프로젝트 상세설계보고서

보고서 및 논문 윤리 서약

1. 나는 보고서 및 논문의 내용을 조작하지 않겠습니다.
2. 나는 다른 사람의 보고서 및 논문의 내용을 내 것처럼 무단으로 복사하지 않겠습니다.
3. 나는 다른 사람의 보고서 및 논문의 내용을 참고하거나 인용할 시 참고 및 인용 형식을 갖추고 출처를 반드시 밝히겠습니다.
4. 나는 보고서 및 논문을 대신하여 작성하도록 청탁하지도 청탁받지도 않겠습니다.

나는 보고서 및 논문 작성 시 위법 행위를 하지 않고, 명지인으로서 또한 공학인으로서 나의 양심과 명예를 지킬 것을 약속합니다.

보고서명 : 프로젝트 상세설계보고서

학 과 : 컴퓨터공학과
과 목 : 캡스톤디자인
담당교수 : 김상귀 교수님
마 감 일 : 2024년 4월 14일
제 출 일 : 2024년 4월 14일

팀 명 : 가 려 조

팀 장	학번 :	60192218	이름 :	우승화	(서명)
팀 원	학번 :	60192208	이름 :	석성훈	(서명)
팀 원	학번 :	60192230	이름 :	이훈근	(서명)
팀 원	학번 :	60192231	이름 :	임준원	(서명)

설계/프로젝트 상세설계보고서

실시간 초상권 보호 및 유해 콘텐츠 AI 필터링 프로그램

(Real-time portrait rights protection and harmful content AI filtering program)

제안 상품명 : 실시간 초상권 보호 및 유해 콘텐츠 AI 필터링 프로그램
발 주 자 : 명지대학교 컴퓨터공학과 김상귀 교수님
보고기관 : 명지대학교 컴퓨터공학과
설 계 팀 : 가 려 조 (명지대학교 컴퓨터공학과 캡스톤디자인1 A-3조)
작성일자 : 2024.04.01. ~ 2024.04.14.
문서버전 : V 1.0

<목차>

요약문

본 프로젝트는 안전하고 건전한 방송 콘텐츠를 제작하기 위해 등록되지 않은 사람들의 초상권을 보호하고 유해 콘텐츠를 필터링하는 프로그램을 개발하는 것을 목표로 한다. 최근 국내 인터넷 방송 시장이 성장하면서 해당 기업의 엄격한 방송 규제가 예상되며, 초상권 보호 및 방송 통신 심의 규정 준수를 위한 AI 기술을 활용하여 영상을 자동 분석하고 검열하는 프로그램을 개발할 예정이다. 이를 통해 수작업 방식에 비해 인적 자원 소모를 줄이고, 개인 방송업계의 수위 기준 준수를 더 원활하게 하고자 본 프로젝트를 진행한다. 이러한 프로그램은 안전한 방송 환경 조성과 개인 방송 시장의 활성화에 기여할 것으로 기대하고 있다.

Summary

This project aims to develop a program to protect the rights of unregistered individuals and filter harmful content in order to produce safe and wholesome broadcast content. With the recent growth of the domestic internet broadcasting market, strict broadcasting regulations are anticipated, prompting the utilization of AI technology to automatically analyze and censor videos to comply with regulations on protecting personal rights and broadcasting communications review. By reducing the reliance on manual labor, the project seeks to facilitate compliance with standards in the individual broadcasting industry more smoothly. It is expected that such a program will contribute to creating

1. 설계/프로젝트 개요

1.1 프로젝트 목적

본 프로젝트는 등록되지 않은 사람들의 초상권을 보호하고 시청자들에게 안전한 콘텐츠를 제공하기 위해 실시간으로 인물을 가려주고 유해 콘텐츠를 필터링하는 프로그램을 개발하는 것이다. 이를 통해 방송사와 콘텐츠 제작자들은 법적 문제나 규제에 대한 걱정 없이 안전하고 건전한 방송 콘텐츠를 제작할 수 있게 될 것이다.

1.2 프로젝트 시장성

최근 해외 인터넷 방송 플랫폼이 한국 서비스 철수함에 따라 국내 기업에서 새로운 플랫폼을 출시하는 등 인터넷 방송 시장이 국내 비중이 증가하게 되었다. 또한 해당 기업의 방송 규제 수준이 국내 방송법 준수 이상이기에 엄격한 방송 규제가 예상된다. 대한민국의 초상권이 타국에 비해 지나치게 넓게 인정되는 경향도 있기에 방송인의 편의 및 규제 준수를 위해 자동 필터링 서비스의 수요가 높아질 것이라 예상된다.

1.3 프로젝트 목표

본 프로젝트의 목표는 인터넷 영상 플랫폼에서 요구되는 초상권 보호 및 방송 통신 심의 규정 준수를 위해 필요한 영상의 검열을 자동화하는 것으로 AI 컴퓨터 비전 기술을 활용하여 영상을 실시간으로 분석하고, 불필요한 콘텐츠를 탐지하고 삭제하는 프로그램을 개발하는 것이다.

1.4 프로젝트 기대효과

본 프로젝트는 영상 제작 및 실시간 영상 송출과 관련된 검열 작업을 AI 기술을 활용하여 자동화하는 것을 목표로 한다. 이를 통해 기존의 수작업 방식에 비해 인적자원 소모량을 획기적으로 줄일 수 있을 것으로 예상되며 실시간 화면 검열 기능을 제공함으로써 개인 방송업계에서 요구되는 수위 기준을 더 원활하게 준수할 수 있도록 하여 개인 방송 시장의 활성화에 기여할 것으로 기대된다.

1.5 프로젝트 팀

1.5.1 팀 명

가려조

1.5.2 팀 구성원

직책	이름	학번	연락처	기술 및 배경
팀장	우승화	60192218	010-7534-4019	C#, Java, Js, React-Native, spring-boot, MySQL
팀원	석성훈	60192208	010-3304-2783	C, C++, Django, Java
팀원	이훈근	60192230	010-3290-0642	C, C++, python, Android Studio, Unity
팀원	임준원	60192231	010-7339-6846	C, C++, Spring-Boot, Oracle, MySQL, Android Studio

[표 1] 팀 구성 및 기술/배경

2. 현실적 제한조건

현실적 제한조건	상세 제한조건	내용
관리적 제한	시간	제품 개발기간을 10주로 한다.
	예산	제품 개발 및 제작비용은 5만원 이하로 한다. 제품 서버/테스트 비용은 10만원 이하로 한다.
	인적자원	제품 개발에 투여되는 인적자원 (MM(Man-Month))을 12로 한다.
	물적자원	▷제품 제작에 사용될 작업 환경 사양 - Laptop (Windows/Intel-i5) 4대 - Desktop (Windows/Intel-i5) 3대 ▷제품이 사용될 테스트 환경 사양 - Android(v12 이상), Ios(v5.0 이상) 시반의 모바일 - 노트북, 데스크탑 PC
시스템적 요구	성능	서비스 요청에 대한 응답시간은 1초 이하로 한다. 실시간 방송의 지연시간은 5초 이하로 한다. 영상 frame의 fps는 30 이상 송출이 가능하게 한다.
	기능성	제품의 기능적 요구사항인 초상권 모자이크, 유해 콘텐츠 필터링 기능을 제공해야 한다.
	이식성	제품은 대규모 수정 없이 Windows, Mac, Ios, Android의 웹/앱 환경에서 작동할 수 있어야 한다.
	유지보수성	수정 예상 LOC(Line of Codes)가 200 이하가 되도록 한다.
	보안성	제품을 제작하는데 필요한 API 및 서비스 데이터가 외부로 유출되지 않도록 한다. 제품의 사용자의 데이터가 유출되거나 변조되지 않도록 한다.
	효율성	CPU 점유율을 30% 이하가 되도록 한다. GPU 점유율을 50% 이하가 되도록 한다.

[표 2] 현실적 제한조건

3. 설계/프로젝트 목적

3.1 프로젝트 핵심 목표

- ✓ 딥러닝을 활용하여 초상권을 보호하는 프로그램을 개발한다.
- ✓ 야외 무단 촬영 문제 및 개인 방송에서의 노출 문제를 해결하기 위해 자동 얼굴 인식 모자이크 기능을 제공한다.
- ✓ 야외에서 영상을 촬영할 때 행인의 얼굴에 대해 따로 모자이크하는 과정을 대신한다.
- ✓ 유해 콘텐츠 감지와 필터링을 통해 방송 심의 미준수에 대한 불이익을 최소화한다.
- ✓ 유해 콘텐츠를 실시간으로 필터링하여 방송 환경의 안전성을 높인다.
- ✓ 국내 방송 플랫폼에서의 이러한 기능들을 제공함으로써 규제 요구사항을 충족시키고, 시청자들에게 안전하고 적절한 콘텐츠를 제공한다.
- ✓ 기술적으로 정교한 딥러닝 알고리즘을 구현하여 실시간으로 처리할 수 있는 효율적인 시스템을 구축한다.
- ✓ 최종적으로, 이 프로그램을 통해 방송 환경의 안전성과 편의성을 향상하고, 국내 방송 플랫폼의 경쟁력을 강화한다.

3.2 기능적/비기능적 요구분석

구분	내용
기능적 요구사항	1. 자동 얼굴 인식 및 모자이크 기능 - 프로그램은 딥러닝을 사용하여 실시간으로 영상에서 얼굴을 인식하고 모자이크 처리해야 한다. - 야외에서의 무단 촬영이나 개인 방송에서의 얼굴 노출을 방지하기 위해 이 기능이 필요하다. 2. 유해 콘텐츠 감지와 필터링 - 딥러닝 알고리즘을 활용하여 콘텐츠를 실시간으로 분석하고 유해 콘텐츠를 식별해야 한다. - 유해 콘텐츠는 음란물, 폭력적인 내용, 불법 행위 등을 포함한다. - 발견된 유해 콘텐츠는 즉시 필터링되어 방송 중단되거나 해당 부분이 모자이크 처리되어야 한다. 3. 국내 방송 플랫폼 규제 요구사항 충족 - 프로그램은 국내 방송 플랫폼의 운영 규정을 준수해야 한다. - 부적합한 콘텐츠 제한, 콘텐츠 등급 정책, 가이드라인 위반 시 조치 등을 준수해야 한다. 4. 효율적인 시스템 구축

	- 프로그램은 딥러닝 알고리즘을 효율적으로 구현하여 실시간으로 처리할 수 있는 시스템이어야 한다. - 실시간 처리를 위한 빠른 속도와 정확도가 요구된다.
비기능적 요구사항	1. 정확성과 신뢰성 -얼굴 인식 및 콘텐츠 분석은 높은 정확성과 신뢰성을 가져야 한다. -오분류나 오류가 발생할 경우 사용자의 개인 정보 보호와 콘텐츠의 신뢰성에 영향을 줄 수 있다. 2. 실시간 처리와 응답 시간 - 프로그램은 실시간으로 영상을 처리하고 콘텐츠를 분석해야 한다. - 응답 시간이 빠르고, 지연 없이 영상에 대한 처리가 이루어져야 한다. 3. 확장성과 유지보수성 - 시스템은 확장 가능하고 유지보수가 용이해야 한다. - 새로운 기능 추가나 업그레이드, 버그 수정 등을 용이하게 할 수 있어야 한다. 4. 보안 및 개인 정보 보호 - 사용자의 개인 정보와 영상 내용은 보안적으로 처리되어야 한다. - 모자이크 처리된 영상이나 콘텐츠 분석 결과는 외부로 유출되지 않아야 한다. 5. 사용자 경험과 편의성 - 프로그램 사용이 쉬워야 하며, 사용자에게 편의성을 제공해야 한다. - 복잡한 설정이나 조작 없이 자동으로 작동하여 사용자가 부담 없이 이용할 수 있어야 한다.

[표 4] 기능적/비기능적 요구 분석

4. 설계 결정에 대한 진행 상황

Model-View-Controller (MVC) 아키텍처

MVC 아키텍처는 사용자 인터페이스와 비즈니스 로직을 분리하여 개발하기 쉽고 유지보수에 유리하다. 객체 감지 및 필터링은 모델 부분에 포함될 수 있다. 또한, PyQt를 사용하여 GUI를 구축하는 데 적합하다.

Pipeline 아키텍처

입력 데이터를 일련의 단계로 처리하여 결과를 생성한다. 웹캠 입력을 받아 객체 감지 및 필터링을 수행하는 연속적인 단계로 구성할 수 있다. 실시간 서비스에 적합한 아키텍처이다.

Layered 아키텍처:

여러 계층으로 구성된 구조로, 계층 간의 강한 의존성이 있을 수 있다. 웹캠 입력 처리, 객체 감지 및 필터링, GUI 표현 등의 계층으로 나눌 수 있지만, 실시간 처리를 위해 이러한 계층 간의 지연을 최소화해야 한다.

Event-Driven 아키텍처

이벤트 기반 시스템으로, 사용자 입력이나 객체 감지 결과와 같은 이벤트에 반응하여 처리된다. 실시간 서비스에 적합하며, PyQt와의 통합도 잘된다.

Service-Oriented 아키텍처 (SOA)

서비스 단위로 구성된 구조로, 각 서비스는 특정 기능을 제공한다. 사용자 입력 처리, 객체 감지 및 필터링, GUI 표현 등의 서비스로 구성할 수 있다. 하지만, 이는 단일 사용자의 경우에는 적절하지 않을 수 있다.

아키텍처	실시간 성능	사용자 편의성	개발 용이성
Model-View-Controller (MVC)	높음	높음	높음
Pipeline	높음	높음	중간
Layered	중간	중간	높음
Event-Driven	높음	높음	중간
Service-Oriented (SOA)	중간	중간	중간

4.1 문제해결에 적용된 이론

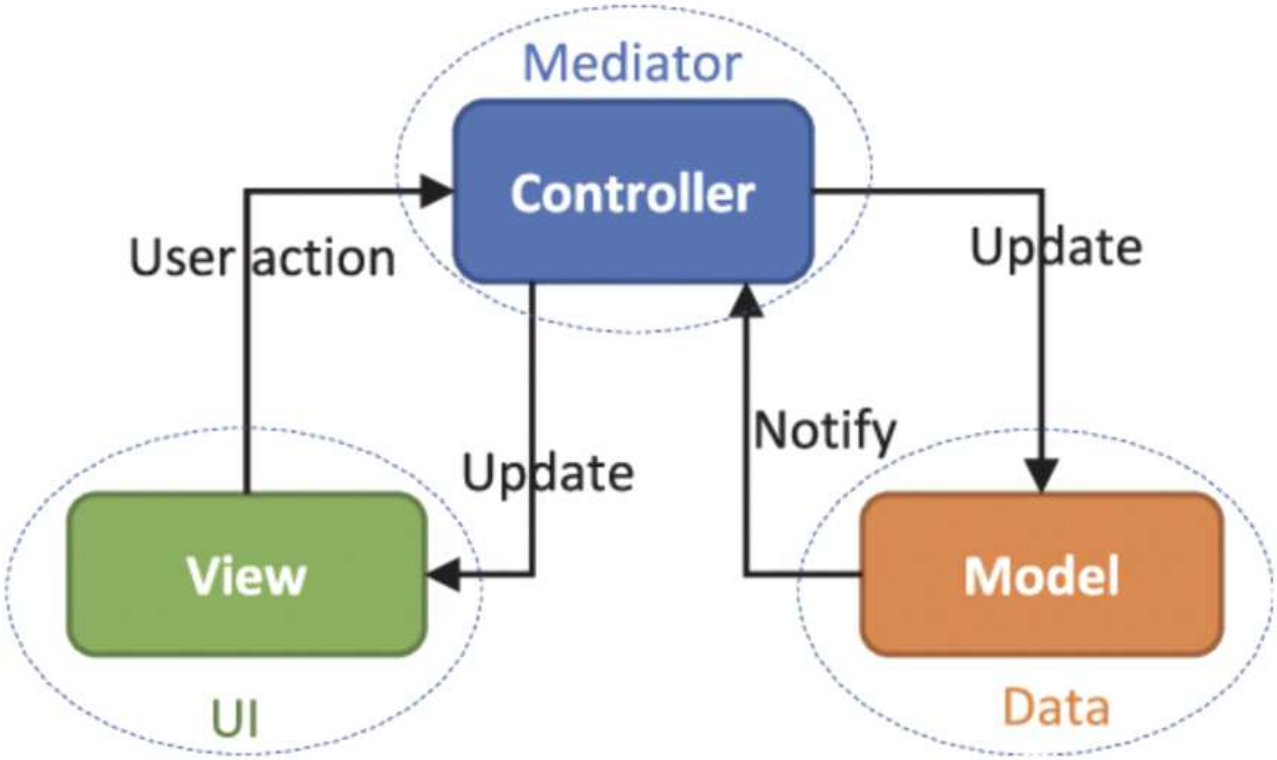
4.1.1 GPGPU : GPU 범용 연산

YOLO와 DLIB처럼 딥러닝 기반 Face Recognition과 Object Detection은 대용량의 이미지나 비디오 데이터를 처리하기 때문에 고성능의 연산이 필요하다. 또한 GPU는 많은 수의 코어를 갖고 있기에 병렬적으로 처리될 수 있는 작업에 대해 높은 성능을 보일 수 있다. 결정적으로 실시간 필터링

작업을 중요시하는 이 프로젝트에서 CPU 연산의 성능 문제로 인해 프레임 저하가 일어나는 상황을 예방하기 위하여 GPU를 사용하는 CUDA 모델을 사용하기로 결정했다.

그러나 GPU를 사용한 연산을 기본 사양으로 결정하였기 때문에 사용자의 디바이스에 일정 수준 이상의 GPU가 장착되어 있어야 하는 제약 조건이 발생할 수 있다.[1)]

4.1.2 개발 아키텍처 - MVC



Model-View-Control(MVC) design pattern.

< MVC 패턴 >

모델-뷰-컨트롤러(model-view-controller, MVC)는 소프트웨어를 구성하는 세 가지 주요 부분을 분리하여 개발하는 아키텍처 패턴이다. 사용자 인터페이스로부터 비즈니스 로직을 분류하여 시각적 요소와 내부 기능을 서로 영향 없이 쉽게 고칠 수 있는 애플리케이션을 만들 수 있다.

모델은 애플리케이션의 데이터와 비즈니스 로직을 나타낸다. 이 소프트웨어에서는 입력된 이미지로부터 객체와 얼굴을 탐지하고, 인식하는 기능이 해당한다.

뷰는 사용자에게 보여주는 화면(UI)이 해당된다. 이 소프트웨어는 독립적으로 작동하는 응용 프로그램으로 사용자에게 보여지는 GUI를 생성하는 기능이 해당한다.

컨트롤러는 모델과 뷰 사이를 이어주는 인터페이스 역할을 한다. 모델은 뷰를 통해 데이터를 전달받지 않으며 뷰는 모델이 가진 데이터를 따로 저장하지 않는다. 즉 모델과 뷰는 서로의 존재를 알지 못해야 하는데, 뷰의 요청에 따라 모델이 데이터를 어떻게 처리할지 알려주는 역할을 한다.

기능별로 코드를 분리하여 구성요소들을 독립시키기 때문에, 협업할 때 각자 맡은 부분의 개발에 집중할 수 있다는 장점이 있다. GUI, Object Detection, Face Recognition의 개발 항목이 뚜렷하게 나뉘어 있는 프로젝트 특성상 개발의 효율성을 크게 높일 수 있을 것이라 판단되어 해당 설계 패턴을 채택했다.

1) 부록. CPU/GPU 성능 테스트

4.1.3 Model

DLIB는 얼굴 인식과 표정 분석에 주로 사용되며, HOG (Histogram of Oriented Gradients)와 SVM (Support Vector Machine)을 기반으로 한 얼굴 검출기를 포함하고 있다. 이를 통해 이미지에서 얼굴을 찾아내고 주요 포인트를 식별하여 얼굴을 특징화하며 인식한다. 또한, DLIB는 얼굴 이미지의 정렬에도 활용되며, 얼굴을 표준화된 형태로 회전, 크기 조정, 이동시킨다. 이렇게 정렬된 얼굴 이미지는 얼굴 인식 및 분석 작업에서 더 정확한 결과를 얻을 수 있다. 더불어 DLIB는 객체 탐지에도 사용되며, 이 과정에서 HOG와 SVM을 활용하여 객체를 탐지하고 경계 상자를 생성한다.

YOLOv8은 객체 탐지 알고리즘 중 하나로, 이미지를 한 번만 보고 객체의 경계 상자와 클래스를 예측하는 Single Shot Detector(SSD) 형식을 따른다. 이로 인해 다른 탐지 알고리즘에 비해 훨씬 빠른 속도로 객체를 감지한다. YOLOv8는 다양한 기술과 아키텍처를 결합하여 높은 정확도와 효율성을 달성한다.

YOLOv8의 특징 중 하나인 Anchor Boxes는 이미지를 그리드로 나누고 각 그리드 셀에서 여러 개의 경계 상자를 예측하는 방식을 사용한다. 이를 통해 다양한 크기와 종횡비를 가진 객체를 효과적으로 탐지한다. Darknet 아키텍처를 기반으로 하는 YOLOv8는 가벼우면서도 효율적인 신경망 구조를 가진다. 이 구조를 통해 다양한 백본 네트워크를 적용하여 특징 추출을 개선할 수 있다. 또한, Feature Pyramid Network (FPN)을 활용하여 다양한 크기의 특징 맵에서 객체를 탐지하며, 여러 최적화 기법을 도입하여 모델의 정확도와 속도를 향상시킨다.

CNN은 두 기술 모두에서 핵심적으로 활용되며, 이미지에서 객체나 얼굴의 특징을 추출하는 데 중요한 역할을 한다. 백프로파게이션은 신경망을 효율적으로 학습시키는 데 활용되며, 앵커 박스는 객체의 가능한 위치를 사전에 정의하여 빠르게 객체를 감지한다. YOLOv8은 객체 감지 정확도를 향상시키기 위해 Non-Maximum Suppression (NMS)와 Attention 메커니즘을 활용하며, Spatial Pyramid Pooling (SPP)과 Path Aggregation Network (PANet)을 통해 다양한 크기와 위치의 객체를 효과적으로 탐지한다. 추가로 YOLOv8는 Focus Layer, CutMix, Mosaic과 같은 데이터 증강 기술을 적용하여 모델의 일반화 성능을 향상시킨다. 이 모든 기술과 방법들을 종합적으로 활용하여 객체 감지와 위치 지정의 정확도와 속도를 극대화한다.[2)]

4.1.4 Fine tuning

YOLOv8은 다양한 객체들을 인식하기 위해 사용하는 범용 모델로 우리가 제작하려는 유해 객체 필터링이라는 특정 분야에 집중된 기능을 수행하기엔 전반적으로 성능이 떨어졌다. 이를 해결하기 위해 전이 학습 중 하나인 Fine tuning을 이용하여 유해 객체 필터링에 적합한 모델을 생성했다.

Fine-tuning은 기계 학습에서 모델을 세밀하게 조정하여 특정 작업에 더 잘 맞추도록 하는 프로세스를 가리킨다. 주로 사전 학습된 모델을 가져와서 특정 작업에 맞게 조정하는 데 사용된다.

일반적으로, 사전 학습된 모델은 큰 데이터세트에서 학습되어 다양한 특징을 학습한다. 그런 다음 이러한 모델을 가져와 새로운 작업에 적용하기 위해 미세 조정(fine-tuning)한다. 예를 들어, 자연어 처리에서 BERT나 GPT와 같은 사전 학습된 언어 모델을 가져와 특정 언어 이해 작업이나 생성 작업에 맞게 조정할 수 있다.

Fine-tuning은 보통 다음과 같은 단계로 이루어진다:

기존 모델 불러오기: 사전 학습된 모델을 선택하고 불러온다.

출력 레이어 변경: 새로운 작업에 맞게 모델의 출력 레이어를 변경하거나 추가한다.

하이퍼파라미터 조정: 모델의 학습률, 배치 크기 등의 하이퍼파라미터를 조정한다.

새로운 작업 데이터로 모델 학습: 새로운 작업에 맞게 데이터를 사용하여 모델을 학습시킨다.

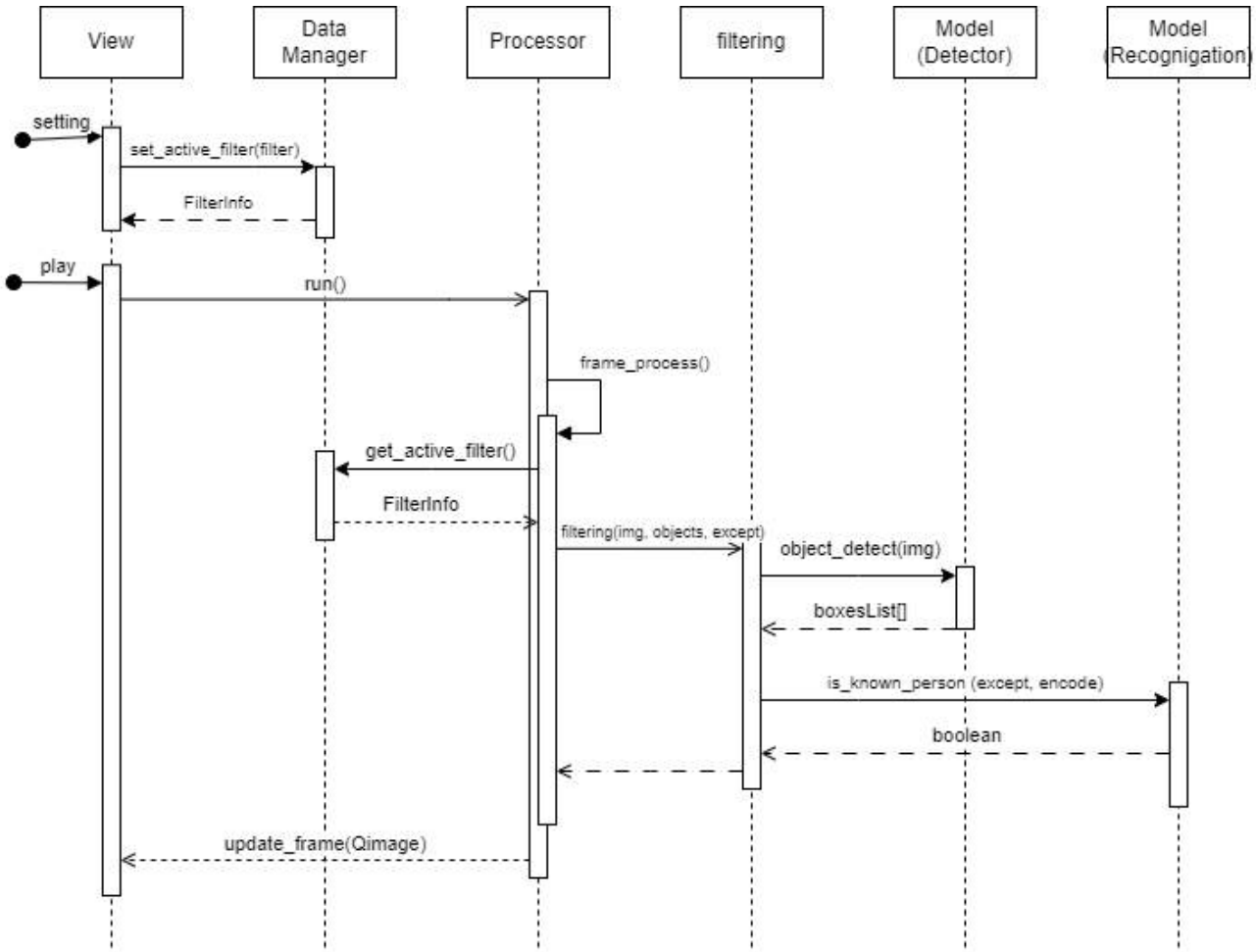
성능 평가: 학습된 모델을 평가하고 필요한 경우 추가적인 조정을 수행한다.

2) 부록. 얼굴 필터링 모델 테스트

4.2 SW를 이용한 설계

위와 같이 결정된 소프트웨어 설계를 체계적으로 정리하고, 작동 흐름을 정확하게 공유하기 위해 drawio를 사용하여 Sequence Diagram, Class Diagram, Flow Chart를 작성하였다. 또한 UI/UX 설계를 위해 디자인 툴인 Figma를 사용하였다.

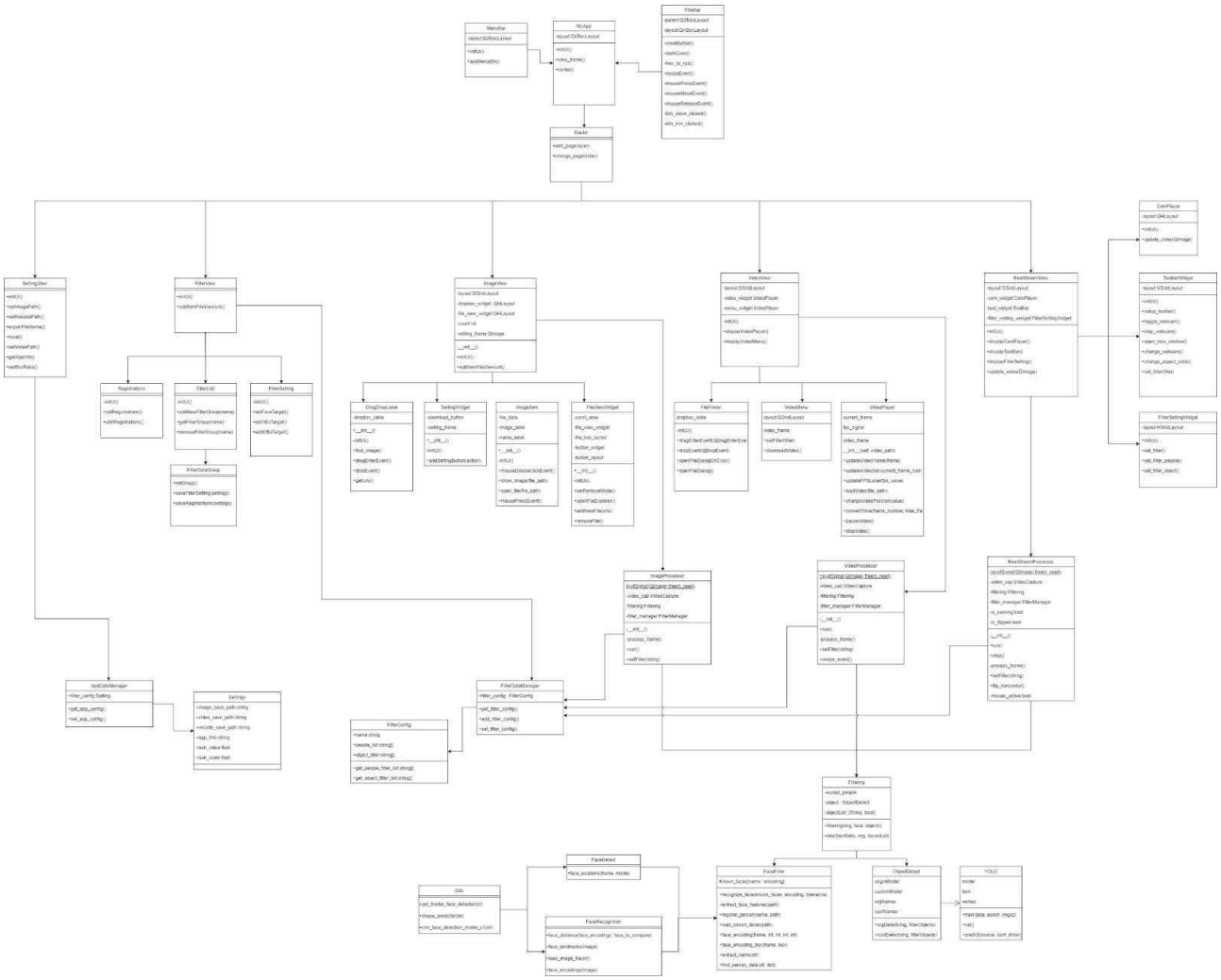
4.2.1 Sequence Diagram



< drawio을 사용한 Sequence Diagram >

해당 시퀀스 다이어그램은 프로그램 사용 시 뷰 - 컨트롤러 - 모델 간의 요청이 어떤 순서로 이루어지는지 나타낸다. 기본적으로 View단에서 사용자의 요청을 받은 뒤 요청에 따라 Data Manager와 Processor를 선택하여 호출한다. Processor는 전송받은 데이터를 추출하여 Filtering 클래스에 전송한다. Filtering은 전송받은 데이터를 Model들에게 전달하여 객체 및 얼굴 처리를 실행하고, 처리가 완료된 데이터를 반환한다.

4.2.2 Class Diagram



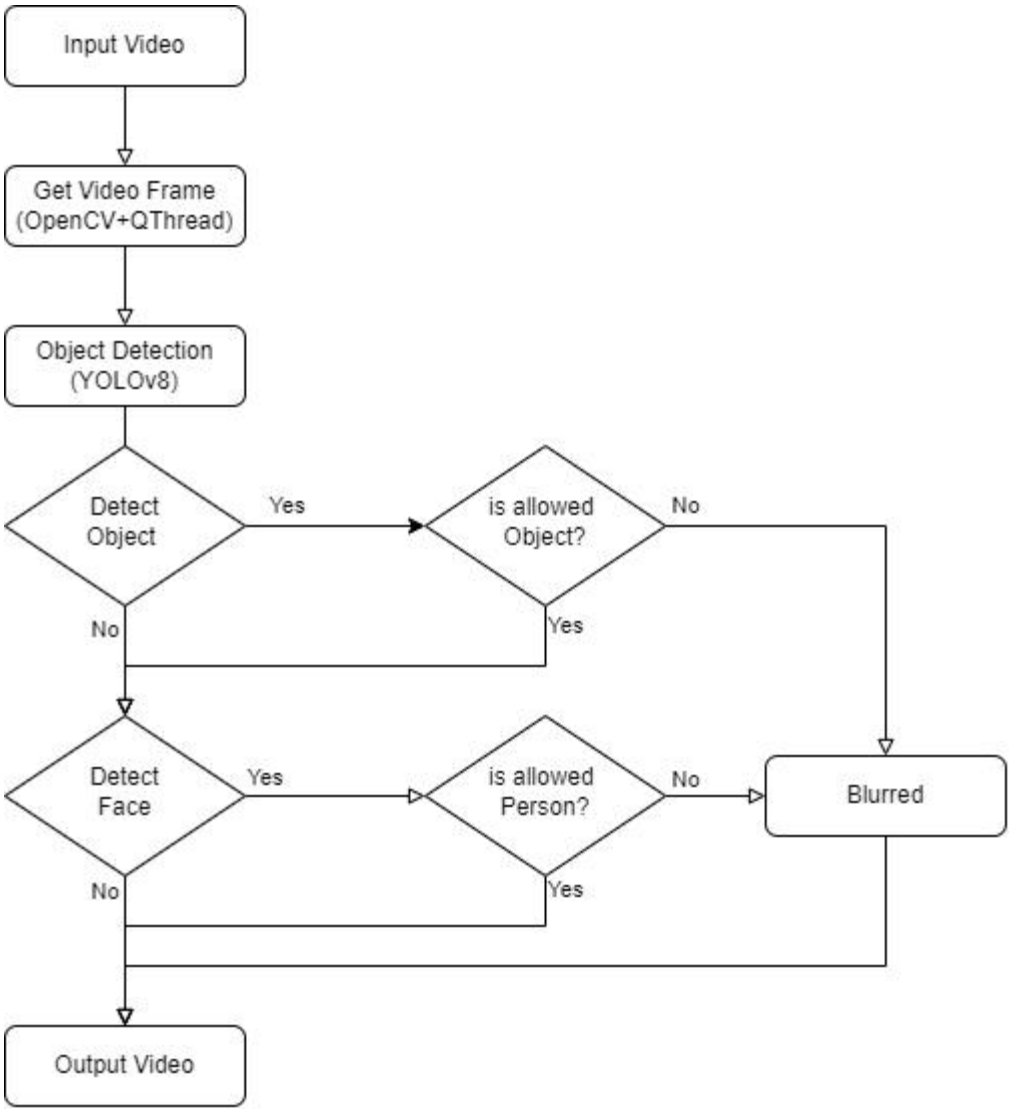
< drawio을 사용한 Class Diagram >

이 소프트웨어의 클래스 구조는 크게 3가지로 나눌 수 있다. 먼저 사용자에게 보여줄 GUI를 생성하는 클래스로 PyQt5 라이브러리를 통해 구현할 SettingView, FilterView, ImageView, VideoView, RealStreamView가 존재한다. 각각의 View들은 Label, Widget, Menu 등의 GUI 요소들을 가지며, 해당 요소들로 Controller를 통해 데이터를 주고받게 된다.

다음으로는 사용자가 GUI 통해 프로그램에 요청을 보냈을 때 어떤 요청인지, 무슨 데이터를 입력받았는지에 따라 작업을 수행하는 Processor 클래스들이 있다. ImageProcessor, VideoProcessor, RealStreamProcessor가 존재하며, 각각 Filtering 클래스를 활용하여 입력받은 데이터를 가공하고 필터링된 데이터를 GUI를 통해 출력한다.

마지막으로 객체와 얼굴을 탐지하고 구분하는 비즈니스 로직을 구현할 Filtering 클래스가 있다. Filtering 클래스는 각각 객체와 얼굴을 담당하는 FaceFilter와 ObjectDetect를 사용하여 구현된다. 각각 Face Recognition 모델인 DLIB과 Object Detection 모델인 YOLO를 사용하여 비즈니스 로직에 필요한 기능을 구현한다.

4.2.3 Flow Chart



< drawio을 사용한 Flow Chart >

개발 과정에서 각 기능에 특화된 API와 라이브러리를 사용하는 것이 가장 효율적이라는 판단하에, 각 기능에 해당하는 시스템을 개별적으로 만든 후 데이터 처리에 필요한 기능을 단계적으로 수행하는 구조가 적합하다고 결정했다. 이러한 구조는 개발 효율성을 높여줄 뿐만 아니라, 유지보수와 확장성 측면에서도 이점을 제공한다. 각 시스템이 독립적으로 작동하면서 필요한 데이터 처리를 수행하므로, 시스템 간의 의존성을 줄이고 개별 시스템의 변경이 다른 시스템에 미치는 영향을 최소화할 수 있다. 이는 전체적인 개발 프로세스를 단순화하고, 효율적으로 관리할 수 있게끔 도와준다.

4.2.4 UI/UX 설계

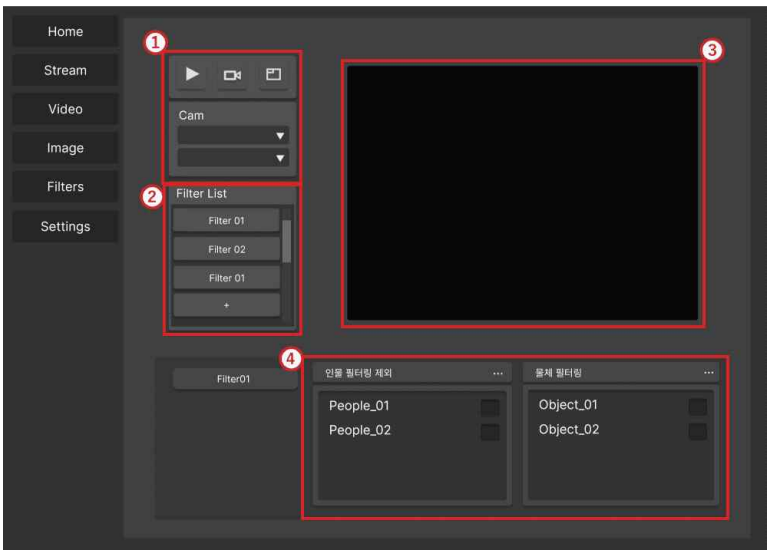
< figma를 사용한 UI/UX 설계 >

UI는 하나의 프레임에 각 기능을 담당하는 페이지를 추가하는 방식으로 구성하였다.
UI를 하나의 프레임에 각 기능을 담당하는 페이지를 추가하는 결정은 개발의 효율성과 사용자의 편의성을 동시에 고려한 결과이다. 이러한 구조는 개발 과정에서 각 페이지를 개별적으로 관리하고 수정할 수 있어서 팀원들 간의 협업을 용이하게 만들어주며 사용자는 각기 다른 기능에 대한 페이지를 명확하게 구분하여 접근할 수 있으므로 사용자 경험이 향상된다. 이는 사용자가 필요로 하는 기능에 더욱 빠르고 효과적으로 접근할 수 있게 도와줄 것이다.[3)]

3) 부록. 참고 GUI 설계

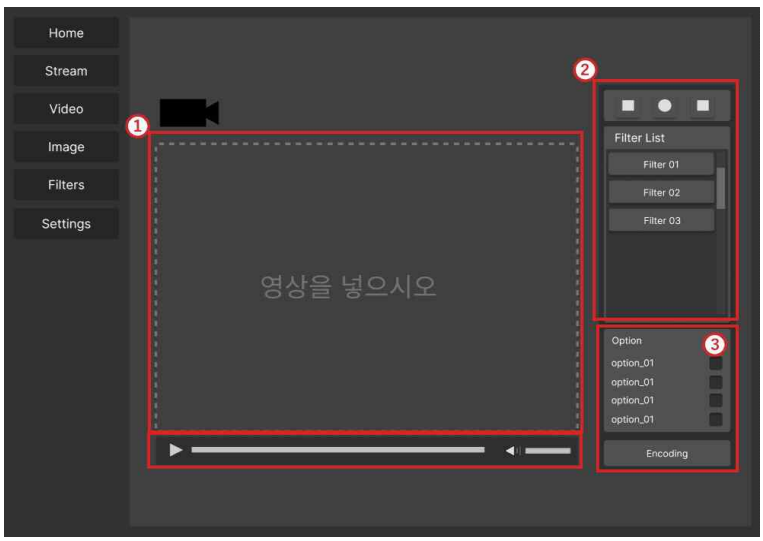
4.2.5 기능 설계

[1] 실시간 방송 필터링



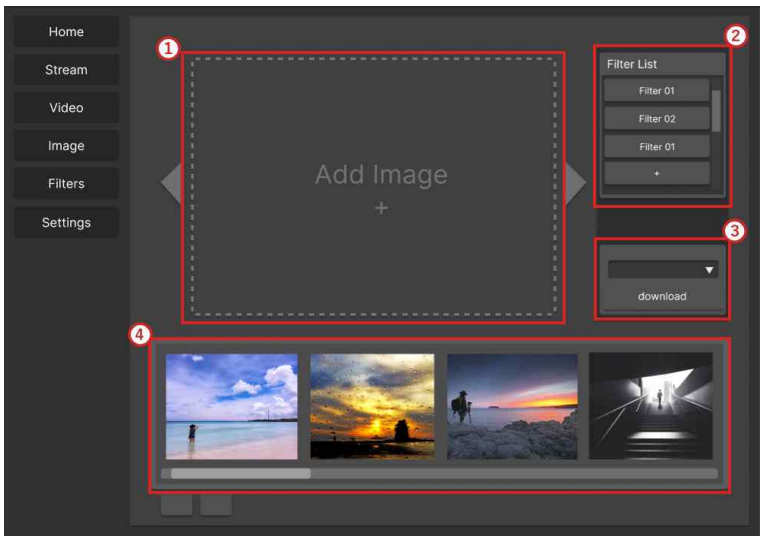
① **실시간 영상 송출 설정:** 실시간 스트리밍 CAM 및 영상 송출 환경에 관한 설정 기능 제공 (송출/녹화/새창 및 Cam 지정 기능, 송출 해상도 설정 등)
② **필터 리스트:** 사용자가 사전에 지정해준 필터 목록을 확인하고 적용할 수 있는 위젯
③ **영상 플레이어:** 현재 실시간 방송 송출 화면이 어떻게 디스플레이 되고 있는지 확인할 수 있는 송출 확인용 영상 플레이어
④ **필터 설정창:** 사용자 필터에서 부분적으로 설정할 수 있는 인스펙터 창, 기본적인 설정에서 간단한 설정 가능

[2] 비디오 영상 필터링



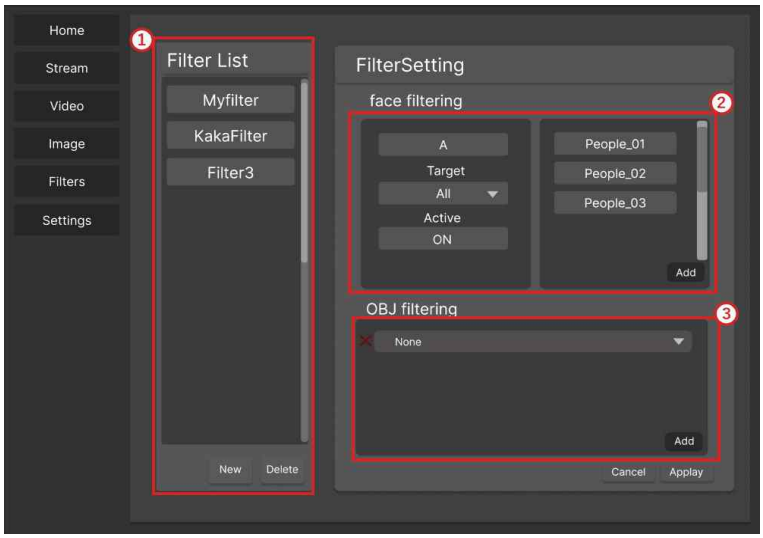
① **비디오 플레이어:** 편집할 영상을 재생하고 필터가 적용되었는지 확인하기 위한 Video Player, Drag&Drop으로 편집할 영상을 등록
② **필터 리스트:** 사용자가 사전에 지정해준 필터 목록을 확인하고 적용할 수 있는 위젯
③ **다운로드 및 설정:** 편집이 완료된 영상을 인코딩하고 다운로드 가능한 형태로 설정 후 다운로드 할 수 있는 기능 탑재

[3] 사진 필터링



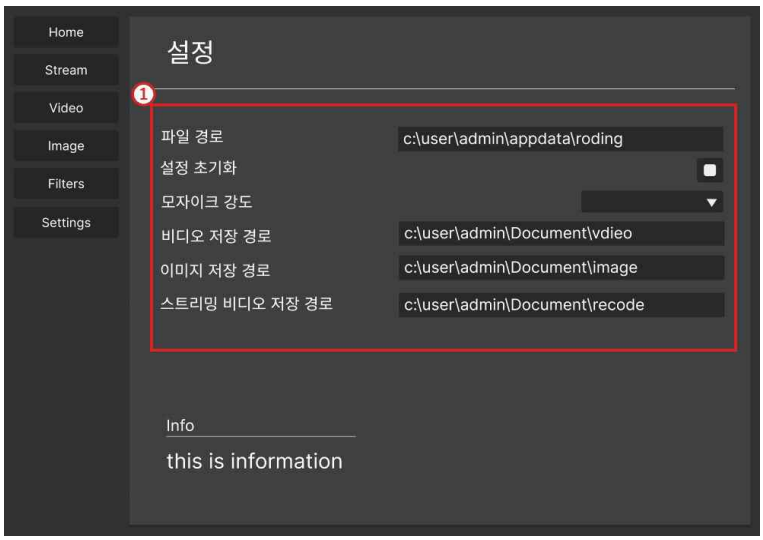
① **이미지 뷰어:** 편집할 사진을 열고 실제로 필터가 적용되었는지 확인하기 위한 Image Viewer, Drag&Drop으로 편집할 사진을 등록할 수 있으며 양 버튼을 통해 사진을 넘길 수 있다.
② **필터 리스트:** 사용자가 사전에 지정해준 필터 목록을 확인하고 적용할 수 있는 위젯
③ **다운로드 및 설정:** 필터가 적용된 사진을 다운로드할 형식과 방법을 설정하고 다운로드하는 기능
④ **이미지 뷰어 리스트:** 현재 편집이 가능한 이미지를 띄어줄 이미지 뷰어 리스트, 사진을 더블 클릭하면 (1) 이미지 뷰어로 이동하여 확인이 가능

[4] 필터 설정



① **필터 리스트:** 사용자가 사전에 지정해준 필터 목록을 확인하고 추가 및 삭제가 가능, 편집하고 싶은 필터는 선택하여 우측 설정(2), (3)을 통하여 필터 정보 설정
② **필터링 인물 등록 및 설정**: 필터링에서 예외 시킬 혹은 필터링시킬 인물 등록하고 활성화 여부 등록, 필터링 인물 등록시 필터링 인물 등록 창이 띄워지며 필터링할 인물의 사진을 넣어 등록
③ **유해 필터링 등록 및 설정:** 유해 매체로 필터링할 객체를 등록하고 활성화 여부를 설정

[5] 응용 프로그램 설정



① 외 APP에서 기본 설정 및 세부 설정을 확인하고 편집할 수 있는 설정창, 다른 시스템에서 사용되는 설정 등을 제어, (앱 파일, 다운로드 경로 결정, 모자이크 설정, 앱 정보등 확인하고 설정가능)

5. 설계/프로젝트 평가

5.1 시작품의 설계 및 제작 시험법

5.1.1 시작품의 설계

- ✓ 방송 플랫폼 이용되는 프로그램을 분석하여 해당 기술이 어떻게 적용되는지 확인한다.
- ✓ 현재 사용되는 사물 인식 프로그램에 대해 분석하고 조사하여 프로토타입을 설계한다.
- ✓ 방송에서 주로 사용되는 사물이나 화면을 분석하여 데이터를 수집하고 이를 AI 모델에 학습하여 딥러닝 모델을 구축한다.
- ✓ 학습된 모델과 기존 사물/인물 인식 프로그램의 모델과 병합하고 기능에 맞는 최적화된 프로그램 개발한다.

5.1.2 제작 시험법

- ✓ 시작품 설계에서 제작된 프로토타입을 사용하여 카메라/웹 캠을 사용하여 물체 인식 및 사람 안면 인식에 대한 식별을 수치화하고 프로그램의 안정성을 테스트한다.
- ✓ 프로토타입 안정성 검사가 완료될 시 예외 상황, 부적절한 시도 시 어떤 오류가 있는지 요구 사항 테스트를 진행하여 예외 상황에 대한 검증을 실시한다.
- ✓ 프로토 타입 안정성 및 오류 검사가 완료된 후 실제 실시간 방송에서 어떻게 작동되는지 테스트하고 일정 기간을 통해 지속적으로 테스트하고 모니터링한다.

5.2 자체 평가 (독립적 평가)

평가 항목	평가 내용
사용성	응용 프로그램을 사용자가 쉽게 사용할 수 있는가?
실용성	응용 프로그램을 통해 사용자의 편리함이 개선되는가?
정확성	응용 프로그램이 정확하게 사람의 얼굴을 인식하고, 필터링할 수 있는가?
경제성	응용 프로그램으로 경제적 효과를 얻을 수 있는가?

[표 7] 자체 평가 표

사용성: 필터링할 영상을 입력하고 필터링된 영상을 출력받는 구조의 프로그램으로 설계하여 사용자는 쉽게 사용할 수 있을 것이라 예상된다.

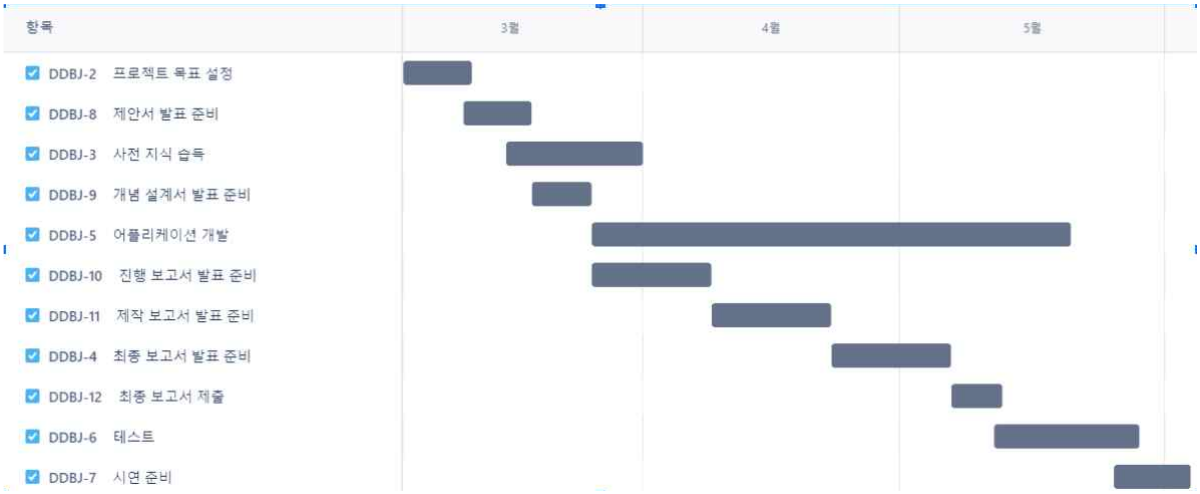
실용성: 실시간 야외 방송 콘텐츠를 살펴보면, 대부분의 방송인이 행인이 화면에 노출되지 않도록 신경 쓰는 모습을 보인다. 이러한 부분을 이 프로그램을 통해 개선할 수 있으리라 예상한다.

정확성: OpenCV의 Face Detection 시연 영상을 보았을 때, 얼굴을 반 정도 가리는 상황에서도 사람의 얼굴임을 인식하고 추적할 수 있는 모습을 보인다. 따라서 정면 모습이 아닌 옆모습의 사람도 정확하게 인식하고 필터링을 적용할 수 있을 것으로 예상된다.

경제성: 검열을 수작업으로 할 경우 많은 시간의 투자가 필요하다. 따라서 해당 프로그램을 사용할 경우 이러한 과정을 생략함으로써 자원을 절약할 수 있다.

6. 설계/프로젝트 계획

6.1 프로젝트 일정표



6.2 프로젝트 팀 업무

역할	이름	담당 업무
팀장	우승화	Front/Backend 개발, Project Management
팀원	석성훈	AI 개발(face recognition), Backend 개발
팀원	이훈근	Frontend(GUI), UI/UX 개발
팀원	임준원	AI 개발(object detection), AI Model 학습

[표 8] 팀원 업무분담표

7. 참고자료

OpenCV 공식 웹사이트
https://opencv.org/

DeepFace 공식 웹사이트
https://viso.ai/computer-vision/deepface/

PyQt 공식 웹사이트
https://www.qt.io/qt-for-python

wxPython 공식 웹사이트
https://www.wxpython.org/

Kivy 공식 웹사이트:
https://kivy.org/

파이썬 Opencv 라이브러리란 무엇일까
https://gr-st-dev.tistory.com/894

YOLOv8, ultralytics 홈페이지
https://docs.ultralytics.com/#where-to-start

face_recognition 깃허브 페이지
https://github.com/ageitgey/face_recognition/blob/master/README_Korean.md

PyTorch 홈페이지
https://pytorch.org/

Face Recognition 공식 문서
https://github.com/ageitgey/face_recognition/blob/master/README_Korean.md

Fine tuning 참고 문서, fiftyone
https://docs.voxel51.com/tutorials/yolov8.html

데이터 파이프라이닝, IBM
https://www.ibm.com/kr-ko/topics/data-pipeline

Layered Architecture, O'Reilly Media
https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html

MVC Architecture, mdn web docs
https://developer.mozilla.org/ko/docs/Glossary/MVC

Event driven Architecture, Microsoft

https://learn.microsoft.com/ko-kr/azure/architecture/guide/architecture-styles/event-driven

Service Oriented Architecture, IBM
https://www.ibm.com/kr-ko/topics/soa

<MVC패턴>Controller & Servicec & Repository란 무엇일까?
https://velog.io/@0andwild/MVC%ED%8C%A8%ED%84%B4Controller-Servicec-Repository%EB%9E%80-%EB%AC%B4%EC%97%87%EC%9D%BC%EA%B9%8CTIL.220526

8. 부록

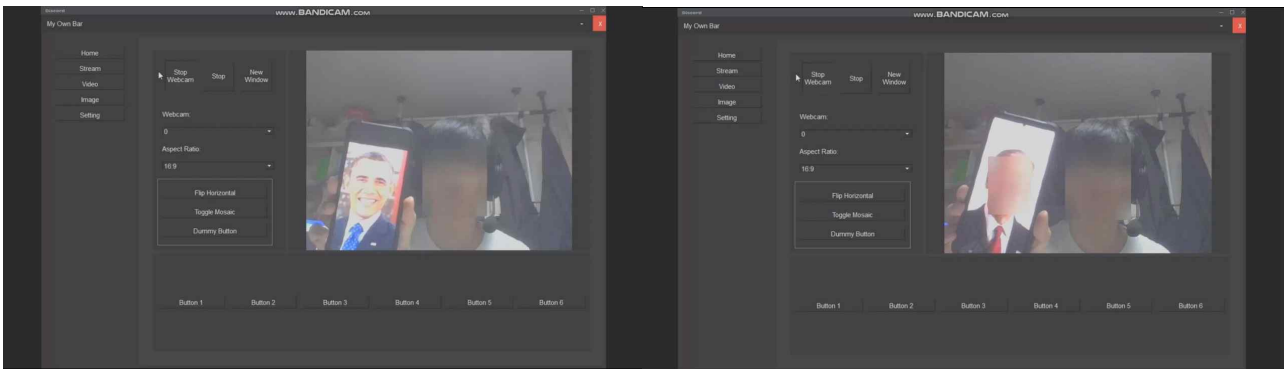
1. [CPU GPU 성능 비교]

각 CPU/GPU를 사용한 모델 성능 비교
동작 환경
CPU = i5-11500k | GPU = 1050ti

CPU 테스트	GPU 테스트
C:\Users\TlesMes\OneDrive\Desktop>python encoding_use_CPU.py Enter the file path: dataset load image file from path: dataset encoding dataset using CPU... Processing time: 27.374s	C:\Users\TlesMes\OneDrive\Desktop>python encoding_use_GPU.py Enter the file path: dataset load image file from path: dataset encoding dataset using GPU... Processing time: 0.182s
소요시간 = 27.374s	소요시간 = 0.182s

모델 성능 비교 결과 소요 시간 약 150배 성능 차이가 발생하였다.

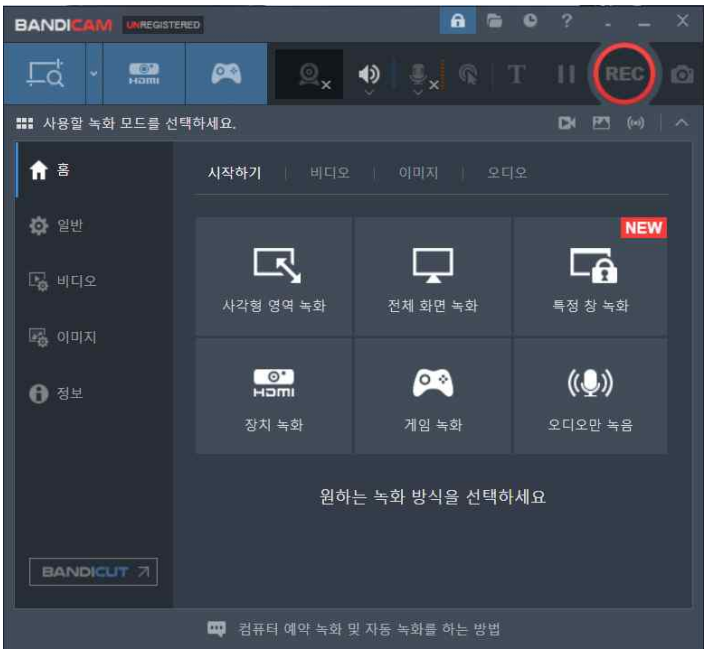
2. [얼굴 필터링 모델 테스트]



좌. 등록된 사진에 따른 얼굴을 인식하고 필터링 제외, **우** 등록되지 않는 사람에 대한 얼굴 필터링

3. [GUI 디자인 참고]

(1) 화면 녹화 프로그램 – 반디캠



(2) 방송 녹화 및 송출 프로그램 – OBS