1. Items done this session:

In the end of the last lab, we thought that we had made the last final design works properly. Yet in this time, we found that the model doesn't work accurately well if waiting for few more times of swing. The robot starts with some rapid swing then stop moving, also the q2 reading became abnormal (accumulating error) if run longer time.

There are couples of issues that we still need to fix this time.

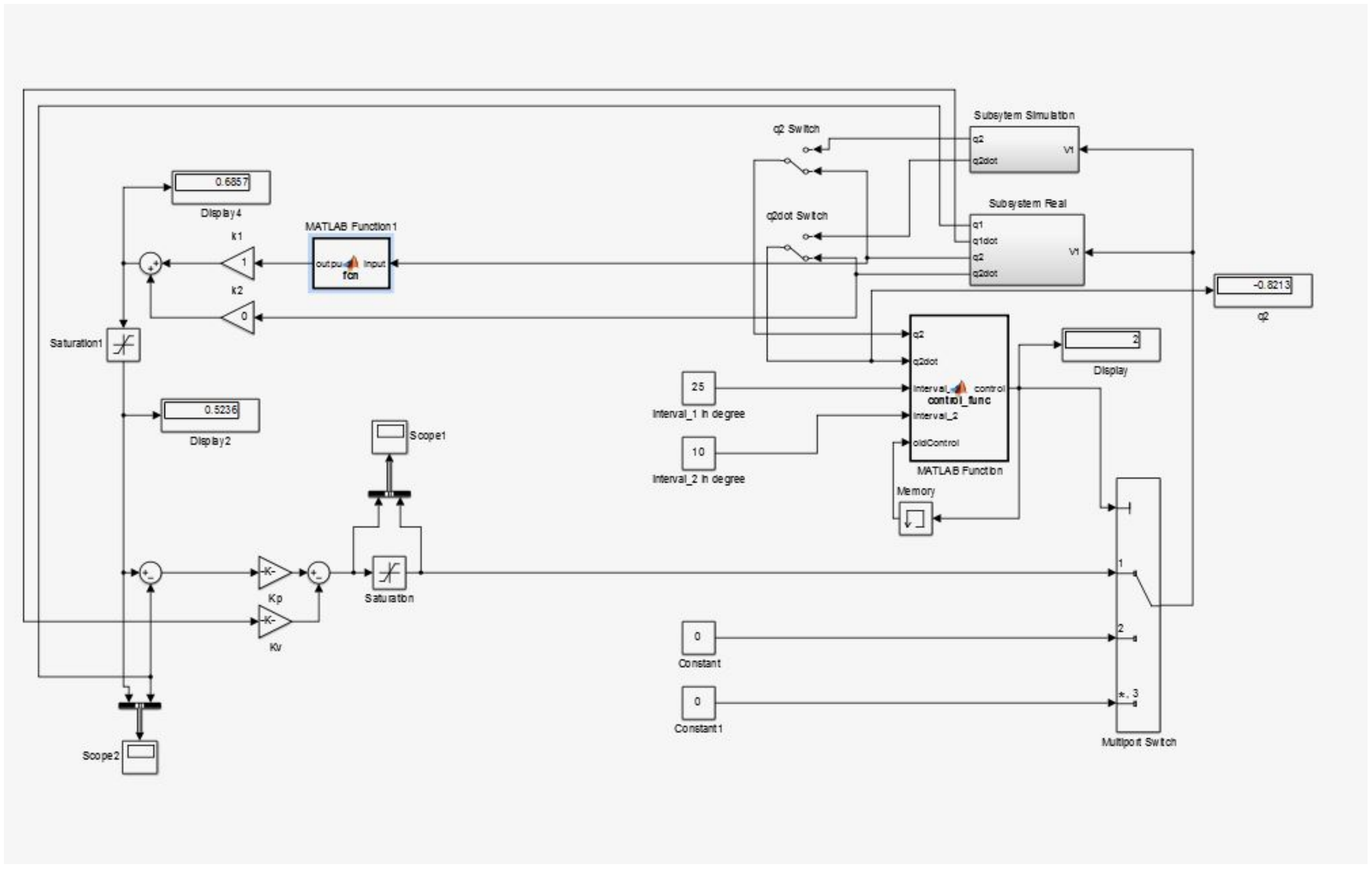
First of all, the sign of after Kv should "negative" rather than "positive", since we're looking for difference (error) of between desired and actual of [q1, q1dot].

The second potential mistake we'd made is using different [q2, q2dot] and [q2_original, q2dot_original] to feed into [k1, k2] and control_func respectively. To fix this, we'd simplified the "Subsystem Real" by deleting the "mod function" previously inside the Subsystem to the original version of only four outputs for the Subsystem, and letting the range transformation job handled outside the subsystem separately. In order to let the "control_func" corporates with the new range of [q2], we added the same atan2 transformation as the function before k1 in the beginning of "control_func" and changed the range in conditional statements correspondingly.

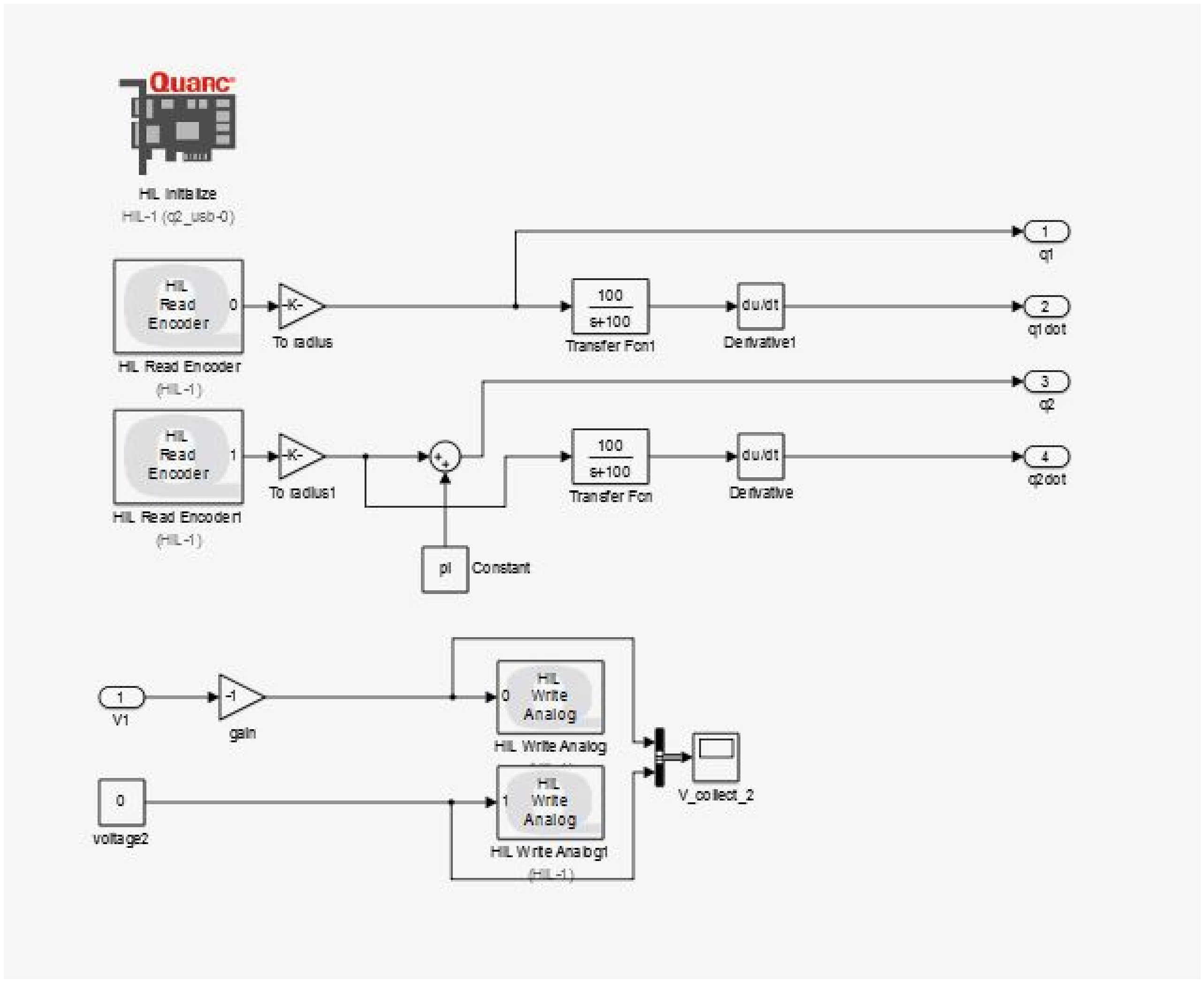
The last few minor changes, we added a negative sign for input voltage and after "HL Read Encoder-0" for [q1, q1dot]. Furthermore, we'd changed the saturation value for [q1, q1dot]_desired from [pi/4, -pi/4] to [pi/6, -pi/6]. Lastly putting the adding condition for the robot to wait in region one until the absolute value of q2dot smaller than the setting value of 0.5.

Then, the model finally succinctly swing toward to top, drop to the down and wait for stick settling down then swing again.

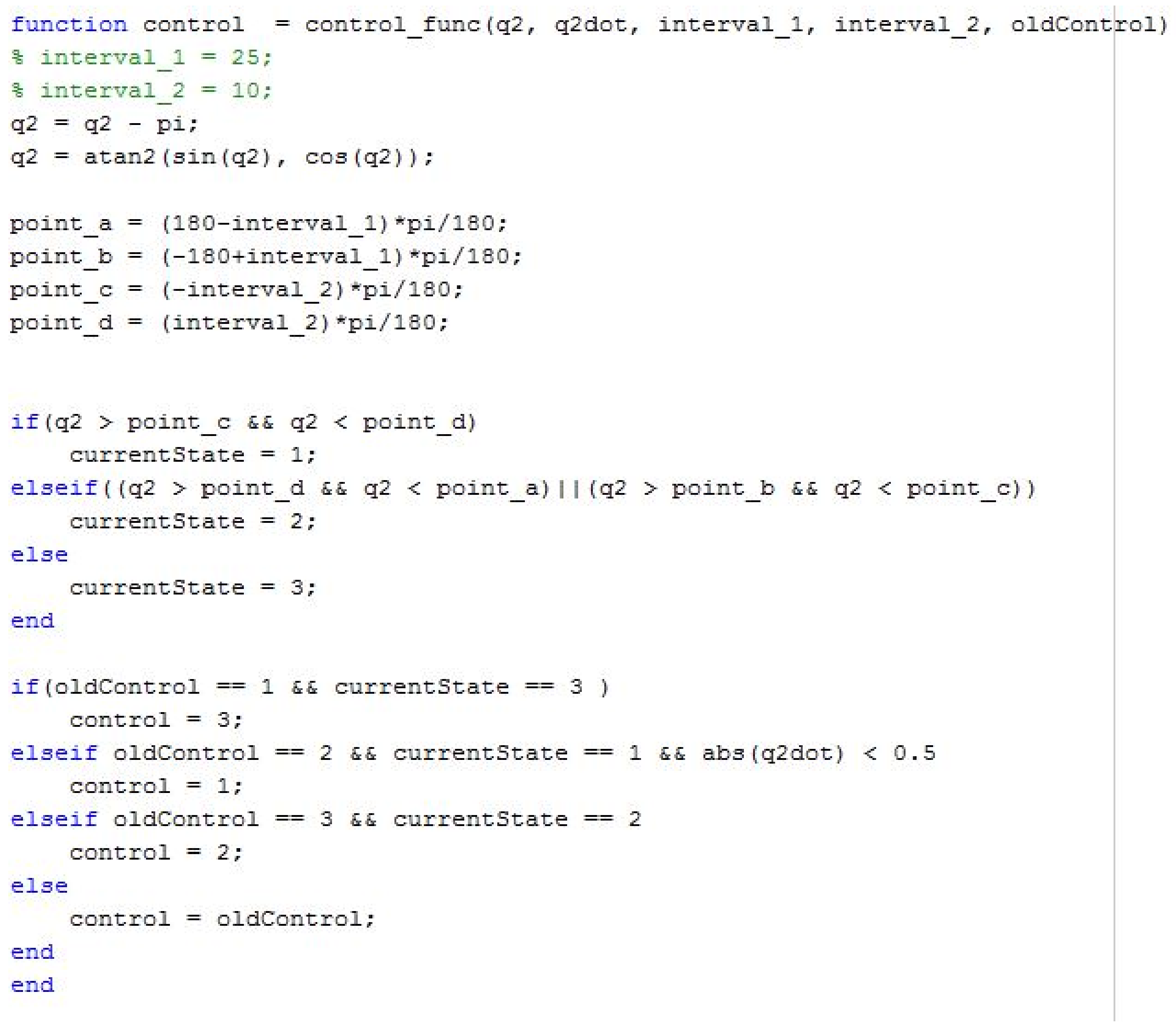
Fig(1): Final Design



Fig(2): Real System



Fig(3): Control Function



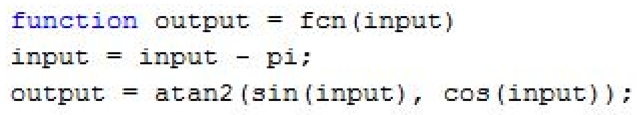
```
function control  = control_func(q2, q2dot, interval_1, interval_2, oldControl)
% interval_1 = 25;
% interval_2 = 10;
q2 = q2 - pi;
q2 = atan2(sin(q2), cos(q2));

point_a = (180-interval_1)*pi/180;
point_b = (-180+interval_1)*pi/180;
point_c = (-interval_2)*pi/180;
point_d = (interval_2)*pi/180;


if(q2 > point_c && q2 < point_d)
    currentState = 1;
elseif((q2 > point_d && q2 < point_a)||(q2 > point_b && q2 < point_c))
    currentState = 2;
else
    currentState = 3;
end

if(oldControl == 1 && currentState == 3 )
    control = 3;
elseif oldControl == 2 && currentState == 1 && abs(q2dot) < 0.5
    control = 1;
elseif oldControl == 3 && currentState == 2
    control = 2;
else
    control = oldControl;
end
end
```

Fig(4): Transfer Function - Transfer q1 from [-inf, +inf] to [-pi, pi].



```
function output = fcn(input)
input = input - pi;
output = atan2(sin(input), cos(input));
```

2. Items for next session:

Keep going on the next assignment for the robot balance on the top region.

3. Problems / Concerns:
It seems working well by now, yet there might be still inaccuracy of the implementation that might be discovered next time.