ABA AIOT PVT LTD
Pratik Rohane

Face ROS

OBJECTIVE

1. Control the face at different positions
2. Control the face movements

REQUIREMENT

HARDWARE REQUIRED

1. HS-755MG Servo Motors
2. Arduino UNO

SOFTWARE REQUIRED

1. Arduino IDE
2. VS Code

APPLICATION REQUIRED

1. IoTMQTTPanal

EXPERIMENT SETUP

Control Face at Different Position

OVERVIEW

All motors are initially fixed at 100 Degrees, in this experiment we are trying to move a motor to a different position with a span of 60 Degree on both sides. Input to set the position will receive from the IoTMQTTPanel Dashboard into ROS MQTT python code and through the serial communication with Arduino motors will control.

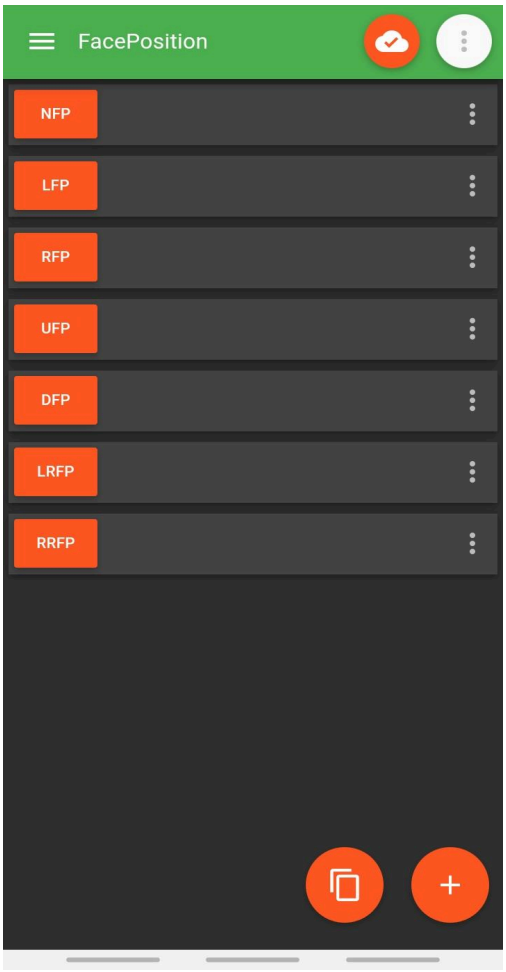
PROCEDURE

Positions

- Normal face position
- Left face position
- Right face position
- Up face position
- Down face position
- Left roll face position
- Right roll face position

Creating MQTT Dashboard

To send the command to achieve each position we have to create the MQTT dashboard in the application.



After clicking the button on the panel the payload message will be received on the subscriber side.

Position	**Panel Name**	**Payload**
Normal Face Position	NFP	NFP
Left Face Position	LFP	LFP
Right Face Position	RFP	RFP
Up Face Position	UPF	UPF
Down Face Position	DFP	DFP
Left Roll Face Position	LRFP	LRFP
Right Roll Face Position	RRFP	RRFP

Creating CSV File

After receiving the payload message on the MQTT subscriber, it will check for the respective angle values for each motor in the CSV file. So we have to specify the angle position for each servo motor for the respective payload.

Position	S1	S2	S3
NFP	100	100	100
RFP	160	100	100
LFP	40	100	100
UFP	100	160	100
DFP	100	40	100
RLFP	100	100	160
RRFP	100	100	40

Code

ROS Python Code

```
#!/usr/bin/env python3

"""ROS Facial Orientation"""

import rospy
from std_msgs.msg import Int16MultiArray
import paho.mqtt.client as mqtt
import pandas as pd

broker = "localhost"
port = 1883
topic = "faceposition"

dataFile = ('/home/dhruya/catkin_ws/src/servo_control/csv/servo_positioning
.csv')
dataFrame = pd.read_csv(dataFile)


def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
    if rc == 0:
        print("Connected to MQTT Broker!")
    else:
        print("Failed to connect, return code %d\n", rc)
```

```
    client.subscribe(topic)

def on_message(client, userdata, msg):
    dataMsg = msg.payload.decode()
    rospy.loginfo(dataMsg)
    #Creating Data to Publish
    index = dataFrame.index[dataFrame["Position"]== dataMsg].tolist()[0]
    s1 = dataFrame.S1[index]
    s2 = dataFrame.S2[index]
    s3 = dataFrame.S3[index]
    Data.data = [s1, s2, s3]

    rospy.loginfo(Data)
    pub.publish(Data)


def MQTTSub():
    rospy.init_node('facialOrientation', anonymous=True)
    rate = rospy.Rate(1)

    client = mqtt.Client()
    client.connect(broker, port, 60)

    client.on_connect = on_connect
    client.on_message = on_message

    client.loop_forever()

if __name__ == '__main__':
    pub = rospy.Publisher('servoControl', Int16MultiArray, queue_size=1)

    Data = Int16MultiArray()
    Data.data = []

    try:
        MQTTSub()
    except rospy.ROSInterruptException:
        client.disconnect()
        pass
```

Arduino Code

```
#include <ros.h>
#include <std_msgs/Int16MultiArray.h>
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>

Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();

#define MIN_PULSE 585     //500
#define MAX_PULSE 2465    //2500
#define SERVO_FREQ 50

ros::NodeHandle  nh;
std_msgs:: Int16MultiArray data;

int arr[3] = {100, 100, 100};
int s1, s2, s3;


void setServo(uint8_t pwm_pin, int angle) {
  int pulse_wide;
  int analog_value;

  pulse_wide = map(angle, 0, 200, MIN_PULSE, MAX_PULSE);
  analog_value = int(float(pulse_wide) / 1000000 * SERVO_FREQ * 4096);

  pwm.setPWM(pwm_pin, 0, analog_value);
}


void messageCb( const std_msgs::Int16MultiArray& servoData) {
  for (int i = 0; i < 3; i++)arr[i] = servoData.data[i];
}


ros::Subscriber<std_msgs::Int16MultiArray> sub("servoControl", &messageCb
);


void setup()
{
  nh.initNode();
```

```
  nh.subscribe(sub);
  pwm.begin();
  pwm.setPWMFreq(SERVO_FREQ);
}


void loop()
{
  nh.spinOnce();

  s1 = arr[0], s2 = arr[1], s3 = arr[2];
  setServo(0, s1);
  setServo(1, s2);
  setServo(2, s3);

  delay(10);
}
```

Control the Face Movement

OVERVIEW

In this experiment, the motor will move from one position to another position. Range of the angle defined in the code to move a motor while iterating in it.

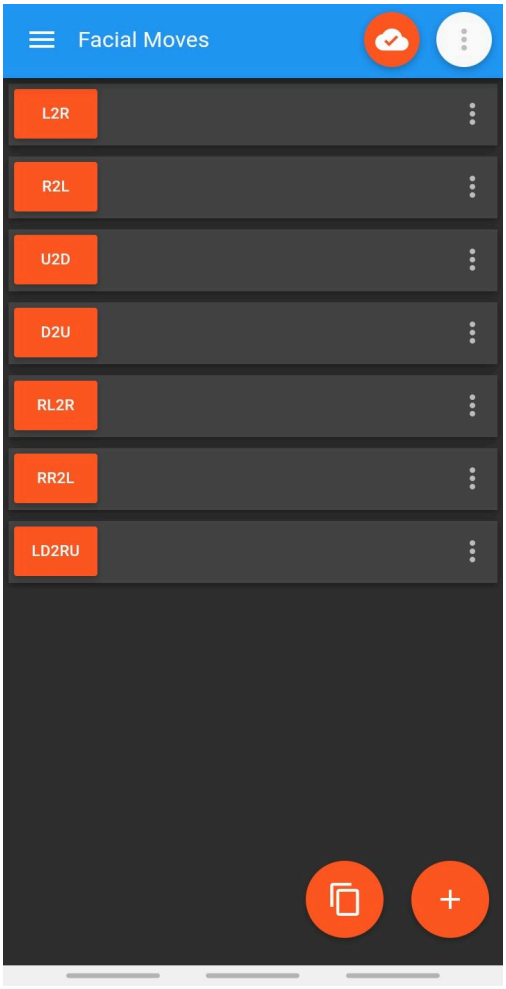
PROCEDURE

Movements

- Left to Right
- Right to Left
- Up to Down
- Down to Up
- Roll Left to Right
- Roll Right to Left
- Left Down to Right Up

Creating MQTT Dashboard

To send the command to achieve each position we have to create the MQTT dashboard in the application.



After clicking the button on the panel the payload message will be received on the subscriber side.

Moves	**Panel Name**	**Payload**
Left to Right	L2R	L2R
Right to Left	R2L	R2L
Up to Down	U2D	U2D
Down to Up	D2U	D2U
Roll Left to Right	RL2R	RL2R
Roll Right to Left	RR2L	RR2L
Diagonal	LD2RU	LD2RU

Functions

For each move, the function is defined in the code, after receiving the payload the respective function will call and data will send to Arduino to move a motor. To control the speed of the motor, the range of angles going to iterate in the function with a specific intervial.

Code

ROS Python Code

```
#!/usr/bin/env python3

""" Facial Moves"""

import rospy
from std_msgs.msg import Int16MultiArray
import time

def moveLeft2Right():
    for i in range(160, 40, -1):
        s1 = 100
        s2 = 100
        s3 = i
        Data.data = [s1, s2, s3]
        rospy.loginfo(Data)
        pub.publish(Data)
        time.sleep(0.015)

def moveRight2Left():
    for i in range(40, 160):
        s1 = 100
        s2 = 100
        s3 = i
        Data.data = [s1, s2, s3]
        rospy.loginfo(Data)
        pub.publish(Data)
        time.sleep(0.01)

def moveUp2Down():
    for i in range(40,161):
        s1 = 100
        s2 = i
        s3 = 100
```

```
        Data.data = [s1, s2, s3]
        rospy.loginfo(Data)
        pub.publish(Data)
        time.sleep(0.015)

def moveDown2Up():
    for i in range(160,39,-1):
        s1 = 100
        s2 = i
        s3 = 100
        Data.data = [s1, s2, s3]
        rospy.loginfo(Data)
        pub.publish(Data)
        time.sleep(0.015)

def moveRollLeft2Right():
    for i in range(40,161):
        s1 = i
        s2 = 100
        s3 = 100
        Data.data = [s1, s2, s3]
        rospy.loginfo(Data)
        pub.publish(Data)
        time.sleep(0.01)

def moveDiagonal():
    for i in range(160,39,-1):
        s1 = 100
        s2 = i
        s3 = i
        Data.data = [s1, s2, s3]
        rospy.loginfo(Data)
        pub.publish(Data)
        time.sleep(0.015)

def arduinoPub():
    rospy.init_node('node_ros', anonymous=True)
    //MQTT


if __name__ == '__main__':
    pub = rospy.Publisher('servoControl', Int16MultiArray, queue_size=1)
```

```
    Data = Int16MultiArray()
    Data.data = []

    try:
        arduinoPub()
    except rospy.ROSInterruptException:
        pass
```

Arduino Code

The same code can be used for this also.

CONCLUSION

Two options to manipulate with CSV file in python one to use built-in module csv or pandas library which has to install. But adding dependencies in the project is not a good idea. So if we want to work on a small data set csv module is the best option.

RESULT

Pandas module gives better performance on large data in CSV files and helps to perform the operation on data.
MQTT and ROS serial communion working well for this experiment.