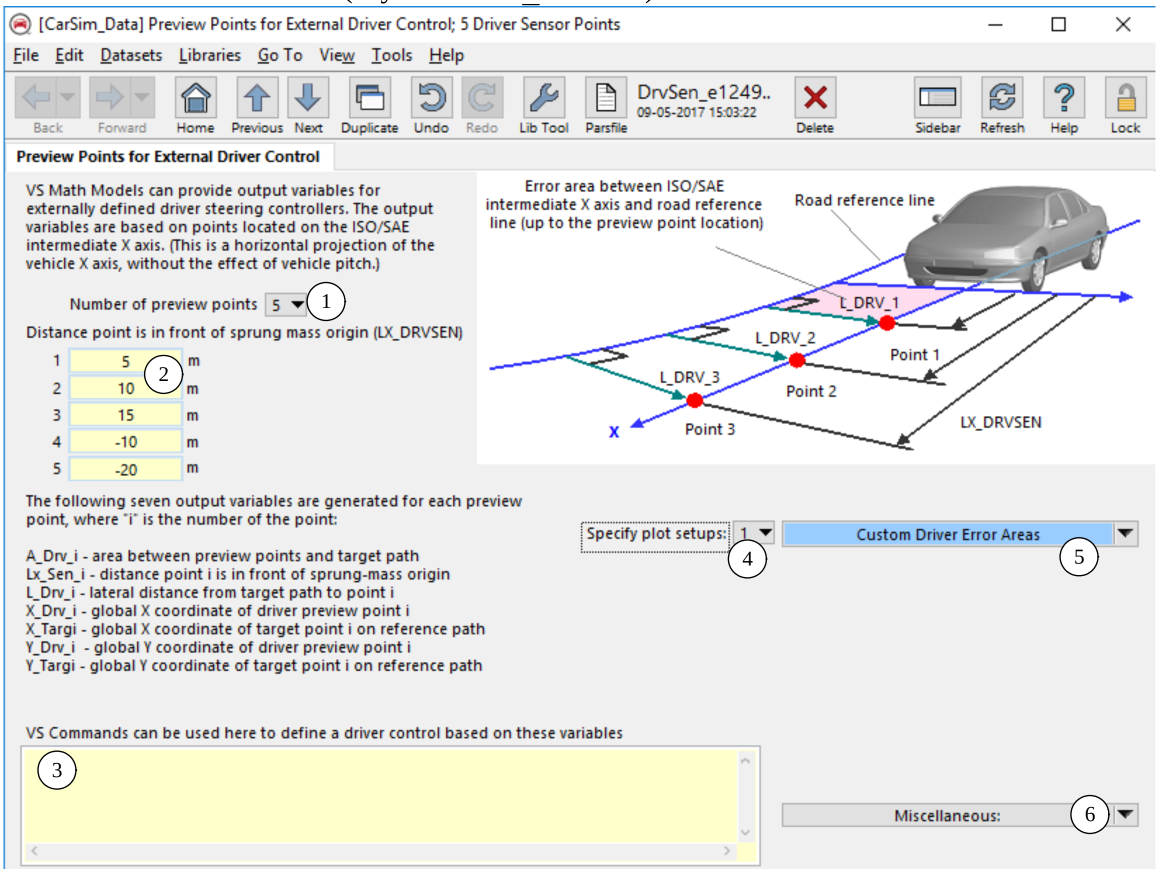

Figure 8, the points are located on the vehicle intermediate system X axis.

(3) This field is available for advanced users to provide VS Commands.

(4) Number of optional plots setups. This screen can associate plots with the preview points (output variables are listed in Table 6); use this control to specify the number of plot setup links (up to eight) to be used.

(5) Links to **Plot: Setup** datasets that define sets of variables to be plotted.

(6) Miscellaneous link for advanced users to include more data.

Names of Output Variables

Table 6 lists the output variables associated with preview point *i*.

Table 6: Summary of output variables for driver preview point i.

Variable	**Description**
`A_Drv_`*i*	Area between the target reference path and vehicle X axis up to the preview point
`Lx_Sen_` *i*	Longitudinal distance that the preview point is in front of the sprung mass origin
`L_Drv_` *i*	Lateral distance from the target reference path to the preview point
`X_Drv_` *i*	X global coordinate of the preview point
`X_Targ` *i*	X global coordinate of a point on the target reference path, matched to the preview point
`Y_Drv_` *i*	Y global coordinate of the preview point
`Y_Targ` *i*	Y global coordinate of a point on the target reference path, matched to the preview point