Altair

HyperWorks

Altair MotionView 2019 Tutorials

MV-1000: Interactive Model Building and Simulation

altairhyperworks.com

MV-1000: Interactive Model Building and Simulation

Multi Body Dynamics (MBD) Overview

MBD Definition

Multi Body Dynamics (MBD) is defined as the "study of dynamics of a system of interconnected bodies". A mechanism (MBD system) constitutes a set of links (bodies) connected (constrained) with each other to perform a specified action under application of force or motion. The motion of mechanisms is defined by its kinematic behavior. The dynamic behavior of a mechanism results from the equilibrium of applied forces and the rate of the change of momentum.

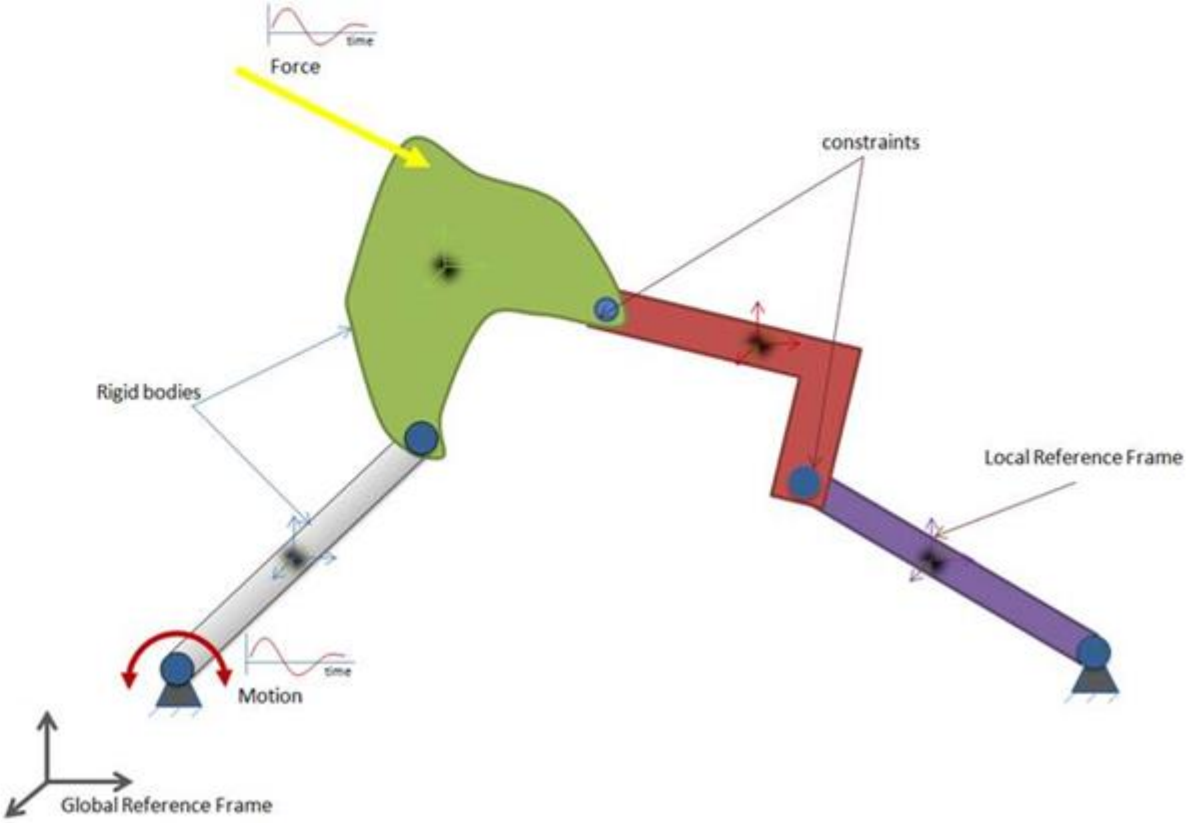
MBD Modeling

A classical MBD formulation uses a rigid body modeling approach to model a mechanism. A rigid body is defined as a body in which deformation is negligible.

In general, in order to solve an MBD problem, the solver requires following information:

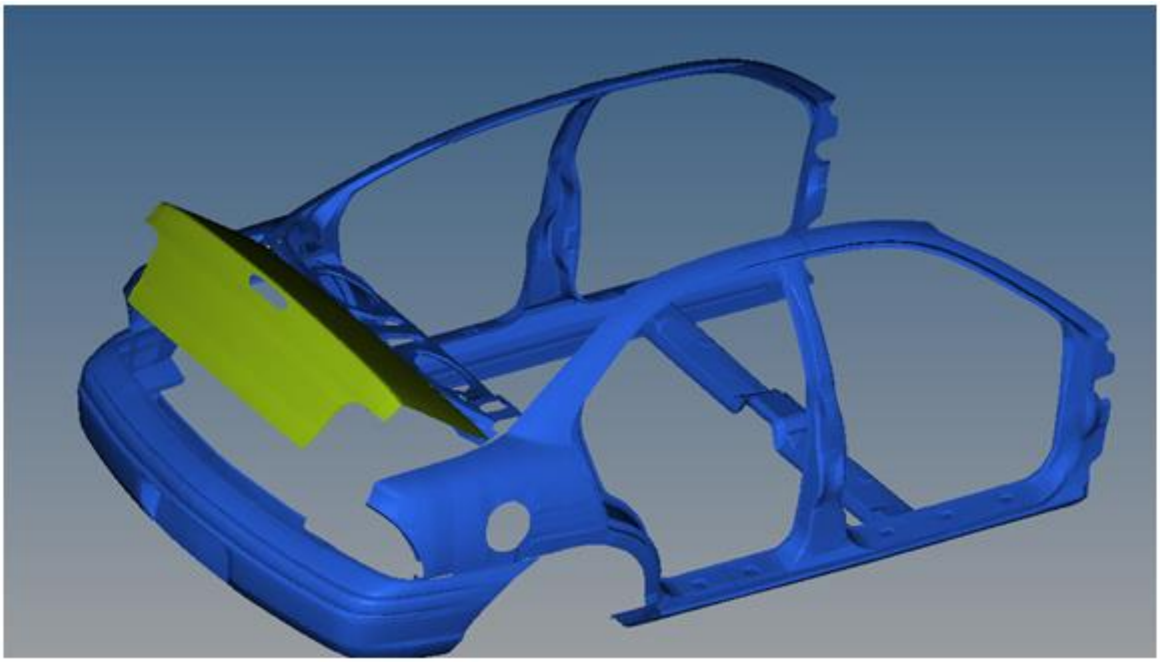
- Rigid body inertia and location
- Connections – type, bodies involved, location, and orientation
- Forces and motions – bodies involved, location, orientation, and value

MotionView facilitates quick and easy ways of modeling items, such as a system, through graphical visualization.



In this tutorial, you will learn how to:

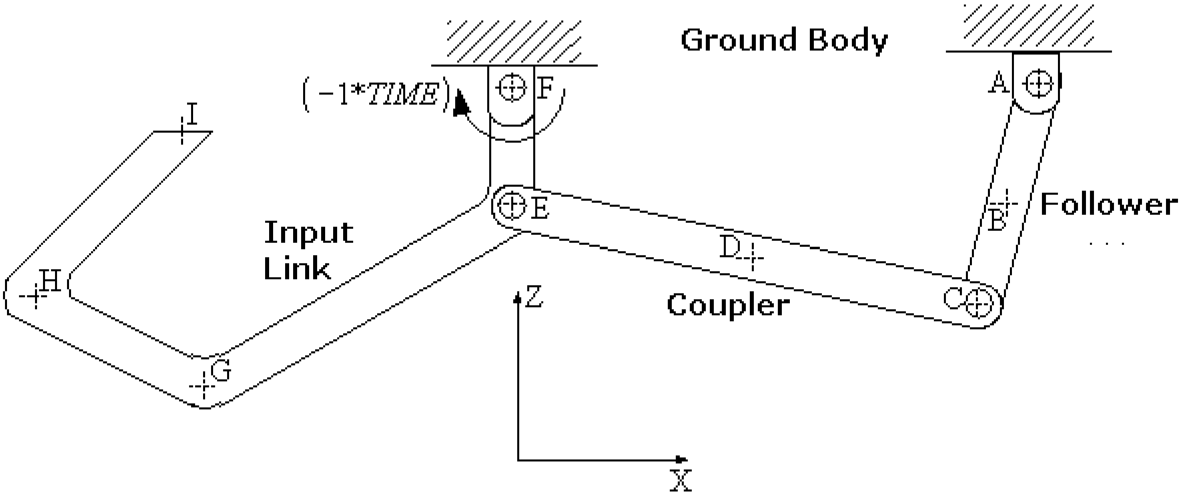
- Create a model of a four-bar trunk lid mechanism interactively through the MotionView graphical user interface.
- Perform a kinematic analysis on the model using MotionSolve.
- Post-process the MotionSolve results in the animation and plot windows.



Car Trunk-Lid Mechanism

The trunk-lid shown in the image above uses a four-bar mechanism for its opening and closing motions.

A schematic diagram of the four-bar mechanism is shown below:



The four links (bodies) in four-bar mechanism are namely; Ground Body, Follower, Coupler, and Input Link. In this example, the Ground Body is the car body and Input Link is the trunk-lid body. The remaining two bodies (Follower and Coupler) form the part of the mechanism used to aid the opening and closing of car trunk-lid.

The following entities are needed to build this model:

- Points
- Bodies
- Constraints (Joints)
- Graphics
- Input (Motion or Force)
- Output

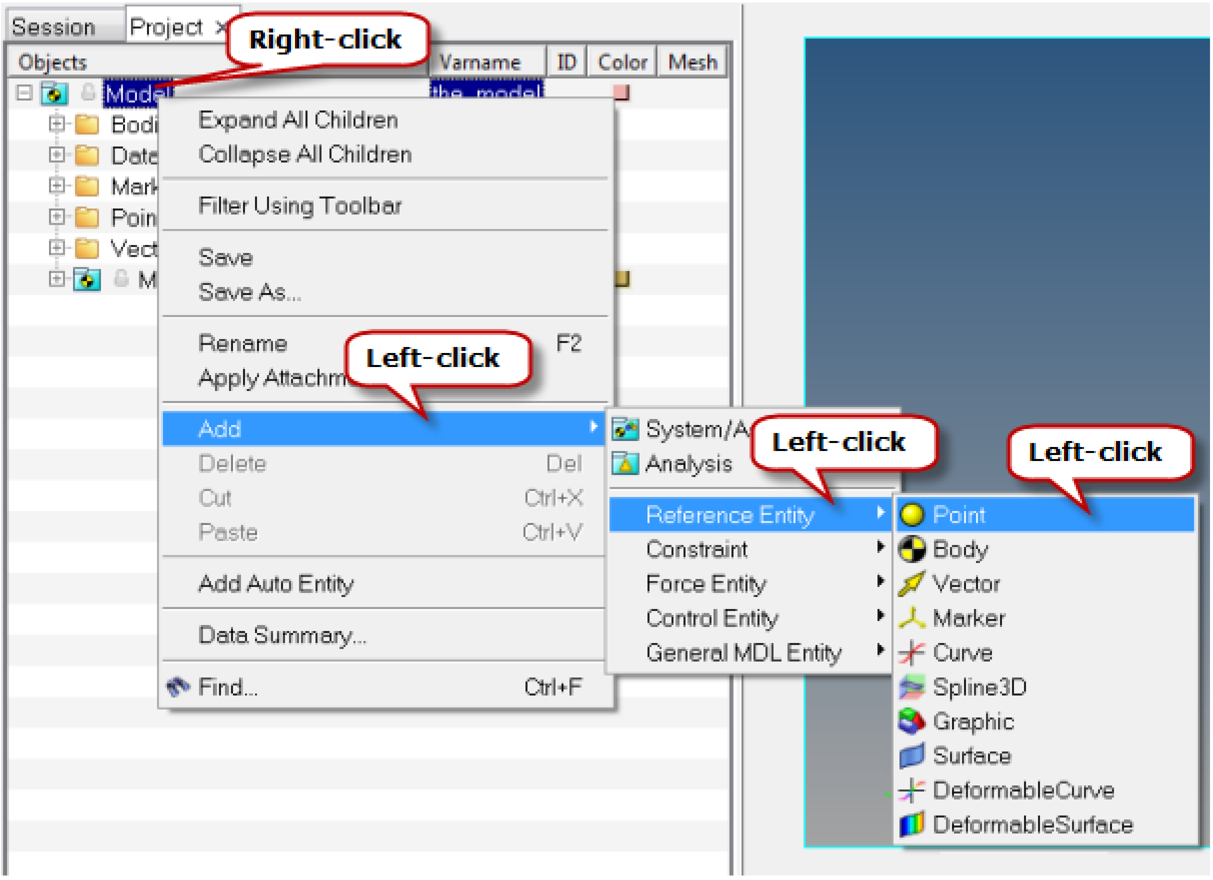
Copy `trunk.hm` and `trunklid.hm`, located in the `mbd_modeling\interactive` folder, to the `<working directory>`.

Exercise

Step 1: Creating points.

1. Start a new MotionView Session.
2. Add a point using one of the following methods:

   – From the **Project Browser**, right-click on ***Model*** and select ***Add > Reference Entity > Point*** from the context menu.



OR

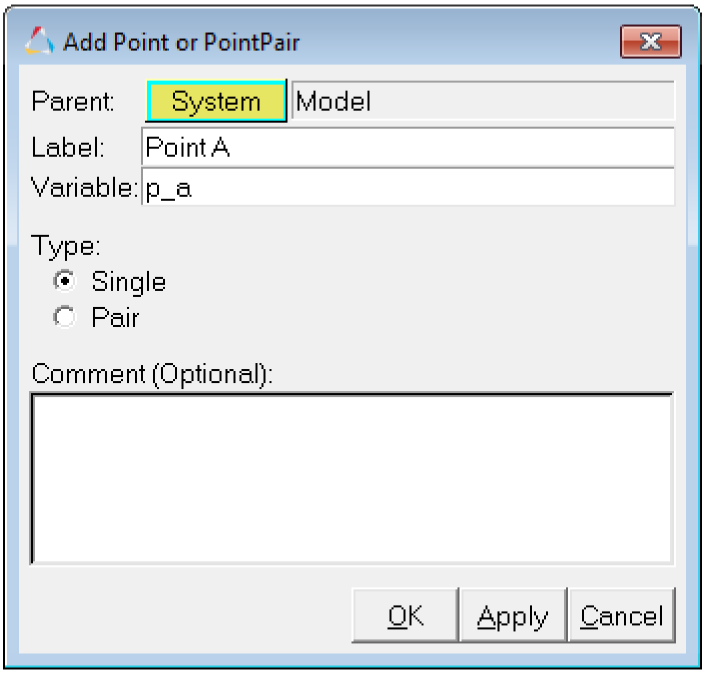
– Right-click on the ***Points*** icon, ●, on the **Reference Entity** toolbar.

The **Add Point or PointPair** dialog is displayed.

Note Other entities like Bodies, Markers, etc. can also be created using either of the methods listed above (Project Browser or toolbar).

3. For **Label**, enter `Point A`.

4. For **Variable**, enter `p_a`.


Add Point or PointPair

Parent: System Model

Label: Point A

Variable: p_a

Type:
● Single
○ Pair

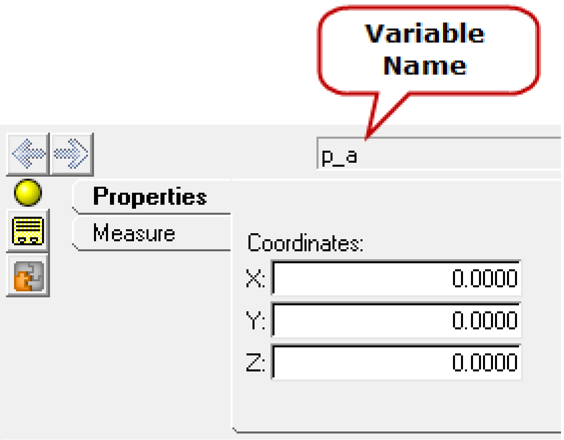
Comment (Optional):

The label allows you to identify an entity in the graphical user interface, while the variable name is used by MotionView to uniquely identify an entity.

Note When using the **Add "Entity"** dialog for any entity, you can use the label and variable defaults. However as a best modeling practice, it is recommended that you provide meaningful labels and variables for easy identification of the entities. For this exercise, please follow the prescribed naming conventions.

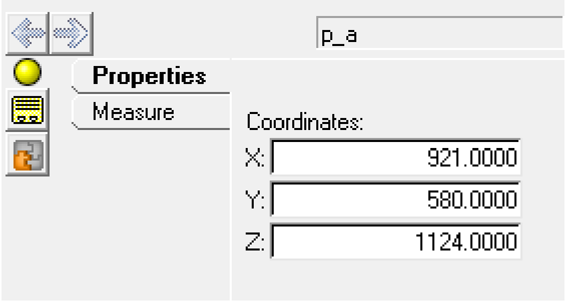
5. Click ***OK***.

   The **Points** panel is displayed. **Point A** is highlighted in the **Points** list of the **Project Browser**.



Points Panel - Properties Tab

6. Enter the values for the ***X***, ***Y***, and ***Z*** coordinates for ***point A***, listed in the table below.



The table below lists the coordinates of the points needed for this model:

Point		Location		
Label	**Variable**	**X**	**Y**	**Z**
Point A	p_a	921	580	1124
Point B	p_b	918	580	1114
Point C	p_c	915	580	1104
Point D	p_d	896	580	1106
Point E	p_e	878	580	1108
Point F	p_f	878	580	1118
Point G	p_g	830	580	1080

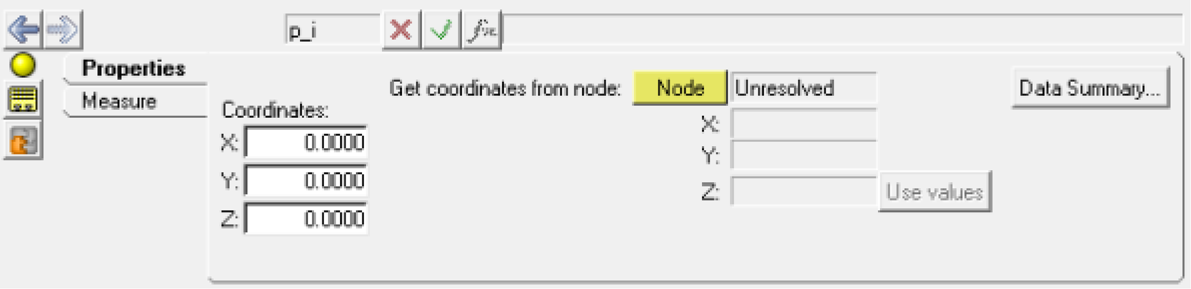
Point		Location		
Point H	p_h	790	580	1088
Point I	p_i	825	580	1109

7. Other (multiple) points can be entered using the following method:

   – Repeat steps 2 through 4 and click ***Apply*** to create points B through I. Remember to substitute B, C, etc., for A when entering the label and variable names in the **Add Point or PointPair** dialog. Clicking the ***Apply*** button allows you to continue to add points without exiting the **Add "Entity"** dialog.

   – After keying in the label and variable name for Point I, click ***OK*** to close the dialog.

   The points panel for Point I is displayed.



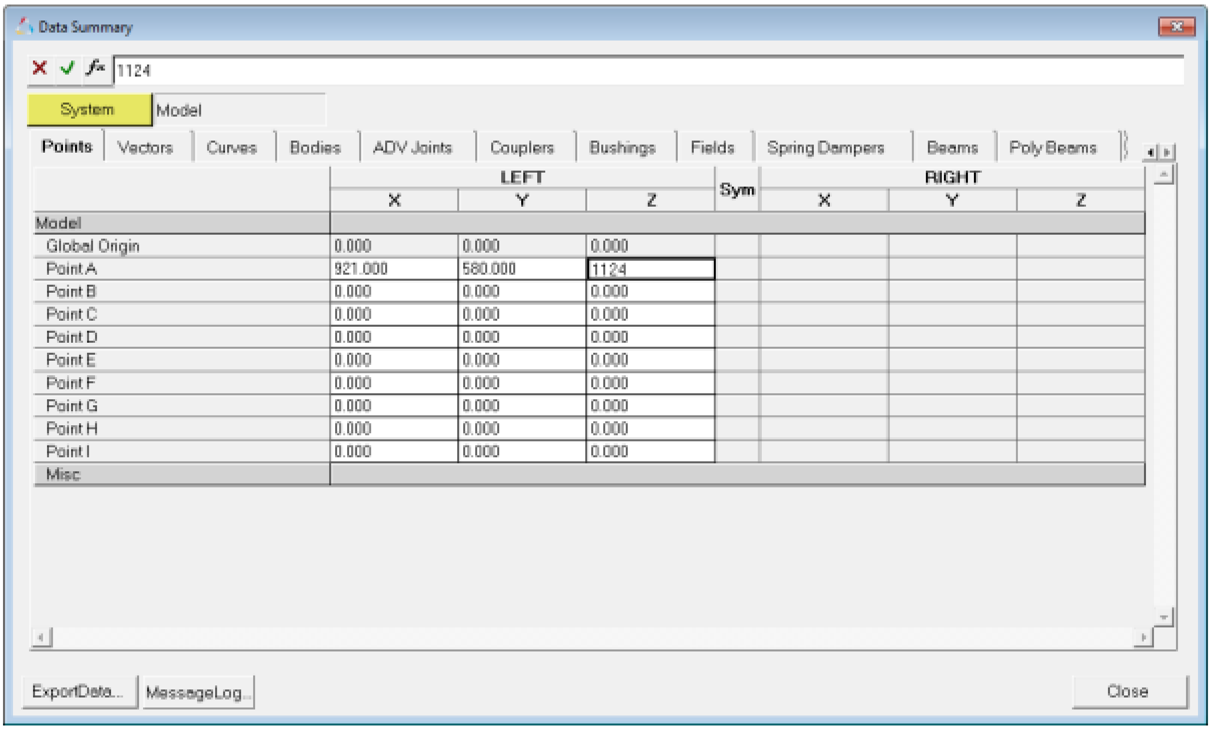
p_i

Coordinates:
X: 0.0000
Y: 0.0000
Z: 0.0000

Get coordinates from node: Node Unresolved
X:
Y:
Z:

– Click the ***Data Summary…*** button located in the upper right corner of the **Points** panel.

The **Data Summary** dialog shows the table of points and you can enter all the coordinates in this table.



1124

	LEFT			Sym	RIGHT		
	X	Y	Z		X	Y	Z
Model							
Global Origin	0.000	0.000	0.000				
Point A	921.000	580.000	1124				
Point B	0.000	0.000	0.000				
Point C	0.000	0.000	0.000				
Point D	0.000	0.000	0.000				
Point E	0.000	0.000	0.000				
Point F	0.000	0.000	0.000				
Point G	0.000	0.000	0.000				
Point H	0.000	0.000	0.000				
Point I	0.000	0.000	0.000				
Misc							

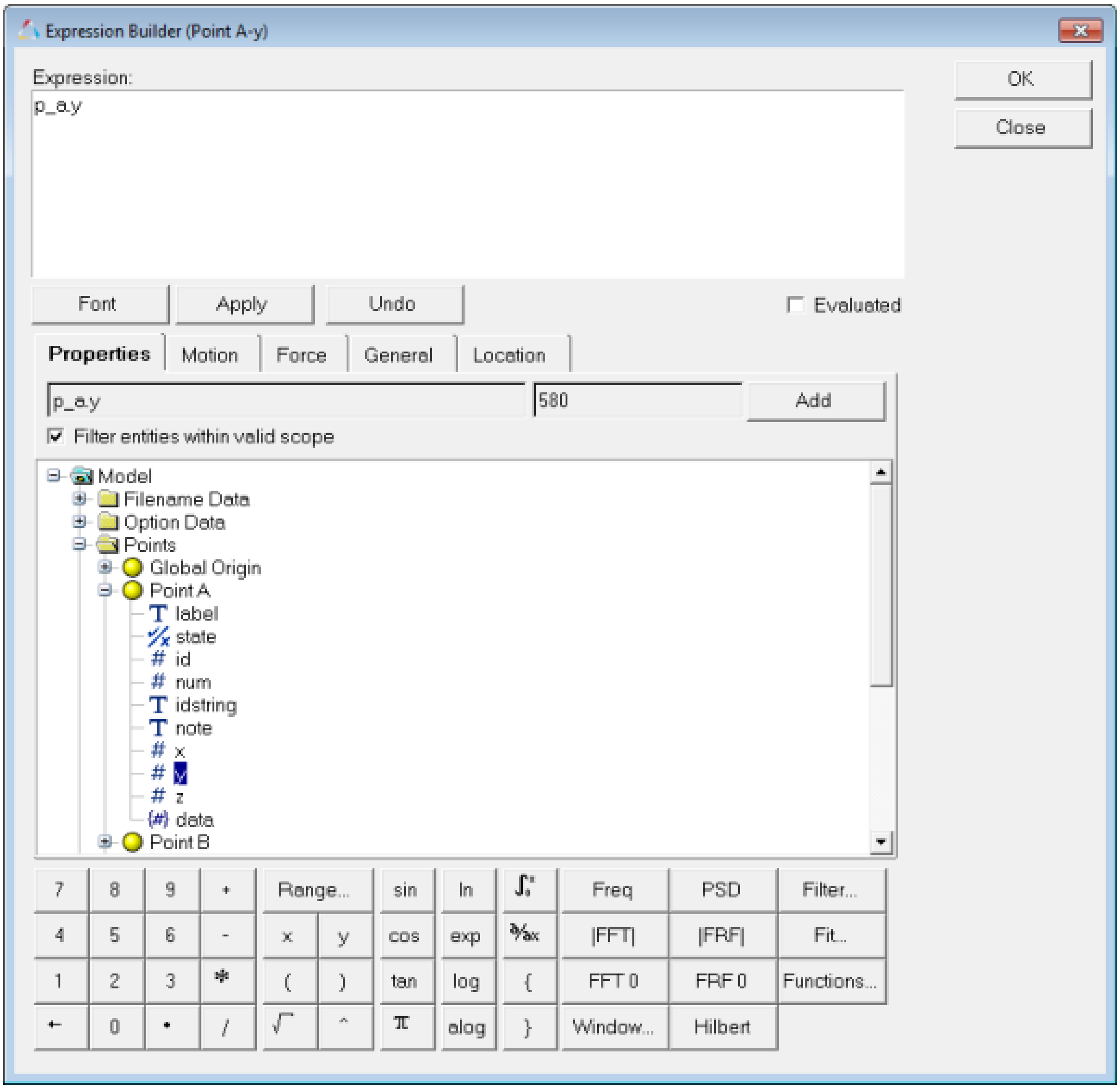
Data Summary Dialog

– Since the Y value of all the points are the same, you can parameterize the value for the points **Point B** to **Point I** to the Y value of **Point A** as follows:

➢ Select the Y coordinate field along **Point B**.

➢ Click on the ƒx button to invoke the **Expression Builder.**

➢ Select the Y value of **Point A** as shown in figure below:



Expression Builder (Point A-y)

Expression: p_a.y

☐ Evaluated

p_a.y 580

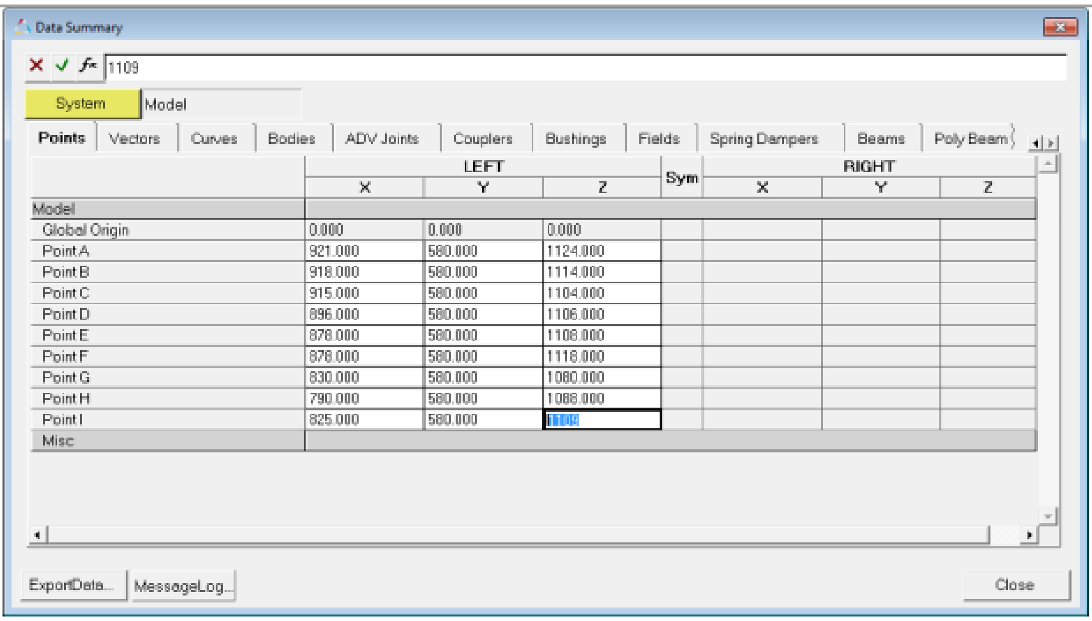
☑ Filter entities within valid scope

Expression Builder

➢ Copy the above expression and paste it into the Y coordinate field of other remaining points.

➢ Enter the X and Z coordinates as listed in the table above.

Note Press ENTER on the keyboard to move to the next field.



1109

	LEFT			Sym	RIGHT		
	X	Y	Z		X	Y	Z
Model							
Global Origin	0.000	0.000	0.000				
Point A	921.000	580.000	1124.000				
Point B	918.000	580.000	1114.000				
Point C	915.000	580.000	1104.000				
Point D	896.000	580.000	1106.000				
Point E	878.000	580.000	1108.000				
Point F	878.000	580.000	1118.000				
Point G	830.000	580.000	1080.000				
Point H	790.000	580.000	1088.000				
Point I	825.000	580.000	1109				
Misc							

- Click ***Close***.

8. Change the view to left, by clicking on the ***XZ Left Plane View*** icon on the **Standard Views** toolbar.

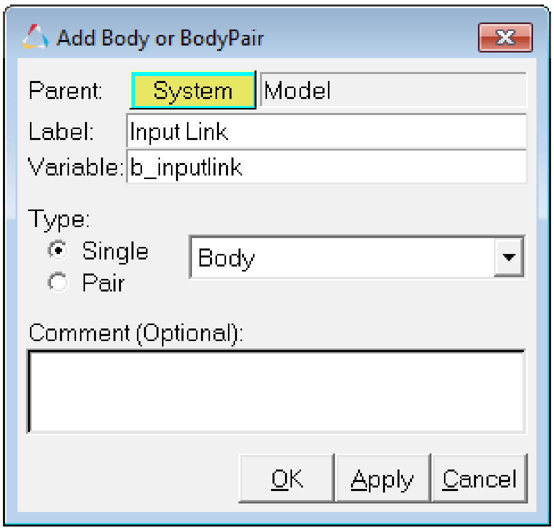
Step 2: Creating bodies.

The mechanism consists of four rigid-body links: Ground (car body), Input Link, Coupler, and Follower. Ground Body is available by default when a new MotionView client is invoked, hence creating the Ground Body separately is not required. In this step, you'll create the Input Link, Coupler, and Follower rigid-body links in the mechanism.

1. Right-click on the ***Bodies*** icon, , on the **Reference Entity** toolbar.

   The ***Add Body or BodyPair*** dialog is displayed.

2. Specify the label as `Input Link` and the variable name `b_inputlink`.



Parent: System Model

Label: Input Link

Variable: b_inputlink

Type:

● Single

○ Pair

Body

Comment (Optional):

3. Click ***OK***.

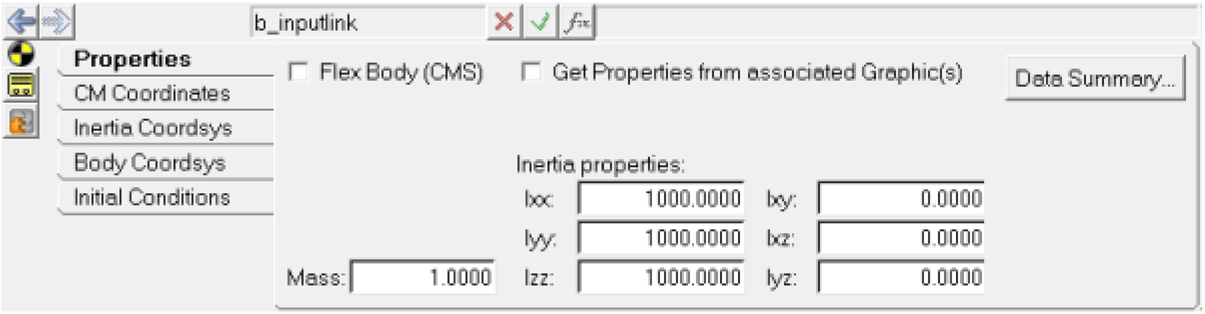
   The ***Bodies*** panel is displayed. The new body that you just added is highlighted in the model tree of the **Project Browser**.

4. Click the ***Properties*** tab.

5. Enter the following values for mass and inertia:

   – ***Mass*** = 1

   – ***Ixx***, ***Iyy***, ***Izz*** = 1000, ***Ixy***, ***Ixz***, ***Iyz*** = 0



b_inputlink

□ Flex Body (CMS) □ Get Properties from associated Graphic(s)

Inertia properties:

		Ixx:	1000.0000	Ixy:	0.0000
		Iyy:	1000.0000	Ixz:	0.0000
Mass:	1.0000	Izz:	1000.0000	Iyz:	0.0000

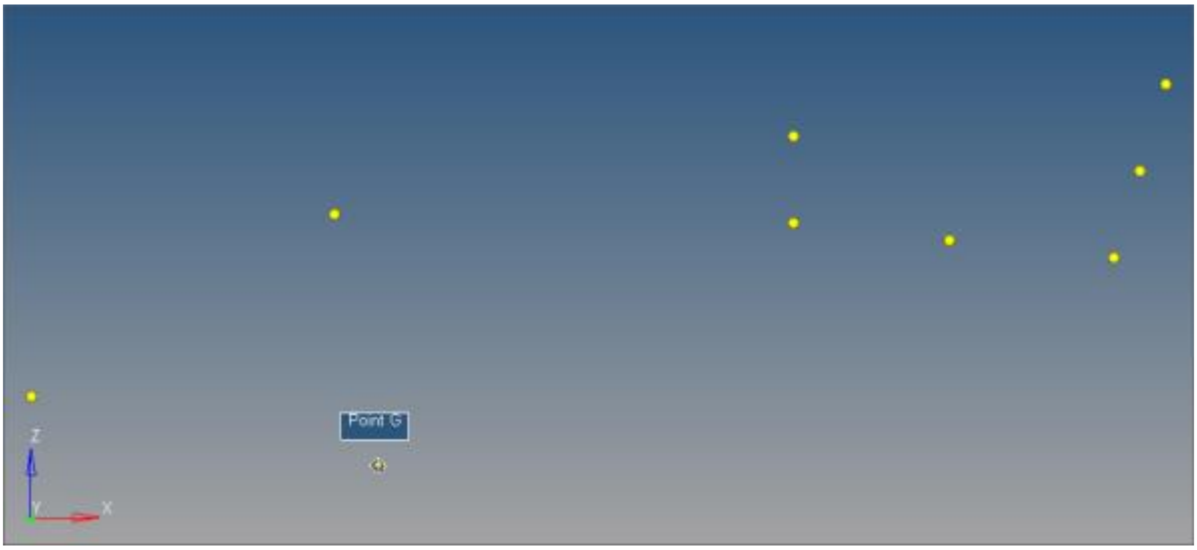
   – Click the ***CM Coordinates*** tab to specify the location of the center of mass of the body.

6. Select the ***Use center of mass coordinate system*** check box.

7. Under the ***Origin***, click the ***Point*** collector Point.

   A cyan border appears around the collector indicating that the collector is now active for selection.

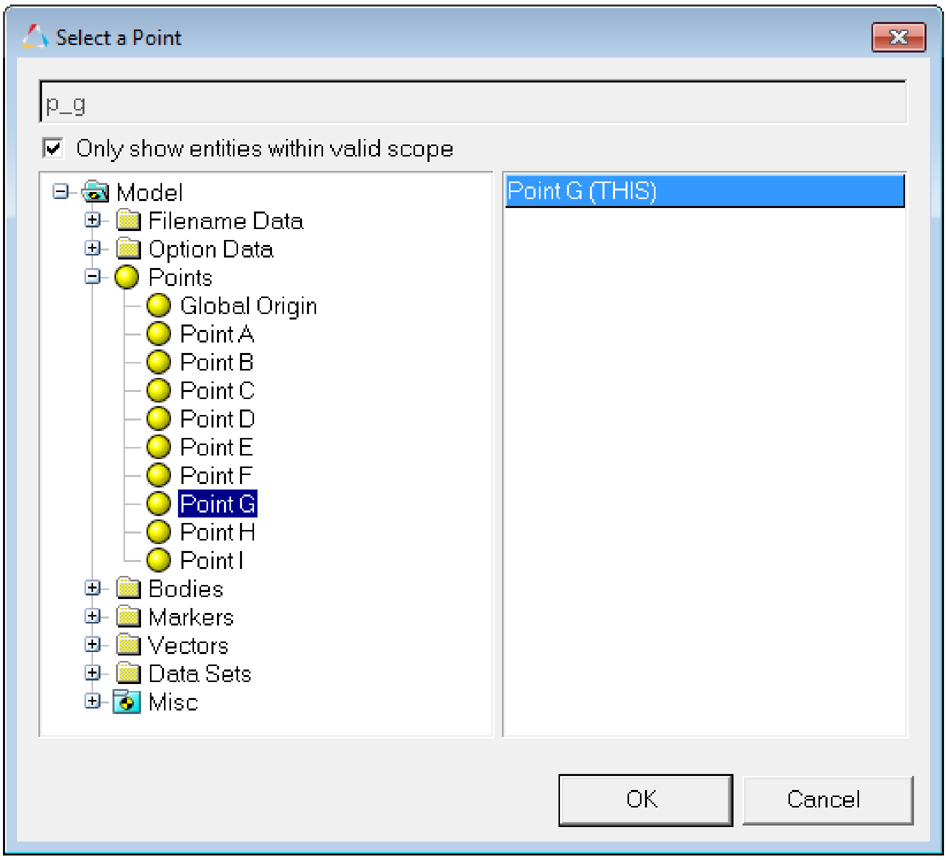
8. From the graphics area, select ***Point G*** on the model by using the left click of the mouse. While selecting, keep the left mouse button pressed and move the cursor over the points to see the label. Release the mouse button when Point G is located.



OR

– Click Point again to launch the ***Select a Point*** dialog.

– Select ***Point G*** from the model tree.


Select a Point

p_g

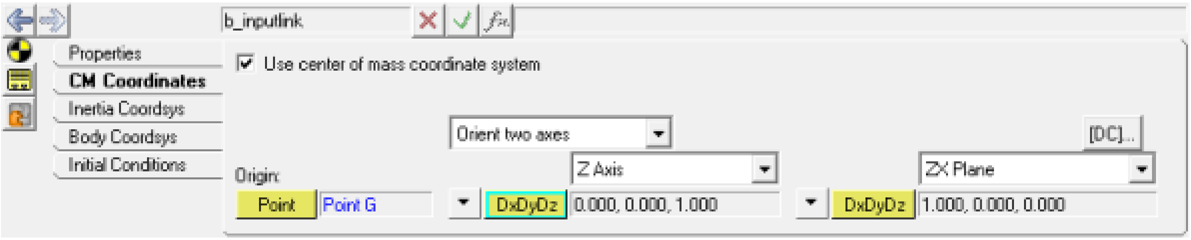
☑ Only show entities within valid scope

- Model
  - Filename Data
  - Option Data
  - Points
    - Global Origin
    - Point A
    - Point B
    - Point C
    - Point D
    - Point E
    - Point F
    - Point G
    - Point H
    - Point I
  - Bodies
  - Markers
  - Vectors
  - Data Sets
  - Misc

Point G (THIS)

OK Cancel

– Click ***OK***.



b_inputlink

Properties
CM Coordinates
Inertia Coordsys
Body Coordsys
Initial Conditions

☑ Use center of mass coordinate system

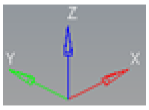
Orient two axes [DC]...

Origin: Point | Point G

Z Axis: DxDyDz | 0.000, 0.000, 1.000

ZX Plane: DxDyDz | 1.000, 0.000, 0.000

Point G is selected as the origin of the center of mass marker for the input link.

Note - The above-mentioned methods for selecting a point can also be applied to other entities such as: body, joint, etc. For selecting the Ground Body or the Global Origin, you can click on the triad representing the Global Coordinate System on the screen .

– Retain the default orientation scheme (Orient two axes) and accept the default values for DxDyDz.

9. Repeat steps 1 through 8 to create the two remaining links with the following label and variable names:

Label	Variable Name
`Follower`	`b_follower`
`Coupler`	`b_coupler`

10. Specify the mass and inertia for these links as:

    – ***Mass*** = `1`

    – ***Ixx***, ***Iyy***, ***Izz*** = `1000`, ***Ixy***, ***Ixz***, ***Iyz*** = `0`

11. Specify points ***B*** and ***D*** as the origin of the center of mass marker for **Follower** and **Coupler**, respectively.

12. Retain the default orientation (Global coordinate system) for the CM marker.

Step 3: Creating revolute joints.

The mechanism consists of revolute joints at four points: A, C, E, and F. The axis of revolution is parallel to the global Y axis.

1. From the **Project Browser**, right-click on ***Model*** and select ***Add > Constraint > Joint*** from the context menu.

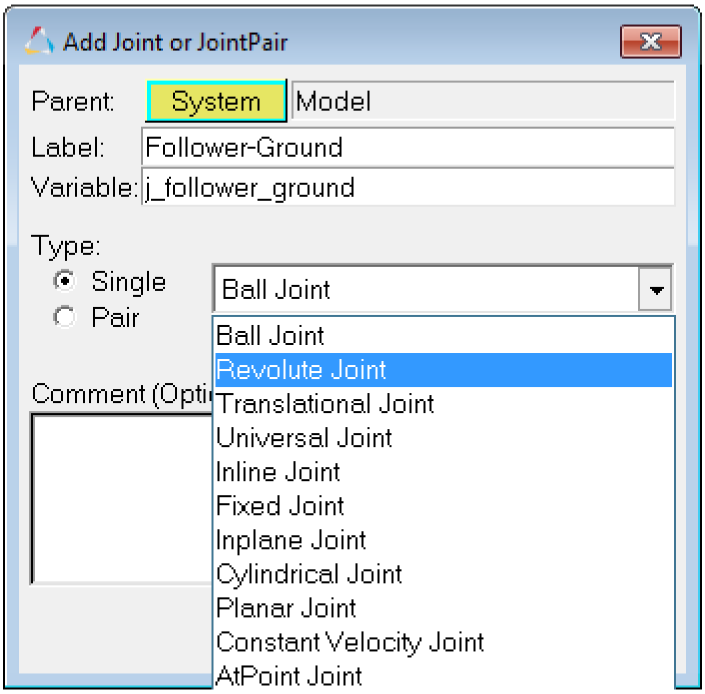
   OR

   – Click the ***Joints*** icon, , on the Constraint toolbar.

   The **Add Joint or JointPair** dialog is displayed.

2. Specify the label as `Follower-Ground` and variable name as `j_follower_ground` for the new joint.

3. Under **Type**, select ***Revolute Joint*** from the drop-down menu.



Add Joint or JointPair

Parent: System Model

Label: Follower-Ground

Variable: j_follower_ground

Type:
● Single
○ Pair

Ball Joint

Comment (Opti

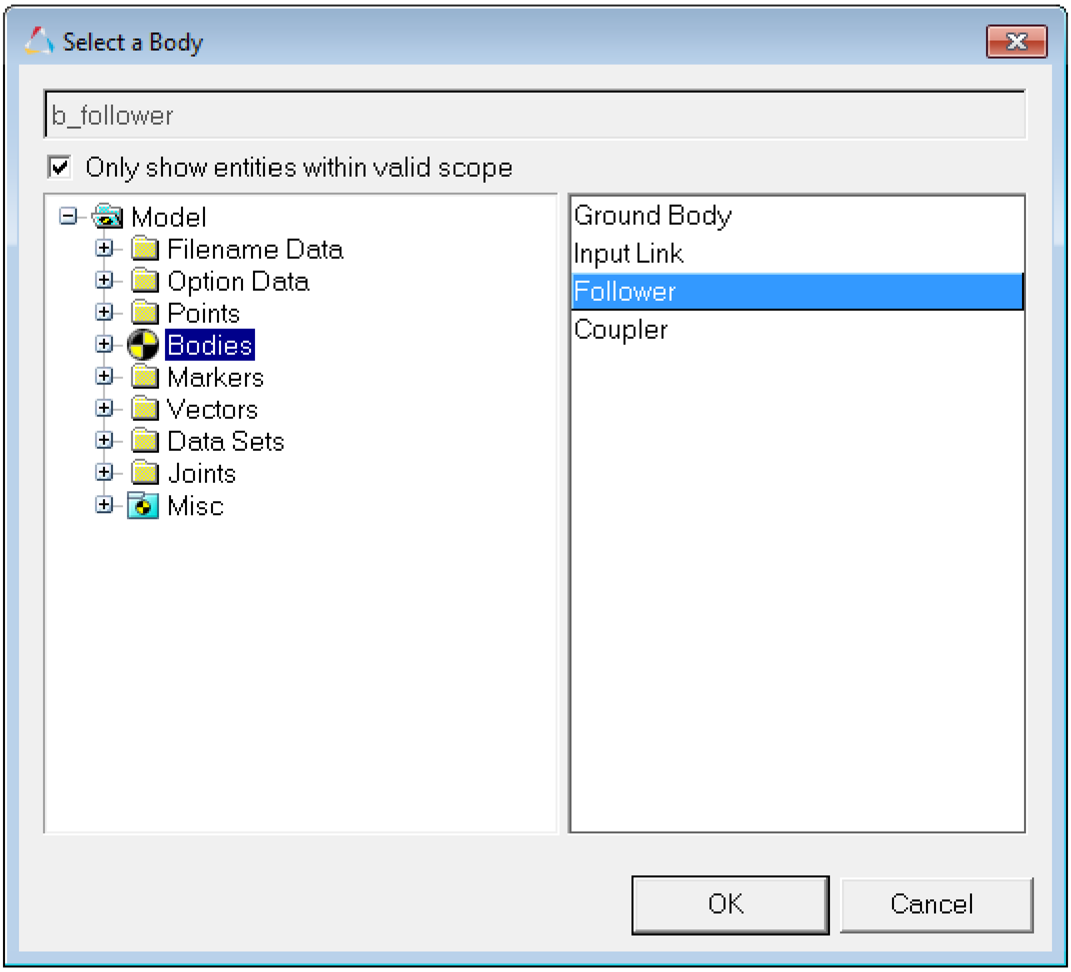
4. Click ***OK***.

   The ***Joints*** panel is displayed. The new joint you added is highlighted in the model tree in the **Project Browser**.

5. Under the ***Connectivity*** tab, double click the first Body collector Body 1.

   The **Select a Body** dialog is displayed.

6. From the model tree, select ***Bodies*** from the left-hand column and ***Follower*** from the right-hand column.



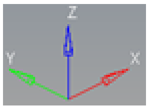
Select a Body

b_follower

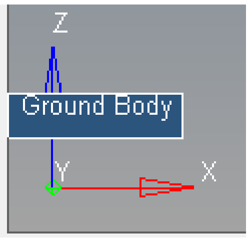
☑ Only show entities within valid scope

7. Click ***OK***.

   Notice that in the **Joints** panel the **Follower** Body is selected for Body 1 and the cyan border moves to Body 2.

8. Click in the graphics window. With the left mouse button pressed move the cursor to the global XYZ triad .

9. Release the left mouse button when ***Ground Body*** is displayed in the graphics window.



10. Under ***Origin***, double click the Point collector Point.

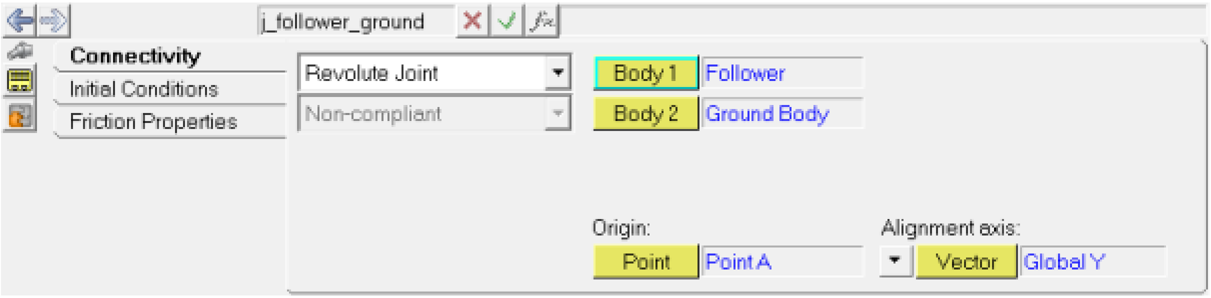
    The ***Select a Point*** dialog is displayed.

11. Select ***Point A*** as the joint origin.

12. Click ***OK***.

13. To specify an axis of rotation, under ***Alignment Axis***, click the downward pointing arrow next to ***Point*** and select ***Vector***.

14. Specify the ***Global Y*** axis vector as the axis of rotation of the revolute joint.



j_follower_ground

Connectivity | Initial Conditions | Friction Properties

Revolute Joint

Non-compliant

Body 1: Follower

Body 2: Ground Body

Origin: Point: Point A

Alignment axis: Vector: Global Y

15. Repeat steps 1 through 14 to create the three remaining revolute joints: points ***C***, ***E***, and ***F***.

Revolute Joint Label	Variable Name	Body 1	Body 2	Point	Vector
Follower-Ground	j_follower_ground	Follower	Ground	A	Global Y
Follower-Coupler	j_follower_coupler	Follower	Coupler	C	Global Y
Coupler-Input	j_coupler_input	Coupler	Input Link	E	Global Y
Input-Ground	j_input_ground	Input Link	Ground	F	Global Y

Revolute joint information

Step 4: Specify a motion for the mechanism.

The input for this model will be in the form of a Motion. A Motion can be specified as **Linear**, **Expression**, **Spline3D**, or **Curve**. In this step, a Motion is specified using an expression.

1. From the **Project Browser**, right-click on ***Model*** and select ***Add > Constraint > Motion*** from the context menu.

   OR

   – Right-click on the ***Motion*** icon, , on the **Constraint** toolbar.

   The **Add Motion or MotionPair** dialog appears.

2. Specify the label as `Motion_Expression` and the variable name as `mot_expr` for the new motion.

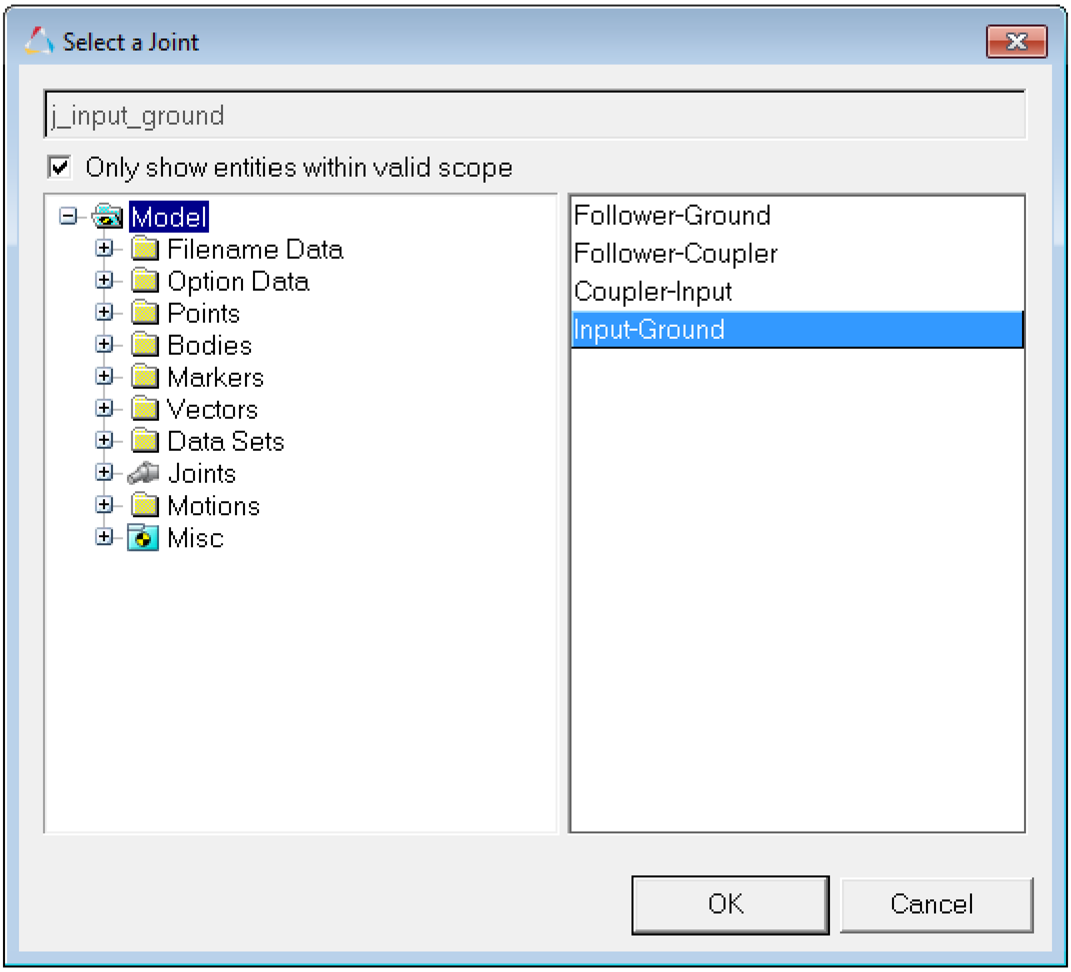
3. Click ***OK***.

   The ***Motion*** panel is displayed. The new motion is highlighted in the model tree in the **Project Browser**.

4. From the ***Connectivity*** tab, double click on the ***Joint*** collector Joint.

   The ***Select a Joint*** dialog is displayed.

5. From the model tree, select the revolute joint at **Point F** (**Input-Ground**) that you created in the previous step.



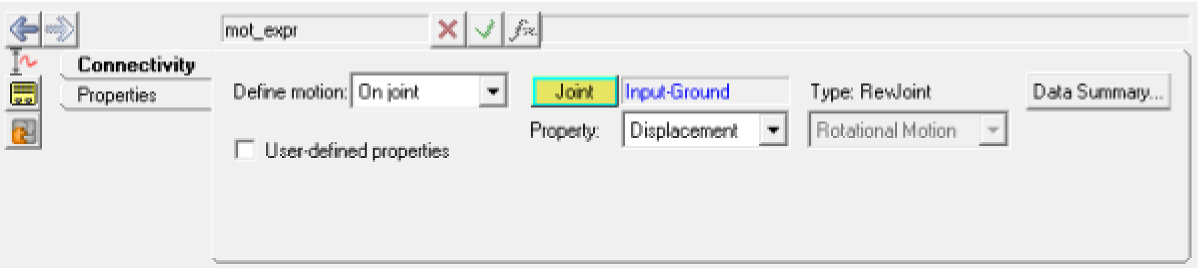
Select a Joint

j_input_ground

☑ Only show entities within valid scope

6. Click ***OK***.

   The **Motion** panel is displayed.



mot_expr

Define motion: On joint

Joint: Input-Ground

Type: RevJoint

Property: Displacement

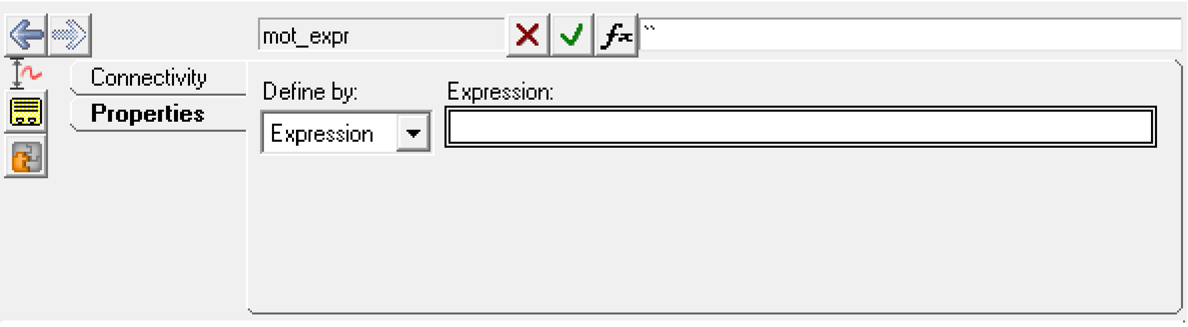
Rotational Motion

□ User-defined properties

7. From the **Properties** tab, select ***Expression*** by clicking on the downward arrow next to **Linear**.

8. Click in the ***Expression*** field.

   The **Expression Builder** is activated.

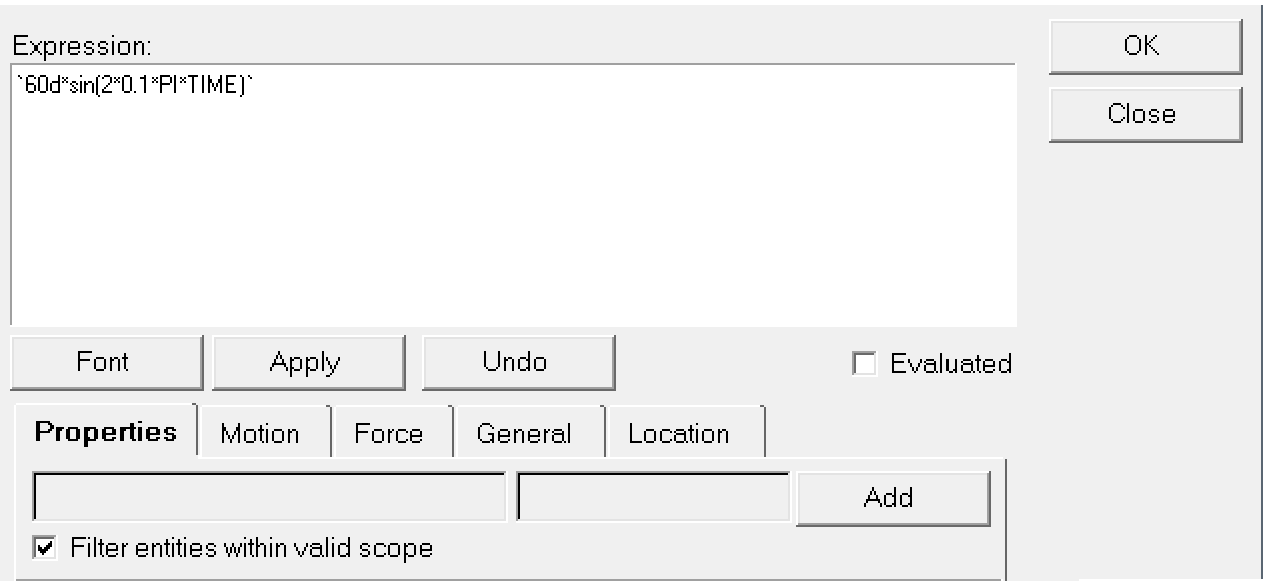


mot_expr

Define by: Expression

Expression:

9. Click on the fx button to open the **Expression Builder** and enter following expression between the back quotes `` `60d*sin(2*0.1*PI*TIME)` ``.



Expression:

`60d*sin(2*0.1*PI*TIME)`

☐ Evaluated

☑ Filter entities within valid scope

The above expression is a SIN function with an amplitude of 60 degrees and frequency of 0.1 Hz. With the above expression the trunk lid is opened to an angle of 60 degrees and back in a total time period of 5 seconds.

10. Click ***OK***.

    Note This method of creating an expression can also be used for specifying non-linear properties for other entities like Force, Spring Damper, Bushing, etc.

Step 5: Creating outputs.

You can create outputs using bodies, points and markers. You can also directly request force, bushing, and spring-damper entity outputs. Another way to create outputs is to create math expressions dependent on any of the above mentioned entities.

In this step, you will:

- Add a displacement output between two bodies using the default entities.
- Add another output to record the displacement of a particular point G on the input link relative to the global frame based on Expressions.

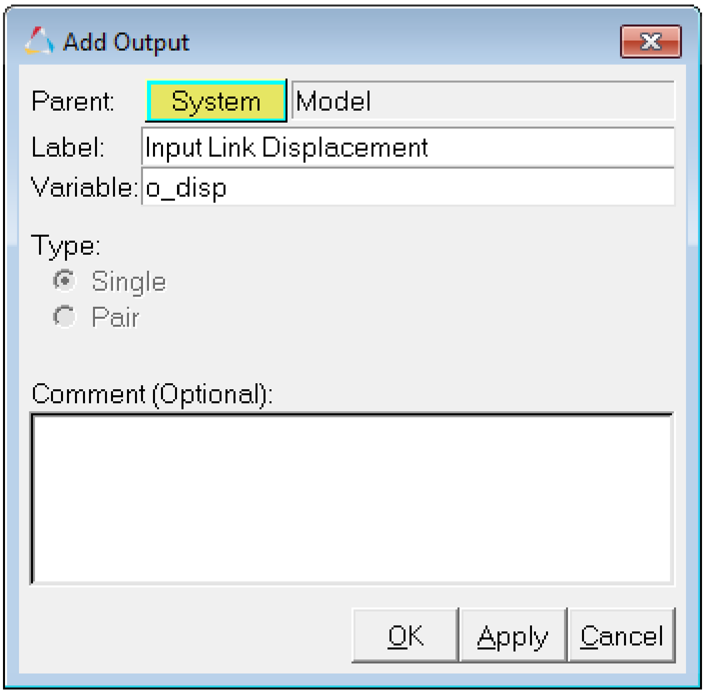
1. From the **Project Browser**, right-click on ***Model*** and select ***Add > General MDL Entity > Output*** from the context menu.

   OR

   – Right-click on the ***Outputs*** icon, , on the **General MDL Entity** toolbar.

   The **Add Output** dialog is displayed.

2. Specify the label as `Input Link Displacement` and the variable name as `o_disp` for the new output.



Add Output

Parent: System Model

Label: Input Link Displacement

Variable: o_disp

Type:

● Single

○ Pair

Comment (Optional):

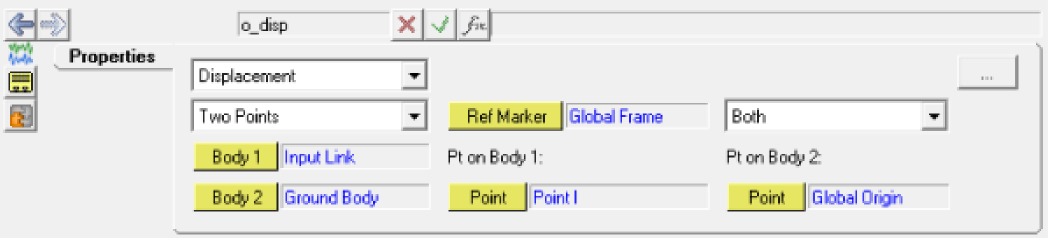
3. Click ***OK***.

4. To create a **Displacement** output between two points on two bodies:

   – For **Body 1** and **Body 2**, select ***Input Link*** and ***Ground Body***, respectively.

   – For **Pt on Body 1** and **Pt on Body 2**, select ***point I*** and the ***Global Origin*** point, respectively.

   – Record the displacement on ***Both*** points, one relative to the other.



o_disp

Properties

Displacement

Two Points

Ref Marker: Global Frame

Both

Body 1: Input Link

Body 2: Ground Body

Pt on Body 1: Point: Point I

Pt on Body 2: Point: Global Origin

5. Add one more output with the label as `Input Link CM Displacement` and the variable name as `o_cm_disp` to calculate the X displacement between the CM markers Input Link and the global origin:

   – From the drop-down menu, select ***Expressions***.

   – Click in the ***F2*** field. This activates the *f=* button.

   – Click on the *f=* button.

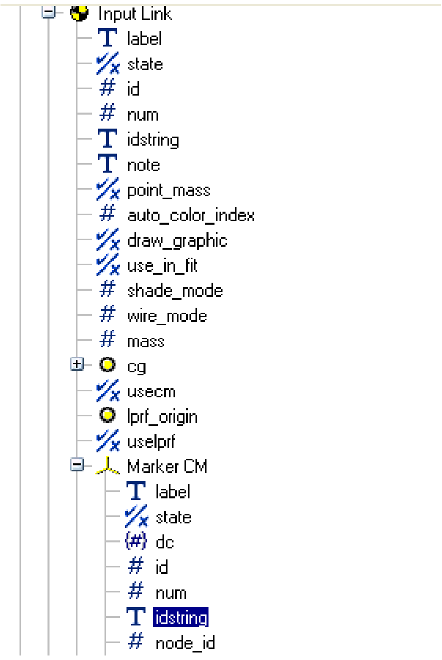
   The **Expression Builder** dialog is displayed.

   – From the **Motion** tab, select ***DX***.

   – Place the cursor inside the brackets after `DX`.

   – From the **Properties** tab, expand the following trees: ***Bodies/Input Link/Marker CM***.

   – Select ***idstring***.



```
Input Link
  T label
  state
  # id
  # num
  T idstring
  T note
  point_mass
  # auto_color_index
  draw_graphic
  use_in_fit
  # shade_mode
  # wire_mode
  # mass
  cg
  usecm
  lprf_origin
  uselprf
  Marker CM
    T label
    state
    dc
    # id
    # num
    T idstring
    # node_id
```

– Click ***Add*** to populate the expression.

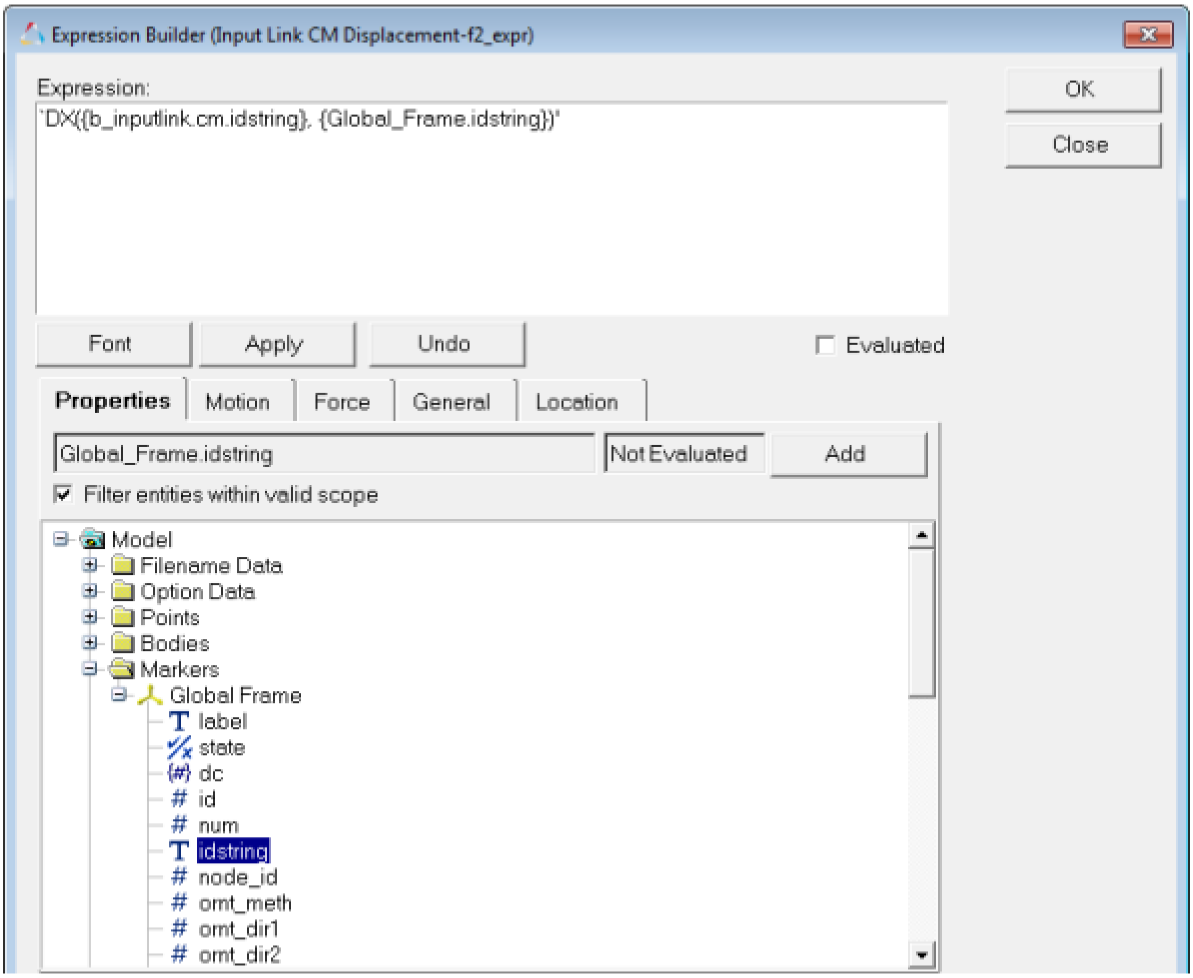
– Add a comma to separate the next expression.

– Add a pair of curly brackets "`{}`".

– Place the cursor inside the added brackets.

– From the **Properties** tab, expand the following items in the tree: ***Markers/Global Frame***.

– Select ***idstring***.



Expression Builder (Input Link CM Displacement-f2_expr)

Expression:

```
`DX({b_inputlink.cm.idstring}, {Global_Frame.idstring})`
```

□ Evaluated

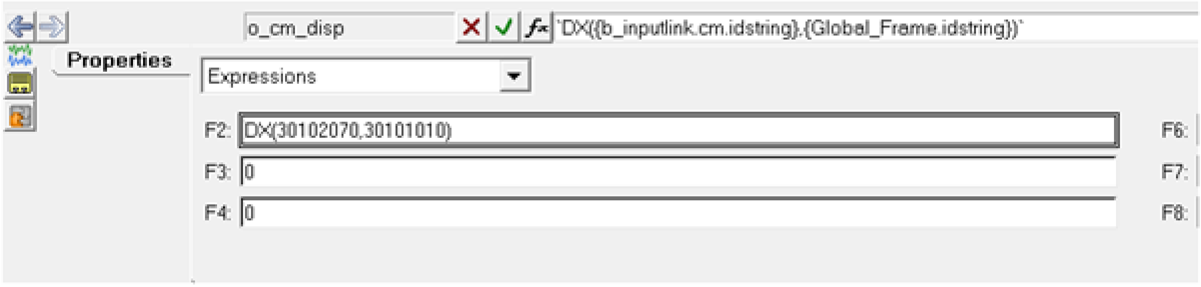
☑ Filter entities within valid scope

Global_Frame.idstring | Not Evaluated

– Click ***Add*** to populate the expression.

6. Click ***OK***.

7. To check for errors, go to the ***Tools*** menu and select ***Check Model***. Any errors in your model topology are listed in the **Message Log**.



o_cm_disp `DX({b_inputlink.cm.idstring},{Global_Frame.idstring})`

Properties

Expressions

F2: DX(30102070,30101010)

F3: 0

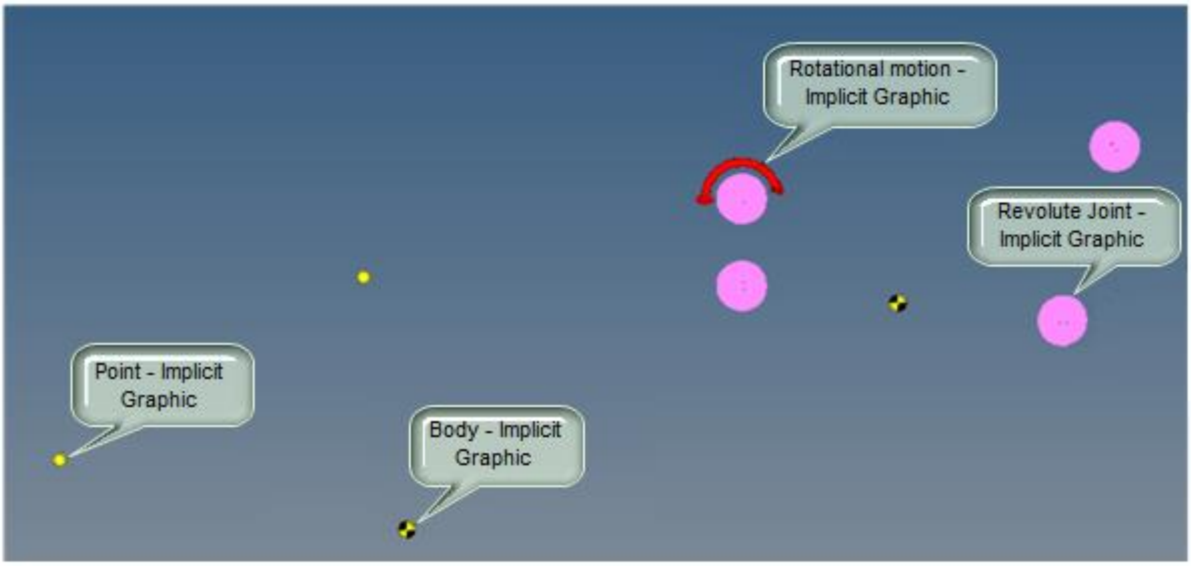
F4: 0

The above function DX measures the distance between Input Link's CM (center of mass) marker and marker representing the Global Frame in the X direction of the Global Frame. Refer to the *MotionSolve Reference Guide* for more details regarding the syntax and usage of this function.

Note The back quotes in the expression are used so that the MDL math parser evaluates the expression. Entity properties like `idstring`, `value`, etc. get evaluated when they are placed inside curly braces `{}`, otherwise they are understood as plain text. Refer to the *Evaluating Expressions in MotionView* topic to learn more about various kinds of expressions and form of evaluation adopted by MotionView.

Step 6: Add graphic primitives.

At this stage your trunk lid model does not contain any graphics, and the entities created in previous steps are represented only by implicit graphics (which are not available in solver deck or results file).


Rotational motion - Implicit Graphic
Revolute Joint - Implicit Graphic
Point - Implicit Graphic
Body - Implicit Graphic

Trunk-lid with only implicit graphics

In this step you will add graphics for visualization of a mechanism. MotionView graphics can be broadly categorized into three types: implicit, explicit, and external graphics.

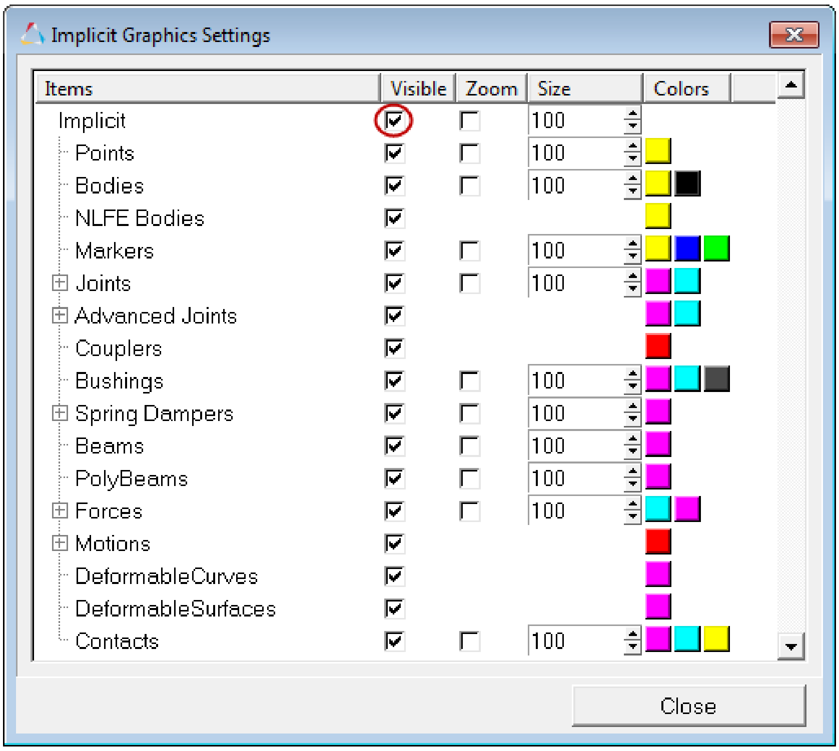
Implicit Graphics	The small icons that you see in the MotionView interface when you create entities like points, bodies, joints, etc. are called **implicit** graphics. These are provided only for guidance during the model building process and are not visible when animating simulations.
Explicit Graphics	These graphics are represented in form of a tessellation, are written to the solver deck and subsequently available in the results. Explicit graphics are of two types.
Primitive Graphics	These graphics help in better visualization of the model and are also visible in the animation. The various types of **Primitive Graphics** in MotionView are **Cylinder**, **Box**, **Sphere**, etc.
External Graphics	One can import in various **CAD** formats or **Hypermesh** files into MotionView. The '**Import CAD or FE using HyperMesh..**' utility in MotionView can be used to convert a **CAD** model or a **Hypermesh** model to **h3d** graphic format which can be imported into MotionView. One can also import **.g**, ADAMS View **.shl** and wavefront **.obj** files directly into MotionView.

MotionView allows you to turn on and off implicit graphics for some of the commonly used modeling entities.

1. To turn on all implicit graphics:

– From the **Model** main menu, select ***Implicit Graphics...***.

– Turn on the ***Visible*** check box.



Items	Visible	Zoom	Size	Colors
Implicit	☑	□	100	
Points	☑	□	100	🟨
Bodies	☑	□	100	🟨⬛
NLFE Bodies	☑			🟨
Markers	☑	□	100	🟨🟦🟩
⊞ Joints	☑	□	100	🟪🟦
⊞ Advanced Joints	☑			🟪🟦
Couplers	☑			🟥
Bushings	☑	□	100	🟪🟦⬛
⊞ Spring Dampers	☑	□	100	🟪
Beams	☑	□	100	🟪
PolyBeams	☑	□	100	🟪
⊞ Forces	☑	□	100	🟦🟪
⊞ Motions	☑			🟥
DeformableCurves	☑			🟪
DeformableSurfaces	☑			🟪
Contacts	☑	□	100	🟪🟦🟨

Note - Implicit graphics of Individual entities can be turned on or off by using the **Visible** check box for each entity.

– Click ***Close***.

The state of the implicit graphics (whether on or off) is not saved in your model (`.mdl`) or session (`.mvw`) files. MotionView uses its default settings when:

- You create a new model in another model window.
- You start a new session.
- You load an existing `.mdl/.mvw` file into a new MotionView session.

To visualize the four-bar mechanism, you need to add **explicit** graphics to the model. In this step, you will add cylinder graphics for Follower, Coupler, and Input Links.

To add **explicit** graphics to your model:

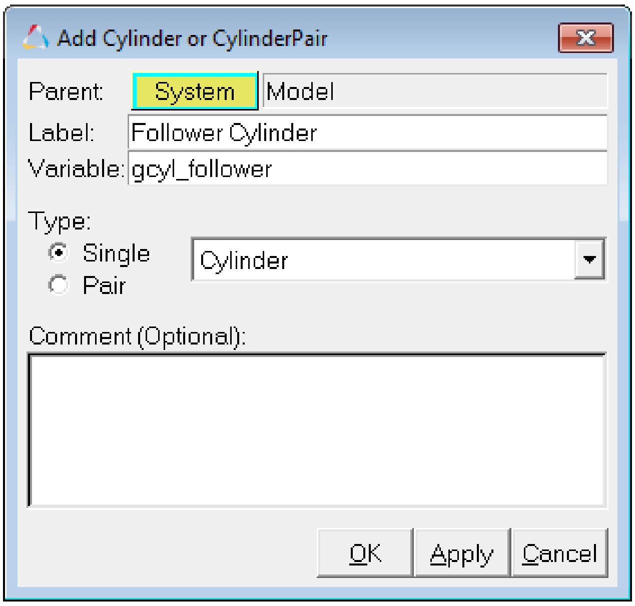
– From the **Project Browser**, right click on ***Model*** and select ***Add > Reference Entity > Graphic*** from the context menu.

OR

– Right-click on the ***Graphics*** panel icon, , on the **Reference Entity** toolbar.

The ***Add Cylinder or CylinderPair*** dialog is displayed.

2. In the ***Add Cylinder or CylinderPair*** dialog, enter the label as `Follower Cylinder` and the variable name as `gcyl_follower`.



Add Cylinder or CylinderPair

Parent: System | Model

Label: Follower Cylinder

Variable: gcyl_follower

Type:
● Single
○ Pair

Cylinder

Comment (Optional):

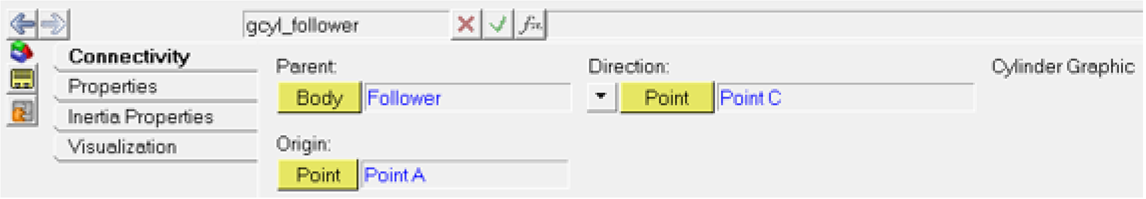
Note The name of the dialog changes with the graphic type. For example, the dialog name changes to **Add Box or BoxPair** when the **Box** graphic type is selected.

3. From the **Type** drop-down menu, select ***Cylinder***. Click ***OK***.

4. In the **Connectivity** tab, double-click the ***Body*** button Body below **Parent**. Select the ***Follower*** from the **Select a Body** list and click ***OK***.

This assigns the graphics to the parent body.

5. To select the origin point of the cylinder, click Point below **Origin**.
6. Pick ***Point A*** in the graphics area.
7. Click Point under ***Direction***.
8. Select ***Point C*** for Point.



gcyl_follower

Connectivity
Properties
Inertia Properties
Visualization

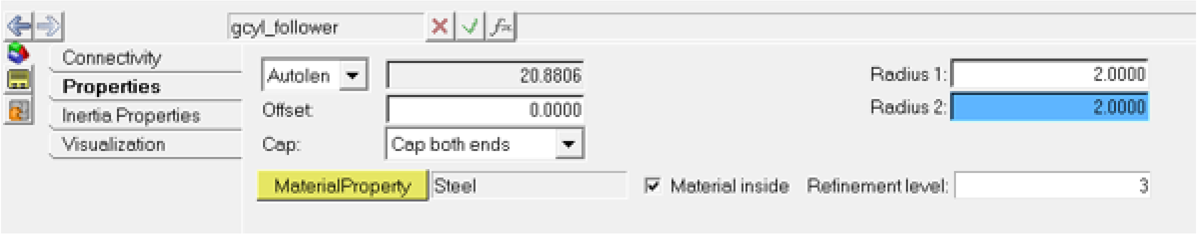
Parent: Body Follower

Direction: Point Point C

Cylinder Graphic

Origin: Point Point A

9. From the **Properties** tab, enter 2 in the **Radius 1:** field.



gcyl_follower

Connectivity
Properties
Inertia Properties
Visualization

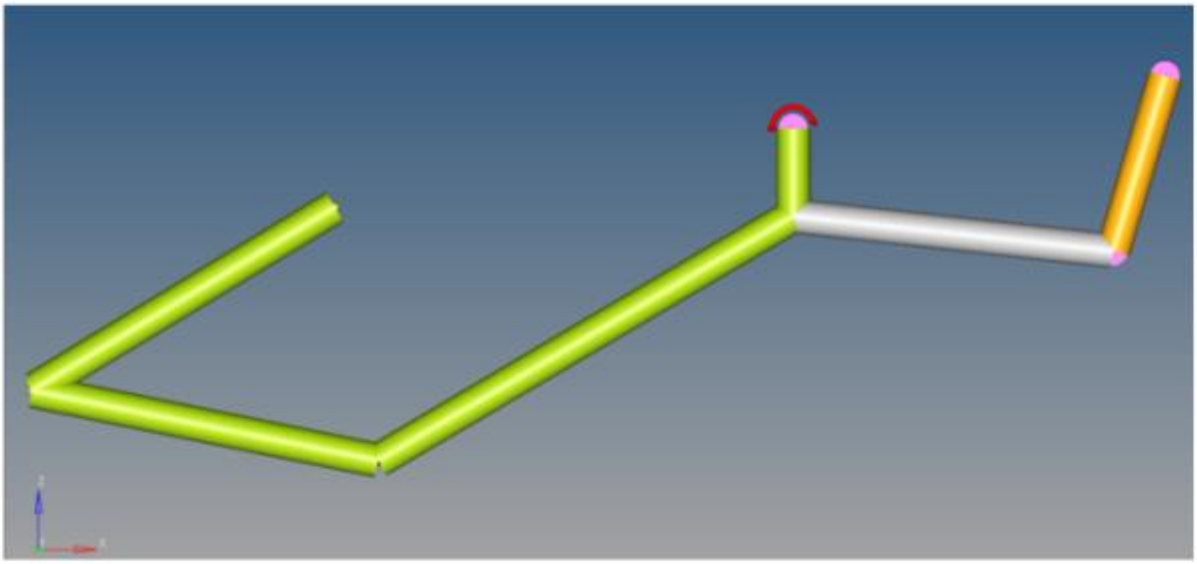
Autolen 20.8806
Offset: 0.0000
Cap: Cap both ends
MaterialProperty Steel
☑ Material inside
Refinement level: 3
Radius 1: 2.0000
Radius 2: 2.0000

Note The cylinder graphic can also be used to create a conical graphic. By default, the **Radius 2** field is parameterized with respect to **Radius 1**, such that **Radius 2** takes the same value of **Radius 1**. Specify different radii to create a conical graphic.

10. For the remaining bodies in your model, follow steps 2 through 9 to create the appropriate explicit graphics for other links.

Label	Variable name	Graphic type	Body	Origin	Direction	Radius
Follower Cylinder	gcyl_follower	Cylinder	Follower	Point A	Point C	2
Coupler Cylinder	gcyl_coupler	Cylinder	Coupler	Point C	Point E	2
Input Link Cylinder 1	gcyl_inputlink_1	Cylinder	Input Link	Point F	Point E	2
Input Link Cylinder 2	gcyl_inputlink_2	Cylinder	Input Link	Point E	Point G	2
Input Link Cylinder 3	gcyl_inputlink_3	Cylinder	Input Link	Point G	Point H	2
Input Link Cylinder 4	gcyl_inputlink_4	Cylinder	Input Link	Point H	Point I	2

After the addition of cylinder graphics for all three links, the Model will look as shown below:



Four-bar mechanism of the trunk-lid assembly

Step 7: Add external graphics and convert a HyperMesh file to an H3D file.

MotionView has a conversion utility that allows you to generate detailed graphics for an MDL model using HyperMesh, Catia, IGES, STL, VDAFS, ProE, or Unigraphics source files. MotionView uses HyperMesh to perform the conversion.

In this step, you will use this conversion utility to convert a HyperMesh file of a car trunk lid into the H3D format.

1. From the **Tools** menu, select ***Import CAD or FE using HyperMesh***.

   The **Import CAD or FE using HyperMesh** dialog is displayed.

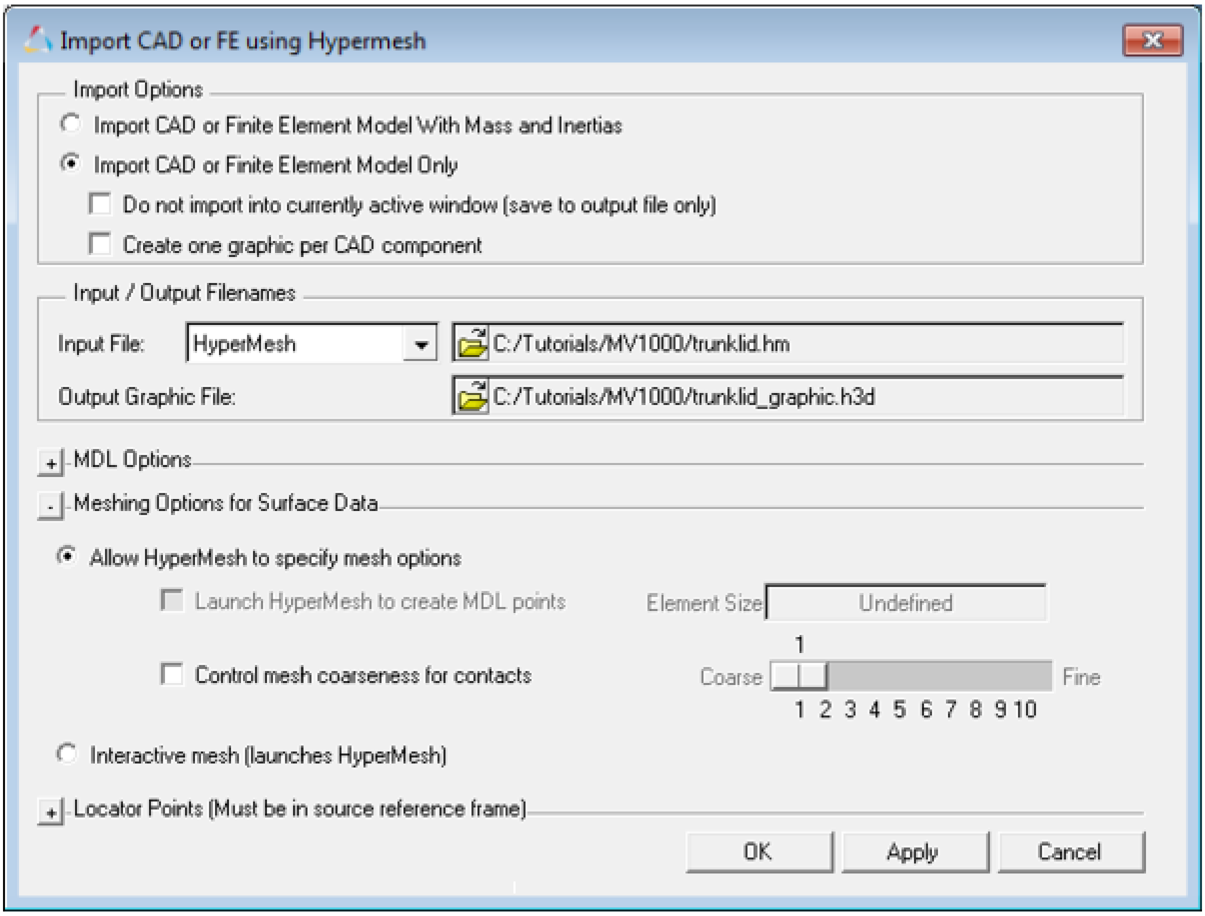
2. Activate the ***Import CAD or Finite Element Model Only*** radio button.

3. From the **Input File** option drop-down menu, select ***HyperMesh***.

4. Click the browser button 🗁 next to **Input File** and select `trunklid.hm`, located in `<working directory>`, as your input file.

   The **Output Graphic File** field is automatically populated with the `trunklid_graphic.h3d` file from the `<working directory>`.

5. Click ***OK*** to begin the import process.



Import CAD or FE using Hypermesh

Import Options

○ Import CAD or Finite Element Model With Mass and Inertias

● Import CAD or Finite Element Model Only

□ Do not import into currently active window (save to output file only)

□ Create one graphic per CAD component

Input / Output Filenames

Input File: HyperMesh C:/Tutorials/MV1000/trunklid.hm

Output Graphic File: C:/Tutorials/MV1000/trunklid_graphic.h3d

+ MDL Options

- Meshing Options for Surface Data

● Allow HyperMesh to specify mesh options

□ Launch HyperMesh to create MDL points

Element Size Undefined

□ Control mesh coarseness for contacts

Coarse 1 Fine (1 2 3 4 5 6 7 8 9 10)

○ Interactive mesh (launches HyperMesh)

+ Locator Points (Must be in source reference frame)

The **Import CAD or FE using HyperMesh** utility runs HyperMesh in the background to translate the HyperMesh file into an H3D file.

Note The H3D file format is a neutral format in HyperWorks. It finds wide usage such as graphics and result files. The graphic information is generally stored in a tessellated form.

6. When the import is complete the **Message Log** appears with the message "***Translating/Importing the file succeeded!***". ***Clear*** the **Message Log**.

7. Use steps 1 through 6 to import the trunk graphics by converting the `trunk.hm` file to `trunk.h3d`.

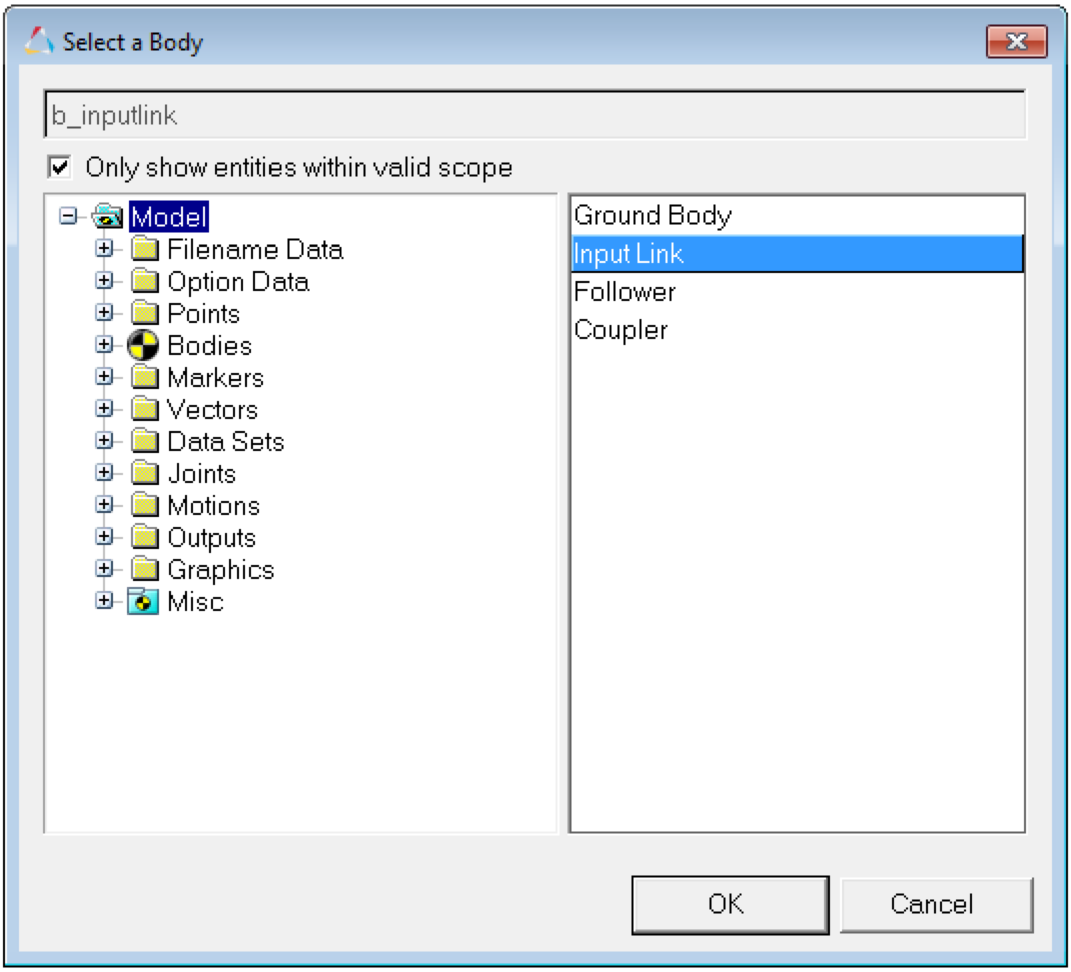
Step 8: Attach H3D objects to the input link and ground bodies.

In this step, you will attach the trunk lid H3D object to the input link and the trunk H3D object to Ground.

1. Click the ***Graphics*** icon on the **Reference Entity** toolbar.

2. Select ***g_trunklid_graphic*** from the graphics area.

3. In the **Connectivity** tab, double click the ***Body*** collector Body under **Parent**. Select ***Input Link*** from the **Select a Body** list.



Select a Body

b_inputlink

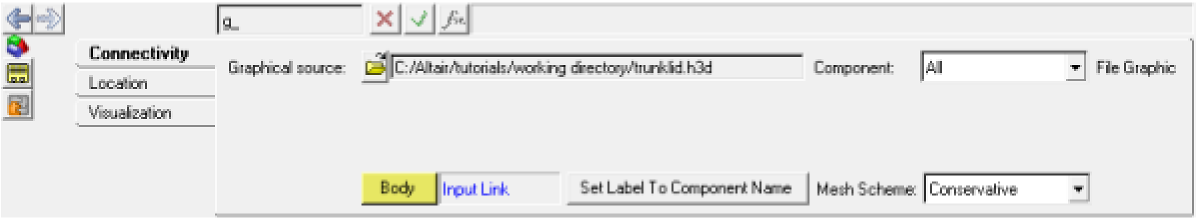
☑ Only show entities within valid scope

- Model
  - Filename Data
  - Option Data
  - Points
  - Bodies
  - Markers
  - Vectors
  - Data Sets
  - Joints
  - Motions
  - Outputs
  - Graphics
  - Misc

Ground Body
Input Link
Follower
Coupler

4. Click ***OK***.

The **Graphics** panel is displayed.



g_

Connectivity
Location
Visualization

Graphical source: C:/Altair/tutorials/working directory/trunklid.h3d

Component: All

File Graphic

Body: Input Link

Mesh Scheme: Conservative

Note Observe the change in the trunk lid graphic color to the **Input Link** body color.

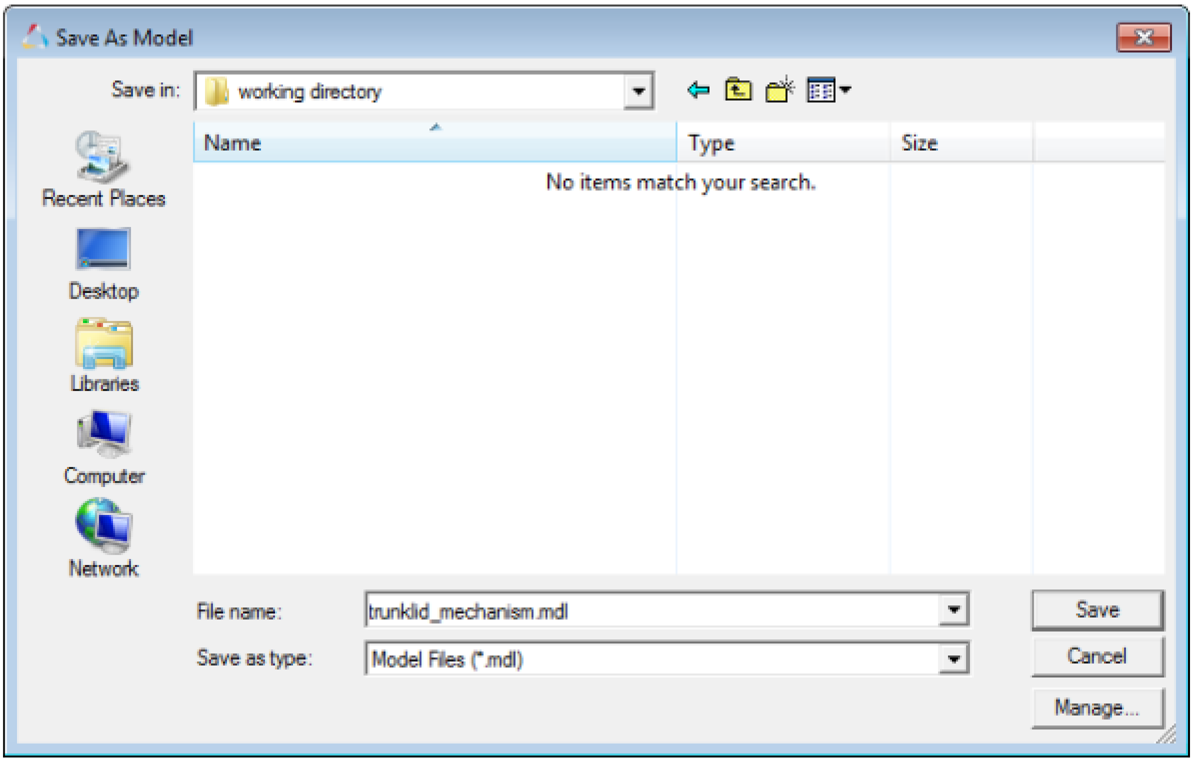
5. Similarly, select the newly created ***g_trunk_graphic*** graphic from the **Project Browser** and set the Body as ***Ground Body***.

6. Click the ***Save Model*** icon on the **Standard** toolbar.

   If the model is new you will be prompted to input the name of the model, otherwise the model will be saved in the working directory with the existing name.

   Note Existing models can be saved to another file using the ***Save As > Model*** option located in the **File** menu.

7. From the **Save As Model** dialog, browse to your working directory and specify the **File name:** as `trunklid_mechanism.mdl`.



Save As Model

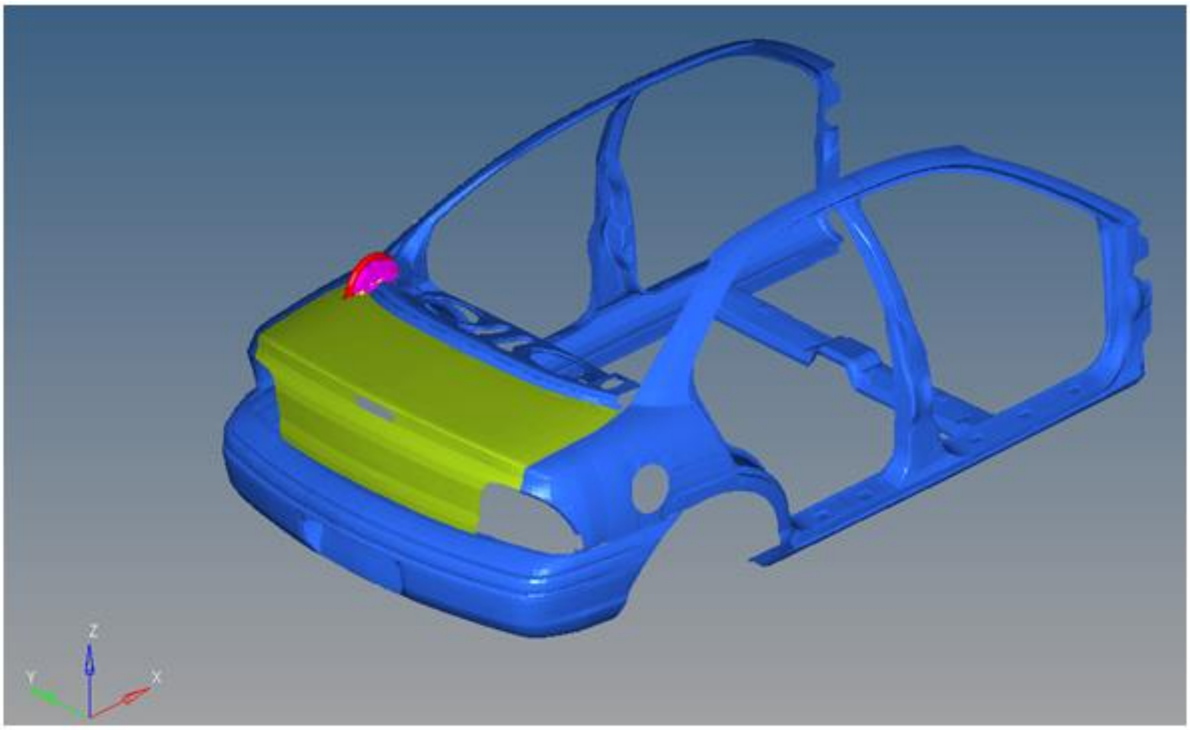
Save in: working directory

File name: trunklid_mechanism.mdl

Save as type: Model Files (*.mdl)

No items match your search.

8. Click ***Save.***



Trunk-lid mechanism

Step 9: Solve the model with MotionSolve.

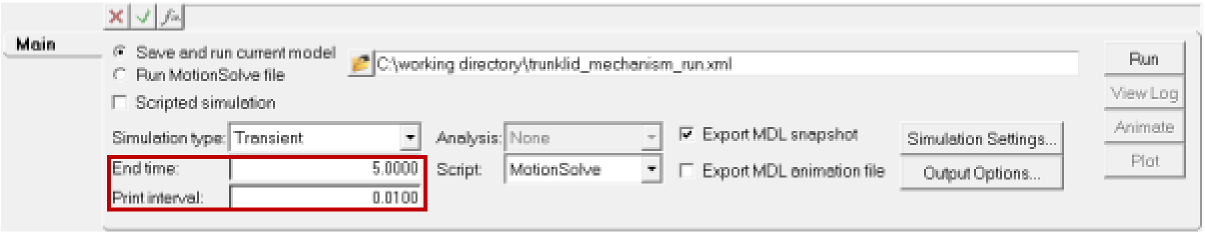
MotionSolve can be used to perform kinematic, static, quasi-static, and dynamic analyses of multi-body mechanical systems. The input file for MotionSolve is an XML file called MotionSolve XML. The solution in MotionSolve can be executed from MotionView.

In this step, you will use MotionSolve to perform a kinematic simulation of the mechanism for a simulation time of 5 seconds, with a step size of 0.01 second.

1. Click the ***Run*** icon, RUN, on the **General Actions** toolbar.
2. Click on the ***Check Model*** button ✓ on the **Model Check** toolbar to check the model for errors.
3. From the **Main** tab of the **Run** panel, specify ***Transient*** as the **Simulation type**.
4. In the field located to the right of the **Save and run current model** option, specify the name for the XML file as `trunklid_mechanism_run`.

   MotionView uses the base name of your XML file for other result files generated by MotionSolve. See the *MotionView User's Guide* for details about the different result file types.

5. Activate the ***Export MDL snapshot*** check box (in order to save the model at the stage in which the Run is executed).



Main

● Save and run current model
○ Run MotionSolve file
□ Scripted simulation

C:\working directory\trunklid_mechanism_run.xml

Simulation type: Transient
End time: 5.0000
Print interval: 0.0100
Analysis: None
Script: MotionSolve
☑ Export MDL snapshot
□ Export MDL animation file

6. Specify an **End time** of 5 for your simulation and a **Print interval** of `0.01` (the time unit is **second** by default).

Note - You can access the **Units** form from the **Forms** panel, ▦.

7. Click the ***Run*** button located on the right side of the panel to solve the model using MotionSolve.

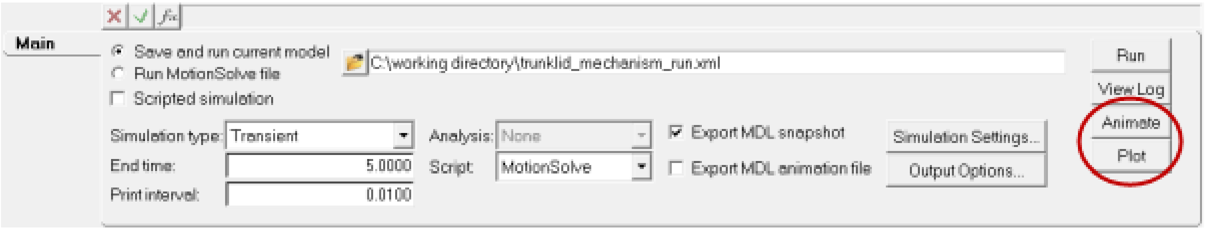
8. Check the **Message Log** for more information.

   Upon clicking **Run**, MotionSolve is invoked and solves the model. The HyperWorks Solver View window appears which shows the progress of the solution along with messages from the solver (Run log). This log is also written to a file with the extension `.log` to the solver file base name.

9. Review the window for solution information and be sure to watch for any warnings/errors.

Step 10: View animation and plot results on the same page.

Once the run is successfully complete, both the **Animate** and **Plot** buttons are active.



Main

● Save and run current model
○ Run MotionSolve file
□ Scripted simulation

C:\working directory\trunklid_mechanism_run.xml

Simulation type: Transient
End time: 5.0000
Print interval: 0.0100
Analysis: None
Script: MotionSolve
☑ Export MDL snapshot
□ Export MDL animation file

1. Click the ***Animate*** button.

   This opens HyperView in another window and loads the animation in that window.

2. To start the animation, click the ***Start/Stop Animation*** icon, ▶, on the toolbar.

3. To stop/pause the animation, click the ***Start/Stop Animation*** icon again, ⏸, on the toolbar.

4. Return to the MotionView window.

5. Click the ***Plot*** button.

   This opens HyperGraph and loads the results file in a new window.

6. Leave the X-axis **Data Type** as ***Time***.

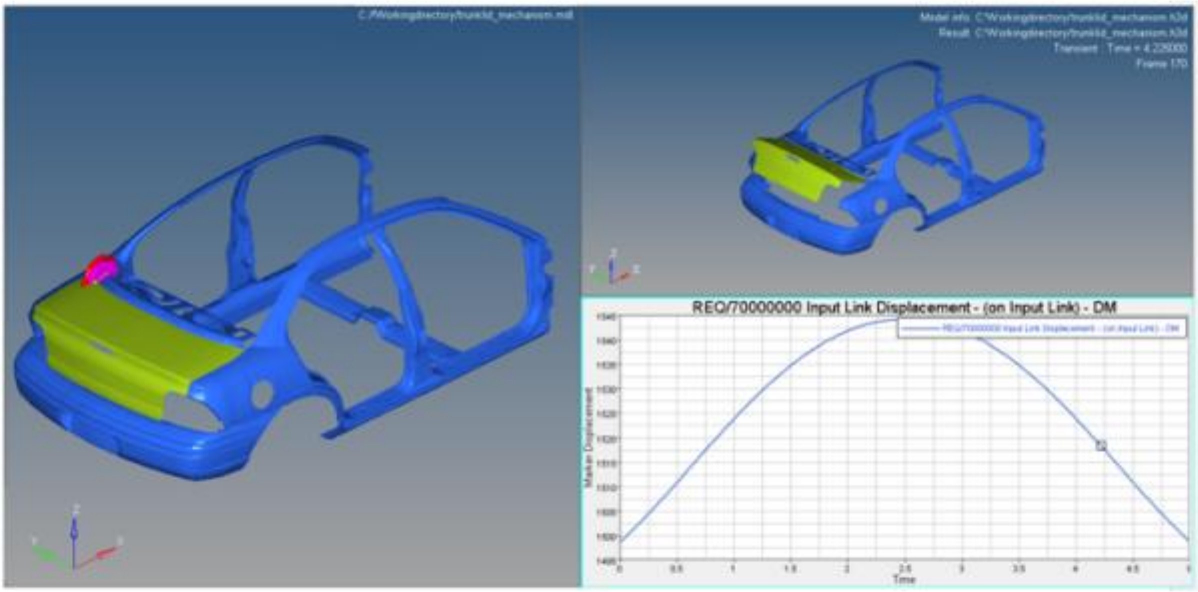
7. Input the following y-axis data:

Y Type	Marker Displacement
Y Request	REQ/70000000 Input Link Displacement (on Input Link)
Y Component	DM (Magnitude)

8. Click ***Apply***.

   This plots the magnitude of the displacement of **Point I** relative to the **Global Origin**.

   Your session should look like the figure below:


C:/Workingdirectory/trunklid_mechanism.mdl
Model info: C:\Workingdirectory\trunklid_mechanism.h3d
Result: C:\Workingdirectory\trunklid_mechanism.h3d
Transient : Time = 4.220000
Frame 170
REQ/70000000 Input Link Displacement - (on Input Link) - DM
REQ/70000000 Input Link Displacement - (on Input Link) - DM
Marker Displacement
Time
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

Session with model, plot, and animation

Step 11: Save your work as a session file.

1. From the **File** menu, select ***Save As > Session File***.

2. Specify the file name as `trunklid_mechanism` for your session.

3. Click ***Save***.

   Your work is saved as `trunklid_mechanism.mvw` session file.

Appendix

Evaluating Expressions in MotionView

Expressions in MotionView are evaluated by two kinds of parsers:

Math Parser

A MotionView parser that evaluates a MotionView expression as real/integer/string for a field as appropriate.

- Real

  This type of field can contain a real number or the parametric expression that should evaluate to a real number. This type of field is found in Points, Bodies , Force – Linear. Note that only the value of the expression as evaluated goes into the solver deck and not the parametric equation.

  Example: `p_a.x, b_0.mass`

- String

  This type of field can contain a string or a parametric expression that should evaluate to a string. This type of field is found in entity such as DataSets with strings as Datamember, SolverString etc. As in case of Linear field, only the value of the expression as evaluated goes into the solver deck and not the parametric expression.

  Example: `b_inputlink.label`

- Integer

  This type of field can contain an integer or a parametric expression that evaluates to an integer. This type of field is found such as DataSets with an integer as Datamember. Even in this case, only the value of the expression as evaluated goes into the solver deck and not the parametric equation.

Templex Parser

A math program available in HyperWorks that can perform more complex programming than the math parser, other than evaluating a MotionView expression.

The following type of fields in MotionView are evaluated by the templex parser that evaluates a parameterized expression:

- Expressions

  This type of field is different than the three listed above because it can contain a combination of text and parametric expression. It is generally used to define a solver function (or a function that is recognized by the solver). This type of expression is embedded within back quotes (` `) and any parametric reference is provided within curly braces {}. The presence of back quotes suggests the math parser to pass the expression through Templex. Templex evaluates any expression within curly braces while retaining the other text as is.

  For example in the expression `` ` DX({b_inputlink.cm.idstring}, {Global_Frame.idstring})` `` , the templex evaluates the ID (as a string) of the cm of the Input link body (`b_ inputlink`) and that of marker Global Frame while retaining "`DX`" as is.

  These fields are available in entity panels such as: Bushings, Motions, Forces with properties that toggle to Expression, Independent variable for a curve input in these entities, and Outputs of the type Expression.