Project Report

Of

Advanced Control Systems

(ME 8281)

Submitted to,

Prof. Perry Y. Li

By,

Amey Sanjay Joshi

MS, Mechanical Engineering

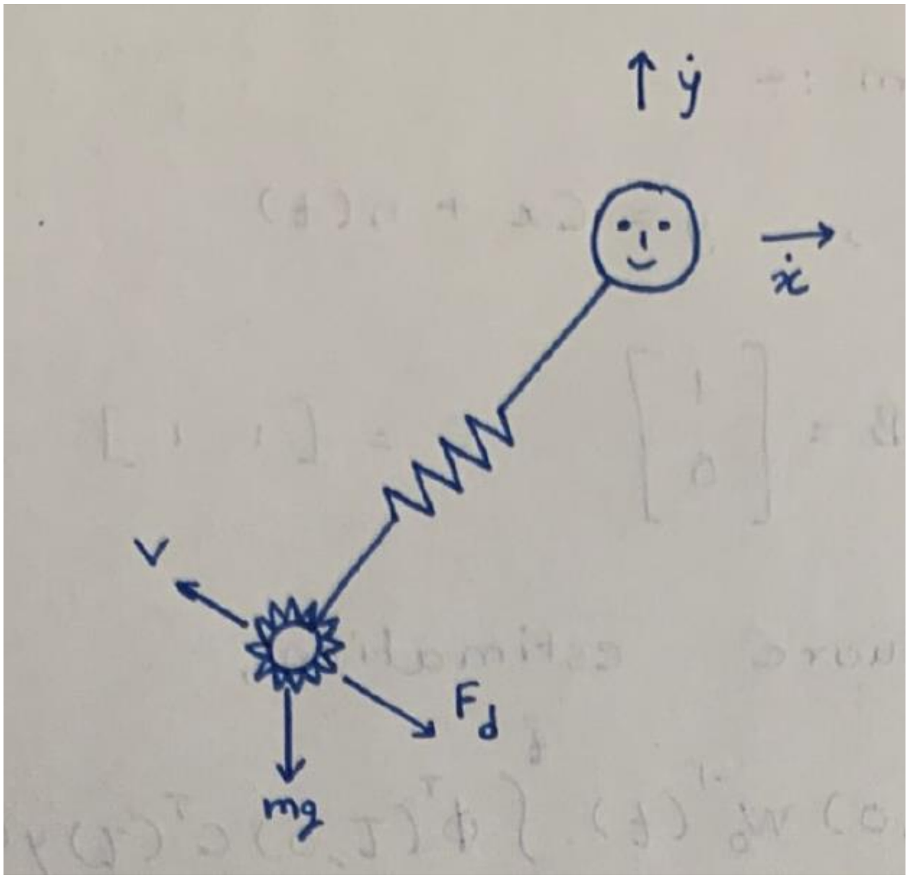
UMN, Twin Cities

Date: - 5/15/2019

Problem Definition:

This is Papi spiky ball system. Basically, this is spring damper system.
In this system, I am considering constants as m_b = 1 Kg, K = 1 N/m, b = 2 N/(m/s).

Free body diagram of system is defined as below,



Equations of system is given as below,

$$m\ddot{x}_b - k(x - x_b) + b\dot{x}_b = 0$$

$$m\ddot{y}_b - k(y - y_b) + b\dot{y}_b = 0$$

$$\dot{x} = u_x$$

$$\dot{y} = u_y$$

$$\ddot{x}_b = -\frac{b}{m}\dot{x}_b + \frac{k}{m}(x - x_b)$$

$$\ddot{y}_b = -\frac{b}{m}\dot{y}_b + \frac{k}{m}(y - y_b)$$

In state space form with A, B, C, D matrices it is written as like below,

$$\frac{d}{dt}\begin{bmatrix} x \\ y \\ x_b \\ y_b \\ \dot{x_b} \\ \dot{y_b} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{k}{m} & 0 & \frac{-k}{m} & 0 & \frac{-b}{m} & 0 \\ 0 & \frac{k}{m} & 0 & \frac{-k}{m} & 0 & \frac{-b}{m} \end{bmatrix} \begin{bmatrix} x \\ y \\ x_b \\ y_b \\ \dot{x_b} \\ \dot{y_b} \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ x_b \\ y_b \\ \dot{x_b} \\ \dot{y_b} \end{bmatrix}$$

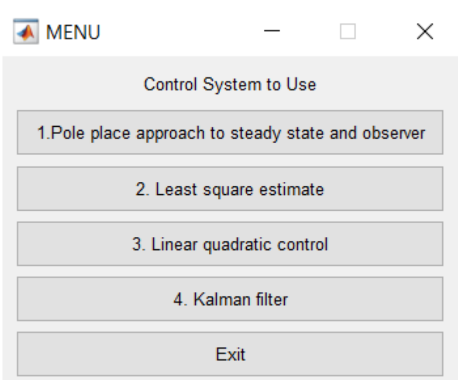
I analyzed this system using 4 control methods.

1. Pole place approach to state-feedback and observer design
2. Least square estimation
3. Linear quadratic control
4. Kalman filter

Although in this system I made following assumptions,

1. There is no gravity
2. Enemies can't kill Papi

I created GUI for the system in which we can choose which control system we want to see, as below.



We can choose whichever control system we want to use and then code will run for that control system.

For all the videos of all control system please follow following google drive link: -

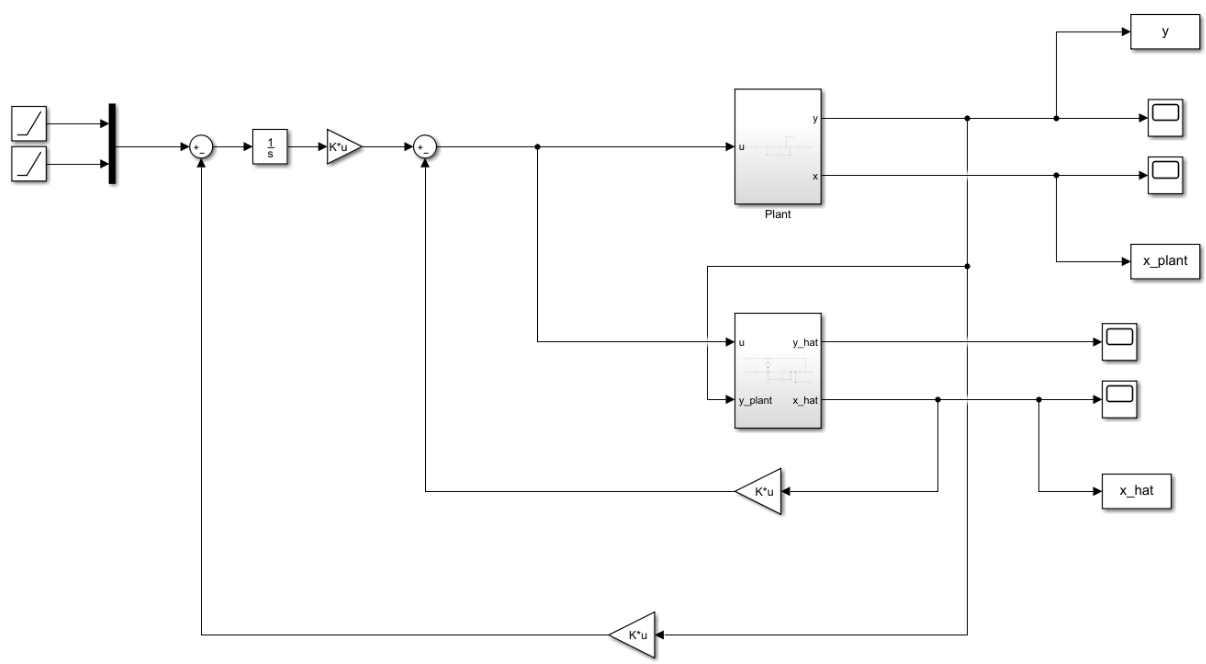
https://drive.google.com/drive/folders/1TCxv9n2Ys-vLjYsKLVV5kQYjgHtGACBR?usp=sharing

Pole place approach to state-feedback and observer design

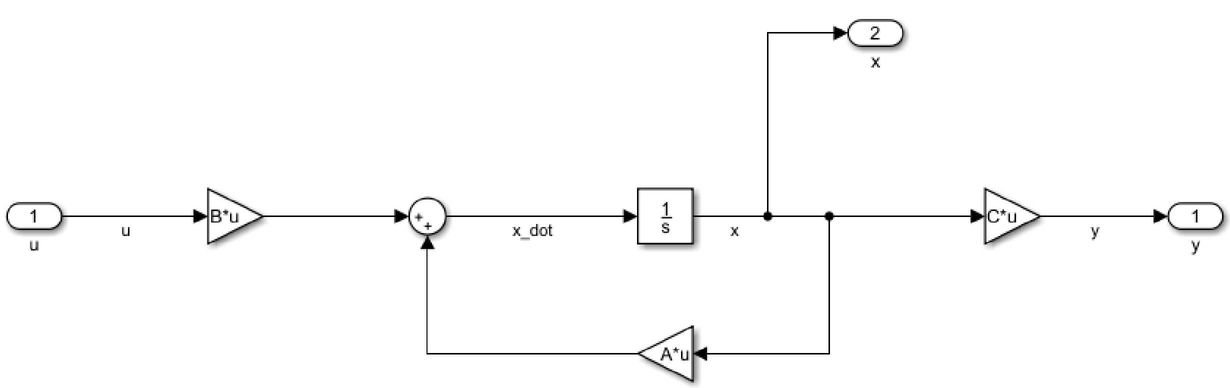
In this method, I designed controller and observer for the system as shown in Simulink figure shown below. I considered to have 0.5 sec settling time with 0.5 damping ratio in both x and y direction. So, for these values we get two poles for each x and y direction at -5. And we can set other poles far from these poles. Similarly, for observer poles. I placed observer poles at 10 times of current desired poles. And then calculated L values for the system using acker MATLAB. Video of overall simulation submitted separately.

I defined 4 enemies, for each case with initial position as (10, 20), (-10, 20), (-10, -20), (10, -20) and moving in straight line with constant velocity with slope (2, -1, 3, 2). In papi spiky ball system, both balls start at (0, 0) origin. This system tracks enemy path and hit enemies after some time.

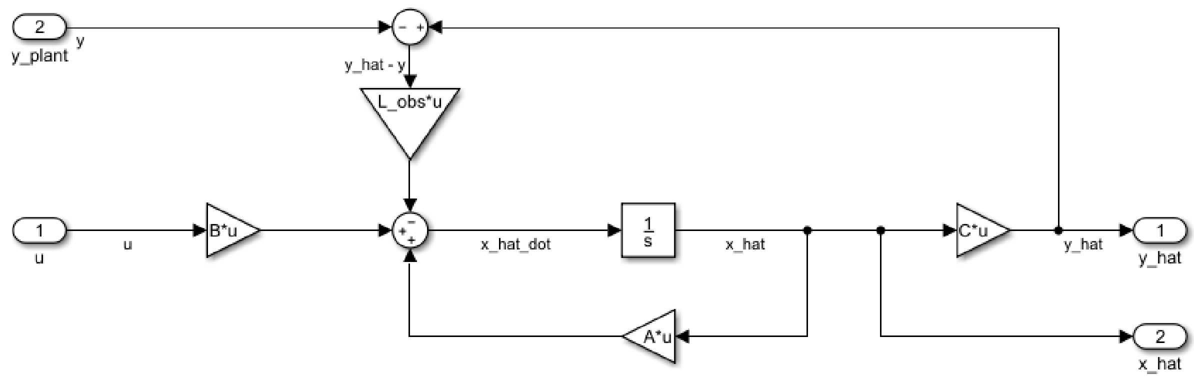
Overall simulink model for the system is as below,



Plant model is as below,

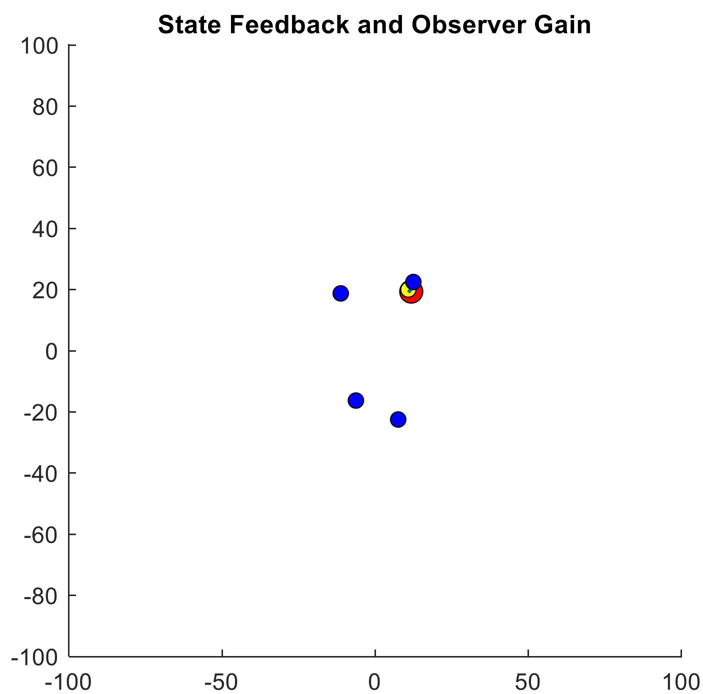


Observer model is below.

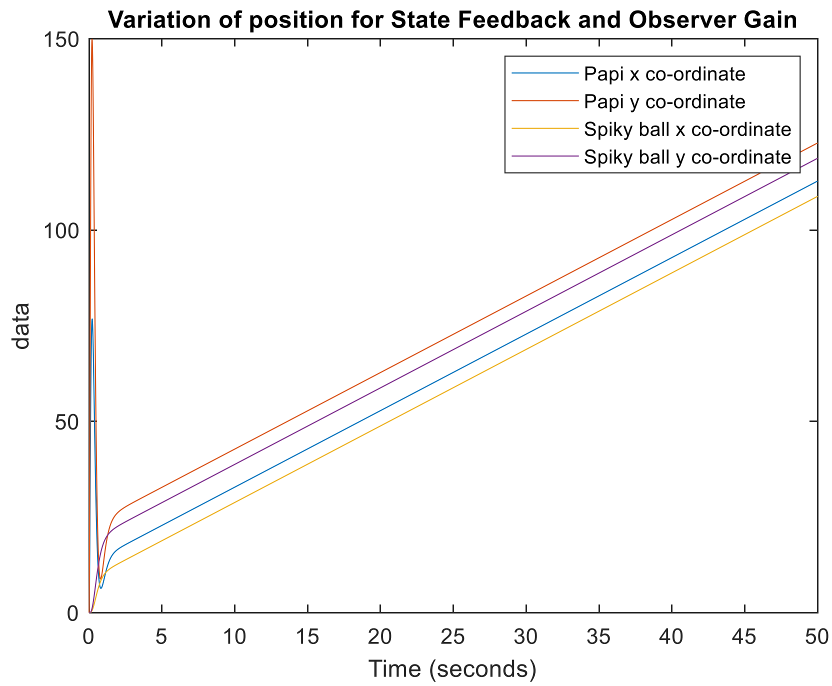


Figures when papi hit enemies and path of x y co-ordinates of papi, spiky balls. For all conditions are as below.

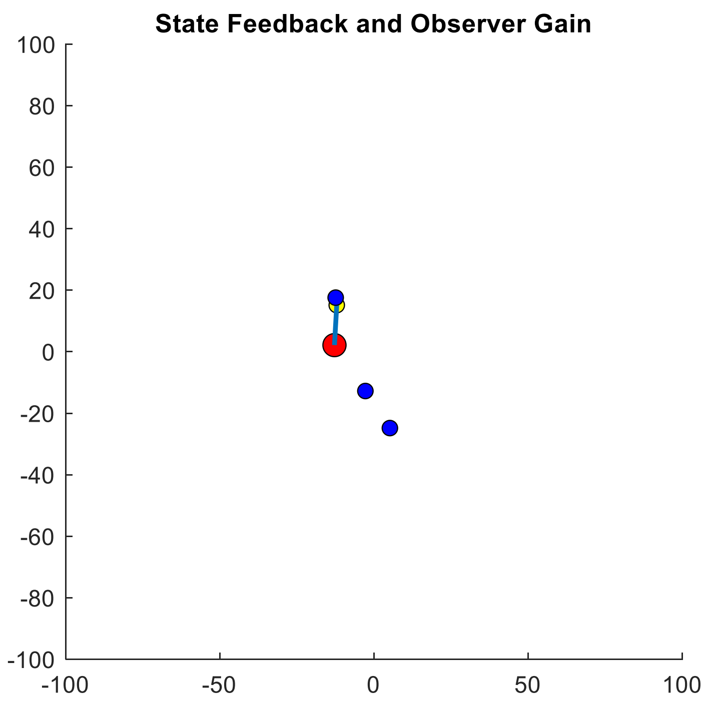
First enemy, when spiky ball hit enemy.



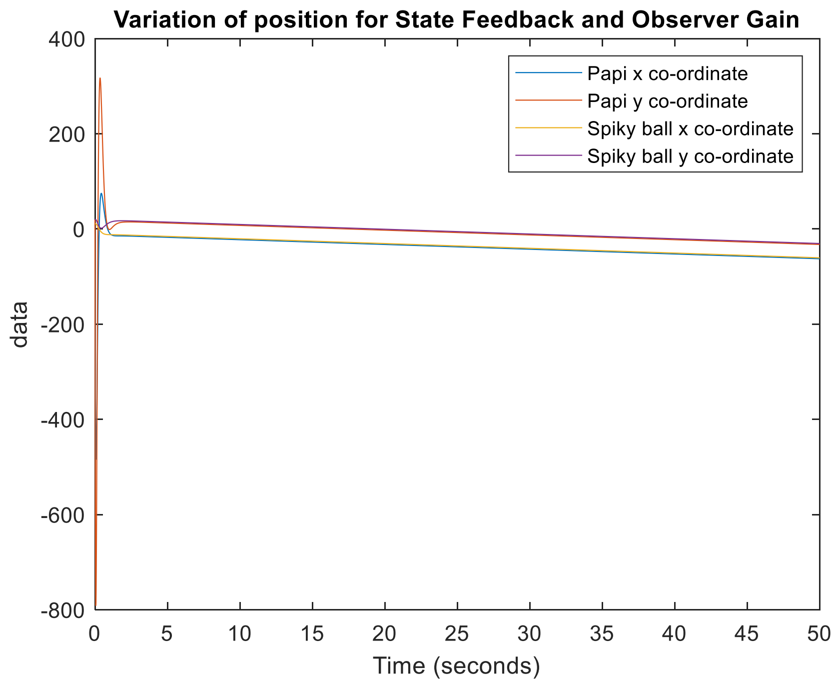
X Y co-ordinates of spiky ball and spiky ball.



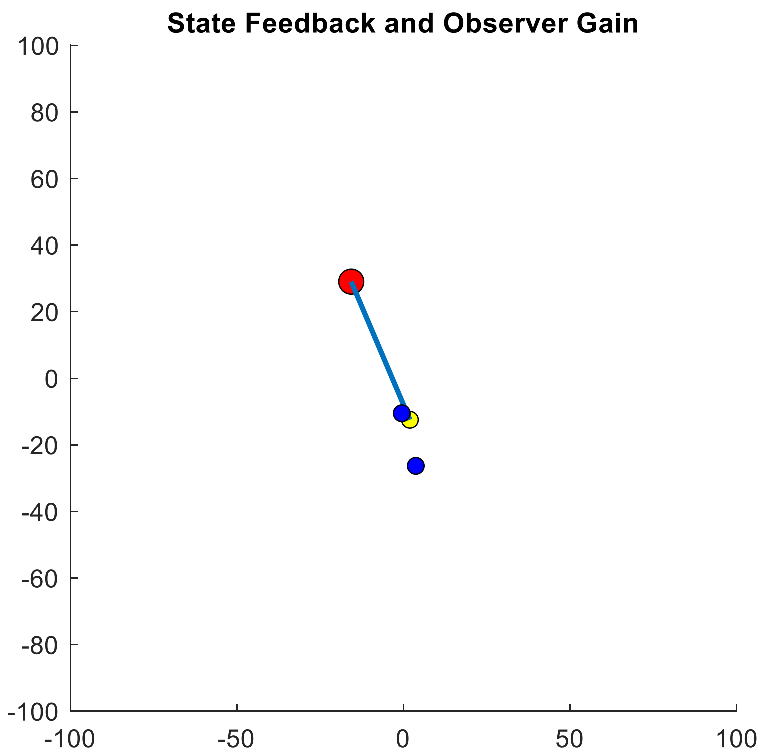
Second enemy, when spiky ball hit enemy,



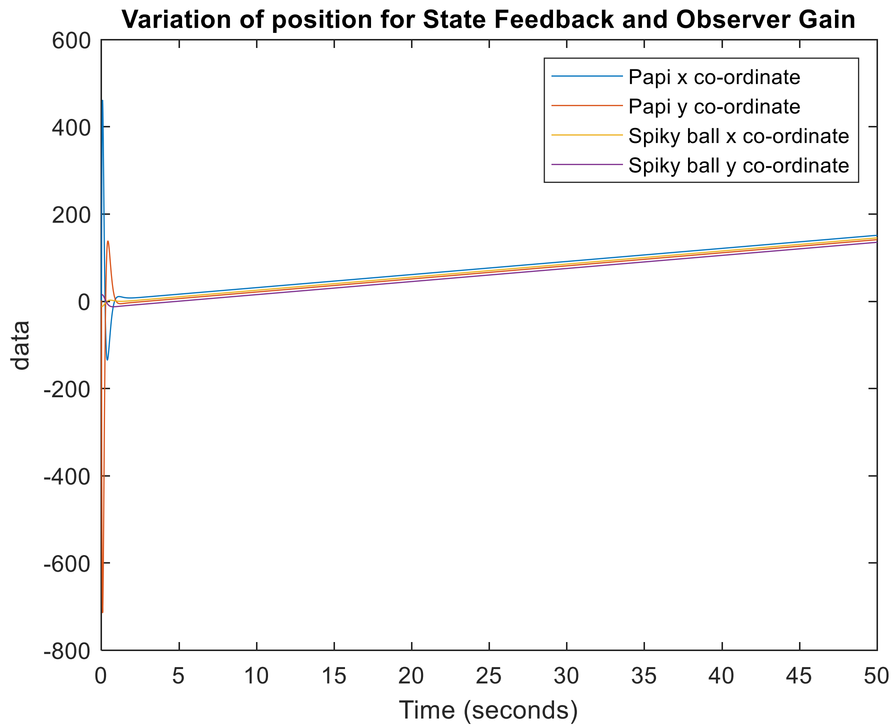
X Y co-ordinates of Papi and Spiky ball.



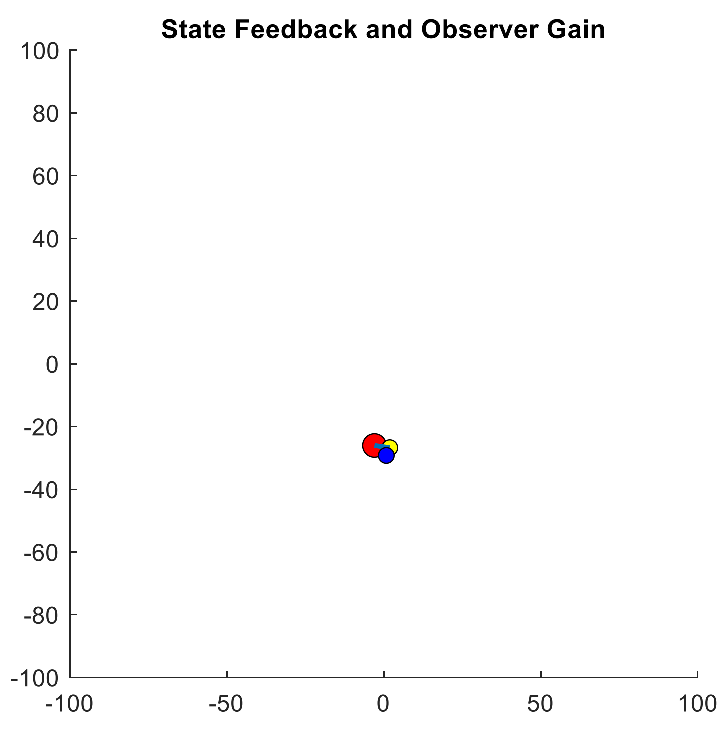
Third enemy, when spiky ball hit enemy,



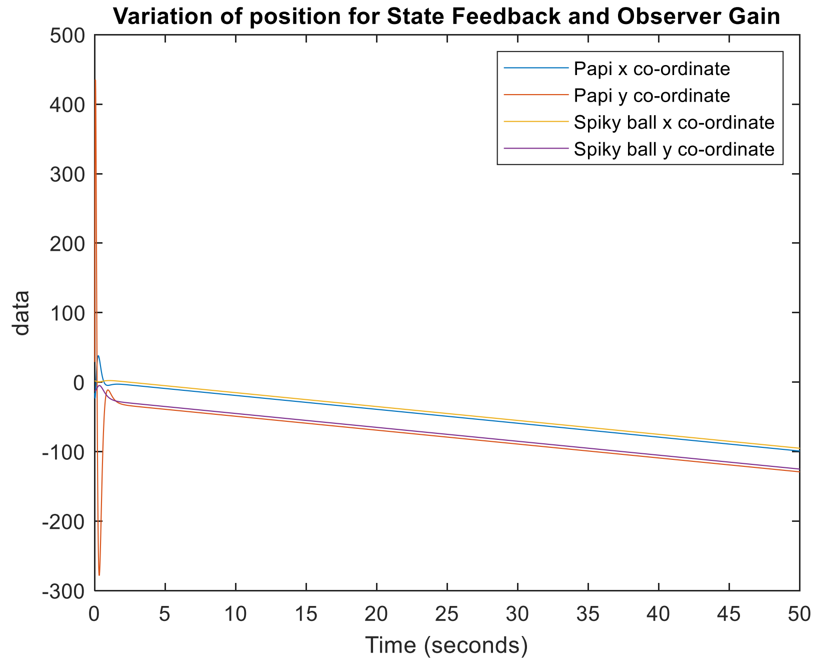
X Y co-ordinates of Papi and Spiky ball



Forth enemy, when spiky ball hit enemy,



X Y co-ordinates of Papi and Spiky ball,



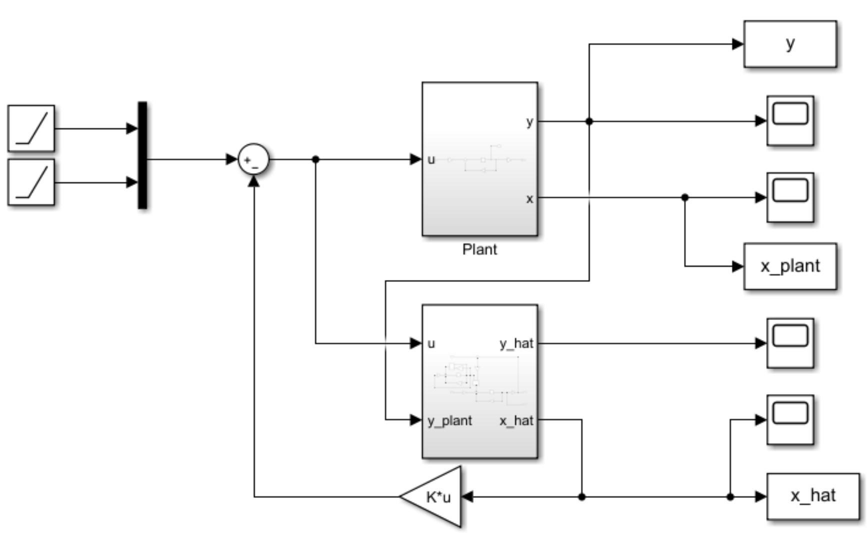
Least Square Estimate

In this method, I designed observer based on y output from plant and u input. L value for observer is designed using riccati equation. I added riccati equation, in simulink to calculate P value and multiply with C transpose to get L value. Riccati equation is as below,

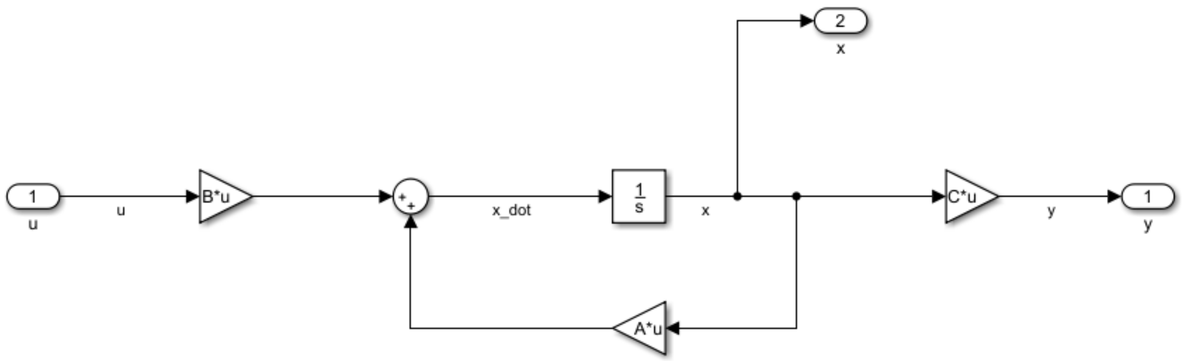
$$\dot{P(t)} = A(t)P(t) + P(t)A^T(t) - P(t)C^T(t)C(t)P(t)$$

Other enemy, plant design is same as before. Video of overall simulation is submitted separately.

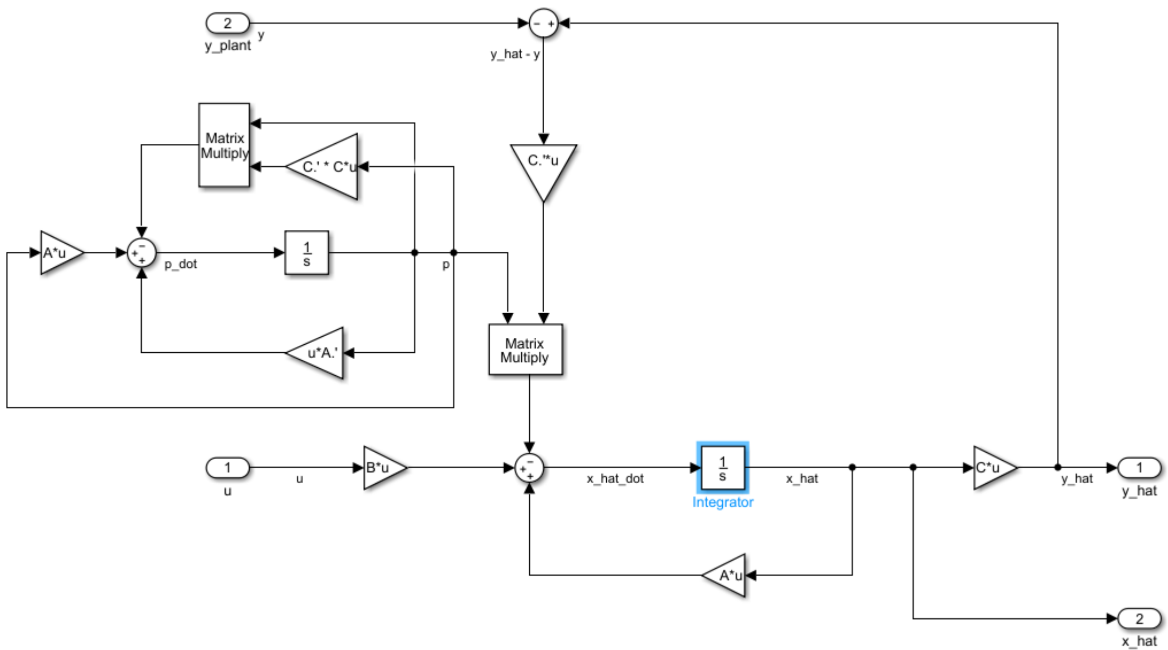
Overall Simulink model for system is below,



Plant model is same and is as below,



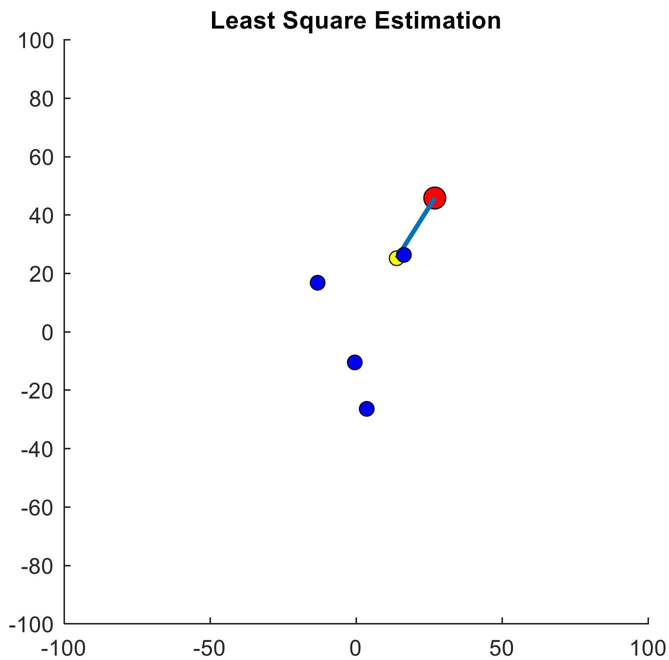
Observer model is as below,



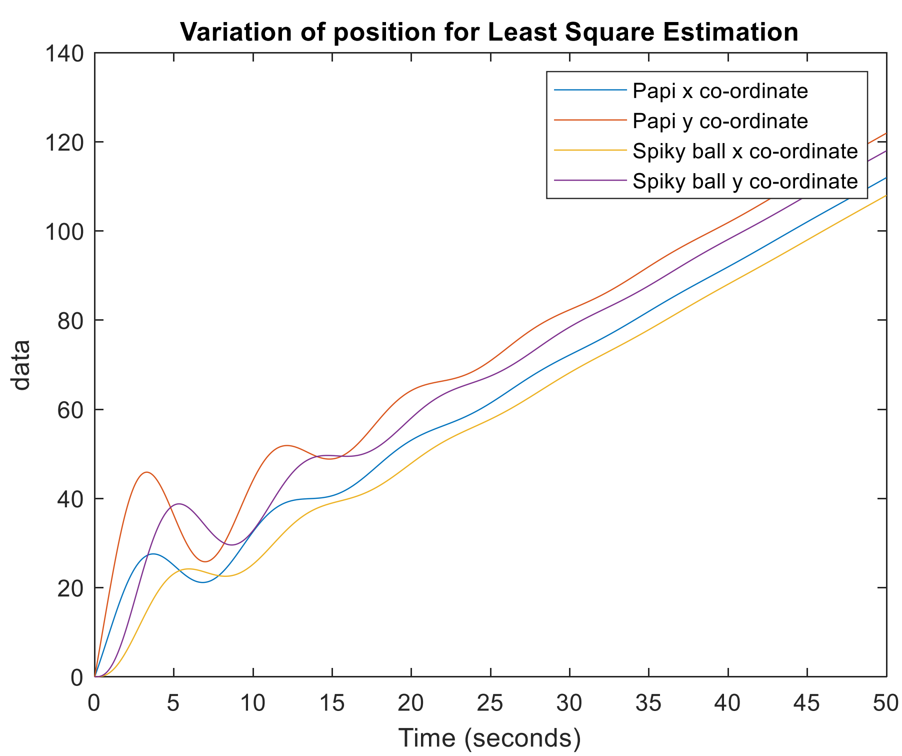
As you can see that, I added, riccati equation, P value in observer and multiply with C transpose to get L value. Everything else is almost same as previous one. But in this case, we don't have to vary pole values to get desired output. System calculates L value depending on P infinite value and most optimized value of L we will get.

Figures when papi hit enemies and path of x y co-ordinates of papi, spiky balls. For all 4 enemies are as below.

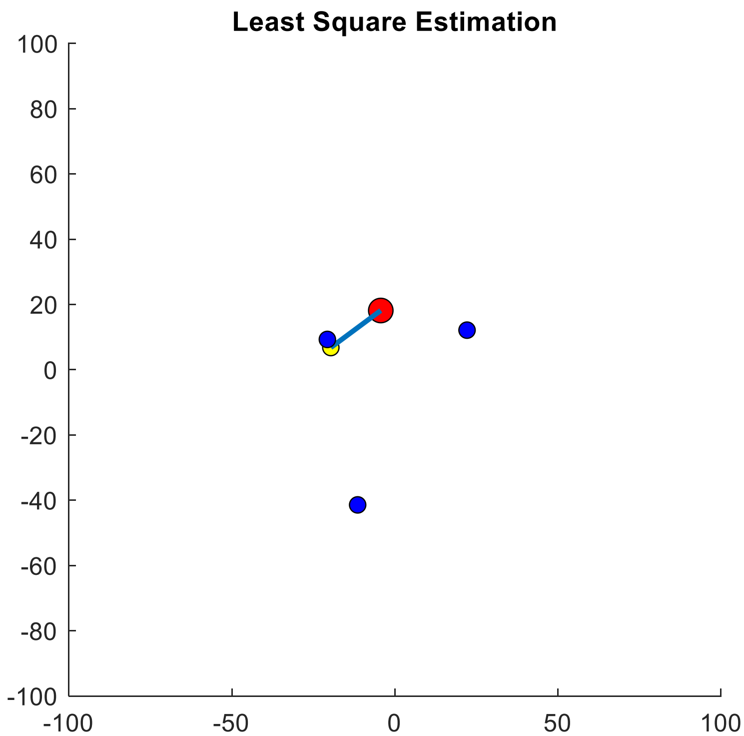
First enemy, when spiky ball hit enemy.



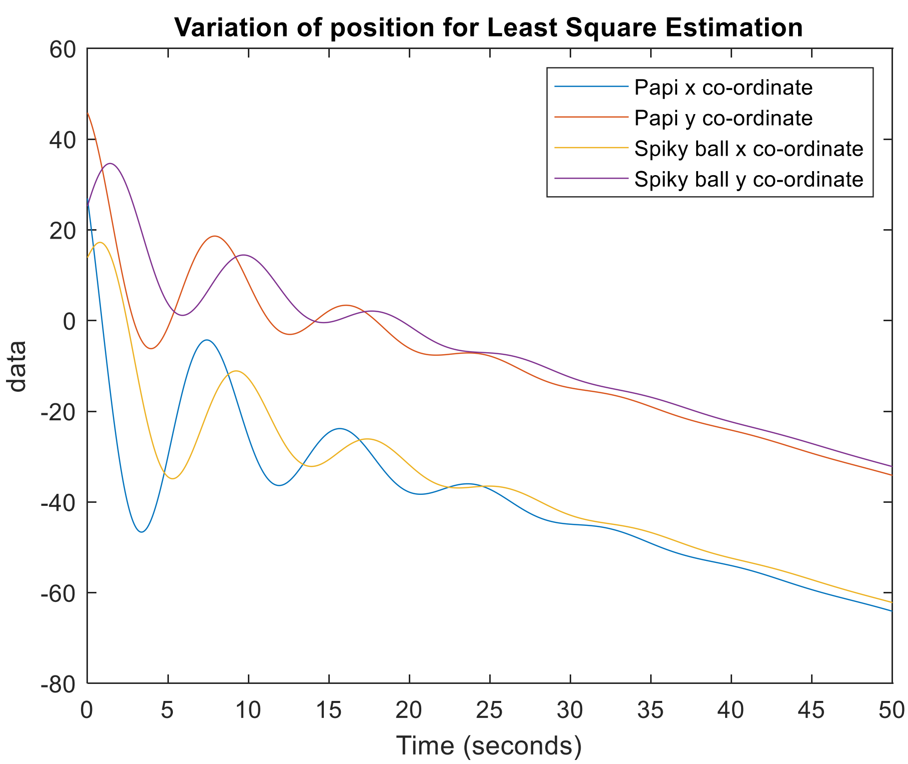
X Y co-ordinates of papi and spiky ball,



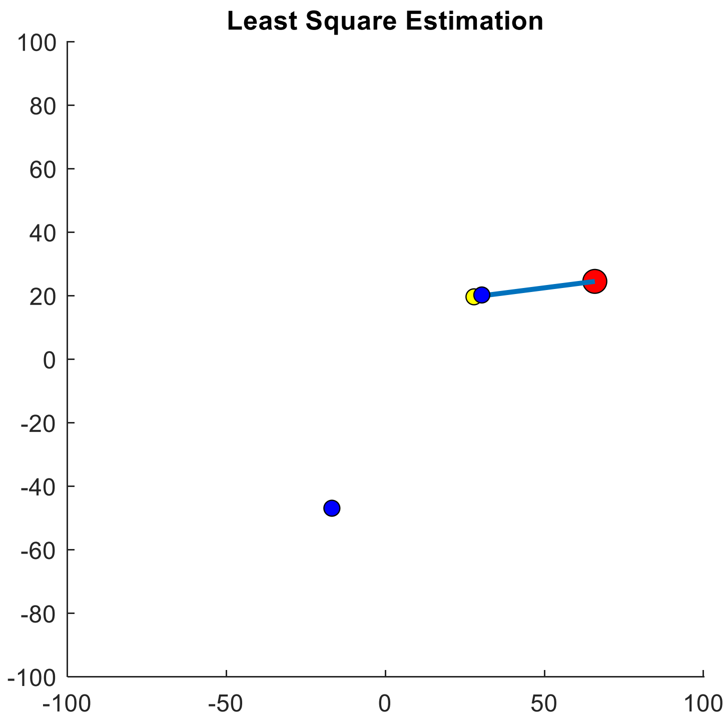
Second enemy, when spiky ball hit enemy,



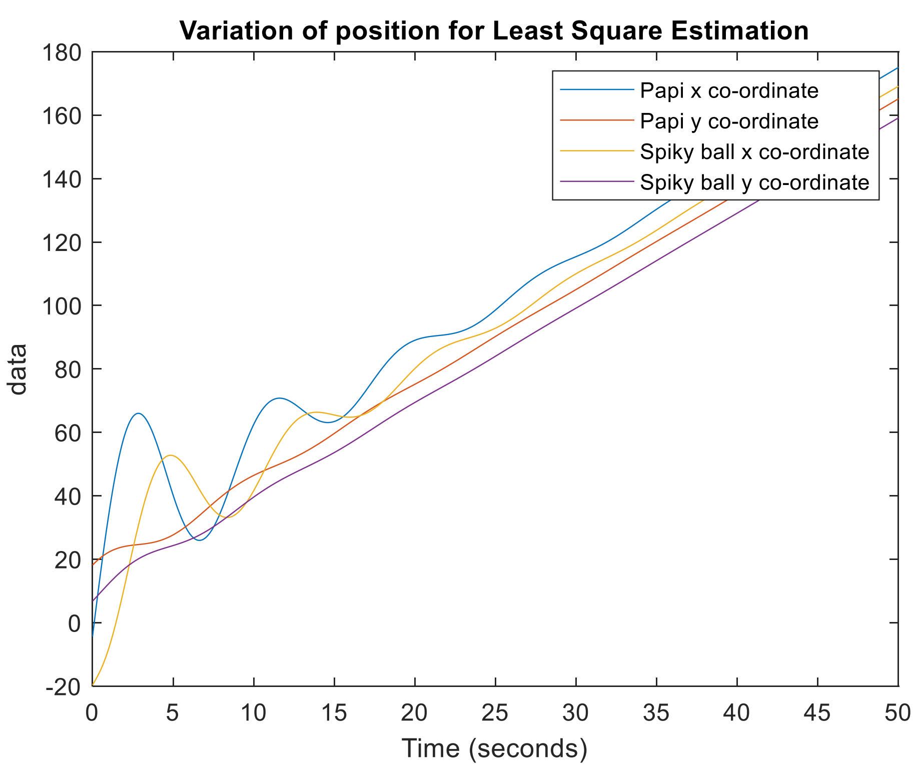
X Y co-ordinates of papi and spiky ball,



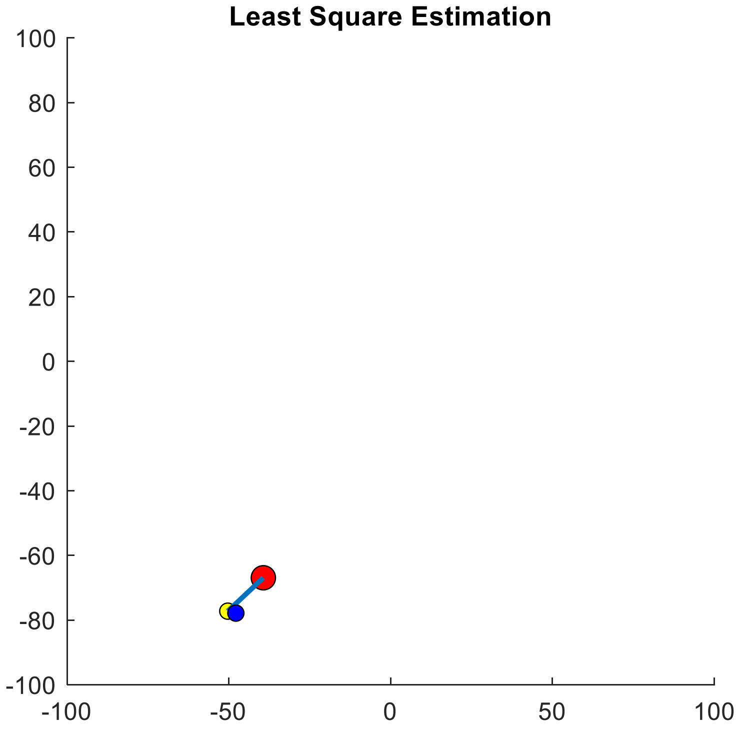
Third enemy, when spiky ball hit enemy,



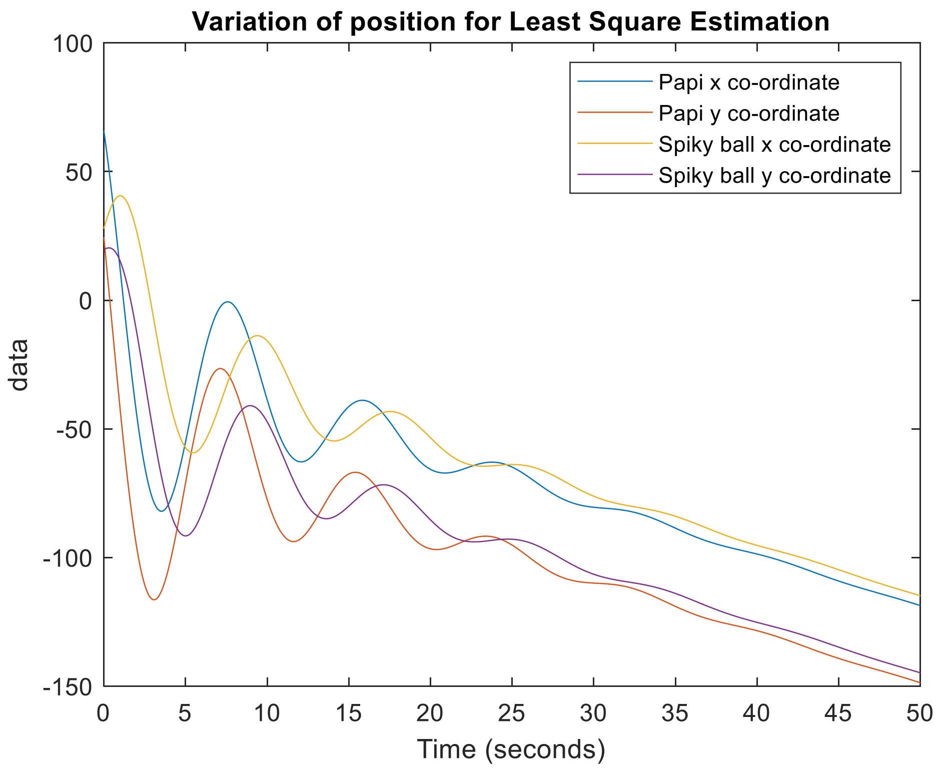
X Y co-ordinates of papi and spiky ball,



Fourth enemy, when spiky ball hit enemy,



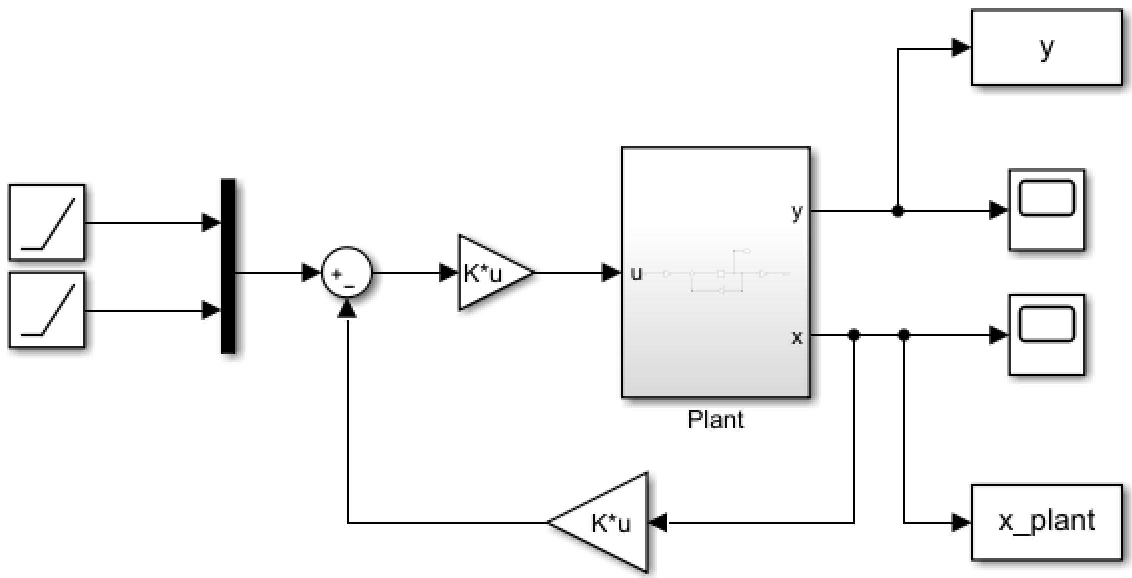
X Y co-ordinates of papi and spiky ball,



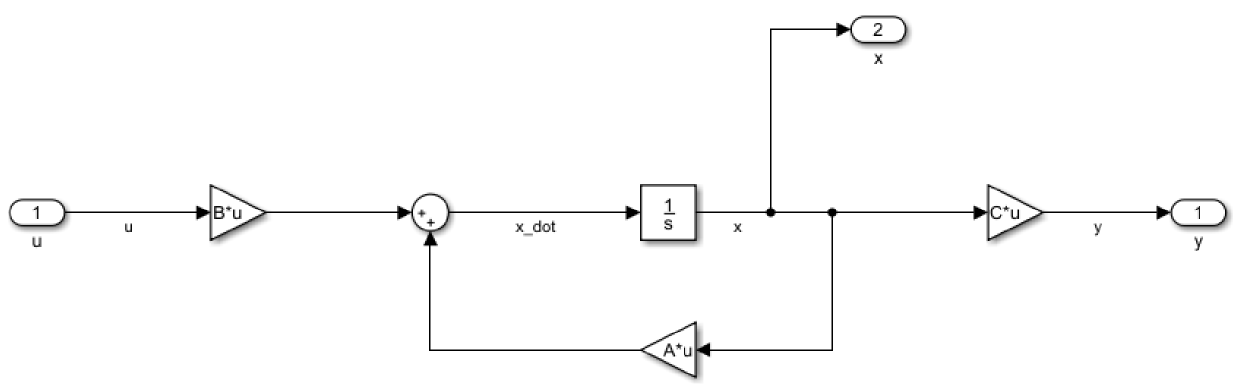
Linear Quadratic Control

In this system, I choose Q and R matrix based on cost function. I defined very simple cost function in such a way that, Q is identity matrix and R is 0.1. Then I find out K values in MATLAB using acker function and implemented that in MATLAB Simulink. Other enemy, plant design is same as before. Video of overall simulation is submitted separately.

Overall Simulink model for system is below,

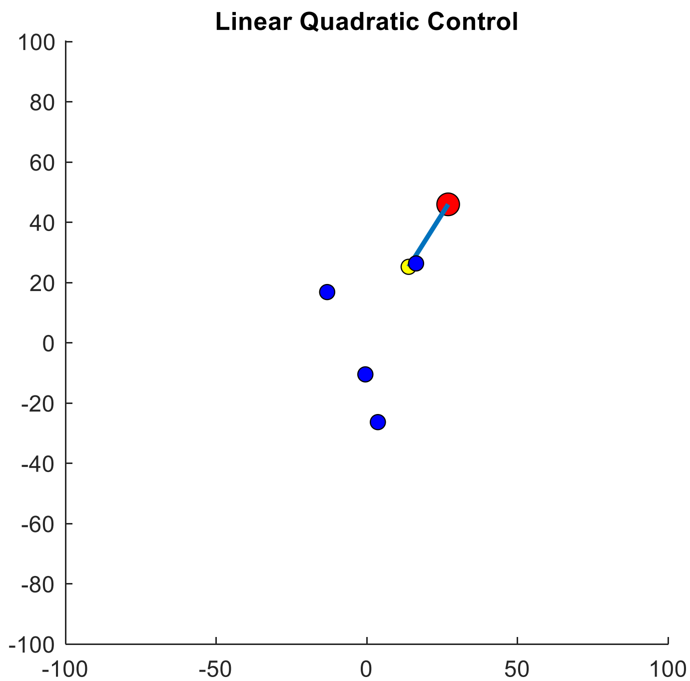


Plant model is as below,

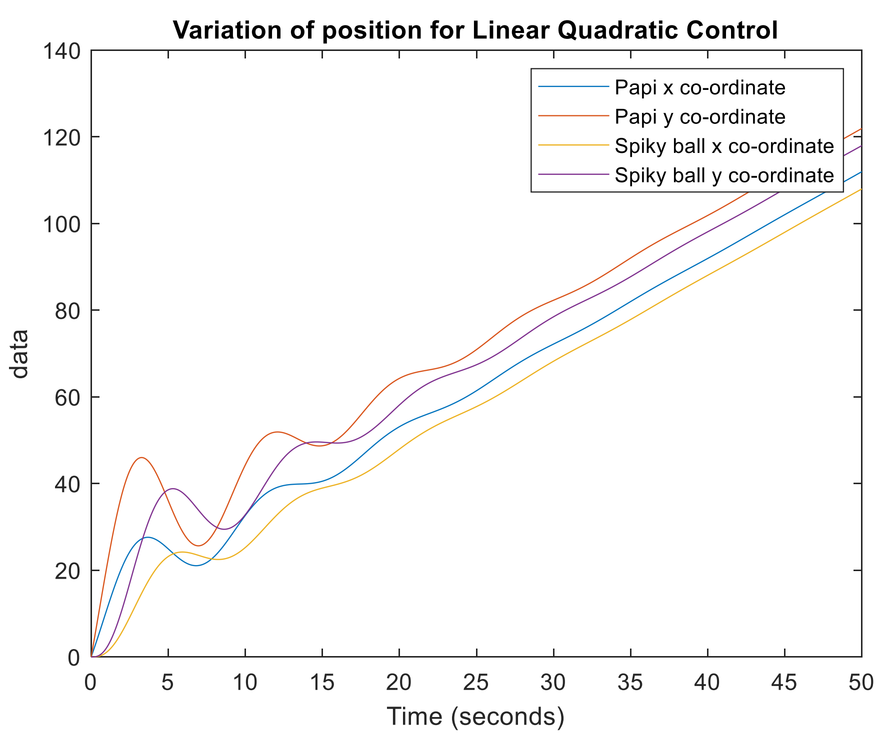


Figures when papi hit enemies and path of x y co-ordinates of papi, spiky balls. For all 4 enemies are as below.

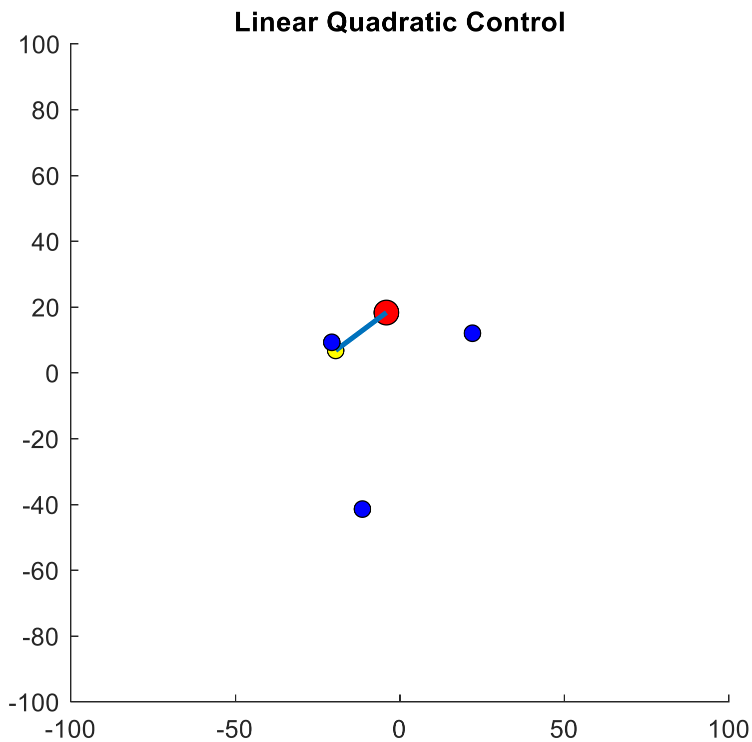
First enemy, when spiky ball hit enemy.



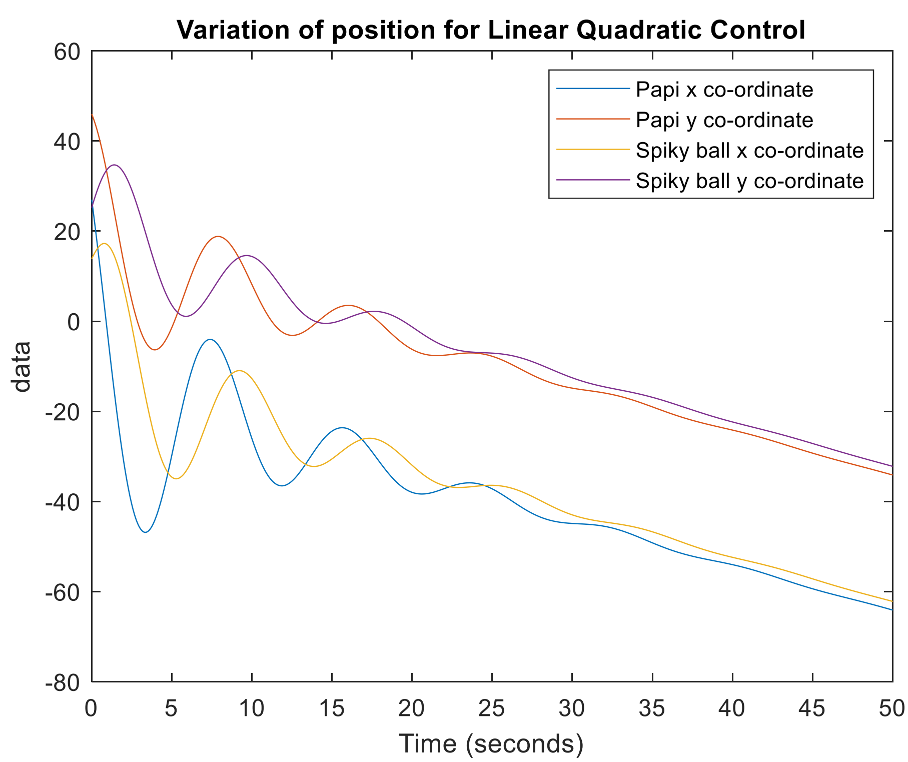
X Y co-ordinates of papi and spiky ball.



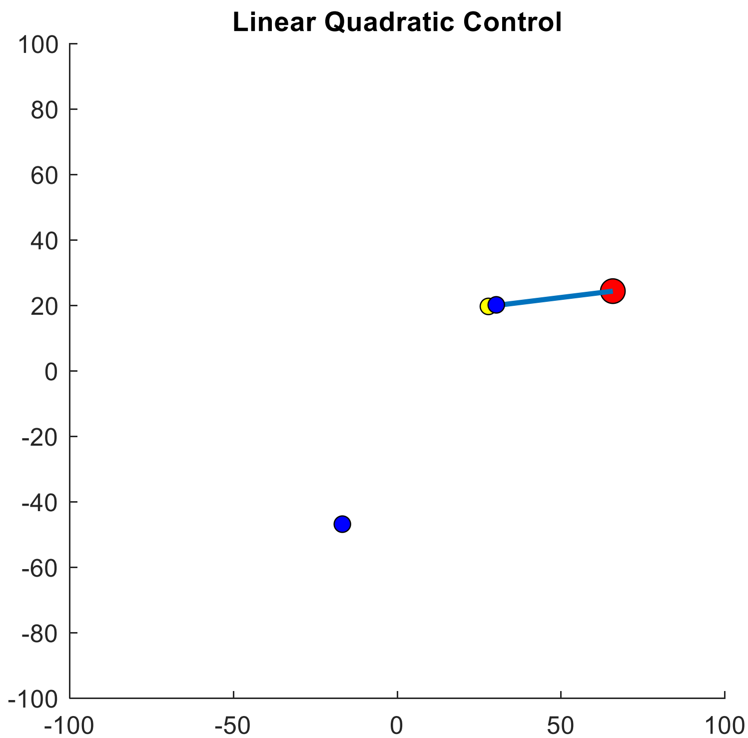
Second enemy, when spiky ball hit enemy.



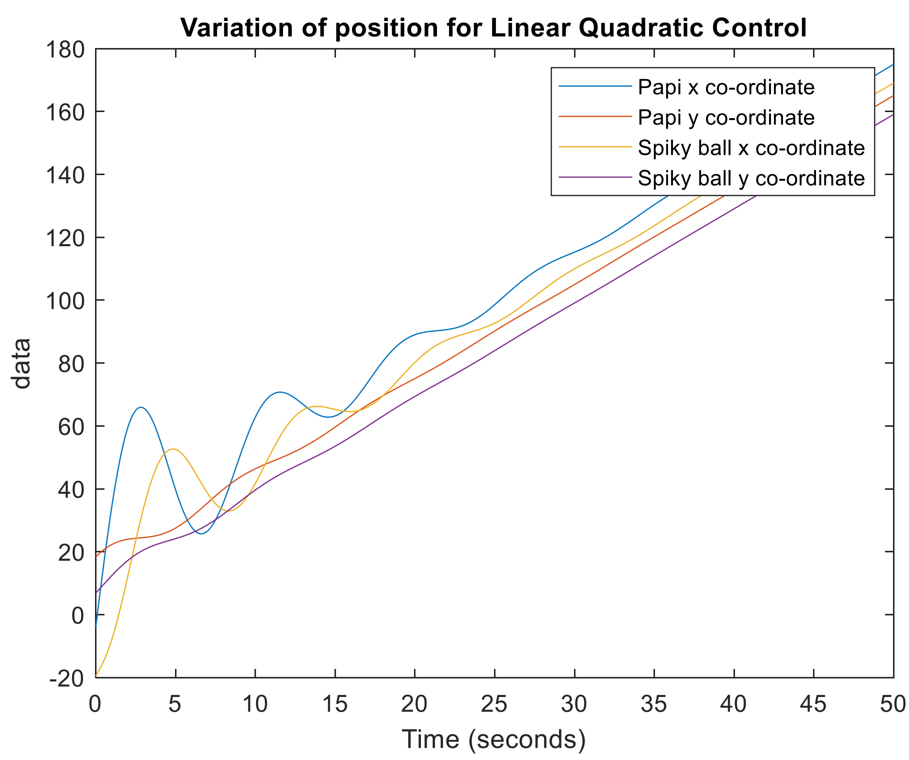
X Y co-ordinates of papi and spiky ball.



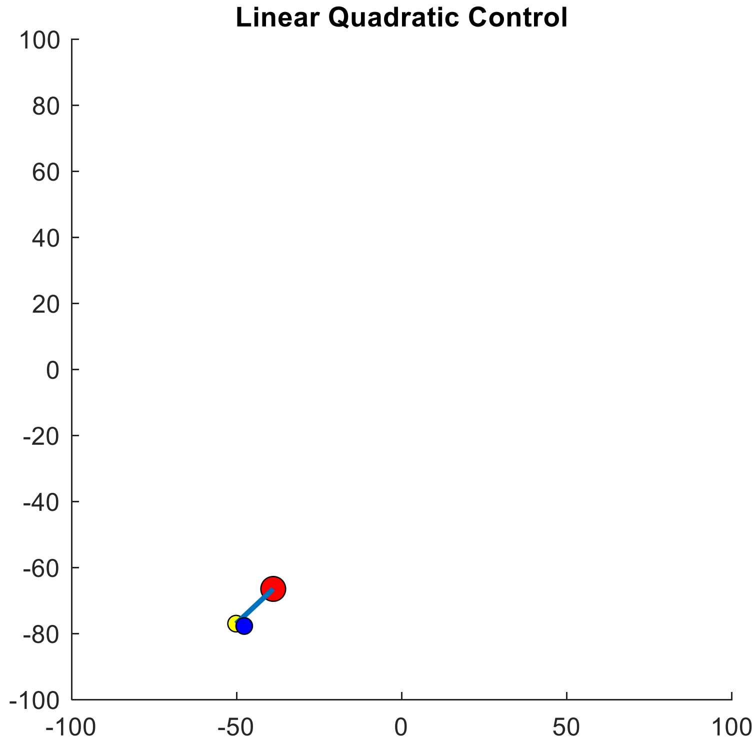
Third enemy, when spiky ball hit enemy.



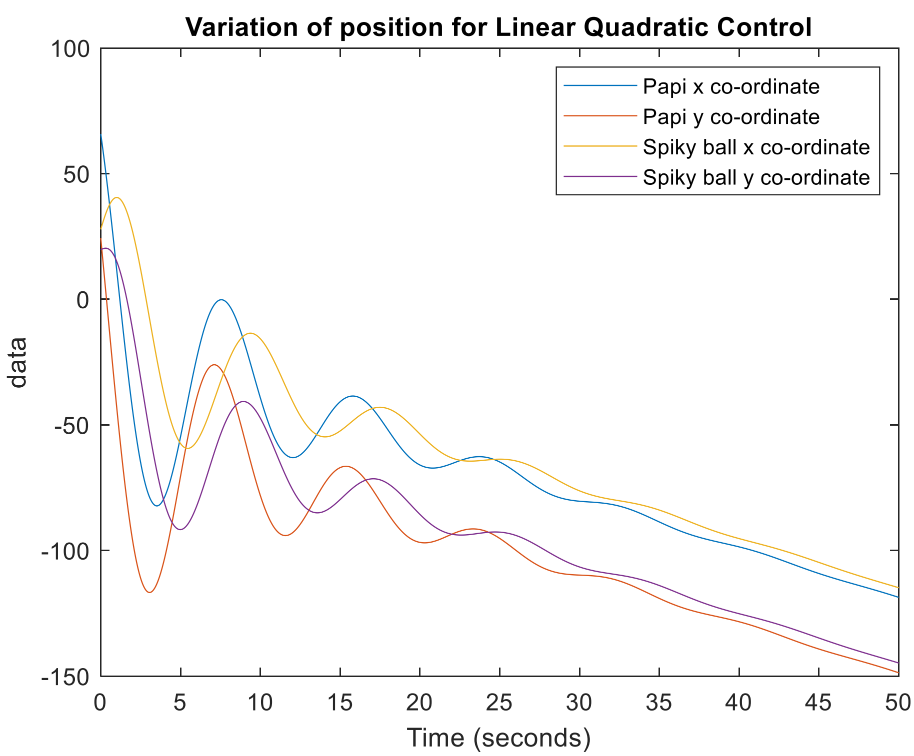
X Y co-ordinates of papi and spiky ball.



Fourth enemy, when spiky ball hit enemy.



X Y co-ordinates of papi and spiky ball.

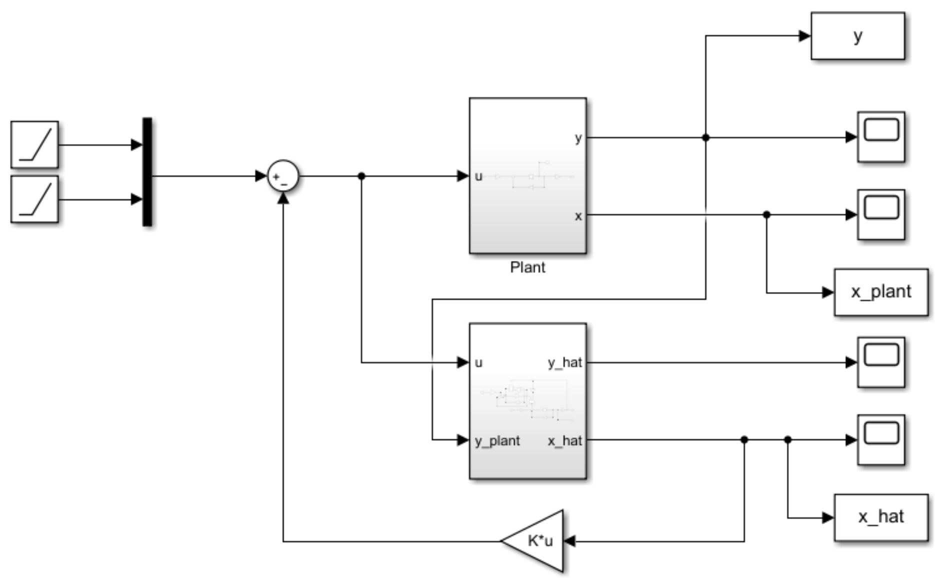


Kalman Filter

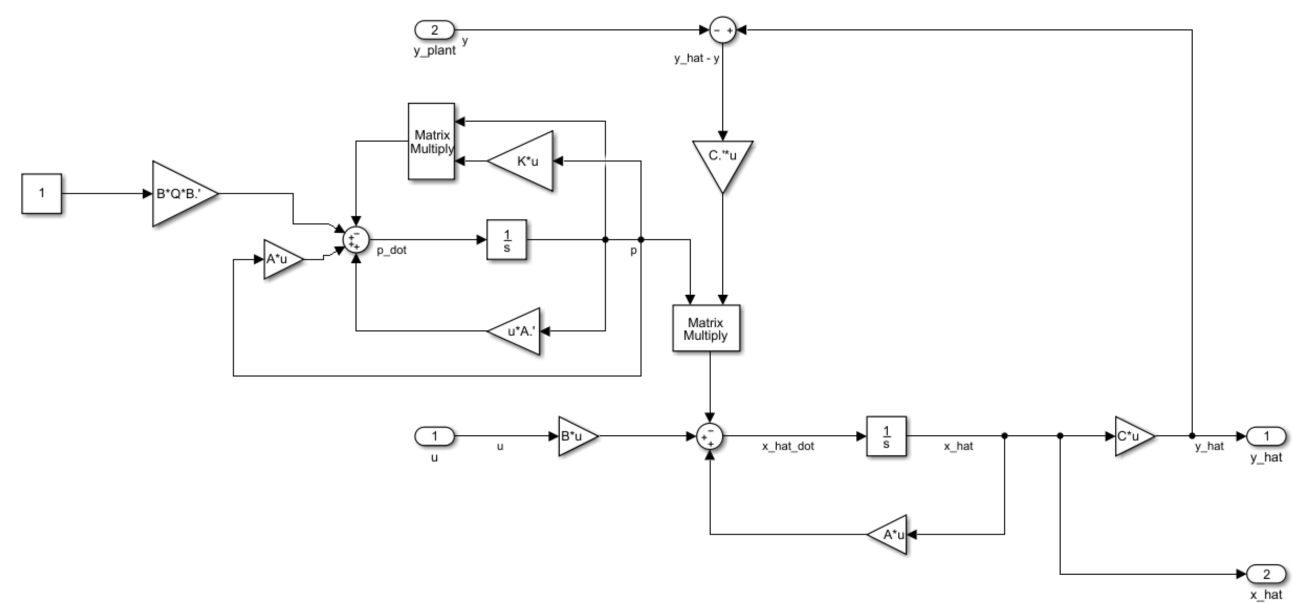
In this system, I designed an observer. I choose Q and R values from cost function. I assumed simple cost function in which, Q is identity matrix and R is 300. Then I use, riccati equation to find out P values to calculate L. Riccati equation, which I used is as below,

$$\dot{P} = A(t)P + PA^T(t) + B_w(t)Q(t){B_w}^T(t) - PC^T(t)R^{-1}(t)C(t)P$$

Other enemy, plant design is same as before. Video of overall simulation is submitted separately. Overall Simulink model for system is below,



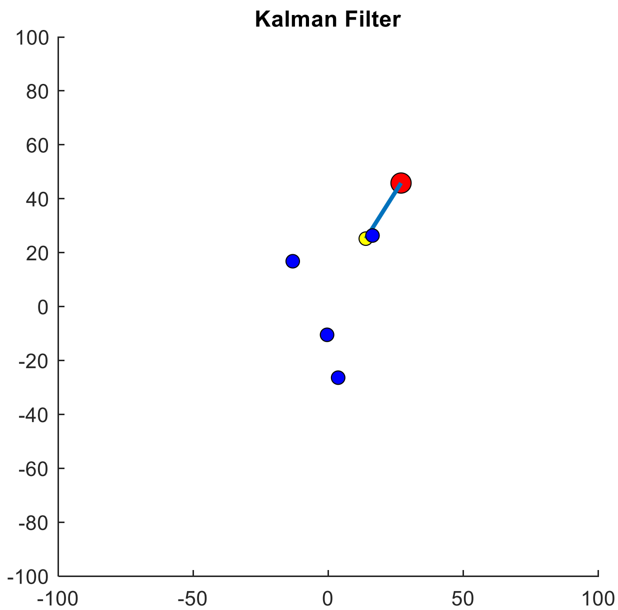
Observer model of system is as below,



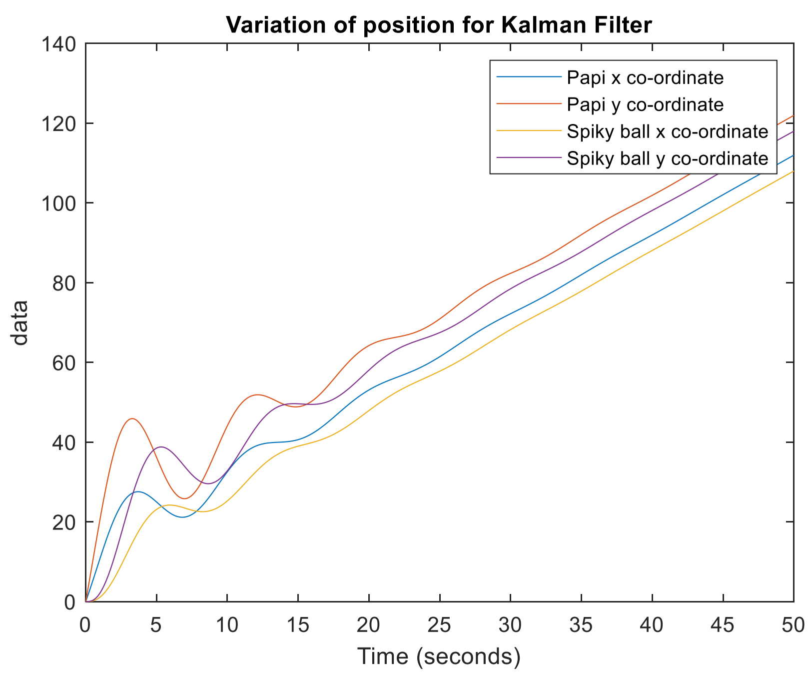
As you can see, I added riccati equation in a system as shown above. Depending upon, Q and R value, simulation will calculate P values and multiplied with C transpose will give L value. Everything else is same as Least square method, only difference is we can control variation in P using Q and R value.

Figures when papi hit enemies and path of x y co-ordinates of papi, spiky balls. For all 4 enemies are as below.

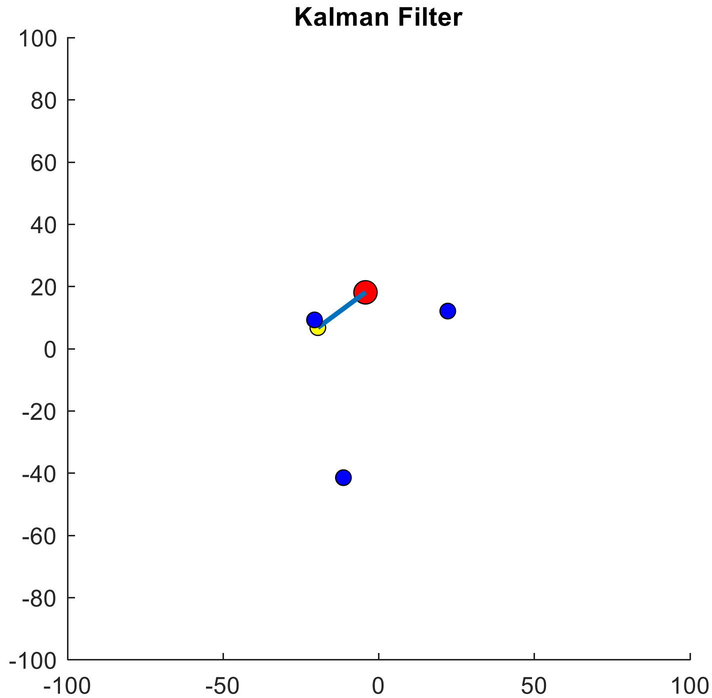
First enemy, when spiky ball hit enemy,



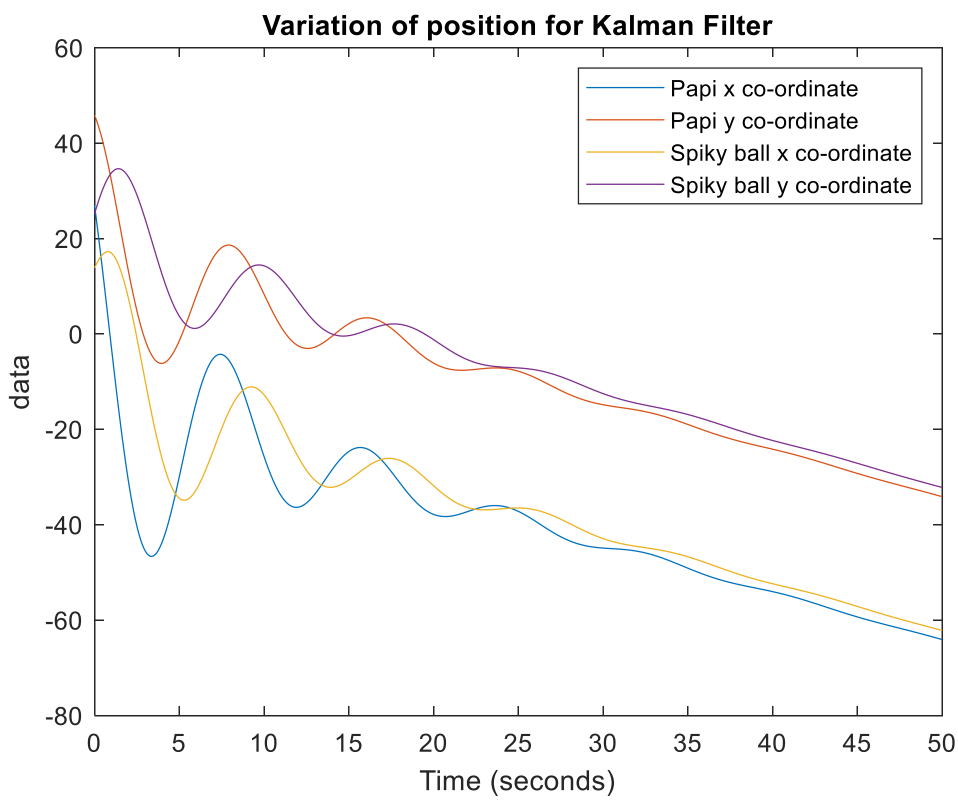
X Y co-ordinates of papi and spiky ball.



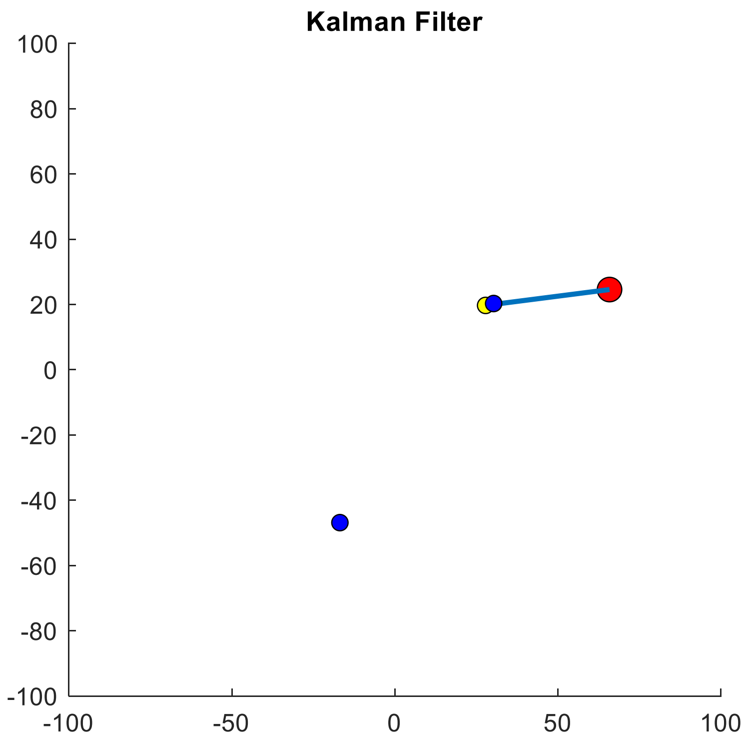
Second enemy, when spiky ball hit enemy,



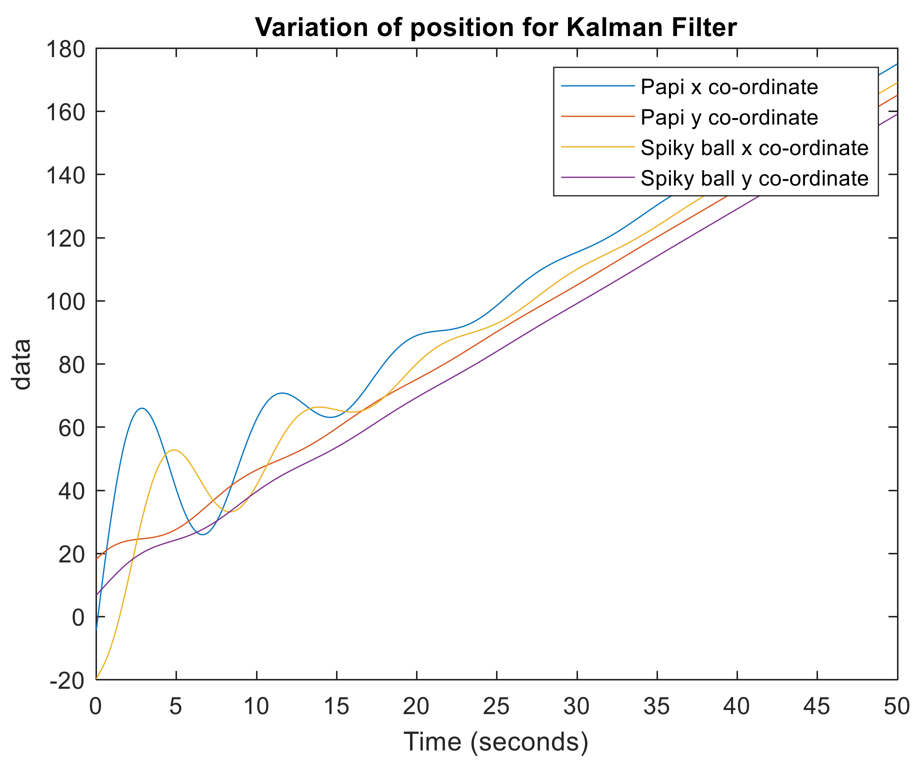
X Y co-ordinates of papi and spiky ball.



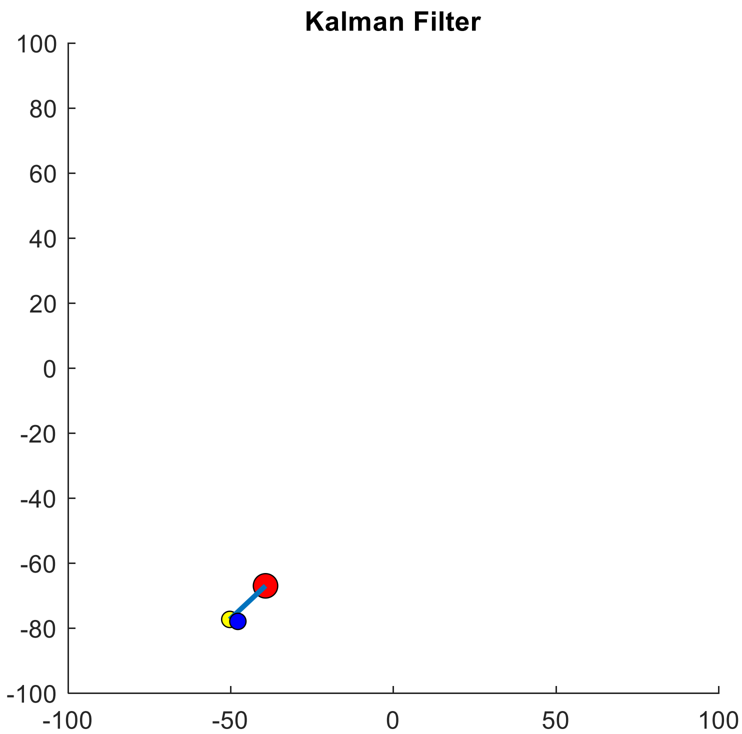
Third enemy, when spiky ball hit enemy,



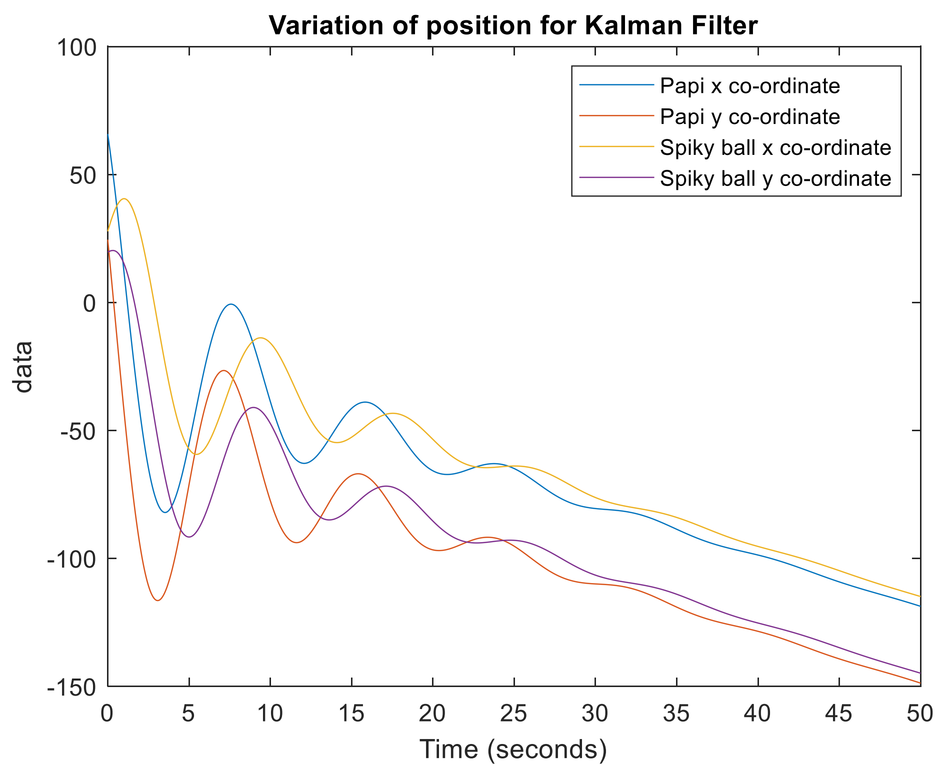
X Y co-ordinates of papi and spiky ball.



Fourth enemy, when spiky ball hit enemy,



X Y co-ordinates of papi and spiky ball.



Discussion of Results

I have done 4 control systems in this project, Pole placement to steady state feedback and observer design, least square estimation, linear quadratic control and Kalman filter. I simulated all the control systems and uploaded videos for each separately. In this game I assumed these conditions, there is no gravity, enemies can't kill papi.

From output lots of each of the control system we can compare and analyze our results. For simple pole placement and observer design, as I have taken less settling time it settles down to enemy trajectory fast but there is huge overshoot in the beginning. Which we don't want, and which is bad for any control system. So, as we can see in video also, that, for spiky ball hit the enemy each time quickly but there is huge overshoot which makes papi go out of the screen for each enemy.

In least square estimate control system, we don't have to choose any parameters, except S. Which in this case, I considered to one. So, this system calculates observer gain based on its riccati equation and give optimum L values. But we don't have any control over system. So, if we want to change overshoot and settling time of a system based on parameters which we want we have to use system like Kalman filter.

In linear quadratic control (LQR) control system, I choose Q and R for a system and calculated gain K using MATLAB. There is no observer in this case, so we can say that this one is better than pole placement and observer design as there is no observer and, it has better performance in overshoot than other system. Also settling time is optimal. But when there are more enemies this system fail to hit all targets in time.

Kalman filter control system, I choose simple Q and R for a system. We can get observer gain values, from riccati equation, which is same as least square estimate. If there is any disturbance in system, it avoid that disturbance and still hit enemy target. From video we can see that, it has better overshoot optimal settling time and more control over performance in our hand.

Overall from all four systems which I studied, I think each system has its own advantages and disadvantages. In all the cases, spiky ball hit enemies' targets and win the game. Pole placement and observer has huge overshoot but settling time is less. Least square estimate has less overshoot but has less control over system as we don't have to choose Q and R. LQR doesn't need an observer but has more settling time, so it takes longer time to hit target. Kalman filter has good overshoot and optimal settling and has certain control over system through Q and R value, plus it can also avoid disturbance. So, if I have to choose between these four control systems then I would choose Kalman filter as it satisfies most of required conditions.

For all the videos please following google drive link: -

https://drive.google.com/drive/folders/1TCxv9n2Ys-vLjYsKLVV5kQYjgHtGACBR?usp=sharing