Yolo_v4 Smart Scarecrow 만들기

고층 건물 실외기에 비둘기로 인하여 오염되는 것을 방지하고자

비둘기 인식 및 추적, 자동 물총 발사를 통한 Smart Scarecrow 제작

😁구성도😁

- PC (Webcam) ↔ Raspberry Pi4(Servo Motor) 간 Socket Connection 구현
- PC (Webcam)을 통한 Yolo_v4 비둘기 학습 및 구현
- Raspbeery Pi4(Servo Motor)에 PC (Webcam)을 결합하여 회전시키기

Requirements

- Python
- Yolo_v4
- Raspberry Pi 4
- Servo Motor (SG90, MG996R)
- USB Webcam

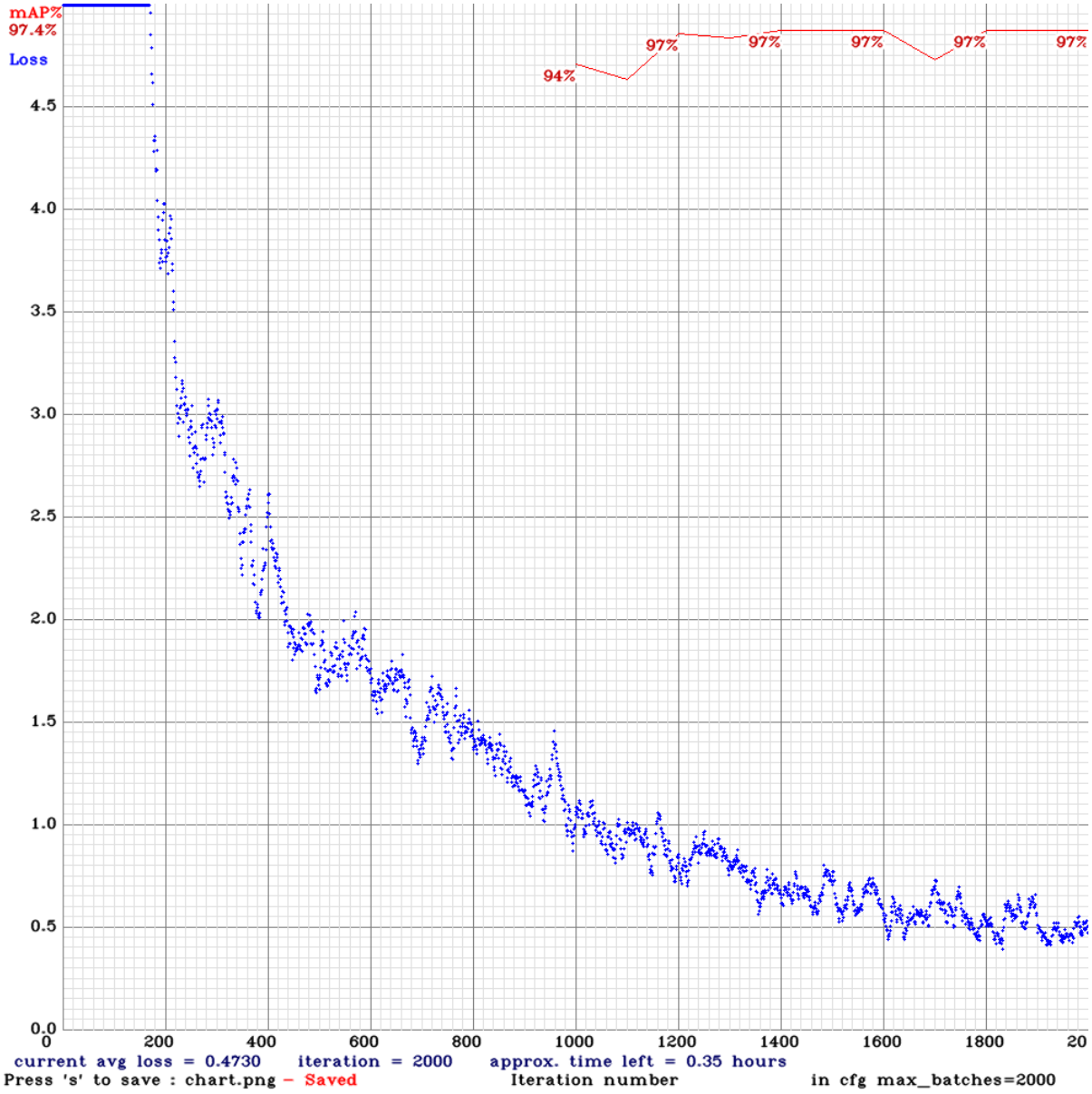
Yolo_v4 설치

https://www.youtube.com/watch?v=5pYh1rFnNZs

- 위 영상 참고하여 설치

Yolo_v4 비둘기 Custom Training

- Image Dataset 만들기
- 비둘기 Image 300장 수집
  (비둘기가 겹치지 않은 상태로 온전하게 있는 사진)
- Yolo_Mark를 사용하여 비둘기 Labeling
- 학습을 위한 cfg파일 수정
  - Width, Height = 512
  - Max_batches = Class 수(1) * 2000 = 2000
  - steps = Max_batches의 80%, 90%
  - classes = 1
  - filters = (4+1+classes) * 3 = 18
- obj.names 수정
  - pigeon
- obj.data 수정
  - classes = 1
  - train =/darknet/train.txt
  - valid = /darknet/valid.txt
  - names = /darknet/obj.names
  - backup = /darknet/backup
- weight 파일
  - darknet53.conv.74
- Training(CMD)
  - darknet.exe detector train data/obj.data yolo-obj.cfg yolov4.conv.137
  - 맨 뒤에 -map 을 붙여서 Training을 하면 실시간 학습 그래프 확인 가능
- 결과 확인



https://www.youtube.com/watch?v=e0q-Pqr5URo

Bbox 좌표 획득을 위한 코드 수정

- darknet_video.py

```
from time import sleep
global bbox_center_coordinates
def convertBack(x, y, w, h):
    xmin = int(round(x - (w / 2)))
    xmax = int(round(x + (w / 2)))
    ymin = int(round(y - (h / 2)))
```

```
    ymax = int(round(y + (h / 2)))
    return xmin, ymin, xmax, ymax

def cvDrawBoxes(detections, img):
    global bbox_center_coordinates
    for detection in detections: #for문을 통해 xmin, xmax, ymin, ymax 좌표 출력
        x, y, w, h = detection[2][0],\
            detection[2][1],\
            detection[2][2],\
            detection[2][3]
        xmin, ymin, xmax, ymax = convertBack(
            float(x), float(y), float(w), float(h))

        xcenter1=int((xmin+xmax)//2) //# bbox의 x center coordinates
        ycenter1=int((ymin+ymax)//2) //# bbox의 y center coordinates
        pt1 = (xmin, ymin) //bbox의 좌측 하단 좌표 (x,y)
        pt2 = (xmax, ymax) //bbox의 우측 상단 좌표 (x,y)
        cv2.rectangle(img, pt1, pt2, (0, 255, 0), 1) # bbox 표시
        cv2.circle(img,(xcenter1,ycenter1),5,(255,255,255)) # bbox의 정중앙에 점 표시
        cv2.putText(img,
                    detection[0].decode() +
                    " [" + str(round(detection[1] * 100, 2)) + "]",
                    (pt1[0], pt1[1] - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
                    [0, 255, 0], 2)
        bbox_center_coordinates=256-int((xmin+xmax)//2)
# cfg에서 설정한 해상도 512의 중앙값 256에서 bbox의 center x값을 빼서 화면 중앙값과 bbox와의 거리차이 획득
        f = open('./coordinates/coordinates.txt', 'w')
        data = str(bbox_center_coordinates)
        f.write(data)
        f.close()
        print('pigeon detected',bbox_center_coordinates)
# 비둘기가 인식됐다는 것을 알리고, 화면 중앙과 bbox 중앙좌표의 좌표 차이 print

    return img
```

```
def YOLO():

    global metaMain, netMain, altNames
    configPath = "./cfg/yolov4-custom.cfg"  # 설정해놓은 cfg 경로
    weightPath = "./backup/yolov4-custom_last.weights" #Custom Training을 진행한 weight 경로
    metaPath = "./data/voc.data" # 설정해놓은 obj.data 경로
    if not os.path.exists(configPath):
        raise ValueError("Invalid config path `" +
                         os.path.abspath(configPath)+"`")
    if not os.path.exists(weightPath):
        raise ValueError("Invalid weight path `" +
                         os.path.abspath(weightPath)+"`")
    if not os.path.exists(metaPath):
        raise ValueError("Invalid data file path `" +
                         os.path.abspath(metaPath)+"`")
    if netMain is None:
        netMain = darknet.load_net_custom(configPath.encode(
            "ascii"), weightPath.encode("ascii"), 0, 1)  # batch size = 1
    if metaMain is None:
        metaMain = darknet.load_meta(metaPath.encode("ascii"))
    if altNames is None:
```

```
        try:
            with open(metaPath) as metaFH:
                metaContents = metaFH.read()
                import re
                match = re.search("names *= *(.*)$", metaContents,
                                  re.IGNORECASE | re.MULTILINE)
                if match:
                    result = match.group(1)
                else:
                    result = None
                try:
                    if os.path.exists(result):
                        with open(result) as namesFH:
                            namesList = namesFH.read().strip().split("\n")
                            altNames = [x.strip() for x in namesList]
                except TypeError:
                    pass
        except Exception:
            pass
    cap = cv2.VideoCapture(0)  # 웹캠 실행
    #cap = cv2.VideoCapture("test.mp4")  # 영상 실행
    #사용하고자 하는 기능을 주석을 제거해서 실행
    cap.set(3, 1280)
    cap.set(4, 720)
    out = cv2.VideoWriter(
        "output.avi", cv2.VideoWriter_fourcc(*"MJPG"), 10.0,
        (darknet.network_width(netMain), darknet.network_height(netMain)))
```

```
while True:
        prev_time = time.time()
        ret, frame_read = cap.read()
        frame_rgb = cv2.cvtColor(frame_read, cv2.COLOR_BGR2RGB)
        frame_resized = cv2.resize(frame_rgb,
                                   (darknet.network_width(netMain),
                                    darknet.network_height(netMain)),
                                   interpolation=cv2.INTER_LINEAR)

        darknet.copy_image_from_bytes(darknet_image,frame_resized.tobytes())

        detections = darknet.detect_image(netMain, metaMain, darknet_image, thresh=0.25)
        image = cvDrawBoxes(detections, frame_resized)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        print(1/(time.time()-prev_time))
        cv2.imshow('Demo', image)
        cv2.waitKey(3)
        print(detections)
        if detections == [] : #객체를 인식하지 않았을 때, 좌표에 nothing을 입력하여 초기화
            f = open('./coordinates/coordinates.txt','w')
            data = "nothing"
            f.write(data)
            f.close()
            print('nothing')
    cap.release()
    out.release()
```

```
if __name__ == "__main__":
    YOLO()
```

Socket 통신을 위한 코드 작성(PC)

- Socket_connection(client).py

```
import socket
import threading
import time

HEADER = 64
PORT = #port number
FORMAT = "utf-8"
DISCONNECT_MESSAGE = "!DISCONNECT"
SERVER = #"Raspberry Pi4 IP adress"
ADDR = (SERVER, PORT)


client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(ADDR)
print("Connected by", ADDR)

#def send(msg):
#    message = msg.encode(FORMAT)
#    msg_length = len(message)
#    send_length = str(msg_length).encode(FORMAT)
#    send_length += b' ' * (HEADER - len(send_length))
#    client.send(send_length)
#    client.send(message)
copy = 'nothing' # copy를 nothing으로 초기화,
while True: # 좌표를 실시간으로 읽어오기 위한 while문
        f = open('./coordinates/coordinates.txt', 'r')
        s = f.read()
        f.close()

        if copy == s : # 새로운 좌표 차이값이 갱신되지 않을 경우
            continue
        else : # 새로운 좌표 차이값이 갱신되었을 경우
            print('coordinates : ', s)
            copy = s
            client.sendall(s.encode()) #Rsp4로 새로운 좌표 차이값 갱신

client.close()
```

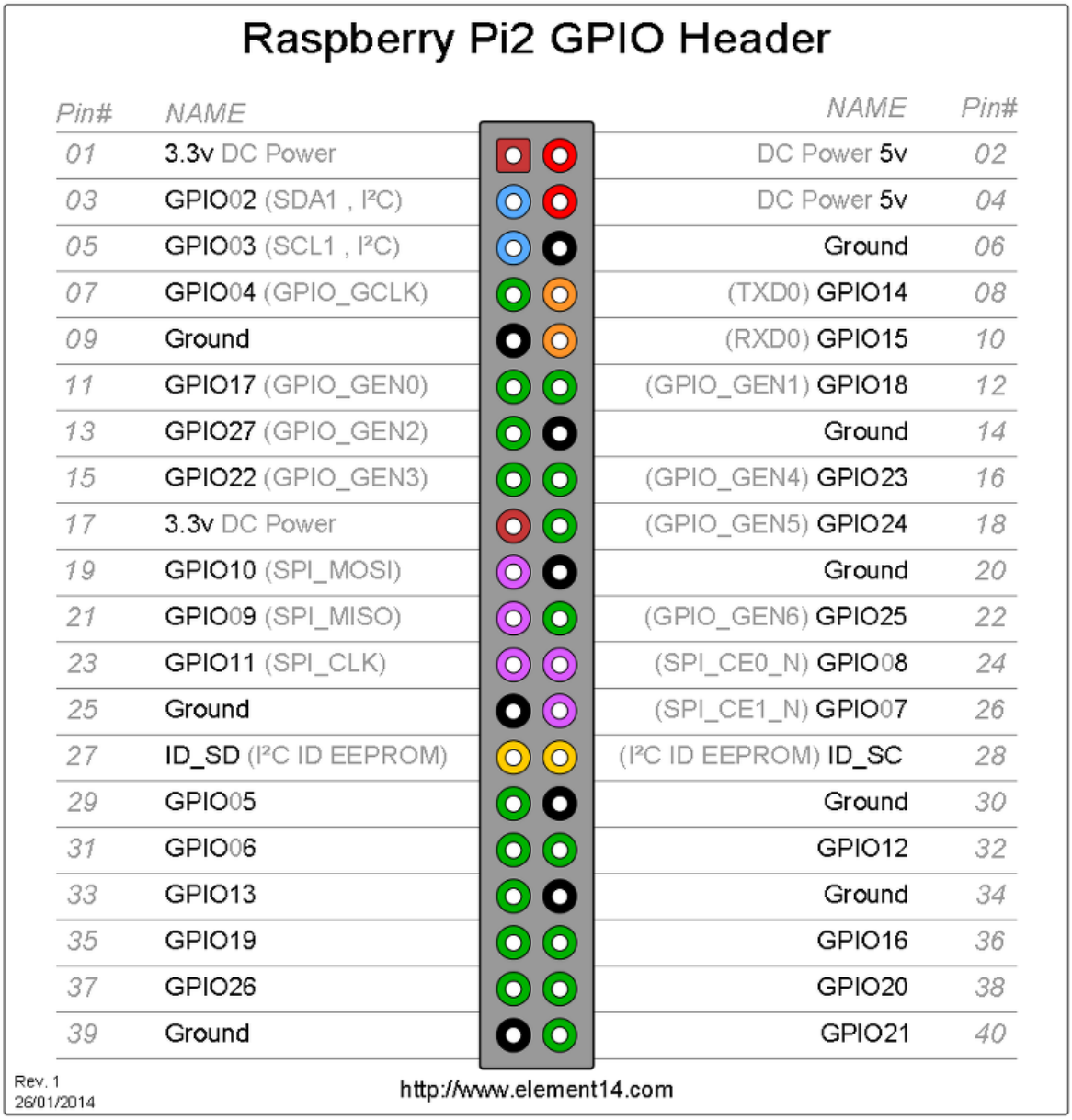
Raspberry Pi4 설정

- VNC Viewer 사용
- Rsp (Server) ↔PC (Client) Socket 통신을 위한 코드 작성

- Socket_connection(server).py

```
import socket
import RPi.GPIO as GPIO
import time
from time import sleep

HOST = #'RSP4 IP adress'
PORT = # port number

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((HOST,PORT))

server.listen()

client, addr = server.accept()
print("Connected by", addr)

copy = 'nothing'
while True:
  data = client.recv(1024)
  print('coordinates : ', repr(data.decode())) #pc에서 보낸 좌표 차이값 받기
  copy = data # 좌표 차이값 text에 입력
  f = open('/home/minsu/coordinates.txt', 'w')
  f.write(data.decode())
  f.close()


server.close()
```

Raspberry Pi4 Servo Motor 연결



- 5V DC Power : pin#02, pin#04
- Ground : pin#06, pin#14
- PWM : pin#12(GPIO18), pin#32(GPIO12)

Raspberry Pi4 Servo Motor 제어 코드 작성

- Servo_motor.py

```
import pigpio  # PWM을 밀리세컨드가 아닌 마이크로세컨드로 보내는 Library
from time import sleep
n_pulsewidth = 600 #(position)
```

```
pulsewidth = 10 #(stride)
pi = pigpio.pi()
default = open('/home/minsu/coordinates.txt', 'w+')
data ="nothing"
default_write = default.write(data)
default.close()
while True: #좌표 차이 값이 입력된 txt 읽기
        f = open('/home/minsu/coordinates.txt', 'r')
        c = f.read()
        f.close()
        print(c)

        try:
                c = int(c)
        except ValueError:
                c = "nothing"
        # -30 to 30이 오차 허용 범위. 이 안에 객체가 들어오면 추적 정지
        if c != "nothing": #좌표 차이값이 존재할 경우
                if c < -30 : #bbox가 화면 우측에 위치할 경우
                        n_pulsewidth -= 10
                        pi.set_servo_pulsewidth(18,n_pulsewidth)
                        print('*' * 20)
                        print('(2-1)Decrease by 1 degree to the right')
                        print('(2-1)Angle : ',(n_pulsewidth - 600)/10, ' degree' )
                        print('(2-1)Featured c : ',c)
                        print('*' * 20)
                        print('this is c <-30')
                        sleep(.5)

                elif -30 <= c <= 30: #bbox가 화면 중앙에 위치할 경우
                        print('*' * 20)
                        print('(2-2)Target is on center')
                        print('(2-2)degree is : ',(n_pulsewidth-600)/10,' degree')
                        print('(2-2)Featured c : ', c)
                        print('*' * 20)
                        pi.set_servo_pulsewidth(12,2200)
            # 회전할때 쓰는 18번 SG90이 아닌, 12번 SG90을 회전시켜 물총트리거 작동
                        print('this is -30 < c <30')
                        sleep(3)

                elif 30 < c : #bbox가 화면 좌측에 위치할 경우
                        n_pulsewidth += 10
                        pi.set_servo_pulsewidth(18,n_pulsewidth)
                        print('*' * 20)
                        print('(2-3)Increases by 1 degree to the left')
                        print('(2-3)Angle : ',(n_pulsewidth - 600)/10, ' degree' )
                        print('(2-3)Featured c : ',c)
                        print('this is 30 < c')
                        print('*' * 20)
                        sleep(.5)

        else :  #if c == 'nothing' : #정찰모드, 0도부터 180도, 180도에서 0도로 회전 반복
                pi.set_servo_pulsewidth(12,600)
                n_pulsewidth += pulsewidth
                if n_pulsewidth >= 2400 :
                        pulsewidth = -pulsewidth
                        n_pulsewidth = 2400
```

```
                elif n_pulsewidth <= 600 :
                        pulsewidth = -pulsewidth
                        n_pulsewidth = 600

                pi.set_servo_pulsewidth(18,n_pulsewidth)
                print('*' * 20)
                print('(1-1) Rotation')
                print('(1-1) Angle :',(n_pulsewidth - 600)/10, ' degree')
                print('this is nothing detected')
                print('*' * 20)
                sleep(.1)
```

결과

https://www.youtube.com/watch?v=5KEdQR_yqyU

- 좌측 상단 : Socket_connection(Client).py
- 좌측 하단 : Darknet_video.py
- 우측 상단 : Servo_motor.py
- 우측 하단 : Socket_connection(Server).py

1. 탐색 모드(0 ~ 180, 180 ~ 0 Degree)로 계속 회전하며 실시간 감시 진행
2. 비둘기가 인식될 경우, 화면 x 중앙 좌표와 비둘기 bbox의 x 중앙 좌표의 차를 계산
3. 이를 소켓통신으로 Raspberry Pi4에 전송
4. 받은 좌표 차이 값에 따라 Servo Motor Control
5. bbox가 오차허용범위(-30 ~ +30)에 들어왔을 경우, 손에 들고있는 SG90 회전(물총발사)
6. 물총위에 서보모터+WEBCAM을 장착 후, 물총 트리거에 발사용 SG90을 설치
7. 인식하고자 하는 객체를 Custom Training 하여 비둘기 외에도 여러 방면으로 활용 가능