Zagazig University
Faculty of Engineering
Computer and Systems Engineering Dept.

Lab No. 5

Digital Position Control of DC Motor Using Arduino

By

Eng. Ahmed Abdelbasit Mohamed

Email: ahmedam@zu.edu.eg

Introduction:

This tutorial introduces the explanation and the practical steps to implement digital PID controller. The main objective it is to apply such controller in the problem of position and/or speed control of DC motors. We will concentrate on the physical meaning and the practical issues rather than the theory and the mathematical concerns. Moreover, we will discuss the sensors and how to handle the measurement noise. So, after this tutorial you will be able to use such control system in many projects especially mobile robots' navigation.

Overview:

In servo systems where the process in needed to follow up some reference value, the closed-loop control is proposed. It consists of three main parts; the process, the controller and the feedback. In our tutorial we work on position control so, these parts are:

- The process: The DC Motor.
- The controller: Here we are working with PID controller implemented on Arduino.
- The feedback: we are using shaft encoder.

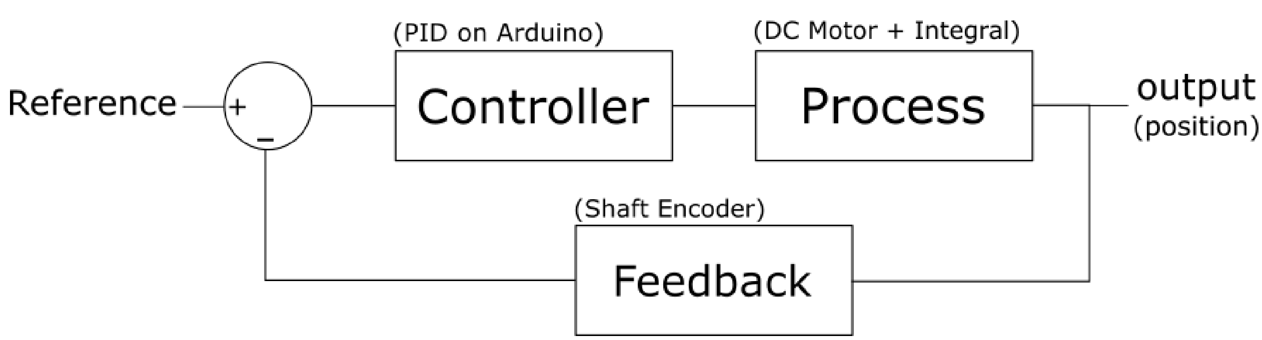


Fig (1)

In the following sections we will discuss each part, identify its main problems and learn how to handle them to reach a stable performance.

List of Hardware:

1. Arduino Nano (you can use any board but you should pay attention for the difference in pin function)
2. Geared DC Motor
3. DC Motor Driver
4. Power supply
5. Magnetic shaft encoder
6. Wires

Arduino Nano

DC Motor Driver 4-channel

DC Motor with Gear box

Shaft Encoder 2PPR

Fig ()

Wiring the circuit:

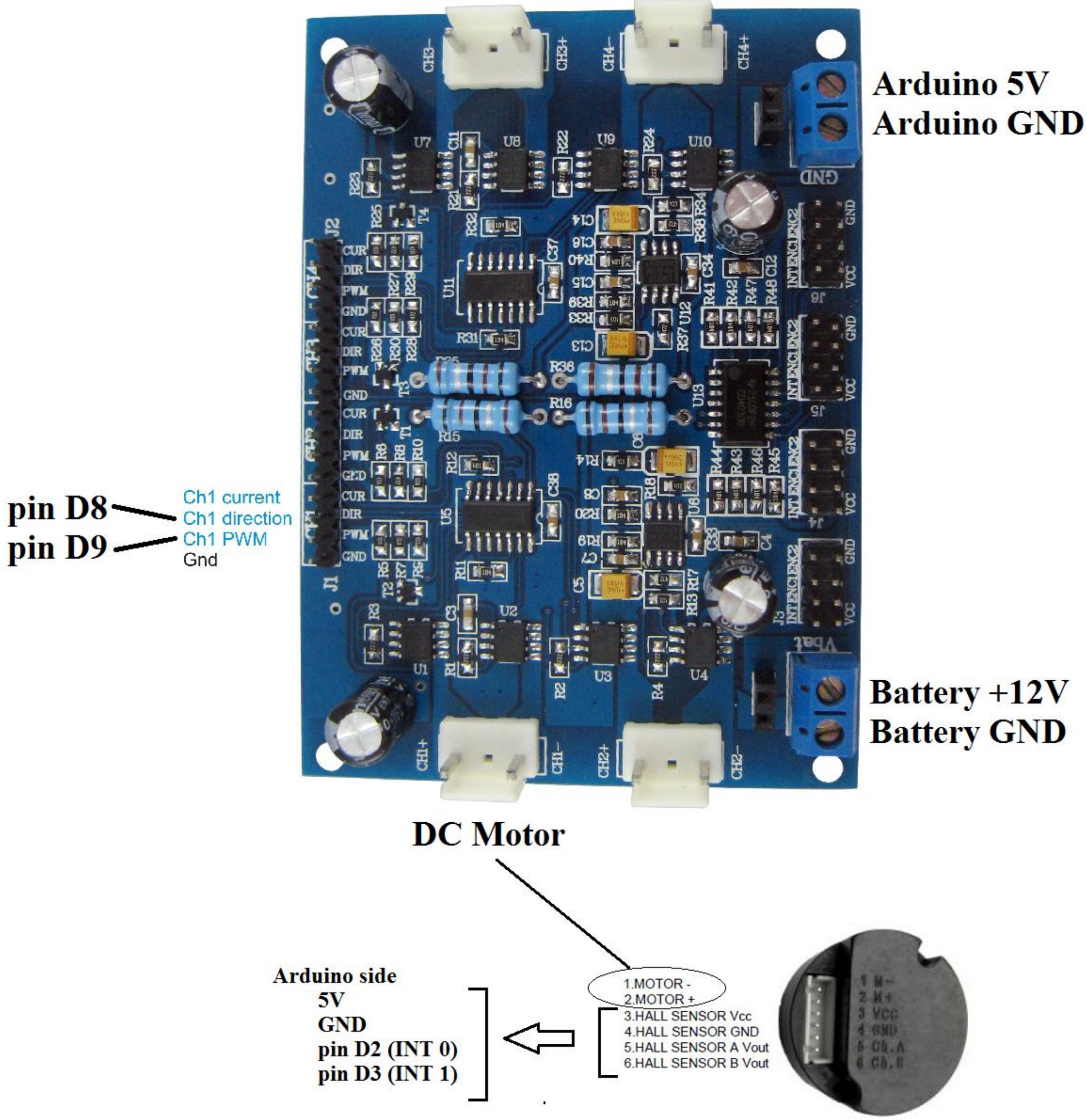


Fig ()

About the work strategy in this tutorial:

In this tutorial we will start with a sustainable system reached by many tries. Then, we will change a single parameter to identify its effect on the overall performance reached. So, we will study the effect of the following parameters:

1. Sample Time
2. Both of the Law Pass Filters' factor
3. The three effectors of PID controller - i.e. Kp, Ki and Kd -.
4. The velocity feedback gain.

Step (1): Hands on the process:

Our objective is to implement a position control system in which the process is the dc motor followed by an integral. For such integral we cannot implement an open loop control system. To know the reason, apply the following steps:

1. Set the control type to SPEED_CONROL 'to visualize the open loop position control'.
2. Set the reference to 30 rpm and notice the system performance 'position curve'.
3. Insert the output curve (position Vs time) here and write down a comment on the result.

Insert your result here

Fig ( )

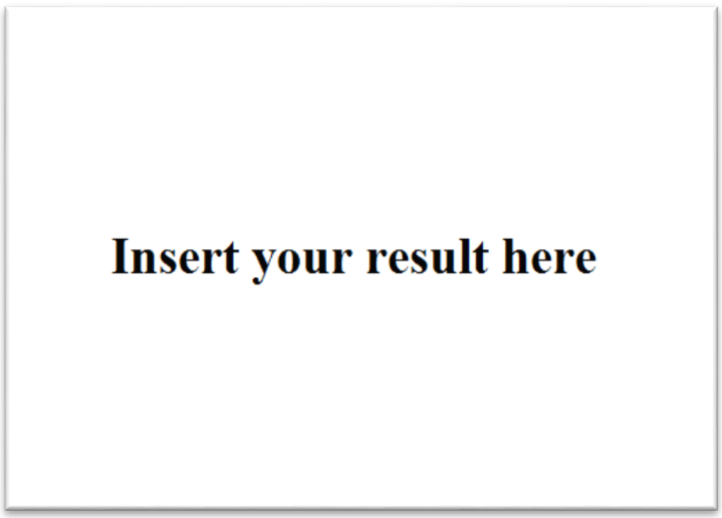
Write your comment here:

Problems concerning the control of DC motor:

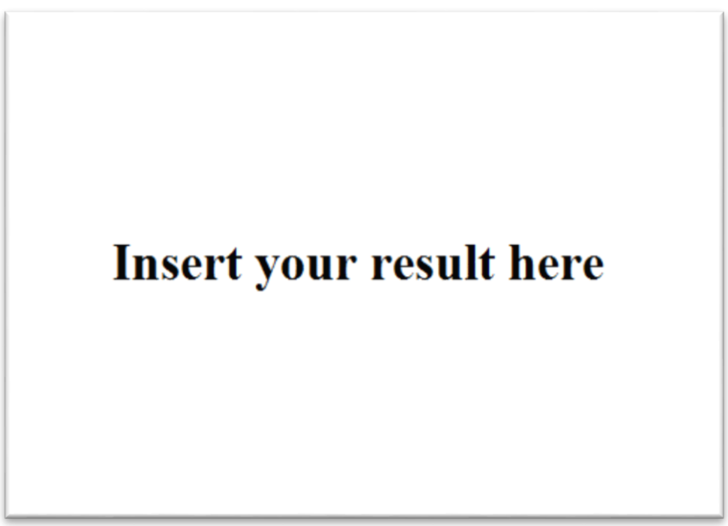
In case we need to control the speed of DC motor using we usually face two main problems concerning its small time constant and the dead zone. In other words, there is a band of small inputs for which the output shaft speed is zero. Outside this band the motor responds fast to the upcoming relatively large input causing a sudden shock in the system response and leading oscillations until steady state reached.

To figure out this problem, apply a ramp input with small slope, try to find the dead band in forward and backward directions and insert the response curve (ramp input versus speed).

Hint: Use ramp_input.ino file to apply this test.



(a)



(b)
Fig ()

Write down you results here:

Forward dead band:
Backward dead band:

Introducing a Low Pass Filter before the DC Motor:

The main operation of the low pass filter is to slow down the system by rejecting the high frequencies from the incoming signal. In our digital system when we deal with discrete numbers, we can figure out this effect by replacing the word 'high frequency' by 'large changes' in the input signal. The equation representing the digital low pass filter can be written as:

$$y_k = (1-a) \times y_{k-1} + a \times u_k$$

The parameter a can be defined as a ratio determining how much the output is affected by a change in the input. The smaller the value of a, the slower the response of the system.

To figure out the effect of such parameter follow up the following steps:

1. Set the control type to speed control.
2. Set the reference speed to 60 rpm.
3. set it to values 1, 0.25, 0.1, 0.05, 0.025, and 0.01 and insert the response curve for each value. The curve. (step Input of 60rpm as a reference versus speed output).

Insert your result here

Insert your result here

Insert your result here

Insert your result here

Insert your result here

Insert your result here

Step (2): Choosing Sample Time:

In digital systems, the sample time is a critical parameter that might affect the system stability. There are two main precautions to consider when determining its value in practical way.

1. It must be less than the least time constant in the system.
2. It must be applicable considering the processing speed of the controller.

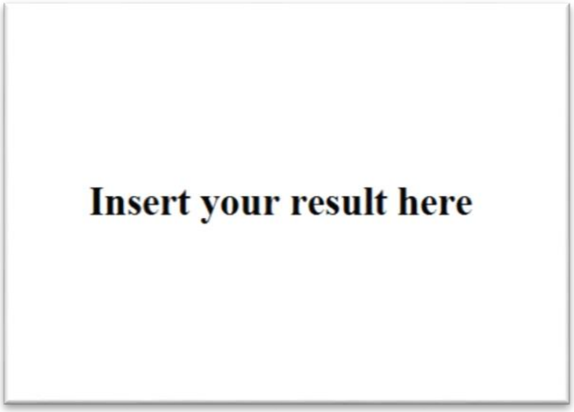
While the sample time gets smaller and smaller, the response of the system approaches that of the continuous system. So, it is a rule of thumb to select the smallest applicable value.

Lab Work:

Run the code with sample time of values **10, 30, 60 and 100** milliseconds and insert the response curve for each value. Write a comment on your results describing the effect of increasing sample time. Also explain how a long sample time may affect the system stability.

Insert your result here

Insert your result here



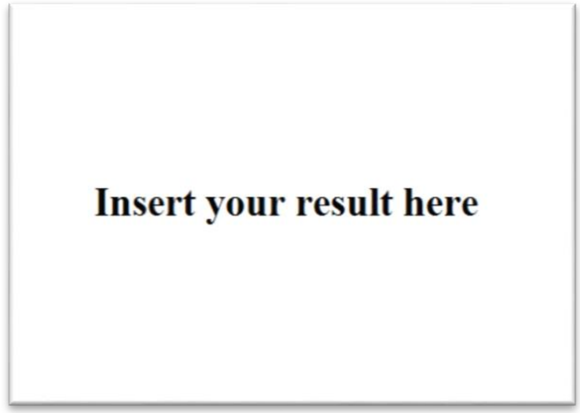


Fig ():

Step (3): Handing Feedback Noise:

In closed loop control systems, the quality of feedback signals directly affects the control efficiency. This is reasonably logical because the accurate feedback leads to a correct control signal to be applied on the system. In this lab the feedback is the shaft rotations measured using a shaft encoder. It consists of some circuitry that generates a number of pulses for each revolution. By counting these pulses and identifying its order we can know how many revolutions the motor have done and in what direction. Here is a figure showing the operation of the digital shaft encoder:

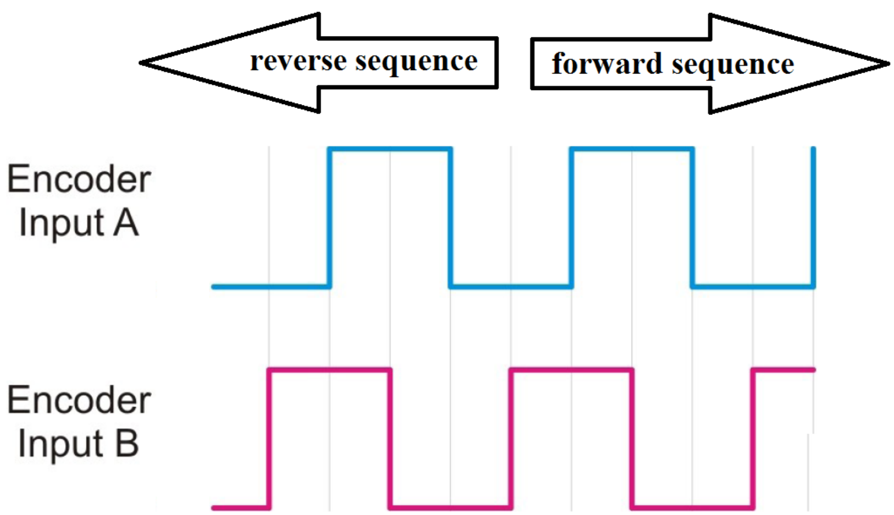


Fig ():

As the figure shows, the determination of movement direction depends on the edge of interrupting pulse – rising or falling – with the level of the second signal – high or low –. A table describing this operation is as follows:

New Input B	New Input A	Old Input B	Old Input A	Result
0	0	0	0	No movement
0	0	0	1	+1
0	0	1	0	-1
0	0	1	1	+2
0	1	0	0	-1
0	1	0	1	No movement
0	1	1	0	-2
0	1	1	1	+1
1	0	0	0	+1
1	0	0	1	-2
1	0	1	0	No movement
1	0	1	1	-1
1	1	0	0	+2
1	1	0	1	-1
1	1	1	0	-1
1	1	1	1	No movement

Handling feedback oscillations

One solution for such problem is to use the mean of a set of consequent readings. In this way, the velocity feedback signal would be more smooth. In another way we can introduce a low pass filter on the encoder signal. But a side effect of using such filter is that the system will get a delayed feedback causing a larger overshot percentage.

1. Set the control type to Speed Control.
2. Set the reference speed to 60 rpm.
3. Set the feedback damping factor to values 1, 0.5, 0.25, 0.1, 0.05 and 0.01. insert the response curve in each case (speed Vs time).
4. Write a comment describing the effect of changing the feedback damping factor on the rise time

Effect of Velocity Feedback:

In position control we take two main factors in our consideration when we design the controller; speed of response and elimination of overshoot. In case we work on a mobile robot, our goal is to make it reach the destination as fast as possible without any collisions. Any introduces overshoot means there is a probability that the robot hit some barrier located beyond the referenced location.

Hands on the PID controller: '<u>to be modified</u>'

PID stands for Proportional-Integral-Derivative controller. It is a simple but effective digital controller and is commonly used in industrial processes. Each term contributes to rapidly correct the error in system.

Understanding operation of each effector:

Proportional term (Kp) works by multiplying the error by a factor causing the manipulated signal to be larger and so the plant is given a higher signal to recover such error in the system output. The larger the proportional gain, the smaller the steady state error. But increasing Kp may increase the overshoot of the system. Moreover, in common digital controllers there is a limit in output voltage which means that if the manipulated output exceeded this limit, the signal is clipped causing nonlinearity in the system.

The integral term works on eliminating the steady state error. It accumulates the whole past errors and multiply the accumulation by a factor to get the manipulated signal. But the bad side of the integral term in digital controllers is that it may cause oscillations around the steady state value because of the quantization error of digital signals. So, the accumulation reaches zero after many oscillations.

The derivative term can be used to overcome the problem of both Proportional and Integral terms. It works on the error change rate. So, if the error is recognized and the system going toward the steady state value, the derivative term decreases the manipulated signal so that it can x the output just when it reaches the steady state for the first time without oscillations or overshooting. The case is called `critical response'. Now, let's talk about the implementation of this type of controllers.

Practical tuning:

Problems of using Integral Controller:

As we knew from the last part the integral controller helps eliminating the steady state error. But unfortunately in our case it may introduces some sort of instability. Regarding the problem of sensors resolution and the dead band of the motor, the integral controller accumulates reasonably large discrete errors until the output is sufficient to overcome the dead band of the motor causing the process to oscillate around the reference value. So, we may see that accepting small errors is more suitable than such uncontrolled oscillations.