시스템 설계 문서 (SDD)

프로젝트 제목: FOD 제거 및 조류 퇴치 AMR
버전: 1.2
날짜: 2025.06.30

---

1. 개요

FOD 제거 및 조류 퇴치 AMR 시스템은 공항 및 산업 환경에서 이물질(FOD) 제거와 조류 유입 방지를 위해 설계된 자율 이동 로봇 기반 솔루션으로, 객체 탐지 및 추적 기능을 통해 자동화된 순찰 및 대응을 수행합니다.

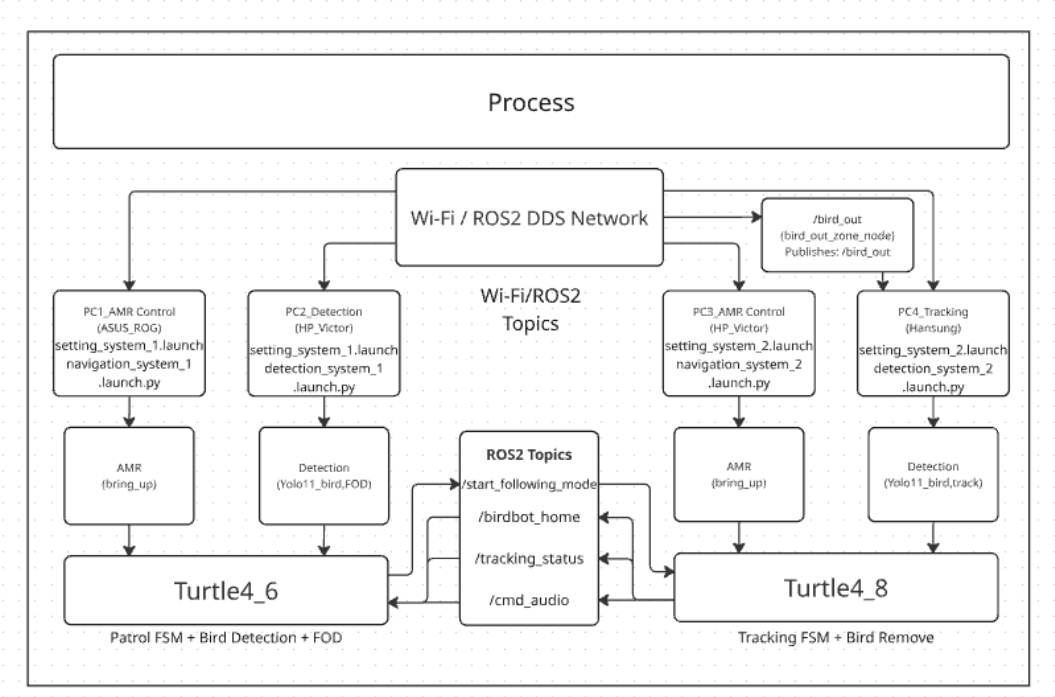
---

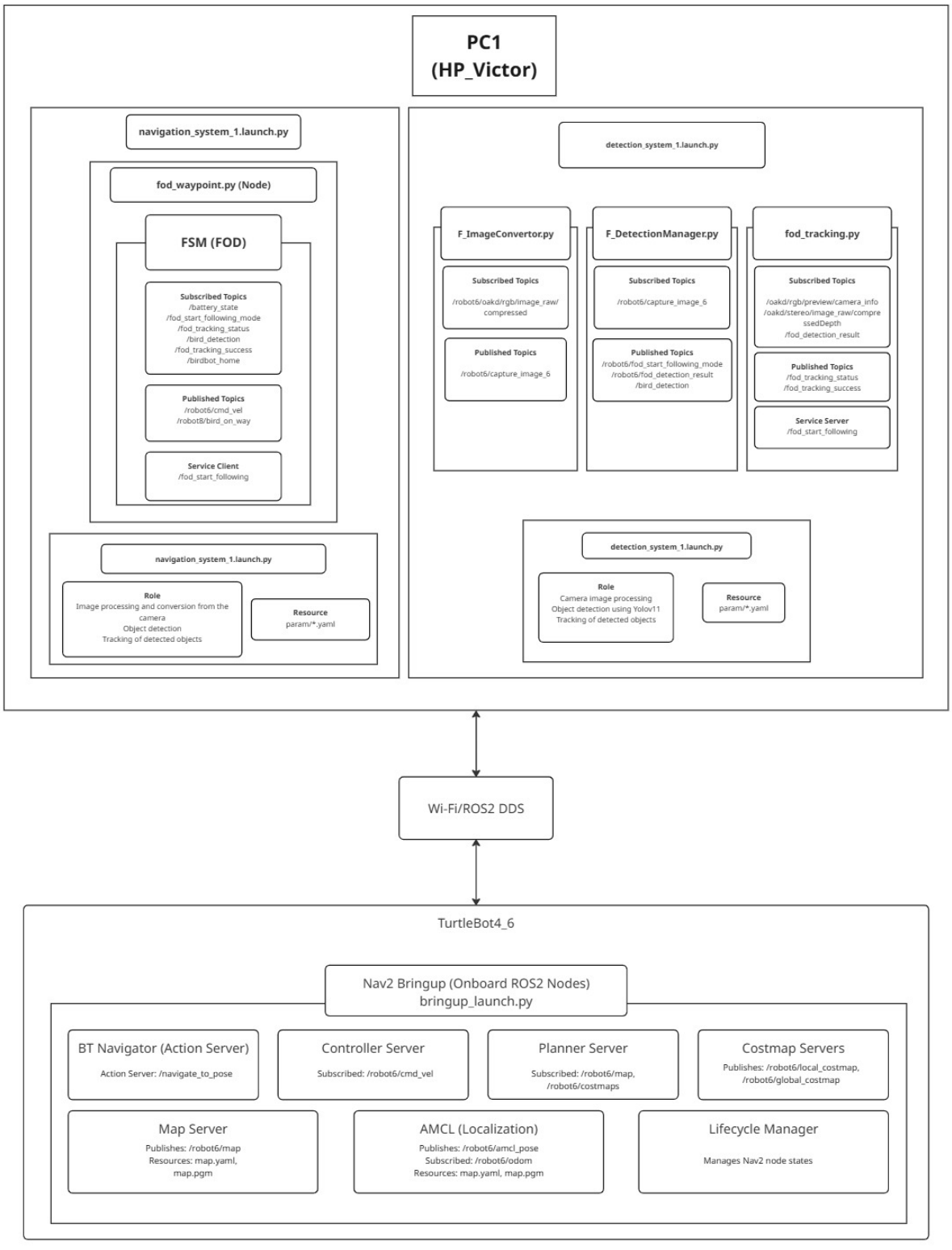
2. 시스템 아키텍처

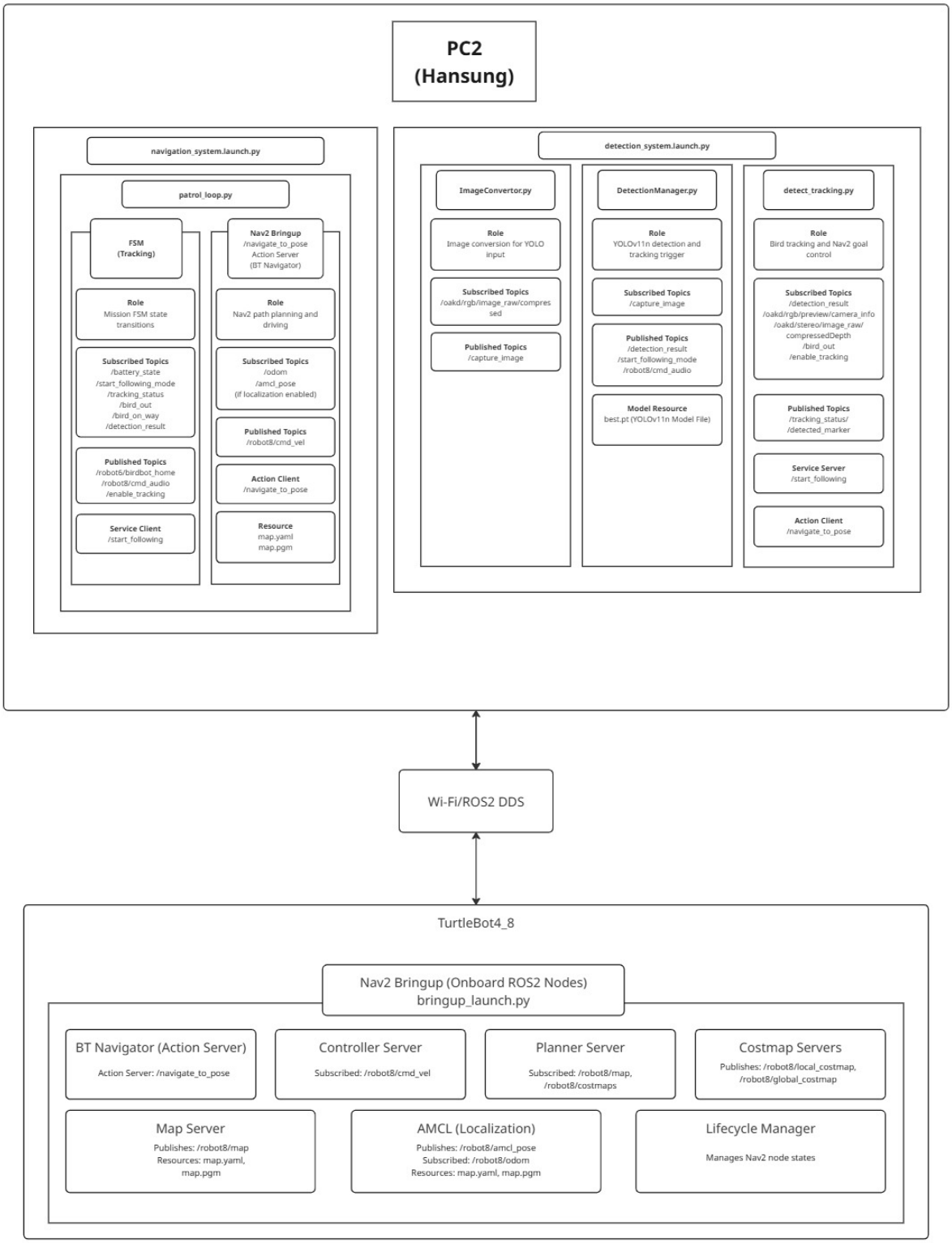
이 시스템은 2 대의 AMR TurtleBot4 로 구성되며, 데이터 처리, 네비게이션, 객체 탐지 및 추적 등 기능을 1 대의 AMR 당 2 대의 PC 에서 처리 후, WIFI 및 ROS2 DDS 통신을 보내어 RaspberryPi 를 통해 터틀봇 4 의 AMR 을 제어합니다. .

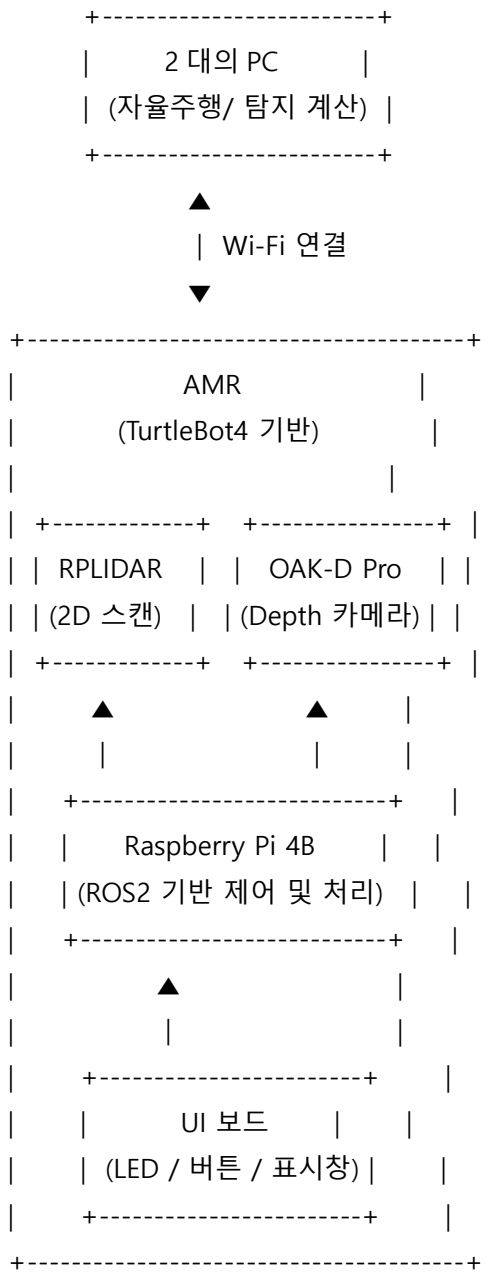
AMR 은 사용자 대시보드와 Wi-Fi 를 통해 통신하며, 실시간 상태 출력은 UI 보드로 제공됩니다.

2.1 고레벨 아키텍처 다이어그램




PC1
(HP_Victor)
navigation_system_1.launch.py
fod_waypoint.py (Node)
FSM (FOD)
Subscribed Topics
/battery_state
/fod_start_following_mode
/fod_tracking_status
/bird_detection
/fod_tracking_success
/birdbot_home
Published Topics
/robot6/cmd_vel
/robot8/bird_on_way
Service Client
/fod_start_following
navigation_system_1.launch.py
Role
Image processing and conversion from the camera
Object detection
Tracking of detected objects
Resource
param/*.yaml
detection_system_1.launch.py
F_ImageConvertor.py
Subscribed Topics
/robot6/oakd/rgb/image_raw/compressed
Published Topics
/robot6/capture_image_6
F_DetectionManager.py
Subscribed Topics
/robot6/capture_image_6
Published Topics
/robot6/fod_start_following_mode
/robot6/fod_detection_result
/bird_detection
fod_tracking.py
Subscribed Topics
/oakd/rgb/preview/camera_info
/oakd/stereo/image_raw/compressedDepth
/fod_detection_result
Published Topics
/fod_tracking_status
/fod_tracking_success
Service Server
/fod_start_following
detection_system_1.launch.py
Role
Camera image processing
Object detection using Yolov11
Tracking of detected objects
Resource
param/*.yaml
Wi-Fi/ROS2 DDS
TurtleBot4_6
Nav2 Bringup (Onboard ROS2 Nodes)
bringup_launch.py
BT Navigator (Action Server)
Action Server: /navigate_to_pose
Controller Server
Subscribed: /robot6/cmd_vel
Planner Server
Subscribed: /robot6/map,
/robot6/costmaps
Costmap Servers
Publishes: /robot6/local_costmap,
/robot6/global_costmap
Map Server
Publishes: /robot6/map
Resources: map.yaml,
map.pgm
AMCL (Localization)
Publishes: /robot6/amcl_pose
Subscribed: /robot6/odom
Resources: map.yaml, map.pgm
Lifecycle Manager
Manages Nav2 node states


PC2
(Hansung)
navigation_system.launch.py
patrol_loop.py
FSM
(Tracking)
Role
Mission FSM state transitions
Subscribed Topics
/battery_state
/start_following_mode
/tracking_status
/bird_out
/bird_on_way
/detection_result
Published Topics
/robot6/birdbot_home
/robot8/cmd_audio
/enable_tracking
Service Client
/start_following
Nav2 Bringup
/navigate_to_pose
Action Server
(BT Navigator)
Role
Nav2 path planning and driving
Subscribed Topics
/odom
/amcl_pose
(if localization enabled)
Published Topics
/robot8/cmd_vel
Action Client
/navigate_to_pose
Resource
map.yaml
map.pgm
detection_system.launch.py
ImageConvertor.py
Role
Image conversion for YOLO input
Subscribed Topics
/oakd/rgb/image_raw/compressed
Published Topics
/capture_image
DetectionManager.py
Role
YOLOv11n detection and tracking trigger
Subscribed Topics
/capture_image
Published Topics
/detection_result
/start_following_mode
/robot8/cmd_audio
Model Resource
best.pt (YOLOv11n Model File)
detect_tracking.py
Role
Bird tracking and Nav2 goal control
Subscribed Topics
/detection_result
/oakd/rgb/preview/camera_info
/oakd/stereo/image_raw/compressedDepth
/bird_out
/enable_tracking
Published Topics
/tracking_status/
/detected_marker
Service Server
/start_following
Action Client
/navigate_to_pose
Wi-Fi/ROS2 DDS
TurtleBot4_8
Nav2 Bringup (Onboard ROS2 Nodes)
bringup_launch.py
BT Navigator (Action Server)
Action Server: /navigate_to_pose
Controller Server
Subscribed: /robot8/cmd_vel
Planner Server
Subscribed: /robot8/map,
/robot8/costmaps
Costmap Servers
Publishes: /robot8/local_costmap,
/robot8/global_costmap
Map Server
Publishes: /robot8/map
Resources: map.yaml,
map.pgm
AMCL (Localization)
Publishes: /robot8/amcl_pose
Subscribed: /robot8/odom
Resources: map.yaml, map.pgm
Lifecycle Manager
Manages Nav2 node states



---

3. 구성 요소 설계

3.1 하드웨어 구성 요소

1. PC (모니터링 시스템)
    - **운영 체제:** Ubuntu 22.04

- **카메라:** FOD (Foreign Object Debris/Damage)와 bird 를 시각적 모니터링을 위한 USB 카메라
- **네트워크:** AMR 과 안정적인 Wi-Fi 연결 통신 지원

2. **자율 이동 로봇 (AMR)**
   - TurtleBot4 2 대 (ID 6 & 8)
   - **프로세서:** Raspberry Pi 4B (온보드 AI 처리 및 데이터 처리)
   - **운영 체제:** Ubuntu 22.04 및 ROS2 (로봇 제어)
   - **센서:**
     - **RPLIDAR:** 네비게이션 및 벽 감지
     - **카메라(OAK-D Pro):** 동적객체와 정적객체 탐지용
     - **배터리: 2 시간 30 분 ~ 4 시간**

3. **충전 스테이션**
   - 수동 개입 없이 AMR 의 충전을 위한 자율 도킹 스테이션

3.2 소프트웨어 구성 요소

1. PC 소프트웨어
   - **Python3:** 인터페이스 관련 작업을 위한 기본 언어
   - **ROS2:** AMR 과의 통신을 위한 플랫폼
   - **OpenCV 및 roboflow (YOLOv8 & v11):** PC 에서 추가 객체 탐지가 필요한 경우의 이미지 처리 용도
2. AMR 소프트웨어
   - **ROS2 (로봇 운영 시스템 2):** 네비게이션, 제어 및 데이터 통합의 핵심 플랫폼
   - **Python3:** AMR 작동 및 AI 처리에 사용하는 기본 프로그래밍 언어
   - **OpenCV 및 roboflow (YOLO):** Jetson-Orin 에서 실행되는 객체 감지 및 위협 분석

---

4. 데이터 흐름 설계

4.1 데이터 수집

- 카메라:
  - Oak D pro 를 이용해 다양한 거리에서 사(bird) 및 FOD 데이터를 345 장 수집하여 bounding box 를 설정하고, 640x480 으로 size 를 조절.

- AMR 센서:
    - AMR 은 LiDAR, 카메라를 사용해서 자유 네비게이션을 수행하고 FOD 를 탐지하고, 사(Bird)를 탐지한다.

4.2 데이터 처리 및 범위 및 논리

- F_ImageConvertor.py
    - OakD 의 /oakd/rgb/image_raw/compressed 에서 복잡한 형식의 카메라 이미지를 OpenCV 로 변환
    - /capture_image_6 이라는 일반 sensor_msgs/Image 통보로 보내지고, YOLO detection 의 입력으로 사용
- F_DetectionManager.py
    - YOLOv11 기반의 object detector 2 개 (bird / fod) 모델 반영
    - /capture_image 에서 사용자 이미지를 수집해 배경이 바뀌면 detection 수행
    - detection 결과 및 bounding box 를 /fod_detection_result 로 전달
    - bird 검색 성공 시 /bird_detection 도 publish
- FodTracking.py
    - /fod_detection_result 를 subscribe 하여 detection 의 bbox 와 depth 값을 통해 3D goal point 계산
    - TF2 를 이용해 공간상 transform 결과를 map 프레임으로 변환
    - 이 goal 을 navigate_to_pose 에 보내어 AMR 이동 진행

4.3 데이터 저장

- AMR 로컬 저장:
    - 각 AMR 은 FOD, 사 검색 결과와 위치 및 배터리 상황을 48 시간간 로컬에 저장
- PC 로그 저장:

    - DetectionManager 보고서에서 보내지는 detection result 및 detection 보내지 전 검색 보고를 로그로 저장

4.4 경고 및 알림 전달

- /fod_tracking_success 또는 /bird_detection 에 따라 AMR 가 /bird_on_way, /cmd_audio 통보를 발행

- 이 것은 DDS 통신 연결로 통보가 건너가 bird_out_zone_node 로 보내지면 목적 PC 에서 /bird_out 통보로 보이는 것이 가능

---

5. 상세 설계

5.1 AMR 네비게이션 및 동적, 정적 객체 탐지

- 네비게이션:
    - ROS2 와 RPLiDAR 기반의 SLAM(동시 위치 추정 및 매핑)을 사용하여 AMR 순찰 구역을 자율적으로 이동하며, 정적 장애물을 회피하고 새(Bird)와 FOD 를 제거

- 조류 탐지:
    - Detection PC 에서 실행되는 YOLOv11 기반 객체 감지를 통해 새(RC 조종 포함)와 같은 조류를 인식하며, 감지 즉시 추적 모드로 전환

- 통신 프로토콜:
    - AMR 은 로컬 Wi-Fi 를 통해 모니터링 PC 와 통신하며, 감지 이벤트 및 상태 정보를 실시간으로 전송함

- 작업 종료 판단:
    - /fod_tracking_success 또는 /bird_home 신호를 기반으로 FSM 은 추적 모드를 종료하고 원위치로 복귀
    - 감지된 객체가 지정된 관심 영역(ROI)을 벗어나면 자동 종료되며, UI 에서도 종료 상태가 표시됨

5.2 Tracking 노드

- /fod_detection_result 에서 검색된 객체 중 class=='fod' 가 존재하면 3D 위치 계산 결과를 goal 에 전달
- 내부적 배반 범위와 통과 정보를 고려해 goal update 중지
- 최종 값은 /fod_tracking_success 로 publish

5.3 Detection Pipeline

- capture_image_6 데이터가 image convertor 에서 발송
- DetectionManager 는 YOLOv11 을 통해 bird 또는 fod 발견 시 bird_detection / fod_start_following_mode publish

- Tracking 노드는 /fod_detection_result 를 subscribe 하여 3D goal point 계산 및 이동 수행

---

6. 보안 설계

- **데이터 암호화:** AMR 과 PC 간 전송되는 모든 데이터는 TLS 1.3 을 사용해 암호화됩니다.
- **접근 제어:** 대시보드는 다중 요소 인증을 통해 접근이 제한되어 있습니다.
- **로컬 저장 보안:** PC 에 저장된 로그 및 경고 데이터는 AES-256 을 사용해 암호화하여 민감한 정보를 보호합니다.

---

7. 성능 요구사항

- 반응 시간: FOD 및 새 탐지 시 /cmd_audio 또는 /bird_on_way 메시지는 1 초 내 송신 완료

- 게임 진행 시간: 추적 명령 수신부터 도착까지 약 3~10 초 (거리, 장애물에 따라 상이)

- 배터리 지속시간: 실측 기준 약 2.5~4 시간 연속 운영 가능

- 증가 가능성: 노드 분산 구조이므로 최대 4 대까지 병렬 운영 가능 (Wi-Fi / namespace 기반 확장 구조)

- 다른 장치 도입 여부: ROS2 기반 구조로 추가 센서 및 노드 유입이 유연함

---

8. 오류 처리 및 복구

- **네트워크 손실:** Wi-Fi 연결이 끊기면 AMR 은 제한된 기능으로 자율적으로 작동을 유지하며, 연결이 복원되면 데이터 전송을 재개합니다.
- **하드웨어 오류:** AMR 은 정기적으로 센서 및 하드웨어 상태를 점검합니다. 문제가 감지되면 모니터링 PC 에 오류를 보고합니다.
- **배터리 부족:** 배터리가 낮을 경우 AMR 은 자율적으로 도킹 스테이션으로 복귀합니다.

9. 테스트 및 검증

1. **단위 테스트:** ROS2 노드 및 Flask 서버를 포함한 모든 소프트웨어 구성 요소를 개별적으로 테스트하여 기능을 검증합니다.
2. **통합 테스트:** AMR 과 모니터링 PC 를 통합하여 원활한 데이터 흐름과 명령 실행을 보장합니다.
3. **사용자 승인 테스트(UAT):** 보안 담당자와 함께 대시보드, 경고 및 수동 제어 기능을 실제 상황에서 검증합니다.
4. **현장 테스트:** 목표 보안 구역에서 AMR 을 테스트하여 자율 네비게이션, 위협 탐지 및 경고 기능을 실제 조건에서 검증합니다.

---

10. 배포 및 유지보수 계획

- **배포 단계:**
  - PC 에 ROS2, Flask, SQLite3 를 설치합니다.
  - 모니터링 대시보드를 구성하고 AMR 을 Wi-Fi 로 연결합니다.
  - 초기 네비게이션 경로를 설정하고 경고 기능을 테스트합니다.

- **유지보수 일정:**
  - 매월 AMR 과 PC 의 소프트웨어 업데이트로 보안 패치 적용
  - 분기마다 AMR 의 하드웨어 점검, 센서 보정 및 배터리 검사

---

11. 부록

- **라이브러리 및 종속성:** ROS2, OpenCV, YOLO, Flask, SQLite3 등 사용된 소프트웨어 종속성 목록
- **용어 사전:** AMR, SLAM, YOLO 등 주요 용어의 정의
- **참고 문헌:** ROS2, Jetson-Orin, TurtleBot3 및 관련 기술에 대한 문서