프로젝트 개념설계보고서

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 개념설계보고서

학 과 : 컴퓨터공학과
과 목 : 캡스톤디자인
담당교수 : 김상귀 교수님
마 감 일 : 2024년 3월 31일
제 출 일 : 2024년 3월 31일

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 개념설계보고서

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 : 2024.03.18. ~ 2024.03.31.
문서버전 : V 1.0

<제목 차례>

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 인적 자원 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

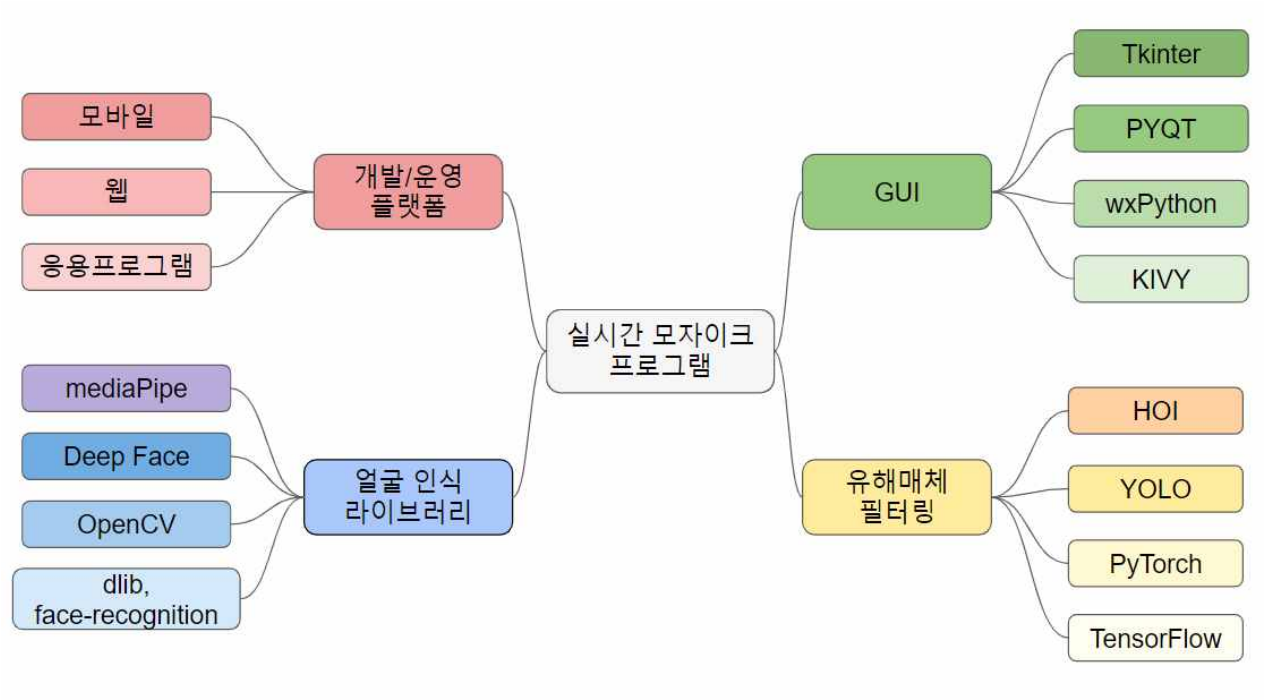
가려조

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. 아이디어 창출



실시간 모자이크 프로그램의 개발을 목표로 브레인스토밍을 진행하여 다양한 아이디어를 도출하였다. 이 과정에서 실시간 모자이크 프로그램의 핵심적인 부분을 파악하고 그에 필요한 주요 부분을 '개발/운영 플랫폼', 'GUI', '얼굴 인식 라이브러리', 그리고 '유해매체 필터링'으로 분류하였다.

개발/운영 플랫폼: 프로그램의 이용성과 접근성을 높이기 위해 모바일, 웹, 그리고 응용 프로그램을 대상으로 아이디어를 도출하였다. 이를 통해 사용자들이 다양한 플랫폼에서도 쉽게 이용할 수 있는 환경을 제공하고자 하였다.
얼굴 인식 라이브러리: 프로그램의 핵심 기능 중 하나로, mediaPipe, Deep Face, OpenCV, 그리고 dlib의 face-recognition과 같은 얼굴 인식 라이브러리를 활용하여 실시간으로 얼굴을 인식하고 모자이크 처리를 진행하는 기술을 연구하고 도입하려고 한다.
유해매체 필터링: 사용자에게 안전한 환경을 제공하기 위해 HOI, YOLO, PyTorch, Tensorflow와 같은 기술을 활용하여 유해매체나 콘텐츠를 실시간으로 탐지하고 차단하는 필터링 시스템을 구축하려고 하였다.
GUI: 사용자 인터페이스는 프로그램의 사용성과 사용자 친화성을 결정짓는 중요한 요소이다. 따라서 Tkinter, PyQt, wxPython, Kivy와 같은 GUI 라이브러리를 고려하여 사용자가 직관적으로 프로그램을 이용할 수 있는 인터페이스를 구성하려고 하였다.

이렇게 도출된 아이디어들을 깊게 조사하고, 각 파트의 기술들을 어떻게 효과적으로 결합하여 사용할지를 비교 분석하는 과정을 거쳐, 가장 효율적이고 안정적인 실시간 모자이크 프로그램의 아이디어를 선정하고자 한다.

3. 아이디어 다듬기

3.1 개발/운영 플랫폼

난이도: 소프트웨어 개발의 복잡성과 기술적 요구
성능: 작업 처리 능력으로, 더 빠르고 효율적인 작업 수행 정도
확장성: 시스템이나 소프트웨어의 크기나 기능을 확장할 수 있는 정도
접근성: 사용자가 소프트웨어에 쉽게 접근하고 사용할 수 있는 정도

판정기준 (가중치)	난이도 (30)	성능 (20)	확장성 (20)	접근성 (30)	총점 (100)	최종 순위
모바일	21	16	14	24	75	2
웹	18	14	16	27	75	2
응용프로그램	24	16	18	27	85	1

모바일 애플리케이션
난이도: 21 (모바일 개발 플랫폼은 점차 발전하고 있지만, 여전히 특정 플랫폼에 대한 개발이 다소 복잡할 수 있음)
성능: 16 (모바일 기기의 성능은 점차 향상되고 있지만, 여전히 제한적일 수 있음)
확장성: 14 (모바일 애플리케이션은 특정 플랫폼에 종속되기 때문에 플랫폼 간의 이식성이 낮을 수 있음)
접근성: 24 (모바일 애플리케이션은 휴대폰이나 태블릿과 같은 기기에서 쉽게 접근할 수 있으며, 다양한 사용자에게 접근성을 제공할 수 있음)

웹 애플리케이션
난이도: 18 (웹 개발은 다양한 기술과 표준을 사용하기 때문에 초기 학습 곡선이 있을 수 있음)
성능: 14 (웹 애플리케이션은 서버와의 통신을 필요로 하기 때문에 네트워크 지연이나 서버 부하 등의 이슈가 발생할 수 있음)
확장성: 16 (웹 애플리케이션은 다양한 플랫폼에서 실행될 수 있으며, 클라우드 기술과 같은 기술을 사용하여 확장성을 높일 수 있음)
접근성: 27 (웹 애플리케이션은 웹 브라우저를 통해 쉽게 접근할 수 있으며, 웹 표준과 접근성 지침을 따르면 다양한 사용자에게 접근성을 제공할 수 있음)

응용 프로그램
난이도: 24 (응용 프로그램 개발은 특정 플랫폼에 최적화되어 있으며, 다양한 기능과 성능을 요구하기 때문에 개발 과정이 복잡할 수 있음)
성능: 16 (응용 프로그램은 클라이언트의 로컬 환경에서 실행되기 때문에 빠르고 효율적인 작동이 기대됨)
확장성: 18 (응용 프로그램은 다양한 플랫폼에서 실행될 수 있으며, 모듈화된 아키텍처를 사용하여 쉽게 확장할 수 있음)
접근성: 27 (응용 프로그램은 사용자의 로컬 기기에 설치되어 있으므로 쉽게 접근할 수 있음)

3.2 얼굴 인식 라이브러리

난이도: 소프트웨어 개발의 복잡성과 기술적 요구
성능: 작업 처리 능력으로, 더 빠르고 효율적인 작업 수행 정도
확장성: 시스템이나 소프트웨어의 크기나 기능을 확장할 수 있는 정도
지원 및 커뮤니티: 지원 서비스 품질, 공식 문서 완성도, 업데이트 빈도, 학습 자료 품질을 기준

판정 기준 (가중치)	난이도 (30)	성능 (30)	확장성 (20)	지원 및 커뮤니티 (20)	총점 (100)	최종 순위
OpenCV	25	25	15	10	75	1
MediaPipe	20	20	15	15	70	3
DeepFace	22	15	15	10	72	2
Dlib	20	20	15	15	70	3

OpenCV

난이도: 25 (OpenCV는 비교적 배우기 쉽고 사용하기 쉬운 API를 제공하기 때문에 난이도가 낮음)
성능: 25 (다양한 하드웨어 플랫폼 지원과 GPU 가속을 지원하기 때문에 성능이 뛰어남)
확장성: 15 (다양한 모듈을 제공하나, 코드 구조가 복잡하여 확장 및 수정이 어려울 수 있음)
지원 및 커뮤니티: 10 (활발한 커뮤니티와 다양한 문서 및 튜토리얼이 제공되지만, 공식 지원이 부족하여 문제 해결에 어려움이 있을 수 있음)

MediaPipe

난이도: 20 (Python API를 제공하며 상대적으로 배우기 쉬움, C++ 코드 기반의 고급 기능 사용이 어려울 수 있음)
성능: 20 (최신 기술 기반으로 모바일 플랫폼도 지원하지만, 고성능 환경에서는 속도가 느릴 수 있음)
확장성: 15 (모듈 기반 구조로 확장 및 수정이 비교적 용이하나 기능이 제한적일 수 있음)
지원 및 커뮤니티: 15 (Google 공식 지원으로 다양한 문서 및 튜토리얼이 제공되지만, 커뮤니티 규모가 상대적으로 작아 문제 해결에 어려움이 있을 수 있음)

DeepFace

난이도: 22 (다양한 모델과 예제를 제공하지만, 모델 학습 및 사용 과정에 전문 지식이 필요함)
성능: 15 (높은 정확도를 제공하지만, 속도가 느리며 실시간 처리가 어려울 수 있음)
확장성: 15 (모델 학습 및 커스터마이징이 가능하지만, 학습 과정이 복잡하고 전문 지식이 필요함)
지원 및 커뮤니티: 10 (활발한 커뮤니티와 다양한 모델 및 예제가 제공되지만, 공식 지원이 부족하여 문제 해결에 어려움이 있을 수 있음)

Dlib

난이도: 20 (C++ 기반으로 코드를 직접 수정할 수 있지만, C++ 프로그래밍 지식이 필요하며 최적화 및 병렬 처리가 어려울 수 있음)
성능: 20 (가볍고 빠른 성능을 제공하지만, 높은 정확도를 위해 추가적인 최적화가 필요할 수 있음)
확장성: 15 (C++ 기반으로 확장 및 병렬 처리가 어려울 수 있음)
지원 및 커뮤니티: 15 (활발한 커뮤니티와 다양한 문서 및 튜토리얼이 제공되지만, 공식 지원이 부족하여 문제 해결에 어려움이 있을 수 있음)

3.3 유해매체 필터링

난이도: 소프트웨어 개발의 복잡성과 기술적 요구
성능: 작업 처리 능력으로, 더 빠르고 효율적인 작업 수행 정도
확장성: 시스템이나 소프트웨어의 크기나 기능을 확장할 수 있는 정도
지원 및 커뮤니티: 지원 서비스 품질, 공식 문서 완성도, 업데이트 빈도, 학습 자료 품질을 기준

판정기준 (가중치)	난이도 (30)	성능 (30)	확장성 (20)	지원 및 커뮤니티 (20)	총점 (100)	최종 순위
YOLO	24	30	14	14	82	1
PyTorch	21	24	18	18	81	2
TenserFlow	27	21	12	18	78	3

YOLO
난이도: 24 (YOLO는 사용자에게 비교적 쉽게 사용 가능한 딥러닝 기반의 객체 탐색 라이브러리이므로 난이도가 낮음)
성능: 30 (YOLO는 실시간 객체 탐지에 높은 성능을 제공하는 것으로 알려져 있음)
확장성: 14 (YOLO는 다양한 환경 및 데이터셋에 대해 잘 확장 가능하나, 사용자 정의 모델 및 알고리즘 추가에 제한적일 수 있음)
호환성: 14 (다양한 플랫폼 및 환경에서 YOLO를 실행할 수 있지만, 다른 라이브러리들과의 통합성에 대한 문제가 발생할 수 있음)

PyTorch
난이도: 21 (PyTorch는 사용자에게 유연하면서도 강력한 딥러닝 프레임워크를 제공하지만, TensorFlow보다는 사용이 복잡할 수 있음)
성능: 24 (PyTorch는 성능이 우수하지만, TensorFlow보다는 약간 떨어질 수 있음)
확장성: 18 (PyTorch는 확장성이 뛰어나고, 사용자가 쉽게 모델을 구축하고 확장할 수 있음)
호환성: 18 (다양한 플랫폼 및 환경에서 PyTorch를 실행할 수 있으며, 다른 라이브러리들과의 통합성도 높음)

TensorFlow
난이도: 27 (TensorFlow는 사용자에게 쉽게 사용 가능한 딥러닝 프레임워크이며, TensorFlow 2.x 버전부터는 사용이 더욱 간편해짐)
성능: 21 (TensorFlow는 성능이 매우 우수하지만, PyTorch나 YOLO보다는 미세하게 떨어질 수 있음)
확장성: 12 (TensorFlow는 PyTorch에 비해 확장성이 다소 제한적일 수 있음)
호환성: 18 (다양한 플랫폼 및 환경에서 TensorFlow를 실행할 수 있으며, 다른 라이브러리들과의 통합성도 높음)

3.4 GUI

난이도 : 배우기 쉽고 사용하기 간편한 정도
성능 : GUI 애플리케이션의 실행 속도 및 응답성
확장성 : 추가 기능 및 라이브러리와의 통합 용이성
호환성 : 다양한 운영체제 및 플랫폼에서 작동하는 정도

판정기준 (가중치)	난이도 (30)	성능 (30)	확장성 (20)	호환성 (20)	총점 (100)	최종 순위
Tkinter	25	22	11	20	68	4
PYQT	25	26	20	20	91	1
wxPython	22	26	20	20	88	2
KIVY	26	22	18	20	86	3

Tkinter

난이도: 25 (Tkinter는 학습하기 쉽고 기본적인 GUI 개발에 적합하여 난이도가 낮음)
성능: 22 (Tkinter는 간단한 GUI에 대한 성능은 충분하지만, 복잡한 기능에서는 제한적일 수 있음)
확장성: 11 (Tkinter는 기본적인 GUI 개발을 위한 모듈을 제공하나, 확장성이 상대적으로 낮음)
호환성: 20 (다양한 플랫폼과 버전에서 잘 동작하며 호환성이 좋음)

PyQt

난이도: 25 (PyQt는 학습 난이도가 상대적으로 높으나, 성능과 확장성이 뛰어나므로 높은 난이도를 가짐)
성능: 26 (C++로 구현되어 있어 높은 성능을 자랑하며, 다양한 기능을 제공하여 성능이 뛰어남)
확장성: 20 (복잡한 기능과 확장성에서 우수하며, 다양한 기능을 추가할 수 있음)
호환성: 20 (다양한 플랫폼과 버전에서 잘 동작하며 호환성이 좋음)

wxPython

난이도: 22 (wxPython은 학습 난이도가 상대적으로 높으나, 성능은 뛰어나므로 중간 수준의 난이도를 가짐)
성능: 26 (성능 면에서 PyQt와 유사하며, 다양한 기능을 제공하여 성능이 뛰어남)
확장성: 20 (확장성이 좋으며, 다양한 기능을 추가할 수 있음)
호환성: 20 (다양한 플랫폼과 버전에서 잘 동작하며 호환성이 좋음)

KIVY

난이도: 26 (KIVY는 OpenGL을 기반으로 하며, 그래픽 처리에 능하나, PyQt에 비해 학습 난이도가 높음)
성능: 22 (성능 면에서 PyQt와 동등하며, OpenGL을 기반으로 그래픽 처리에 특화되어 있음)
확장성: 18 (확장성이 좋으나, PyQt에 비해 다양한 복잡한 기능을 제공하지 못함)
호환성: 20 (다양한 플랫폼과 버전에서 잘 동작하며 호환성이 좋음)

4. 아이디어 최종선택

4.1 퓨 방식을 통한 아이디어 선택

아이디어 다듬기 과정에서 선정된 아이디어 중 높은 점수를 받은 아이디어를 뽑아 새로운 기준을 추가하여 각 아이디어를 종합적으로 퓨 방식을 사용하여 평가한다. 이 방식은 다양한 평가 기준과 가중치를 적용하여 아이디어의 전반적인 적합성을 정확하게 파악하고, 프로젝트의 성공 가능성을 높이기 위해 실시한다. 이를 통해 팀의 노력과 자원을 효율적으로 활용하는 아이디어를 선정한다.

4.1.1 얼굴 인식 라이브러리

	OpenCV	Deep face
기능성과 범용성	+	-
정확도와 신뢰성	-	+
실행속도와 리소스사용	+	-
개발과 유지보수의 용이성	+	-

기능성과 범용성: OpenCV는 광범위한 컴퓨터 비전 관련 기능을 제공하는 반면, DeepFace는 주로 얼굴 인식과 관련된 기능에 초점을 맞추고 있다.
정확도와 신뢰성: DeepFace는 특히 얼굴 인식 분야에서 높은 정확도를 제공하는 반면, OpenCV는 다양한 컴퓨터 비전 작업을 위한 범용 라이브러리이다. 얼굴 인식의 정확도가 프로젝트의 중요한 요소라면 DeepFace가 더 적합할 수 있다.
실행 속도와 리소스 사용: DeepFace는 일반적으로 더 깊은 신경망 모델을 사용하기 때문에 더 많은 계산 리소스와 시간을 요구할 수 있다. OpenCV는 효율적인 알고리즘을 많이 제공하여 상대적으로 빠른 실행 속도와 낮은 리소스 사용을 제공할 수 있다.
개발과 유지 보수의 용이성: OpenCV는 C++, Python 등 여러 언어로 사용할 수 있어 개발자에게 유연성을 제공한다. DeepFace는 Python 기반 라이브러리로, 특정 언어에 익숙한 개발팀에게 더 적합할 수 있다.

4.1.2 유해매체 필터링

	YOLO	PyTorch
적합도	+	-
학습곡선	+	-
모델 성능	-	+
활용범위	-	+
배포 및 운영	+	-

적합도: Pugh 매트릭스 평가 결과, YOLO가 PyTorch보다 총점에서 우수했다. YOLO는 객체 인식 프로젝트에 특히 적합하며, 빠른 속도가 요구되는 프로젝트에서 유리하다.
학습곡선: YOLO는 더 적합한 경우 학습곡선이 빠르고 간단하게 만들어질 수 있다. YOLO는 특히 초기 학습 단계에서 빠르게 결과를 얻을 수 있어, 학습 시간을 단축하는 데 도움을 준다.
모델 성능: PyTorch는 모델의 성능과 정확도 측면에서 더 뛰어나다. PyTorch는 다양한 딥 러닝 알고리즘과 모델을 구현하고 최적화하는 데 높은 자유도와 유연성을 제공한다.
활용범위: PyTorch는 다양한 딥 러닝 모델 개발 및 다양한 분야에 활용할 수 있다. PyTorch는 다양한 알고리즘과 구조를 지원하여 다양한 문제에 적용할 수 있다.
배포 및 운영: YOLO는 배포 및 운영이 간편하며, 빠른 속도가 중요한 프로젝트에 적합하다. YOLO는 실시간으로 객체 인식을 수행하는 데 최적화되어 있어, 실시간 처리가 필요한 응용 프로그램에서 높은 효율성을 보인다. 반면, PyTorch는 모델 관리 및 운영에 유연성이 필요하며, 다양한 환경과 플랫폼에서 모델을 배포하고 운영하는 데 더 적합하다.

4.1.3 GUI

	PYQT	wxPython	KIVY
기능 및 성능	+	-	-
개발 환경 및 지원	+	+	-
라이선스 및 배포	-	+	+
UI 디자인	+	-	+

기능 및 성능: PyQt는 Qt 프레임워크를 기반으로 하여 다양한 기능과 위젯을 제공하는 데 중점을 둡니다. wxPython은 다양한 플랫폼에서 작동하며 다양한 기능과 위젯을 제공하는 반면, Kivy는 멀티터치 제스처 지원으로 모바일 앱 개발에 특화되어 있다.
개발 환경 및 지원: PyQt와 wxPython은 활발한 개발 및 커뮤니티 지원을 받고 있다. PyQt는 상용 버전과 LGPL 버전을 제공하며 다양한 IDE 및 툴에서 사용할 수 있다. Kivy는 개발 환경 및 지원 측면에서는 약간의 부족함이 있지만 빠르게 발전하고 있다.
라이선스 및 배포: PyQt는 상용 버전과 LGPL 버전을 가지고 있어 상업적 사용 시 라이센스 비용이 발생할 수 있다. 반면, wxPython과 Kivy는 자유 라이센스를 갖고 있어 상업적 사용에 제약이 없다.
UI 디자인: PyQt는 Qt 프레임워크를 기반으로 하여 UI 디자인 및 커스터마이징 가능성이 크다. Kivy는 UI 디자인에 특화되어 있어 디자인 측면에서 장점이 있다.
모바일 앱 개발: Kivy는 멀티터치 제스처를 지원하는 특성이 있어 모바일 앱 개발에 적합합니다.

4.2 기능/부품의 설계 및 구현 계획

번호	기능/부품	계획 사항 (설계/구현/향후)	설계 난이도 (상/중/하)	구현 난이도 (상/중/하)
1	사용자가 원하지 않는 객체 혹은 인물을 모자이크 등으로 필터링해야 한다.	설계	중	상
2	필터링 대상 혹은 필터링 제외 대상에 대한 사진 등록을 통해 필터링을 관리할 수 있어야 한다.	설계	중	중
3	현재 연결된 웹캠 디바이스 중에서 필터링 작업을 수행할 디바이스를 선택할 수 있다	설계	중	하
4	사용자가 필터링 예외처리하고자 하는 인물 사진 파일을 프로그램에 입력하는 기능이 있어야 한다.	설계	중	하
5	필터링 된 웹캠 영상을 출력하는 기능이 있어야 한다.	설계	하	하
6	필터링할 영상 파일을 프로그램에 입력하는 기능이 있어야 한다.	설계	중	하
7	입력된 영상을 필터링하는 기능이 있어야 한다.	설계	중	중
8	웹캠과 같이 실시간으로 전송되는 영상을 필터링할 수 있어야 한다.	설계	중	상
9	사용자가 영상 필터링 항목을 설정하고 관리할 수 있어야 한다.	설계	중	중
10	입력된 영상에서 얼굴 및 물체 정보를 검출할 수 있어야 한다.	설계	중	중
11	검출된 대상에서 물체가 무엇인지 인식할 수 있어야 한다.	설계	중	상
12	검출된 대상에서 사람의 얼굴을 인식할 수 있어야 한다.	설계	중	상
13	실시간으로 인식된 객체가 필터링 대상인지 판별할 수 있어야 한다.	설계	중	상
14	UI를 통해 웹캠을 통한 실시간 필터링, 동영상 파일 필터링, 사진 필터링의 기능을 선택할 수 있어야 한다.	설계	중	하
15	사용자가 선택한 기능에 세부적인 설정을 관리하는 UI가 있어야 한다. 세부 설정으로는 필터링 항목 관리, 입력 파일 등록, 출력 방식(화면 출력, 파일 출력) 정하기 등이 있다.	설계	중	하

번호	기능/부품	계획 사항 (설계/구현/향후)	설계 난이도 (상/중/하)	구현 난이도 (상/중/하)
16	검출된 대상 간의 상호작용을 인식하여 유해 콘텐츠 여부를 판단할 수 있어야 한다.	향후	상	상
17	필터링 대상의 경우 별도의 사진 등록 없이 객체의 이름을 이용해 필터링 대상으로 관리할 수 있도록 한다.	설계	중	중
18	영상 필터링의 경우 시간이 오래 걸리면 로딩 화면을 표시한다.	설계	중	중
19	실시간 영상 필터링은 속도가 느려질 경우 프레임을 낮춰 필터링한 영상을 출력한다.	설계	중	중
20	별도의 환경 설정 없이 실행할 수 있도록 하는 Installer 구현해야 한다.	설계	중	중

5. 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