ECEN415 Advanced Control Systems Engineering
Assignment Four - 2021

Due 29th of October

Submit pdf documents describing your solutions via the online submission system. Hand written solutions are fine, but these must be scanned and converted to pdf.

Solutions should use standard mathematical notation (not pseudo-matlab), explain your reasoning solutions and include computer generated figures where appropriate. If you do use Matlab routines to derive your answers then you should explain the critical Matlab calls, including input arguments and the pertinent outputs. Your written submitted solutions should not include code unless it is explicitly requested. Note that I will not read your code to understand what you are trying to achieve. Your written answers should contain your complete description of your approach and solutions.

Unless otherwise noted, you can use Matlab (or other suitable tool) to complete these problems. **For this assignment, in addition to your explanatory text you should upload your code as a separate .m file.**

---

1 Task

Recall the rocket launch problem that we saw in assignment two. For this assignment your task is to design a feedback controller that shapes the trajectory of the rocket during its takeoff phase. You can use any control approach that you like, but must thoroughly explain your approach and demonstrate its effectiveness through simulation results.

The goal of your controller is to deliver the rocket to a particular state space configuration after 15 second of flight time. At this time the rocket should

1. Be at an altitude (x_1) of at least 1500 m (higher is better), and travelling upwards (x_2) at as high a speed as possible.

2. Be between 50 m and 100 m downrange (x_3) of the takeoff point with a downrange velocity (x_4) of no more than 20 m/s.

3. Have a pitch (x_5) of 10°and a pitch rate (x_6) of 0 °/s.

4. Have no remaining fuel (x_7).

This final state is included in the template code as `x_f`.

2 Instructions

Template code that runs the simulation of the rocket launch can be found on the course web page. It is substantially the same as that used in assignment 2, with some minor structural changes to make things a little clearer.

In terms of coding, you need only modify the `controller_command` function at the bottom of the code. This function takes the current time and the rocket's state as input, and must return u_1 and u_2, which are the inputs that determine the upwards thrust and pitch rate respectively. You can make other changes to the code, but only your controller function will be used during marking.

2.1 global and persistent variables

You may not have previously encountered global variables in Matlab. They can be useful for parameters of a system that a function might need to know. In this case we have the physical parameters of the rocket, and its initial state declared as global, so you can use them within your controller function. The example controller shows how to use the variables `eta` and `x_0`, but you can add others if you find them useful. (Only variables defined as global in line 14 can be used in this way.)

A similar mechanism can be used for variables that you want to reuse in subsequent calls to your function, but do not appear outside. For example, if you wanted to integrate something, then you might want to have an integrator function to remember the value between function calls. In this case you should look at Matlab's `persistent` keyword.

2.2 Control Hints

Your goal is to deliver the rocket to a particular state that is not precisely defined. That is there is no exact value given for the final altitude and vertical velocity. You will probably find it useful to define a particular final state for use in your design, and then refine things by changing that final state to a more challenging one.

The specifications suggest that both x_1 and x_2 should be made as large as possible. If trading off between these two, then choose a combination that will result in the rocket having the highest possible apogee. You may wish to increase the simulation running time in line 101 to assess this.

If you decide to use a feedback controller, then you will likely find it useful to act on the difference between the state and the desired state. That is, you might like to use a control law of form $u_i = -\boldsymbol{K}(\boldsymbol{x} - \boldsymbol{x}_{\text{desired}})$, where $\boldsymbol{x}_{\text{desired}}$ is the state you would like to have. This is a slight extension of some of the regulator work that we have done in the course.

If the bottom right figure (the fuel graph) turns red during your simulation, then this means you have run out of fuel. This may or may not be what you want.