

|  |
| --- |
| Image Processing |
| Team Project Report |

폰트, 그래픽, 시계, 로고이(가) 표시된 사진

자동 생성된 설명

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 제출일 | 2023. 12. 09 |  | 전공 | 소프트웨어학부 |
| 과목 | 영상처리 |  | 조 | Team 11 |
| 담당교수 | 최광남 교수님 |  | 이름 | 20201721 공현솔  20200525 김수현  20201665 민재연  20216950 신지원 |

목차

[1. 개발 동기 3](#_Toc152981321)

[2. 배경 지식 3](#_Toc152981322)

[3. 프로그램 설명 4](#_Toc152981323)

[4. 구현 4](#_Toc152981324)

[4.1 색상 필터 4](#_Toc152981325)

[4.2 윤곽선 검출: 색 구분 7](#_Toc152981326)

[4.3 색상 설명 8](#_Toc152981327)

[5. 최종 결과물 9](#_Toc152981328)

[6. 느낀점 11](#_Toc152981329)

# 개발 동기

수업 시간과 실습 시간에서 공부한 내용을 기반으로, 사용자에게 어떤 의미 있는 경험을 제공할 수 있을까 고민하던 중, “색약 안경”이 떠오르게 되었다. 색약자에게 도움을 줄 수 있는 색약 안경은, 일종의 색 필터를 안경에 씌워 사용자로 하여금 색을 잘 구별하고 받아들일 수 있게 도움을 준다. 하지만 이 안경은 구하기 힘들기도 하고, 개인의 색약 정도에 따라 맞추기도 쉽지 않다. 그렇기에 색약 안경과 같은 효과를 프로그램을 통해 개인에 맞추어, 이미지에 RGB값의 변화를 줌으로써 손쉽게 제공할 수 있을 것 같다는 생각이 들게 되었다.

색약자는 일상생활을 사는 데 큰 어려움은 없어도 색을 보는 부분에서 불편한 부분이 존재하기 때문에 이를 개선하고, 사진을 볼 때 더 편리하게 볼 수 있는 프로그램을 만들고자 한다.

# 배경 지식

두 가지 원추세포만 가지고 있는 경우를 색맹(dichromacy), 세 가지 원추세포가 모두 존재하지만 원추 세포의 기능이 정상적이지 못한 경우를 색약(anomalous trichromacy)이라고 한다. 전 세계 남성 인구의 약 8%, 여성 인구의 약 0.5%가 색각 이상자이다.

색맹은 적색맹(protanopia), 녹색맹(deuteranopia), 청색맹(tritanopia)로 나뉜다.

적녹생맥(가장 많은)은 스펙트럼의 중간 녹색이 무색이나 회색으로 보이고, 그보다 단파장측은 청색으로 장파장측은 황색으로 보이는 경우이다. 디스플레이 장치로부터 보여지는 색이 청색과 황색의 두 가지 색으로만 보이게 되고, 교통 신호 등을 잘 식별할 수 없다.

청색맹은 극히 드물며, 모든 것이 적색과 녹색의 두 가지 색으로 보이게 되며, 교통 신호등을 식별하는 것은 의외로 쉽다. 한편, 세 가지 원추 세포가 모두 없는 경우를 전색맹이라고 한다. 이 경우는 모든 색이 흑백 또는 회색으로만 보이기 때문에 시력이 매우 나쁘다.

색약은 적색약(protanomaly), 녹색약(deuteranomaly), 청색약(tritanomaly)으로 나뉜다.

적록색약은 적색과 녹색은 약간 볼 수 있으나, 그 정도가 적록색맹과 다름없을 정도로 심한 경우에서부터 아주 약한 경우까지 있다.

색맹은 근본적으로 원추세포가 존재하지 않기 때문에, 원추세포가 존재하지만 기능이 정상적이지 않아 구분에 어려움이 존재하는 색약자에게 특정 색을 더 강조하여 구분이 쉽도록 도와주는 것이 이번 과제의 목표라고 할 수 있다.

# 프로그램 설명

색약자를 위한 디지털 이미지 변환 프로그램으로, 색약 단계를 입력하거나 프로그램에서 간단한 테스트로 값을 설정하면 자신의 상태에 맞게 이미지를 변환해줄 수 있다.

* 색상 필터  
   색맹 유형에 따라 특정 색상을 강조하는 필터를 적용하여, 색약자가 더 색을 잘 판별할 수 있도록 도움을 준다.
* 색상 설명 및 식별  
   이미지의 특정 구간을 선택 시, 해당 색상의 RGB값과 함께 색상을 설명하여 색약 혹은 색맹 사용자가 특정 구간의 색 이름을 알 수 있게 한다.
* 윤곽선 검출  
   색약자들은 구분이 어려운 색끼리 붙어있을 시, 구분을 하지 못해 그림을 인식하는 데 어려움을 느낄 수 있기 때문에 색상을 기반으로 윤곽선을 검출해서 그림의 윤곽을 알 수 있게 해준다.

# 구현

언어는 Python을 사용했으며, 별도의 딥러닝 프레임워크를 사용하지 않고 OpenCV만을 이용하여 개발하였다.

색상 필터는 사용자의 색약 정도를 파악하고 그를 기반으로 진행해야 하기 때문에 OpenCV의 TrackBar 기능을 이용하여 사용자가 가장 색을 잘 판별할 수 있는 정도를 직접 입력한 후, 그 정보를 기반으로 변환을 수행한다.

색상 설명 및 식별은 이미지의 일부 부분에 대한 RGB값의 평균을 계산하여, 해당 구간이 평균적으로 어떤 색깔을 가지고 있는지에 대한 정보를 사전에 입력된 색의 분류 중 하나로 판별하여 사용자에게 알려준다.

윤곽선 검출은 각 색상 이미지에 sobel edge detection을 적용하여 구현하였다.

## 색상 필터

사용자에게 맞춤 필터를 제공하기 위한 기능이며, 트랙바를 구현하고, GUI를 구현하기 위해 PyQt5 라이브러리를 사용하였다.

Applyfilter는 색약자들이 구분하기 어려운 색들을 강조하여 다른 색과의 구분을 가능하게 하도록 한다. 사용자가 Applyfilter를 눌렀을 때 상단에는 RGB 각각의 값을 조절할 수 있는 트랙바가 있고 하단에는 녹색약, 적색약, 청색약 버튼을 제공하여 사용자가 본인에게 해당하는 버튼을 누르게 된다면 해당 색상이 강조가 되어 다른 색상과 구별하기 쉽도록 하였다. 색약 버튼 아래에는 초기화 버튼을 두었다.

* **initUI**

트랙바와 레이블을 나란히 배치하고, RGB 각각의 트랙바를 나타내었다. 사용자가 적색약인지, 녹색약인지, 청색약인지 구분하여 트랙바를 default 값으로 두고 이후 색조 조절을 사용자 스스로 조절하게 하기 위해 적색약, 녹색약, 청색약 각각의 버튼을 만들었고, 값 초기화를 위한 버튼을 만들었다.  
텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* **createSlider**

트랙바를 생성하는 함수로, 트랙바의 default 값을 0으로 초기화하고, -255부터 255까지 조절할 수 있도록 하였다. 색 조절 시 overhead 발생을 막기 위해 RGB 각각의 조절 가능 범위를 제한해두었다.  
텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* **sliderChanged**

createSlider에서 RGB 값 조절 가능 범위를 제한할 때 사용된다.

* **resetSliders**

초기화 버튼에 사용할 함수로 트랙바의 값을 모두 0으로 초기화 한다.

* **increaseRed, Green, Blue**

각각 RGB 값을 50씩 증가 시키는 버튼이다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* **updateImage**

트랙바로 조절한 RGB 값을 이미지에 적용하는 함수이다. frame에 받아온 image를 복사해서 frame의 값을 변경한다. r, g, b에 각각 트랙바의 값을 넣고, 트랙바로 조절한 r, g, b값이 0보다 크면 frame의 RGB값이 128보다 큰 픽셀에 트랙바의 값인 r, g, b를 frame의 RGB값에 더하여 넣어준다. 만약 r, g, b값이 0보다 작거나 같다면 frame의 RGB값이 128보다 작은 픽셀에 해당 r, g, b 값을 더하여 넣어준다. 또한, frame의 RGB값에 overflow가 발생하지 않도록 0~255값으로 제한을 둔다.  
텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

## 윤곽선 검출: 색 구분

색약을 가진 사람들은 특정 색을 구별하기 어려워한다. 이 점에서 착안하여 수업시간에 배운 ‘Sobel Edge Detection’을 사용하여 이미지에서 윤곽선을 추출해 서로 다른 색을 가지는 부분을 확인하고 구별 가능하게 했다. 윤곽선 추출을 함으로써 서로 다른 색을 가지는 부분이 어디인 지 확인할 수 있고, 추후 우리 프로그램에서 제공하는 다양한 기능(색상 설명 등)을 통해 이미지의 색상을 더 잘 이해할 수 있다.

* **calculate(color\_img)**

인자로 들어온 이미지에 대해서 sobel edge detection을 수행한다.

이미지를 순회하면서 각 픽셀에 대해 마스크의 크기와 같은 3x3 크기만큼 part image를 만들고, 해당 이미지와 소벨 마스크의 같은 위치에 있는 원소끼리 곱하는 아다마르 곱을 진행한다. 이 부분은 4중 for문으로 수행할 수도 있지만, 연산의 효율성을 높이기 위해서 numpy 라이브러리를 import해서 수행해주었다.  
텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

이렇게 얻어낸 x 방면의 결과값과 y 방면의 결과값을 절댓값을 취해 더해주고, 이 연산 결과를 저장한다.

모든 픽셀에 대해 연산을 마쳤다면, 이 값을 0 혹은 255로 normalize하기 위해 전체 픽셀의 연산 결과의 평균값을 기준으로 그것보다 크면 255, 그렇지 않다면 0으로 정규화 해주었다.  
텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* **erode**

검출된 윤곽선을 더욱 깔끔하게 볼 수 있도록 하기 위해, erode function을 사용했다. 이는 ‘침식’ 연산이라고도 하며, 이미지를 전체 순환하면서 각 픽셀에 대해 주변을 순회하며 단 하나라도 255의 값을 가지지 않는다면, 해당 위치의 픽셀 값을 0으로 설정하고, 모든 주변 픽셀이 255의 값을 가진다면 해당 위치의 픽셀 값을 0으로 설정하는 연산이다.

이를 통해 윤곽선은 더 얇고 뚜렷한 모습을 가질 수 있게 된다.  
텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

위 함수들을 기반으로, 메인 프로그램 흐름은 다음과 같다.

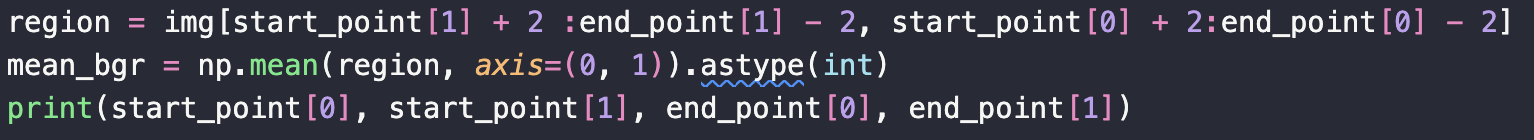
이미지를 red, green, blue의 색으로 분리하고, 각 색에 대해서 sobel edge detection을 수행한다. 이를 통해 다양한 색을 기반으로 윤곽선 추출 및 정규화가 진행된다. 이렇게 만들어진 이미지를 erode function을 통해 윤곽선을 정제한다. 그 후, 원본 이미지에 윤곽선의 위치를 표시하여 보여준다.

이를 통해 우리가 얻고자 하는, 색을 기반으로 추출된 윤곽선을 얻을 수 있다.

## 색상 설명

사용자에게 이미지의 특정 구간의 평균 색상이 무엇인 지 언어로써 설명해주는 기능을 제공한다. 사용자가 궁금한 부분을 마우스로 드래그 하여 영역을 선택하면, 그 부분의 RGB평균값을 내고 그것과 가장 거리가 짧은 색상을 도출해 냄으로써 이 기능을 구현했다.

마우스 콜백함수를 사용해서 사용자가 마우스로 드래그 하는 이벤트가 발생하면 이벤트의 종류, 마우스의 위치, 그리고 플래그를 매개변수로 받게 된다.  


사용자가 영역을 선택하면, 해당 영역 내부의 모든 픽셀 평균 RGB값을 numpy 라이브러리를 이용해 연산한다.  


위 과정을 통해 얻은 RGB값을 이용해 가장 비슷한 색상을 찾을 수 있다. RGB의 평균값과 실제 색상들의 RGB값의 차를 구하고, 이를 모든 색상 데이터에 대해서 진행하여 그 차가 가장 작은 색을 결과값으로 나오도록 한다  
텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

color\_map 에는 가장 비슷한 색을 찾을 때 높은 정확도를 갖도록 직접 329가지의 색을 넣어 두었다. 이 결과값을 바탕으로 사용자는 언어로써 해당 부분이 무슨 색인 지 짐작이 가능하며 느낄 수 있다.  
텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

# 최종 결과물

**메인 화면**  
텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**색상 설명**텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명 **텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명**

**색상 필터  
텍스트, 스크린샷, 소프트웨어, 컴퓨터이(가) 표시된 사진

자동 생성된 설명**

**윤곽선 검출  
텍스트, 컴퓨터, 운영 체제, 소프트웨어이(가) 표시된 사진

자동 생성된 설명**

**윤곽선 검출 + 색상 설명  
텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명**

# 느낀점

공현솔: 영상처리 수업 시간에 배웠던 것을 응용하여 우리가 만들고 싶은 프로그램을 직접 구현하고 눈으로 확인할 수 있어서 뜻깊은 시간이었습니다. 색약과 색맹은 우리가 직접 겪어보지 않은 상황과 환경이기 때문에 처음에 이것을 이렇게 구현 해도 옳은 것인가, 사용자에게 도움이 되는 것인가에 대한 고민이 많았고 팀원끼리 많은 의견을 나눴었는데, 나름의 해결책을 찾아낸 것 같아서 좋았고, 프로그램이 잘 실행되는 것을 보니 뿌듯했습니다. 기회가 된다면 여러 가지 방면으로 발전시켜보고 싶다는 생각이 들었습니다.

김수현: 팀플을 진행하며, 각자가 개발한 다양한 기능들을 통합하고, GUI를 통해 하나의 프로그램으로 만드는 과정에서 외적으로, 기능적으로 더 나은 방향을 고민하고, 발전시킬 수 있었습니다. 팀플 경험은 개인의 역량을 향상시키는 데에만 그치지 않고, 협업과 창의성을 바탕으로 한 더 나은 결과물을 창출하는 과정에서의 가치를 깨닫게 해주었습니다.

민재연: 이번 발표를 통해서 색약자들의 유형이나 그들이 겪는 어려움을 알게 되고, 불편을 조금이라도 해소 할 수 있도록 프로그램을 만들어 의미 깊은 시간이었던 것 같습니다.

신지원: 영상처리 실습에서 배운 내용을 응용하여 구현할 수 있게 되어 좋았습니다. 이렇게 팀 프로젝트로 opencv 라이브러리를 사용하면서 아이디어 구상부터 구현까지 어려운 점도 많았지만 좋은 경험이 된 거 같아서 좋았습니다. 특히 영상처리 수업에서는 사용하지 않았던 기능들을 파이썬으로 구현하면서 실력 향상에 도움이 된 거 같아서 좋았습니다.