ABA AIOT PVT LTD
Pratik Rohane

Eyelid ROS

OBJECTIVE

1. Control the eyelid at a different position
2. Control the eyelid movement

REQUIREMENT

HARDWARE REQUIRED

1. 9g Servo Motors
2. Arduino UNO

SOFTWARE REQUIRED

1. Arduino IDE
2. VS Code

APPLICATION REQUIRED

1. IoTMQTTPanal

EXPERIMENT SETUP

Control the Eyelid at Different Position

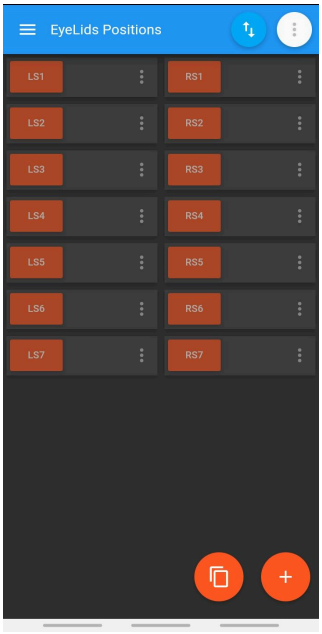
PROCEDURE

Positions

- Stage 1 at 20 Degree
- Stage 2 at 40 Degree
- Stage 3 at 60 Degree
- Stage 4 at 80 Degree
- Stage 5 at 100 Degree
- Stage 6 at 120 Degree
- Stage 7 at 140 Degree

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	Left Eyelid		Right Eyelid	
	Panel Name	Payload	Panel Name	Payload
Stage 1	LS1	LS1	RS1	RS1
Stage 2	LS2	LS2	RS2	RS2
Stage 3	LS3	LS3	RS3	RS3
Stage 4	LS4	LS4	RS4	RS4
Stage 5	LS5	LS5	RS5	RS5
Stage 6	LS6	LS6	RS6	RS6
Stage 7	LS7	LS7	RS7	RS7

Creating CSV File

After receiving the payload message on the MQTT subscriber, it will check for the respective angle values for that motor in the CSV file. For both motor the topics are different.

LE	RE	Angle
LS1	RS1	20
LS2	RS2	40
LS3	RS3	60
LS4	RS4	80
LS5	RS5	100
LS6	RS6	120
LS7	RS7	140

Code

ROS Python Code

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

""" ROS EyeLids """

import rospy
from std_msgs.msg import Int64
import paho.mqtt.client as mqtt

broker = "localhost"
port = 1883
topicRE = "rightE"
topicLE = "leftE"


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(topicRE)
    client.subscribe(topicLE)

def on_message(client, userdata, msg):
    if msg.topic == topicRE:
        dataMsg = msg.payload.decode()
        data.data = int(dataMsg)
        pubR.publish(data.data)
```

```
    if msg.topic == topicLE:
        dataMsg = msg.payload.decode()
        data.data = int(dataMsg)
        pubL.publish(data.data)


def MQTTSub():
    rospy.init_node('eyeLids', 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__':
    pubR = rospy.Publisher('rightEye', Int64, queue_size=1)
    pubL = rospy.Publisher('leftEye', Int64, queue_size=1)

    data = Int64()

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

Arduino Code

```
#include <ros.h>
#include <std_msgs/Int64.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:: Int64 pubdata;

ros::Publisher chatter("chatter", &pubdata);
int RE = 90, LE = 90;

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

  pulse_wide = map(angle, 0, 180, MIN_PULSE, MAX_PULSE);
  analog_value = int(float(pulse_wide) / 1000000 * SERVO_FREQ * 4096);

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


void messageLCb( const std_msgs::Int64& servoData) {
  LE = servoData.data;
}
void messageRCb( const std_msgs::Int64& servoData) {
  RE = servoData.data;
}


ros::Subscriber<std_msgs::Int64> subR("rightEye", &messageRCb );
ros::Subscriber<std_msgs::Int64> subL("leftEye", &messageLCb );


void setup()
{
  nh.initNode();
  nh.subscribe(subR);
  nh.subscribe(subL);
  nh.advertise(chatter);
  pwm.begin();
  pwm.setPWMFreq(SERVO_FREQ);
}


void loop()
{
```

```
  nh.spinOnce();
  pubdata.data = RE;
  chatter.publish(&pubdata);
  setServo(4, RE);
  setServo(5, LE);
  delay(10);
}
```

Control the Eyelid Movement

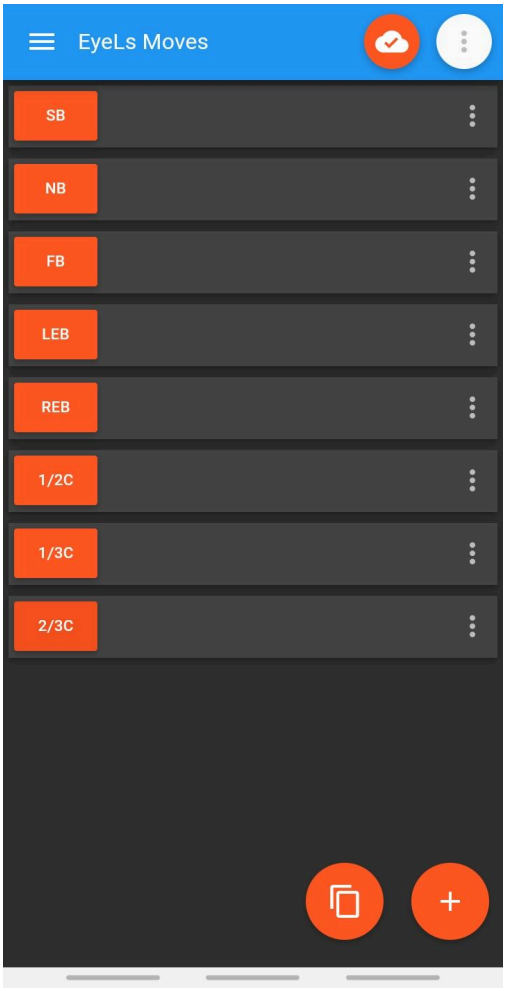
PROCEDURE

Movements

- Slow blink
- Normal blink
- Fast blink
- Halfway blink
- One-third blink
- Two-third blink

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**
Slow Blink	SB	SB
Normal Blink	NB	NB
Fast Blink	FB	FB
Left Eye Blink	LEB	LEB
Right Eye Blink	REB	REB
Halfway Blink	1/2B	1/2B
One-third Blink	1/3B	1/3B

ROS Python Code

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

""" Eyelids Movement"""

import rospy
from std_msgs.msg import Int64
import paho.mqtt.client as mqtt
import time

broker = "localhost"
port = 1883
topic = "eyeMoves"

upperLimit = 0
lowerLimit = 150


def blink(initialAngle, finalAngle, delay):
    for i in range(initialAngle, finalAngle):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
        time.sleep(delay)

    for i in range(finalAngle, initialAngle-1, -1):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
        time.sleep(delay)

def slowBlink():
    for i in range(0, 141):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
        time.sleep(0.010)

    for i in range(140, -1, -1):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
```

```
        time.sleep(0.010)

def normalBlink():
    for i in range(0, 141):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
        time.sleep(0.005)

    for i in range(140, -1, -1):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
        time.sleep(0.005)

def fastBlink():lowerLimit
    for i in range(0, 141):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
        time.sleep(0.001)
    for i in range(140, -1, -1):
        data.data = i
        pubL.publish(data.data)
        pubR.publish(data.data)
        time.sleep(0.001)

def leftEyeBlink():
    for i in range(0, 141):
        data.data = i
        pubL.publish(data.data)
        time.sleep(0.005)

    for i in range(140, -1, -1):
        data.data = i
        pubL.publish(data.data)
        time.sleep(0.005)

def rightEyeBlink():
    for i in range(0, 141):
        data.data = i
        pubR.publish(data.data)
        time.sleep(0.005)
```

```
    for i in range(140, -1, -1):
        data.data = i
        pubR.publish(data.data)
        time.sleep(0.005)

def halfwayClose():
    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 onethirdClose():
    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 twothirdClode():
    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 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()
    if dataMsg == 'SB': blink(upperLimit, lowerLimit, 0.014)
    if dataMsg == 'NB': blink(upperLimit, lowerLimit, 0.007)
    if dataMsg == 'FB': blink(upperLimit, lowerLimit, 0.001)
    if dataMsg == 'LEB': leftEyeBlink()
    if dataMsg == 'REB': rightEyeBlink()
    if dataMsg == '1/2C': blink(upperLimit, int(1*lowerLimit/2), 0.007)
    if dataMsg == '1/3C': blink(upperLimit, int(1*lowerLimit/3), 0.007)
    if dataMsg == '2/3C': blink(upperLimit, int (2*lowerLimit/3), 0.007)



def MQTTSub():
    rospy.init_node('eyeLids', 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__':
    pubR = rospy.Publisher('rightEye', Int64, queue_size=1)
    pubL = rospy.Publisher('leftEye', Int64, queue_size=1)

    data = Int64()

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

Arduino Code

The same code can be used for this also.

CONCLUSION

To control the speed of the Servo motor I am not able to find the optimal way as servo motor speed can’t be controlled other than the iteration in the loop with a certain delay period.

RESULT

Iteration is not a good solution as it does not give smoothness in action.