REPORT

PERSONAL PROJECT

SUNFLOWER ROBOT

AUTHOR: NGUYEN TUAN VINH
GITHUB: SUNFLOWER ROBOT ON GITHUB

HO CHI MINH CITY, 07/2024

TABLE OF CONTENTS

LIST OF TABLES

LIST OF IMAGES

Chapter 1

Overview

Overview of the project

Sunflower Robot is a robot based on the behavior of sunflowers in nature. It will always orient its face in the direction opposite to that of the user. The robot uses Computer Vison technology to detect and track facial movements on camera or webcam and controls servos to move according to those position parameters.

Chapter 2

Components

All softwares, library and components which I used for implementing the project.

2.1 Softwares

2.1.1 Integrated development environment (IDE)

I used Visual Studio Code for development.

Visual Studio Code

Figure 2.1: Visual Studio Code Logo

On VS Code, I installed PlatformIO for scripting and upload code to the Arduino board.

Figure 2.2: PLatformIO logo

2.1.2 Languages

- C++
- Python

2.1.3 Frameworks

Arduino framework

2.1.4 Libraries

On PLatformIO, I used Servo library by Michael Margolis at version 1.2.1 for communicate easyly with servos.

On Python, I used 2 library:

- pyserial 3.5 for sending face location to serial.
- OpenCV 4.10.0 for detecting human face.

2.2 Hardwares

2.2.1 Arduino UNO Board

The Arduino UNO is one of the most popular microcontroller boards in the Arduino family. It's designed for beginners and hobbyists to learn electronics and programming. The board is open-source and can be used for a variety of DIY projects and prototypes.

Figure 2.3: Arduino UNO and Adapter

Key features:

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Input Voltage: 7-12V (recommended)
- Digital I/O Pins: 14 (of which 6 can provide PWM output)
- Analog Input Pins: 6
- DC Current per I/O Pin: 20 mA
- Flash Memory: 32 KB (ATmega328P) of which 0.5 KB used by the bootloader
- SRAM: 2 KB (ATmega328P)
- EEPROM: 1 KB (ATmega328P)
- Clock Speed: 16 MHz
- USB Connection: Type B

- Dimensions: 68.6 mm x 53.4 mm

Pinout:

- Digital Pins: Used for general-purpose input/output (GPIO).
- Pins 0-13.
- PWM Pins: Digital pins 3, 5, 6, 9, 10, 11 can be used for PWM output.
- Analog Pins: Pins A0-A5 can be used to read analog signals.
- Power Pins: Includes 3.3V, 5V, GND, and Vin.
- Special Pins:
  - Serial: 0 (RX), 1 (TX) for serial communication
  - External Interrupts: 2 and 3
  - SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK)
  - I2C: A4 (SDA), A5 (SCL)

You can view more detail about Arduino UNO's datasheet in here.

2.2.2 Servo SG90

The SG90 is a micro servo motor that is commonly used for various hobbyist applications due to its compact size, ease of use, and cost-effectiveness. It is capable of rotating 180 degrees (90 degrees in each direction) and can be controlled using a PWM (Pulse Width Modulation) signal.

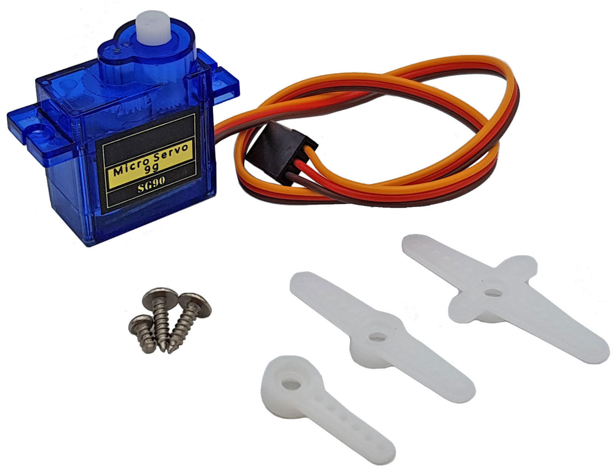

Figure 2.4: Servo SG90 180°

Key Features

- Type: Micro servo
- Weight: 9 grams

- Dimensions: 22.2 mm x 11.8 mm x 31 mm
- Operating Voltage: 4.8V to 6.0V
- Stall Torque:
  - 1.8 kg/cm (at 4.8V)
  - 2.2 kg/cm (at 6.0V)
- Operating Speed: 0.1 s/60 degrees (at 4.8V)
- Temperature Range: -30 to +60 degrees Celsius
- Control System: PWM
- Connector Wire Length: 25 cm
- Gear Type: Plastic

Pinout and Wiring
The SG90 has three wires:

1. Orange (Control Signal): Connect to the PWM signal pin from the microcontroller (e.g., Arduino).
2. Red (Power): Connect to the positive power supply (4.8V to 6.0V).
3. Brown (Ground): Connect to the ground of the power supply and microcontroller.

Control
The SG90 is controlled by sending a PWM signal with specific pulse widths:

- 0 degrees: 1 ms pulse width
- 90 degrees: 1.5 ms pulse width
- 180 degrees: 2 ms pulse width

The frequency of the PWM signal is typically 50 Hz (20 ms period).
You can view more detail about the Servo's data sheet in here.

Chapter 3

Implementation Plan

You can visit the project's plan on Jira by clicking here. Be sure you have a Jira account and logged in already.

3.1 Project's tasks

With the goal of completing the project in 14 days, I have planned the implementation as shown in the table below.

ID	Task name	Duration (day)
SFR-1	Researching information and knowledge	1 (June 22, 2024)
SFR-2	Selecting, shopping and testing hardware	1 (June 22, 2024)
SFR-3	Designing Robot model	2 (June 23, 2024 → June 24,2024)
SFR-4	Designing components and communication diagrams	2 (June 23, 2024 → June 24,2024)
SFR-5	Creating Project and configuration	3 (June 22, 2024 → June 24,2024)
SFR-6	Hardware implementation	4 (June 25, 2024 → June 28,2024)
SFR-7	Software Implementation	3 (June 29, 2024 → July 1,2024)
SFR-8	Testing and fix bugs	2 (July 2, 2024 → July 3,2024)
SFR-9	Writing Report	4 (June 22, 2024 → July 4,2024)
SFR-10	Recording demo video	1 (July 5, 2024)

Table 3.1: All project tasks

3.2 Project timeline

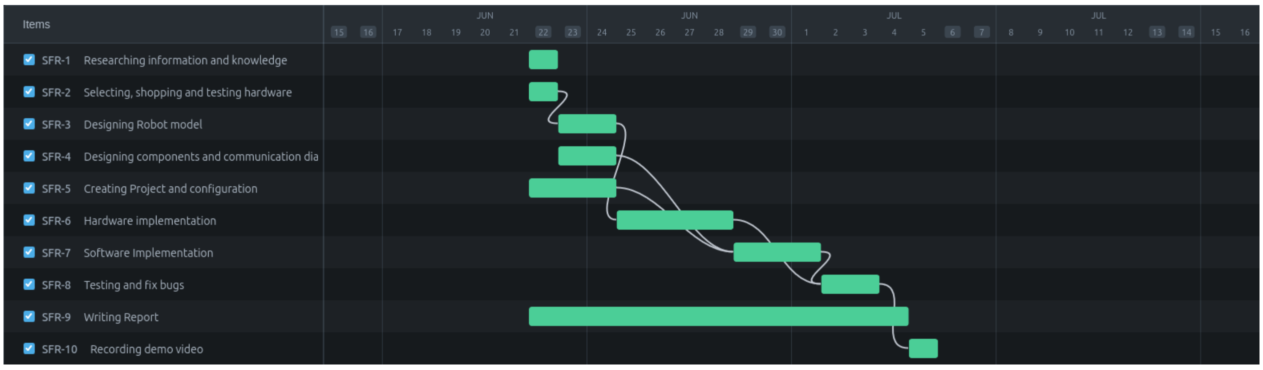

Figure 3.1: Project timeline

Chapter 4

Scripts

Source Code for this project. You can visit project's repository on github to see detail. In this repository, I created 2 branch:

- **master**: for window user,
- **linux**: for linux user.

4.1 Arduino Programming Scripts

In **main.cpp**:

```cpp
#include <Arduino.h>
#include <Servo.h>

#define MIN_ANGLE 0
#define MAX_ANGLE_SERVO1 80
#define MAX_ANGLE_SERVO2 180

#define FRAME_HEIGHT 480
#define FRAME_WIDTH 640

static uint8_t x_angle = MIN_ANGLE;
static uint8_t y_angle = MIN_ANGLE;

Servo myservo1; //rotate range = [0,80] for up-down
Servo myservo2; //rotate range = [0,180] for left-right

String faceLocation;

void setup() {
  Serial.begin(9600);

  myservo1.attach(10);
  myservo2.attach(9);

  myservo1.write(x_angle);
  myservo2.write(y_angle);

}

void loop() {

  if (Serial.available())
  {
    faceLocation = Serial.readStringUntil('\r');
    // Serial.println("faceLocation");
    int x_location = faceLocation.substring(0, faceLocation.indexOf(',')).
   toInt();
    int y_location = faceLocation.substring(faceLocation.indexOf(',') + 1)
   .toInt();

    y_angle = map(x_location, 0, FRAME_WIDTH, MAX_ANGLE_SERVO2, MIN_ANGLE)
   ;
```

```
    x_angle = map(y_location, 0, FRAME_HEIGHT, MIN_ANGLE, MAX_ANGLE_SERVO1
   );
    // Serial.println(x_axis);
    if (x_angle >=0 && x_angle <= MAX_ANGLE_SERVO1)
    {
      myservo1.write(x_angle);

    }

    if(y_angle >= 0 && y_angle <= MAX_ANGLE_SERVO2){
      myservo2.write(y_angle);
    }

  }

}
```

Program 4.1: Programming script for Arduino

4.2 Face Detection Scripts Using Python

In **FaceDetection.py**:

```

import serial
import cv2 as cv
# import time
from serial.tools import list_ports

# ports = list(list_ports.comports())
# for port in ports:
#     print(port.device)

portName = "/dev/ttyACM0"

arduinoData = serial.Serial(portName, 9600)

print(cv.__version__)

def send_coordinates_to_arduino(x, y, w, h):
    # Convert the coordinates to a string and send it to Arduino
    coordinates = f"{x+w/2},{y+h/2}\r"
    arduinoData.write(coordinates.encode())
    print(f"X{x+w/2}Y{y+h/2}\n")

def FaceDetection():
    # x_rotate = 0
```

```python
    # y_rotate = 0

    capture = cv.VideoCapture(0, cv.CAP_V4L2) #Change the number for the camera that you are using, 0 is for the internal laptop camera, 1 is for an external webcam
    face_cascade = cv.CascadeClassifier(cv.data.haarcascades + 'haarcascade_frontalface_default.xml')
    while True:
        isTrue, frame = capture.read()
        # print(len(frame[1]))
        gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.05, 8, minSize=(120,120))

        for (x,y,w,h) in faces:
            cv.rectangle(frame, (x,y), (x+w, y+h), (0,255,0), 5)

            print(f"x={x}, y={y}, w={w}, h={h}")
            send_coordinates_to_arduino(x, y, w ,h)

        cv.imshow("Video", frame)

        if cv.waitKey(20) & 0xFF == ord('d'):
            break

        # send_coordinates_to_arduino(x_rotate, y_rotate ,10 ,10)
        # x_rotate += 2
        # y_rotate += 2

        # if(x_rotate > 80):
        #     x_rotate = 0

        # if(y_rotate > 180):
        #     y_rotate = 0

        # time.sleep(1)
    capture.release()
    cv.destroyAllWindows()


if __name__ == '__main__':
    FaceDetection()
```

Program 4.2: Programming Script for Face Detection using Python

Chapter 5

Demo

You can visit the Project's repository on github to see more code's detail.
You also watch final result video on my youtube channel by clicking here.

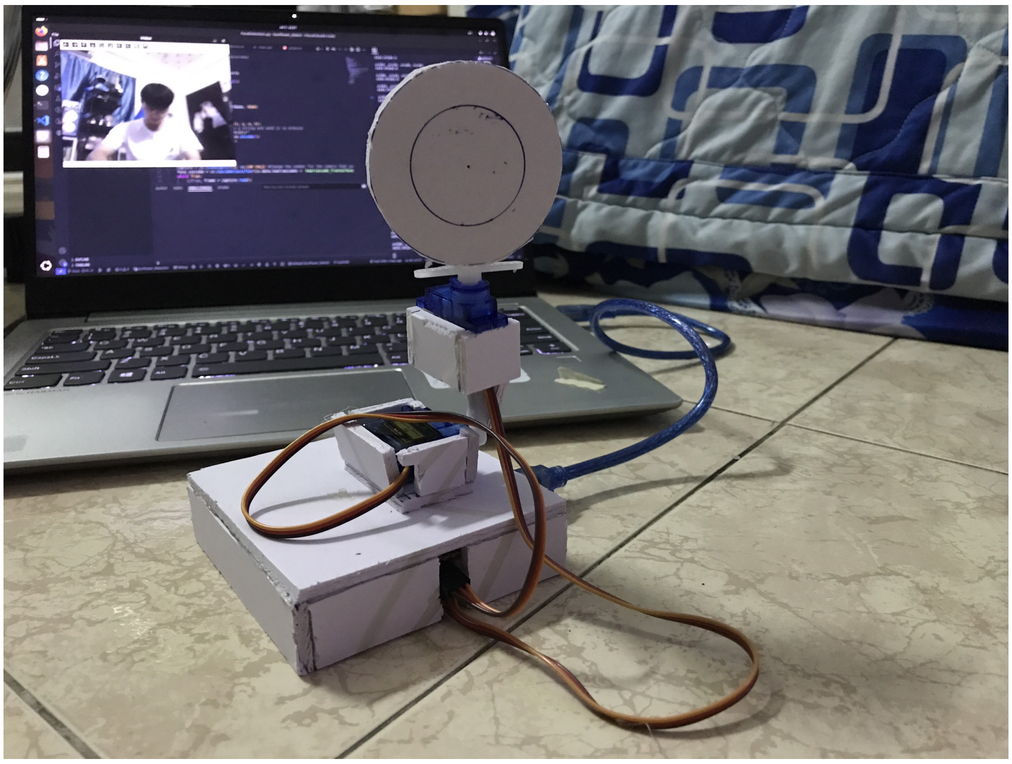

Figure 5.1: Sunflower robot

Thank you for paying your time to reading this Sunflower Robot's report.

End.