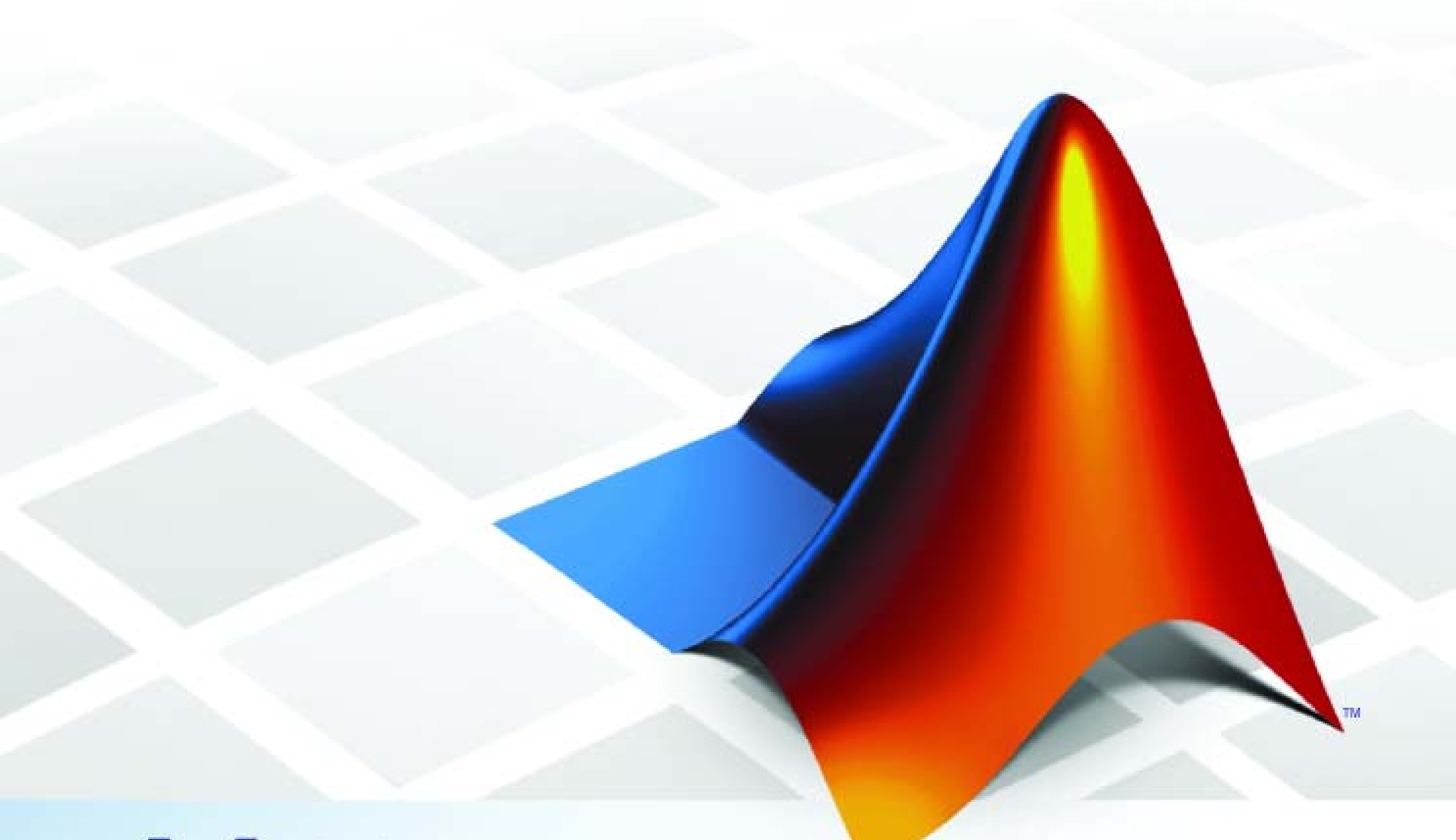
SimMechanics™ Link 1
User's Guide



MATLAB®

The MathWorks™
Accelerating the pace of engineering and science

How to Contact The 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 1.0 (Release 2008b)

Contents

1 Introducing SimMechanics Link Software

Software Overview **1-2**
Software Definition **1-2**
Using SimMechanics Link and SimMechanics Software Together for Complete CAD Translation **1-2**

Related Products **1-4**
Required Products **1-4**
Other Related Products **1-4**

Installing and Linking SimMechanics Link Software .. **1-6**
Requirements for Using SimMechanics Link Software ... **1-6**
Downloading and Installing SimMechanics Link Software **1-7**
Calling MATLAB Through the SimMechanics Link Interface **1-7**
Linking and Unlinking SimMechanics Link Software To a Supported CAD Platform **1-8**
Linking SimMechanics Link Software To an Unsupported External Application **1-9**

What Can You Do With SimMechanics Link Software? **1-10**
Exporting CAD Assemblies Into Physical Modeling XML **1-10**
Creating a Custom Link using the SimMechanics Link API **1-10**

Learning More **1-11**
Using the MATLAB Help System for Documentation and Demos **1-11**
Finding Related Documentation **1-11**

2 Getting Started with Export

Mechanical Export and Translation **2-2**
About CAD Translation: CAD Assembly to SimMechanics Model **2-2**
CAD Export: CAD Assembly to Physical Modeling XML .. **2-3**
XML Import: Physical Modeling XML to SimMechanics Model **2-4**

Designing a CAD Assembly for Export **2-5**
How CAD Assemblies Are Translated into SimMechanics Models **2-5**
Preparing a CAD Assembly for Translation into a SimMechanics Model **2-7**

Exporting and Re-Exporting a CAD Assembly **2-9**
About CAD Assembly Export **2-9**
Configuring the Exporter Controls in Supported CAD Platforms **2-10**
Creating the XML File **2-11**
Re-Exporting an Assembly After Changes **2-11**
Exporting a Robot Arm Assembly for the First Time **2-12**
Re-Exporting a Robot Arm Assembly After Changes **2-14**

Troubleshooting Export Problems **2-16**
Troubleshooting Assembly Export Failures **2-16**
Getting Exporter Help from the CAD Platform Interface .. **2-16**

Retranslating from Assembly to Existing Generated Model **2-17**
Working with Associativity in Common Updating Situations **2-17**
Understanding Associativity in General **2-20**

3 Computer-Aided Design Translation

Introducing the Case Studies **3-2**
For More About CAD Assembly Export **3-2**
For More About SimMechanics Import and Model Generation **3-3**
For More About Simulink and SimMechanics Blocks and Commands **3-3**
Defining Specialized CAD Terms **3-3**

Translating a CAD Part into a Body **3-4**
Locating the Single-Part Assembly Files **3-4**
Viewing the CAD Assembly **3-4**
Exporting the CAD Assembly **3-6**
Generating the SimMechanics Model **3-6**

Translating CAD Constraints into Joints **3-8**
Modeling CAD and SimMechanics Degrees of Freedom ... **3-8**
Locating the Constraint Assembly Files **3-9**
Generating the Two-Part Models: Common Steps **3-10**
Understanding the Two-Part Models: Common Block Structure **3-10**
Modeling a Six-DoF Joint **3-11**
Modeling a Prismatic Joint **3-12**
Modeling a Revolute Joint **3-16**
Modeling an Inplane Joint **3-17**
Modeling a Spherical-Spherical Massless Connector **3-18**

Translating a CAD Robot Arm **3-21**
Locating the Robot Arm Assembly Files **3-21**
Viewing the Robot Arm Assembly **3-22**
Exporting the Robot Arm Assembly **3-23**
Generating and Completing the Robot Arm Model **3-23**
Simulating and Observing the Robot Arm Motion **3-27**

Translating a CAD Stewart Platform **3-28**
Introducing the Stewart Platform **3-28**
Introducing the Stewart Platform Assembly **3-28**
Viewing the Stewart Platform Assembly **3-29**
Exporting the Stewart Platform Assembly **3-30**

Generating the Stewart Platform Model **3-30**
Visualizing the Stewart Platform Motion **3-33**

4 Creating Custom Links to External Applications

Custom Export with the SimMechanics Link API **4-2**
About CAD Translation with Custom Export **4-2**
What's the Same: Translation Steps in Common with Standard Export **4-3**
What's Different: Translation Steps with Customized Export **4-4**
Requirements for Creating a Custom Exporter **4-4**

Understanding the Mapping of API Objects **4-6**
About Mapping API Objects from CAD Format to Physical Modeling XML **4-6**
Selecting CAD Assembly Data for Export **4-8**
Constructing Selective CAD Representations **4-8**
Converting Selective CAD Representations into Physical Modeling XML **4-9**

Designing Custom Exporter Modules **4-10**
Requirements for Custom Exporter Modules **4-10**
Implementing Translation with the CAD and SimMechanics Link APIs **4-11**

Programming with the SimMechanics Link API **4-13**
Including, Linking to, and Running with the API Function Library **4-13**
A Custom Exporter Module Example **4-14**

Index

Introducing SimMechanics Link Software

- "Software Overview" on page 1-2
- "Related Products" on page 1-4
- "Installing and Linking SimMechanics Link Software" on page 1-6
- "What Can You Do With SimMechanics Link Software?" on page 1-10
- "Learning More" on page 1-11

Software Overview

In this section...

"Software Definition" on page 1-2

"Using SimMechanics Link and SimMechanics Software Together for Complete CAD Translation" on page 1-2

Software 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 combines the power of Simulink® and SimMechanics software with CAD.

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

Using SimMechanics Link and SimMechanics Software Together for Complete CAD Translation

With the SimMechanics Link utility, you export

- CAD assemblies into Physical Modeling XML format.
- Graphics files to define the body geometries of the assembly parts.

This export step constitutes the first part of complete translation of CAD assemblies into models that you simulate with SimMechanics and Simulink software.

The second 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.

- See Chapter 3, “Computer-Aided Design Translation” for examples of complete assembly-to-model translation.
- For more about XML import and model generation, see the *SimMechanics Visualization and Import Guide*.

Related Products

In this section...

"Required Products" on page 1-4
"Other Related Products" on page 1-4

Required Products

You must have current versions of the following products installed to use SimMechanics Link software:

- MATLAB®

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 should 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* for further details.

Physical Modeling Product Family

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

- 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" on page 1-6

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

"Calling MATLAB Through the SimMechanics Link Interface" on page 1-7

"Linking and Unlinking SimMechanics Link Software To a Supported CAD Platform" on page 1-8

"Linking SimMechanics Link Software To an Unsupported External Application" on page 1-9

Requirements for Using SimMechanics Link Software

To use SimMechanics Link software, you must first install it. Once you have installed it, you must either

- Link it 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 1-8.
- Create your own application program interface (API) to your CAD platform or other external application, as described in "Linking SimMechanics Link Software To an Unsupported External Application" on page 1-9.

This section explains how to download, install, and link SimMechanics Link software, as well as register MATLAB as an automation server.

Installing SimMechanics Link Software After Installing MATLAB

You install the SimMechanics Link utility as part of your system's MATLAB files. But you must install the SimMechanics Link utility after you install MATLAB, in a separate step.

Downloading and Installing SimMechanics Link Software

Once 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.

Follow these steps to download and install the SimMechanics Link utility.

1. Point your Web browser to `http://www.mathworks.com/products/simmechanics/download_smlink.html`.

2. Follow the instructions on that page and proceed to the confirmation page.

3. Download the `install_addon.m` file and the platform-specific ZIP archive file.

   Archives for different operating systems are named accordingly; for example, `win32` for 32-bit Windows®.

4. Start your installed version of 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 unpacked to your *`$matlabroot`* directory. Any needed paths are added to the MATLAB path dynamically.

You must now register MATLAB and link SimMechanics Link software.

Calling MATLAB Through the SimMechanics Link Interface

When called from an external application, the SimMechanics Link utility attempts to connect to MATLAB. For this connection to succeed, MATLAB must be registered as an automation server on your system. See "MATLAB COM Automation Server Support" in *MATLAB External Interfaces*.

Once MATLAB is thus registered, you must link SimMechanics Link software to your CAD platform.

Registering MATLAB as an Automation Server

MATLAB is registered as an automation server with your system when you install it. But if, when you attempt to connect to the SimMechanics Link utility from your CAD platform, you get an error saying, "Could not connect to MATLAB," manually register your MATLAB as an automation server with one of these methods.

- Enter the system command `matlab -regserver`.

  This command opens a MATLAB session in non-desktop mode. Then just close that MATLAB session.

- Enter the command `regmatlabserver` at the MATLAB command line.

Calling MATLAB as an Automation Server from an External Application

Invoked from an external application, the SimMechanics Link utility connects to MATLAB in one of these ways.

- It opens MATLAB as an automation server, if a MATLAB session is not already open in automation server mode.
- It connects 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 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

First download and install SimMechanics Link software. Then make sure MATLAB is registered as an automation server on your system.

To link to a supported external CAD platform, consult the SimMechanics Link reference chapter for your platform, listed in this table. Follow these reference chapter instructions to unlink as well.

- "Linking and Using Pro/ENGINEER® Toolkit"
- "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 not directly supported, see Chapter 4, "Creating Custom Links to External Applications".

What Can You Do With SimMechanics Link Software?

In this section...

"Exporting CAD Assemblies Into Physical Modeling XML" on page 1-10

"Creating a Custom Link using the SimMechanics Link API" on page 1-10

Exporting CAD Assemblies Into Physical Modeling XML

With the SimMechanics Link utility, you can create a SimMechanics model from a CAD assembly, in two steps. The first 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.

SimMechanics Link export converts the mass and inertia of each part in the assembly and the constraint definitions between parts in the XML format. These XML representations of parts and constraints become bodies and joints in a SimMechanics model.

Exporting Body Geometries of CAD Parts

Export also captures the body geometries of the assembly parts as external graphics files. The generated SimMechanics model uses these body geometry graphics files to visualize the bodies.

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.

Learning More

In this section...

"Using the MATLAB Help System for Documentation and Demos" on page 1-11

"Finding Related Documentation" on page 1-11

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 an online index and search system.

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” on page 2-2
- “Designing a CAD Assembly for Export” on page 2-5
- “Exporting and Re-Exporting a CAD Assembly” on page 2-9
- “Troubleshooting Export Problems” on page 2-16
- “Retranslating from Assembly to Existing Generated Model” on page 2-17

Mechanical Export and Translation

In this section...

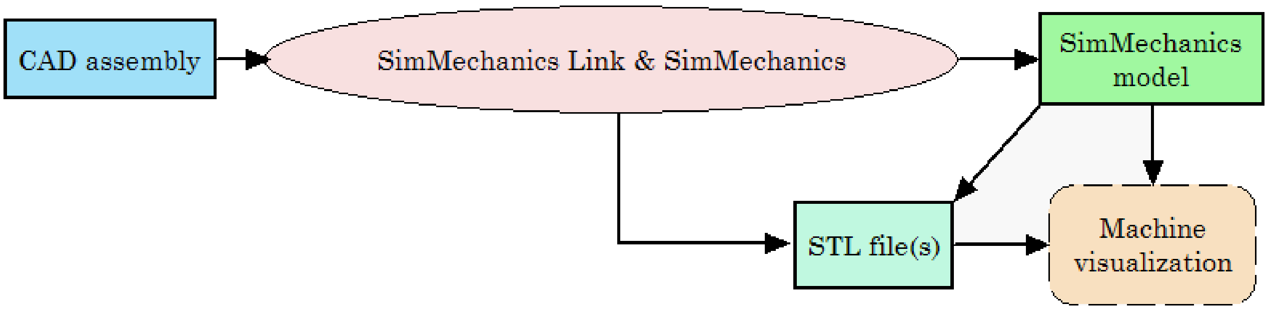
"About CAD Translation: CAD Assembly to SimMechanics Model" on page 2-2

"CAD Export: CAD Assembly to Physical Modeling XML" on page 2-3

"XML Import: Physical Modeling XML to SimMechanics Model" on page 2-4

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

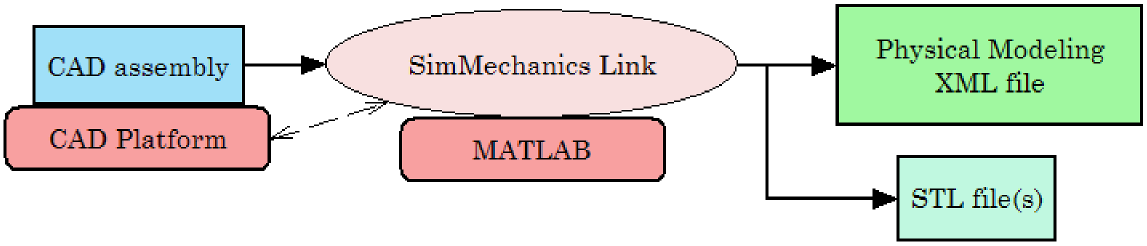
CAD Export: CAD Assembly to Physical Modeling XML

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:

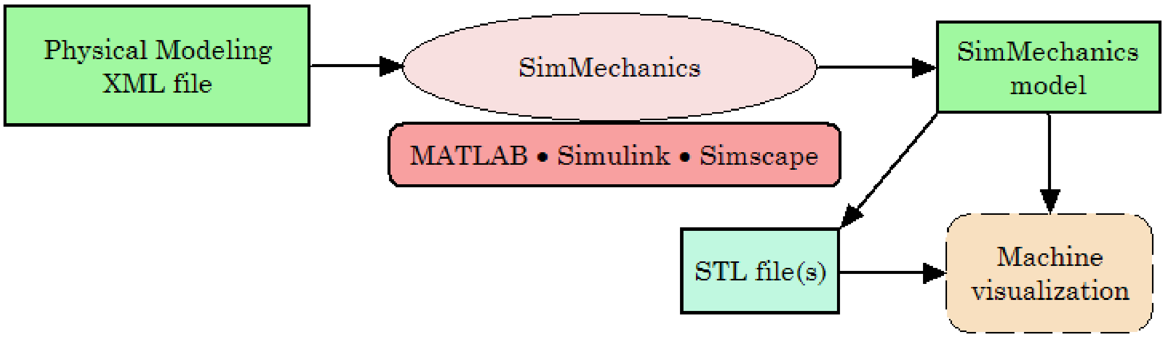
- A CAD platform or application
- MATLAB registered as a server
- The SimMechanics Link utility, installed and linked to your CAD platform

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.

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

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" on page 2-5

"Preparing a CAD Assembly for Translation into a SimMechanics Model" on page 2-7

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 Blocks
Part	Body
Constraints*	Joints
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

* Constraints on parts in a CAD assembly are sometimes 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.

See "Translating a CAD Robot Arm" on page 3-21 for an example of subassembly hierarchy.

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 each of two bodies 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.

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. (See “Subassemblies and Hierarchies” on page 2-6 for more on fixed parts.)
- 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 3-4 for an example.

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.

See "Translating a CAD Robot Arm" on page 3-21 for an example of configuring constraints.

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” on page 2-9

“Configuring the Exporter Controls in Supported CAD Platforms” on page 2-10

“Creating the XML File” on page 2-11

“Re-Exporting an Assembly After Changes” on page 2-11

“Exporting a Robot Arm Assembly for the First Time” on page 2-12

“Re-Exporting a Robot Arm Assembly After Changes” on page 2-14

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, “Creating Custom Links to 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. Geometrical and numerical differences smaller than the tolerances are treated as zero. The entries implicitly have the same units selected in your CAD platform settings.

- **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.

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

Complete export of the assembly to a Physical Modeling XML file with these steps.

1. If you changed the assembly or any subassemblies, you need to 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 finishes.

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

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 whatever directory 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 2-10.

Re-exporting an assembly after such changes is, by default, identical to the initial export steps. See "Creating the XML File" on page 2-11. With no changes to the XML file save procedure, your re-export overwrites the existing XML and STL files with the same names.

Tip You might want to preserve the older XML and STL files.

- You can use the old directory and give the re-exported XML file a new name. The exporter still overwrites the old STL files.
- You can use a new directory 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 2-17.

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 directory, open the robot arm assembly file, `robot.`*`ASSEMBLYFILETYPE`*, using a supported CAD platform linked to the SimMechanics Link utility.

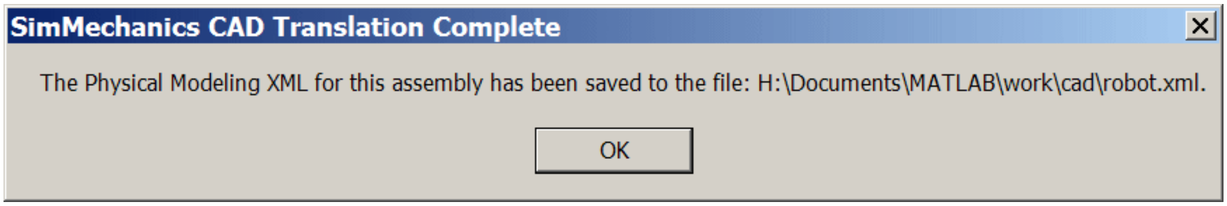
2 If you wish, open the settings pane or dialog box from the SimMechanics Link menu in your CAD interface. (See "Configuring the Exporter Controls in Supported CAD Platforms" on page 2-10.)

   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.

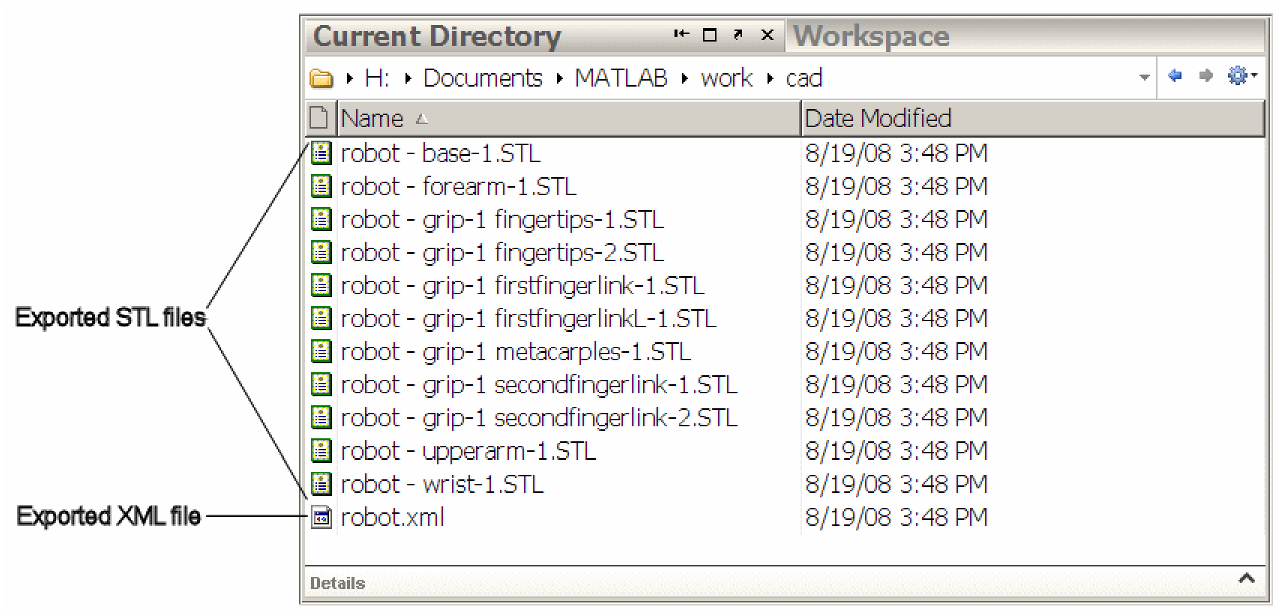
4 Change to a different directory to export the XML file. Leave the file name as the default, `robot.xml`.

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:


SimMechanics CAD Translation Complete

The Physical Modeling XML for this assembly has been saved to the file: H:\Documents\MATLAB\work\cad\robot.xml.

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



Name	Date Modified
robot - base-1.STL	8/19/08 3:48 PM
robot - forearm-1.STL	8/19/08 3:48 PM
robot - grip-1 fingertips-1.STL	8/19/08 3:48 PM
robot - grip-1 fingertips-2.STL	8/19/08 3:48 PM
robot - grip-1 firstfingerlink-1.STL	8/19/08 3:48 PM
robot - grip-1 firstfingerlinkL-1.STL	8/19/08 3:48 PM
robot - grip-1 metacarples-1.STL	8/19/08 3:48 PM
robot - grip-1 secondfingerlink-1.STL	8/19/08 3:48 PM
robot - grip-1 secondfingerlink-2.STL	8/19/08 3:48 PM
robot - upperarm-1.STL	8/19/08 3:48 PM
robot - wrist-1.STL	8/19/08 3:48 PM
robot.xml	8/19/08 3:48 PM

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.

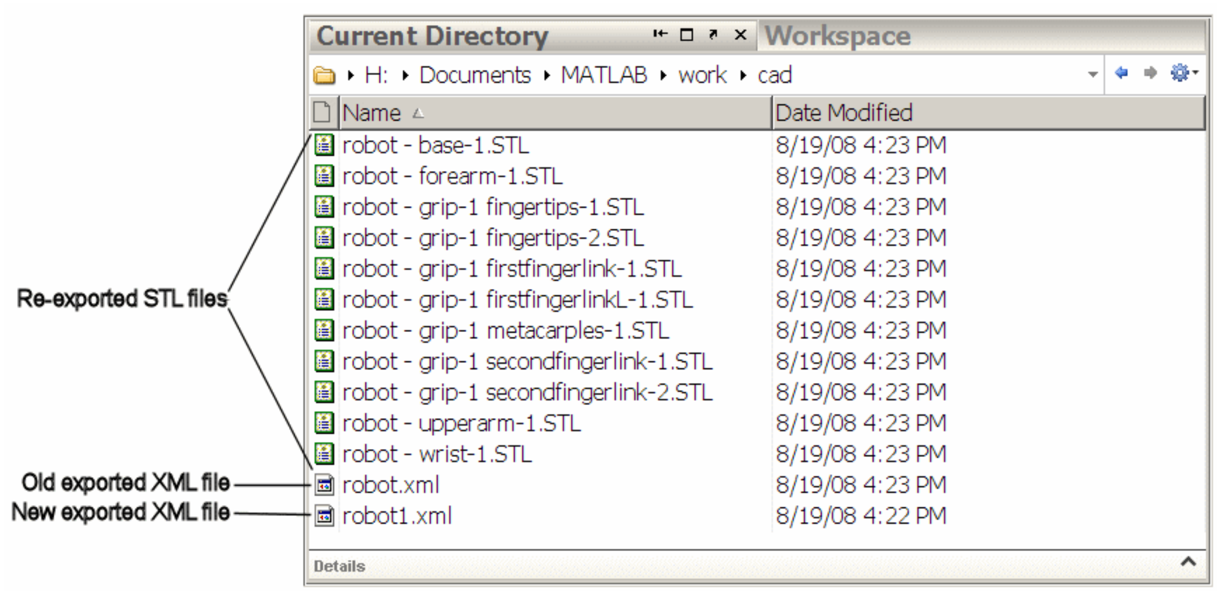
- These changes can include modifications of the assembly or subassembly parts.
- They can also include modifications to the export settings. (See “Configuring the Exporter Controls in Supported CAD Platforms” on page 2-10.)

With the assembly open from your CAD platform interface:

1. Open the export interface from the SimMechanics Link menu. Remain in the same directory 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 directory as the original export. The exporter overwrites the old stereolithographic files with new ones, named with the same names.



Name	Date Modified
robot - base-1.STL	8/19/08 4:23 PM
robot - forearm-1.STL	8/19/08 4:23 PM
robot - grip-1 fingertips-1.STL	8/19/08 4:23 PM
robot - grip-1 fingertips-2.STL	8/19/08 4:23 PM
robot - grip-1 firstfingerlink-1.STL	8/19/08 4:23 PM
robot - grip-1 firstfingerlinkL-1.STL	8/19/08 4:23 PM
robot - grip-1 metacarples-1.STL	8/19/08 4:23 PM
robot - grip-1 secondfingerlink-1.STL	8/19/08 4:23 PM
robot - grip-1 secondfingerlink-2.STL	8/19/08 4:23 PM
robot - upperarm-1.STL	8/19/08 4:23 PM
robot - wrist-1.STL	8/19/08 4:23 PM
robot.xml	8/19/08 4:23 PM
robot1.xml	8/19/08 4:22 PM

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

Troubleshooting Export Problems

In this section...

"Troubleshooting Assembly Export Failures" on page 2-16

"Getting Exporter Help from the CAD Platform Interface" on page 2-16

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 examines 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 map one or more constraints into joints, it issues one or more error dialog boxes and logs the errors into a text file. The error dialog indicates the name of this error log file, which is located in the same directory 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. The constraints that failed to export properly are converted instead into rigid welds. These errors reappear as MATLAB command line warnings if you generate a SimMechanics model from the XML file.

Getting Exporter Help from the CAD Platform Interface

Independent of details, 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.

Retranslating from Assembly to Existing Generated Model

In this section...

"Working with Associativity in Common Updating Situations" on page 2-17

"Understanding Associativity in General" on page 2-20

Working with Associativity in Common Updating Situations

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. See "Understanding Associativity in General" on page 2-20.

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.

These unique parallel identities 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.

CAD Assembly–SimMechanics Model Parallel Identities Captured by Associativity

CAD Assembly Components Correspond to...	...Imported SimMechanics Model Components
Parts and grounds	Body and Ground blocks
Constraints between parts (allowed motions)	Joint blocks
Constraints between parts (positions and orientations)	Coordinate systems attached to Joints
Subassembly hierarchy	Subsystem hierarchy

The following translation cases are the basic possibilities. It is possible to 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 of the types listed in the first column of the table, CAD Assembly–SimMechanics™ Model Parallel Identities Captured by Associativity on page 2-18. When you import the XML and generate a SimMechanics model from it, the corresponding model components listed in the 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, you can reuse an existing SimMechanics model previously translated from the same assembly by updating it.

Modifying a CAD assembly means changing the properties of its associated components of the types listed in the first column of the table, CAD Assembly–SimMechanics™ Model Parallel Identities Captured by

Associativity on page 2-18, 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, you can reuse an existing SimMechanics model previously translated from the same CAD assembly by updating it with the extensions.

Extending a CAD assembly means adding associated components of the types listed in the first column of the table, CAD Assembly–SimMechanics™ Model Parallel Identities Captured by Associativity on page 2-18. 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 leaves them unchanged 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 remain unchanged.

- However, 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 thus cannot be associated.

Understanding Associativity in General

Associativity consists of defined relationships between certain components of CAD assemblies and certain components of SimMechanics models based on them. For practical examples, see "Working with Associativity in Common Updating Situations" on page 2-17.

What Is Associativity?

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).

In some ways, associativity is symmetric between a CAD assembly and a SimMechanics model generated from it. In other ways, associativity is not symmetric, because the translation process moves in one direction only, from CAD assembly to SimMechanics model.

When and Why Associativity Is Needed

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:

- Parts
- Constraints (mates)
- Subassemblies

When you use the SimMechanics importer to generate a SimMechanics model from the XML file, the identifiers are preserved in these SimMechanics model features:

- Bodies (generated from CAD parts)
- Joints (generated from CAD constraints)
- Body coordinate systems connected to the generated Joints
- Subsystems (generated from CAD subassemblies)

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 thus 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.

Computer-Aided Design Translation

These case studies illustrate in detail how to translate mechanical systems defined externally, as computer-aided design (CAD) assemblies, into mechanical models.

- “Introducing the Case Studies” on page 3-2
- “Translating a CAD Part into a Body” on page 3-4
- “Translating CAD Constraints into Joints” on page 3-8
- “Translating a CAD Robot Arm” on page 3-21
- “Translating a CAD Stewart Platform” on page 3-28

Introducing the Case Studies

Requirements for CAD Translation The following CAD assembly and exporting examples require the SolidWorks CAD platform. To complete all the steps, you also need SimMechanics Link and SimMechanics software.

You can recreate these examples with other CAD platforms. The assembly, geometric, kinematic, and part details differ from platform to platform. In SolidWorks, constraints on CAD parts are called *mates*.

The case studies of this chapter illustrate the stages of converting 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 3-4 presents the simplest case, translating an assembly with a single part or body.
- "Translating CAD Constraints into Joints" on page 3-8 examines in detail how you translate assemblies with constrained parts into bodies with degrees of freedom represented by SimMechanics Bodies and Joints.
- "Translating a CAD Robot Arm" on page 3-21 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 3-28 illustrates the translation of a moderately complex assembly with many degrees of freedom, subassembly-subsystem hierarchy, and visualization of its motion.

For More About CAD Assembly Export

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

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.

See the *SimMechanics Visualization and Import Guide* for complete information about mechanical import and model generation and editing.

For More About Simulink and SimMechanics Blocks and Commands

For SimMechanics, consult the block and command reference of the SimMechanics documentation.

For Simulink, consult the block and function reference of the Simulink documentation.

Defining Specialized CAD Terms

The Glossary of the SimMechanics documentation defines specialized terms used in the context of computer-aided design (CAD).

Translating a CAD Part into a Body

In this section...

"Locating the Single-Part Assembly Files" on page 3-4

"Viewing the CAD Assembly" on page 3-4

"Exporting the CAD Assembly" on page 3-6

"Generating the SimMechanics Model" on page 3-6

Locating the Single-Part Assembly Files

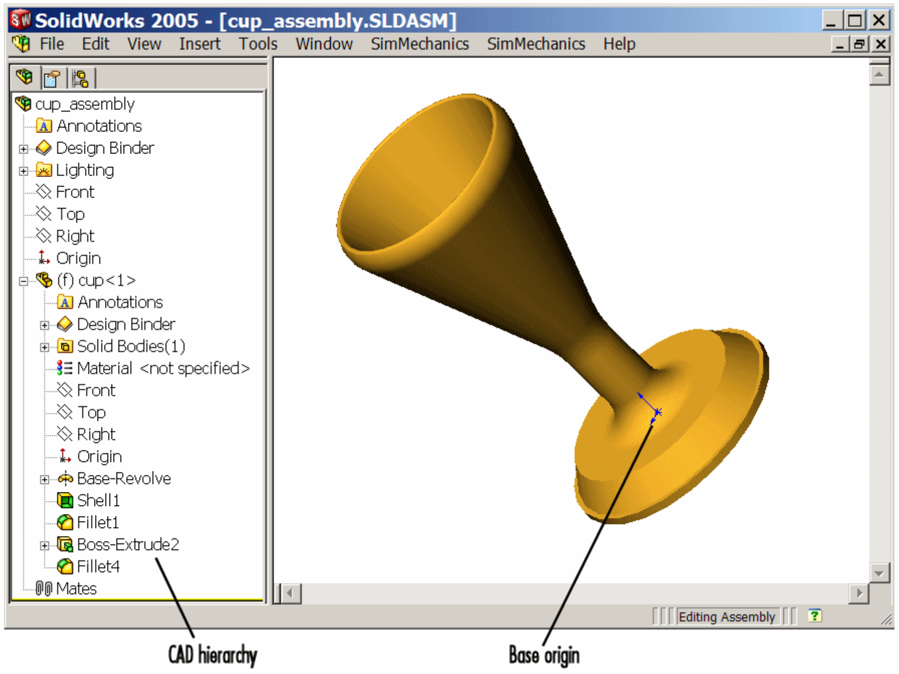
In this example, you export an assembly with one part and no constraints. Locate the two example CAD files in the SimMechanics Link demos directory:

- 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 3-5. The *x*-axis is the cup's axis of symmetry, and the *y*- and *z*-axes point across the cup.

Exporting the CAD Assembly

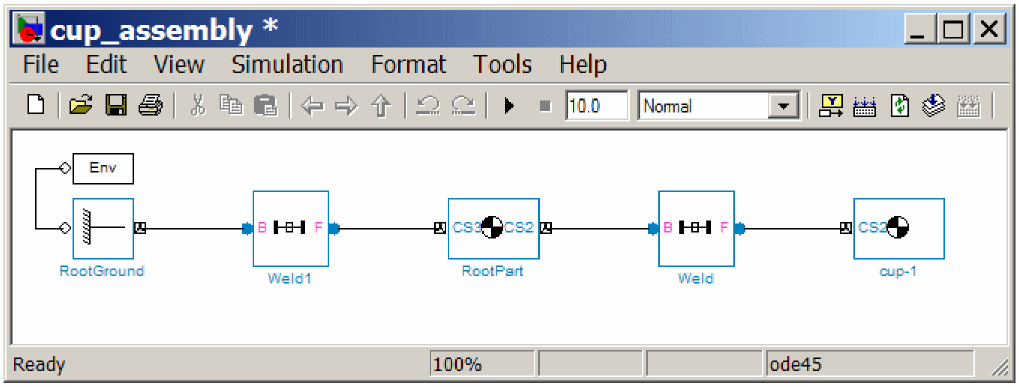
Using the SimMechanics Link interface to your CAD platform, export the assembly into Physical Modeling XML format. The XML file `cup_assembly.xml` appears in your working CAD directory.

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 directory 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 Body 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" on page 3-8

"Locating the Constraint Assembly Files" on page 3-9

"Generating the Two-Part Models: Common Steps" on page 3-10

"Understanding the Two-Part Models: Common Block Structure" on page 3-10

"Modeling a Six-DoF Joint" on page 3-11

"Modeling a Prismatic Joint" on page 3-12

"Modeling a Revolute Joint" on page 3-16

"Modeling an Inplane Joint" on page 3-17

"Modeling a Spherical-Spherical Massless Connector" on page 3-18

Modeling CAD and SimMechanics Degrees of Freedom

In "Translating a CAD Part into a Body" on page 3-4, you create and export an assembly composed of a single part. CAD translation converts the part into a body, with a mass, an inertia tensor, and Body coordinate systems (CSs). 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. Each example assembly consists of two instances of the same part file, representing two identical cubes. The study shows how to find the needed files, presents the essential steps for generating models from them, and discusses the structure common to all the generated models. It then proceeds to specific assembly cases.

In different assemblies, the two cubes are constrained with different constraint combinations to create different relative DoFs between the cubes. You can typically represent a set of DoFs with a large number of different constraint combinations. Each constraint combination here, in general, is *not* the unique way to create the corresponding set of DoFs. These assembly examples include:

- Two cubes with no constraints, so that the cubes have the full six degrees of freedom relative to one another
- Two cubes constrained in two different ways so as to produce the same result, a single prismatic (translational) DoF between them
- Two cubes constrained so as to allow only a single revolute (rotational) DoF between them
- Two cubes constrained so as to allow two prismatic (translational) DoFs between them
- Two cubes constrained so as to allow relative spherical joint motion, with the two cubes separated by a constant nonzero distance

Locating the Constraint Assembly Files

Locate the CAD assembly files for this study in the SimMechanics Link demos directory. 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 steps for exporting a two-part assembly and generating SimMechanics models based on it are 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`.

   In the CAD hierarchy, note 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 this CAD assembly into Physical Modeling XML. The XML file is saved in your current working CAD directory.

Generating a Model

You can generate a SimMechanics model based on this assembly.

1. Move or copy the XML file to a working MATLAB directory. Then open MATLAB in that directory.

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 "Understanding the Two-Part Models: Common Block Structure" on page 3-10. A set of Joints represents the DoFs between the two cubes.

Understanding the Two-Part Models: Common Block Structure

All the models that you generate in this study from the example CAD assemblies have a common structure because each assembly has a fundamental root and two moving parts. Each model has eight blocks.

- *The assembly's fundamental root.* As in any generated CAD-based model, the four-block combination Machine Environment – Root Ground – Root Weld – Root Part represents the assembly's fundamental root. The Root Part is a nonmoving, zero-mass/zero-inertia body.
- *The moving bodies.* The bodies representing the assembly's 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 question 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 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.

Restricting Degrees of Freedom with Constraints

CAD platforms normally 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. There is always one assembly part 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 such an assembly:

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

In the CAD hierarchy, the constraints (Mates) node has no entries. Therefore, relative to one another, the cubes are unconstrained in their motion and have six relative DoFs.

2 Export this CAD assembly into `sixDOF.xml`.

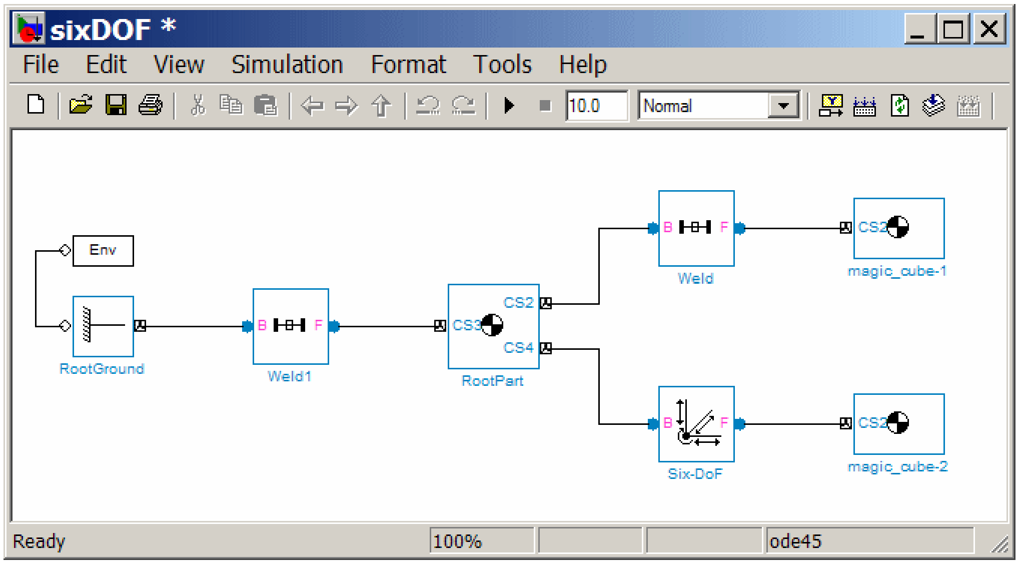
Generating the Model

To generate a model based on this 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 accomplish this; you can experiment with other constraints to find more. In the translated SimMechanics models, this single DoF is a prismatic joint.

Prismatic as a Planar Constraint and a Cylindrical Constraint

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

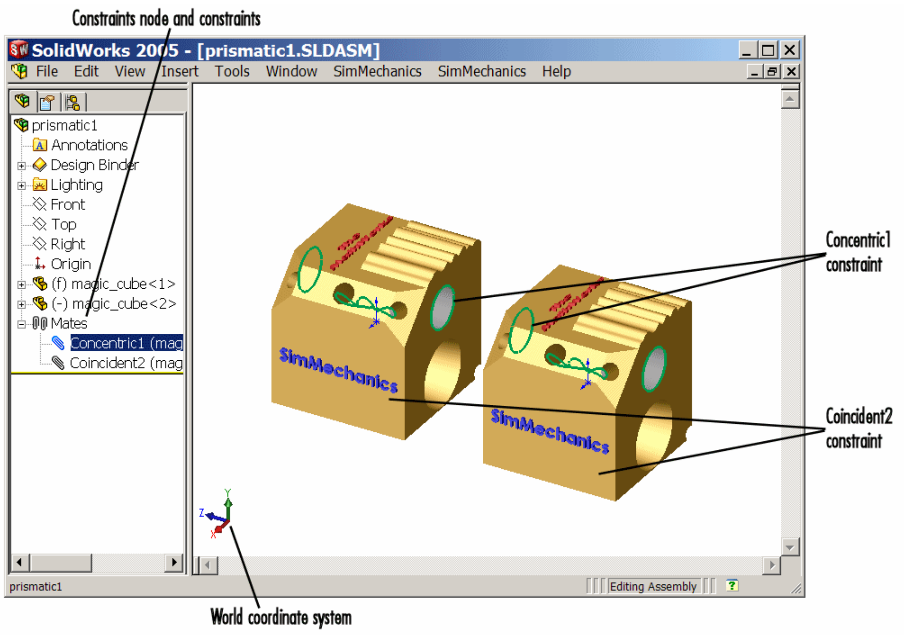
1. Open the assembly file `prismatic1.`*`ASSEMBLYFILETYPE`* and examine the CAD hierarchy.

2. Locate and expand the constraints (Mates) node. There are two constraints on the two cubes.

   - Highlight the first constraint, Concentric1. The constraint geometry is highlighted in the assembly. This constraint allows the two cubes only 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 second constraint, Coincident2. The constraint geometry is highlighted 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.

These two constraints mean that 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.



Planar and Cylindrical Constraints on Two Cubes

Prismatic as Two Orthogonal Planar Constraints

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

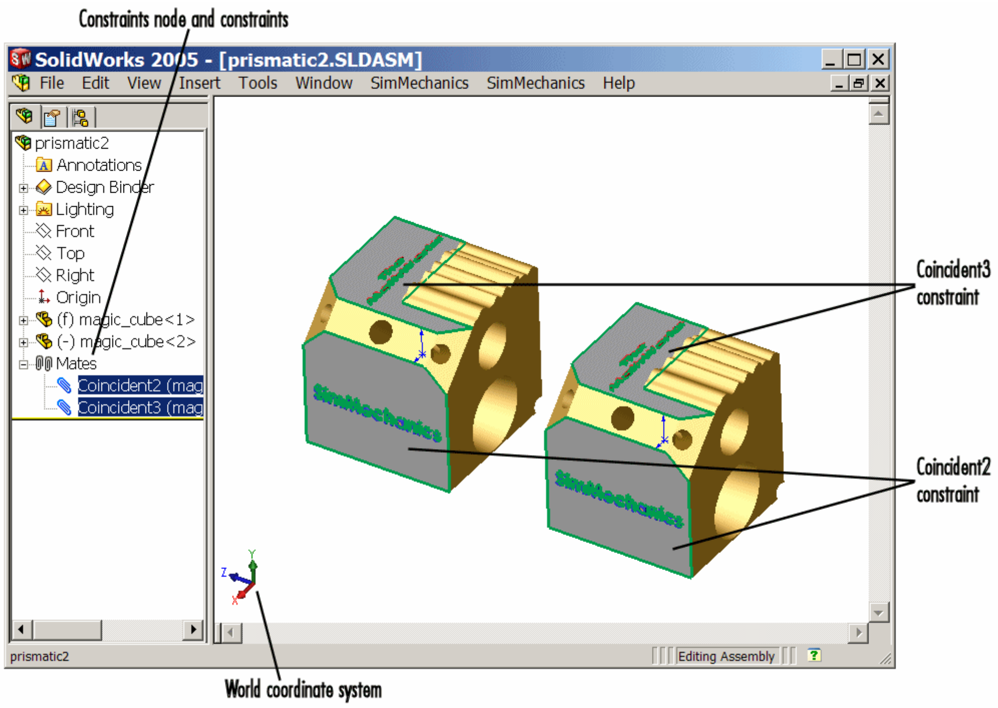
1 Open the assembly file `prismatic2.`*`ASSEMBLYFILETYPE`* and examine the CAD hierarchy.

2 Locate and expand the constraints (Mates) node. There are two constraints on the two cubes.

- Highlight the first constraint, Coincident2. The constraint geometry is highlighted 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 constraint, Coincident3. The constraint geometry is highlighted 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. It 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.

These two constraints mean that 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

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 directory. 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 the CAD hierarchy.

2 Locate and expand the constraints (Mates) node. There are two constraints on the two cubes.

- Highlight the first constraint, Concentric1. The constraint geometry is highlighted 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 second constraint, Coincident1. The constraint geometry is highlighted in the assembly. This constraint allows the two cubes to slide along the *x*-*y* plane, with the parallel sides sharing a common plane. It 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.

These two constraints mean that 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 this assembly:

1. Export the assembly as `revolute.xml`. Copy or move it to a MATLAB working directory.

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 the CAD hierarchy.

2. Locate and expand the constraints (Mates) node. There are two constraints on the two cubes.

   - Highlight the first constraint, Coincident2. The constraint geometry is highlighted 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. 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 constraint, Parallel1. The constraint geometry is highlighted 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.

These two constraints mean that 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 this assembly:

1. Export the assembly as `inplane.xml`. Copy or move it to a MATLAB working directory.

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:

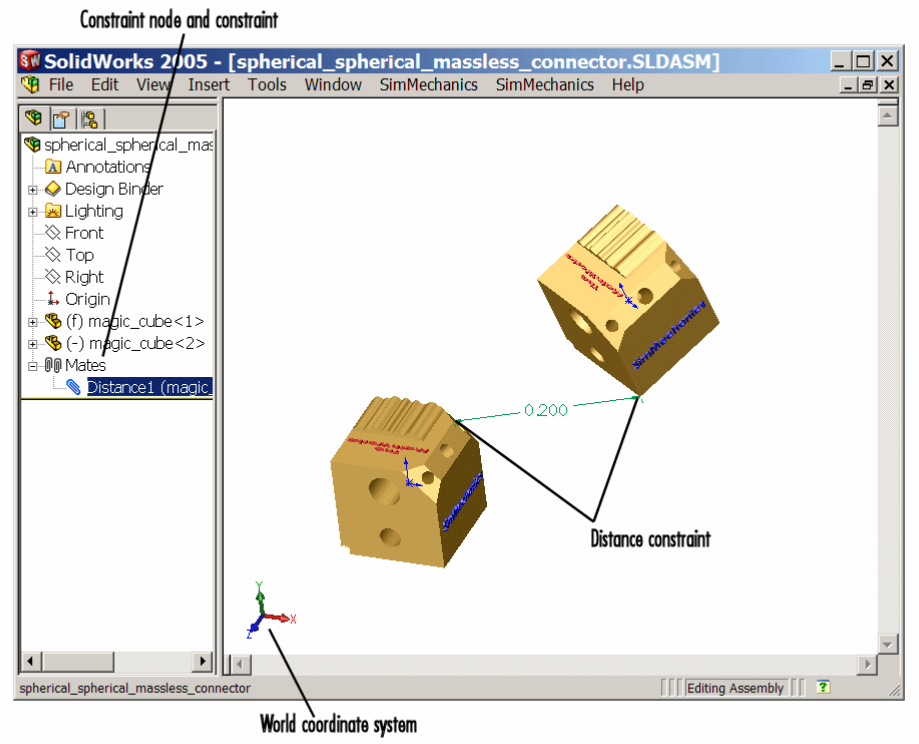
1. Open the assembly file

   ```
   spherical_spherical_massless_connector.ASSEMBLYFILETYPE
   ```

and examine the CAD hierarchy.

2 Locate and expand the constraints (Mates) node. There is one constraint on the two cubes.

Highlight this constraint, Distance1. The two spherical pivot points are highlighted as small red or green squares, one on each cube. These points are the endpoints of the rigid massless connector. The cubes can move such that the distance between these two points (the length of the massless connector) does not change. The constraint allows the two cubes to pivot independently about their connector endpoints.


Constraint node and constraint
SolidWorks 2005 - [spherical_spherical_massless_connector.SLDASM]
File Edit View Insert Tools Window SimMechanics SimMechanics Help
spherical_spherical_mas
Annotations
Design Binder
Lighting
Front
Top
Right
Origin
(f) magic_cube<1>
(-) magic_cube<2>
Mates
Distance1 (magic
0.200
Distance constraint
spherical_spherical_massless_connector
Editing Assembly
World coordinate system

Distance Constraint on Two Cubes

Exporting the Assembly and Generating the Model

To generate a model based on this assembly:

1. Export the assembly as

   ```
   spherical_spherical_massless_connector.xml
   ```

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

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 “Understanding the Two-Part Models: Common Block Structure” on page 3-10.

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.

Translating a CAD Robot Arm

In this section...

"Locating the Robot Arm Assembly Files" on page 3-21

"Viewing the Robot Arm Assembly" on page 3-22

"Exporting the Robot Arm Assembly" on page 3-23

"Generating and Completing the Robot Arm Model" on page 3-23

"Simulating and Observing the Robot Arm Motion" on page 3-27

Locating the Robot Arm Assembly Files

The example of this study is based on a more complex CAD assembly, a robot arm. It includes multiple parts, two closed loops, multiple constraints, and a subassembly.

Locate the 11 CAD files for the robot arm in the SimMechanics Link demos directory. They are

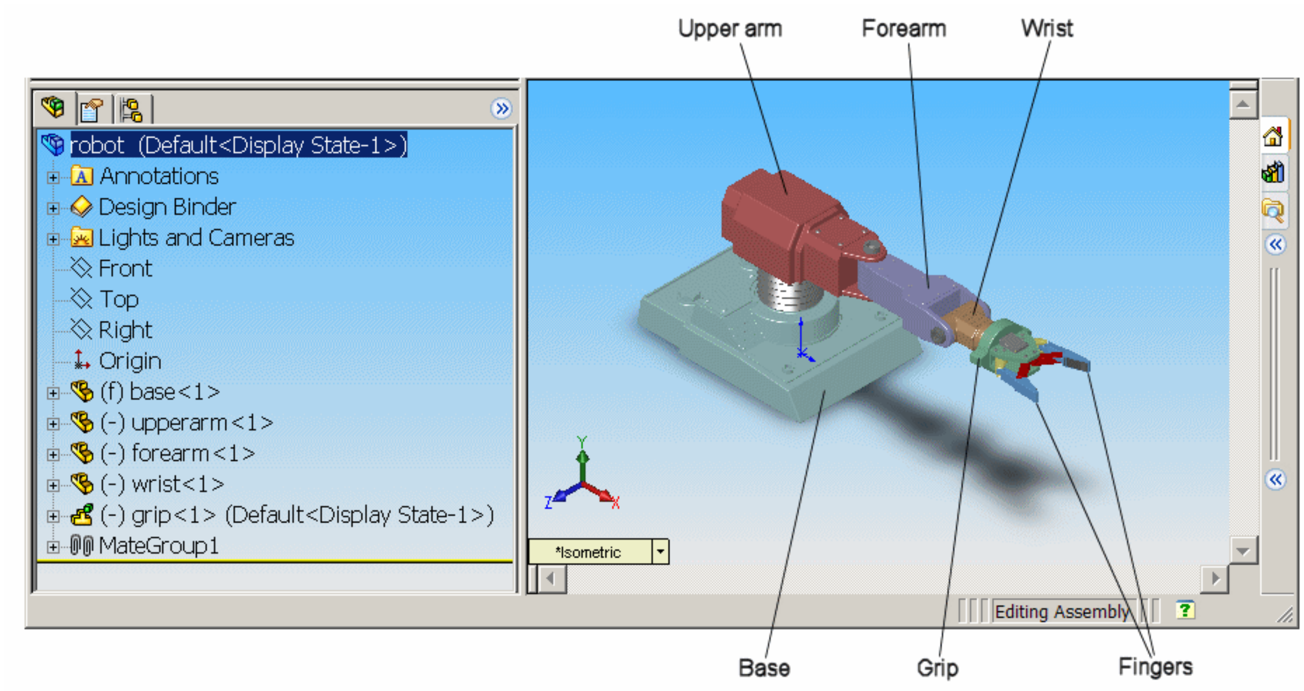
File Name	CAD Filetype
`robot.`*`ASSEMBLYFILETYPE`*	Assembly
`grip.`*`ASSEMBLYFILETYPE`*	Subassembly (flexible)

`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

Then 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 directory.

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 directory.

Generating the Initial Model

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

1. Generate the model by entering at the command line

   ```
   mech_import('robot')
   ```

   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:

  - 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 SphericalSpherical Joints configured with the proper primitives to represent the original CAD assembly's DoFs.

- The first non-Weld Joint encountered 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.

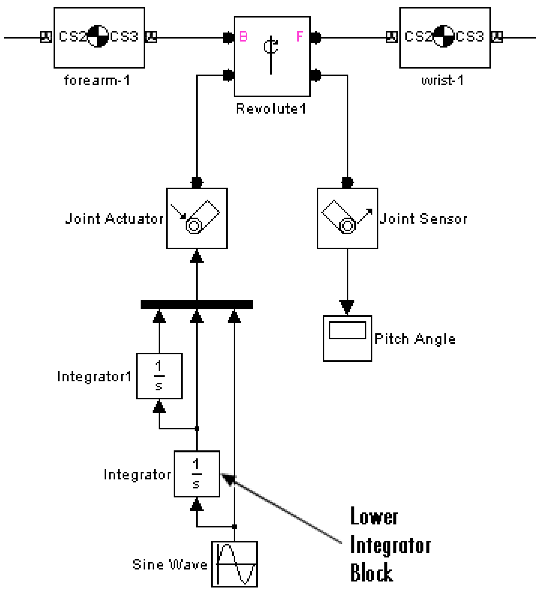
Adding an Actuator and a Sensor

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 unchanged.

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

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" on page 3-28

"Introducing the Stewart Platform Assembly" on page 3-28

"Viewing the Stewart Platform Assembly" on page 3-29

"Exporting the Stewart Platform Assembly" on page 3-30

"Generating the Stewart Platform Model" on page 3-30

"Visualizing the Stewart Platform Motion" on page 3-33

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

This study uses a complex computer-aided design (CAD) assembly that models the Stewart platform.

Note The Stewart platform assembly of this study 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*.

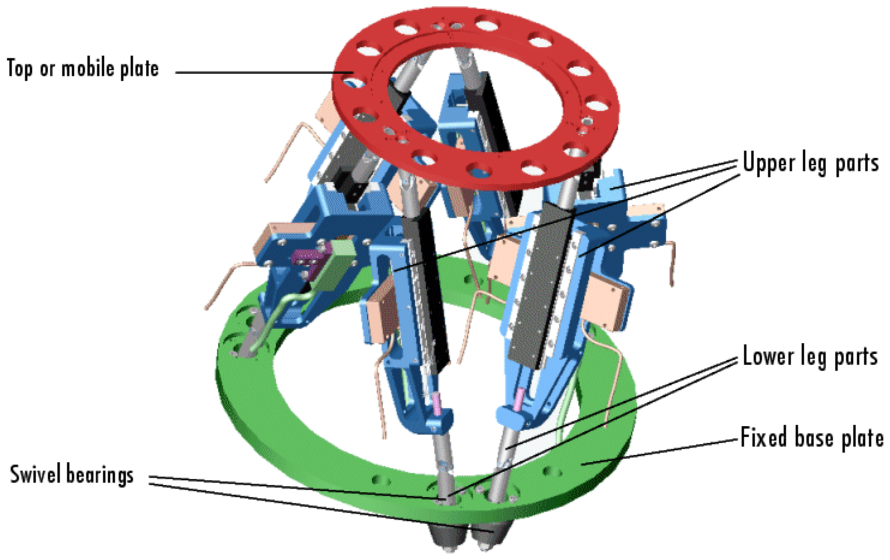
Locating the Stewart Platform Assembly Files

Locate the 45 CAD files for the Stewart platform in the SimMechanics Link demos directory. 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 the XML file `stewart_platform.xml` in your working CAD directory.

Generating the Stewart Platform Model

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

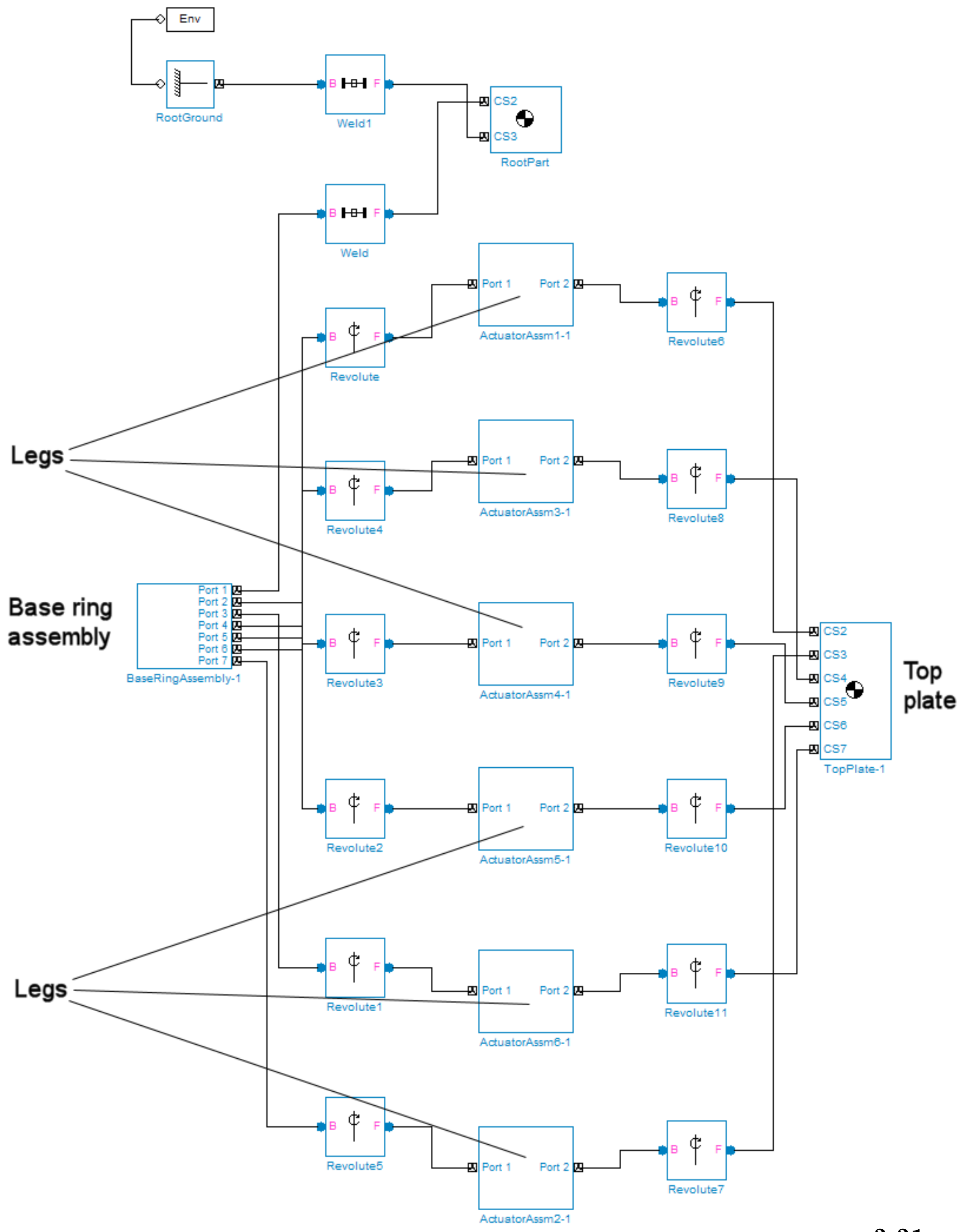
To generate a new Simulink model, enter at the MATLAB command line

```
mech_import('stewart_platform')
```

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, if 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 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.

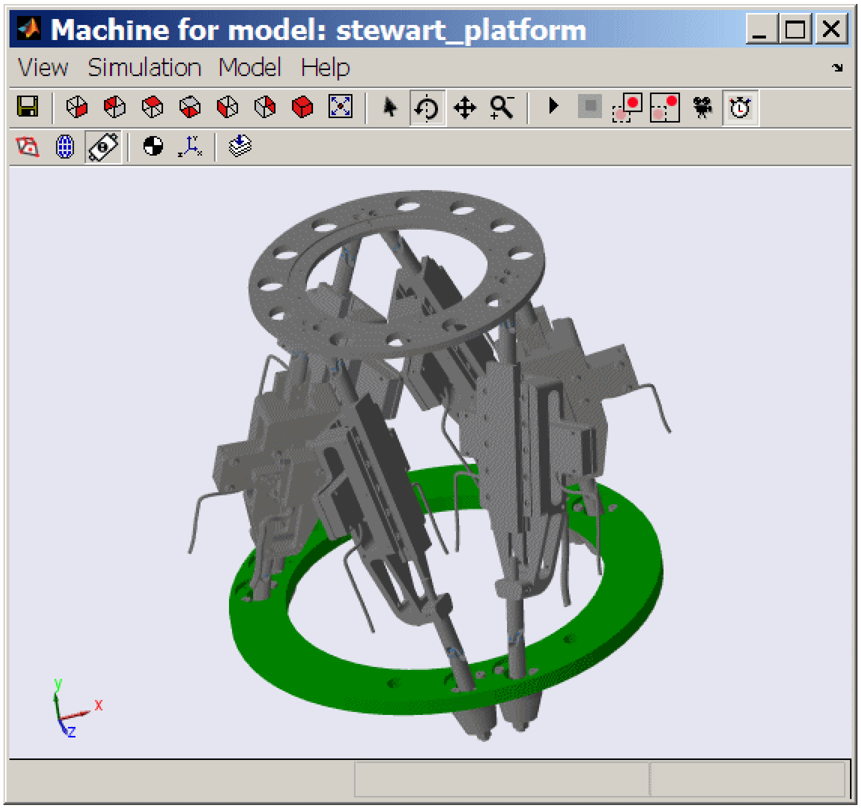
2 Select **Display machines after updating diagram** and **Show animation during simulation**. Click **Apply** or **OK**.

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.


Machine for model: stewart_platform
View Simulation Model Help
y
x
z

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

Creating Custom Links to External Applications

The SimMechanics Link software includes an application program interface (API). Along with the API of an external application program, the SimMechanics Link API allows you to selectively transfer data from the external application to the SimMechanics Link software.

- "Custom Export with the SimMechanics Link API" on page 4-2
- "Understanding the Mapping of API Objects" on page 4-6
- "Designing Custom Exporter Modules" on page 4-10
- "Programming with the SimMechanics Link API" on page 4-13

Custom Export with the SimMechanics Link API

In this section...

"About CAD Translation with Custom Export" on page 4-2

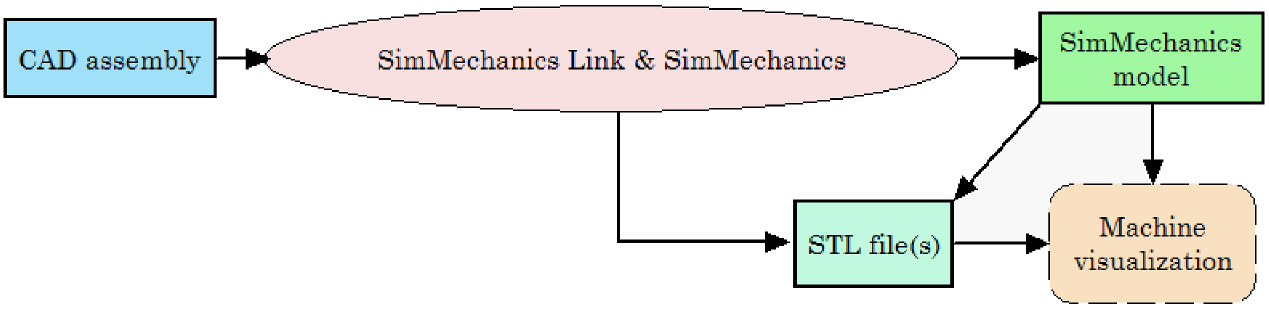
"What's the Same: Translation Steps in Common with Standard Export" on page 4-3

"What's Different: Translation Steps with Customized Export" on page 4-4

"Requirements for Creating a Custom Exporter" on page 4-4

About CAD Translation with Custom Export

The overall process of translation 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. 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.



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 is not supported by the SimMechanics Link utility.
- The standard export from a supported CAD platform does yield the results you want.

Requirements for CAD Translation with Custom Export

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

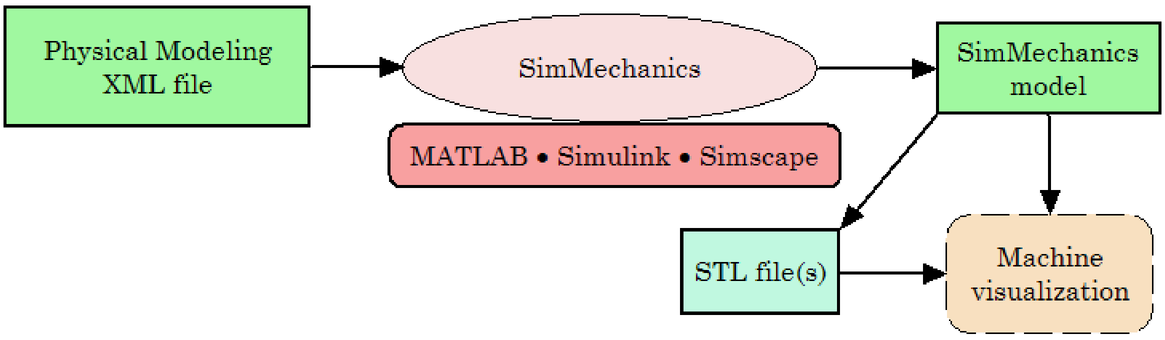
- A CAD platform or application, including access to its application program interface (API)
- MATLAB installed and registered as a server
- The SimMechanics Link utility, which includes an API
- A *custom exporter module* that you create and that transfers assembly information from the CAD platform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

Import requires SimMechanics software, based on Simulink and Simscape software.

What's the Same: Translation Steps in Common with Standard Export

You generate SimMechanics models from Physical Modeling XML files using mechanical import and 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's bodies. This part of CAD translation is the same as the standard export case.

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

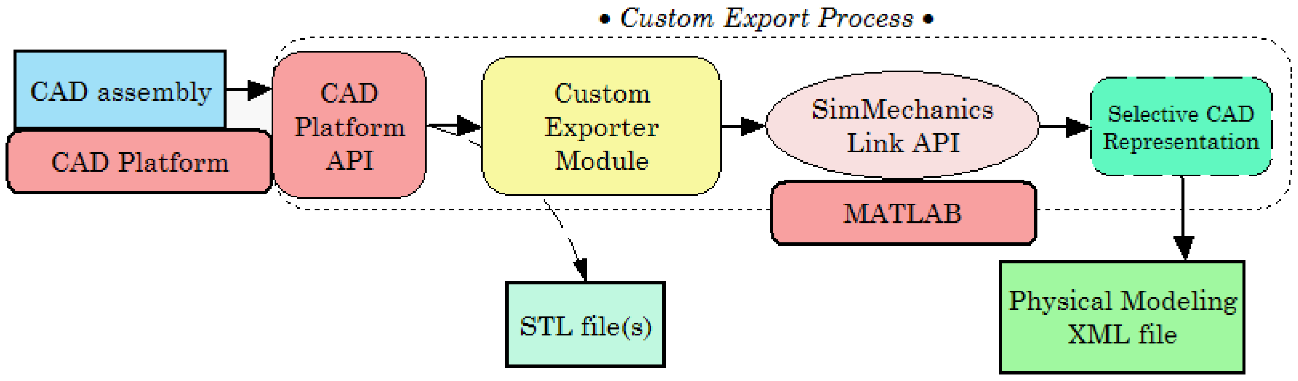


From Physical Modeling XML to Visualizable SimMechanics™ Model

What's Different: Translation Steps with Customized Export

In the custom export case, 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 temporary selective representation of the machine. It then finishes by creating the same files as the standard export does:

- A Physical Modeling XML file representing selected data needed to generate a SimMechanics model and written from the selective assembly representation
- A set of STL files to represent the surface geometries of the assembly's 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 supported by MATLAB.

To create a custom exporter, you need to:

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

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

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

Understanding the Mapping of API Objects

In this section...

"About Mapping API Objects from CAD Format to Physical Modeling XML" on page 4-6

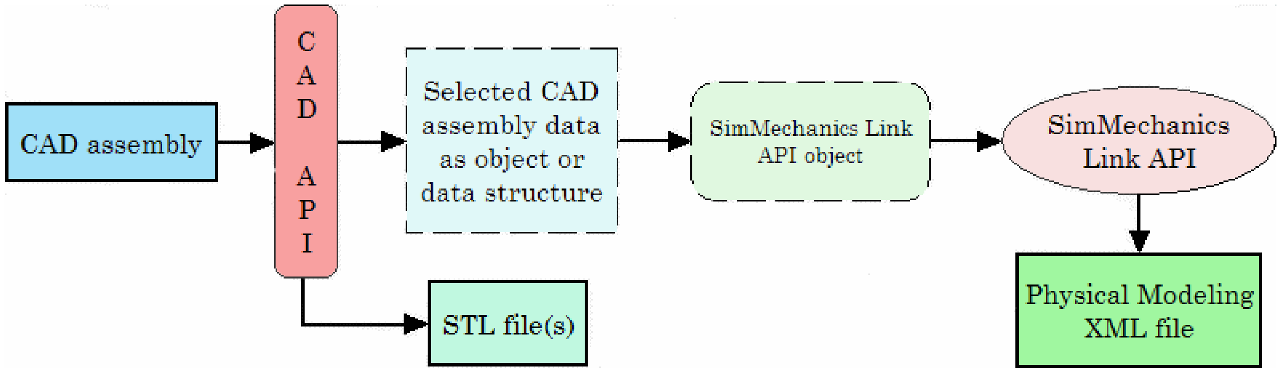
"Selecting CAD Assembly Data for Export" on page 4-8

"Constructing Selective CAD Representations" on page 4-8

"Converting Selective CAD Representations into Physical Modeling XML" on page 4-9

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 the assembly.


CAD assembly
CAD API
Selected CAD assembly data as object or data structure
SimMechanics Link API object
SimMechanics Link API
STL file(s)
Physical Modeling XML file

Mechanical Data Needed from a CAD Assembly

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

- Assembly and parts origin

- Assembly and subassembly hierarchy
- Part masses and inertia tensors, colors, and body geometry file names (needed by the generated SimMechanics model)
- Constraint restrictions on relative positions and motions of parts

Mechanical Data Needed to Construct a SimMechanics Model

The information required for a Physical Modeling XML is determined by the data needed for an imported SimMechanics model. The needed SimMechanics model data consist of:

- Ground and World coordinate system
- Model and subsystem hierarchy
- Bodies, including their mass properties, geometries, and colors
- Joints between pairs of Bodies
- Body coordinate systems connected to Joints

Relation of API Object Mapping to Exporter Functions

A CAD exporter module to create Physical Modeling XML transforms the selected CAD assembly data structure or object into a SimMechanics Link API object. You pass this object to the SimMechanics Link API to write the Physical Modeling XML file.

Body Geometry File 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 the 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

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 assembly ground.

For each component, you need to extract:

- 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)
- How its constraints 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 with respect to its parent or it can move

Constructing Selective CAD Representations

Before you can export the CAD assembly, you must construct a selective CAD representation, a collection of objects that capture the needed CAD data. These objects include these or these types:

- 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 the other objects

You construct a full CAD representation as follows.

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 this and the preceding steps 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 the assembly hierarchy.

Converting Selective CAD 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" on page 4-10

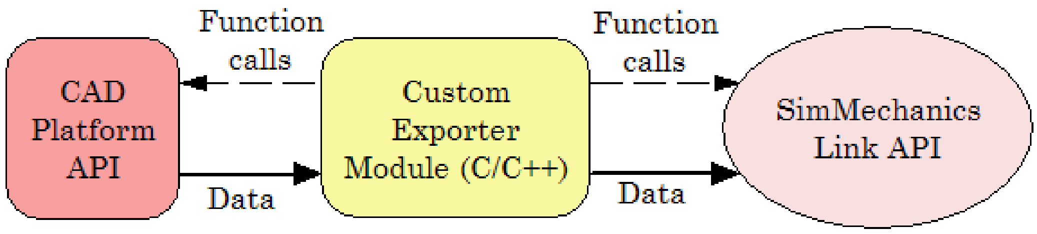
"Implementing Translation with the CAD and SimMechanics Link APIs" on page 4-11

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 "Understanding the Mapping of API Objects" on page 4-6.

As shown, 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, though 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 with the SimMechanics Link API" on page 4-13 describes how to create, compile, and execute custom exporter modules in C/C++ with the SimMechanics Link API function library.

Implementing Translation with the CAD and SimMechanics Link APIs

Extracting Selective CAD Assembly Data

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

Constructing the Selective CAD Representation

To construct the selective representation of an assembly, you need these functions from the SimMechanics Link API, in the order shown. The API also contains other functions you might need in special cases.

SimMechanics Link API Function	Purpose
`pmit_set_units`	Set length and mass units for CAD assembly
`pmit_set_tolerances`	Set translational, rotational, and numerical tolerances for CAD assembly
`pmit_create_cadmodel`	Create object to contain data for CAD assembly, subassembly, or part
`pmit_create_cadmodelref`	Create reference to CAD model object
`pmit_add_refincadmodel`	Add reference within existing CAD model object to child model object by adding transform
`pmit_create_assemcomp`	Create object to contain data for CAD assembly component (part or rigid subassembly)
`pmit_add_refincomp`	Add reference within existing CAD model object to child assembly component object
`pmit_create_constrain`	Create object to contain data for constraint between CAD components (parts or rigid subassemblies)
`pmit_add_constrain`	Add constraint object to CAD model object

Exporting the Selective CAD Representation into Physical Modeling XML

To export the selective representation of the assembly into a Physical Modeling XML file, you need these functions from the SimMechanics Link API, in the order shown.

SimMechanics Link API Function	Purpose
`pmit_create_cad2sm`	Enable translation of CAD assembly data into SimMechanics model
`pimt_write_xml`	Write Physical Modeling XML file from selective CAD representation of assembly

Programming with the SimMechanics Link API

In this section...

"Including, Linking to, and Running with the API Function Library" on page 4-13

"A Custom Exporter Module Example" on page 4-14

Including, Linking to, and Running with 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 (e.g., `win32` for 32-bit Windows).

Using and Including the API Header File

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

This header file contains the function definitions that you must follow to call the SimMechanics Link API functions from your module. You must 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, you also need to consult your CAD platform's API documentation to determine requirements for linking to the platform's API.

On Windows Platforms. When 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. When 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

In addition to including the MATLAB executable directory *`$matlabroot`*`/bin/`*`ARCH`*`/` on your system path, you also need to consult your CAD platform's API documentation to determine requirements, if any, for including the platform's API on the path.

On Windows. Before you start execution of the module, be sure the directory *`$matlabroot`*`\bin\`*`ARCH`*`\` is on the path.

On UNIX, LINUX, and Mac Platforms. Before you start execution of the module, be sure the directory *`$matlabroot`*`/bin/`*`ARCH`*`/` is included in the system path environment variable `LD_LIBRARY_PATH`.

A Custom Exporter Module Example

The directory `matlab/toolbox/physmod/smlink/api/example/cadapi_example/` contains a simple C++ example of a custom module based on the API, `cadapi_example.cpp`.

- This example custom module converts two CAD parts with one constraint into two SimMechanics bodies and one joint and 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's API. Compared to the figure, Custom Exporter Module Communicates with APIs (Detail) on page 4-10, the leftmost portion of the diagram is missing.
- To extend the example module to process real assembly data, you must add code that calls a CAD platform's API for assembly data to replace the internally defined data.

Index

A

API. *See* application program interface (API)
application program interface (API)
  and custom export 4-2
  for SimMechanics™ Link 4-2
  function library 4-13
  objects 4-6
  requirements for using 4-2
assembly 2-5
  CAD 2-5
  components 2-5
  designing for simulation 2-7
  exporting 2-9
  preparing for export 2-5
  See also computer-aided design; subassembly
associativity
  general properties of 2-20
  of exported assemblies with generated models 2-17

C

CAD. *See* computer-aided translation (CAD)
computer-aided design (CAD)
  case studies 3-2
  translation 2-2
constraints
  in computer-aided design 3-8

D

demo models
  running 1-11

E

export
  and re-export 2-11
  custom 4-2
  custom module 4-10
  defined 2-3
  getting online help 2-16
  of CAD assemblies 2-9
  requirements 2-3
  troubleshooting errors 2-16
extensible markup language (XML)
  exporting 2-9
  importing 2-4
  in Physical Modeling format 2-3

G

generated models. *See* import

I

import
  defined 2-4
  generating SimMechanics™ models 2-4
  requirements 2-4

P

parts
  in computer-aided design 3-4

R

robot arm
  in computer-aided design 3-21

S

stereolithographic (STL) files
  custom exporting 4-6
  defined 2-3
  exporting 2-9
  referenced by generated SimMechanics™
    models 2-4
Stewart platform
  in computer-aided design 3-28
STL. *See* stereolithographic (STL) files
subassembly
  and SimMechanics™ subsystems 2-5
  hierarchy 2-5

T

translation
  defined 2-2
  requirements 2-2

U

updated models. *See* export

X

XML. *See* extensible markup language (XML)