Pusan National University
Computer Science and Engineering
Technical Report 2024-10

지능형 무인 경계 로봇 구현

202055513 김대영

202055540 박민재

201812157 조영진

지도교수 탁성우

목 차

1. 과제 개요

1.1. 과제 배경

최근 몇 년간 전 세계적으로 나타난 여러 추세들이 고급 보안 및 감시 솔루션의 필요성을 부각시켰습니다. 본 프로젝트는 세계 각지에서 직면한 세 가지 주요 과제에 대응하기 위해 구상되었습니다.

미국은 심각한 가택 침입 문제로 어려움을 겪고 있습니다. 이 문제는 주택 소유자들 사이에서 개인 안전과 재산 보안에 대한 우려를 불러일으켰습니다. 기존의 보안 시스템들은 실시간 개입 능력을 제공하는 데 종종 부족함을 보여, 더욱 정교하고 신속하게 대응할 수 있는 주택 보호 기술에 대한 수요를 창출했습니다.

또한 미국의 광대한 농업 지대는 야생동물 관리에 있어 독특한 과제를 제시합니다. 농부들은 다양한 형태의 야생동물로부터 농작물과 가축을 보호하는 데 지속적인 어려움을 겪고 있습니다. 이러한 농장들의 방대한 규모로 인해 기존의 해충 및 포식자 통제 방법은 비효율적이고 종종 효과가 없습니다. 최소한의 인력으로 넓은 지역의 야생동물을 모니터링하고 관리할 수 있는 혁신적인 솔루션이 필요한 상황입니다.

한편 한국은 국가 안보에 중대한 영향을 미치는 인구 변화에 직면해 있습니다. 국가의 출산율 저하로 인해 국경 경비 작전에 투입할 수 있는 인력이 감소하고 있습니다. 이러한 인적 자원의 감소는 국경을 효과적으로 모니터링하고 보안을 유지하는 국가의 능력에 잠재적 위험을 초래합니다. 이러한 상황은 국경 감시 및 보안에서 인간의 능력을 보완할 수 있는 기술적 솔루션을 요구하고 있습니다.

이러한 다양하면서도 상호 연관된 과제들은 다목적의 지능형 감시 시스템의 필요성을 부각시킵니다. 이러한 시스템은 주택 보안을 위한 실시간 모니터링 및 신속한 대응 능력 제공, 대규모 농업 부동산을 위한 광역 감시 제공, 그리고 감소된 인력 요구사항으로 국경 보안 작전 강화와 같은 능력을 갖추어야 합니다.

이러한 문제들을 해결함으로써, 본 프로젝트는 개인 주택에서부터 국경에 이르기까지 다양한 보안 및 감시 요구에 적응할 수 있는 최첨단 솔루션을 개발하는 것을 목표로 합니다. 궁극적인 목표는 인공지능, 컴퓨터 비전, 원격 제어 하드웨어와 같은 첨단 기술을 활용하여 효과적이고 효율적이며 확장 가능한 보안 솔루션을 제공하는 시스템을 만드는 것입니다. 이를 통해 다양한 환경과 상황에서 발생하는 보안 문제에 대해 종합적이고 혁신적인 해결책을 제시할 수 있을 것으로 기대됩니다.

1.2. 과제 목표

우리는 실시간 영상 처리와 인공지능 기술을 활용하여 침입자 탐지 및 대응 시스템을 개발할 것입니다. 이를 위해 첫 번째로 딥러닝 모델을 통해 실시간 영상에서 사람, 동물, 차량 등의 객체를 탐지하고 분류해야 합니다. 그림 1과 같이 카메라로 촬영된 영상을 프레임 단위로 분석하여 각 객체의 위치와 종류를 식별할 수 있습니다.

또한 시간의 흐름에 따라 객체들의 위치가 변화하기 때문에, 각 프레임마다 객체 탐지를 지속적으로 수행해야 합니다. 객체 탐지를 성공적으로 마쳤다면, 그 결과에서 객체의 크기, 이동 속도, 이동 방향 등의 feature를 추출합니다. 추출한 feature를 바탕으로 주어진 조건에 맞춰 모델을 훈련하여 위협 수준을 판단하고, 위협으로 분류된 객체에 대해 적절한 대응 방식을 결정합니다.

마지막으로 객체 탐지와 위협 수준 판단을 완료한 결과를 실시간으로 시각화하여 사용자에게 제공할 것입니다. 탐지 과정에서 사용된 정보를 수치화하여 사용자가 활용 가능한 추가적인 정보를 제공할 예정입니다. 더 나아가 과거의 유사한 위협 상황 데이터를 쉽게 참고할 수 있도록 시스템을 구성하고자 합니다. 수치화된 추가 정보를 기반으로 과거 사례를 쉽게 검색할 수 있는 기능을 제공할 예정입니다.

연구를 통해 제작한 객체 탐지 모델과 위협 수준 판단 모델을 바탕으로 제작한 보안 솔루션은 사용자에게 다음과 같은 이점을 제공할 것입니다.

- 실시간으로 침입자나 위협 요소를 AI 모델을 통해 1차로 탐지하여 신속한 대응 방향을 제시합니다.
- 탐지된 객체의 위치, 크기, 이동 경로를 명확하게 표시하여 상황을 쉽게 파악할 수 있습니다.
- AI가 판단한 위협 수준과 유사한 과거 사례들을 나열하여 비교 분석을 쉽게 할 수 있도록 UI를 제공합니다.
- 자동화된 대응 시스템을 통해 신속하고 효과적으로 위협을 제거할 수 있습니다.

이 시스템은 가정, 농장, 국경 등 다양한 환경에서 활용될 수 있으며, 인력 부족 문제를 해결하고 24시간 지속적인 보안 감시를 가능하게 합니다. 또한, 실시간 데이터 분석을 통해 보안 취약점을 파악하고 개선할 수 있는 인사이트를 제공할 것입니다.

1.3. 과제 내용 및 요구사항

본 프로젝트는 실시간 객체 탐지 및 대응 시스템 개발을 목표로 하며, 다음과 같은 주요 요구사항과 특징을 갖추어야 합니다.

첫째, 프레임에 탐지된 객체를 정확하게 조준할 수 있는 하드웨어 시스템을 구축해야 합니다. 이 시스템은 고해상도 카메라와 정밀한 모터 제어 장치를 포함하여, 실시간으로 탐지된 객체의 위치를 추적하고 정확하게 조준할 수 있어야 합니다. 조준 시스템은 빠른 반응 속도와 높은 정확도를 갖추어 다양한 크기와 속도의 객체에 대응할 수 있어야 합니다.

둘째, 사용자는 객체가 탐지되었을 때의 실시간 영상, 혹은 녹화한 영상을 확인할 수 있어야 합니다. 이를 위해 고성능 영상 처리 및 스트리밍 기술이 구현되어야 하며, 사용자가 언제 어디서나 모바일 기기나 컴퓨터를 통해 실시간 영상을 확인할 수 있어야 합니다. 또한, 중요한 이벤트 발생 시 자동으로 영상을 녹화하고 저장하는 기능이 필요하며, 사용자가 이러한 녹화 영상을 쉽게 검색하고 재생할 수 있는 인터페이스를 제공해야 합니다.

셋째, 탐지된 객체를 지속해서 추적할 수 있어야 합니다. 이는 고급 컴퓨터 비전 알고리즘을 활용하여 구현되어야 하며, 객체가 일시적으로 가려지거나 화면에서 사라졌다가 다시 나타나는 경우에도 지속적인 추적이 가능해야 합니다. 또한, 여러 객체가 동시에 탐지될 경우 각 객체를 개별적으로 식별하고 추적할 수 있어야 합니다. 추적 과정에서 객체의 이동 경로, 속도, 방향 등의 정보를 실시간으로 분석하고 이를 사용자에게 제공해야 합니다.

이러한 기능들을 구현하기 위해서는 강력한 하드웨어 성능과 최적화된 소프트웨어가 필요합니다. 시스템은 24시간 연속 작동이 가능해야 하며, 다양한 환경 조건에서도 안정적으로 동작해야 합니다. 또한, 사용자 친화적인 인터페이스를 통해 시스템의 설정을 쉽게 변경하고 상태를 모니터링할 수 있어야 합니다.

마지막으로, 시스템은 확장성을 고려하여 설계되어야 합니다. 향후 새로운 기능 추가나 성능 향상이 용이하도록 모듈화된 구조를 가져야 하며, 다양한 환경에 적용될 수 있도록 유연성을 갖추어야 합니다. 이러한 요구사항들을 모두 충족시킴으로써, 본 프로젝트는 효과적이고 신뢰할 수 있는 실시간 객체 탐지 및 대응 시스템을 구현할 수 있을 것입니다.

2. 과제 구성

2.1. 배경지식

2.1.1. 라즈베리파이 GPIO 제어 (PWM) 및 I2C 제어

라즈베리파이(Raspberry Pi)는 저비용, 고성능 싱글보드 컴퓨터로, GPIO(General Purpose Input/Output) 핀을 통해 다양한 전자 장치를 제어할 수 있습니다. PWM(Pulse Width Modulation)은 디지털 신호를 사용하여 아날로그 결과를 시뮬레이션하는 기술로, 모터 속도 제어나 LED 밝기 조절 등에 사용됩니다. I2C(Inter-Integrated Circuit)는 여러 장치 간의 통신을 위한 직렬 통신 프로토콜로, 센서나 디스플레이와 같은 주변 장치와의 통신에 사용됩니다.

라즈베리파이의 GPIO를 통한 PWM 제어는 하드웨어 PWM과 소프트웨어 PWM 두 가지 방식으로 가능합니다. 하드웨어 PWM은 더 정확하고 CPU 부하가 적지만 사용 가능한 핀이 제한적입니다. 소프트웨어 PWM은 더 유연하게 사용할 수 있지만 CPU 부하가 높습니다. I2C 통신은 SDA(Serial Data)와 SCL(Serial Clock) 두 개의 선을 사용하여 여러 장치와 통신할 수 있어, 복잡한 시스템 구성에 유용합니다.

2.1.2. YOLO 모델

YOLO(You Only Look Once)는 실시간 객체 탐지를 위한 최첨단 딥러닝 알고리즘입니다.[1] 이 알고리즘은 이미지를 한 번만 보고 여러 객체를 동시에 탐지하고 분류할 수 있어, 매우 빠른 처리 속도를 자랑합니다. YOLO는 이미지를 그리드로 나누고, 각 그리드 셀에서 객체의 존재 확률과 경계 상자를 예측합니다.

YOLO의 주요 특징은 다음과 같습니다:

1. 단일 신경망을 사용하여 경계 상자 예측과 클래스 확률을 동시에 계산
2. 전체 이미지의 전역적 특징을 고려하여 예측 수행
3. 테스트 시 실시간 처리 가능 (45 FPS 이상)
4. 다양한 버전 (YOLOv3, YOLOv5, YOLOv8 등)을 통해 지속적인 성능 개선

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779-788, 2016.

YOLO 모델은 본 프로젝트의 실시간 객체 탐지 기능 구현에 핵심적인 역할을 합니다. 빠른 처리 속도와 높은 정확도를 바탕으로, 카메라로 입력되는 영상에서 사람, 동물, 차량 등 다양한 객체를 실시간으로 탐지하고 분류할 수 있습니다. 이는 보안 시스템의 즉각적인 위협 감지 및 대응 능력을 크게 향상시킵니다.

2.1.3. DeepOCSORT 알고리즘

DeepOCSORT(Deep Online and Continuous Sort)는 다중 객체 추적(Multiple Object Tracking, MOT)을 위한 알고리즘으로, 기존의 SORT(Simple Online and Realtime Tracking) 알고리즘을 개선한 버전입니다.[2] 이 알고리즘은 딥러닝 기반의 특징 추출과 결합하여 더 강력한 추적 성능을 제공합니다.

DeepOCSORT의 주요 특징은 다음과 같습니다:

1. 딥러닝 기반 특징 추출기를 사용하여 객체의 외형 정보를 더 잘 포착
2. 칼만 필터를 사용하여 객체의 움직임을 예측
3. 헝가리안 알고리즘을 통한 효율적인 데이터 연관
4. ReID(Re-Identification) 기술을 통해 장기간 추적 성능 향상

DeepOCSORT 알고리즘은 프로젝트의 다중 객체 추적 기능을 구현하는 데 사용됩니다. 복잡한 환경에서도 안정적인 추적 성능을 제공하여, 여러 객체가 동시에 존재하는 상황에서도 각 객체의 이동을 정확하게 추적할 수 있습니다. 이는 지속적인 위협 평가와 정확한 조준 시스템 운용에 필수적입니다.

2.1.4. Flask

Flask는 Python으로 작성된 마이크로 웹 프레임워크입니다.[3] '마이크로'라는 용어는 Flask가 핵심 기능만을 제공하고, 필요에 따라 확장할 수 있는 유연한 구조를 가지고 있다는 것을 의미합니다. Flask는 간단한 API 서버부터 복잡한 웹 애플리케이션까지 다양한 규모의 프로젝트에 사용될 수 있습니다.

---

[2] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Weng, Y. Yuan, and P. Luo, "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," arXiv preprint arXiv:2110.06864, 2021.

[3] M. Grinberg, "Flask Web Development: Developing Web Applications with Python," O'Reilly Media, 2018.

Flask의 주요 특징은 다음과 같습니다:

1. 경량화: 최소한의 설정으로 빠르게 개발 시작 가능
2. 유연성: 다양한 확장 기능을 통해 필요한 기능만 추가 가능
3. Jinja2 템플릿 엔진: 동적 HTML 페이지 생성 용이
4. WSGI 호환: 다양한 웹 서버와 호환 가능
5. 내장 개발 서버 및 디버거 제공

Flask는 본 프로젝트의 하드웨어 서버 구현에 사용됩니다. 경량화되고 유연한 특성 덕분에 라즈베리파이의 제한된 리소스 환경에서도 효율적으로 동작하며, RESTful API를 쉽게 구현할 수 있어 클라이언트 애플리케이션과의 원활한 통신을 가능하게 합니다.[4] 또한, 필요에 따라 기능을 확장할 수 있어 프로젝트의 요구사항 변화에 유연하게 대응할 수 있습니다.

2.2. 통신 방식 개요

2.2.1. TCP/IP 소켓 통신

TCP/IP 소켓 통신은 네트워크 상에서 두 노드 간의 양방향 통신 채널을 구축하는 방법입니다.[5] 이는 전송 제어 프로토콜(TCP)과 인터넷 프로토콜(IP)을 기반으로 합니다.

작동 원리

1. 서버는 특정 포트에서 연결을 수신 대기합니다.
2. 클라이언트가 서버의 IP 주소와 포트로 연결을 요청합니다.
3. 연결이 수립되면, 양측은 데이터를 주고받을 수 있는 소켓을 얻게 됩니다.
4. 데이터는 바이트 스트림 형태로 전송됩니다.

특징

---

[4] V. Mazzia, A. Khaliq, F. Salvetti, and M. Chiaberge, "Real-Time Apple Detection System Using Embedded Systems With Hardware Accelerators: An Edge AI Application," IEEE Access, Vol. 8, pp. 9102-9114, 2020.

[5] W. R. Stevens, B. Fenner, and A. M. Rudoff, "UNIX Network Programming: The Sockets Networking API," Vol. 1, 3rd Ed., Addison-Wesley Professional, 2003.

- 연결 지향적: 데이터 전송 전 연결이 먼저 수립됩니다[6].
- 신뢰성 있는 데이터 전송: 패킷 손실, 중복, 순서 오류 등을 처리합니다.
- 전이중(Full-duplex) 통신: 동시에 양방향 데이터 전송이 가능합니다.
- 낮은 수준의 제어: 개발자가 프로토콜 세부사항을 직접 관리할 수 있습니다.

프로젝트 내 사용 사례

라즈베리파이(alphatronRPI.py)와 하드웨어 서버(main.py) 간의 통신에 사용됩니다. 이를 통해 라즈베리파이에서 캡처한 영상 데이터를 하드웨어 서버로 실시간 전송합니다.

2.2.2. WebSocket 통신

WebSocket은 단일 TCP 연결을 통해 전이중 통신 채널을 제공하는 컴퓨터 통신 프로토콜입니다.[7]

작동 원리

1. 클라이언트가 HTTP를 통해 WebSocket 연결을 요청합니다(WebSocket 핸드셰이크).
2. 서버가 이 요청을 수락하면, HTTP 연결이 WebSocket 연결로 업그레이드됩니다.
3. 이후 양방향 통신이 이루어집니다.

특징

- 실시간 양방향 통신: 서버와 클라이언트가 언제든 메시지를 주고받을 수 있습니다.
- 낮은 오버헤드: 한 번 연결이 수립되면 최소한의 추가 데이터로 통신이 가능합니다.
- 웹 호환성: 대부분의 브라우저에서 지원되며, 방화벽 친화적입니다.
- 다양한 데이터 형식: 텍스트와 이진 데이터를 모두 지원합니다.

---

[6] A. S. Tanenbaum and D. J. Wetherall, "Computer Networks," 5th Ed., Prentice Hall, 2011.

[7] V. Wang, F. Salim, and P. Moskovits, "The Definitive Guide to HTML5 WebSocket," Apress, 2013.

프로젝트 내 사용 사례

1. 하드웨어 서버(main.py)와 WebSocket 서버(WebSocketHandler.java) 간의 통신
2. WebSocket 서버와 Flutter 앱 간의 실시간 비디오 스트리밍 데이터 전송

2.2.3. RESTful API를 통한 HTTP 통신

REST(Representational State Transfer)는 웹 서비스를 위한 소프트웨어 아키텍처 스타일입니다. RESTful API는 이 원칙을 따르는 웹 API를 의미합니다.

작동 원리

1. 클라이언트가 특정 리소스에 대한 요청을 HTTP 메서드(GET, POST, PUT, DELETE 등)를 사용하여 보냅니다.
2. 서버는 요청을 처리하고 적절한 응답을 반환합니다.
3. 각 요청은 독립적이며, 서버는 클라이언트의 상태를 저장하지 않습니다(Stateless).

특징

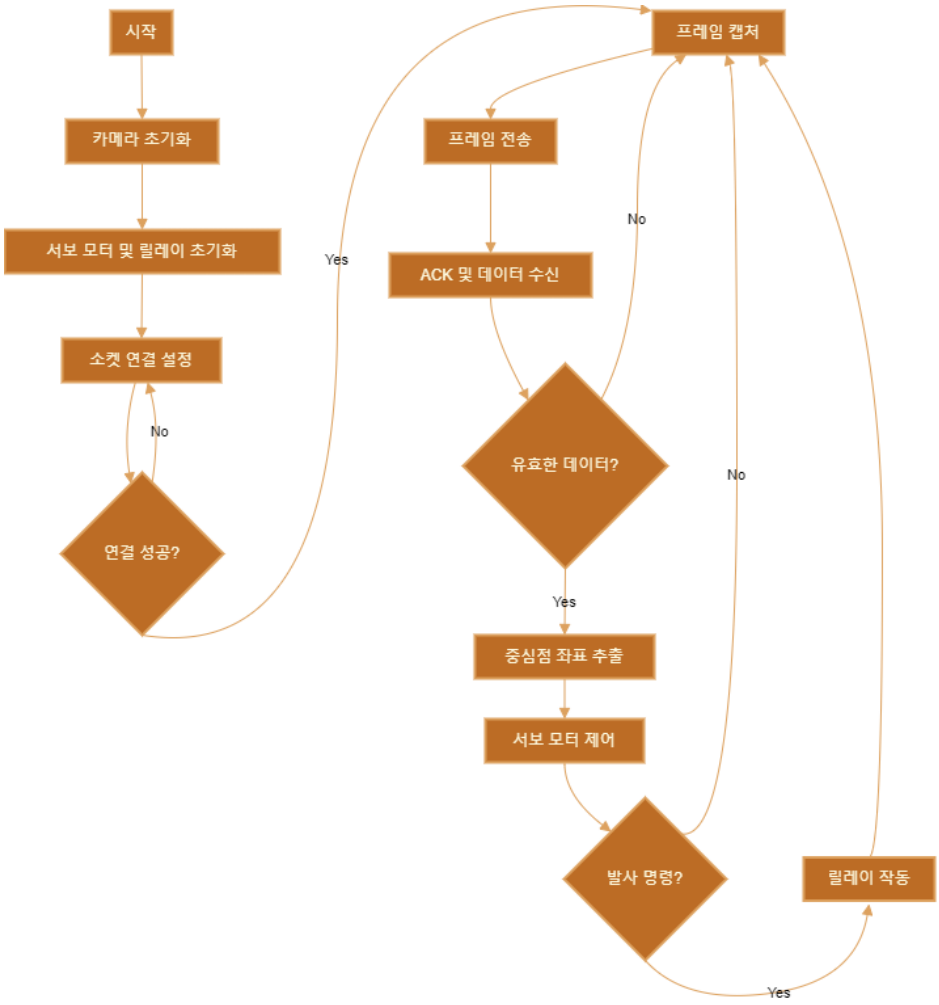
- 자원 기반 구조: 모든 것을 '자원'으로 표현하고, 각 자원은 고유한 URI를 가집니다.
- Stateless: 각 요청은 독립적이며, 서버는 클라이언트의 상태를 저장하지 않습니다.
- 캐시 가능: 응답을 캐시할 수 있어 성능을 향상시킬 수 있습니다.
- 균일한 인터페이스: 일관된 방식으로 자원을 조작합니다.

프로젝트 내 사용 사례

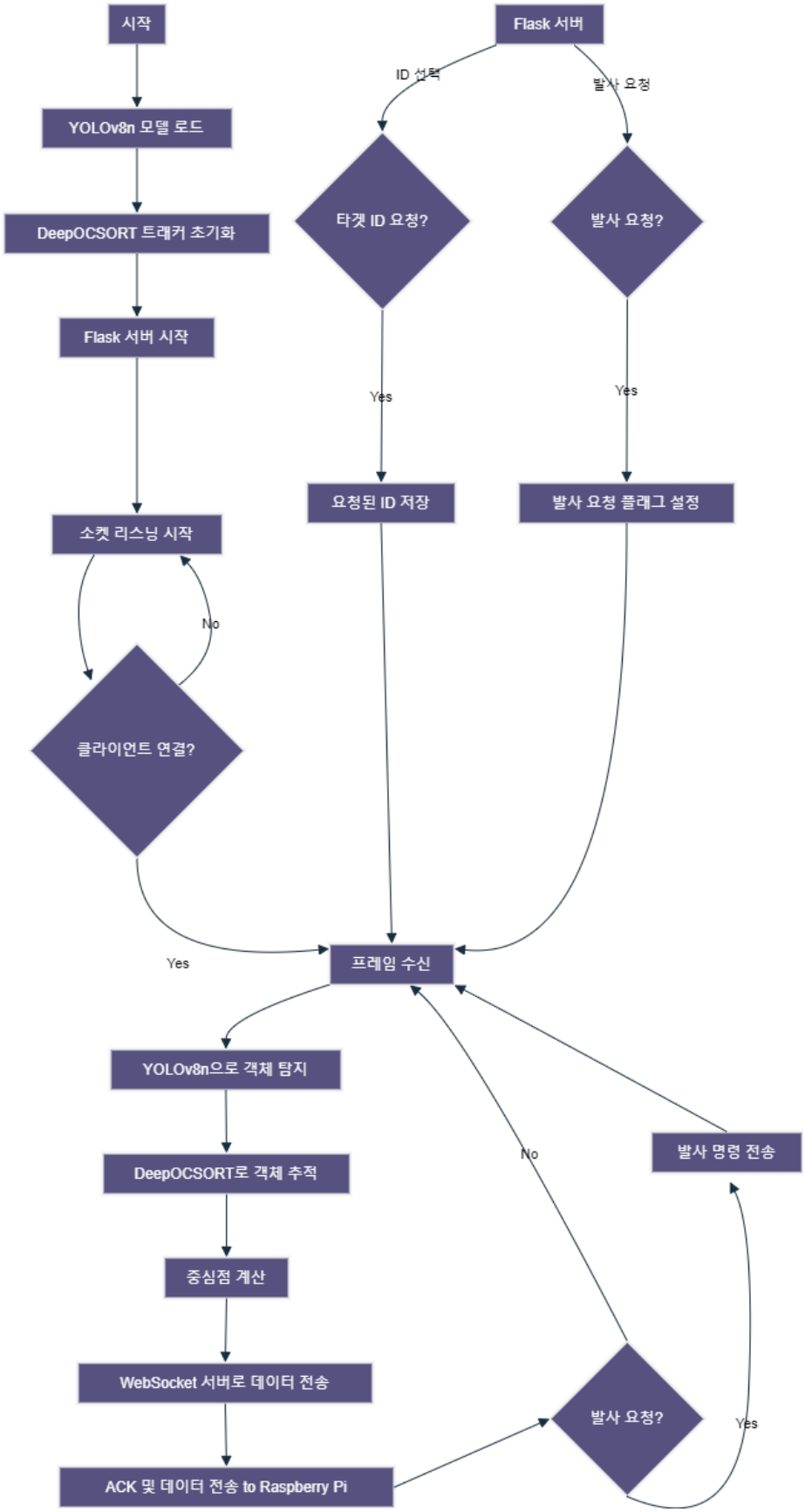
1. Flutter 앱과 WebSocket 서버 간 통신:
   - ■ 비디오 리스트 조회
   - ■ 비디오 다운로드 요청
2. Flutter 앱과 하드웨어 서버 간 통신:
   - ■ 타겟 ID 전송

2.3. 시스템 흐름도

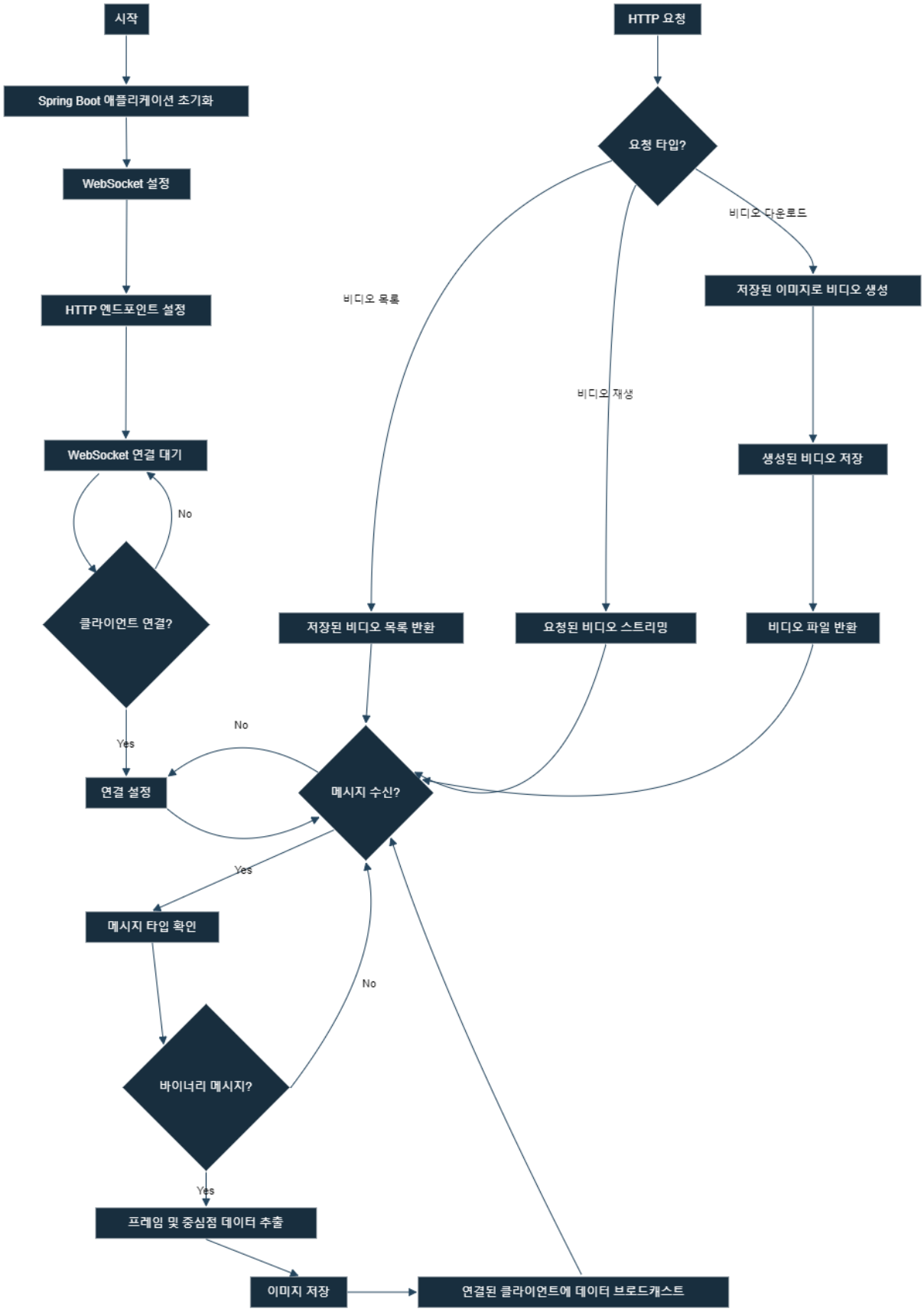
2.3.1. Hardware (Raspberry Pi)



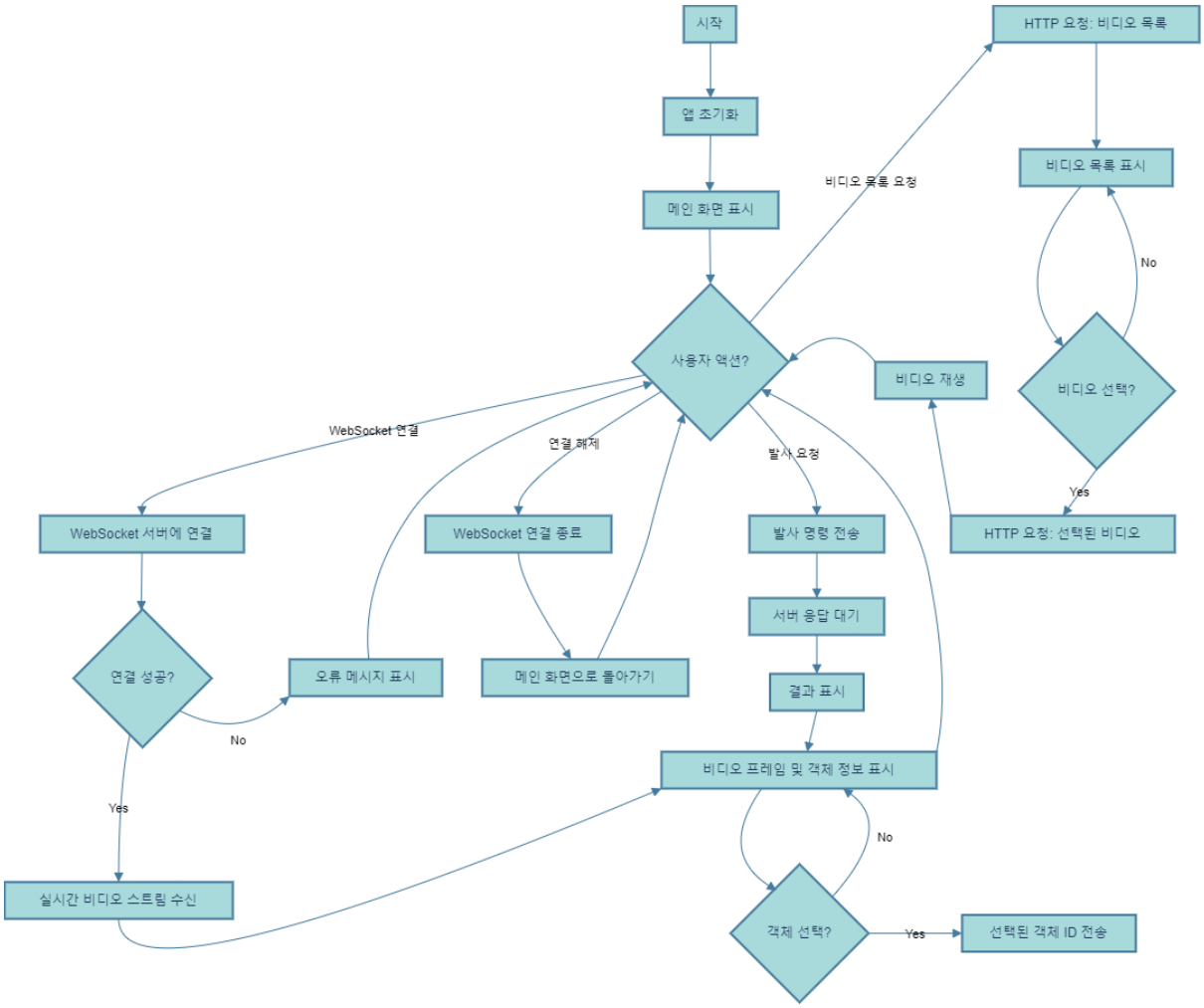
2.3.2. Hardware Server



2.3.3. Web Socket Server



2.3.4. Flutter Application



2.4. 개발 환경

- **하드웨어 (H/W):**
  - 플랫폼: 라즈베리파이
  - 프로그래밍 언어: Python
  - 주요 용도: 센서 데이터 수집, 하드웨어 제어
- **하드웨어 서버:**
  - 프로그래밍 언어: Python
  - 프레임워크: Flask
  - 주요 용도: 하드웨어와의 통신, 데이터 처리, 웹 서버 기능 제공

- **웹소켓 서버:**
  - 프레임워크: Java Spring Boot
  - 추가 기능: RESTful API 제공
  - 주요 용도: 클라이언트와의 실시간 양방향 통신, 데이터 관리
- **사용자 애플리케이션:**
  - 개발 플랫폼 모바일 애플리케이션 개발 (iOS, Android)

3. 과제 설계 상세

3.1. Hardware

본 프로젝트에서는 라즈베리파이 4B를 중심으로 하는 하드웨어 시스템을 구축할 예정입니다. 이 시스템은 실시간 영상 캡처, 객체 탐지 및 추적, 자동 조준 기능을 갖추게 될 것입니다.

먼저, 라즈베리파이에 연결된 카메라 모듈을 통해 실시간 영상을 캡처할 계획입니다. 이를 위해 picamera2 라이브러리를 활용하여 카메라를 제어하고, 적절한 해상도와 프레임 레이트를 설정할 것입니다. 카메라 모듈의 정상 작동 여부는 연결 상태, 캡처 속도, 영상 품질 등을 종합적으로 테스트하여 확인할 예정입니다.

카메라의 방향 제어를 위해 서보모터를 사용할 계획입니다. GPIO 핀을 통해 서보모터를 제어하여 카메라의 팬/틸트 움직임을 구현할 것입니다. 서보모터의 성능은 각도 제어의 정확성, 움직임의 부드러움, 반응 속도 등을 측정하여 검증할 예정입니다.

객체 탐지 및 추적 알고리즘이 완성되면, 이를 하드웨어 제어 시스템과 통합하여 탐지된 객체를 자동으로 조준할 수 있는 시스템을 구축할 것입니다. 카메라로 캡처한 영상을 실시간으로 처리하고, 탐지된 객체의 위치 정보를 바탕으로 서보모터를 제어하여 카메라의 방향을 자동으로 조정할 계획입니다.

탐지된 객체의 정보는 프레임에 시각적으로 표시하고, 이 정보와 함께 처리된 프레임을 웹소켓 서버로 전송할 예정입니다. 전송 데이터의 구조를 효율적으로 설계하여 실시간 통신이 원활히 이루어지도록 할 것입니다.

또한, Flutter 애플리케이션으로부터 발사 명령을 수신하고 처리할 수 있는 시스템을 구축할 계획입니다. GPIO를 통한 릴레이 제어 방식을 사용하여 발사 메커니즘을 작동시킬

예정이며, 안전을 고려한 설계를 진행할 것입니다. 발사 명령의 정확한 실행과 시스템의 안전성을 확보하기 위한 다양한 테스트를 수행할 예정입니다.

모든 하드웨어 구성 요소와 소프트웨어 모듈을 통합한 후에는 종합적인 시스템 테스트를 진행할 계획입니다. 이 테스트에서는 실시간 영상 처리 성능, 객체 탐지 및 추적의 정확도, 자동 조준 시스템의 반응 속도와 정확성, 발사 메커니즘의 안정성 등을 검증할 것입니다. 또한, 장시간 운영 테스트와 다양한 네트워크 환경에서의 성능 평가를 통해 시스템의 안정성과 신뢰성을 확보할 예정입니다.

3.2. 객체 탐지 및 추적

본 프로젝트에서는 실시간 객체 탐지 및 추적 시스템을 구현할 예정입니다. 이 시스템은 라즈베리파이로부터 전송되는 실시간 영상 프레임을 처리하여 사람 객체를 탐지하고 추적하는 기능을 수행할 것입니다.

먼저, 라즈베리파이로부터 소켓 통신을 통해 실시간으로 프레임 데이터를 수신할 계획입니다. 수신된 데이터는 즉시 디코딩되어 처리 가능한 이미지 형태로 변환될 것입니다. 이 과정에서 데이터 수신의 안정성과 디코딩의 정확성을 지속적으로 모니터링하여 시스템의 신뢰성을 확보할 예정입니다.

디코딩된 프레임 데이터는 YOLO(You Only Look Once) 알고리즘을 사용하여 객체 탐지 과정을 거칠 것입니다. 특히, 우리 시스템에서는 사람 객체에 초점을 맞추어 탐지를 수행할 계획입니다. YOLO 모델의 성능을 최적화하여 실시간 처리에 적합하도록 구성하고, 사람 객체 탐지의 정확도를 높이기 위한 다양한 방법을 적용할 예정입니다.

탐지된 객체는 DeepOCSORT(Deep Online and Continuous Sort) 알고리즘을 통해 추적될 것입니다. 이 알고리즘은 연속된 프레임에서 동일한 객체를 식별하고 추적하는 데 사용될 예정입니다. 각 추적된 객체에는 고유한 ID가 할당되며, 객체의 중심 좌표(CX, CY)가 계산될 것입니다. 이 정보들은 추후 처리 및 시각화에 활용될 예정입니다.

각 프레임 처리가 완료될 때마다, 시스템은 처리 결과를 라즈베리파이에 ACK(Acknowledgement) 메시지로 전송할 계획입니다. 기본적으로 이 ACK 메시지는 'None,None,None,0' 형태로 전송되며, 특정 객체가 선택되었을 때 해당 객체의 정보로 업데이트될 것입니다.

처리된 프레임에는 탐지 및 추적된 객체의 테두리와 정보(ID, CX, CY)가 시각적으로 표시될 예정입니다. 이렇게 처리된 프레임과 객체 정보는 WebSocket 통신을 통해 Spring

서버로 전송될 것입니다. 이 과정에서 데이터 전송의 효율성과 안정성을 확보하기 위한 방안을 마련할 계획입니다.

Flutter 애플리케이션에서 특정 ID 값을 선택하여 POST 요청을 보내면, 우리 시스템은 이를 처리하여 해당 ID를 가진 객체의 정보를 추적하고 업데이트할 것입니다. 이 정보는 다시 라즈베리파이로 전송되어 카메라의 방향 조정 등에 활용될 예정입니다.

또한, Flutter 애플리케이션에서 발사 요청이 전송되면, 시스템은 현재 추적 중인 객체의 정보와 함께 발사 여부를 포함한 데이터를 라즈베리파이로 전송할 계획입니다. 단, 추적 중인 객체가 화면에서 사라졌을 경우에는 발사 요청이 전송되더라도 이를 무시하는 안전 메커니즘을 구현할 예정입니다.

이러한 설계를 통해 우리는 실시간으로 작동하는 정확하고 안정적인 객체 탐지 및 추적 시스템을 구축하고자 합니다. 시스템의 각 구성 요소는 지속적인 테스트와 최적화 과정을 거쳐 성능을 향상시킬 예정이며, 전체 시스템의 안정성과 신뢰성을 확보하기 위해 다양한 시나리오에서의 종합적인 테스트를 수행할 계획입니다.

3.3. 웹소켓 서버

본 프로젝트에서는 실시간 데이터 통신을 위한 웹소켓 서버를 구축할 예정입니다. 이 서버는 객체 탐지 및 추적 시스템과 사용자 애플리케이션 간의 효율적인 양방향 통신을 지원하게 될 것입니다.

웹소켓 서버의 구축은 Spring Framework를 기반으로 진행할 계획입니다. 서버는 다수의 클라이언트 연결을 동시에 처리할 수 있도록 설계될 것이며, 연결의 안정성과 성능을 최적화하기 위한 다양한 기술을 적용할 예정입니다.

서버 구축 후에는 연결 관리 기능의 안정성을 검증하기 위한 테스트를 수행할 계획입니다. 이 과정에서 클라이언트 세션이 정상적으로 설정되는지, 연결이 지속적으로 유지되는지를 확인할 것입니다. 또한, 다수의 클라이언트가 동시에 연결을 시도하는 상황에서도 서버가 안정적으로 작동하는지 검증할 예정입니다.

데이터 전송 기능의 신뢰성을 테스트하기 위해 더미 데이터를 사용한 전송 테스트를 실시할 계획입니다. 이 테스트에서는 다양한 크기와 형태의 데이터를 전송하여 서버의 처리 능력과 전송 속도, 데이터 무결성을 확인할 것입니다. 특히, 실제 운용 환경과 유사한 조건에서의 성능을 평가하기 위해 대용량 비디오 스트림 데이터 전송 테스트도 포함할 예정입니다.

연결 관리의 완전성을 보장하기 위해, 클라이언트의 연결 종료 시 적절한 알림 메커니즘을 구현할 계획입니다. 서버는 클라이언트의 연결 해제를 감지하고, 이를 로그로 기록하며, 필요한 경우 다른 연결된 클라이언트들에게 알림을 전송할 수 있도록 설계될 것입니다. 이러한 기능의 정확성은 다양한 네트워크 상황을 시뮬레이션하여 검증할 예정입니다.

프로젝트의 중요한 요구사항 중 하나인 비디오 데이터 저장에 대해서는 심도 있는 고민 끝에 MySQL의 LOB(Large Object) 형식을 사용하기로 결정했습니다. 이 결정은 데이터의 일관성, 트랜잭션 지원, 그리고 기존 관계형 데이터베이스 시스템과의 통합 용이성을 고려한 것입니다. LOB 형식을 사용함으로써 대용량 비디오 파일을 효율적으로 저장하고 관리할 수 있을 것으로 예상됩니다.

비디오 데이터 저장 시스템의 성능을 최적화하기 위해, 인덱싱 전략, 청크 단위 저장 및 검색 방법, 그리고 캐싱 메커니즘 등을 구현할 계획입니다. 또한, 저장된 비디오 데이터의 빠른 검색과 스트리밍을 위한 API를 개발하여, 클라이언트 애플리케이션에서 원활하게 비디오를 재생할 수 있도록 지원할 예정입니다.

이러한 설계를 통해 우리는 안정적이고 효율적인 웹소켓 서버를 구축하고자 합니다. 서버의 각 구성 요소는 단위 테스트, 통합 테스트, 그리고 부하 테스트를 거쳐 그 신뢰성을 검증할 것이며, 실제 운용 환경에서의 성능을 보장하기 위해 지속적인 모니터링과 최적화 과정을 거칠 예정입니다. 이를 통해 실시간 데이터 통신과 비디오 데이터 관리를 효과적으로 수행할 수 있는 강력한 백엔드 시스템을 구축할 수 있을 것으로 기대합니다.

3.4. 사용자 애플리케이션

본 프로젝트에서는 Flutter 프레임워크를 사용하여 안드로이드와 iOS 플랫폼에서 동작하는 모바일 애플리케이션을 개발할 예정입니다. 이 애플리케이션의 핵심 기능은 실시간 영상 표시, 객체 탐지 및 선택, 그리고 선택된 객체 정보 전송입니다. 실시간 영상 표시 기능은 웹소켓 서버로부터 H.264로 인코딩된 영상을 수신하여 지연 없이 화면에 렌더링합니다. H.264 인코딩을 활용함으로써 네트워크 대역폭 사용을 최소화하면서도 고품질의 영상을 유지할 수 있을 것으로 기대됩니다. 애플리케이션에서는 하드웨어 가속을 이용한 효율적인 디코딩 및 렌더링을 구현할 계획입니다.

수신된 프레임에서 탐지된 객체는 경계 상자와 고유 ID로 표시되며, 사용자가 특정 객체를 선택할 수 있는 인터페이스를 제공합니다. 사용자가 선택한 객체의 ID 정보는 웹소켓을 통해 하드웨어 서버로 실시간 전송됩니다. 성능 최적화를 위해 비동기 프로그래밍

기법을 활용하며, 특히 H.264 디코딩과 영상 처리 부분에서 효율적인 리소스 관리를 구

현할 것입니다. 또한, 애플리케이션의 안정성을 위한 예외 처리와 오류 복구 메커니즘을 구현할 계획입니다.

애플리케이션의 품질 보장을 위해 다양한 테스트를 수행할 예정입니다. 이는 단위 테스트, 통합 테스트, 사용자 인터페이스 테스트, 그리고 다양한 네트워크 환경에서의 성능 테스트를 포함합니다. 특히 H.264 디코딩 성능과 관련된 테스트를 통해 다양한 디바이스에서의 안정성과 성능을 검증할 계획입니다. 보안 측면에서는 데이터 암호화와 안전한 인증 메커니즘을 구현하여 시스템의 안전성을 보장할 것입니다.

4. 과제 수행 내용

4.1. 개발 일정 및 역할

이름	역할
김대영	- 하드웨어 시스템 설계 및 구현 - 라즈베리파이와 컴퓨터의 TCP/IP 소켓 연결 구현 - 보고서 작성
박민재	- 객체 탐지 및 추적 코드 작성 - 하드웨어 서버 구현
조영진	- 웹소켓 서버 구현 - 사용자 Application 구현

표 1. 구성원별 역할

5월		6월				7월				8월				9월				10월			
3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
착수 보고서 작성, 지도 확인서 제출																					
		사용 기술 조사																			
					모델, 서버, 앱 개발																
									시스템 테스트 및 중간보고서 작성												
													모델 정확도 테스트								
													라즈베리 파이 각 모듈별 제어 코드 작성 및 테스트								
																로봇 제작					
																			최종 테스트		
																				최종 보고서 작성 및 발표 준비	

표 2. 무인 경계 로봇 개발 일정

4.2. 개발 환경 구축

본 프로젝트의 개발 환경 구축은 하드웨어와 소프트웨어 양면에서 세심한 고려가 필요했습니다.

하드웨어 플랫폼으로는 Raspberry Pi 4B 보드를 선택하였습니다. 이 보드는 높은 성능과 다양한 인터페이스를 제공하여 복잡한 시스템 구현에 적합했습니다. 운영체제로는 Raspberry Pi OS 64비트 버전을 채택하였는데, 이는 데비안 북웜(Bookworm) 기반의 시스템입니다. 초기에는 32비트 버전으로 개발 환경을 구축하려 했으나, 일부 필요한 라이브러리와 도구들이 32비트 시스템을 지원하지 않아 어려움을 겪었습니다. 이에 64비트 버전으로 전환하여 이러한 문제를 해결하였습니다.

서버 측 개발에는 Spring Boot 프레임워크를 활용하였습니다. 처음에는 실시간 통신을 위해 WebRTC 기술의 도입을 고려했으나, 구현의 복잡성과 프로젝트의 요구사항을 고려하여 최종적으로 웹소켓 기술을 채택하였습니다. 이러한 선택은 시스템의 실시간성을 유지하면서도 개발 및 유지보수의 용이성을 확보할 수 있게 해주었습니다.

전반적인 개발은 Windows 11 환경에서 진행되었으며, 크로스 플랫폼 개발을 위해 Flutter 프레임워크를 사용하여 모바일 애플리케이션을 구현하였습니다.

4.3. Hardware

본 프로젝트의 하드웨어 구현에 있어 Raspberry Pi 4B를 중심으로 다양한 컴포넌트들을 통합하는 과정에서 여러 기술적 도전과 해결 과정이 있었습니다. 먼저, 실시간 영상 입력을 위해 picamera2 모듈을 사용하여 카메라 모듈을 테스트하였습니다. 이를 통해 안정적인 실시간 영상 스트리밍이 가능함을 확인하였습니다.

서보 모터 제어 구현에 있어서는 초기에 GPIO18 핀을 사용하여 PWM 제어 방식을 적용하였습니다. 이 과정에서 모터 제작사의 공식 문서와는 달리 PWM duty cycle이 2에서 11 사이의 범위에서 작동함을 발견하였고, 이에 맞춰 제어 로직을 조정하였습니다. 추가적인 서보 모터 제어를 위해 GPIO13 핀의 사용을 시도하였으나, 라즈베리 파이의 5V 전압 출력으로는 충분한 전력을 공급하기 어려웠습니다. 이 문제를 해결하기 위해 Servo Driver HAT을 도입하기로 결정하였습니다.

Servo Driver HAT 사용 시 외부 전력 공급이 필요하여, 제조사의 안내에 따라[8] 0옴 저항을 제거하고 6V 전압을 공급하였습니다. 그러나 이 설정으로는 서보 모터가 제대로 작동하지 않았습니다. 제조사의 공식 스펙을 재확인한 결과, 정격 전압이 6V에서 12V 사이임을 알게 되었고, 9V 배터리를 사용하여 전력을 공급함으로써 서보 모터의 정상 작동을 확인할 수 있었습니다.

무기 시스템 통합 과정에서는 기존의 클릭 스위치를 제거하고 GPIO17 핀과 릴레이 박스를 연결하여 소프트웨어 제어가 가능한 발사 메커니즘을 구현하였습니다. 이를 통해 프로그램의 명령에 따라 정확한 타이밍에 발사가 가능해졌습니다.

카메라 모듈의 정확한 화각 계산은 정밀한 조준 시스템 구현에 핵심적이었습니다. 제조사에서 제공한 120도의 대각선 화각을 바탕으로[9], 640x480 해상도에서의 정확한 가로 및 세로 화각을 계산하였습니다. 정밀한 삼각함수 계산을 통해 가로 시야각(HFOV)이 109.3도, 세로 시야각(VFOV)이 92.7도임을 도출하였습니다. 이 계산된 화각을 바탕으로 서보 모터의 각도 제어 알고리즘을 개발하였습니다. 서보 모터의 각도가 90도일 때 화면의 중앙을 가리키도록 설정하고, 객체의 중점 좌표에 따라 서보 모터의 각도를 동적으로 조절하여 정확한 조준이 가능하도록 구현하였습니다.

이러한 하드웨어 구성 및 최적화 과정을 통해, 본 프로젝트는 실시간 영상 처리, 정확한 객체 추적, 그리고 정밀한 조준 및 발사 기능을 갖춘 통합 시스템으로 완성되었습니다. 각 컴포넌트의 특성을 정확히 이해하고 최적화함으로써, 전체 시스템의 안정성과 성능을 크게 향상시킬 수 있었습니다.

4.4. 객체 탐지 및 추적

본 프로젝트에서 객체 탐지 및 추적 기능의 구현은 실시간 처리와 정확도 사이의 균형을 맞추는 것이 주요 과제였습니다. 초기에는 라즈베리 파이에서 직접 실행할 수 있는 경량화된 YOLO v5 모델을 고려하였습니다. 이 모델은 상대적으로 작은 크기에도 불구하고 준수한 정확도를 제공하여 매력적인 선택지였습니다. 그러나 실시간으로 프레임을 처리하면서 동시에 YOLO와 DeepSORT 알고리즘을 실행해야 하는 요구사항으로 인해 라즈

[8] Waveshare Electronics. (2024). Servo Driver HAT [Online]. Available: https://www.waveshare.com/wiki/Servo_Driver_HAT (downloaded 2024, Oct. 15)

[9] Raspberry Pi Foundation. (2024). Camera Module 3 [Online]. Available: https://www.raspberrypi.com/products/camera-module-3/ (downloaded 2024, Oct. 15)

베리 파이의 처리 능력에 상당한 부담이 되었습니다.

이러한 문제를 해결하기 위해 TensorFlow Lite를 사용한 모델 경량화도 고려되었습니다. 하지만 이 접근 방식은 정확도의 불가피한 저하를 동반했고, 총기 관련 시스템의 특성상 높은 정확도가 필수적이었기 때문에 적합하지 않았습니다.

따라서 우리는 실시간성을 유지하면서도 높은 정확도를 확보하기 위해 분산 처리 방식을 채택하였습니다. 라즈베리 파이에서 캡처한 프레임 데이터를 소켓 통신을 통해 별도의 하드웨어 서버(로컬 컴퓨터)로 전송하고, 이 서버에서 객체 탐지 및 추적 모델을 실행하는 방식을 구현하였습니다. 이 접근 방식은 여러 가지 이점을 제공했습니다.

첫째, 라즈베리 파이의 부하를 크게 줄일 수 있었습니다. 이로 인해 카메라 모듈과 서보 모터 제어 등 다른 중요한 기능에 더 많은 리소스를 할당할 수 있게 되었습니다. 둘째, 더 강력한 하드웨어에서 모델을 실행함으로써 처리 속도와 정확도를 동시에 개선할 수 있었습니다. 이는 실시간 추적의 품질을 크게 향상시켰습니다.

또한, 이 접근 방식은 시스템의 확장성과 유연성을 크게 높였습니다. 하드웨어 서버의 성능 향상만으로도 더 복잡하고 정확한 모델로의 업그레이드가 가능해졌습니다. 특히 Ultralytics에서 제공하는 모델을 활용함으로써, 인터넷 연결만 있다면 언제든지 최신 모델을 다운로드받아 사용할 수 있게 되었습니다. 이는 시스템의 지속적인 개선과 최신 기술 적용을 용이하게 만들었습니다.

결과적으로, 이러한 분산 처리 방식의 채택은 실시간 객체 탐지 및 추적의 성능을 크게 향상시켰습니다. 라즈베리 파이의 하드웨어 한계를 극복하고, 동시에 시스템의 정확도와 확장성을 확보할 수 있었습니다. 이는 본 프로젝트의 핵심 요구사항인 높은 정확도와 실시간 처리 능력을 모두 만족시키는 효과적인 해결책이 되었습니다.

4.5. 웹소켓 서버

본 프로젝트에서는 Spring Framework를 기반으로 하는 강력하고 효율적인 웹소켓 서버를 구현하였습니다. 서버의 핵심 구성요소와 주요 기능을 상세히 설명하겠습니다.

먼저, WebSocketConfig 클래스를 통해 웹소켓 설정을 구성하였습니다. 이 클래스에서는 "/python"과 "/flutter" 두 개의 엔드포인트를 등록하여 각각 파이썬 클라이언트와 Flutter 애플리케이션의 연결을 지원하도록 하였습니다. 성능 최적화를 위해 웹소켓의 버퍼 크기를 50MB로 확장하고, 세션 타임아웃을 10분으로 설정하여 장시간 연결에도 안정적인 통신이 가능하도록 하였습니다.

WebSocketHandler 클래스는 실제 웹소켓 통신의 핵심 로직을 구현하고 있습니다. 이 클래스에서는 클라이언트 연결 관리, 메시지 처리, 그리고 데이터 브로드캐스팅 기능을 제공합니다. 특히 주목할 만한 점은 바이너리 메시지 처리 로직으로, 중점 좌표 데이터와 이미지 프레임을 효과적으로 분리하여 처리합니다. 이를 통해 객체 탐지 결과를 실시간으로 전송할 수 있게 되었고, 동시에 이미지 데이터를 파일 시스템에 저장하는 기능도 구현하였습니다.

비디오 데이터 관리를 위해 VideoFile 엔티티를 설계하였습니다. 이 엔티티는 비디오 파일의 메타데이터와 실제 비디오 데이터를 저장합니다. JPA를 활용하여 VideoFileRepository를 구현함으로써 데이터베이스와의 효율적인 상호작용이 가능해졌습니다. VideoService 클래스에서는 비디오 파일의 저장, 조회, 전체 목록 조회 등의 비즈니스 로직을 제공하여 애플리케이션의 다른 부분에서 쉽게 비디오 데이터를 관리할 수 있도록 하였습니다.

WebSocketController 클래스는 HTTP 기반의 REST API를 제공합니다. 이 컨트롤러를 통해 비디오 파일 목록 조회, 개별 비디오 파일 다운로드, 새로운 비디오 파일 생성 등의 기능을 구현하였습니다. 특히 주목할 만한 점은 ffmpeg를 활용한 이미지 시퀀스를 비디오로 변환하는 기능입니다. 이 기능을 통해 객체 탐지 과정에서 저장된 이미지들을 하나의 연속된 비디오로 만들어 블랙박스 영상을 생성할 수 있게 되었습니다.

마지막으로, 실시간성이 크게 요구되지 않는 기능들에 대해서는 HTTP 프로토콜을 사용하여 구현하였습니다. 예를 들어, 발사 명령, 특정 객체에 대한 인식 요청, 비디오 저장 및 불러오기 등의 기능은 RESTful API를 통해 처리됩니다. 이러한 접근 방식은 웹소켓의 실시간 통신 부하를 줄이고, 전체 시스템의 안정성을 향상시키는 데 기여하였습니다.

이러한 종합적인 구현을 통해, 본 프로젝트는 실시간 객체 탐지 및 추적 데이터의 전송, 고품질 비디오 스트리밍, 효율적인 비디오 파일 관리 등 다양한 기능을 제공하는 안정적이고 확장 가능한 웹소켓 서버를 성공적으로 구축하였습니다. 이 서버는 복잡한 실시간 통신 요구사항을 충족시키면서도, 비실시간 작업의 안정성과 효율성을 보장하는 균형 잡힌 시스템으로 완성되었습니다.

4.6. 사용자 애플리케이션

본 프로젝트에서는 Flutter 프레임워크를 사용하여 크로스 플랫폼 모바일 애플리케이션을 개발하였습니다. 애플리케이션의 핵심 기능은 실시간 영상 스트리밍, 객체 탐지 결과 표시, 그리고 사용자 상호작용을 통한 객체 선택 및 정보 전송입니다.

실시간 영상 스트리밍을 구현하기 위해 WebSocket 기술을 활용하였습니다. WebSocket 클래스를 사용하여 서버와의 연결을 관리하고, 데이터 스트림을 처리하였습니다. 초기 계획과는 달리 H.264 디코딩을 직접 구현하지 않았는데, 이는 Flutter에서 H.264 데이터 스트림을 실시간으로 영상으로 변환할 수 있는 적절한 패키지를 찾지 못했기 때문입니다. 대신, 서버로부터 JPEG 형식의 이미지 프레임을 수신하여 화면에 표시하는 방식을 채택하였습니다.

VideoStreamModel 클래스에서는 WebSocket 연결 관리와 데이터 수신 로직을 구현하였습니다. 이 클래스는 서버로부터 받은 바이너리 데이터를 Uint8List 형태로 변환하여 스트림 형태로 제공합니다. VideoStreamController 클래스에서는 이 데이터를 처리하여 화면에 표시할 수 있는 형태로 변환합니다.

사용자 인터페이스는 VideoStream 위젯을 중심으로 구성되었습니다. 이 위젯은 실시간으로 수신되는 영상을 표시하고, 탐지된 객체에 대한 정보를 오버레이 형태로 제공합니다. 사용자는 화면에 표시된 객체를 선택할 수 있으며, 선택된 객체의 ID는 서버로 전송됩니다. 이를 위해 HTTP POST 요청을 사용하여 서버와 통신하는 로직을 구현하였습니다.

애플리케이션의 상태 관리와 비즈니스 로직 처리를 위해 GetX 라이브러리를 활용하였습니다. 이를 통해 반응형 프로그래밍 패턴을 적용하여 UI의 동적 업데이트와 상태 관리를 효율적으로 구현하였습니다.

또한, 녹화된 영상을 관리하고 재생하는 기능을 구현하였습니다. VideoListView 위젯에서는 서버에 저장된 영상 목록을 표시하고, 사용자가 선택한 영상을 재생할 수 있는 기능을 제공합니다. 영상 재생을 위해 VideoPlaybackView 위젯을 구현하였으며, 이 위젯은 서버로부터 영상 데이터를 스트리밍 방식으로 받아 재생합니다.

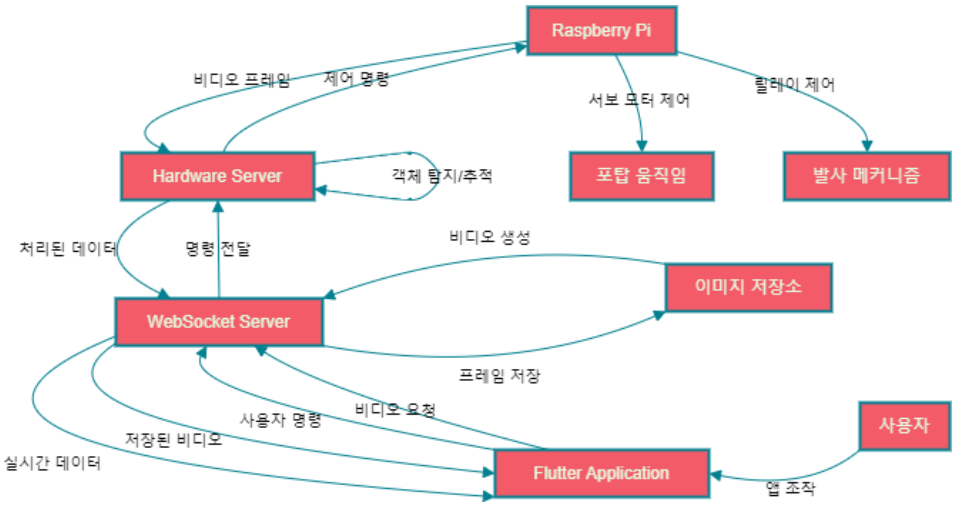
성능 최적화를 위해 비동기 프로그래밍 기법을 적극 활용하였습니다. Future와 Stream을 사용하여 네트워크 요청과 데이터 처리를 비동기적으로 수행함으로써, UI의 반응성을 유지하면서도 효율적인 리소스 관리를 달성하였습니다.

보안 측면에서는 HTTPS 프로토콜을 사용하여 서버와의 통신을 암호화하였습니다. 또한, 사용자 인증 기능을 구현하여 애플리케이션의 안전한 사용을 보장하였습니다.

전체적으로, 본 애플리케이션은 실시간 영상 스트리밍, 객체 탐지 결과 표시, 사용자 상호작용, 그리고 녹화된 영상 관리 기능을 성공적으로 구현하였습니다. Flutter의 크로스 플랫폼 특성을 활용하여 iOS와 Android 모두에서 동작하는 애플리케이션을 개발함으로써, 다양한 사용자 환경을 지원할 수 있게 되었습니다.

5. 과제 수행 결과

5.1. 전체 흐름도



5.2. 최종 테스트

5.2.1. Hardware Server : 탐지한 객체에 id를 부여하고, 추적하고 있는 화면

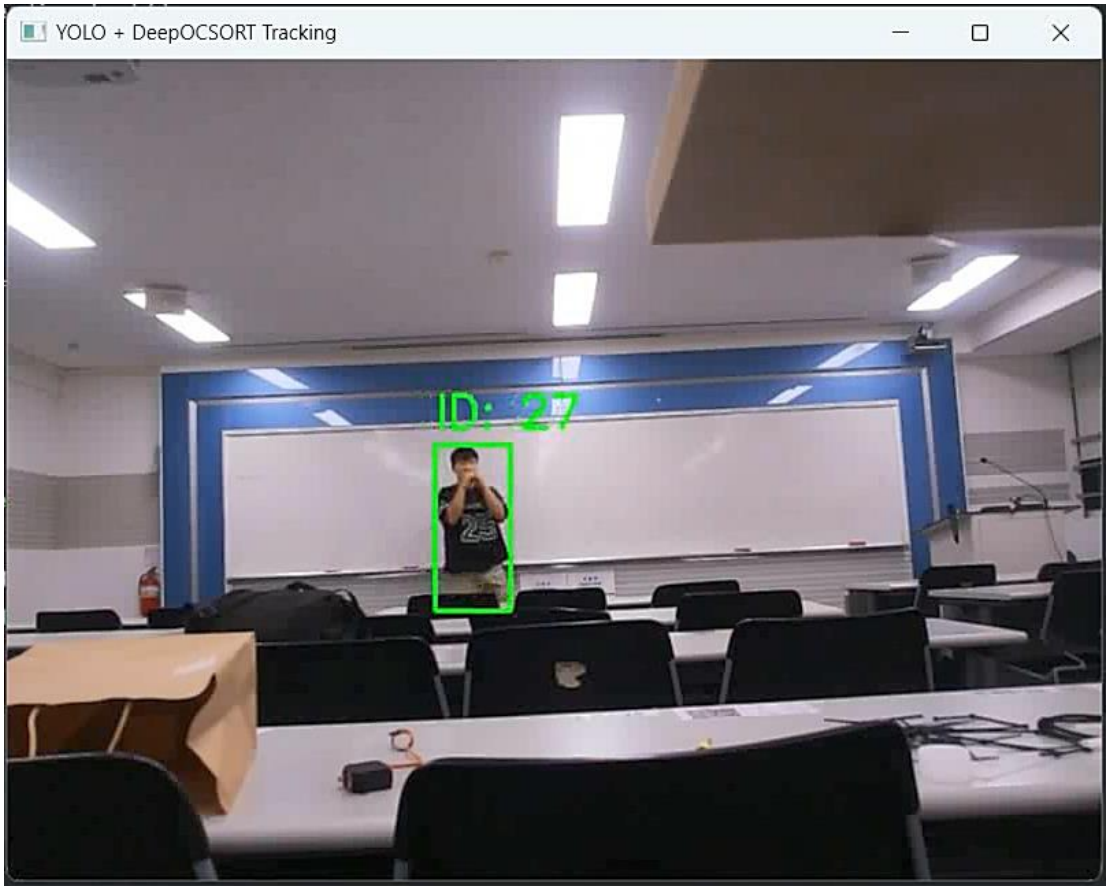


그림 1. 탐지한 객체에 id를 부여하고, 추적하고 있는 화면

5.2.2. Hardware Server : 하드웨어 서버 실행화면 전체

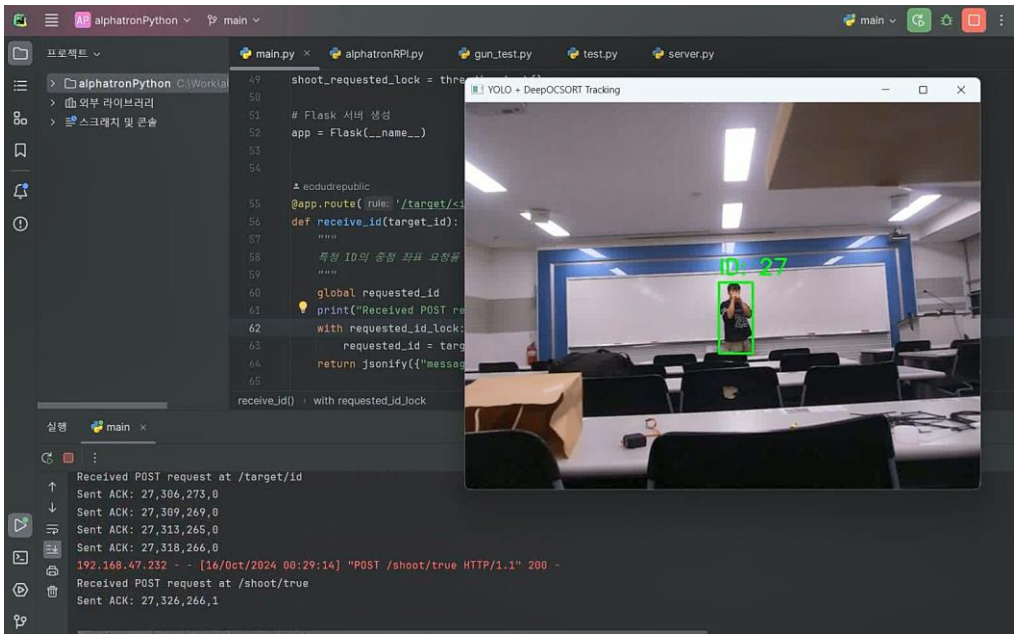


그림 2. 하드웨어 서버 실행화면 전체 (발사 명령 수신 포함)

5.2.3. Flutter Application : 애플리케이션 시작 화면

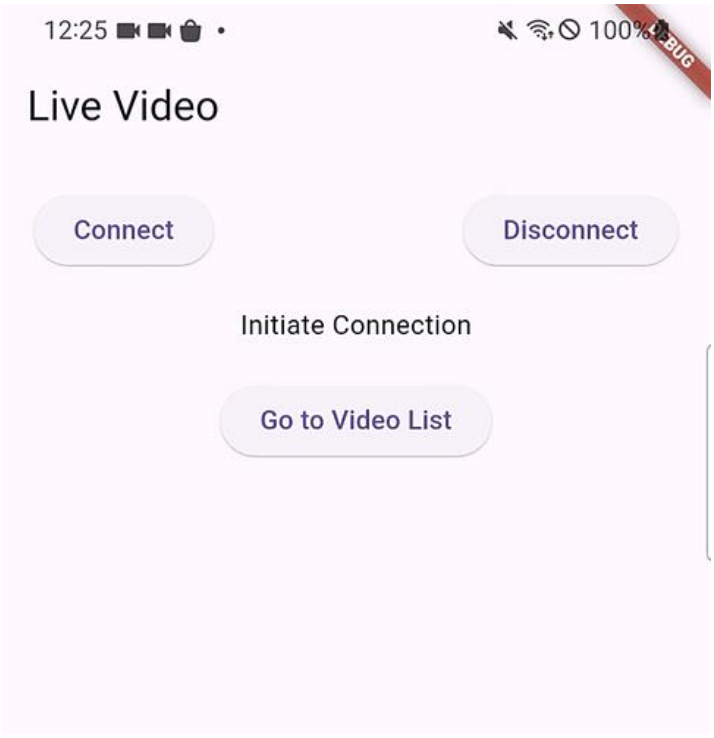


그림 3. 애플리케이션 시작 화면

→ Connect : 웹소켓 서버와 연결, Go to Video List : 녹화한 영상 리스트 확인

5.2.4. Flutter Application : 애플리케이션 실행 화면 – 1

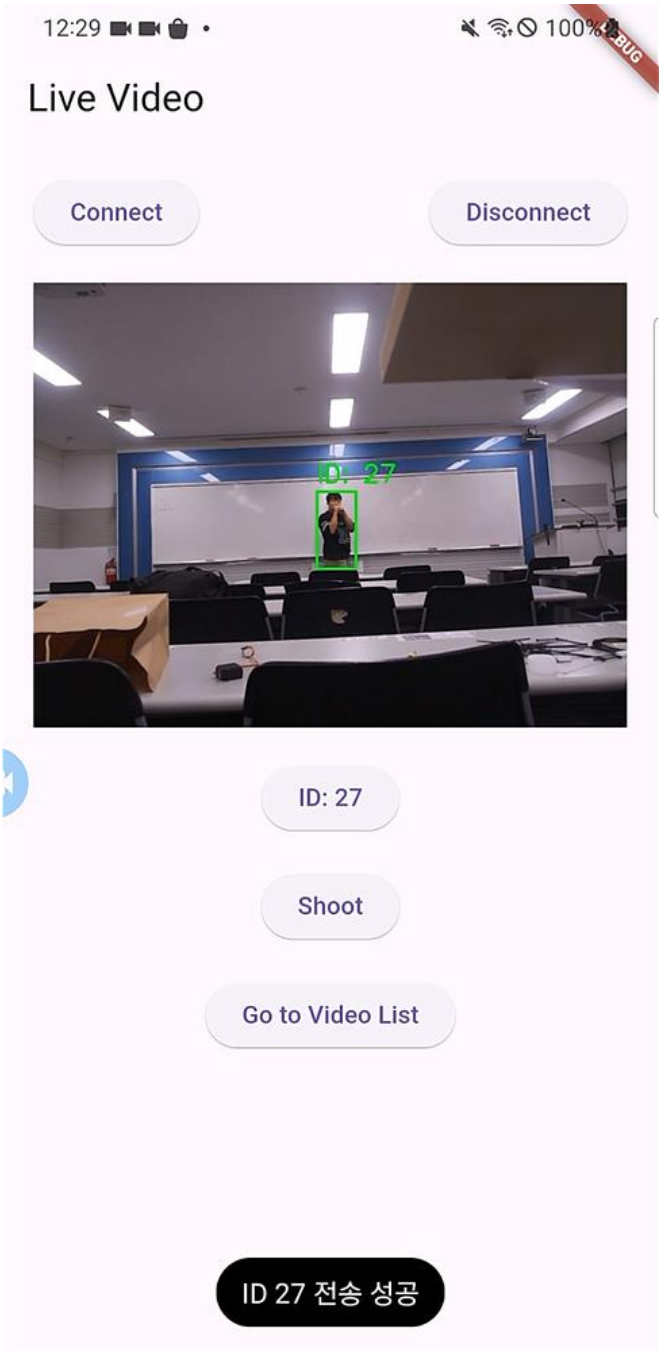


그림 4. 애플리케이션 실행 화면 – 1

1. Connect 버튼을 눌러 웹소캣 서버와 연결하여, 실시간 영상이 보이고 있는 화면

2. Hardware Server에서 탐지한 객체의 영역을 표시한 프레임을 받아와서 영상으로 표시

3. ID: n 버튼을 누르면 해당 ID의 객체 추적

4. Shoot 버튼을 누르면 추적중인 객체에 발사 (추적중인 객체가 없다면 무효)

5.2.5. Flutter Application : 애플리케이션 실행 화면 – 2 & 3

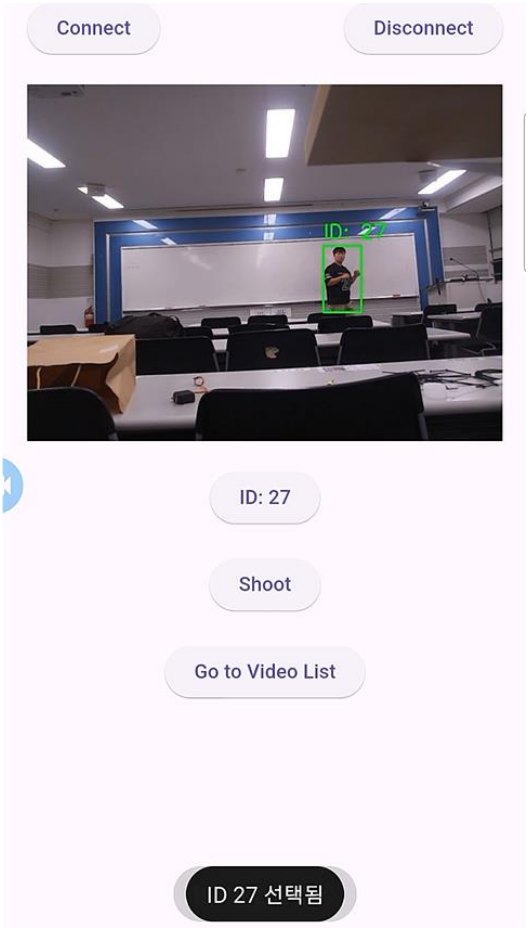


그림 6. Flutter Application : ID 27 선택 화면

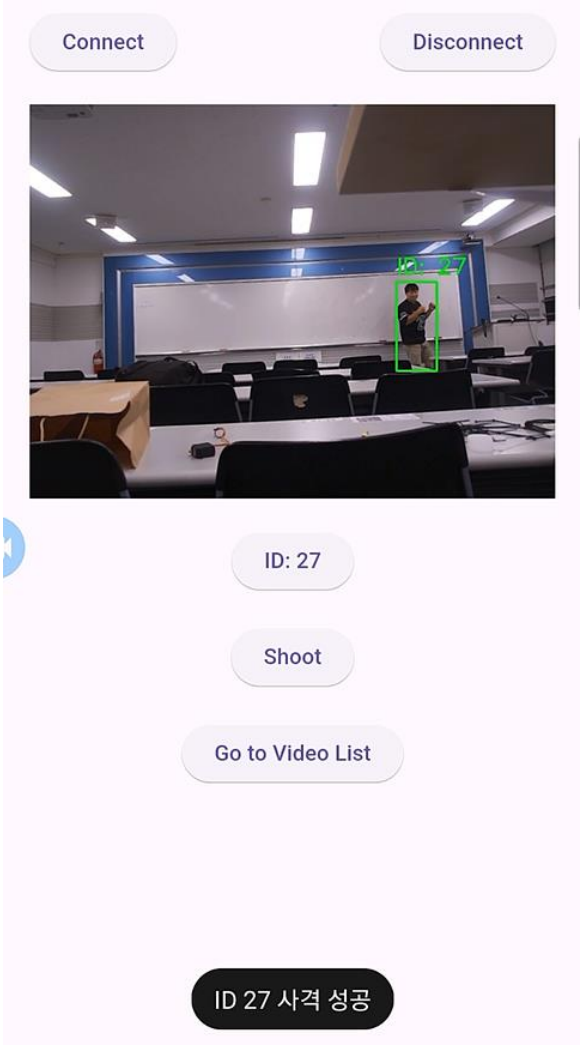


그림 5. Flutter Application : ID 27 사격 화면

5.2.6. 침입자 시점

그림 7. 침입자 시점 : 총구가 조준되고 있음

5.2.7. 로봇 시점 – 1 & 2

그림 8. 로봇 시점 - 1

그림 9. 로봇 시점 - 2

6. 결론 및 향후 연구 방향

6.1. 결론

6.1.1. 프로젝트 요약 및 주요 성과

본 시스템의 핵심 기능은 실시간 영상 처리를 통한 객체 탐지 및 추적, 원격 포탑 제어, 그리고 실시간 영상 스트리밍 및 녹화입니다. 이러한 기능들은 라즈베리 파이, 고성능 서버, 웹소켓 서버, 그리고 모바일 애플리케이션의 유기적인 통합을 통해 구현되었습니다.

특히, 다음과 같은 주요 성과를 달성하였습니다.

- YOLOv8n과 DeepOCSORT 알고리즘을 활용한 고효율 실시간 객체 탐지 및 추적 시스템 구현
- 서보 모터와 릴레이를 이용한 정밀한 원격 제어 포탑 시스템 개발
- 웹소켓 기술을 활용한 저지연 실시간 영상 스트리밍 및 녹화 기능 구현
- 라즈베리 파이, 고성능 서버, 웹소켓 서버, 플러터 기반 모바일 앱 등 다양한 기술의 성공적인 통합
- TCP/IP, 웹소켓, HTTP 등 다양한 통신 프로토콜을 활용한 효율적인 데이터 전송 시스템 구축

6.1.2. 기술적 의의

본 프로젝트는 최신 컴퓨터 비전 기술인 YOLOv8n과 DeepOCSORT를 실제 시스템에 적용하여 그 효용성을 입증하였습니다. 또한, 웹소켓 기술을 활용한 실시간 데이터 통신 구현을 통해 저지연 영상 전송 및 제어가 가능한 시스템을 개발하였습니다. 플러터를 이용한 크로스 플랫폼 모바일 애플리케이션 개발은 시스템의 접근성과 사용성을 크게 향상시켰습니다.

6.1.3. 응용 분야 및 잠재적 영향

본 시스템은 다양한 분야에서 활용될 수 있는 잠재력을 가지고 있습니다.

- 보안 및 감시 시스템: 기존 CCTV 시스템을 뛰어넘는 지능형 감시 시스템으로 활용 가능
- 군사적 응용: 경계 구역 감시 및 자동화된 방어 시스템으로의 확장 가능성

- 자동화된 모니터링: 산업 현장이나 재난 지역에서의 실시간 모니터링 및 대응 시스템으로 활용

이러한 응용 가능성은 보안, 국방, 산업 안전 등 다양한 분야에서 혁신을 가져올 수 있을 것으로 기대됩니다.

6.2. 향후 연구 방향

본 프로젝트의 성공적인 완료를 바탕으로, 우리는 시스템의 성능과 기능을 더욱 향상시키기 위한 다양한 향후 연구 방향을 제시합니다. 이러한 연구 방향은 현재 시스템의 한계를 극복하고 더욱 발전된 무인 경계 로봇 시스템을 구현하는 것을 목표로 합니다.

먼저, 하드웨어 성능 개선 및 고도화에 중점을 둘 것입니다. 현재 시스템은 라즈베리 파이를 기반으로 YOLOv8n 모델을 사용하고 있습니다. 향후 연구에서는 더 강력한 연산 능력을 갖춘 고성능 마이크로 컨트롤러를 도입하여 시스템의 전반적인 성능을 향상시킬 계획입니다. 이를 통해 더 정확하고 복잡한 객체 인식 모델을 적용할 수 있게 되어, 탐지 및 추적의 정확도와 속도를 크게 개선할 수 있을 것으로 기대됩니다.

다음으로, 현재의 단일 카메라 시스템을 확장하여 다중 카메라 시스템을 구현할 예정입니다. 여러 대의 카메라를 연동하여 더 넓은 영역을 효과적으로 커버할 수 있는 시스템을 개발함으로써 광역 감시 능력을 강화할 것입니다. 또한, 다중 카메라의 데이터를 종합하여 3D 공간에서의 정확한 객체 위치 추적 및 분석 기능을 구현하여 입체적인 객체 추적이 가능하도록 할 것입니다.

시스템의 보안성 강화도 중요한 연구 방향 중 하나입니다. 카메라에서 서버, 서버에서 클라이언트로의 모든 데이터 전송 과정에 강력한 엔드-투-엔드 암호화를 적용할 것입니다. 또한, 사용자 인증 및 권한 관리 시스템을 구축하여 승인된 사용자만이 시스템에 접근할 수 있도록 하고, 시스템의 모든 활동을 기록하고 분석할 수 있는 보안 로깅 시스템을 구현할 계획입니다.

객체 인식의 정확도를 높이고 더 스마트한 대응을 위해 지능형 객체 분류 및 대응 시스템을 개발할 것입니다. 다양한 객체 카테고리를 정의하고, 이를 바탕으로 더 세밀한 객체 분류 시스템을 구현할 예정입니다. 사용자가 특정 객체 카테고리에 대한 대응 방식을 직접 설정할 수 있는 인터페이스를 개발하고, 인공지능을 활용하여 상황을 분석하고 최적의 대응 방식을 자동으로 선택하는 시스템을 구현할 것입니다.

마지막으로, 현재의 고정식 포탑 시스템을 더욱 발전시켜 이동이 가능한 경계 로봇 시스템을 개발할 계획입니다. 지정된 경로를 따라 자율적으로 이동하며 감시할 수 있는 자율 주행 기능을 구현하고, 실시간으로 주변 환경을 인식하고 최적의 경로를 찾아 이동할 수 있는 알고리즘을 개발할 것입니다. 더 나아가 여러 대의 모바일 경계 로봇이 서로 협력하여 효율적으로 넓은 지역을 감시할 수 있는 다중 로봇 협업 시스템을 구축할 예정입니다.

이러한 향후 연구 방향을 통해, 우리의 무인 경계 로봇 시스템은 더욱 정확하고, 안전하며, 지능적인 보안 솔루션으로 발전할 것입니다. 이는 다양한 분야에서의 활용 가능성을 더욱 높이고, 궁극적으로 더 안전한 사회 구축에 기여할 것으로 기대됩니다.

7. 참고 문헌

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779-788, 2016.

[2] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Weng, Y. Yuan, and P. Luo, "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," *arXiv preprint arXiv:2110.06864*, 2021.
[3] M. Grinberg, "Flask Web Development: Developing Web Applications with Python," O'Reilly Media, 2018.

[4] V. Mazzia, A. Khaliq, F. Salvetti, and M. Chiaberge, "Real-Time Apple Detection System Using Embedded Systems With Hardware Accelerators: An Edge AI Application," *IEEE Access*, Vol. 8, pp. 9102-9114, 2020.

[5] W. R. Stevens, B. Fenner, and A. M. Rudoff, "UNIX Network Programming: The Sockets Networking API," Vol. 1, 3rd Ed., Addison-Wesley Professional, 2003.

[6] A. S. Tanenbaum and D. J. Wetherall, "Computer Networks," 5th Ed., Prentice Hall, 2011.

[7] V. Wang, F. Salim, and P. Moskovits, "The Definitive Guide to HTML5 WebSocket," Apress, 2013.

[8] Waveshare Electronics. (2024). Servo Driver HAT [Online]. Available: https://www.waveshare.com/wiki/Servo_Driver_HAT (downloaded 2024, Oct. 15)

[9] Raspberry Pi Foundation. (2024). Camera Module 3 [Online]. Available: https://www.raspberrypi.com/products/camera-module-3/ (downloaded 2024, Oct. 15).