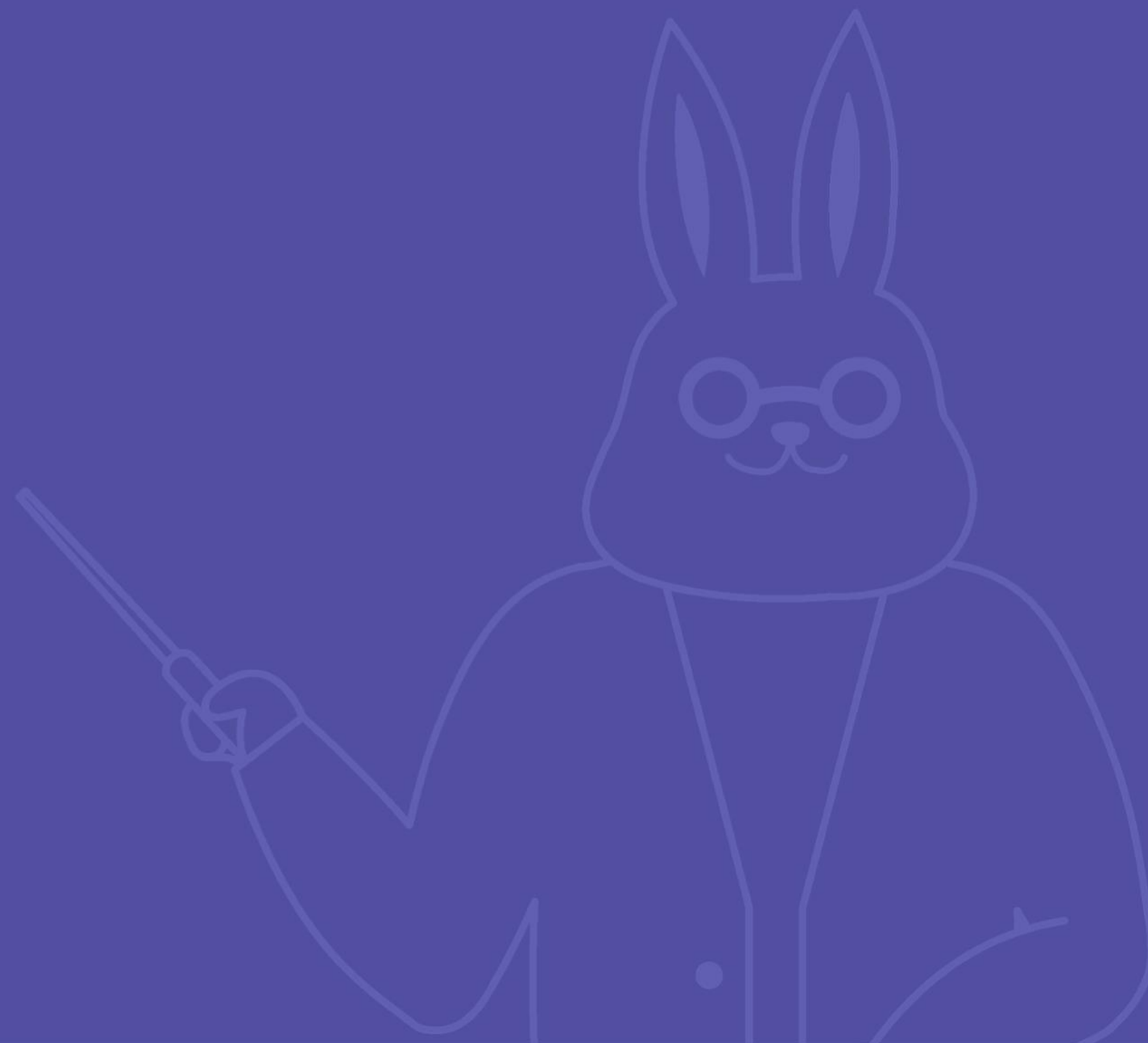




이미지 처리

01 OpenCV로 배우는 영상처리



목차

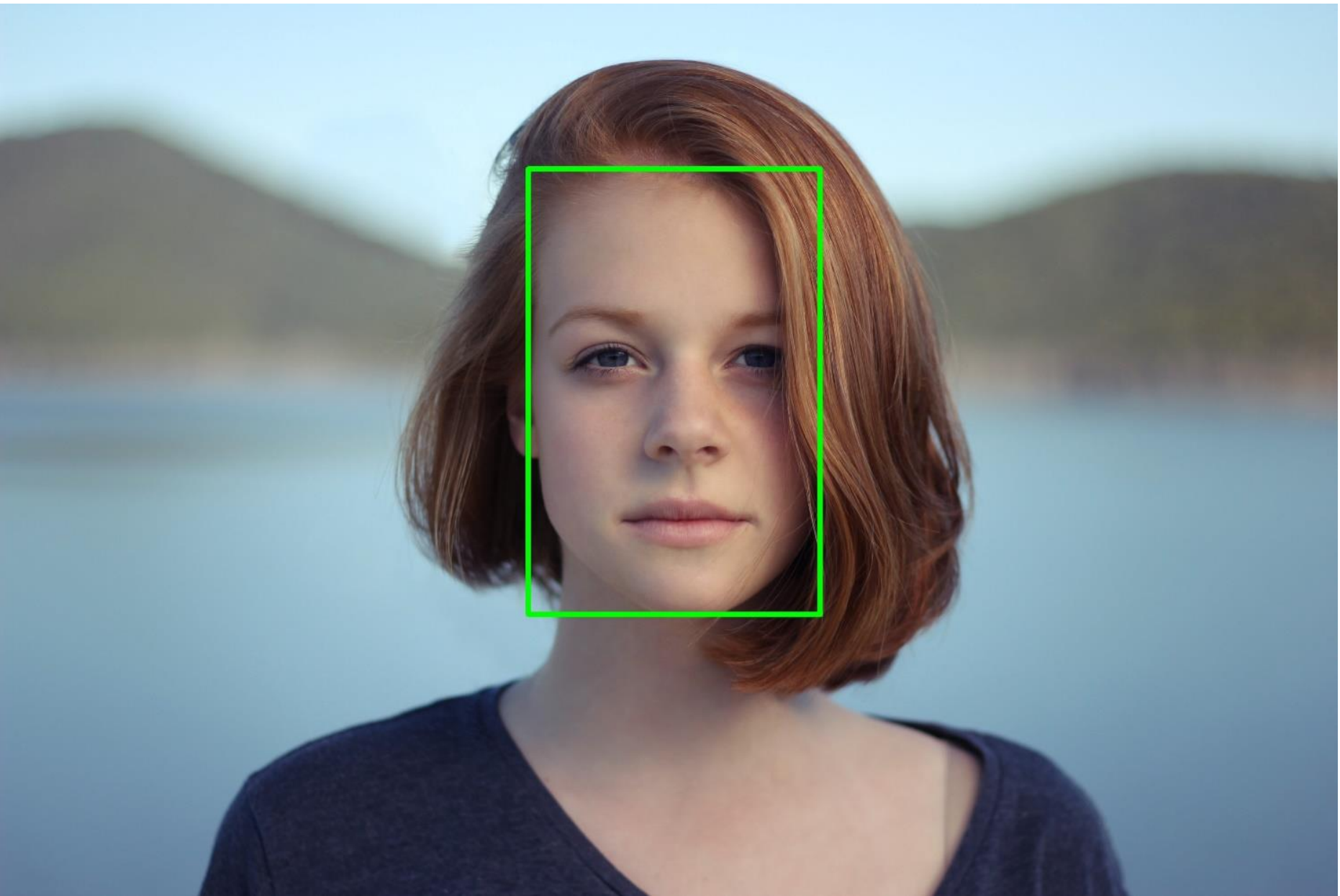
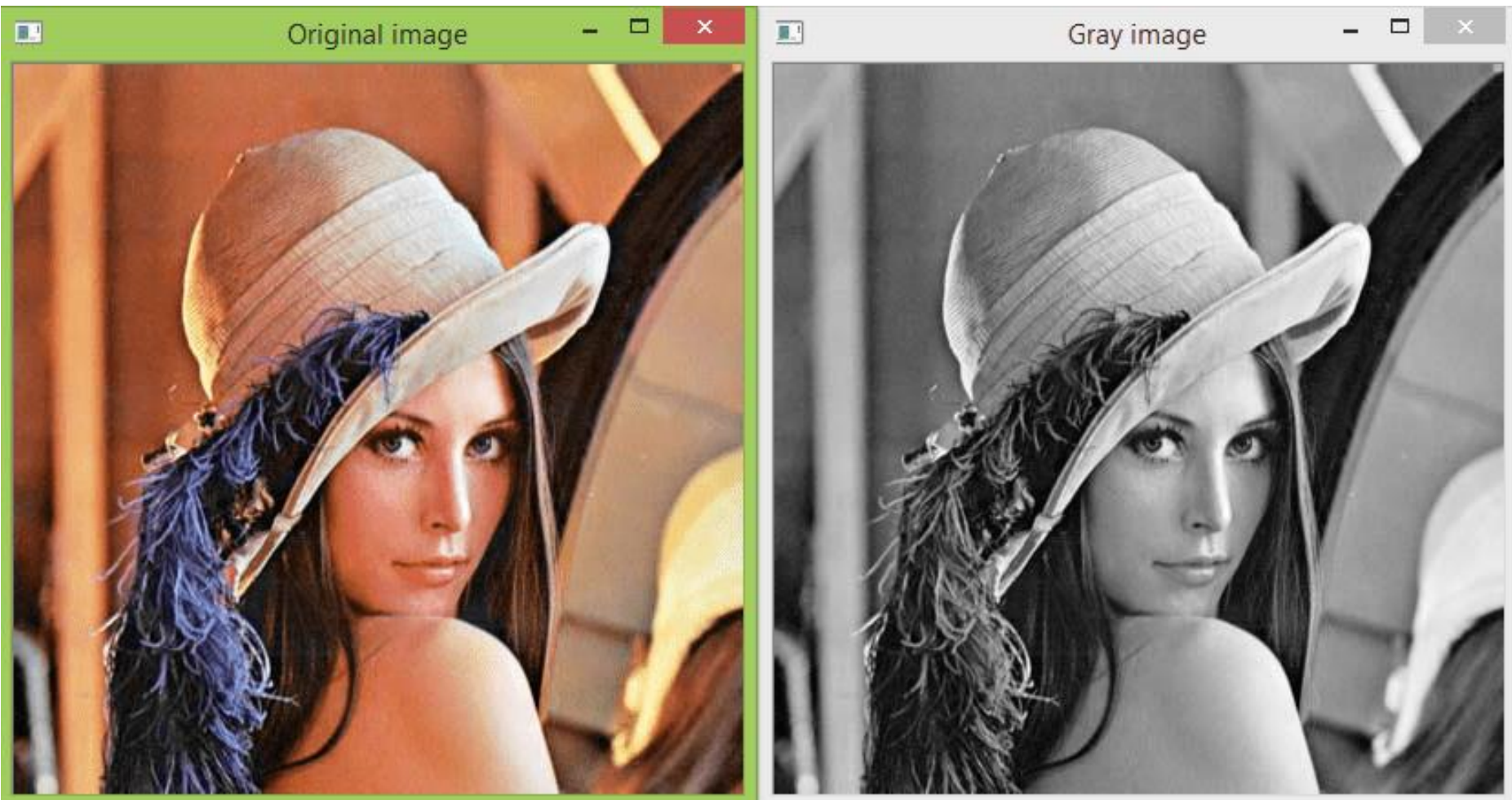
01. 영상 처리와 컴퓨터 비전
02. 영상의 표현
03. 개발환경 구축하기
04. 이미지 I/O 처리
05. 이미지 위에 정보 표시하기
06. OpenCV GUI 인터페이스 활용하기
07. 윤곽선의 이해
08. 회선처리

01

영상 처리와 컴퓨터 비전



✓ 영상 처리란



입력 영상을 원하는 목적 영상으로 만들거나

영상의 특성을 얻어내는 것

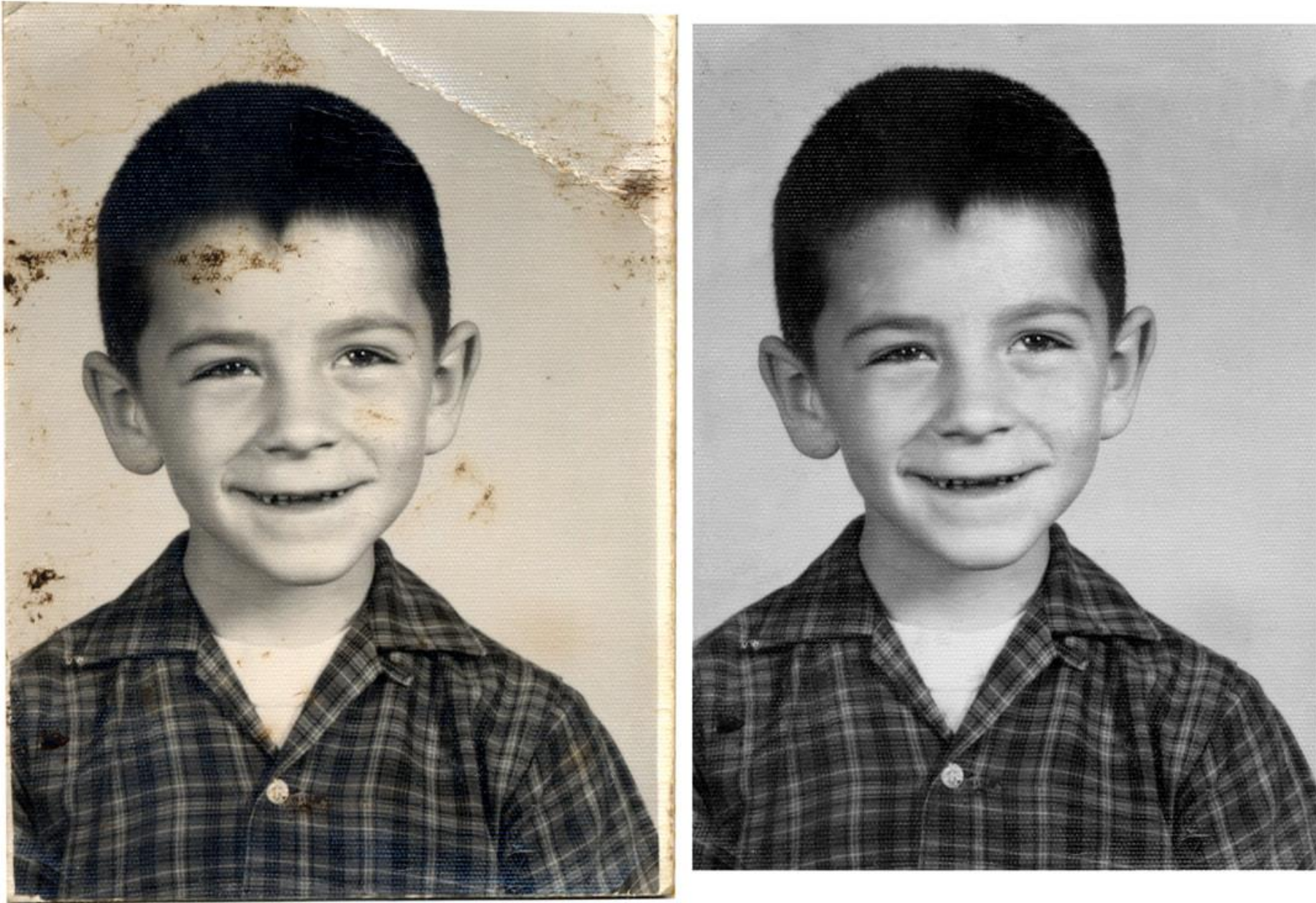
✔ 저수준 영상 처리와 고수준 영상 처리

저수준 영상 처리	고수준 영상 처리
출력의 결과가 영상	출력의 결과가 영상의 특성 또는 영상을 해석한 결과
영상 개선 영상 복원 영상 분석 영상 압축 ...	영상 인식 특징 추출 영상 분할 ...

✓ 저수준 영상 처리의 예



개선



복원

✓ 저수준 영상 처리의 예



original image

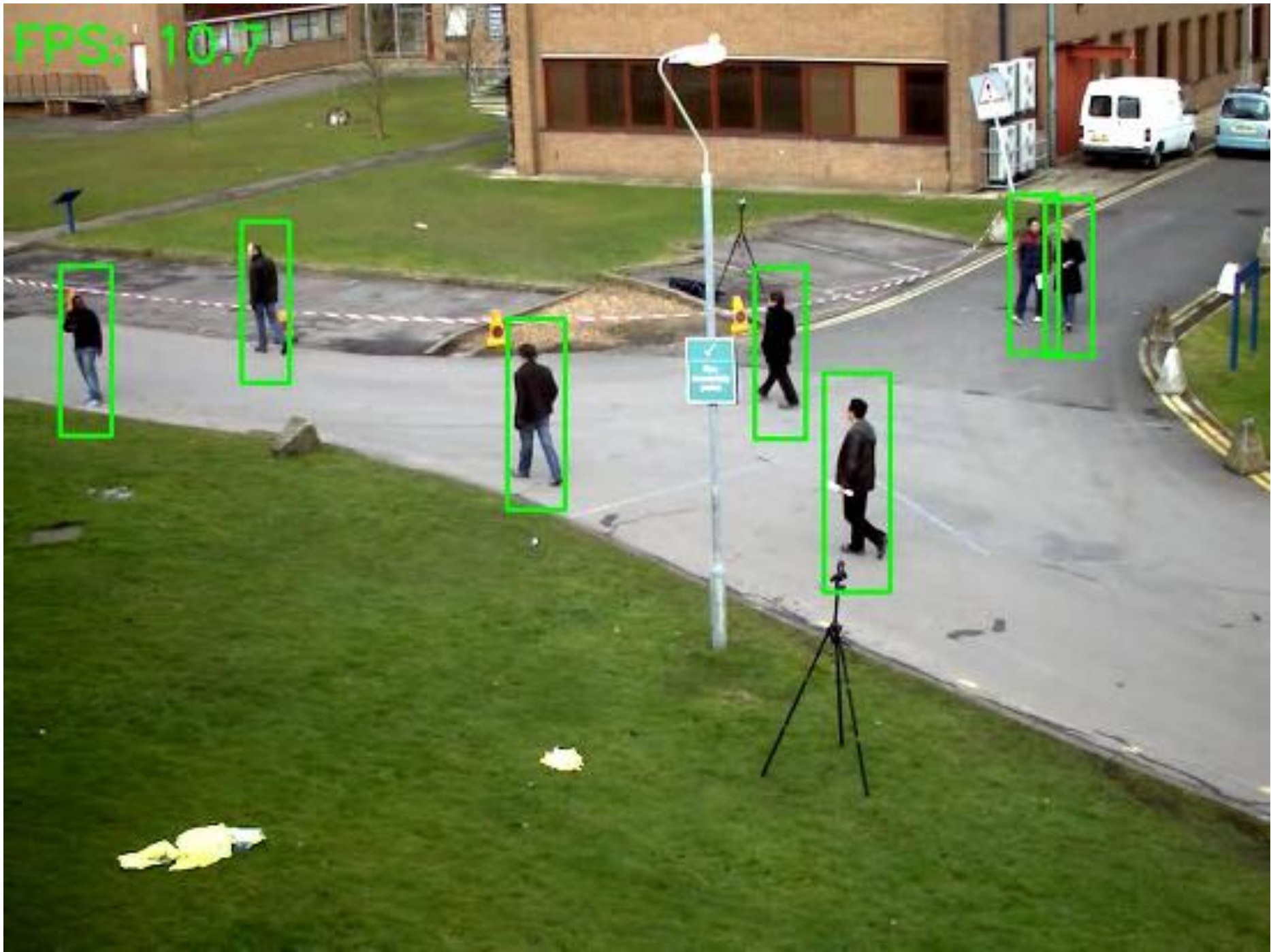


lossy JPEG format
with "artifacts"

분석(윤곽선 검출)

압축

✓ 고수준 영상 처리의 예



객체 인식



배경 분할

✔ 워터마킹, 머신 비전, 영상 합성

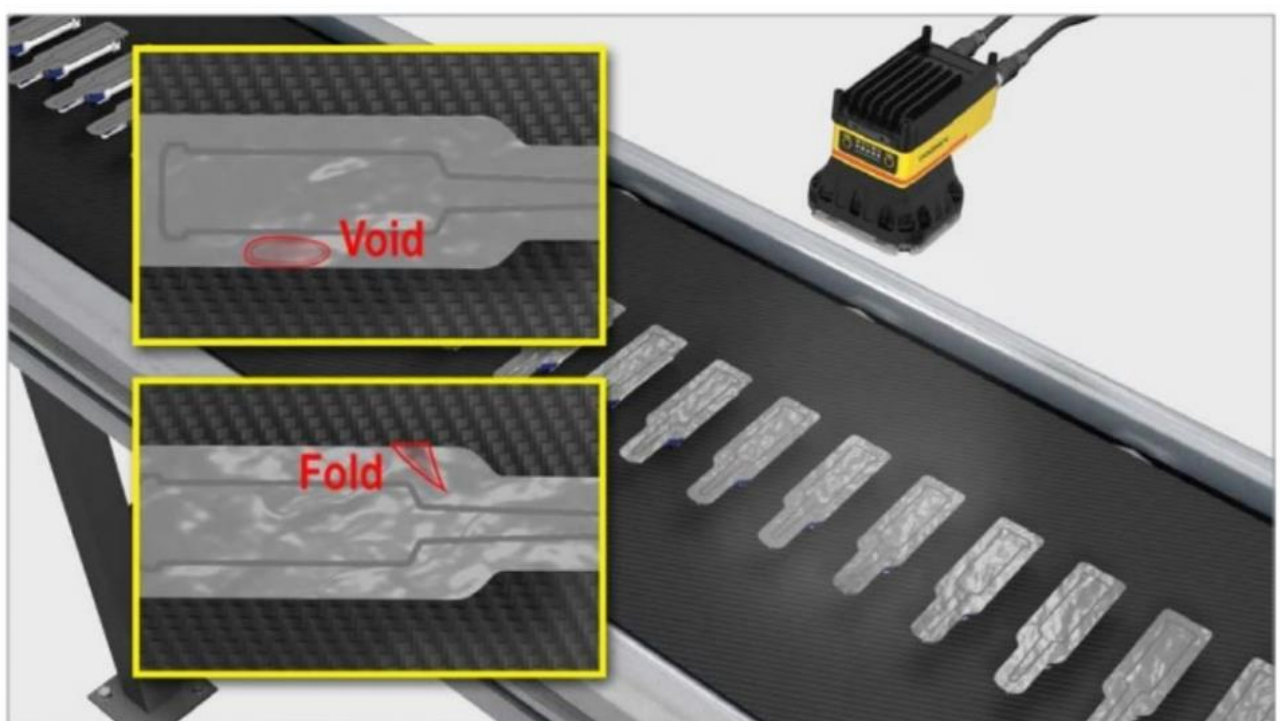
Before Watermarking



After Watermarking



워터마크 삽입으로
저작권 보호



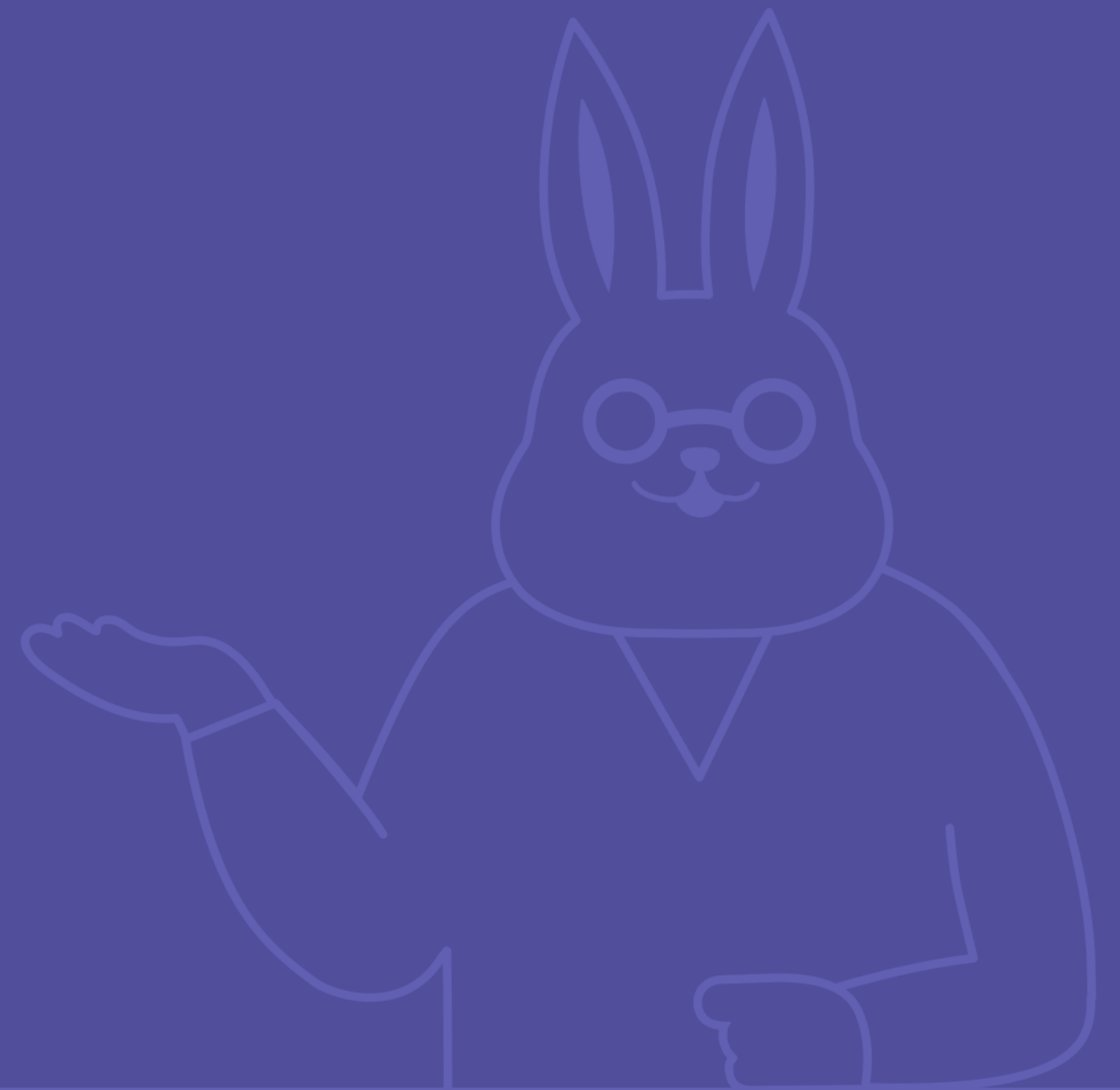
공장 라인의
불량품 검출



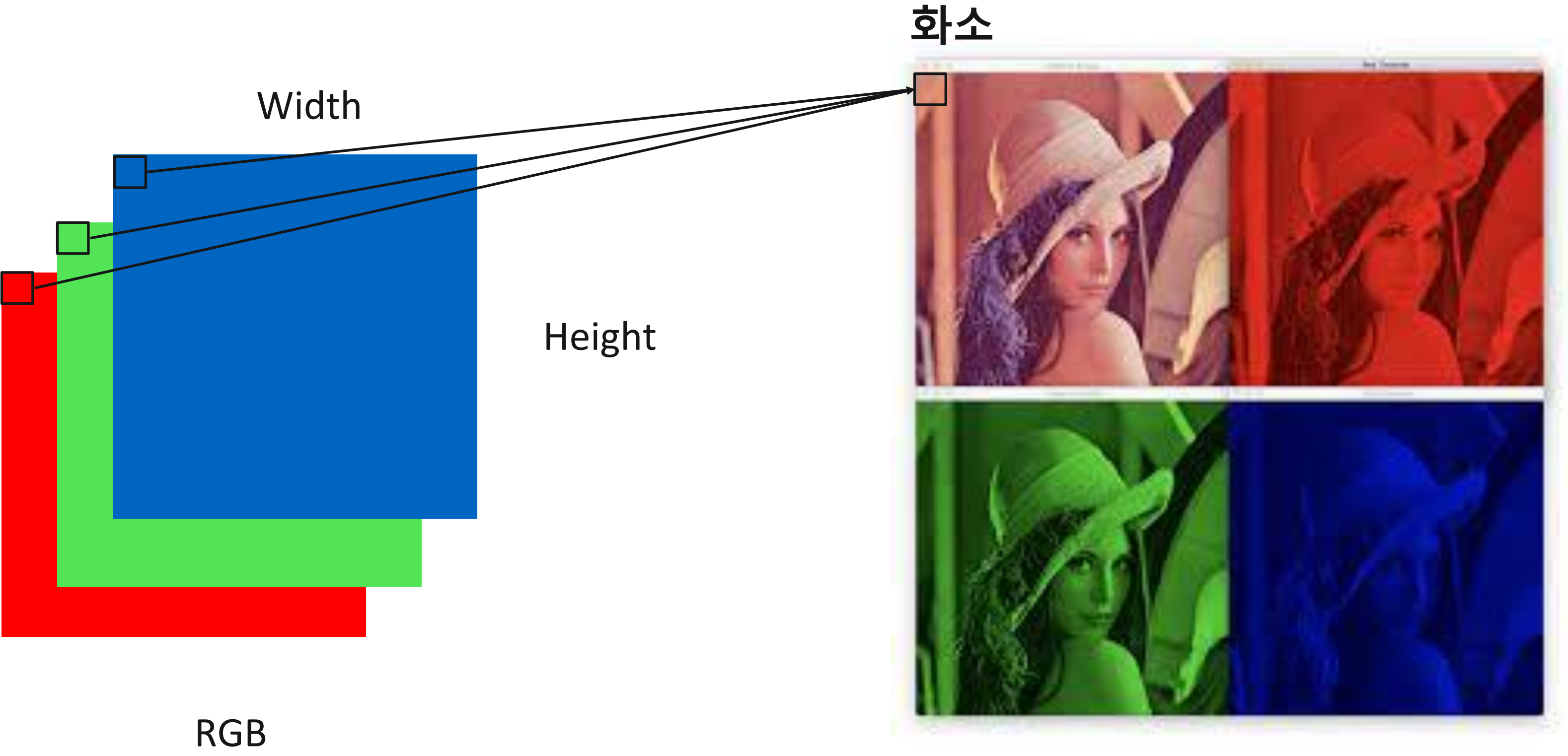
미디어에
효과 입히기

02

영상의 표현

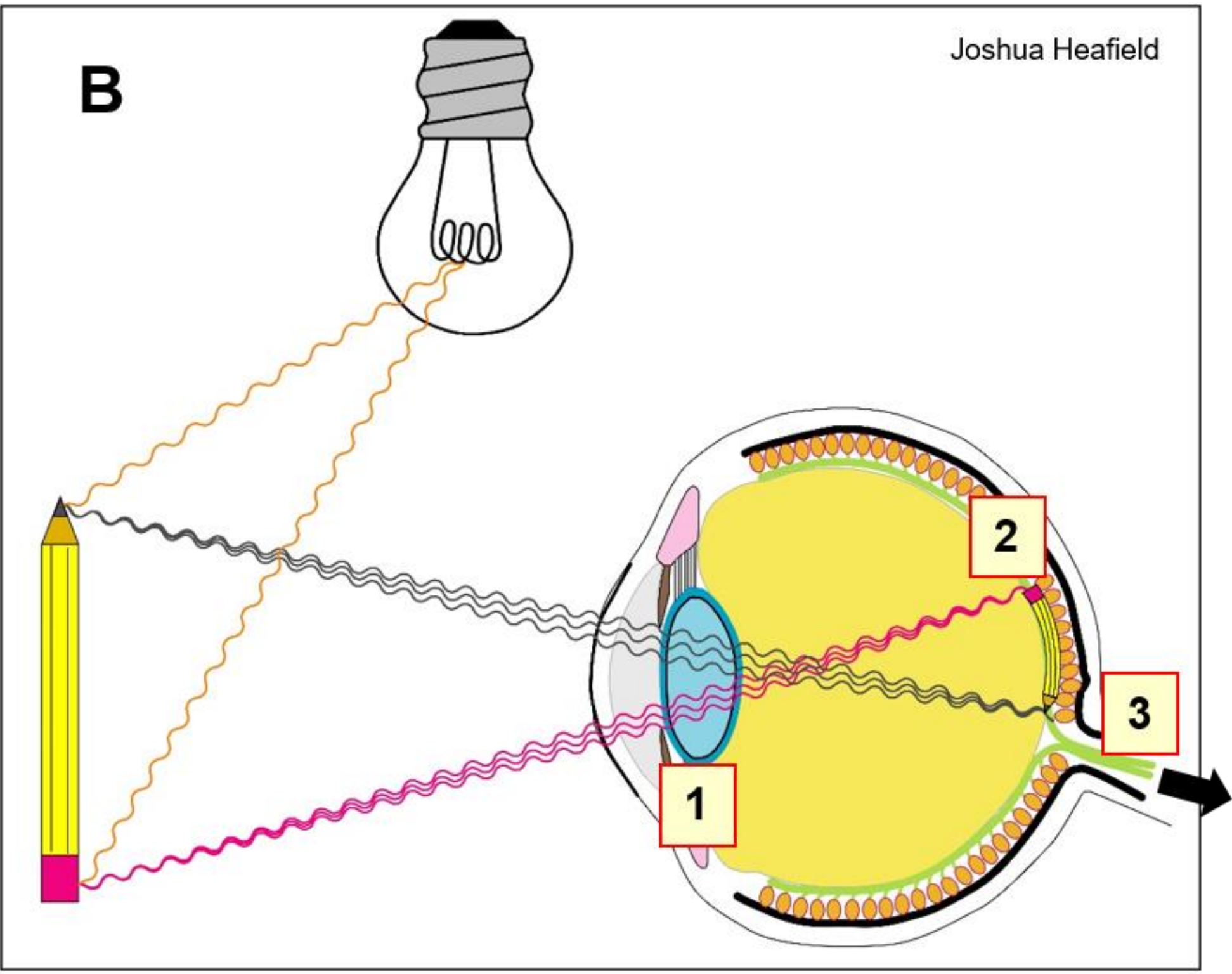
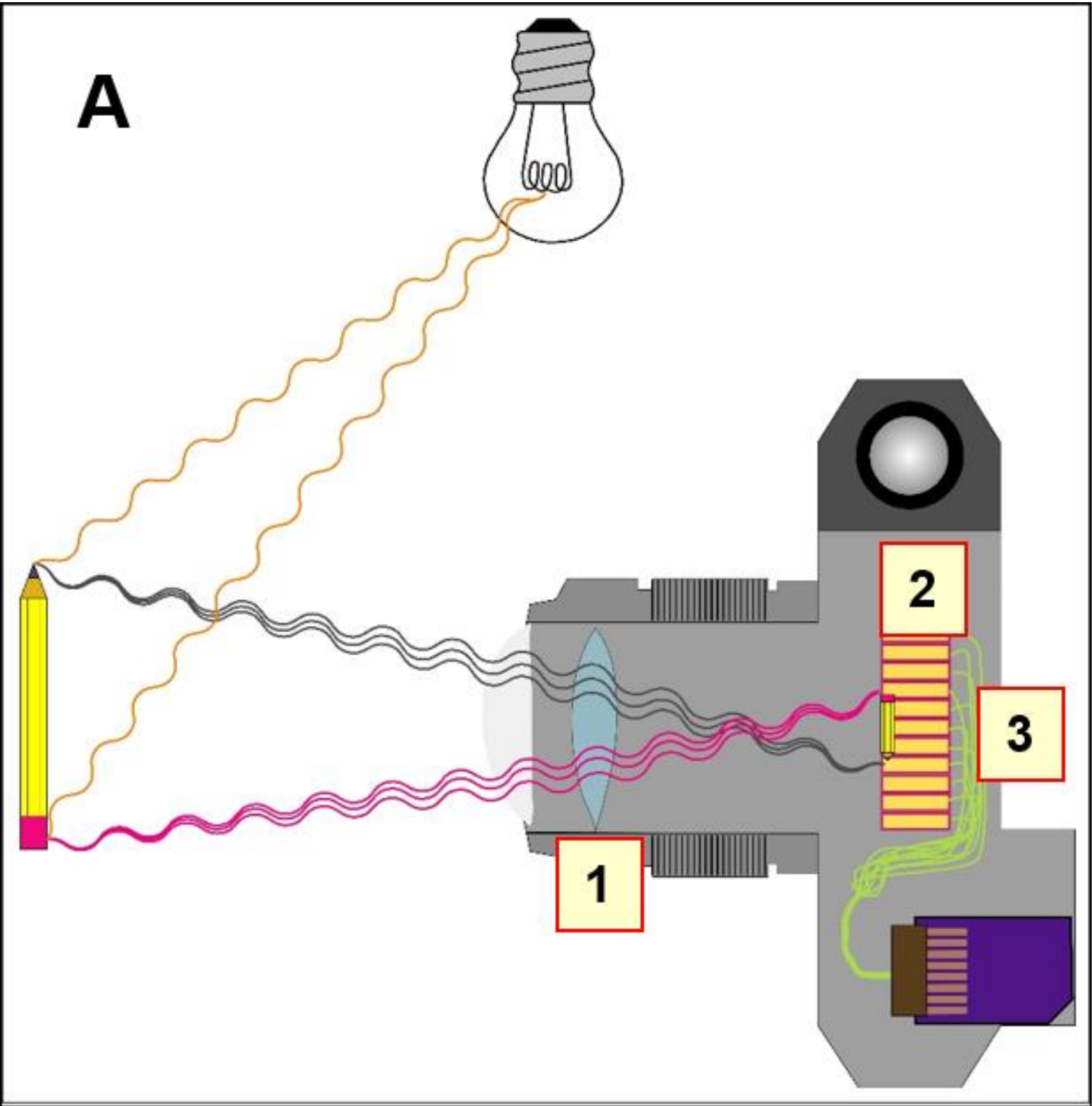


✓ RGB 컬러모델

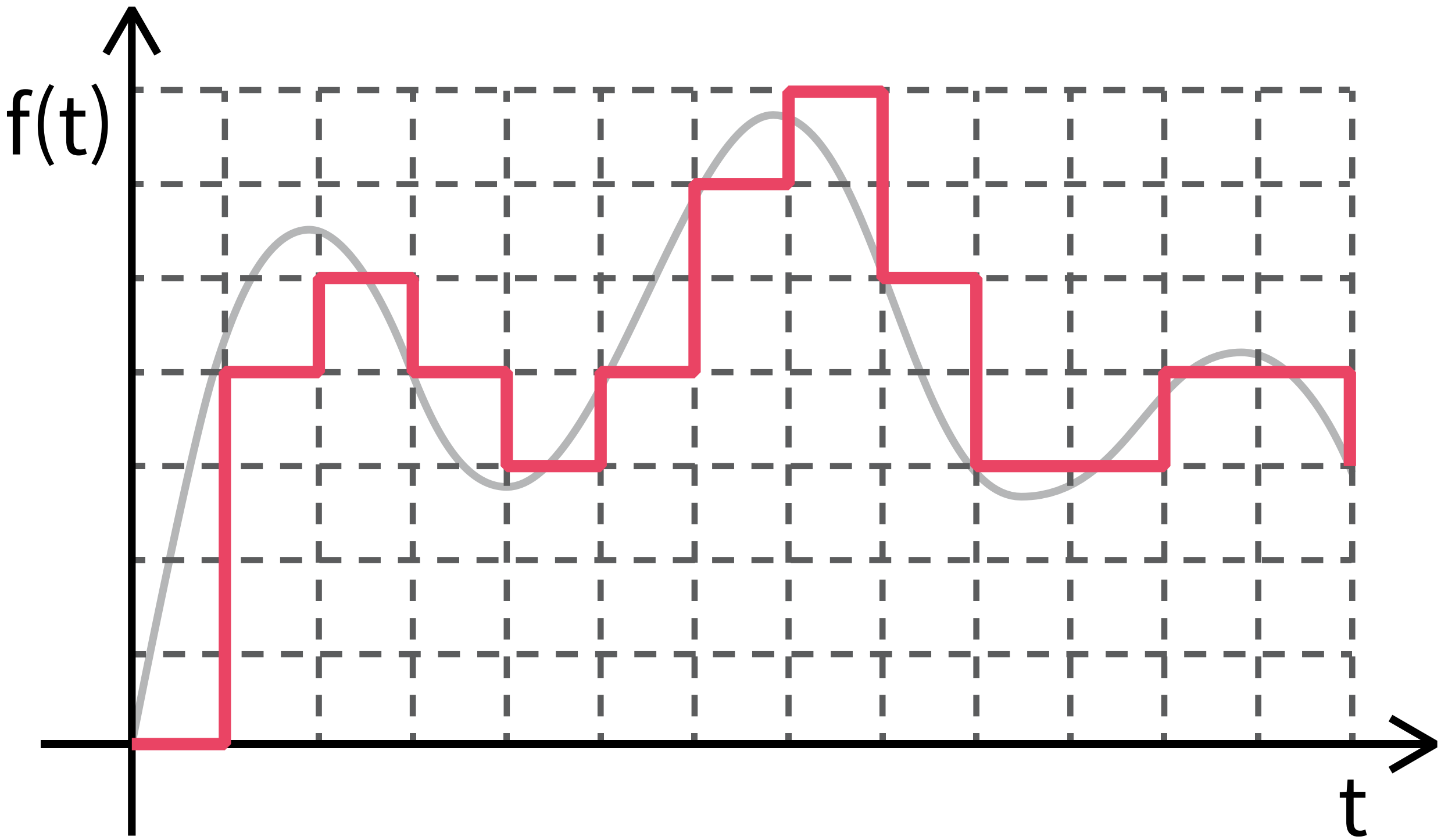


색상(채널), 명도, 해상도

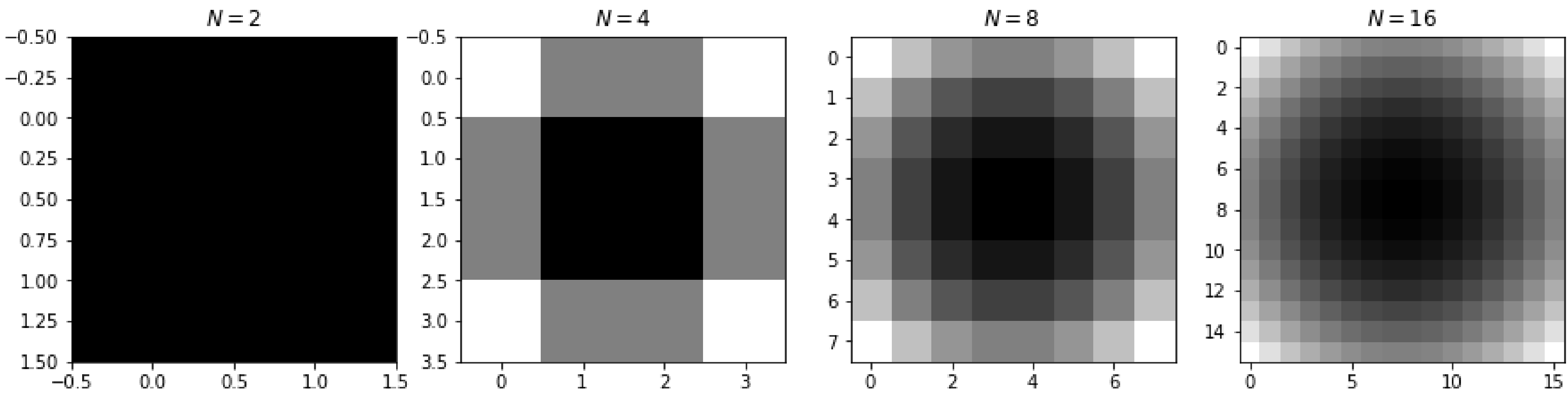
✓ 아날로그와 디지털 신호



✓ 아날로그와 디지털 신호



✔ 디지털 신호로 표현



표본화

✓ 디지털 신호로 표현



Original
(256 colors)



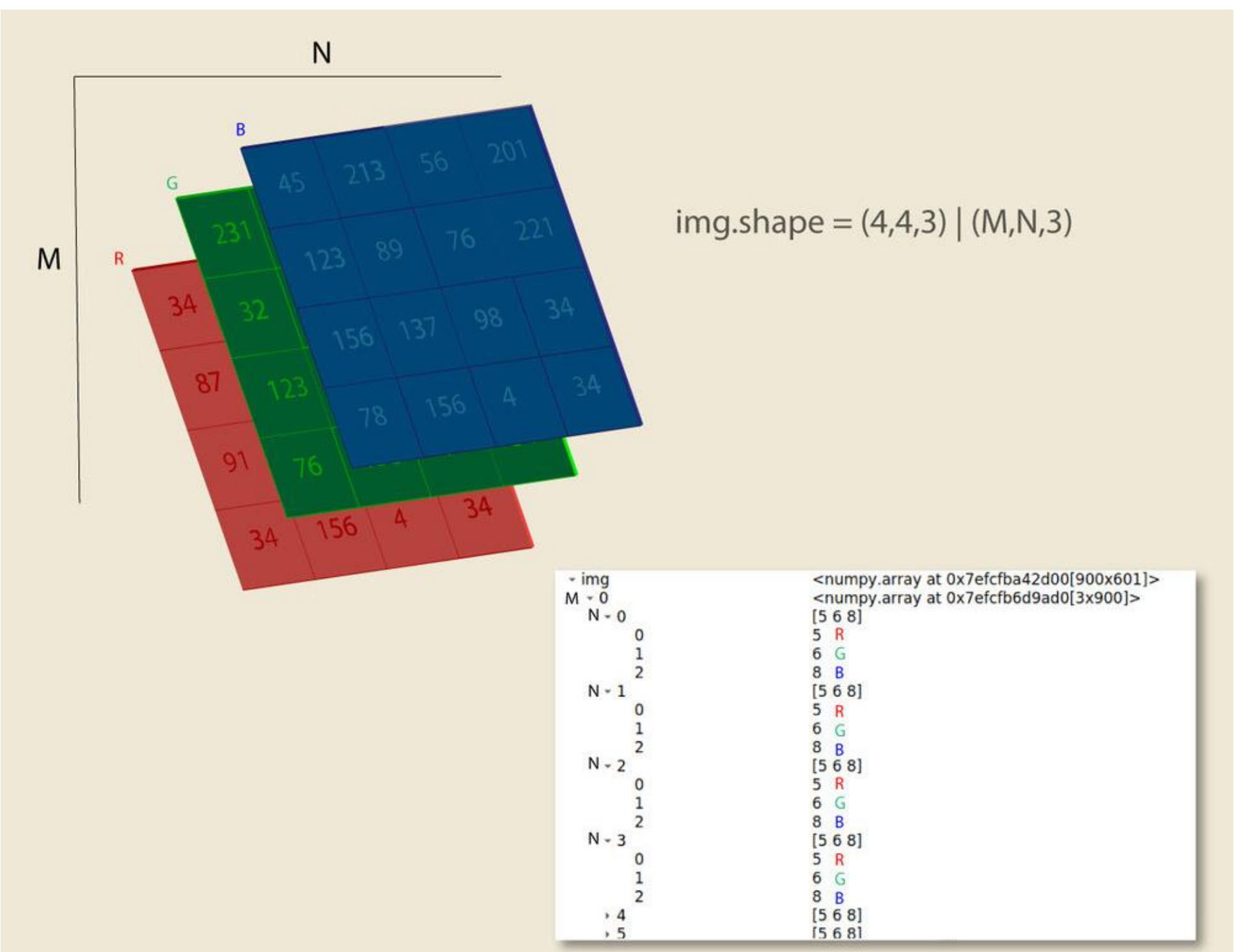
8 colors



4 colors

양자화

✓ OpenCV가 영상을 표현하는 방법



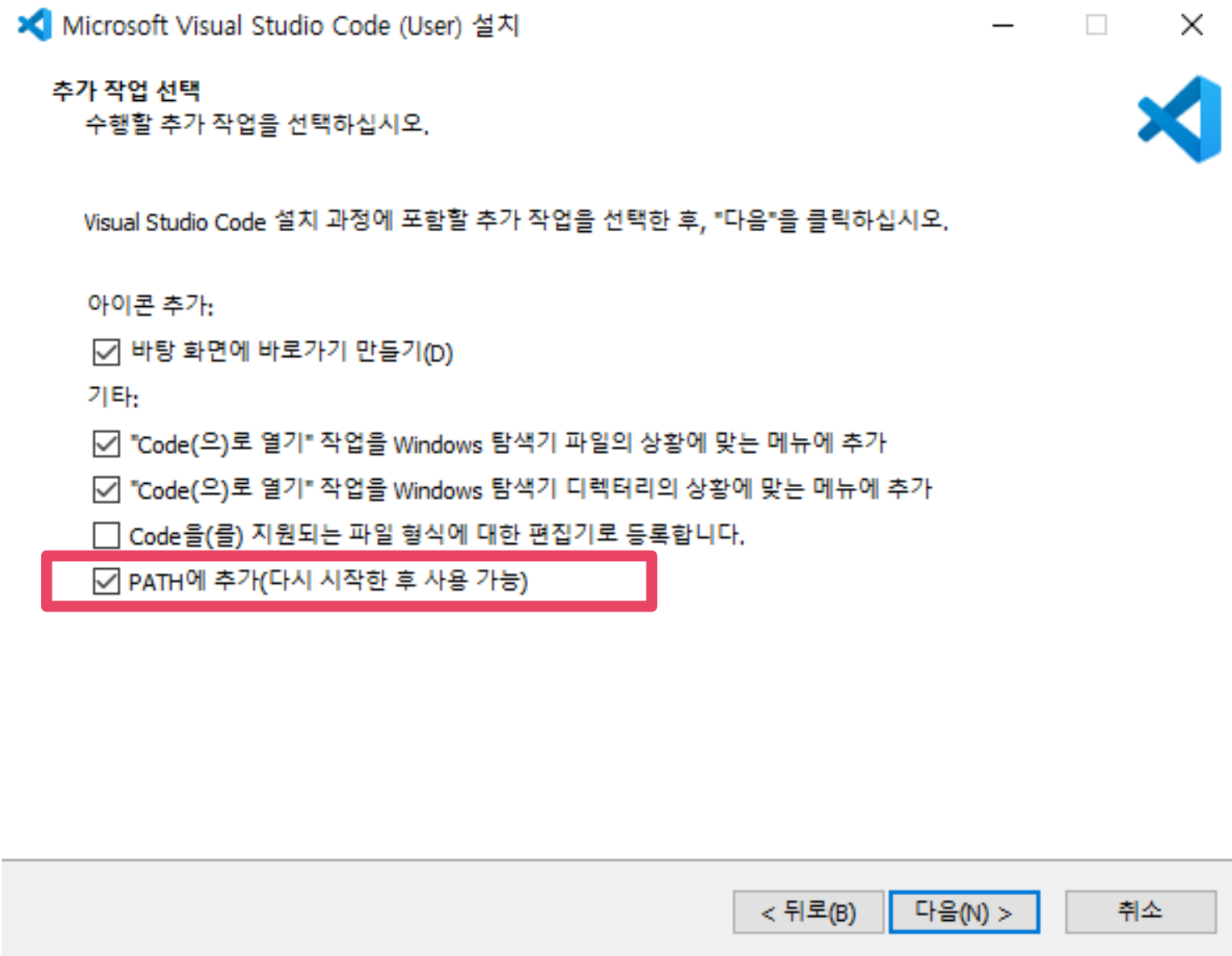
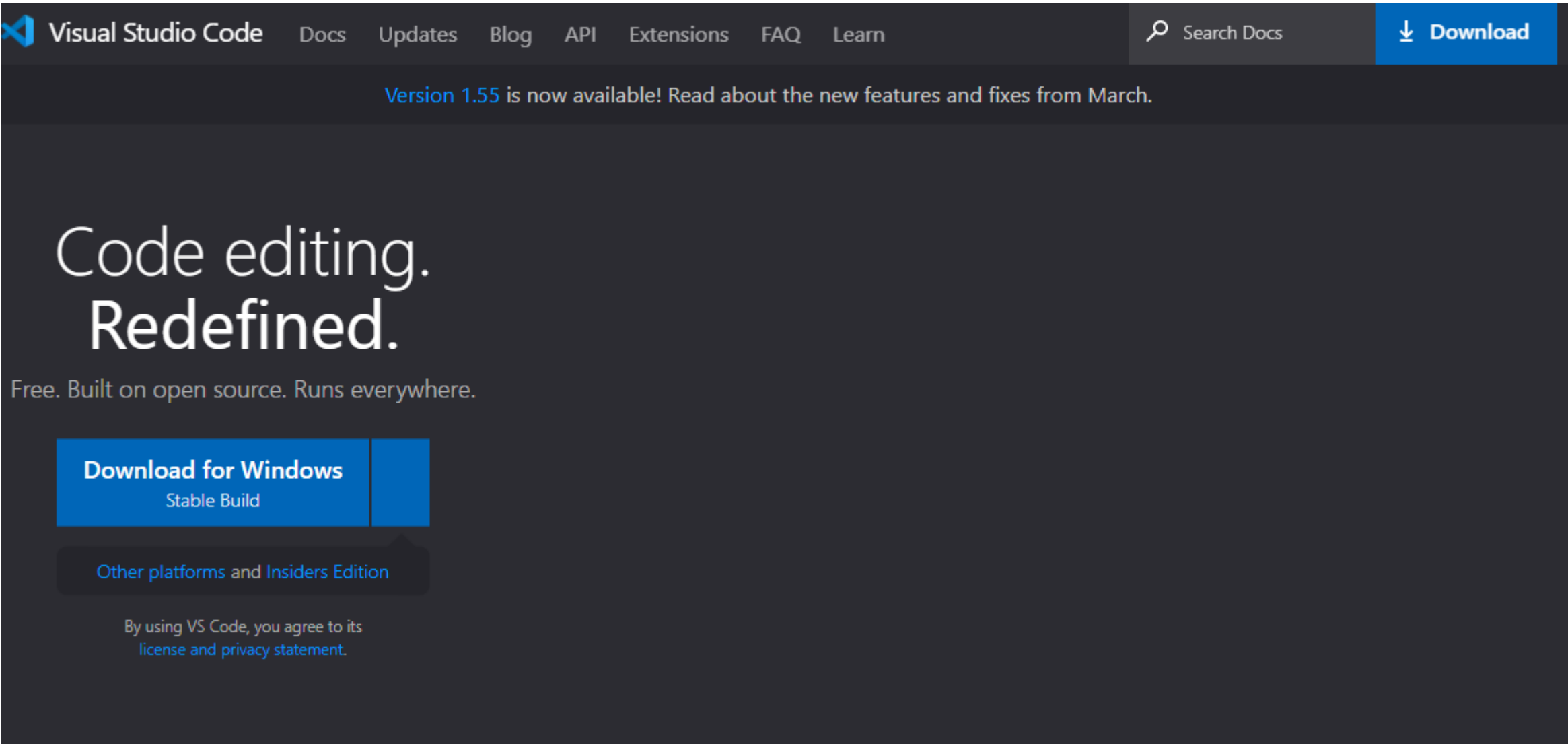
unit8로 화소 표현
0~255 사이의 값으로 명도 표현

03

개발환경 구축하기

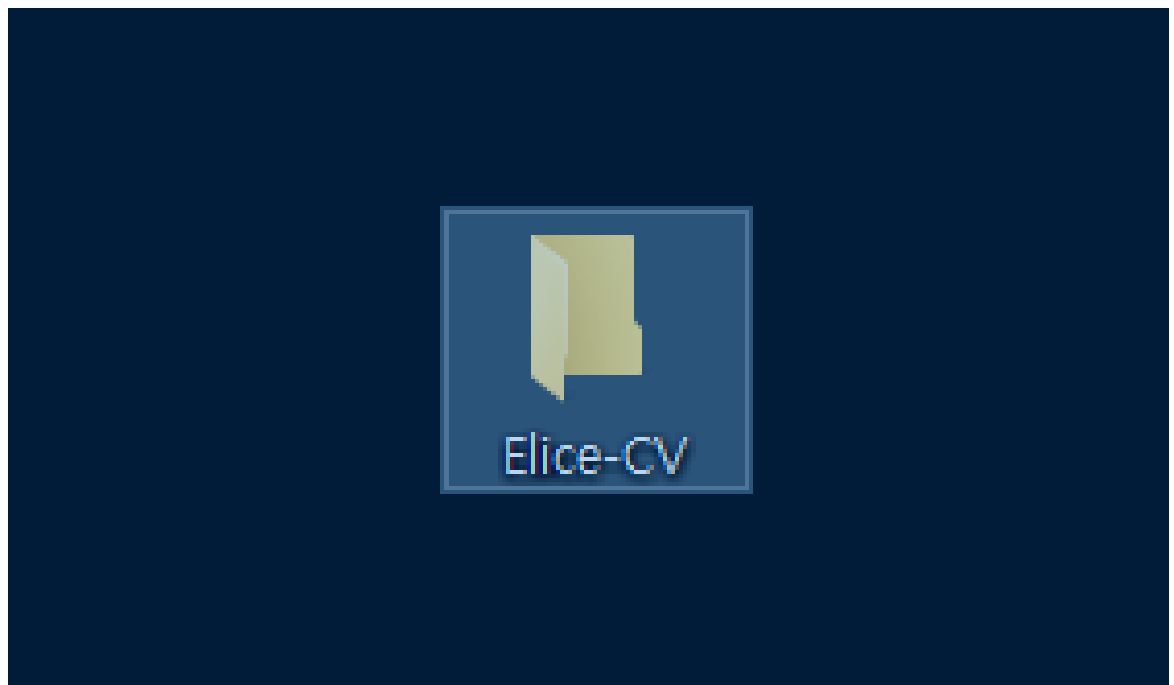
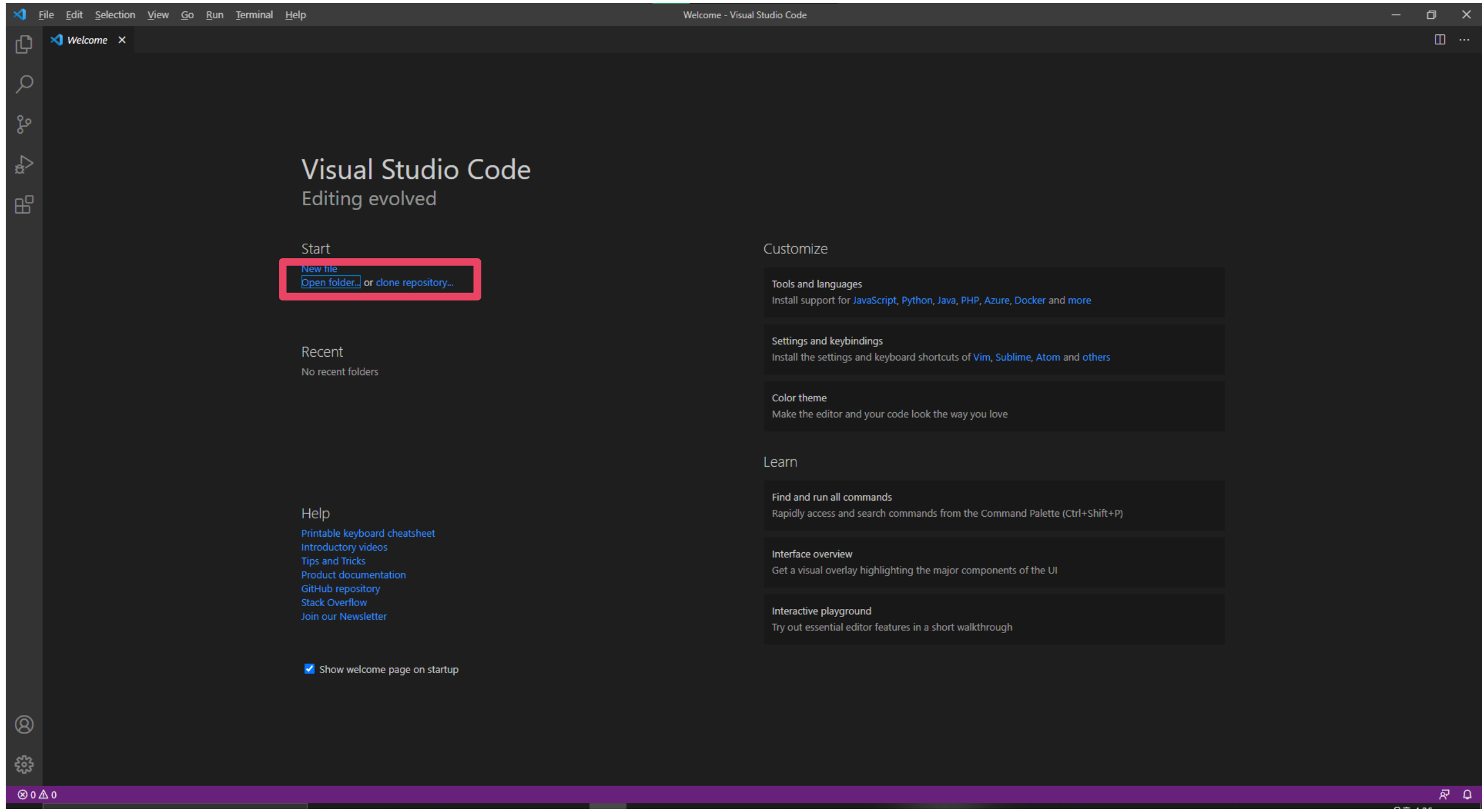


Visual Studio Code 설치하기



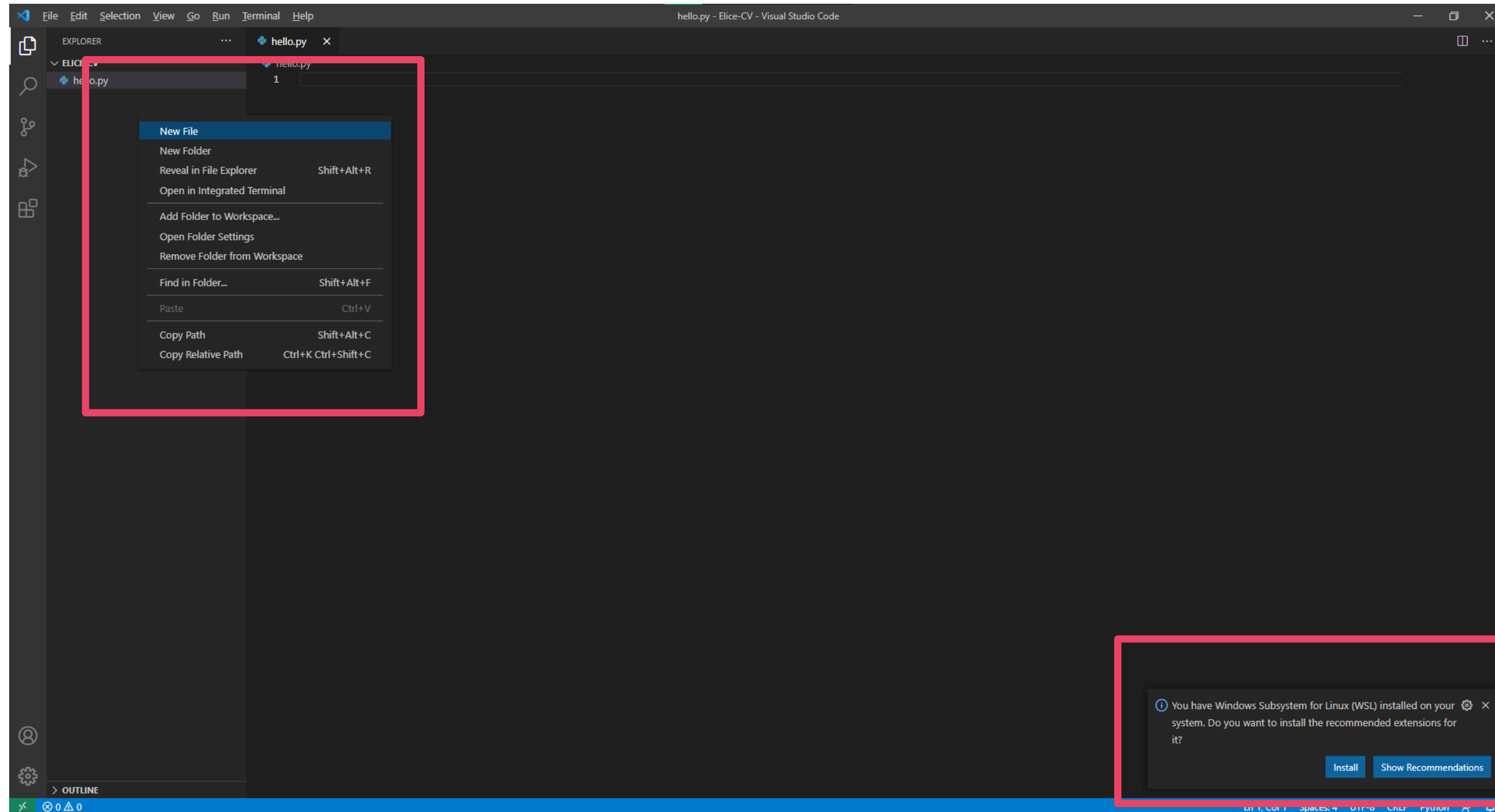
VS Code를 다운받고 설치 후 설치 옵션 중 PATH에 추가 옵션을 체크

✓ 프로젝트 생성하기



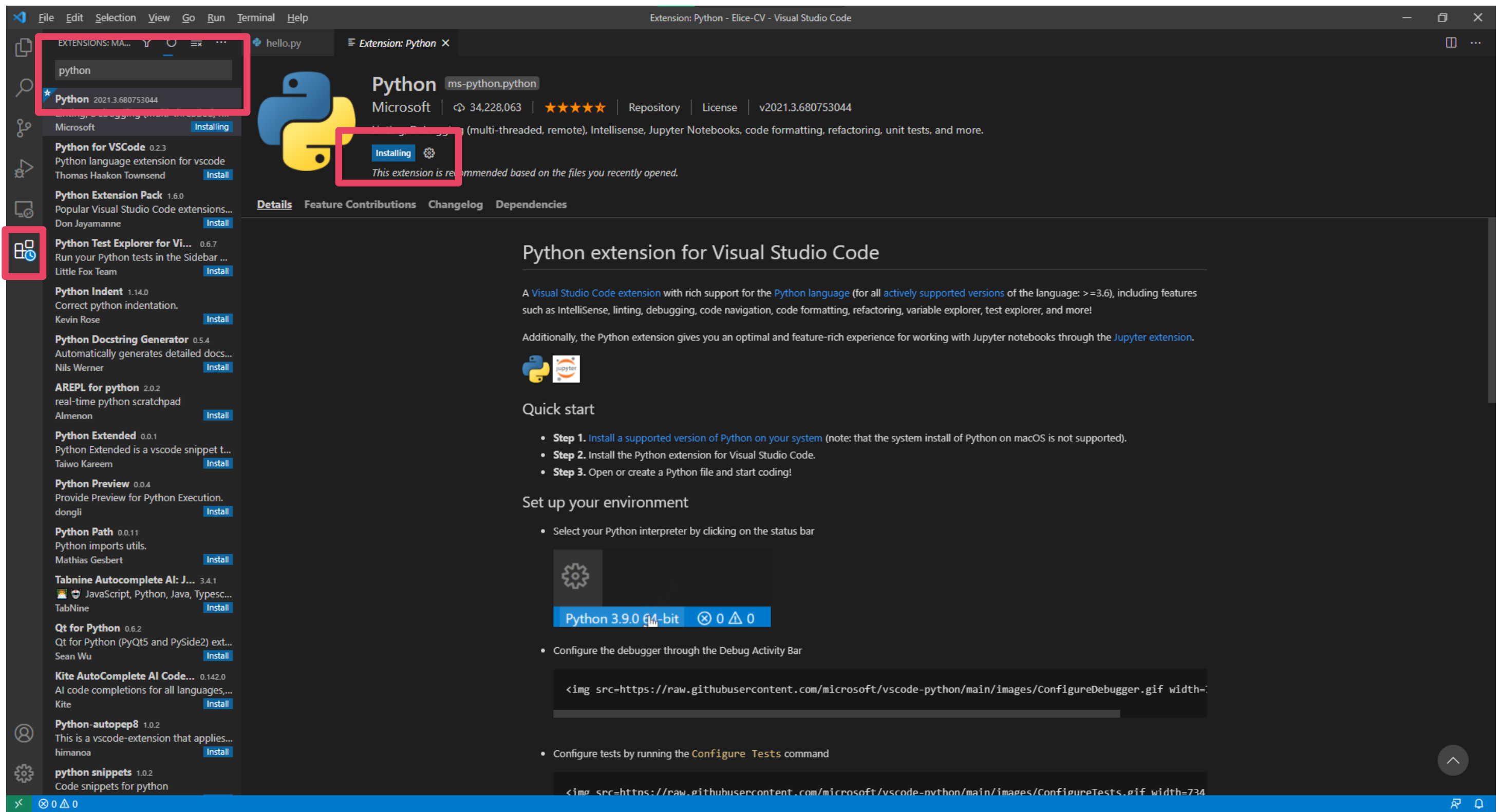
바탕화면에 프로젝트 폴더 만들고 워크 스페이스 지정하기

✓ 파이썬 코드 파일 만들기



오른쪽 마우스 버튼을 눌러 파이썬 파일을 추가
파이썬이 설치 되어있지 않으면 VS Code에서 자동으로 설치할 것을 권고

✓ 파이썬 인터프리터 설치하기



CTRL + SHIFT + X 또는 좌측의 확장기능 설치 아이콘을 눌러서 파이썬을 설치

✓ 주석 단축키

Example

```
"""  
Hello Computer Vision world!  
"""  
  
# print('hello world', end=' - jisu.choi - \n')  
# print('hello world', end=' - jisu.choi - \n')
```

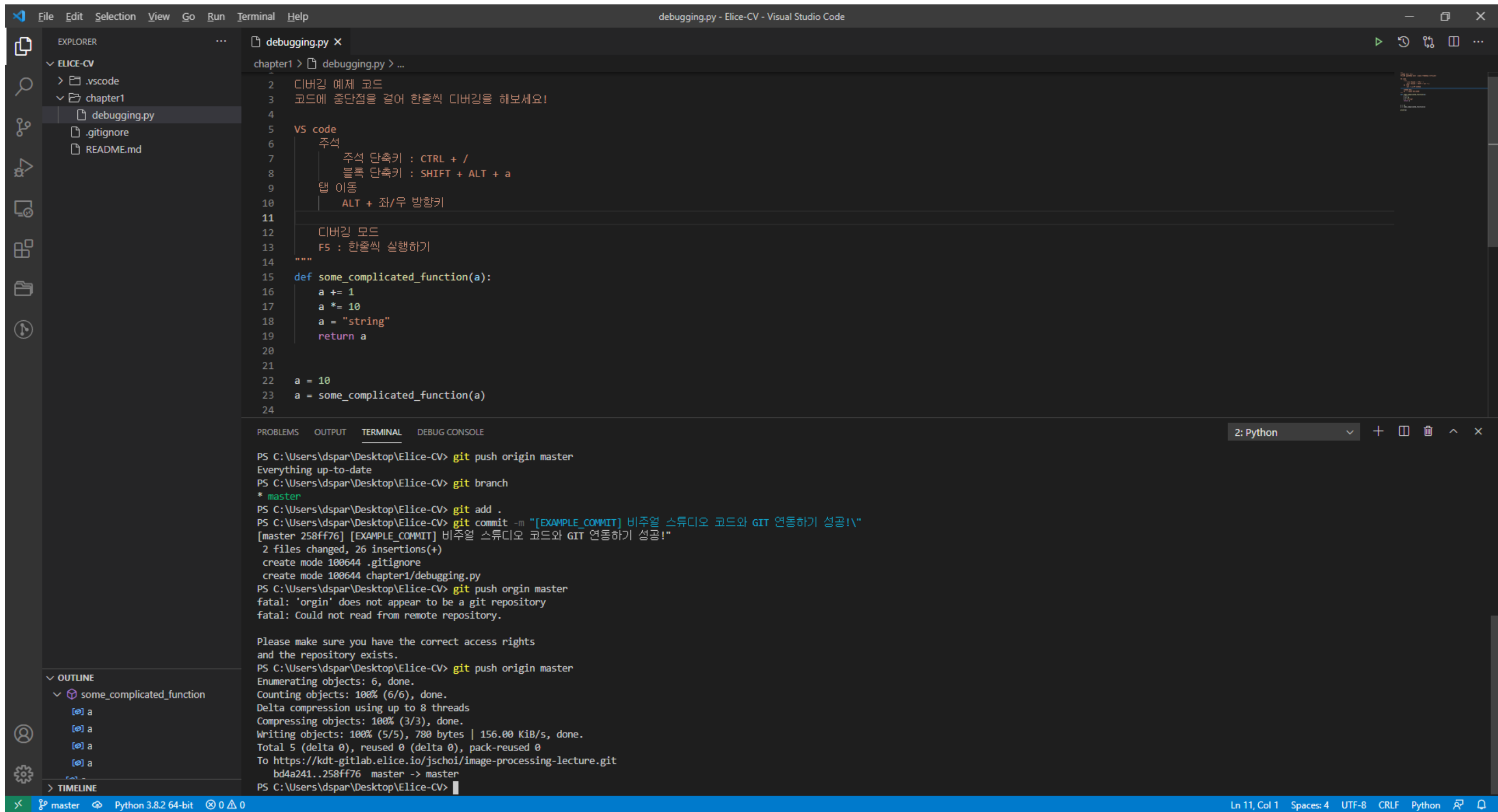
라인 주석 : 코드를 주석처리 할 때 사용

CTRL + /

블록 주석 : 코드의 설명문을 달거나, 주석 처리하고 싶은 코드 범위가 길 때 사용

SHIFT + ALT + A

✓ 터미널 실행시키기



단축키: CTRL + `

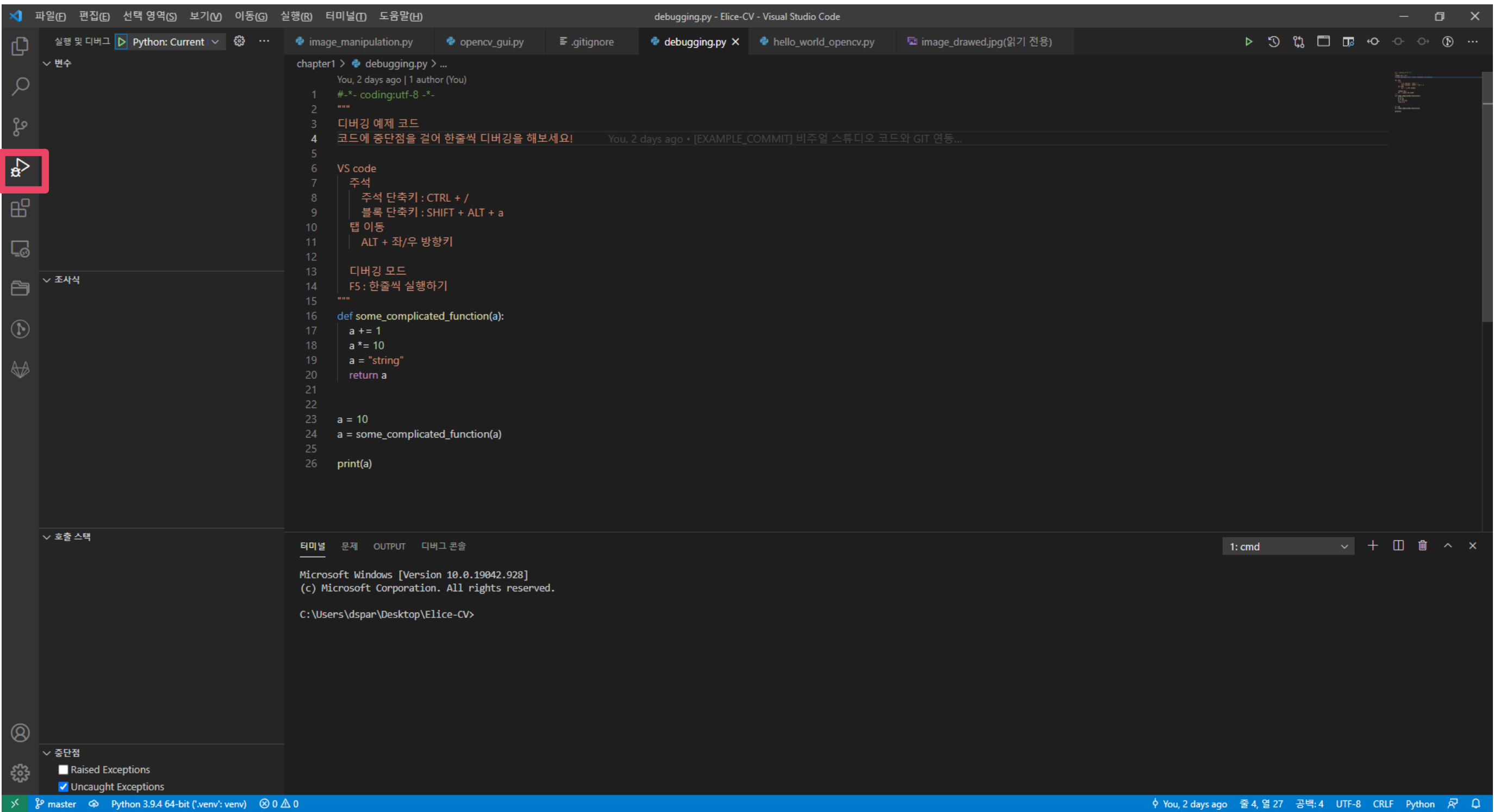
✓ 코드 디버깅하기 : 예제 코드 붙여넣기

Example

```
def some_complicated_function(a):  
    a += 1  
    a *= 10  
    a = "string"  
    return a  
  
a = 10  
a = some_complicated_function(a)  
  
print(a)
```

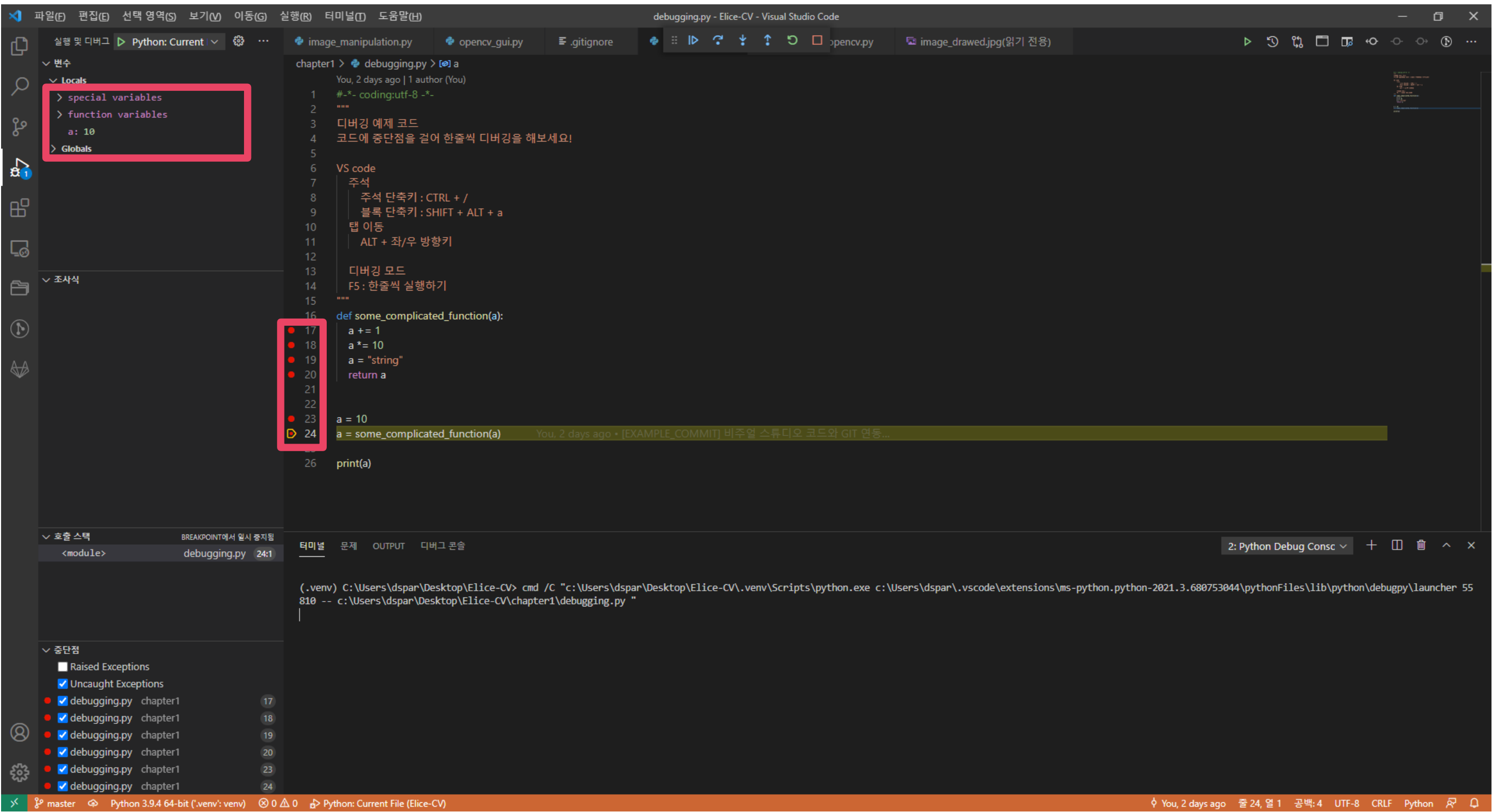
개발을 하다 보면 코드와 로직이 복잡해지기
때문에 코드를 추적하여 어디서 잘못된 값이
들어가는지 확인 필요

✓ 코드 디버깅하기



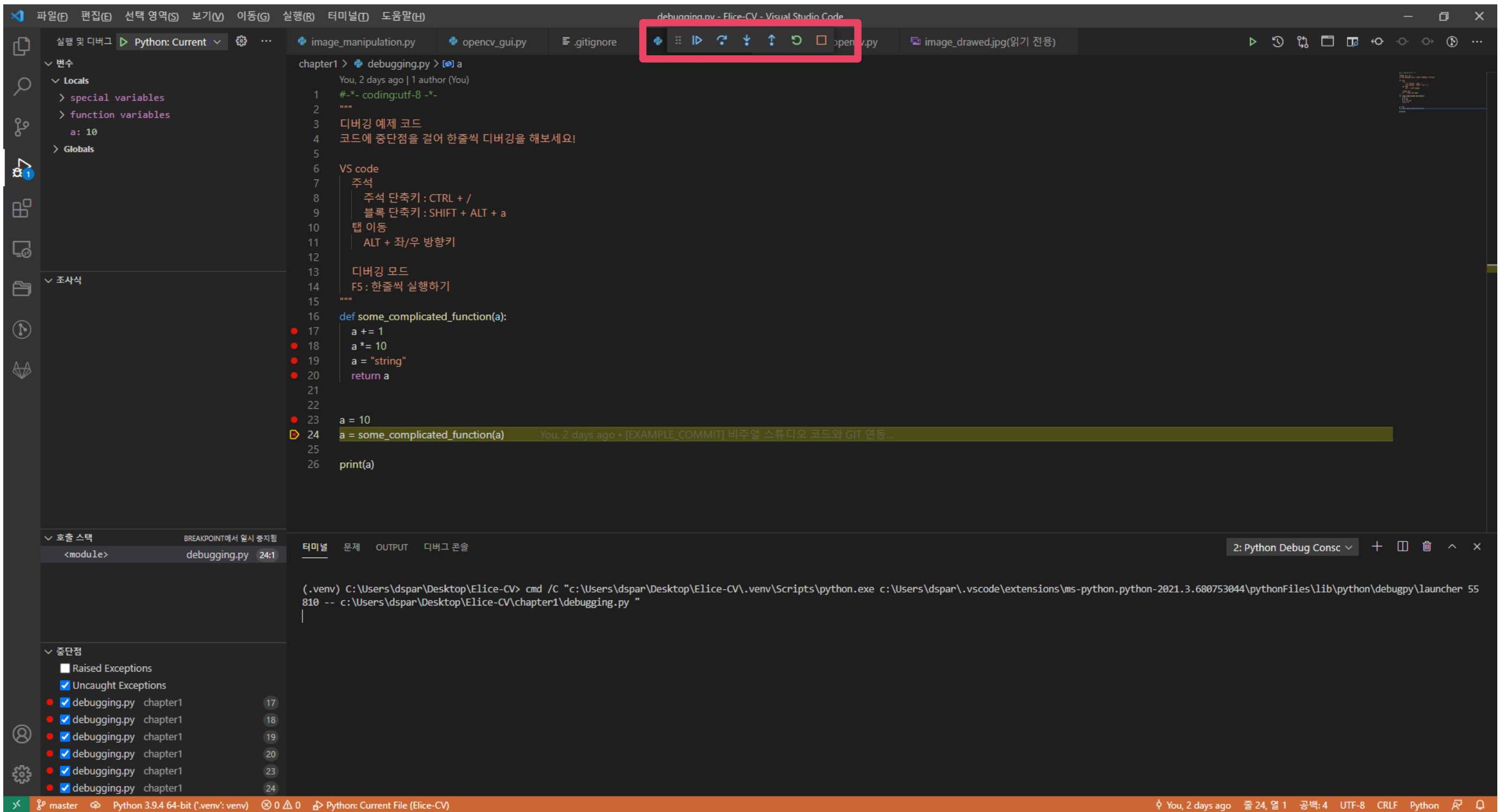
좌측 상단의 디버깅 아이콘 클릭

✓ 코드 디버깅하기



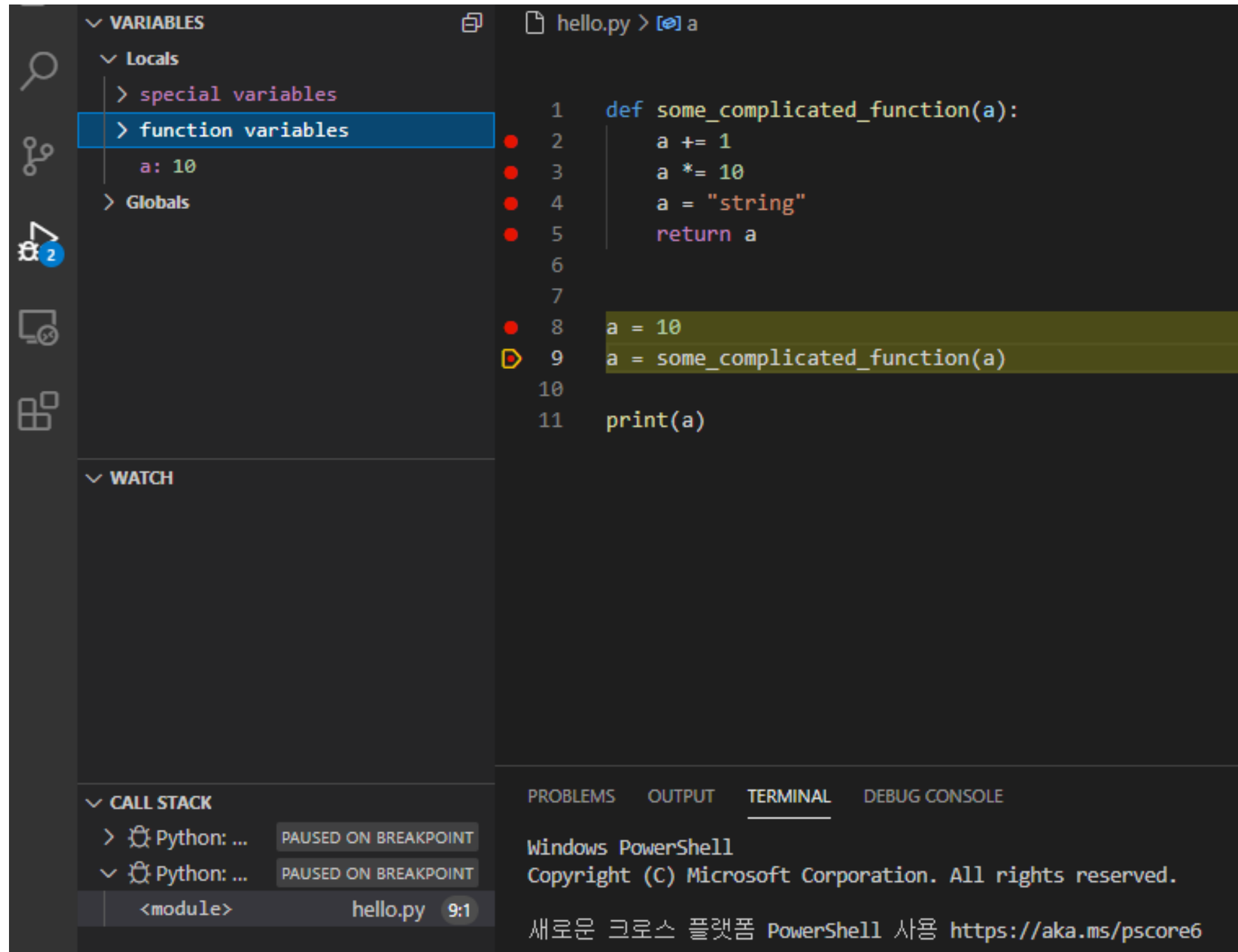
코드 에디터 창에 마우스로 중단점 찍고 디버깅 모드로 실행

✓ 코드 디버깅하기



F5를 눌러서 a값의 변화 보기

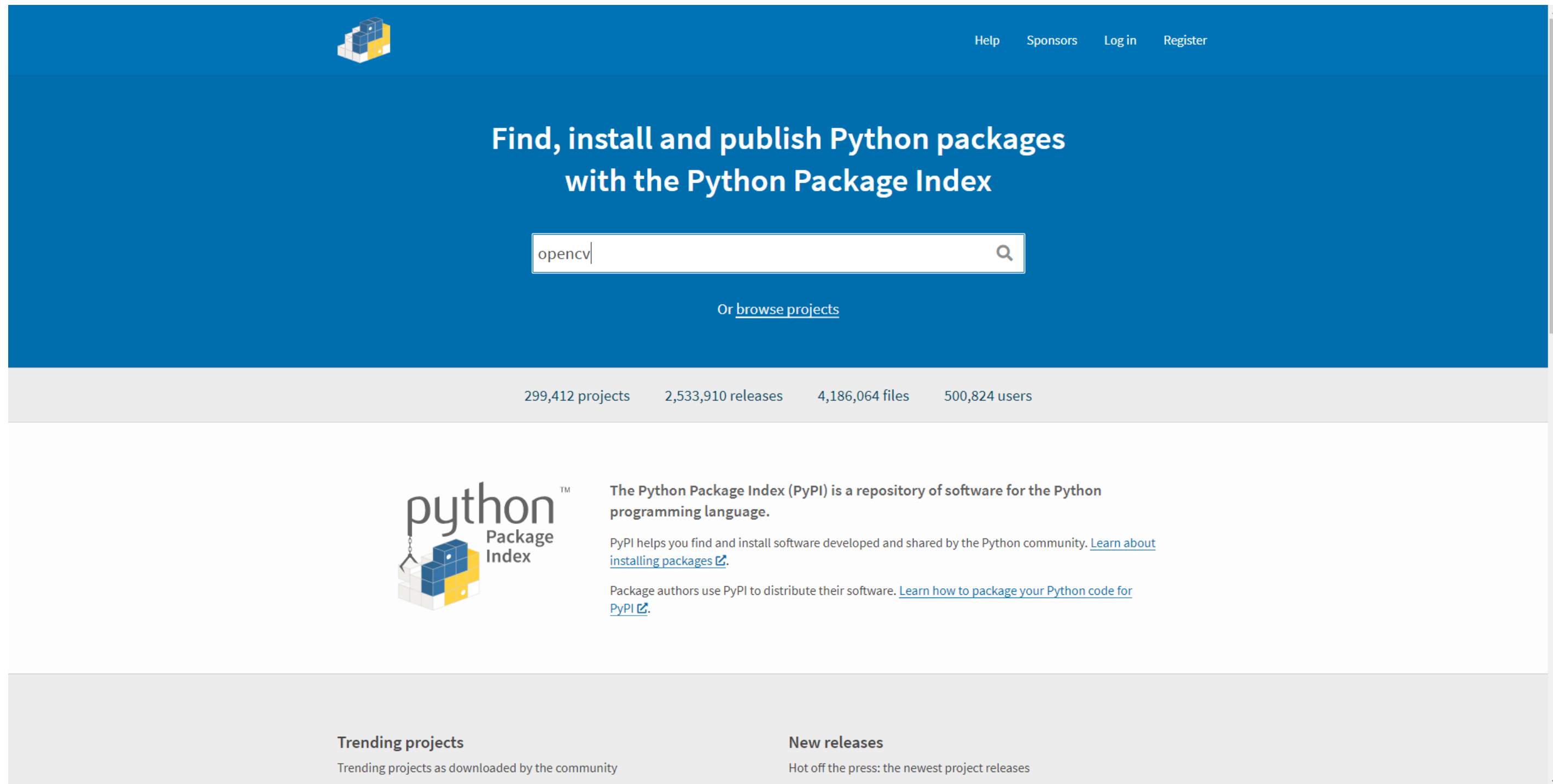
✓ 코드 디버깅하기



좌측의 VARIABLES 창은
중단점 위치에 따른 변수 **a**의 값을 보여줌


해당 기능을 통해 나의 코드가 어디서부터
의도대로 동작하지 않는지 추적 가능

✓ PyPi 채널을 활용하기



PyPi 채널에는 다양한 패키지들이 존재하므로
검색을 통해 적절한 패키지들을 설치하고 활용

✓ OpenCV 검색해보기



[Help](#) [Sponsors](#) [Log in](#) [Register](#)

Filter by classifier

Framework

Topic

Development Status

License

Programming Language

Operating System

Environment

Intended Audience

Natural Language

Typing

10,000+ projects for "opencv python"

Order by

Relevance



opencv-python 4.5.1.48

Wrapper package for OpenCV python bindings.

Jan 2, 2021



opencv-python-headless 4.5.1.48

Wrapper package for OpenCV python bindings.

Jan 2, 2021



opencv-contrib-python 4.5.1.48

Wrapper package for OpenCV python bindings.

Jan 2, 2021



opencv-python-aarch64 3.3.0.1

Wrapper package for OpenCV python bindings.

Sep 21, 2017



python-opencv-utils 0.0.4

Python OpenCV Utilities

Mar 12, 2021



opencv-contrib-python-headless 4.5.1.48

Wrapper package for OpenCV python bindings.

Jan 2, 2021



opencv-opencvino-contrib-python 4.1.1.26

Wrapper package for OpenCV python bindings.

Sep 18, 2019



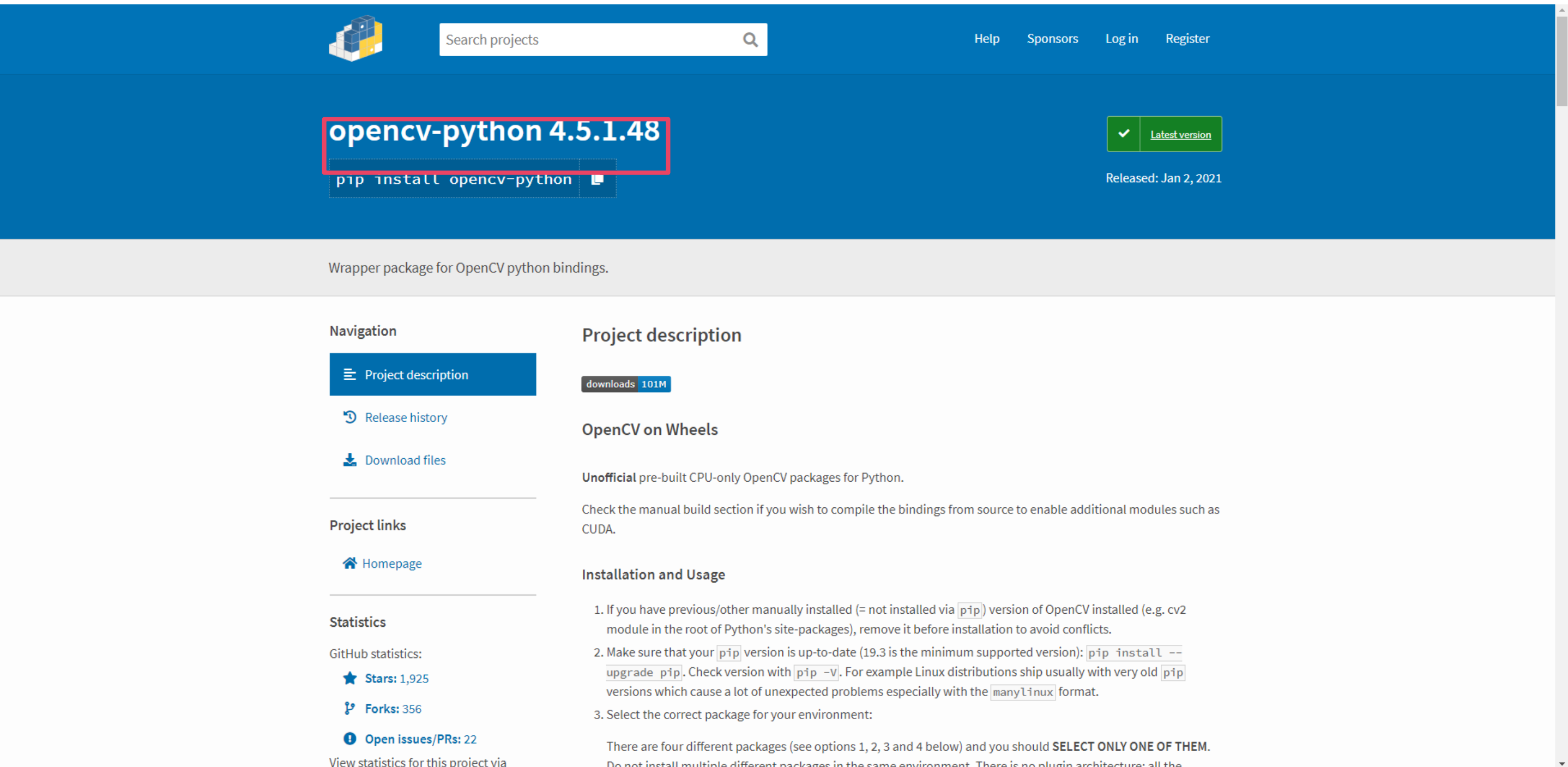
opencv-python-inference-engine 2021.4.13

Wrapper package for OpenCV with Inference Engine python bindings

Apr 14, 2021

좌측에 있는 필터들을 이용해서 목적에 맞게 필터링 가능,
opencv-python 패키지 확인

✓ OpenCV 설치하기



패키지 상세페이지에서 설치방법, 개발 상태, 저자 등 상세한 내용 확인 가능

✓ OpenCV 설치하기

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS C:\Users\admin\PycharmProjects\flaskOpencv> pip install opencv-python
Collecting opencv-python
  Using cached opencv_python-4.5.1.48-cp36-cp36m-win_amd64.whl (34.9 MB)
Requirement already satisfied: numpy>=1.13.3 in c:\python\python36\lib\site-packages (from opencv-python) (1.19.5)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.5.1.48
PS C:\Users\admin\PycharmProjects\flaskOpencv> 
```

```
pip install opencv-python
```

Pip는 파이썬 패키지를 설치해주는 도구로
종속성 있는 패키지들을 자동으로 파악하여 설치

✓ 설치확인

Example

```
import cv2

print(cv2.__version__)
```

모든 소프트웨어 패키지들은 버전명을 나타내는 변수를 내부적으로 가지고 있음
올바르게 설치 됐는지 확인하기 위해 OpenCV 패키지 버전 변수를 출력

04

이미지 I/O 처리



✓ 이미지 읽기

Example

```
# 컬러로 읽기
img_color = cv2.imread(img_path, cv2.IMREAD_COLOR)

# 흑백 이미지로 읽기
img_gray = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
```

```
cv2.imread(filename, flag)
```

인자

fileName : 문자열 타입의 파일 경로

flag : 이미지를 읽을 때 사용할 옵션

반환값 : 이미지를 제대로 읽으면 numpy.ndarray 타입의 이미지가, 그렇지 못할 경우 None을 반환

✓ 읽은 이미지 차원 확인하기

Example

```
# 컬러 이미지 행렬 차원 확인하기
print(type(img_color), img_color.shape)

# 흑백 이미지 행렬 차원 확인하기
print(type(img_gray), img_gray.shape)
```

OpenCV는 numpy를 이용하여 행렬의 형태로 이미지를 표현
그래서 shape을 확인하면 이미지가 컬러인지, 흑백인지 확인 가능

type()과 print()를 이용해 결과 확인

✓ 흑백으로 변환된 이미지 저장하기

Example

```
if cv2.imwrite(save_path, img_gray):  
    print("image save done")  
else:  
    print("image save failed")
```

```
cv2.imwrite(filename, image)
```

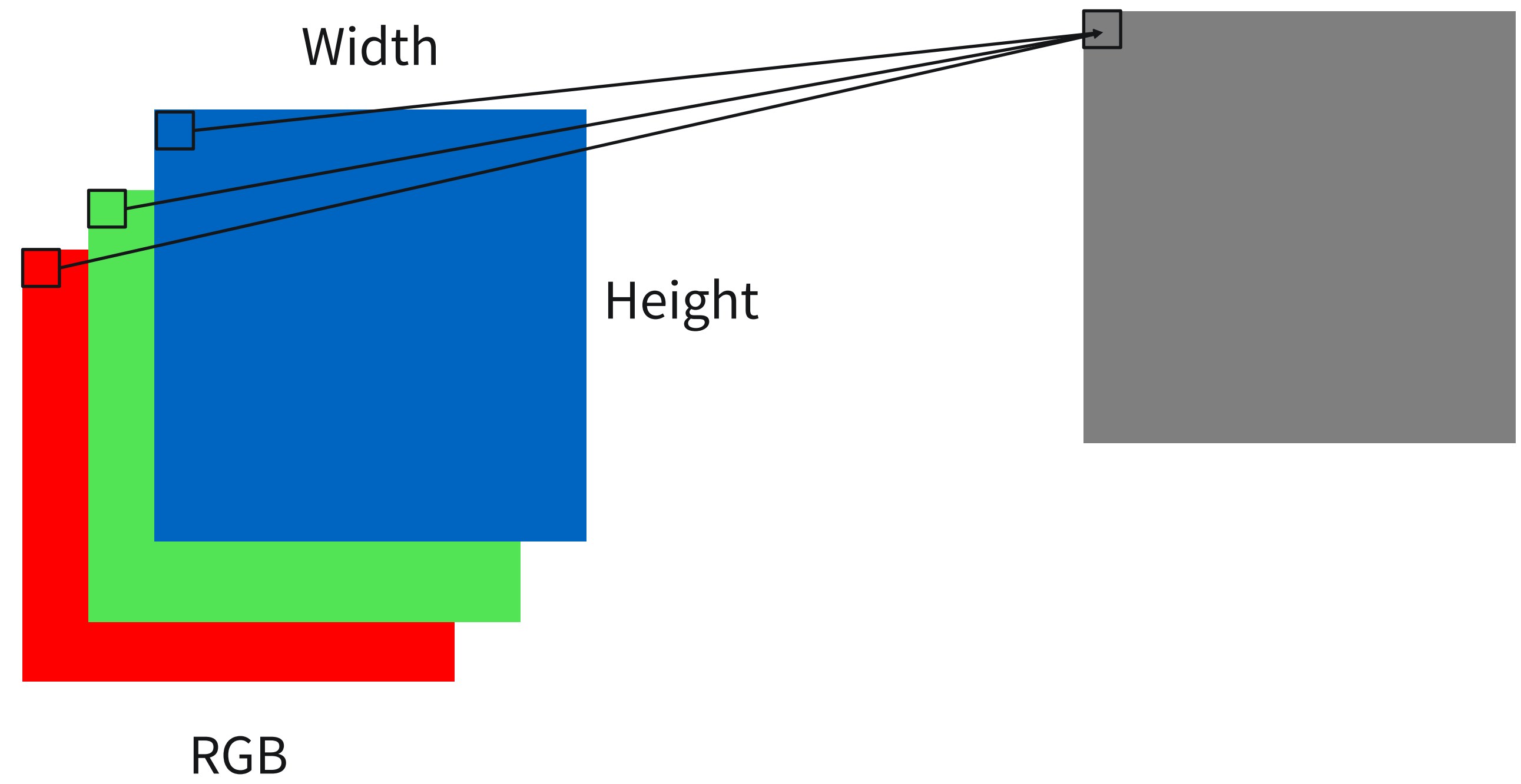
인자

fileName : 문자열 타입의 파일 경로

image : numpy.ndarray 타입 이미지

반환값 : 이미지 저장 실패 여부가 bool 타입으로 반환

✔ 흑백 이미지 컬러 변환하기



$$\text{Gray}(x, y) = \text{int}((R(x, y) + B(x, y) + G(x, y)) / 3)$$

✓ 흑백 이미지 컬러 변환하기

Example

```
# 흑백 이미지를 다시 컬러 이미지로 바꾸기
img_gray_to_color = cv2.cvtColor(img_gray, cv2.COLOR_GRAY2BGR)

# 흑백 이미지 컬러로 변환 후 행렬 차원 확인하기
print("흑백 이미지를 다시 컬러로 변환 : ", type(img_gray_to_color), img_gray_to_color.shape)
```

`cv2.cvtColor(image, option)`

인자

image : numpy.ndarray 타입 이미지

option : 이미지를 읽을 때 사용할 옵션

반환값 : 이미지를 제대로 읽으면 numpy.ndarray 타입의 이미지가, 그렇지 못할 경우 None을 반환

✓ 이미지를 지정된 사이즈로 크기 조정하기

Example

```
# 이미지 리사이징 하기
img_resized = cv2.resize(img_color, (512, 512))
print("이미지를 512 x 512로 크기 변환 : ", type(img_resized), img_resized.shape)
```

`cv2.resize(image, dsize)`

인자

image : numpy.ndarray 타입 이미지

dsize : 목적 이미지 크기. 튜플 형태로 (가로, 세로)로 지정

반환값 : 이미지를 제대로 읽으면 numpy.ndarray 타입의 이미지가, 그렇지 못할 경우 None을 반환

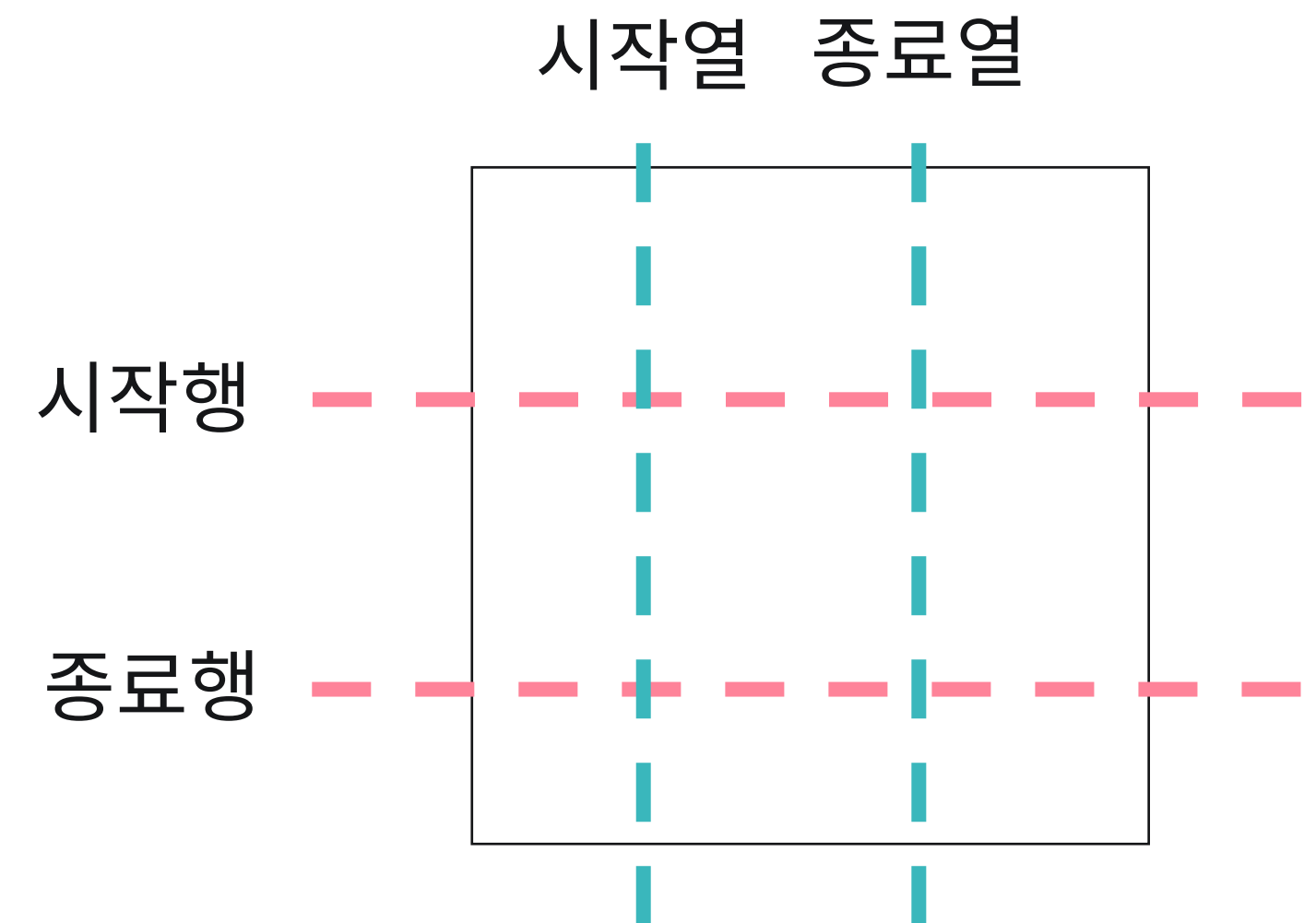
✓ 이미지를 크롭하여 이미지 일부만 남기기

Example

```
# 이미지 크롭 하기
img_cropped = img_color[0:img_color.shape[0] // 2, :]
print("이미지를 횡으로 크롭하기: ", type(img_cropped), img_cropped.shape)
```

크롭 = 2차원 배열을 슬라이싱 하는 것
OpenCV는 이미지를 행렬로 다루므로
numpy의 슬라이싱 기능을 그대로 쓸 수 있음

`image_cropped = [시작행 : 종료행, 시작열 : 가로열]`



05

이미지 위에 정보 표시하기



✓ 선 그리기

Example

```
# 검정 바탕 이미지에 빨간 선분 그리기
start_point, end_point = (0, 0), (img_black.shape[1], img_black.shape[0])
color, line_thickness, line_type = (0, 0, 255), 1, cv2.LINE_AA
img_black = cv2.line(img_black, start_point, end_point, color, line_thickness, line_type)
```

```
cv2.line(image, start_point, end_point, color, line_thickness, line_type)
```

인자

start_point, end_point : 튜플로 구성된 **x, y 좌표로 행렬 좌표가 아님**

color : 선의 색상을 지정. (B, G, R)로 구성되어 있으며 0~ 255 사이 값을 사용

line_thickness : 선의 굵기

line_type : 선의 종류

✓ 원 그리기

Example

```
# 검정 바탕 이미지 중심에 반지름이 30인 파란 원 그리기
center_point = (img_black.shape[1] // 2, img_black.shape[0] // 2)
radius, color, line_thickness, line_type = 30, (255, 0, 0), 1, cv2.LINE_AA
img_black = cv2.circle(img_black, center_point, radius, color, line_thickness, line_type)
```

```
cv2.circle(image, center_point, radius, color, line_thickness, line_type)
```

인자

center_point : 원의 중심점

radius : 반지름 길이

✓ 사각형 그리기

Example

```
# 검정 바탕 이미지 중심에 20 x 20 초록 사각형 그리기
width, height = 20, 20
top_left = center_point[0] - (width // 2), center_point[1] - (height // 2)
bottom_right = center_point[0] + (width // 2), center_point[1] + (height // 2)
color, line_thickness, line_type = (0, 255, 0), 1, cv2.LINE_AA
img_black = cv2.rectangle(img_black, top_left, bottom_right, color, line_thickness, line_type)
```

```
cv2.rectangle(image, top_left, bottom_right, color, line_thickness, line_type)
```

인자

top_left : 사각형이 이미지에 위치할 좌상단 모서리 좌표

bottom_right : 사각형이 이미지에 위치할 우하단 모서리 좌표

✓ 텍스트 쓰기

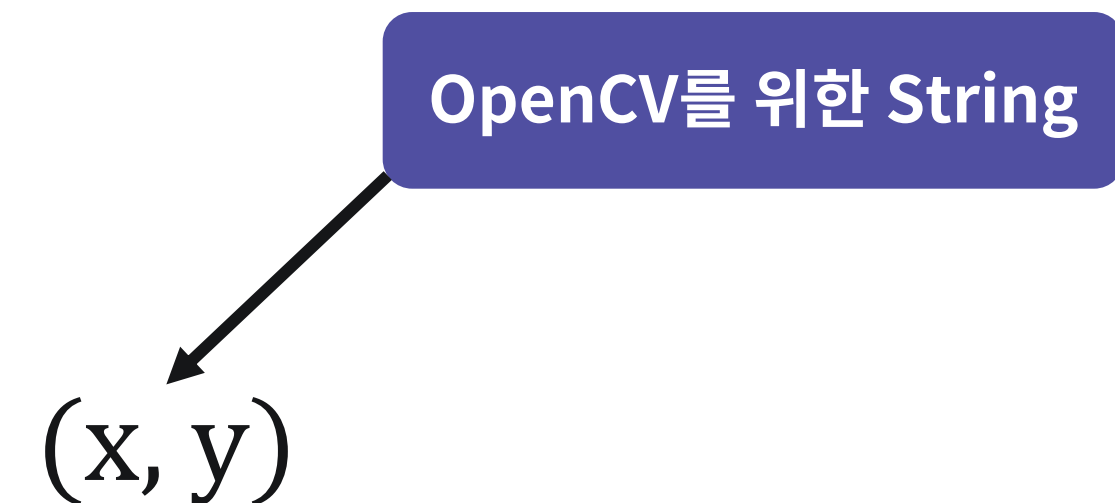
Example

```
# 검정 바탕 이미지에 (30, 30)위치에 하얀 글씨 쓰기
text_bottom_left = (30, 30)
font_type, font_scale = cv2.FONT_HERSHEY_COMPLEX, 0.4
color, line_thickness, line_type = (255, 255, 255), 1, cv2.LINE_AA
img_black = cv2.putText(img_black, "OpenCV", text_bottom_left, font_type, font_scale, color,
                        line_thickness, line_type)
```

```
cv2.putText(image, string, text_bottom_left, font_type, font_scale, color, line_thickness, line_type)
```

string : 이미지에 쓸 문자열로 **영문만 쓸 수 있음**

text_bottom_left : 문자열이 시작되는 좌측 하단 좌표



06

OpenCV GUI 인터페이스 활용하기



✓ 윈도우 생성하기

Example

```
# 윈도우 생성
window_name = "GUI_EXAMPLE"
cv2.namedWindow(window_name, cv2.WINDOW_NORMAL)
```

```
cv2.nameWindow(window_name, flag)
```

OpenCV 는 알고리즘 개발자들이 분석을 쉽게 할 수 있도록 간단한 GUI를 제공

cv2.namedWindow() 함수는 이미지가 그려질 창을 생성하며, 이미지가 그려지는 창의 구분은 문자열 타입의 윈도우 이름으로 식별

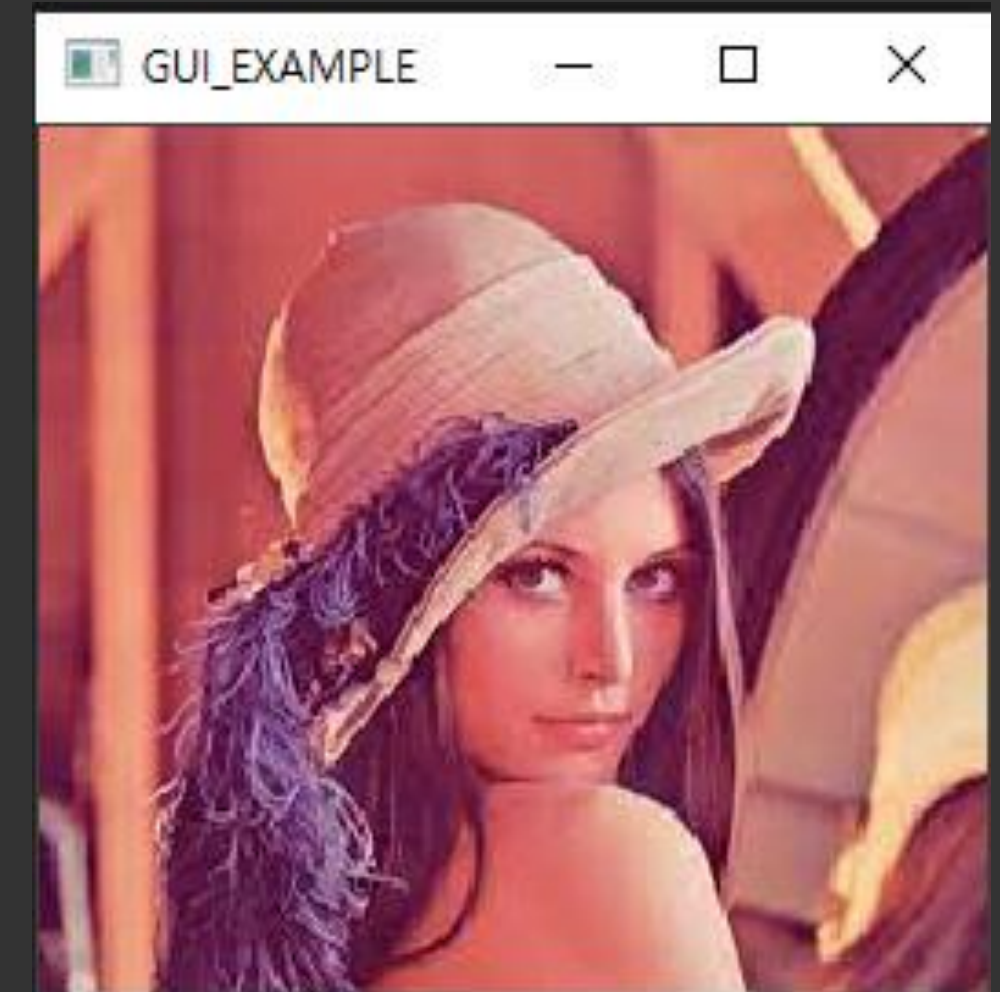
cv2.WINDOW_NORMAL, cv2.WINDOW_FULLSCREEN 등 창을 최대화 했을 때 자동으로 리사이징 할 건지, 리사이징 한다면 종횡비를 지키며 리사이징 할 건지 다양하게 선택 가능

✓ 이미지 창에 그리기

Example

```
# 이미지 읽기
img_path = "./lena.jpg"
img_color = cv2.imread(img_path, cv2.IMREAD_COLOR)

# 윈도우 생성
window_name = "GUI_EXAMPLE"
cv2.namedWindow(window_name, cv2.WINDOW_NORMAL)
cv2.imshow(window_name, img_color)
```



```
cv2.imshow(window_name, image)
```

image를 **window_name** 이름을 가진 창에 출력

✓ 마우스 활용하기

Example

```
# 콜백함수 등록
param = [param1, param2]
cv2.setMouseCallback(window_name, onMouse, param)
```

콜백함수: 특정 조건이 걸릴 때마다 호출되는 함수

Example

```
def onMouse(event, x, y, flags, param):
    # 마우스를 더블 클릭한 경우
    if event == cv2.EVENT_LBUTTONDBLCLK:
        center = (x, y)
        radius = 10
        color = param[0]
        img = param[1]
        cv2.circle(img, center, radius, color, 2)
```

```
cv2.setMouseCallback(window_name,
                      callback_function, param)
```

마우스 왼쪽 버튼 더블 클릭 이벤트가 발생 시 원을 이미지에 그리는 콜백을 등록

✓ 키보드 활용하기

Example

```
# 활성화된 윈도우 창에서 q를 누르면 창이 꺼지도록 종료
key_pressed = cv2.waitKey(1)
if key_pressed == ord("q"):
    exit(0)
```

```
key_pressed = cv2.waitKey(milliseconds)
```

cv2.namedWindow()나 cv2.imshow()로 생성된 윈도우가 키보드 입력을 받을 수 있도록 키 입력이 될 때까지 **milliseconds**만큼 입력을 기다림

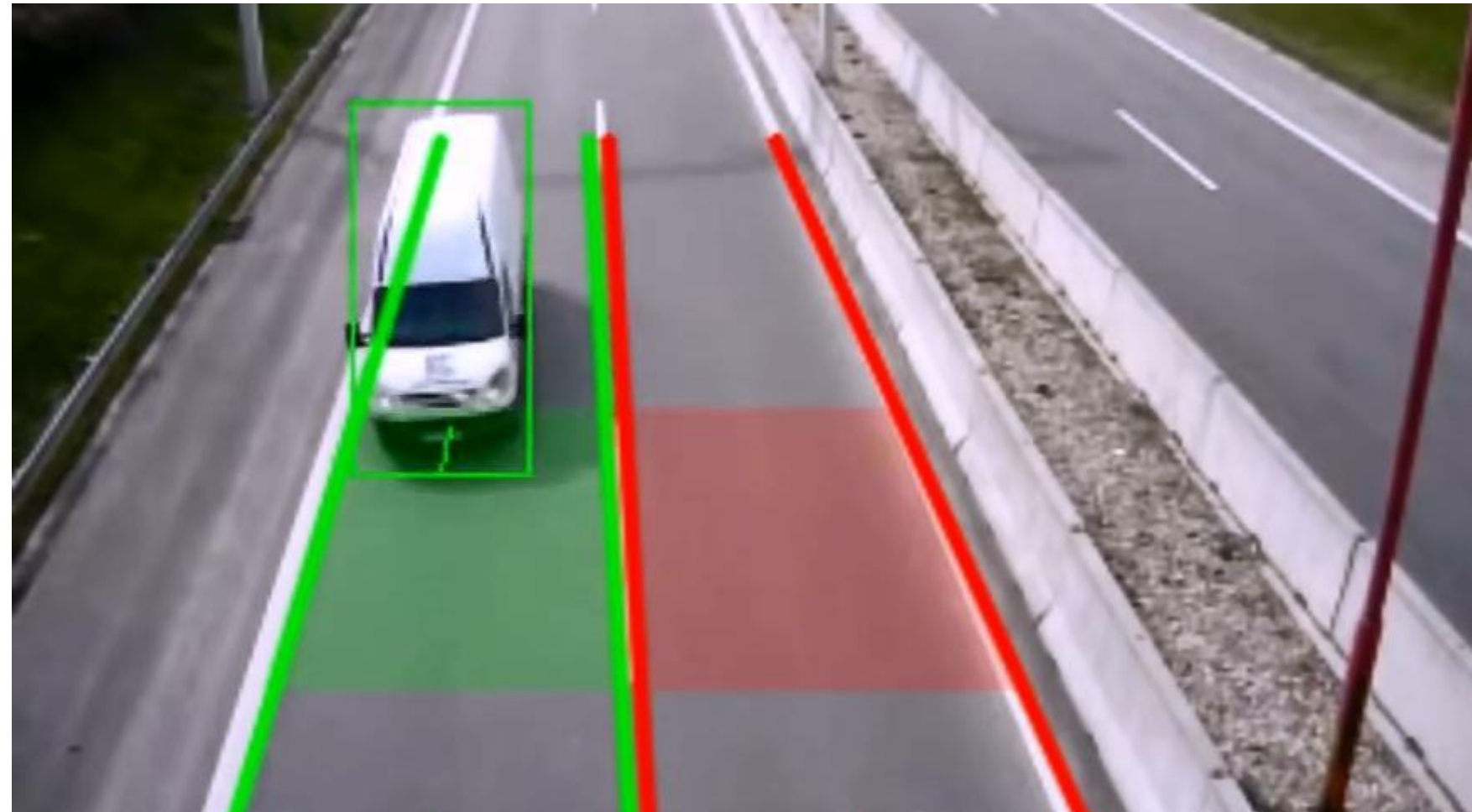
눌린 키의 16진수 값을 반환 후 ord() 내장 함수를 이용하여 16진수가 아닌 문자를 비교

✓ ROI란

ROI(Region of interest)는 **관심 영역**이라는 뜻으로
이미지 영역 중에서 특정 영역만 알고리즘을 적용하거나,
특정 영역에서 검출된 정보만을 사용

불필요한 주변 픽셀을 다 사용하지 않고
관심이 가는 영역을 지정하여
알고리즘의 성능을 높이고 싶을 때 설정

✓ ROI란



차량 차선 위반 감지의
여기서 관심영역은 **도로** 영역

✓ ROI 박스 그리기

Example

```
roi = cv2.selectROI(window_name, image, showCrosshair=False)
print(roi)
```

```
cv2.selectROI(window_name, image, showCrosshair=False)
```

Image를 보여주는 윈도우를 새로 생성하거나, 이미 생성한 윈도우에 ROI를 그림
알맞게 박스를 잡았다면 **Space 또는 Enter**를 눌러 ROI 박스 좌표를 얻을 수 있고, **C**키를 눌러 취소 가능

```
roi = [top_left_x, top_left_y, bottom_right_x, bottom_right_y]
```

ROI 박스는 좌상단 모서리 좌표와 우하단 모서리 좌표를 담음

07

윤곽선의 이해

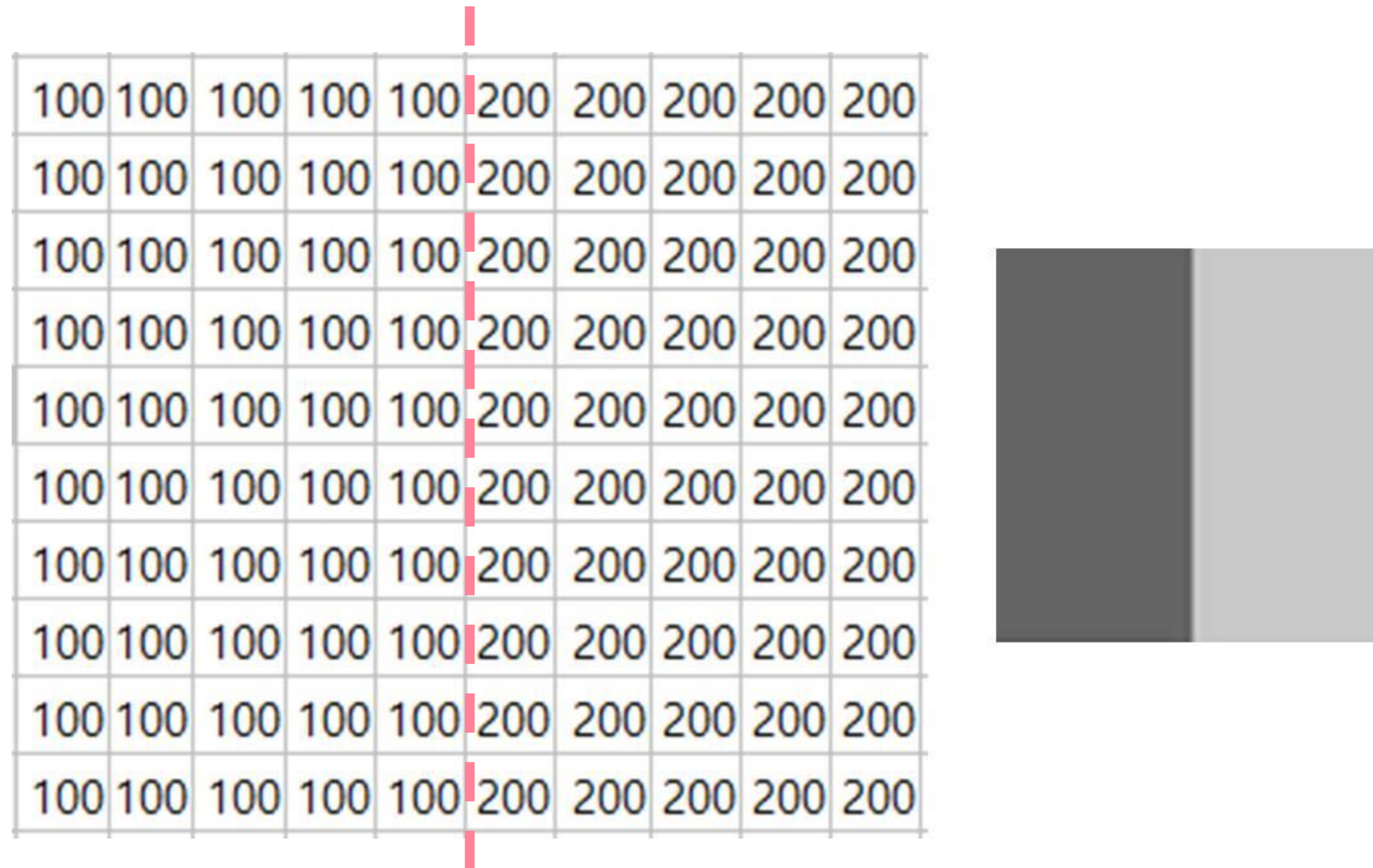


✓ 윤곽선이란

사람의 눈은 영상이 얼마나 어두운지 밝은지에 대한 정보보다
상대적으로 얼마나 밝거나 어두운지에 더 예민하게 반응함

이처럼 영상에서도
물체의 경계는 **인근 화소들 간의 명도차가 급격히 나타나는 구간**을
‘윤곽선’ (Edge)이라고 부름.

✓ 윤곽선이란



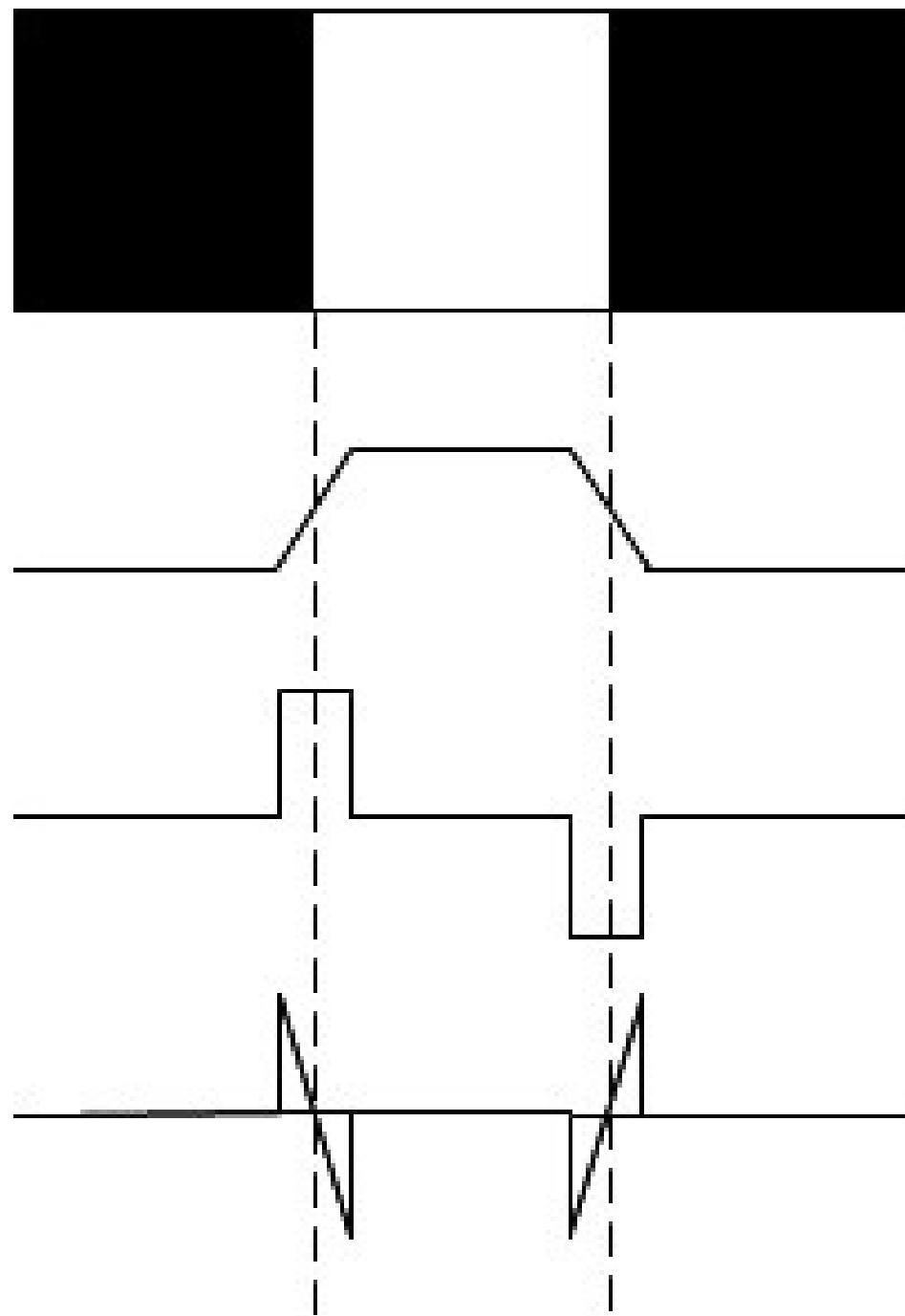
위의 그림은 극단적으로 영상의 경계를 나타낸 예시 이미지임
중앙선을 기준으로 명도의 대비가 극심한 것을 볼 수 있음

✓ 윤곽선이란

디지털 영상 처리에서 윤곽선이란

- 영상의 밝기 변화가 일어나는 지점
ex. 낮은 명도 -> 높은 명도 또는 높은 명도에서 -> 낮은 명도
- 디지털 영상을 구성하는 객체 간의 경계
- 물체 식별, 위치/모양/크기 등을 인지하고 방향성을 탐지할 수 있는 정보를 제공함

✓ 디지털 영상에서 윤곽선의 특징



영상

명암변화

1차 미분

2차 미분

디지털 영상에서의 미분은
‘인접 화소 간의 차’ 이므로
‘**차분**’이라는 용어를 사용한다.

수평 차분 : $f(x-1, y) - f(x+1, y)$

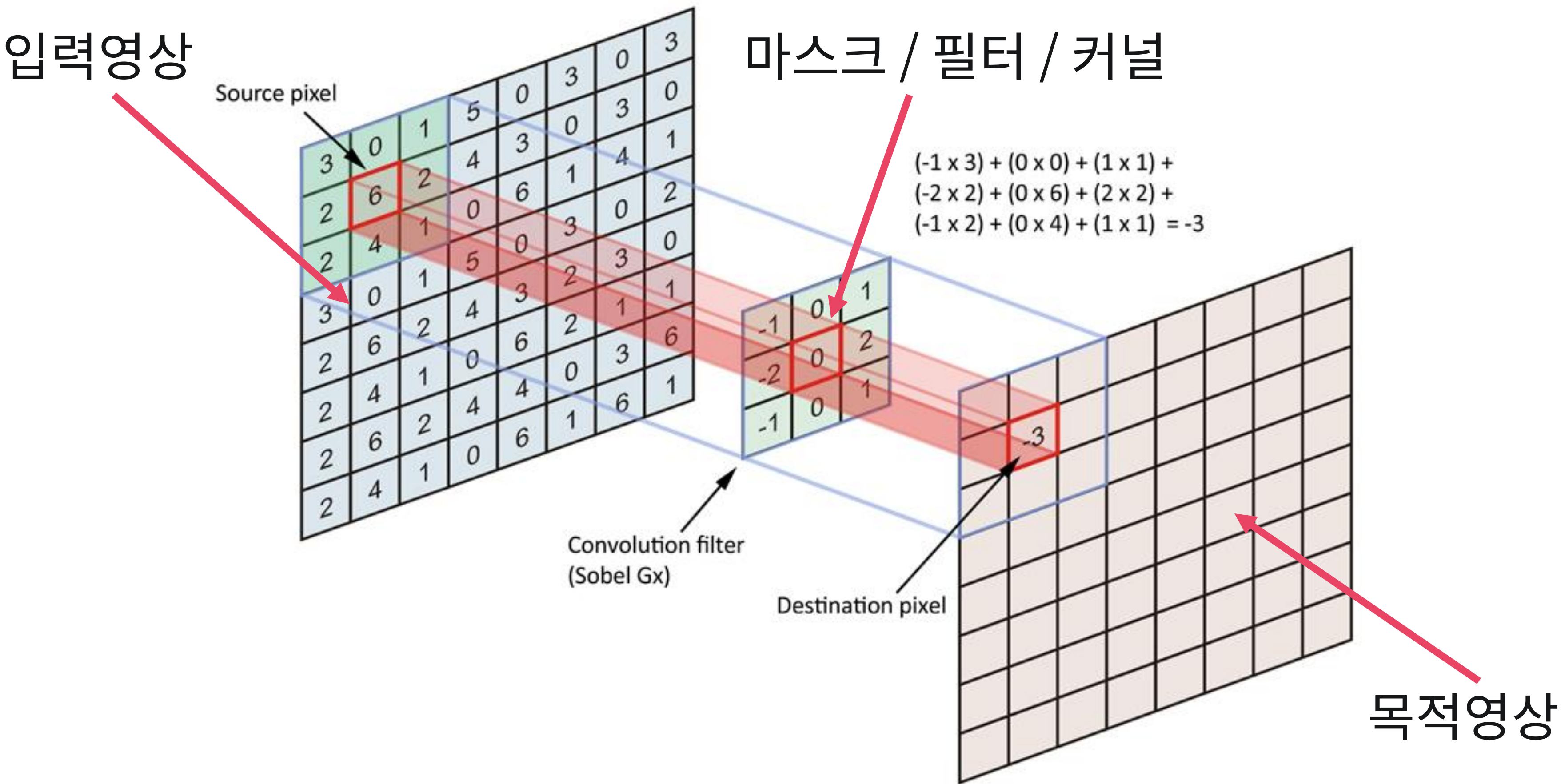
수직 차분 : $f(x, y-1) - f(x, y+1)$

08

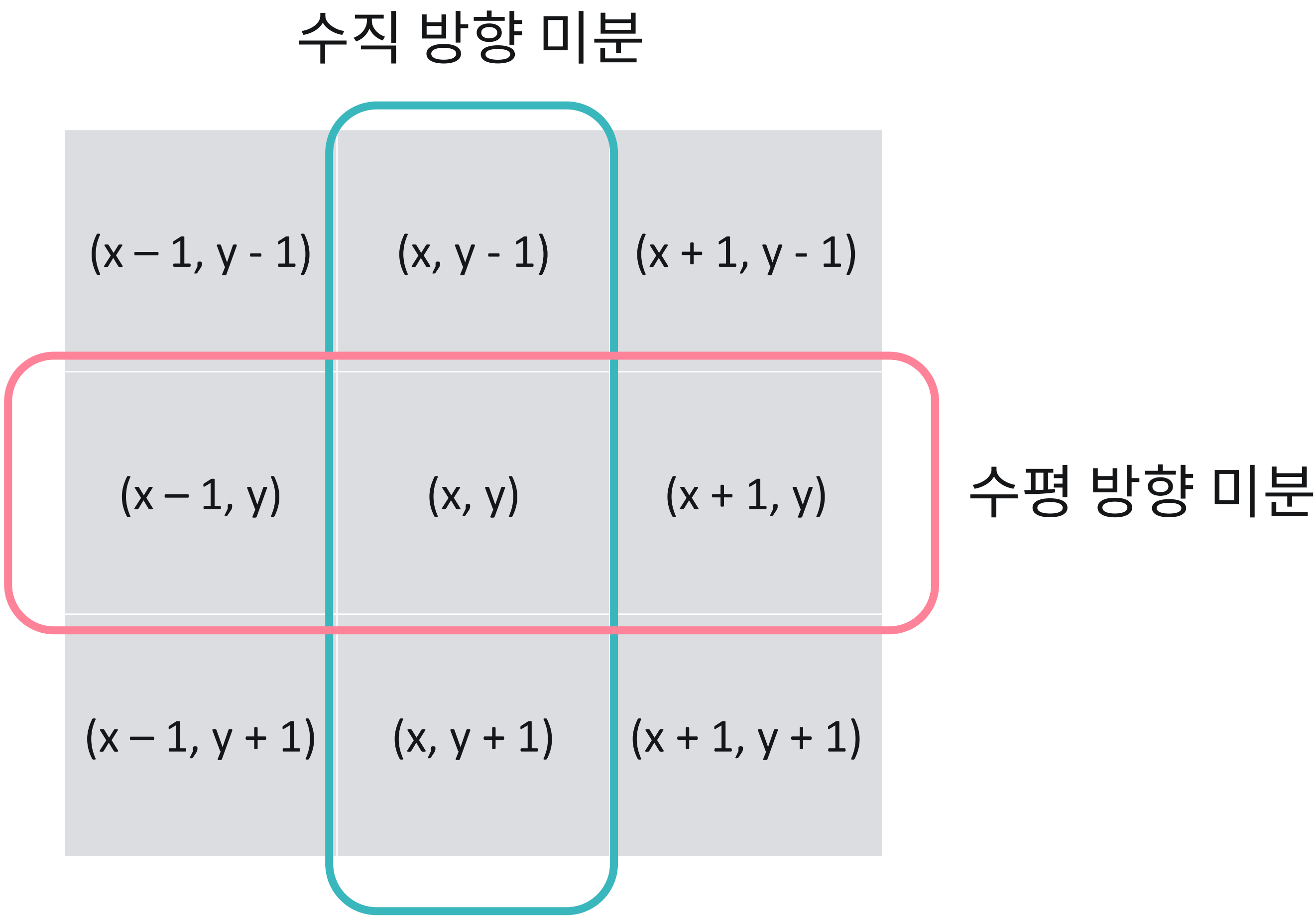
회선처리



✓ 필터, 마스크, 커널, 회선처리



✓ 수평 윤곽선 필터와 수직 윤곽선 필터의 설계



✔ 1차 미분 에지 검출 필터 소개 : 수평, 수직 윤곽선 필터

0	0	0
-1	1	0
0	0	0

수평 윤곽선 필터

0	-1	0
0	1	0
0	0	0

수직 윤곽선 필터

✓ 로버츠 필터

-1	0	0
0	1	0
0	0	0

↖ 강조 필터

0	-1	0
0	1	0
0	0	0

↗ 강조 필터

✔ 프리윗 필터

-1	-1	-1
0	0	0
1	1	1

수평 윤곽선 필터

-1	0	1
-1	0	1
-1	0	1

수직 윤곽선 필터

크레딧

/* elice */

코스 매니저

이재성

콘텐츠 제작자

최지수

강사

최지수

감수자

최지수

디자이너

강혜정

연락처

TEL

070-4633-2015

WEB

<https://elice.io>

E-MAIL

contact@elice.io

