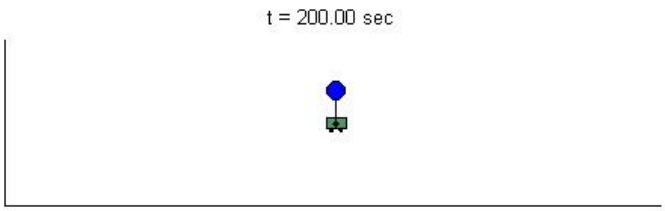
Question 4

Code

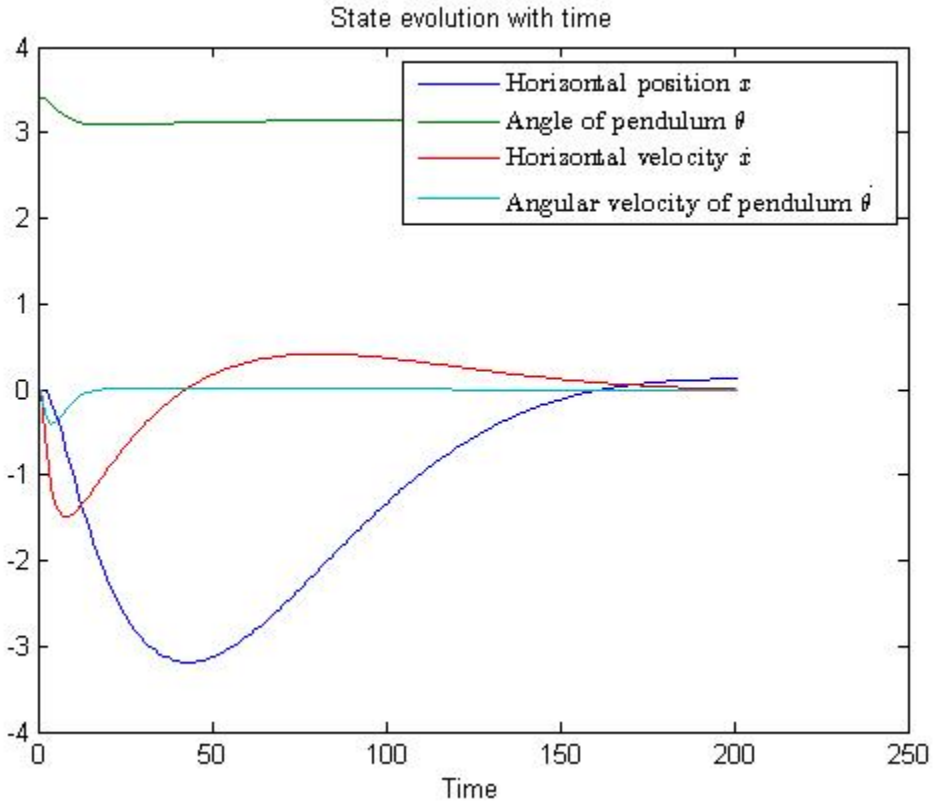
```
function [L, P] = lqr_infinite_horizon_solution(Q, R)

% find the infinite horizon L and P through running LQR back-ups
%   until norm(L_new - L_current, 2) <= 1e-4
dt = 0.1;
mc = 10; mp = 2.; l = 1.; g= 9.81;

% TODO write A,B matrices
a1 = mp*g/mc;
a2 = (mc+mp)*g/(l*mc);
dfds = [0 0 1 0;
    0 0 0 1;
    0 a1 0 0;
    0 a2 0 0];
dfdu = [0; 0; 1/mc; 1/(l*mc)];
A = eye(4) + dt*dfds;
B = dt*dfdu;

% TODO implement Riccati recursion
k = 1;
while k==1 || norm(L_new - L_current, 2) > 1e-4
    if k == 1
        L_current = 0;
        P_current = Q;
    else
        L_current = L_new;
        P_current = P_new;
    end
    L_new = -inv(R + B'*P_current*B)*(B'*P_current*A);
    P_new = Q + L_new'*R*L_new + (A + B*L_new)'*P_current*(A + B*L_new);
    diff = norm(L_new - L_current, 2);
    k = k+1;
end
L = L_new;
P = P_new;
end
```

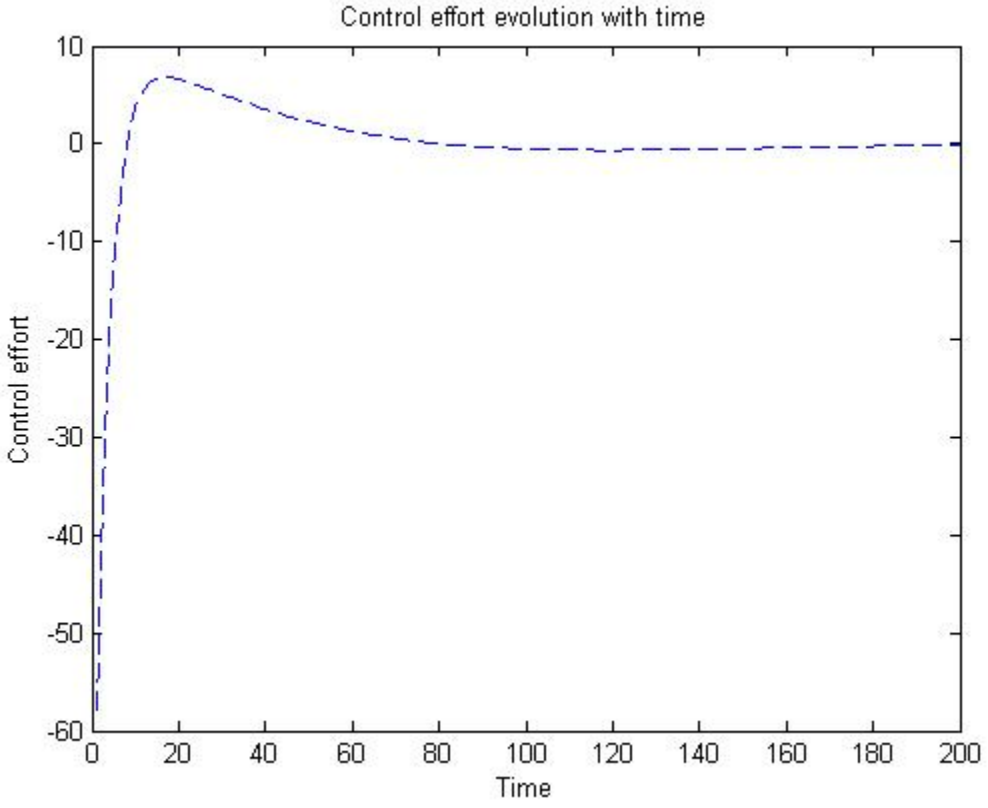
Plots

All the simulations end at this point in the animation

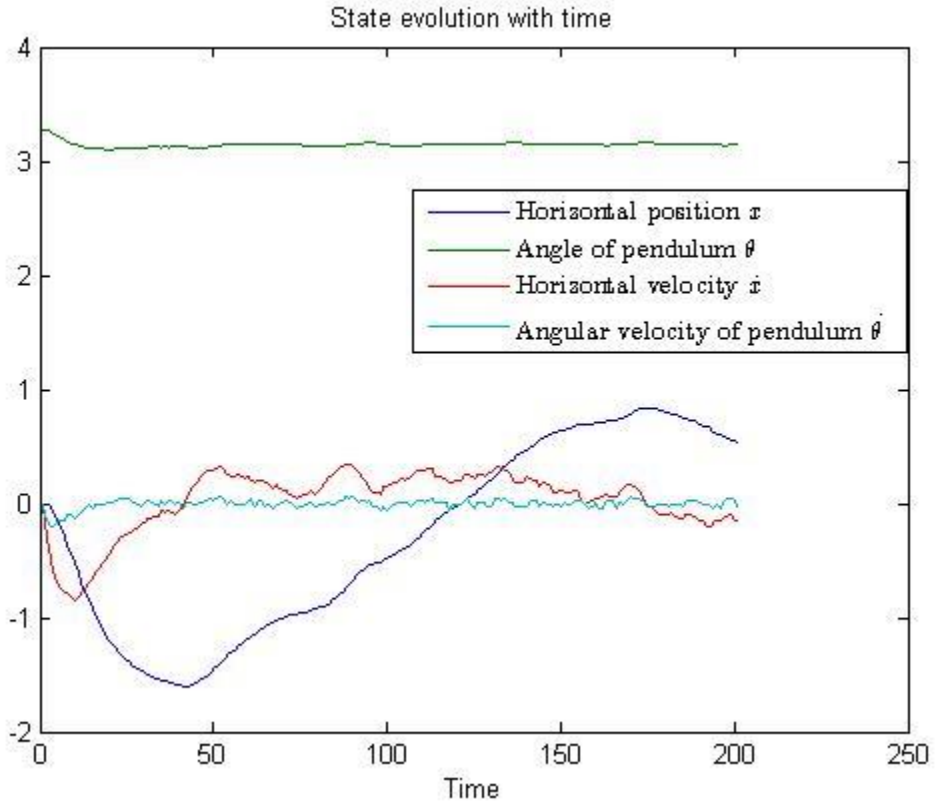


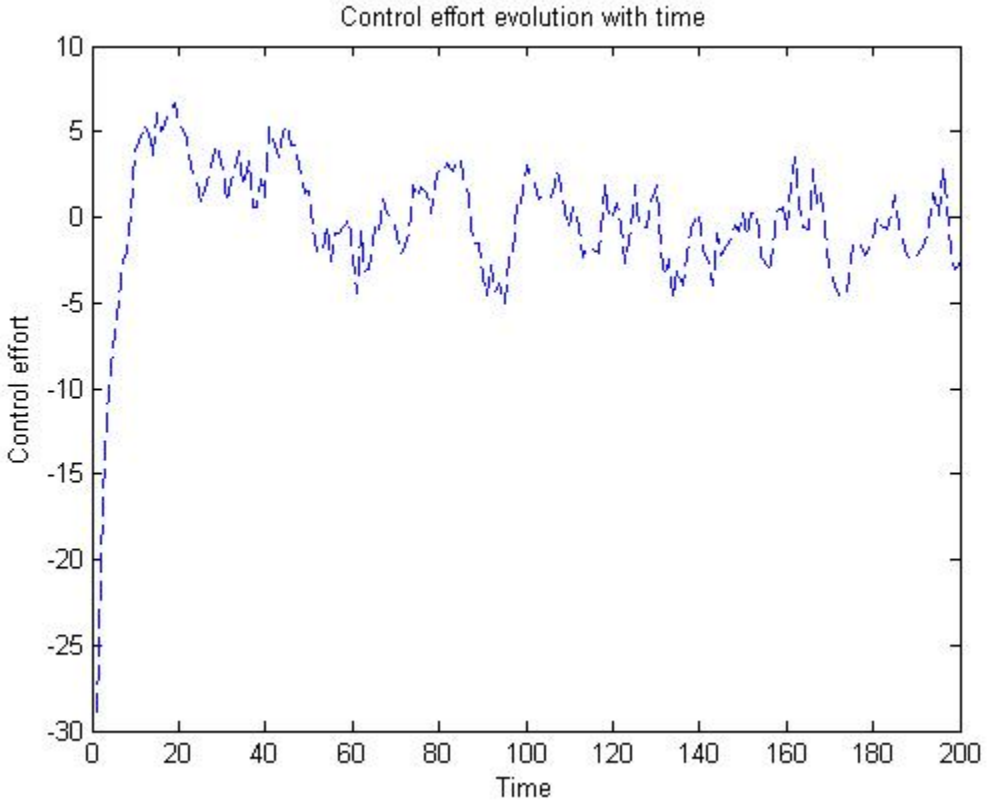
Without Noise





With Noise


State evolution with time
Horizontal position x
Angle of pendulum θ
Horizontal velocity $\dot{x}$
Angular velocity of pendulum $\dot{\theta}$
Time


Control effort evolution with time
Control effort
Time