HW 01 - REPORT

소속 : 정보컴퓨터공학부
학번 : 202155592
이름 : 이지수

서론

실습 목표

1. Anaconda를 설치하여 실습 환경을 구축한다.
2. 예제 코드를 통해 PIL 및 numpy 사용법, pixel 값 조작 방법 등을 익힌다.

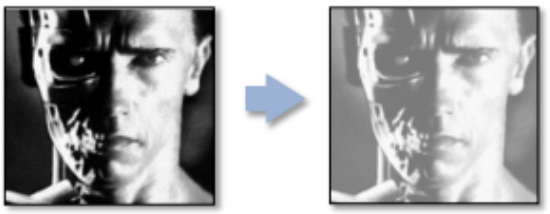
이론적 배경

1. What is an image?
    - A grid (matrix) of intensity values
    - 0 = black, 255 = white

2. Images as functions
    - An image contains discrete numbers of pixels
    - Can think of image as a function f
        - Grayscale
            - f (x,y) gives intensity at position (x,y)
            - f: [a,b] x [c,d] ⇒ [0,255]
        - Color

- f (x,y) = [r(x,y), g(x,y), b(x,y)]

3. Image transformations



$g(x,y) = f(x,y) + 20$

$g(x,y) = f(-x,y)$

4. Noise reduction

- Mean filtering : 이미지의 특정 좌표 값을 주변 픽셀 값들의 산술 평균으로 설정
- Linear filtering (cross-correlation, convolution)
  - Replace each pixel by a linear combination (a weighted sum) of its neighbors
  - The prescription for the linear combination is called the “kernel”

$$S[f](m,n) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(m+i, n+j)$$

  - Cross-correlation

1	2	3
4	5	6
7	8	9

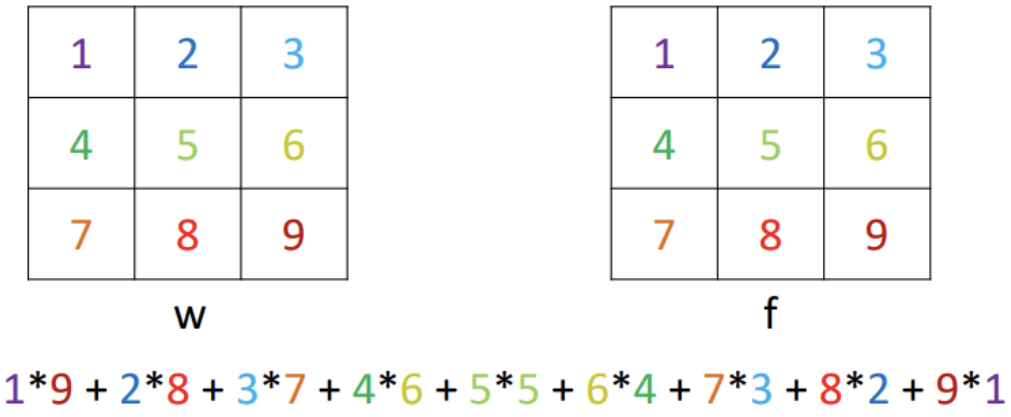
w

1	2	3
4	5	6
7	8	9

f

1*1 + 2*2 + 3*3 + 4*4 + 5*5 + 6*6 + 7*7 + 8*8 + 9*9

- Convolution



본론

1. 실습 환경 구축
   - `conda create -n cv2024 python=3.11.4` : 'cv2024'라는 이름의 새로운 가상 환경을 생성하고 python 3.11.4 ver을 설치한다.
   - `conda activate cv2024` : 생성한 가상 환경 'cv2024'를 활성화한다.
   - `conda install pillow numpy` : pillow와 numpy 라이브러리를 설치한다.

2. 코드 분석

```
from PIL import Image
import numpy as np

im = Image.open('chipmunk.png')
print (im.size, im.mode, im.format)

im.show()
```

실습에 필요한 라이브러리를 호출한다.

'chipmunk.png' 파일을 열어 Image 객체로 만든다.

이미지의 크기(size), 픽셀 포맷(mode), 파일 포맷(format)을 출력한 결과는 아래와 같다.

```
[Running] python -u "c:\Users\1\Desktop\computer-vision\hw1\vision_hw1.py"
(750, 599) RGB JPEG
```

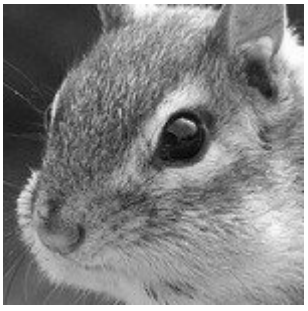
이후 이미지를 임시 파일로 저장하고 해당 파일을 열어 이미지를 보여준다.

```
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)
```

현재 이미지 `im`을 흑백으로 변환한다.

다람쥐의 머리가 포함된 영역을 추출하기 위해 (280,150) 위치에서 시작하여 (430,300) 위치로 끝나는 사각 영역을 선택하여 자른다.

크롭한 이미지 `im2`를 'chipmunk_head.png'로 저장한다. 저장한 파일은 아래와 같다.



`im2`를 numpy 배열로 변환하고 평균 밝기(픽셀 값의 평균)를 계산한다.

```
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` 복사본인 `im3_array`를 만든다.

반복문을 돌면서 각 픽셀의 밝기 값을 50만큼 증가시키는데, 픽셀 값이 255를 넘지 않도록 min 함수를 사용한다.

`im3_array`를 이미지로 변환하고 'chipmunk_head_bright.png'로 저장한다. 저장한 파일은 아래와 같다.

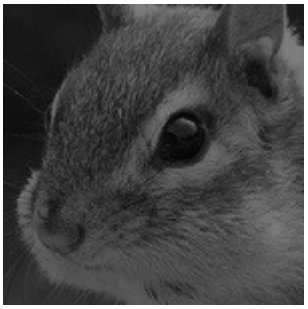
```
im4_array = im2_array.copy()

im4_array = im4_array * 0.5

im4_array = im4_array.astype('uint8')

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

`im2_array` 복사본인 `im4_array`를 만든다.

배열의 모든 픽셀 값에 0.5를 곱함으로써 이미지를 어둡게 만든다.

float 배열을 uint8 배열로 변환한다.

`im4_array`를 이미지로 변환하고 'chipmunk_head_dark.png'로 저장한다. 저장한 파일은 아래와 같다.



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

grad = np.tile(grad,[256,1])

im5 = Image.fromarray(grad.astype('uint8'))
im5.save('gradient.png','PNG')
```

0부터 255까지의 값을 가지는 1차원 배열 `grad` 생성한다.

생성된 1차원 배열 `grad`를 256번 반복하여 256*256 크기의 2차원 배열을 만든다.

uint8로 변환한 배열을 이미지로 변환하고 ‘gradient.png’로 저장한다. 저장한 파일은 아래와 같다.

결론

- 파이썬의 pillow 및 numpy 라이브러리를 활용하여 기본적인 이미지 처리 기법을 학습할 수 있다.

- 이미지 처리 작업을 시작하기 전에 이미지의 크기, 픽셀 포맷, 파일 포맷 등 이미지에 대한 기본적인 정보를 파악해야 한다.
- 이미지 처리 작업 시 필요에 따라 원하는 영역을 잘라서 사용한다.
- 픽셀 값을 증가시키면 밝은 이미지를, 감소시키면 어두운 이미지를 생성할 수 있다.
- numpy를 통해 새로운 이미지를 생성하고 이를 파일로 저장할 수 있다.