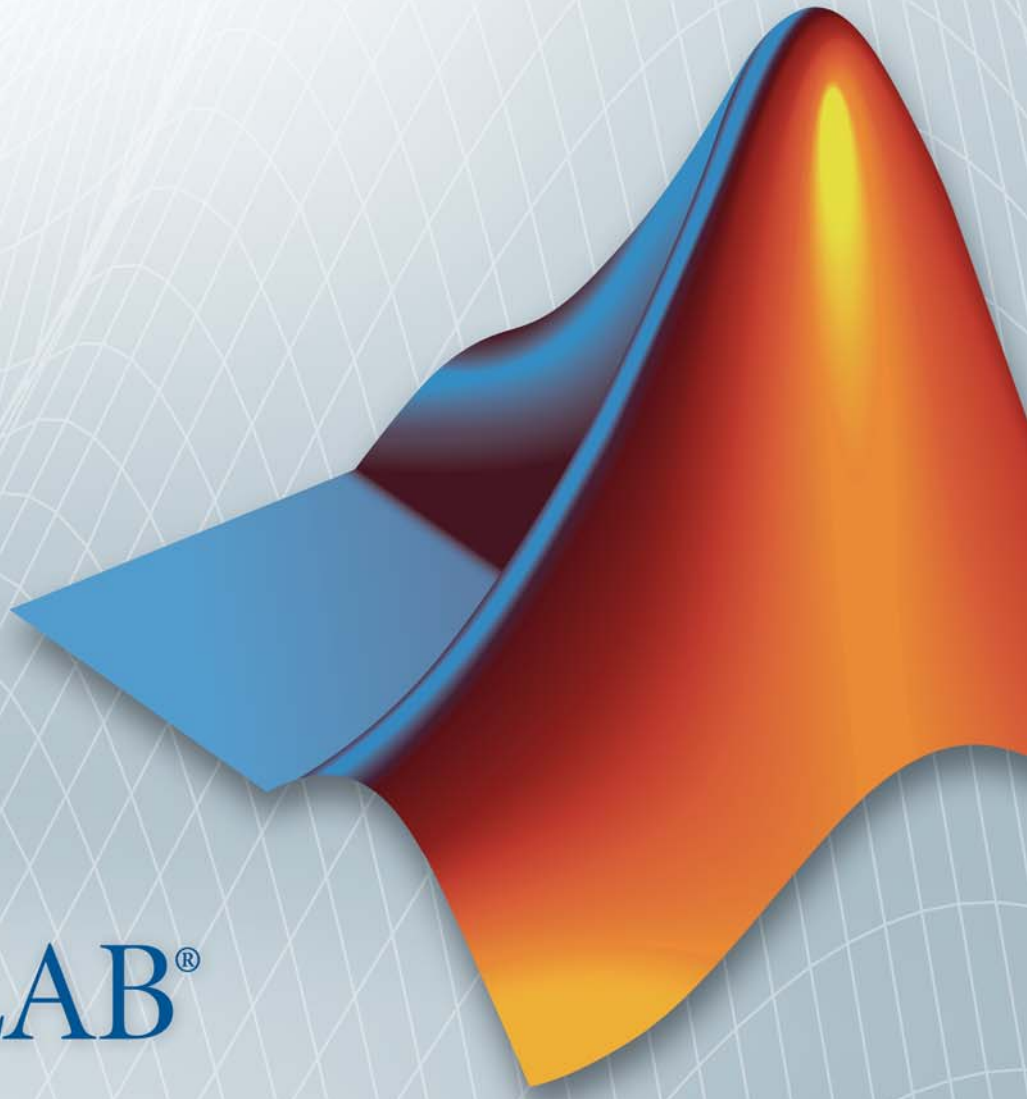
SimMechanics™ Link

User's Guide

R2012a



MATLAB®

MathWorks®

How to Contact MathWorks

`www.mathworks.com`	Web
`comp.soft-sys.matlab`	Newsgroup
`www.mathworks.com/contact_TS.html`	Technical Support

`suggest@mathworks.com`	Product enhancement suggestions
`bugs@mathworks.com`	Bug reports
`doc@mathworks.com`	Documentation error reports
`service@mathworks.com`	Order status, license renewals, passcodes
`info@mathworks.com`	Sales, pricing, and general information

508-647-7000 (Phone)

508-647-7001 (Fax)

The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

SimMechanics™ Link User's Guide

Revision History

October 2008	Online only	Version 3.0 (Release 2008b)
March 2009	Online only	Revised for Version 3.1 (Release 2009a)
September 2009	Online only	Revised for Version 3.1.1 (Release 2009b)
March 2010	Online only	Revised for Version 3.2 (Release 2010a)
September 2010	Online only	Revised for Version 3.2.1 (Release 2010b)
April 2011	Online only	Revised for Version 3.2.2 (Release 2011a)
September 2011	Online only	Revised for Version 3.2.3 (Release 2011b)
March 2012	Online only	Revised for Version 3.2.4 (Release 2012a)

Contents

Getting Started

1 Introducing SimMechanics Link Software

Getting Started with Export

2

3 Computer-Aided Design Translation

4 Custom Linking to CAD and Other External Applications

Getting Started

1

Introducing SimMechanics Link Software

Product Overview

In this section...

Product Definition

Computer-aided design (CAD) is an integral part of engineering design in many industries. CAD tools allow engineers to model their mechanical systems in 3-D space. Although this approach is excellent for geometric modeling, incorporating controllers into this environment is difficult. Simulink with SimMechanics software uses a block-diagram schematic approach for modeling control systems around mechanical devices. The SimMechanics Link utility bridges the gap between geometric modeling and block diagram modeling and simulation, by combining the power of Simulink® and SimMechanics software with CAD.

Using SimMechanics Link and SimMechanics Software Together for Complete CAD Translation

Tip You can start learning how to use SimMechanics Link software with the "Watching a Demo" on page 1-11 section.

With the SimMechanics Link utility, you can create a SimMechanics model from a CAD assembly, in two steps. For examples of complete assembly-to-model translation, see Chapter 3, "Computer-Aided Design Translation".

Exporting CAD Assemblies Into Physical Modeling XML

The first translation step is to use the SimMechanics Link exporter to create an intermediate Physical Modeling XML file from a CAD assembly. Using SimMechanics software, you can then import that XML file to automatically generate a SimMechanics model.

With SimMechanics Link export, you initiate the translation of CAD assemblies into dynamical block diagram models. You export:

- CAD assemblies into Physical Modeling XML format. The XML file captures the mass and inertia of each part in the assembly and the constraint definitions between parts.
- Graphics files to define the body geometries of the assembly parts. The graphics files capture the body geometries of the assembly parts.

Importing Physical Modeling XML to Generate SimMechanics Models

The second translation step is to import the Physical Modeling XML to generate the SimMechanics model, then use that model together with the body geometry graphics files to simulate and visualize the original mechanical system.

- The XML representations of parts and constraints become bodies and joints in a SimMechanics model.
- The generated SimMechanics model uses the exported body geometry graphics files to visualize the bodies.

For more information about XML import and model generation, see the *SimMechanics Visualization and Import Guide*.

Creating a Custom Link using the SimMechanics Link API

For CAD platforms not directly supported, you can use the SimMechanics Link API to connect to your CAD platform API. You create a custom export link that achieves the same export results as for supported CAD platforms, export of the XML and body geometry files.

Related Products

In this section...

Required Products

SimMechanics Link software requires MATLAB® software.

Supported Operating Systems

You can use the SimMechanics Link product on any operating system that supports MATLAB.

Supported CAD Platforms

To build and export CAD assemblies, you need a CAD platform. Such a platform must either be supported directly by the SimMechanics Link utility or have an API you can use to write a custom SimMechanics Link interface.

Other Related Products

SimMechanics Product

Note The SimMechanics Link utility does not require the SimMechanics, Simscape™, or Simulink products.

The Physical Modeling XML and body geometry graphics files exported by the SimMechanics Link utility are intended for use with the SimMechanics product. See the *SimMechanics Visualization and Import Guide*.

Physical Modeling Product Family

Use the Physical Modeling product family to model physical systems in Simulink. In addition to SimMechanics software, it includes:

- Simscape the platform and unifying environment for Physical Modeling products
- SimDriveline™ for modeling and simulating drivetrain systems
- SimElectronics® for modeling and simulating electronic systems
- SimHydraulics® for modeling and simulating hydromechanical systems
- SimPowerSystems™ for modeling and simulating electrical power systems

Installing and Linking SimMechanics Link Software

In this section...

Requirements for Using SimMechanics Link Software

You must install MATLAB first, then install the SimMechanics Link utility. The SimMechanics Link utility works with MATLAB if:

- The MATLAB version corresponds to the SimMechanics Link version as part of the same release.
- The MATLAB, SimMechanics Link, and CAD platform installation architectures match each other.

Steps to Installing and Linking SimMechanics Link Software

1. Install the SimMechanics Link utility as part of your system MATLAB files.

   See "Downloading and Installing SimMechanics Link Software" on page 1-7.

2. Ensure that your MATLAB installation is registered as an automation server on your system, or register it manually.

   See "Registering MATLAB as an Automation Server" on page 1-8.

3. Connect the SimMechanics Link utility to your CAD platform by either:

- Linking SimMechanics Link software to a supported external computer-aided design (CAD) platform, as described in "Linking and Unlinking SimMechanics Link Software To a Supported CAD Platform" on page 1-9.
- Creating your own application programming interface (API) to your CAD platform or other external application, as described in "Linking SimMechanics Link Software to an Unsupported External Application" on page 1-10.

Downloading and Installing SimMechanics Link Software

After you have completed your main MATLAB installation, you can install the SimMechanics Link utility as an add-on to MATLAB by downloading the necessary files from the Web.

1. Go to `http://www.mathworks.com/products/simmechanics/download_smlink.html`.
2. Follow the instructions there and proceed to the confirmation page.
3. Download the `install_addon.m` file and the platform-specific ZIP archive file. Do not change the name of the ZIP file.

   Archives for different operating systems are named accordingly. For example, `win32` indicates an archive for 32-bit Windows®.

   Caution Always be sure to match your MATLAB installation architecture (32-bit versus 64-bit) to the platform-specific ZIP archive that you download.

   For example, download a 32-bit SimMechanics Link archive for a 32-bit MATLAB installation.

   ---

4. Start MATLAB and install the SimMechanics Link add-on by entering the following MATLAB command:

   ```
   install_addon('<add-on ZIP file name>')
   ```

The specified ZIP file is extracted to your *matlabroot* folder. Any required paths are added to the MATLAB path dynamically.

Registering MATLAB as an Automation Server

Note When the SimMechanics Link utility is called from an external application, it attempts to connect to the version of MATLAB corresponding to the SimMechanics Link version. For this connection to succeed, this correct version of MATLAB must be registered as an *automation server* on your system.

When you install MATLAB, it is registered as an automation server with your system. Only one version of MATLAB can be registered as an automation server on your system at any one time.

See "MATLAB COM Automation Server Support" in *MATLAB External Interfaces*.

MATLAB Connection Errors — Manually Registering MATLAB as an Automation Server

When you attempt to connect to the SimMechanics Link utility from your CAD platform, you might get an error indicating that the utility could not connect to MATLAB.

If you see this error, manually register the correct MATLAB version as an automation server using one of these methods:

- Enter the operating system command, `matlab -regserver`.

  This command opens a MATLAB session in non-desktop mode. Close this MATLAB session.

- At the MATLAB command line, enter the command `regmatlabserver`.

Maintaining MATLAB Automation Server Registration After Upgrading, Reinstallation, or Multiple Installations

If you install another version of MATLAB on a system which already has a MATLAB installation, you might accidentally cancel the automation server

registration of the original version when you register the other version. Attempts by the original SimMechanics Link version to connect to the other MATLAB version fail.

Fix the problem by registering (or re-registering) the correct MATLAB version as an automation server. The SimMechanics Link utility then successfully connects to this correct MATLAB version.

Connecting to MATLAB as an Automation Server from an External Application

When you invoke the SimMechanics Link utility from an external application, it connects to MATLAB by either:

- Opening MATLAB as an automation server, if a MATLAB session is not already open in automation server mode.
- Connecting to an existing MATLAB automation server session, if one is available.

  If you want to open a MATLAB automation server session, start MATLAB by entering the system command:

  ```
  matlab -automation -desktop
  ```

  If you have already started MATLAB without the `-automation` option, you can convert that running session to an automation session by entering the MATLAB command:

  ```
  enableservice('AutomationServer',true)
  ```

Linking and Unlinking SimMechanics Link Software To a Supported CAD Platform

To link and unlink SimMechanics Link software, you must have already downloaded and installed the software. Make sure that MATLAB is registered as an automation server on your system.

To link or unlink to a supported external CAD platform, consult the *SimMechanics Link Reference* chapter for your platform.

Supported CAD Platform	Reference Chapter
Autodesk® Inventor®	"Linking and Using the Autodesk Inventor Add-In"
Pro/ENGINEER®	"Linking and Using the Pro/ENGINEER Toolkit"
SolidWorks	"Linking and Using the SolidWorks Add-In"

Linking SimMechanics Link Software to an Unsupported External Application

If you want to link SimMechanics Link software to a CAD platform or other external application that the SimMechanics Link software does not directly support, see Chapter 4, "Custom Linking to CAD and Other External Applications".

Uninstalling and Upgrading SimMechanics Link Software

There is no uninstaller available for the SimMechanics Link utility. If you no longer want to use this software, unlink it from your CAD platform.

If you want to link a newer version of SimMechanics Link software to your CAD platform, unlink the older version and then install and link the newer version.

Watching a Demo

In this section...

About the Demo

The demo consists of a series of steps that:

- Start with a CAD assembly.
- Show how, with SimMechanics Link and SimMechanics software, to translate the assembly into a SimMechanics model through export and import.
- Show how to modify the original assembly, then reexport and reimport it with successive changes that modify the imported model.
- Show how to manually modify the initial generated model, then update it with changes to the original assembly, without losing your manual changes to the model.

The assembly models a double pendulum, subsequently modified to a triple pendulum.

What the Demo Requires

The complete procedure requires the original CAD assembly and platform, as well as SimMechanics Link and SimMechanics software.

Watching the Demo Video

A video of the demo is available at the MathWorks Web site, `www.mathworks.com`. If you are reading this in a browser and have access to the Internet, click here to access the demo page.

What the Video Requires

The video requires a Web browser with a compatible streaming video application.

Learning More About the Demo

The demo video performs the steps of the case study, Updating and Retranslating a CAD Pendulum, contained in Chapter 3, “Computer-Aided Design Translation”.

Learning More

In this section...

Using the MATLAB Help System for Documentation and Demos

You can get help online in a number of ways to assist you while you use SimMechanics Link software. The MATLAB help browser allows you to access the documentation and demo models for all the MATLAB and Simulink based products that you have installed. The online help includes search.

Consult the *MATLAB Getting Started Guide* for more about the MATLAB help system.

Finding Related Documentation

Full use of SimMechanics Link software requires generating, modifying, simulating, and visualizing SimMechanics models based on exported Physical Modeling XML files. For more information about these topics, consult the *SimMechanics Visualization and Import Guide*.

Getting Started with Export

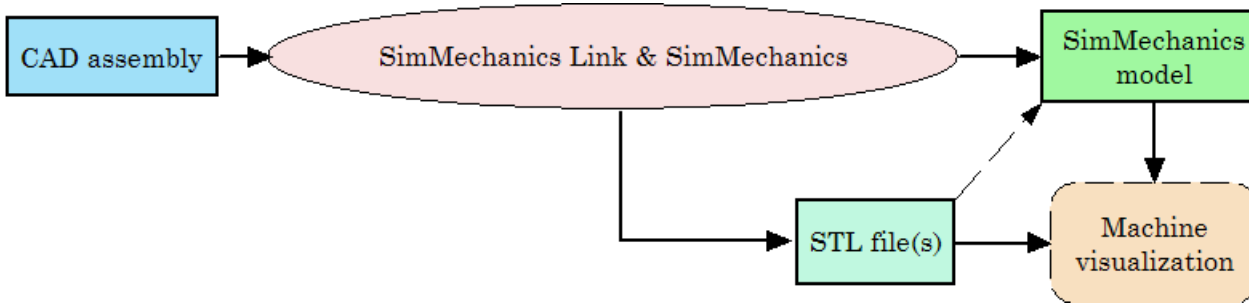
The SimMechanics Link exporter allows you to translate a machine defined externally (such as a computer-aided design assembly) into an intermediate representation. From this intermediate representation, you can generate a SimMechanics model representing the original machine and simulate its motion in the Simulink environment. The intermediate representation allows you to separate the export of external machine data with the SimMechanics Link utility and the generation of the dynamical model with the SimMechanics importer.

Mechanical Export and Translation

In this section...

About CAD Translation: CAD Assembly to SimMechanics Model

Mechanical export using the SimMechanics Link exporter translates mechanical system data from an external application such as a computer-aided design (CAD) platform. You can use this translated data to generate a SimMechanics model of the original mechanical system.



Requirements for CAD Translation

To translate a CAD assembly into a SimMechanics model requires:

- A CAD platform or application
- MATLAB registered as a server
- The SimMechanics Link utility, installed and linked to your CAD platform
- SimMechanics software, based on Simulink and Simscape software

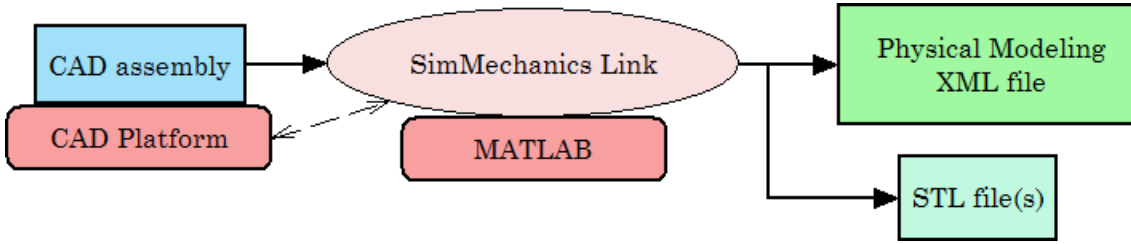
CAD Export: CAD Assembly to Physical Modeling XML

The translation of a CAD assembly into a SimMechanics model uses an portable intermediate representation. From the assembly, the SimMechanics Link exporter creates:

- A Physical Modeling XML file representing selected data needed to dynamically simulate the assembly
- A set of stereolithographic (STL) files to represent the surface geometries of the assembly's bodies

---

Tip STL files can be formatted as binary or ASCII type. The SimMechanics Link exporter creates binary STL files.



From CAD Assembly to Physical Modeling XML

Requirements for CAD Assembly Export

To export a CAD assembly into an intermediate representation requires:

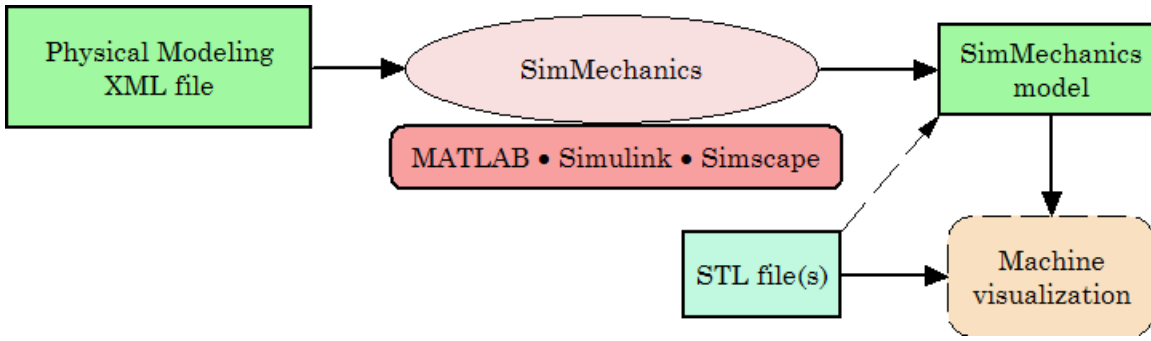
- A CAD platform or application
- MATLAB registered as a server
- The SimMechanics Link utility, installed and linked to your CAD platform

See "Installing and Linking SimMechanics Link Software" on page 1-6.

Tip CAD assembly export does not require SimMechanics, Simscape, or Simulink software.

XML Import: Physical Modeling XML to SimMechanics Model

The automatic generation of a SimMechanics model from external definition as a CAD assembly starts with a portable intermediate representation of Physical Modeling XML. The generated model references the exported STL files for visualizing the surface body geometries of the assembly's bodies.



From Physical Modeling XML to Visualizable SimMechanics™ Model

Requirements for Physical Modeling XML Import

To import an intermediate XML representation into a SimMechanics model requires SimMechanics software, based on Simulink, Simscape, and MATLAB software.

For complete information about generating SimMechanics models from Physical Modeling XML files and using STL files for visualization, see the *SimMechanics Visualization and Import Guide*.

Tip Physical Modeling XML import does not require the SimMechanics Link utility, the original CAD assembly, or a CAD platform.

Designing a CAD Assembly for Export

In this section...

How CAD Assemblies Are Translated into SimMechanics Models

The SimMechanics Link utility creates a Physical Modeling XML file that represents the assembly's parts as bodies and maps the constraints between the parts into joints. This section explains the requirements a CAD assembly must satisfy to produce a valid SimMechanics model when the translation process is complete.

CAD Assembly Component	Corresponding SimMechanics Components
Part	Body
Constraints*	Joints
Reference coordinate systems	Body coordinate systems unattached after import
Assembly Reference	Fundamental Root: Ground – Root Weld – Root Body
Subassembly	Subsystem
Subassembly Reference: [[*trunk*]] – constraint(s) – *subassembly*	Subassembly Root: [[Root Body – Root Weld – Fixed Body]] – Joint – *subsystem* ....
Fixed Part (in a subassembly)	Root Body – Weld – Body

* In some CAD platforms, constraints on parts in a CAD assembly are called *mates*.

Origins, References, Roots, and Root Bodies

Every CAD assembly has a single *assembly origin* and one or more *assembly references* that do not move with respect to the origin. The positions and orientations of all parts refer directly or indirectly to this origin.

A *root body* is a zero-mass, zero-inertia body used in the generated SimMechanics model to represent one or more assembly references. A root body is always welded to ground, so that its zero mass and zero inertia do not affect the model's dynamics. A root body is necessary to represent a fixed anchor for part constraints in the original assembly. This body can carry multiple coordinate systems for this purpose, while the single Ground block in the generated model can carry only one.

Assembly Origin Mapped to SimMechanics World Origin

CAD translation maps the CAD assembly origin to the World coordinate system origin.

Subassemblies and Hierarchies

You can isolate a collection of CAD components (parts and their constraints) into a *subassembly*. CAD translation converts subassemblies into SimMechanics subsystems.

The main assembly is like the trunk of a tree, and its subassemblies are like the branches of the tree. Subassemblies can have subassemblies, and so on. This tree is the assembly's *hierarchy*. Each CAD subassembly has its own *subassembly origin* and *references*. A *fixed part* of a CAD subassembly is a part that is welded to a subassembly reference. It cannot move relative to the subassembly origin.

For an example of subassembly hierarchy, see "Translating a CAD Robot Arm" on page 3-42.

Mass Properties of Assembly Parts

The CAD assembly's parts need to have masses and inertia tensors. When you generate the SimMechanics model, this mass property information is used to specify the properties of the SimMechanics Body block corresponding to each assembly part.

Constraint Geometries

The constraints in your CAD assembly restrict how the assembly's parts can move with respect to each other. Without any constraints, a pair of CAD parts can move with six unrestricted degrees of freedom (DoFs) relative to one another. Constraints between pairs of parts reduce the six to fewer DoFs. SimMechanics joints express DoFs between bodies because SimMechanics bodies by themselves carry no DoFs. CAD constraints and SimMechanics joints are complements of one another.

Each joint is connected to two bodies, each at a body coordinate system (CS). The constraint geometry determines the joints into which CAD export translates the constraints and controls the position and orientation of the body CSs. Each of these body CSs has an origin and axis triad fixed relative to its body. CAD translation creates body CSs on the bodies, as necessary, for connecting joints.

Reference Coordinate Systems

Your CAD assembly can contain reference coordinate systems inserted on parts or elsewhere in the assembly. In general, these coordinate systems are not associated with constraints.

Export of such reference coordinate systems is optional. If these coordinate systems are translated into a SimMechanics model, they appear as coordinate systems on Body blocks, but not associated with Joint connections. When the model is initially imported, the reference coordinate systems are unconnected.

Preparing a CAD Assembly for Translation into a SimMechanics Model

You need to specify enough information in your CAD assembly for the SimMechanics importer to construct a valid model from the XML file.

CAD Export Requires an Assembly

When you export from your CAD platform, you must export a complete assembly into XML, not just a part. If you have only one part, you must embed it in an assembly.

Simplifying Your Assembly with Subassemblies

Use subassemblies to organize your assembly hierarchically. This simplifies your subsequent SimMechanics model by grouping blocks into corresponding subsystems. Follow these guidelines to ensure that your CAD assembly translates into a functioning SimMechanics model:

- You must have at least one fixed part inside each subassembly. (For more about fixed parts, see "Subassemblies and Hierarchies" on page 2-6.)
- Put as many welded components as you can inside rigid subassemblies or combine welded components into a single equivalent part.

  If your assembly has a group of parts that do not move relative to one another, model them as a single part, eliminating unnecessary Body and Joint blocks from the generated SimMechanics model.

Specifying Mass Properties of Assembly Parts

Your CAD platform might compute masses and inertia values from the mass density and geometry of the assembly parts. Otherwise, you must specify the mass and inertia tensor with respect to the part's center of gravity. The SimMechanics Link utility computes the center of gravity of each part automatically.

See "Translating a CAD Part into a Body" on page 3-5 for an example.

Inserting and Naming Reference Coordinate Systems

Depending on your CAD platform and SimMechanics Link interface, you might be able to insert reference coordinate systems on assembly parts or elsewhere in your assembly. If you choose to export these coordinate systems, translation creates corresponding unattached Body coordinate systems in the generated SimMechanics model. You control the selective export of these reference coordinate systems by giving names with a distinctive prefix to these coordinate systems.

You can use reference coordinate systems in several ways.

- Prepare a CAD assembly for manual addition or replacement of Joints after import. These manually added Joints are independent of Joints automatically generated from assembly constraints.

Some CAD constraints are not supported for export. When you translate an assembly into a model, constraints that fail to translate into moving Joint blocks appear as rigid Welds. To represent your assembly correctly requires manual replacement of these Welds with moving Joints.

- Prepare a CAD assembly for manual addition of Constraints, Drivers, Actuators, and Sensors after import. To attach these blocks to Bodies, you need additional Body coordinate systems, apart from the Body coordinate systems automatically generated to connect Joints representing assembly constraints.

You might find it easier to prepare for manually adding Joints, Constraints, Drivers, Actuators, and Sensors by setting up reference coordinate systems before export and attaching these Body coordinate systems to manually added blocks after import.

Specifying Constraint Geometries

You must specify the constraint geometry in the CAD assembly consistently and in enough detail to reconstruct the assembly's DoFs as joints. The relationship between constraints in CAD and SimMechanics joints is not, in general, a simple mapping. Some SimMechanics joints have only one DoF, while others represent more than one DoF. CAD translation often combines multiple DoFs into one joint. Constraint specification details depend on the specific CAD platform.

For an example of configuring constraints, see "Translating a CAD Robot Arm" on page 3-42.

Avoiding Redundant Constraints

Keep constraints simple and few enough to avoid creating unnecessary joints in your SimMechanics model.

For example, consider three parts, P1, P2, and P3, in an assembly. Suppose P1 and P2 are constrained so that there is no movement possible between them. When you attach P3, you could put one constraint between P3 and P1 and the other between P3 and P2. This leads to a redundant joint in the SimMechanics model, making it harder to understand and troubleshoot than if you created only one constraint. In this example, it is better to create a constraint just between P3 and P2, since P2 cannot move with respect to P1 anyway.

Exporting and Re-Exporting a CAD Assembly

In this section...

About CAD Assembly Export

This section explains at a high level how to export CAD assemblies from CAD platforms supported by the SimMechanics Link utility.

Requirements for Export with a Supported CAD Platform

The *SimMechanics Link Reference* presents platform-specific information on:

- Linking a supported CAD platform with the SimMechanics Link utility
- Finding, changing, and applying export settings
- Exporting assemblies in the Physical Modeling XML format through the CAD platform interface

Requirements for Export with an Unsupported CAD Platform

If the SimMechanics Link utility does not support your CAD platform, you can still export assemblies to Physical Modeling XML. But you must first construct your own custom exporter through the SimMechanics Link application program interface (API).

See Chapter 4, "Custom Linking to CAD and Other External Applications".

Configuring the Exporter Controls in Supported CAD Platforms

Once it is linked to the SimMechanics Link utility, each CAD platform's interface has a SimMechanics Link menu that allows you to access the export settings pane or dialog box. Before you export an assembly, you can check, change, and apply the export settings.

Opening the Settings Pane or Dialog Box

1. Open your CAD assembly.
2. From the CAD platform's SimMechanics Link menu, open the settings interface.

At any time, you can:

- Apply your settings.
- Cancel your settings. You lose whatever new settings you have entered.

Configuring the Export Tolerances

In the settings interface, you can configure one or more of the export tolerances. Geometric and numerical differences smaller than the tolerances are treated as zero.

- **Linear tolerance** specifies the smallest significant difference in length.
- **Angular tolerance** specifies the smallest significant difference in angle.
- **Relative roundoff** specifies the smallest significant numerical difference.

Configuring Reference Coordinate Systems for Export

In the export coordinate systems interface, you can require the export of all, some, or none of the reference coordinate systems in your assembly. You control selective export by adding a distinctive prefix to the names of assembly coordinate systems that you want to export.

- You can export all the reference coordinate systems by choosing export and specifying no prefix.

- You can export some of the reference coordinate system by choosing export and specifying a prefix. Reference coordinate systems labeled by names with that prefix are exported.
- You can also choose not to export any reference coordinate systems.

For example, if your assembly has 11 reference coordinate systems (five with names beginning with `A_`, three with names beginning with `B_`, and three with names beginning with `C_`), you can choose to export:

- All 11 coordinate systems by specifying no prefix.
- The five coordinate systems with names prefixed by `A_` by specifying the prefix `A_`.
- The three coordinate systems with names prefixed by `B_` by specifying the prefix `B_`.
- The three coordinate systems with names prefixed by `C_` by specifying the prefix `C_`.

Note Your CAD platform and SimMechanics Link interface might not support inserting and exporting reference coordinate systems. For details, consult the *SimMechanics Link Reference*.

Applying the Export Settings

Apply your export settings by clicking the **OK** button in the settings pane or dialog box. The settings interface closes.

Creating the XML File

To complete export of the assembly to a Physical Modeling XML file:

1. If you changed the assembly or any subassemblies, you must rebuild the assembly and resave it in its native format before exporting it to XML.
2. Export the assembly through the CAD platform's SimMechanics Link interface.

The assembly is saved in a new form as an XML file. The exporter displays a dialog box when it is finished.

The default XML file name and folder are the same as those of the CAD assembly file. With the export interface, you can change the XML file name and folder if you want.

Automatic Export of Stereolithographic Files

In addition to the XML file, the exporter creates a stereolithographic (STL) file for each part in the assembly that represents the part's body surface geometry.

- The STL files automatically receive names based on their respective part names in the assembly.
- The STL files are saved in the folder where the XML file is saved.

Re-Exporting an Assembly After Changes

The results of a CAD export are changed if you:

- Change the assembly or any of its subassemblies or parts
- Change the export settings. See "Configuring the Exporter Controls in Supported CAD Platforms" on page 2-11.

Re-exporting an assembly after such changes is, by default, identical to the initial export steps. See "Creating the XML File" on page 2-12. With no changes to the XML file save procedure, your re-export overwrites the existing XML and STL files with the same names.

Tip For later use and comparison, you might want to preserve the older XML and STL files.

- You can use the old folder and give the re-exported XML file a new name. The exporter still overwrites the old STL files.
- You can use a new folder and avoid overwriting any old files.

Consequences of Re-Exporting Assemblies for Generated Models

While the re-export procedure is the same as, or only slightly different from, the initial export, the resulting new Physical Modeling XML file represents a revised CAD assembly, not a new assembly. To import this XML file with the SimMechanics importer requires choosing how to implement the revisions:

- Create a new SimMechanics model
- Update an existing generated SimMechanics model

You can make more detailed adjustments to the second choice as well, using the persistent properties that the XML file retains when you revise and re-export an existing assembly. See "Retranslating from Assembly to Existing Generated Model" on page 2-18.

The *SimMechanics Visualization and Import Guide* describes in fuller detail the updating of generated SimMechanics models from revised assemblies and re-exported XML files.

Exporting a Robot Arm Assembly for the First Time

In this example, you export a CAD assembly for the first time into Physical Modeling XML.

1. From the SimMechanics Link demos folder, open the robot arm assembly file, `robot.ASSEMBLYFILETYPE`, using a supported CAD platform linked to the SimMechanics Link utility.

   The demo CAD files are under `$matlabroot/toolbox/physmod/smlink/smlinkdemos/`, in the subfolder appropriate to your supported CAD platform. The file extensions of these files are specific to each CAD platform.

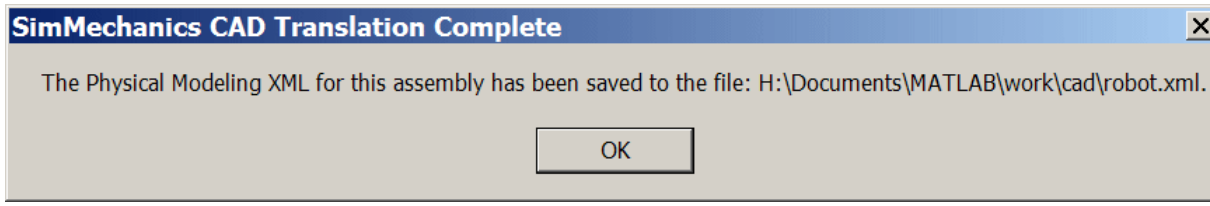
2. If you want, open the settings pane or dialog box from the SimMechanics Link menu in your CAD interface. (See "Configuring the Exporter Controls in Supported CAD Platforms" on page 2-11.)

   Make any adjustments you want to the export settings and apply the settings. The settings interface closes.

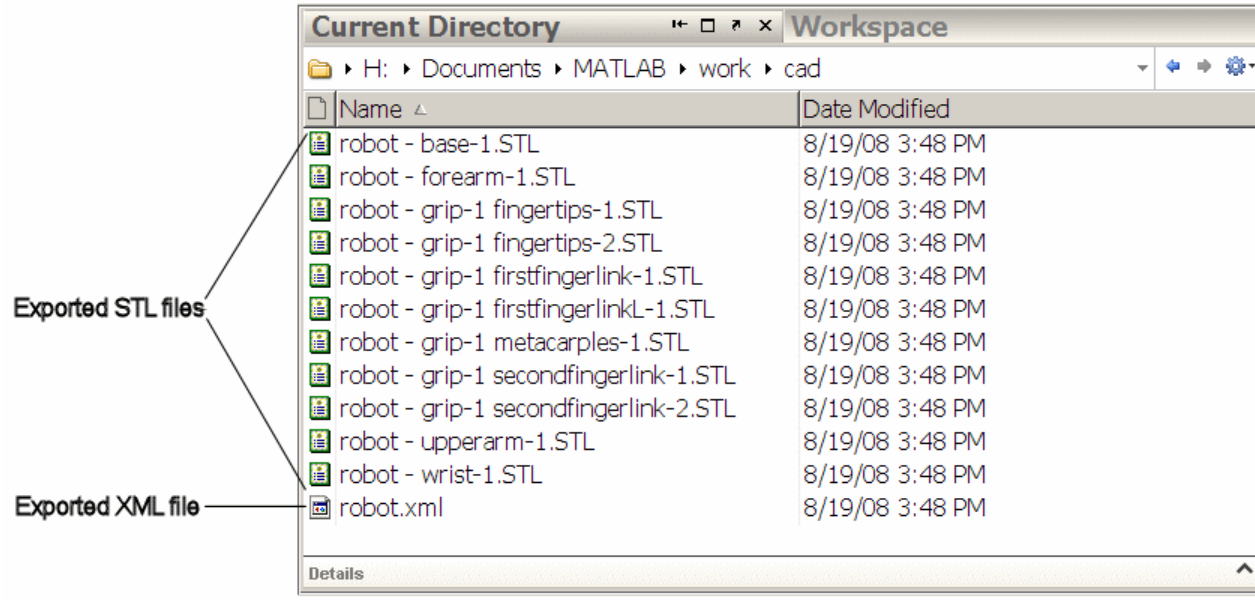
3. From the CAD platform's SimMechanics Link menu, open the export interface.

4. Change to a different folder to export the XML file. Leave the file name as the default, `robot.xml`.

5. Start the export. The exporter begins converting and saving the XML file. When the exporter is finished, it displays a message or dialog box similar to this example from SolidWorks:



The XML file is saved to the folder you chose. The exporter also automatically writes the stereolithographic (STL) body surface geometry files to the folder you choose.



Exported Robot Arm XML and STL Files Viewed in MATLAB® Desktop on Windows®

Re-Exporting a Robot Arm Assembly After Changes

In this example, you re-export the robot arm CAD assembly after making some changes. You save the new version of the XML file under a new name, but overwrite the old stereolithographic (STL) files.

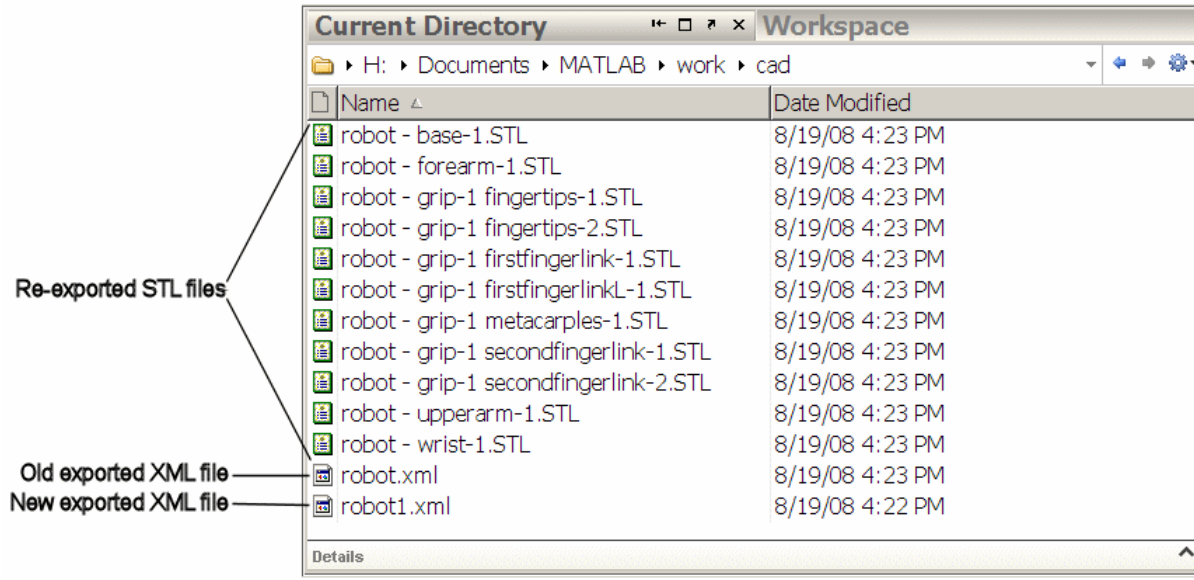
- These changes can include modifications of the assembly or subassembly parts.
- They can also include modifications to the export settings. (See "Configuring the Exporter Controls in Supported CAD Platforms" on page 2-11.)

With the assembly open from your CAD platform interface:

1. Open the export interface from the SimMechanics Link menu. Remain in the same folder in which you saved the original exported files.
2. Change the exported file name to `robot1.xml`.

3 Start the export.

The new XML file, with the new name, is saved in the same folder as the original export. The exporter overwrites the old stereolithographic files with new ones, named with the same names.



Re-Exported Robot Arm XML and STL Files Viewed in MATLAB® Desktop on Windows®

Retranslating from Assembly to Existing Generated Model

In this section...

About Associativity and Updating

To update an existing generated SimMechanics model with changes to its original external definition (a CAD assembly, for example), the intermediate Physical Modeling XML file and the model itself must retain information about the identities of at least some of its components. This section explains this "identity memory" or *associativity*.

What Is Associativity?

Associativity is a key concept for understanding the relationship between CAD assemblies and SimMechanics models based on them, and the export and updating process that defines SimMechanics models from CAD assemblies.

Associativity Between CAD Assemblies and SimMechanics Models

Associativity is a persistent (session-independent) parallel relationship among certain components of a CAD assembly, Physical Modeling XML files exported from it, and SimMechanics models generated from the XML files.

This relationship preserves the identities and parallelisms of certain CAD components and the corresponding imported components of the SimMechanics model. The SimMechanics Link exporter defines these unique identities from the CAD assembly components and embeds them in the exported Physical Modeling XML file. SimMechanics models generated from the XML file in turn retain these identities.

You actualize associativity when you generate a SimMechanics model from a CAD assembly. Associativity is a mapping between parts, constraints, and subassemblies in a CAD assembly and the corresponding Body and Joint blocks, coordinate systems, and subsystems in the SimMechanics model generated from that CAD assembly. It uniquely captures the identities of

these CAD components, their corresponding blocks, and their topology (how they are connected to one another).

Associativity is not completely symmetric between the CAD and SimMechanics worlds, because the translation process moves in one direction only, from CAD assembly to generated SimMechanics model.

When and Why You Need Associativity

Associativity is required for updating a generated SimMechanics model when its originating CAD assembly has been changed.

How Associativity Is Implemented

When you use the SimMechanics Link exporter to create a Physical Modeling XML file from a CAD assembly, these components receive unique XML identifiers. When you use the SimMechanics importer to generate a SimMechanics model from the XML file, the identifiers are preserved in the corresponding SimMechanics model features.

Parallel Identities Between CAD Assembly and SimMechanics Model Components Captured by Associativity

CAD Assembly Components	Corresponding Imported SimMechanics Model Components
Parts and grounds	Body and Ground blocks
Constraints between parts (allowed motions)	Joint blocks
Constraints between parts (positions and orientations)	Paired coordinate systems attached to Joints
Reference coordinate systems	Body coordinate systems unattached after import
Subassembly hierarchy	Subsystem hierarchy

Changing Assemblies, Generated Models, and Their Associativity

The associativity of CAD assembly and generated SimMechanics model is open, modifiable, and extensible. As long as a generated SimMechanics model

retains at least one associated imported component, this model retains some associativity with its originating CAD assembly.

Preserving associativity. You preserve the original associativity if you do not remove or reconnect associated components in either the CAD assembly or the SimMechanics model.

Changing the properties of an associated component, without removing or reconnecting it, both uses and preserves associativity.

Extending associativity. You extend the original associativity if you add new, associable components to the CAD assembly, export the assembly, and update-import the generated SimMechanics model. The new components generated in the updated SimMechanics model are associated with the new components of the CAD assembly.

Modifying associativity. You modify the original associativity if you remove or reconnect one or more associated components in the SimMechanics *model*.

- The associativity of the removed or reconnected associated components is destroyed.
- The associativity of the other associated components, and of the SimMechanics model as a whole, remains intact.
- You recreate the original associativity of the removed or reconnected components in the SimMechanics model if you reimport the unchanged components from the CAD assembly.

Replacing associativity. You replace the original associativity if you remove or reconnect one or more associated components in the CAD *assembly*.

Once you export the CAD assembly and update-import the SimMechanics model, the associativity of the removed or reconnected components is destroyed. In this case, the component is either connected in a new way, with a new associativity, or it is removed altogether.

Working with Associativity in Common Updating Situations

The unique parallel identities created by associativity allow you to revise and expand CAD assemblies, then export the changed CAD assemblies and update existing SimMechanics models based on them. While you can also create entirely new SimMechanics models from the updated XML, associativity saves the effort invested in editing and testing by reusing existing SimMechanics models.

The following translation cases cover the basic possibilities. You can combine some of them into more complex, compound cases. For example, you can change a CAD assembly by both revising existing component properties and adding new components.

Exporting a CAD Assembly and Generating a SimMechanics Model for the First Time

When you first export a CAD assembly to create the Physical Modeling XML file, new and unique XML identifiers tag the assembly components listed in the first column of the table, Parallel Identities Between CAD Assembly and SimMechanics™ Model Components Captured by Associativity on page 2-19. When you import the XML file and generate a SimMechanics model from it, the corresponding model components listed in the preceding table's second column receive these parallel identities.

Updating a Generated SimMechanics Model by Modifying CAD Assembly Properties

If you modify a CAD assembly and export a new Physical Modeling XML file from it, updating the model with the modifications allows you to reuse an existing SimMechanics model that was previously translated from the same assembly.

Modifying a CAD assembly means changing the properties of its associated components of the types listed in the first column of the table, Parallel Identities Between CAD Assembly and SimMechanics™ Model Components Captured by Associativity on page 2-19, without changing the identity of the components. Updating the existing translated SimMechanics model by update-import means that the existence and identity of the corresponding model components are not changed, only their properties.

Associativity identifies the components in the existing generated SimMechanics model so that the importer can update their properties.

Updating a Generated SimMechanics Model by Extending the CAD Assembly

If you add more components to a CAD assembly and export a new Physical Modeling XML file from it, updating the model with the extensions allows you to reuse an existing SimMechanics model previously translated from the same CAD assembly.

Extending a CAD assembly means adding associated components listed in the first column of the table, Parallel Identities Between CAD Assembly and SimMechanics™ Model Components Captured by Associativity on page 2-19. Updating the existing translated SimMechanics model by update-import means extending the machine corresponding to the CAD assembly with new model components corresponding to the new assembly components. The update does not disturb the existence and identity of the SimMechanics model components corresponding to the original CAD assembly components.

Associativity identifies the original components in the existing generated SimMechanics model so that the importer does not change them while adding the new associated components.

Modifying a Generated SimMechanics Model Associated with a CAD Assembly, Then Updating Its Associated Components

You can also manually add nonassociated components to an existing SimMechanics model previously generated from a CAD assembly, separately revise the assembly, then retranslate the assembly by update-importing the SimMechanics model with the revisions.

- The associated SimMechanics model components are updated with the CAD assembly revisions.
- The nonassociated SimMechanics model components are not unchanged.
- If the nonassociated SimMechanics model components are connected in the original model to associated blocks, they might become disconnected after update-import.

- Nonassociated model components can include Constraints, Drivers, Actuators, and Sensors that you manually added and connected to associated, imported Bodies and Joints.
- Nonassociated model components can also include Bodies and Joints added manually after you generated the original SimMechanics model. These Bodies and Joints were not import-generated and therefore cannot be associated.

Troubleshooting Export Problems

In this section...

Troubleshooting Assembly Export Failures

The SimMechanics Link exporter can encounter difficulties when it attempts to represent your assembly in the Physical Modeling XML file as SimMechanics bodies and joints. This section describes some of these problems and how to obtain online help while you work with export.

Constraint Export Errors

Constraint export errors occur when you specify a constraint in your CAD assembly that is not supported for export. Constraints supported for a specific CAD platform are listed in the *SimMechanics Link Reference*.

If the exporter fails to translate one or more constraints into joints, the exporter issues one or more error dialog boxes and logs the errors into a text file. The error dialog box indicates the name of the error log file, which is located in the same folder as the exported XML file. The XML file is generated regardless of constraint export errors.

The XML file itself contains the same errors, each paired with the corresponding failed joint. If you generate a SimMechanics model from the XML file, these errors reappear as MATLAB command line warnings.

Fixing Constraint Export Failures

In your generated model, Joints imported from failed constraint translation appear as rigid Welds. You can fix such errors in two ways:

- Reconfigure the constraints in your CAD assembly, and export it again.
- In the generated model, manually replace these Welds with the proper moving Joints.

Tip If you need reference coordinate systems on Bodies to facilitate repairing mistranslated Joints after import, see "Preparing a CAD Assembly for Translation into a SimMechanics Model" on page 2-7 and "Configuring the Exporter Controls in Supported CAD Platforms" on page 2-11.

Getting Exporter Help from the CAD Platform Interface

Each CAD platform's SimMechanics Link interface has a SimMechanics Link menu with a **Help** option. Selecting this menu item opens the online SimMechanics Link help in MATLAB.

3

Computer-Aided Design Translation

Using realistic examples, these case studies illustrate how to translate mechanical systems that are defined externally, as computer-aided design (CAD) assemblies, into SimMechanics models.

About the CAD Translation Case Studies

In this section...

Introducing the Case Studies

These case studies illustrate converting computer-aided design (CAD) assemblies, through SimMechanics Link export, into SimMechanics models. Each study presents an example or set of examples based on a specific machine type represented in CAD. The SimMechanics models in each study are generated with default settings.

- "Translating a CAD Part into a Body" on page 3-5 presents the simplest case, translating an assembly with a single part or body.
- "Translating CAD Constraints into Joints" on page 3-9 describes how you translate assemblies with constrained parts into bodies with degrees of freedom represented by SimMechanics Bodies and Joints.
- "Updating and Retranslating a CAD Pendulum" on page 3-22 presents an example of translating a CAD assembly into a SimMechanics model, modifying the original assembly in several ways, then retranslating it into several updated models.

  Note This study requires Pro/ENGINEER.

- "Translating a CAD Robot Arm" on page 3-42 illustrates the translation of an assembly representing a simple mechanism, including subassembly-subsystem hierarchy. This study also introduces post-translation additions to the model.
- "Translating a CAD Stewart Platform" on page 3-49 presents the translation of a moderately complex assembly with many degrees of freedom, subassembly-subsystem hierarchy, and visualization of its motion.

Note This study is not supported for Autodesk Inventor.

Requirements for the CAD Translation Case Studies

The following assembly and exporting examples require a CAD platform supported by the SimMechanics Link utility. To complete all the steps, you also need SimMechanics Link and SimMechanics software.

The demo CAD files are under `$matlabroot/toolbox/physmod/smlink/smlinkdemos/`, in the subfolder appropriate to your supported CAD platform. The file extensions of these files are specific to each CAD platform.

The assembly, geometric, kinematic, and part details differ from platform to platform. In SolidWorks, constraints on CAD parts are called *mates*.

For More About CAD Assembly Export

For an overview of export procedures and concepts, see Chapter 2, "Getting Started with Export".

For More About SimMechanics Import and Model Generation

Note Mechanical import and model generation require SimMechanics software and a previously exported Physical Modeling XML file.

For complete information about mechanical import and model generation and editing, see the *SimMechanics Visualization and Import Guide*.

For More About Simulink and SimMechanics Blocks and Functions

For more information about SimMechanics software, consult the block and command reference of the SimMechanics documentation.

For more information about Simulink software, consult the block and function reference of the Simulink documentation.

About Specialized CAD Terms

For definitions of CAD terminology, see the glossary of the SimMechanics documentation.

Translating a CAD Part into a Body

In this section...

Locating the Single-Part Assembly Files

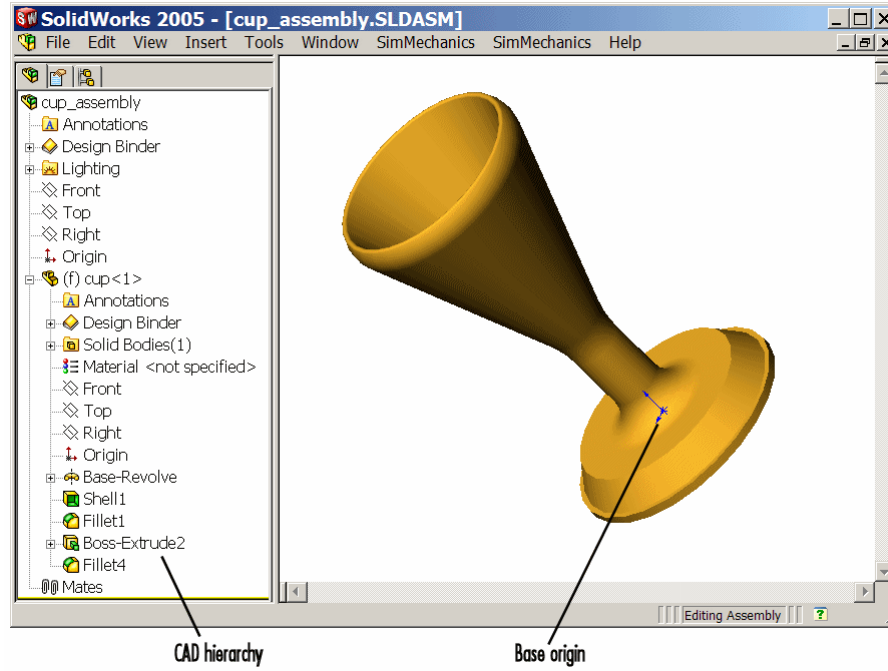
In this example, you export an assembly with one part and no constraints. Look for the following two example CAD files in the SimMechanics Link demos folder:

- The full assembly file, `cup_assembly.`*`ASSEMBLYFILETYPE`*
- The part, a cup, in a file called `cup.`*`PARTFILETYPE`*

Although it has only one part, you must export the full assembly into XML, not just the cup part.

Viewing the CAD Assembly

Open the cup assembly file in your CAD platform and check its geometry and mass properties.



Cup Assembly in a CAD Platform

Property	Value
Volume	0.0001 cubic meters (m^3)
Surface area	0.0381 square meters (m^2)
Density	3.0 grams/cm^3 = 3000 kg/m^3
Mass	0.2906 kilograms (kg)
Principal moments of inertia at the center of gravity	I_x = 0.00015, I_y = 0.00067, I_z = 0.00067 kg-m^2

The inertia tensor is computed at the center of gravity, with the coordinate axes aligned with assembly base-origin axes, indicated in the figure, Cup Assembly in a CAD Platform on page 3-6. The *x*-axis is the cup's axis of symmetry, and the *y*- and *z*-axes point across the cup.

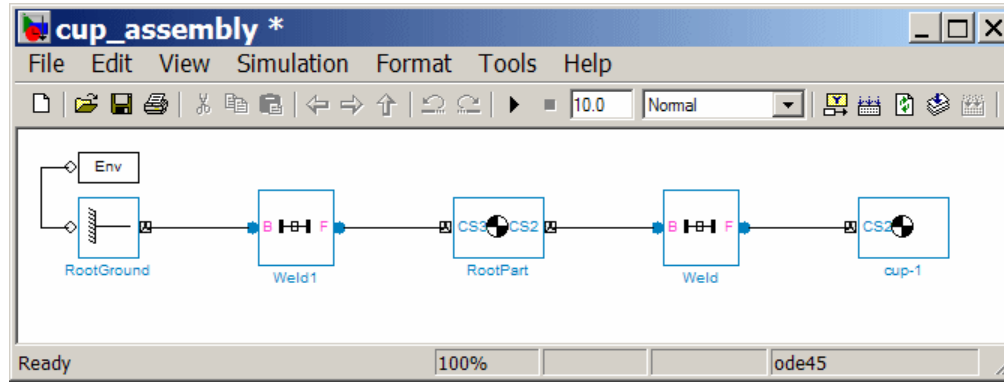
Exporting the CAD Assembly

Using the SimMechanics Link interface to your CAD platform, export the assembly into Physical Modeling XML format. The XML file `cup_assembly.xml` appears in your working CAD folder.

Generating the SimMechanics Model

To import the Physical Modeling XML with the SimMechanics importer:

1. Move or copy the exported XML file into a MATLAB working folder to generate a SimMechanics model from the file.
2. Generate the model from `cup_assembly.xml` with the `mech_import` command.



Once you generate the SimMechanics model, it has six blocks, a combination representing the entire assembly:

Machine Environment – Root Ground – Weld – Root Part – Weld – Cup

- The Ground origin is coincident with the World origin and maps the CAD assembly origin.
- The Root Part is a nondynamical zero-mass/zero-inertia body inserted between ground and the cup.
- The second joint is a Weld because the original CAD assembly has no degrees of freedom.

Deleting the Root Part and one of the Welds does not physically change the model, as long as you reconnect the remaining blocks.

Translating CAD Constraints into Joints

In this section...

Modeling CAD and SimMechanics Degrees of Freedom

In "Translating a CAD Part into a Body" on page 3-5, you create and export an assembly composed of a single part. Because there are no other parts in that CAD assembly, the SimMechanics body is welded to ground and has no degrees of freedom (DoFs). This lack of DoFs is not realistic for most assemblies.

This study presents a set of complete CAD assemblies with both parts and constraints and possessing DoFs. Each example assembly consists of two instances of the same part file, representing two identical cubes. The study shows how to find the required files, presents the essential steps for generating models from them, and describes the structure common to all the generated models. It ends with specific assembly cases that include two cubes:

- With no constraints, so that the cubes have the full six degrees of freedom relative to one another
- Constrained in two different ways so as to produce the same result, a single prismatic (translational) DoF between them

- Constrained so as to allow only a single revolute (rotational) DoF between them
- Constrained so as to allow two prismatic (translational) DoFs between them
- Constrained so as to allow relative spherical joint motion, with the two cubes separated by a constant nonzero distance

Different Constraint Combinations Can Represent the Same Degrees of Freedom

In different assemblies, the two cubes are constrained with different constraint combinations to create different relative DoFs between the cubes. In most cases, you can represent one set of DoFs with a large number of different combinations of constraints.

Locating the Constraint Assembly Files

Look for the CAD assembly files of this study in the SimMechanics Link demos folder. The assemblies have the generic name, `<assembly-name>.ASSEMBLYFILETYPE`. The cube part is in `magic_cube.PARTFILETYPE`.

Assembly Name	Assembly Configuration
`sixDOF`	Two cubes with no constraints
`prismatic1`	Two cubes with planar and cylindrical constraints
`prismatic2`	Two cubes with planar constraints
`revolute`	Two cubes with planar and cylindrical constraints
`inplane`	Two cubes with planar constraints
`spherical_spherical_ massless_connector`	Two cubes with a distance constraint

Generating the Two-Part Models: Common Steps

The procedure for exporting a two-part assembly and generating SimMechanics models based on it is essentially the same for all the examples of this study.

Viewing and Exporting an Assembly

To see a two-part assembly and export it into Physical Modeling XML:

1. Open the assembly, `<assembly-name>.ASSEMBLYFILETYPE`. The two parts are `magic_cube-1` and `magic_cube-2`.

   Locate any constraints imposed on the parts. These constraints define the relative DoFs between the parts.

2. Using the SimMechanics Link interface to your CAD platform, export the CAD assembly into Physical Modeling XML. The XML file is saved in your current working CAD folder.

Generating a Model

You can generate a SimMechanics model based on the assembly.

1. Move or copy the XML file to a working MATLAB folder. In this folder, open MATLAB.

2. At the command line, enter `mech_import('<assembly_name>')` to automatically generate a model, `<assembly_name>.mdl`, based on `<assembly_name>.xml`.

3. Open the subsystem. The blocks are arranged in the common structure described in “About the Common Block Structure of the Two-Part Models” on page 3-11. A set of Joints represents the DoFs between the two cubes.

About the Common Block Structure of the Two-Part Models

In this study, all the models that you generate each have eight blocks. From the example CAD assemblies, the models have a common structure because each assembly has a fundamental root and two moving parts:

- *The assembly fundamental root.* As in any generated CAD-based model, the four-block combination Machine Environment – Root Ground – Root Weld – Root Part represents the assembly fundamental root. The Root Part is a nonmoving, zero-mass/zero-inertia body.
- *The moving bodies.* The bodies representing the assembly parts are `magic_cube-1` and `magic_cube-2`.
- *The joints.* In all the models, the first cube is connected by a Weld to Root Part and cannot move. The second cube is connected to RootPart by a Joint that represents the appropriate degrees of freedom (DoFs).

  Depending on the DoFs in a particular assembly, CAD translation configures the Joint to represent different DoFs with combinations of prismatic, revolute, and spherical primitives. The second cube can move with respect to the first cube through the DoFs represented by the Joint.

Some of the blocks in the generated models are redundant. You can manually edit and simplify the models without changing their physical properties.

Degrees of Freedom Restricted by Constraints

CAD platforms usually assume that two parts with no constraints between them have the complete six relative DoFs possessed by any rigid body relative to another body. You restrict the DoFs between parts by connecting them with constraints in the CAD assembly. Constraints restrict relative body motion and reduce the number of relative DoFs between body pairs. One assembly part is always welded to ground.

Modeling a Six-DoF Joint

The simplest assembly with two parts has no constraints between the parts. The parts can move with respect to one another with all six degrees of freedom (DoFs).

Exporting the Assembly

To see and export the assembly:

1. Open the assembly, `sixDOF.`*`ASSEMBLYFILETYPE`*.

The cube parts have no constraints. Therefore, relative to one another, the cubes are unconstrained in their motion and have six relative DoFs.

2 Export this CAD assembly into the file, `sixDOF.xml`.

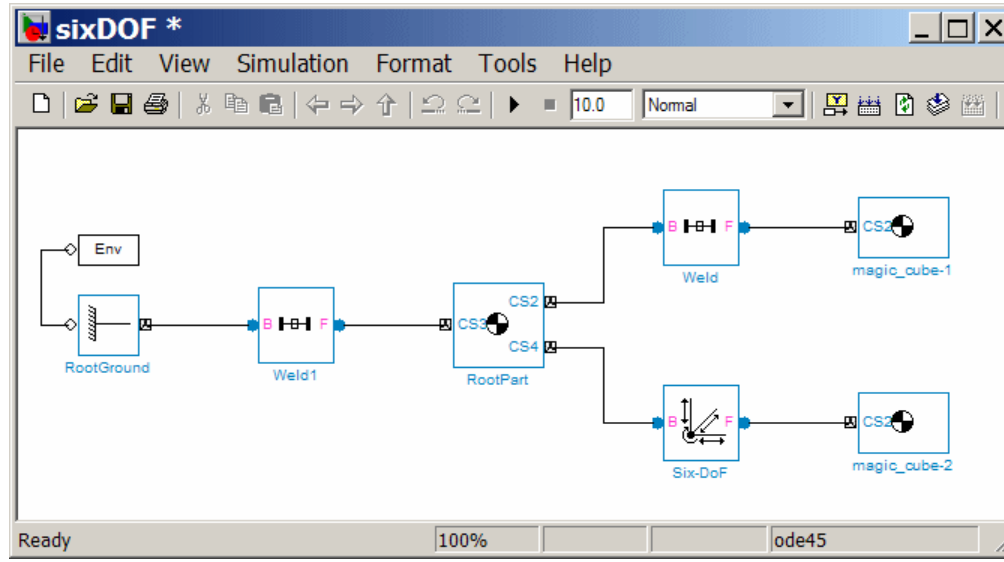
Generating the Model

To generate a model based on the assembly:

1 At the MATLAB command line, enter `mech_import('sixDOF')` to generate a model, `sixDOF.mdl`.

2 Inspect the model. There are eight blocks.

The Six-DoF Joint represents the six DoFs between the two cubes with one spherical and three prismatic primitives.



Modeling a Prismatic Joint

In the following two assemblies, the two cubes are constrained to have only a single translational degree of freedom (DoF) between them. These assemblies illustrate two ways to express one translational degree of freedom.

You can experiment with other constraints to find more. In the translated SimMechanics models, this single DoF is a prismatic joint.

Prismatic as a Cylindrical Constraint and a Planar Constraint

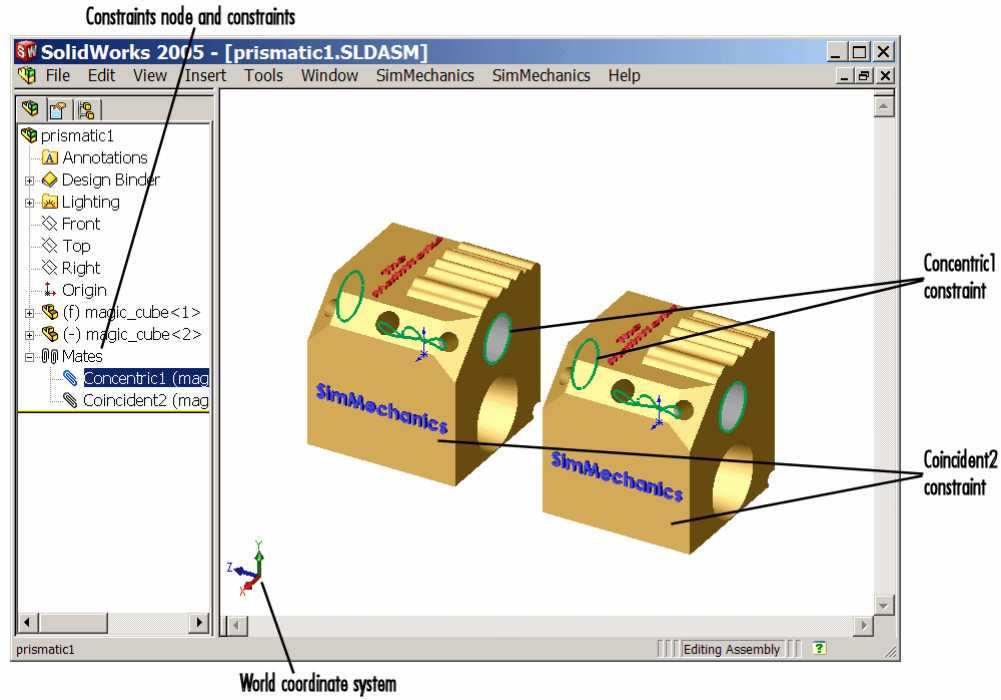
To see the first way of constraining the DoFs to produce a prismatic joint:

1. Open the assembly file, `prismatic1.`*`ASSEMBLYFILETYPE`*, and examine its CAD hierarchy.

2. Locate the two constraints on the two cubes.

   - Highlight the cylindrical constraint in the assembly. This constraint allows the two cubes to only slide along and rotate about the *z*-axis running through the center of the parallel and concentric upper holes of each cube.
   - Highlight the planar constraint in the assembly. This constraint allows the two cubes to slide along the *y*-*z* plane, with the two sides marked "SimMechanics" sharing a common plane, representing two translational DoFs. The constraint also allows the two cubes to rotate about the *x*-axis. The cubes are not allowed to rotate about any other axis, or to translate perpendicular to the *y*-*z* plane.

With these two constraints, the two cubes can only slide along the *z*-axis common to the two upper concentric holes. The second constraint prevents rotation about this axis, leaving the whole assembly with only one translational DoF.



Cylindrical and Planar Constraints on Two Cubes (SolidWorks®)

Prismatic as Two Orthogonal Planar Constraints

To see the second way of constraining the DoFs to produce a prismatic joint:

1. Open the assembly file, `prismatic2.`*`ASSEMBLYFILETYPE`*, and examine its CAD hierarchy.

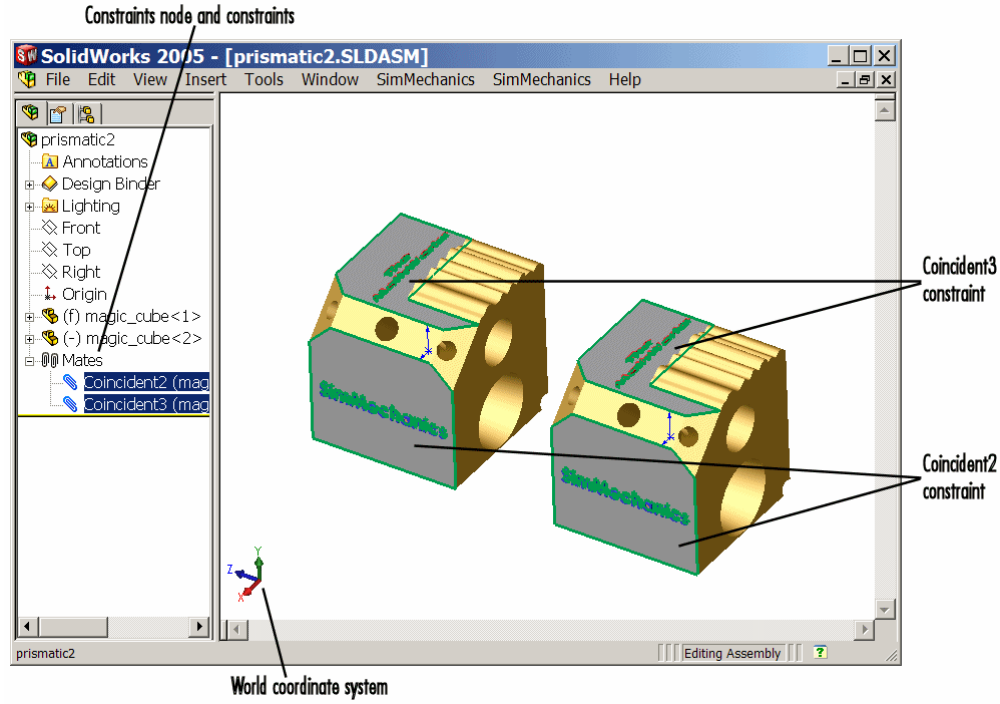
2. Locate the two constraints on the two cubes.

   - Highlight the first planar constraint in the assembly. This constraint allows the two cubes to slide along the *y*-*z* plane, with the two sides marked “SimMechanics” sharing a common plane, representing two

translational DoFs. It also allows the two cubes to rotate about the *x*-axis. The cubes are not allowed to rotate about any other axis, or to translate perpendicular to the *y*-*z* plane.

- Highlight the second planar constraint in the assembly. This constraint allows the two cubes to slide along the *x*-*z* plane, with the two sides marked "The MathWorks" sharing a common plane, representing two translational DoFs. The constraint also allows the two cubes to rotate about the *y*-axis. The cubes are not allowed to rotate about any other axis, or to translate perpendicular to the *x*-*z* plane.

With these two constraints, the two cubes can only slide along the *z*-axis common to the two planes *y*-*z* and *x*-*z*, leaving the whole assembly with only one translational DoF.



Two Planar Constraints on Two Cubes (SolidWorks®)

Exporting the Assemblies and Generating SimMechanics Models

To create models from the assemblies:

1. Export the two assemblies into the XML files, `prismatic1.xml` and `prismatic2.xml`.
2. Copy or move them to a MATLAB working folder. At the MATLAB command line, generate SimMechanics models using `mech_import`.

In both models, the assemblies are translated into block diagrams of eight blocks each. The Prismatic Joint represents the single translational DoF between the two cubes with one prismatic primitive along the *z*-axis.

Modeling a Revolute Joint

In the following assembly, the two cubes are constrained to have only a single rotational degree of freedom (DoF) between them. In the translated SimMechanics model, this single DoF is a revolute joint.

Viewing the Assembly

To see an assembly with one rotational DoF:

1. Open the assembly file, `revolute.`*`ASSEMBLYFILETYPE`*, and examine its CAD hierarchy.
2. Locate the two constraints on the two cubes.
   - Highlight the cylindrical constraint in the assembly. This constraint allows the two cubes to slide along and rotate about the *z*-axis running through the center of the parallel and concentric upper holes of each cube.
   - Highlight the planar constraint in the assembly. This constraint allows the two cubes to slide along the *x*-*y* plane, with the parallel sides sharing a common plane. The constraint also allows the two cubes to rotate about the *z*-axis. The cubes are not allowed to rotate about any other axis, or to translate perpendicular to the *x*-*y* plane.

With these two constraints, the two cubes can only rotate about the *z*-axis orthogonal to the *x*-*y* plane, leaving the whole assembly with only one rotational DoF.

Exporting the Assembly and Generating the Model

To generate a model based on the assembly:

1. Export the assembly as the file, `revolute.xml`. Copy or move it to a MATLAB working folder.
2. At the MATLAB command line, generate a SimMechanics model using `mech_import`.

The assembly is translated into a block diagram of eight blocks. The Revolute Joint represents the single rotational DoF between the two cubes with one revolute primitive about the *z*-axis.

Modeling an Inplane Joint

In the following assembly, the two cubes are constrained to have only two translational degrees of freedom (DoFs) between them. In the translated SimMechanics model, these two DoFs are two prismatic joints.

Viewing the Assembly

To see an assembly with two translational DoFs:

1. Open the assembly file, `inplane.`*`ASSEMBLYFILETYPE`*, and examine its CAD hierarchy.
2. Locate the two constraints on the two cubes.
   - Highlight the first planar constraint in the assembly. This constraint allows the two cubes to slide along the *y*-*z* plane, with the two sides marked “SimMechanics” sharing a common plane. The constraint also allows the two cubes to rotate about the *x*-axis. The cubes are not allowed to rotate about any other axis, or to translate perpendicular to the *y*-*z* plane.
   - Highlight the second planar constraint in the assembly. This constraint allows the two cubes to slide parallel to the *x*-*z* plane, with the two

sides marked “The MathWorks” parallel but not necessarily in the same plane. It also allows the two cubes to translate perpendicular to the *x*-*z* plane and to rotate about the *y*-axis. The cubes are not allowed to rotate about any other axis.

With these two constraint, the two cubes can only slide in the *y*-*z* plane, leaving the whole assembly with only two translational DoFs.

Exporting the Assembly and Generating the Model

To generate a model based on the assembly:

1. Export the assembly as the file, `inplane.xml`. Copy or move it to a MATLAB working folder.
2. At the MATLAB command line, generate a SimMechanics model using `mech_import`.

The assembly is translated into a block diagram of eight blocks. The In-Plane Joint represents the two translational DoFs between the two cubes with two prismatic primitives, along the *y*-axis and the *z*-axis.

Modeling a Spherical-Spherical Massless Connector

In the following assembly, the two cubes are constrained to have six rotational degrees of freedom (DoFs) between them, represented by two spherical primitives. The spherical primitives pivot independently about two pivot points at a fixed relative distance. In the translated SimMechanics model, a spherical-spherical massless connector represents these six DoFs.

Viewing the Assembly

To see an assembly with three rotational DoFs separated from three other rotational DoFs:

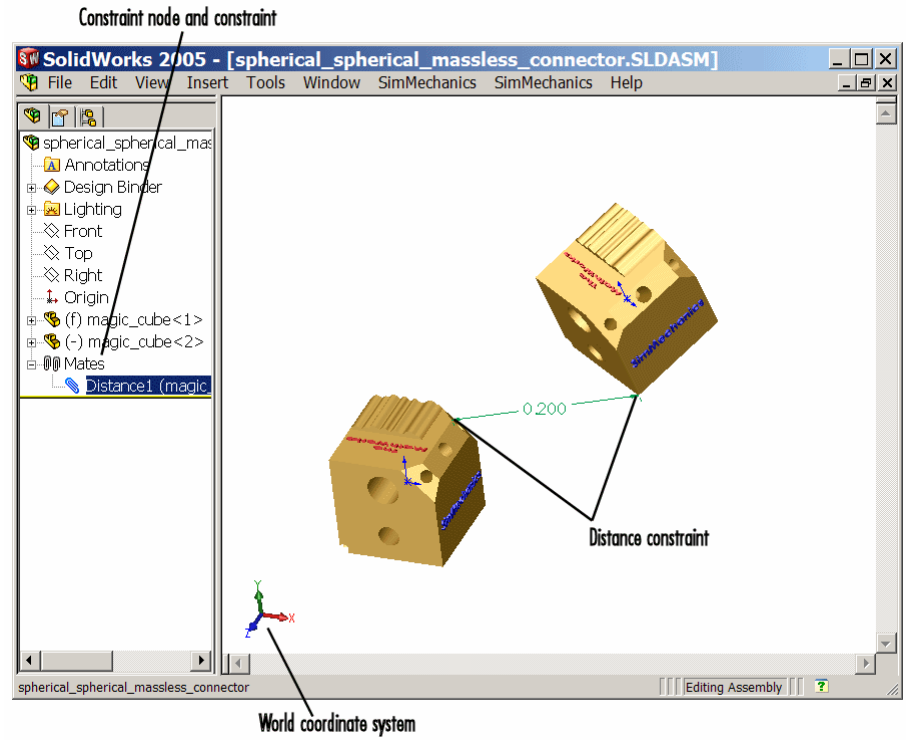
1. Open the assembly file,

   `spherical_spherical_massless_connector.`*`ASSEMBLYFILETYPE`*

   and examine its CAD hierarchy.

2 Locate the one constraint on the two cubes.

Highlight this distance-offset constraint. The two spherical pivot points occur one on each cube and mark the endpoints of the rigid massless connector. The cubes can move so that the distance between these two endpoints (the length of the massless connector) does not change. The constraint allows the two cubes to pivot independently about their connector endpoints.



Distance Constraint on Two Cubes (SolidWorks®)

Exporting the Assembly and Generating the Model

To generate a model based on the assembly:

1. Export the assembly as a file,

   ```
   spherical_spherical_massless_connector.xml
   ```

2. Copy or move it to a MATLAB working folder.

3. At the MATLAB command line, generate a SimMechanics model using `mech_import`.

The assembly is translated into a block diagram of eight blocks, arranged in the common structure described in "About the Common Block Structure of the Two-Part Models" on page 3-11.

The Spherical-Spherical massless connector Joint block represents the two spherical primitives, each with three rotational DoFs, independently pivoting at each end of the massless, rigid connector connecting the two cubes.

Updating and Retranslating a CAD Pendulum

In this section...

About Assembly Re-Export and Model Update

Note The CAD assembly files of this case study require Pro/ENGINEER.

The following example shows how to update a previously generated SimMechanics model with a new XML file exported after changes have been made to the original CAD assembly. The example starts with a double pendulum assembly.

For a general discussion of retranslation and update-import, see "Retranslating from Assembly to Existing Generated Model" on page 2-18.

Locating the Assembly Files

Look for the following seven CAD files of this case study in the SimMechanics Link demos folder.

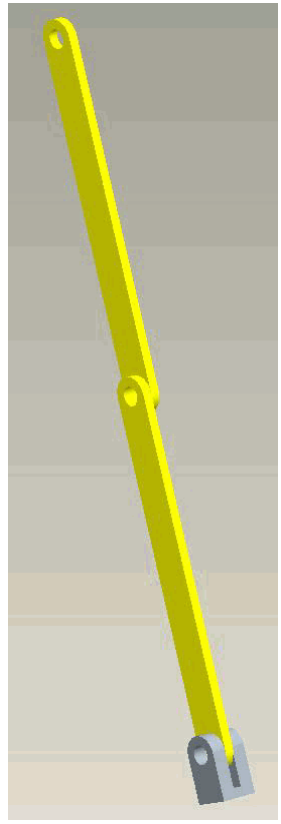
File Name	CAD File Type
`dpen.`*`ASSEMBLYFILETYPE`*	Assembly
`tpen.`*`ASSEMBLYFILETYPE`*	Assembly

File Name	CAD File Type
hook.*ASSEMBLYFILETYPE*	Subassembly
mid_part.*PARTFILETYPE* pend.*PARTFILETYPE*	Parts (main assembly)
hook.*PARTFILETYPE* hookbase.*PARTFILETYPE*	Parts (subassembly)

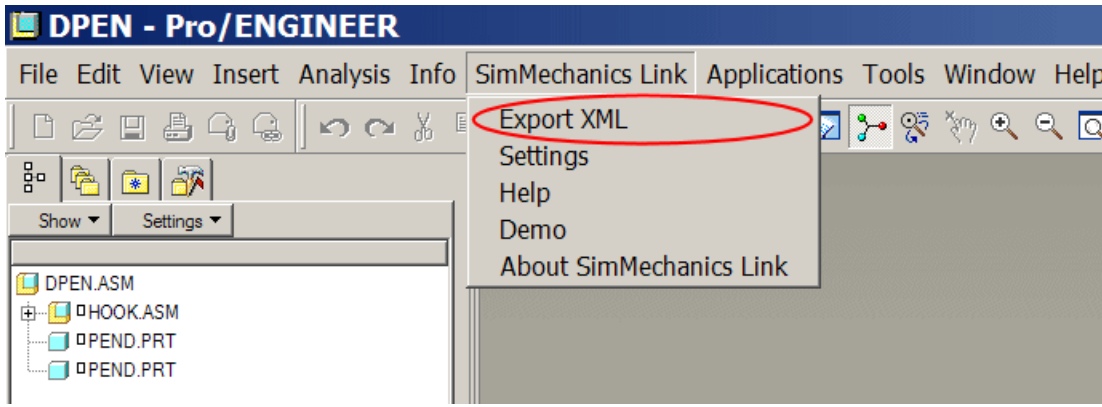
Translating the Assembly For the First Time

Exporting the Assembly for the First Time

Open the dpen assembly file and export it into an XML file.



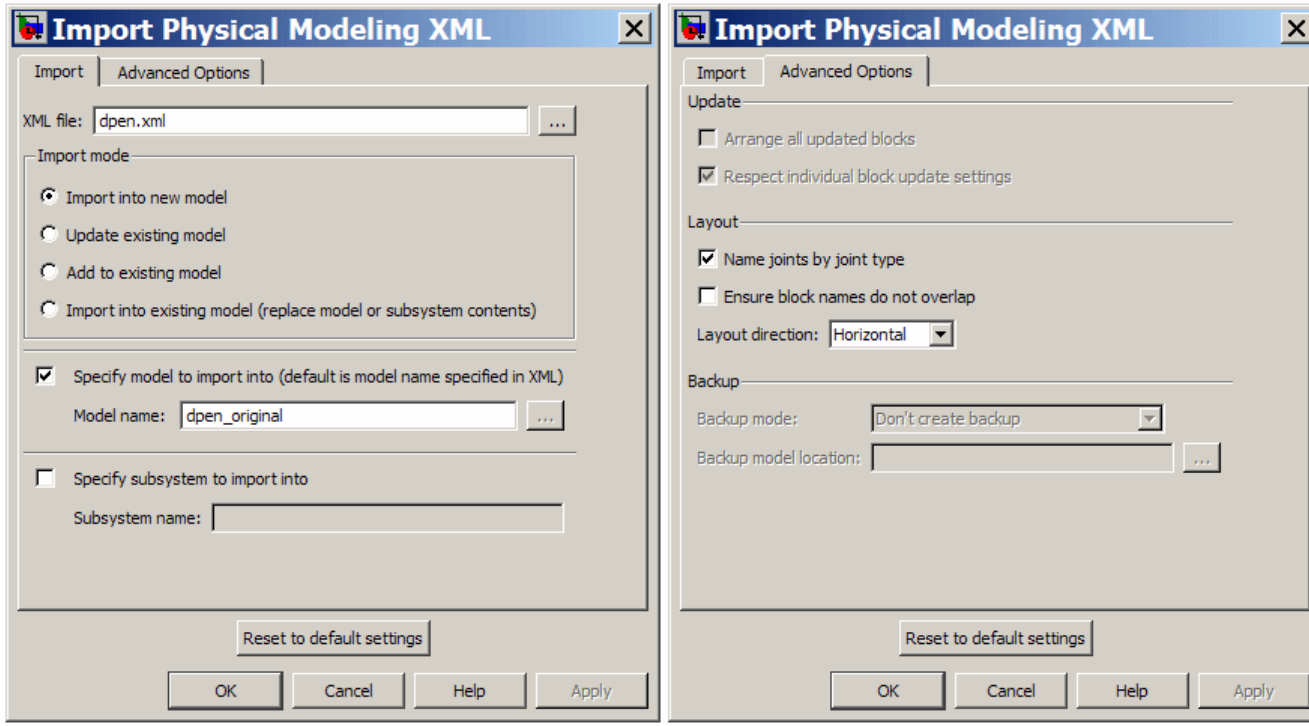
Export the assembly through the **SimMechanics Link** menu of the CAD platform's menu bar. The result is an XML file called `dpen.xml`.



Importing the Assembly and Generating the Initial Model

To import the XML file and generate the SimMechanics model, you can either use the import dialog box or work with the full command at the MATLAB command line.

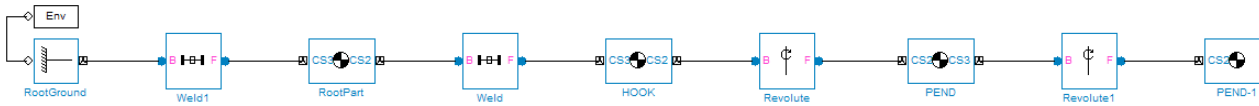
Open the dialog box by entering `mech_import` at the command line.



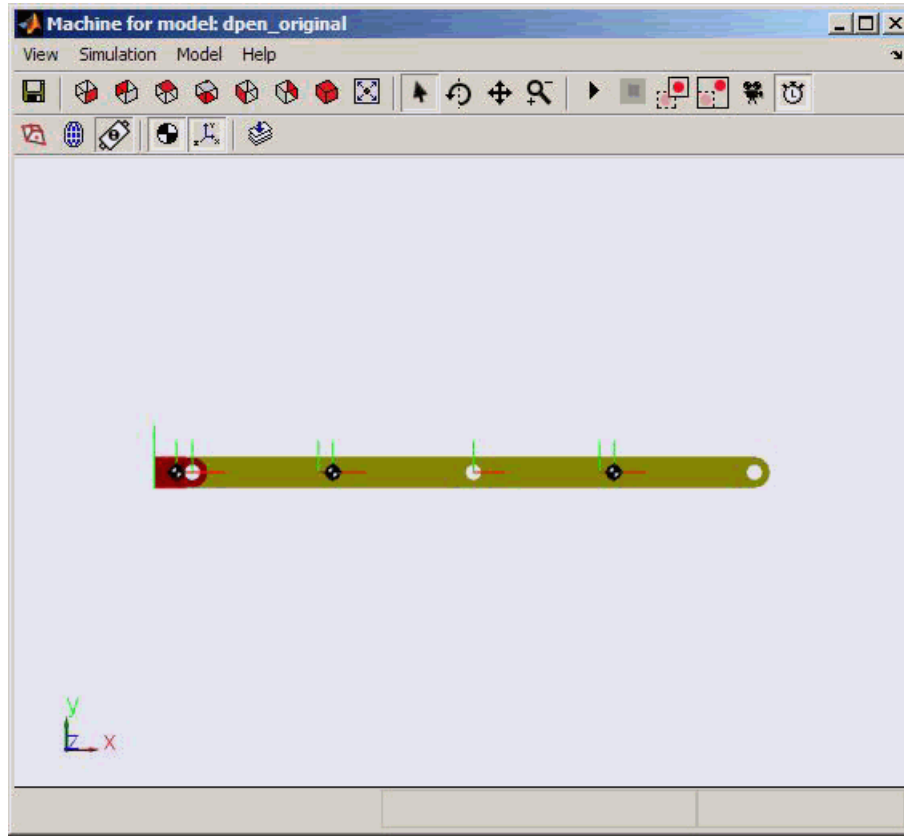
The alternative is to use the full command with specified import options:

```
mech_import('dpen.xml','ImportMode',0,...
        'ModelToImportInto','dpen_original');
```

This model results from the import step. The blocks are created at the top level.



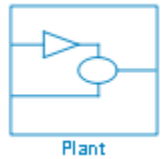
With visualization enabled, you see this SimMechanics model with custom bodies defined by STL files exported when the XML file was exported.



Importing the Assembly into a Plant Subsystem

If you want to import the assembly into a model subsystem, rather than at the top level of a model, use the `SubsystemToImportInto` option of the `mech_import` command. (You can also use the parallel options in the import dialog box.) For example, you can import the assembly into a subsystem named Plant.

```
mech_import('dpen.xml','ImportMode',0,...
        'ModelToImportInto','dpen_original',...
        'SubsystemToImportInto','Plant');
```



Updating the Original Imported Model with Changes to Bodies

You can change part properties in the original assembly and reimport the assembly, with those changes, to update the model. You can also make direct changes by hand to the original generated model and preserve those changes even as you update it with assembly changes.

In this example, you now cut out material from the links to make them lighter. You also connect joint and body sensors in the SimMechanics model to implement a controller. Now you want to update the SimMechanics model with changes from the assembly without undoing the additions you made by hand.

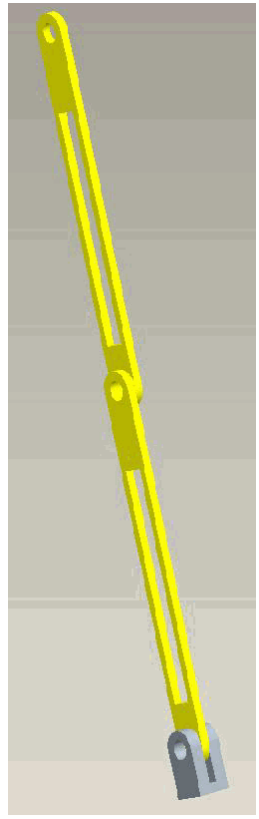
Exporting the Modified Assembly

Change the double pendulum assembly and generate a new XML file from it.

1. Open the `pend` part file and unsuppress the Extrude2 feature to enable the hole in the parts. Do this by right-clicking Extrude2 in the tree view and selecting **Resume**.

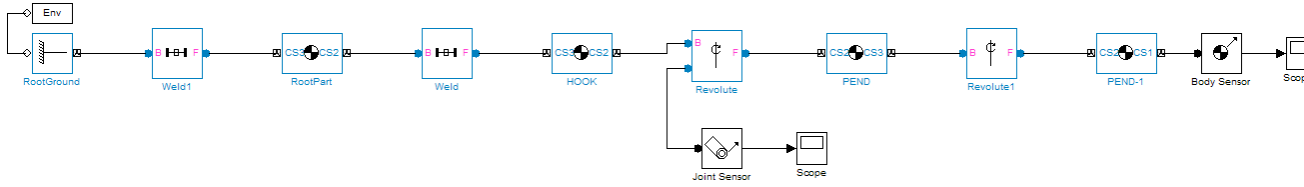
   Save the part file.

2. Open and re-export the `dpen` assembly file to obtain the new XML file, which is assumed to be named `dpen_with_holes.xml`.

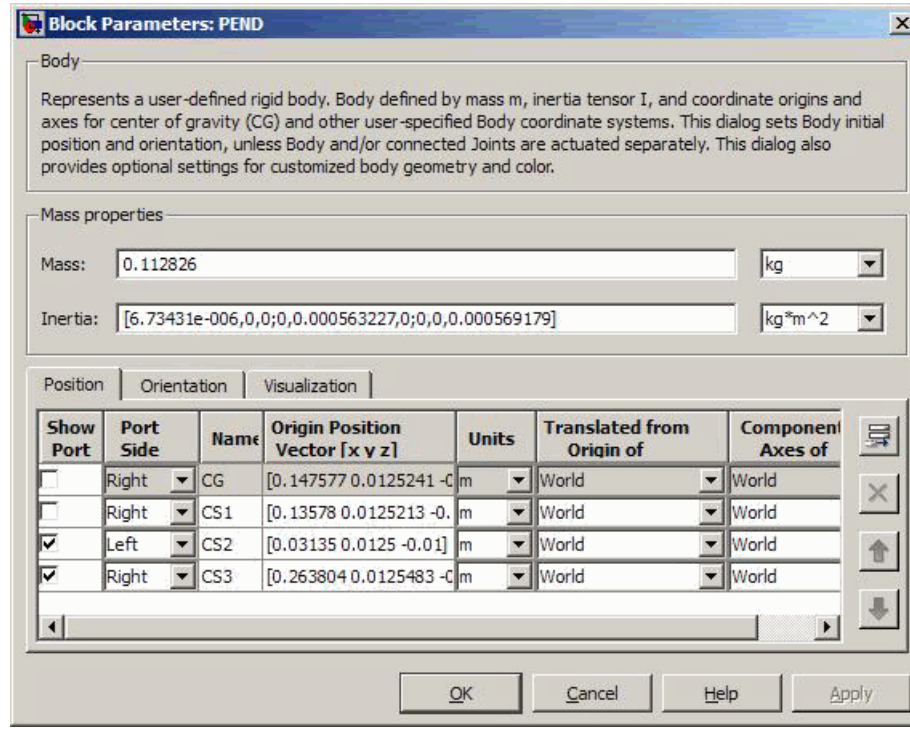


Non-CAD Modifications to the Original Generated Model

In this example, you have separately made the following changes, by hand, to the original generated model, now renamed dpen_withsensors. These changes include insertion of a Joint Sensor, a Body Sensor, and two Scope blocks.

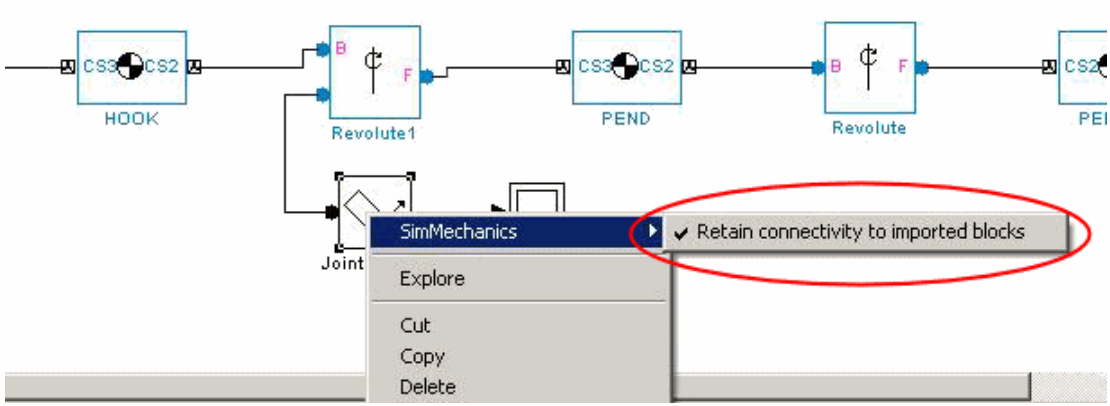


These are the properties of the PEND body before the update-import. PEND-1 has the same mass properties.



Preparing for Update-Import

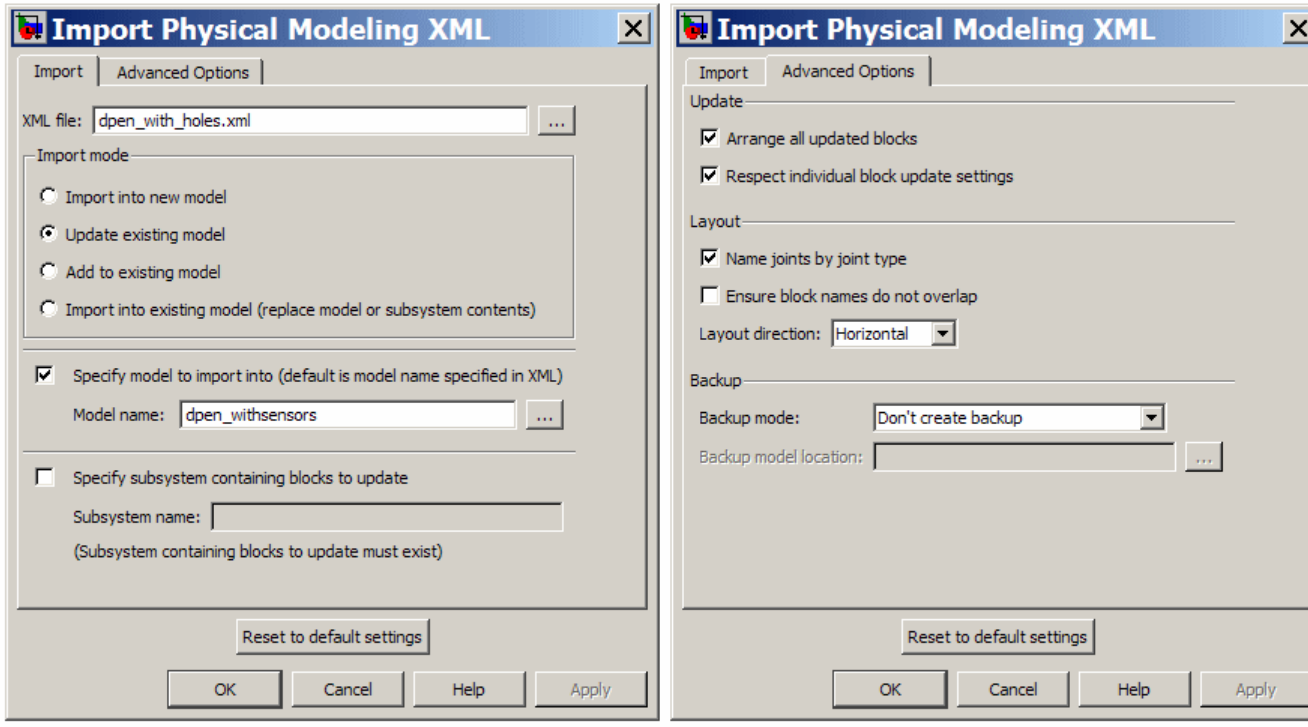
Before proceeding to the update-import from the updated CAD assembly, designate the manually added SimMechanics blocks that you want to retain and keep connected to the originally imported blocks. Right-click the sensor blocks and select the **Retain connectivity to imported blocks** option under the **SimMechanics** submenu. This option is selected by default.



Importing the Modified Assembly and Updating the Generated Model

Now update-import in such a way as to modify the PEND and PEND-1 body mass properties, without changing the manually added blocks and connections. You can either use the import dialog box or the command line with fully specified options to update the existing model. This example uses the dialog box, opened by entering `mech_import` at the command line.

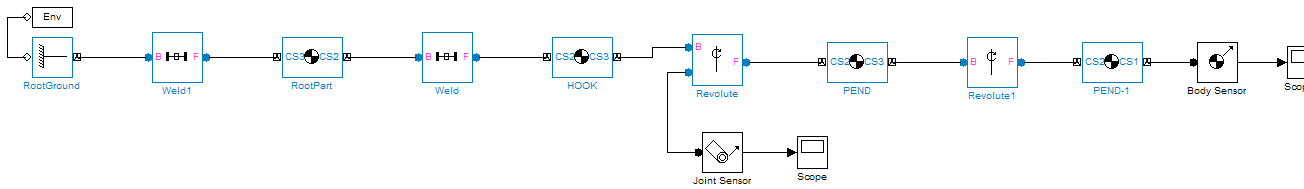
The dialog box is configured so that `dpen_with_holes.xml` is the file imported to update the modified dpen_withsensors model. Because the dialog box is set to update an existing model, the **Update** and **Backup** options in the **Advanced Options** tab are now enabled.



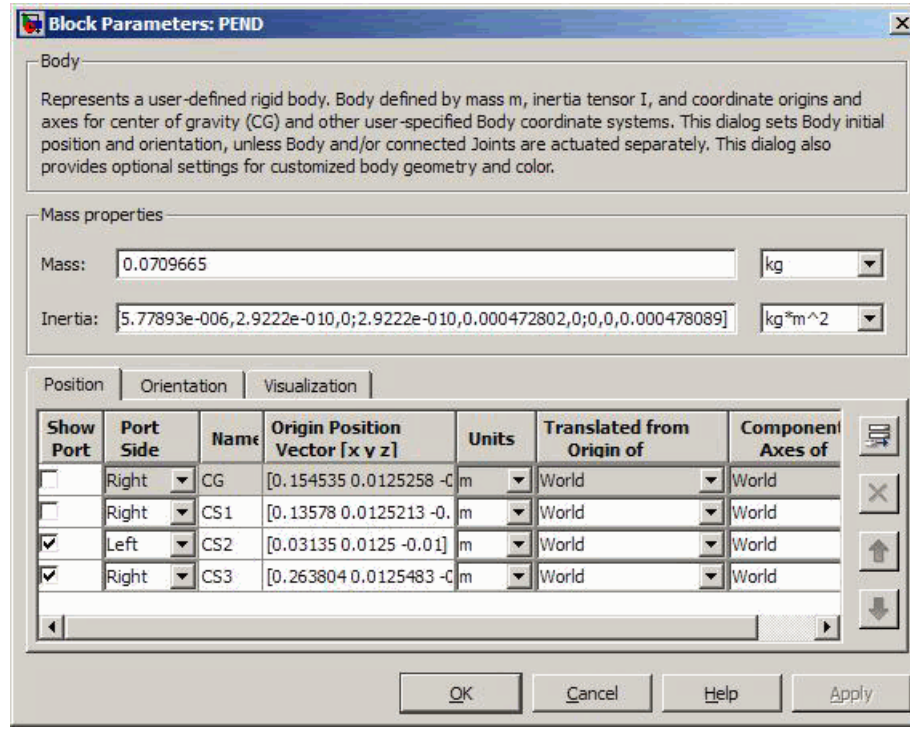
Alternatively, if you want to use the command with fully specified options, enter:

```
mech_import('dpen_with_holes','ImportMode',1,...
        'ModelToImportInto','dpen_withsensors','LayoutWithUpdate',1,...
        'EnableIndvlBlkUpdCtrl',1,'BackupMode',0)
```

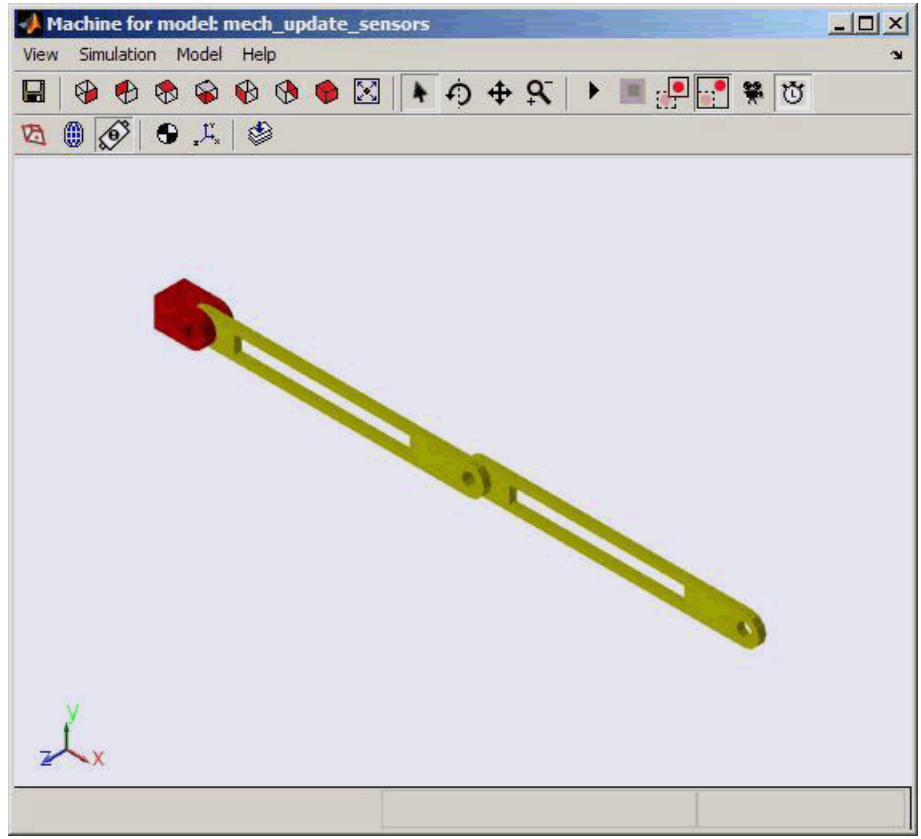
The updated model looks like the following. The sensors and Scopes are retained.



However, the PEND body properties are now different, because of the new holes in the parts. The PEND-1 mass properties are changed in the same way.



The updated bodies, displayed in SimMechanics visualization, appear differently as well, as determined by the new STL files exported from the updated assembly at the same time the new XML file was created.



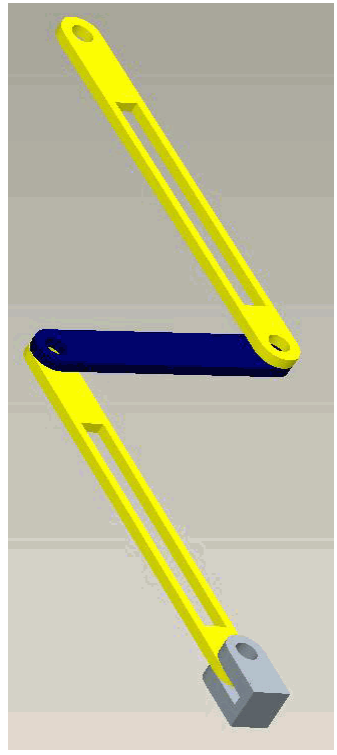
Adding a New Body to Create a Triple Pendulum

The changes to an assembly that you can re-export are not limited to changes in existing parts (bodies). You can add or remove parts, re-export, then update-import an older version of the generated model with the changes.

In this example, in addition to the holes in the parts you previously made, you change the original assembly further by adding a new part (link) between the two original pendulum links, making the assembly a triple pendulum. This addition changes the topology of the assembly (the assembly's connectivity), beyond the property changes to the individual links.

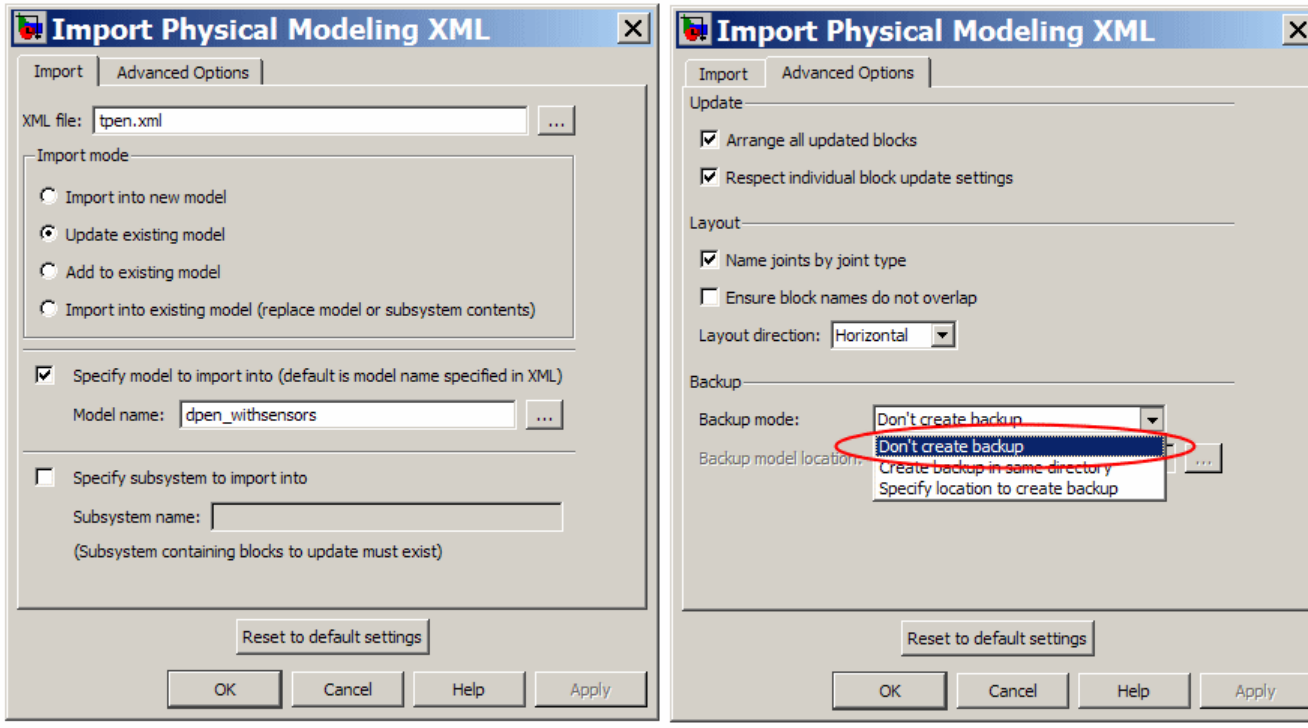
Modifying and Re-exporting the Assembly

To see such a changed assembly, open the `tpen` assembly file. The `tpen` assembly is a copy of the `dpen` assembly with a link inserted. Export a new XML file, which is named `tpen.xml`.



Updating the Existing Generated Model

Update the existing dpen_withsensors model with the changes to the assembly. These are the required options in the import dialog box.



The **Advanced Options** tab shows important options selected for this example.

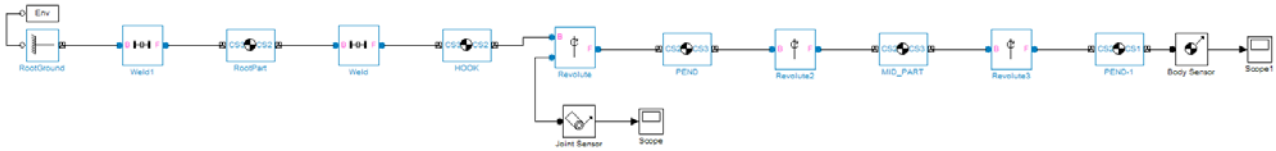
Advanced Option	Effect of Option
Arrange all updated blocks	Selecting this option causes the importer to rearrange all the imported and updated blocks. If cleared, the importer does not alter the layout of the existing blocks, whether updated or not, and places the newly imported blocks around the bounding box for the already existing blocks. In many cases, this

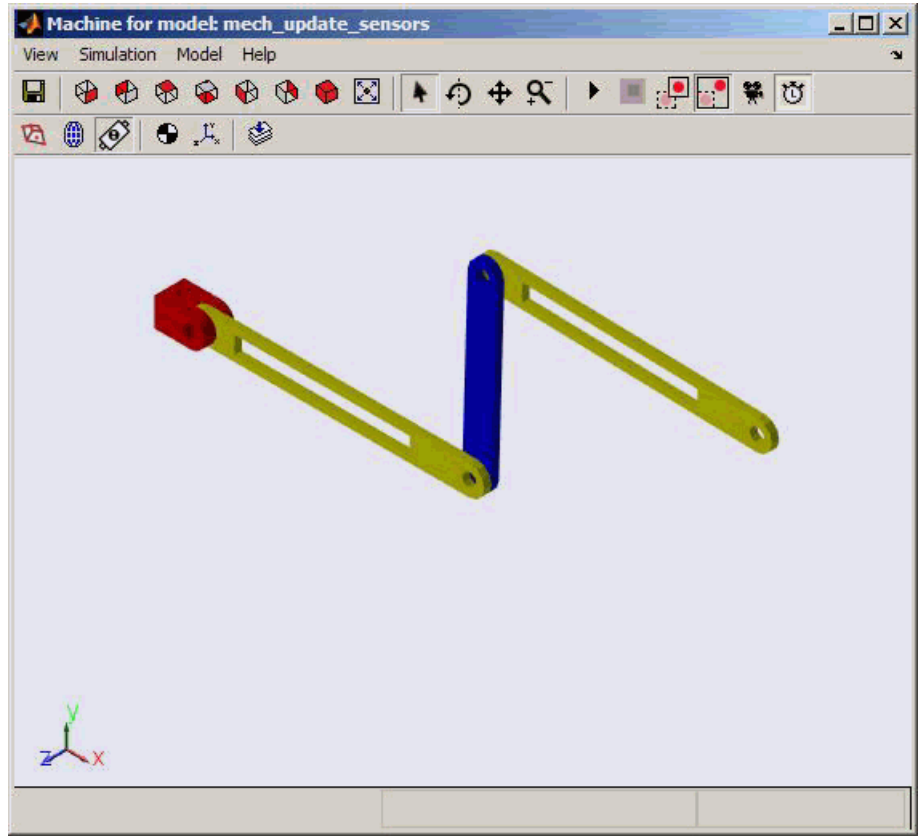
Advanced Option	Effect of Option
	means you have to rearrange the newly imported blocks by hand.
Respect individual block update settings	Selecting this option preserves the associativity information specified on each block.
Backup mode	Using this pull-down menu allows you to choose how the model is backed up during update. the backup options. By default, the backup copy of the model is created in the same folder as the model being updated.

The equivalent command line import command is:

```
mech_import('tpen.xml','ImportMode',1,...
        'ModelToImportInto','dpen_withsensors','LayoutWithUpdate',1,...
        'EnableIndvlBlkUpdCtrl',1,'BackupMode',0)
```

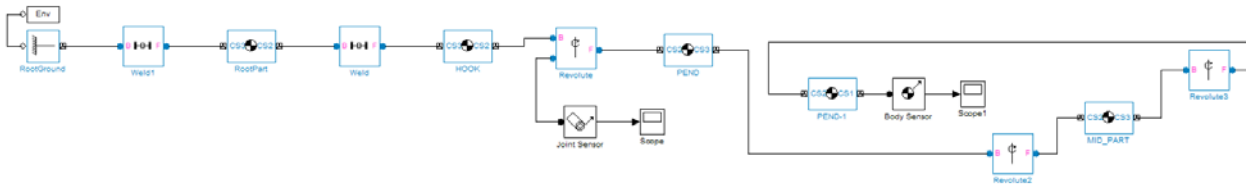
The block diagram shows that the part MID_PART inserted between the two links PEND and PEND-1. The sensors are retained. The properties of PEND and PEND-1 have been updated (to incorporate the holes that are now present). Because the **Arrange all updated blocks** option was selected, the blocks PEND and PEND-1 were moved to accommodate the block MID_PART and the joints Revolute2 and Revolute3.





Disabling Automatic Block Arrangement

The following block diagram shows the updated model that results when the **Arrange all updated blocks** option is cleared. The importer places the newly imported blocks around the bounding box of the existing blocks. The layout of the existing blocks is unchanged.



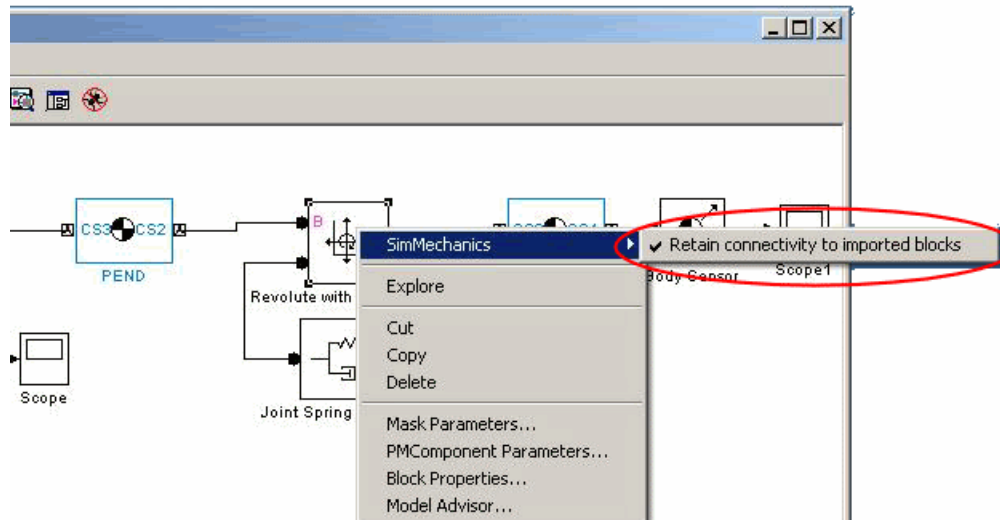
Updating an Existing Generated Model While Retaining Manual Joint Replacements

In circumstances where a set of assembly constraints imposes ambiguous restrictions on the motions of two parts, the SimMechanics Link exporter cannot resolve the constraint set into a moving joint. In such cases, it exports a Weld joint. If so, it is up to you to manually replace the Weld joint with the appropriate moving joint in the generated model. Even in situations where the exporter creates a moving joint, it might not reflect the degrees of freedom that you want, and you might want replace it with another. When you have replaced such joints in an imported model, and you want to update the model with changes from the CAD assembly, you generally want to retain the replacements you made manually.

1. Create a dpen_jointrep model in the following way.

   Open the dpen_withsensors model and replace the Revolute joint between the two link bodies by a Planar joint that has some compliance in the plane perpendicular to the original revolute axis. Rename the Planar joint to Revolute with Compliance.

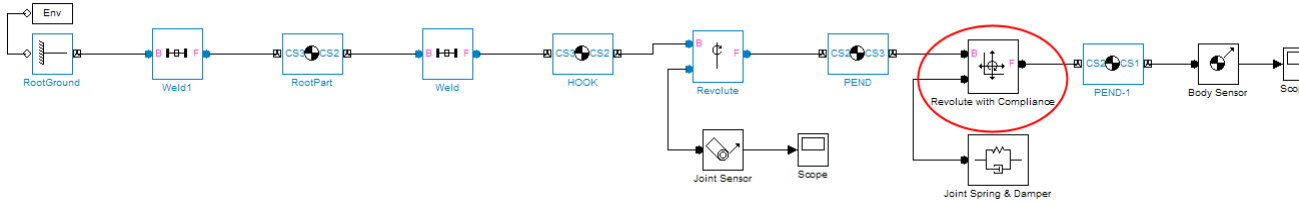
2. To update this model with changes to the link body properties, but to still retain the new Planar joint, right-click the Revolute with Compliance block and select the **Retain connectivity to imported blocks** option. It is selected by default.



3 Update this model with the new XML file that has changes to the properties of the links.

```
mech_import('dpen.xml','ImportMode',1,...
        'ModelToImportInto','dpen_jointrep','LayoutWithUpdate',1,...
        'EnableIndvlBlkUpdCtrl',1,'BackupMode',0)
```

The updated model retains the Revolute with Compliance joint.



Selectively Updating an Existing Generated Model

You can update certain bodies while not updating other bodies with changes. Prevent a particular body from being updated during reimport by:

1. Right-clicking the body and selecting the **Retain without updating properties** option under the **SimMechanics** submenu.

2. Selecting the **Respect individual block update settings** option in the import dialog box.

   If you use the full command instead, you need to set the `EnableIndvlBlkUpdCtrl` option to true.

Clearing the **Respect individual block update settings** check box causes the importer to update all blocks, disregarding the update settings on the individual blocks.

Translating a CAD Robot Arm

In this section...

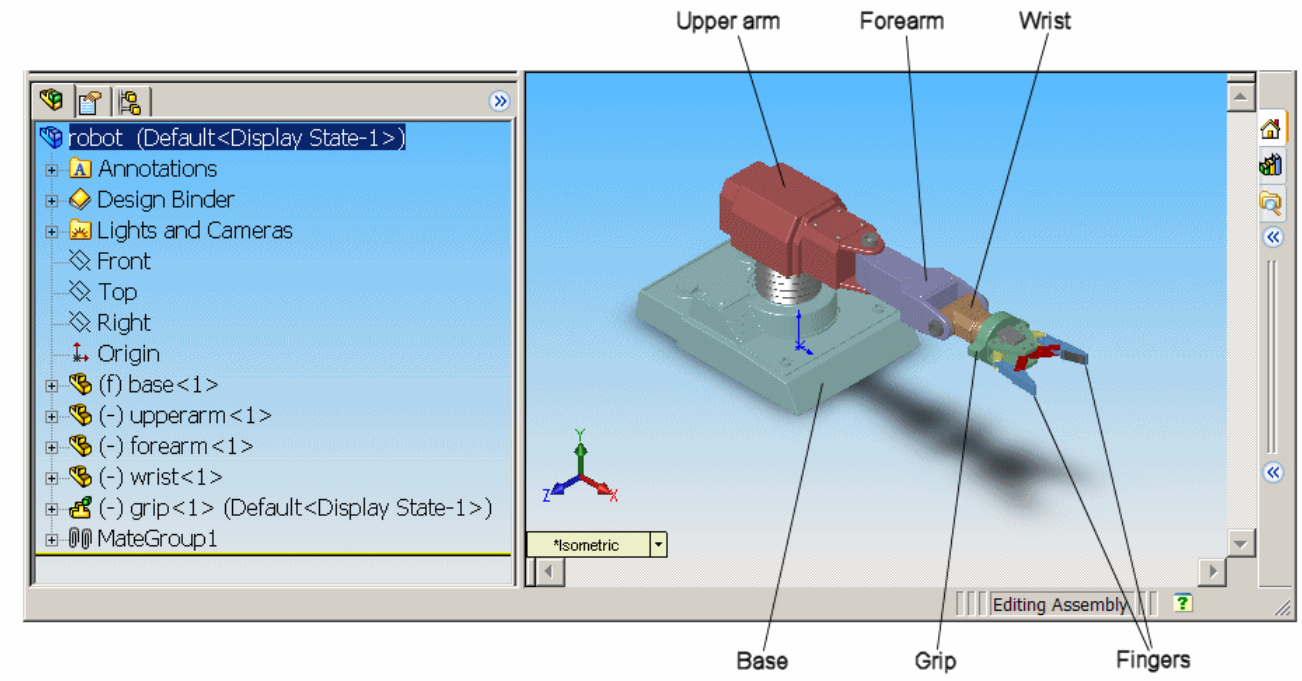
Locating the Robot Arm Assembly Files

The following example is based on a more complex CAD assembly, a robot arm. It includes multiple parts, two closed loops, multiple constraints, and a subassembly. Look for the 11 CAD files of this case study in the SimMechanics Link demos folder.

File Name	CAD File Type
`robot.`*`ASSEMBLYFILETYPE`*	Assembly
`grip.`*`ASSEMBLYFILETYPE`*	Subassembly (flexible)
`base.`*`PARTFILETYPE`* `forearm.`*`PARTFILETYPE`* `upperarm.`*`PARTFILETYPE`* `wrist.`*`PARTFILETYPE`*	Parts (main assembly)
`fingertips.`*`PARTFILETYPE`* (twice) `firstfingerlink.`*`PARTFILETYPE`* `firstfingerlinkL.`*`PARTFILETYPE`* `metacarpal.`*`PARTFILETYPE`* `secondfingerlink.`*`PARTFILETYPE`* (twice)	Parts (subassembly)

Viewing the Robot Arm Assembly

Open the assembly file for the whole robot.



Robot Arm Assembly in a CAD Platform

In the assembly tree to the left of the window, examine the CAD hierarchy:

- Five of the part files are grouped into the subassembly `grip`. The subassembly uses two instances each of `fingertips` and `secondfingerlink`.
- The subassembly has its own group of 18 constraints, MateGroup1.

  Two constraints, Angle1 and Angle2, are not active. If they were, they would lock the grip fingers into the open position. Here, each grip finger can move separately.
- The other four part files are separate and grouped into the main assembly.

- The main assembly has its own MateGroup1, consisting of seven constraints.

The whole assembly has eight DoFs. The `grip` subassembly alone contains two, allowing each finger to open and close separately. The main assembly has six DoFs:

- The upper arm can move relative to the base by pitching, yawing, and rolling (three DoFs).
- The forearm can yaw relative to the upper arm (one DoF).
- The wrist can pitch relative to the forearm (one DoF).
- The grip can rotate about its symmetry axis (one DoF).

Exporting the Robot Arm Assembly

Apply any changes you want to the assembly configuration or settings. If you change the assembly or any subassemblies, you need to rebuild the assembly before exporting it to XML.

Using the SimMechanics Link interface to your CAD platform, export the assembly into Physical Modeling XML. The XML file `robot.xml` appears in your working CAD folder.

Generating and Completing the Robot Arm Model

Generate a SimMechanics model for the robot arm based on the file `robot.xml`. You can use this preconfigured demo file or export your own version of the XML file from the robot arm CAD assembly. In either case, copy or move the XML file to your MATLAB working folder.

Generating the Initial Model

The preconfigured `robot.xml` file is in the SimMechanics demos folder.

1. Generate the model by entering mech_import('robot') at the command line.

   The status bar opens and indicates the progress of model generation. A model window, named **robot**, opens and is populated with blocks.

2 Save this initial body-joint model as robot, and note these properties:

- The top level of the model has 13 blocks and the grip-1 subsystem.
- The grip-1 subsystem has 18 blocks.

  The original robot arm assembly has eight DoFs, with two in the grip subassembly and six at the top level. These translate into eight DoFs in the SimMechanics model, where:

  - Six DoFs occur at the top level. These include the upper arm relative to the base, the forearm relative to the upper arm, the wrist relative to the forearm, and the grip relative to the wrist.
  - Two DoFs occur in the grip-1 subsystem. These are the rotational DoFs of the two grip fingers.

    There are eight revolute primitives in the subsystem. They occur in two closed loops as two independent four-bar mechanisms. Each four-bar mechanism actually has only one independent DoF because each four-bar loop closes on itself.

Note For more about counting mechanical degrees of freedom, see the *SimMechanics User's Guide*.

Obtaining Simulink and Additional SimMechanics Blocks

To modify and extend the robot arm model, you need blocks from the SimMechanics and Simulink block libraries. Open these libraries by entering `mechlib` and `simulink`, respectively, at the command line.

You can also open the Simulink library from the MATLAB window menu or toolbar.

Editing the Bodies

Some of the bodies in the generated robot arm model are redundant. You can remove them without affecting the model's dynamics, as long as you properly reconnect the remaining blocks.

- At the top level, the SimMechanics_RootPart block is a zero-mass, zero-inertia Root Body. You can delete it, along with the connected Weld1 block, then reconnect the Root Ground to the base-1 block through the Weld block.
- In the grip-1 subsystem, you can delete the grip-1 (Root Body) block and the connected Weld block because they are unnecessary. You can also delete the associated Body coordinate system on the metacarpal-1 Body block.

See the reference page for more details about the Body block.

Editing the Joints

The non-Weld Joint blocks, those that carry DoFs, are Revolute and Spherical Joints configured with the proper primitives to represent the original CAD assembly's DoFs.

- The first non-Weld Joint you encounter as you move away from the Ground block is a Spherical, representing three DoFs.
- Each Revolute block contains a single revolute primitive, representing one rotational DoF.

Save this intermediate model as robot2.

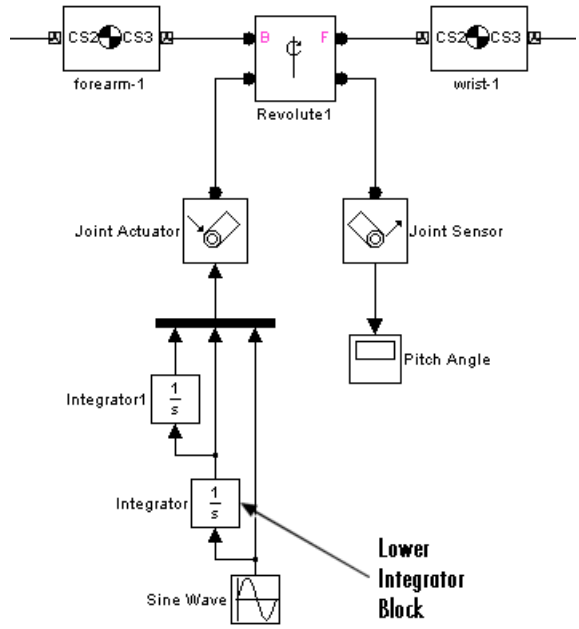
Adding an Actuator and a Sensor

You can motion-actuate the wrist relative to the forearm.

1. Double-click the Revolute that connects the forearm-1 and wrist-1 Body blocks. Change the **Number of sensor/actuator ports** to 2.
2. Click **OK**. Two new ports appear on the Joint.
3. From the SimMechanics Sensors & Actuators library, insert and attach a Joint Actuator and a Joint Sensor to these new ports.
4. Configure the Joint Actuator to accept motion signals. Be sure the angular units are `deg` (degrees).

5 From the Simulink library, insert a Sine Wave, a Mux, two Integrator blocks, and one Scope block. Connect them to the previous blocks as shown in the following figure. Rename the Scope block to Pitch Angle.

Consult the Simulink documentation for more about these Simulink blocks.



6 In the Sine Wave block, set the **Amplitude** to `60*pi*pi` and the **Frequency** to `60`. Leave all other defaults.

7 In the lower Integrator block, set **Initial condition** to `-60*pi`. Leave all other defaults.

Configuring Tolerances

The original robot arm CAD assembly requires looser tolerances than the SimMechanics defaults, and its motion can lead to singularities. To avoid simulation errors or slowdown, you need to reconfigure the assembly tolerances and constraint solver:

1. Open the Machine Environment block.
2. On the **Parameters** tab, reset the **Linear assembly tolerance** to `1e-2 m` (meters) and the **Angular assembly tolerance** to `1e-1 rad` (radians).
3. On the **Constraints** tab, select the **Use robust singularity handling** check box. Leave all other defaults. Click **OK**.
4. Resave your finished model as robot3.

Simulating and Observing the Robot Arm Motion

Run `robot3` and examine its motion.

To use the motion sensor:

1. Double-click the Pitch Angle block to open a scope.
2. Click the **Start simulation** button. The scope plot displays a trace of the pitch angle motion.

To visualize the body motions:

1. From the **Simulation** menu, select Configuration Parameters, then the **SimMechanics** node.
2. Select **Display machines after updating diagram** and **Show animation during simulation**. Click **OK**.
3. Select **Update Diagram** from the **Edit** menu. The SimMechanics visualization window opens.
4. In the **SimMechanics** menu of the visualization window, select **Machine Display**, then **Ellipsoids**. The display now shows the robot arm's component bodies as ellipsoids.
5. Click the **Start** button. The simulation begins. Observe the robot arm motion in the SimMechanics window.

Translating a CAD Stewart Platform

In this section...

Introducing the Stewart Platform

The Stewart platform consists of two plates connected by six mobile and extensible legs. The lower or base plate is immobile. The upper or mobile plate has six degrees of freedom, three rotational and three translational. The platform is a six-degree-of-freedom (DoF) mechanical system used for accurate positioning applications. It is highly stable and easy to control.

The platform's six legs each have two parts, an upper and a lower leg, with a piston-like cylindrical DoF between each pair of parts. The legs are connected to the base plate and the top plate by universal joints at each end of each leg. (These universals are not just sets of abstract DoFs. Each also contains a spider-like body, while also having two DoFs.) The upper part of each leg can slide into and out of the lower leg, allowing each leg to be varied in length. The position and orientation of the mobile platform (top plate) varies depending on the lengths to which the six legs are separately adjusted.

Once the top is connected to the legs, the entire Stewart platform assembly has 36 DoFs. Only six DoFs are *independent*, the same as the top plate would have if it were disconnected. You can think of these independent DoFs as the six adjustable leg lengths or as equivalent to the six DoFs of the mobile plate.

Introducing the Stewart Platform Assembly

The following example uses a complex computer-aided design (CAD) assembly that models the Stewart platform.

Note The Stewart platform assembly in this example is an advanced example of computer-aided design. You should work through the previous case studies before attempting to work with this assembly.

To learn more about the Stewart platform, see the Motion, Control, and Real-Time Simulation chapter of the *SimMechanics User's Guide*.

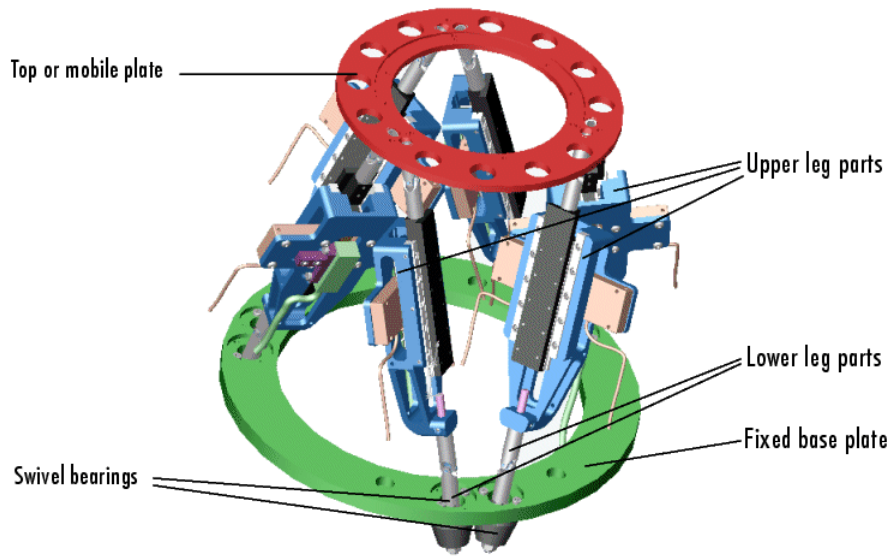
Locating the Stewart Platform Assembly Files

Look for the 45 CAD files of this case study in the SimMechanics Link demos folder. The master assembly file is:

`stewart_platform.`*`ASSEMBLYFILETYPE`*

Viewing the Stewart Platform Assembly

Open the master assembly file, `stewart_platform.`*`ASSEMBLYFILETYPE`*. Click the assembly and rotate it to view the top and bottom plates and the legs.



Stewart Platform CAD Assembly

The CAD hierarchy for the Stewart platform contains assemblies for the top and base plates, as well as assemblies for the six legs. All the constraints on the assembly parts are grouped into one group, containing 30 constraints. There are 448 component parts and 38 subassemblies, which you can open individually to examine the separate parts.

The base plate is about 24 centimeters (cm) in diameter; the top plate about 16.5 cm. When centered and oriented flat, the top plate is about 20 cm above the base. The assembly models the platform material as aluminum (about 2.7 grams per cubic cm).

Exporting the Stewart Platform Assembly

Apply any changes you want to the assembly configuration or settings. If you change the assembly or any subassemblies, you need to rebuild the assembly before exporting it to XML.

Using the SimMechanics Link interface to your CAD platform, export the assembly into Physical Modeling XML. Because the assembly is so complex, the export process takes longer than it does for simpler assemblies. As the export proceeds, various parts and subassemblies are highlighted. When the highlighting stops, the export is finished.

The exported model appears as `stewart_platform.xml` in your working CAD folder.

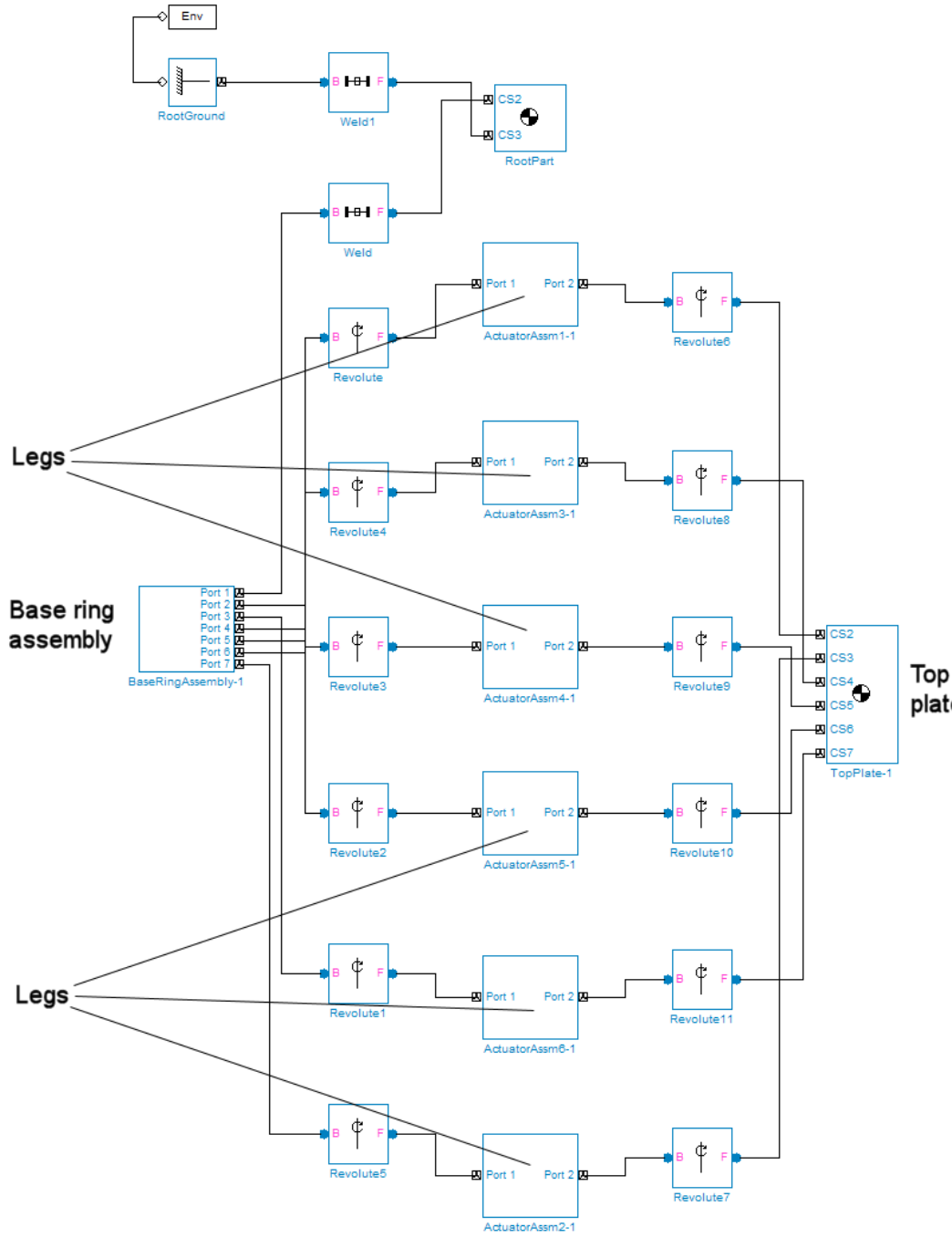
Generating the Stewart Platform Model

Move or copy the `stewart_platform.xml` file into your working MATLAB folder.

To generate a new Simulink model, enter `mech_import('stewart_platform')` at the MATLAB command line, and wait for the model generation `stewart_platform` to finish.

Inspecting the Generated Model and Counting Its DoFs

The complete Stewart platform model contains seven subsystems.



Stewart Platform Model: Base, Legs, and Top Plate

The subsystems correspond to the subassemblies of the original CAD assembly — the base plate and the six platform legs.

- The base plate subassembly `BaseRingAssembly-1` contains six subassemblies, modeling a base swivel bearing for each leg.
- The six leg subassemblies, `ActuatorAssm`, model the upper and lower halves of each leg and represent part of their DoFs.

For each leg, there are six DoFs. Two pairs of revolutes associated with each leg represent the two universal joints connecting each leg to the top and base plates, respectively. Each of these universals has two DoFs.

- At the top level, there are two revolutes, one attached to either end of a leg subassembly, connecting each leg to the base and top plates, respectively.
- Within each leg subassembly, there are two other revolutes, each one connecting the leg to the top and base plates, respectively.

One of the revolutes inside the leg subassembly pairs with one of the revolutes outside the leg assembly to make up a two-DoF universal. These pairs occur twice on each leg, one connecting the leg to the top plate, the other connecting the leg to the base plate.

- Within each leg subassembly, there is one prismatic, representing the leg's freedom to expand or contract along its shaft.
- Within each swivel bearing subassembly, itself located within the base ring assembly, is another revolute representing each leg's freedom to rotate about its shaft.

Each leg has six DoFs. However, the constraints imposed by attaching each leg to fixed points on the base and top plates, respectively, reduce these to one independent DoF for each leg — the freedom to expand or contract along its shaft.

- The rotational DoFs associated with the universals at the attachment points are completely dependent on the leg's prismatic DoF.
- The rotational DoFs associated with the cylindricals in each leg are completely dependent on the universals at the top and bottom of each leg.

Deleting Unnecessary Bodies and Joints

The generated model contains a large number of redundant Root Weld and zero-mass Root Part blocks. You can delete these and not affect the model's dynamics, as long as you take care to reconnect the remaining bodies properly after deleting each Weld.

Adding Actuators and Sensors

If you want the motion of the platform to be controlled by something other than gravity, you need to add the appropriate Actuators to the model. To quantify the model's motion, you need to make precise measurements with Sensors. You can drive the actuators with external control signals to model an open-loop controller for the Stewart platform. If you introduce feedback from the sensors to the actuators, you can model a closed-loop controller.

Visualizing the Stewart Platform Motion

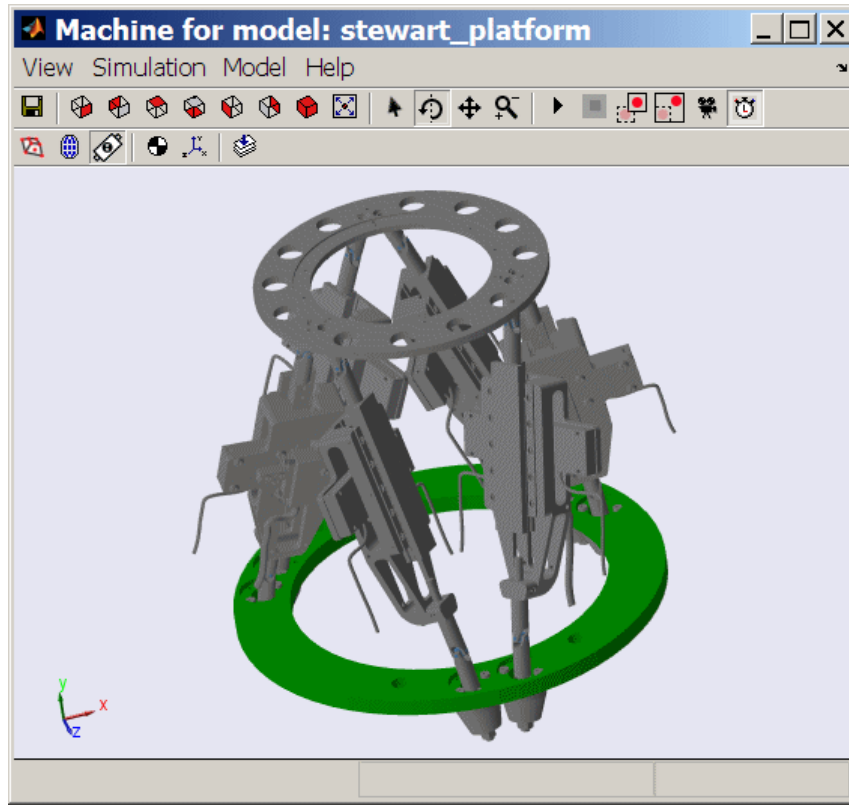
Note You can find more information about SimMechanics visualization in the *SimMechanics Visualization and Import Guide*.

Without any external forces acting, apart from gravity, the platform collapses under its own weight. You can verify this by running and visualizing your Stewart platform model.

1. From the **Simulation** menu, select **Configuration Parameters**. The Configuration Parameters dialog opens. Choose the **SimMechanics** node.
2. Select **Display machines after updating diagram** and **Show animation during simulation**. Click **Apply** or **OK**.
3. From the **Edit** menu, select **Update Diagram**. The SimMechanics visualization window opens with the SimMechanics controls. The window displays the Stewart platform in its initial position.
4. Start the simulation by clicking the **Start** button in the toolbar of either the visualization window or the model window.

The mobile plate falls under its own weight and reaches the base plate in about 0.2 seconds. Because there is nothing to stop the legs or the top plate, the platform continues to collapse: the mobile plate falls below the base plate, and the upper and lower parts of each leg come apart.

This visualization of the Stewart platform uses custom body visualization with the STL body geometry files exported from the original CAD assembly.



SimMechanics™ Visualization of the CAD-Based Stewart Platform (Custom Body Geometries)

4

Custom Linking to CAD and Other External Applications

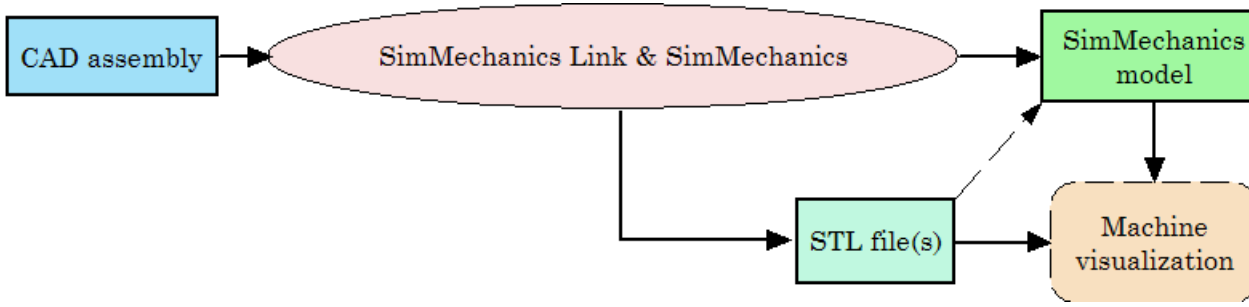
The SimMechanics Link software includes an application programming interface (API). With the API of an external application program and the SimMechanics Link API, you can selectively transfer data from the external application to the SimMechanics Link software.

Custom Export with the SimMechanics Link API

In this section...

About CAD Translation with Custom Export

The translation process from an externally defined machine representation to a SimMechanics model is the same whether you use the *standard* export with a supported external third-party platform or create your own *custom* exporter. The SimMechanics Link exporter translates mechanical system data from an external application, such as a computer-aided design (CAD) platform. You can use this translated data to generate a SimMechanics model of your original mechanical system.



Reasons for Custom Export

You must create and use your own custom exporter when:

- You want to export an assembly from a CAD platform that the SimMechanics Link utility does not support.
- The standard export from a supported CAD platform does not yield the results that you want.

Requirements for CAD Translation with Custom Export

CAD translation includes CAD assembly export and SimMechanics model import. Export requires:

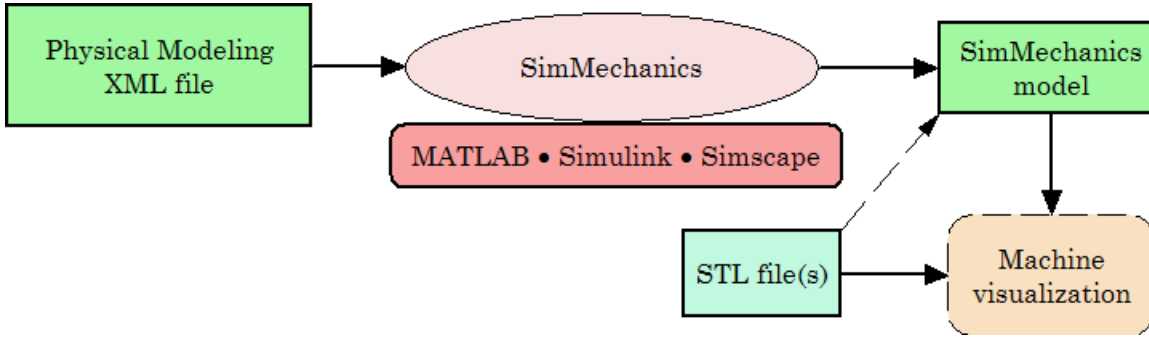
- A CAD platform or application, including access to its application programming interface (API)
- MATLAB installed and registered as a server
- The SimMechanics Link utility, which includes its own API
- A *custom exporter module* that you create and that transfers assembly information from the CAD platform API to the SimMechanics Link API

Import requires SimMechanics software.

Custom Translation Steps in Common with Standard Export

You generate SimMechanics models from Physical Modeling XML files. You use mechanical import and then visualize the models with stereolithographic (STL) files. *Custom* export creates an XML file to represent a CAD assembly and a set of STL files to represent the surface geometries of the assembly bodies. This part of CAD translation is the same as *standard* export.

See Chapter 2, “Getting Started with Export” and Chapter 3, “Computer-Aided Design Translation”.

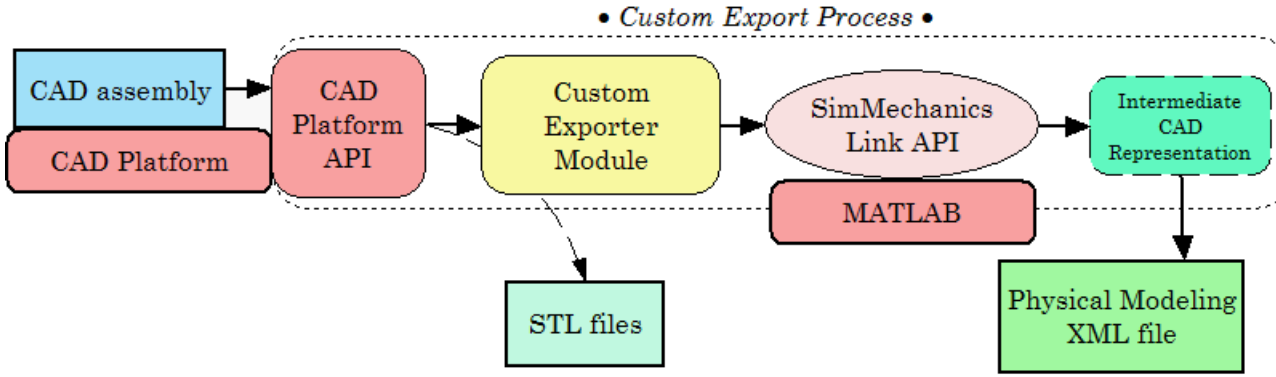


From Physical Modeling XML to Visualizable SimMechanics™ Model

Custom Translation Steps Different from Standard Export

For *custom* export, you must create a custom exporter module that interacts with the APIs of both the CAD platform and the SimMechanics Link utility. From a CAD assembly, this custom exporter module creates a selective, intermediate representation of the machine that persists for one session. The module then finishes by creating the same files as the standard export does:

- A Physical Modeling XML file representing selected data required to generate a SimMechanics model, which is written from the selective representation
- A set of STL files to represent the surface geometries of the assembly bodies



Custom Export: From CAD Assembly to Physical Modeling XML

Requirements for Creating a Custom Exporter

Note The SimMechanics Link API is supported on all platforms that MATLAB supports.

To create a custom exporter:

- Access the API of your CAD platform and understand how to call it from an external program. See your CAD platform documentation.

- Access the SimMechanics Link API and understand how to call it from an external program.
- Write the custom exporter module in C/C++.

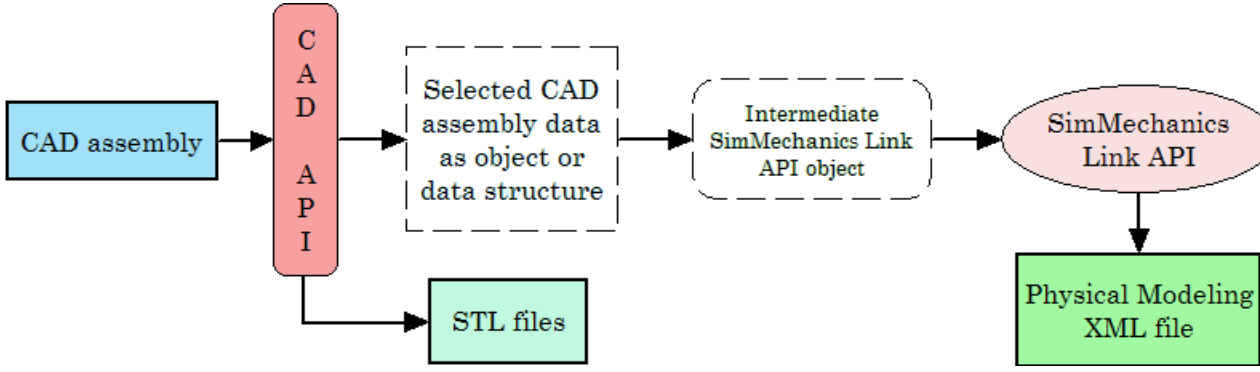
  The module might be an executable or a linked library, depending on your CAD platform requirements and API.

Translating Machines from External to Physical Modeling Representations with the API

In this section...

About Mapping API Objects from CAD Format to Physical Modeling XML

A complete CAD assembly contains information both relevant and not relevant for exporting a Physical Modeling XML file to represent your assembly.



Data Required from a CAD Assembly to Create a SimMechanics Model

You obtain the selected assembly data through the CAD platform application programming interface (API) in the form of an object or a data structure. The data required from an assembly to construct a SimMechanics model include:

Mapping of Selected Mechanical Data to SimMechanics Model Components

Data Selected from CAD Assembly or Other External Machine Definition Mapped to...	...Required SimMechanics Model Component
Assembly and parts origin	Ground and World coordinate system
Assembly and subassembly hierarchy	Model and subsystem hierarchy
Part masses and inertia tensors, colors, and body geometry file names	Bodies, including their mass properties, geometries, and colors
Constraint restrictions on relative positions and motions of parts	Joints between pairs of Bodies Body coordinate systems connected to Joints

Using the Exporter Functions to Complete the Mapping of CAD Assembly to Physical Modeling XML

Refer to the table, Mapping of Selected Mechanical Data to SimMechanics™ Model Components on page 4-7. Using your CAD exporter module, you transform the selected data listed in the first column of the table into the form required in the second column with the SimMechanics Link API functions and data types. You collect the selected data into an intermediate SimMechanics Link API object. With the SimMechanics Link API, you then write the intermediate API representation to a Physical Modeling XML file.

See “Designing Custom Exporter Modules” on page 4-11 following for detailed information on these API functions and data types.

Body Geometry Files Exported Directly from CAD API

For each moving Body in the model, the final SimMechanics model requires a separate body geometry file in STL format. You export these body geometry files directly from the CAD API, with file names matching the body geometry file names specified in the exported XML file. You specify these file names in the SimMechanics Link API object.

Tip You need body geometry files only for individual moving CAD parts or rigid subassemblies, not for flexible subassemblies.

Selecting CAD Assembly Data for Export

Tracing the Assembly Hierarchy: Nodes and Components

The assembly data that you need to retrieve through the CAD API starts with the assembly-subassembly hierarchy, with nodes. Each node in the hierarchical tree is an assembly *component* (a part or a subassembly).

For an individual component, the next component immediately up in the hierarchy is its *parent*. All dependent components connected below in the hierarchy are its *children*. The top parent in the assembly hierarchy is the whole assembly itself.

Identifying and Extracting Component Information

For each component, determine:

- Its geometric transformation with respect to its parent, its mass, its inertia tensor, its body geometry and color (for a part, or for a rigid subassembly exported as a single rigid body)
- The constraints that restrict how its children can move with respect to each other
- Whether or not the component is rigid or flexible; that is, whether all its children are to be treated as a single rigid body or as individual moving bodies
- Whether or not the component is *fixed*; that is, whether it is rigid or moving with respect to its parent

Constructing Intermediate API Representations

Before you can export the CAD assembly, you must construct a selective intermediate CAD representation, a collection of objects that capture the required CAD data.

Objects Required for Intermediate Representations

- *CAD models* (not to be confused with CAD assemblies or SimMechanics models). These represent assemblies, subassemblies, and parts.
- CAD model references
- Components (parts or subassemblies)
- Constraints
- A translator object constructed from all the other objects

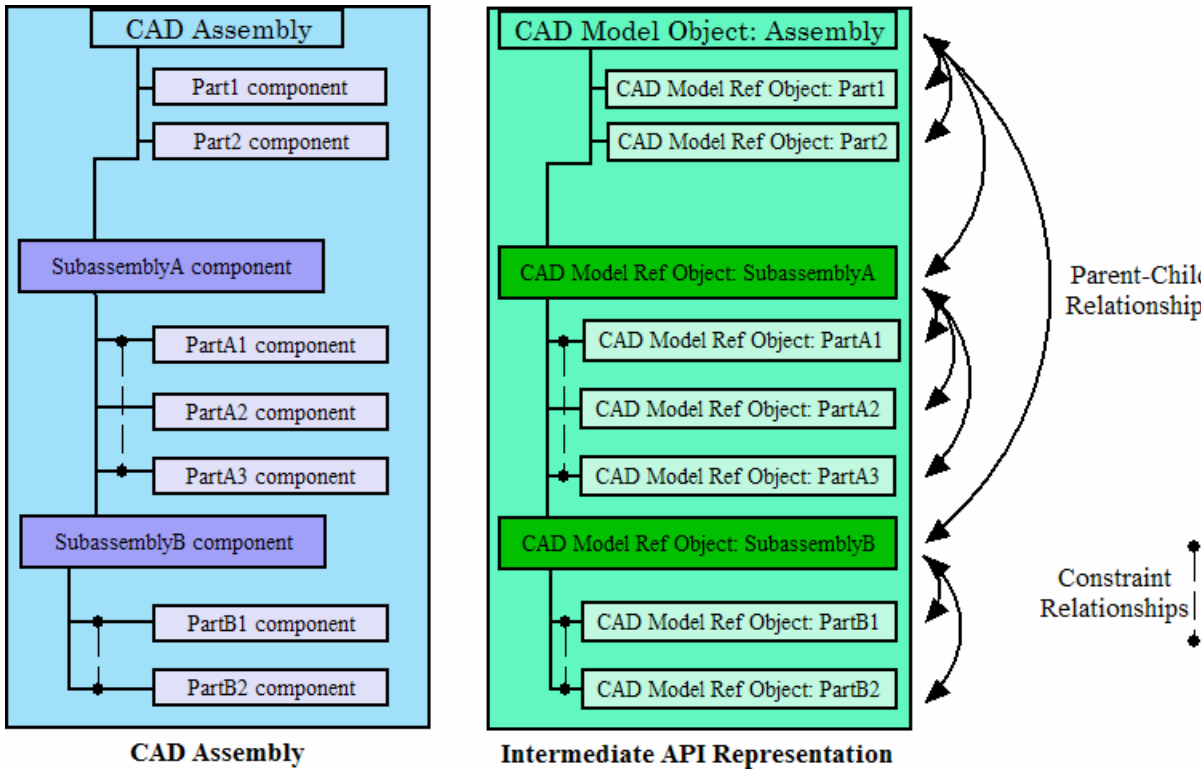
Putting Objects Together into Intermediate Representations

To construct a full CAD representation:

1. Represent the entire assembly with a CAD model object.
2. Represent child parts and subassemblies with additional CAD model objects.

   If a subassembly is flexible (its parts can move with respect to each other), the subassembly is a parent to child parts, represented by their own CAD model objects.
3. Add child CAD model reference objects to their parent CAD model objects, including parent-to-child transforms.

   You might need to repeat steps 1 through 3 recursively until you have mapped the entire CAD assembly hierarchy.
4. Represent constraints between components with constraint objects. These are also children to parent assembly and subassembly objects.
5. Form a single translator object from all these objects, referenced by one another in a hierarchy that parallels the original assembly hierarchy.



CAD Assembly and Intermediate API Representation: Parallel Hierarchies

Converting Selective API Representations into Physical Modeling XML

The final translator object is the last intermediate form of the CAD assembly translation. From this translator object, you create the Physical Modeling XML file representing the assembly.

Designing Custom Exporter Modules

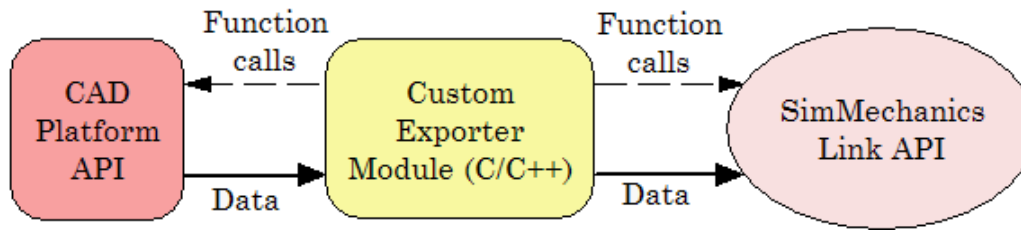
In this section...

Requirements for Custom Exporter Modules

A custom exporter module works with intermediate representations of CAD assembly data. For an overview of the types of CAD assembly and SimMechanics Link data and their relationship, see "Translating Machines from External to Physical Modeling Representations with the API" on page 4-6.

A custom exporter module:

- Issues function calls to the APIs of both the CAD platform and the SimMechanics Link utility.
- Transfers assembly representation data from the assembly, through the CAD API, to the Physical Modeling XML file, through the SimMechanics Link API.



Custom Exporter Module Communicates with APIs (Detail)

Using the SimMechanics Link API Functions

The SimMechanics Link API is a library of API functions that you call to construct a unified, selective representation of an assembly and write it as an XML file. "Programming Custom Exporter Modules with the SimMechanics

Link API" on page 4-15 describes how to create, compile, and execute custom exporter modules in C/C++ with the SimMechanics Link API function library.

Implementing Translation with CAD and SimMechanics Link APIs

Extracting Selective CAD Assembly Data

To extract the selected assembly data required for export, consult your CAD platform API documentation and determine which functions of the CAD platform API you need to accomplish this.

Constructing the Intermediate API Representation

To construct the intermediate representation of an assembly, use the following functions from the SimMechanics Link API, in the order shown. The API also contains other functions you might need in special cases. For more information, consult the *SimMechanics Link Reference*.

Step	To...	Use SimMechanics Link API Function...	...and SimMechanics Link API Objects
1.	Set length and mass units for CAD assembly. The defaults are meters (m) and kilograms (kg), respectively.	`pmit_set_units`	—
2.	a. Create CAD model object to contain data for CAD assembly	`pmit_create_cadmodel`	`PmitCadModelH`
	b. Create API representation translator object	`pmit_create_cad2sm`	`PmitCad2SMH`
3.	Set translational, rotational, and numerical tolerances for CAD assembly	`pmit_set_tolerances`	`PmitCad2SMH`

Step	To...	Use SimMechanics Link API Function...	...and SimMechanics Link API Objects
4.	a. Create object to contain data for CAD subassemblies and parts	`pmit_create_cadmodel`	`PmitCadModelH`
	b. Create reference to CAD model object	`pmit_create_cadmodelref`	`PmitCadModelH`, `PmitCadModelRefH`
	c. Add reference within existing CAD model object to child model object by adding transform	`pmit_add_refincadmodel`	`PmitCadModelH`, `PmitCadModelRefH`
5.	a. Create object to contain data for CAD assembly component (part or rigid subassembly)	`pmit_create_assemcomp`	`PmitAssemCompH`
	b. Add reference within existing CAD model object to child assembly component object	`pmit_add_refincomp`	`PmitAssemCompH`, `PmitCadModelRefH`
6.	a. Create object to contain data for constraint between CAD components (parts or rigid subassemblies)	`pmit_create_constrain`	`PmitAssemCompH`, `PmitConstrainH`, `PmitConstrainType`, `PmitGeomType`
	b. Add constraint object to CAD model object	`pmit_add_constrain`	`PmitCadModelH`, `PmitConstrainH`

Exporting the Intermediate API Representation into Physical Modeling XML

To export the selective representation of your assembly into a Physical Modeling XML file, use the following functions from the SimMechanics Link API, in the order shown.

Step	To...	Use SimMechanics Link API Function...	...and SimMechanics Link API Objects
1.	Enable translation of CAD assembly data into SimMechanics model	`pmit_create_cad2sm`	`PmitCad2SMH`
2.	Write Physical Modeling XML file from selective CAD representation of assembly	`pmit_write_xml`	`PmitCad2SMH`

Programming Custom Exporter Modules with the SimMechanics Link API

In this section...

Including, Linking to, and Calling the API Function Library

In the following procedures, replace:

- *`matlabroot`* with the root of your MATLAB installation.
- *`ARCH`* with the specific operating system architecture abbreviation for your system (for example, `win32` for 32-bit Windows).

Including and Using the API Header File

To write your C/C++ custom exporter module, use the C header file located in *`matlabroot`*`/toolbox/physmod/smlink/api/include/`.

This header file contains the function definitions that you follow to call the SimMechanics Link API functions from your module. Include this header file in your module C/C++ source code.

Compiling and Linking the Custom Exporter Module

In addition to linking your custom module to the SimMechanics Link API, consult your CAD platform's API documentation to determine requirements for linking to the platform's API.

On Windows Platforms. After you compile your module, link it to the binary API function library located in *`matlabroot`*`\toolbox\physmod\smlink\api\lib\`.

On UNIX, LINUX, and Mac Platforms. After you compile your module, link it to the binary API function library:

matlabroot/bin/*ARCH*/libmwpmi_api.*OS-SPECIFIC-EXTENSION*

OS-SPECIFIC-EXTENSION is the API file extension specific to your operating system.

Executing the Custom Exporter Module

To determine if you need to include your CAD platform API on the path, consult your CAD platform API documentation.

On Windows. Before you start execution of the module, verify that the folder *matlabroot*\bin*ARCH*\ is on the path.

On UNIX, LINUX, and Mac Platforms. Before you start execution of the module, verify that the folder *matlabroot*/bin/*ARCH*/ is included in the system path environment variable LD_LIBRARY_PATH.

Locating API Code Examples

You can find API examples in this folder:

matlabroot/toolbox/physmod/smlink/api/example/

The subfolder, cadapi_example/, contains CAD-based examples.

A Custom Exporter Module Example

The folder *matlabroot*/toolbox/physmod/smlink/api/example/cadapi_example/ contains a simple C++ example of a custom module based on the API, cadapi_example.cpp.

- The example custom module converts two CAD parts with one constraint into two SimMechanics bodies and one joint and then writes the result to a Physical Modeling XML file.
- The example program defines the CAD assembly data internally, rather than taking the data from a CAD platform through the platform API.

Compared to the figure, Custom Exporter Module Communicates with APIs (Detail) on page 4-11, the leftmost portion of the diagram is missing.

- To extend the example module to process real assembly data, add code that calls a CAD platform's API for assembly data to replace the internally defined data.

Index

P

R

S

T

U

X