[Extra Project] Smart Lecture Recording System

Date: 2023-06-25

Name/Student ID: Lee-Kiyoon/ (21800501), Lee_chanKeun(21900575)

Result Video:

DLIP_Extra_DEMO_for_SSL_youtube

DataSet Link: I used pretrained Yolov5s data, therefore refer to Yolov5

Github LInk: DLIP_Extra

Contents

1. Introduction

Existing Lecturer tracking camera(expensive)

I will make it cheaper and efficient using 2 webcams

When a professor goes on a business trip or it is difficult to conduct a class due to personal matters, the professor records the lecture himself and uploads the lecture video to the LMS site. However, unlike other Internet lecture sites, in the case of university professors, cameramen do not take pictures when filming lectures. Therefore, when recording a lecture, there is a disadvantage of having to do troublesome tasks such as manual work, such as changing the angle alone or capturing the handwriting on the blackboard. This system records blackboard lectures very easily by itself and automatically detects the professor's gestures and even captures the handwriting on the blackboard. Through this, it is expected that professors will be able to easily record lectures by themselves. Sure, There is already similar product in website. But, It is too expensive. Therefore, I will make this system using webcams to make it more cheaper and efficient for customers.

2. Requirement

2.1 required hardware and software

2.1.1. Software

- YOLO v5
- Pytorch 1.6.0
- CUDA 11.6
- Python 3.9.12 (py39)
- Arduino IDE
- Fusion 360 (for gears design)

2.1.2. Hardware

- Adafruit Motor Shield v2.3
- 12V DC Motor
- Arduino Uno WiFi
- Webcam x2

2.2 Setup

2.2.1. Software Setup

Anaconda settings

check your cuda version and donwload nvidia driver click here. Refer to installation_guide.

```
# create a conda env name=py39
conda create -n py39 python=3.9.12
conda activate py39
conda install -c anaconda seaborn jupyter
pip install opencv-python

# pytorch with GPU
conda install -c anaconda cudatoolkit==11.3.1 cudnn seaborn jupyter
conda install pytorch=1.10 torchvision torchaudio cudatoolkit=11.3 -c pytorch
pip install opencv-python torchsummary

# Check GPU in Pytorch
conda activate py39
python
import torch
print("cuda" if torch.cuda.is_available() else "cpu"
```

YOLOv5 Installation

Go to YOLOv5 github (https://github.com/ultralytics/yolov5) and download Repository as below. After entering the /`yolov5-master` folder, copy the path address. Then executing Anaconda prompt in administrator mode, execute the code below sequentially.

```
conda activate py39
cd $YOLOv5PATH$ // [ctrl+V] paste the copied yolov5 path
pip install -r requirements.txt
```

Labeling

The DarkLabel 2.4 program was used to generate custom data. Using this program, bounding box labeling was performed directly for each frame of the video to create an image and label dataset. Compared to other labeling programs, labeling work through images is possible, so it is possible to generate a lot of training data quickly. Go to DarkLabel 2.4 and download the DarkLabel 2.4 program below. if it is not available, please download here.

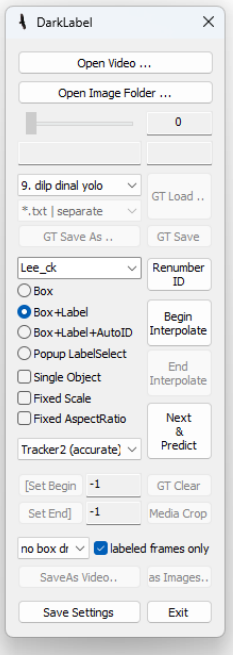
After executing DarkLabel.exe, labeling is performed using the desired image or image.

1. Image file path for using image files
2. Using Darknet yolo labeling method
3. To using your customized labels for model training, the number of data and class name of coco dataset must be changed.
4. Adjust for yml file and add the custom class.

darklabel.yml

```yaml
## Predefined Classes Sets (you can define and add your own classes set here)
my_classes1: ["person", "vehicle", "bicycle", "motorbike", "animal", "tree", "building"]
coco_classes: ["person", "bicycle", "car", "motorcycle", "airplane", "bus", "train", "truck", "boa
voc_classes: ["aeroplane", "bicycle", "bird", "boat", "bottle", "bus", "car", "cat", "chair", "cow
dlip_class: ["Lee_ck", "Lee_ky","Oscilsicope","Red", "Green", "Blue", "Purple", "Gray", "Yellow"]

format9:  # darknet yolo (predefined format)
  fixed_filetype: 1               # if specified as true, save setting isn't changeable in GUI
  data_fmt: [classid, ncx, ncy, nw, nh]
  gt_file_ext: "txt"               # if not specified, default setting is used
  gt_merged: 0                   # if not specified, default setting is used
  delimiter: " "                   # if not spedified, default delimiter(',') is used
  classes_set: "dlip_class"     # if not specified, default setting is used
  name: "dilp dinal yolo"          # if not specified, "[fmt%d] $data_fmt" is used as default format name
```



and then when you train, you have to match yolov5 yaml to the yaml you used in darklabel 2.4.

```yaml
train: ../datasets/DLIP_final/images/training/
val: ../datasets/DLIP_final/images/validation/

# Classes
nc: 9
# class_name = ['Lee_ck', 'Lee_ky', 'Oscilliscope','Red','Green','Blue','Purple','Gray','Yellow']
names: ['0', '1', '2','3', '4', '5', '6', '7', '8']
```

Finally, We used the GPU server to train the datasets. If you input this in cmd, You can train

```
python -m torch.distributed.run –nproc_per_node 1 train.py –batch 8 –epochs 100 –data data.yaml –weights yolov5s.pt –device 0,1
```

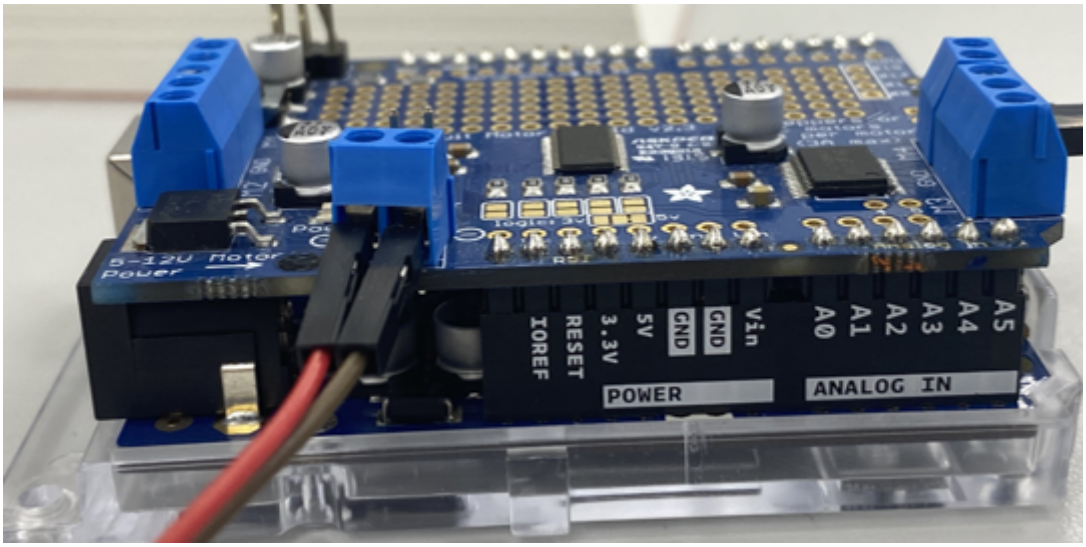
Mediapipe installation

you can install 'mediapipe' library by just writing this command.

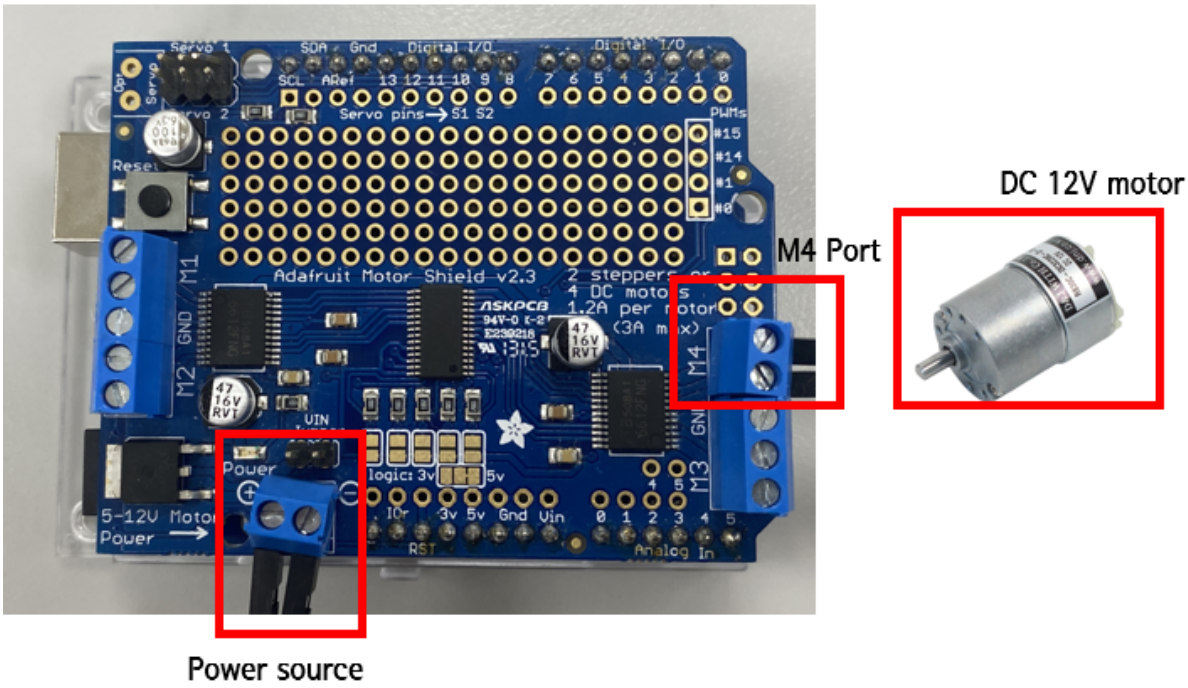
```
pip install mediapipe -opencv
```

2.2.2. Hardware setup

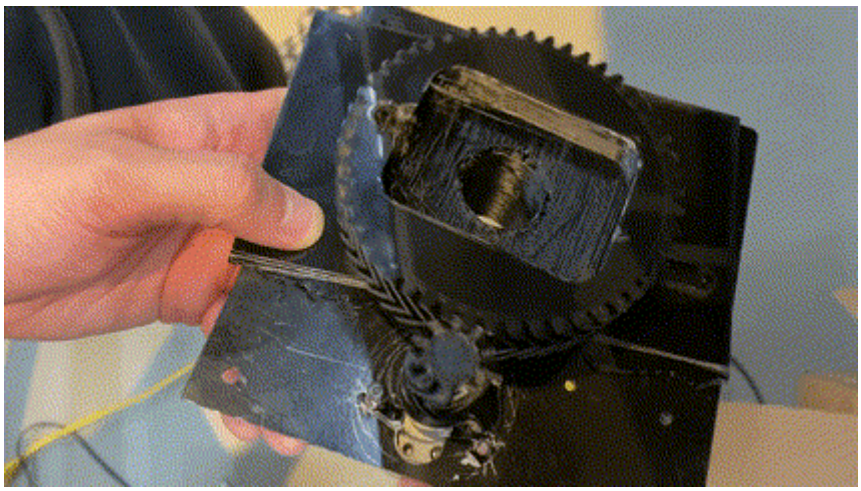
First, we used **Adafruit motorshield v2.3** and **arduino uno wifi**. you can combine motor driver and arduino by just lapping.



We used the 12V DC motor, therefore you need another power source for using this motor. It cannot be used only from arduino power source. Therefore, You need to connect 12V source from power supplier. and then we select the M4 port for motor driving.

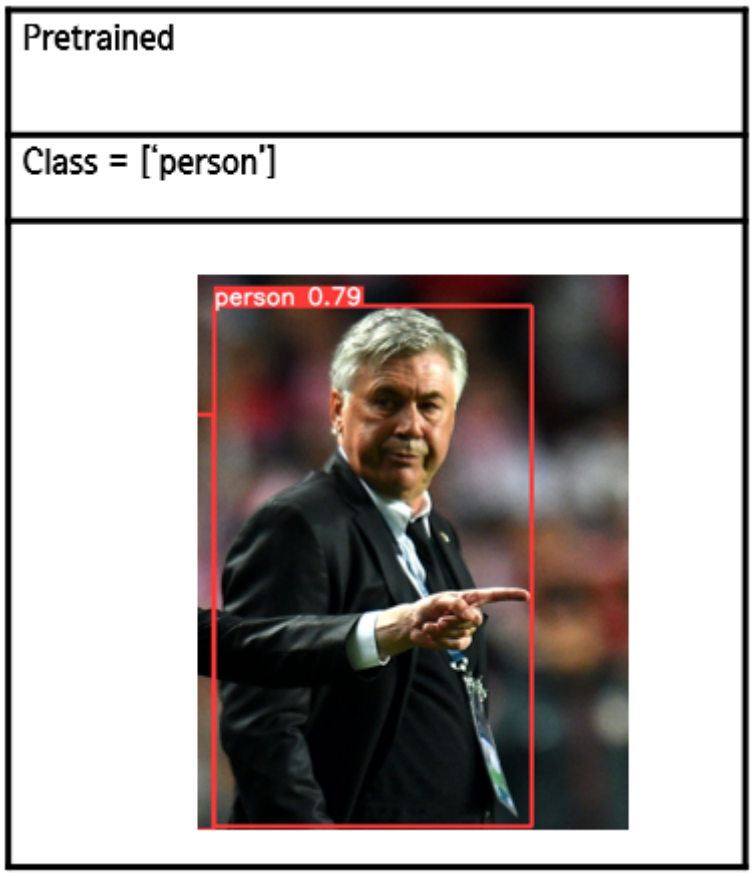


Finally, we designed two gears. You can design two gears by using 'Fusion 360'. Below video is the motor and gear test video.



Finally, Everything is ready for starting this project.

3. Dataset

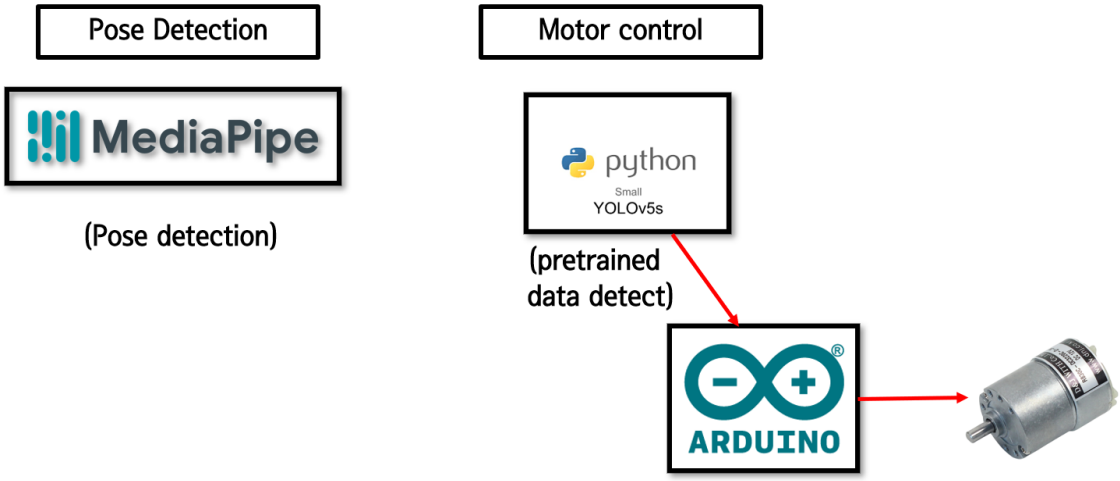


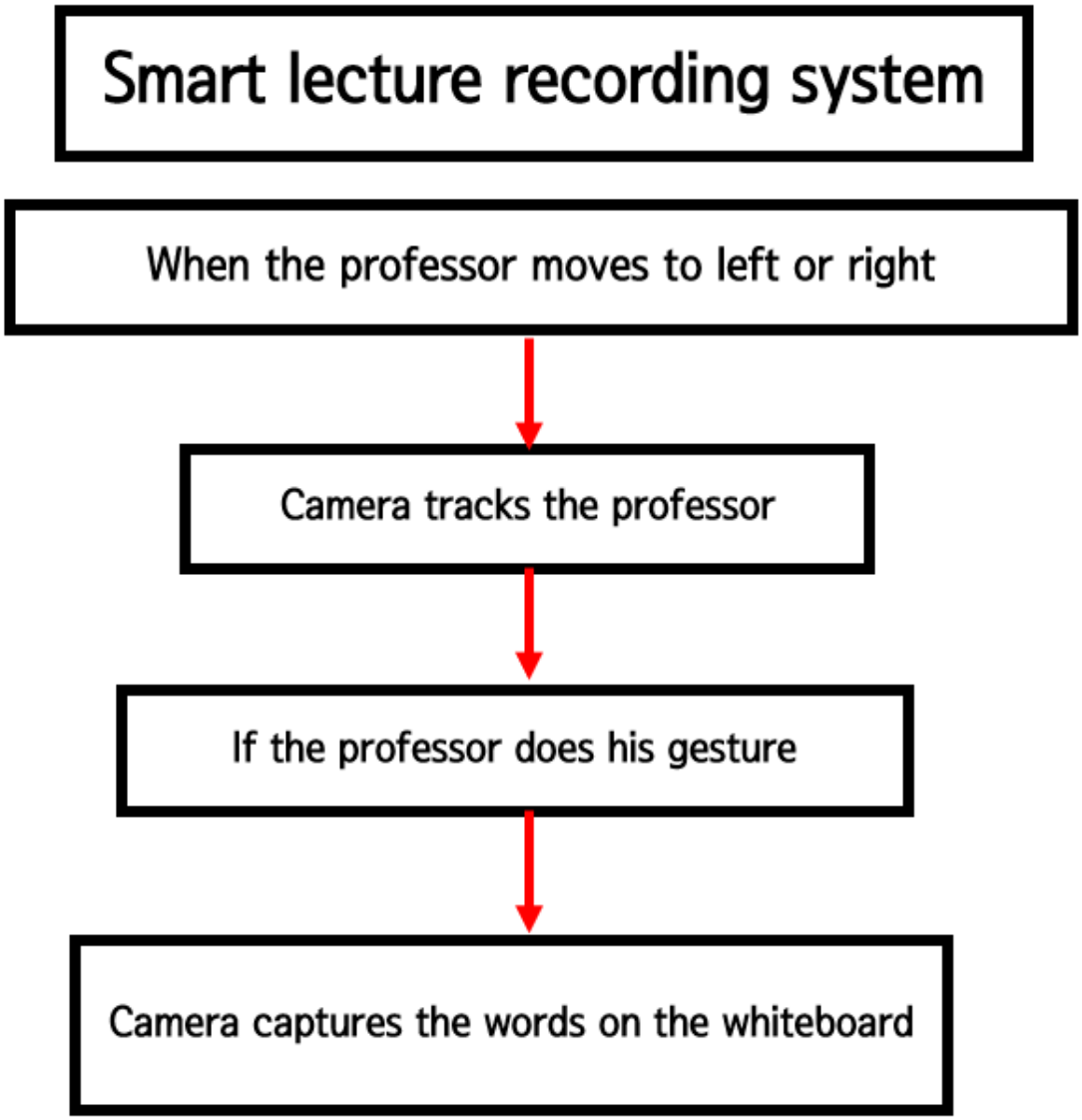
In this project, I used pretrained Yolov5s data, therefore refer to Yolov5 http://naver.me/G5Q9u72g)

4. Flow Chart

First of all, the first feature of the lecture recording assistance system I made was designed so that the camera could recognize the professor and follow the professor in one-take without having to change the camera angle even if the blackboard was horizontally long. The second feature is that when there is an important handwriting during the lecture, when the professor makes a biceps motion, he automatically captures the handwriting on the blackboard and

automatically saves the image. To this end, 'mediapipe', a model that detects the professor's gestures, the 'Yolov5s' model that recognizes people, and 'Arduino' to control the motor were used. The software structure and flow chart are as follows

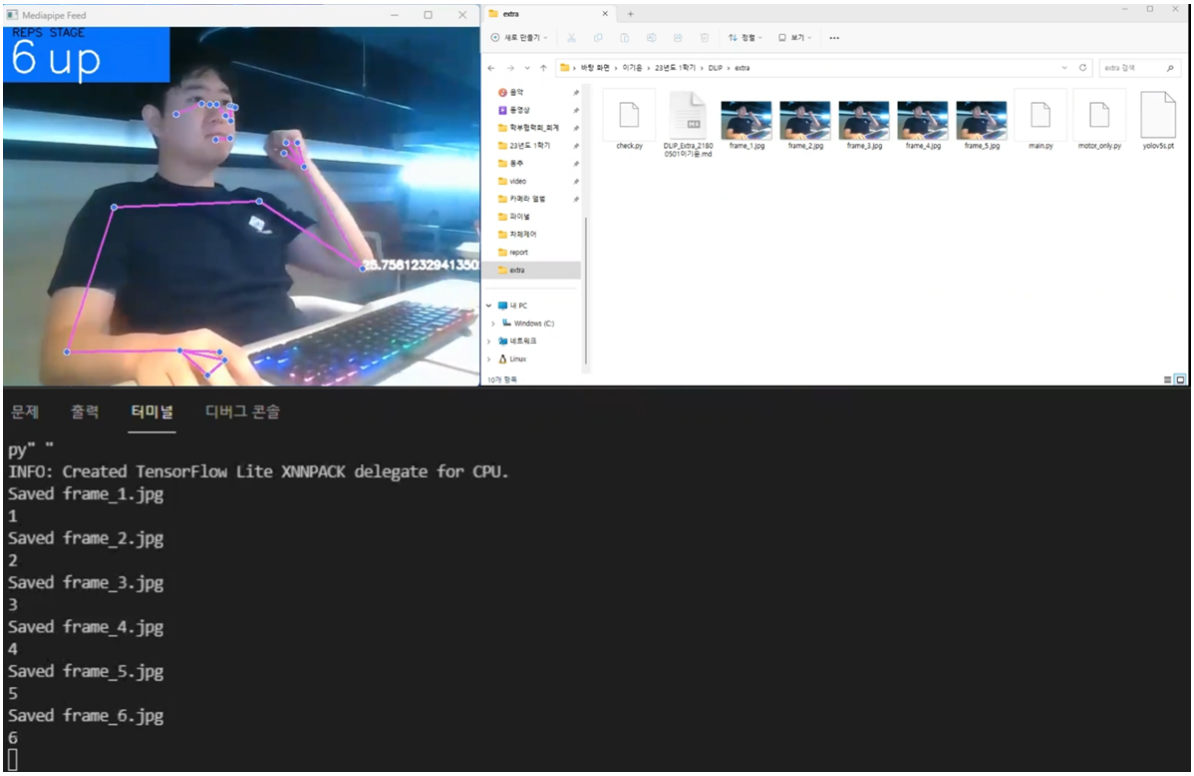




5. Tutorial Procedure

According to the order described in the flowchart above, it will be explained in the order of 'Pose detection', 'Person Tracking algorithm', and 'arduino motor control'. The core algorithm applied to each part will be explained focusing on functions.

5.1 Pose detection



I will explain the main functions applied to 'pose detection'. 'calculate_angle' is a function that calculates and returns the angle of the arm. First, pixel coordinates of a portion corresponding to the shoulder, elbow, and wrist are calculated. These are points automatically detected by the meidiapipe algorithm and will be used to calculate the angle by storing the corresponding pixel coordinates

```
# Extract landmarks
        try:
            landmarks = results.pose_landmarks.landmark

            # Get coordinates
            shoulder =
[landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x,landmarks[mp_pose.PoseLan
dmark.LEFT_SHOULDER.value].y]
            elbow =
[landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x,landmarks[mp_pose.PoseLandma
rk.LEFT_ELBOW.value].y]
            wrist =
[landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x,landmarks[mp_pose.PoseLandma
rk.LEFT_WRIST.value].y]

          except:
                pass
```

Next, The three points of the arm are designated by the user in advance, and the angle is calculated using the points as a reverse tangent function.

```
def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End

    radians = np.arctan2(c[1]-b[1], c[0]-b[0]) - np.arctan2(a[1]-b[1], a[0]-
b[0])
    angle = np.abs(radians*180.0/np.pi)

    if angle >180.0:
        angle = 360-angle

    return angle
```

Finally, the angle was calculated and the flag was set to capture the corresponding frame image every time the person made a gesture based on the angle change. Initialize flag after capture.

```
# Curl counter logic
    if angle > 160:
        stage = "down"
    if angle < 30 and stage =='down':
            stage="up"
            counter +=1
            flag=1
            print(counter)

    if flag==1:
            cv.imwrite('/path/to/your/directory/frame.jpg', frame)
            flag = 0
```

Finally, plot the results.

```
# Render curl counter
# Setup status box
cv.rectangle(image, (0,0), (225,73), (245,117,16), -1)

# Rep data
cv.putText(image, 'REPS', (15,12),
           cv.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv.LINE_AA)
cv.putText(image, str(counter),
           (10,60),
           cv.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 2, cv.LINE_AA)

# Stage data
cv.putText(image, 'STAGE', (65,12),
           cv.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv.LINE_AA)
cv.putText(image, stage,
           (60,60),
           cv.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 2, cv.LINE_AA)


# Render detections
mp_drawing.draw_landmarks(image, results.pose_landmarks,
mp_pose.POSE_CONNECTIONS,
```

```
                            mp_drawing.DrawingSpec(color=(245,117,66),
thickness=2, circle_radius=2),
                            mp_drawing.DrawingSpec(color=(245,66,230),
thickness=2, circle_radius=2)
                           )

cv.imshow('Mediapipe Feed', image)

if cv.waitKey(10) & 0xFF == ord('q'):
    break

    cap.release()
    cv.destroyAllWindows()
```

The result is below

5.2 Person Tracking Algorithm

First of all, I divide the width of the frame by 10 segments and then define the area between segment and segment.

```
#this is frame's area divided by x axis
area = [100,150,225,300,375,450,525,600]
```

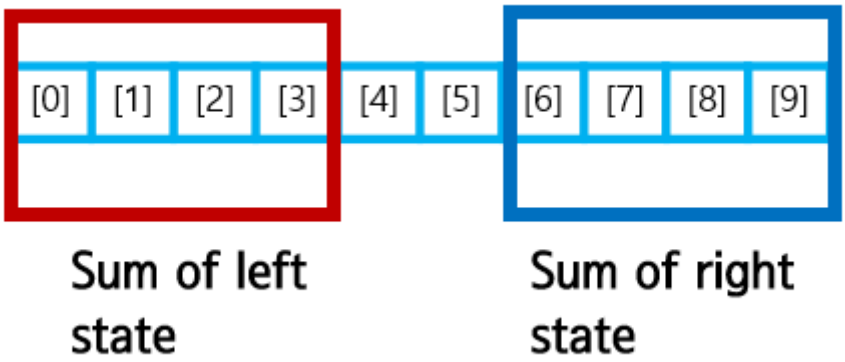
Above code is the code that stacks the state history. and the person doesn't always fit the area. We have to find the closest area to determine the person's position in the frame. the function is as follows.

```python
# find closest area of the object
def find_closest_index(target, arr = area):
    closest_diff = float('inf')
    closest_index = None

    for i, num in enumerate(arr):
        diff = abs(target - num)
        if diff < closest_diff:
            closest_diff = diff
            closest_index = i

    return closest_index
```

and then the function that saves detected Person's (x1, y1, x2, y2) in **person** array. Afterwards, the x1x2 coordinates are averaged from the stored array, the x-means coordinates for each person are calculated, and updated to the state_list by +1 at the closest position in the area.



To track the person, It is important to move the camera following the person. So, using **the area state array** that we made, divide left and right. If the sum of the left states is bigger than the sum of the right states, the person is on the left of the frame. Then, the camera has to move left toward. and we didn't divide by half of the width. If we set [0]~[4] left and [5]~[6] right the motor turns left and right too fast therefore We give the [4] [5] areas empty for preventing turning too fast for the detected person. Below code is the function that calculates the sum of the states. Finally, We save the direction and velocity for **array[2]**. and using Serial communication, we are going to convey these commands to arduino finally to turn the camera to track the person.

The following function is a code implementation of the above description, and the driving direction of the motor is determined by comparing the sum of the weights on the left and the sum of the weights on the right.

```python
def calculate_weighted_index(arr):
    # all_sum = sum(arr)
    left_sum = sum(arr[:5])
    right_sum = sum(arr[7:])
    direction=0
    speed=-1
    if (left_sum>right_sum):
        direction=-1
        # speed = abs(4-index)
    elif(right_sum>left_sum):
        direction=1
        # speed = abs(5-index)
    return direction, speed
```

and then send the array which includes 'direSction' and 'vcelocity' from the algorithm to the arduino.

```python
def send_array_to_arduino(array):
    array_string = ','.join(map(str, array))
    try:
        arduino.write((array_string + '\n').encode())
    except Exception as e:
        print(f"Error: {e}")
```

5.3 Arduino motor control

Finally arduino receives the array from python, motor_only.py and move the motor according to the commands from the server.py

```
#This is arduino code
#include <Wire.h>
#include <Adafruit_MotorShield.h>

// Adafruit_MotorShield object
Adafruit_MotorShield AFMS = Adafruit_MotorShield();

// DC Motor is connected M4 Port
Adafruit_DCMotor *myMotor = AFMS.getMotor(4);
int speed = 0;

void setup() {
  Serial.begin(9600); //Start serial communication

  // Adafruit Motor Shield Start
  AFMS.begin();

  // DC Motor Basic velocity set
  myMotor->setSpeed(100); // Select Velocity (0~255)
}

void loop() {
  if (Serial.available())// Check if data exists in Serial Port
```

```
  {
    String command = Serial.readStringUntil('\n');  // Read Data
    // Convert Strings to array. ',' is the divide standard
    int array[2];
    int index = 0;
    for (int i = 0; i < command.length(); i++) {
        int commaIndex = command.indexOf(',', i);
        if (commaIndex == -1) {
            array[index] = command.substring(i).toInt();
            break;
        } else {
            array[index] = command.substring(i, commaIndex).toInt();
            index++;
            i = commaIndex;
        }
    }
    int direction = array[0]; //direction command from server.py
    int velocity  = array[1]; // velocity command from server.py

    if( direction != 0){
      myMotor->setSpeed(velocity);
      if(direction == 1){
        myMotor->run(BACKWARD);
      }else{
        myMotor->run(FORWARD);
      }
    }else{
      myMotor->run(RELEASE);
    }
  }
}
```

6. Result and Analysis

6.1 Evaluation Standard

First, the first evaluation criterion was set as the frequency of errors in tracking whether people follow well when recording lectures. If a professor moves and the camera follows a person, it is not appropriate from the beginning to record a lecture alone, which is a problem suggested by the system. Therefore, it was used as a criterion for evaluating the accuracy of tracking. Next, the second evaluation criterion was set as the error frequency of the screen capture function according to the gesture. Since how well the instructor detects the gesture to capture the handwriting recorded by the professor is also an important part of this system, the evaluation criteria were set as above.

6.2 Objective

Accuracy was targeted at more than 80 percent for both attendance mode and CCTV mode.

6.3 Result

The evaluation of Smart Lecture Recording System Demo

Demo (date)	Person Tracking Accuracy	Gesture Detection accuracy
Demo(6/25)	(7/7)	(5/5)
Demo(6/26)	(8/8)	(6/7)

In the first (6/25) demo I achieved 100% accuracy. I think this is the result of preparing and proceeding with an optimal experiment on how to recognize gestures because I knew how to take classes from the location of the camera to the distance of the lecture board and the screen composition. However, in the second demonstration, I was able to confirm in the experiment that there was an error in gesture recognition about 1 time.

6.4 Analysis

Analyzing this, first, it seemed to be the limitation of the custom model for human face recognition. [Lee_ck] and [Lee_ky] present in the train data set showed very accurate detection with more than 90% accuracy, but other people which belong to were also recognized as Lee_ck or Lee_ky three out of eight times. Previously, while preparing for dataset learning, accessories such as headsets, hoods, and hats were added to Lee_ck or Lee_ky's face to enhance characteristics compared to others, but it was confirmed that there was insufficient to obtain high accuracy by learning a person's side profile with a custom model. Second, it was confirmed that there was a problem in the labeling process. As a result of re-checking the dataset used for training, it was confirmed that about 500 out of 6,500 sheets were lost due to errors due to the coordinates deviating between 0 and 1 during the training process. This is estimated to have shown insufficient train performance due to 10% loss of learning data as a result of turning the learning without checking it and sometimes labeling the coordinates out of the frame using the space bar in the dark label program.

Therefore, in order to improve this, first, it is expected that performance can be improved using the yolo model, which has better performance than the existing yolov5s custom model. To distinguish a person from others with a profile, a more recent version of yolo can be used to expect improved performance with the same number of train data. Second, it seems that it can be improved algorithmically. Current algorithms capture when a person first appears in a frame, and do not capture and analyze faces separately when leaving. However, if you capture a person's image both when entering and leaving, compare the two values, and set them to be labeled if both are recognized as Lee_ky or Lee_ck, you will be able to show higher accuracy than before.

7. Appendix

7.1 movement_detection code

```
import cv2 as cv
import mediapipe as mp
import numpy as np
import torch
mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose

cap = cv.VideoCapture(0)

# Curl counter variables
counter = 0
stage = None
image_number=1

def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End

    radians = np.arctan2(c[1]-b[1], c[0]-b[0]) - np.arctan2(a[1]-b[1], a[0]-
b[0])
    angle = np.abs(radians*180.0/np.pi)

    if angle >180.0:
        angle = 360-angle

    return angle


## Setup mediapipe instance
with mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5) as
pose:
    while cap.isOpened():
        ret, frame = cap.read()

        if cv.waitKey(1) & 0xFF == 27:
            break
        # Recolor image to RGB
        image = cv.cvtColor(frame, cv.COLOR_BGR2RGB)
        image.flags.writeable = False

        # Make detection
        results = pose.process(image)

        # Recolor back to BGR
        image.flags.writeable = True
        image = cv.cvtColor(image, cv.COLOR_RGB2BGR)

        # Extract landmarks
        try:
            landmarks = results.pose_landmarks.landmark

            # Get coordinates
```

```
            shoulder =
[landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x,landmarks[mp_pose.PoseLan
dmark.LEFT_SHOULDER.value].y]
            elbow =
[landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x,landmarks[mp_pose.PoseLandma
rk.LEFT_ELBOW.value].y]
            wrist =
[landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x,landmarks[mp_pose.PoseLandma
rk.LEFT_WRIST.value].y]

            # Calculate angle
            angle = calculate_angle(shoulder, elbow, wrist)

            # Visualize angle
            cv.putText(image, str(angle),
                           tuple(np.multiply(elbow, [640, 480]).astype(int)),
                           cv.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2,
cv.LINE_AA
                                )

            # Curl counter logic
            if angle > 160:
                stage = "down"
                # cv.imwrite(cv.imwrite(f'C:/Users/LeeKiYoon/Desktop/이기윤/23년도
1학기/DLIP/extra/{counter}.jpg', image))



            if angle < 30 and stage =='down':
                stage="up"
                counter +=1
                cv.imwrite(f'frame_{image_number}.jpg', frame)
                print(f'Saved frame_{image_number}.jpg')
                image_number += 1
                print(counter)

            # cv.imwrite(cv.imwrite(f'C:/Users/LeeKiYoon/Desktop/이기윤/23년도 1학
기/DLIP/extra/{counter}.jpg', image))
            # if flag==1:

            #     flag = 0



        except:
            pass

        # Render curl counter
        # Setup status box
        cv.rectangle(image, (0,0), (225,73), (245,117,16), -1)

        # Rep data
        cv.putText(image, 'REPS', (15,12),
                    cv.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv.LINE_AA)
        cv.putText(image, str(counter),
```

```
                    (10,60),
                    cv.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 2, cv.LINE_AA)

        # Stage data
        cv.putText(image, 'STAGE', (65,12),
                    cv.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv.LINE_AA)
        cv.putText(image, stage,
                    (60,60),
                    cv.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 2, cv.LINE_AA)


        # Render detections
        mp_drawing.draw_landmarks(image, results.pose_landmarks,
mp_pose.POSE_CONNECTIONS,
                                  mp_drawing.DrawingSpec(color=(245,117,66),
thickness=2, circle_radius=2),
                                  mp_drawing.DrawingSpec(color=(245,66,230),
thickness=2, circle_radius=2)
                                   )

        cv.imshow('Mediapipe Feed', image)

        if cv.waitKey(10) & 0xFF == ord('q'):
            break

    cap.release()
    cv.destroyAllWindows()
```

7.2 motor_only.py

```
import cv2 as cv
import timeit as time
import torch
import datetime
import os
import socket
import pickle
from queue import Queue
import serial
import time

area = [100,150,225,300,375,450,525,600]
AREA_NUMBER = len(area)

def find_closest_index(target, arr = area):
    closest_diff = float('inf')
    closest_index = None
    for i, num in enumerate(arr):
        diff = abs(target - num)
        if diff < closest_diff:
            closest_diff = diff
            closest_index = i
    return closest_index
```

```
def calculate_weighted_index(arr):
    # all_sum = sum(arr)
    left_sum = sum(arr[:5])
    right_sum = sum(arr[7:])
    direction=0
    speed=-1
    if (left_sum>right_sum):
        direction=-1
        # speed = abs(4-index)
    elif(right_sum>left_sum):
        direction=1
        # speed = abs(5-index)
    return direction, speed

def send_array_to_arduino(array):
    array_string = ','.join(map(str, array))
    try:
        arduino.write((array_string + '\n').encode())
    except Exception as e:
        print(f"Error: {e}")

#load pretrained model
cap = cv.VideoCapture(1)
model = torch.hub.load('ultralytics/yolov5', 'yolov5s', pretrained=True)


#arduino
port = 'COM3'
baudrate = 9600
arduino = serial.Serial(port, baudrate)
time.sleep(3) #wait for arduino

# save img

if cap.isOpened():
    array_received=[]
    namevalue = []
    while True:
        state_list = [0 for _ in range(AREA_NUMBER)]
        #exit
        if cv.waitKey(1) & 0xFF == 27:
            break
        ret, frame = cap.read()
        height, width = frame.shape[:2]
        #about model
        results= model(frame)
        person=[]
        for result in results.pandas().xyxy[0].iterrows():
            if (result[1]['name'] in ('person')):
                x1, y1, x2, y2 = int(result[1]['xmin']), int(result[1]['ymin']),
int(result[1]['xmax']), int(result[1]['ymax'])
                person.append([x1,y1,x2,y2])
                cv.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
        for plot in person:
```

```
            state_list[find_closest_index((plot[0]+plot[2])/2)]+=1
        #default
        array=[0,1]
        array[0], array[1] = calculate_weighted_index(state_list)
        print(array)

        send_array_to_arduino(array)

        cv.imshow("Webcam", frame)
else:
    print('cannot open the camera')
cap.release()
cv.destroyAllWindows()
```

7.3 Arduino code

```
#include <Wire.h>
#include <Adafruit_MotorShield.h>

// Adafruit_MotorShield object creation
Adafruit_MotorShield AFMS = Adafruit_MotorShield();

// connect DC motor to M4 Port
Adafruit_DCMotor *myMotor = AFMS.getMotor(4);
int speed = 0;

void setup() {
  Serial.begin(9600);  //Begin Serial communication

  // Adafruit Motor Shield Set
  AFMS.begin();

  // DC motor basic velocity set
  myMotor->setSpeed(100);
}

void loop() {
  if (Serial.available()) {  //Check the data exists in serial Port
    String command = Serial.readStringUntil('\n');  // Read data
    // Convert strings to array and save array
    int array[2];
    int index = 0;
    for (int i = 0; i < command.length(); i++) {
        int commaIndex = command.indexOf(',', i);
        if (commaIndex == -1) {
            array[index] = command.substring(i).toInt();
            break;
        } else {
            array[index] = command.substring(i, commaIndex).toInt();
            index++;
            i = commaIndex;
```

```
        }
    }
    int direction = array[0]; // direction command from server.py
    int velocity =  array[1]; // velocity command from server.py

    if( direction != 0){
      myMotor->setSpeed(velocity);
      if(direction == 1){
        myMotor->run(BACKWARD);
      }else{
        myMotor->run(FORWARD);
      }
    }else{
      myMotor->run(RELEASE);
    }
  }
}
```

8. Reference

YOLOv5 : Click here

Darklabel: Click here

9. Demo Video

9.1 Demo Video

DLIP_Extra_DEMO_for_SSL_youtube