Prelab	Participation	Lab
√+ √ √− 0	√+ √ √− 0	√+ √ √− 0

Name: ______________________

7 Lab: Motor control for orientation and angular speed

Control systems help satellites to track distant stars, airplanes to follow a desired trajectory, cars to travel at a designated speed, disk-drives to spin at desired angular speeds, and humans to walk, hear, and regulate body temperature.

The purpose of control system design is to determine an appropriate input to an ***actuator*** (e.g., voltage to a ***motor***) to obtain a desired (or nominal) output (e.g., motor speed).

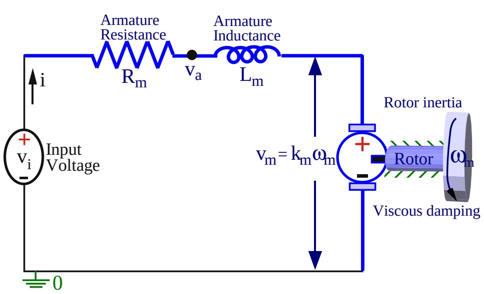
This lab uses system identification techniques to determine a motor's relevant physical parameters. You will implement a control law for motor orientation θ and angular speed ω. You will look at a motor's step response and its transient behavior for various inputs.

7.1 PreLab: Working Model

1. Download the following Working Model (.wm2d) simulations.
   Get these files from: www.MotionGenesis.com ⇒ Textbooks ⇒ Resources

   MotorControlWithOnOffAndDeadBand.wm2d **MotorControlWithKpKi.wm2d**
2. Run the Working Model simulations.
   Record results on the Working Model PreLab (.pdf on www.MotionGenesis.com).

7.2 Experimental



The system to be controlled is a motor whose rotor (shaft) is attached to a rod. Using $T = I\,\alpha$ and circuit analysis, the equation relating v_i to motor orientation is as follows (this should be familiar from the homework):

$$\text{From given model:} \quad \frac{L\,I_{\text{eff}}}{k_m}\,\dddot{\tilde{\theta}} \;+\; \frac{L\,b_m + R\,I_{\text{eff}}}{k_m}\,\ddot{\tilde{\theta}} \;+\; \left(\frac{R\,b_m}{k_m} + k_v\right)\dot{\tilde{\theta}} \;=\; \tilde{v}_i$$

where $I_{\text{eff}} \triangleq I_{\text{rod}} + I_m$. Knowing $I_{\text{rod}} \gg I_m$, approximate the value of I_{eff}.

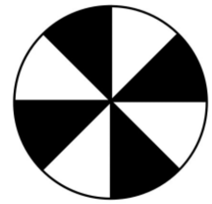
mass of rod $m_{rod} \approx$ ______ kg length of rod $L_{rod} \approx$ ______ m

$$I_{\text{eff}} \approx I_{\text{rod}} = \frac{1}{12}\,m_{rod}\,L_{rod}{}^2 \approx \text{\_\_\_\_\_\_\_\_\_\_\_\_} \text{ kg m}^2$$

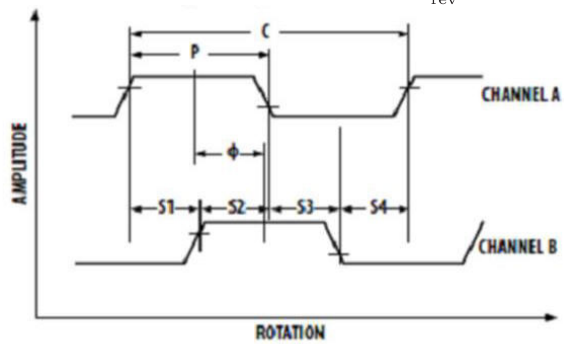
7.3 Equipment

We use several pieces of equipment to measure and control the motor's angular speed, namely, we use an encoder, an Arduino microprocessor, a transceiver, and a computer.[14]

- **Encoder:**
  The optical quadrature **encoder** measures motor rotational speed by detecting alternating light and dark patterns on a disk. For example, the encoder on the right shows 8 transitions (from light to dark or vice-versa). A quadrature encoder can detect both angular speed **and** direction. Our encoder has 1000 transitions (500 black sections and 500 white sections) and counts 1000 $\frac{\text{tics}}{\text{rev}}$.



  Shown to the right is the output signal from the encoder to the Arduino microprocessor.



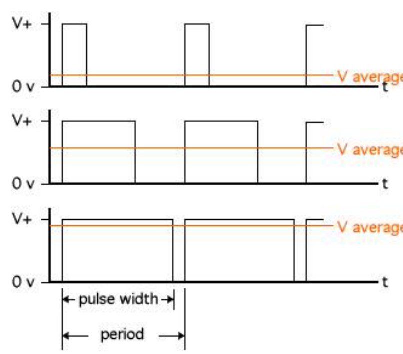
- **Computer:**
  The computer connects to the Arduino microprocessor via a USB cable (Serial Communication). Computer bits (ones and zeros) are transferred between the computer and Arduino microprocessor. You will use the compiled Arduino-specific executable file (Lab7.ino) to communicate between the computer and motor. When you run Lab7.ino, you will be prompted to select proportional or velocity control (choose the appropriate selection for the associated lab question).

- **Arduino UNO microprocessor (the interface between the computer and motor):**
  In this lab, the Arduino will use PWM (Pulse Width Modulation)[15] to vary the average voltage delivered to the motor, which indirectly controlling the motor's speed.

$$\text{PWM} \;\Rightarrow\; \text{Average Voltage} \;\Rightarrow\; \text{Motor current} \;\Rightarrow\; \text{Motor torque} \;\Rightarrow\; \alpha \;\Rightarrow\; \omega$$

  The frequency of this PWM signal is 30 KHZ.
  The Arduino receives the digital signal from the encoder and counts the transitions from "high" (5 Volts) to "low" (0 Volts) of the signal in 2 milliSecond intervals.
  The on-screen data (i.e., x in the following equation) is in units of $\frac{1 \text{ tic}}{2 \text{ ms}}$. The equation converting units displayed on the computer screen to the motor's angular speed (in RPM) is

$$\frac{x \text{ tics}}{2 \text{ ms}} * \frac{1000 \text{ ms}}{1 \text{ second}} * \frac{60 \text{ sec}}{1 \text{ minute}} * \frac{1 \text{ rev}}{1000 \text{ tics}} = \frac{30 \text{ rev}}{1 \text{ minute}}$$



- **Motor driver/Arduino Interface Board:**
  The motor driver circuit receives the PWM signal from the Arduino microprocessor and controls the voltage delivered to the motor via the 12 Volt wall adapter.

---

[14] Most motors do not come attached to a rotary encoder and assembled with a encoder, microprocessor, transceiver, and computer.

[15] A PWM signal is on (designated by a "high-bit" 1) or off (designated by a "low-bit" 0) for different intervals.

7.4 Data collection

1. Power the Arduino by plugging-in (in order):
   (a). 12 Volt adaptor (between the board and wall socket)
   (b). USB cable (between the board and the computer)
2. Navigate to the Lab7 folder and open Lab7.ino
3. Click the magnifying glass button (or type Ctrl+Shift+m) to open the serial monitor
4. On the serial monitor screen, a menu should appear. Select ***position*** control.
5. Enter values for k_p, k_i, and k_d and enter a desired (nominal) value (see below).
6. Enter "r" to stop recording data (when sufficient data has been collected).
7. Plot the data (e.g., using Excel, MATLAB®, or PlotGenesis) and determine ζ, ω_n, etc.
   For the yAxis, convert encoder counts to radians by multiplying by $\frac{2\,\pi}{1000}$
   For the xAxis, convert to seconds by dividing by 500 (i.e., each tick is 2 ms)
   Note: Use the oscilloscope to ensure $Vcc = 12$ volts.
8. Email the data files and/or graphs to yourself and your group members.
9. Ensure the power to the board is off and the setup is neat for the next lab.

7.5 Determination of ζ (damping ratio) and ω_n (natural frequency)

Using an input step response of $\theta_{\text{desired}} = 10$ rad and a **PD** control law of

$$\widetilde{v}_i \;=\; \text{-}k_p\,\widetilde{\theta} \;-\; k_d\,\dot{\widetilde{\theta}} \qquad \text{with } k_p = 1 \text{ and } k_d = 0.01$$

Express ζ in terms of logDecrement and ω_n in terms of logDecrement, ζ, and τ_{period}.
Experimentally determine values for ω_n and ζ (assume $L \approx 0$ from here on).
Note: Whether $L \approx 0$ is a good approximation is to-be-determined in a non-existent later lab. However, this assumption is marginally tested since the control system actually works on the real equipment.

Result:

$$\zeta = \sqrt{\frac{\boxed{}}{\boxed{} + \boxed{}}} \approx \boxed{}\ \text{noUnits} \qquad \omega_n = \frac{\boxed{}}{\boxed{}} \approx \boxed{}\ \frac{\text{rad}}{\text{sec}}$$

7.5.1 Analytical expressions for ζ and ω_n in terms of k_p, k_d, ...

Using the governing ODE and previous **PD** control law, write the equations for ω_n and ζ as functions of k_d and k_p (do this symbolically - without numbers).

Result:

$$\omega_n = \sqrt{\frac{\boxed{}}{\boxed{}}} \qquad \zeta = \frac{\boxed{}}{2\sqrt{\boxed{}}}$$

7.5.2 System identification for motor armature resistance and motor damping

Determine the value for the motor resistance R and motor's damping b_m for the motor with a rod attached. (Note: The last lab showed $k_m = k_v \approx 0.041\,\frac{\text{N m}}{\text{Amp}}$.)

Result:

$$R = \frac{\boxed{}}{\boxed{}} \approx \boxed{}\ \text{Ohms} \qquad b_m = \boxed{} - \frac{\boxed{}}{\boxed{}}(k_v + k_d) \approx \boxed{}\ \text{N m sec}$$

7.6 Angle (θ) control - with PD control

Write expressions for damping ratio ζ and natural frequency ω_n in terms of maximum overshoot M_p and settling time t_{settling}. Approximate ζ and ω_n when $M_p = 0.257$ and $t_{\text{settling}} = 1.75$ sec.

Result:

$$\zeta = \sqrt{\frac{\square}{\square + \square}} \approx \square \qquad \omega_n = \frac{\square}{\square} \approx \square$$

Express k_p and k_d in terms of ζ, ω_n and I_{eff}, b_m, k_m, k_v, R. Approximate numerical values for k_p and k_d corresponding to $M_p = 0.257$ and $t_{\text{settling}} = 1.75$ sec.

Result:

$$k_p = \frac{\square}{\square}\,\omega_n^2 \approx \square \qquad k_d = 2\,\zeta\,\omega_n\,\frac{\square}{k_m} - \left(\frac{\square}{k_m} + k_v\right) \approx \square$$

Try these values for k_p and k_d on the actual system. Graph $\theta(t)$ and show it to your lab TA. Use the graph to estimate the experimental maximum overshoot M_p and settling time t_{settling}.

Result:

Experimental: $M_p \approx \square$ $\qquad t_{\text{settling}} \approx \square$

Calculate the percent error between the desired $M_p = 0.257$ and $t_{\text{settling}} = 1.75$ sec and experimental (actual) values of M_p and t_{settling}. Note: Percent error $= 100\,\frac{\text{(analytical − experimental)}}{\text{experimental}}$.

Result:

Error in $M_p \approx \square$ % $\qquad$ Error in $t_{\text{settling}} \approx \square$ %

Experimentally adjust k_p and k_d in Lab7.ino until $M_p \approx 0.257$ and $t_{\text{settling}} \approx 1.75$ sec.

Result:

$k_{p_{\text{experimental}}} \approx \square$ $\qquad k_{d_{\text{experimental}}} \approx \square$

Experiment with k_p and k_d to get critical damping and the fastest settling time. [Hint: Pick a k_p value and solve for k_d so the system is critically damped. Then keep trying!]

Result:

$t_{\text{settling}} = \square$ $\qquad k_p = \square$ $\qquad k_d = \square$

Why can't you get a faster settling time?

7.7 Angle (θ) control - with PID control

When k_p is small, is there any steady-state error? What do you think causes it?

Write a new **PID** control law (with integral control).

Result:

$$\tilde{v}_i = \square + \square + \square$$

Try various values for k_i on the actual system. What happens? How does integral control eliminate steady-state error? Do you have to be careful with your value for k_i?

7.8 Optional: Angular velocity (ω) PI control

Write a **PI** controller for $\widetilde{v}_i$ so the motor at the constant desired (nominal) rate $\omega_{\text{nom}} = 24\ \frac{\text{rad}}{\text{sec}}$.

Result:

$$\widetilde{v}_i = \boxed{} + \boxed{} \qquad \text{where} \qquad \widetilde{\omega} \triangleq \omega - \omega_{\text{nom}}$$

Determine the governing ODE for $\widetilde{\omega}$ (error in ω) when using this **PI** controller.

Result:

$$\ddot{\widetilde{\omega}} + \frac{\boxed{}}{\boxed{}}\dot{\widetilde{\omega}} + \frac{\boxed{}}{\boxed{}}\widetilde{\omega} = 0$$

Adjust k_p and k_i to give a peak value $M_p = 30\ \frac{\text{rad}}{\text{sec}}$ at $t_{\text{peak}} = 2$ sec for a 24 $\frac{\text{rad}}{\text{sec}}$ step.

Note: The microcontroller's Arduino code rounds ω_{nom} to increments of $\frac{2\,pi}{5} \approx 1.26\frac{rad}{sec}$. This number arises from the combination of running the control software at 500 Hz and having 500 gaps in the encoder.

Result:

$$k_p = \boxed{} \qquad\qquad k_d = \boxed{}$$