Chris Bonney

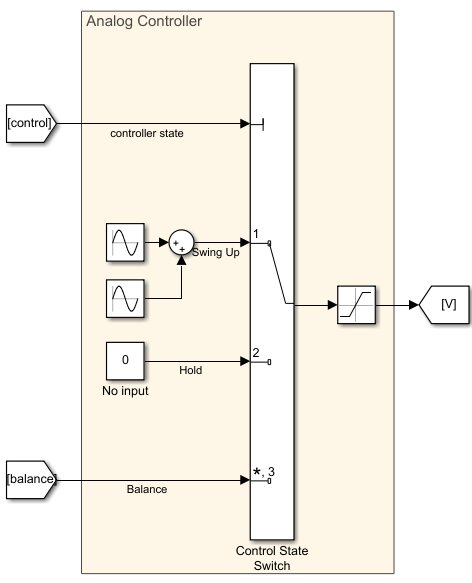
ESE 447

3/30/20

Digital Selector

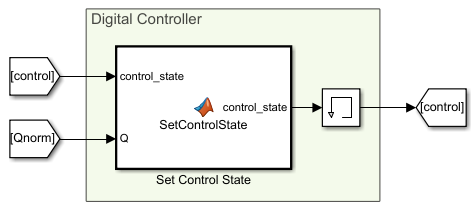
The goal for this section is to create the digital part of a hybrid controller that will balance the pendulum arm of the robot. The purpose of the digital controller is to select the mode of the analog controller.

There will be three modes: swing up, wait, and balance. Swing up will attempt to get the pendulum arm close to vertical, where the balance controller can kick in; this is important because the balance controller will be based on a linearized model of the system, which will only be accurate close to the vertical fixed point. The balance mode will obviously attempt to balance the arm. And finally, the wait mode will feed a 0 input through to the motor, allowing the arm to settle back again. The controller can be made in Simulink using a multiport switch, as shown in Figure 1.



*Figure 1: Multiport Switch for Analog Controller*

Now, we have to do is design logic for the controller. For this we will use a Matlab Function block. The block will take in the previous time step’s control state, as well as a normalized version of the system’s states. This is shown in Figure 2. The logic is as follows: the controller can only go between states in order of



and depends on the second joint’s position and velocity:

* When in balance mode (3), switch to wait (2) if the second joint is more than 25o away from vertical
* When in wait (2), switch to swing up (1) if the second joint is within 10o of straight down, and its angular velocity is less than 0.01 rad/sec
* When in swing up (1), switch to balance (3) if the second joint is within 20o of vertical.

The code for this function is shown in Figure 3.

*Figure 2: Setting the Control State*

function control\_state = SetControlState(control\_state, Q)

q2 = Q(2);

q2dot = Q(4);

q2 = 2\*atan2(tan(q2/2),1);

switch control\_state

case 1

if (abs(q2) < 20\*pi/180)

control\_state = 3;

end

case 2

if (abs(q2) > pi-5\*pi/180 && abs(q2dot) < 0.01)

control\_state = 1;

end

case 3

if abs(q2) > 25\*pi/180

control\_state = 2;

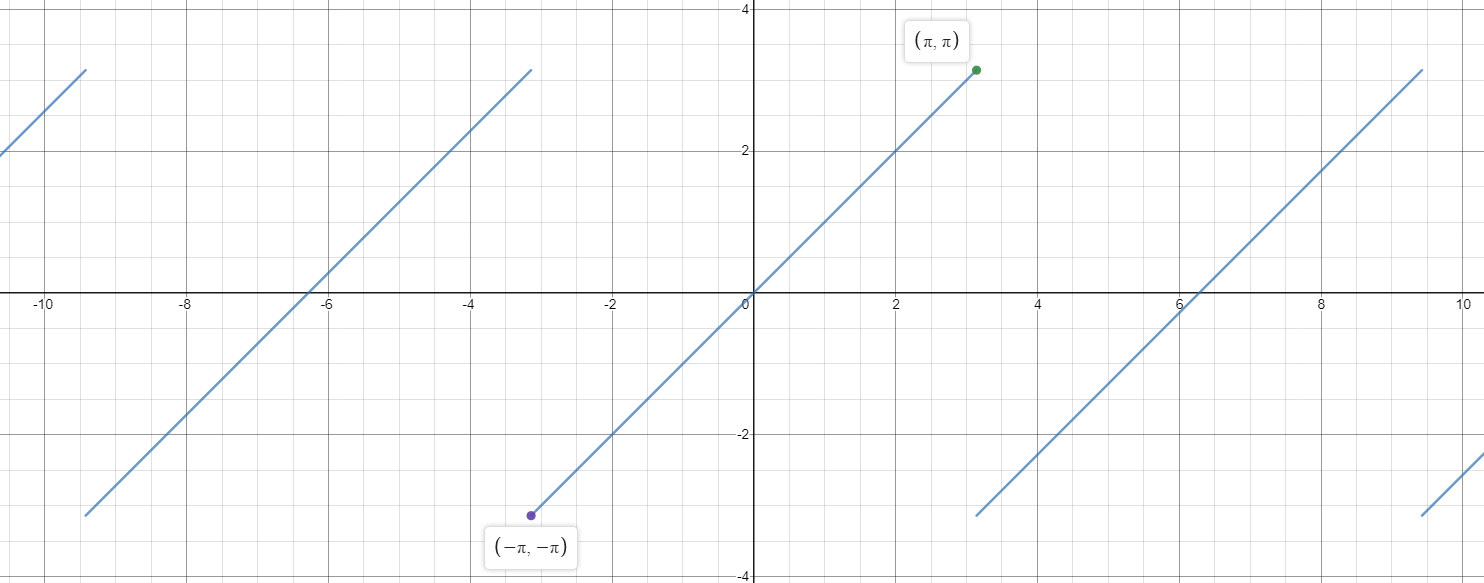
end

end

*Figure 3: Setting the Control State*

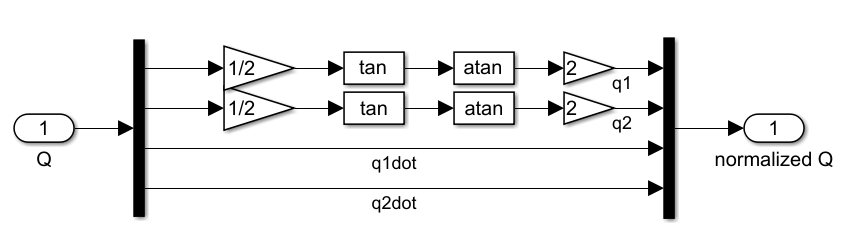
Lastly, we have to normalize the *q2*. The goal is to limit the joint angle to between , with switching between the two endpoints. This is important because the controller uses the angle of the joints relative to 0 for its calculations. To do this, I used the following equation:

I created a graph in Desmos to plot this in Figure 4, and confirm that it matches the behavior desired. This normalization also works for normalizing *q1,* which will be important for the analog controller, even though it is not needed for the digital selector.



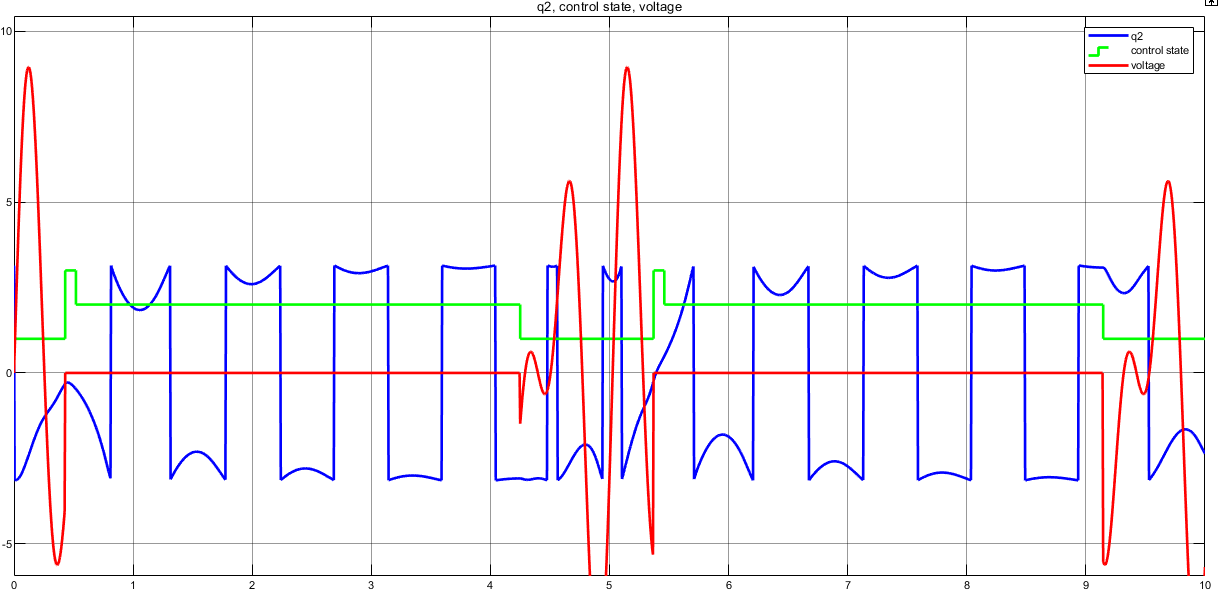
*Figure 4: Normalized Joint Angles*

I created a subsystem in Simulink to do this calculation, shown in Figure 5.



*Figure 5: Trig Normalizer for Joint Angles*

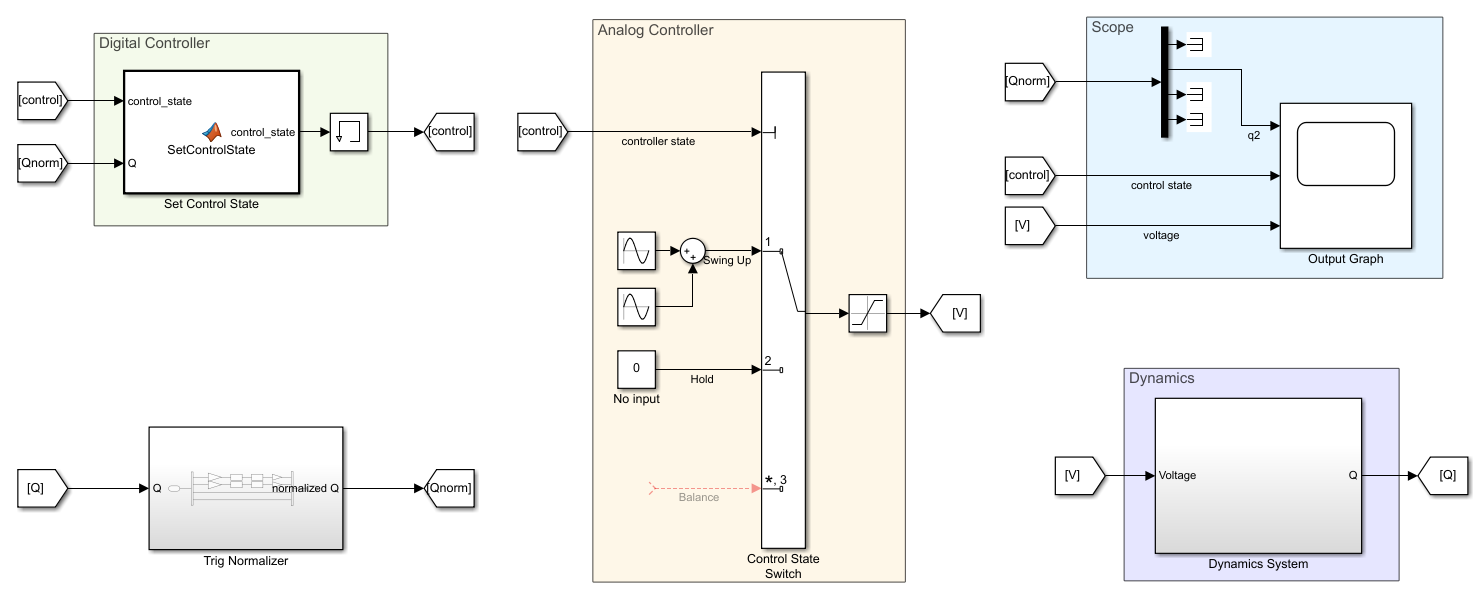
Using a swing up controller already designed, I can show that the behavior of the digital selector is as desired. I ran the full model, plotting the state *q2,* the control state, and the input voltage. This is shown in Figure 6.



*Figure 5: Simulation of Digital Selector with 0 Balance Input*

We can see that at time 0.5, the state *q­2* gets within 20o the control state switches to balance (3), until it falls outside that range, whereupon it goes into wait (2), until the arm has come close to being at rest, after this the process restarts. We can also observe the control input shutting off for states 2 and 3, because for this simulation I set the balance input to be 0, and the wait input should be 0 to allow the arm to swing to rest.

The full Simulink model is shown below in Figure 7.



*Figure 7: Full Simulink Model for Simulating the Digital Controller*