

1. 서론

앞으로 있을 컴퓨터비전개론의 실습을 준비하기 위해 파이썬 가상환경을 구축하고, 관련 라이브러리를 설치해 보고 해당 환경이 잘 작동하는지 확인하기 위해 간단한 예제코드를 실행해 본다.

2. 본론

2-1. 가상환경 구축

가상환경을 구축하는 방법은 아래와 같다. ComputerVision이라는 이름으로 가상환경을 만든다.

```
python -m venv ComputerVision
```

만든 가상환경을 활성화한다.

```
source ComputerVision/bin/activate
```

```
minmunui@bagdongjin-ui-MacBookAir ComputerVisionPrinciples % ls
ComputerVision HW1
minmunui@bagdongjin-ui-MacBookAir ComputerVisionPrinciples %
```

가상환경이 활성화된 모습

pip명령어를 사용하여 필요한 라이브러리를 설치한다.

```
pip install pillow numpy image
```

이로써 필요한 라이브러리가 설치된 가상환경 구축이 완료되었다.

2-2. vision_hw1.py

예제코드 vision_hw1.py가 포함한 코드의 내용은 아래와 같다.

```
from PIL import Image
import numpy as np

im = Image.open('chipmunk.png')

print (im.size, im.mode, im.format)

im.show()
```

해당 프로젝트에 필요한 라이브러리 Image와 numpy를 불러온다.

같은 디렉토리에 존재하는 파일 `chipmunk.png` 를 `image object` 를 열어 `im`에 저장한다.

해당 이미지 `im` 의 정보 (크기, 모드, 포맷)을 출력한다.

이미지 `im` 창을 띄운다.

```
im = im.convert('L')

im2 = im.crop((280,150,430,300))

im2.save('chipmunk_head.png', 'PNG')

im2_array = np.asarray(im2)

average = np.mean(im2_array)
```

이미지 `im` 을 흑백으로 변환한다.

`im` 을 크롭한 이미지를 `im2` 에 저장한다.

`im2` 를 `chipmunk_head.png` 이름으로 저장한다.

`im2` 를 numpy라이브러리의 array로 변환하여 `im2_array` 에 저장한다.

`im2_array` 의 entry평균을 계산하여 `average` 에 저장한다.

```
im3_array = im2_array.copy()

for x in range(0,150):
    for y in range(0,150):
        im3_array[y,x] = min(im3_array[y,x] + 50, 255)

im3 = Image.fromarray(im3_array)
im3.save('chipmunk_head_bright.png', 'PNG')
```

`im2_array` 를 카피하여 `im3_array` 에 저장한다.

`im3_array` 의 모든 엔트리에 50을 더한다. 만약 값이 최대치인 255를 초과할 경우 255를 저장한다.

`im3_array` 배열을 이미지로 변환하여 `im3` 에 저장한다.

`im3` 이미지를 `chipmunk_head_bright.png` 라는 이름으로 저장한다.

```
im4_array = im2_array.copy()

im4_array = im4_array * 0.5

im4_array = im4_array.astype('uint8')

im4 = Image.fromarray(im4_array)
im4.save('chipmunk_head_dark.png', 'PNG')
```

`im4_array` 에 `im2_array` 를 카피한다.

`im4_array` 의 모든 엔트리의 값을 절반으로 한다.

`im4_array` 의 모든 엔트리의 자료형을 uint8로 변환한다.

`im4_array` 를 `im4` 에 이미지형태로 저장한다.

`im4` 를 `chipmunk_head_dark.png` 이름으로 저장한다.

```
grad = np.arange(0,256)

# repeat this 1-D array 256 times to create a 256x256 2-D array
grad = np.tile(grad,[256,1])

# convert to uint8 and then to a PIL image and save
im5 = Image.fromarray(grad.astype('uint8'))
im5.save('gradient.png', 'PNG')
```

`grad` 에 0부터 255까지 포함된 1차원 배열을 저장한다.

`grad` 배열을 256번 반복하여 2차원 이미지를 만든다.

각 엔트리의 타입을 uint8로 변경한 배열을 `Image` 로 변경하여 `im5` 에 저장한다..

`im5` 를 `gradient.png` 이름으로 저장한다.

생성된 파일은 아래와 같다.



chipmunk.png



chipmunk_head
_bright.png



chipmunk_head
_dark.png



gradient.png

3. 결론

파이썬 가상환경 설정이 완료되었고, 필요한 라이브러리도 설치했다. 예제코드를 돌려보고 설정이 완료되었음도 확인했다.

결과적으로는 다람쥐의 머리 부분을 150*150크기로 크롭하여 흑백처리한 사진

`chipmunk_head.png`, 각 픽셀에 50을 더해 전체적으로 밝게 변한 사진

`chipmunk_head_btight.png`, 각 픽셀의 값을 절반하여 전체적으로 어두어진 사진

`chipmunk_head_dark.png`, 0부터 255까지의 밝기를 2D크기로 표현한 사진 `gradient.png`을 얻었다.

`image` 라는 클래스를 통해서 이미지파일을 불러올 수도 있고, `numpy` 를 이용한 배열의 형태로 변형할 수도 있다. 배열로 변경된 이미지는 각각의 엔트리를 수정할 수도 있었다. 수정된 파일은 이미지형태로 저장할 수도 있음을 확인했다.