Action Recognition Fan

201433013 나희찬, 201433002 김동연

Action Recognition Fan

01

Action Recognition Fan

행동(동작) 인식 선풍기

02

Function - Ultrasonic Sensor

기능

초음파 센서

03

Function – Action Recognition

기능

행동(동작) 인식

04

Picture and Improvement

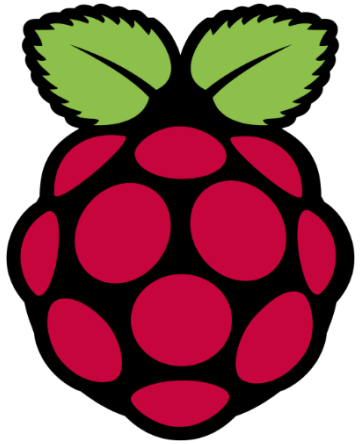
사진 및 개선사항

Action Recognition Fan

동작(행동) 인식 선풍기

- 초음파 센서를 통해 사용자와의 거리와 방향을 측정하여 사용자의 위치에 따라서 바람의 방향을 조절
- 사용자의 행동을 인식하여 바람의 세기를 조절
- 사무용 선풍기로 적합

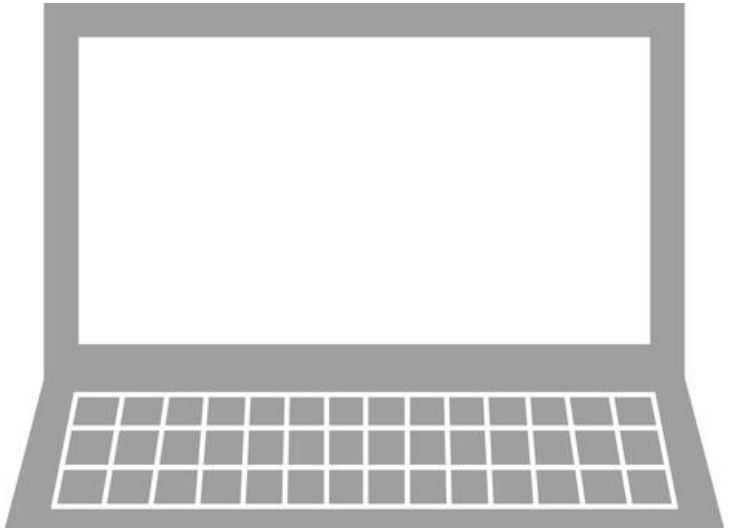
Action Recognition Fan



영상 데이터

ZMQ 소켓 통신

출력 데이터

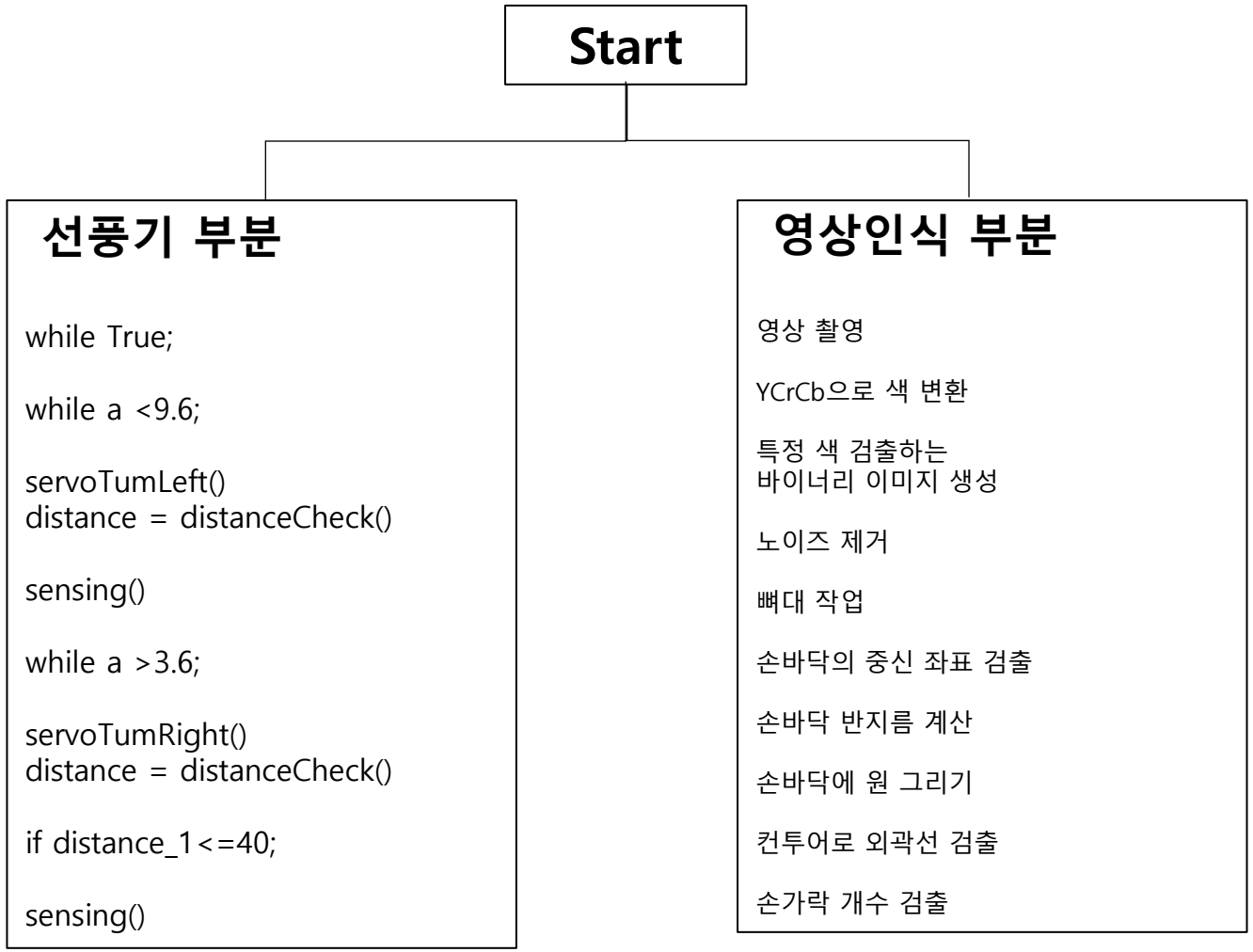


1, GPIO제어
(모터와 센서)

2. 영상촬영

openCV영상 데이터 처리
(동작 인식)

Action Recognition Fan



Main 스레드(선풍기 부분)와 부스레드(영상인식 부분)을 나눔

Function – Ultrasonic Sensor

초음파 센서

```
def sensing():
    state = 1
    print("sensing!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")

    while state:
        distance = distanceCheck()
        distance_1 = distance[0]
        distance_2 = distance[1]
        print("distance_1 : ", distance_1)
        print("distance_2 : ", distance_2)

        # 1 is closer
        if 2 < distance_2 - distance_1:
            print("1 is closer!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
            while distance_2 - distance_1 > 0.2:
                distance = distanceCheck()
                distance_1 = distance[0]
                distance_2 = distance[1]
                print("distance_1 : ", distance_1)
                print("distance_2 : ", distance_2)

                # out of 1 is close
                if distance_1 > 40:
                    print("out of 1 is closer!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
                    break

                # servo limit
                global a
                if 10.1 < a:
                    print("out of 1 servo!!!!!!!!!!!!!!!!!!!!!!!!!")
                    break
```

Function – Ultrasonic Sensor

초음파 센서

```
        servoTurnLeft_sensing()

distance = distanceCheck()
distance_1 = distance[0]
distance_2 = distance[1]

# 2 is closer
if 2 < distance_1 - distance_2:
    print("2 is closer!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
    while distance_1 - distance_2 > 0.2:
        distance = distanceCheck()
        distance_1 = distance[0]
        distance_2 = distance[1]
        print("distance_1 : ", distance_1)
        print("distance_2 : ", distance_2)

        # out of 2 is close
        if distance_2 > 40:
            print("out of 2 is closer !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
            break

        # servo limit
        if a < 2.9:
            print("out of 2 servo!!!!!!!!!!!!!!!!!!!!!!!!!!!")
            break

        servoTurnRight_sensing()

# sensing out
if distance_1 > 40:
    print("sensing out!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
```

Function – Action Recognition

동작(행동) 인식

```
import cv2 as cv
import numpy as np

cap = cv.VideoCapture(0)
cap.set(cv.CAP_PROP_FRAME_WIDTH, 640)
cap.set(cv.CAP_PROP_FRAME_HEIGHT, 420)

while True:
    ret, img_mycam = cap.read()
    if ret == False:
        break
    height, width = img_mycam.shape[:2]

    #색 변환
    img_ycrcb = cv.cvtColor(img_mycam, cv.COLOR_BGR2YCrCb)

    # 살색영역 구분하는 곳
    lower_skin = (0, 133, 77)
    upper_skin = (255, 173, 140)
    #바이너리 이미지 생성
    img_mask = cv.inRange(img_ycrcb, lower_skin, upper_skin)

    # 노이즈 없애주기(morphologyEx Opening & Closing)
    kernel2 = np.ones((10, 10), np.uint8)
    img_mask = cv.morphologyEx(img_mask, cv.MORPH_OPEN, kernel2)
    img_mask = cv.morphologyEx(img_mask, cv.MORPH_CLOSE, kernel2)
```

Function – Action Recognition

동작(행동) 인식

```
#일반화면에서 살색만 보여주기
img_result = cv.bitwise_and(img_mycam, img_mycam, mask = img_mask)

# 뼈대 보여주기
dist_transform = cv.distanceTransform(img_mask, cv.DIST_L2, 5)
dist_result = cv.normalize(dist_transform, None, 255, 0, cv.NORM_MINMAX, cv.CV_8UC1)
#손바닥까지의 반지름 구하기( maxVal가 반지름임(중앙에서부터 배경까지 가장 가까운 거리))
minVal, maxVal, minLoc, maxLoc = cv.minMaxLoc(dist_transform)
radius = maxVal
#손바닥 중심 좌표 구하기
centerX, centerY = int(maxLoc[0]), int(maxLoc[1])

cv.circle(img_mycam, (centerX, centerY), int(radius * 1.9), (0, 255, 0), 2)  # 손바닥 영역 표시
cv.circle(img_mycam, (centerX, centerY), 2, (0, 0, 255), 10)# 손바닥 중심 표시

#손가락 개수 검출`
# 바이너리 이미지 하나 더 생성(손가락 검출용)
ret, img_mask_finger = cv.threshold(img_mask, 100, 255, cv.THRESH_BINARY)
# 새로운 바이너리 이미지에 손바닥 동그라미 그려서 내부까지 다 칠하기
cv.circle(img_mask_finger, (centerX, centerY), int(radius * 1.7), (0, 0, 255), -1)

#새로운 바이너리 이미지에 컨투어로 외각선 따기
contours, _ = cv.findContours(img_mask_finger, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)

# 외곽선이 없을 때 == 손 검출 X
if contours == 0:
    break
```

Function – Action Recognition

동작(행동) 인식

```
#손가락개수 = 그려지는컨투어개수 - 1(손목)
fingerCount = len(contours) - 1
print("fingerCount : ", fingerCount)

# 화면에 손가락 개수 표시
location = (0, 50)
font = cv.FONT_HERSHEY_SIMPLEX
fontScale = 2
if fingerCount == 1:
    cv.putText(img_mycam, '1', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
elif fingerCount == 2:
    cv.putText(img_mycam, '2', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
elif fingerCount == 3:
    cv.putText(img_mycam, '3', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
elif fingerCount == 4:
    cv.putText(img_mycam, '4', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
elif fingerCount == 5:
    cv.putText(img_mycam, '5', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
else:
    cv.putText(img_mycam, '0', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)


key = cv.waitKey(1)
if key == 27:
    break

cv.imshow("mycam", img_mycam) #영상 출력
cv.imshow("img_mask", img_mask) #바이너리 이미지로 보여줌
cv.imshow("img_result", img_result) #살색영역만 보여줌
cv.imshow("dist_result", dist_result) #뼈대 보여줌
```

Function – Socket Communication

소켓통신

```python
import cv2 as cv
import imagezmq
import numpy as np

image_hub = imagezmq.ImageHub()
sender = imagezmq.ImageSender(connect_to='tcp://192.168.0.10:5555')

power = "0"
src = cv.imread("powerpic.jpg")
print("server start")

while True:
    rpi_name, img_mycam = image_hub.recv_image()
    image_hub.send_reply(b'OK')

    sender.send_image(power, src)
```

Function – Socket Communication

소켓통신

```
# 화면에 손가락 개수 표시
location = (0, 50)
font = cv.FONT_HERSHEY_SIMPLEX
fontScale = 2
if fingerCount == 1:
    cv.putText(img_mycam, '1', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
    power = "power1"
elif fingerCount == 2:
    cv.putText(img_mycam, '2', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
    power = "power2"
elif fingerCount == 3:
    cv.putText(img_mycam, '3', location, font, fontScale, (0, 0, 255), 3, cv.LINE_AA)
    power = "power3"
```

Function – Socket(Client) + Thread

소켓(클라이언트) + 스레드

```
#########################################################################################################################
#thread

image_hub = imagezmq.ImageHub()
sender = imagezmq.ImageSender(connect_to='tcp://192.168.0.12:5555')

rpi_name = socket.gethostname() # send RPi hostname with each image

picam = VideoStream(usePiCamera=True).start()
time.sleep(2.0)  # allow camera sensor to warm up

flag_exit = False
def t1():
    while True:
        image = picam.read()
        sender.send_image(rpi_name, image)

        power, src = image_hub.recv_image()
        image_hub.send_reply(b'OK')
        # print("power : ", power)

thread = threading.Thread(target=t1)
thread.start()
#########################################################################################################################
```

Action Recognition Step

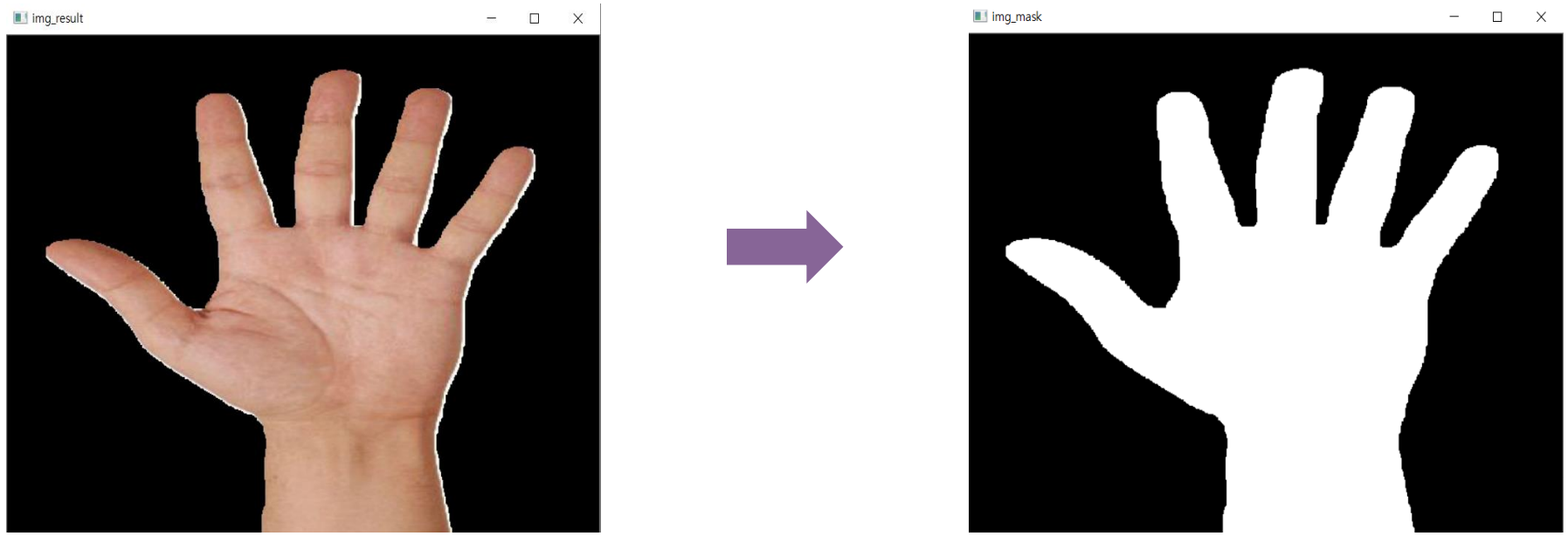
동작(행동) 인식 단계

- BRG 이미지를 Ycrcb 이미지로 변경
- 노이즈에 상대적으로 강해짐

Action Recognition Step

동작(행동) 인식 단계

- 이미지에서 살구색 영역만 추출 후 바이너리 이미지로 변경
- openCV의 모폴로지 연산으로 노이즈를 줄임



Action Recognition Step

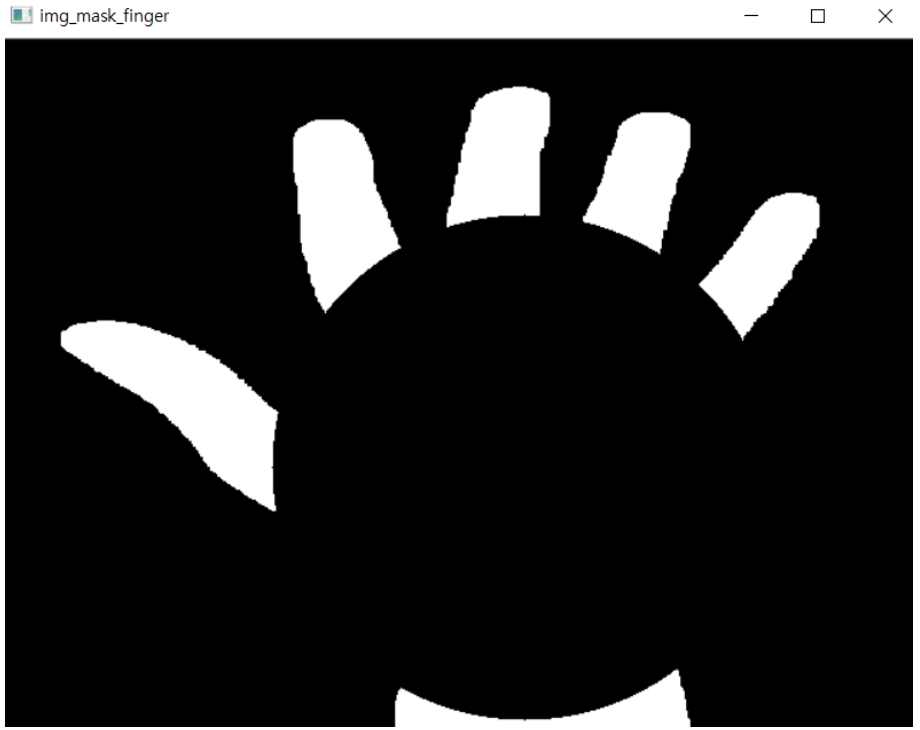
동작(행동) 인식 단계

- distanceTransform 함수를 이용하여 배경과 멀어질수록 픽셀 값이 커지는 화면을 출력
- 이 작업을 통해 손바닥의 중심과 손바닥의 반지름 값을 얻음

Action Recognition Step

동작(행동) 인식 단계

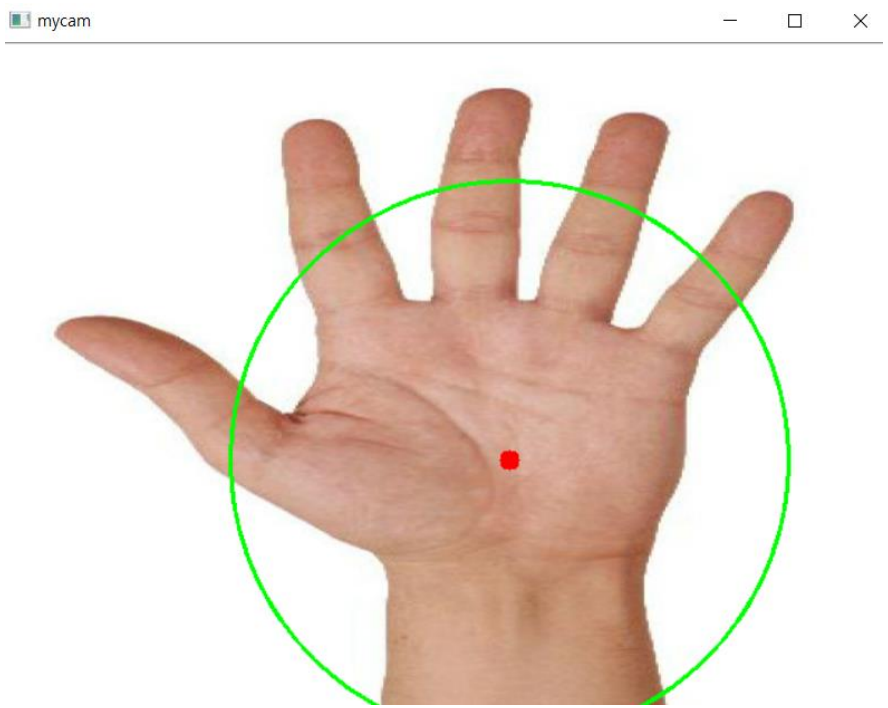
- 바이너리 이미지의 손바닥 중심에 원을 그린다.
- 컨투어로 외곽선을 추출



Action Recognition Step

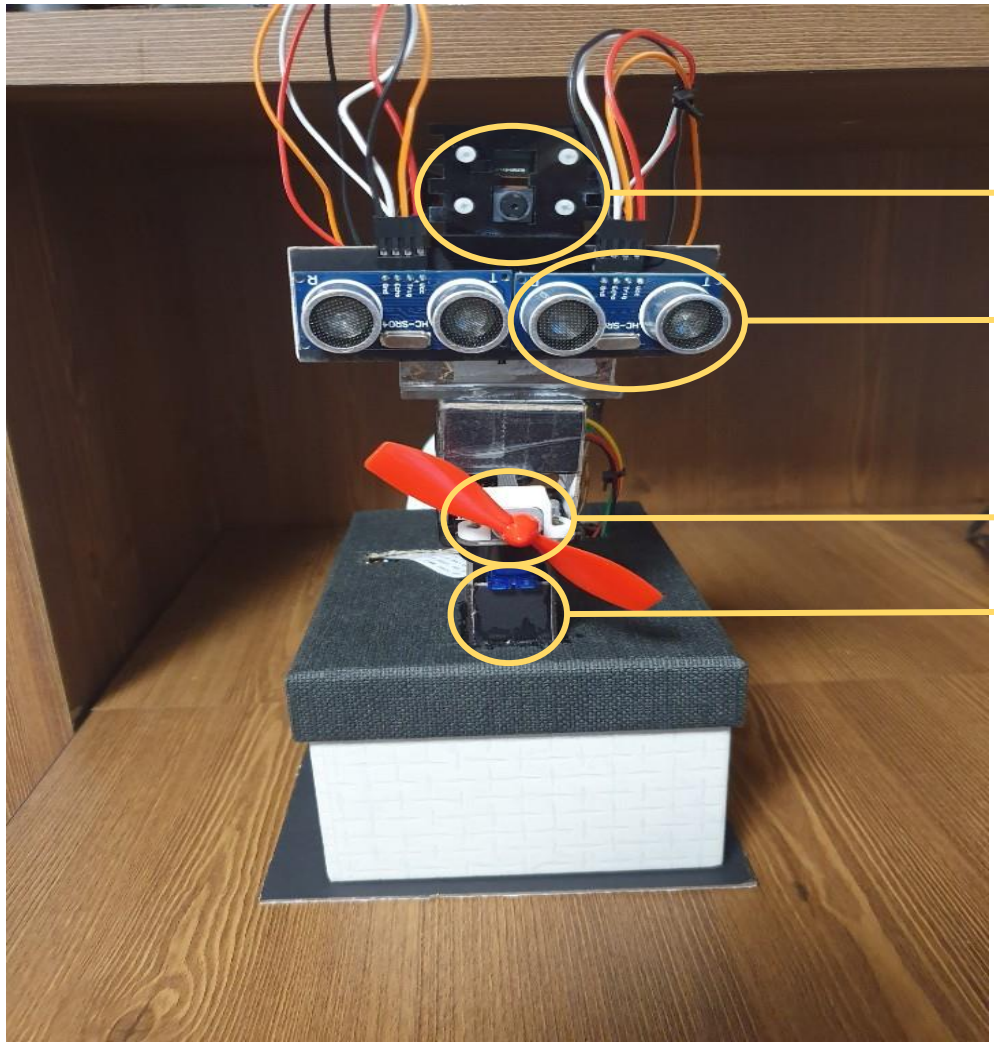
동작(행동) 인식 단계

- 검출된 컨투어의 개수에서 손목의 영역 -1
- 손가락 개수를 검출



Action Recognition Fan - Picture

사진



카메라 모듈

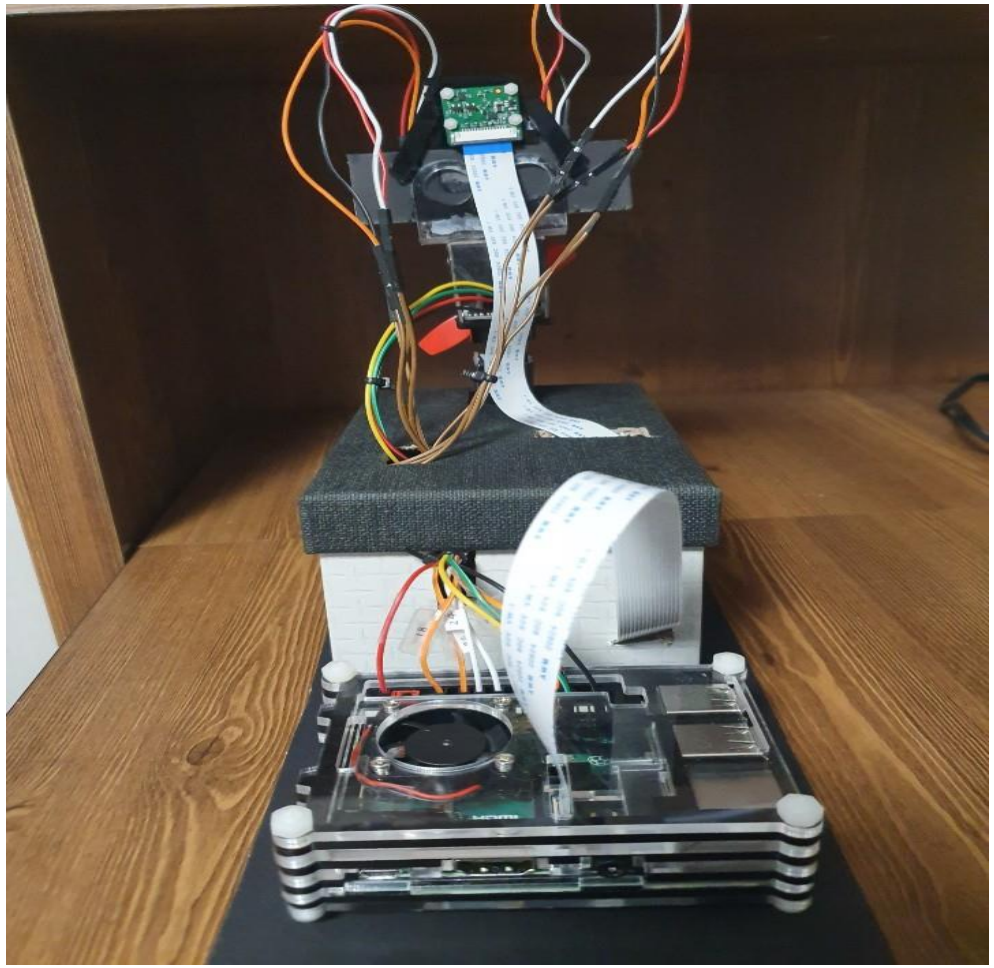
초음파 센서 2EA

DC 모터

SERVO 모터

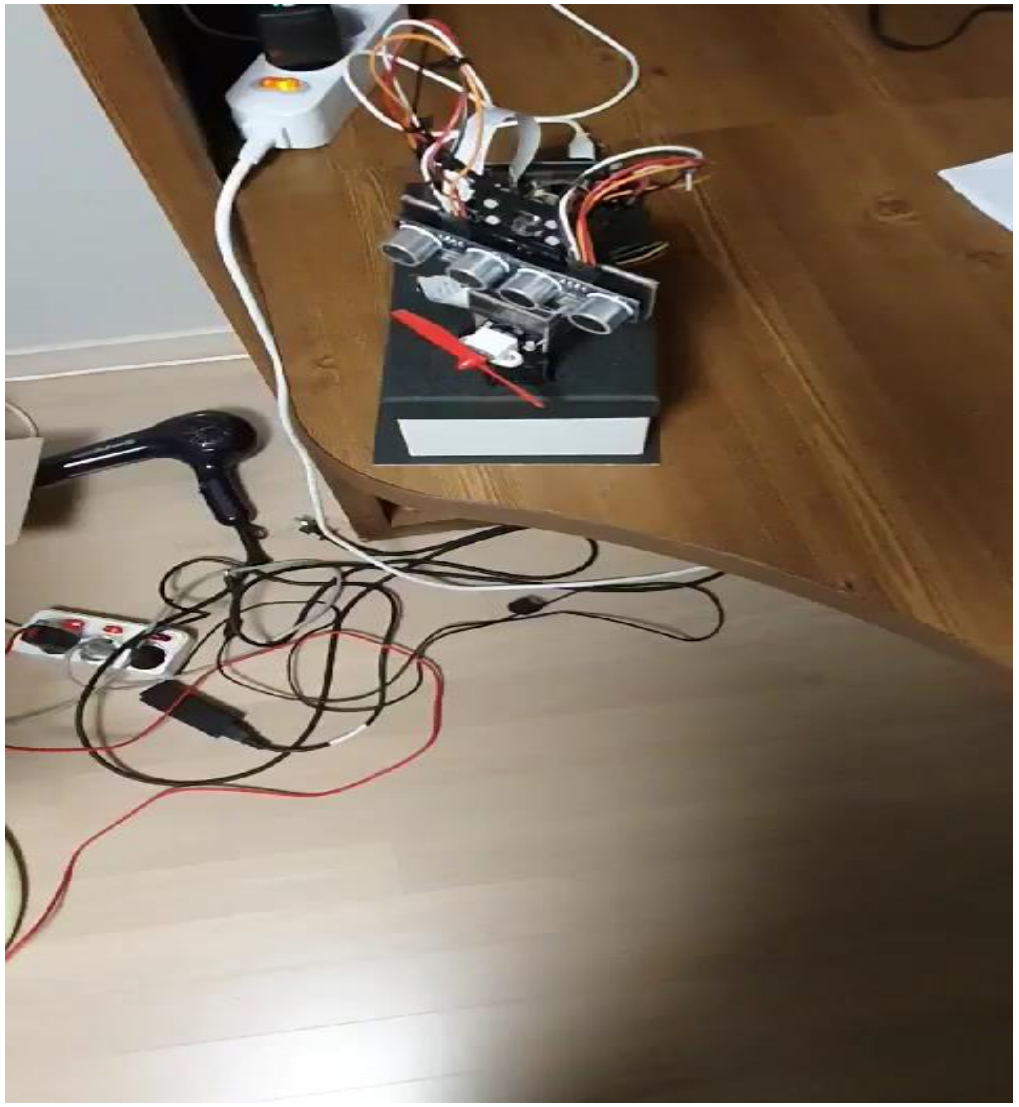
Action Recognition Fan - Picture

사진



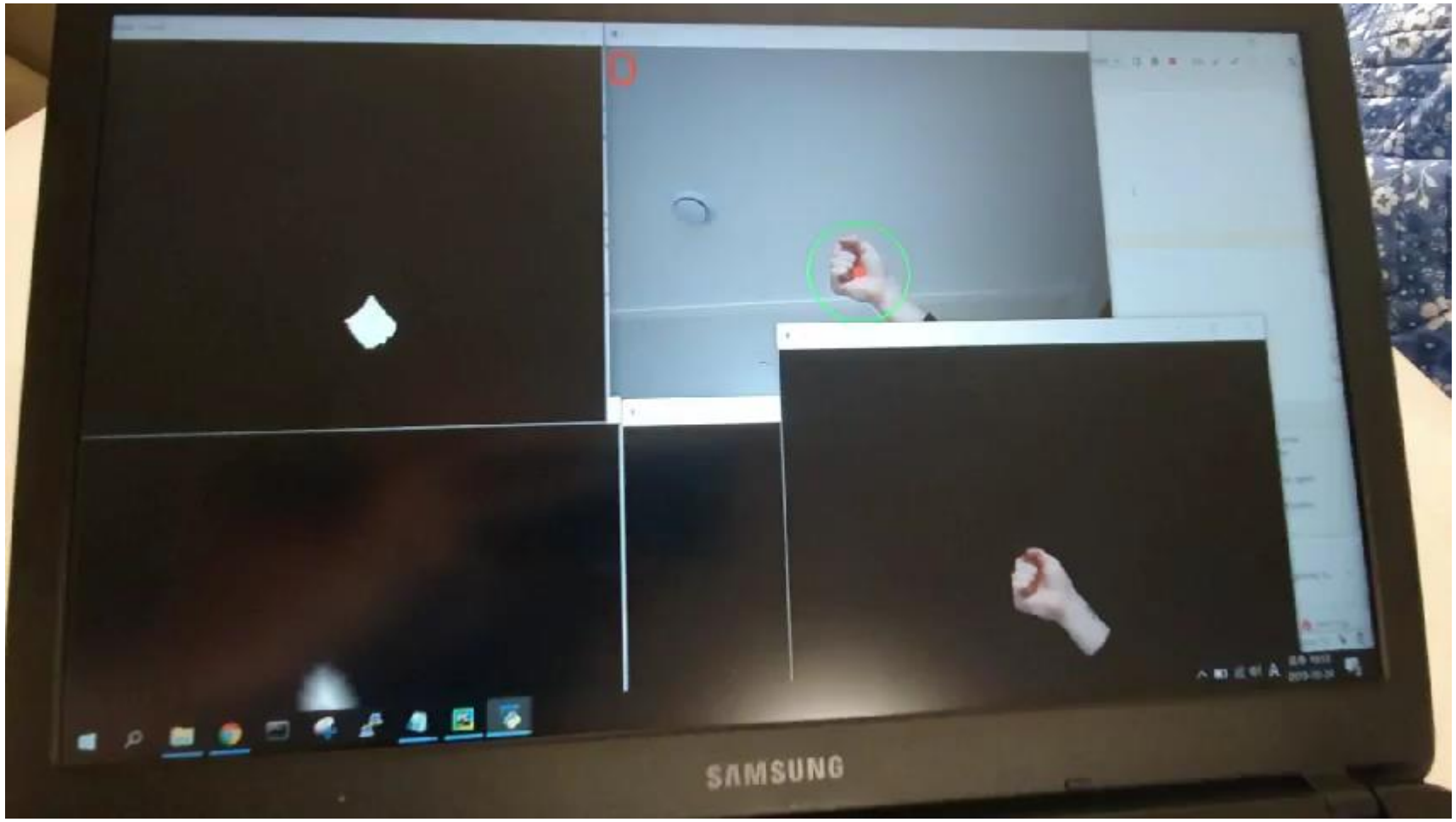
Action Recognition Fan - Video

영상



Action Recognition Fan - Video

영상



Thank You!