[과제2] Chapter 3 영상처리

2021098 노하림

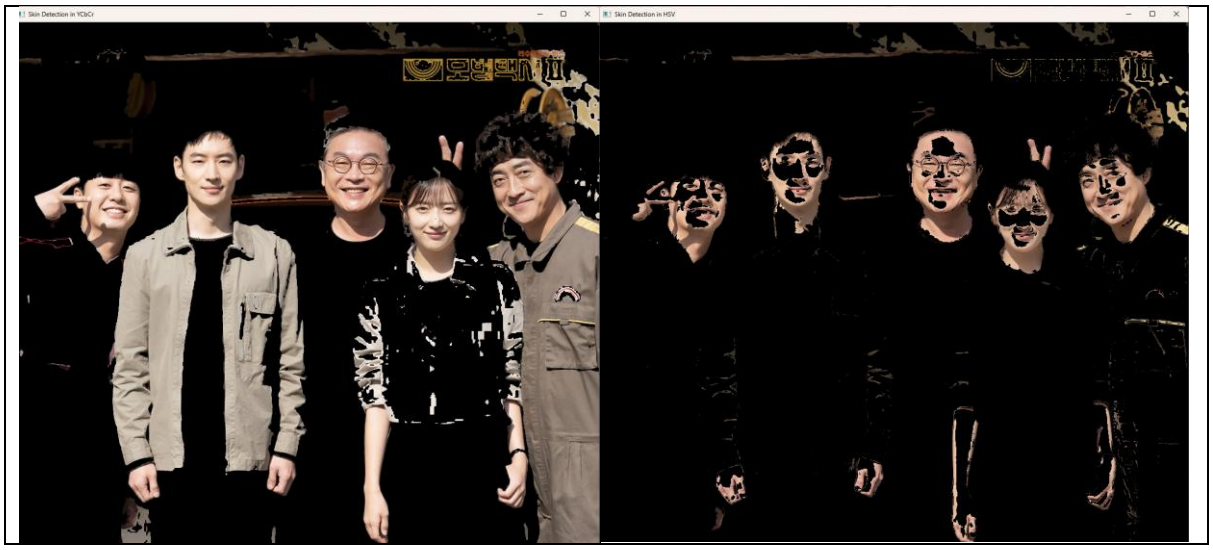
HW#2-1 : skin color detection

[코드 및 설명]

```
# 필요한 라이브러리를 임포트한다.
import cv2 as cv

# 'face2.jpg' 이미지를 불러온다.
img = cv.imread('face2.jpg')

# 이미지가 제대로 로드되었는지 확인한다.
if img is None:
    print("no img")

# 무한루프로 사용자의 키 입력을 기다린다.
while True:
    # 원본 이미지를 화면에 표시한다.
    cv.imshow("face img", img)
    key = cv.waitKey()


    # 사용자가 'q' 키를 누르면 프로그램을 종료한다.
    if key == ord('q'):
        break

    # 1 Keybord 로 컬러모델 선택
    # 사용자가 'y' 키를 누르면 YCbCr 컬러 모델을 사용하여 피부색을 검출한다.
    elif key == ord('y'):

        # 2 선택된 컬러모델로 변환
        img_color = cv.cvtColor(img, cv.COLOR_BGR2YCrCb)
        # 3 픽셀 별로 피부색인지 확인
        mask = cv.inRange(img_color, (0, 133, 77), (255, 173, 127))
        # 4 피부색 출력
        skin_detection = cv.bitwise_and(img, img, mask=mask)
        cv.imshow('Skin Detection in YCbCr', skin_detection)
        cv.waitKey(0)

    # 사용자가 'h' 키를 누르면 HSV 컬러 모델을 사용하여 피부색을 검출한다.
    elif key == ord('h'):
```

```
        # 2 선택된 컬러모델로 변환
        img_color = cv.cvtColor(img, cv.COLOR_BGR2HSV)
        # 3 픽셀 별로 피부색인지 확인
        mask = cv.inRange(img_color, (0, 70, 50), (50, 150, 255))
        # 4 피부색 출력
        skin_detection = cv.bitwise_and(img, img, mask=mask)
        cv.imshow('Skin Detection in HSV', skin_detection)
        cv.waitKey(0)

# 모든 창을 닫는다.
cv.destroyAllWindows()
```

[결과]



- 왼쪽 사진은 'y'를 눌러 YCbCr컬러 모델로 변환해 피부색인지 확인 후 피부색이면 그대로 아니면 검정색으로 출력하는 결과를 확인할 수 있다.
- 오른쪽 사진은 'h'를 눌러 HSV컬러 모델로 변환해 피부색인지 확인 후 피부색이면 그대로 아니면 검정색으로 출력하는 결과를 확인할 수 있다.

HW#2-1 : 동영상에서 skin color detection *Bonus

[코드 및 설명]

```
# 필요한 라이브러리를 임포트한다.
import cv2 as cv

# 'face2.mp4' 동영상을 불러온다.
cap = cv.VideoCapture('face2.mp4')

color_mode=''
# 동영상의 각 프레임을 읽어온다.
while True:
    ret, frame = cap.read()

    # 프레임 읽기에 실패하면 루프를 종료한다.
    if not ret:
        print('프레임 획득에 실패하여 루프를 나갑니다.')
        break

    key = cv.waitKey(1)
    # 사용자에게 키를 받는다
    if key == ord('y'):
        color_mode = 'COLOR_BGR2YCrCb'
    elif key == ord('h'):
        color_mode = 'COLOR_BGR2HSV'
    elif key == ord('q'):
        break



    if color_mode=="COLOR_BGR2HSV":
        img_color = cv.cvtColor(frame, cv.COLOR_BGR2HSV)
        # 3 픽셀 별로 피부색인지 확인
        mask = cv.inRange(img_color, (0, 70, 50), (50, 150, 255))
        # 4 피부색 출력
        skin_detection = cv.bitwise_and(frame, frame, mask=mask)
        cv.imshow('face img', skin_detection)

    elif color_mode=='COLOR_BGR2YCrCb':
        # 2 선택된 컬러모델로 변환
        img_color = cv.cvtColor(frame, cv.COLOR_BGR2YCrCb)
        # 3 픽셀 별로 피부색인지 확인
        mask = cv.inRange(img_color, (0, 133, 77), (255, 173, 127))
        # 4 피부색 출력
```

```
            skin_detection = cv.bitwise_and(frame, frame, mask=mask)
            cv.imshow('face img', skin_detection)
        else:
            cv.imshow('face img',frame)

cap.release()
cv.destroyAllWindows()
```

[결과]

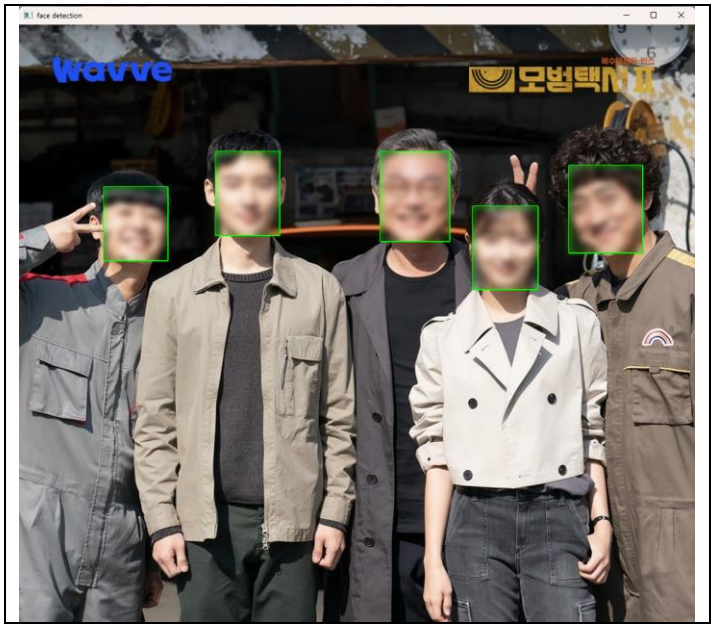
- 'y'를 눌러 YCbCr컬러 모델로 변환해 피부색인지 확인 후 피부색이면 그대로 아니면 검정색으로 출력하는 결과를 확인할 수 있다.
- 'h'를 눌러 HSV컬러 모델로 변환해 피부색인지 확인 후 피부색이면 그대로 아니면 검정색으로 출력하는 결과를 확인할 수 있다.
-

HW#2-2 : face mosaic

[코드 및 설명]

```
# 필요한 라이브러리를 임포트한다.
import cv2 as cv
import cvlib as cvl

ksize = 31

# 'face2.jpg' 이미지를 불러온다.
img = cv.imread('face2.jpg')

# 1 얼굴 검출
# 이미지에서 얼굴을 감지한다.
faces, confidences = cvl.detect_face(img)
# 2 검출된 얼굴 부분만 추출
# 감지된 각 얼굴에 대하여 블러 효과를 적용한다.
for (x, y, x2, y2), conf in zip(faces, confidences):
    cv.rectangle(img, (x, y), (x2, y2), (0, 255, 0), 2)
    roi = img[y:y2, x:x2]  # 얼굴 영역을 지정한다.
    # 3 추출된 얼굴 부분만 모자익 처리
    roi = cv.GaussianBlur(roi, (ksize, ksize), 0.0)  # 가우시안 블러를 적용한다.
    # 4 모자익 처리된 얼굴을 원래 영상에 적용
    img[y:y2, x:x2] = roi  # 원본 이미지에 블러 처리된 얼굴 영역을 대체한다.

# 결과를 표시한다.
cv.imshow('face detection', img)
cv.waitKey()
cv.destroyAllWindows()
```

[결과]



- 얼굴 영역을 검출해서 모자익 처리 후 원래 영상에 적용한 결과를 확인할 수 있다.

HW#2-2 : 동영상에서 Face mosaic *Bonus

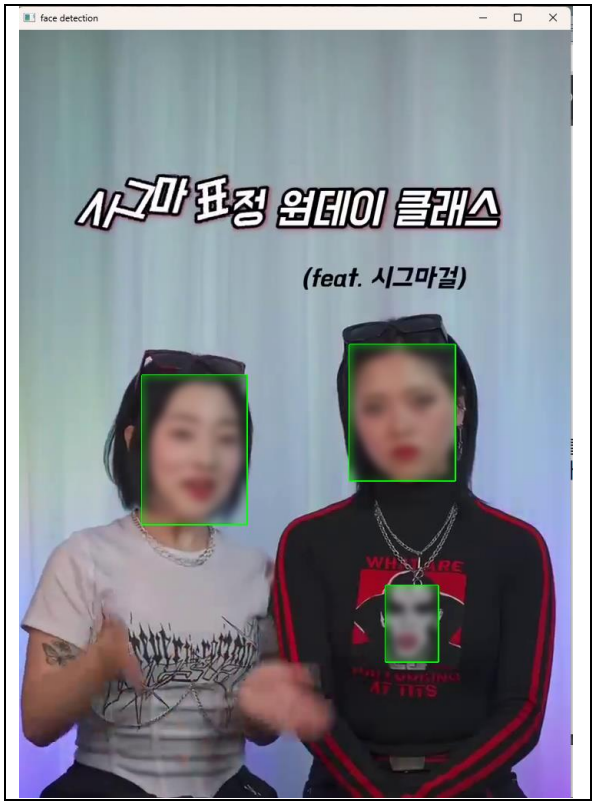
[코드 및 설명]

```
# 필요한 라이브러리를 임포트한다.
import cv2 as cv
import cvlib as cvl

cap = cv.VideoCapture('face2.mp4')
ksize = 31

# 동영상의 각 프레임을 읽어온다.
while True:
    ret, frame = cap.read()
    # 프레임 읽기에 실패하면 루프를 종료한다.
    if not ret:
        print('프레임 획득에 실패하여 루프를 나갑니다.')
        break
```

```
    # 프레임에서 얼굴을 감지한다.
    faces, confidences = cvl.detect_face(frame)
    # 감지된 각 얼굴에 대하여 블러 효과를 적용한다.
    for (x, y, x2, y2), conf in zip(faces, confidences):
        cv.rectangle(frame, (x, y), (x2, y2), (0, 255, 0), 2)
        roi = frame[y:y2, x:x2]
        roi = cv.GaussianBlur(roi, (ksize, ksize), 0.0)
        frame[y:y2, x:x2] = roi

    # 결과를 표시한다.
    cv.imshow('face detection', frame)

    # 'q' 키를 누르면 종료한다.
    key = cv.waitKey(1)
    if key == ord('q'):
        break

cap.release()
cv.destroyAllWindows()
```

[결과]

- 동영상에서 프레임별로 얼굴을 감지하고 해당 얼굴 영역에 가우시안 블러 효과를 적용한 결과를 확인할 수 있다.



HW#2-3 : rotate by 90 degrees

[코드 및 설명]

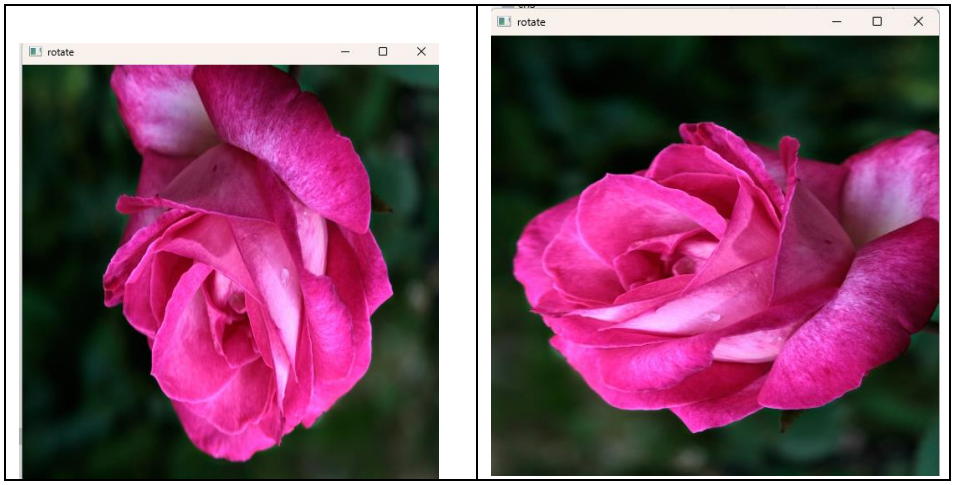
```
import cv2 as cv

import numpy as np


# 이미지를 불러온다.

img = cv.imread('rose.png')


# 이미지의 크기를 조절한다.

img = cv.resize(img, dsize=(500, 500))

rows, cols = img.shape[:2]
```

```
current_img = img.copy()

# 마우스 콜백 함수
def rotate(event, x, y, flags, param):
    global current_img  # 전역 변수 사용

    # 1 왼쪽 마우스 버튼 클릭 시 시계 반대 방향으로 회전
    if event == cv.EVENT_LBUTTONDOWN:
        src_point = np.float32([[0, 0], [0, rows - 1], [cols - 1, 0]])
        dst_point = np.float32([[0, rows - 1], [cols - 1, rows - 1], [0, 0]])
        affine_matrix = cv.getAffineTransform(src_point, dst_point)
        current_img = cv.warpAffine(current_img, affine_matrix, (cols, rows))
        cv.imshow('rotate', current_img)

    # 2 오른쪽 마우스 버튼 클릭 시 시계 방향으로 회전
    elif event == cv.EVENT_RBUTTONDOWN:
        src_point = np.float32([[0, 0], [0, rows - 1], [cols - 1, 0]])
        dst_point = np.float32([[cols - 1, 0], [0, 0], [cols-1, rows - 1]])
        affine_matrix = cv.getAffineTransform(src_point, dst_point)
        current_img = cv.warpAffine(current_img, affine_matrix, (cols, rows))
        cv.imshow('rotate', current_img)

# 이미지 표시 및 마우스 콜백 함수 설정
cv.namedWindow('rotate')
cv.imshow('rotate', img)
cv.setMouseCallback('rotate', rotate)

# 사용자 키 입력 대기
while(True):
    if cv.waitKey(1) == ord('q'):
```

```
        cv.destroyAllWindows()

        break
```

[결과]

- 왼쪽 버튼을 누를 때마다 시계반대방향으로 90도 회전하는 결과를 확인할 수 있다.



- 오른쪽 버튼을 누를 때마다 시계방향으로 90도 회전하는 결과를 확인할 수 있다.

HW#2-2 : 동영상에서 Face mosaic *Bonus

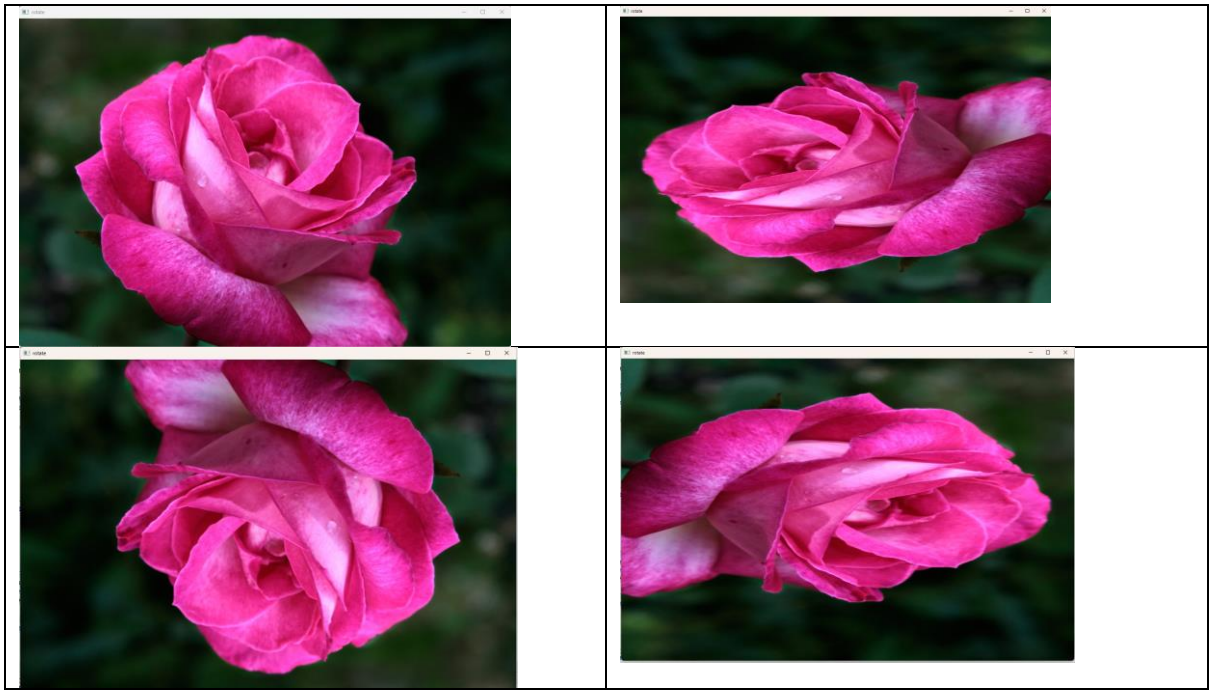
[코드 및 설명]

```
import cv2 as cv
import numpy as np

# 'rose.png' 이미지를 불러온다.
img = cv.imread('rose.png')
rows, cols = img.shape[:2]
# 현재 이미지를 복사하여 누적 회전을 위한 초기 이미지로 사용한다.
current_img = img.copy()

# 마우스 콜백 함수 정의
def rotate(event, x, y, flags, param):
    global current_img  # 전역 변수를 사용하여 현재 이미지를 업데이트한다.

    # 왼쪽 마우스 버튼 클릭 시 시계 반대 방향으로 회전
    if event == cv.EVENT_LBUTTONDOWN:
        src_point = np.float32([[0, 0], [0, rows - 1], [cols - 1, 0]])
```

```
            dst_point = np.float32([[0, rows - 1], [cols - 1, rows - 1], [0, 0]])
            affine_matrix = cv.getAffineTransform(src_point, dst_point)
            current_img = cv.warpAffine(current_img, affine_matrix, (cols, rows))
            cv.imshow('rotate', current_img)

    # 오른쪽 마우스 버튼 클릭 시 시계 방향으로 회전
    elif event == cv.EVENT_RBUTTONDOWN:
            src_point = np.float32([[0, 0], [0, rows - 1], [cols - 1, 0]])
            dst_point = np.float32([[cols - 1, 0], [0, 0], [cols-1, rows - 1]])
            affine_matrix = cv.getAffineTransform(src_point, dst_point)
            current_img = cv.warpAffine(current_img, affine_matrix, (cols, rows))
            cv.imshow('rotate', current_img)

# 이미지를 화면에 표시하고 마우스 콜백 함수를 설정한다.
cv.namedWindow('rotate')
cv.imshow('rotate', img)
cv.setMouseCallback('rotate', rotate)
# 사용자의 키 입력을 대기한다. 'q' 키를 누르면 프로그램을 종료한다.
while(True):
    if cv.waitKey(1) == ord('q'):
            cv.destroyAllWindows()
            break
```

[결과]

- 왼쪽 버튼을 누를 때마다 시계반대방향으로 90도 회전하는 결과를 확인할 수 있다.



- 오른쪽 버튼을 누를 때마다 시계방향으로 90도 회전하는 결과를 확인할 수 있다.