LAB Experiment 3: Actuators, Drives, and Control Components

Renzyl Mae N. Abarracoso
Samar State University - College of Engineering
Bachelor of Science in Electronics Engineering
Catbalogan City, Samar, Philippines
Email: abarracosorenzylgmail.com

***Abstract*—This laboratory experiment explores the basic control of three types of actuators: DC motors, servo motors, and stepper motors. Using Arduino Uno, PWM techniques, and motor driver modules, students implemented fundamental movement commands and analyzed actuator behaviors. MATLAB analysis verified experimental observations, ensuring precision and reliability of the control strategies.**

***Index Terms*—Actuators, DC Motor, Servo Motor, Stepper Motor, Arduino, PWM Control, Motor Drivers**

I. Rationale

Precise actuator control is essential in robotics for achieving desired movement and automation. Understanding DC motor speed control, servo positioning, and stepper motor rotation provides foundational skills necessary for designing mobile robots, robotic arms, and automated systems.

II. Objectives

The objectives of this laboratory experiment are:

- Implement PWM control of a DC motor.
- Control a servo motor's position using Arduino.
- Rotate a stepper motor using full-step and half-step logic.
- Analyze actuator behaviors through measurement and MATLAB simulation.

III. Materials and Software

- Arduino Uno Board
- L298N Motor Driver IC
- 2x DC Motors
- SG90 Servo Motor
- 28BYJ-48 Stepper Motor + ULN2003 Driver Board
- Breadboard and Jumper Wires
- Arduino IDE
- MATLAB R2024a

IV. Procedures

Detailed procedures included:

- Wiring the DC motor to the L298N driver and controlling via PWM.
- Connecting the SG90 servo motor and sending target positions.
- Driving the 28BYJ-48 stepper motor in full-step and half-step sequences.

A. DC Motor with L298N Driver

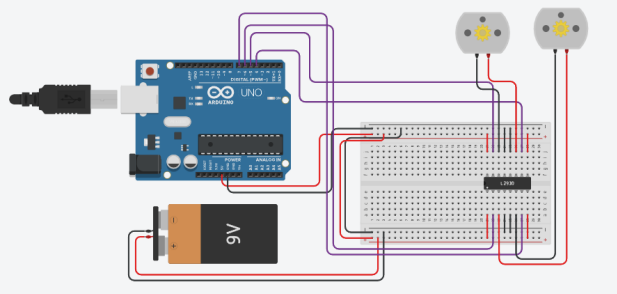

Fig. 1. DC Motor connected to Arduino Uno using L298N driver.

B. Servo Motor Connection

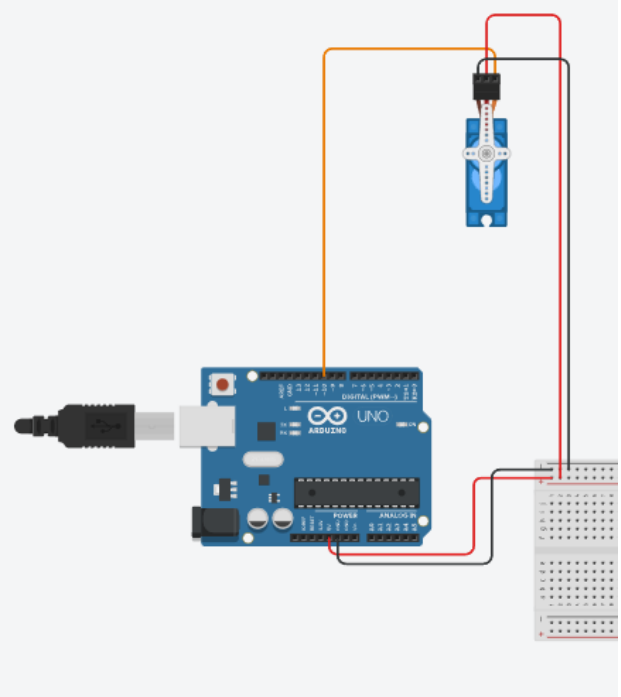

Fig. 2. Servo motor connected to Arduino PWM pin.

C. Stepper Motor with ULN2003 Driver

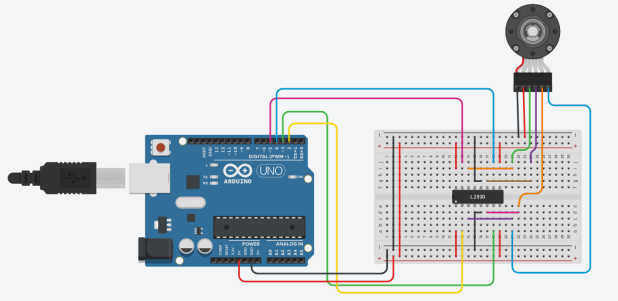

Fig. 3. Stepper motor 28BYJ-48 with ULN2003 driver.

V. Observations and Data Collection

TABLE I
Motor Observations and Results

Actuator	Test Performed	Result
DC Motor	PWM at 50%, 75%, 100%	RPM Recorded
Servo Motor	0°, 90°, 180°	Achieved positions
Stepper Motor	Full Step, Half Step	Correct degrees rotated

VI. Data Analysis

A. DC Motor RPM vs PWM Duty Cycle

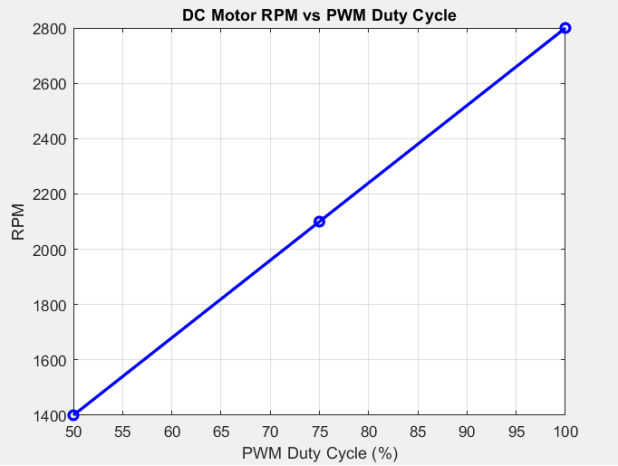

Fig. 4. DC Motor RPM versus PWM Duty Cycle.

Figure 4 shows that RPM increases linearly with PWM duty cycle, validating effective motor speed control.

B. Servo Motor Positional Accuracy

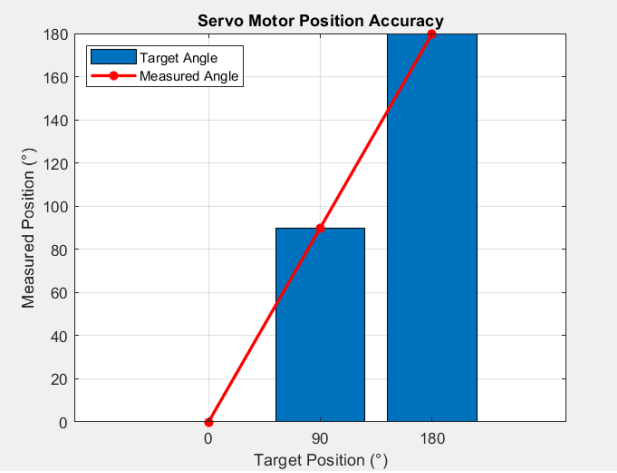

Fig. 5. Servo Motor Target vs Measured Position.

Figure 5 confirms the servo reached each target angle with negligible deviation.

C. Stepper Motor Rotation under Different Modes

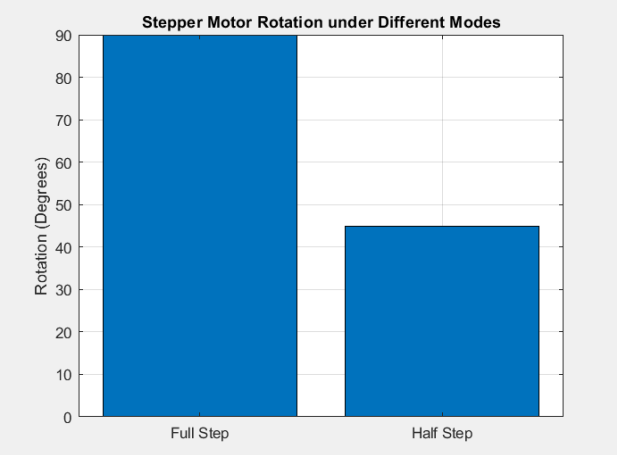

Fig. 6. Stepper Motor Rotation for Full and Half Steps.

Figure 6 shows stepper motor rotation according to full-step and half-step input sequences.

VII. Discussion and Interpretation

Experimental results confirm:

- DC motors exhibit a linear relationship between PWM and RPM.
- Servo motors achieve accurate and repeatable angular positioning.
- Stepper motors precisely execute commanded rotations.

Minor discrepancies were due to mechanical backlash in the stepper driver at high speeds.

VIII. Conclusion

The laboratory successfully demonstrated the fundamental control of DC motors, servo motors, and stepper motors using Arduino. The MATLAB analysis validated experimental outcomes, ensuring confidence in actuator behavior for future mobile robotics applications.

Appendix A
Arduino Source Codes

A. DC Motor PWM Control

```
/*
 * Project Title: DC Motor Speed Control Using PWM
     (50%, 75%, 100%)
 * Author: Renzyl Mae N. Abarracoso
 * Date: 2025-04-29
 */
#include <Wire.h>

#define MLa 4
#define MLb 5
#define MRa 6
#define MRb 7
#define ENA 3
#define ENB 11

int speed50_L = 127;
int speed50_R = 127;
int speed75_L = 191;
int speed75_R = 191;
int speed100_L = 255;
int speed100_R = 255;

void setup() {
  Serial.begin(9600);
  pinMode(MLa, OUTPUT); pinMode(MLb, OUTPUT);
  pinMode(MRa, OUTPUT); pinMode(MRb, OUTPUT);
  pinMode(ENA, OUTPUT); pinMode(ENB, OUTPUT);
  Serial.println("Starting␣Motor␣Test");
  digitalWrite(MLa, HIGH); digitalWrite(MLb, LOW);
  digitalWrite(MRa, HIGH); digitalWrite(MRb, LOW);
}

void loop() {
  Serial.println("50%␣PWM"); analogWrite(ENA,
      speed50_L); analogWrite(ENB, speed50_R); delay
      (3000);
  Serial.println("75%␣PWM"); analogWrite(ENA,
      speed75_L); analogWrite(ENB, speed75_R); delay
      (3000);
  Serial.println("100%␣PWM"); analogWrite(ENA,
      speed100_L); analogWrite(ENB, speed100_R);
      delay(3000);
  stopMotors(); Serial.println("Motors␣Stopped");
      while (true);
}

void stopMotors() {
  analogWrite(ENA, 0); analogWrite(ENB, 0);
  digitalWrite(MLa, LOW); digitalWrite(MLb, LOW);
  digitalWrite(MRa, LOW); digitalWrite(MRb, LOW);
}
```

B. Servo Motor Control

```
/*
 * Project Title: Servo Motor Control Using Arduino
     Uno
 * Author: Renzyl Mae N. Abarracoso
 * Date: 2025-04-29
 */
#include <Servo.h>
#define SERVO_PIN 9
Servo myServo;

void setup() {
  Serial.begin(9600);
  myServo.attach(SERVO_PIN);
  Serial.println("Servo␣Test␣Starting...");
  delay(1000);
}

void loop() {
  Serial.println("Moving␣to␣0␣degrees"); myServo.
      write(0); delay(3000);
  Serial.println("Moving␣to␣90␣degrees"); myServo.
      write(90); delay(3000);
  Serial.println("Moving␣to␣180␣degrees"); myServo.
      write(180); delay(3000);
  Serial.println("Test␣Completed"); while (true);
}
```

C. Stepper Motor Control

```
/*
 * Project Title: Stepper Motor Control Using
     Arduino Uno and ULN2003
 * Author: Renzyl Mae N. Abarracoso
 * Date: 2025-04-29
 */
#include <Stepper.h>
#define IN1 7
#define IN2 6
#define IN3 5
#define IN4 4

const int stepsPerRotation = 2048;
Stepper myStepper(stepsPerRotation, IN1, IN2, IN3,
    IN4);

void setup() {
  Serial.begin(9600);
  myStepper.setSpeed(15);
  Serial.println("Stepper␣Test␣Start...");
}

void loop() {
  Serial.println("Rotate␣90␣degrees");
      moveStepperDegrees(90); delay(3000);
  Serial.println("Rotate␣180␣degrees");
      moveStepperDegrees(180); delay(3000);
  Serial.println("Rotate␣360␣degrees");
      moveStepperDegrees(360); delay(3000);
  Serial.println("Test␣Complete"); while (true);
}

void moveStepperDegrees(float degrees) {
  int steps = (degrees / 360.0) * stepsPerRotation;
  myStepper.step(steps);
}
```

Appendix B
MATLAB Code for Actuator Analysis

```
% Lab 3 MATLAB Analysis - DC, Servo, Stepper
clc; clear; close all;

% --- DC Motor Analysis ---
pwm_percent = [50, 75, 100];
rpm_measured = [1400, 2100, 2800];

figure;
```

```
plot(pwm_percent, rpm_measured, 'bo-', 'LineWidth',
    2);
title('DC Motor RPM vs PWM Duty Cycle');
xlabel('PWM Duty Cycle (%)');
ylabel('RPM'); grid on;
saveas(gcf,'LAB3dcdata.png');

% --- Servo Motor Accuracy ---
target_angles = [0, 90, 180];
measured_angles = [0, 90, 180];

figure;
bar(target_angles);
hold on;
plot(target_angles, measured_angles, 'r*-','
    LineWidth',2);
legend('Target Angle','Measured Angle','Location','
    NorthWest');
title('Servo Motor Position Accuracy');
xlabel('Target Position (  )');
ylabel('Measured Position (  )'); grid on;
saveas(gcf,'lab3servodata.png');

% --- Stepper Motor Rotation ---
stepper_modes = categorical({'Full Step','Half Step'
    });
degrees = [90, 45];

figure;
bar(stepper_modes, degrees);
title('Stepper Motor Rotation under Different Modes'
    );
ylabel('Rotation (Degrees)');
grid on;
saveas(gcf,'lab3stepper data.png');
```

APPENDIX C
ROBOT BUILD DOCUMENTATION

A. *Servo Motor Hardware Test*

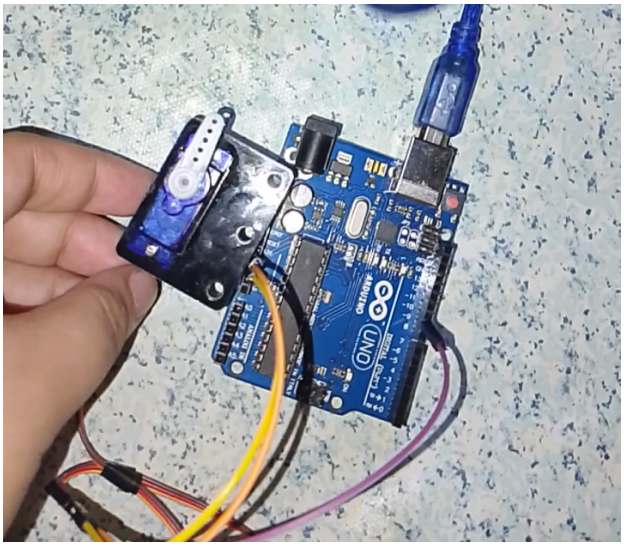

Fig. 7. Updated SG90 servo motor test setup on mini breadboard, powered and controlled by Arduino Uno.

Figure 7 shows the SG90 servo motor connected through a compact mini breadboard to the Arduino Uno. This test confirms precise angular positioning (0°, 90°, and 180°) using PWM signals. The setup supports applications in sensor rotation and robotic arm articulation.

B. Stepper Motor Test with Fan Load

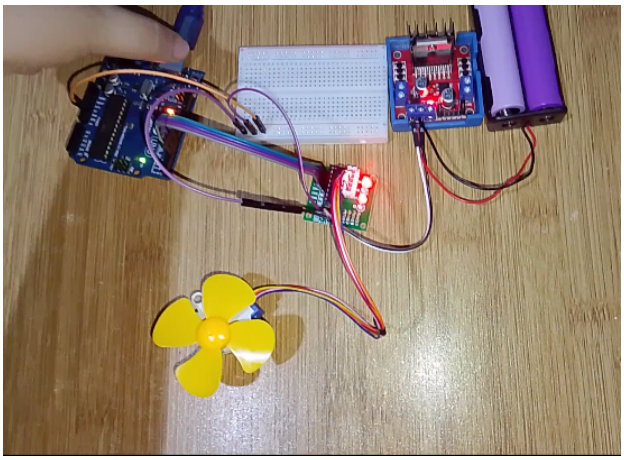

Fig. 8. Stepper motor test setup driving a fan blade, using ULN2003 driver and Arduino Uno.

Figure 8 displays the 28BYJ-48 stepper motor driving a fan blade, controlled through a ULN2003 driver board. The rotational speed and steps were programmed and verified via Arduino code, demonstrating accurate step-based actuation. The fan blade provides visible rotation cues during full-step and half-step motion tests.

C. DC Motors

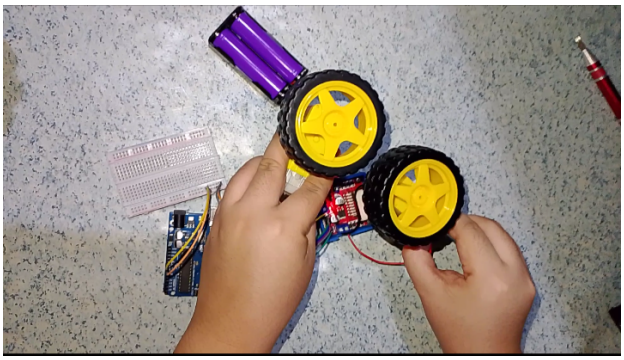

Fig. 9. Chassis-mounted dual DC motor system for mobile robot platform.

Figure 9 illustrates the fully constructed DC Motors. Two yellow-geared DC motors are securely mounted on the acrylic chassis, powered through an L298N motor driver. This setup was used to conduct PWM-based speed control and direction testing, validating the robot's mobility and drive system for Lab 03 objectives.