SINCE 1947
SEO KYEONG
UNIVERSITY

[영상처리 1차 레포트]

과목명	디지털영상처리
담당 교수님	김진헌 교수님

전공	컴퓨터공학과
학번	2018305068
이름	정동기

제출일	2022.09.21

미션 소개

cv.imread() 함수로 읽어 낸 3채널 영상을 칼라와 모노로 matplotlib 화면에 2x2 서브창의 4개 화면에 출력한다. 여기서 화면은 interactive 모드로 작동해야 하며 화면에 마우스를 클릭하거나 키보드를 누를 때마다 다음 루틴으로 넘어간다.
1) 서브 화면 1: 원본 컬러 영상(title = “Original Color Image”
2) 서브 화면 2 : 모노 영상(title = “Mono Image”)
3) 서브 화면 3 : 원본 컬러 영상에서 각 컬러 채널에 1.5를 곱하여 밝게 만든 영상(title = “Original * 1.5)
4) 서브 화면 4: 원본 컬러 영상에서 R 컬러 채널만 2로 곱하여 Red 색상만 강화된 영상(Original[..., 0] = 2)

미션 구현 방법

1. 이미지를 opencv 에서 받을 때에는 RGB 의 순서가 BGR 로 바뀌어 들어오게 된다. 우리는 이번 미션 과제에서 matplotlib 를 이용하여 출력을 해야되기 때문에 이 부분부터 바꾸어 주어야 된다.
2. 이미지를 바꾸어 주었다면 interactive 모드로 설정을 해준다.
3. 빈 화면에 타이틀과 서브 타이틀을 출력한 후 사용자가 입력을 하면 다음으로 넘어가게 만들어 준다.
4. 서브 화면 (1,1) 화면에는 원본 영상이 올라가므로 앞서 변환해준 영상을 imshow()함수를 이용하여 출력해준다. 이때 아래에서도 마찬가지이겠지만 축과 라벨을 끄고 타이틀을 위에서 정해준 값으로 출력하게 해준다. 이 후 출력을 다 하였으면 사용자의 입력을 받아야 넘어갈 수 있도록 해준다.
5. 서브 화면 (1,2) 화면에는 모노 영상이 올라가므로 cv.cvtColor 함수를 이용하여 3채널 영상을 1채널 영상으로 변환을 해준다. 후에 출력 시 cmap 함수를 이용하여 단일 채널 이미지를 gray 채널에 매핑해준다. 앞서 했던 것과 마찬가지로 축과 라벨을 끄고 타이틀을 위에서 정해준 값으로 출력 후 사용자의 입력을 받아야 넘어가게 해준다.
6. 서브화면 (2,1) 화면에는 원본 컬러 영상에서 RGB 순서를 바꾸어준 이미지의 정규화를 먼저 해준다. 정규화해준 이미지에 1.5를 곱하게 되면 RGB 값에 각각 1.5가 곱해진다. 이후 imshow()가 자동적으로 clipping 을 해주기 때문에 이를 이용하여 출력을 해준다. 앞과 마찬가지로 축과 라벨을 끄고 타이틀을 출력 후 사용자의 입력을 받아야 넘어가게 해준다.
7. 서브화면 (2,2) 화면에는 R 컬러 채널만 2를 곱하여 주는 것이기 때문에 RGB 순서를 바꾼 이미지의 R 값[...,0] 값만 2를 곱하여 준다. 후에 imshow()를 이용하여 clipping 을 해주고 앞서 출력한 것과 같이 출력을 해주면 된다.

코드

```python
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt

Path = '../data/'
Name = 'RGBColors.JPG'
FullName = Path + Name

image = cv.imread(FullName)     # read as 3 channel. default.
assert image is not None, f"No image file: [{FullName}]....!"  # 입력 영상을 제대로 읽어오지 못하여 NULL을 반환.

imgRGB = image[..., ::-1]   # = image[:, :, ::-1]. 같은 표현
plt.ion()       # interactive mode로 설정

plt.figure(num='Report1: Original image & its processed images')
plt.suptitle("Report1: Original image & its processed images", fontsize=10, fontweight='bold')
plt.waitforbuttonpress() # 사용자가 입력학면 true

# 1) 서브 화면 1 - 원본 컬러 영상, title="Original Color Image"
plt.subplot(221) #row = 2, col = 2, index = 1 (1,1 위치)
plt.imshow(imgRGB) # 위에서 변환해준 이미지를 보여준다.
plt.axis('off')             # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title('Original Color Image')
plt.waitforbuttonpress()        # 키 혹은 버튼 입력을 기다린다.

# 2) 서브 화면 2 - 모노 영상, title="Mono Image"
plt.subplot(222)
imgM = cv.cvtColor(image, cv.COLOR_BGR2GRAY)    # 3채널 영상을 1채널 gray로 변환
plt.imshow(imgM, cmap='gray') #단일 채널 이미지를 gray 생상 채널에 매핑해준다.
plt.axis('off')             # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title("Mono Image")
plt.waitforbuttonpress()        # 키 혹은 버튼 입력을 기다린다.

# 3) 서브 화면 3 - 원본 컬러 영상에서 각 컬러 채널에  1.5를 곱하여 밝게 만든 영상, title="Original * 1.5"
imgF = imgRGB/255       # RGB 원본영상의 정규화 영상을 만든다. 0~1 범위의 부동소수 영상이다.
imgShow = imgF * 1.5
plt.subplot(223)
# [0, 1] 범위를 넘어서서는 값들은 imshow() 함수 내부에서 모두 truncation 다른 말로는 clip 동작이 일어난 후 화면에 출력된다.
# clip 동작: 0보다 작으면 0, 1보다 더 크면 1으로 변환되고, 그 사이의 값들은 그대로 유지된다.
plt.imshow(imgShow) # 위에서 변환해준 이미지를 보여준다.
plt.axis('off')             # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title('Original * 1.5')
plt.waitforbuttonpress()        # 키 혹은 버튼 입력을 기다린다.


# 4) 서브 화면 4 - 원본 컬러 영상에서 R 컬러 채널만  2로 곱하여 Red 색상만 높게 반영한 영상, title="Original[..., 0] * 2"
imgF = imgRGB/255       # RGB 원본영상의 정규화 영상을 만든다. 0~1 범위의 부동소수 영상이다.
imgF[..., 0] = imgF[..., 0] * 2
plt.subplot(224)
plt.imshow(imgF) # 위에서 변환해준 이미지를 보여준다.
plt.axis('off')             # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title('Original[..., 0] * 2')
plt.waitforbuttonpress()        # 키 혹은 버튼 입력을 기다린다.
```

코드 설명

1. import

```
import numpy as np
import cv2 as cv
from matplotlib import pyplot as plt
```

2. 경로 및 이름

```
Path = '../data/'
Name = 'RGBColors.JPG'
FullName = Path + Name
```

3. opencv 함수 imread()를 이용하여 이미지 읽기

```
image = cv.imread(FullName)      # read as 3 channel. default.
assert image is not None, f"No image file: [{FullName}]....!"  # 입력 영상을 제대로 읽어오지 못하여NULL을 반환.
```

:opencv 로 이미지를 읽게 되면 RGB 의 순서가 BGR 순서로 바뀌어 읽힌다.

4. matplotlib 에서 사용하기 위해 RGB 순서 바꿔주기

```
imgRGB = image[..., ::-1]    # = image[:, :, ::-1]. 같은 표현
```

5. interactive mode 로 설정

```
plt.ion()        # interactive mode로 설정
```

6. 최초의 창 생성 및 타이틀과 서브 타이틀 출력

```
plt.figure(num='Report1: Original image & its processed images')
plt.suptitle("Report1: Original image & its processed images",
fontsize=10,fontweight='bold')
plt.waitforbuttonpress() # 사용자가 입력하면 true
```

*waitforbuttonpress(): 사용자가 입력을 대기하다가 사용자가 입력을 하면 true 가 됨

7. 서브 1: 원본 컬러 영상 출력

```
# 1) 서브 화면1 – 원본 컬러 영상, title="Original Color Image"
plt.subplot(221) #row = 2, col = 2, index = 1 (1,1 위치)
plt.imshow(imgRGB) # 위에서 변환해준 이미지를 보여준다.
plt.axis('off')              # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title('Original Color Image')
plt.waitforbuttonpress()         # 키 혹은 버튼 입력을 기다린다.
```

:(1,1)위치에 위에서 RGB 순서를 바꾼 원본 사진을 출력한다.

*plt.subplot(221): row = 2, col = 2, index = 1 , (1,1) 위치

*plt.imshow() : 이미지 출력

*plt.axis(): 축과 라벨 끄기

*plt.title(): 타이틀 적어주기

8. 서브 2: 모노 영상 출력

```
# 2) 서브 화면2 – 모노 영상, title="Mono Image"
plt.subplot(222)
imgM = cv.cvtColor(image, cv.COLOR_BGR2GRAY)    # 3채널 영상을1채널gray로 변환
plt.imshow(imgM, cmap='gray') #단일 채널 이미지를gray 색상 채널에 매핑해준다.
plt.axis('off')              # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title("Mono Image")
plt.waitforbuttonpress()        # 키 혹은 버튼 입력을 기다린다.
```

: 원본 영상을 gray 로 변환해준 후 gray 색상 채널에 매핑하여 출력

*plt.subplot(222): (1,2) 위치

*cv.cvtColor(image, cv.COLOR_BGR2GRAY): 3채널 영상을 1채널 gray 로 변환

*plt.imshow(imgM, camp='gray'): 단일 채널 이미지를 cmap 을 이용하여 gray 색상 채널에 매핑

9. 서브 3: 원본 컬러 영상에서 각 컬러 채널에 1.5를 곱하여 밝게 만든 영상 출력

```
# 3) 서브 화면3 – 원본 컬러 영상에서 각 컬러 채널에1.5를 곱하여 밝게 만든 영상,
title="Original * 1.5"
imgF = imgRGB/255        # RGB 원본영상의 정규화 영상을 만든다. 0~1 범위의
부동소수 영상이다.
imgShow = imgF * 1.5
plt.subplot(223)
# [0, 1] 범위를 넘어서서는 값들은imshow() 함수 내부에서 모두truncation 다른
말로는clip 동작이 일어난 후 화면에 출력된다.
plt.imshow(imgShow) # 위에서 변환해준 이미지를 보여준다.
plt.axis('off')              # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title('Original * 1.5')
plt.waitforbuttonpress()        # 키 혹은 버튼 입력을 기다린다.
```

:정규화를 해준 영상에 1.5를 곱하여 RGB 값에 1.5를 각각 곱하여 imshow 로 자동 clipping 하여 출력해준다.

*imgF = imgRGB/255: 정규화

*imgShow = imgF * 1.5 : 1.5를 그냥 곱해주면 RGB 값에 각각 1.5배를 해주는 것과 동일

*plt.subplot(223): (2,1) 위치

*plt.imshow(imgShow): [0, 1] 범위를 넘어서는 값들은 imshow() 함수 내부에서 모두 clipping 동작이 일어난 후 화면에 출력됨.

*clip 동작: 0보다 작으면 0, 1보다 크면 1로 변환되고, 그 사이의 값들은 그대로 유지

10. 서브 4: 원본 컬러 영상에서 R 채널만 2를 곱한 영상 출력

```
# 4) 서브 화면4 – 원본 컬러 영상에서R 컬러 채널만2로 곱하여Red 색상만 높게 반영한
영상, title="Original[..., 0] * 2"
imgF = imgRGB/255        # RGB 원본영상의 정규화 영상을 만든다. 0~1 범위의
부동소수 영상이다.
imgF[..., 0] = imgF[..., 0] * 2
plt.subplot(224)
plt.imshow(imgF) # 위에서 변환해준 이미지를 보여준다.
plt.axis('off')              # plt.xticks([]), plt.yticks([]) 축과 라벨 끄기
plt.title('Original[..., 0] * 2')
plt.waitforbuttonpress()        # 키 혹은 버튼 입력을 기다린다.
```

:R 값만 바꾸기 위해 정규화한 이미지[...,0]에 2를 곱하여 주고 imshow 를 이용하여 출력시 자동으로 clipping 이 되게 출력해준다.

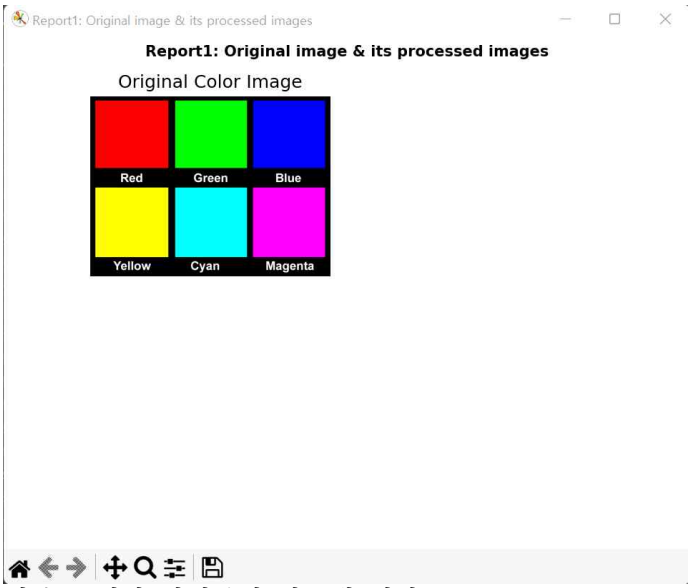
*imgF = imgRGB/255: 정규화
*imgF[...,0] = imgF[...,0]*2 :그냥 곱하면 RGB 값에 다 2를 곱하는 것과 마찬가지 이므로 R 값만 바꾸어주기 위해 [...,0]값만 바꾸어줌
*plt.subplot(224): (2,2) 위치

결과 출력 및 설명

1. 첫 번째

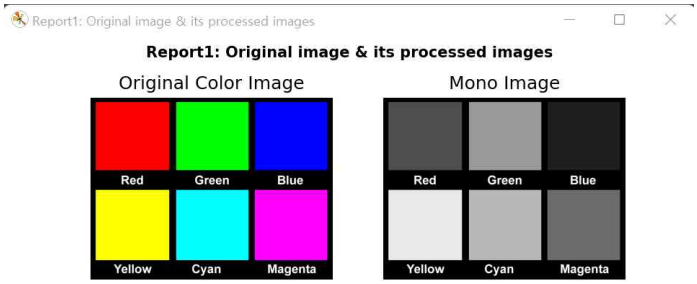
실행을 하게 되면 plt.figure()로 인해 타이틀과 서브 타이틀을 지정해준 새 창이 나오게 된다.

2. 두 번째



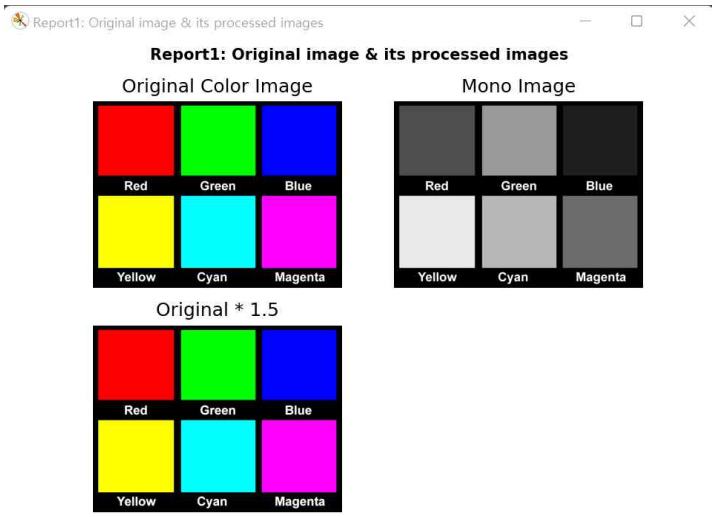
원본 그림과 타이틀이 나오게 된다.

3. 세 번째



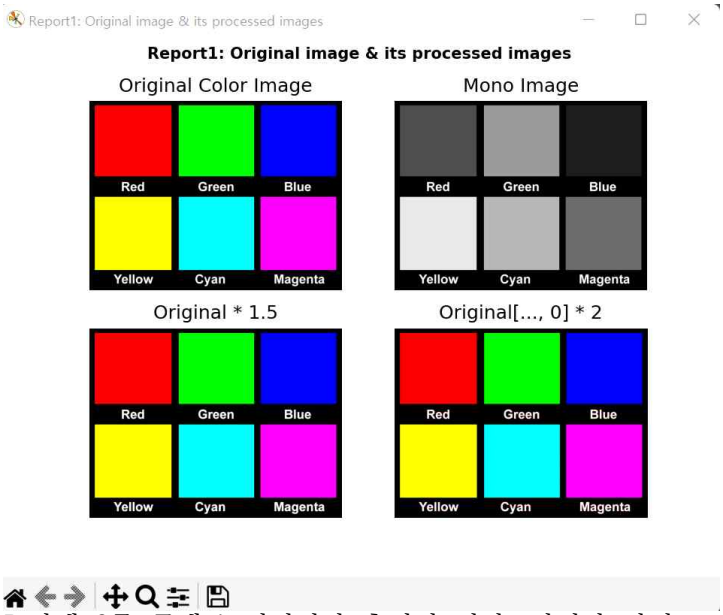
원본 파일에서 gray 로 변환한 이미지가 출력이 된다.

4. 네 번째



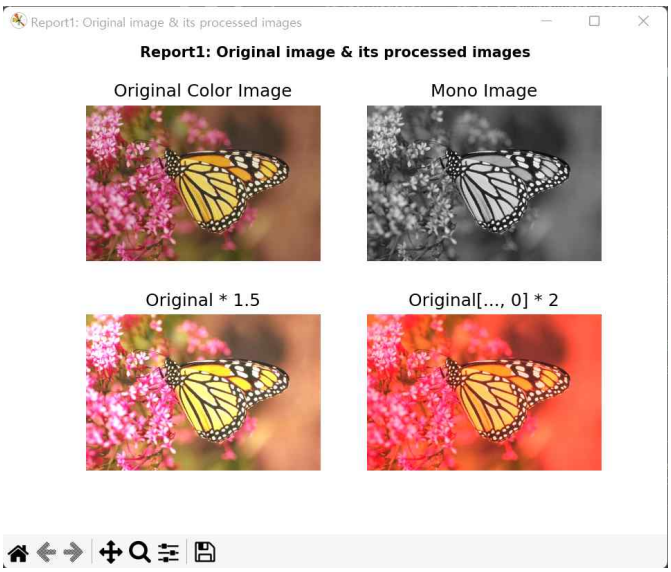
원본 사진에 RGB 값에 1.5를 곱한 이미지가 출력된다. 여기서 밝기가 안 바뀐 이유는 영상 속 이미지의 RGB 값이 거의 1이나 0이기 때문에 곱해도 clipping 으로 인해 0 또는 1로 바뀌기 때문이다.

5. 다섯 번째



R 값에 2를 곱해준 이미지가 출력이 된다. 마지막 사진도 앞선 이유와 마찬가지로 값이 거의 0이나 1이기 때문에 clipping 으로 인해 값이 거의 안 바뀐 것이다.

*또 다른 예시(나비)



잘 변환이 되는 모습을 볼 수 있다.

마무리

이번 과제는 영상을 받아 원본, 모노, 밝기 변환을 하는 과제였다. 이번 과제를 하면서 천천히 다시 복습하며 따라가다보니 수업 때 들었던 이야기를 다시 한 번 복습할 수 있는 계기가 되었다. 확실이 이번 과제를 통해 opencv 로 받은 이미지를 matplot 와 같은 다른 라이브러리를 사용할 때 RGB 순서를 바꾸어 주어야 된다는 것과 영상을 모노로 만들 때 컬러매핑이 필요하다는 것, 정규화의 중요성 및 plt.imshow 가 자동적으로 clipping 을 해준다는 것, interactive mode 로 바꾸는 방법 등등 이전 수업 때 배웠던 것들을 대부분 사용을 해보며 다시 한번 내용을 상기 시키고 스스로 영상처리를 해보며 영상처리라는 학문에 흥미를 좀 더 느끼게 되는 계기가 되었다. 또한 이번 결과 분석을 통해 처음에 마지막 두 상황에서 어떠한 이유로 밝기의 변화가 있었는지 그냥 넘어갈 뻔했던 것도 결과 분석을 함으로써 정확한 이유를 찾고 넘어갈 수 있었어서 좋았다. 이번 과제를 하면서 복습도 하고 새로운 사실을 알아갈 수 있었기에 얻어가는 것이 많은 것 같다.