HW 01 – REPORT

소속 : 정보컴퓨터공학부

학번 : 201824523

이름 : 안혜준

1. 서론

이번 실습 목표는 파이썬의 PIL과 NumPy를 사용하여 기초적인 디지털 이미지 처리를 직접 해 보는 것이다. 실습의 내용으로는 이미지 파일을 열고, 그레이 스케일로 변환하고, 이미지의 sub-region을 추출하며, 간단한 이미지 밝기 조작과 새로운 이미지 생성이 있다.

그레이 스케일의 이미지는 픽셀당 1 byte를 할당하여 각 0~255의 값을 가진다. 이는 결국 픽셀의 배열로 나타내질 수 있다. 즉, (h x w) 크기의 이미지는 각 1 byte가 할당된 (h x w)의 uint8자료형 2차원 배열이다. 이번 실습에서는 이 배열에 filtering을 통하여 원하는 이미지로 변환을 할 것이다.

실습에 사용되는 파이썬 라이브러리는 NumPy와 PIL이다. NumPy는 기존 파이썬 리스트보다 더욱 강력하게 배열을 처리할 수 있도록 도와주는 라이브러리이다. PIL은 Python Imaging Library의 약자로 다양한 이미지 파일 형식을 지원하고 강력한 이미지 처리와 그래픽 기능을 제공하는 자유-오픈 소스 소프트웨어 라이브러리이다.[1] PIL을 통해 간편하게 local의 이미지를 불러오고, 이미지를 볼 수 있고, 그레이 스케일로의 변환, 크기 자르기, 배열을 이미지로 변환, local에 이미지 저장을 수행할 수 있다.

아래는 실습에서 사용되는 다람쥐의 이미지이다. 이 이미지로부터 다람쥐의 머리 이미지, 밝은 머리 이미지, 어두운 머리 이미지를 생성할 것이다.

그림 1 chipmunk.png

[1] Wikipedia, https://ko.wikipedia.org/wiki/Python_Imaging_Library

2. 본론

2.1. chipmunk_head

먼저, chipmunk.png를 프로그램으로 불러와 grayscale로 변환한다. 이때 'L'은 Luminance를 의미한다. 다른 모드로는 'RGB' 등이 있다.

```
im = Image.open('chipmunk.png')
im = im.convert('L')
```

im을 사용하여 다람쥐의 머리 부분을 포함한 부분만 crop하여 chipmunk_head.png에 저장한다.

```
im2 = im.crop((280,150,430,300))
im2.save('chipmunk_head.png','PNG')
```

그림 2 chipmunk_head.png

2.2. chipmunk_head_bright

Im2를 NumPy array로 변환한다. 0에서 255로 갈수록 흰색에 가까워지므로 array의 모든 item에 일괄적으로 값을 증가시켜주면 이미지가 밝아진다. 각 item의 value에 50을 더해 저장한다. 이때 value의 최대값은 255이다.

```
im2_array = np.asarray(im2)
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)
```

변형된 array를 다시 Image로 바꾸고 chipmunk_head_bright.png로 저장한다.

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

그림 3 chipmunk_head_bright.png

2.3. chipmunk_head_dark

어둡게 만드는 것은 반대로 value를 작게 만들면 된다. 여기서는 간단하게 array에 0.5를 곱하여 어둡게 만든다. 하지만 0.5를 곱했을 시 float형으로 변환되어 버리므로 다시 uint8 타입으로 바꾸어 chipmunk_head_dark로 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')
```

그림 4 chipmunk_head_dark.png

2.4. gradient

이번엔 다람쥐의 이미지를 변형시키지 않고 이미지 배열을 NumPy를 이용하여 새로 만들어 내었다. 먼저 arange를 통해 검정에서 흰색까지 그라데이션인 pixel row를 생성한다.

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

tile을 통해 grad로 (256 x 256) 의 그라데이션 이미지를 만든다. Tile의 2번째 인자인 [256, 1]은 1번째 인자인 grad의 items들이 1번 들어가는 배열이 256개인 배열을 만들도록 지시한다.

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

생성한 배열은 int형이기 때문에 uint8로 변환해준 뒤, 이미지로 변환하여 gradient로 저장한다.

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

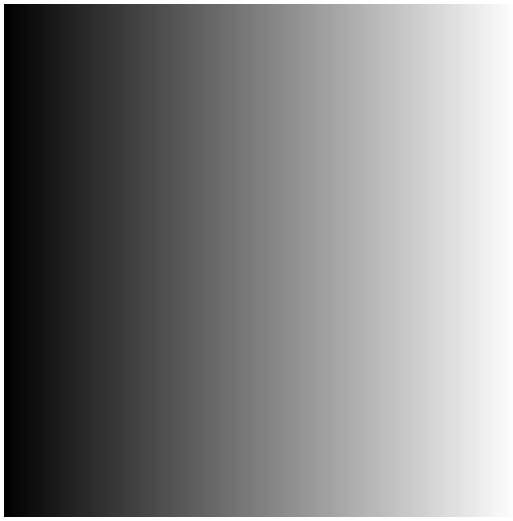

그림 5 gradient.png

3. 결론

이번 실습을 통해 기초적인 디지털 이미지 처리를 수행하였다. 파이썬 환경에서의 이미지 처리를 위한 라이브러리 PIL과 배열을 처리하는데 있어 도움을 주는 NumPy를 사용하였고, 다람쥐의 이미지에 간단한 filtering을 통해 밝게/어둡게 이미지를 변환하였다. 실습에서는 단순히 filtering의 kernel size를 1로하여 각 픽셀의 정보가 하나의 픽셀에만 적용되었다. Kernel size를 늘리고 kernel을 직접 만들어 본다면 filtering을 더욱 잘 이해할 수 있게 될 것이다.