A Design Study Approach to Classical Control

Randal W. Beard Timothy W. McLain
Brigham Young University

Updated: April 27, 2016

Homework B.a

Create a simulink animation of the pendulum on a cart system. The inputs should be sliders for z and θ.

Solution

Consider the image of the inverted pendulum shown in Figure 1, where the configuration is completely specified by the position of the cart z, and the angle of the rod from vertical θ. The physical parameters of the system are the rod length L, the base width w, the base height h, and the gap between the base and the track g. The first step in developing the animation is to determine the position of points that define the animation. For example, for the inverted pendulum in Figure 1, the four corners of the base are

$$(z + w/2, g), (z + w/2, g + h), (z - w/2, g + h), \text{ and } (z - w/2, g),$$

and the two ends of the rod are given by

$$(z, g + h) \text{ and } (z + L \sin\theta, g + h + L \cos\theta).$$

Since the base and the rod can move independently, each will need its own figure handle. The `drawBase` command can be implemented with the following Matlab code:

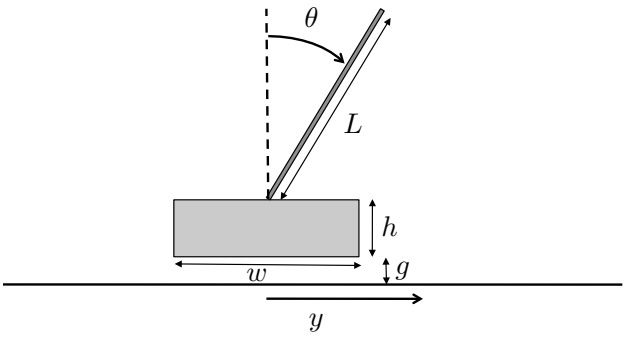

Figure 1: Drawing for inverted pendulum. The first step in developing an animation is to draw a figure of the object to be animated and identify all of the physical parameters.

```
1   function handle = drawBase(z, width, height, gap, handle)
2     X = [z-width/2, z+width/2, z+width/2, z-width/2];
3     Y = [gap, gap, gap+height, gap+height];
4     if isempty(handle),
5       handle = fill(X,Y,'m');
6     else
7       set(handle,'XData',X,'YData',Y);
8     end
```

Lines 2 and 3 define the X and Y locations of the corners of the base. Note that in Line 1, `handle` is both an input and an output. If an empty array is passed into the function, then the `fill` command is used to plot the base in Line 5. On the other hand, if a valid handle is passed into the function, then the base is redrawn using the `set` command in Line 7.

The Matlab code for drawing the rod is similar and is listed below.

```
1   function handle = drawRod(z, theta, L, gap, height, handle)
2     X = [z, z+L*sin(theta)];
3     Y = [gap+height, gap + height + L*cos(theta)];
4     if isempty(handle),
5       handle = plot(X, Y, 'g');
6     else
7       set(handle,'XData',X,'YData',Y);
8     end
```

The main routine for the pendulum animation is listed below.

```
1      function drawPendulum(u)
2          % process inputs to function
3          z         = u(1);
4          theta     = u(2);
5          t         = u(3);
6
7          % drawing parameters
8          L = 1;
9          gap = 0.01;
10         width = 1.0;
11         height = 0.1;
12
13         % define persistent variables
14         persistent base_handle
15         persistent rod_handle
16
17         % first time function is called, initialize plot
18         % and persistent vars
19         if t==0,
20           figure(1), clf
21           track_width=3;
22           % plot track
23           plot([-track_width,track_width],[0,0],'k');
24           hold on
25           base_handle = drawBase(z, width, height, gap, []);
26           rod_handle  = drawRod(z, theta, L, gap, height, []);
27           axis([-track_width,track_width,-L,2*track_width-L]);
28         % at every other time step, redraw base and rod
29         else
30             drawBase(z, width, height, gap, base_handle);
31             drawRod(z, theta, L, gap, height, rod_handle);
32         end
```

The routine **drawPendulum** is called from the Simulink file shown in Figure 2, where there are three inputs: the position z, the angle θ, and the time t. Lines 3-5 rename the inputs to z, θ, and t. Lines 8-11 define the drawing parameters. We require that the handle graphics persist between function calls to **drawPendulum**. Since a handle is needed for both the base and the rod, we define two persistent variables in Lines 14 and 15. The `if` statement in Lines 19-32 is used to produce the animation. Lines 20–27 are called once at the beginning of the simulation, and draw the initial animation. Line 20 brings the figure 1 window to the front and clears it. Lines 21 and 23 draw the ground along which the pendulum will move. Line 25 calls the

`drawBase` routine with an empty handle as input, and returns the handle `base_handle` to the base. Line 26 calls the `drawRod` routine, and Line 25 sets the axes of the figure. After the initial time step, all that needs to be changed are the locations of the base and rod. Therefore, in Lines 30 and 31, the `drawBase` and `drawRod` routines are called with the figure handles as inputs.

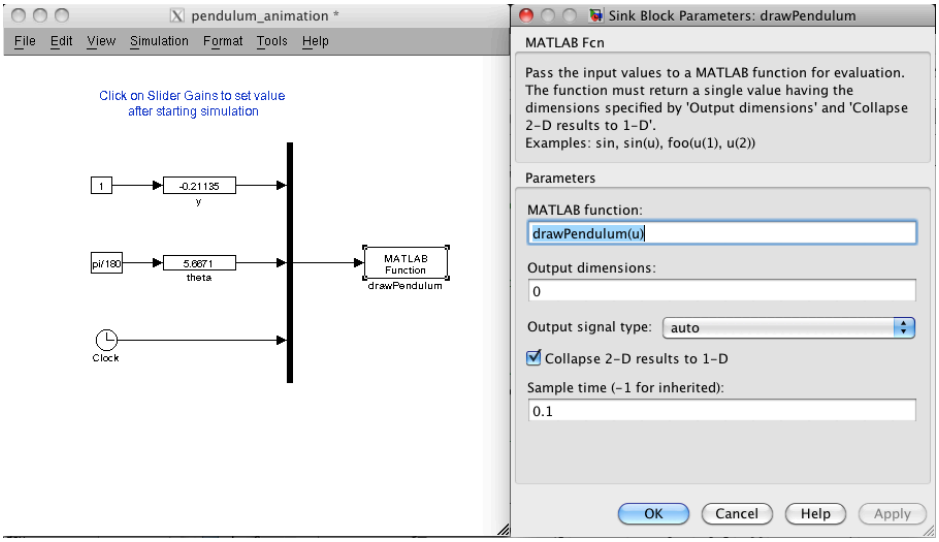

Figure 2: Simulink file for debugging the pendulum simulation. There are three inputs to the Matlab m-file `drawPendulum`: the position z, the angle θ, and the time t. Slider gains for z and θ are used to verify the animation.