LABORATORY 3

ACTUATORS, DRIVES, AND CONTROL COMPONENTS

Ma. Gellan A. Caples
College of Engineering
Bachelor of Science in Electronics Engineering
Samar State University
magellancaples@gmail.com

***Abstract*—This experiment explores the different types of actuators and control components used in robotics, examining their interaction with microcontrollers. Students will manage various actuators using PWM and other control signals, testing and validating their performance both on physical hardware and in simulations.**

***Index Terms*—Actuators, PWM, DC motor, Servo motor, Stepper motor, Robotics, Webots.**

I. Rationale

This experiment focuses on understanding the types of actuators and control components in robotics and how they interact with microcontrollers. Students will control various actuators through PWM and other control signals.

II. Objectives

- Interface and control at least three actuators (DC motor, servo motor, and stepper motor) using the Arduino, ensuring correct control of speed and direction.
- Use PWM signals to control the speed of a DC motor and observe the change in motor speed across a range from 50% to 100%.
- Simulate the movement of motors in Webots with a success rate of 95% for correct motor movement and speed.

III. Materials and Software

A. Materials

- Arduino Uno
- DC motors
- Servo motors
- Stepper motor
- L298N motor driver
- ULN2003 stepper motor driver
- Breadboard
- Jumper Wires

B. Software

- Arduino IDE
- Webots Simulation Environment

IV. Procedures

1) Set up Arduino to control DC motors, servo motors, and stepper motors.
2) Generate PWM signals to vary motor speed and direction.
3) Write the control code for motor management in Arduino IDE.
4) Simulate the actuators in Webots and check for correct performance.

V. Results

The stepper motor has been set to complete one full revolution approximately every 15 seconds, which results in a frequency of around 0.067 Hz. This configuration allows for precise and controlled rotation over a set period of time.

For the DC motor, three distinct speed modes have been defined, with PWM values of 55, 125, and 255 used to regulate its speed. In addition, the servo motor has been programmed to sweep continuously from 0 to 180 degrees with each loop iteration, providing a full range of back-and-forth motion.

VI. Discussion

The stepper motor demonstrated accurate but relatively slow movement, completing one full rotation in approximately 15 seconds, corresponding to a frequency of 0.067 Hz. This performance matched expectations when using the ULN2003 driver at the chosen stepping rate, showing reliable and controlled motion over time.

The DC motor responded linearly to adjustments in the PWM duty cycle. By using analogWrite values of 60, 130, and 255, we observed distinct low, medium, and high-speed modes, confirming that increasing the duty cycle led to a proportional increase in both torque and rotational speed (RPM). At the same time, the servo motor consistently completed a full sweep from 0° to 180° with each iteration, effectively demonstrating precise position control via standard PWM signal modulation.

VII. Conclusion

This experiment effectively showcased the integration and control of a stepper motor, DC motor, and servo motor using PWM signals produced by an Arduino Uno. The DC motor displayed distinct speed modes that clearly corresponded to adjustments in the PWM duty cycle, demonstrating predictable and consistent behavior.

The stepper motor operated with stable, low-frequency rotational movement, maintaining a steady pace as expected. Meanwhile, the servo motor consistently executed full-range sweeps from 0° to 180°, validating its ability to achieve accurate position control through PWM signal modulation.

REFERENCES

[1] STM32f103c6 Documentation, "STM32f103c6 ", https://www.st.com/en/microcontrollers-microprocessors/stm32f103c6.html.
[2] Cyberbotics Ltd., "Webots User Guide", 2024.
[3] ULN2003 stepper motor driver, https://www.ti.com/product/ULN2003A

APPENDIX

Actuators

```
#include <Servo.h>

Servo myservo;

//karan stepper pins
int pin1 = 8;
int pin2 = 9;
int pin3 = 10;
int pin4 = 11;

//balyu e kun ano an iyo mga gin gamit
int dc_motor_pin1 = 7;
int dc_motor_pin2 = 6;


int pinslist[] = {pin1, pin2, pin3, pin4};
int speed_switch_pin = 3;


//stepper controller
void stepper(int ms_delay) {
  digitalWrite(pin1, HIGH);
  digitalWrite(pin2, LOW);
  digitalWrite(pin3, LOW);
  digitalWrite(pin4, LOW);
  delay(ms_delay);

  digitalWrite(pin1, LOW);
  digitalWrite(pin2, HIGH);
  digitalWrite(pin3, LOW);
  digitalWrite(pin4, LOW);
  delay(ms_delay);

  digitalWrite(pin1, LOW);
  digitalWrite(pin2, LOW);
  digitalWrite(pin3, HIGH);
  digitalWrite(pin4, LOW);
  delay(ms_delay);

  digitalWrite(pin1, LOW);
  digitalWrite(pin2, LOW);
  digitalWrite(pin3, LOW);
  digitalWrite(pin4, HIGH);
  delay(ms_delay);
}


// mode 1 = max speed; mode 2 = medium; mode 3 = slow;
void dc_motor_driver(bool reverse, int speed_mode){

  switch(speed_mode){
    case 1:
      if(!reverse){
        digitalWrite(dc_motor_pin1, 0);
        analogWrite(dc_motor_pin2, 255);
      }else{
        digitalWrite(dc_motor_pin2, 0);
        analogWrite(dc_motor_pin1, 255);
      }
    break;
    case 2:
      if(!reverse){
          digitalWrite(dc_motor_pin1, 0);
          analogWrite(dc_motor_pin2, 125);
        }else{
          digitalWrite(dc_motor_pin2, 0);
          analogWrite(dc_motor_pin1, 125);
        }
    break;
    case 3:
      if(!reverse){
          digitalWrite(dc_motor_pin1, 0);
          analogWrite(dc_motor_pin2, 55);
        }else{
          digitalWrite(dc_motor_pin2, 0);
          analogWrite(dc_motor_pin1, 55);
        }
    break;
    default:
    break;

  }

}

void servo_turn(){

  myservo.write(0);
  delay(500);
  myservo.write(180);

}



void setup(){

  for(int i=0; i < sizeof(pinslist)/sizeof(pinslist[0]); i++){
    pinMode(pinslist[i], OUTPUT);
    digitalWrite(pinslist[i], 0);
  }
  pinMode(dc_motor_pin1, OUTPUT);
  pinMode(dc_motor_pin2, OUTPUT);
  myservo.attach(9);
  pinMode(speed_switch_pin, INPUT);


}

void loop(){

  if(digitalRead(speed_switch_pin)){

    stepper(300);
    dc_motor_driver(true, 1);
```

```
116       servo_turn();
117
118     }
119     //delay(100);
120
121 }
```