Experiment M5
Servo Motors
Procedure

Deliverables: Checked score sheet, tech memo

Overview

Background

In this lab, you will construct a small mechatronic measurement system using a servo motor and an ultrasonic rangefinder. This will essentially be a sonar system that will produce a 270° polar plot of nearby objects.

A servo motor is a simple DC motor with a gear box and a sensor (potentiometer, quadrature encoder, or magnetic sensor) for measuring the shaft position, which provides feedback so that the shaft can be set to a precise angular position. Servos are very good at controlling angular position (or angular speed for a "continuous servo"), but they are typically not good for controlling torque.

You will use an inexpensive servo motor in this lab. These small servos are typically used by hobbyists for building remote control cars and planes. Industrial grade servo motors are usually capable of controlling position, speed, and torque. However, they are very expensive and require high voltage power that presents a shock hazard.

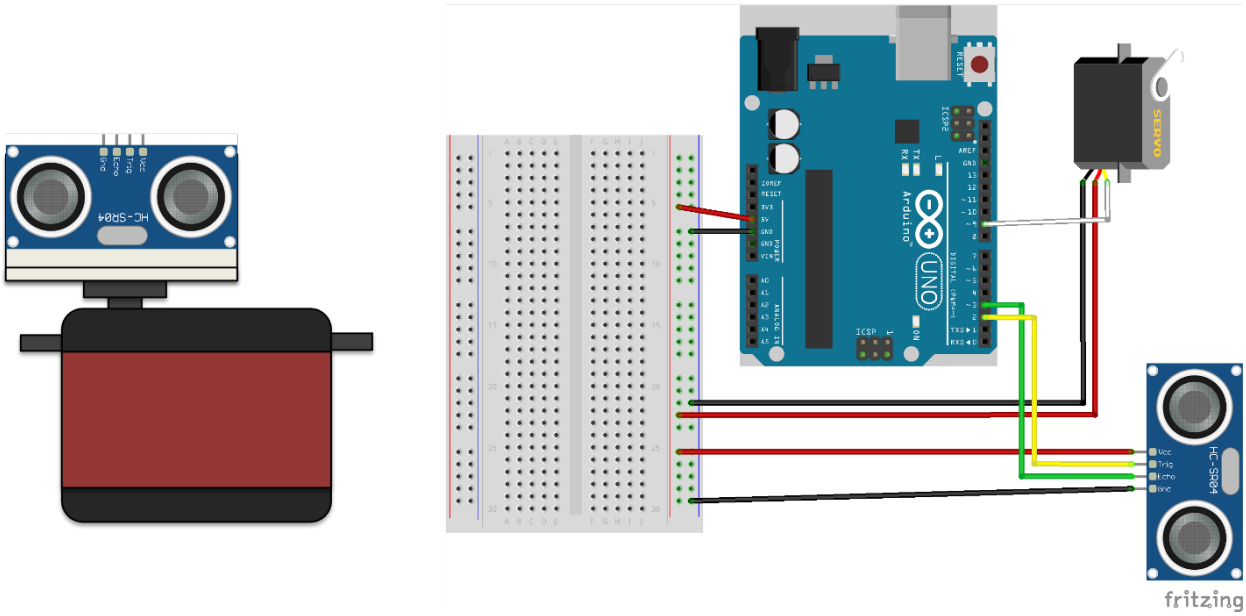

Figure 1 – (Left) An ultrasonic rangefinder is mounted on top of a servo motor. (Right) A wiring diagram shows how the ultrasonic rangefinder and servo motor are connected to the Arduino microcontroller.

Part I: Testing the Servo

You will begin by connecting the servo motor to the function generator. These small servo motors read in a 50 Hz digital pulse with a width between 1000 – 2000 μs. The width of the pulse tells the servo what position to set the shaft:

- 1000 μs (1 ms) pulse width turns the shaft to its minimum position. (In this case 0°.)
- 2000 μs (2 ms) pulse width turns the shaft to its minimum position. (In this case 270°.)
- Any pulse width in between is linearly mapped to an angle between the two limits.

Procedure

1. Mount the servo to basswood sheet with the hole pattern that you made, so that the shaft is pointing upward. Use the 8-32 stand-offs, nuts, nuts and flathead screws. (A drill is available if you need to alter your hole pattern.)
2. Connect the 5V and GND pins on the Arduino to the vertical bus lines of the breadboard as shown in Fig. 1.
3. Connect the red and black wires of the servo to the vertical bus lines of the breadboard as shown in Fig. 1.
4. With the BNC-to-minigrabber cable, connect the 'T' on the "50Ω" output of the function generator to the servo. Connect the red mini-grabber to the white wire on the servo. The black mini-grabber is connected to ground.
5. Connect the other side of 'T' on the function generator to CH1 on the oscilloscope (scope).
6. BNC-to-minigrabber cable, connect V_{out} to CH2 of the oscilloscope. Make sure the black minigrabber is connected to ground, and the red one is connected to the resistor-capacitor junction.
7. Turn on the function generator. Press the "sine" button, then "output menu", then "load impedance", then "High Z", then press "Top Menu" to exit to the main menu.
8. Use the Tektronix function generator to apply a 50 Hz pulse to the third wire (yellow or white) of the servo.
   a. Set the Amplitude to a high value of 5V, low value of 0V.
   b. Set the duty/width to 1 ms.
   c. Press the "ON" button above the BNC T on the function generator. (Not the power button. This is a different "ON" button located in the lower-middle portion of the device panel.)
9. Make sure the Arduino is connected to the computer.
10. Adjust the duty/width of the pulses on the function generator to values between 1 and 2 ms. You should see the servo rotate.
11. **Demonstrate this to the TA to receive credit on your score sheet.**

Part II: Microcontroller Implementation

You will now use the Arduino UNO microcontroller to generate the pulses and control the position of the servo.

Procedure

1. Disconnect the function generator from the servo. Connect the white signal wire on the servo to pin 9 on the Arduino.
2. Download the Servo Example 1 code from the course website. Fill in the '***' with a value between 0 and 180. Save it to your code library, then upload it to the Arduino. You should see the servo rotate.
3. Repeat the previous step with a value of 0. Then again with a value of 180. Did the servo rotate 180 degrees? Or did it rotate to a different angle?
4. Use the oscilloscope to measure the output from pin 9 of the Arduino. In particular, what is the pulse width in ms when the servo.write() value is 0? What is the pulse width in ms when the servo.write() value is 180? Write the mapping down in your lab notebook.
5. Download the Servo Example 2 code from the course website. Read it, try to understand how it works. Save it to your code library, then upload it to the Arduino. You should see the servo periodically rotate, then return to its home position.
6. **Demonstrate this to the TA to receive credit on your score sheet.**

Part III: Design Challenge 1

Construct a system where a user can control the angular position of the servo by turning the potentiometer.

- Use the potentiometer, circuit, and code from the M1 lab.
- Use the map() function to map the minimum value of the potentiometer to the minimum value of servo, and the maximum value of the potentiometer to the maximum value of servo.
- The servo should rotate the same direction as the potentiometer knob.

Part IV: Ultrasonic Rangefinder

An ultrasonic rangefinder measures distance by sending out an ultrasonic 'ping' from a transmitter, then measuring the time it take for the echo to return to a receiver. This is how active sonar work, except sonar is typically used in water while the ultrasonic rangefinder is used in air.

Procedure

1. Insert the ultrasonic rangefinder into the plastic mount with the pins pointing up.
2. Wire up the ultrasonic rangefinder by making the following connections:
    a. GND on Rangefinder → GND bus line on breadboard
    b. Vcc on Rangefinder → +5V bus line on breadboard
    c. Trig on Rangefinder → Pin 2 on Arduino
    d. Echo on Rangefinder → Pin 3 on Arduino
3. Add the HCSR04 ultrasonic rangefinder code library to the Arduino IDE:
    a. Download the HCSR04 Rangefinder Library from the M5 webpage. Right click the link, then click "Save link as…".
    b. Go to "Sketch" > "Include Library" > "Add ZIP Library...", navigate to the file you just downloaded "HCSR04-ultrasonic-sensor-lib-master.zip", and select it.
    c. If it gives you an error that the library "already exists", don't worry. That simply means the student before you already loaded the library.
4. Download the Ultrasonic Example code from the course website. Read it, try to understand how it works. Save it to your code library, then upload it to the Arduino.
5. Go to "Tools" > "Serial Monitor" (or press "Ctrl + Shift + M") to view the output from the "Serial.print()" commands at the bottom of the screen. Make sure the baud rate is set to 9600. You should see the measured distances printed repeatedly.
6. Point the sensor at various nearby objects. Use a tape measure or meterstick to check its accuracy.

NOTE: The rangefinder outputs 0 if it does not receive a strong return echo. This usually means the nearest object is > 200 cm away.

7. Attach the ultrasonic transducer to the servo shaft using one of the plastic shaft couplings and small screws.
8. Combine the various codes to create a program where the servo steps through angles between 0 and 270 degrees and measures the distance to the nearest object at each angle.
    a. Use "Serial.print()" commands to display the angle θ in degrees followed by the distance r in cm. Both values should be separated by a comma and a space.
9. **Demonstrate this to the TA to receive credit on your score sheet.**

Part V: Pneumatic Missile Launcher

A solenoid valve will be used to release a puff of compressed air that will launch a small foam rocket.

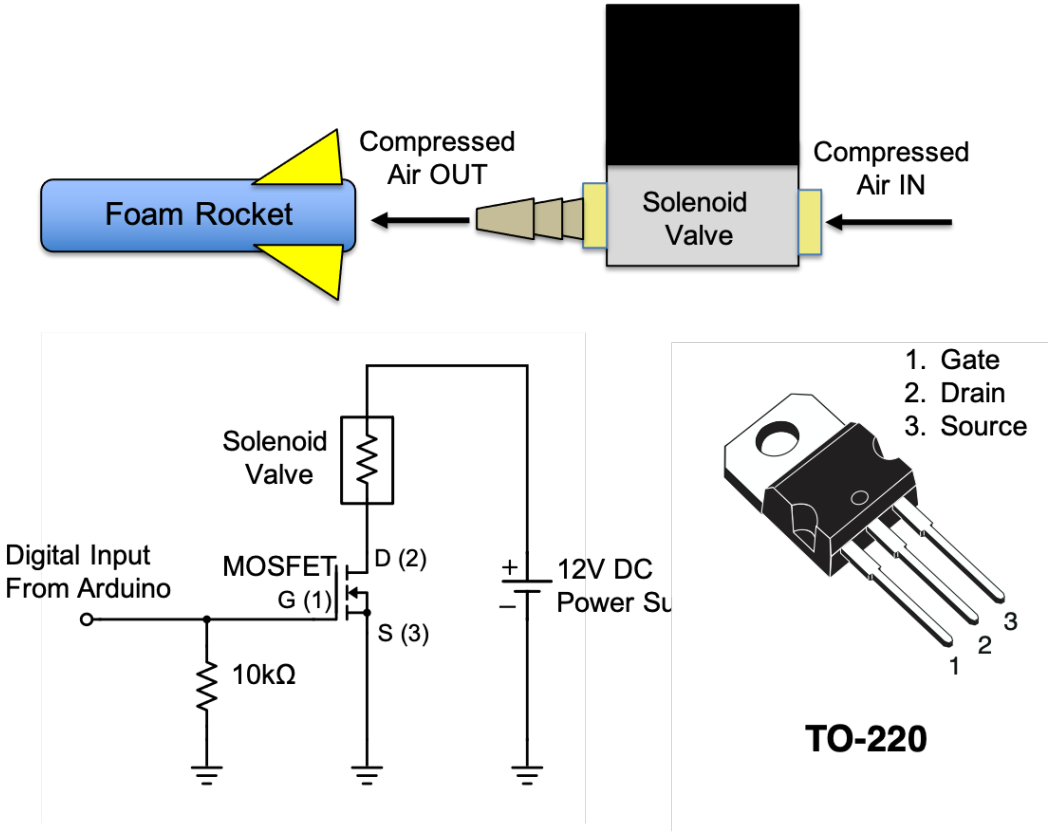

Figure 2 – (Top) A puff of compressed air from the solenoid valve propels the foam rocket (missile) forward. (Bottom) A MOSFET circuit is used to switch the solenoid valve.

Procedure

1. Locate the 12V DC power supply. (It looks like a laptop charger.)
   a. Connect the 5.5mm barrel connector on the 12V DC power supply to the inline kill switch. Make sure the kill switch is in the OFF position.
   b. Connect the other end of the kill switch to the screw terminal adapter.
   c. Connect red and black wires on the pneumatic cylinder to the + and – screw terminals.
2. Use the inline kill switch to turn the power supply ON and OFF. You should here the solenoid valve click.
3. Open the quarter-turn ball valve to pressurize the system.
4. Toggle the 12V power supply kill switch to open and close the solenoid valve, thus releasing the compressed air.
5. Place the foam rocket on the brass nozzle coming off the solenoid valve. Launch the rocket by opening the solenoid valve.

6. Disconnect the solenoid valve from the DC power supply.
7. Connect the 12V power supply to the other vertical bus lines on the breadboard. Establish a common ground by connecting the negative bus lines together.
8. Construct the MOSFET circuit shown on the bottom of Fig. 2. Use one of the digital outputs from the Arduino to drive the gate of the MOSFET, thus switch the solenoid valve.
9. Write an Arduino code that uses serial.println() to pring a 10 second countdown to the serial monitor (10, 9, 8, 6, 5, 4, 3, 2, 1, FIRE!). Then switches the MOSFET gate to HIGH for 1 second, thus energizing the solenoid and firing the rocket. After 1 second, it should go back to LOW.
10. **Demonstrate this to the TA to receive credit on your score sheet.**

Part VI: Design Challenge 2

Build a system using the servo, ultrasonic rangefinder, and pneumatic missile launcher that is able to detect the nearest stationary object and fire the pneumatic missile at the target.

- Design and build a 'turret' that mounts to the shaft of the servo. The ultrasonic rangefinder and pneumatic launcher should both be mounted to the servo. Use any of the plastic shaft couplings, foam core board, hot glue, tape, and velcro command strips to accomplish this.
- Do NOT put hot glue on any part of the pneumatic solenoid valve or the servo. Use command strips instead.
- Consider the weight of solenoid valve in your design. Try to minimize the rotational inertia of the turret.
- Combine your previous codes to rotate the turret, locate the target, aim, and fire. The turret should rotate the full 270 degrees and record the location of the closest object. Then, aim the missile at that bearing and fire.
- The system should print the current angle and range to the serial monitor as it scans. When it has determined the location of the target, it should print "Bearing to target: **angle*".
- Be mindful of the wires and tubing, so they do not get tangle or unplugged as the turret rotates.
- Think about filtering data by computing some type of average on the measured ranges. Also, the rangefinder will output a distance of 0 if it does not get a good reading.

Clean-up

To receive full credit, you must return the lab bench to its initial state:

- Unplug the DC power supply. Remove the kill switch and screw terminal adapter.
- Disassemble the circuit. Disconnect the wires from the stepper driver, Arduino, and breadboard.
- Sweep away and dispose any dried hot glue, foam core scraps, and balsa wood scraps.
- Put anything that belongs in your tool kit back into the tote bin.
- Remove the servo from the mount. Put the 8-32 stand-offs, screws, and nuts in a plastic bag in your tote bin. Put the tote bin and basswood sheet in the cabinet above your lab bench.

Appendix A

Equipment

- 12V DC power supply (looks like a laptop charger)
- BNC to minigrabber cable
- Arduino Microcontroller
- HC-SR04 Ultrasonic Rangefinder with mount (Amazon Part # B07SC1YJ21)
- 270 degree servo motor (Amazon Part # B07DHP2922)
- 12V Pneumatic solenoid valve w/ 1/4” barbed hose fitting (Amazon Part # B09H42GCHG)
- 1/4” tubing w/ quick connect pneumatic coupling and quarter turn valve
- Toy foam rockets
- Hot glue guns
- Foam core
- Balsa wood