디지털 영상처리 연구실 연구보고서

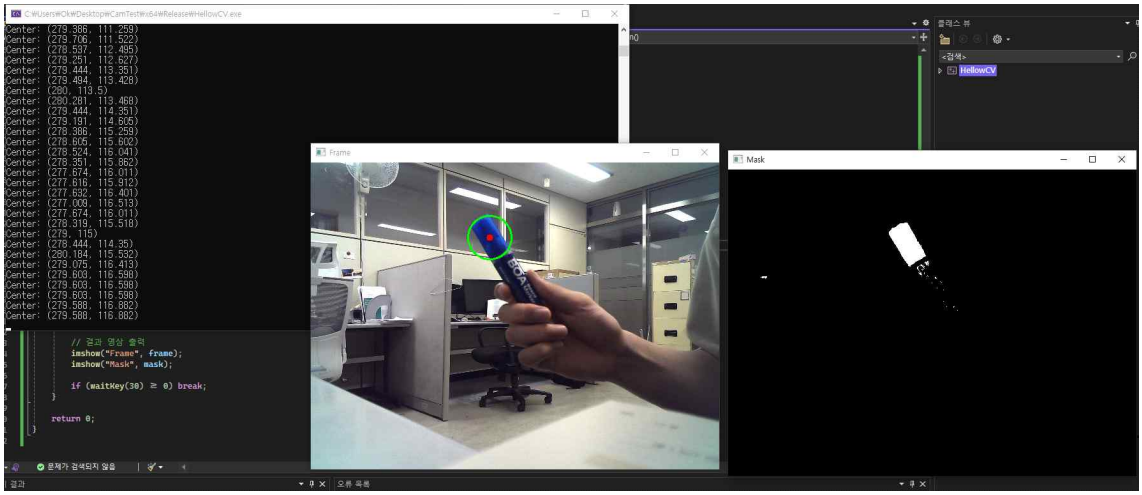
정지우

<메이커톤>

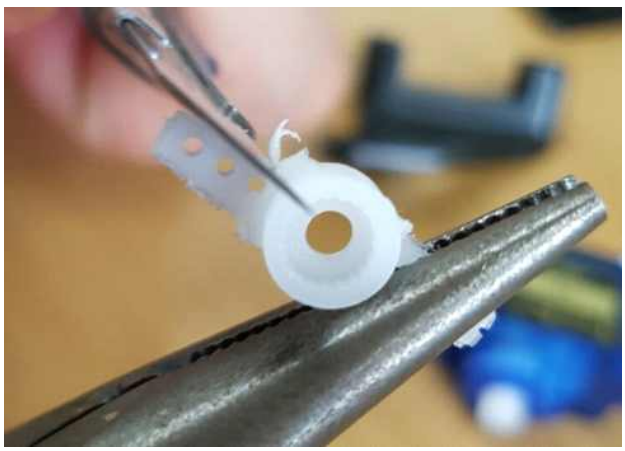
주제: 음성인신 ai기반의 IOT 조명장치

추가할 것: 넓은 작업대의 환경에서 작업하는곳(손)을 자동으로 비춰주는 시스템

첫 번째) 특정물체를 트레킹하면서 좌표를 실시간 반환



두 번째) 서보모터를 활용하여 펜틸트 만들고 구현하기



```
#include <Servo.h>

Servo myservox;  // create Servo object to control a servo
Servo myservoy;  // create Servo object to control a servo
// twelve Servo objects can be created on most boards

int pos = 0;    // variable to store the servo position

void setup() {
  myservox.attach(9);  // attaches the servo on pin 9 to the Servo object
  myservoy.attach(8);  // attaches the servo on pin 9 to the Servo object
}

void loop() {
  for (pos = 80; pos <= 150; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservox.write(pos);              // tell servo to go to position in variable 'pos'
    myservoy.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15 ms for the servo to reach the position
  }
  delay(30);
  for (pos = 150; pos >= 80; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservox.write(pos);              // tell servo to go to position in variable 'pos'
    myservoy.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15 ms for the servo to reach the position
  }
}
```

서보 클래스의 객체생성에서 핀번호 입력후,

write함수에 각도를 넣어주면 제어완료.

+) 주의사항: 서보모터 마다 하드웨어적으로 최대각도가 존재하는데 코드로 억지로 돌리려고 하면 열난다.

세 번째) visual studio와 아두이노의 정보교환

1. 시리얼통신

Arduino and C++ (for Windows)

As I found it pretty hard finding the good information, or an already working code to handle Serial communication on windows based system, I finally made a class that do what is needed for basic Serial Communication, thanks to help of several forumers. Please note that this code might not be completely perfect so I encourage you to make any update needed so that it might become even better.

아두이노 공식 사이트에 방법이 존재하며, 정해진 헤더파일과 소스코드를 작동시키면 **"유선"**으로 정보 교환이 가능.

+) 하지만, 제가 못하는건지 5시간동안 노력하였지만 실패
그래서 이것저것 찾다가 와이파이 모듈로 웹서버를 만들어 통신을 찾음

2. 웹서버

공유기의 ip주소로 웹서버 만들어서 정보의 교환

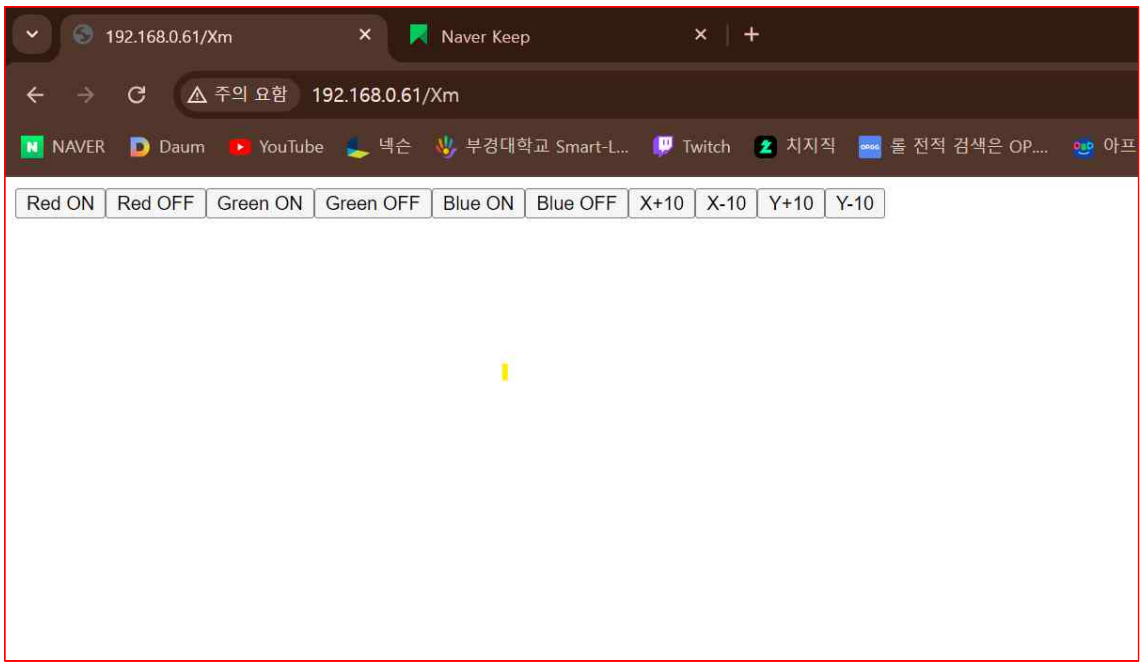
```
#include <SPI.h>
#include <WiFiNINA.h>
#include <Servo.h>

// WiFi network information
char ssid[] = "dip00";        // your network SSID (name)
char pass[] = "dip00805";    // your network password (use for WPA, or use as key for WEP)
int keyIndex = 0;                 // your network key index number (needed only for WEP)

int status = WL_IDLE_STATUS;
WiFiServer server(80);
```

```
client.print("<html><body>");
client.print("<button onclick=\"location.href='/RH'\">Red ON</button>");
client.print("<button onclick=\"location.href='/RL'\">Red OFF</button>");
client.print("<button onclick=\"location.href='/GH'\">Green ON</button>");
client.print("<button onclick=\"location.href='/GL'\">Green OFF</button>");
client.print("<button onclick=\"location.href='/BH'\">Blue ON</button>");
client.print("<button onclick=\"location.href='/BL'\">Blue OFF</button>");
client.print("<button onclick=\"location.href='/Xp'\">X+10</button>");
client.print("<button onclick=\"location.href='/Xm'\">X-10</button>");
client.print("<button onclick=\"location.href='/Yp'\">Y+10</button>");
client.print("<button onclick=\"location.href='/Ym'\">Y-10</button>");
client.print("</body></html>");
client.println();
```

```
14:48:55.525 -> Attempting to connect to Network named: dip00
14:49:09.320 -> SSID: dip00
14:49:09.320 -> IP Address: 192.168.0.61
14:49:09.320 -> signal strength (RSSI):-43 dBm
14:49:09.320 -> To see this page in action, open a browser to http://192.168.0.61
```



이렇게 웹 서버가 생성되고 버튼을 클릭하면 아두이노에서 정보를 받아 처리가능하다.

3. visual studio에서 웹서버 통신

다행히도 CURL이라는 라이브러리를 통해 데이터 전송이 가능하다.

> curl은 데이터 전송과 관련된 프로그램의 빠른 작성을 위해서 사용하는 command line tool이다.
>
> HTTP(:12), FTP(:12), LDAP(:12), TELNET(:12), HTTPS(:12), DICT(:12)와 같은 프로토콜을 지원하며, SSL(:12)을 가지는 각각의 프로토콜 역시 지원한다.
>
> 또한 HTTP기반의 upload, proxies, cookies, user+password 인증을 사용할 수도 있다.

이렇게 웹서버를 지원한다.

해더파일을 #include <curl/curl.h> 를 사용하고,

```
int main() {
    std::string arduinoIP = "192.168.0.21"; // 아두이노의 IP 주소
    std::string command; // 보낼 명령어

    // 예시로 Red LED를 켜고 끄는 명령어 전송
    command = "RH";
    sendDataToArduino(arduinoIP, command);
    std::this_thread::sleep_for(std::chrono::seconds(1));

    command = "RL";
    sendDataToArduino(arduinoIP, command);

    return 0;
}
```

ip를 지정해주고 보낼 정보를 정해주면 된다.

네 번째) 영상처리 결과를 서버를 통해 정보전송 후 아두이노 제어

```
    // 중심 좌표 출력
    cout << "Center: (" << max_center.x << ", " << max_center.y << ")" << endl;

    // 이전 중심 좌표와의 차이 계산
    int diff_x = max_center.x - previous_center.x;
    int diff_y = max_center.y - previous_center.y;

    cout << "Diff: (" << diff_x << ", " << diff_y << ")" << endl;

    // X, Y 축으로 얼마나 떨어졌는지를 기준으로 명령어 전송
    if (diff_x > threshold) {
        sendDataToArduino(arduinoIP, "Xp");
        //std::this_thread::sleep_for(std::chrono::milliseconds(500)); // 0.5초 대기
    }
    else if (diff_x < -threshold) {
        sendDataToArduino(arduinoIP, "Xm");
        // std::this_thread::sleep_for(std::chrono::milliseconds(500)); // 0.5초 대기
    }

    if (diff_y > threshold) {
        sendDataToArduino(arduinoIP, "Yp");
        //std::this_thread::sleep_for(std::chrono::milliseconds(500)); // 0.5초 대기
    }
    else if (diff_y < -threshold) {
        sendDataToArduino(arduinoIP, "Ym");
        //std::this_thread::sleep_for(std::chrono::milliseconds(500)); // 0.5초 대기
    }




    // 현재 중심 좌표를 이전 중심 좌표로 업데이트
    previous_center = max_center;
}
```

영상 프레임의 중심좌표를 찾고,
(트레킹좌표 - 중심좌표)를 하여 이동량을 구해서 일정량 이상이면 특정방향으로 회전는 정보를 서버에 전달한다.

이후 중심좌표는 이전 프레임의 좌표로 치환한다.

구현만 완성되었고 20%짜리 코드이기 때문에
동작방식 알고리즘, 프레임단위, 특이사항 등 고쳐나갈 예정이다.