**차량비전시스템  
Assignment 1**

|  |  |
| --- | --- |
| 학번 | 5533082 |
| 전공 | 컴퓨터공학전공 |
| 이름 | 최승환 |

**Q1) 개발환경**

Windows에서 Anaconda, VSCode를 이용하여 python 개발환경 구축

🡪 conda=22.9.0 / python=3.8.16

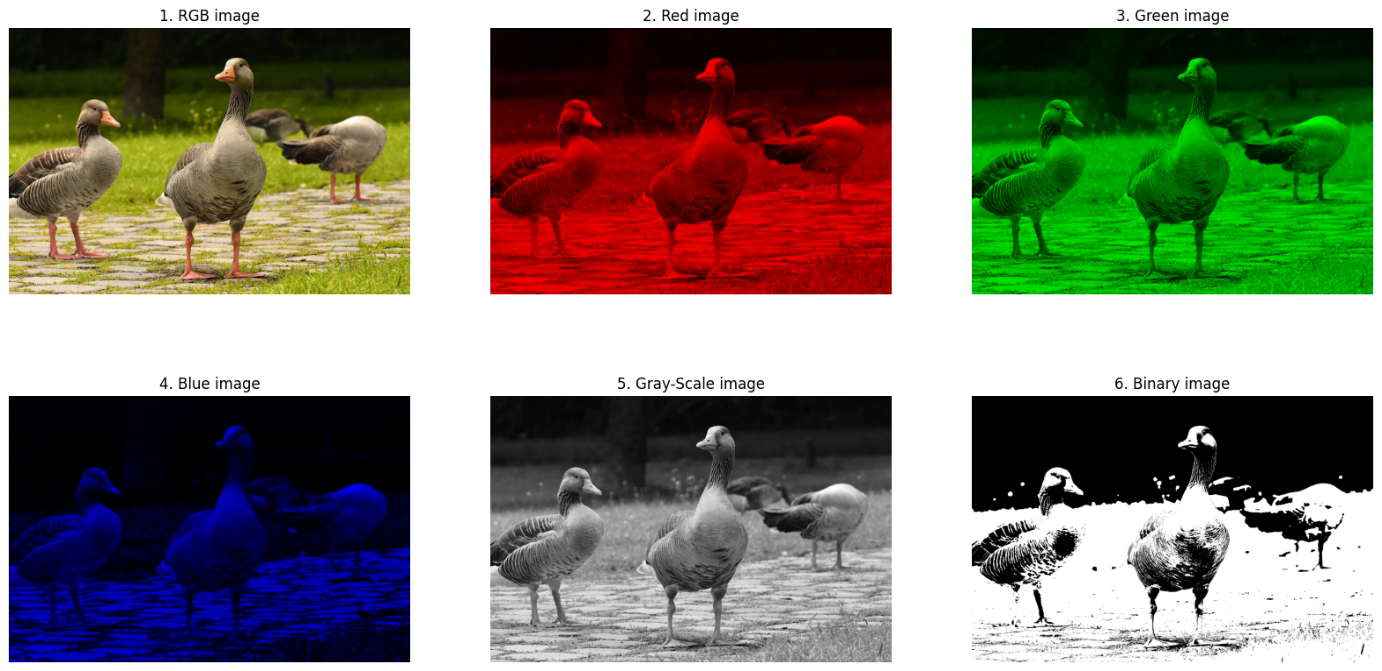
**필요 라이브러리**

**🡪** numpy=1.24.2 / matplotlib=3.7.1 / opencv-python=4.7.0.72

**Q2. 1~6)**

|  |  |  |
| --- | --- | --- |
| 1) RGB 이미지 출력 | 2) R-channel 출력 | 3) G-channel 출력 |
| 4) B-channel 출력 | 5) Gray scale 이미지로 변환 | 6) Binary 이미지로 변환 |

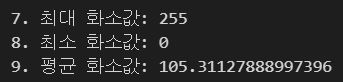
|  |
| --- |
| import cv2  import matplotlib.pyplot as plt  origin\_img = cv2.imread('./img/duck.jpg')  # RGB 이미지로 변환  RGB\_img = cv2.cvtColor(origin\_img, cv2.COLOR\_BGR2RGB)  # 채널 분리  Red\_img, Green\_img, Blue\_img = cv2.split(RGB\_img)  # 그레이스케일 이미지로 변환  Gray\_img = cv2.cvtColor(RGB\_img, cv2.COLOR\_RGB2GRAY)  # 그레이스케일 이미지에 이진화  # ret: 설정 임계값, Binary\_img: 결과 이미지  ret, Binary\_img = cv2.threshold(Gray\_img, 100, 255, cv2.THRESH\_BINARY)  plt.figure( figsize=(20,10) )  # RGB  plt.subplot(2, 3, 1)  plt.title("1. RGB image")  plt.axis("off")  plt.imshow(RGB\_img)  # R-channel  RGB\_img[:,:,0] = Red\_img  RGB\_img[:,:,1] = 0  RGB\_img[:,:,2] = 0  plt.subplot(2, 3, 2)  plt.title("2. Red image")  plt.axis("off")  plt.imshow(RGB\_img)  # G-channel  RGB\_img[:,:,0] = 0  RGB\_img[:,:,1] = Green\_img  RGB\_img[:,:,2] = 0  plt.subplot(2, 3, 3)  plt.title("3. Green image")  plt.axis("off")  plt.imshow(RGB\_img)  # B-channel  RGB\_img[:,:,0] = 0  RGB\_img[:,:,1] = 0  RGB\_img[:,:,2] = Blue\_img  plt.subplot(2, 3, 4)  plt.title("4. Blue image")  plt.axis("off")  plt.imshow(RGB\_img)  # Gray-Scale  plt.subplot(2, 3, 5)  plt.axis("off")  plt.title("5. Gray-Scale image")  # 컬러맵을 지정하지 않으면 임의로 색상이 지정되어 출력된다.  plt.imshow(Gray\_img, cmap='gray')  # Binarization  plt.subplot(2, 3, 6)  plt.axis("off")  plt.title("6. Binary image")  plt.imshow(Binary\_img, cmap='gray')  plt.show() |



**Q2. 7~9)**

7) 5)번 작업을 수행한 후 출력 이미지에서 최대 화소값 출력   
8) 5)번 작업을 수행한 후 출력 이미지에서 최소 화소값 출력   
9) 5)번 작업을 수행한 후 전체 화소값의 평균값 출력

|  |
| --- |
| import numpy as np  print(f"7. 최대 화소값: {np.max(Gray\_img)}")  print(f"8. 최소 화소값: {np.min(Gray\_img)}")  print(f"9. 평균 화소값: {np.mean(Gray\_img)}") |



**Q3-1) 두 이미지 모두 gray scale로 변환**

import cv2

import matplotlib.pyplot as plt

pic1 = cv2.imread('./img/duck.jpg')

pic2 = cv2.imread('./img/cat.jpg')

pic1\_gray = cv2.cvtColor(pic1, cv2.COLOR\_BGR2GRAY)

# 픽셀 resize

pic1\_gray = cv2.resize(pic1\_gray, dsize=(510, 340), interpolation=cv2.INTER\_AREA)

pic2\_gray = cv2.cvtColor(pic2, cv2.COLOR\_BGR2GRAY)

plt.figure( figsize=(10, 10) )

plt.subplot(1,2,1)

plt.title("pic1 gray-scale image")

plt.axis("off")

plt.imshow(pic1\_gray, cmap='gray')

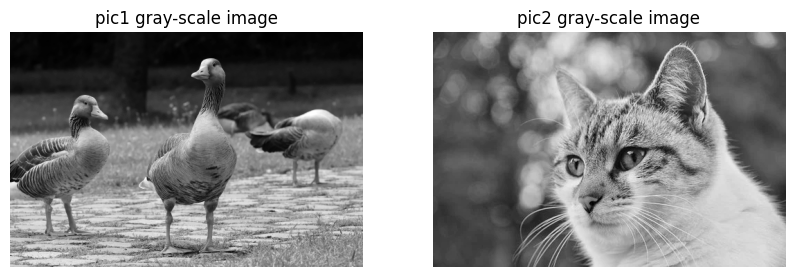
plt.subplot(1,2,2)

plt.title("pic2 gray-scale image")

plt.axis("off")

plt.imshow(pic2\_gray, cmap='gray')

plt.show()



**Q3-2) 두 이미지의 화소값을 모두 더하시오.**



plt.figure( figsize=(10, 10) )

# Saturation

pic\_saturation = cv2.add(pic1\_gray, pic2\_gray) # 자동 Saturation

plt.subplot(1,2,1)

plt.title("2-a. Saturation")

plt.axis("off")

plt.imshow(pic\_saturation, cmap='gray')

# Wrapping

pic\_wrapping = pic1\_gray + pic2\_gray # 자동 Wrapping

plt.subplot(1,2,2)

plt.title("2-b. Wrapping")

plt.axis("off")

plt.imshow(pic\_wrapping, cmap='gray')

plt.show()