Team 1 Final Report (S)

주제: 이미지/영상 카툰화 필터

1. Outline of program
   이 프로그램은 이미지나 영상을 카툰화하는 프로그램으로, 입력된 이미지 또는 영상을 만화처럼 보이게 만드는 필터를 적용해서 출력한다. 해당 필터는 원본 이미지의 세부 디테일은 줄이고, 간단한 색상과 뚜렷한 윤곽선을 강조하여 실사 극화체적인 만화의 느낌을 부여한다.

   여기에 사람 인식 모델을 활용하여 사람을 인식하고 해당 부분만 카툰화를 하게 하여, 실사 사진에 사람을 그린 듯한 화면을 연출하는 것을 목표로 진행하였다. 또한 GUI를 구현하여 사용자가 쉽고 용이하게 이미지 카툰화를 수행할 수 있도록 하였다. 결과적으로 사용자는 이미지, 사전 준비된 영상, 그리고 실시간 카메라를 통한 촬영 영상을 GUI를 통해 변환하고 획득할 수 있다.

2. Design of program
   이 프로그램은 다음의 단계를 거쳐서 이미지를 카툰화한다.

   1) detect_people_and_generate_matrix() : 이미지에서 인간 감지하기
      - Model : torchvision의 Mask RNN - RestNet50
      - 처리할 이미지를 입력 받는다.
      - 딥러닝 모델 함수를 사용하여, 사람을 감지한다.
      - 감지한 사람을 이용하여, i번째 사람이 있는 image의 pixel을 i로 할당한 matrix를 return한다 (i >= 1)
      - ps. 이 딥러닝 모델을 사용하여, 우리가 적용할 기술을 이미지 전체에 적용하는 것이 아니라, 인간에게만 적용하여 영화 포스터, 인물 배경 사진 등에서 더욱 극적인 효과를 이끌어내어 사진을 세련되게 만들어 주었다.

   2) process_image_to_sketch() : pencil drawing 이미지 생성
      - 처리할 입력 이미지/영상을 입력받는다.
      - 사람의 영역이 표현되는 마스크를 입력받는다. i번째 사람에게 i로 할당되어 있다.
      - 공식을 통해 이미지/영상을 그레이스케일로 변환한다.
      - 가우시안 커널에 기반한 필터를 정의한다.
      - 반전된 그레이스케일 이미지/영상을 생성하고 가우시안 블러를 적용한다. 이때 커널 크기와 표준 편차를 입력받아야 한다.
      - 블러 처리된 이미지/영상을 다시 반전한다.
      - 위의 과정들을 조합하여 스케치를 계산하고 마스크된 영역에만 스케치를 적용해 최종 스케치 이미지/영상을 생성한다.
      - 스케치화된 이미지/영상을 반환한다.

   3) color_quantization() : simplified color 이미지 생성

앞선 스테이지에서 얻은 Human Detection 데이터와 원본 이미지를 받아서 각각의 사람에 대해 색상의 값을 K개로 제한하는 역할을 수행하는 단계이다. 상세한 과정은 아래와 같다.

[1] Human Detection Data에서 **Label**을 추출
Human Detection Data에서 0은 인간이 아닌 영역, 1이상부터 각 사람마다 Label이 순서대로 붙으므로 이 라벨이 몇까지 존재하는지 추출해서 unique_labels에 넣는다. 이때 0은 제외한다.

<여기서부터 **[2]~[3]**을 사람 수만큼 **iterate>**
[2] Kmeans알고리즘을 수행할 파라미터를 설정
[2 - 1] Human Detection Data에서 label이 i인 데이터만 선별하여 Kmeans알고리즘의 DataSet에 넣는다.
[2 - 2] 제한할 색상의 수와 Kmeans 반복횟수를 설정해주고 위와 같은 3개의 파라미터로 Kmeans알고리즘을 수행한다.
[3] Kmeans알고리즘을 수행하여 각 클러스터의 중심 좌표**(RGB)**와 각 **RGB**의 라벨을 리턴값으로 획득 **(RGB**값을 **3**차원 공간좌표로 해석하여 수행**)**
[3 - 1] 클러스터 좌표 Dataset 기준으로 랜덤 초기화
<여기서부터 반복횟수만큼 **iterate>**
[3 - 2] 파라미터로 넣은 DataSet에 대해 각각의 클러스터 좌표와의 거리를 계산해서 가장 가까운 클러스터의 인덱스를 각각의 DataSet에게 할당하여 해당 클러스터에 포함시킨다.
[3 - 3] 각각의 클러스터의 중심좌표(RGB)를 각 클러스터에 속한 point들의 좌표(RGB)의 평균으로 Update한다.

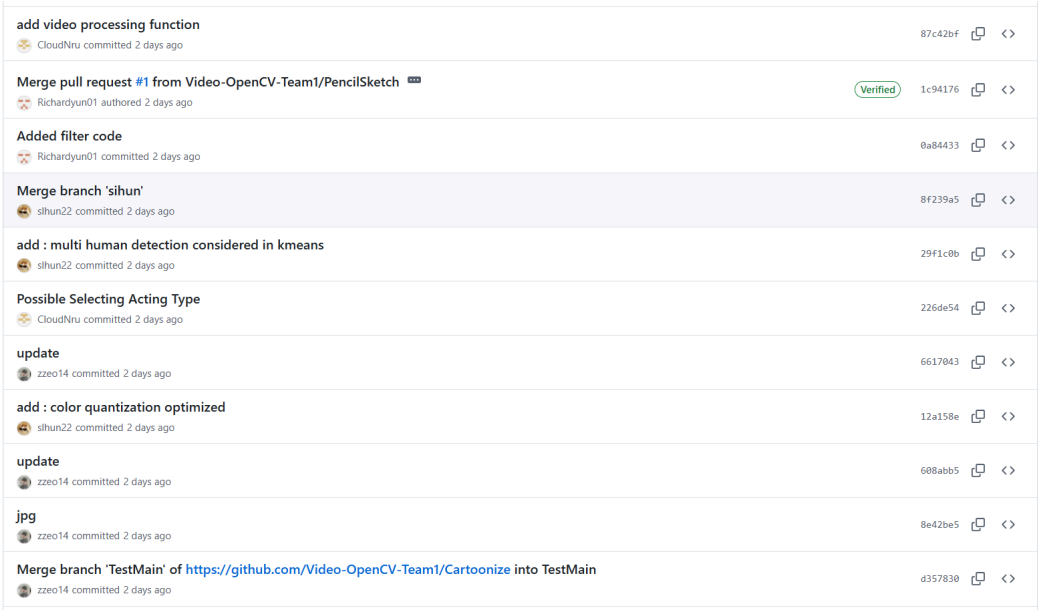
[4] RGB값마다 속해있는 클러스터의 중심 좌표**(RGB)**로 색을 변경 후**,** 결과 이미지를 리턴

이렇게 만들어진 2개의 이미지와 1개의 마스크를 이용하여, composite 작업을 실행한다.

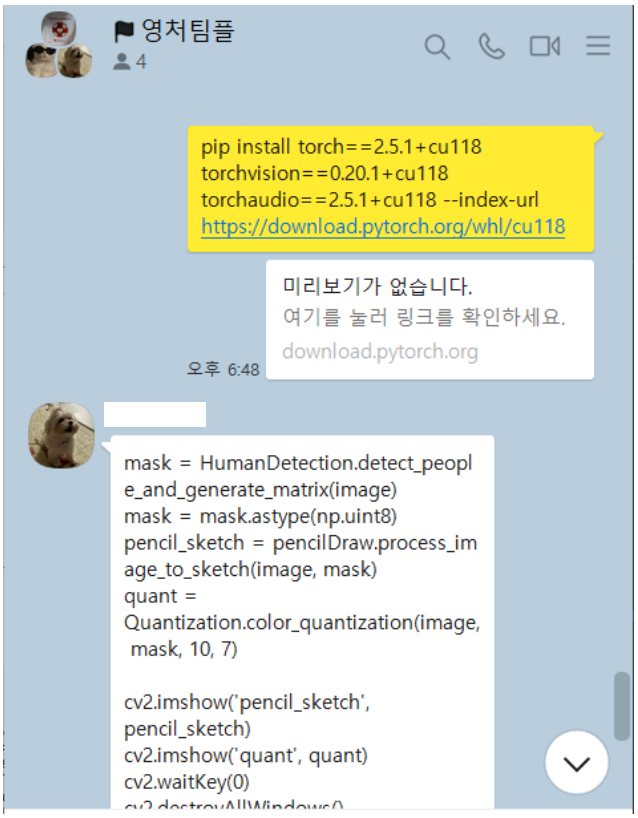
3. The project timeline
프로젝트를 본격적으로 시작하기 전 보고서와 중간 발표 PPT는 공동으로 제작하였으며 이를 위한 사전 조사 및 정보 검색 또한 해당 작업 중에 수행하였다. 그리고 프로젝트를 제작하기 위해서 우선 보고서와 발표 자료에 적힌 대로 작업해야 할 부분을 조사하였고 목록을 만들었다. 이를 토대로 각각 역할을 나누었고, 나누어진 역할을 각각 맡아 수행하였다.

결과적으로 중간 보고서 및 중간 발표 PPT의 제작에는 1일이 소요되었으며, 실제 프로그램 제작에는 약 2~3일이, 그리고 최종 보고서 및 최종 발표 PPT의 제작에도 1~2일이 소요되었다.

4. Design process of group discussion
   - GitHub 커밋 이력



- 카카오톡 대화 내용



5. Team's thoughts on this project

-

우선 기본적인 카툰화를 진행하는 것이 조금 창의성이 부족한 아이디어라고 생각했다. 그래서 보통 화면 전체를 CG처리 하거나, Green Screen 배경을 CG처리하는 경우가 많은데 우리 조에서는 조금 특이하게 '사람'만을 CG처리해보는 게 어떻겠냐고 나온 아이디어가 인상이 깊었다.
우리가 카툰 필터를 어떻게 진행하는지를 몰라서, 이 정보 저 정보를 찾아보며 여러 가지 방향성을 알게 되었다. 그런데 그중에서 K-Means

알고리즘을 사용하여 이미지의 색을 Clustering을 하고, 그것으로 사람을 단색화시키는 것이 애니메이션에서 나오는 캐릭터처럼 카툰화에 굉장히 잘 와닿았다. 마침 본 학기에 인공지능 수업을 수강하고 있었기 때문에 K-Means 알고리즘을 어느 정도 잘 구현할 수 있었으며, 좋은 결과를 얻을 수 있었다. 생각보다 결과가 만족스럽게 나와서, 친구들과 사진을 찍거나 어떤 그림을 그렸을 때 이 프로그램으로 친구들에게 좋은 사진을 남겨줄 수 있을 것 같다.

-

직접 카툰필터를 만들게 되면 퀄리티가 낮게 나올까봐 염려했었는데 Kmeans 알고리즘으로 색상을 제한하고 여기에 pencil sketching효과까지 합치니까 꽤 좋은 퀄리티의 결과가 나온 것 같아서 기분이 좋다. 특히 Kmeans알고리즘을 이미지 파일에 적용해서 색상 수를 제한한다는 것은 인공지능 강의 시간에 들어보기만 했던 내용이라 직접 구현해낼 수 있을지 걱정이 되었는데 잘 작동되어서 다행이었다.

-

OpenCV의 함수들을 사용하지 않고 직접 함수들을 손으로 구현하는 과정이 쉽지 않았으나, 각종 자료들을 참고하여 구현한 결과를 확인해보니 상당히 양호한 결과를 출력하는 것을 확인하였다. 이를 통해 마스크 영역, 그레이스케일링, 필터 등 각종 개념들을 이해할 수 있었고 이것이 실제로 잘 맞물려 정상적으로 동작하는 것을 확인하니 다행이라고 생각한다. 또한 이 프로젝트를 진행하면서 배운 지식들을 응용하면 더 다양한 종류의 필터를 구현할 수 있을 것이라고 생각한다.

-

실제로 작동 여부를 확인하기 위해 노트북 카메라로 촬영하는 실시간 영상으로 테스트를 진행하던 때가 많이 인상깊었다. 먼저 우리가 생각한 방식대로 진행했을 때, 예상보다 더 좋은 결과물이 나와서 많이 놀랐다. 걱정과는 다르게 잘 나온 결과물에 서로가 기뻐하는 장면이 인상이 깊었다. 한편으로는 그냥 카메라로 찍힌 영상을 출력하던 때와 다르게 모델과 각종 작업을 처리하면서 fps가 눈에 띌 정도로 줄어드는 장면을 볼 수 있었다. 기존에 보던 다른 카메라 앱을 이용할 때와 다르게 처리 속도에서 많은 아쉬움이 느꼈다. 좀 더 시간이 있었다면 최적화를 하며 더 빠른 처리 속도를 낼 수 있을 것이라고 생각한다.

6. Etc.
   - 프로그램 실행 방법: cmd나 git bash로 실행한다.
     1. python venv 가상환경에 접속한다.

```
python -m venv <가상환경이름> // 가상환경 생성
source <가상환경이름>/Scripts/activate
```

     2. 명령줄에서 cuda와 pytorch를 다운로드한다.

```
pip install torch==2.5.1+cu118
torchvision==0.20.1+cu118 torchaudio==2.5.1+cu118
--index-url
https://download.pytorch.org/whl/cu118
```

3. requirements.txt를 다운로드한다.

```
pip install -r requirements.txt
```

4. main.py가 있는 위치로 이동하여 코드를 실행한다.

```
python3 main.py
```

- 출력된 이미지나 영상은 해당 프로그램의 폴더에 저장된다.