## 작성자: 김상화 (201978314)

### 문)

다음의 사진을 이용하여 반은 정상인데 반은 사자인 얼굴을 어색하지 않게 만들어보자.

사용파일: man\_face, lion\_face

#### <힌트>

두 영상을 절반씩 가져다가 새로운 영상으로 단순히 합성하면 그냥 두 개의 영상을 이어 붙인 것처럼 보이기 때문에 어색하다.

따라서 수업시간에 배운 알파블렌딩을 이용하여 두 영상이 만나는 지점의 일정 부분을 서서히 변하게 하면서 자연스럽게 두 개의 얼굴이 하나의 얼굴로 합성할 수 있도록 해보자.

2. X축 - 알파 블랜딩 적용

영상의 15%를 알파 블렌딩의 범위로 지정합니다.

1. X축 - 그냥 두 영상을 합성

#### 답)

#### import cv2 import numpy as np # 영상의 15%를 알파 블렌딩의 범위로 지정 import numpy as np alpha\_width\_rate = 15 # 영상의 15%를 알파 블렌딩의 범위로 지정 alpha\_width\_rate = 15 # 합성할 두 영상 읽기 img face = cv2.imread('../img/man face.ipg') # 합성할 두 영상 읽기 img\_face = cv2.imread(' img skull = cv2.imread('../img/lion face.jpg') img\_face = cv2.imread('../img/man\_face.jpg') img\_skull = cv2.imread('../img/lion\_face.jpg') # 입력 영상과 같은 크기의 결과 영상 준비 img\_comp = np.zeros\_like(img\_face) # 입력 영상과 같은 크기의 결과 영상 준비 img\_comp = np.zeros\_like(img\_face) # 연산에 필요한 좌표 계산 # 연산에 필요한 좌표 계산 height, width = img\_face.shape[:2] height, width = img\_face.shape[:2] middle = width//2 middle = width//2 # 영상의 중앙 좌표 middle = width//2 # 영상의 중앙 alpha\_width = width \* alpha\_width\_rate // 100 # 알파 볼렌딩 start = middle - alpha\_width//2 # 알파 볼렌딩 alpha\_width = width \* alpha\_width\_rate // 100 # 알파 블렌딩 범위 start = middle - alpha\_width//2 # 알파 블렌딩 시작 지점 step = 100/alpha\_width step = 100/alpha width # 알파 값 간격 # 입력 영상의 절반씩 복사해서 결과 영상에 합성 img\_comp[:, :middle, :] = img\_face[:, :middle, :].copy() img\_comp[:, middle:, :] = img\_skull[:, middle:, :].copy() cv2.imshow('half', img\_comp) # 입력 영상의 절반씩 복사해서 결과 영상에 합성 img\_comp[:, :middle, : ] = img\_face[:, :middle, :].copy() $img\_comp[:, middle:, :] = img\_skull[:, middle:, :].copy()$ cv2.imshow('half', img\_comp) # 알파 값을 바꾸면서 알파 블렌딩 적용 [파 값을 바꾸면지 끌피 돌편으 그는 i in range(alpha\_width+1): alpha = (100 - step \* i) / 100 # 중감 간격에 따른 알파 값 (1~0) beta = 1 - alpha # 베타 값 (0~1)| # 알파 볼텐딩 적용 img\_comp[:, start+i] = img\_face[:, start+i] \* ₩ alpha + img\_skull[:, start+i] \* beta # 알파 값을 바꾸면서 알파 블렌딩 적용 for i in range(alpha\_width+1): alpha = (100 - step \* i) / 100 # 증감 간격에 따른 알파 값 (1~0) beta = 1 - alpha # 베타 값 (0~1) # 알파 블렌딩 적용 img\_comp[:, start+i] = img\_face[:, start+i] \* \(\psi\) alpha + img skull[:, start+i] \* beta cv2.imshow('half skull', img\_comp) print(i, alpha, beta) cv2.waitKey( cv2.destroyAllWindows() cv2.imshow('half skull', img\_comp) cv2.waitKey() cv2.destroyAllWindows()

#### 3. Y축 - 그냥 두 영상을 합성



# 4. Y축 - 알파 블랜딩 적용



```
import cv2
import numpy as np
# 영상의 15%를 알파 블렌딩의 범위로 지정
alpha_height_rate = 15
# 합성할 두 영상 읽기
img_face = cv2.imread('../img/man_face.jpg')
img_skull = cv2.imread('../img/lion_face.jpg')
# 입력 영상과 같은 크기의 결과 영상 준비
img_comp = np.zeros_like(img_face)
# 연산에 필요한 좌표 계산
height, width = img_face.shape[:2]
                                          # 영상의 중앙 좌표
middle = height//2
alpha_height = height * alpha_height_rate // 100 # 알파 블렌딩 범위
                                       # 알파 블렌딩 시작 지점
start = middle - alpha height//2
                                          # 알파 값 간격
step = 100/alpha_height
# 입력 영상의 절반씩 복사해서 결과 영상에 합성
img_comp[:middle, :, :] = img_face[:middle, :, :].copy()
img_comp[middle:, :, :] = img_skull[middle:, :, :].copy()
cv2.imshow('half', img_comp)
# 알파 값을 바꾸면서 알파 블렌딩 적용
for i in range(alpha_height+1):
   alpha = (100 - step * i) / 100 # 증감 간격에 따른 알파 값 (1~0)
   beta = 1 - alpha
                               # 베타 값 (0~1)
   # 알파 블렌딩 적용
   img_comp[start+i,:] = img_face[start+i,:] * \footnote{\psi}
                           alpha + img_skull[start+i,:] * beta
   print(i, alpha, beta)
cv2.imshow('half skull', img_comp)
```

cv2.destroyAllWindows()

```
import cv2
import numpy as np
# 영상의 15%를 알파 블렌딩의 범위로 지정
alpha_height_rate = 15
#합성할 두 영상 읽기
# 월경불 구 8명 최기
img_face = cv2.imread('../img/man_face.jpg')
img_skull = cv2.imread('../img/lion_face.jpg')
# 입력 영상과 같은 크기의 결과 영상 준비
img_comp = np.zeros_like(img_face)
# 연산에 필요한 좌표 계산
height, width = img_face.shape[:2]
                                                                            # 영상의 중앙 좌표
middle = height//2
alpha_height = height * alpha_height_rate // 100 # 알파 볼렌딩 범위
start = middle - alpha_height//2 # 알파 볼렌딩 시작 지점
step = 100/alpha_height # 알파 값 간격
# 입력 영상의 절반씩 복사해서 결과 영상에 합성
|img_comp[:middle, :, :] = img_face[:middle, :, :].copy()
img_comp[middle:, :, :] = img_skull[middle:, :, :].copy()
cv2.imshow('half', img_comp)
# 알파 값을 바꾸면서 알파 볼렌딩 적용
for i in range(alpha_height+1 ):
    alpha = (100 - step * i) / 100 # 중감 간격에 따른 알파 값 (1~0)
    tata = 1 - alpha # 베타 값 (0~1)
       img_comp[start+i,:] = img_face[start+i,:] * \ alpha + img_skull[start+i,:] * beta
      print(i, alpha, beta)
cv2.imshow('half skull', img_comp)
cv2.waitKey()
cv2.destroyAllWindows()
```