

# A Design Study Approach to Classical Control

Randal W. Beard      Timothy W. McLain  
Brigham Young University

Updated: December 28, 2020

## Homework F.a

Create a simulink animation of the planar VTOL system. The inputs should be sliders for  $z_v$ ,  $z_t$ ,  $h$ , and  $\theta$ . Turn in a screen capture of the animation.

## Solution

The drawing function for the ball on beam system is listed below.

```
1 function VTOL_animation(u, P)
2
3     % process inputs to function
4     z      = u(1);
5     h      = u(2);
6     theta  = u(3);
7     %z_dot  = u(4);
8     %h_dot  = u(5);
9     %theta_dot = u(6);
10    target  = u(7);
11    t       = u(8);
12
13    % define persistent variables
14    persistent VTOL_handle
15    persistent target_handle
16
17    L = 10;
18
```

```

19     % first time function called, initialize plot and persistents
20     if t==0,
21         figure(1), clf
22         plot([0,L],[0,0],'k'); % plot track
23         hold on
24         VTOL_handle = drawVehicle(z, h, theta, []);
25         target_handle = drawTarget(target, []);
26         axis([-L/5, L+L/5, -L, L]);
27
28
29     % at every other time step, redraw base and rod
30     else
31         drawVehicle(z, h, theta, VTOL_handle);
32         drawTarget(target, target_handle);
33     end
34 end
35
36
37 %
38 %=====
39 % drawVTOL
40 % draw VTOL system
41 % return handle if 3rd argument is empty, otherwise use 3rd arg
42 % as handle
43 %=====
44 %
45 function handle = drawVehicle(z, h, theta, handle)
46
47     x1 = 0.1;
48     x2 = 0.3;
49     x3 = 0.4;
50     y1 = 0.05;
51     y2 = 0.01;
52     pts = [...
53         x1, y1;...
54         x1, 0;...
55         x2, 0;...
56         x2, y2;...
57         x3, y2;...
58         x3, -y2;...
59         x2, -y2;...
60         x2, 0;...
61         x1, 0;...
62         x1, -y1;...
63         -x1, -y1;...

```

```

64     -x1, 0;...
65     -x2, 0;...
66     -x2, -y2;...
67     -x3, -y2;...
68     -x3, y2;...
69     -x2, y2;...
70     -x2, 0;...
71     -x1, 0;...
72     -x1, y1;...
73     x1, y1;...
74     ];
75     % rotate points (must do first)
76     R = [cos(theta), sin(theta); -sin(theta), cos(theta)];
77     pts = pts*R;
78     % translate points
79     pts = pts + repmat([z,h],size(pts,1),1);
80
81     if isempty(handle),
82         handle = fill(pts(:,1),pts(:,2),'b');
83     else
84         set(handle,'XData',pts(:,1),'YData',pts(:,2));
85         drawnow
86     end
87 end
88
89 %
90 %=====
91 % drawTarget
92 % draw the Target
93 % return handle if 3rd argument is empty, otherwise use 3rd arg
94 % as handle
95 %=====
96 %
97 function handle = drawTarget(z, handle)
98
99     w = 0.1;
100    h = 0.05;
101    pts = [...
102        w/2, h;...
103        w/2, 0;...
104        -w/2, 0;...
105        -w/2, h;...
106        w/2, h;...
107    ];
108

```

```
109     % translate points
110     pts = pts + repmat([z,0],size(pts,1),1);
111
112     if isempty(handle),
113         handle = fill(pts(:,1), pts(:,2), 'r');
114     else
115         set(handle, 'XData',pts(:,1), 'YData',pts(:,2));
116         drawnow
117     end
118 end
```

The complete solution is given on the wiki associated with the book.