Introductory Tutorial on SimMechanics

Creation of a Fourbar Mechanism

Colin Lea
University at Buffalo
Automation, Robotics, and Mechatronics lab
Summer 2009

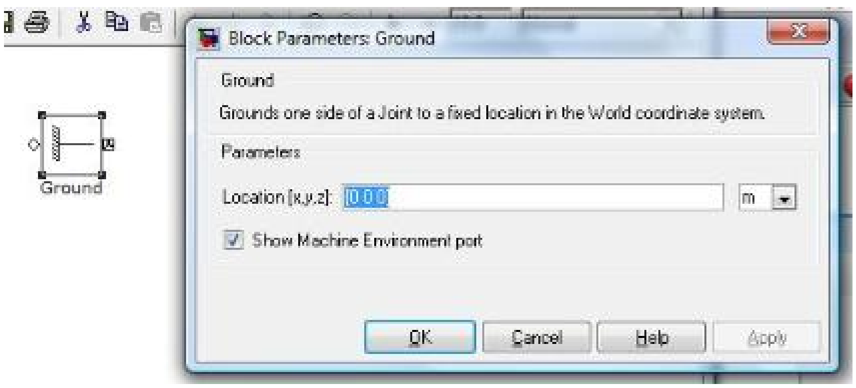
SimMechanics

SimMechanics allows for creation of rigid bodies for dynamic simulation. It is possible to create a collection of rigid bodies and joints with actuators and sensors without having to develop the kinematic equations. I will start off with an overview of how to create a SimMechanics model. All parts can be found in the SimMechanics folder under Simscape in the Simulink Library. Additional information and tutorials can be found in the Simulink/SimMechanics Help.

Description of Parts:

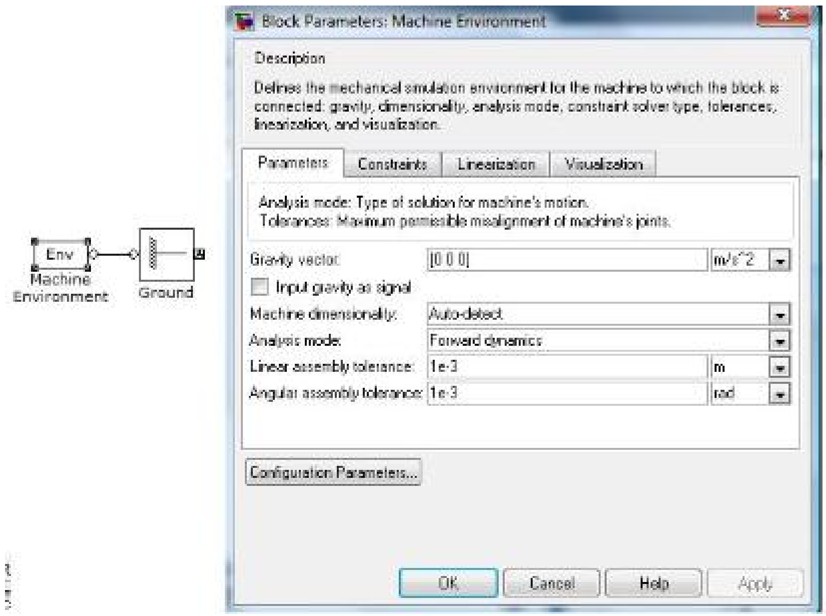
Ground Block

Every mechanism must start with a ground block. This acts as an origin for the rest of the bodies. Parameters for this block are the X,Y, and Z coordinates.



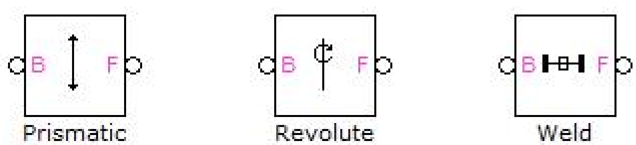
Machine Environment Block

Every model needs at least one Machine Environment block connected to a ground. It is possible to modify gravity and tolerances for the model as well as set the type of analysis. This block must connect to the back of a ground block. In order to make the connection the “Show Machine Environment port” on the ground block must be checked.

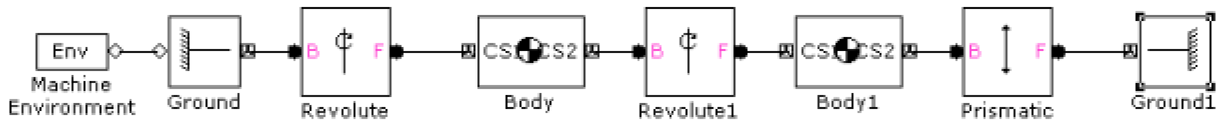


Joints

Many types of connections are included under the *Joints* dropdown within the Simulink Library. The most basic ones are *Revolute*, *Prismatic* and a *Weld*. A revolute joint allows one degree of rotational freedom. An example is if you have two pieces of metal pinned together with one screw. A prismatic joint allows one degree of translational freedom. For example if you have one cylinder fit inside a larger cylinder allowing it to move in one axis. A weld is fixed and allows no degrees of freedom.

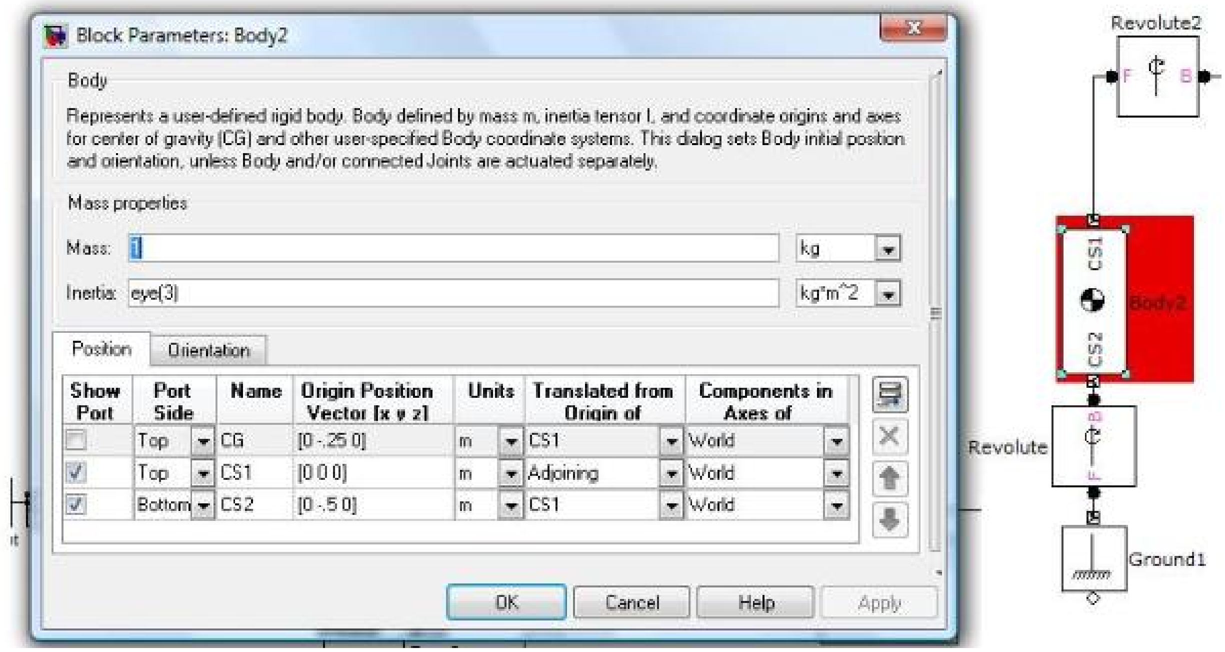


For a prismatic and revolute joint it is possible to change the direction of the degree of freedom within the parameters. It should be noted that joints have direction. For an open loop (no end connection) the ‘B’ (base) should be closest to the ground and the ‘F’ (follower) should be going towards the end effector. For a closed system the general direction doesn’t matter but it must be consistent.



Bodies

Bodies generally include at least 3 link parameters: a port on both sides and a center of gravity. The ports are denoted as CS[#] within the block parameters. They represent different points on the body. It is possible to describe these in relative or absolute coordinates. To use absolute coordinates chose ‘World’ under ‘Translated from Origin of.’ To use relative pick ‘Adjoining’ or reference another coordinate system (CS). In my experience if you are learning or playing around relative is more convenient because you don’t have to know the actual point in space. Generally I choose the Base link as ‘Adjoining’ with an origin of [0, 0, 0] and the rest of the links relative to that CS.



It is also possible to change the orientation of each coordinate system using the ‘Orientation’ tab. If you use this you may want to change the ‘Components in Axes of’ parameter under ‘Position.’

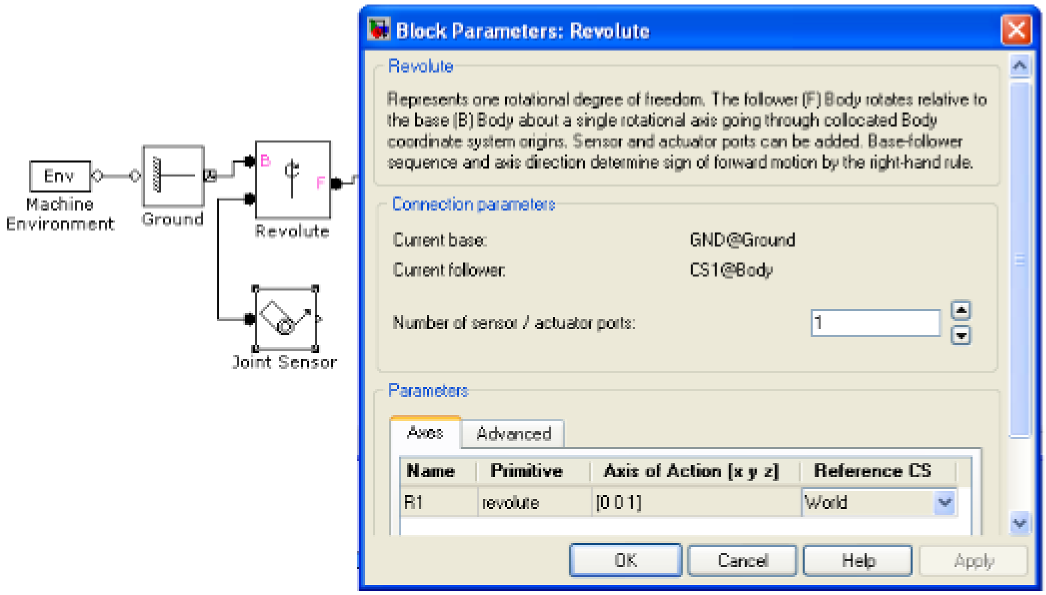
Mass and Inertia are also parameters. Inertia is in the form of a tensor matrix. In order to connect two bodies or a body to a ground you must use some type of joint.

Sensors

There are two main types of sensors: Joint and Body.

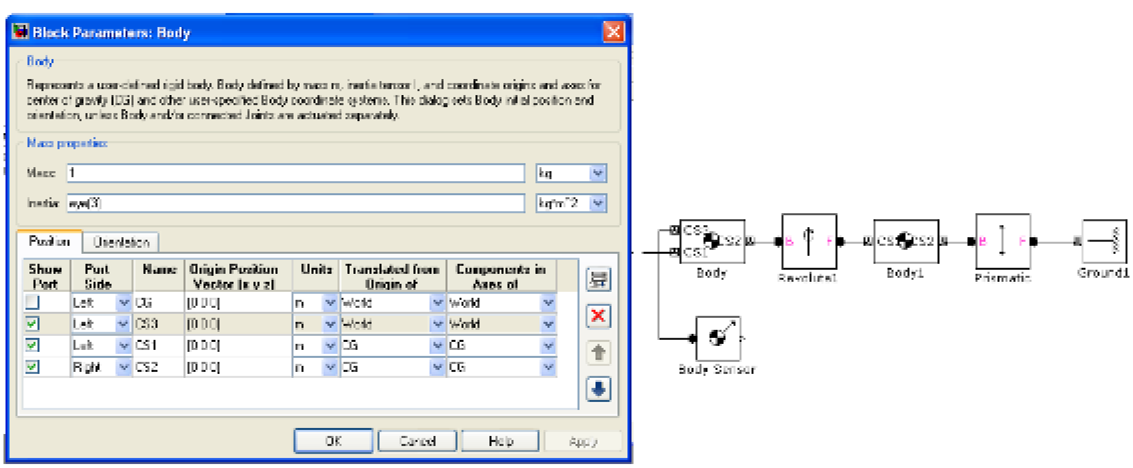
Joint sensors are capable of measuring angle, angular velocity, angular acceleration, computed torque, reaction torque, and reaction forces that can taken from either the base or follower. This outputs values for each in the X, Y, and Z axes and can send them to a scope or to other blocks.

To use a joint sensor you must attach it to an open port on joint. To create a new port open the joint's block parameters and increase the number of sensor/actuator ports.



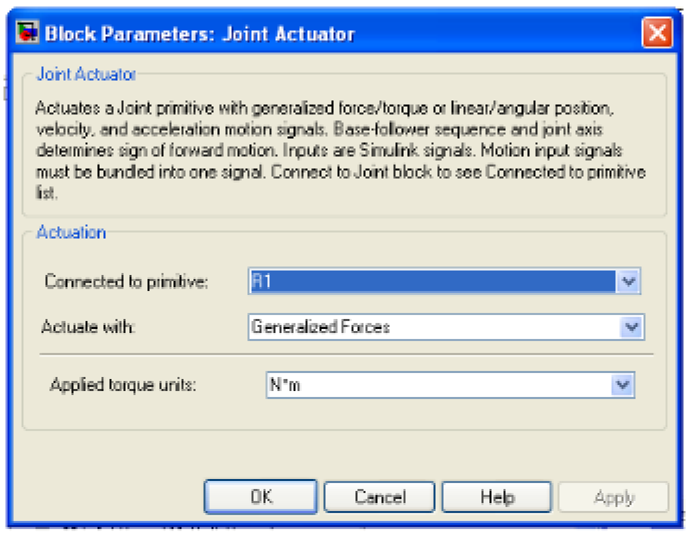
Body sensors can measure position, velocity, angular velocity, a rotation matrix, acceleration, and angular acceleration in absolute or local coordinates.

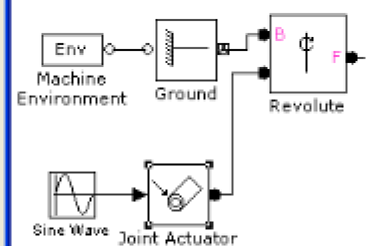
To use a body sensor you must have a free CS port on a rigid body. To add a port open the block parameters and click the 'Add' icon on the right. If you want to reference an existing point it is easiest to first select the row you want to reference and the click the 'Add button.' This new CS will have the same parameters as the one highlighted. Make sure 'Show Port' is checked. You may then connect the body sensor to the new port.



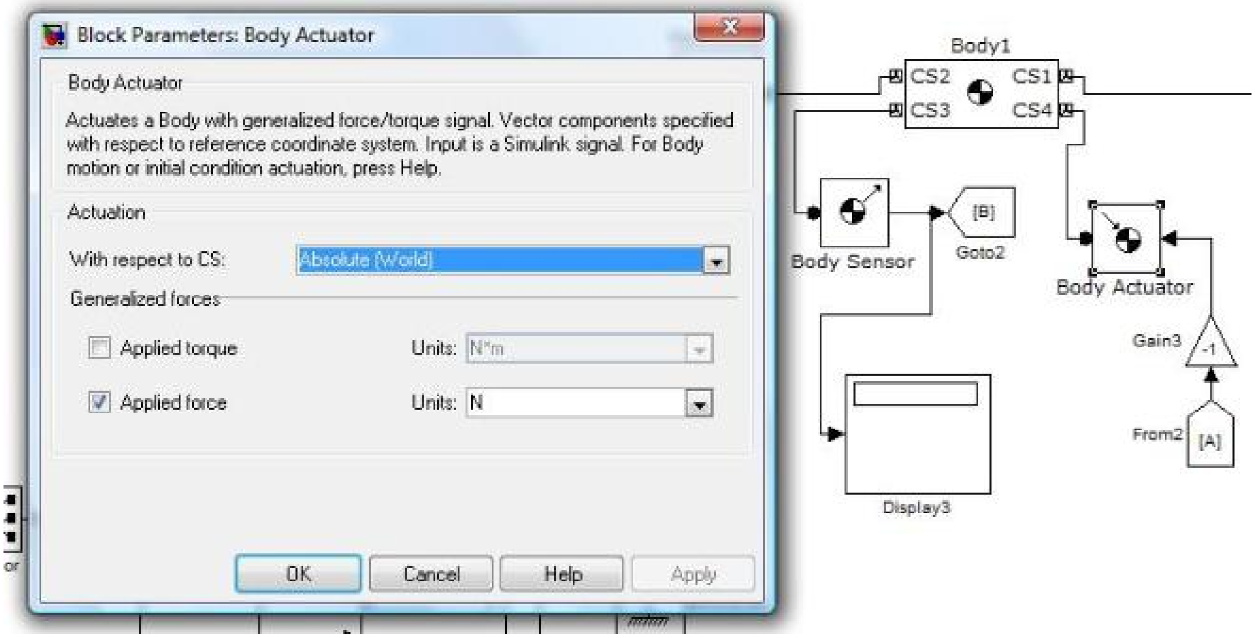
Actuators
Actuators are added in the same way as sensors.

Joint Actuators have one input and one output. You can input a value for torque using the 'Actuate with: Generalized Forces' option. You can also input actual values of angular position, velocity, and acceleration using 'Actuate with: Motion.' Inputs can come from any numerical source of the correct dimension. For example you could use a constant or sinusoidal function. The output connects to an open sensor/actuator port on a joint. The forces affect the follower body.



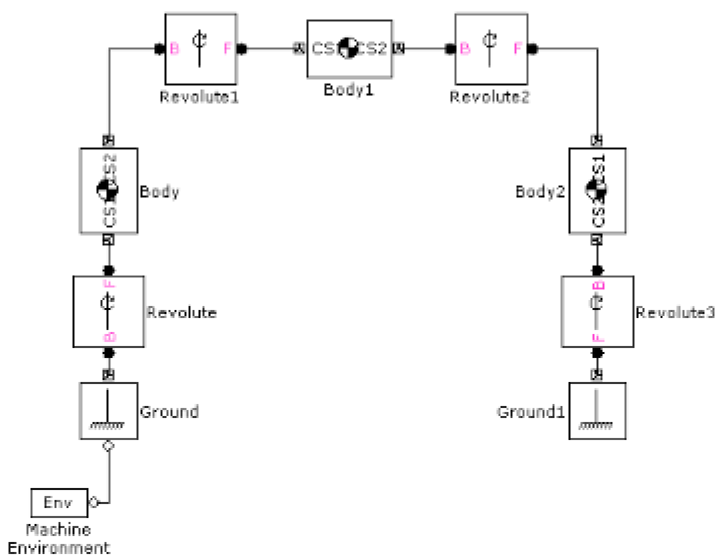


Body Actuators also have one input and one output. The input is any value provided from another block (i.e. Constant or Sine Wave). It can be applied to an individual part of the body or in reference to the world. Torques and forces can be applies.



Creating a Model

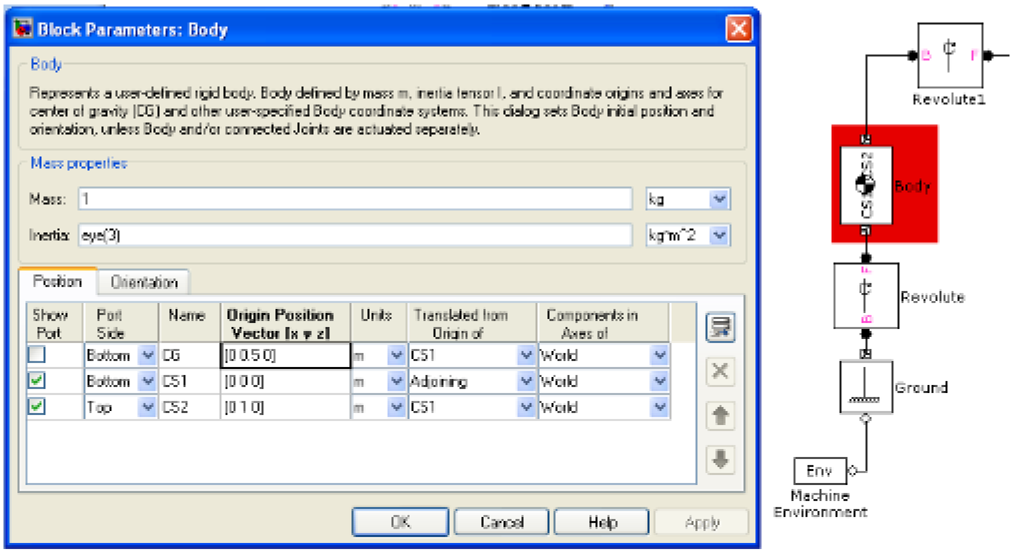
Knowing the function of each part should make laying out a model fairly trivial. To get started we will create a standard parallelogram linkage fourbar mechanism.



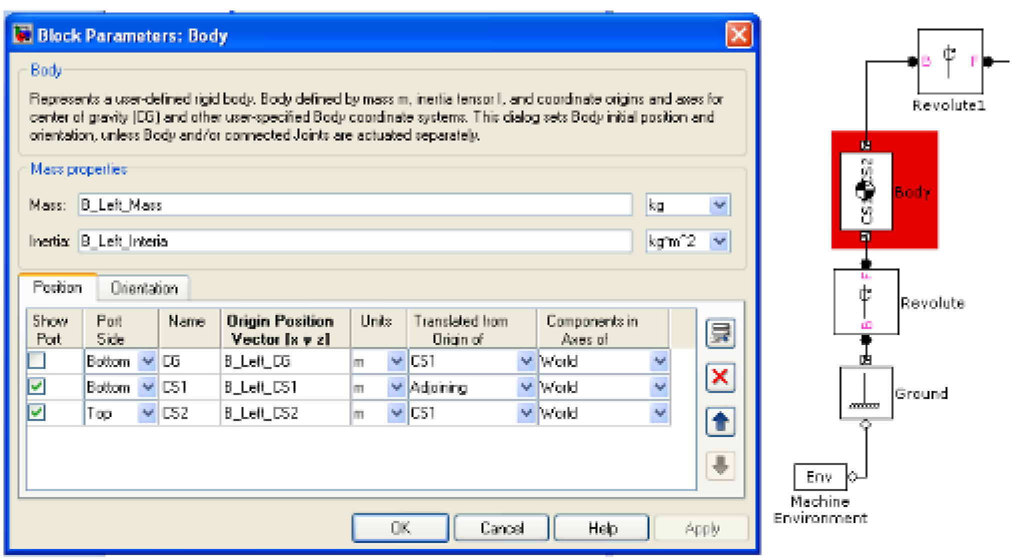
Tip: Instead of connecting nodes by clicking and dragging, click on the first block, hold down control, and click on each successive block.

Tip: To rotate a block press control+'R'. *To flip a block press* control+'I'.

There are multiple methods of configuring a model. The most straightforward way is to modify each body using numerical values, using one side of the body as a reference point starting at [0, 0, 0] connecting to the adjoining body.



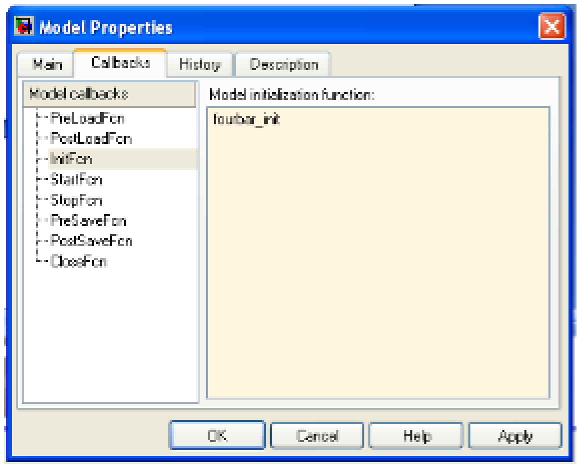
A more efficient and robust method is to label each position with a variable name. Global variables can be set from an external Matlab file. This makes it is easier to modify values later.



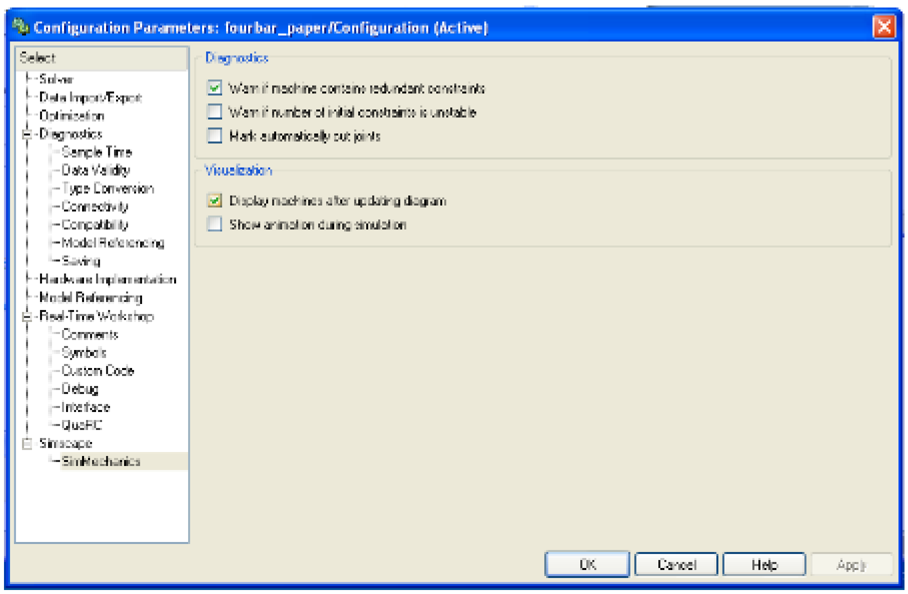
Each body and ground block should be modified to include position variables and the correct origins.

```matlab
%define environment values
global GRAVITY

%define Grounds as global variables
global G_Left G_Right

%define Bodies as global variables
global B_Left_Mass      B_Left_Inertia      B_Left_CG      B_Left_CS1      B_Left_CS2
global B_Coupler_Mass   B_Coupler_Inertia   B_Coupler_CG   B_Coupler_CS1   B_Coupler_CS2
global B_Right_Mass     B_Right_Inertia     B_Right_CG     B_Right_CS1     B_Right_CS2

%% Environtment values
GRAVITY = [0 0 0];

%% Ground positions
G_Left = [0 0 0];
G_Right = [1 0 0];

%% Body mass, inertia, and positions
B_Left_Mass = 1;
B_Left_Inertia = eye(3);
B_Left_CS1 = [0 0 0];
B_Left_CS2 = [0 1 0];
B_Left_CG = (B_Left_CS1 + B_Left_CS2)/2;

B_Coupler_Mass = 1;
B_Coupler_Inertia = eye(3);
B_Coupler_CS1 = [0 0 0];
B_Coupler_CS2 = [1 0 0];
B_Coupler_CG = (B_Coupler_CS1 + B_Coupler_CS2)/2;

B_Right_Mass = 1;
B_Right_Inertia = eye(3);
B_Right_CS1 = [0 0 0];
B_Right_CS2 = [0 -1 0];
B_Right_CG = (B_Right_CS1 + B_Right_CS2)/2;

%%
```

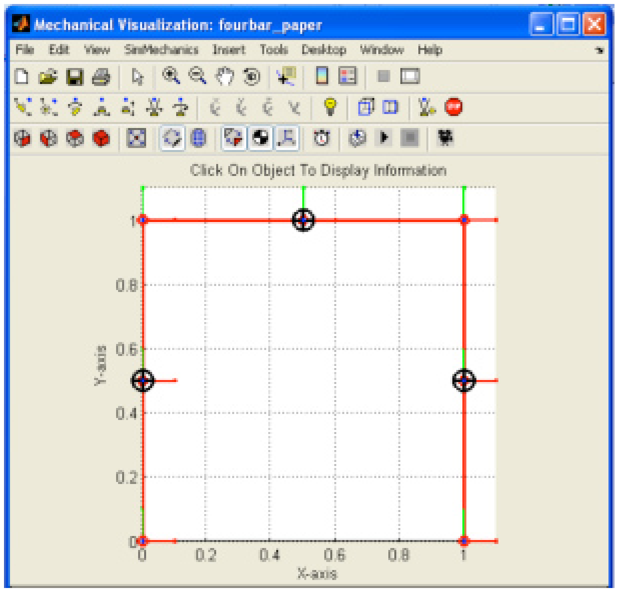
Before running the model this m-file must be called. To set it up so it automatically initializes every time the model is called go to Model Properties under the File menu. Under InitFcn in Callbacks add the name of the initialize function.



In order to view the mechanism a visualization setting must be turned on. In the Simulink window go to Configuration Parameters under the Simulation menu. Select SimMechanics in the Simscape rolldown. Check *Display Machines after updating diagram*.



To visualize your mechanism press Ctrl+D or go to Update Diagram under the Edit menu. You should see bars for each body created.



The motion can be analyzed visually or by exporting datapoints. The latter can be done by using a *To Workspace* block from the Simulink Sinks.

Concluding Remarks

The same procedure can me used to assemble increasingly complex mechanisms with hard or unsolvable analytical solutions. SimMechanics can additionally be used for actuated systems. It is possible to control the mechanism in realtime using additional hardware.