ME477, Lab 7

Ali Jones

3/4/2022

Description:

This program combines elements from previous labs to control the velocity of a motor with a PI controller. The `main()` function initializes communication channels and the interrupt thread, and creates a table to be printed on the LCD that allows the user to communicate with the motor in real time. In this table, the user can set values including the desired steady state velocity, the BTI length, and the proportional and integral gain. The interrupt thread finds the motor controller by calling `cascade()`, a function that uses a biquad to calculate a difference equation. The IRQ also calls `vel()` to communicate with the encoder and find the real-time motor velocity.

```
main
|_ Irq_RegisterDiIrq()      - reserves interrupt, configures DI and IRQ
|_ pthread_create()         - creates new thread to service interrupt
|_ Irq_UnregisterDiIrq()    - unregisters interrupt
|_ wait()                   - waits for 5ms
|_ctable2()                 - creates user-interface table
|
Timer_Irq_Thread            - function in response to interrupt
|_ printf_lcd()             - prints to LCD screen
|_ Irq_Wait()               - waits for IRQ number or ready signal
|_ Aio_Read()               - reads an input channel
|_ Aio_Write()              - writes to an output channel
|_ cascade()                - calculates the controller
|_ vel()                    - finds motor velocity in BDI/BTI
|
cascade                     - calculates the controller
|
vel                         - finds motor velocity in BDI/BTI
|_ Encoder_Counter()        - communicates with encoder
```

Testing:

1. Run the program
    a. The LCD screen will display a table of values. Use the up and down keys to scroll through the table
    b. Set a desired motor steady-state speed
    c. The motor will start up, and “V_act” will eventually settle to almost exactly “V_ref”
2. Apply a disturbance torque
    a. “V_act” might change at first, but will eventually return to “V_ref”
    b. The output voltage, “VDA_out”, will increase
3. Set “V_ref” to zero
    a. The motor will quickly return to zero, and will resist external inputs
4. Set “V_ref” to a negative number
    a. The motor will spin in the opposite direction

Results:

The program behaved as intended. When starting the motor from zero, it ramped up in speed and eventually reached the desired steady-state value. The tachometer showed that the true/measured motor speed almost exactly matched the reported motor speed. With the motor in steady-state, I gently applied a steady load torque to the motor. The actual speed remained fairly constant (had a little oscillation) but the control voltage drastically increased. I could saturate the control voltage by holding the motor steady, preventing movement. This shows how the controller responded to a disturbance torque to keep the output velocity constant. By sending more voltage, it could counteract the load until it saturated.

In addition to analyzing the steady-state behavior, we can adjust the control parameters *Ki* and *Kp* to see how they affect the response. The default *Kp* value was 0.1 V-s/rad, and I varied it from 0.05 to 0.2. There was more oscillation in the motor response when *Kp* was low, as this reduced the damping ratio. Conversely, I varied *Ki* from 1 to 10 and saw more oscillation when it was high. A large value of *Ki* means the time constant and the damping ratio are low, which causes a steeper initial slope, more overshoot, and more oscillation.

Finally, with the base parameters ($Kp = 0.1$, $Ki = 2.0$), I recorded a step starting at -200 rpm and ending at 200 rpm. This velocity response can be seen below, compared with the analytical transfer function response.

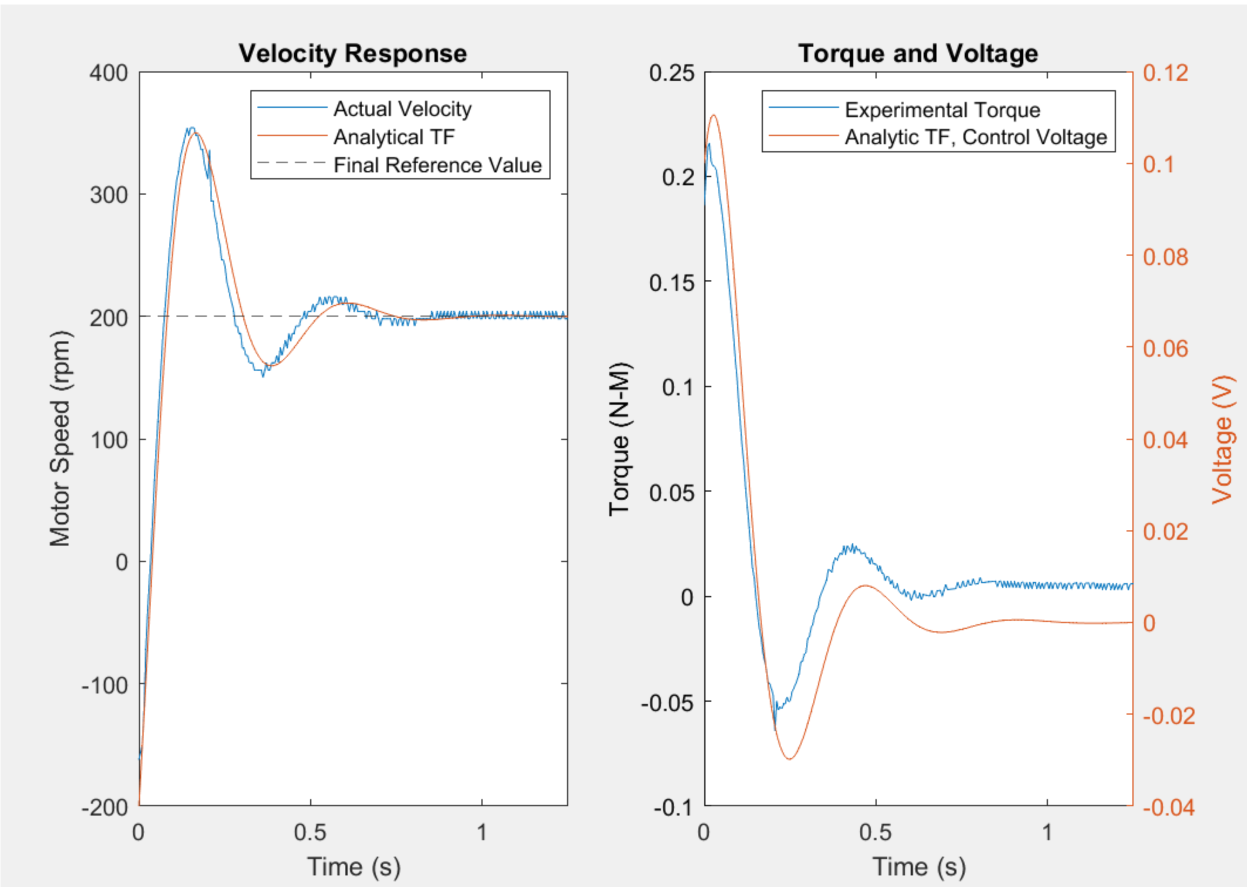

Figure 1. Experimental vs Analytical Responses

The motor velocity behavior followed the analytical transfer function prediction very closely, and the torque displayed the same shape as the control voltage. The model is accurate under our operating conditions. This lab could be broadened by including other operating conditions to see where the model fails, such as variable damping or the addition of a restoring force.

Figure 2. In-lab, showing tachometer measurement, reference velocity, and actual velocity