

# Hiker Guardian

구주헌, 유광재, 이민규, 최원빈

## [목 차]

<b>I. 서</b>	<b>론</b>	<b>3</b>
1.1	개발 개요	3
1.1.1	개발 배경 및 목표	3
1.1.2	작품 개요	4
<b>II. 본</b>	<b>론</b>	<b>5</b>
2.1	개발 환경 및 개발 도구 설명	5
2.1.1	전체 개발 환경	5
2.2	작품 구성	6
2.2.1	부품 리스트	6
2.2.2	회로도	7
2.2.3	시스템 구성도	8
2.2.4	작품 외관	8
2.3	보드별 동작 흐름도	9
2.3.1	전체 흐름도	9
2.3.2	Jetson Nano	10
2.3.3	Arduino	10
2.3.4	Raspberry Pi	11
2.4	개발 과정 및 주요 기능	12
2.4.1	단계별 진행 과정	12
2.4.2	주요 동작 및 특징	12
2.5	구현 결과	20
2.5.1	시연 영상	20
<b>III. 결</b>	<b>론</b>	<b>20</b>
3.1	결론	20
3.2	발전 가능성	20
<b>IV. 참고</b>	<b>자료</b>	<b>21</b>
4.1	참고 문헌	21
4.2	소스 코드	21

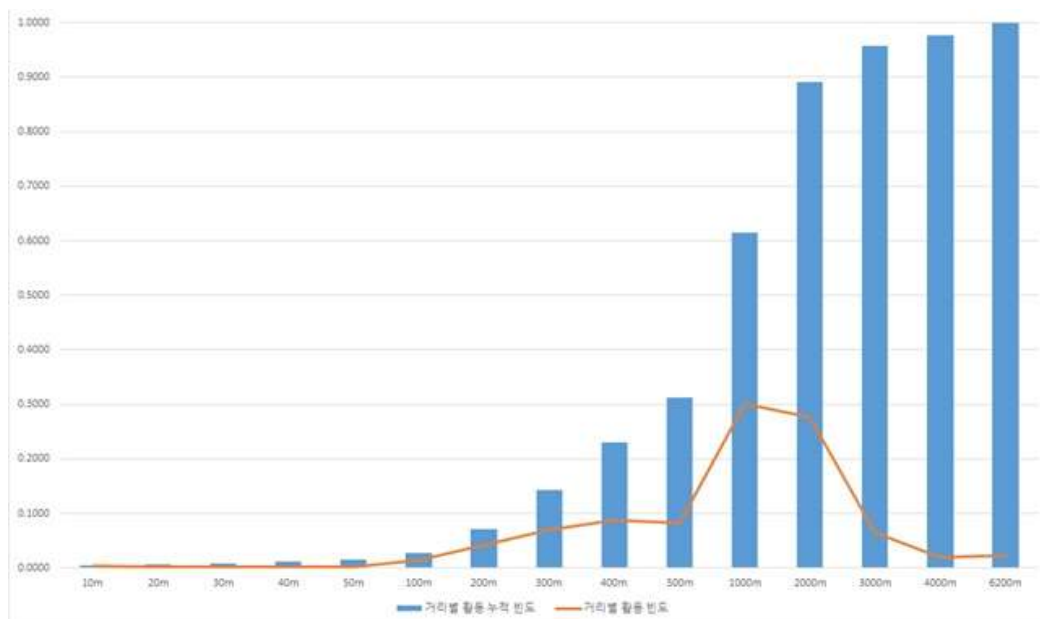
# I. 서 론

## 1.1 개발 개요

### 1.1.1 개발 배경 및 목표

최근 한 남성이 반달가슴곰에게 먹이를 주다 물려 팔이 절단된 사고가 발생하였다. 반달가슴곰이 불곰 같은 다른 대형 포유류와 비교했을 때 상대적으로 작은 체구로 인해 안전하다는 인식도 원인 중에 하나로 파악된다. 그러나 반달가슴곰은 몸길이 130~190cm, 수컷은 최대 200kg 까지 성장한다고 한다. 일본 등 해외에선 반달가슴곰에 의한 인명피해가 꾸준히 발생한다고 한다.

현재 국립공원공단에 따르면 국내 반달가슴곰은 총 86 마리로 확인되었다. 이는 최적 60 마리, 최대 78 마리 생활권을 넘어 포화 상태에 이르렀다는 분석도 나오고 있다. 또한 2017 년 경북 김천의 수도산에서 발견된 반달가슴곰의 위치추적기는 작동하지 않는 상태였고 새로 태어난 새끼 곰을 모두 추적하기엔 무리가 있는 상황이다.



[그림 1] 탐방로 이격거리별 반달가슴곰 활동 빈도

국립공원공단이 분석한 자료[그림 1]에 따르면 반달가슴곰이 지리산 탐방로 10m 이내에서 관찰된 빈도는 0.44%, 1km 이내는 61.43%라고 한다. 반경 200m 이후부터는 빈도가 점차 상승하고 꾸준히 야생동물 관련 사고가 발생하는 만큼 대책이 필요하다고 판단하였다.

따라서 등산로 주변에서 반달가슴곰을 포함한 야생동물이 나타났을 때 퇴치 시스템과 등산객들이 미리 알림을 받고 대처 방법을 상기시킬 수 있는 어플리케이션과 관련 정보를 데이터 베이스에 저장하여 분석할 수 있는 시스템을 제작하기로 하였다.

### 1.1.2. 작품 개요

이 시스템은 통행량이 많은 등산로 또는 야생동물이 자주 출몰하는 지점에 야생동물 출현을 알림과 동시에 야생동물이 등산로 주변에 접근하지 못 하도록 경보음과 퇴치 스프레이를 분사한다. 또한, 어플리케이션을 설치한 사용자에게 출현 알림과 위치정보를 전송한다.

야생 동물 감지를 위해 Jetson Nano를 이용해 모델 데이터를 학습시키고, 이를 통해 객체 감지가 이루어진다. 만약 학습된 데이터가 감지된다면 감지된 객체 수, 카메라 ID, 감지된 시간과 위치정보를 데이터 베이스에 전송한다.

본 시스템은 소켓 통신을 사용한다. Raspberry Pi4는 전체 시스템의 서버 역할을 하며, 데이터 베이스에 추가되는 데이터를 감시하고 만약 추가된 객체 감지 데이터 정보가 있다면 이를 사용자 어플리케이션과 Arduino에 전송한다.











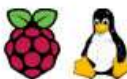








Arduino는 Raspberry Pi4와 블루투스로 연결하고 데이터를 전송받는다. 전송된 ID가 설정된 ID와 일치한다면 야생동물 출현을 알림과 퇴치를 위해 경고음을 울리고 LCD에 메시지를 출력하며, 퇴치 스프레이를 작동시켜 야생동물을 쫓아낸다.

사용자 어플리케이션은 클라이언트로, 로그인 되어있는 사용자에게 감지된 위치정보를 알리고, 야생동물을 마주쳤을 시 대처 방법을 제공한다.

## 표. 본 론

### 2.1 개발 환경 및 개발 도구 설명

#### 2.1.1 전체 개발 환경

	Device1	Device2	Main Server	Application
Main Server				
Language				
OS				
Library	Servo.h SoftwareSerial.h Wire.h LiquidCrystal_I2C.h			
Tool				

[그림 2]

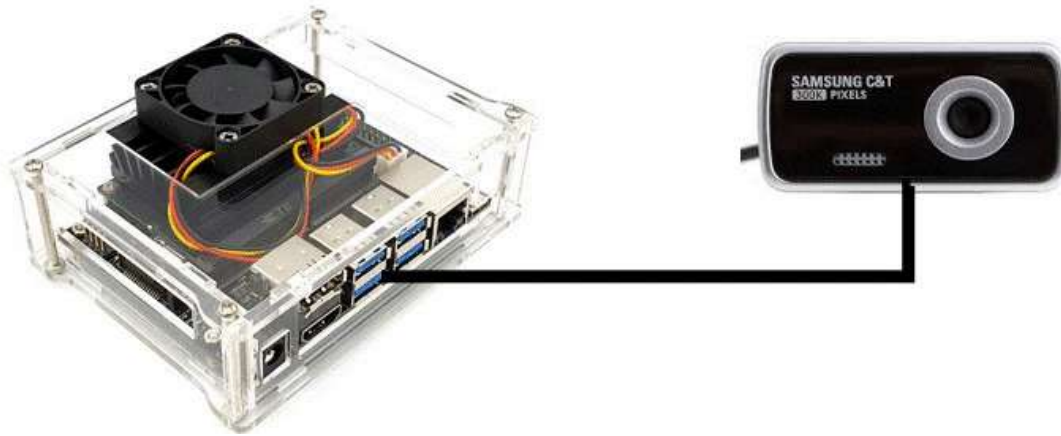
## 2.2 작품 구성

### 2.2.1 부품 리스트

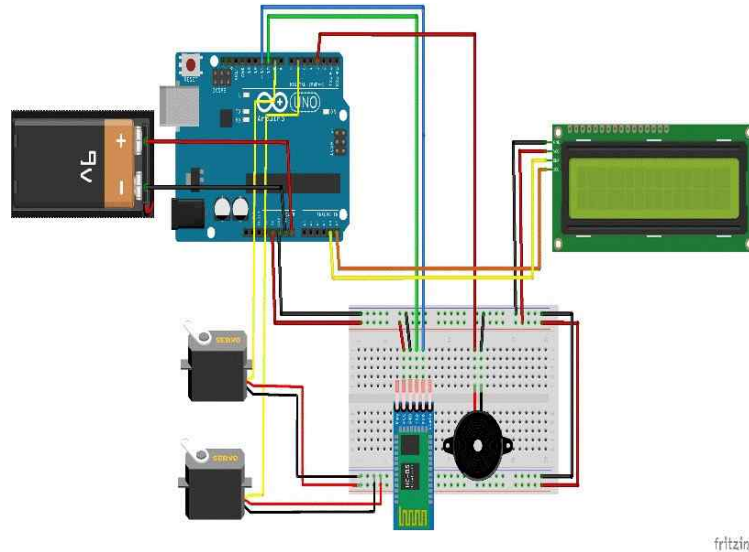
시스템	부품	이미지	링크(제품 코드)
영상	Jetson Nano		<a href="#">1376882</a>
	Jetson Nano 아크릴 팬 케이스		<a href="#">12494766</a>
	PLEOMAX W-210		<a href="#">개별 구매</a>
아두이노	Arduino Uno		<a href="#">1245596</a>
	블루투스 모듈		<a href="#">1376882</a>
	미니 서보모터		<a href="#">1128421</a>
	LCD		<a href="#">12500006</a>
	부저		<a href="#">2733</a>
DB & SERVER	Raspberry Pi 4		<a href="#">12234533</a>

## 2.2.2 회로도

### Jetson Nano 회로도

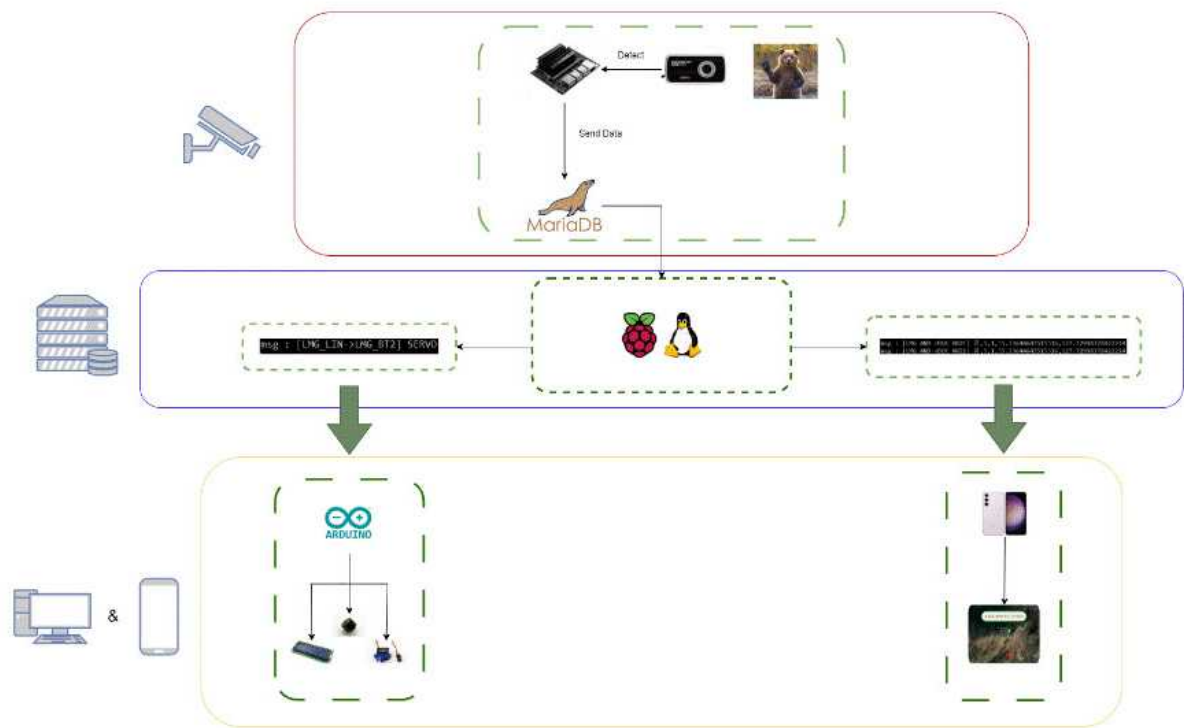


### Arduino 회로도



하드웨어 회로도 [그림 3], [그림 4]

### 2.2.3 시스템 구성도



[그림 5] 전체 시스템 구성도

### 2.2.4 작품 외관



[그림 6] 작품 외관1



[그림 7] 작품 내부 구성

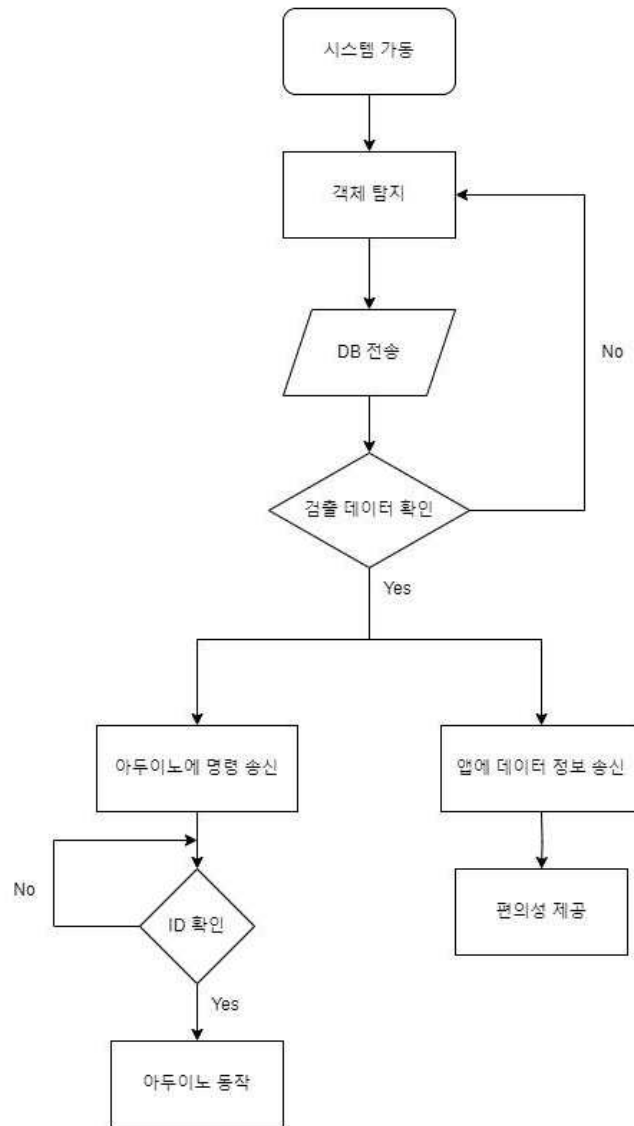


[그림 8] 작품 외관2



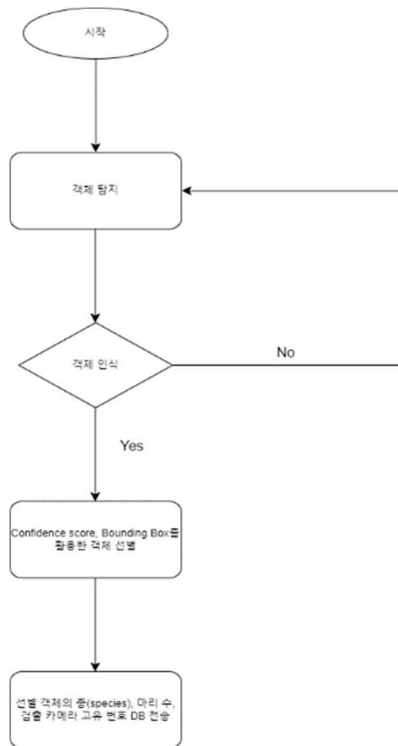
## 2.3 보드별 동작 흐름도

### 2.3.1 전체 흐름도



[그림 9] 전체 흐름도

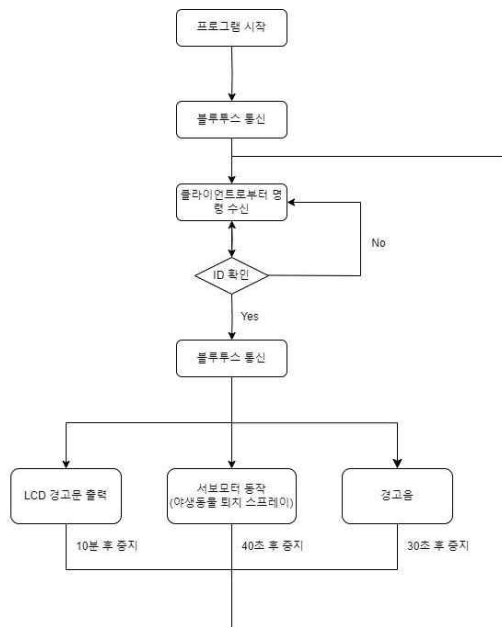
### 2.3.2 Jetson Nano



[그림 10] Jetson Nano 흐름도

Jetson Nano는 Yolo를 이용하여 객체를 탐지하는 주요한 역할을 담당한다. 선정한 기준으로 객체의 분류를 반복하며, 최종적으로 선별된 객체의 종(Species), 마리 수 및 검출 카메라 고유 번호를 데이터 베이스로 전송한다.

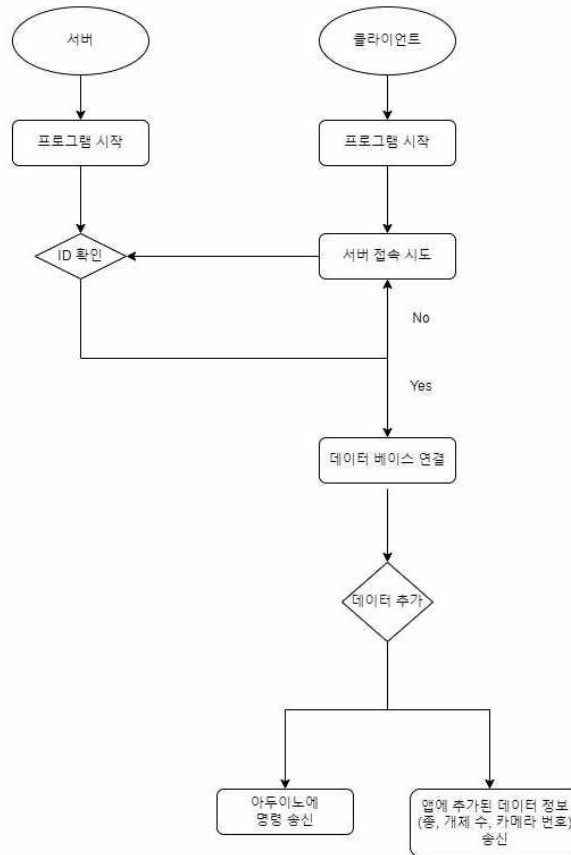
### 2.3.3 Arduino



Jetson Nano로부터 새로운 데이터가 데이터 베이스에 추가되면 메인 서버의 Raspberry에 블루투스 클라이언트의 ID와 명령을 수신 받는다.

해당 ID의 블루투스 클라이언트는 블루투스 통신으로 연결되어 있는 Arduino에 명령을 전달하고 Arduino는 LCD, 서보 모터, 부저를 각각 10분, 40초, 30초 동안 제어를 한다

#### **2.3.4 Raspberry Pi**



[그림 12] Raspberry Pi 흐름도

Jetson Nano로부터 데이터를 수신 받을 메인 서버와 앱, Arduino에 연결할 클라이언트를 생성한다. 올바른 ID의 클라이언트가 접속을 시도하면 서버는 클라이언트의 접속을 허용한다. 클라이언트는 서버에 접속을 하면 데이터 베이스와 연결을 하고 Jetson Nano로부터 새로운 데이터가 데이터 베이스에 추가될 때마다 트리거를 이용하여 서버에 클라이언트 ID와 명령어, 데이터 정보를 송신한다. 해당 ID의 클라이언트는 Application이나 Arduino에 각각 명령어와 데이터 정보를 송신한다.

## 2.4. 개발 과정 및 주요 기능

### 2.4.1 단계별 진행 과정

분류	작업제목	3.15~3.16	3.17~3.18	3.19~3.20	3.21~3.22	3.23~3.24	3.25~3.26	3.27~3.28	3.29~3.31
주제 선정 및 착수	주제 선정 및 구체화								
	보드 조사 및 구매물품 선정								
	피드백 및 최종계획 수립								
제품 개발 (영상 처리)	Jetson Nano 및 opencv 개발환경 셋팅								
	Yolov5를 이용한 야생 동물 모델 학습								
	학습된 모델 Jetson Nano에 적용								
제품 개발	데이터 결과에 따른 모터 제어								
	Raspberry pi 서버와 연결								
	제품 회로 배치 및 외관 제작								
제품 개발 (앱, 웹)	DB 구축								
	웹서버 구축								
	어플리케이션 개발								
최종 종합 검증	전체 구동 검사								
	오류확인 및 해결								
	프로젝트 최종 보고서 작성								

[그림 13] 단계별 개발 과정

## 2.4.2 주요 동작 및 특징

### 1) Arduino 제어

Arduino 시스템에선 Arduino와 블루투스를 통해 연결된 Raspberry Pi의 ID로 명령어가 들어오게 되면 부저, 서보 모터, LCD가 동작하게 된다.

각각의 기능으로는 부저는 대표적으로 지리산에 서식하는 반달곰이 싫어하는 소리를 낼 수 있도록 최대한 종소리와 비슷하게 구현하도록 노력했다.

서보 모터는 동물 퇴치 기능을 구현한 스프레이와 연결하여 약 40초 동안 동작하며 스프레이를 분사한다.

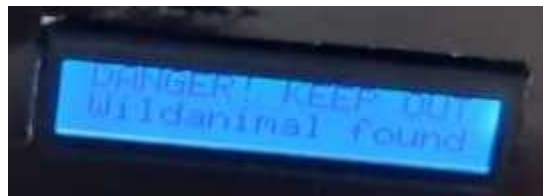


[그림 14] 서보모터와 연결된 스프레이



[그림 15] 스프레이 분사

마지막으로 LCD에선 최근 10분 안에 야생 동물이 탐지된 적이 있는지 text를 통해 알려주게 된다



[그림 16] 탐지 문구

## 2) Jetson Nano를 활용한 객체 감지

### 2-1) Yolo Model 생성

목표 객체를 인식하기 위해 Roboflow 를 이용한 곰, 멧돼지, 고라니의 Data 를 3000 장 이상 수집 및 labeling 작업 이후 생성한 모델을 활용해, YOLOv5 를 이용한 작업을 진행하였으며, 검출 결과는 아래[그림 17], [그림 18]와 같다.



[그림 17] Bounding Box 1



[그림 18] Bounding Box 2

## 2-2) Bounding Box 기준 설정

단순히 검출이 되었을 때, 데이터 베이스에 전송이 된다면, 데이터가 무분별하게 전송될 것이고 이는 편의성 증대를 위해 진행한 프로젝트의 취지에 부적절하다고 판단하였다. 따라서 센서를 이용한 검출 객체의 접근 거리를 판단하는 기능을 추가하여 이를 개선하고자 하였으나 적외선 등의 거리 센서는 야외에서의 정확성이 낮아지는 단점이 있으며, 특히 산에서는 나무와 돌 등, 변수로 작용할 요소가 많기에 실질적인 사용은 불가능하다고 판단하였다.

이에 따라 JetsonNano 를 이용한 객체 선별 과정에서 특정 기준의 Bounding Box 의 크기를 기준으로 객체의 접근 거리를 판단하는 알고리즘을 설계하여 해당 문제점을 개선해 보고자 하였다.

아래 [그림 19]과 같이 Bounding Box 기준 선정 과정은 높이 1.8m ~ 1.9m 모델의 자세조정으로 4 족 보행을 하는 반달 가슴곰을 비롯하여, 선정한 동물 객체들의 크기 평균치에 최대한 근접하게 맞추었으며, 카메라의 위치는 모델 기준, 직선 거리 6m, 지면 기준, 높이 1.9m 에서 진행하였다. 이를 통해 [그림 20]와 같이 0.0043 의 Bounding Box 의 크기가 검출 되었고 따라서 선별된 객체의 Bounding Box 크기가 0.0043 보다 크다면, 6m 이내로 접근했다는 기준 하에 프로젝트를 진행하였다.

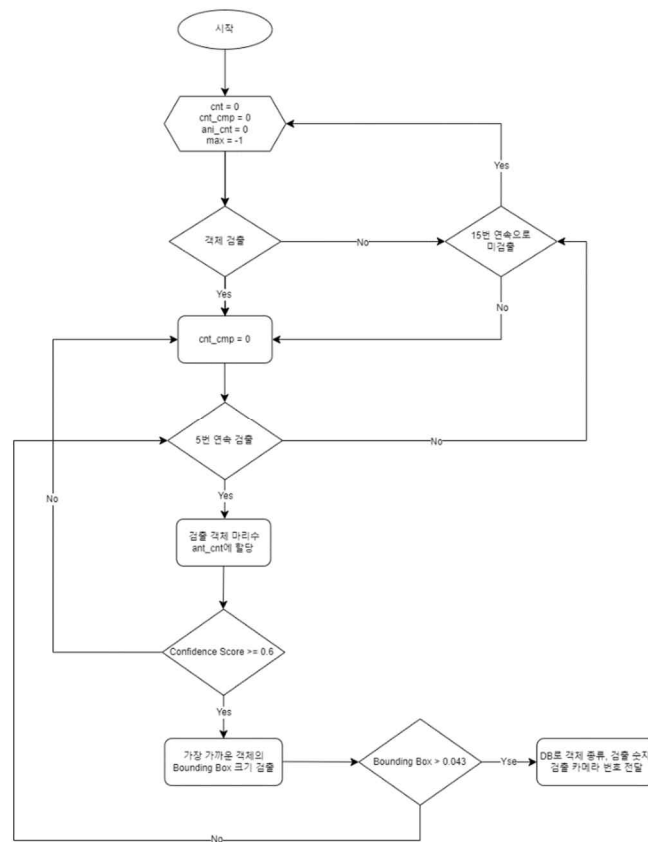


[그림 19] 모델 기준 Bounding Box

```
ptop, 1 clock, (cnt_det): 4  
area : 0.043
```

[그림 20] Bounding Box 크기 검출 예시

## 2-3) 객체 선별 알고리즘 설계



[그림 21] 객체 선별 알고리즘 흐름도

Data Base로 전송되는 객체의 선별 알고리즘은 위 [그림21]를 참고하면 된다. 우선 카메라가 작동을 시작하면, cnt(누적 객체 검출 횟수), cnt\_cmp(누적 객체 미검출 횟수), ani\_cnt(선정 동물 마리 수), max(선별 동물들 중 가장 큰 Bounding Box 크기)를 각각 초기화 이후 객체 검출 과정이 실시된다. 만약 15번 연속으로 객체가 미검출 될 경우, [그림 22]와 같이 cnt가 초기화가 되고, 5번 연속으로 검출 시 검출 객체 숫자가 ani\_cnt에 할당된다. 다만 한번이라도 검출이 된다면 cnt\_cmp는 즉시 초기화 되며, 이후 검출된 동물들 중 Confidence Score가 0.6보다 크거나 같은 동물이 한마리라도 존재할 시에는 Bounding Box의 크기를 검출하는 알고리즘으로 진입한다. 그 중 0.043보다 큰 동물의 종(species), 동물의 마리 수를 비롯해 검출된 카메라의 고유번호의 정보를 [그림 23]와 같이 데이터 베이스로 전송한다.

```
0: 480x640 1 Bear, (cnt_det): 1
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 1
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 2
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 3
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 4
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 5
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 6
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 7
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 8
50: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 9
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 10
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 11
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 12
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 13
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 14
0: 480x640 (cnt_det): 1, 0: 480x640 (cnt_ndet): 15
0: 480x640 (Reset cnt_det): 0
```

[그림 22] cnt 초기화

```
0: 480x640 1 water deer, (cnt_det): 1
0: 480x640 1 Boar, (cnt_det): 2
0: 480x640 1 Boar, (cnt_det): 3
0: 480x640 1 Boar, (cnt_det): 4
0: 480x640 1 Boar, (animal_cnt): 1, 0: 480x640 1 Boar, (Send D
B)
```

[그림 23] 데이터 베이스 전송



### 3) 데이터 베이스 및 서버-클라이언트

Jetson Nano에서 야생동물을 감지해 지리산 테이블에 데이터를 저장한다. 그 뒤 저장된 데이터의 camera\_id를 확인한 뒤에 트리거를 활용해 블루투스 클라이언트의 ID를 분류하고 테이블에 문자열을 저장한다. 트리거에서는 새로운 데이터 값의 camera\_id, 명령어, 종, 위치 정보와 같은 중요한 데이터 값들을 문자열 형태로 합친 뒤 명령어를 저장하는 테이블에 들어가도록 구성하였다.

```
MariaDB [wild]> CREATE TRIGGER wild_animal_insert_trigger_and2 AFTER INSERT ON wild_animal2 FOR EACH ROW
-> BEGIN
->   DECLARE message_text VARCHAR(255);
->   DECLARE i INT;
->   SET i = 1;
->   WHILE i <= 2 DO
->     SET message_text = CONCAT('[BUK_AND', i, ']', NEW.species, ',', NEW.count, ',', NEW.camera_id, ',', NEW.latitude, ',', NEW.longitude, '\n');
->     INSERT INTO and_message_queue2 (message) VALUES (message_text);
->     SET i = i + 1;
->   END WHILE;
-> END//

-> SET message_text = CONCAT('[LMG_BT', NEW.camera_id, ']SERVO\n');
```

[그림 24, 25] 트리거의 구성1, 2

아래 [그림 26]는 데이터가 추가된 뒤 3번째열인 camera\_id 값에 따라 테이블에 메시지가 저장되는 예시이다.

우선 지리산만을 위한 테이블 하나를 만들었고 다른 산의 데이터도 추가하고 싶으면 테이블을 추가로 생성해 [그림 27]의 구조와 같이 테이블을 생성하면 된다. 해당 [그림 28]에서의 camera\_id는 1이고 명령어 저장 테이블에 저장되는 블루투스 클라이언트의 ID는 LMG\_BT1, 명령어는 SERVO이다.

명령어가 저장되는 테이블은 선입선출 구조로 만들어 저장되어 먼저 들어온 데이터부터 처리되게 하였다.

메트덱지	1	1	2024-03-29 21:32:04	35.3215445456	127.155354546464
------	---	---	---------------------	---------------	------------------

[그림 26] 지리산 테이블에 데이터가 추가된 예시

Field	Type	Null	Key	Default	Extra
species	varchar(20)	YES		NULL	
count	varchar(20)	YES		NULL	
camera_id	varchar(20)	YES		NULL	
createdAt	timestamp	NO		current_timestamp()	
latitude	varchar(20)	YES		NULL	
longitude	varchar(20)	YES		NULL	

[그림 27] 지리산 테이블 구조

id	message
740	[LMG_BT1]SERVO

[그림 28] 명령어가 저장되는 테이블

이후 클라이언트가 서버에 접속해 있는 상황이면 [그림 29]의 테이블에서 저장되어있는 메시지를 가져오고 가져온 메시지를 서버에 전송한다. 메시지가 서버에 전송되면 해당 ID의 디바이스(안드로이드 클라이언트, 블루투스 클라이언트) 메시지를 수신 받고 메시지에 포함되어 있는 명령어를 실행하거나 데이터 정보를 활용한다.

```
while (1) {
    // 데이터베이스에서 새로운 메시지를 가져오는 쿼리 실행
    if (mysql_query(conn, "SELECT * FROM message_queue2") != 0) {
        fprintf(stderr, "mysql_query() failed\n");
        continue;
    }

    result = mysql_store_result(conn);
    if (result == NULL) {
        fprintf(stderr, "mysql_store_result() failed\n");
        continue;
    }

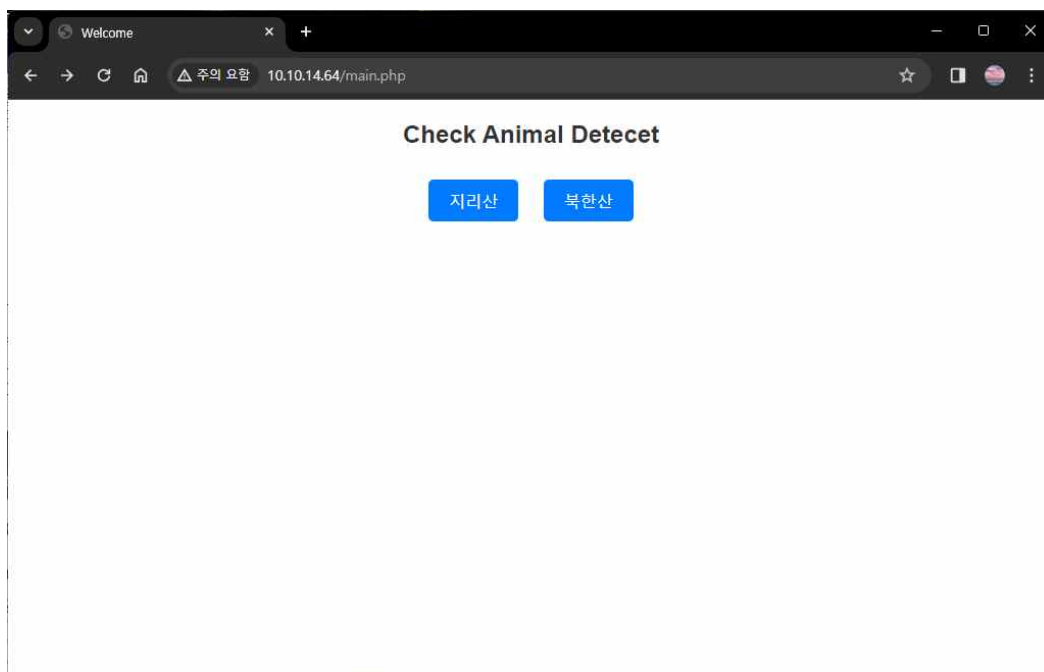
    // 가져온 메시지를 서버에 전송
    while ((row = mysql_fetch_row(result)) != NULL) {
        write(*sock, row[1], strlen(row[1]));
        mysql_query(conn, "DELETE FROM message_queue2");
    }

    mysql_free_result(result);
    sleep(1); // 1초마다 데이터베이스 확인
}
```

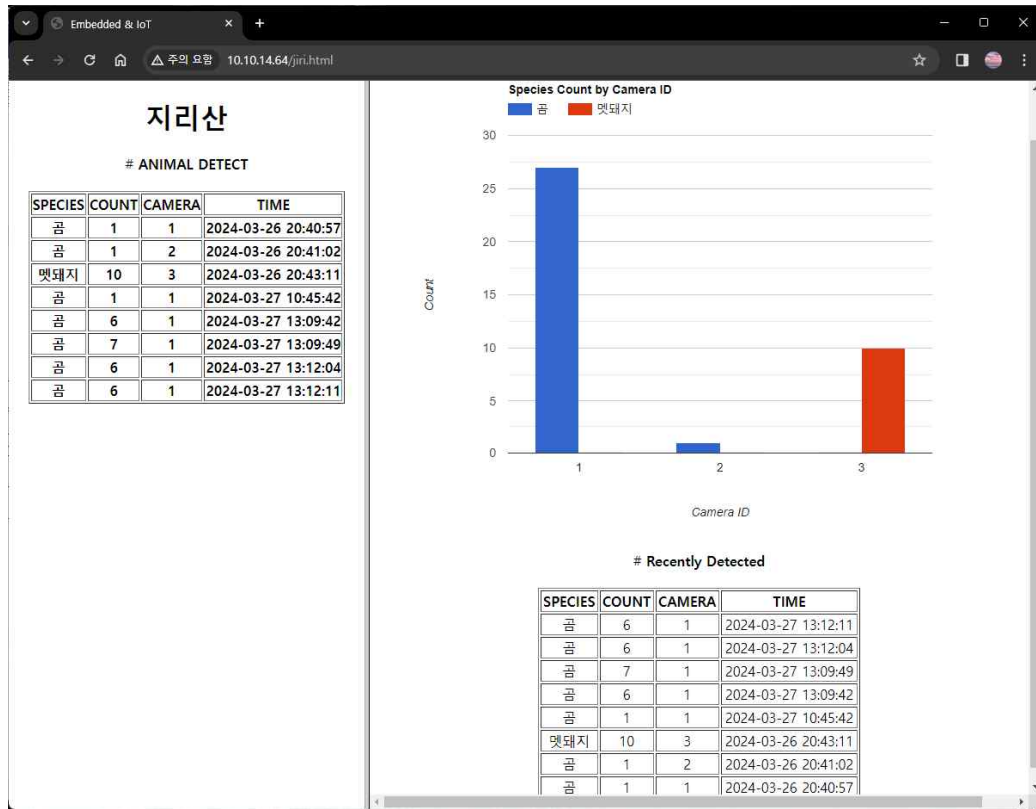
[그림 29] 클라이언트

#### 4) 웹 서버

웹 서버에서는 각각 산에 있는 데이터를 볼 수 있는 테이블과 Camera ID 별로 동물이 출현한 빈도를 확인할 수 있는 통계 그래프를 볼 수 있다. 이를 통해 여러 데이터 분석에 용이하게 사용이 가능하다. 그리고 최근 감지 되었던 동물 또한 확인이 가능하다.



[그림 30] 메인 홈페이지



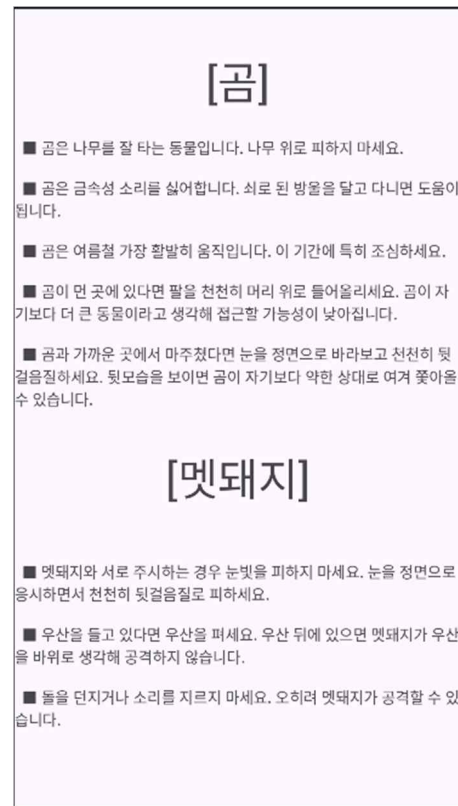
[그림 31] 각 게시판 접속 화면

## 5) 어플리케이션을 활용한 대처 방법

본 시스템은 야생동물을 마주쳤을 시 대처 방법과 원하는 지역을 선택하여 해당 지역의 야생동물 알람 기능을 받을 수 있다.



[그림 32] 어플리케이션 초기 화면



[그림 33] 대처 방법 문구

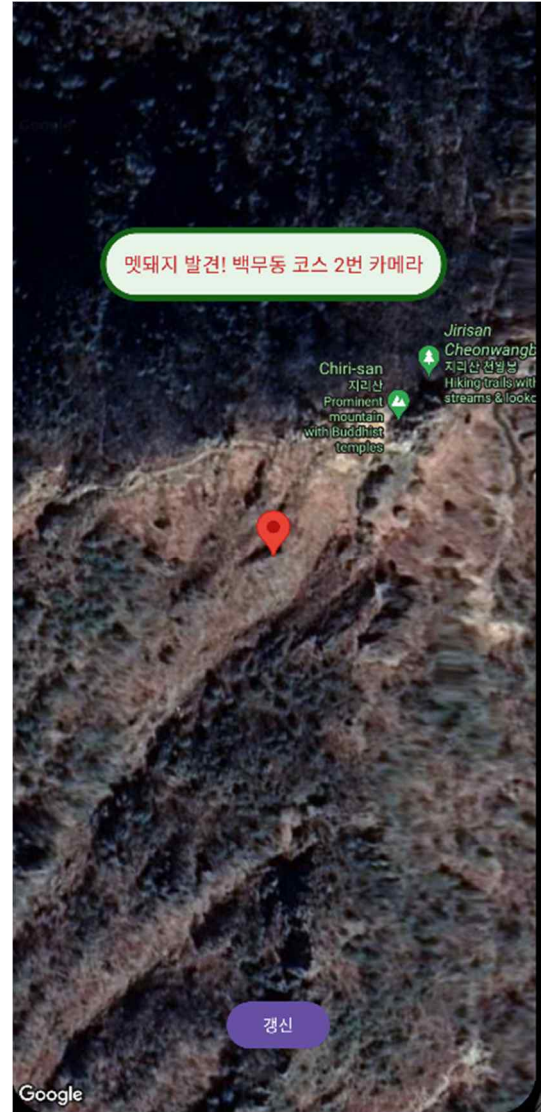
[그림 32]의 은 야생동물을 마주쳤을 시 대처 방법을 확인할 수 있는 버튼이다. 클릭하면 [그림 33]과 같이 대처 방법이 제시된다.

## 6) 어플리케이션을 통한 알림 시스템

[그림 32]의 는 특정 산의 버튼을 클릭하여 해당 산에서 발생하는 야생동물 알림을 받을 수 있다. 해당 버튼을 클릭하면 Raspberry Pi4의 서버와 소켓 통신을 하여 데이터를 받는다. 본 시스템에서는 '지리산' 버튼만 구현하였다. 그러나 '지리산' 버튼과 동일한 원리로 각 산에 해당하는 데이터 베이스 테이블에서 데이터를 받을 수 있음을 미리 밝힌다.



[그림 34] 지리산 초기 지도 화면



[그림 35] 야생동물 탐지

'지리산' 버튼을 클릭하면 지리산 중심의 지도가 화면에 나타난다. 지도는 Google Maps Android API를 활용하였다. 이때 설치된 카메라를 통해 야생동물이 감지되어 정보가 어플리케이션으로 전송되면 [그림 35]처럼 Marker 기능으로 감지된 카메라의 위치를 표시하고 토스트 메시지를 띄워 발견된 동물의 종류와 카메라 위치 정보를 알려준다. 하단의 '경신' 버튼을 누르게 되면 표시된 Marker들을 삭제한다.

## 2.5 구현 결과

### 2.5.1 시연 영상

첨부파일 참고

## Ⅲ. 결 론

### 3.1 결론

Roboflo 를 활용한 데이터셋을 학습시켜 모델을 만들고 Jetson Nano 에서 검출한 데이터를 데이터 베이스에 전송하는데 성공하였다. 이를 토대로 Arduino 와 어플리케이션은 각각 퇴치 시스템과 알림 시스템 기능 구현에 성공하였다. 등산객들은 등산을 하면서 실시간으로 야생동물 감지 알림을 받기에 등산로 주변에서 야생동물이 감지될 경우 미리 대비할 수 있을 것으로 기대된다. 또한 야생동물이 자주 출몰하는 곳을 파악하여 경계를 강화하는 데에 도움이 될 것이다.

### 3.2 발전 가능성

#### 1) 시스템 확장성

본 시스템은 지리산에서 두 개의 등산로에 총 두 대의 카메라와 연결한 상황을 가정하여 제작하였다. 그러나 각 카메라마다 위치 좌표를 활용하여 보다 더 많은 카메라와 연동이 가능할 것이다. 또한 지리산을 뿐만 아니라 설악산, 덕유산 등으로 확장하게 될 경우 해당 산에 해당하는 데이터 베이스의 테이블을 추가하여 각 산마다 야생동물 관리 시스템을 구축할 수 있게 된다.

#### 2) 분야 확장성

본 시스템은 현재 등산객 보호를 목적으로 야생동물 감지 시스템을 제작하였다. 그러나 해당 시스템을 도심이나 농가에 적용하여 보다 더 다양한 동물들을 학습한 모델을 제작하여 유해 동물이 침입할 시 퇴치 기능 구현이 가능할 것이다. 또한 데이터 베이스 분석을 토대로 자주 나타나는 동물, 발견 시기, 발견 장소 등을 파악하고 취약한 부분을 집중적으로 보완하여 환경 개선 도움될 것으로 기대된다.

퇴치 기능에 목적을 두는 것이 아닌 생태학 연구에도 효과적으로 사용될 것으로 기대된다. 특정 지역에서 멸종 위기 종의 활동을 추적하고 분석하여 연구에 활용할 수 있을 것이다. 이러한 방식으로 동물 퇴치 및 감지 시스템은 도시, 농업, 자연 보호 등 다양한 분야에서 유용하게 활용될 것이다.



## IV. 참고 자료

### 4.1 참고 문헌

- 오상훈, "곰이 팔 물고 안 놓아줘서..." 스스로 팔 절단한 남성, 헬스조선, 2024.02.01
- 정은혜, "등 보인 채 도망, 큰일 납니다" ... 지리산 곰 주의보 뒀다, 중앙일보, 2023.06.05
- 이준원, 메이커를 위한 아두이노의 모든 것, 프리렉, 2020
- 정재곤, 안드로이드 앱 프로그래밍, 이지스퍼블리싱, 2023
- 서영진, 사물인터넷을 위한 리눅스 프로그래밍 with 라즈베리 파이, 제이펍, 2023
- 윤성우, 윤성우의 열혈 TCP/IP 소켓 프로그래밍, 오렌지미디어, 2023

### 4.2 소스 코드

첨부파일 참고