

2023 ICT 융합 프로젝트 공모전

Fishify

허 준, 김승환, 김윤희, 박선규, 유다원

[목 차]

I. 서론	3
1 절 개발 개요	3
1. 개발 배경	3
2. 개발 목표	3
3. 작품 개요	4
II. 본론	5
1 절 개발 환경 및 개발 도구 설명	5
1. 전체 개발 환경	5
2. 보드 및 Tools 설명	6
3. YOLOv5	7
2 절 작품 구성	11
1. 부품 리스트	11
2. 회로도	12
3. 시스템 구성도	13
4. 작품 외관	13
3 절 보드별 동작 흐름도	14
1. 전체 흐름도	14
2. Jetson Nano	15
3. Arduino	15
4. Raspberry Pi	16
4 절 개발 과정 및 주요 기능	17
1. 계획 수립	17
2. 단계별 진행 과정	17
2. 주요 기능	18
5 절 구현 결과	22
1. 시연 영상	22
III. 결론	23
1. 향후 목표	23
2. 기대 효과	23
IV. 참고 자료	24
참고 문헌	24
소스 코드	24

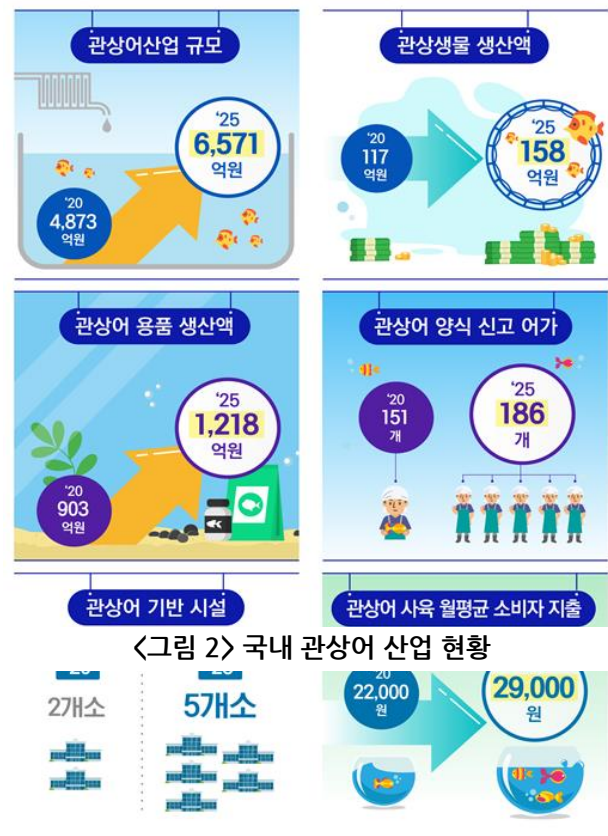
I. 서 론

1 절 개발 개요

1. 개발 배경

우리의 주변에서의 수족관에서는 보기 드물고 접하기 어려운 다양한 관상어를 볼 수 있다. 과거에는 매니아 층을 위한 취미활동으로 인식이 된 관상어 양식은 현대인들 사이에서의 인기있는 취미활동으로 차츰 관심이 높아지고 있다. 최근 국내외적으로 관상어가 가진 가치가 주목받으면서 단순한 양식 사업과 수입 의존형 양식이 아닌, 독보적인 양식 기술을 이용한 국내 자생력을 확대시키기 위해 지속적인 성장세를 보이고 있다. 이미 관상어 시장은 우리나라와 일본 등의 국지적인 시장을 제외하고는 식용어에 비해 규모가 큰 상황이다.

하지만, 치어의 양식과정에서 많은 문제점이 발생되고 있다. 최적 생육 조건을 유지하기 위하여 일정 용량의 수조에 이동하는 과정을 거치고 있으며, 약 100,000 마리의 치어를 직접 작업자의 눈으로 판별하고 분류하는 노동 집약적산업이 이어지고 있다. 대부분의 관상어는 크기가 매우 소형이기에 수작업으로 진행되는 분류작업은 인력과 시간이 많이 소요되며, 정확성과 일관성을 보장하기 어려운 실태이다.



<그림 2> 국내 관상어 산업 현황

2. 개발 목표

1) 아이디어 도출

이러한 문제를 해결하기 위하여 이미지 인식, 패턴, 특징 추출 등의 기술을 적용할 수 있는 머신러닝(Machine Learning) 기술에 자동 분류 시스템(Auto Classification System)을 혼합하여 치어를 정확하고 신속하게 분류하고 계수할 수 있는 시스템을 개발하였다.

2) 필요성

어류 산업에서는 어종별로 성장 속도나 사육 환경 등이 다르기 때문에, 정확한 어종 분류가 필요하다. 이를 수작업으로 진행하면 많은 시간과 비용이 소요된다. 하지만 자동화된 치어 분류 시스템을 도입하면 생산성을 향상시킬 수 있다. 또한, 분류 작업 결과를 데이터베이스(이하 DB)에

<그림 1> 국내 관상어 산업 현황

저장하기 때문에, 데이터 분석과 추적이 용이 해져 생산성과 효율성을 높일 수 있다. 이러한 이유들로 인해 치어 분류기 시스템은 어류 산업에서 매우 중요한 역할을 맡을 수 있고, 국내 관상어 양식 산업의 효율적인 발전을 기대할 수 있다.

어류 관상어(디스커스)	비어류 관상어 (크리스탈레드쉬림프)
	
- 단색 치어 수준의 경우 마리당 1.5 만원 ~ 2 만원으로 유통되나 품종 개량 및 크기에 따라 수십, 수백만원의 가치를 가짐	- 원종은 2\$에 불과하지만 품종 개량을 통할 경우 3,500\$ 이상에 판매 - 국내가격 10 만원~200 만원대

<표 1> 대표적인 어류 관상어 및 비어류 관상어

3. 작품 개요

이 시스템은 여러 종류의 치어가 수로를 통해 이동할 때 종류를 분류하는 시스템이다. 이를 구현하기 위해 수로는 컨베이어 벨트를 사용하고, 각각의 치어는 3D 프린터를 통해 제작하였다. 또한 카운트 시스템을 구성하기 위해 레이저와 CdS(광)센서를 사용하였다.









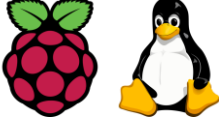





치어를 감지하기 위해 입구에 설치된 CdS 센서가 감지를 시작하며, 학습된 모델을 이용해 분류가 이루어진다. 만약 1 번 종류의 치어가 감지되면 Jetson Nano(Main Server)는 해당 신호를 Arduino(Client)에게 전달하여 1 번 모터가 작동되어 1 번 수조로 이동한다. 이때 수조 통로에 달린 CdS 센서를 통해 계수가 이루어진다.

Arduino 에서 감지된 치어는 정확하게 인식되었음을 확인하기 위해 Jetson Nano 로 수신 신호를 보낸다. 최종 수신부인 Raspberry Pi 에서는 인식된 치어들의 종과 수량을 Jetson Nano 로부터 Socket 통신을 통해 데이터를 수신 받아 Maria DB 서버에 저장한다. 이때, 데이터의 안정성을 위해 데이터 전송 중 손실이나 오류가 발생할 경우를 대비하여 LCD 상으로 에러 확인이 가능하다. 데이터 삭제 기능으로 관리자는 효율적인 데이터 관리도 가능하다.

II. 본 론

1 절 개발 환경 및 개발 도구 설명

1. 전체 개발 환경

	Device	Server Main	Application
Board	 ----- Arduino Mega 2560	 ----- Jetson Nano	 ----- Raspberry Pi 4 Model B
Language			
OS	 ----- Windows 10	 ----- JetPack 4.6.0	 ----- Raspbian GNU / Linux 11
Library	Servo.h -----	 ----- OpenCV / PyTorch / YOLOv5	 ----- Qt
System	 ----- VS Code	 ----- Google Colab	 ----- MariaDB

<표 2> 전체 개발 환경

2. 보드 및 Tools 설명

1) Jetson Nano

Jetson Nano 는 이미지 분류, 객체 감지, 세그멘테이션, 음성 처리 등 높은 연산을 필요로 하는 데이터 처리를 실행하는 소형 컴퓨터 중에서 뛰어난 성능을 가진 컴퓨터이다. 인터페이스와 확장성에 면에 뛰어난 기능을 가지고 있어 최근에는 AI 를 활용한 프로젝트에서 많이 사용되고 있다. 본 시스템에서는 머신 러닝을 통한 치어 분류를 위한 주 시스템으로 사용되었다.

2) Arduino

오픈소스 하드웨어 플랫폼이며 마이크로 컨트롤러를 탑재하고 있는 Arduino 는 다양한 장치들을 제어하고 센서 데이터 수집, 제어기능 등으로 활용이 가능하다. 용도에 따른 보드를 선택할 수 있고 라이브러리가 오픈 소스로 제공된다. 이에 따라 빠르고 쉽게 응용 프로그램을 개발이 가능하다. 본 시스템에서는 CdS 센서, 서보 모터, LED 를 제어하는 용도로 활용하였다.

3) Raspberry Pi

Raspberry Pi 는 리눅스 운영체제 기반으로 동작하는 싱글 보드로, 다양한 개발도구와 라이브러리를 포함하여 개발 환경을 쉽게 구축할 수 있다. 컴퓨터지만 크기가 작아 휴대성이 뛰어나고 전력 소모 또한 낮은 편이다. 많은 인터페이스를 제공하여 개발, 통신, 데이터베이스 등 여러가지 용도로 사용이 가능하다. 본 시스템에는 Maria DB 를 사용한 Database 용 및 Qt 로 구현된 GUI 를 LCD 상으로 출력하는데 사용되었다.

4) OpenCV

OpenCV 는 영상 처리 분야에서 널리 사용되는 오픈 소스 라이브러리로, 객체감지를 비롯한 다양한 함수와 알고리즘을 제공한다. C++, Python, Java 등 다양한 프로그래밍 언어로 지원이 되고, Windows, Linux, Mac OS 등 다양한 플랫폼에서 사용이 가능하다. 이미지나 비디오 등에서 물체 검출, 추적, 특징 추출, 패턴 인식 등 다양한 컴퓨터 비전 작업을 수행할 수 있다. 최근에는 OpenCV 를 활용해 자율 주행, 보안 시스템, 의료 분야 등에서도 널리 활용되고 있다. 본 시스템에서는 객체감지를 위한 기능을 오 실시간 이미지 프로세싱을 핵심 기능으로 사용하였다.

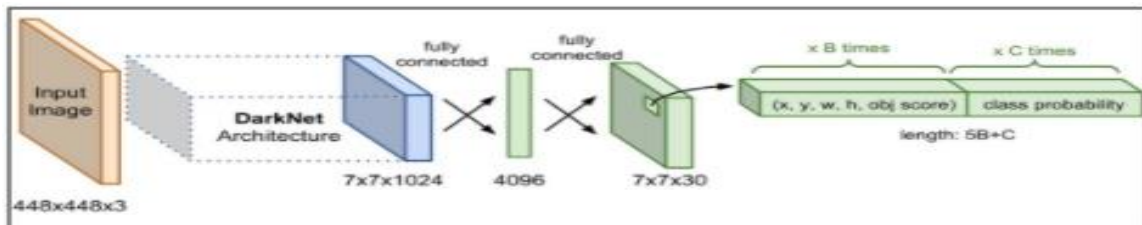
5) Qt

Qt 는 크로스 플랫폼 어플리케이션 개발을 위한 프레임 워크이다. 특히, GUI(그래픽 사용자 인터페이스)를 구축하는데 중점을 둔 프레임 워크로, 다양한 플랫폼에서 일관성 있는 UI 를 만들 수 있도록 지원한다. 본 시스템에서는 Python 코드를 이용하여 GUI 를 구축하였으며, Maria DB 와 연동을 통하여 데이터를 수집하여 관리자가 용이하게 관리할 수 있도록 활용하였다.

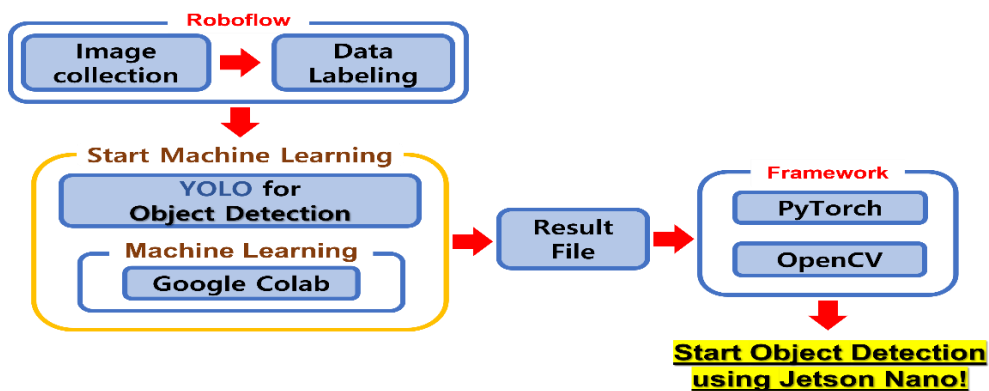
3. YOLOv5

1) YOLOv5

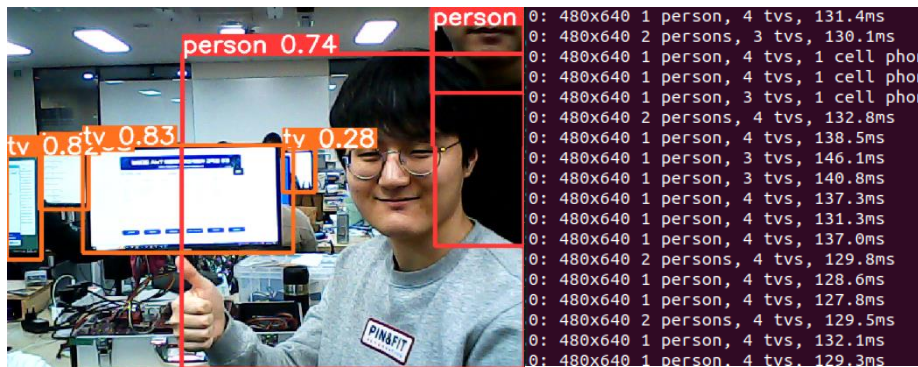
YOLOv5 는 실시간 객체감지를 위한 딥 러닝 모델 중 하나다. YOLOv5 신경망 아키텍처는 Backbone 네트워크, Neck 네트워크, Head 네트워크 총 3 가지로 구성된다. Backbone 은 이미지로부터 특징 맵을 추출, Neck 은 다양한 스케일의 특징 맵을 결합, Head 는 추출된 특징 맵을 바탕으로 물체의 위치를 찾는다. 아래 그림에서 DarkNet 부분이 Backbone 이며 YOLOv5 에서는 심도 배수(Depth Multiple)와 폭 배수(width multiple)를 기준으로 크기가 작은 순부터 yolov5s, m, l, x 총 4 가지 버전이 있다. 작은 모델일수록 더 적은 매개변수를 가지고 있어 학습하는데 적은 시간이 걸리지만 예측 정확도가 낮다. Head 에서는 미리 정의된 사각형 형태의 Anchor Box(Default Box)를 처음에 설정하고, 이를 이용해 최종적인 Bounding Box 를 생성한다. 모델이 예측한 물체의 위치와 실제 물체의 위치 사이의 차이를 계산하여 손실 값을 얻는다. 이 손실 값을 최소화하기 위해서는 모델의 가중치를 업데이트를 해야한다.



<그림 2> YOLOv5 아키텍처



<그림 3> Machine Learning Structure



<그림 4> YOLOv5 로 인식된 사람 및 사물

2) 데이터 학습 과정

Roboflow 를 이용하여 4500 장 이상의 물고기 이미지 데이터를 수집하고 라벨링을 시행하였다. 학습 데이터의 경로(train, val, test), 클래스 개수(nc), 클래스 종류(names)로 아래와 같은 yaml 파일을 제작한다.

```
1 train: ../train/images
2 val: ../valid/images
3 test: ../test/images
4
5 nc: 3
6 names: ['Arapaima gigas', 'Pterophyllum', 'ikan-mas']
7
```

<그림 5> data.yaml

```
!python train.py --img 256 --batch 16 --epochs 250 --data {dataset.location}/data.yaml --weights yolov5s.pt --cache
```

Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
0/249	0.585G	0.0918	0.0229	0.03783	30	256: 100% 86/86 [00:18<00:00, 4.76it/s]
	Class	Images	Instances	P	R	mAP50 mAP50-95: 100% 11/11 [00:03<00:00, 2.80it/s]
	all	334	454	0.176	0.39	0.176 0.0962
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
1/249	0.74G	0.05618	0.02207	0.02892	38	256: 100% 86/86 [00:12<00:00, 6.70it/s]
	Class	Images	Instances	P	R	mAP50 mAP50-95: 100% 11/11 [00:02<00:00, 4.47it/s]
	all	334	454	0.283	0.456	0.29 0.172

<그림 6> 학습 코드

코드명	시행 내용
img 256	이미지 사이즈 256*256
batch 16	batch 사이즈 16 으로 한 번에 학습할 데이터의 개수
epochs 250	전체 학습 데이터셋을 250 번 학습

<표 3> 코드 시행 내용

3) Custom 데이터 평가 및 예측

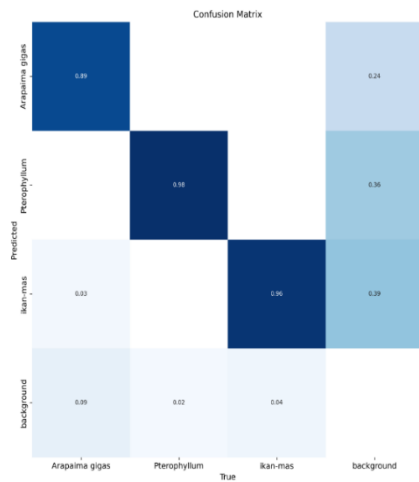
Google Colab 에서 train.py 를 사용하여 데이터 학습을 진행한다. yaml 파일을 가져오고 사전에 학습된 모델 가중치인 yolov5.pt 를 이용하여 학습시킨다. YOLOv5 에서 제공한 Detect.py 를 사용하여 test 파일에 있는 이미지로 객체 감지를 수행하였다.

```
!python detect.py --weights runs/train/exp/weights/best.pt --img 416 --conf 0.1 --source {dataset.location}/test/images
```

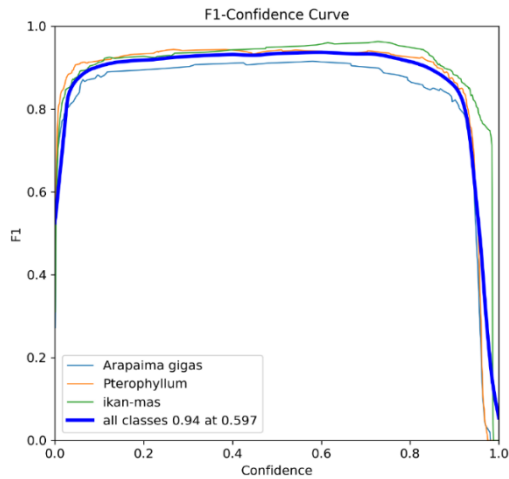
<그림 7> 학습 결과 Test 코드

코드명	시행 내용
conf 0.1	감지 임계값(Confidence Threshold)을 0.1 로 설정

<표 4> 코드 시행 내용



<그림 8> Confusion Matrix



<그림 9> F1-Confidence Curve

위 그림과 같이 Tensor Board 를 활용하여 학습한 모델의 분류 성능을 평가하고 시각화 해줄 그래프를 확인하고 검증하였다. Confusion Matrix 는 실제 값과 모델이 예측한 값을 비교하여 행렬로 나타낸다. 이를 통해 정확도(Accuracy), 정밀도(Precision), 재현율(Recall)을 계산한다.

Confusion Matrix 좌표	설 명
True Positive (TP)	실제 값과 모델 예측이 모두 긍정인 경우
False Positive (FP)	실제 값은 부정이지만 모델 예측이 긍정인 경우
True Negative (TN)	실제 값과 모델 예측이 모두 부정인 경우
False Negative (FN)	실제 값은 긍정이지만 모델 예측이 부정인 경우

<표 5> Confusion Matrix 좌표별 설명

성능 지표	설 명	수 식
정확도	전체 예측 중에서 맞는 예측의 비율	$\frac{TP + TN}{TP + TN + FP + FN}$
정밀도	Positive 로 예측한 것 중에 실제로 Positive 인 비율	$\frac{TP}{TP + FP}$
재현율	실제로 Positive 인 것 중에 모델이 Positive 로 예측한 비율	$\frac{TP}{TP + FN}$

<표 6> 성능 지표별 설명

F1-Confidence Curve 그래프는 모델의 분류 결과인 정확도를 x 축, F1-Score 를 y 축으로 하는 그래프로 이때 F1-Score 는 정밀도와 재현율의 조화평균으로 계산되는 성능 지표다. 이 그래프를 통해 모델이 특정 정확도 이상에서 어느 정도의 성능을 보이는지 파악할 수 있으며, F1-Score 가 최대가 되는 정확도를 선택하여 최적의 성능을 얻을 수 있다.

$$(F1 - Score) = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

<수식 1> F1-Score 수식



<그림 10> 최종적으로 웹캠을 이용하여 Custom Dataset 모델을 검증

2 절 작품 구성

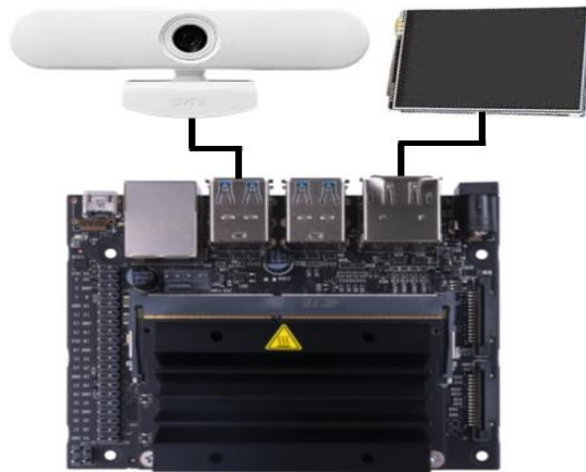
1. 부품 리스트

시스템	부품명	이미지	링 크 (고유 번호)
영상 분석	Jetson Nano		https://url.kr/46k8ft (12513656)
	SKY-BWC1F		https://url.kr/gh1e8u
Arduino 동작제어	Arduino Mega 2560		https://url.kr/mxrfc3 (34405)
	컨베이어 벨트 키트		https://url.kr/uj2n7t
	SG90 서보 모터		https://url.kr/nsidtj (1324334)
	CdS 센서		https://url.kr/wtgelm (1324334)
	LED		https://url.kr/aj72iq (1324334)
	레이저 포인터		개별 구매 품목
DB & GUI	Raspberry Pi 4 Model B		https://url.kr/hov8m9 (12234534)
	5" HDMI Display		https://url.kr/as76w5 (1382229)

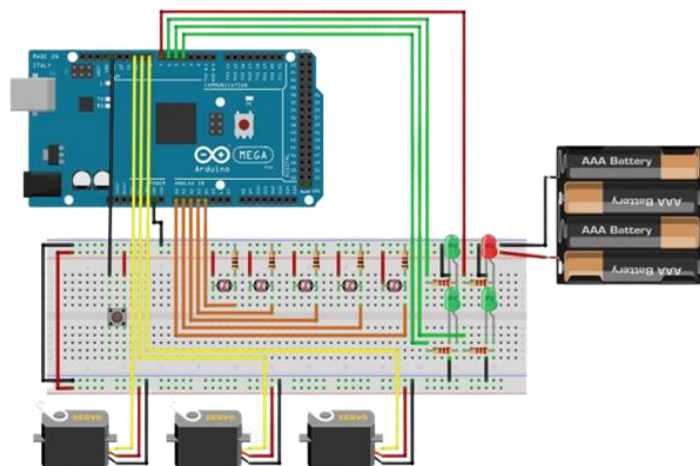
<표 7> 부품 리스트

2. 회로도

Jetson Nano 회로도



Arduino 회로도

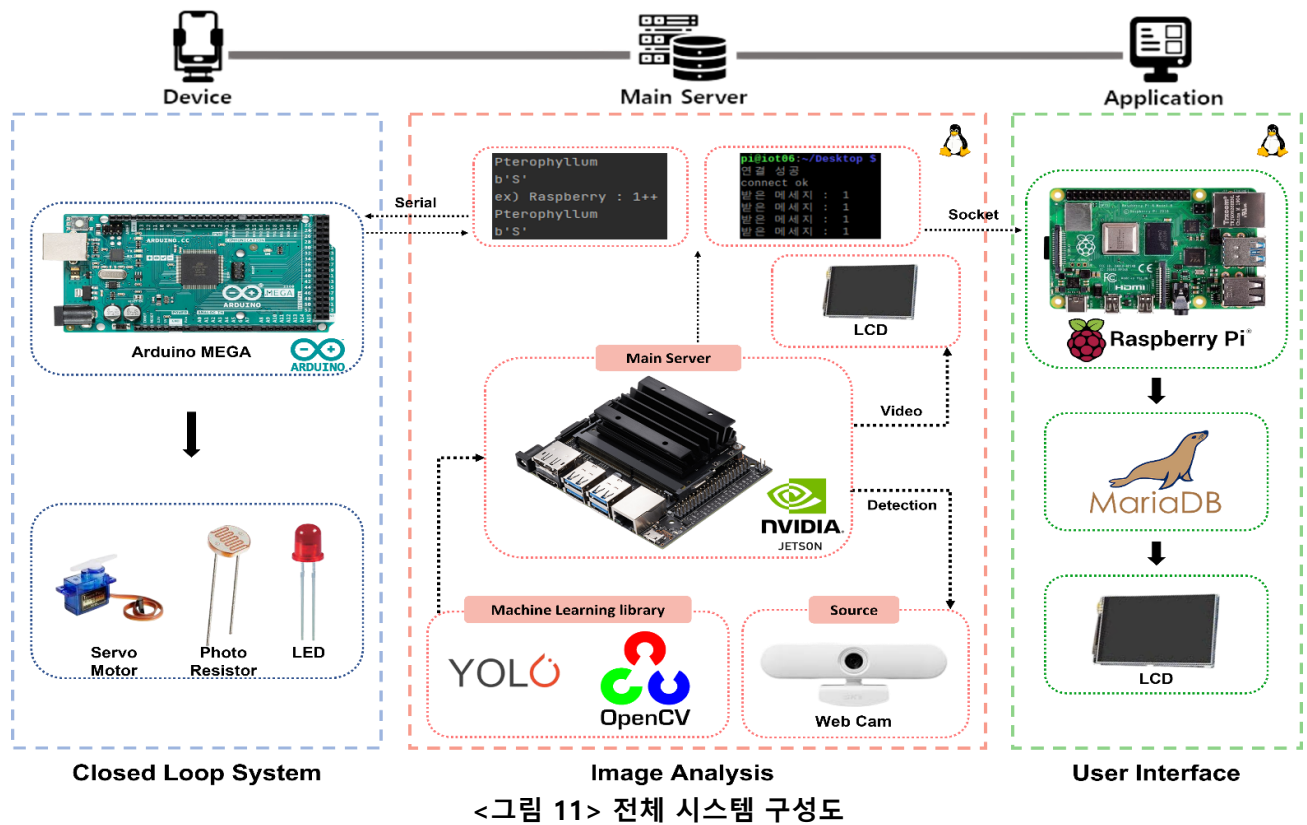


Raspberry Pi 회로도



<표 8> 하드웨어 회로도

3. 시스템 구성도

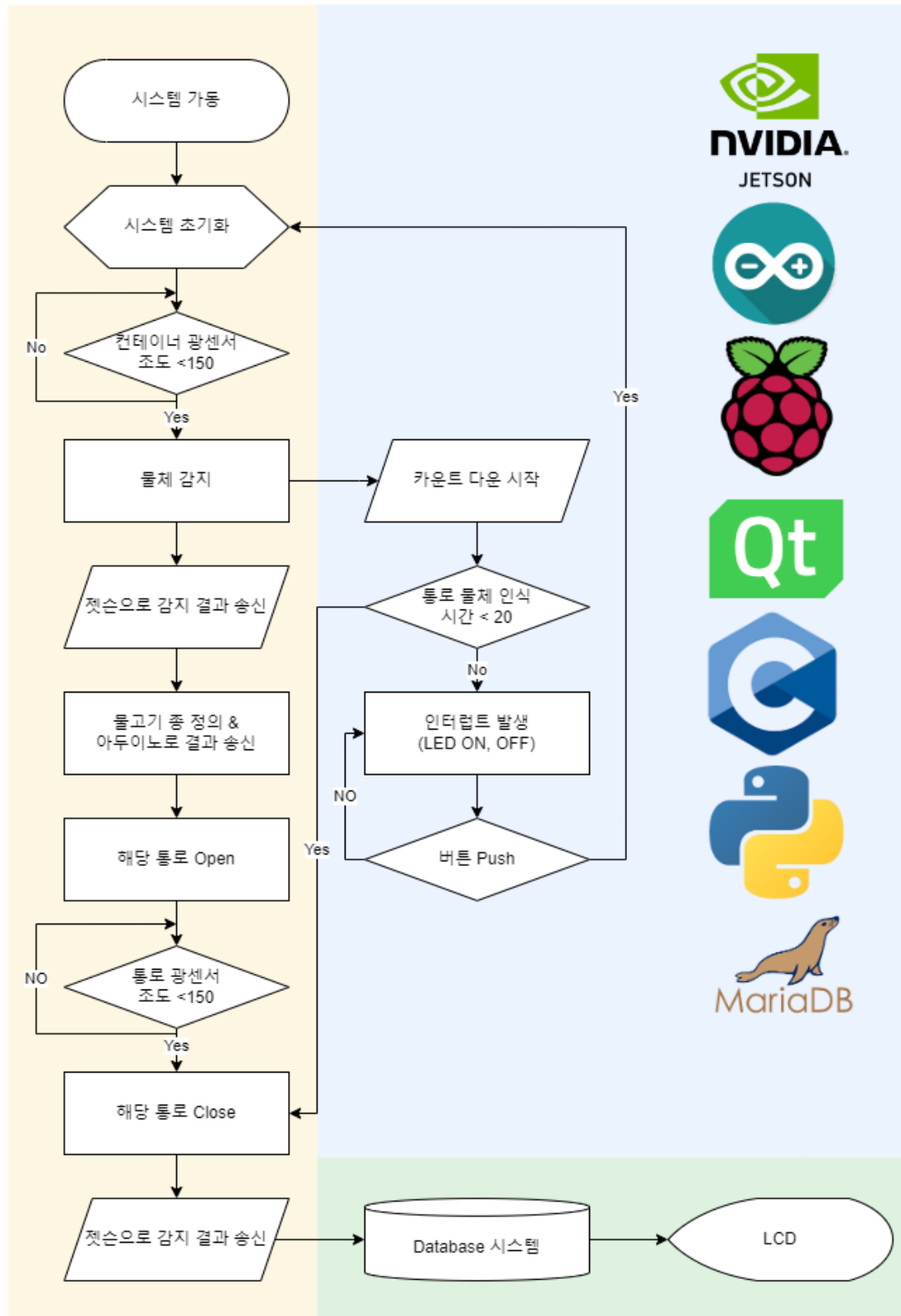


4. 작품 외관

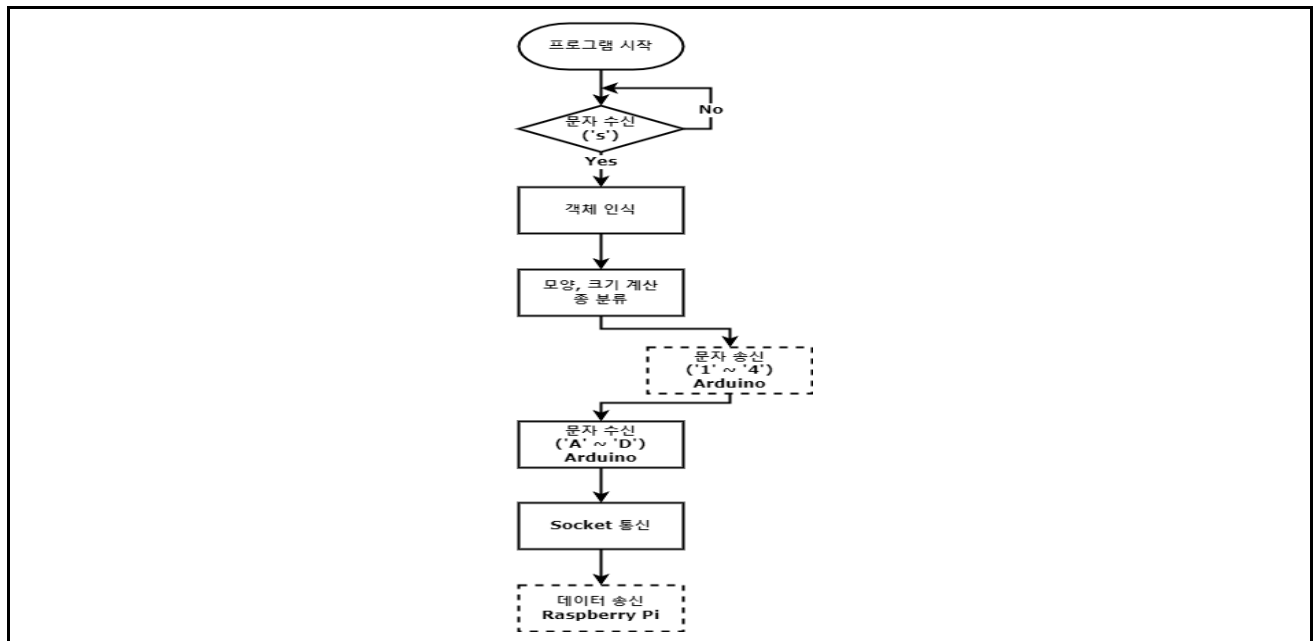


3 절 보드별 동작 흐름도

1. 전체 흐름도

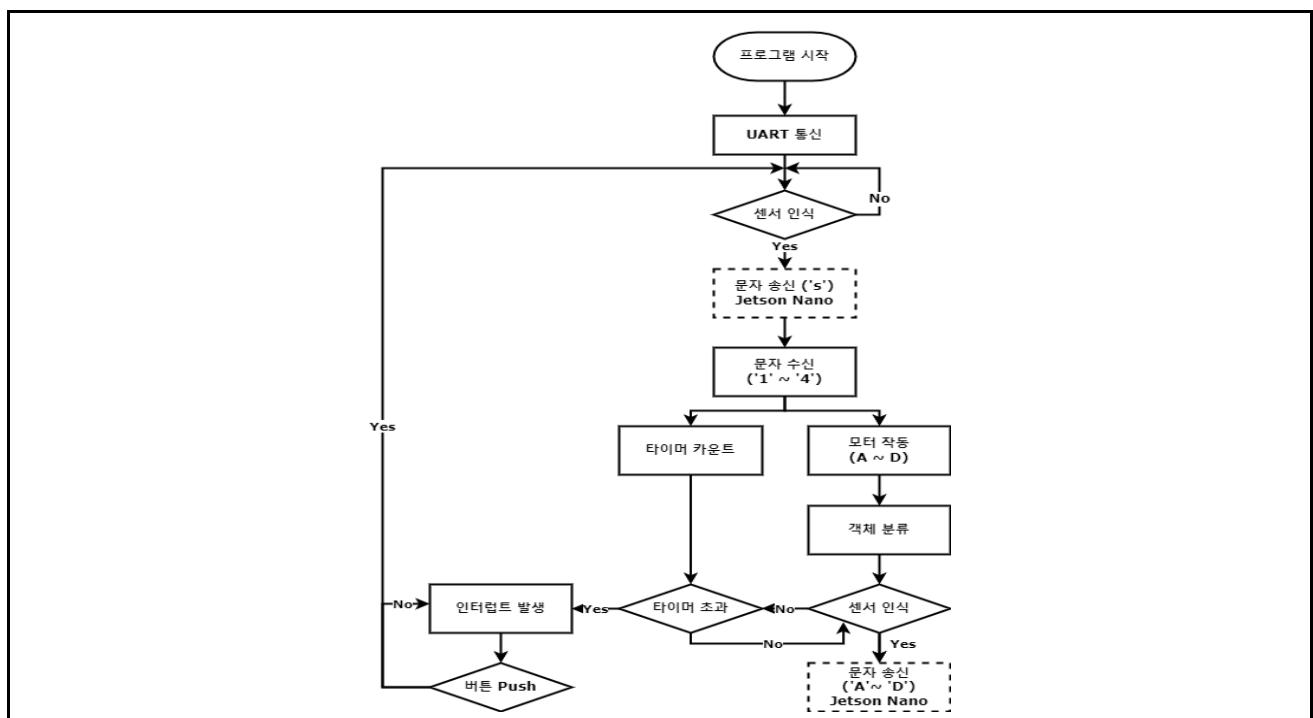


2. Jetson Nano



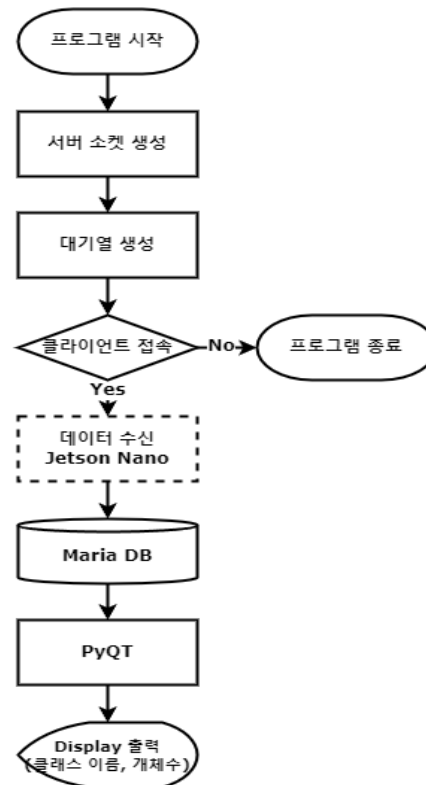
Jetson Nano 는 머신 러닝을 이용해 객체를 감지하는 핵심 역할을 담당한다. Arduino 와 문자 송수신을 위해 PORT 로 UART 통신을, Raspberry Pi 와는 문자열 송수신을 위해 Host IP 와 PORT 로 Socket 통신을 시작한다. 웹캠으로 객체를 인식하고, 인식된 결과를 문자열로 송수신한다. 이를 통해 Arduino 를 제어가 가능하게 한다.

3. Arduino



Jetson Nano 와 UART 통신으로 연결된 Arduino 는 Jetson Nano 로부터 수신 받은 데이터로 서보 모터, 타이머를 제어한다. 분류가 완료되면 Jetson Nano 에게 데이터를 송신하여 작업 완료를 알린다.

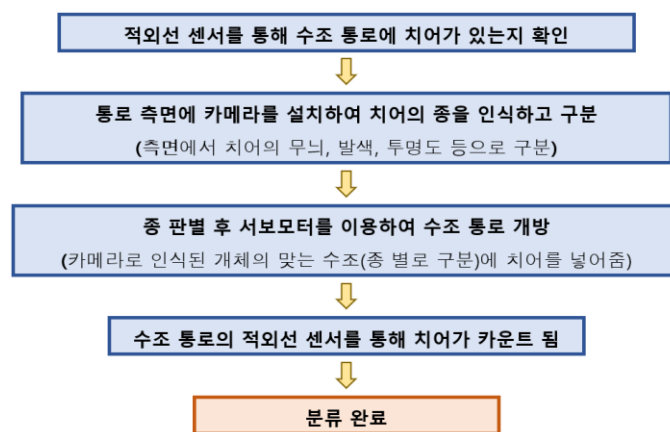
4. Raspberry Pi



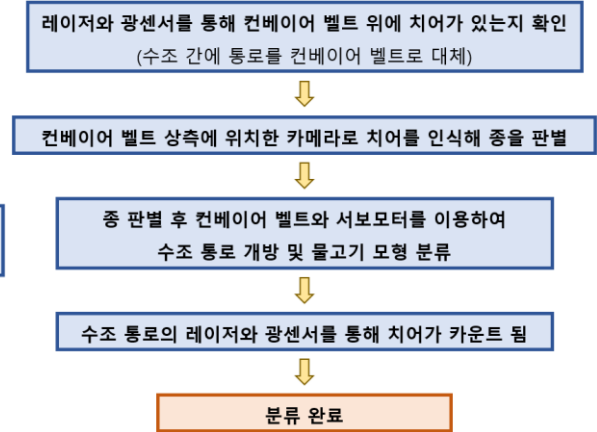
데이터를 수신 받기 위해 서버 소켓을 생성한다. 클라이언트가 접속하기 전까지 대기 상태를 유지하다가 접속이 되면 통신을 시작한다. 클라이언트인 Jetson Nano로부터 전송된 분류 결과 데이터를 문자열 형식으로 수신 받아 Maria DB에 저장하고, Qt로 제작한 GUI에 저장된 데이터를 LCD로 출력한다. 이를 통해 실시간으로 데이터를 모니터링하고, 필요한 조치를 취할 수 있다.

4 절. 개발 과정 및 주요 기능

1. 계획 수립



<그림 14> 이론상 구현 계획



<그림 15> 실제 구현 계획

2. 단계별 진행 과정

Fishify 개발 일정

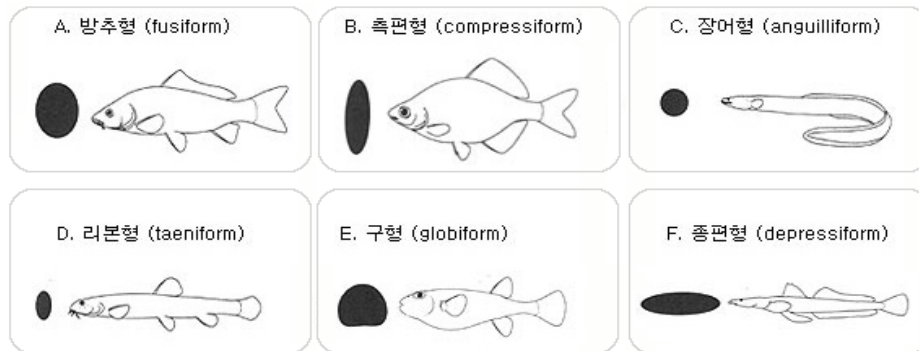
분류	작업제목	1주차	2주차	3주차	4주차	5주차	6주차	7주차	8주차
프로그램 구상 및 착수	주제선정 및 구체화								
	센서 및 보드조사								
	구매물품 선정								
	피드백 및 최종계획 수립								
제품 개발 (개발 환경 셋팅 및 데이터 수집)	Jetson Nano 및 OpenCV 개발환경 셋팅								
	YOLOv5를 이용한 사물/사람 Detect Test								
	사물 및 사람 Detect 후 데이터 값 출력								
	기기별 개발환경 검토								
제품 개발 (영상 처리)	영상/사진 데이터 수집								
	데이터 셋 가공(전처리)								
	모델 학습								
	학습된 모델 Jetson Nano 에 적용								
	학습된 모델 최종 파일 Test								
제품 개발	데이터 결과에 따른 모터 및 센서제어								
	제품 회로 배치 및 외관 제작								
최종 종합 검증	전체 구동 검사								
	오류확인 및 해결								
	프로젝트 최종 보고서 작성								

<표 9> Fishify 개발 일정

3. 주요 기능

1) 어류 학습 및 분류

어류는 아래 그림과 같이 방추형, 측판형, 장어형 등 뚜렷한 특징으로 분류되며, 꼬리 지느러미의 모양도 종류에 따라 다르다. 본 시스템에서는 대표적으로 3 가지 형태를 선택하여 학습시켰다.



<그림 16> 다양한 치어 형태

컨테이너에서 웹캠으로 감지한 치어는 어류 이미지를 종에 따라 학습시켰으며 데이터를 각각 분류했다. 이때, 인식 정확도(Recognition accuracy) 95% 이상의 결과를 도출해 냄으로써 보다 정확한 분류가 가능하게 하였다. 3 개의 종을 학습시켜 분류한 어류들은 실제 구현 상황을 시연하기 위해 아래와 같이 3D 프린터로 제작하였다. Class No.4 는 다른 치어가 감지되었을 때를 가정하여 Trash(No Detect)로 분류하도록 예외 처리를 하였다.

Class No.	Class Name	실제사진	모형사진
1	Pterophyllum		
2	Arapaima gigas		
3	Ikan-mas		
4	Trash (No Detect)	Class No.4 는 다른 종류의 물고기 혹은 물체를 감지하면 'Trash'로 분류하도록 설정	

<표 10> 물고기 학습 및 분류

2) CdS 센서를 활용한 제어

본 시스템에서는 물체를 감지하기 위해 CdS 센서를 활용하였으며, 처음 물체를 감지하는 센서 1 개, 분류 작업 이후 치어를 계수하는 센서 4 개로 총 5 개를 사용하였다. 컨베이어 벨트에 첫 번째로 위치한 CdS 센서는 치어를 감지하기 전까지 일정한 레이저 값을 받아들이고 있다. 이후 치어가 레이저를 가리면 Arduino 에서 Jetson Nano 로 문자를 송신해 치어가 어떤 종인지 판별을 시행하도록 한다. 아래 그림에서는 물체를 감지할 때마다 문자를 송신하는 상황이다.

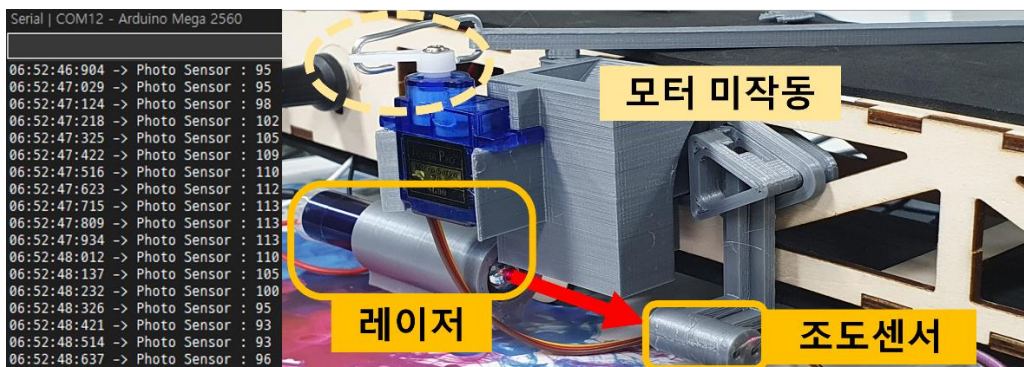


<그림 17> 물체 감지 대기 상태

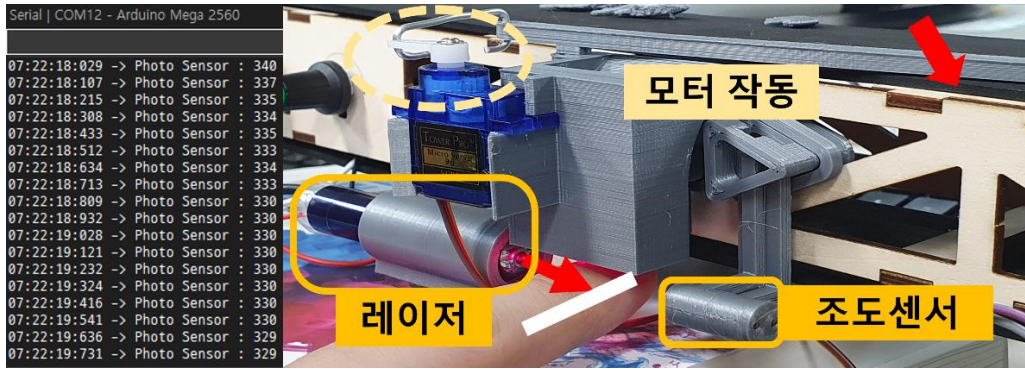


<그림 18> 물체를 감지한 상태

Jetson Nano 에서 치어를 판별하고 결과를 송신하면 Arduino 에서 판별 결과를 수신받고 해당 서보 모터를 작동해 통로를 개방한다. 이후 치어가 통로에 들어올 때까지 카운트 인터럽트가 시작되고 대기 상태에 돌입하며, 설정한 인터럽트 시간 이전에 통로 아래에 설치된 치어가 감지되면 서보 모터가 작동되면서 통로를 폐쇄하면서 Jetson Nano 로 분류 결과를 송신한다. 실험을 위하여 사람의 손을 대신하여 서보 모터를 제어하였다.

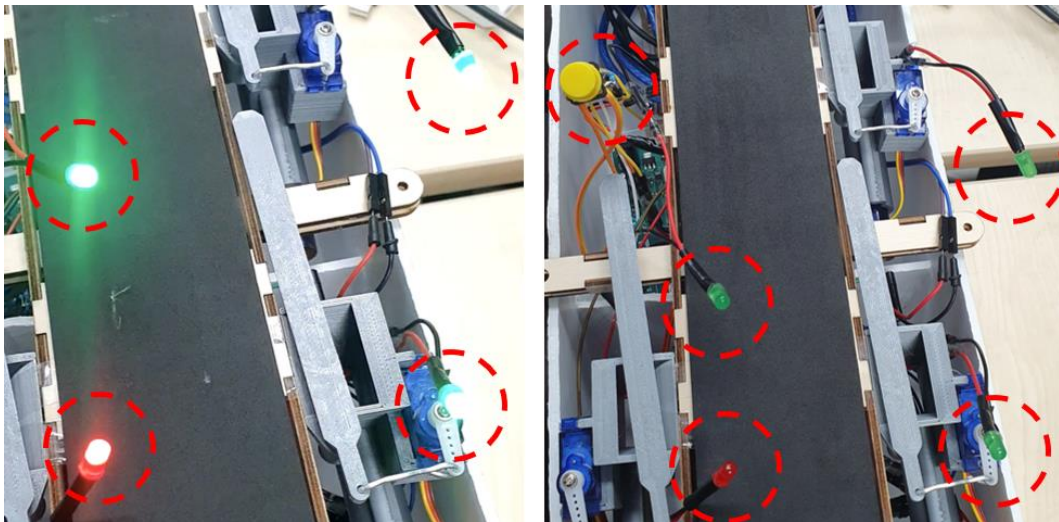


<그림 19> 센서 미감지로 개방된 통로



<그림 20> 센서 감지로 폐쇄된 통로

만약 인터럽트 시간이 초과되면 메인 함수의 상태가 전환이 되면서 LED 가 일정 시간 ON/OFF 되고 통제가 불가능하게 된다. 이후 Push Button 을 누르면 메인 함수 상태가 다시 원상 복귀되고 인터럽트 상태에서 해제가 된다.



<그림 21> 인터럽트 발생 및 해제 과정

3) Data Management

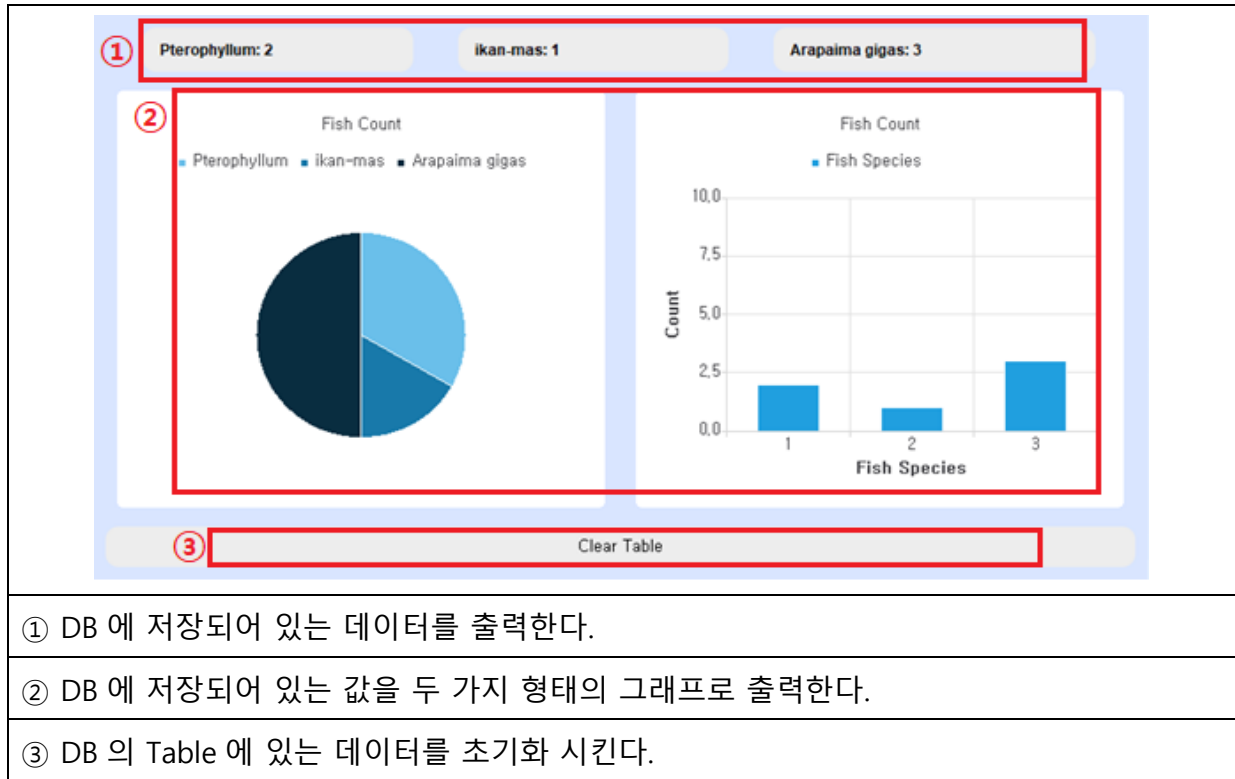
분류 후 생성된 데이터는 Socket 통신을 통하여 Jetson Nano 에서 Raspberry Pi 로 문자열을 송신한다. 이후 Raspberry Pi 에 구축한 Maria DB 에 데이터를 저장하며, Timer 함수를 통한 데이터를 일정 시간(0.5 초)마다 DB 에서 가져온다. Table 은 number(데이터 순서), classification(어종), quantity(인식된 객체 수), detect_time(DB 저장 시간)으로 구성되어 있으며, 문자열을 수신하면 Table 에 데이터를 실시간으로 치어 인식 결과에 대한 정보를 저장한다.

number	classification	quantity	detect_time
5	trash	1	2023-03-15 20:32:09
6	trash	1	2023-03-15 20:32:09
7	Pterophyllum	1	2023-03-15 20:32:37
8	Arapaima gigas	1	2023-03-15 20:32:39
9	ikan-mas	1	2023-03-15 20:32:40
10	trash	1	2023-03-15 20:32:40

<그림 22> 데이터 수신과 동시에 Maria DB 에 데이터 저장

4) GUI 인터페이스

실시간으로 쌓이는 데이터를 GUI에 그래프와 개수로 시각화 시킨다. 데이터가 있는 테이블은 'Clear Table' 버튼을 눌러 모든 데이터를 삭제할 수 있다.



<표 11> GUI 인터페이스

5) 경고 기능

DB에서 가져온 데이터가 설정한 데이터가 아닌 예외처리한 데이터를 수신하면 알림 창으로 사용자에게 알리며, 알림 창을 눌러 데이터를 삭제한다. 예외 처리 데이터(Trash)가 있을 경우, 팝업 기능과 조회 기능을 통해 데이터를 관리할 수 있다. 이 기능을 통해 에러를 사전에 방지하고 조기 대응할 수 있으므로 데이터 관리에 매우 유용한 이점을 가지고 있다.



<그림 23> Trash 값 수신 시 출력되는 팝업창

6) Monitoring

웹캠과 연결된 LCD 를 이용하여, 관리자가 운영중인 모든 시스템 및 프로세스를 지속적으로 모니터링하고 분석하여 문제점을 즉시 탐지하고 조치할 수 있다. 이는 시스템의 안정성과 신뢰성 보장 및 장애 최소화, 가용성 향상에 큰 기여를 한다.



<그림 24> 웹캠으로 연결된 LCD 로 지속적인 Monitoring 가능

5 절. 구현 결과

1. 시연 영상

첨부파일 참고

Ⅲ. 결 론

1. 향후 목표

Fishify 시스템에서는 현재 수로를 대신하여 컨베이어 벨트를 사용하고 일반 웹캠으로 치어를 감지하였으며, 치어의 모양을 기준으로 데이터를 학습시켰다. 이점을 보완하여 실제 치어 분류에 적용하기 위하여 아래와 같은 향후 목표를 제안한다.

1) 적외선 센서 도입 및 무늬 판별 기술 적용

투명하고 작은 외관(약 2mm ~ 12mm)을 가지고 있는 대부분의 치어를 고려해 보았을 때, 내부 패턴(장기의 구성)을 통한 구분도 요구되며 정확한 감지가 필요하다. 이를 고려해 보았을 때 육안으로 보이는 레이저가 아닌, 초 근접에서 적외선 센서를 이용하면 보다 정확한 시스템 구축이 가능할 것으로 기대된다.

2) 빅데이터 분석 기술 적용

치어 데이터를 대량으로 수집하고 분석하여 치어 분류에 대한 통계 및 예측 시스템을 개발하면 더 높은 정확도를 기대할 수 있다. 빅데이터를 수집하고 딥 러닝 기술을 활용해 만든 시스템은 빠르고 정확한 분류가 가능하다. 이를 통해 치어 분류 작업을 자동화하여 효율성을 높일 수 있고, 이 시스템뿐만 아니라 다른 분야에서도 적용 가능할 것으로 기대된다.

2. 기대 효과

1) 치어들을 정확한 수치와 학습된 데이터로 관리가 가능

현재 치어들을 분류할 때 육안으로 구별하기 어려워 정확한 수치파악이 불가하고 관리가 어렵다. 그러나 본 시스템을 도입하면 치어에 대하여 데이터화가 가능하다. 이를 통해 치어 분류의 정확도와 데이터 관리의 효율성이 대폭 향상될 것으로 전망된다.

2) 관상어 산업 및 양식업 산업 활성화

앞서 언급한대로, 최근 관상어에 대한 수요 증가와 함께 국내 자생력의 확대 및 보다 효율적인 양식업 산업 구축에 대한 노력이 진행되고 있다. 이러한 추세를 따라, 본 시스템의 적용이 관상어 산업 및 양식업 산업의 활성화를 견인할 것으로 기대가 된다. 또한, 이를 통해 다양한 관상어의 양식 및 판매가 이루어 질 수 것이라고 본다.

3) 다양한 산업 적용 가능

본 시스템에서는 머신 러닝을 기반으로 한 비전 시스템(Vision System)이 주된 기술로 적용되었다. 시스템에서 활용된 학습된 데이터는 치어 분류뿐만 아니라 개인 및 일반 수족관에서도 적용이 가능하다. 더불어, 관상어의 성장 및 스트레스 관리 등 다양한 측면에서 딥 러닝을 통한 제어가 가능할 것으로 기대가 된다. 이러한 시스템 적용으로 인해, 단순한 환경 조절뿐만 아니라 관상어의 성장과 생태계의 유지에도 기여할 것으로 예상된다.

IV. 참고 자료

• 참고 문헌

- 김두철, 「양식 치어 계수기 개발」, (중소기업청, 제주대학교, 2010).
- 곽도흔, 「해수부, 관상어 산업 2025년까지 6500억 원 규모로 키운다」, 『이투데이』, 2021년 03월 22일.
- 이성종, 「관상용 스마트 수조 개발 및 상품화」, 해양수산부, 2020, 9-17.
- 전성우, 「YOLO 기반 차선 검출 시스템」 (석사학위논문, 배재대학교, 2020), 15-26.
- 서영진, 『사물인터넷을 위한 리눅스 프로그래밍 with 라즈베리파이』 (제이펍, 2021), 510-45.
- 황선규, 『OpenCV 4로 배우는 컴퓨터 비전과 머신러닝』 (길벗, 2022), 124-131, 170-76.
- 박진현 외 3명, 「어종 분류를 위한 CNN의 적용」, (한국정보통신학회논문지, 2019), 23(1), 39-46.
- Glenn Jocher, "yolov5", 18 February 2023, <https://github.com/ultralytics/yolov5> (2023년 2월 25일 검색).
- 유(YOO), "Colab을 이용한 YOLOv5 Custom 학습 및 Inference", 2021년 5월 17일, <https://why-you-story.tistory.com/m/35> (2023년 3월 2일 검색).
- PtQy5 Maria DB 연동, "파이썬 MySQL 연동, DB 엑셀 데이터 업로드 다운로드", 2019년 1월 6일, <https://myjamong.tistory.com/53> (2023년 3월 13일 검색).

• 소스 코드

첨부파일 참고