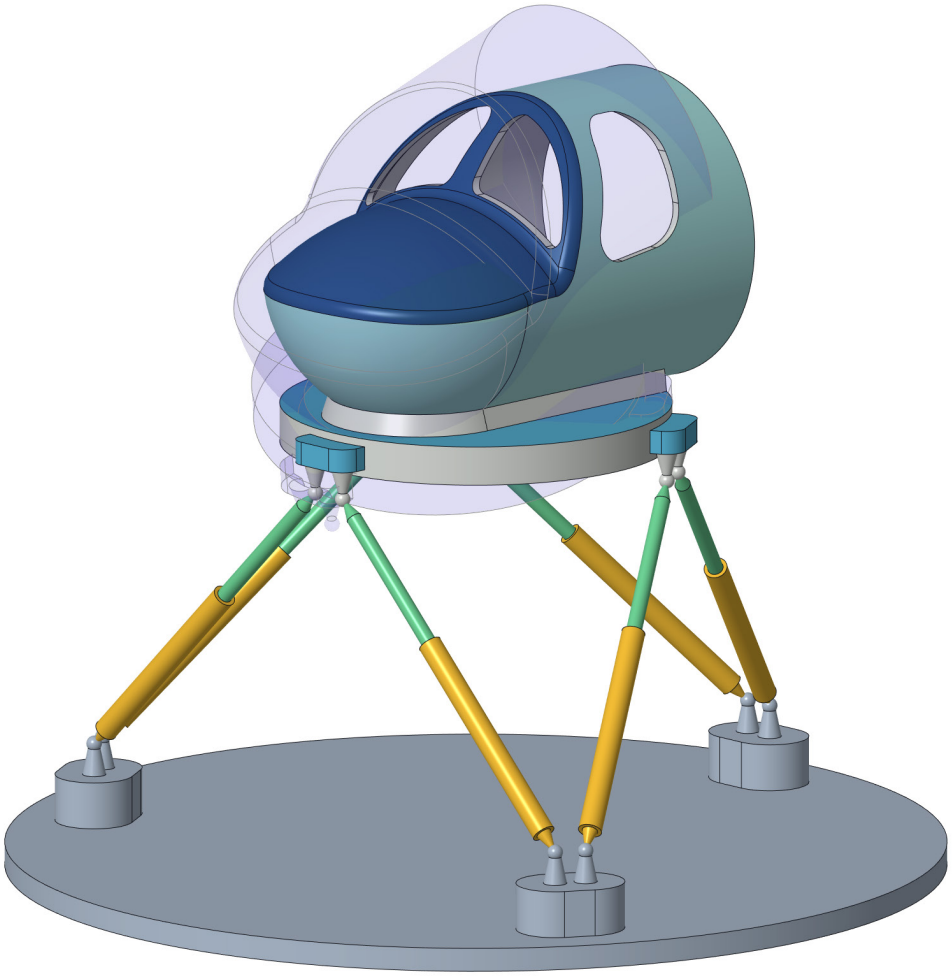
Creo® 7.0 Mechanism Design

A Short Course Tutorial



Roger Toogood, Ph.D.

SDC PUBLICATIONS
Better Textbooks. Lower Prices.
www.SDCpublications.com

Visit the following websites to learn more about this book:

Chapter 3: Creating Motion Drivers

Summary

Specification of servo motors
Motion profiles - existing functions, tables, and user-defined functions

The Slider-Crank

We are going to set up a kinematic driver, or servo, on the crank axis to make it turn at a constant speed.

In the ribbon at the top, select ***Applications*** ▸ ***Mechanism***. In the model tree window, the mechanism tree is now displayed. This will eventually contain all aspects of the mechanism model. Explore this for a moment to find the defined bodies, joints (connections), and so on. As these are selected in the mechanism tree, they will highlight in the graphics window. Observe the small orange icons on the model that represent the various connections we defined earlier. Notice the difference between the pin, cylinder, and slider icons.

In the **Mechanism** ribbon, select ***Servo Motors***. The procedure now is to select something to be driven by this servo[9]. This can be either a joint connection or basic geometry. We will only use servos on connections here. Pick the connection between the crank axis and the ground; you can do this either using the icon on the model or in the mechanism tree where it is listed as the **ROTATION AXIS** of **Connection_1**. Observe the appearance of the servo on the model: a green "pigtail" and a purple arrow. The arrow shows the direction of positive rotation (right hand rule). In Figure 1, that means clockwise when viewed from the FRONT. In the dashboard, the ***Flip*** button will reverse this direction.

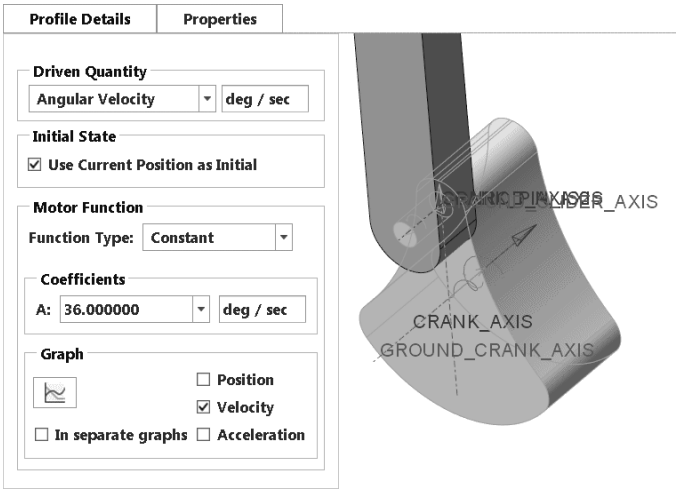


Figure 1 Defining a constant speed servo motor to drive the crank.

Now up in the dashboard, set the driven quantity to **Angular Velocity** in the drop down list and check that the function type is **Constant**. Open the **Profile Details** panel shown in Figure 1. Just in passing, open the drop down list to see what other **Motor Function Types** are available. We'll explore those later. For a constant function, we only need to specify the desired value. This is the coefficient **A** shown in Figure 1. Enter ***36*** (we'll see why that number in a minute or two) and note that units are degrees per second since Creo knows this is a rotational DOF.

---

[9] This is an action-object command sequence. You can, of course, do this using an object-action sequence by picking the pin joint icon on the model and in the RMB pop-up menu selecting the ***Servo Motor*** button.

Middle click to accept the servo definition. Check out where/how it appears in the mechanism tree.

We are finished with this step in the slider-crank model. That was easy! Save the model file. To continue with this model, move ahead to Chapter 4 to define and run an analysis. To find out more about servo options, read on!

More About Servos

Motion drivers or servo motors set up time-dependent geometric constraints in the mechanism. They come in two kinds[10]:

1. Servos that drive the motion axes of connections. In the slider-crank, we placed a servo to drive the motion axis on the pin connection between the crank and the ground. This most closely represents how we would normally think of a mechanism with some kind of motor. We will spend most of our time discussing servos on connection motion axes.

2. Servos that drive geometry like datum planes, surfaces, edges, or even points. See the Schmidt coupling in Chapter 7 for examples of servos driving the position of datum planes.

Servos can do much more than just drive the position of a joint or entity. They can be set up to drive velocity or acceleration. Both of these require additional information about the initial conditions for the simulation. For a velocity driver, you must specify the initial position. For an acceleration driver, you must specify both initial position and initial velocity.

In the slider-crank example, we used a constant velocity driver on the position of the crank. We could also have used a linear ramp function on the position. There are many other functional shapes available.

Servos do not have to specify absolute motion, that is, relative to a fixed ground. Servos defined on connections between any two rigid bodies will define the relative motion between those bodies. For example, a rotating body like a table can carry a translating body moving in a specified path relative to the table.

If you use any of the other function types, the graphing options at the bottom of the servo definition window are useful to confirm that you have defined the function correctly. Select which graph(s) you want by checking the appropriate box, then click the graph icon. The graphs will automatically update when you change any of the defining parameters for the servo.

Let’s explore some of the functions available. Remember that these can be applied to either position, velocity, or acceleration. In the examples to follow, the functions are applied to either the position or the velocity, as stated.

---

[10] Since we are not covering Mechanism Dynamics here, the topic of Force servos is skipped.

Motor Function Types

As mentioned, motor functions can be used to drive position, velocity, or acceleration. We used a constant velocity profile for the slider-crank above. Other types are available using the pull-down list shown in the figure at the right. Details and examples of each of these types are discussed below. For now, note that when you hold the mouse over one of the listed types, a pop-up will show you the general form of the function and how the coefficients are used.

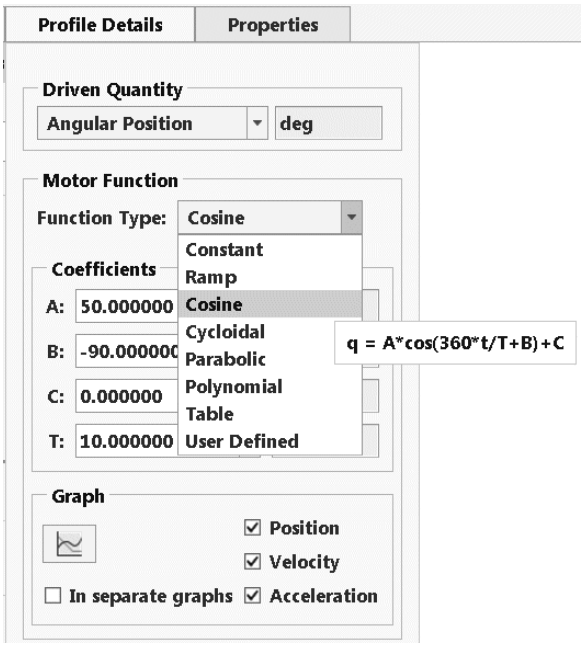


Figure 2 Selecting the motor function type

To examine and verify that the function is doing what you want, select one or more of the check boxes in the Graph panel at the bottom. You can show these all on the same graph (as done below) or in separate graph windows. Then pick the ***Graph*** button on the left to open the graph window and plot the curves. This graph will automatically update when you make changes to the coefficients in the Profile Details panel.

In the following, the quantity q is the one being driven by the function. This can be either position, velocity, or acceleration. The others are obtained automatically either as derivatives or integrals (with specified initial conditions).

Ramp

The graph for a velocity ramp function is shown at the right. This, of course, produces a constant acceleration motion. The position curve is parabolic. The functional shape is

$$q = A + Bt$$

where A = initial value
B = slope

For a velocity function, of course, the initial velocity is A and the acceleration is B. The initial position can be specified either by a parameter or the current position.

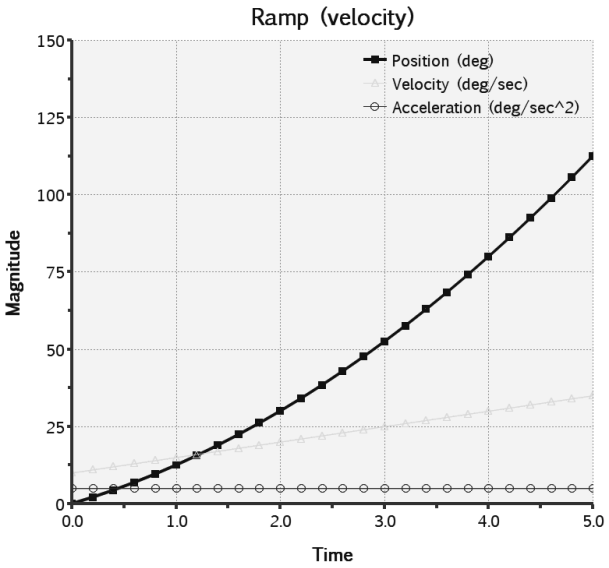


Figure 3 Function parameters: A = 10, B = 5

The same kinematic behaviour could be obtained using a constant acceleration driver, in which the initial position and velocity would be specified. Or even a parabolic position driver (see below) where the initial conditions are built-in to the parabola parameters (eg B would be the initial velocity).

Cosine

The graph for a position cosine function is shown at the right. The functional shape is

$$q = A\ \cos\left(360\,\frac{t}{T} + B\right) + Ct$$

where A = amplitude
B = phase shift
C = offset
T = period

If the motion analysis (covered in the next section) is longer than the period T, then the function repeats as many times as necessary to complete the analysis. Notice that, in that case, all three of position, velocity, and acceleration would be continuous into the next period - this is a potential issue when using repeated drivers, discussed later.

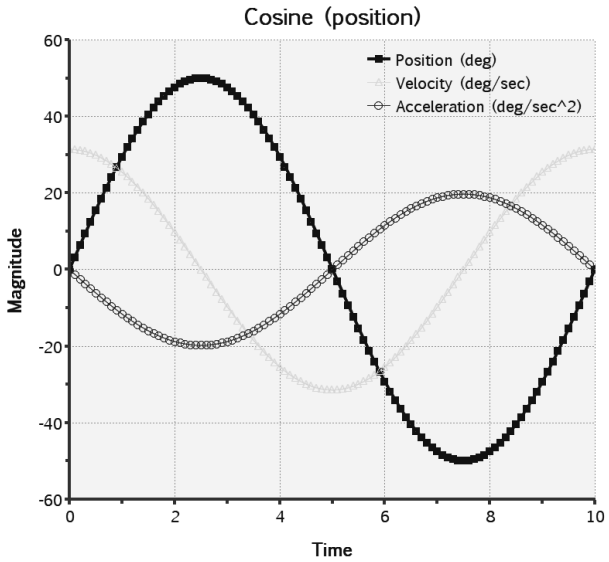


Figure 4 Function parameters: A = 50, B = -90, C = 0, T = 10

Parabolic

The graph for a position parabolic function is shown at the right. This, of course, produces a constant acceleration motion. The velocity curve is linear. The functional shape is

$$q = At + \frac{1}{2}\,B\,t^2$$

where A = initial slope
B = rate of change of slope

This is clearly the same behaviour as a constant acceleration, or ramp velocity function.

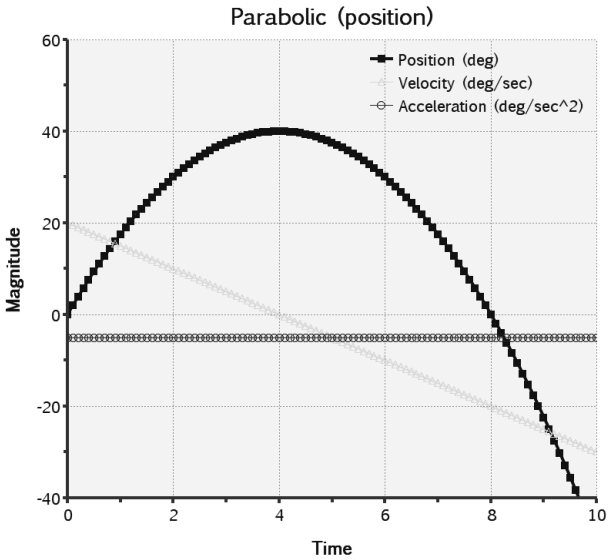


Figure 5 Function parameters: A = 20, B = -5

Polynomial

The graph for a position polynomial function is shown at the right. This is the most general of the (constant, parabolic, ramp) group of functions, from which all the others could be derived.

$$q = A + Bt + Ct^2 + Dt^3$$

where A, B, C, D = polynomial coefficients

The polynomial is interesting in that if it is used to define position, then the jerk of the motion can be defined, where jerk is defined as the rate of change of acceleration.

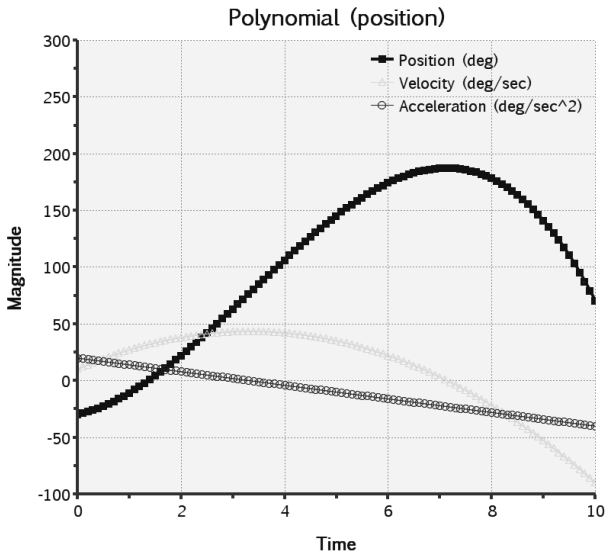


Figure 6 Function parameters: A = -30, B = 10, C = 10, D = -1

Cycloidal

The graph for a position cycloidal function is shown at the right. When used for position, this results in a step change but with smooth velocity and acceleration changes at the beginning and end of the segment.

$$q = L\left[\frac{t}{T} - \frac{1}{2\pi}\sin\left(360\frac{t}{T}\right)\right]$$

where L = rise (+ or -)
T = period

The function is useful in the specification of cam motions since the velocity and acceleration of the motion are both zero at the start and finish.

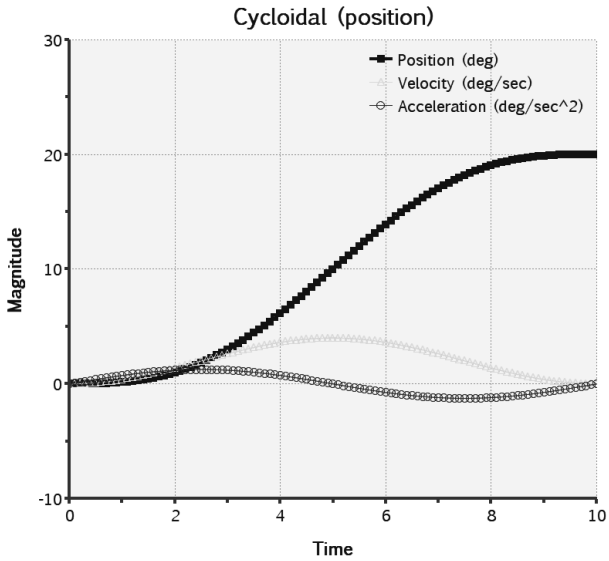


Figure 7 Function parameters: L = 20, T = 10

Table

This is defined by specifying desired values at specific times using a table as shown below. Rows are added to (or deleted from) the table using the buttons at the right. Rows can be re-ordered using the button at the lower right.

The example below is used to drive the position. As shown, table values do not need to be evenly spaced. The driven quantity can, of course, be negative. Below the tables there are options to set how many points should be interpolated between the specified values (default 50).

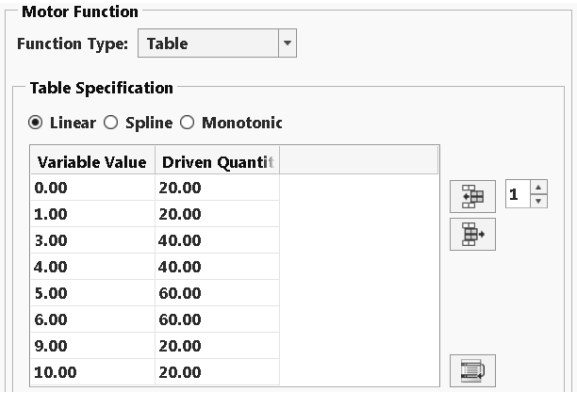


Figure 8 Data entered into table

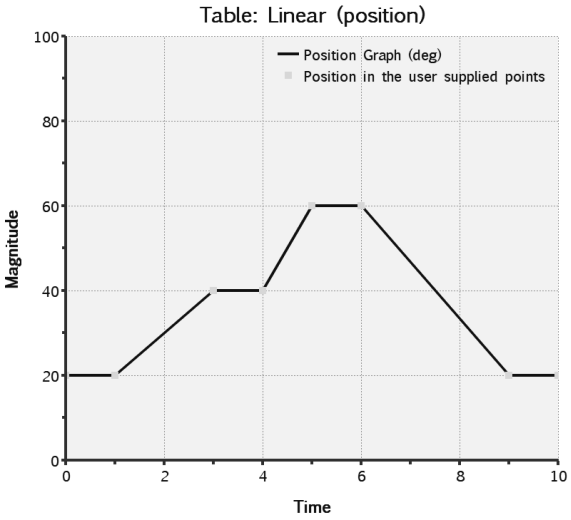


Figure 9 Table driven position using Linear radio button

The three radio buttons in the middle specify how the interpolation should be done between points. The default is **Linear** (Figure 9). The results of selecting the other options are shown below. For the **Spline** interpolation, note the overshoot and undershoot of the position interpolation and the non-zero values of initial and final velocity and acceleration. The **Monotonic** interpolation does not have these effects.

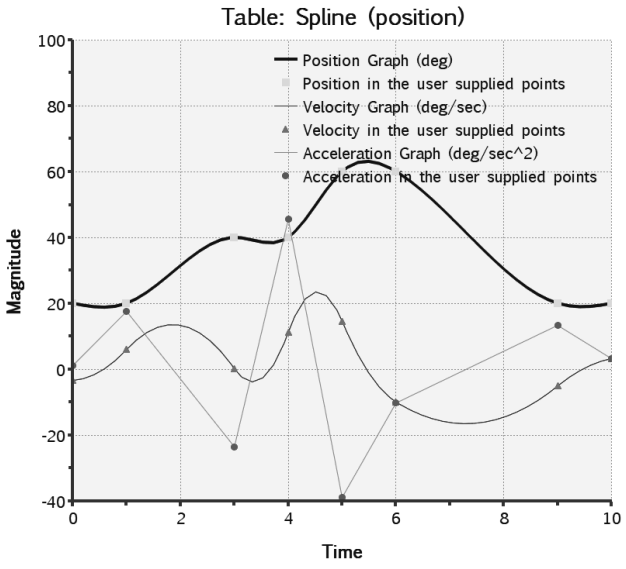


Figure 10 Table data - Spline interpolation

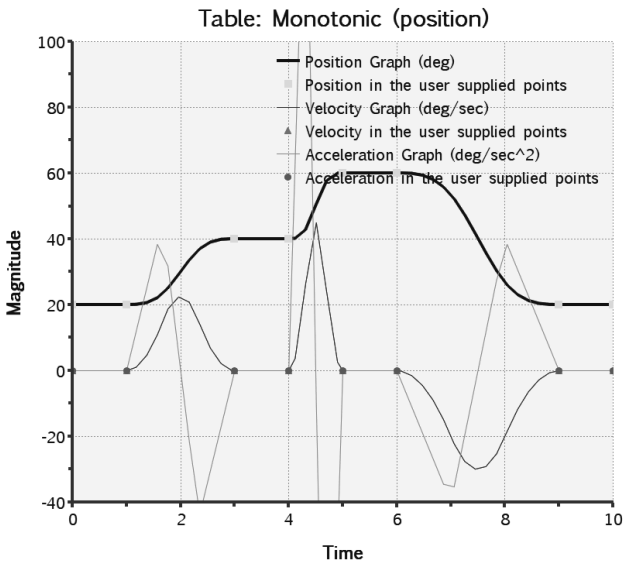


Figure 11 Table data - Monotonic interpolation

User Defined Function

If the provided functions don't give the motion that you want, you can always create your own!

The example below is used to drive the position. The function can contain several expressions that are valid over different time domains. Selecting the ***Edit Expression*** button at the right (in Figure 12) opens the window in Figure 13. This lets you enter whatever function you want. Buttons at the top present some helpful tools (like plotting the function), built-in functions, and symbols. The domain for the expression is set at the bottom of the window. It is up to you to make sure that the function is valid throughout the domain; the program will flag an error if it is not (like division by zero) during the analysis run. The graphs for the expression shown are in Figure 14.

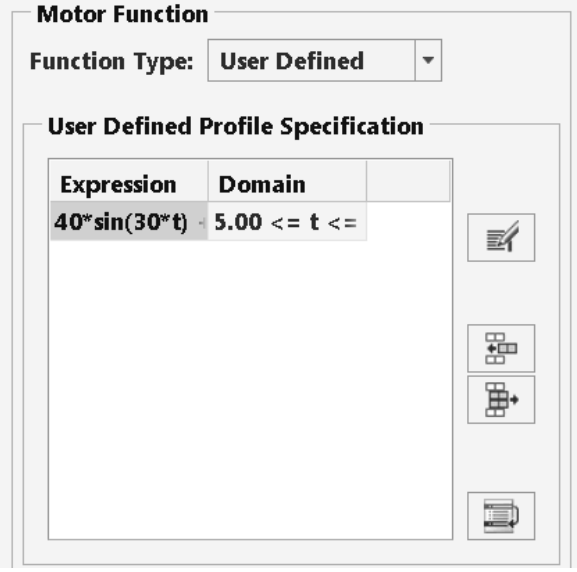


Figure 12 Setting user-defined functions

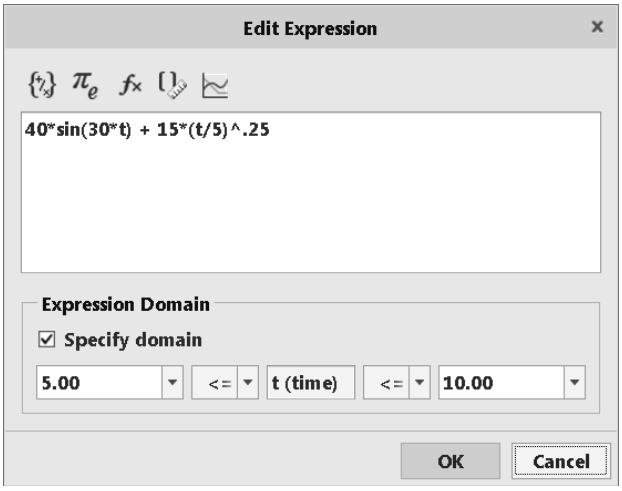


Figure 13 Defining the expression and domain for a user-defined function

For more information on user-defined functions, see the on-line Help page "To Define a User-Defined Motor Function".

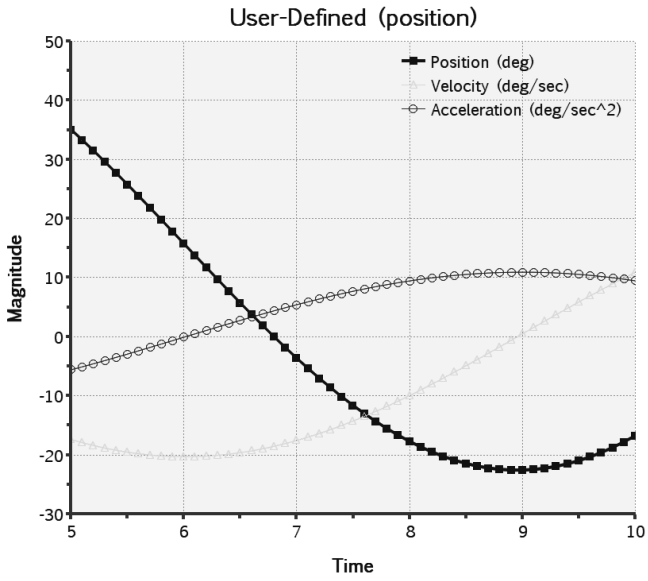


Figure 14 User-defined position function

A Note about Multiple Drivers

It is possible to have several servos defined on the same body, even on the same motion axis. Since these represent geometric constraints on the mechanism, some care needs to be taken to ensure that the motion of the servos is compatible when the analysis is running. For example, two position servos on the same axis cannot be commanding different position requirements at the same time. Similarly, a velocity and acceleration servo (admittedly a rather strange case to consider!) must be consistent with each other, and with the initial position specified. The sequencing and coordination of multiple servos is handled when the analysis run is set up in the next section.