5ELEN018W - Tutorial 4 Exercises

1 PID Control for the Robot Arm Surgeon

Implement in Simulink the PID controller for the Robot Arm Surgeon we have seen in the lecture slides.

Experiment with different values for the parameters of the PID controller and observe the response of the system using a Scope block.

2 Car Cruise Control

Car cruise control can be found in most of the modern cars. The cruise control system keeps the car at a constant speed despite any external disturbances, such as the road surface and incline.

The car's mass m is controlled by a force u. To simplify the situation, we assume that the force u that our controller applies is not affected by other parameters such as tires, etc.

The equation of the system is described by the following:

$$m\dot{v} + bv = u \tag{1}$$

where v is the speed of the car, u is the control action and b is the damping coefficient due to friction.

For the purposes of this exercise, the following values should be used for the system: $m = 1000, b = 50, u = 500$.

2.1 Modelling of Cruise Control

Implement a Simulink modelling of the above system. For the purposes of the control force u, use a *Step* block (`Simulink->Sources`) that has a value equal to 500 from time $t = 0$.

Use a simulation time of 120 secs (set *Stop time* to 120 in the main Simulink tab).

2.2 Control of the System

1. Add a PID controller to the system, following the guidelines we have covered in the last lecture.

   The same reference signal as in the previous section should be used (step with value equal to 500) as the desired response of the system.

   Use a PI controller with some arbitrary K_p and K_i values.

   Is the response of the cruise control system satisfactory when considering the desired speed response?

2. Open the PID block by double clicking on it and click on the *Tune* button. Choose different values for *Response Time* and *Transient Behavior* so as to try to match the desired response for the speed of the system. Click on the *Update Block* button to save your chosen values to the PID controller and open a Scope connected to the speed signal of your implemented system to check the response.

3. Set the values of the PID controller to $K_p = 800, K_i = 40$. Check the desired speed response of the system by running it and opening the Scope connected to the speed signal. Is that a better response from what you have achieved before? Can you do better than that, by re-opening the PID controller tuner and modify its parameters using the sliders?

3 Transfer Functions

1. Using a manual calculation on paper prove that equation (8) in the lecture slides is correct for the Feedback Controller Transfer Function shown in the Figure in the same slide.

2. Using Python's symbolic computation (Sympy) prove that equation (8) in the lecture slides is correct for the Feedback Controller Transfer Function shown in the Figure in the same slide.

   Hint: Use the solve() function in Sympy. Study the relevant documentation at:

   `https://docs.sympy.org/latest/guides/solving/solve-equation-algebraically.html`