문제정의서

팀명: 토끼 용

프로젝트명: Raspberry PI 5를 이용한 Computer Vision Suite 개발

개발 배경 및 필요성

컴퓨터 비전을 적용한 제품을 개발하기 위해서는 적절한 HW와 SW 기술의 조합이 필요하다. Raspberry Pi 5 는 가격이 저렴하고 AI 모델을 실행하기에 좋은 성능을 가지고 있다. RPI 5에서 실행되는 Computer Vision 모델을 실행하기에 좋은 성능을 가지고 있다. RPI에서 실행되는 Computer Vision 모델을 모아서 정리한다면 연구 혹은 제품개발에 손쉽게 사용될 수 있다.

문제해석

RPI 5같은 임베디드 컴퓨터는 로봇 및 컴퓨터 비전 프로젝트에 널리 활용되고 있다. 하지만 다양한 CV 모델들을 비교해 선택하는 과정은 많은 시간이 소요된다.
따라서 Docker Container에 CV 모델들을 넣어 UI로 관리하려 한다.

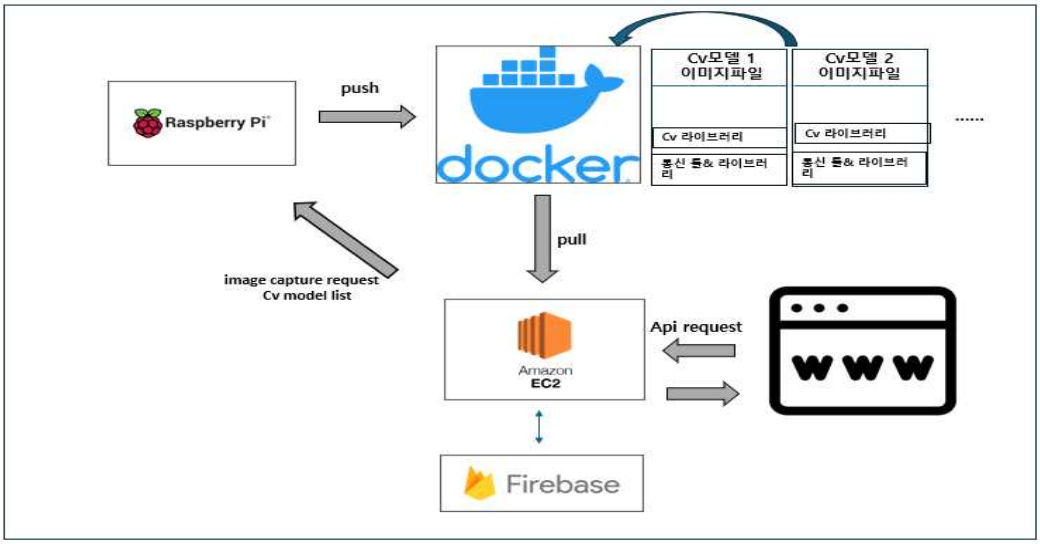
개발 요구 사항

1. Raspberry PI 5 와 CSI 카메라를 준비.
2. 카메라와 AI 프레임워크를 실행을 위한 Docker 환경 준비.
3. 다양한 AI 모델을 선택적으로 실행하고 테스트할 수 있는 Web UI개발.
4. AI 모델을 선택적으로 실행하거나 결과를 전송하는 websoket 등을 이용한 서버를 개발.
5. 통신을 위해서는 JSON등을 이용한 프로토콜을 정의한다.

프로젝트 진행 방향

개발환경 설정

ARM 아키텍처에 맞는 Docker 이미지를 생성하고, 필요한 소프트웨어 및 라이브러리를 설치한다.



1. 사용자가 특정한 AI Computer Vision 모델을 선택해 Raspberry Pi에서 이미지 분석을 요청한다.
2. 사용자의 요청은 서버에 도착, 서버는 API를 통해 이를 처리.
3. 서버는 사용자가 요청한 AI CV 모델을 실행할 도커 컨테이너를 준비, 이를 위해 이미지 촬영 요청을 RPI로 전송.
4. RPI는 CSI 카메라로 이미지를 촬영, 도커에 전달.
5. 도커는 받은 이미지를 선택된 AI CV 모델로 분석, 결과를 서버로 전송.
6. 서버는 분석된 결과를 DB에 저장.
7. 서버는, 클라이언트에게 분석된 결과 전송.

소프트웨어 개발

Docker 환경에서 실행될 Computer Vision 모델들을 선정 및 개발하고 **Docker Container에 넣는다**.

RPI 5에서 실행하기에 적합한 모델(컴퓨팅 리소스가 적은 모델)

1. MobileNet
가벼우면서도 높은 정확도, 모바일 및 임베디드 기기에서의 실행에 최적(이미지 분류, 객체 검출)
2. YOLO
빠른 추론 속도, 실시간 객체 검출에 적합(실시간 객체 검출)
3. Tiny YOLO
YOLO의 경량 버전으로, 매우 제한된 컴퓨팅 리소스에서도 사용 가능.(실시간 객체 검출)
4. EfficientDet
효율성과 정확도 사이의 높은 균형, 리소스 제한 환경에서도 높은 성능 유지(실시간 객체 검출, 이미지 내 다수의 객체 인식 및 분류)
5. SqueezeNet
작은 모델 크기에 초점을 맞춘 CNN 아키텍처, 모바일 또는 임베디드 시스템에서의 실행을 고려하여 설계, AlexNet 수준의 정확도를 유지하면서 모델 크기를 대폭 줄임.(이미지 분류, 기본적인 CV 작업)
6.Siamese Networks
두 입력 이미지 사이의 유사도를 측정하는데 사용, 이를 통해 같은 객체인지 다른 객체인지를 판별(얼굴 인식, 이미지 매칭, 감시 시스템에 사용)

컴퓨팅 자원이 많은 모델

1. OpenPose
사람의 자세를 실시간으로 추정할 수 있는 고급모델. 상대적으로 많은 컴퓨팅 자원을 요구(포즈 추정)
2. Mask R-CNN

객체 검출과 함께 객체의 정확한 영역을 분할하는 것이 가능한 모델, 이미지 내 객체의 정밀한 위치와 형태를 파악하는 데 사용.(객체 검출 및 세그먼테이션, 고정밀 객체 인식 및 추적)

웹 UI 개발 및 서버 개발

1. 사용자가 AI 모델을 선택적으로 실행할 수 있도록 도커 환경 내에 통합한다.
2. 모델의 성능과 호환성 테스트를 한다.
3. 반응형 디자인을 고려해 모바일과 데스크톱에서 모두 사용 가능하도록 한다.
4. AI 모델 선택 및 실행 결과 전송을 위한 웹소켓 서버를 구축한다. JSON등의 프로토콜을 사용해 클라이언트와 서버 간의 통신을 정의한다.

테스트 및 최적화

1. 기능 테스트: 개발된 웹 UI와 웹소켓 서버가 예상대로 작동하는지 테스트한다. 사용자 입장에서의 경험도 평가한다.
2. 성능 최적화: ㄴAI 모델의 실행 시간, 웹 서버의 응답 속도 등을 분석하고 최적화한다. 필요시 하드웨어 리소스 할당을 조정한다.

배포 및 모니터링

1. 배포: 모든 개발이 완료 되면 시스템을 실제 환경에 배포한다
2. 모니터링 및 유지 보수: 시스템의 성능을 지속적으로 모니터링하고, 발생할 수 있는 문제에 대해 즉각적으로 대응한다.