Powered by ITI

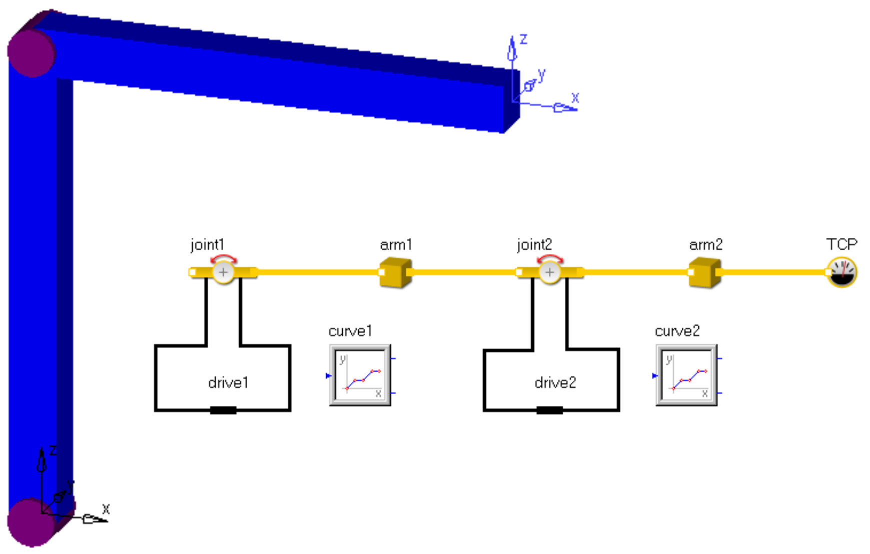
Tutorials

Tutorial 6 - Create a Multi-Body-Mechanics Model

Objective

In this tutorial you will model a simple robot, consisting of two arms and kinematic drives. The exercise will start with the creation of the basic MBS structure, which consists of two rigid body arms connected by revolute joints. The driven variant of the revolute joint allow us to adept 1D drive models to the 3D robot model. The motion of the robot arms is determined by the relative joint angles defined in independent curves.
Drive torques and joint forces as well as the trajectory of the TCP are results used in further robot design studies.

- *Using of the Multi-Body-Mechanics library*
- *Creating 3D models*
- *Paramerization of 3D models*
- *Animation of 3D models*



It is assumed that you are familiar with the basic functionality of SimulationX. Therefore, please refer to “Tutorial 1: Getting Started” for a general introduction on how to select elements from the libraries, how to connect them and enter parameters, how to run a simulation and how to open result windows.

The listed elements will be used in the modeling task. Make sure the libraries are available with your SimulationX license.

Mechanics/MBS Mechanics library

- *Rigid Bodies/Cuboid for modeling the arm*
- *Joints/Actuated Revolute Joint for connecting the arms and defining the models DOF*
- *AbsoluteKinematicSensor representing the TCP and monitoring the trajectory of TCP*

Mechanics/Rotary Mechanics library

- *Constraint as kinematic drives at both joints*

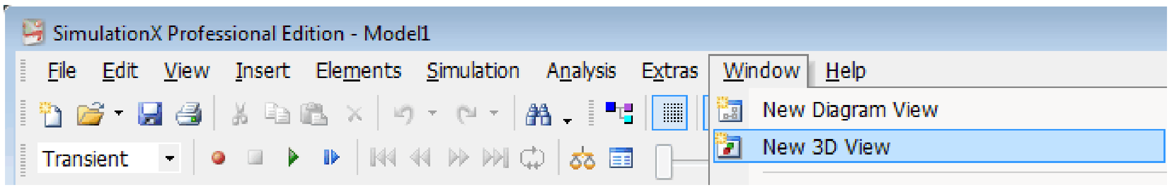
Signal Blocks library

- *Signal Sources/Curve to define the robot arms motion*

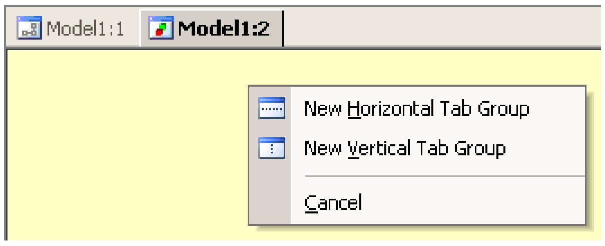
Example:

- *Setting the SimulationX Environment*

We intend to create a 3D model of the robot. This means, there should be a 3D-view in addition to the diagram view, which is to be opened via Menu/Window/New 3D View.



We suggest arranging Diagram View and 3D View in two Vertical Tab Groups. This is realized picking and placing the Tab of the 3D View, see figure below.

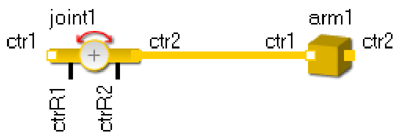


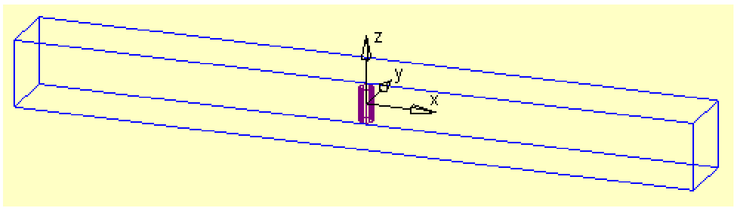
We will start the exercise with Diagram View and 3D View arranged in parallel. You should use the "zoom all" button to refit the 3D View of your model occasionally.

The 3D View shows the actual model only in the non calculated state. If changes are not shown, you have to either use the "zoom all" button or reset your model to start from a previous calculation.

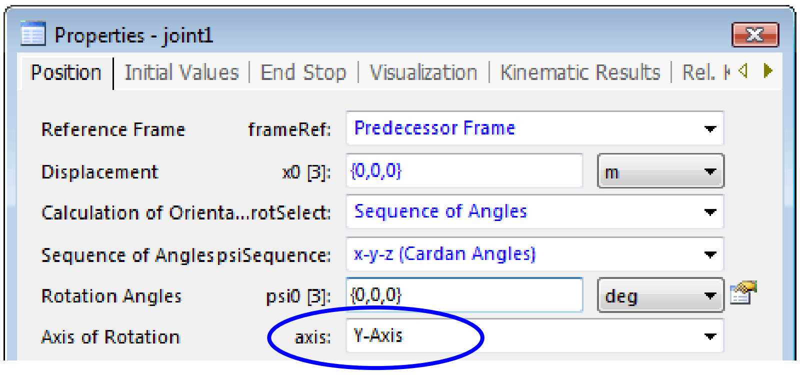
- *Getting started*

Let us create the first arm of our robot. Open the Library Bar / Mechanics / MBS Mechanics. There you find sub-libraries such as Bodies, Joints, Constraints, Forces, … . Select Rigid Bodies/ Cuboid to model the robot arm. Rename the element from cuboid1 to arm1. It is first space fixed (free kinetic connector ctr1). You may test this fact by starting a transient simulation.
The arm1 receives its degree of freedom (DOF) by placing a revolute joint (1 rotational DOF) between ground and arm1. Select the Joints / Actuated Revolute Joint for adding a 1D drive train later. Rename the element actuatedRevoluteJoint1 to joint1. Connect the kinematic connector ctr2 (yellow) of joint1 to the kinetic connector ctr1 (white) of arm1 and get the shown result. Use "wireframe" to see all components.





- *Setting the joint properties*

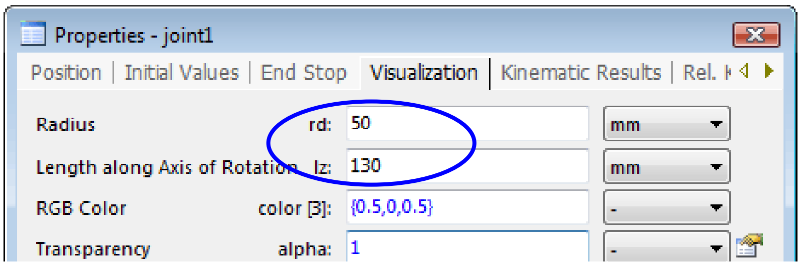


Properties - joint1

Position | Initial Values | End Stop | Visualization | Kinematic Results | Rel. K

Reference Frame	frameRef:	Predecessor Frame	
Displacement	x0 [3]:	{0,0,0}	m
Calculation of Orienta...rotSelect:		Sequence of Angles	
Sequence of Angles	psiSequence:	x-y-z (Cardan Angles)	
Rotation Angles	psi0 [3]:	{0,0,0}	deg
Axis of Rotation	axis:	Y-Axis	

Position and orientation is given referring to its predecessor frame (which is equal to origin for free kinetic connectors). Position and orientation remain unchanged but select Y-Axis as the joints axis of rotation.

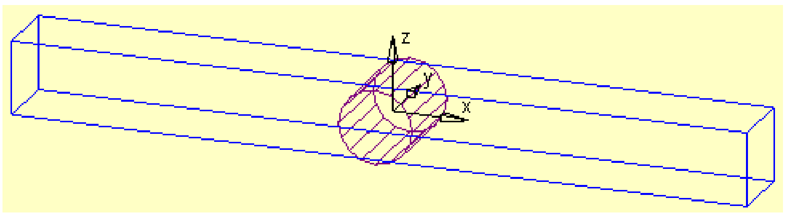


Properties - joint1

Position | Initial Values | End Stop | Visualization | Kinematic Results | Rel. K

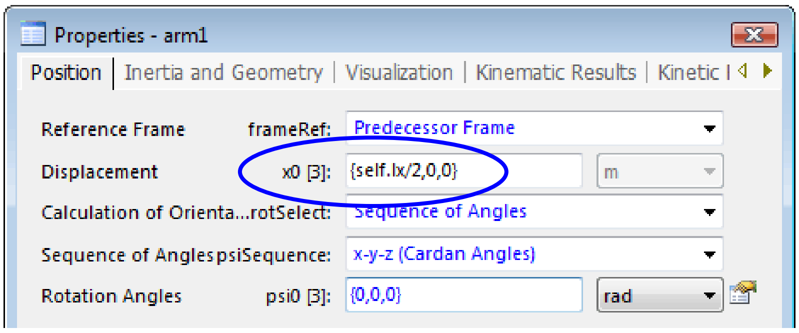
Radius	rd:	50	mm
Length along Axis of Rotation	lz:	130	mm
RGB Color	color [3]:	{0.5,0,0.5}	-
Transparency	alpha:	1	-

Visualization properties are set to radius rd = 50 mm and length lz=130mm. The joint is shown as a cylinder in y-direction with length and radius better fitted to the arms dimensions.



- *Setting properties of the rigid body arm1*

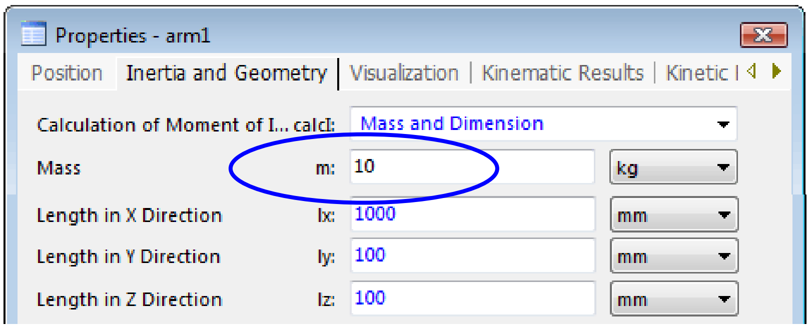
Position and orientation is given referring to its predecessor frame (which is the joint frame, shown as tripod when selecting the joint). Arm frame and joint frame now coincide. The arm has to be shifted half its length in x direction to place its left end at the joint. Change the position vector to {0.5,0,0} m (or {lx/2,0,0} for a parametric model). The orientation remains unchanged.



Properties - arm1

Reference Frame	frameRef:	Predecessor Frame	
Displacement	x0 [3]:	{self.lx/2,0,0}	m
Calculation of Orienta...rotSelect:		Sequence of Angles	
Sequence of Angles psiSequence:		x-y-z (Cardan Angles)	
Rotation Angles	psi0 [3]:	{0,0,0}	rad

Mass should be changed to 10 kg (standard value is 1 kg). Geometry remains unchanged.

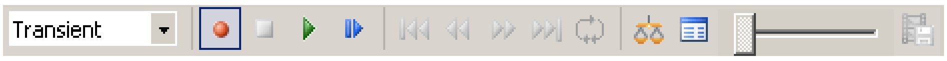


Properties - arm1

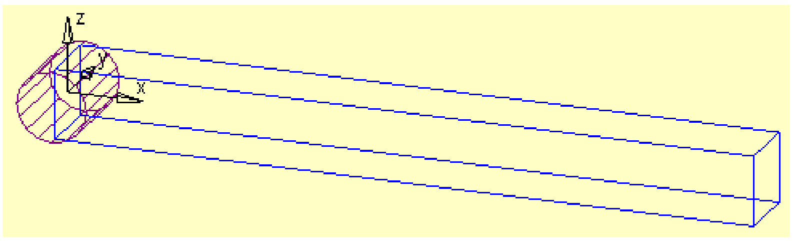
Calculation of Moment of I... calcI:		Mass and Dimension	
Mass	m:	10	kg
Length in X Direction	lx:	1000	mm
Length in Y Direction	ly:	100	mm
Length in Z Direction	lz:	100	mm

- *Intermission*

The first step is done; you have modeled the simple pendulum. You may demonstrate its function by running the transient simulation ▶. To run a few cycles of the pendulum the Stop Time has to be set to 20 s, using the property dialog in the Simulation Control toolbar ("Transient" has to be set as Kind of Simulation).



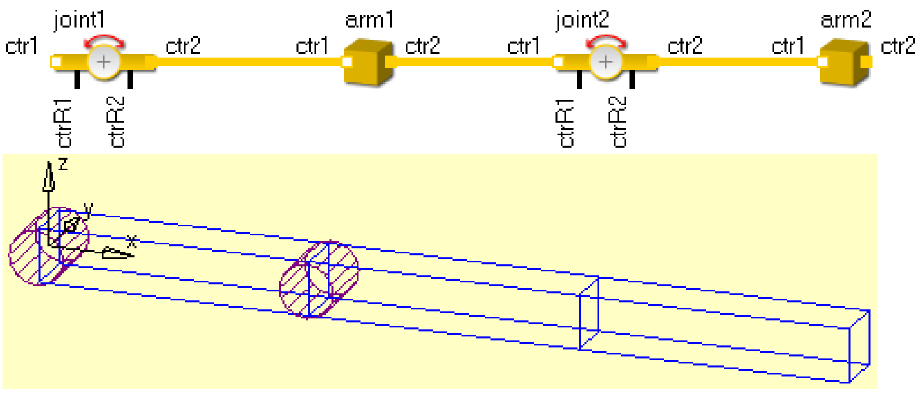
The calculation is done so fast, that 3D-view switches from starting position to final position immediately. Activate the "record" ● to animate the results after the simulation run.



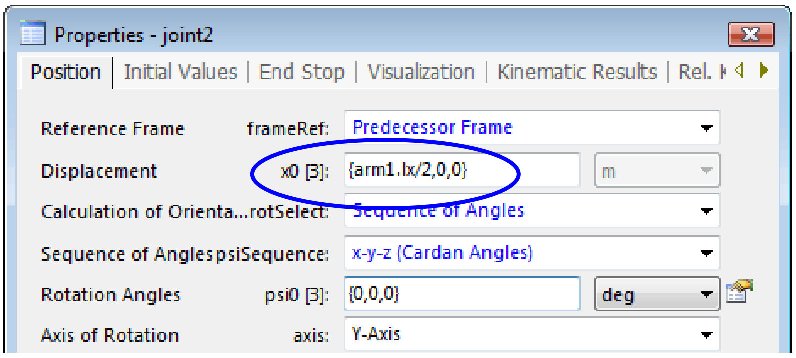
To run the animation you have to switch the "Kind of Simulation" from "Transient" to "Animation". This has to be done in the Simulation Control toolbar.

Remark
There is a predefined vector gravity = {0, 0, -9.8065} m/s² acting at each body in negative z-direction.

- *Creating the second arm*



Create the second arm by simply copy and paste the model and connect joint2.ctr1 to arm1.ctr2. The result is shown on the left.



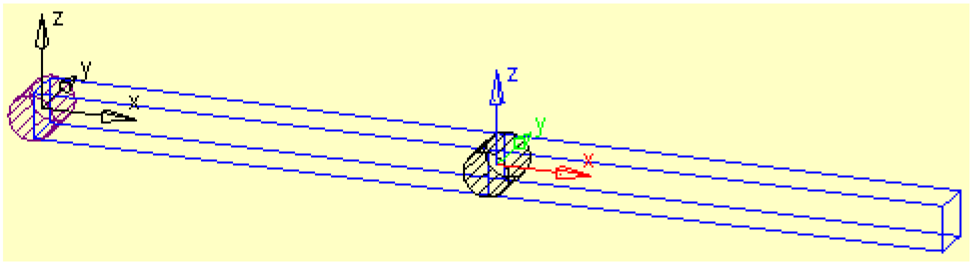
Properties - joint2

Position | Initial Values | End Stop | Visualization | Kinematic Results | Rel. I

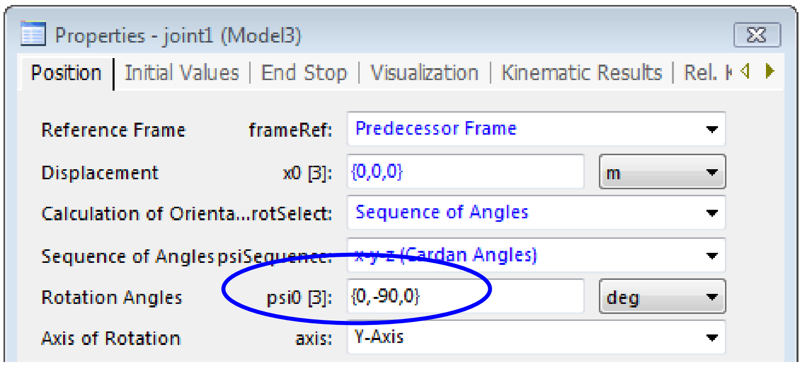
Reference Frame	frameRef:	Predecessor Frame	
Displacement	x0 [3]:	{arm1.lx/2,0,0}	m
Calculation of Orienta...rotSelect:		Sequence of Angles	
Sequence of Angles psiSequence:		x-y-z (Cardan Angles)	
Rotation Angles	psi0 [3]:	{0,0,0}	deg
Axis of Rotation	axis:	Y-Axis	

Change the position vector x0 of joint2 to {0.5,0,0} m (or {arm1.lx/2,0,0} for a parametric model). The orientation remains unchanged. The arm2 is at its correct position due to our former settings for arm1.

The second step is done and you have now modeled the double pendulum.



- *Initial position of the model*



Properties - joint1 (Model3)

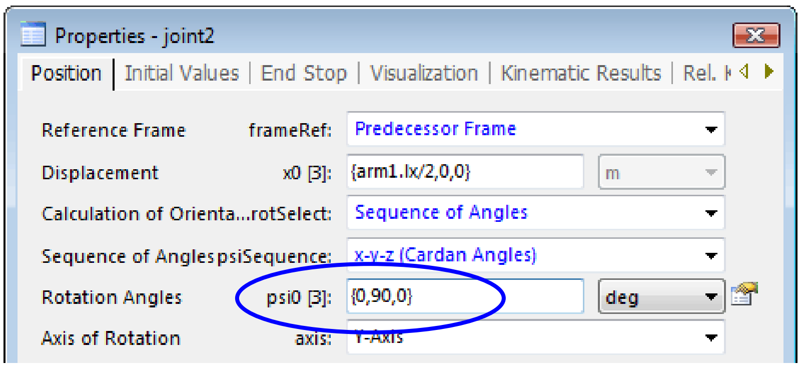
Position | Initial Values | End Stop | Visualization | Kinematic Results | Rel. I

Reference Frame	frameRef:	Predecessor Frame	
Displacement	x0 [3]:	{0,0,0}	m
Calculation of Orienta...rotSelect:		Sequence of Angles	
Sequence of Angles psiSequence:		x-y-z (Cardan Angles)	
Rotation Angles	psi0 [3]:	{0,-90,0}	deg
Axis of Rotation	axis:	Y-Axis	

Setting the initial position of our robot according to the figure at p.1, there are two possible solutions.

1. Set the initial joint angle to -90 deg for joint1 and to 90 deg for joint2.

2. Rotate the joint elements around the y-axis with respect to their reference frames by setting the rotation angles (sequence x-y-z) to {0, -90, 0} deg for joint1 and to {0,90,0} deg for joint2.
This leaves the initial joint angles at


Properties - joint2
Position | Initial Values | End Stop | Visualization | Kinematic Results | Rel.

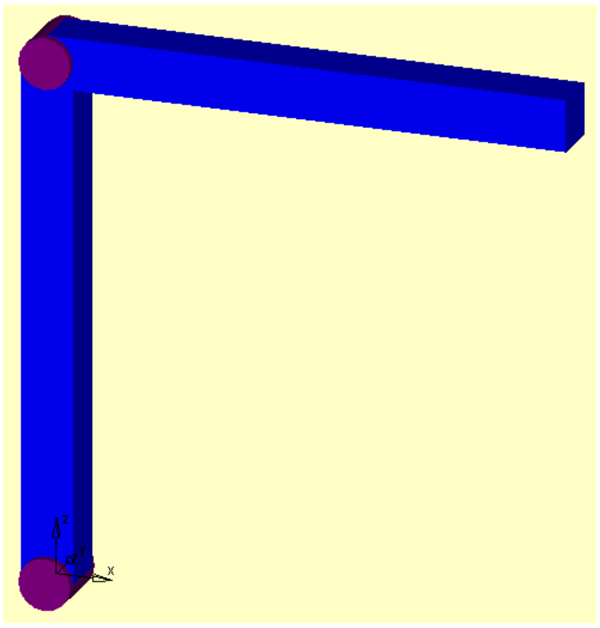
Reference Frame	frameRef:	Predecessor Frame	
Displacement	x0 [3]:	{arm1.lx/2,0,0}	m
Calculation of Orienta...rotSelect:		Sequence of Angles	
Sequence of Angles	psiSequence:	x-y-z (Cardan Angles)	
Rotation Angles	psi0 [3]:	{0,90,0}	deg
Axis of Rotation	axis:	Y-Axis	

0 deg initial value. We prefer this method.

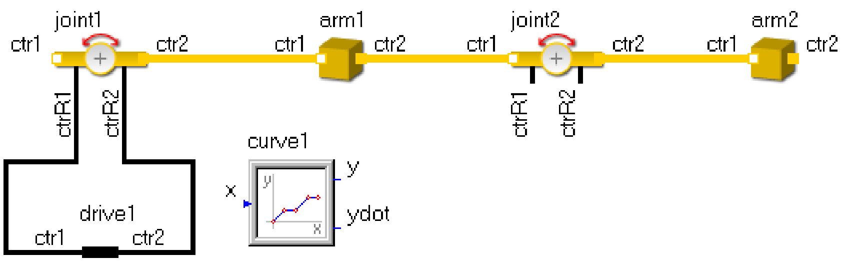
Make sure you have changed the unit from 'rad' do '°' or 'deg'.

- *Intermediate Result*

The MBS part of the model is now complete. Arms and joints have been defined, connected and parameterized. The initial joint angles are 0 deg.
To finish the model, we have to add some drive model. The next point is dealing with this task.

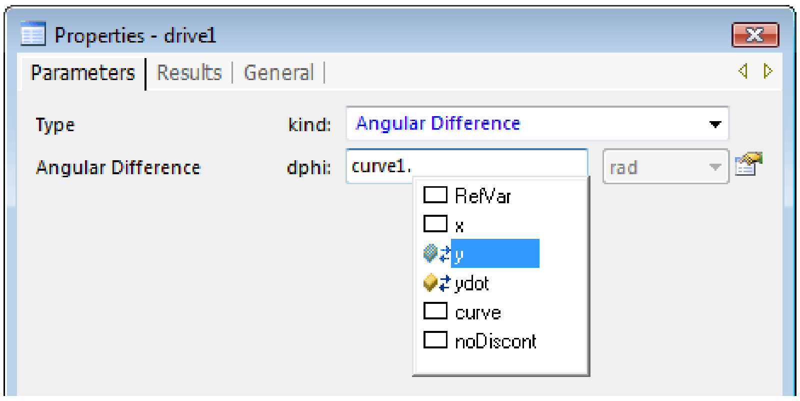


- *Modeling the drives*



Add a constraint element from the library Mechanics / Rotational Mechanics / Constraint to model a kinematic drive. Rename it to drive1 and connect drive1.ctr1 to joint1.ctrR1 and drive1.ctr2 to joint1.ctrR2. Add an element curve1 from the library Signal Blocks / Signal Sources / Curve to define the given motion set.

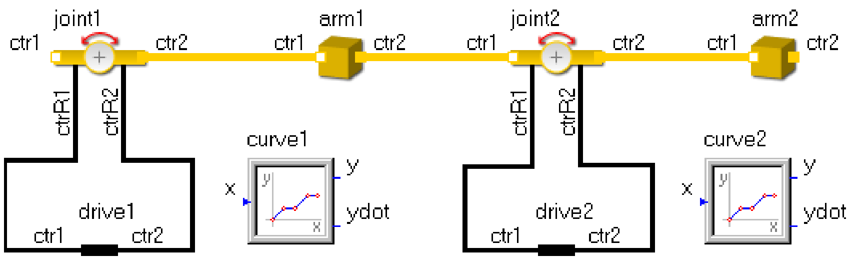
To use the output variable y of curve1 as the parameter of drive1, we set a reference to curve1.y as the angular difference dphi.



Type kind: Angular Difference

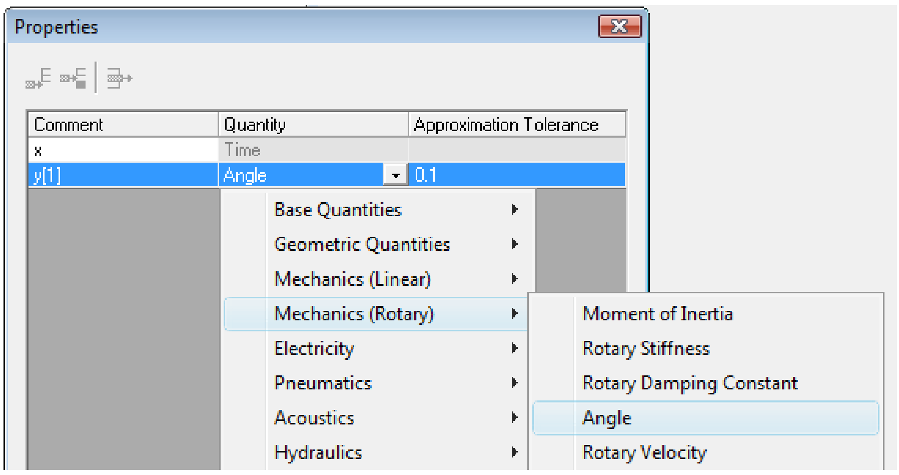
Angular Difference dphi: curve1. rad

Copy and paste the now created kinematic drive to joint2 too and set the reference to curve2.y in drive2.



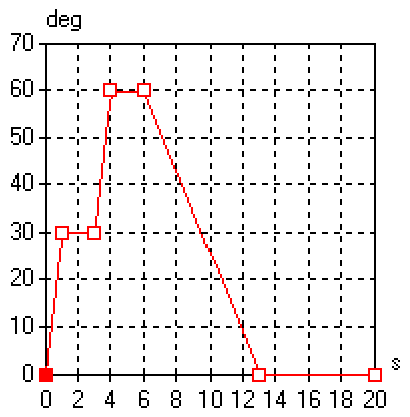
Finally it must be defined a curve for each drive defining the joint angle. It is important to set the physical quantity of the curves y to angle. This is done in the property dialog in the curve assistant.

Make sure to set the quantity for y to angle and the unit to 'deg' or '°'.



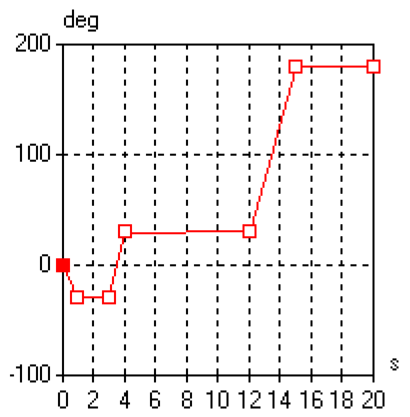
Comment	Quantity	Approximation Tolerance
x	Time	
y[1]	Angle	0.1

For editing curves see Tutorial 2 and 3. After this you may continue with the input of the given curves.



time [s]	angle [deg]
0	0
1	30
3	30
4	60
6	60
13	0
20	0

Curve of relative joint angle 1 (drive1)



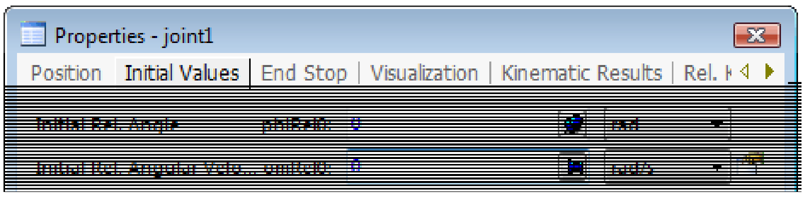
time [s]	angle [deg]
0	0
1	-30
3	-30
4	30
12	30
15	180
20	180

Curve of relative joint angle 2 (drive2)

Confirming the curves, the model is ready for a simulation run.

- *Transient Simulation Run*

If the first simulation run results in an error message: "Calculation of initial values failed!", a modification of initial value settings is required. Checking the given motion curves, you see an initial angle of 0 deg but a gradient (initial velocity) different from 0. This has to be considered in the Initial Values settings of the joints.



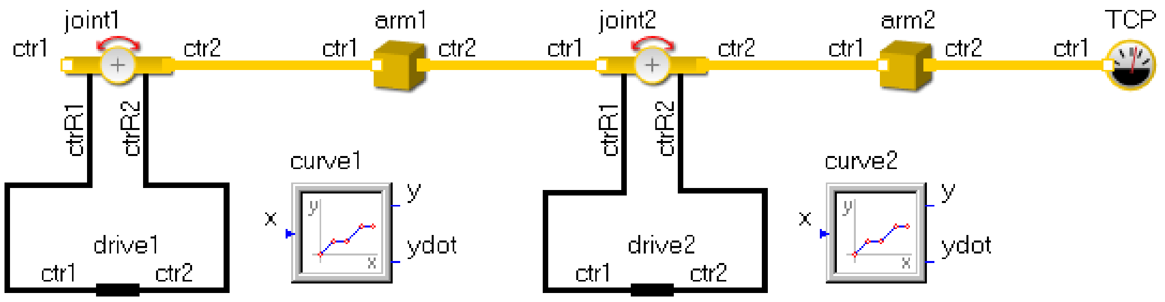
Initial Rel. Angle	phiRel0:	0	rad
Initial Rel. Angular Velo...	omRel0:	0	rad/s

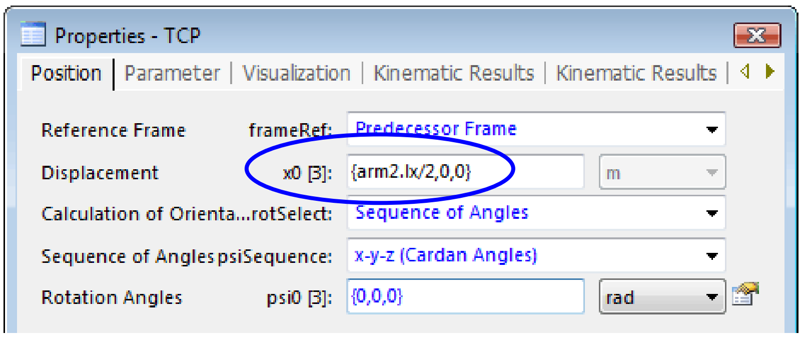
Releasing the pin at the Initial Velocity dialog for both joints, the model runs well. The robot is now executing the given motion.

To run the full motion cycle the Stop Time has to be set to 20 s, using the property dialog in the Simulation Control toolbar ("Transient" has to be set as Kind of Simulation).

- *Viewing Results*

We have to add a sensor element from Mechanics / MBS Mechanics / Sensor /Absolute Kinematic Sensor to monitor the trajectory of the TCP. Rename the sensor to TCP.

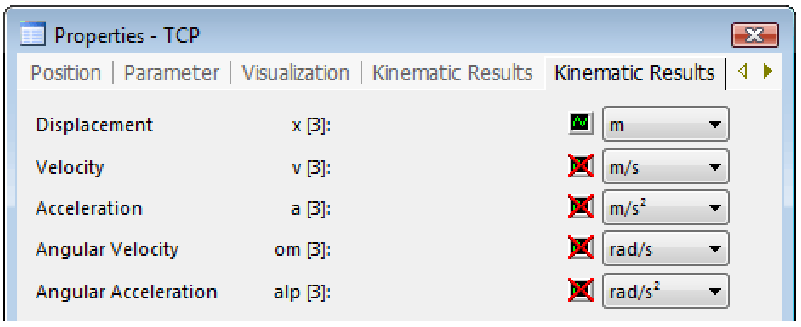



Properties - TCP

Position | Parameter | Visualization | Kinematic Results | Kinematic Results |

Reference Frame	frameRef:	Predecessor Frame	
Displacement	x0 [3]:	{arm2.lx/2,0,0}	m
Calculation of Orienta...rotSelect:		Sequence of Angles	
Sequence of Angles psiSequence:		x-y-z (Cardan Angles)	
Rotation Angles	psi0 [3]:	{0,0,0}	rad

Connecting the sensors kinetic connector ctr1 to the kinematic connector ctr2 of arm2, we have to define its position with respect to the arm2 coordinate system (predecessor frame) to x0={arm2.lx /2, 0, 0}.

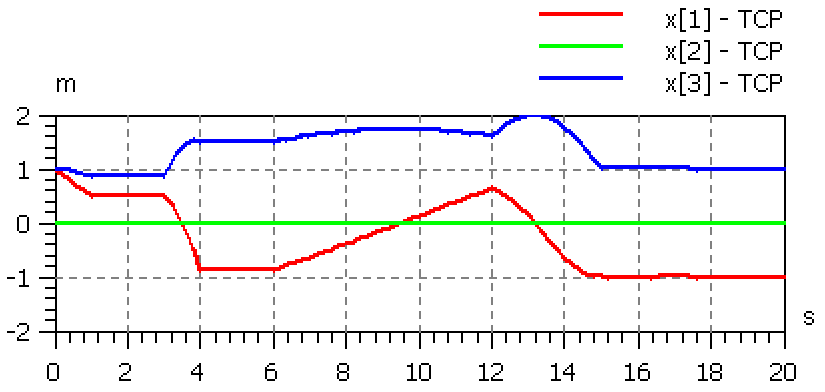

Properties - TCP

Position | Parameter | Visualization | Kinematic Results | Kinematic Results |

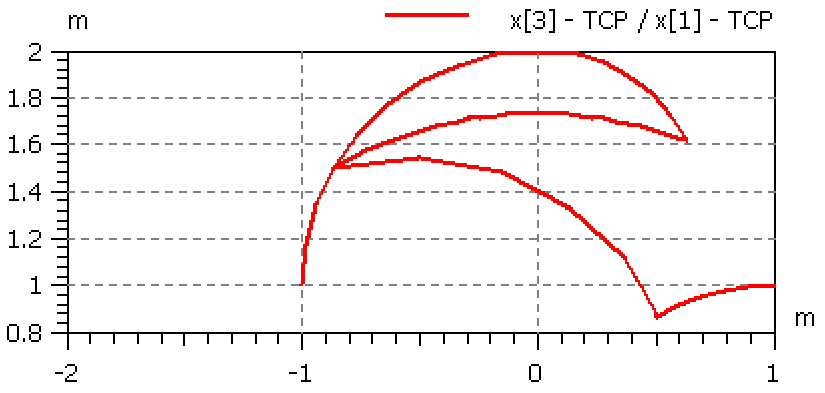
Displacement	x [3]:	✓	m
Velocity	v [3]:	✗	m/s
Acceleration	a [3]:	✗	m/s²
Angular Velocity	om [3]:	✗	rad/s
Angular Acceleration	alp [3]:	✗	rad/s²

Selecting the appropriate Kinematic Result / Displacement x[3] and plotting the result curves we may follow the motion of the TCP in time.

The result curves are shown on the right. We get the x[2] component equal to zero because of the planar motion in x-z-plane.
Thus the x[2] curve should be deleted by picking the item "x[2] – TCP" and using the menu Edit/Clear in the result window.



We can combine the curves x[1](=x) and x[3] (=z) over time to plot the trajectory of TCP z over x as x[3](x[1])=z(x) using the button y(x).



Finally, let us resume a few points concerning the benefits of this tutorial

- *We generated a 3D MBS basic structure using parametric positioning of joints and bodies in a kinematic chain.*
- *The orientation of joints and bodies is set as rotation around the joint axis to meet the given initial position of the robot arms.*
- *The basic MBS structure is driven by 1D kinematic drives following given motion curves.*
- *The trajectory of the TCP is plotted with the introduction of the absolute kinematic sensor.*