



TEAM FACE DETECTIVE

— 최종발표

Assistance Program for Monitor User

목차

- | | |
|--------------|------------|
| 1 주제 및 선정 이유 | 5 거북목 파트 |
| 2 주차별 계획 | 6 졸음 방지 파트 |
| 3 역할 분담 | 7 최종 결과물 |
| 4 구현 방법 | 8 기대효과 |

주제 및 선정이유 01

충남일보 · 2023.04.21.

[의료칼럼] 직장인 허리·목 건강 경고등

직장인들의 허리 건강에 경고등이 켜졌다. 건강보험심사평가원 지난해 허리디스크 환자는 197만5853명으로 약 200만 명에 육리디스크는... 업무 시에는 책상 위 PC 모니터를 너무 높거나 낮

헬스조선 PICK · 2023.01.17. · 네이버뉴스

안구건조증은 참는다? 방치했다간...

PC, 휴대전화 등 각종 전자기기를 사용하는 현대인에게 안구건조증은 때려야 뗄 수 없는 병이다. 우리나라 성인의 75% 이상이 앓고 있을 정도로 흔하고... 하지만 안구건조증은 생각보다 눈 건강에 위협적이다. 원인 다...



진료 환자 증가 ▲

목·허리 디스크는 디스크 내부 수핵이 뼈 틈으로 빠져나와 신경을 누르면 통증이 발생합니다.



출처: 건강보험심사평가원

출처: 건강보험심사평가원

직장인 고질병인
눈 피로, 졸음 및 거북목

대부분의 직장인이
모니터를 장시간 사용하는 것에서 착안해
모니터에서 효과적으로 동작될 프로그램을 기획

주제 및 선정이유 01

ASSISTACNE PROGRAM FOR MONITOR USER

1. 눈의 졸음 및 피로를 감지
→ 미로 게임을 통한 눈 운동

2. 사용자의 자세를 관찰
→ 자세가 흐트러지면 경고 알림

3. 자세 촬영 및 평가



주차별 진행 상황 02

8주차



START!

- 주제 선정
- 구현 기능 논의
- 선행기술 및 Library 조사

9주차



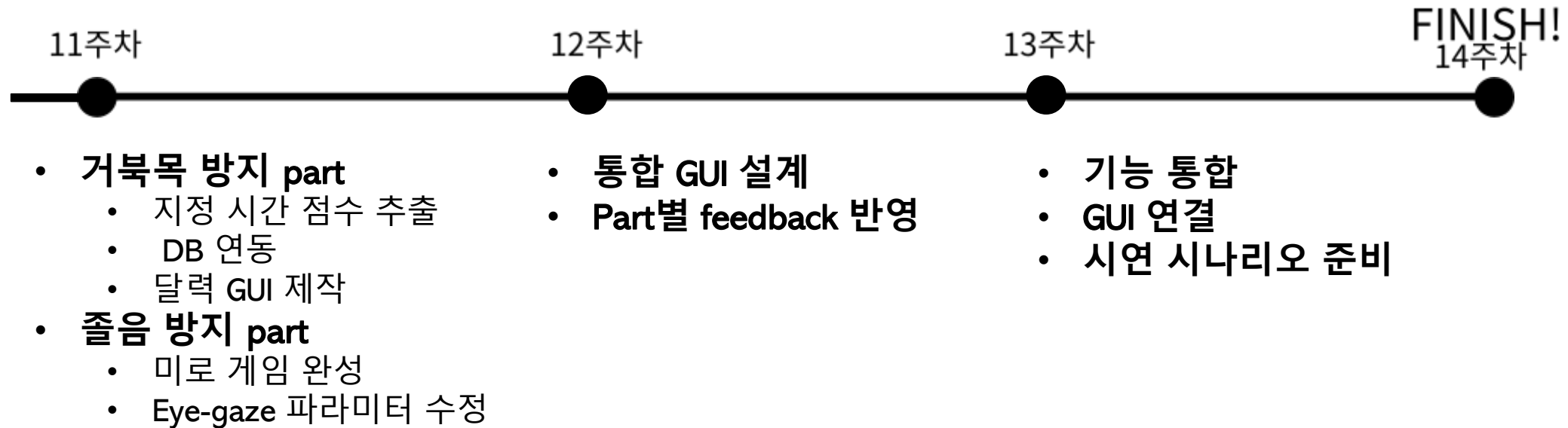
- 전체적 시나리오 구상
- 기능별 역할 분담

10주차



- 거북목 방지 part
 - 목과 어깨 keypoint
 - 거북/정상 목 구분
- 졸음 방지 part
 - 눈동자 데이터 수집
 - 파라미터 조정
 - 게임 시나리오 구성

주차별 진행 상황 02



역할분담 03

목경서

- Eye-Tracking 데이터 전처리 및 라벨링
- 줄음 감지 모델 pt 파일 생성
- 줄음감지, 미로 코드 통합

이다온

- 거북목 알람 및 UI 수정
- DB 연결
- 점수 캘린더 연동
- 전체 GUI 제작

최세인

- 프로그램 설명 이미지 제작
- 미로기능 구현, 디자인
- 줄음감지 2단계 구성하여 알림 추가
- 줄음감지, 미로 코드 통합

장수민

- 거북/정상 목 단계별 구분
- 사용자별 거북/정상 목 촬영
- 지정 시간 점수 추출
- 시연용 GUI 제작

구현 방법 04

졸음 방지

- dlib을 이용한 얼굴 랜드마크 인식으로 눈 감지
- EAR 알고리즘을 이용한 눈 깜빡임 측정
- 2단계로 나누어 졸음 알람
- Yolo5x 모델을 이용하여 눈동자 객체 인식 (상하좌우 판단)
- 눈동자 움직임으로 미로 해결 및 눈 운동

거북 목 방지

- OpenPose MPI모델로 관절 추출
- 목과 양쪽 어깨 정보만 사용
- 어깨길이 / 목 측정
- 일정 값 이상이면 거북목 판단
- Mysql 사용하여 자세 점수 저장
- pyQT로 달력 및 날짜 별 점수 공개

구현 방법 - 졸음감지 05



구현 방법 - 졸음감지 05

```
def eye_aspect_ratio(eye):          # EAR 계산
    A = distance.euclidean(eye[1], eye[5])
    B = distance.euclidean(eye[2], eye[4])
    C = distance.euclidean(eye[0], eye[3])
    ear = (A + B) / (2.0 * C)
    return ear
```

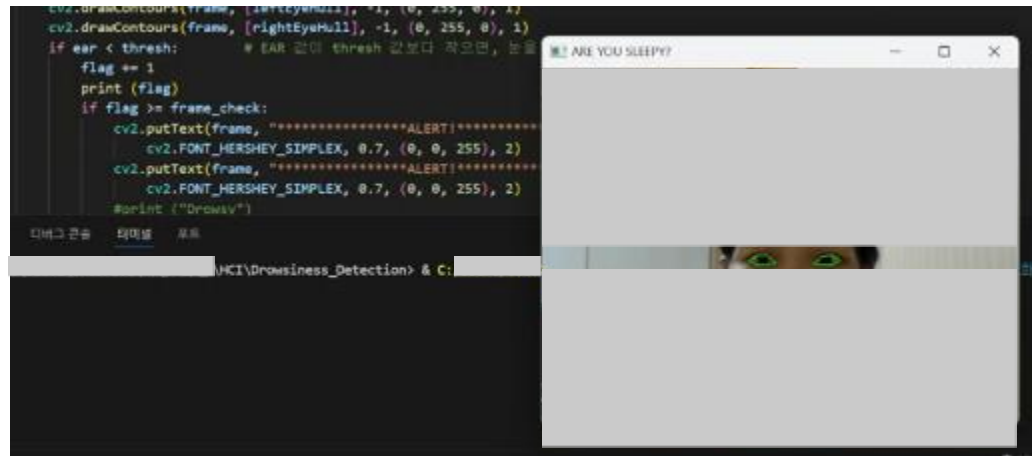
$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

EAR: 눈의 폭과 높이 사이의 비율
각 눈을 6개의 (x, y) 좌표로 표현해 비율을 계산합니다.
좌표는 눈의 왼쪽 모서리에서 시작하여 시계 방향으로
눈 주위를 따라갑니다.

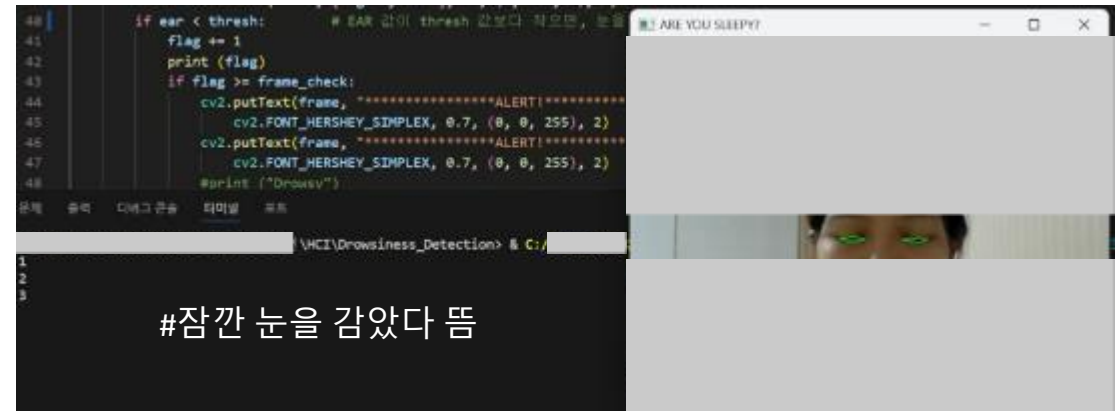
```
if ear < thresh:          # EAR 값이 thresh 값보다 작으면, 눈을 감은 것으로 인식
    flag += 1
    print(flag)
    if flag >= frame_check:
        cv2.putText(frame, "*****ALERT!*****", (10, 30),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        cv2.putText(frame, "*****ALERT!*****", (10, 325),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
```

30개의 연속된 프레임을 확인하여 EAR 값이
0.25(thresh)보다 작으면 경고문구가 보여집니다.
EAR이 작을수록 눈이 감기거나 감은 것으로 간주합니다.

구현 방법 - 졸음감지 05

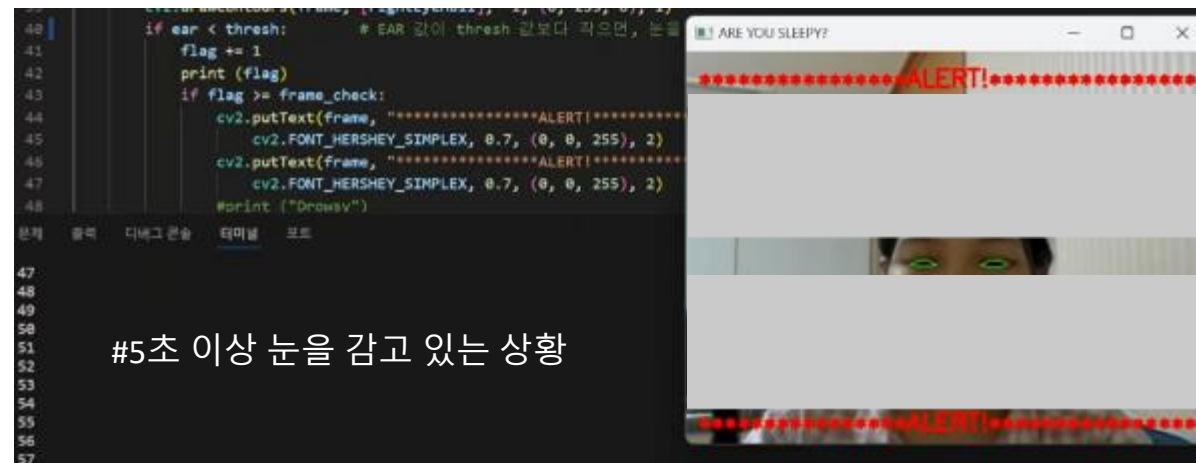


실행 후 눈을 감지하여 위치를 표시



#잠깐 눈을 감았다 뜸

눈을 감는 순간부터 밀리초 단위로 카운트 시작



#5초 이상 눈을 감고 있는 상황

눈이 일정시간 이상 감길 시 화면에 ALERT 문구 띄움

구현 방법 - eye tracking+미로

05

```
# 왼쪽 눈동자의 위치 비율
left_x_iris_center = (left_part[1]['xmin'] + left_part[1]['xmax']) / 2
left_x_per = (left_x_iris_center - left_part[0]['xmin']) / (left_part[0]['xmax'] - left_part[0]['xmin'])
left_y_iris_center = (left_part[1]['ymin'] + left_part[1]['ymax']) / 2
left_y_per = (left_y_iris_center - left_part[0]['ymin']) / (left_part[0]['ymax'] - left_part[0]['ymin'])

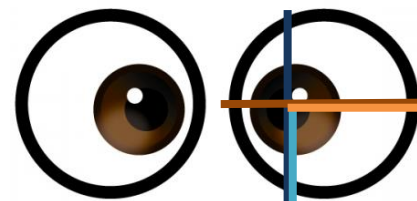
# 오른쪽 눈동자의 위치 비율
right_x_iris_center = (right_part[1]['xmin'] + right_part[1]['xmax']) / 2
right_x_per = (right_x_iris_center - right_part[0]['xmin']) / (right_part[0]['xmax'] - right_part[0]['xmin'])
right_y_iris_center = (right_part[1]['ymin'] + right_part[1]['ymax']) / 2
right_y_per = (right_y_iris_center - right_part[0]['ymin']) / (right_part[0]['ymax'] - right_part[0]['ymin'])

# 왼쪽 눈동자와 오른쪽 눈동자 비율의 평균
avr_x_iris_per = (left_x_per + right_x_per) / 2
avr_y_iris_per = (left_y_per + right_y_per) / 2
```

```
# Threshold 기준으로 눈동자의 위치를 계산
if avr_x_iris_per < (0.5 - iris_x_threshold):
    iris_status = 'Left'
    print("Left : ((" , avr_x_iris_per < (0.5 - iris_x_threshold), ")") , "
    move()
elif avr_x_iris_per > (0.5 + iris_x_threshold):
    iris_status = 'Right'
    print("Right : ((" , avr_x_iris_per > (0.5 + iris_x_threshold), ")") , "
    move()
elif avr_y_iris_per > (0.5 - iris_y_threshold):
    elif avr_y_iris_per > (0.6):
        iris_status = 'Up'
        print("Up : ((" , avr_y_iris_per > (0.6), ")") , "avr_x_iris_per : " , a
        move()
    else:
        iris_status = 'Center'
        print("Center 에서 Up : ((" , avr_y_iris_per > (0.6 - iris_y_threshold
        print("Center : " , "avr_x_iris_per : " , avr_x_iris_per , "iris_x_thres
elif len(eye_list) == 2 and len(iris_list) == 0:
    iris_status = 'Down'
    #print("Down : " , "avr_x_iris_per : " , avr_x_iris_per , "iris_x_threshold
    move()
```

Left/Right_x_per = $\frac{\text{눈동자의 중심과 눈의 오른쪽까지의 길이}}{\text{눈의 가로 전체 길이}}$

Left/Right_y_per = $\frac{\text{눈동자의 중심과 눈의 아래까지의 길이}}{\text{눈의 세로 전체 길이}}$



눈의 비율을 기준으로 계산,
사용자 맞춤 인식이 가능

Avr_x_iris_per : 왼쪽과 오른쪽의 x_per 평균

Avr_y_iris_per : 왼쪽과 오른쪽의 y_per 평균

Iris_x_threshold , Iris_y_threshold : 사전에 정의한 임계값, 0.15, 0.26

If: 눈과 눈동자가 전부 인식되는 경우

Avr_x_iris_per < (0.5-threshold) => 왼쪽

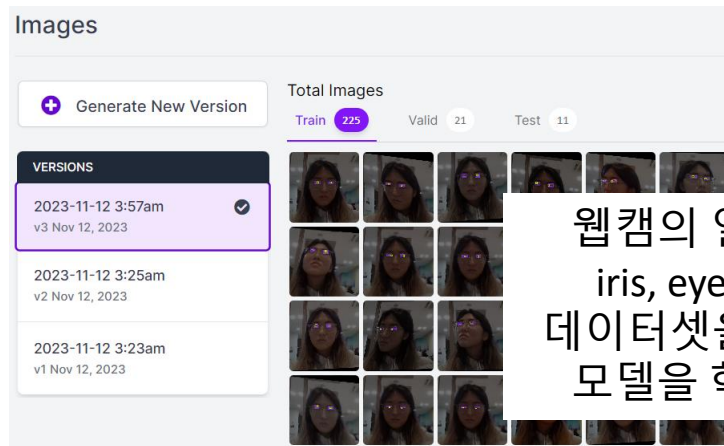
Avr_x_iris_per > (0.5+threshold) => 오른쪽

Avr_y_iris_per > (0.6) => 위쪽

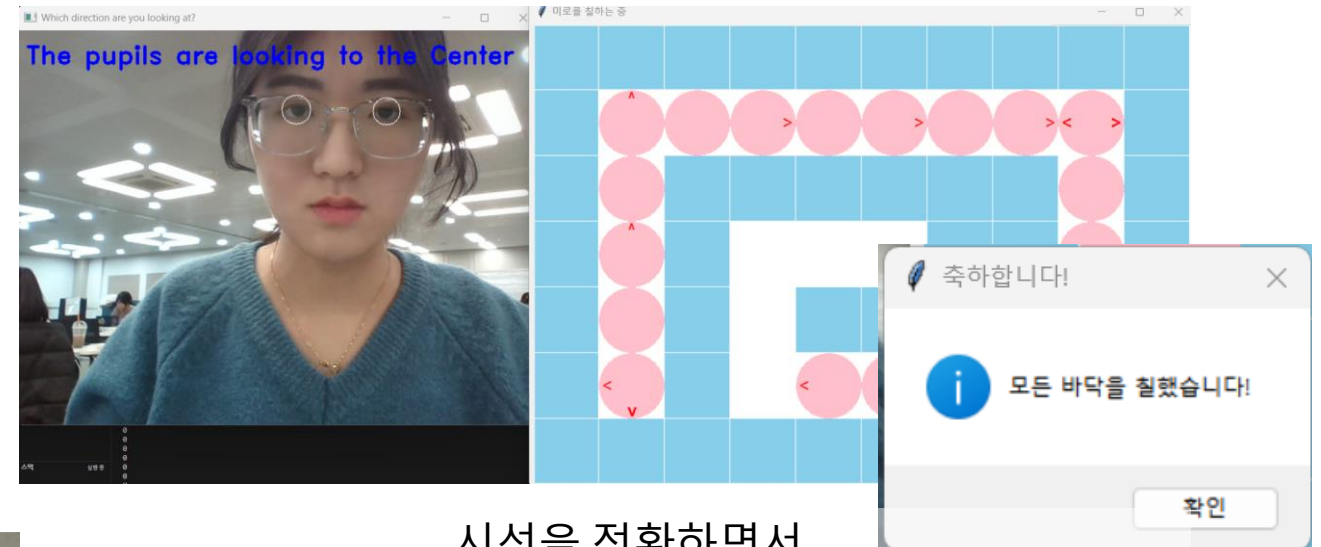
나머지의 경우 => 중앙

Elif: 눈은 인식이 되나 눈동자가 인식되지 않은 경우 => 아래

구현 방법 - eye tracking+미로 05



웹캠의 얼굴을 캡처하여
iris, eye를 레이블링한
데이터셋을 활용하여 YOLO
모델을 학습시켰습니다.

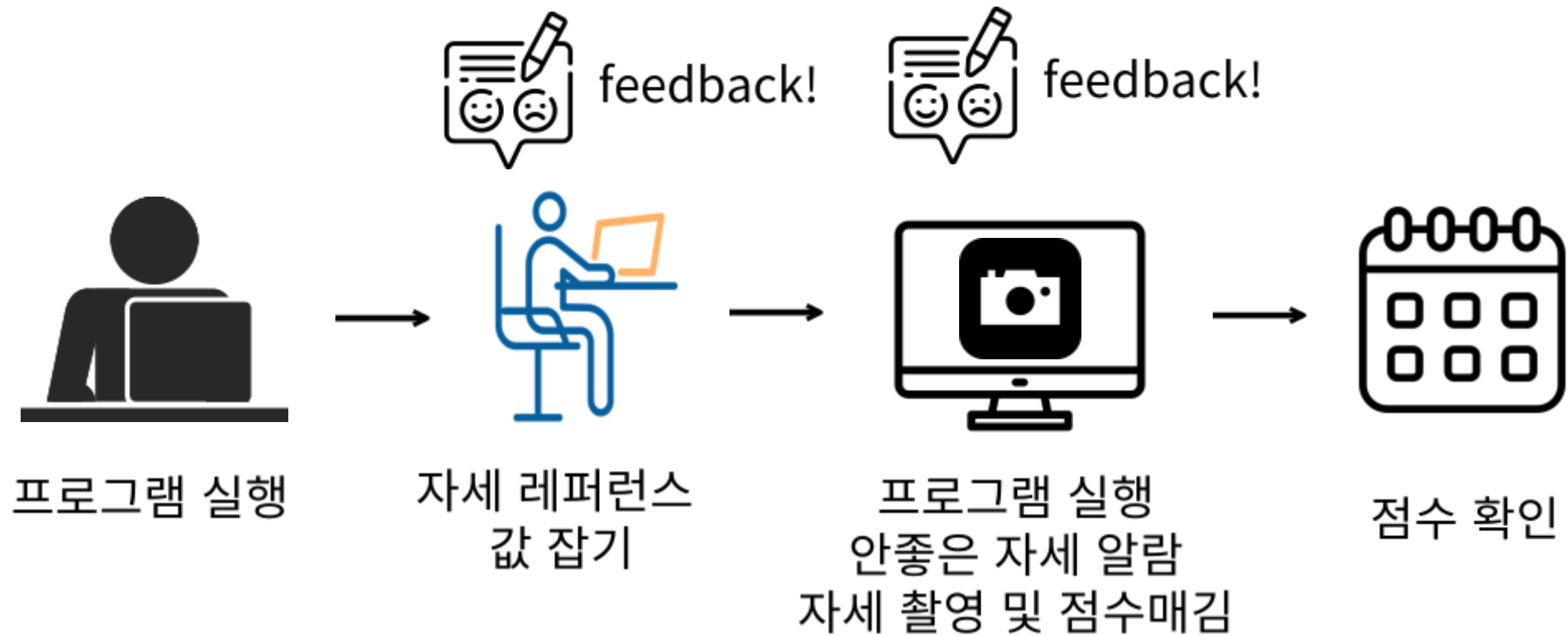


시선을 전환하면서
미로찾기 게임을 진행합니다.
흰 색 경로를 따라 분홍색을 다 칠하면 게임이 종료됩니다.






웹캠은 좌우반전으로 표현됩니다.
상하좌우중앙을 잘 구분하는 모습을 보입니다.

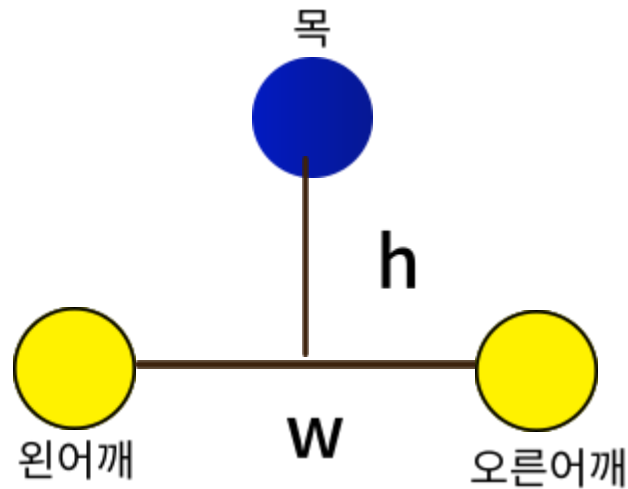
구현 방법 - 자세 06



구현 방법 - 자세 06



	이름	7
 pose_deploy_linevec.prototxt		2
 pose_deploy_linevec_faster_4_stages.prot...		2
 pose_iter_160000.caffemodel		2



openpose 사용

목
오른 어깨
왼 어깨 포인트만 추출해
 w/h 비율을 사용

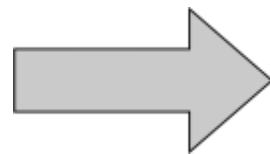
구현 방법 - 자세 06



정상자세 reference 값 촬영



비정상자세 reference 값 촬영



Result Grid			
	id	good	bad
▶	1	4	6
	3	2	8

DB 저장!

구현 방법 - 자세 06

reference 값 추출하기



feedback!

```
x = abs(points[1][0] - points[2][0]) # 밑변
h = abs(points[0][1] - (points[1][1] + points[2][1]) // 2) # 높이
r = x / h # 비율

step = (ext_ratio - ord_ratio)/4

if r <= ord_ratio:
    cv2.line(frame, points[0], points[1], color: (255, 0, 0), thickness: 2)
    cv2.line(frame, points[0], points[2], color: (255, 0, 0), thickness: 2)
    score = 100
elif r <= ord_ratio+step:
    cv2.line(frame, points[0], points[1], color: (0, 255, 0), thickness: 2)
    cv2.line(frame, points[0], points[2], color: (0, 255, 0), thickness: 2)
    score = 80
elif r <= ord_ratio + 2*step:
    cv2.line(frame, points[0], points[1], color: (0, 255, 255), thickness: 2)
    cv2.line(frame, points[0], points[2], color: (0, 255, 255), thickness: 2)
    score = 60
elif r <= ord_ratio + 3*step:
    cv2.line(frame, points[0], points[1], color: (0, 165, 255), thickness: 2)
    cv2.line(frame, points[0], points[2], color: (0, 165, 255), thickness: 2)
    score = 40
else:
    cv2.line(frame, points[0], points[1], color: (0, 0, 255), thickness: 2)
    cv2.line(frame, points[0], points[2], color: (0, 0, 255), thickness: 2)
    score = 20
```

사용자의 좋은 자세와
안 좋은 자세의 비율을 4로 나눠
step 값으로 사용

ex) 좋은 자세 ratio가 3
안좋은 자세 ratio 가 7
step 값은 1

ratio = 3	-> 100점 할당
3 + 1	-> 80점 할당
3 + 1*2	-> 60점 할당
3 + 1*3	-> 40점 할당
else	-> 20점 할당

구현 방법 - 자세 06

```
def isTriangle(points):
    if points[0] and points[1] and points[2]: # 3점 다 찍힘
        if (points[1][0] < points[0][0]) and (points[0][0] < points[2][0]): # x좌표가 RNL 순
            if points[0][1] < min(points[1][1], points[2][1]): # y 좌표가 NOI 작음
                return True
    return False
```

isTriangle() : 올바른 좌표 위치 파악

```
usage = 4000000
def isTime():
    global ind
    curr_hour = datetime.today().hour
    curr_min = datetime.today().minute

    res_hour = resv_time[ind][0]
    res_min = resv_time[ind][1]
    global prev_hour, prev_min

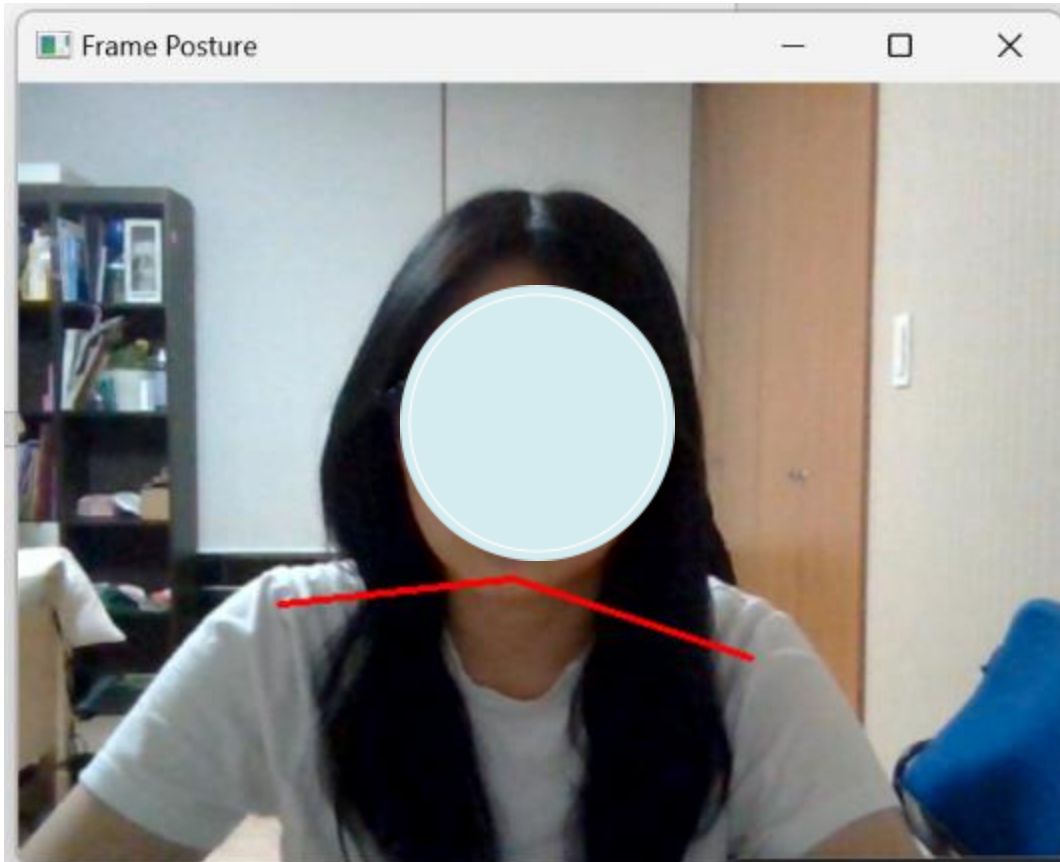
    if (prev_hour == res_hour) and (prev_min == res_min):
        if (curr_hour != res_hour) or (curr_min != res_min):
            ind += 1
    prev_hour, prev_min = curr_hour, curr_min

    if (curr_hour == res_hour) and (curr_min == res_min):
        if not visit[ind]:
            visit[ind] = 1
            return True
    return False
```

isTime() : 한 시간 간격으로 자세 촬영

```
#시간
#9:10, 10:10, 11:10..에 촬영
resv_time = [(9, 10), (10, 10), (11, 10), (12, 10), (13, 10), (14, 10), (15, 10), (16, 10), (17, 10), (18, 10)]
visit = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
ind = 0
curr_hour, curr_min = 0, 0
prev_hour, prev_min = 0, 0
res_hour, res_min = 0, 0
```

구현 방법 - 자세 06



1.2.4

Multidimensional data visualization across les. Explore relationships within and among



feedback!

자세가 좋지 않을 때
window 알람 띄움



Python

안 좋은 자세 알람
자세가 현재 굉장히 안 좋아요! 자세를 바로 세우세
요!

구현 방법 - 자세 06

	id	score	createdAt
	9	40	2023-12-01
	10	40	2023-12-01
	11	60	2023-12-01
	12	40	2023-12-01
	13	60	2023-12-01
	14	40	2023-12-01
	15	20	2023-12-01
	16	20	2023-12-01
	17	20	2023-12-01
	18	100	2023-12-01
	19	80	2023-12-01

수집된 점수의
평균값 사용

11월, 2023						
일	월	화	수	목	금	토
29	30	31	1	2	3	4 80.3점
5	6	7 56.7점	8 77.2점	9 45.8점	10 66.2점	11 20.0점
12	13 78.9점	14 78.0점	15	16 80.0점	17	18
19	20 44.8점	21	22 40.1점	23 54.5점	24	25
26	27 68.2점	28 50.0점	29	30 52.1점	1 61.5점	2
3	4	5	6	7	8	9

최종 결과물



feedback!

07

프로그램 설명서 첨부



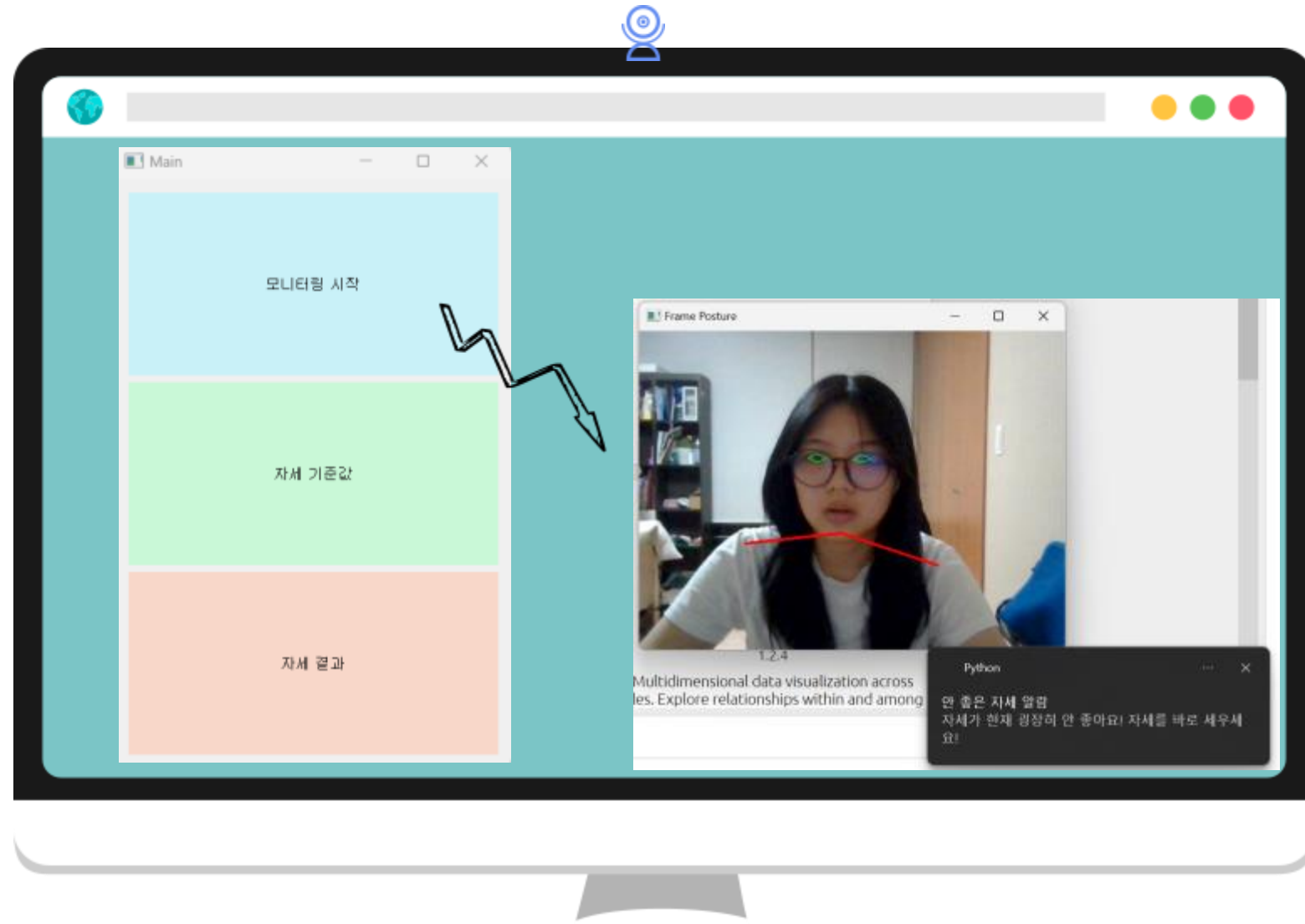
최종 결과물



feedback!

07

- 졸음감지와 자세 측정이 함께 실행됨.



기대 효과 08

작업 효율성 향상

장시간 컴퓨터 작업 시, 졸음이나 부적절한 자세로 인한 피로 감지

→ 알림 및 확인 → 작업 효율성 ↑



개인 건강 관리

자세 점수 캘린더 이용 → 스스로 자세 점수 체크