# 멀티미디어시스템 REPORT

# Image Color Filtering

2020년 06월 26일

소프트웨어학과 32151671 박민혁

# 1. Project 개요

## *1.1 Project 개발 필요성*

- 이미지는 여러 색상의 집합인데 각 이미지들이 얼마나 많은 색상들로 이루었는지를 알고 싶었다. 육안으로는 정확한 색을 구분하는 데에 한계가 있다. 따라서 필터링을 통해 이미지 내사용 빈도가 높은 색상부터 나타내고, 더 나아가 하나의 색상만을 추출해 이미지를 재구성한다.

## *1.2 Project 개발 배경 또는 기존 기술의 현황*

- 인터넷에 이미지를 등록하면 색상을 추출해주는 사이트는 상당히 많다. 하지만 단순히 색상을 얻는 것으로 끝나는 것이 아니라 한 가지 색상을 그림에서 추출하여 원하는 색상을 자신이 선택하여 이미지를 재구성한다. 또한 수업시간에 배운 이론 수업이나, 일반적인 포토샵을 이용하여 이미지 필터링을 한 것이 아닌 MATLAB 프로그램을 이용한 이미지 필터링을 공부하여 사용을 했다.

## *1.3 문제점 및 개선 방안*

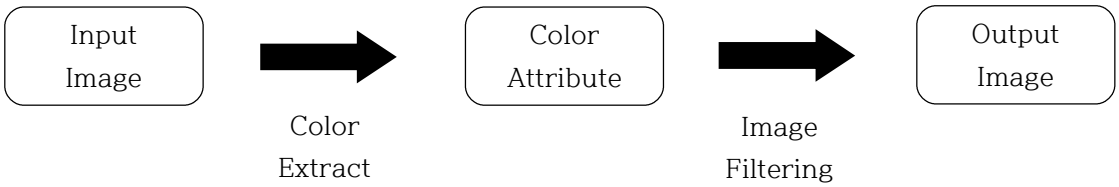
- 색상 범주를 조금 더 구체화할 필요가 있다. 현재 프로그램에서는 색상이 6가지로 나뉘어 있는데, 보다 정확한 색상을 추출하기 위해서는 더 많은 색상들로, 더욱 뚜렷한 범위를 사용해야 할 것이다. 또한 색상을 직접 넣는 방법을 선택했는데 이 부분에 대해서도 개선해야 될 것이다.

## *1.4 기대효과*

- 프로그램을 사용하면 이미지 파일에서 색상을 추출한다. 그리고 자신이 원하는 색상을 추출하여 새로운 이미지를 얻어낼 수 있게 된다. 그렇게 되면 더 나아가 생각 해 보면 일반적인 색상 추출에도 도움이 될 수 있으며, 색상 구분이 어려운 그림 같은 경우에도 활용 할 수 있다고 생각한다.

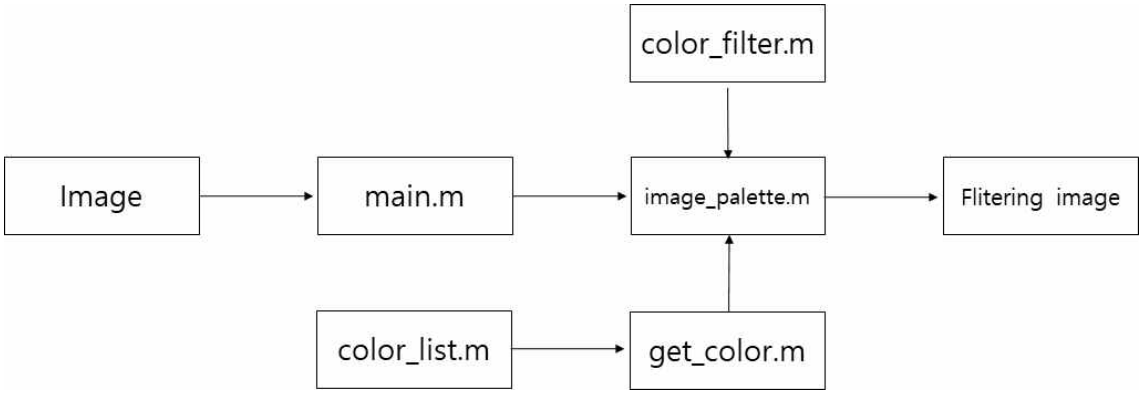
# 2. Project 수행 내용

## *2.1 기본적인 동작 및 기능*




## *2.2 알고리즘 설명*

1. 지정된 이미지 호출
2. 이미지 RGB 요소 LAB으로 변환
3. 이미지의 색상 구성 Confirm
4. 지정한 HSV 색상 영역만큼 필터링 된 이미지 Confirm

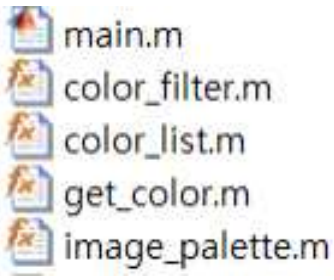



위의 블록 다이어그램을 설명하면, main에서 이미지를 읽어와 image_palette 함수에서 이미지를 구성하고 있는 색상 요소들을 구별한다. color_filter 함수에서는 이미지를 특정한 색상으로 필터링하여 재구성된 모습을 새로운 창에 표시해준다.

get_color 함수와, color_list 함수는 서로 상호 작용하여 추출된 색상을 RGB color list와 일치하는 색상의 이름을 보여주는 데 사용된다. 알고리즘의 구성이 단순하게 되어있기 때문에 처음 접하는 사람도 쉽게 이해가 가능하며 수정하는데 어려움 없을 것이다.

### *2.3 MATLAB 구현*

- 함수별 설명




**main.m** : 이미지를 읽어 image_palette.m 함수를 호출
**color_filter.m** : HSV 범위를 이용하여 특정 색상 필터.
**color_list.m** : RGB 표에 의한 색상 정보
**get_color.m** : color_list.m으로부터 색상 이름을 전달
**Image_palette.m** : main.m에서 이미지를 받아온 후 색상 추출

- 예상 출력 값



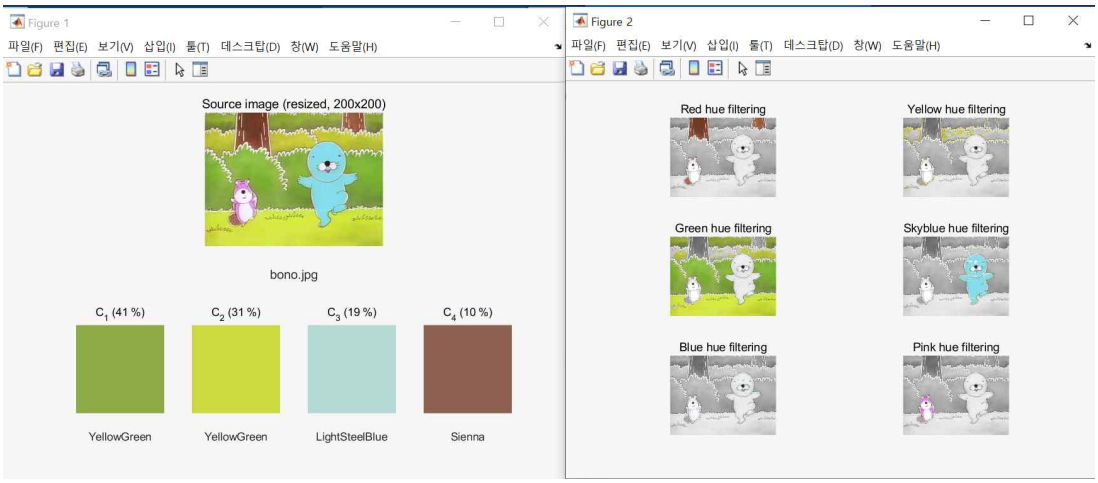
① 지정된 이미지의 컬러 구성에 맞는 색상들을 추출

② 미리 지정한 HSV의 H영역 범위를 이용하여 특정 색상을 제외한 나머지 색상을 필터링하여 이미지를 재구성한다. 하지만 HSV와 RGB는 색 구성 방법부터 다르기 때문에 양쪽 생상의 호환이 이슈가 될 것 같다.

### *2.4 입력 및 출력*

- 입력에 대한 결과물 설명

**① 그림과 실제 사진 비교**



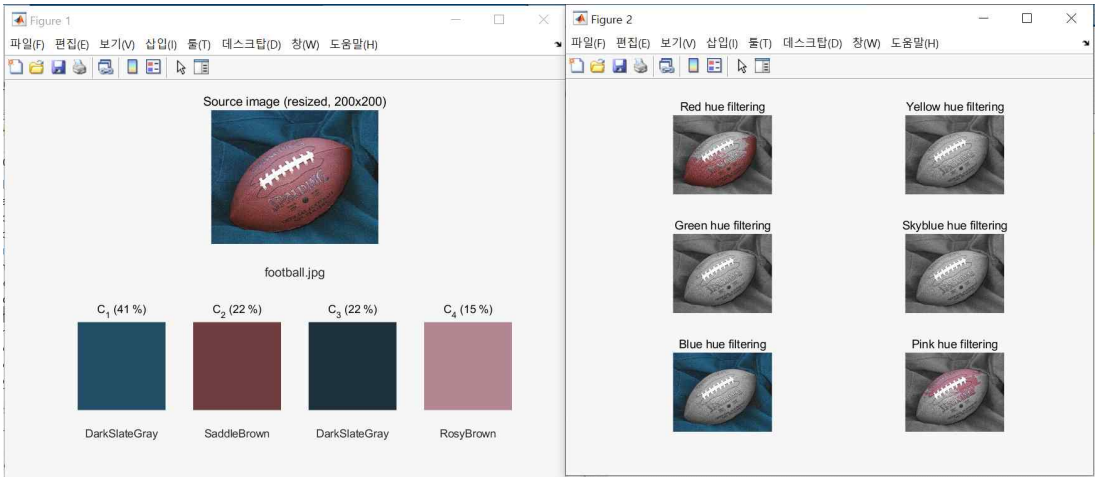


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
football.jpg
C1 (41 %)
C2 (22 %)
C3 (22 %)
C4 (15 %)
DarkSlateGray
SaddleBrown
DarkSlateGray
RosyBrown
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

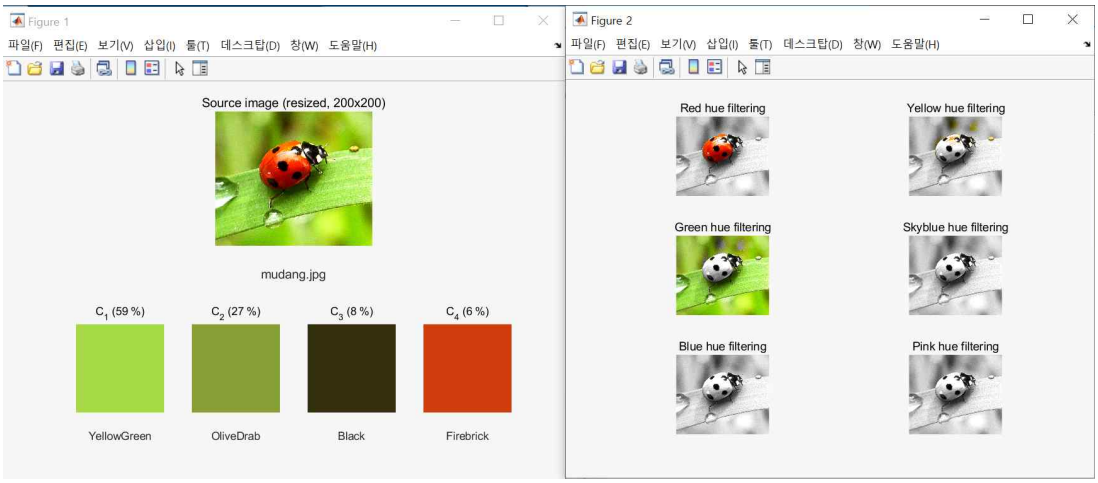


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
mudang.jpg
C1 (59 %)
C2 (27 %)
C3 (8 %)
C4 (6 %)
YellowGreen
OliveDrab
Black
Firebrick
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

## ② 색 구분이 뚜렷한 경우

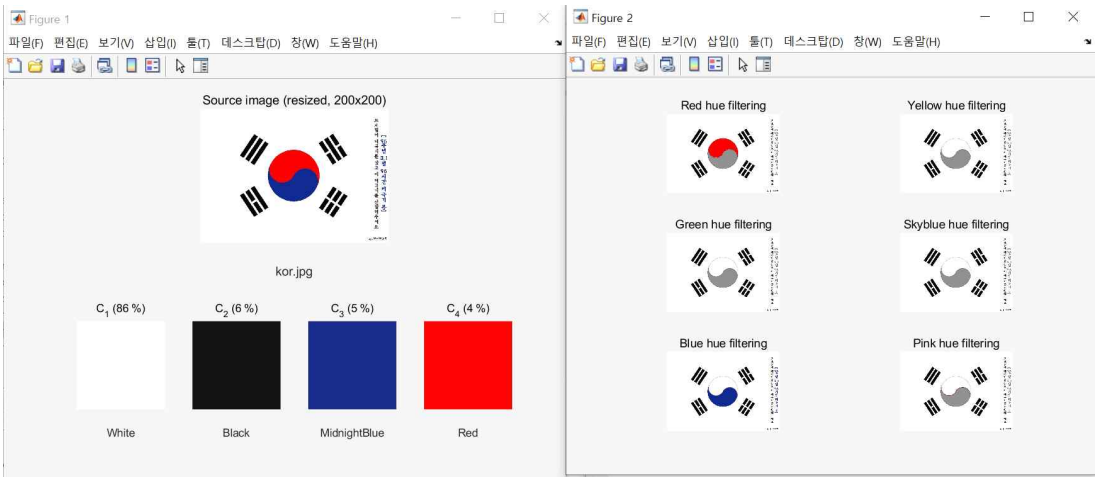


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
kor.jpg
C1 (86 %)
C2 (6 %)
C3 (5 %)
C4 (4 %)
White
Black
MidnightBlue
Red
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

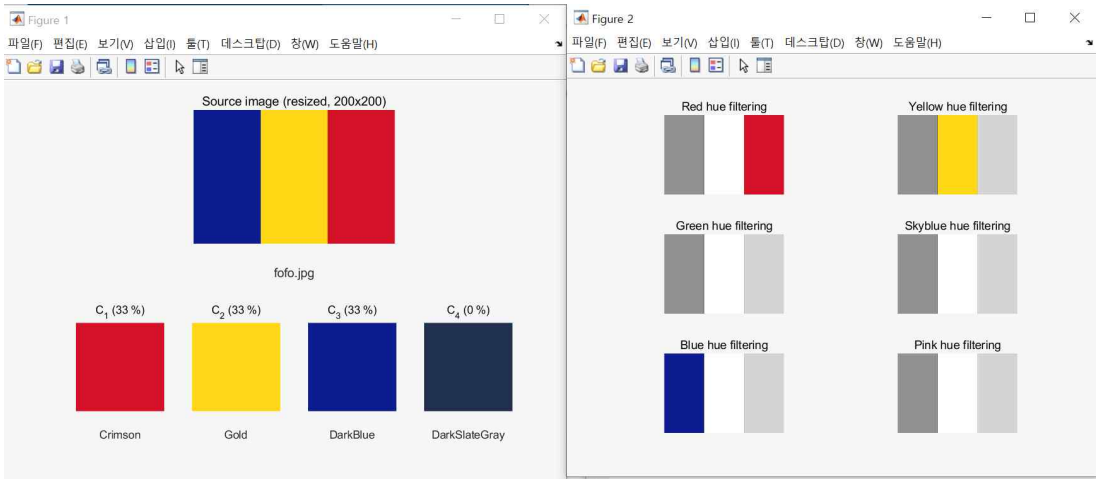


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
fofo.jpg
C1 (33 %)
C2 (33 %)
C3 (33 %)
C4 (0 %)
Crimson
Gold
DarkBlue
DarkSlateGray
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

## ③ 색 구분이 복잡한 경우

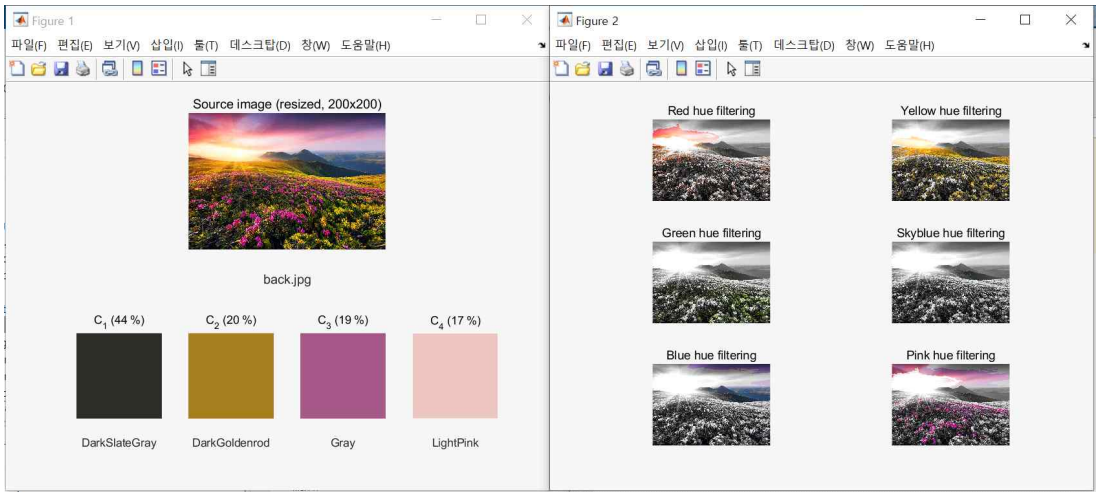


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
back.jpg
C1 (44 %)
C2 (20 %)
C3 (19 %)
C4 (17 %)
DarkSlateGray
DarkGoldenrod
Gray
LightPink
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

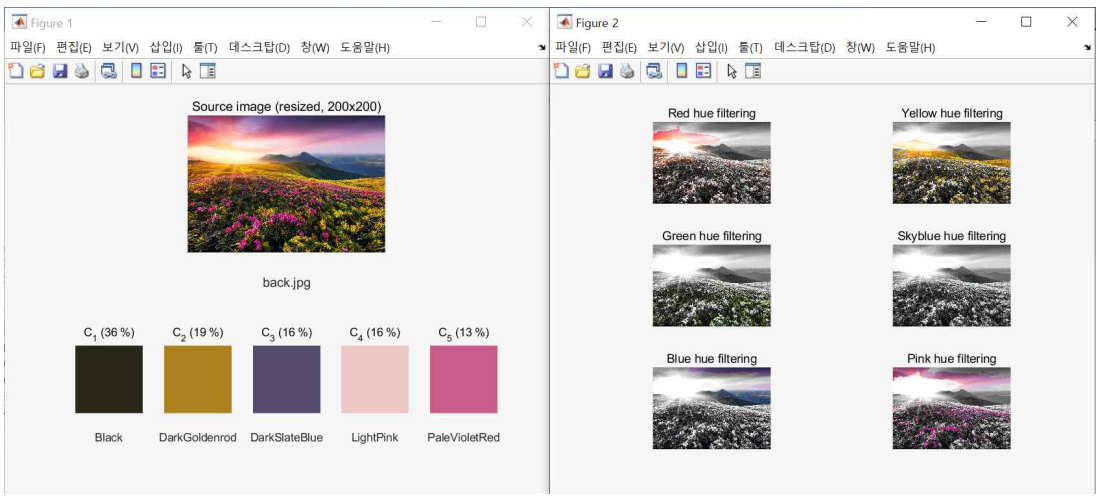


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
back.jpg
C1 (36 %)
C2 (19 %)
C3 (16 %)
C4 (16 %)
C5 (13 %)
Black
DarkGoldenrod
DarkSlateBlue
LightPink
PaleVioletRed
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

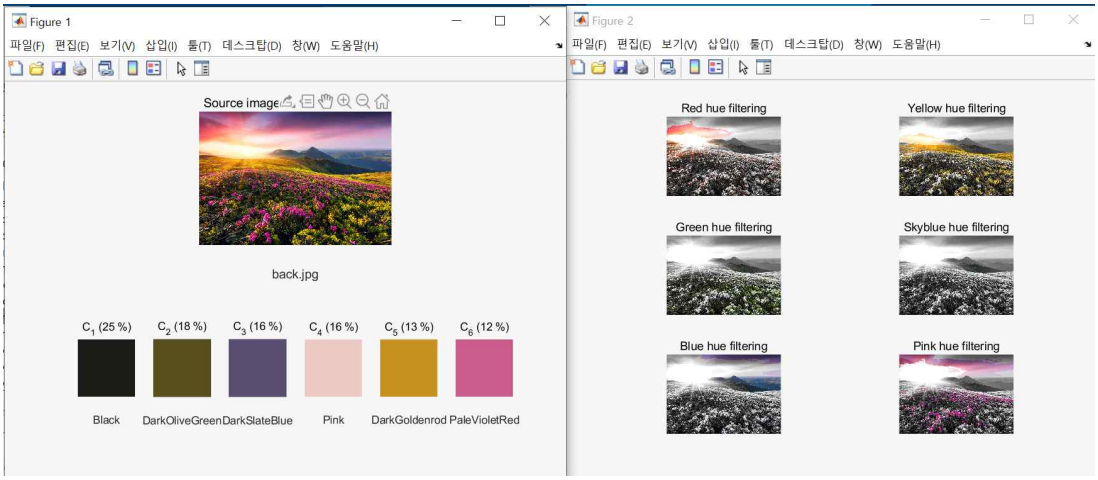


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image
back.jpg
C1 (25 %)
C2 (18 %)
C3 (16 %)
C4 (16 %)
C5 (13 %)
C6 (12 %)
Black
DarkOliveGreen
DarkSlateBlue
Pink
DarkGoldenrod
PaleVioletRed
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

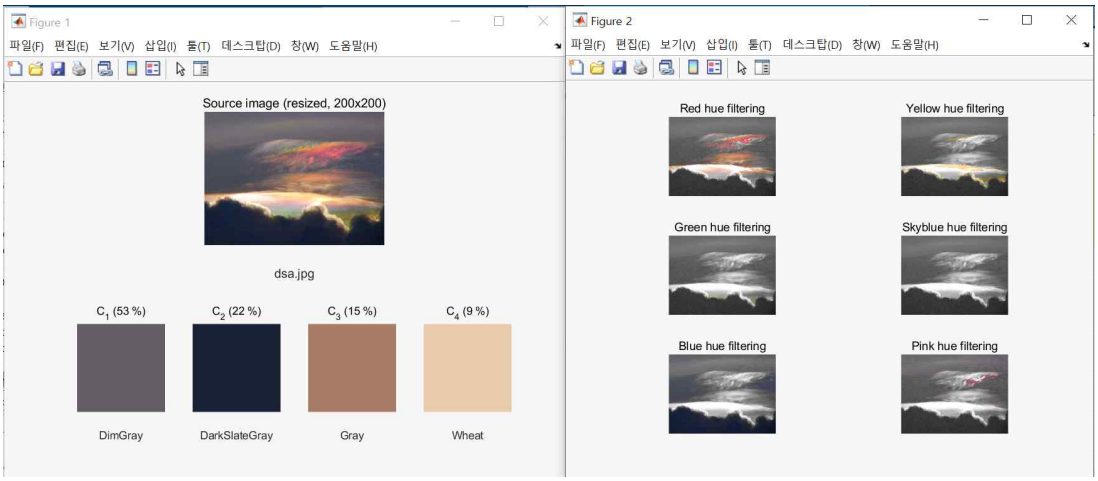


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
dsa.jpg
C1 (53 %)
C2 (22 %)
C3 (15 %)
C4 (9 %)
DimGray
DarkSlateGray
Gray
Wheat
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering

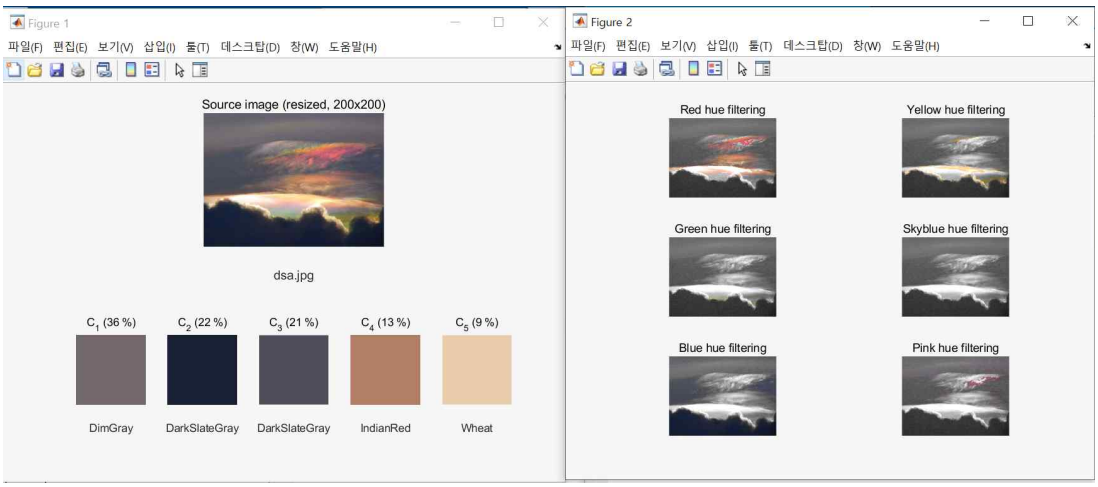
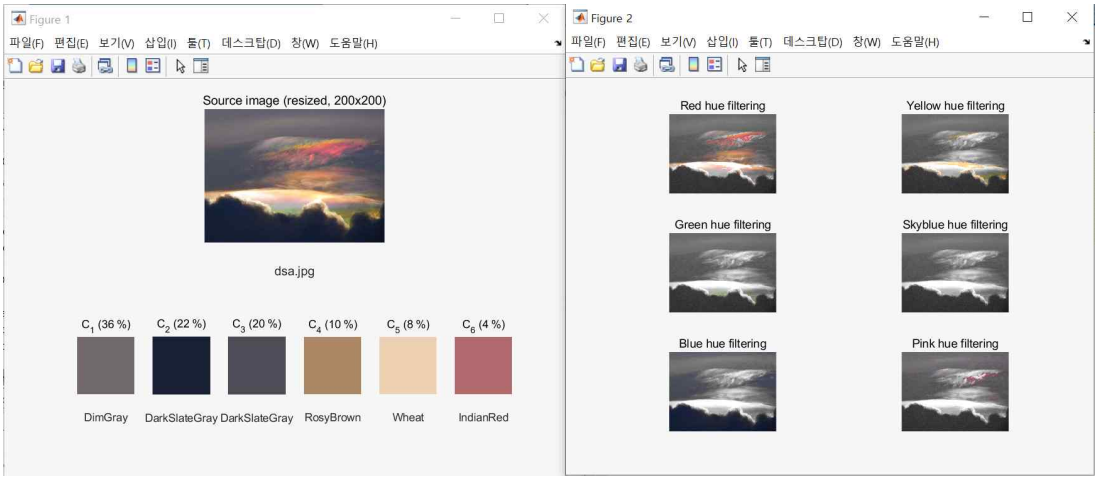


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Source image (resized, 200x200)
dsa.jpg
C1 (36 %)
C2 (22 %)
C3 (21 %)
C4 (13 %)
C5 (9 %)
DimGray
DarkSlateGray
DarkSlateGray
IndianRed
Wheat
Figure 2
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
Red hue filtering
Yellow hue filtering
Green hue filtering
Skyblue hue filtering
Blue hue filtering
Pink hue filtering




- 결과물을 비교하여 알고리즘 성능 분석

결과물은 위의 표에서 볼 수 있듯이 3가지 기준을 갖고 분류했다.

### ① 그림과 실제 사진을 비교한 경우

그림과 실제 사진을 이용해 필터링을 해본 결과 실제 사진에서는 결과가 잘 나오지만 그림에서는 결과 값이 제대로 나오지 않는 것을 볼 수 있다. 따라서 그림보다는 선명한 색감을 지닌 실제 사진을 이용해 필터링을 하는 것이 정확도가 높다는 것을 알 수 있었다.

### ② 색 구분이 뚜렷한 경우

색 구분이 뚜렷한 사진을 이용해 필터링을 해본 결과 국기처럼 색이 잘 구분되어 있는 사진일수록 정확도가 높았다. 두 번째 국기 사진의 경우 3가지 색을 사용했는데 결과 값으로 4가지의 색이 나왔다. 하지만 마지막 색상인 DarkSlateGray 성분이 0% 들어있다는 것을 알 수 있다. 따라서 cluster 수가 사진에 사용된 색상 수보다 많을 경우 임의의 값이 들어간다는 것을 알 수 있었다.

### ③ 여러 색들이 모여 있어 색 구분이 복잡한 경우

마지막으로 색 구분이 복잡할 경우의 사진을 이용해 필터링을 해보았다. 이러한 사진들은 색 구분이 어려워 색상을 성공적으로 분석하지 못할 것이라고 생각했다. 우선 cluster를 4개부터 시작해서 5개, 6개 점점 개수를 증가시키면서 비교했다. 4개보단 5개, 5개보단 6개의 cluster를 사용했을 때가 결과 값이 잘 나왔다. 따라서 색 구분이 복잡한 경우엔 cluster 수를 증가시킴에 따라 정확도가 증가하는 것을 볼 수 있다.

**세 가지를 종합해보면 색 구분이 뚜렷하고, 그림보다는 사진을 이용하며 cluster 수를 증가시킨다면 성능이 좋아질 것이다.**

*2.5 응용사례*

**① 사진 편집 애플리케이션**

사진 편집 애플리케이션은 대부분의 사람들이 사용해 보았을 것이다. 애플리케이션을 살펴보면 정말 다양한 편집 기능이 있다. 하지만 이 편집 기능들의 기초가 되는 것은 무엇일까? 바로 '이미지 컬러 필터링' 기술이다. 예전에 이런 기술이 많이 나오기 전에는 배경 색을 바꾸거나 자신이 원하는 색상만을 강조하여 사진을 편집하는 기술밖에 나오지 않았었다. 그래서 비인기 애플리케이션 이었으며 '이미지 컬러 필터링' 기술 또한 딱히 중요하지 않았었다. 하지만 현재는 이러한 기술들을 이용한 애플리케이션이 많이 나오고 있다. 그래서 중요하다고 생각 했고, 그 만큼 발전하여 예전과 다르다고 생각한다. 그래서 가장 기본적인 ;이미지 컬러 필터링; 기술의 발전은 중요하다고 생각한다. 또한 현재 사진 편집 애플리케이션은 대중적으로 인기 있는 애플리케이션이며 애플리케이션 시장 역시 경제성이 있다고 생각한다.

**② 옷 서칭**

사진 편집 애플리케이션과 같이 요즘 시대에 유행하고 있는 것이 있다면 바로 패션이라고 생각한다. 하지만 패션에는 관심이 있는 사람과 없는 사람이 명확히 갈리게 된다. 하지만 관심이 없는 사람들도 한 번 관심을 갖게 되면 못 빠져 나오는 것이 패션이라고 생각한다. 이러한 생각 때문인지 요즘에 자신에게 맞는 옷을 서칭 해주는 프로그램들이 많이 나오고 있다. 그래서 자신에게 맞는 옷, 또한 이미지 컬러 매칭을 통해 내가 입고 있는 바지와 어울리는 티셔츠 색을 골라주는 것, 그리고 사진을 찍을 때 배경과 맞는 옷을 찾아주는 서칭 프로그램들이 요즘에 서서히 발전하고 있다. 이것 또한 '이미지 컬러 필터링' 기술이다. 시대에 뒤처지지 않게 기술들이 발전하고 있으며 그 기본이 되는 것이 '이미지 컬러 필터링' 이라고 생각한다.

## 3. 본인의 제안 사항

*3.1 MATLAP Project 후기 및 아이디어*

a. MATLAP Project 후기

- 처음 기말 대체 과제가 나오고 주제를 선정 할 때에는 이론에 관한 Project를 진행하려고 했었다. 하지만 교수님이 수업시간에 좀 더 나아갈 수 있으면 MATLAB 프로그램을 활용해 보라고 하셔서 시험공부 대신에 MATLAB 공부를 하게 되었다. 하지만 난이도가 높고 프로그램 자체가 무거운 프로그램이라 좀 더 쉬운 난이도의 주제를 선정하였다.

분석한 코드를 간략하게 설명하면 다음과 같다. 우선 RGB로 되어있는 이미지를 lab으로 바꿔 저장을 한다. 그리고 K-means clustering을 하는데 이것은 대표적인 분리형 균집화 알고리즘이다. 쉽게 말해 각 개체들은 가장 가까운 중심에 할당되고, 같은 중심에 할당된 개체들이 모여 균집을 형성한다. 이를 통해 이미지를 이루고 있는 색상 성분들의 할당량을 %로 알 수 있었다. 그리고 색상을 추출하여 이미지를 재구성하는 코드는 다음과 같다. RGB로 되어있는 이미지를 hsv로 바꿔 특정 범위를 주면, 마스크를 씌워 범위 외의 색은 흑백처리 해주는 알고리즘인데, 6개의 범위를 적용했다. 범위를 더욱 넓게 설정하면 보다 정확한 결과를 도출 해낼 수 있었을 것 같다.

b. 아이디어

- 이러한 기술들을 통해 아이디어를 낼 수 있다. 내가 생각한 아이디어는 인테리어 시뮬레이션 프로그램이다.

예를 들어서 카페를 창업한다고 하자. 카페 뿐 만 아니라 어떤 가게를 창업 한다고 했을 때 인테리어에서 가장 중요하다고 생각하는 것은 조명이라고 생각한다. 요즘 인테리어를 보면 컨셉을 잡고 인테리어 하는 것이 유행하고 있다. 그래서 '이미지 컬러 필터링' 기술을 도입하여 시뮬레이션 프로그램을 돌리는 것이다. 가게 인테리어를 만들고 그에 맞는 조명을 찾는 것이다. 그렇게 되면 인테리어에 맞는 조명을 미리 확인하여 어울리는 조명을 설치해 창업의 성공률을 높이는 것이다.

실제, 나도 인테리어 하는 것을 좋아한다. 하지만 인테리어에 기본이 없다. 그래서 이것저것 많이 사보고 어울리는 제품만 두어 집을 인테리어를 했었다. 그러면서 느낀 점은 인테리어에서 가장 중요한 것은 조명이라 생각했다. 그리고 이것저것 많이 사보면서 잘못 샀다는 생각도 많이 했었다. 이러한 것을 방지하기 위해 미리 시뮬레이션을 돌릴 수 있다면 헛것을 사는 횟수도 줄어들 것이라 생각한다.

또한 요즘 취업이 힘든 경우들이 많다. 그래서 창업에 욕심이 생기는 사람들이 많아졌다. 그래서 이 시뮬레이션 프로그램은 더욱 더 활성화 될 것이라 생각한다.

### *3.2 Multimedia 후기*

- 기말 대체 과제 프로젝트가 주어졌을 때 별로 좋지 않았다. 왜냐하면 재수강이었고 시험에 더 자신이 있었다. 작년에 멀티미디어를 들었을 당시에는 학생회 일과 학점을 너무 꽉꽉 채워 들은 결과 좋지 않은 결과를 맞이했었다. 그래서 재수강인 만큼 더 열심히 공부를 했다. 하지만 코로나 사태로 인하여 시험은 취소되었고 대체 프로젝트 과제가 나왔다. 사실 나에게는 불리한 조건이라고 생각을 했다. 나는 두 번 배우는 과목이어서 시험에 자신이 있었고 공부 하는 데에 어려움이 없었는데 대체 과제가 나와 조금 당황스러웠다. 하지만 나름 열심히 하려고 준비한 것 같다.

MATLAB이라는 프로그램은 여태 다뤄왔던 프로그램에 비해 무거운 소프트웨어라고 생각한다. 일단 컴퓨터 사양도 좋지 않기에 애를 먹었었다. 하다가 저장이 안 되고 나가지는 경우도 있었다. 그럴 때 마다 MATLAB에 관한 프로젝트를 포기하고 싶었다. 하지만 완성을 시키고 싶었기에 계속 진행하였고 원하는 결과를 만들었다고 생각을 한다. 또한 프로젝트를 하면서 Mathwork라는 좋은 사이트도 알게 되었다. 그리고 코드를 참고하여 재구성 하는 것이어서 대강 프로젝트의 기반이 된 코드를 봤을 땐 코드가 어려울 것 같다고 생각이 들었었지만 Matlab 자체가 굉장히 직관적인 언어이고, 구현 되어있는 함수가 많다고 느낀 것이 어려울 것 같다고 생각한 대부분 코드가 읽으면 무슨 역할을 하는 코드인지 바로 해석이 되었다. 그리고 처음 보는 함수가 있으면 help라는 명령어를 통해서 무슨 역할을 하는 함수인지 바로 확인을 할 수 있었고 그래도 이해가 되지 않는 함수는 함수 사용 예제를 통해 학습 하면서 익힐 수 있었다.

그래서 빠르게 어떤 코드가 어떠한 역할을 하는지 파악이 가능했고 프로젝트에 어떤 코드가 필요 없는지 외부함수로 어떤 코드들을 옮겨야 하는지 쉽게 이해가 가능해서 진행할 수 있었다. 그리고 Matlab 자체에 구현되어 있는 함수가 다양하다고 느꼈고 C언어에 비해 쓰는 사람이 쉽게 배울만한 언어라고 생각하였다. 앞으로 다른 프로젝트 과제물을 할 때 유용하게 쓸 수 있을 것 같다.

## 4. Appendix (Project MATLAB source code)

main.m

```
%% 이미지를 읽어 image_palette 함수 호출
nCluster = 4;               % 추출하고 싶은 색 성분 개수
fname ='dsa.jpg';           % 이미지 파일

image_palette( nCluster, fname );
```

get_color.m

```
%% color_list.m으로부터 색상 이름 전달받음
function [matching_label, min_error] = get_color( input )

if size(input,1)>size(input,2)
    input=input';
end
[rgb, labels] = color_list();
input_rep = repmat( input, [size(rgb,1), 1]);
[min_error,matchIdx] = min( mean((input_rep-rgb).^2, 2) );
matching_label = labels{ matchIdx };
end
```

color_list.m

```
%% RGB 표에 의한 색상 정보
function [rgb, labels] = color_list()
labels = {...
    'Pink',
    'LightPink',
    'HotPink',
    'DeepPink',
    'PaleVioletRed',
    'MediumVioletRed',
    'LightSalmon',
    'Salmon',
    'DarkSalmon',
    'LightCoral',
    'IndianRed',
    'Crimson',
    'Firebrick',
    'DarkRed',
    'Red',
    'OrangeRed',
    'Tomato',
    'Coral',
    'DarkOrange',
    'Orange',
    'Yellow',
    'LightYellow',
    'LemonChiffon',
    'LightGoldenrodYellow',
    'PapayaWhip',
    'Moccasin',
    'PeachPuff',
    'PaleGoldenrod',
    'Khaki',
    'DarkKhaki',
    'Gold',
    'Cornsilk',
    'BlanchedAlmond',
    'Bisque',
    'NavajoWhite',
    'Wheat',
    'Burlywood',
    'Tan',
    'RosyBrown',
    'SandyBrown',
    'Goldenrod',
    'DarkGoldenrod',
    'Peru',
    'Chocolate',
    'SaddleBrown',
    'Sienna',
    'Brown',
```

```
                        'Turquoise',
'Maroon',               'MediumTurquoise',  'Fuchsia',
'DarkOliveGreen',       'DarkTurquoise',    'Magenta',
'Olive',                'LightSeaGreen',    'MediumOrchid',
'OliveDrab',            'CadetBlue',        'MediumPurple',
'YellowGreen',          'DarkCyan',         'BlueViolet',
'LimeGreen',            'Teal',             'DarkViolet',
'Lime',                 'LightSteelBlue',   'DarkOrchid',
'LawnGreen',            'PowderBlue',       'DarkMagenta',
'Chartreuse',           'LightBlue',        'Purple',
'GreenYellow',          'SkyBlue',          'Indigo',
'SpringGreen',          'LightSkyBlue',     'DarkSlateBlue',
'MediumSpringGreen',    'DeepSkyBlue',      'SlateBlue',
'LightGreen',           'DodgerBlue',       'MediumSlateBlue',
'PaleGreen',            'CornflowerBlue',   'White',
'DarkSeaGreen',         'SteelBlue',        'Snow',
'MediumAquamarine',     'RoyalBlue',        'Honeydew',
'MediumSeaGreen',       'Blue',             'MintCream',
'SeaGreen',             'MediumBlue',       'Azure',
'ForestGreen',          'DarkBlue',         'AliceBlue',
'Green',                'Navy',             'GhostWhite',
'DarkGreen',            'MidnightBlue',     'WhiteSmoke',
'Aqua',                 'Lavender',         'Seashell',
'Cyan',                 'Thistle',          'Beige',
'LightCyan',            'Plum',             'OldLace',
'PaleTurquoise',        'Violet',           'FloralWhite',
'Aquamarine',           'Orchid',           'Ivory',
```

```
    'AntiqueWhite'      220 20  60      210 180 140
    'Linen'             178 34  34      188 143 143
    'LavenderBlush'     139 0   0       244 164 96
    'MistyRose'         255 0   0       218 165 32
    'Gainsboro'         255 69  0       184 134 11
    'LightGray'         255 99  71      205 133 63
    'Silver'            255 127 80      210 105 30
    'DarkGray'          255 140 0       139 69  19
    'Gray'              255 165 0       160 82  45
    'DimGray'           255 255 0       165 42  42
    'LightSlateGray'    255 255 224     128 0   0
    'SlateGray'         255 250 205     85  107 47
    'DarkSlateGray'     250 250 210     128 128 0
    'Black'             255 239 213     107 142 35
    };                  255 228 181     154 205 50
rgb = [255  192 203     255 218 185     50  205 50
    255 182 193         238 232 170     0   255 0
    255 105 180         240 230 140     124 252 0
    255 20  147         189 183 107     127 255 0
    219 112 147         255 215 0       173 255 47
    199 21  133         255 248 220     0   255 127
    255 160 122         255 235 205     0   250 154
    250 128 114         255 228 196     144 238 144
    233 150 122         255 222 173     152 251 152
    240 128 128         245 222 179     143 188 143
    205 92  92          222 184 135     102 205 170
```

```
60  179 113
46  139 87
34  139 34
0   128 0
0   100 0
0   255 255
0   255 255
224 255 255
175 238 238
127 255 212
64  224 208
72  209 204
0   206 209
32  178 170
95  158 160
0   139 139
0   128 128
176 196 222
176 224 230
173 216 230
135 206 235
135 206 250
0   191 255
30  144 255
100 149 237
70  130 180
65  105 225
0   0   255
0   0   205
0   0   139
0   0   128
25  25  112
230 230 250
216 191 216
221 160 221
238 130 238
218 112 214
255 0   255
255 0   255
186 85  211
147 112 219
138 43  226
148 0   211
153 50  204
139 0   139
128 0   128
75  0   130
72  61  139
106 90  205
123 104 238
    245 245 220
    253 245 230
    255 250 240
    255 255 240
    250 235 215
    250 240 230
    255 240 245
    255 228 225
    220 220 220
    211 211 211
    192 192 193
    169 169 169
    128 128 128
    105 105 105
    119 136 153
    112 128 144
    47  79  79
    0   0   0
    ];
end
```

image_palette.m

```
%% main.m에서 받은 이미지를 바탕으로 색상 추출
function image_palette(nCluster, fname, plotOption)

  % 함수 인자가 3개 미만이면 plotOption 값 true
if nargin<3, plotOption = true; end
sz = [200 200];                          % 사진 크기 초기화

%% Load image file
img = rgb2lab(imread(fname));          % 이미지 파일 읽어오기
img = single(img);
L = img(:,:,1);                          % 이미지의 lab 값을 각각 변수에 저장
a = img(:,:,2);
b = img(:,:,3);

%% Visualize image
if plotOption                            % 불러온 이미지 표시
    figure(1);
    subplot(2,1,1);
    imshow( lab2rgb( img ) );
    title(['Source image (resized, ' num2str(sz(1)) 'x' num2str(sz(2)) ')'])
    xlabel(fname);
end

%% Parse and visualize Lab values
img_parse = [];
img_parse(1,:) = L(:);
img_parse(2,:) = a(:);
img_parse(3,:) = b(:);
```

```
%% K-means Clustering
[G] = kmeans( img_parse', nCluster);    % k-평균 군집화 수행
p = [];
for clusterIdx = 1:nCluster                    % 퍼센트 할당
    p(clusterIdx) = 100*length(find(G==clusterIdx)) / length(G);
end
[~,order]=sortrows(p'); order=order(end:-1:1);

% Palette visualization
for clusterIdx = 1:nCluster                    % 값이 작은 것부터 정렬해서 저장
    id = G==order(clusterIdx);
    LAB = ( img_parse(:, id) );
    mean_LAB = reshape( nanmean(LAB,2)', [1, 1, 3]);    % LAB 색상 저장

     % lab 색상을 rgb 색상표에 따라 이름 부여
    [color_label] = get_color(255*lab2rgb(mean(LAB,2)'));
    if plotOption
        subplot(2,nCluster,nCluster+clusterIdx);
        % LAB색상을 rgb 색상으로 변환후 표시
        imshow( lab2rgb( mean_LAB ) );
        title(['C_{' num2str(clusterIdx) '}
(' num2str(round(100*p(order(clusterIdx))/100)) ' %)' ]);
        xlabel(color_label);
    end
end
```

```
%% 6가지 색상별로 추출하여 이미지 재구성
if plotOption
    figure(2)
            I = imread(fname);
            I = im2double(I);
            I_m = color_filter(I,[350 30]);
            subplot(321)
            imshow(I_m,[]);
            title('Red hue filtering');
            I_m2 = color_filter(I, [31 60]);
            subplot(322)
            imshow(I_m2,[]);
            title('Yellow hue filtering');
            I_m3 = color_filter(I, [61, 130]);
            subplot(323)
            imshow(I_m3,[]);
            title('Green hue filtering');
            I_m4 = color_filter(I, [131 190]);
            subplot(324)
            imshow(I_m4,[]);
            title('Skyblue hue filtering');
            I_m5 = color_filter(I, [191 280]);
            subplot(325)
            imshow(I_m5,[]);
            title('Blue hue filtering');
            I_m6 = color_filter(I, [281 349]);
            subplot(326)
            imshow(I_m6,[]);
            title('Pink hue filtering');
end
end
```