컴쟁이들을 위한 자세 잔소리 프로그램 제안

TEAM. 팀장이감자

20233084 류승희
20233070 권나현
20231762 문재민

INDEX

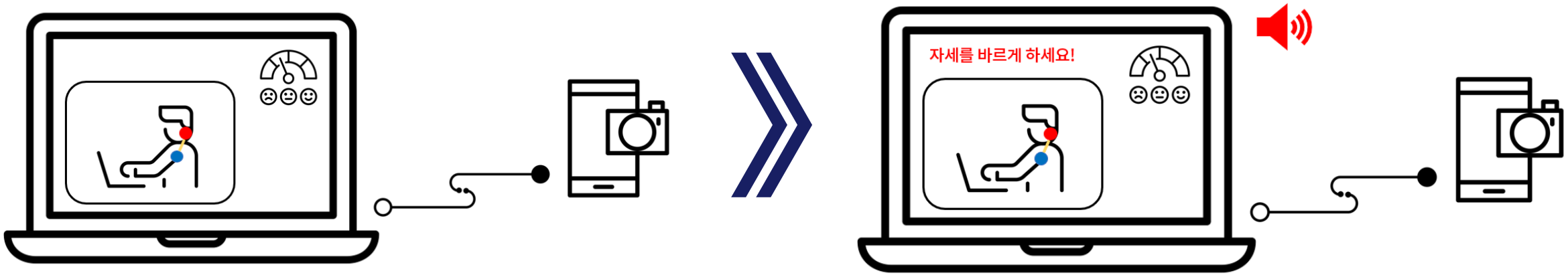
GOAL

BACKGROUND

CONTENTS

GOAL

[대표 그림]

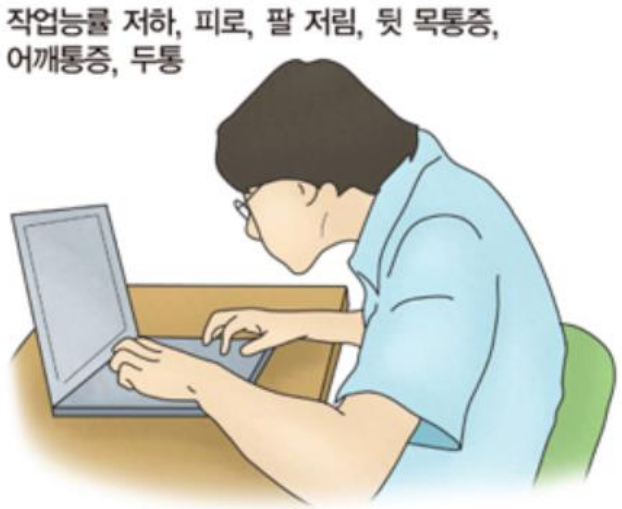


BACKGROUND

[연구 배경]

VDT 증후군(Visual Display Terminal Syndrome)

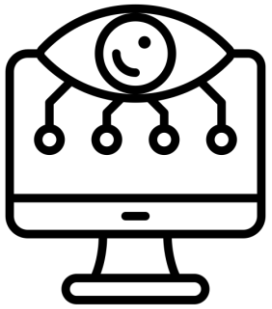
VDT 증후군이란 장시간 동안 모니터를 보며 키보드를 두드리는 작업을 할 때 생기는 각종 신체적, 정신적 장애를 이르는 말이다. 이는 장시간 동안 컴퓨터, 스마트폰, 모바일 디바이스 등을 보는 젊은이에게 많이 나타나는 질환이다.



【거북목 증후군】

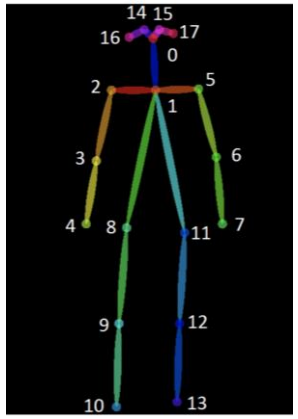
BACKGROUND

[기술]



Computer Vision

컴퓨터 비전은 인공지능의 한 분야로
디지털 이미지, 비디오 및 기타 시각적 입력에서 의미 있는 정보를
추출한 다음 이러한 정보를 바탕으로 작업을 실행할 수 있도록 한다.



Pose Estimation

자세 추정은 컴퓨터 비전을 활용한 기술 중 하나로
ML 모델을 사용하여 주요 신체 관절의 공간적 위치를 추정하여
이미지 또는 비디오로부터 사람의 포즈를 추정하는 작업이다.

BACKGROUND

[사례]

Journal of Korea Multimedia Society Vol. 24, No. 2, February 2021(pp. 285-294)
https://doi.org/10.9717/kmms.2020.24.2.285

웹캠 기반 거북목 판별 알고리즘을 활용한 자세 교정 반응형 헬스케어 시스템

박소연†, 류서진††, 동서연†††

Responsive Healt
Based Turt

Soy

ABSTRACT

This study developed
turtle neck syndrome by
improvement by adjusti
algorithm detecting the t
developed based on mac
line and the shoulder var
problematic due to the i
to the height and posture
In addition, a height-adj
a responsive cradle that
this service enables post
will become an essential

Key words: Forward
Responsi

영상인식 기반 자세교정 시스템*

임윤지°, 에다 카츠토시, 황석형, 김응희, 김수환, 김민경+
{yeinlim12, katsu97625, shwang, ehkim, kimsoohwan, minkyoungkim}@

Attitude Calibration System based on Video Recogr

Yunji Lim°, Katsutoshi Eda, Suk Hyuong Hwang, Eung Hee Kim, Soohwan Kim,
Department of AI & Software Technology, Sunmoon Univer

요 약

기존에 전자기기를 사용함으로써 발생하는 질병(거북목, 허리디스크 등)을 예방하기 위해
진행되어 왔으나, 대부분 전문 의료기기를 이용하여 측정하기 때문에 일상적인 환경(사무실,
용하기 어렵다는 제약이 있다. 따라서, 본 논문에서는 디바이스에 장착된 카메라를 통해 상반
올바르지 못한 자세가 일정시간 유지될 때, 자세 유형별 경고 메세시를 제공할 수 있는 시스템

한국컴퓨터정보학회 동계학술대회 논문집 제30권 제1호 (2022. 1)

거북목 자세를 효율적이고 정확하게 찾기 위한 뼈대 기반 데이터 학습 프레임워크

나홍은°, 김종현*
°강남대학교 소프트웨어응용학부,
*강남대학교 소프트웨어응용학부
e-mail: jonghyunkim@kangnam.ac.kr

Skeleton-Based Data Learning Framework to Efficiently and Accurately Find Text Neck Posture

Hong Eun Na°, Jong-Hyun Kim*
°School of Software Application, Kangnam University,
*School of Software Application, Kangnam University

요 약

본 논문에서는 스마트 기기를 사용할 시 자세가 거북목 자세인지 아닌지 판별하는 시스템을 제안한다. 거북목 증후군이란 목이 구부정하게 앞으로 나오는 자세를 오래 취해 목이 일자목으로 바뀌고 뒷목, 어깨, 허리 등에 통증이 생기는 증상을 말하며, 수술이나 약물치료보다 평소의 자세 습관을 고치는 방법이 효과적이다. 기존의 연구들은 노트북에 내장되어있는 웹캠을 이용한 CNN기반의 학습모델은 영상의 명도와 학습 데이터 등에 많은 영향을 받고 학습 데이터를 모을 때 초상권 문제로 수집이 어렵다. 본 논문에서는 이러한 문제를 예방하고자 Openpose 오픈 소스를 이용한 뼈대를 기반으로 측면에서의 앉은 자세를 한습 모델로 실시간 검증하여, 거북목 자세인지 아닌지를 효율적이고 정확하게 판별한다.

키워드: 거북목 증후군(Text neck posture), 옆모습(Side view), 뼈대(Skeleton),
RGB 카메라(RGB camera), 실시간(Real-time), 딥러닝(Deep learning)

CONTENTS

[문제 정의]

문제 정의

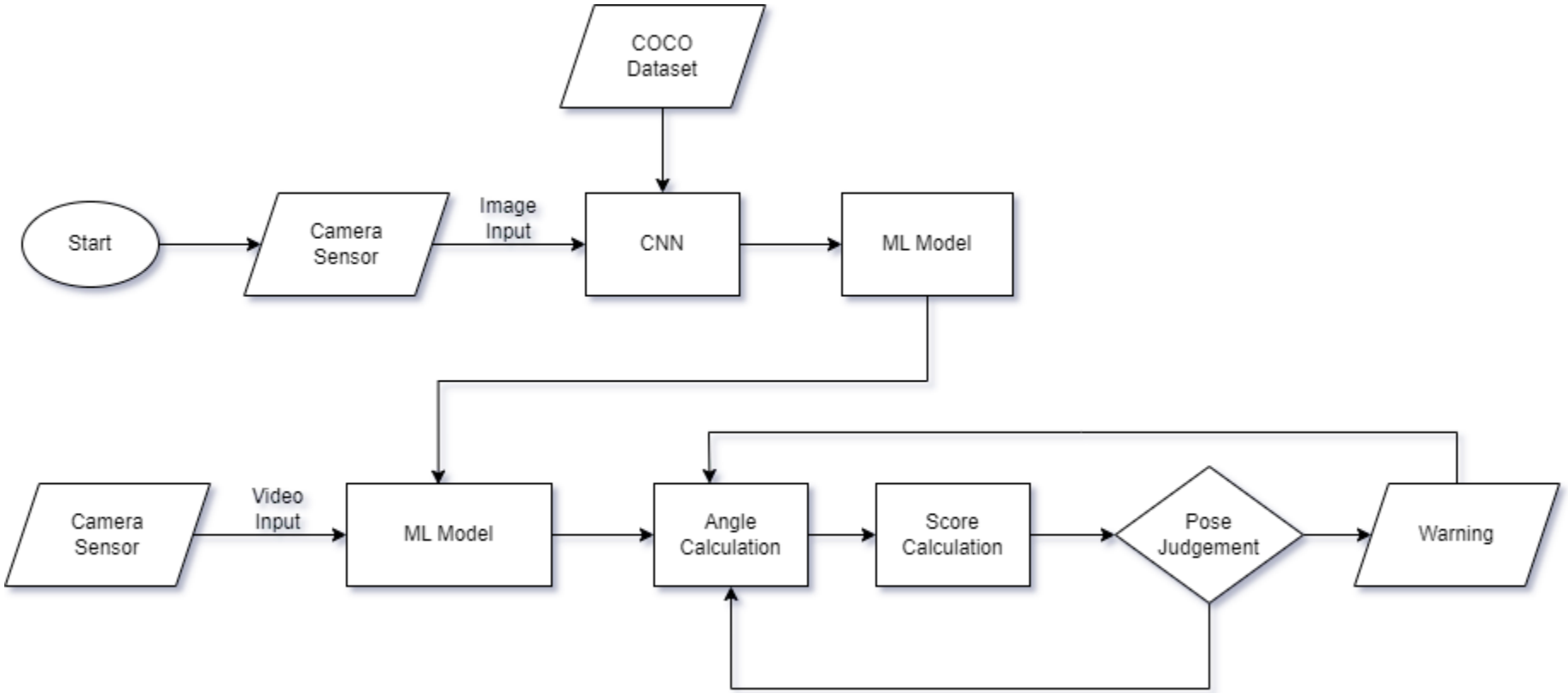
오랜 시간 컴퓨터를 사용하다 보면, 잘못된 자세로 작업할 때가 많고 제 3자가 지적해 주기 전까지는 스스로 인지하기 어렵다.

극복 방안

사용자가 컴퓨터를 하고 있는 측면의 모습을 실시간으로 분석하고 사용자에게 보여주며 바르지 않은 자세에 대해 경고도 해주는 프로그램을 기획하였다.

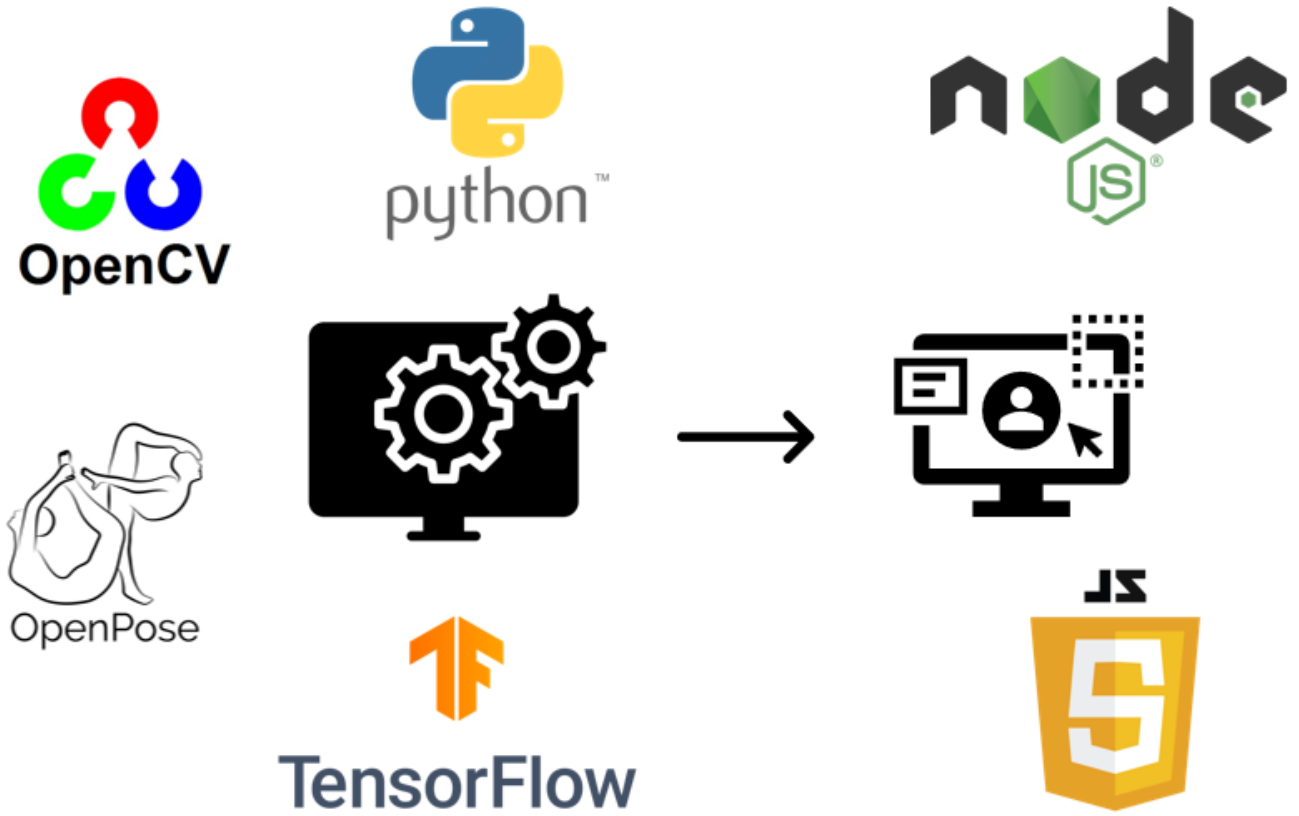
CONTENTS

[순서도]



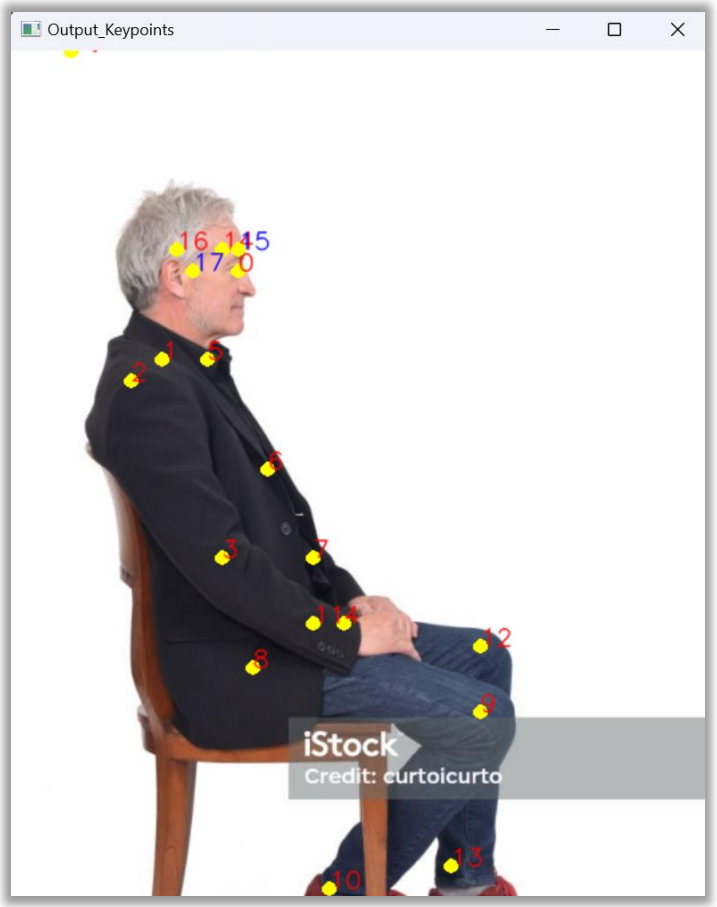
CONTENTS

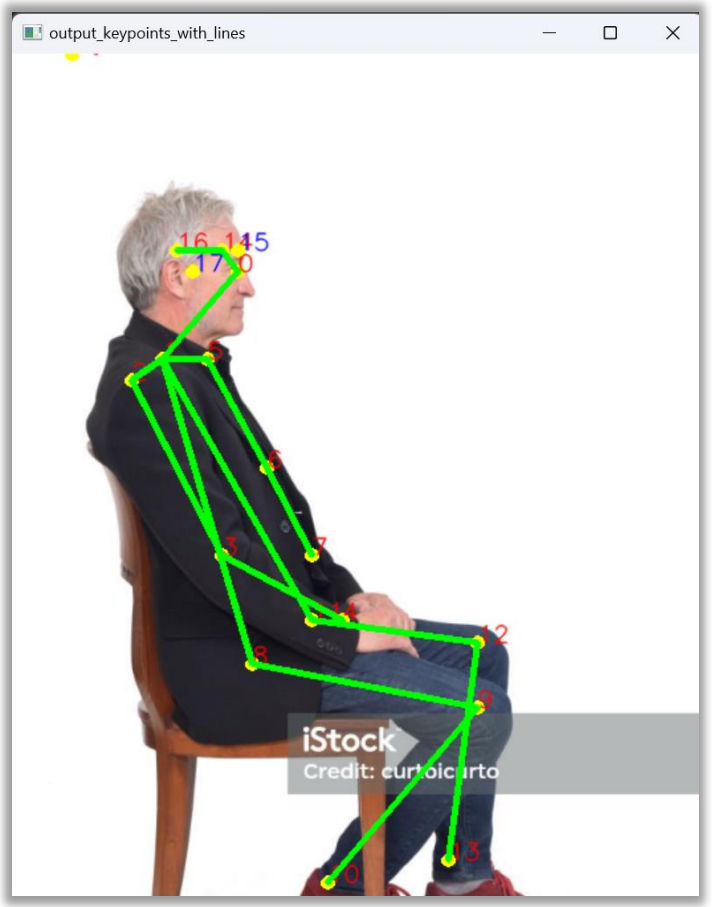
[기술 스택]



CONTENTS

[테스트 시연]





File Edit Selection View Go Run ··· opensource_testcode

EXPLORER

OPENSOURCE_TESTCODE

- openpose-master
- public
- pose_deploy_linevec_faster_4_stages.protot...
- pose_iter_160000.caffemodel
- pose_test1.py
- pose_test2-0.py
- pose_test2-1.py
- pose_test2-2.py
- pose_test2-3.py
- test.py

OUTLINE

TIMELINE

pose_test1.py | pose_test2-0.py | pose_test2-1.py

pose_test2-1.py > ...

```python
# 직선 0-1 과 1-8 사이의 각도 (angle1), 직선 0
import cv2
import math

# 각도 계산 함수
def calculate_angle(x1, y1, x2, y2, x3, y3):
    # 두 점 (x1, y1)와 (x2, y2) 사이의 벡터
    vector1 = (x2 - x1, y2 - y1)
    # 두 점 (x2, y2)와 (x3, y3) 사이의 벡터
    vector2 = (x3 - x2, y3 - y2)

    # 벡터의 내적을 계산
    dot_product = vector1[0] * vector2[0] +

    # 벡터의 크기를 계산
    magnitude1 = math.sqrt(vector1[0] ** 2 +
    magnitude2 = math.sqrt(vector2[0] ** 2 +

    # 두 벡터의 각도를 라디안에서 계산
```

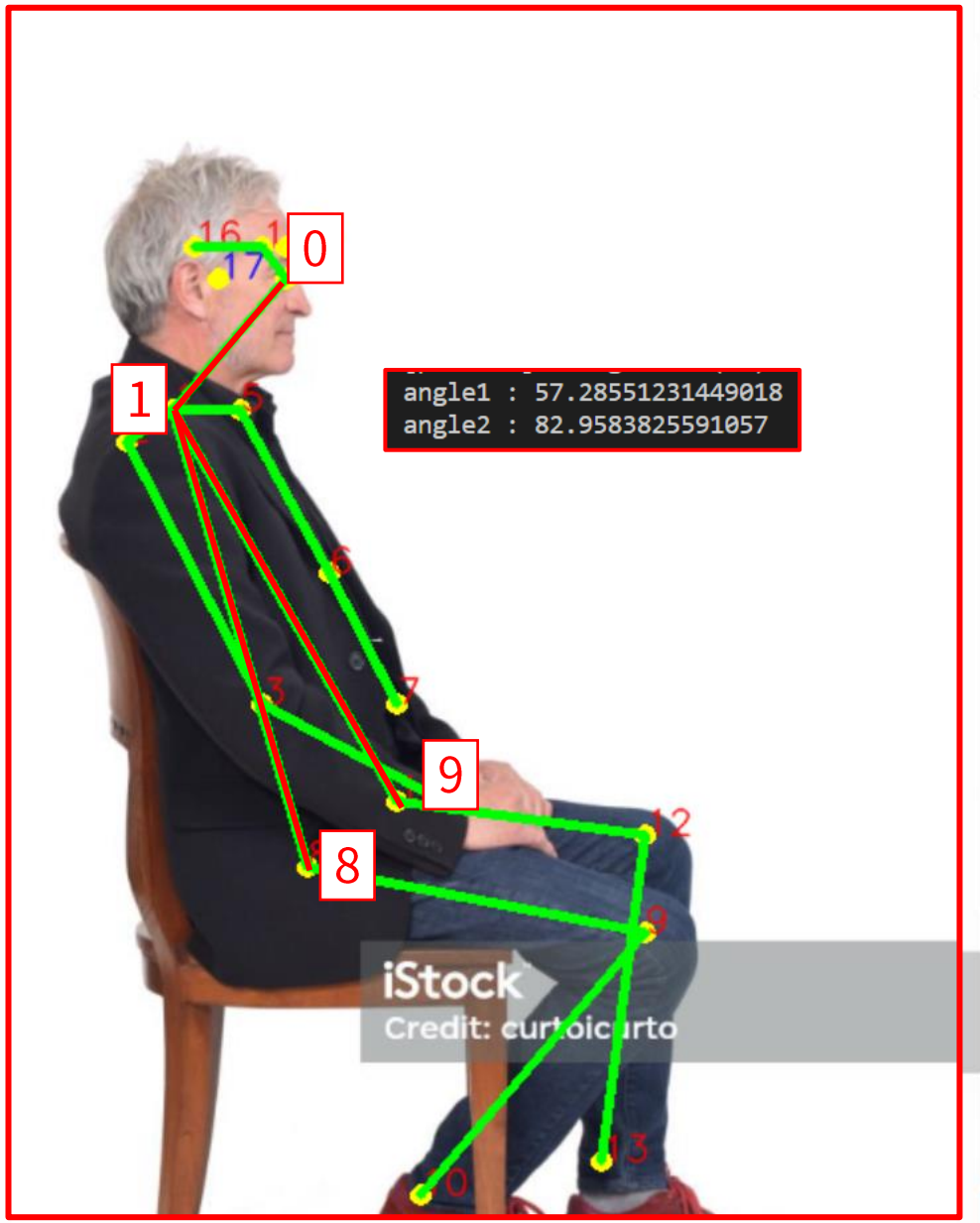
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[pointed] REar (16) => prob: 0.67805 / x: 124 / y: 14
[not pointed] LEar (17) => prob: 0.00571 / x: 136 / y
[pointed] Background (18) => prob: 1.00029 / x: 45 /
angle1 : 57.28551231449018
angle2 : 82.9583825591057

[linked] 0 (170, 164) <=> 1 (113, 230)
[linked] 0 (170, 164) <=> 14 (158, 148)
[not linked] 0 (170, 164) <=> 15 None
[linked] 1 (113, 230) <=> 2 (90, 246)
```

⊗ 0 ⚠ 0 0

output_keypoints_with_lines



12°C 맑음

AI융합경진대회 예선

오후 6:07 2023-10-20

File Edit Selection View Go Run ··· opensource_testcode

EXPLORER

OPENSOURCE_TESTCODE

- openpose-master
- public
- pose_deploy_linevec_faster_4_stages.protot...
- pose_iter_160000.caffemodel
- pose_test1.py
- pose_test2-0.py
- pose_test2-1.py
- pose_test2-2.py
- pose_test2-3.py
- test.py

OUTLINE

TIMELINE

pose_test1.py | pose_test2-0.py | pose_test2-1.py

pose_test2-2.py > ...

```python
# 0(x1, y1), 1(x2, y2) 두 점을 알고 있을 때, y
import cv2
import math

# 각도 계산 함수
def calculate_angle(x1, y1, x2, y2):
    # 두 점 (x1, y1)와 (x2, y2) 사이의 벡터
    vector1 = (x2 - x1, y2 - y1)
    # y축 벡터
    vector2 = (1, 0)

    # 벡터의 내적을 계산
    dot_product = vector1[0] * vector2[0] +

    # 벡터의 크기를 계산
    magnitude1 = math.sqrt(vector1[0] ** 2 +
    magnitude2 = math.sqrt(vector2[0] ** 2 +

    # 두 벡터의 각도를 라디안에서 계산
```

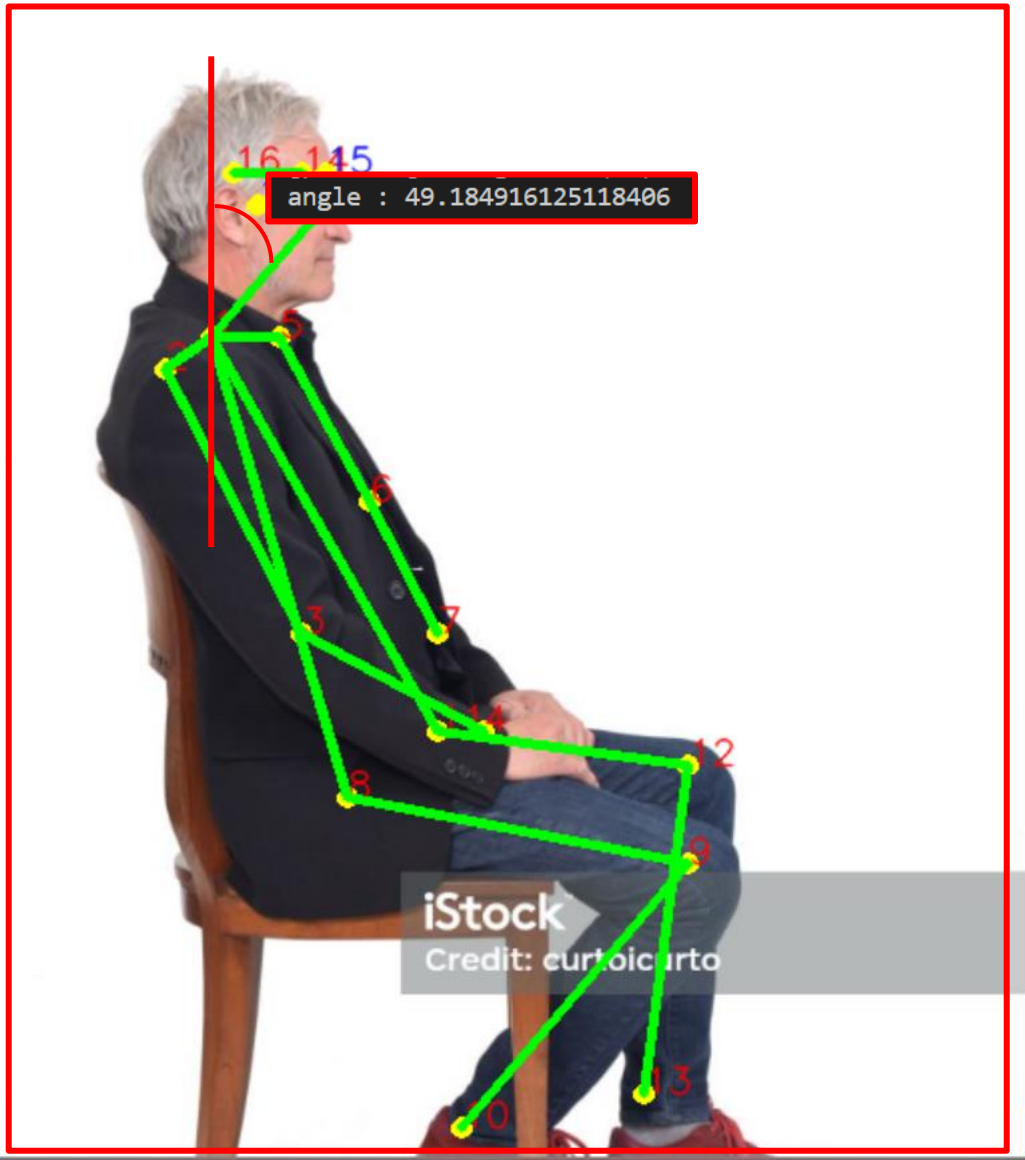
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[not pointed] LEye (15) => prob: 0.09958 / x: 170 / y
[pointed] REar (16) => prob: 0.67805 / x: 124 / y: 14
[not pointed] LEar (17) => prob: 0.00571 / x: 136 / y
[pointed] Background (18) => prob: 1.00029 / x: 45 /
angle : 49.184916125118406

[linked] 0 (170, 164) <=> 1 (113, 230)
[linked] 0 (170, 164) <=> 14 (158, 148)
[not linked] 0 (170, 164) <=> 15 None
[linked] 1 (113, 230) <=> 2 (90, 246)
```

⊗ 0 ⚠ 0 0

output_keypoints_with_lines



12°C 맑음

AI융합 경진대회 예선

오후 6:08 2023-10-20

File Edit Selection View Go Run ···

opensource_testcode

EXPLORER

OPENSOURCE_TESTCODE

- openpose-master
- public
- pose_deploy_linevec_faster_4_stages.protot...
- pose_iter_160000.caffemodel
- pose_test1.py
- pose_test2-0.py
- pose_test2-1.py
- pose_test2-2.py
- pose_test2-3.py
- test.py

OUTLINE

TIMELINE

pose_test1.py | pose_test2-0.py | pose_test2-1.py

pose_test2-3.py > ...

```python
# 재민쿤 아이디어
# 코(0)와 목(1) 사이의 거리가 초기에 입력된 바른
import cv2
import math

# 길이 계산 함수
def calculate_length(x1, y1, x2, y2):
    length = math.sqrt((x2-x1)**2 + (y2-y1)**
    return length

def output_keypoints(frame, proto_file, weigh
    global points

    # 네트워크 불러오기
    net = cv2.dnn.readNetFromCaffe(proto_file

    # 입력 이미지의 사이즈 정의
    image_height = 368
```

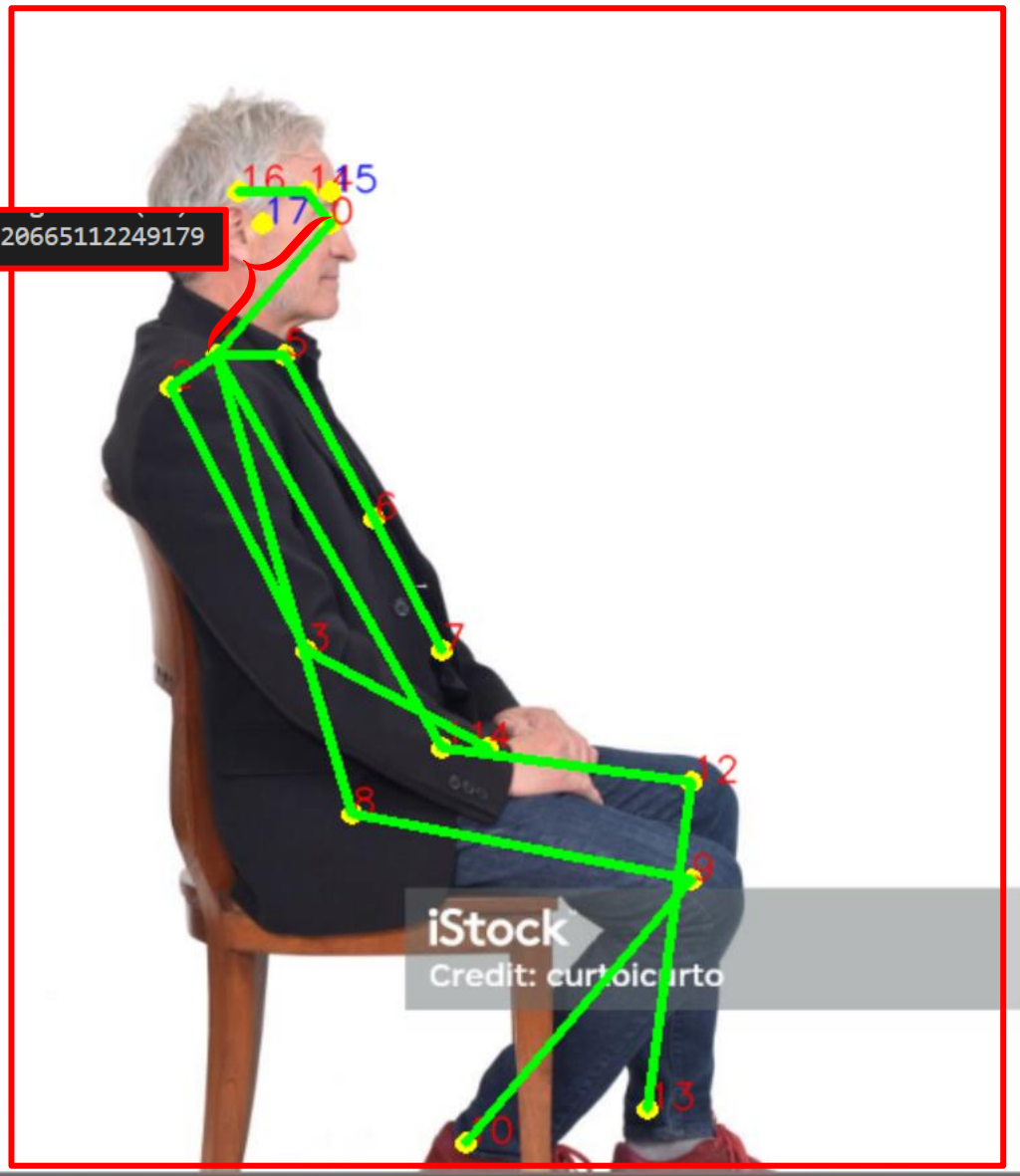
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[pointed] REye (14) => prob: 0.86117 / x: 158 / y: 14
[not pointed] LEye (15) => prob: 0.09958 / x: 170 / y
[pointed] REar (16) => prob: 0.67805 / x: 124 / y: 14
[not pointed] LEar (17) => prob: 0.00571 / x: 136 / y
[pointed] Background (18) => prob: 1.00029 / x: 45 /
length : 87.20665112249179

[linked] 0 (170, 164) <=> 1 (113, 230)
[linked] 0 (170, 164) <=> 14 (158, 148)
[not linked] 0 (170, 164) <=> 15 None
```

output_keypoints_with_lines

length : 87.20665112249179



0 0 0

12°C 맑음

AI융합경진대회 예선

오후 6:09 2023-10-20

참고 문헌

1. https://azure.microsoft.com/ko-kr/resources/cloud-computing-dictionary/what-is-computer-vision/#%EA%B0%9C%EC%B2%B4-%EB%B6%84%EB%A5%98
2. https://www.tensorflow.org/lite/examples/pose_estimation/overview?hl=ko
3. https://github.com/CMU-Perceptual-Computing-Lab/openpose
4. https://opencv.org/
5. https://www.amc.seoul.kr/asan/healthinfo/disease/diseaseDetail.do?contentId=31866
6. https://bkshin.tistory.com/entry/%EC%BB%B4%ED%93%A8%ED%84%B0-%EB%B9%84%EC%A0%84-12-%EC%9E%90%EC%84%B8-%EC%B6%94%EC%A0%95Pose-Estimation%EC%9D%B4%EB%9E%80
7. https://hanryang1125.tistory.com/2
8. https://m.blog.naver.com/rhrkdfus/221531159811