AE 4803 Robotics and Autonomy
Professor Evangelos Theodorou
Homework 2

Luis Pimentel — lpimentel3@gatech.edu
Jackson Crandell — jackcrandell@gatech.edu

November 2, 2020

Part 1.

See MATLAB code implementation.

To run this simulation run the Inverted_Pendulum/main.m file.

Part 2.

See MATLAB code implementation.

To run this simulation run the CartPole/main.m file.

Part 3.

3.1) See MATLAB code implementation.

To run this simulation run the Inverted_Pendulum/main_robustness.m file.

In this problem we are asked to test the robustness of our MPC-DDP policy on a nomial model of the system used in the DDP feedback optimization and a real model used to propagate forward the dynamics.

To test robustness we create the real model by perturbing the nominal model by different levels of uncertainty denoted by σ. The parameters of the inverted pendulum are perturbed as follows.

$$g_{real} = abs(g_{nominal} + (0.5)^{\sigma} g_{nominal}(2rand - 1)\sigma^2)$$

$$m_{real} = abs(m_{nominal} + m_{nominal}(2rand - 1)\sigma)$$

$$b_{real} = abs(b_{nominal} + b_{nominal}(2rand - 1)\sigma)$$

$$l_{real} = abs(l_{nominal} + l_{nominal}(2rand - 1)\sigma)$$

$$I_{real} = abs(I_{nominal} + I_{nominal}(2rand - 1)\sigma)$$

In this update rand is a random number between 0 and 1. As the inverted pendulum can tolerate high levels of σ the parameter gravity is scaled down much more as to model Earth's gravity closer. The absolute value is taken as these parameters cannot be negative.

In our test we increase σ, compute a new real model with this uncertainty level, and rerun MPC-DDP until it cannot converge anymore.

This process is detailed through the following algorithm:

Algorithm 1: MPC-DDP Robustness Test Against Perturbations in Nominal Model

Initialize: $\boldsymbol{F_{nominal}}, \boldsymbol{F_{real}}$: nominal and real dynamics model;
Initialize: σ: $= 0$ uncertainty level, δ_σ;
while do
  $\boldsymbol{F_{real}}$ = update_real_model($\sigma, \boldsymbol{F_{nominal}}$) ;
  while do
    $\boldsymbol{u}$ = fnDDP($\boldsymbol{F_{nominal}}$,$\boldsymbol{x_{now}}$);
    $\boldsymbol{x}_{next}$ = fnDynamics($\boldsymbol{F_{real}}, \boldsymbol{x_{now}}, \boldsymbol{u}$);
    Repeat until task completion. Break if not converged and greater than i_{max} iterations
  end
  $\sigma = \sigma + \delta_\sigma$;
  Break if last MPC-DDP not converged and greater than i_{max} iterations
end

The following plot shows the results of MPC-DDP after this robustness test is run for several levels of uncertainty:

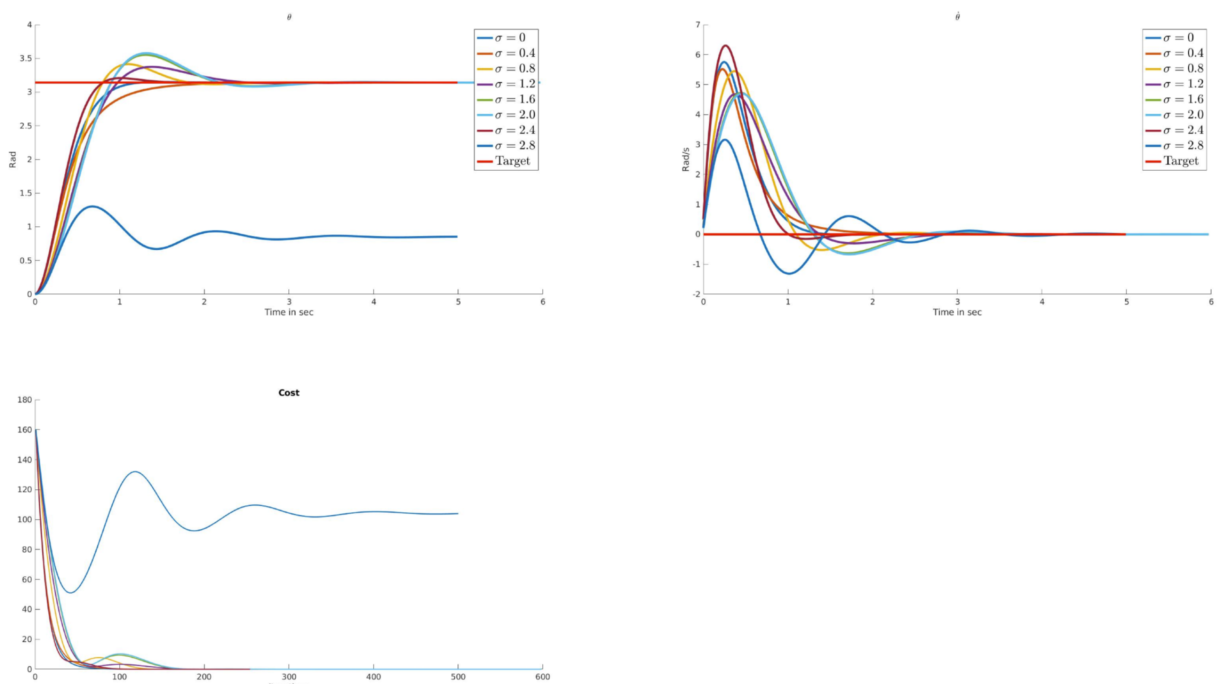

Figure 1: Robustness Test results for Inverted Pendulum.

The following plot shows the results of the perturbation in parameters of the model used to create the real model, for several levels of uncertainty:

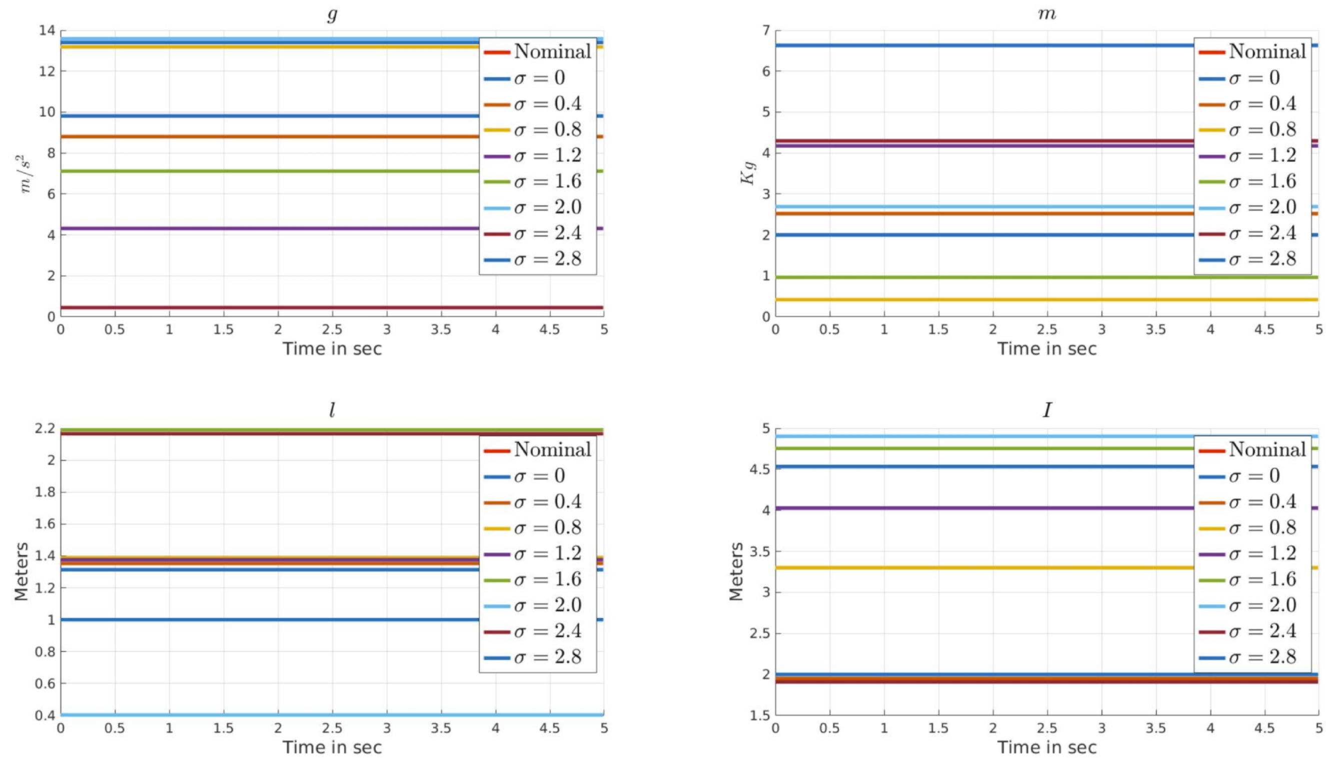

Figure 2: Robustness Test results for Inverted Pendulum.

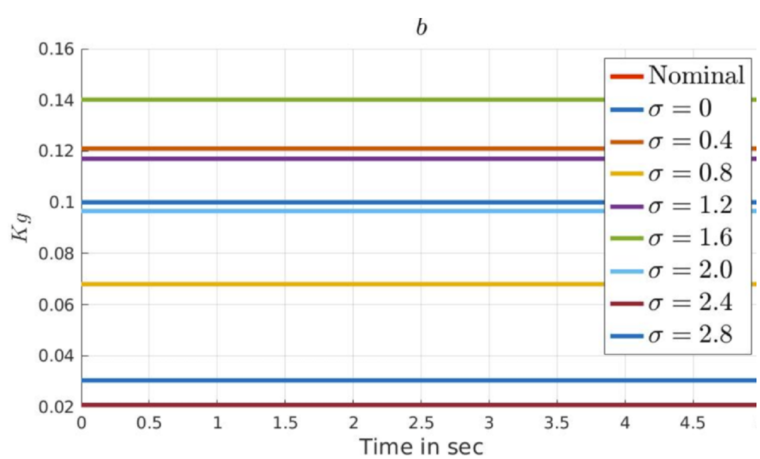

Figure 3: Robustness Test results for Inverted Pendulum.

3.2) See MATLAB code implementation.

To run this simulation run the CartPole/main_robustness.m file.

The robustness test described above was executed for the Cart Pole problem. For this we use the following real model update rule.

$$g_{real} = abs(g_{nominal} + (0.5)g_{nominal}(2rand - 1)\sigma^2)$$

$$m\_c_{real} = abs(m\_p_{nominal} + m\_c_{nominal}(2rand - 1)\sigma)$$

$$m\_p_{real} = abs(m\_c_{nominal} + m\_p_{nominal}(2rand - 1)\sigma)$$

$$l_{real} = abs(l_{nominal} + l_{nominal}(2rand - 1)\sigma)$$

The following are the results.

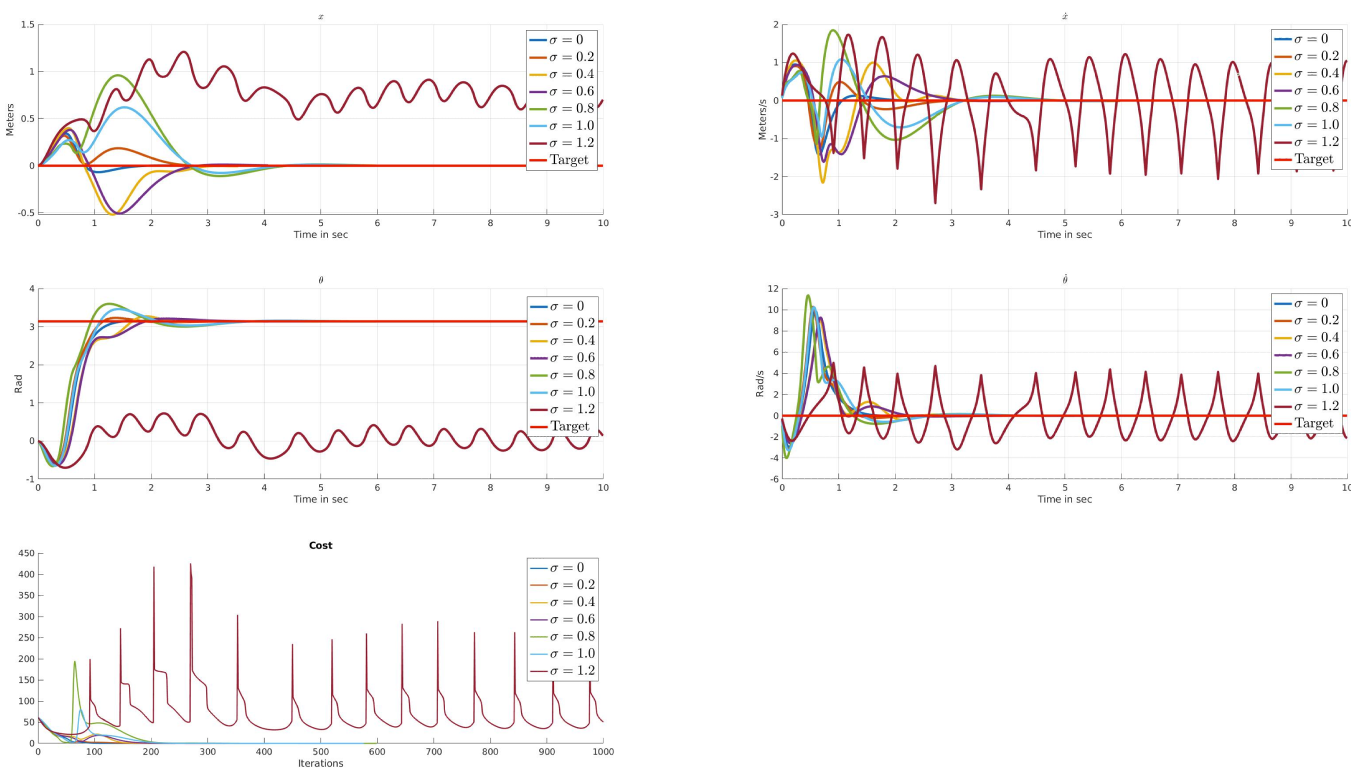

Figure 4: Robustness Test results for Cart Pole.

The following plot shows the results of the perturbation in parameters of the model used to create the real model, for several levels of uncertainty:

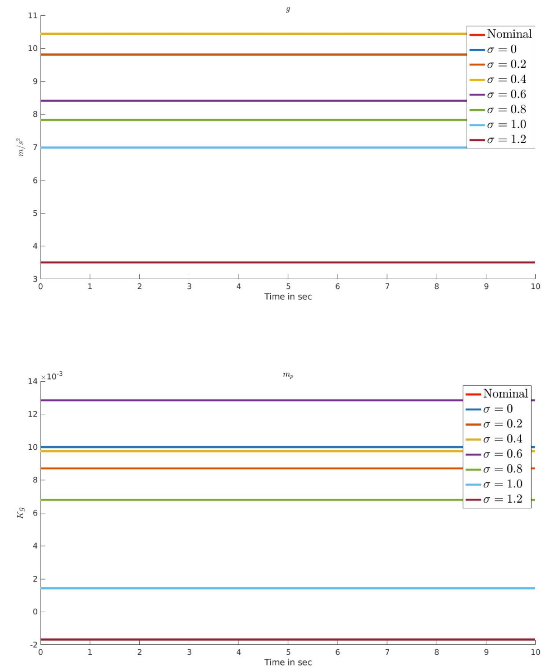

Figure 5: Robustness Test results for Cart Pole.

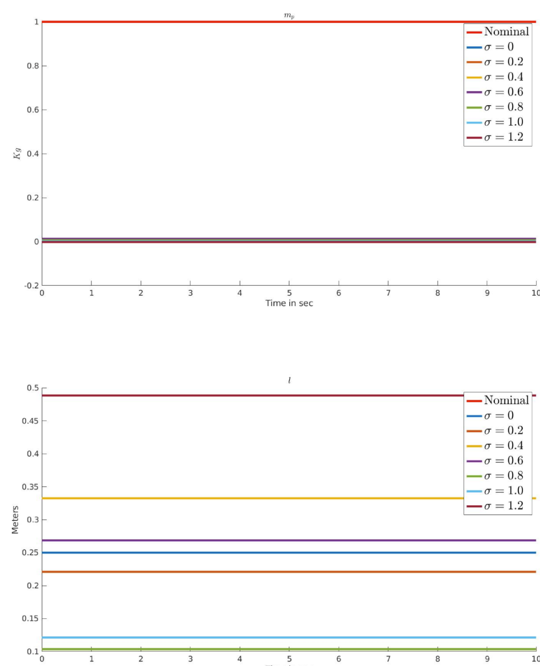

Figure 6: Robustness Test results for Cart Pole.