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

Animator: Reference Frame

VS Visualizer shows moving rigid objects, stationary objects, and internally animated objects such as walking pedestrians or windmills.

When the relationships between a set of points never changes (i.e., they form a rigid structure), they are said to exist in the same *reference frame*. The coordinate systems provided for shapes are fixed in the appropriate reference frame. Animator reference frames are created automatically in BikeSim, CarSim, and TruckSim for the vehicle, road, moving objects, ADAS sensor detection vectors, brake lights, and wheel indicators such as force arrows. In SuspensionSim, they are created automatically for points and vectors.

The **Animator: Reference Frame** screen is used to define reference frames for applications other than those that are handled automatically. The custom frames are used for applications such as:

- Locate the virtual camera and define a "look point" within the reference frame where the camera is aimed.
- Locate animator shapes in a scene that are strictly cosmetic: buildings, signs, trees, fences, etc.
- Setup sets of objects such that only one is visible at a time. For example: traffic lights that may be green, yellow, or red; or walking or waiting pedestrians.
- Setup objects that contain local motion, such as a pedestrian walking, with moving limbs.

Information for VS Visualizer and VS Solvers

The **Animator: Reference Frame** screen (Figure 1) is one of just a few in the VS Browser that provides information mainly to VS Visualizer (VSV). The dataset Parsfile contains commands for VSV, along with values that may be numbers, names of output variables from a VS Solver, or possibly algebraic formulas involving names of output variables and math functions that are recognized by VSV. For example, Figure 2 shows the Parsfile for the data shown in Figure 1.

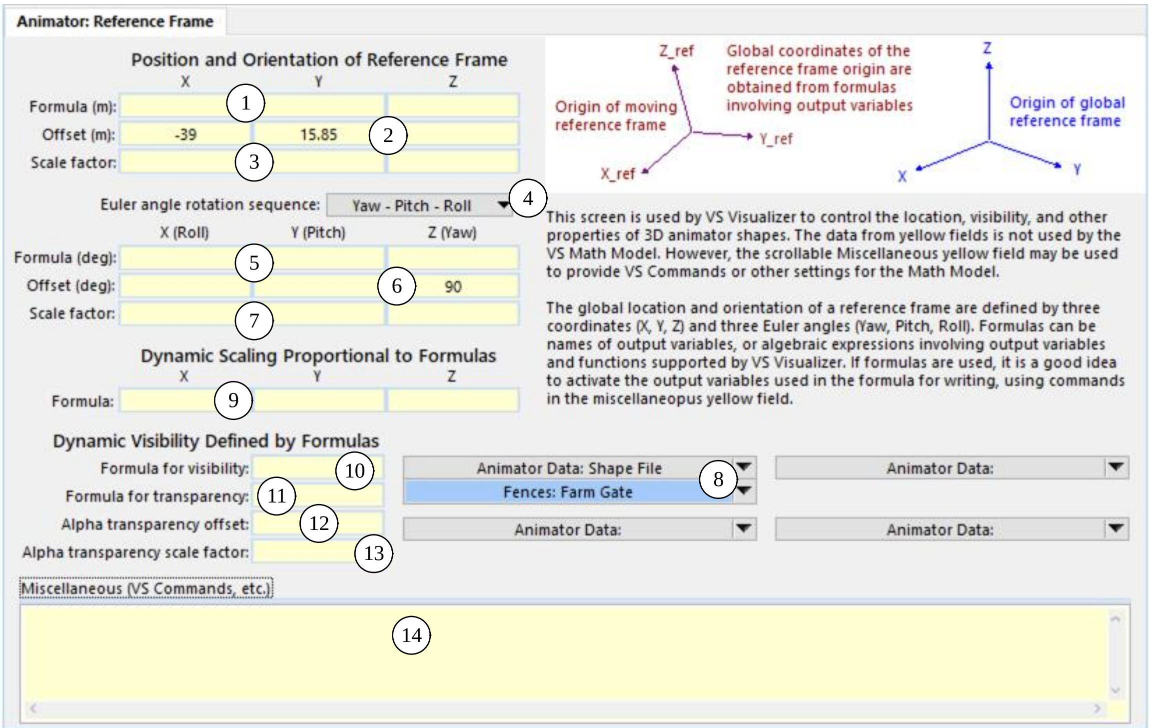

Figure 1. An Animator Reference Frame dataset used to locate a 3D animator shape.

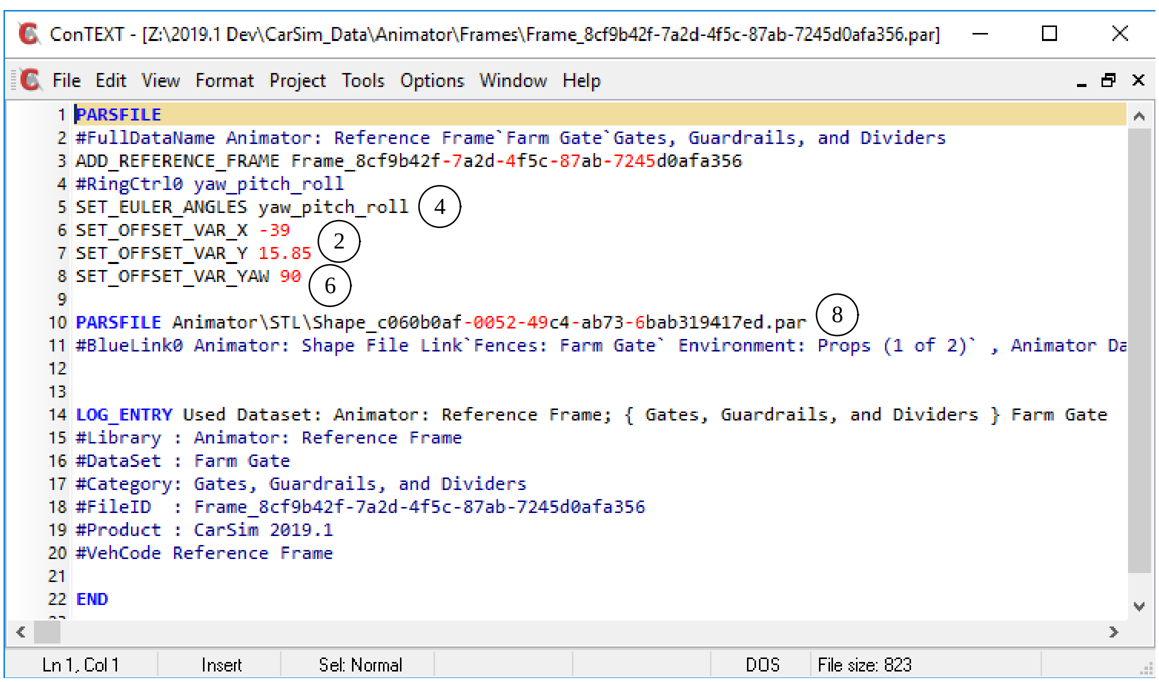

Figure 2. Parsfile for dataset shown in Figure 1.

After the title of the dataset (line 2), the Parsfile contains the VSV command ADD_REFERENCE_FRAME. This command instructs VSV to add a new reference frame with a specified name (in this case, Frame_8cf9b42f-7a2d-4f5c-87ab-7245d0afa356). Until a new reference frame is created, any commands related to a reference frame will apply to this one.

This example is used to locate an animator shape (a farm gate). To do so, the reference frame is located using global X and Y coordinates, specified with constants (2). These are shown in the Parsfile with the commands SET_OFFSET_VAR_X and SET_OFFSET_VAR_Y.

The drop-down control (4) (Figure 1) specifies a sequence of Euler angle rotations, and writes the selection in the form of the VSV command SET_EULER_ANGLES in the Parsfile. The orientation of this reference frame is set with a yaw angle of 90° (6), which is written in the Parsfile with the command SET_OFFSET_VAR_YAW.

The 3D shape file used to show the farm gate is specified in the blue link (8). This is a link to a dataset in the **Animator Shape File Link** (Figure 3). The dataset shows an image of the object (1), and contains a link to an OBJ file with the 3D shape (2).

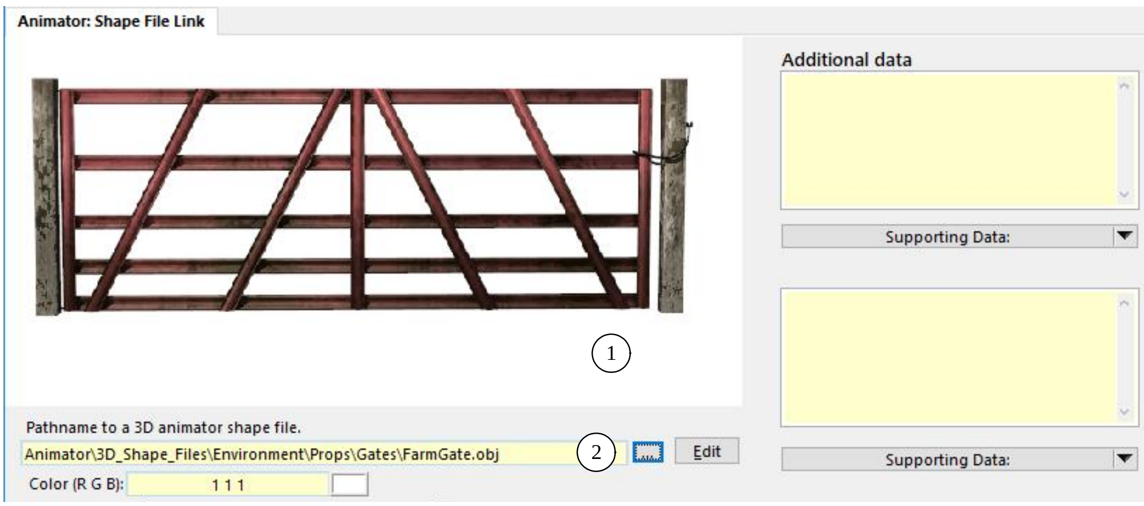

Figure 3. Animator Shape File Link dataset.

Figure 4 shows the Parsfile for the **Animator Shape File Link** dataset. It includes a command for VSV: add_obj with the pathname for an OBJ file (2). It also has a few more VSV commands specifying how the object should be rendered (lines 7, 10, 12, and 16).

This shape from the specified OBJ file is located within the current reference frame, which is the one specified in Figure 1.

Many reference frame datasets do not include links to shape files. When a reference frame is added to VSV, it remains as the "active" frame for all objects that are added, until a new reference frame is added. In many cases, a reference frame is added, and then links to shape files are made from other screens.

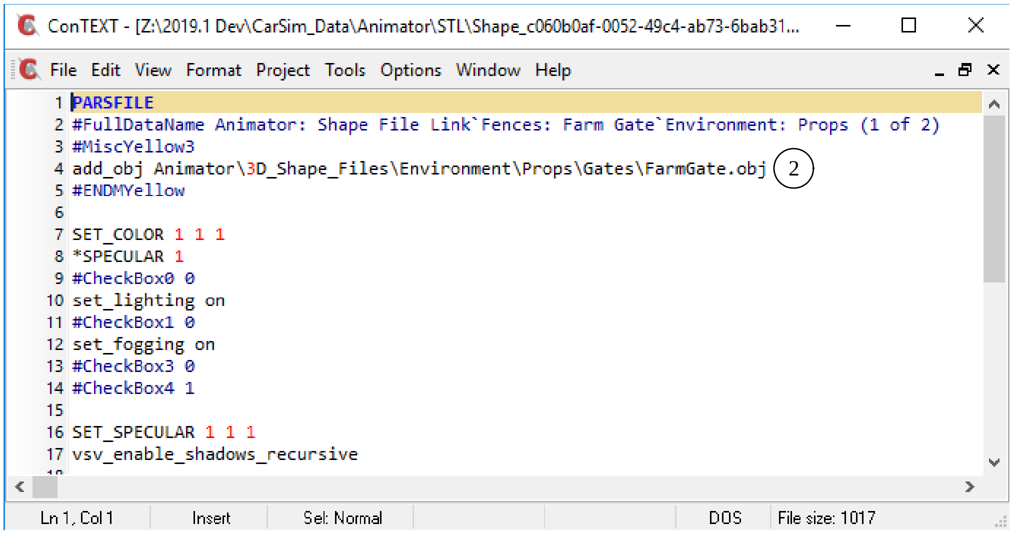

Figure 4. Parsfile for a shape file link.

To locate a camera or render shapes, VSV converts relative coordinates in a moving coordinate system to absolute coordinates in the global coordinate system. This conversion is defined by the global position of the origin of the moving coordinate system of the reference frame (its global X, Y, and Z coordinates), along with the orientations of its three axes. For example, Figure 5 shows a reference frame for a virtual camera that is intended to track a moving vehicle. In this case, there are no constant offsets; instead, global coordinates are specified using the names of output variables from the VS Solver: names `Xo`, `Yo`, and `Zo` (1). A yaw angle is also specified by the name `yaw` (5).

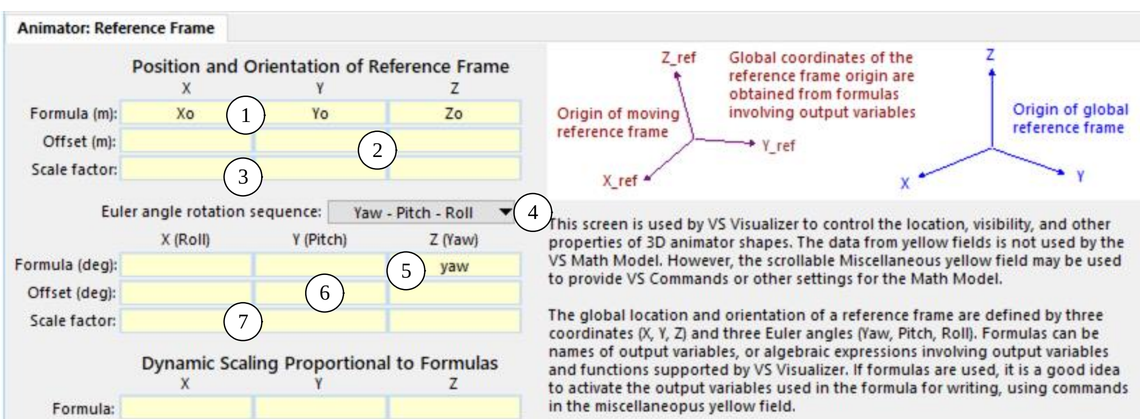

Figure 5. Reference frame for camera.

VSV cannot make use of any output variables in post-processing visualization unless they are provided in the simulation output file (ERD or VS) written by a VS Solver. To ensure that they are activated for writing, the Parsfile has the activation keywords made by adding the prefix `WRT_` to the name of the variables: `WRT_Xo`, `WRT_Yo`, etc. These commands are written to the Parsfile (lines 11-16, Figure 6). In case the simulation will be run with live animation/video, the VS Solver is also provided activation keywords made by adding the prefix `ANI_` to the name of the variables: `ANI_Xo`, etc. The same commands are used for the Yaw angle (lines 17 and 18).

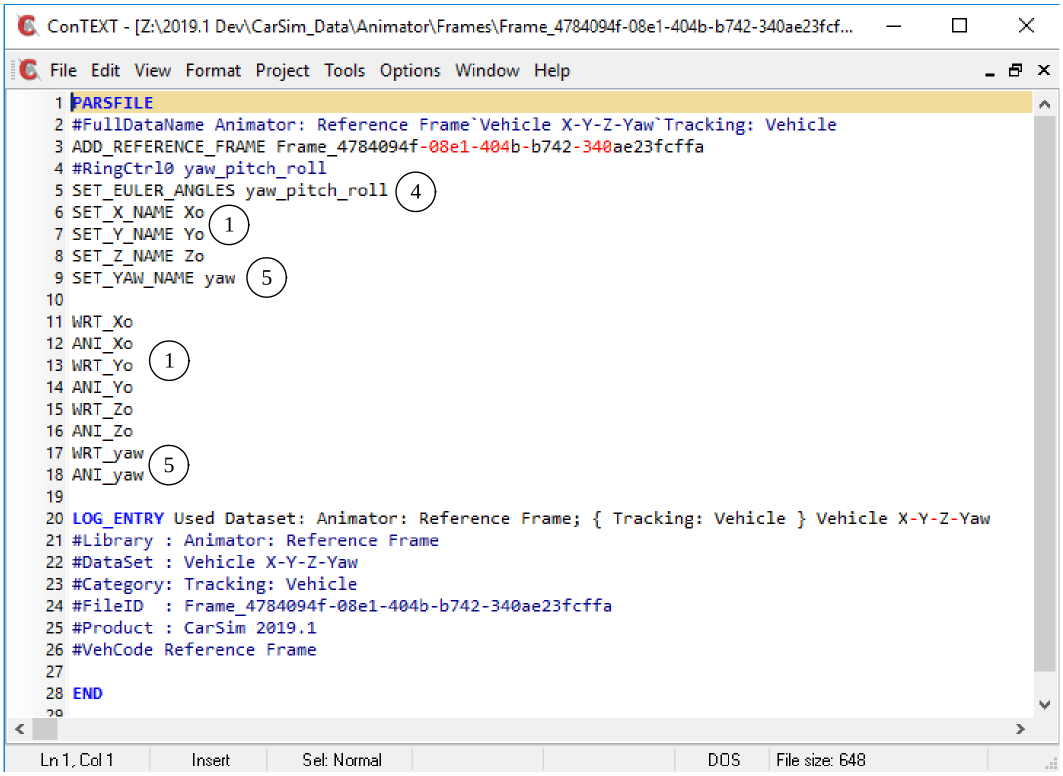

Figure 6. Camera reference frame Parsfile.

The commands to activate output variables for writing to file and streaming for live animation are the only parts of the Parsfile that are used by the VS Solver. All other commands are provided for VSV. In this example, the output names are identified for VSV with the commands `SET_X_NAME`, `SET_Y_NAME`, `SET_Z_NAME`, and `SET_YAW_NAME`.

Motion Variables

To locate a camera or render shapes, VSV converts relative coordinates in a moving coordinate system to absolute coordinates in the global coordinate system. This conversion is defined by the global position of the origin of the moving coordinate system (its global X, Y, and Z coordinates), along with the orientations of its three axes

A reference frame requires up to six variables from the output files generated by a VS Solver for post-processing visualization. For live video during the simulation, it receives the variables as a “live stream” from the VS Solver while a simulation runs. The required variables are the global X-Y-Z coordinates for the origin of the local coordinate system, and three angles to orient the axes of the coordinate system of the reference frame.

Position Variables

The three global coordinates of the origin of the reference frame are calculated as:

$$\textit{coordinate} = C_0 + f \bullet SF_C \tag{1}$$

where *f* is a formula based on output variables from a VS Solver, C_0 is a constant offset, and SF_C is a constant scale factor. These are specified using the yellow fields identified in the figure. In most cases, the formula *f* is simply the name of an output variable (e.g., in Figure 5, the formulas for X, Y, and Z are simply the names `Xo`, `Yo`, and `Zo` 1).

1. Formulas or names of output variables that define X, Y, and Z (keywords = `SET_X_NAME`, `SET_Y_NAME`, `SET_Z_NAME`). These are short names of output variables from the VS Solver, or formulas involving the short names. If the field is left blank, then a constant value of 0.0 is used for the coordinate. A value of 0.0 is also used if the name is not found in the output file generated by the VS Solver, or in the connection made for live video.

   VSV supports the use of algebraic expressions made of output variable names, math operators, Boolean operators, and basic math functions. These are documented in the *VS Visualizer Reference Manual*, and also in the *VS Commands Reference Manual*.

   Alert VSV does not recognize any symbols other than time T and output variables contained in an ERD or VS file, or provided via live streaming during a simulation.

2. Offsets for coordinate variables (keywords = `SET_OFFSET_VAR_X`, `SET_OFFSET_VAR_Y`, `SET_OFFSET_VAR_Z`). These are numbers used to replace the symbol C_0 in Equation 1. If no number is entered, a value of 0.0 is used.

3. Scale factors for coordinate variables (keywords = `SET_SCALE_VAR_X`, `SET_SCALE_VAR_Y`, `SET_SCALE_VAR_Z`). These are numbers used to replace the symbol SF_C in Equation 1. If no number is entered, a value of 1.0 is used.

   Note The offsets and scale factors are legacy features, used in old versions that did not support formulas. In the early versions, the “formula” field was a “variable” field that had to be the name of an output variable.

Rotation Variables

The orientation of a reference frame is defined by three consecutive rotations that are called *Euler angles*. Although it might not be obvious at first if you are not experienced with 3D kinematics, any conceivable orientation of a reference frame can be described with three Euler angles. The values of the angles depend on the sequence of the rotations: different angles are required for sequential rotations about the X-Y-Z axes than about the Z-Y-X axes.

For vehicles, the angles are commonly called *yaw* (rotation about the Z axis), *pitch* (rotation about the “new” Y axis), and *roll* (rotation about the “new-new” X axis).

The standard terminology for vehicle dynamics defines the sequence of rotation as yaw-pitch-roll. Starting such that the X, Y, and Z axes of the moving frame are parallel with those of the global frame, first rotate the frame about its Z axis by a yaw angle. After this is done, the new Z direction is the same as the old, but the X and Y axes are pointed in new directions, called X' and Y'. Next, rotate about the new Y axis, Y', by a pitch angle. After the pitch rotation, the Y axis is still in the

Y' direction, but the X and Z axes are pointed in new directions, called X" and Z". Finally, rotate a third time, about the most recent X axis, X", by the roll angle.

> **Note** In principle, there are 12 possible sequences of body-fixed orientation angles. However, only two sequences are used in VSV to visualize vehicle motions: yaw-pitch-roll, used for large body motions, as described above, and yaw-roll-pitch, used for the spinning wheels.

(4) This control provides the two options (Figure 7) that are supported by VSV.

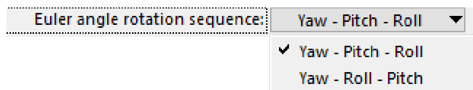

Figure 7. Euler angle sequence.

The VSV keyword is `SET_EULER_ANGLES`, and the two acceptable settings are the text `yaw_pitch_roll` or `yaw_roll_pitch`.

Each Euler angle is calculated with a relationship of the form:

$$angle = A_0 + f \cdot SF_A \qquad (2)$$

where f is a formula based on output variables from a VS Solver for an Euler angle variable from a VS Solver obtained from the output file, A_0 is a constant offset, and SF_A is a constant scale factor. These are specified using the yellow fields identified in the figure.

(5) Formulas or names of output variables for roll, pitch, and yaw (keywords = `SET_ROLL_NAME`, `SET_PITCH_NAME`, `SET_YAW_NAME`). These are short names of output variables from the VS Solver, or formulas involving the short names. If no name is listed, then a constant value of 0.0 is used. A value of 0.0 is also used if the name is not found.

Formulas for angles are handled the same as formulas for coordinates, including support for symbolic expressions and the need to activate output variables for writing to file and streaming for live animation for all variables appearing in formulas.

(6) Offsets for Euler angle variables (keywords = `SET_OFFSET_VAR_ROLL`, `SET_OFFSET_VAR_PITCH`, `SET_OFFSET_VAR_YAW`). These are numbers used to replace the symbol A_0 in Equation 2. If no number is entered, a value of 0.0 is used.

(7) Scale factors for Euler angle variables (keywords = `SET_SCALE_VAR_ROLL`, `SET_SCALE_VAR_PITCH`, `SET_SCALE_VAR_YAW`). These are numbers used to replace the symbol SF_A in Equation 2. If no number is entered, a value of 1.0 is used.

Links to Shape Files

(8) When a reference frame is created to control the position or rendering of a few shapes, it can be convenient to use the miscellaneous links to specify the shape, as shown earlier for the farm gate.

As noted earlier, any links to animator shapes that are loaded by VSV will be placed in the most recently added reference frame. (The links are not necessarily made from the reference frame screen.)

Dynamic Effects

All objects associated with a reference frame can be re-sized statically by ratios of actual lengths relative to reference lengths, using different ratios for the local X, Y, and Z directions. The static scale factors for each direction (X, Y, Z) are determined by two parameters built into VS Visualizer for each reference frame. For example, all X coordinates are scaled by a ratio of two VS Visualizer parameters: `X_LENGTH / X_REF_LENGTH`.

These parameters are typically set in datasets from the libraries **Animator: Shape File Link**, **Animator: Shape Assembly**, or **Animator: Group**.

VS Visualizer also supports dynamically resizable shapes, used for creating force vectors showing tire actions, bar graphs, and other shapes resized based on dynamic variables. For example, Figure 8 shows a dataset for an arrow shape (8) that is resized in proportion to aerodynamic vertical force, with a multiplier of 1/400: `Fz_Air/400` (9).

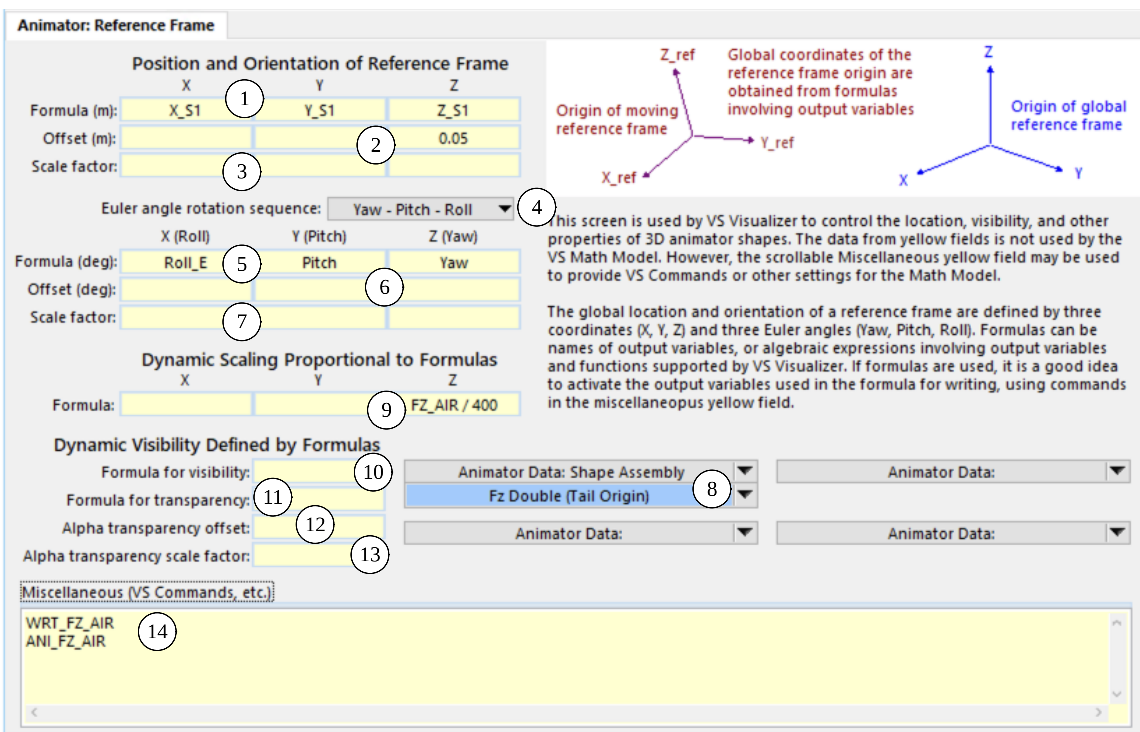

Figure 8. Reference frame with a dynamically scaled arrow.

As noted earlier, it is sometimes necessary to specify the names of output variables that must be activated for writing to file or streaming for animation. Because the Z scaling is specified with a formula, it is necessary to activate the variable `Fz_Air` with commands written in the miscellaneous field (14).

9 Formulas or names of output variables that are used to size the objects in the reference frame dynamically (keywords = SET_FRAME_SCALE_X_NAME, SET_FRAME_SCALE_Y_NAME, and SET_FRAME_SCALE_Z_NAME). If no variable is named, then the scaling in that direction is not changed dynamically.

The complete scaling of the coordinates of the shapes contained in the reference frame is determined in each direction by the relationship shown here for the X direction:

$$X_{new} = X_{old} \bullet f \bullet \mathtt{X\_LENGTH} / \mathtt{X\_REF\_LENGTH} \quad (3)$$

where X_{old} is the original X coordinate for a point in the shape, X_{new} is the rescaled value, f is the formula or output variable identified with the keyword SET_FRAME_SCALE_X_NAME, and X_LENGTH, and X_REF_LENGTH are parameters.

Dynamic Visibility and Transparency

All objects associated with a reference frame can have their opacity changed dynamically. VS Visualizer supports two types of dynamic opacity: visibility and transparency (more properly called translucency).

Visibility

In the definition used by VS Visualizer, visibility controls whether the objects in a reference frame are either fully visible or completely invisible. This change occurs instantly, without any transition.

For example, this is done automatically for the datasets in the **Animator: Vehicles and Sensor Targets** for animating brake lights, where the brake lights are always in one of two mutually exclusive states. It is also used to automatically show 3D vectors connecting ADAS sensors to target points. The same capability may be applied by advanced users for custom setups.

For example, Figure 9 shows a dataset for a pedestrian standing still. The pedestrian is represented as a 3D shape applied to a moving object that has associated output variables. One of these, the speed V_Obj_*o* (where *o* is the object number), is used to set the visibility of the shape.

Note	The names of output variables in this example make use of the Symbol Stack feature in the VS Browser, which enables the same dataset to be applied in for multiple moving objects. The text <<o>> will be replaced in the All.Par file generated by the VS Browser and sent to VS Solvers and VSV. For example, if this dataset is associated with the third moving object, the replacement text is "3" and the speed variable is written V_Obj_3. The motion variables become X_Obj_3, Y_Obj_3, etc.

Another dataset (Figure 10) shows the same moving object motion variables being used in a reference frame that is linked to a different shape file, in this case for a pedestrian walking. In the case of the walking pedestrian, the formula used for visibility is V_Obj_<<o>> 10: the velocity of the moving object. If the object has any speed other than zero, the walking pedestrian shape is shown, moving at the speed of the object.

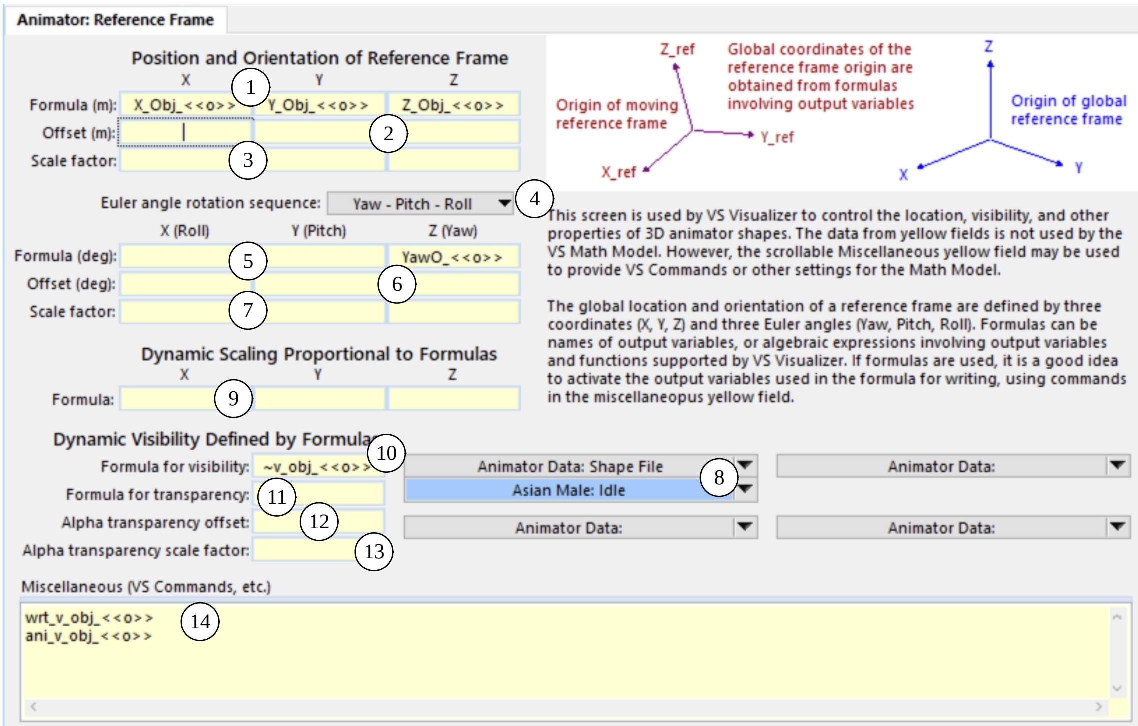

Figure 9. Pedestrian idle (not walking).

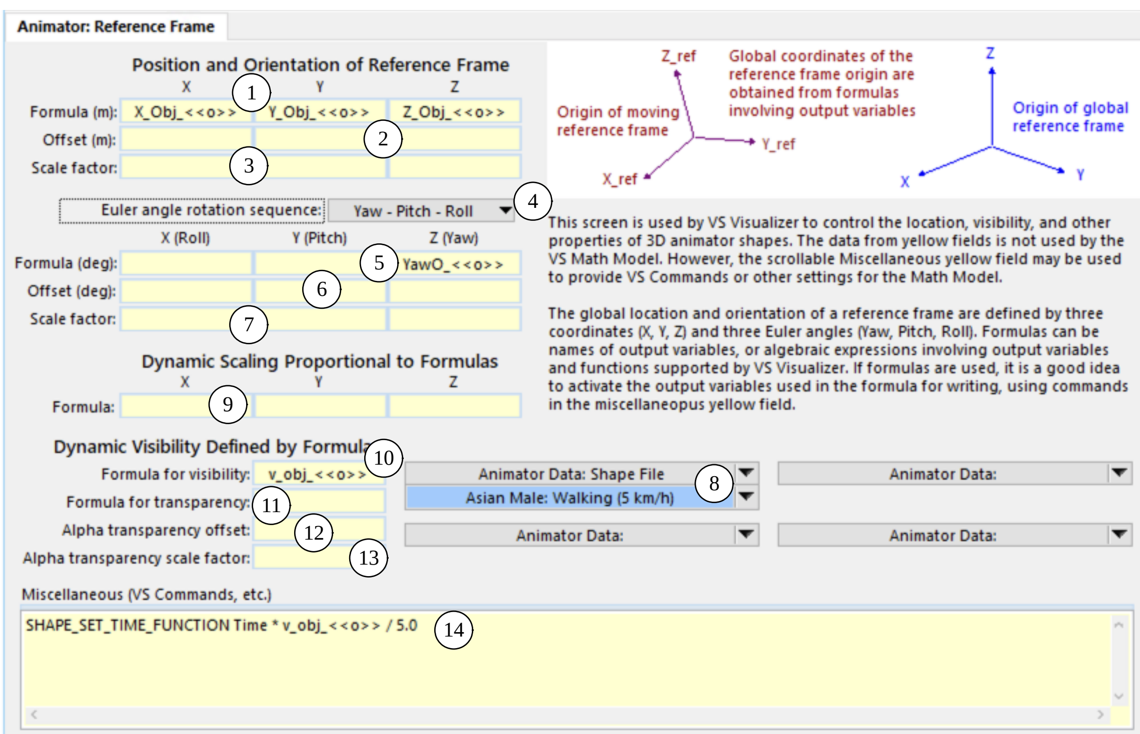

Figure 10. Pedestrian walking.

On the other hand, the dataset for the idle pedestrian (Figure 9) uses a formula for visibility: `~V_Obj_<<o>>` (10). The tilde ~ in front of the speed variable is the Boolean "not" operator. That is, if the speed is zero, the formula has a value of 1, and the idle pedestrian shape is visible. If the speed is anything except zero, the formula has a value of 0, and the idle pedestrian shape is hidden.

(10) Formula or name of an output variable from a VS Solver that is used to hide or show the objects in the reference frame dynamically (keyword = `SET_FRAME_VISIBLE_NAME`). If the function is evaluated and gives has a value of zero, the objects in the frame are invisible; otherwise, they are visible. If no variable is named, then visibility is not changed dynamically.

Transparency

VS Visualizer defines transparency to produce a gradient of opacities. This may be used for animating tire skid marks on the road surface, where more slip would constitute a darker mark. VS Visualizer interprets the transparency of an object on a scale from 0.0 (invisible) to 1.0 (fully visible) given by the following equation:

$$\textit{transparency} = \mathrm{T}_0 + f \bullet \mathrm{SF}_\mathrm{T} \tag{4}$$

where f is a formula based on output variables from a VS Solver for the transparency, T_0 is a constant offset, and SF_T is a scale factor (gain). Transparency values of 0.0 or less are interpreted as completely invisible, and values of 1.0 and greater are interpreted as fully opaque.

(11) Formulas or names of output variables that are used to set the opacity of the objects in the reference dynamically (keyword = `REFERENCE_FRAME_ALPHA_NAME`). This is the variable f in Equation 4. If no name is listed, then a constant value of 1.0 is used. A value of 1.0 is also used if the name is not found in the output file or live connection.

(12) Offset for the transparency of the objects in the reference frame (keyword = `REFERENCE_FRAME_ALPHA_OFFSET`). This value is used to replace the symbol T_0 in Equation 4. If no number is entered, a value of 0.0 is used.

(13) Scale factor for the transparency of the objects in the reference frame (keyword = `REFERENCE_FRAME_ALPHA_SCALE`). This value is used to replace the symbol SF_T in Equation 4. If no number is entered, a value of 1.0 is used.

Miscellaneous

(14) Field for additional information related to the reference frame.

For example, in Figure 8, a `WRT_` command was given to ensure the VS Solver would activate the output variable `Fz_Air` for writing to file (in support of VSV being used for post-processing visualization), and an `ANI_` command was given to ensure the variable would be streamed in support of live animation.

In the example of the walking pedestrian (Figure 10), the command `SHAPE_SET_TIME_FUNCTION` was provided to VSV to scale the local animation time (showing the moving arms and legs of the walking pedestrian) for the reference frame in proportion to the speed. The artist who made the walking pedestrian animation asset timed the leg movement for a forward speed of 5km/h. The local time used in the reference frame is the global time T scaled by the ration `V_Obj_<<o>> / 5`.

For more information about the command `SHAPE_SET_TIME_FUNCTION`, please see the *VS Visualizer Reference Manual*.