카피 (IoT와 함께)

Categorizing Fish by IoT

INDEX

프로젝트 주제

어민들을 위한 병어 자동 분류 시스템

* 병에 걸린 물고기

프로젝트 목적

스쿠티카 병

물고기의 병은
대부분 몸에
시각적으로 보인다.

병어 자동 분류를
통해 양식 어민들의
어려움을 해결

봄철 수온 상승…**양식장** 기생충 발생 '비상' 남도일보 | 2019.05.08. |
또한 많은 양의 수산양식용 종자가 입식되는 시기여서 외부로부터 병원체와 기생충이 유입돼 **전염병** 위험에 노출돼 있다. 품종별로 넙치 **양식장**에서는 스쿠티카충 조피볼락에서는 마이크로코타일 감성돔에서는…

수산물 **전염병** 느는데 정부 관리 헛점 국제신문 | 2019.10.08. |
5년새 국내산 발생 4배로 증가 - 수입산은 16배나 급증했는데- 해수부 질병 5종 관리도 않아- 뒤늦게 연내 관련법 개정 추진 - 해썹 등록 **양식장** 19% 그쳐최근 들어 수산물 **전염병** 발생이 크게 늘고 있지만 정부가 일부…

개발 도구

오픈소스

Inception V3 모델 사용

모듈

파이카메라, 초음파센서, 스텝모터

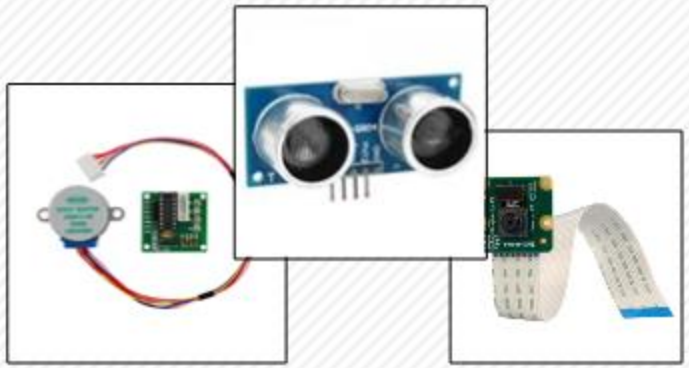
웹

MJPG Streamer

Inception V3

Google AI

딥러닝 기반 이미지 인식



mjpg streamer

개발 순서

준비	데이터 수집, 데이터 가공
구현	파이카메라, 초음파센서, 스텝모터, 스트리밍 연결
제작	아크릴모형 제작, 딥러닝 모델 제작
실행	분류 시뮬레이션

데이터 수집

물고기 이미지 수집

세 가지 개체 (광어, 숭어, 볼락)

- 정상인 물고기 개체별 30장
- 병걸린 물고기 개체별 60장

구현 <아크릴 모형>

큰 통의 물고기 이동

↓

통로를 통해 지나갈 시 초음파 센서로 인식 후
파이카메라로 촬영

↓

찍은 사진으로 미리 만든 모델인 분류 모델 작동

↓

물고기 분류기가 정상 물고기,
병 걸린 물고기를 분류 하여 작동

↓

작은 통 2개에 정상, 병 걸린 물고기가
나뉘어 보관됨

구현 <초음파 센서>

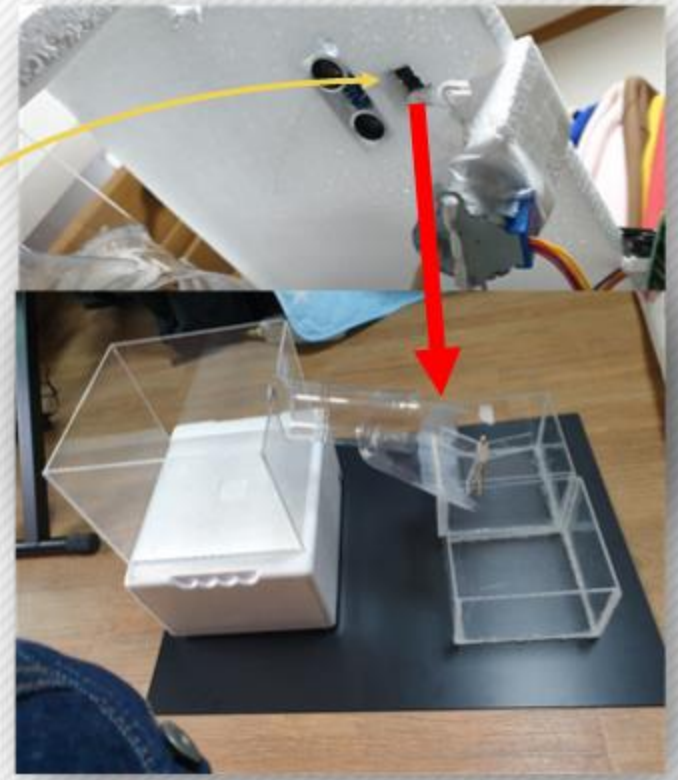
```
while True:
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(TRIGER, GPIO.OUT)
    GPIO.setup(ECHO,GPIO.IN)
    temp=0
    GPIO.output(TRIGER,GPIO.LOW)
    time.sleep(0.5)
    GPIO.output(TRIGER,GPIO.HIGH)
    time.sleep(0.5)
    GPIO.output(TRIGER,GPIO.LOW)

    while GPIO.input(ECHO) == GPIO.LOW:
        startTime = time.time()

    while GPIO.input(ECHO) == GPIO.HIGH:
        endTime = time.time()

    peroid = endTime - startTime
    dist1 = peroid * 17000
    dist2 = round(dist1, 2)

    print('물고기와의 거리', dist2, 'cm')
    if dist2 < 13:
        time.sleep(4)
        urllib.request.urlretrieve("http://192.168.0.15:8090/?action=snapshot",
        GPIO.cleanup()
        break
```

- 초음파 센서를 이용해 물고기의 이동을 관찰한다. 이동 시 카메라에게 촬영을 하라고 신호를 준다.

구현 <파이카메라>

```
if dist2 < 13:
    time.sleep(4)
    urllib.request.urlretrieve("http://192.168.0.15:8090/?action=snapshot", "/home/pi/Desktop/OpenCV_image/fish_photo_test/test_image.jpg")
    GPIO.cleanup()
    break
```

- 물고기 이동 시, 웹 스트리밍 중이던
  웹에 캡처 명령을 내려 **사진을 찍는다.**



구현 <웹 스트리밍>

```
from bottle import route, run, error

@route('/')
@route('/stream')
def stream():
    return """
    <html>
    <head>
        <link href="https://fonts.googleapis.com/css?family=Jua|Song+Myung&d
        <style type="text/css">
            #sm {font-family: 'SongMyung', serif;
                font-size: 32px;
                font-weight: 800}
            #jua {font-family: 'Jua', sans-serif;
                font-size: 40px;}
        </style>
        <style>
            *{margin:0; padding:10;}
            body {
            background-image: url('https://img1.daumcdn.net/thumb/R1280x0/?s
            background-size: cover;
            }
            #d1{
                width: 50px;
                height: 50px;
                margin:0 auto;
            }
            h1{
                text-align: left;
                line-height: 100px;
            }
            p{
                text-align: left;
                line-height: 50px;
            }
            p1{
                text-align: left;
                line-height: 100px;
            }
        </style>
        <title>지능IOT시스템</title>
    </head>
    <body>
        <div id ="jua" style="border-bottom: 5px solid black">
            <h1>지능 IOT 시스템 1팀</h1>
```

- 평상시에는 스트리밍 출력을 하다
  초음파 센서의 신호를 받고
  그때 촬영을 한다.

구현 <Inception V3>

```
imagePath = '/home/pi/Desktop/OpenCV_image/fish_photo_test/test_image.jpg'          # 추론을 진행할 이미지 경로
modelFullPath = '/home/pi/Desktop/tmp/output_graph.pb'                              # 읽어들일 graph 파일 경로
labelsFullPath = '/home/pi/Desktop/tmp/output_labels.txt'                           # 읽어들일 labels 파일 경로

TRIGER = 19
ECHO = 26
startTime = time.time()
temp = 0

def create_graph():
    # 저장된(saved) graph_def.pb로부터 graph를 생성한다.
    with tf.gfile.FastGFile(modelFullPath, 'rb') as f:
        graph_def = tf.GraphDef()
        graph_def.ParseFromString(f.read())
        _ = tf.import_graph_def(graph_def, name='')

def run_inference_on_image():
    answer = None

    if not tf.gfile.Exists(imagePath):
        tf.logging.fatal('File does not exist %s', imagePath)
        return answer

    image_data = tf.gfile.FastGFile(imagePath, 'rb').read()

    create_graph() # 저장된(saved) GraphDef 파일로부터 graph를 생성한다.

    with tf.Session() as sess:

        softmax_tensor = sess.graph.get_tensor_by_name('final_result:0')
        predictions = sess.run(softmax_tensor, {'DecodeJpeg/contents:0': image_data})
        predictions = np.squeeze(predictions)

        top_k = predictions.argsort()[-5:][::-1]  # 가장 높은 확률을 가진 5개(top 5)의 예측값(predictions)을 얻는다.
        f = open(labelsFullPath, 'rb')
        lines = f.readlines()
        labels = [str(w).replace("\n", "") for w in lines]
        for node_id in top_k:
            fish_string = labels[node_id]
            fish_string = fish_string.replace('b\'','')
            fish_string = fish_string.replace('\\n\'','')
            score = predictions[node_id]
            print('%s (score = %.5f)' % (fish_string, score))
            labels[node_id] = fish_string

        answer = labels[top_k[0]]
        return answer, predictions[top_k[0]]
```

- 구글에서 만든 모델 구조와, 파라미터를 사용해 물고기 데이터셋으로 모델을 생성한 뒤 왼쪽의 코드로 리트레이닝하여 물고기의 병 유, 무를 분류하는 코드

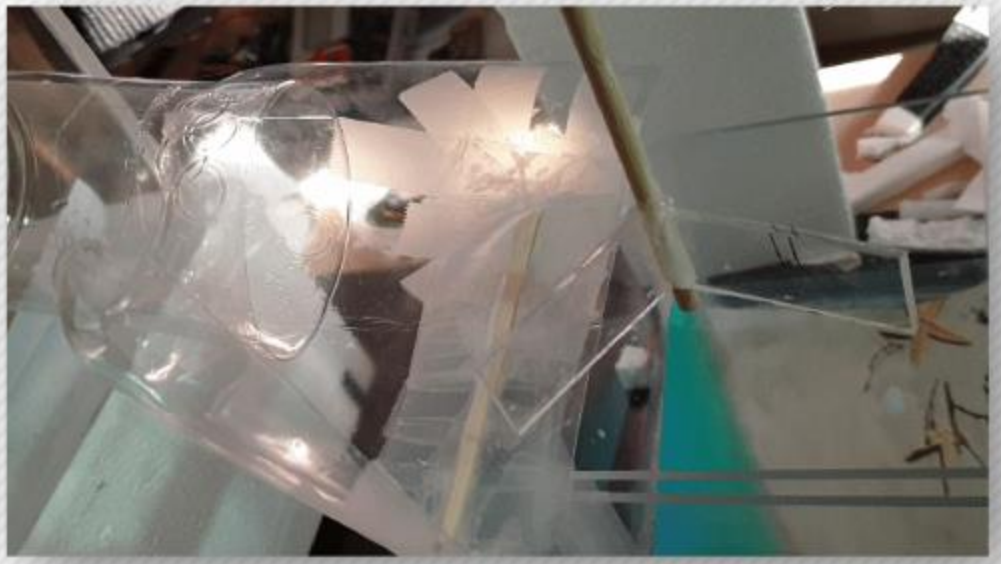
구현 <스텝 모터>

```
GPIO.setmode(GPIO.BCM)
#control_pins = [32,36,37,40]
control_pins = [12,16,20,21]
print(temp)
for pin in control_pins:
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, False)
halfstep_seq = [
    [1,0,0,0],
    [1,1,0,0],
    [0,1,0,0],
    [0,1,1,0],
    [0,0,1,0],
    [0,0,1,1],
    [0,0,0,1]
]

halfstep_seq2 = []
    [0,0,0,1],
    [0,0,1,1],
    [0,0,1,0],
    [0,1,1,0],
    [0,1,0,0],
    [1,1,0,0],
    [1,0,0,0]
]


if temp == 1: # 병걸린 물고기
    for i in range(512):
        for halfstep in range(7):
            for pin in range(4):
                GPIO.output(control_pins[pin], halfstep_seq[halfstep][pin])
            time.sleep(0.001)
elif temp == 2: # 병안걸린 물고기
    for i in range(512):
        for halfstep in range(7):
            for pin in range(4):
                GPIO.output(control_pins[pin], halfstep_seq2[halfstep][pin])
            time.sleep(0.001)

GPIO.cleanup()
```

- 분류 결과에 따라 돌아가는 각도를 달리해서 물고기를 해당하는 통으로 보내기 위해 모터를 회전 시키는 코드

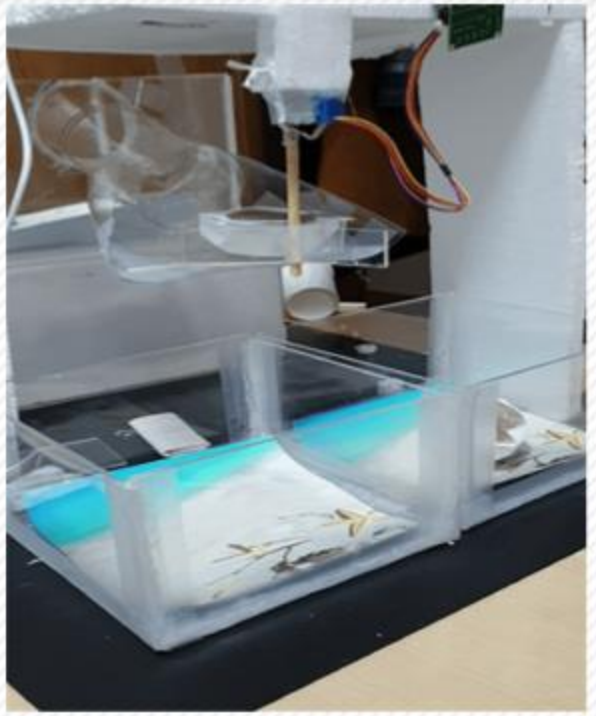
테스트 <정상 물고기>

물고기가 흘러감

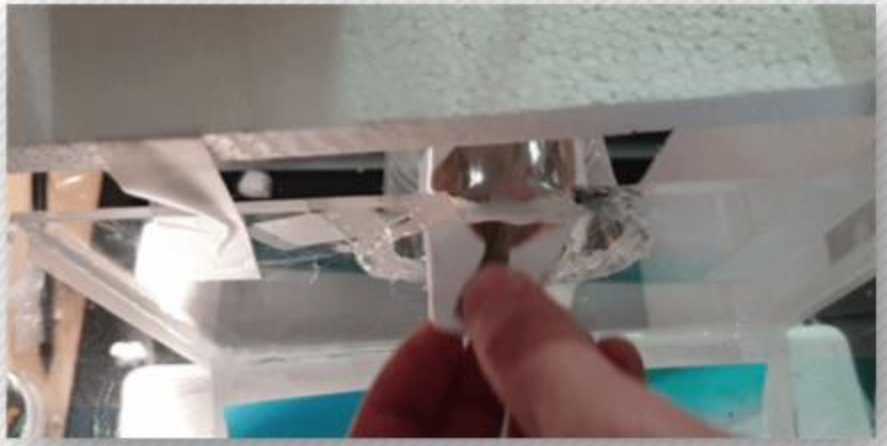


(정상 물고기 모형)

분류되고 있는 모습

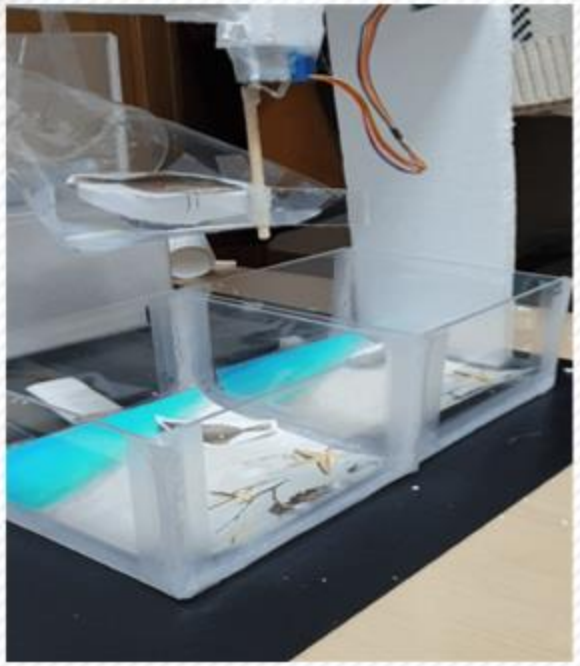


테스트 <병 걸린 물고기>



물고기가 흘러감

(병 걸린 물고기 모형)



분류되고 있는 모습

나아갈 점

빠른
분류 속도

동시
분류기능

더욱 많은
개체 분류

끊김 없는
스트리밍

감사
합니다.