***Introduction to Robotic Systems Course***

**LAB 3B**

**Basic Motor Control**

Contents

[1 Introduction 1](#_Toc30054731)

[1.1 Lab Overview 1](#_Toc30054732)

[2 Requirements 1](#_Toc30054733)

[3 Dynamixel Motors 1](#_Toc30054734)

[4 Task: Setting the IDE Environment 2](#_Toc30054735)

[4.1 Enter the Additional Board Manager URLs 2](#_Toc30054736)

[4.2 Set Board, Port, and Programmer in Tools Menu 2](#_Toc30054737)

[5 Task: Program the Dynamixel Motors 2](#_Toc30054738)

[5.1 Include Header Files and Variable/Constants Definition 2](#_Toc30054739)

[5.2 Edit Setup Function 3](#_Toc30054740)

[5.3 Edit Loop Function 4](#_Toc30054741)

[5.4 Save Sketch and Upload Program 4](#_Toc30054742)

# Introduction

## Lab Overview

In this lab, you will program the robot to perform simple and short movements in a straight line. The program will cause the robot to move forward and backward in a straight line.

# Requirements

The following hardware and software are required to complete this lab:

* **Hardware:** 
  + TurtleBot 3 Burger (robot).
  + ULINK-ME debugger.
* **Software:**
  + Arduino IDE.
  + If you have not used to Arduino IDE to program the board before, you might need to configure the IDE. This is detailed in the “Setting the IDE Environment” section below.

Reference may be made to the OpenCR1.0 manual that can be found using the link <http://emanual.robotis.com/docs/en/parts/controller/opencr10/>.

# Dynamixel Motors

Each wheel in the Turtlebot is attached to a Dynamixel motor. We will use the functions provided in the Dynamixel Workbench library to control the speed and direction of rotation of each Dynamixel Motor.

Details of the Dynamixel motor and its library can be found in the Dynamixel-Workbench using the following link: <http://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_workbench/>.

# Task: Setting the IDE Environment

Before you can write a program to control the Dynamixel Motors in the robot using the Arduino IDE, you must ensure to enter the correct parameters in Arduino IDE. These parameters are the board manager URLs, board, and port. The steps below will guide you on how to enter these parameters.

## Enter the Additional Board Manager URLs

Open Arduino IDE and then do the following:

* Click **File** -> **Preferences**
* In the Additional Boards Manager URLs textbox enter the correct url. The correct url can be found thus:
  + Go to the OpenCr1.0 manual website and expand the panel on the left **Arduino IDE** -> **Install** on <**your OS**> -> and click **Porting to Arduino IDE**.
  + The URL can be found under the Preferences section.
  + Copy the URL and paste it in the Preferences textbox and the click **Ok**.

## Set Board, Port, and Programmer in Tools Menu

On the Arduino IDE, select the correct board, port and programmer. Steps to set these are detailed below.

* Ensure OpenCR is selected for Board. To do this:
  + Click **Tools** -> **Board** and select **OpenCR Board**.
  + If **OpenCR Board** is not listed, click **Board Manager**.
    - Type **OpenCR** and OpenCR by ROBOTIS will appear.
    - Select the version you want to install (this course was tested with 1.3.2).
    - Click **install**.
    - Once installed, select **OpenCR Board**.
* Select the correct port. To do this:
  + Click **Tools** -> **Ports** and select the correct port (the board must be connected to PC).
* Select **DFU\_UTIL** for **Programmer**. To do this:
  + Click **Tools** -> **Programmer** and select **DFU\_UTIL**.

# Task: Program the Dynamixel Motors

In this section, you will write the program to control the Dynamixel Motors. Edit the sketch environment (blank environment) opened when you start the Arduino IDE.

## Include Header Files and Variable/Constants Definition

The first step is to define the header files and other constant variables.

Enter the following at the top of your sketch.

1. #include <DynamixelWorkbench.h>
2. #define DXL\_Serial "/dev/ttyACM0"
3. #define BAUDRATE 1000000
4. #define DXL\_LEFT 1
5. #define DXL\_RIGHT 2
6. DynamixelWorkbench dxl\_wb;

The first line includes the Dynamixel header file in the project. This file contains the definitions of most functions that will be used in the program’s body. These functions can be found in the Dynamixel-Workbench library.

The second line defines a macro DXL\_Serial that is substituted for the string **/dev/ttyACM0**. The third, fourth, and fifth lines also define macros.

## Edit Setup Function

In this step, we will edit the setup function. This function is called once at the start of the program. It is used to initialize variables, set pin modes, and start using defined libraries.

Enter the following lines of code:

1. void setup()
2. {
3. Serial.begin(57600);
4. //while(!Serial); // Open a Serial Monitor
5. dxl\_wb.begin(DXL\_Serial, BAUDRATE);
6. dxl\_wb.ping(DXL\_RIGHT);
7. dxl\_wb.ping(DXL\_LEFT);
8. dxl\_wb.wheelMode(DXL\_LEFT);
9. dxl\_wb.wheelMode(DXL\_RIGHT);
10. }

Serial.begin (57600) sets the data rate for a serial transmission.

The rest of the lines of code inside the setup function make use of the libraries defined in DynamixelWorkbench.h. The device name and baud rate are first initialized. Next, the right and left wheels are pinged to establish a connection to the motors and then set to wheel mode.

## Edit Loop Function

In this step, we will edit the loop function. This function will run indefinitely after the setup function.

Enter the following lines of code:

1. void loop()
2. {
3. dxl\_wb.goalSpeed(DXL\_LEFT, 30);
4. dxl\_wb.goalSpeed(DXL\_RIGHT, 30);
5. delay(2000);
6. dxl\_wb.goalSpeed(DXL\_LEFT, -30);
7. dxl\_wb.goalSpeed(DXL\_RIGHT,-30);
8. delay(2000);
9. }

This function sets the same speed to both wheels in clockwise and anticlockwise rotation. The wheels rotate for the period set by the delay period of 2000 milliseconds before changing directions.

## Save Sketch and Upload Program

Once satisfied with your code, Save the sketch, click the **Verify** button and then the **Upload** button to program the board.

If there is no error, the robot will move forward and backward in a straight line. Edit the delay period to change the distance of each movement.