Project Report: Model Predictive Control on a Cart-Pole model.

Aditya Mohana Sivaraj

N16408439

ams1803

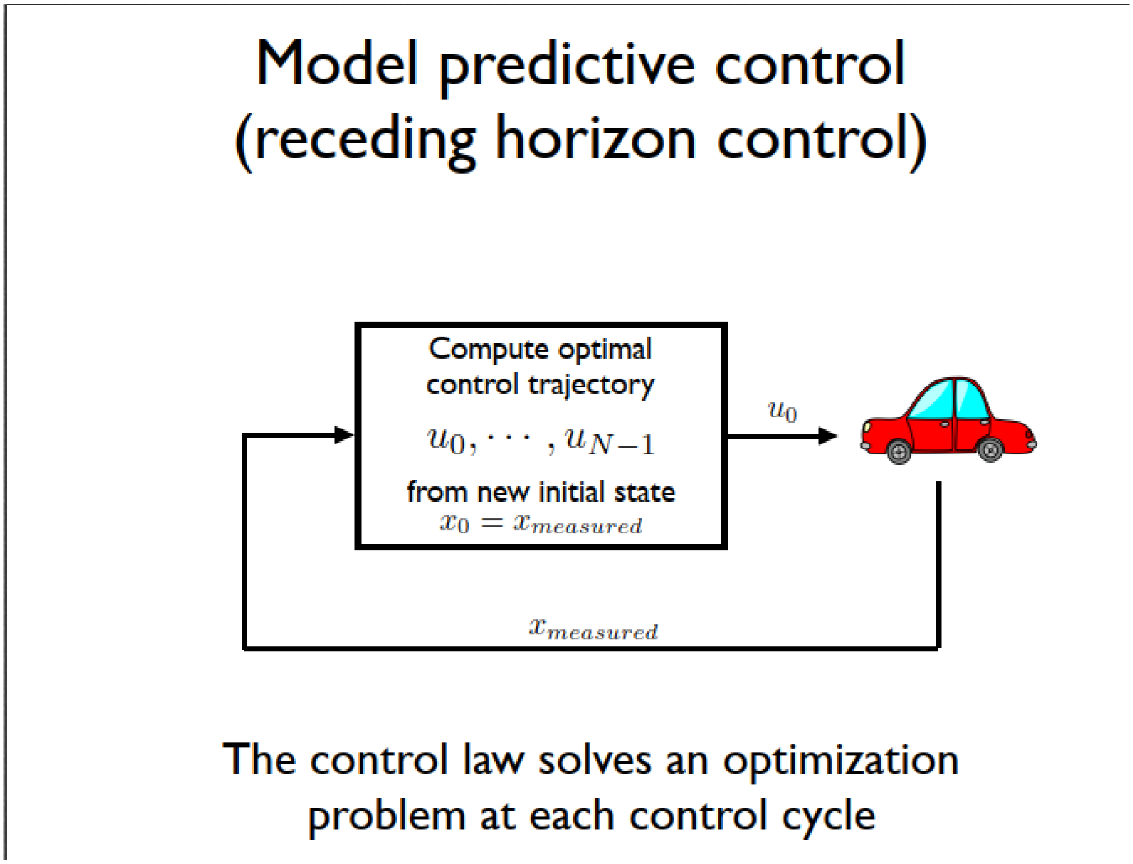
Goal:

The goal of this project is to develop a model predictive control algorithm that will help control a cart pole setup (A cart with an inverted pendulum on top of it).

CVXPY:

In this project a different library called CVXPY which is different from the standard CVXOPT library used in class. In cvxpy they use cvxopt libraries to solve the problems. In cvxopt you have to write your problem in a more standard way for the type of solver you want to use, whereas cvxpy is supposed to adapt your problem based on the structure you use for your problem (they are supposed to select the type of cvxopt solver depending on your problem and pass the variables in an standard cvxopt way). At the end of the day CVXPY is a wrapper that tries to make things easier.

Model Predictive Control:



Model predictive control
(receding horizon control)

The control law solves an optimization problem at each control cycle

Model Predictive Control is a form of suboptimal control algorithm that combines elements of several ideas namely: certainty equivalent control, multistage lookahead, and rollout algorithms.

Its main advantages include that it betters many of the problems faced by the normal Linear Quadratic Model. It can also look after non-linear models well. The short hand version of the algorithm is this: at each and every discrete point defined in space, the model calculates the best possible trajectory or rather solves an optimization problem with respect to that point as if that is the starting point.

The Code:

Initially the global values are set. This includes the length of the pendulum bar, the mass of the cart, mass of the pendulum bob (the pendulum bar is considered massless for the case of simplicity), acceleration, the components of optimization program -> Q and R, number of states, number of inputs, horizon length and finally the time differential.

The next function is *make_equation* which takes an input of the state variable and the input variable. It gives out an output of the state variable which is modified as the standard system state equation.

The next function is the crux of the program which implements the model predictive control algorithm on the given initial state. In this function we declare the state and input variable as cvxpy standard variable. Then we build the cost and the constraint function until horizon length. Then we declare the cvxpy problem and optimize the problem optimality is reached. Finally, we return the optimal parameters.

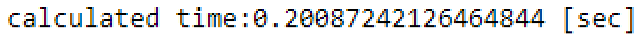
The next function is *model* which returns the coefficient matrix which is built according to the model of the system we plan it to be.

The next function is *cart_pole_frame* which takes in the current position and the angle value from the state matrix, and this returns the current frame as a plot. Then the position of the different objects including the cart, the pendulum bob, telescopic leg, the right and the left wheels is defined. This also includes the angle of the pendulum setup. Then the defined objects are plotted with different colors. I tried to animate the setup, even using the animation function from the cart-pole homework but since the optimization function is done for every step it becomes too complicated to create.

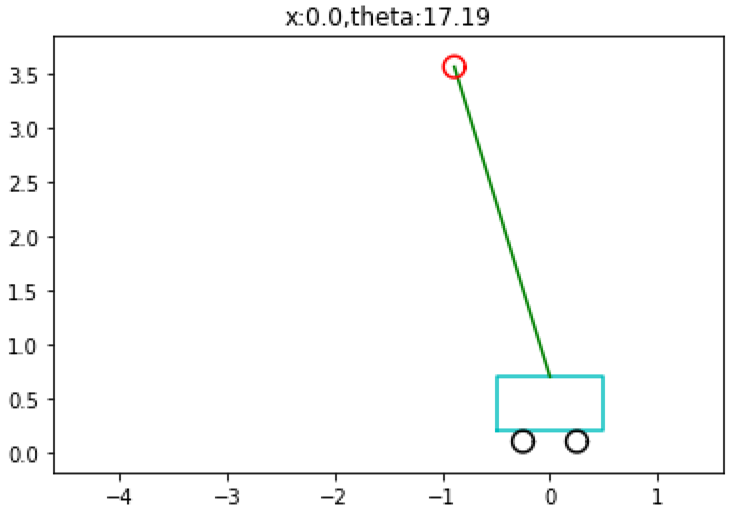
Next cell contains the main parameters which is used to run the program like the initial state and the limit between which the program runs.

Results:

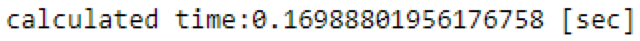
Initial state:



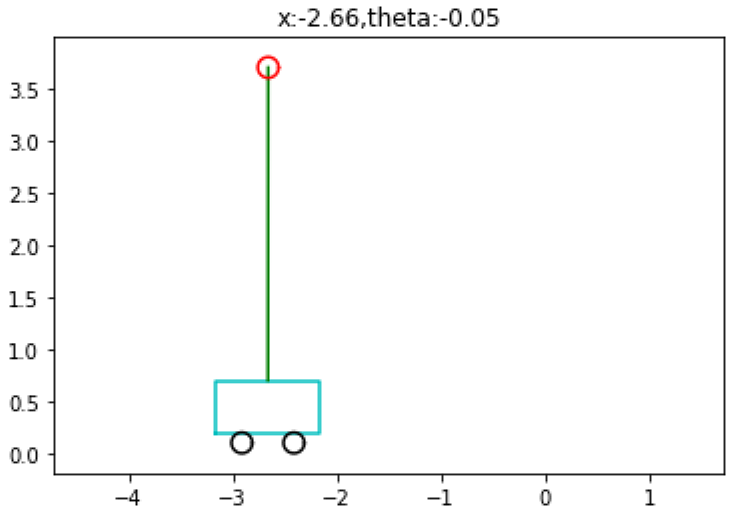
```
calculated time:0.20087242126464844 [sec]
```



Final state:



```
calculated time:0.16988801956176758 [sec]
```



Result: Hence the given cart pole system is optimized to stand upright using the model predictive control algorithm.