Design Example

ENGF0001 Engineering Challenges

Content

- Introduction
- Distance Measurement Subsystem
- Motor Control Subsystem
- Distance Control Logic and Code
- Experimental Results
- Conclusion

Content

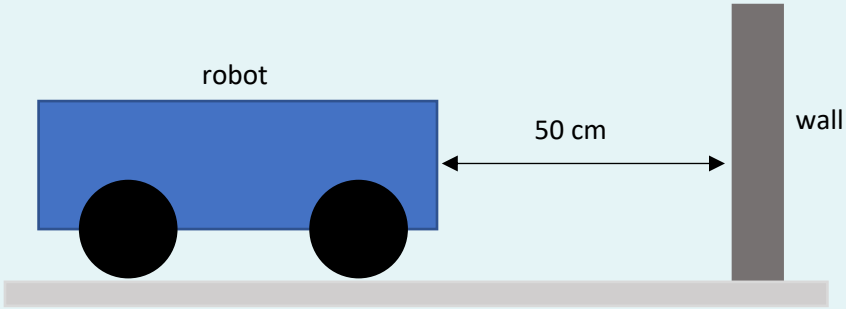
- Introduction
- Distance Measurement Subsystem
- Motor Control Subsystem
- Distance Control Logic and Code
- Experimental Results
- Conclusion

Introduction (1)

- The following is an example of the design process.
- It has no relationship with the bio-reactor project you are working on, but it shows you the steps in completing a project.

Introduction (2)

- Purpose of the distance control system:
  - To maintain a desired distance between the robot and the wall.



  - The system can be divided into two parts:
    - The distance measurement subsystem, and
    - The motor control subsystem.

Content

- Introduction
- Distance Measurement Subsystem
- Motor Control Subsystem
- Distance Control Logic and Code
- Experimental Results
- Conclusion

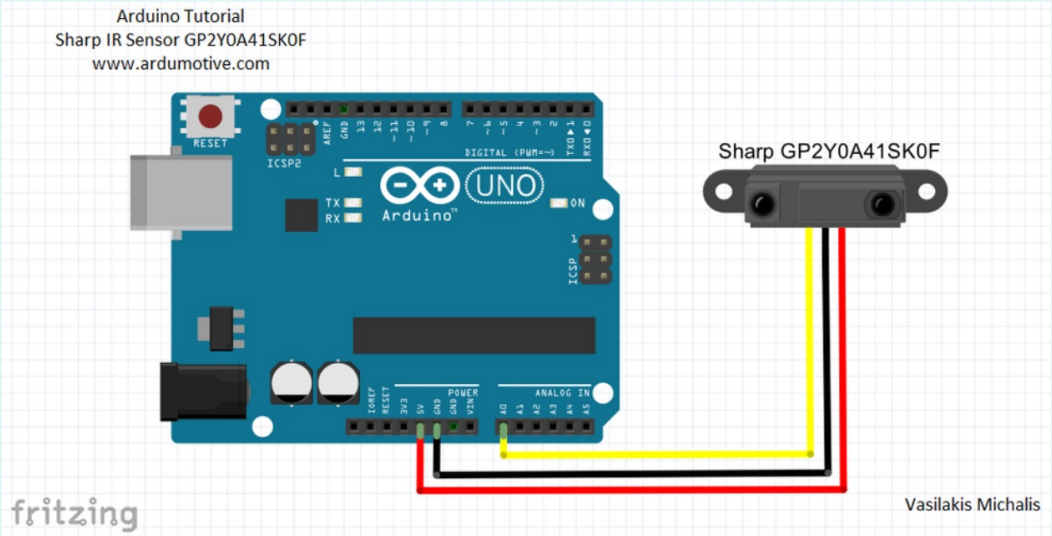
The Sensor

- The sensor used for distance measurement in this project is the Sharp GP2Y0A02YK0F analog infrared sensor [1].

[1] Sharp, “Distance Measuring Sensor Unit Measuring distance: 20 to 150 cm Analog output type”, GP2Y0A02YK0F datasheet [Online]. https://docs.rs-online.com/268d/0900766b81364822.pdf (accessed 5 Sep 2022)

The Sensor Circuit (1)

- The sensor circuit is shown on the right [2]:
  - The Arduino board supplies 5V to the sensor
  - The sensor provides an analog signal corresponding to the distance back to Arduino.



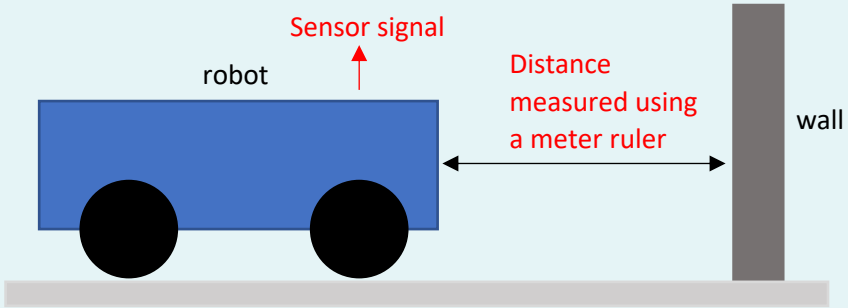
[2] Codebender_cc, “How to Use the Sharp IR Sensor (GP2Y0A41SK0F) - Arduino Tutorial,” [Online]. https://www.instructables.com/How-to-Use-the-Sharp-IR-Sensor-GP2Y0A41SK0F-Arduin/ (accessed 5 Sep 2022)

The Sensor Circuit (2)

- Note for students:
  - This circuit in previous slide was taken from the internet.
    - For your system, you should design your own circuit!
  - Also, the circuit was very simple because all the electronics are integrated within the sensor unit.
    - For your own subsystem, you should include circuits for amplification, filtering etc. where necessary.

Sensor Calibration (1)

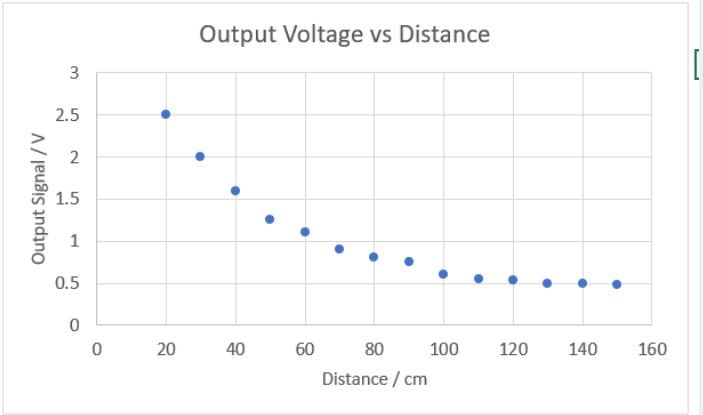
- We need to know the relationship between the measured distance and the sensor output signal.
  - Then, if we get a sensor reading, we will know the distance.
- The process is called calibration.
  - We use a “trusted” measurement device to measure the distance, then read off the sensor output signal.



Sensor Calibration (2)

- By placing the robot at several distances from the wall, we obtain the following data:

Distance / cm	Output Voltage / V
20	2.5
30	2
40	1.6
50	1.25
60	1.1
70	0.9
80	0.8
90	0.75
100	0.6
110	0.55
120	0.53
130	0.5
140	0.49
150	0.48

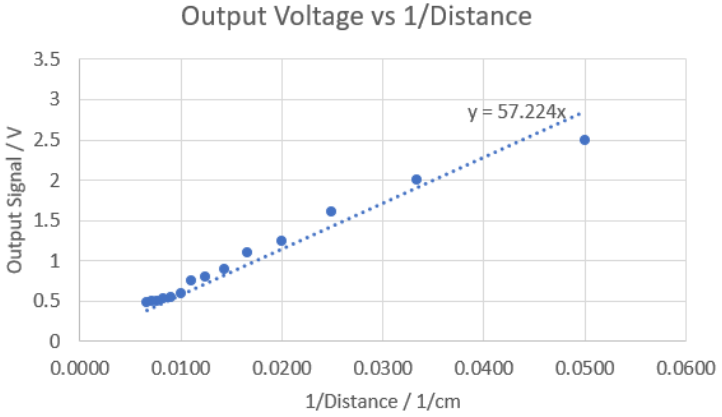


- The relationship between the two is nonlinear.
  - Hard to programmatically determine the distance from the measured output signal.

Sensor Calibration (3)

- However, if we make a plot of output signal vs. 1/distance, we obtain an almost-linear relationship:

Distance / cm	1/Distance / cm^{-1}	Output Voltage / V
20	0.0500	2.5
30	0.0333	2
40	0.0250	1.6
50	0.0200	1.25
60	0.0167	1.1
70	0.0143	0.9
80	0.0125	0.8
90	0.0111	0.75
100	0.0100	0.6
110	0.0091	0.55
120	0.0083	0.53
130	0.0077	0.5
140	0.0071	0.49
150	0.0067	0.48



Sensor Calibration (4)

- Using Excel's curve fitting tool, the linear equation between the two quantities is determined to be:

$$\text{Output Voltage in V} = 57.244 \cdot \frac{1}{\text{Distance in cm}}$$

- Therefore, once the output voltage is read in by Arduino, we can calculate the distance as:

$$\text{Distance in cm} = 57.244 \cdot \frac{1}{\text{Output Voltage in V}}$$

Sensor Calibration (5)

- Note: The sensor's output voltage will go through analog-digital-conversion (ADC) in Arduino.
  - Arduino's ADC is 10 bits.
  - Physical 0V will be converted to a ADC value of 0.
  - Physical 5V will be converted to a ADC value of 1023.
  - Therefore, to get the physical voltage from a given ADC value, the calculation is:

$$\text{Output Voltage in V} = \frac{5 \cdot \text{ADC value}}{1024}$$

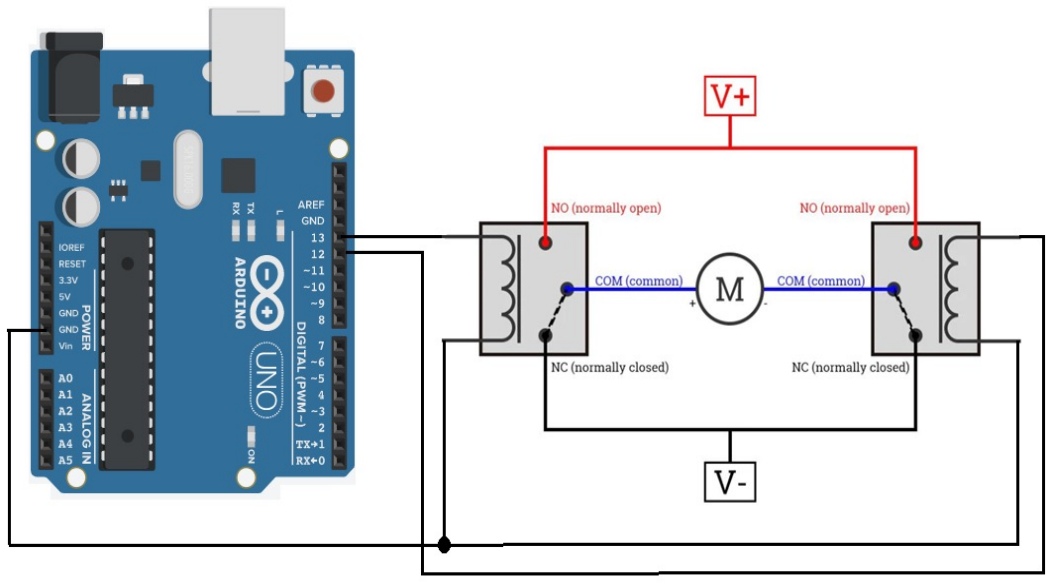
  - In Arduino, we will use the function "map" to do this automatically.

Content

- Introduction
- Distance Measurement Subsystem
- Motor Control Subsystem
- Distance Control Logic and Code
- Experimental Results
- Conclusion

Motor Control Circuit (1)

- To control the distance of the robot from the wall, the circuit needs to have the capability of rotating the motors (connected to the wheels) in both directions.
  - To this means, we have designed a circuit consisting of two relays.



[3] The Pi Hut, "Controlling Motors with Relays", [Online]. https://thepihut.com/blogs/raspberry-pi-tutorials/controlling-motors-with-relays (accessed 5 Sep 2022)

Motor Control Circuit (2)

- How it works:
  - When the input signal to the relay is LOW or 0V, the switch is in the normally open (NO) position.
  - When the input signal to the relay is HIGH or 5V, the switch switches to the normally closed (NC) position.
  - Depending on the digital output signals from pin 12 and 13, we can command the motors in the following ways:

Pin 12 Output	**Pin 13 Output**	**Motor Status**
LOW (0V)	LOW (0V)	Stop
HIGH (5V)	LOW (0V)	Clockwise
LOW (0V)	HIGH (5V)	Anti-Clockwise
HIGH (5V)	HIGH (5V)	Stop

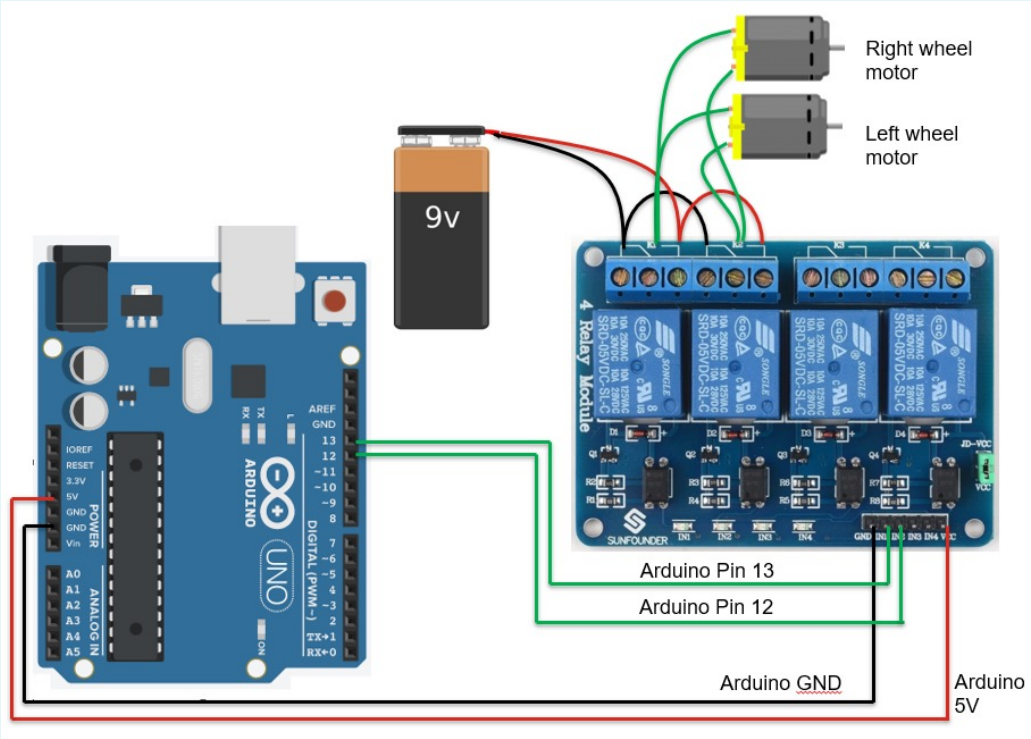
Motor Control Circuit (3)

- The relay is of brand SunFounder, model number SU-TS0001 which requires 5V to turn on [4],
  - Suitable to be operated using Arduino.

[4] Little Bird Electronics, “4 Channel 5V Relay Shield Module [SU-TS0011],” [Online]. https://littlebirdelectronics.com.au/products/4-channel-5v-relay-shield-module (accessed 5 Sep 2022)

Motor Control Circuit (4)

- The physical connection is as follows:

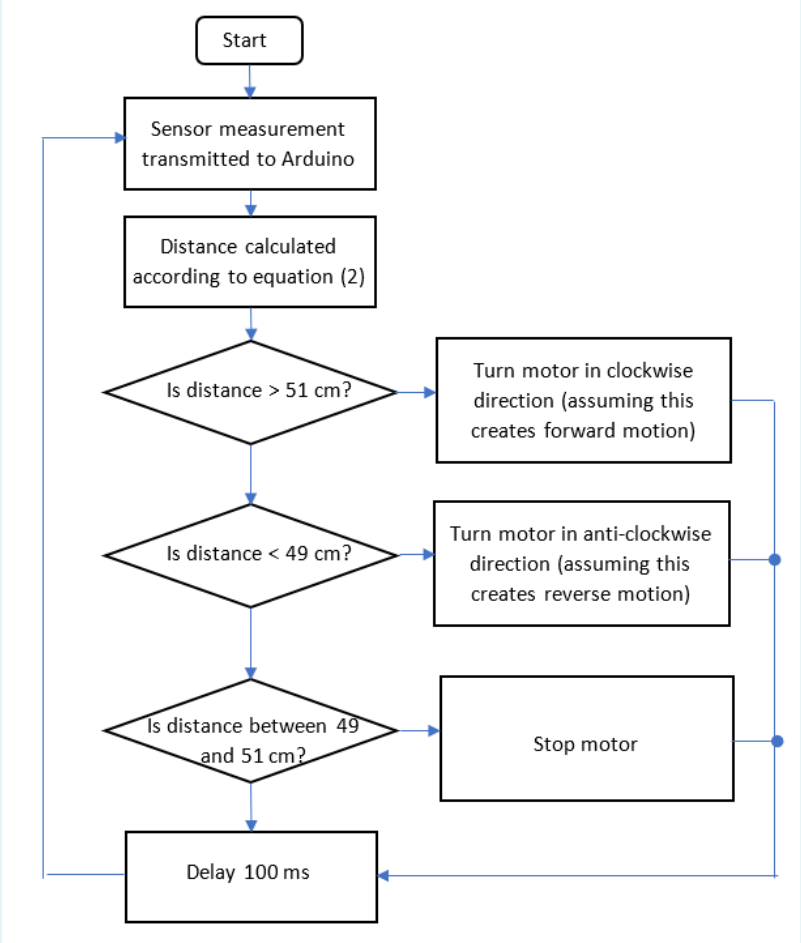


Content

- Introduction
- Distance Measurement Subsystem
- Motor Control Subsystem
- Distance Control Logic and Code
- Experimental Results
- Conclusion

Motor Control Logic

- Flowchart:
- We have opted to achieve a distance between 49 cm and 51 cm
  - as it is quite difficult to reach exactly 50 cm in front of the wall, due to the fast motor speed.



Arduino Code - Setup

```
int left_relay = 12;
int right_relay = 13;
int sensor_pin = A0;
int sensor_reading = 0;
float signal_voltage = 0.0;
float distance_cm = 0.0;

void setup()
{
  // pinMode relay pins as outputs
  pinMode(left_relay, OUTPUT);
  pinMode(right_relay, OUTPUT);

  // make sure motor stops at the beginning
  digitalWrite(left_relay, HIGH);
  digitalWrite(right_relay, LOW);
}
```

Arduino Code - Loop

```
void loop()
{
  // read sensor signal
  sensor_reading = analogRead(sensor_pin);

  // mapping from ADC value back to actual voltage
  signal_voltage = map(sensor_reading, 0, 1023, 0, 5);

  // calculate distance in cm according to calibration
  distance_cm = 57.244 / signal_voltage;

  // if distance > 51, turn motor clockwise
  if (distance_cm > 51)
  {
    digitalWrite(left_relay, HIGH);
    digitalWrite(right_relay, LOW);
  }
```

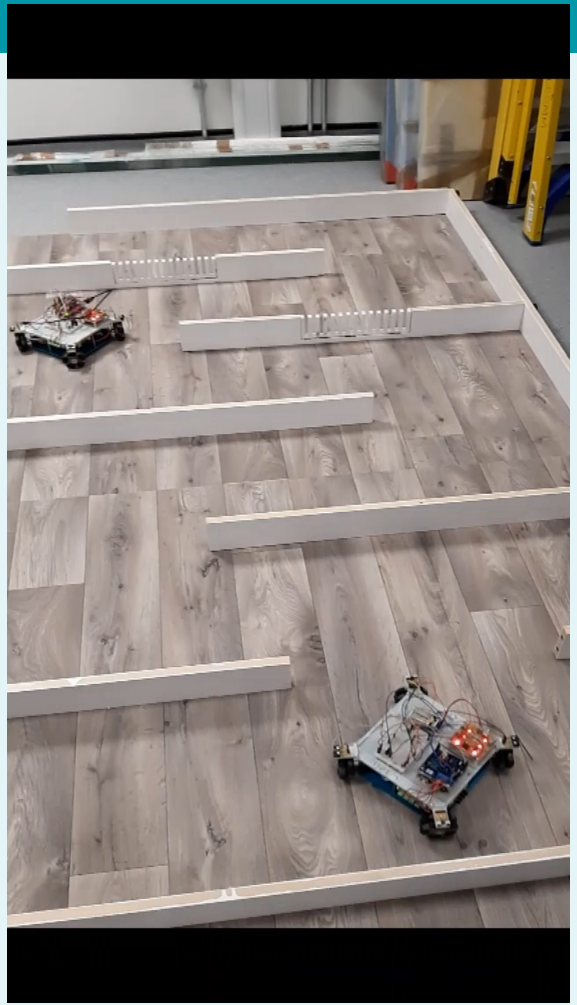
```
  // else if distance < 49, turn motor counter-clockwise
  else if (distance_cm < 49)
  {
    digitalWrite(left_relay, LOW);
    digitalWrite(right_relay, HIGH);
  }

  // else, stop motor
  else
  {
    digitalWrite(left_relay, LOW);
    digitalWrite(right_relay, LOW);
  }

  // delay 100 ms
  delay(100);
}
```

Content

- Introduction
- Distance Measurement Subsystem
- Motor Control Subsystem
- Distance Control Logic and Code
- Experimental Results
- Conclusion

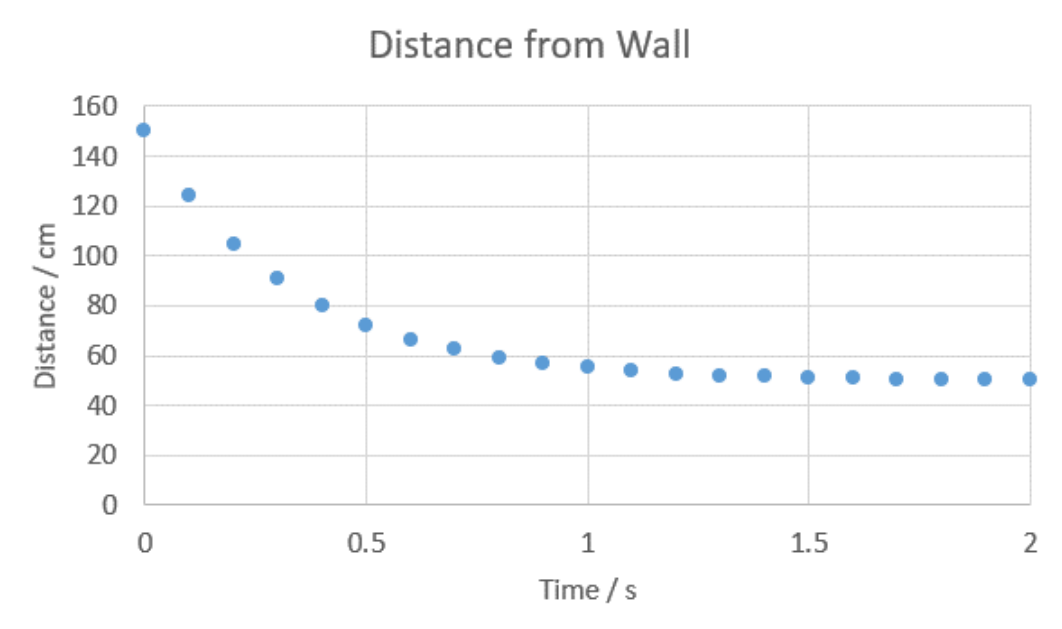
Experiments

- We ran an experiment, and here is a video of the robot movement:
  - Ignore the left/right motion.
  - The forward motion indeed stopped at a desired distance from the wall!



Experimental Data

- We also logged the distance data during the forward motion, and it is as follows:



Content

- Introduction
- Distance Measurement Subsystem
- Motor Control Subsystem
- Distance Control Logic and Code
- Experimental Results
- Conclusion

Conclusion

- In this presentation, we have shown the design process for a distance control system.
  - How to design circuit for sensor, and how to calibrate sensor.
  - How to design circuit for actuator.
  - How to design the control logic.
  - How to translate the control logic into Arduino code.
- We have also done the experiments, which shows that the robot was able to stop at the desired distance from the wall.

Thank you for your attention!

Any questions?