

# 파이썬을 활용한 컴퓨터 비전 입문

## Chapter 02. OpenCV 설치와 기초 사용법

동양미래대학교  
인공지능소프트웨어학과  
권 범

본 강의자료는 길벗 출판사의 『OpenCV 4로 배우는 컴퓨터 비전과 머신 러닝』  
교재 내용을 토대로 작성되었습니다.

## ❖ 2장 OpenCV 설치와 기초 사용법

- 2.1 OpenCV 개요와 설치
- 2.2 OpenCV 사용하기: HelloCV

## 2.1 OpenCV 개요와 설치

2.2 OpenCV 사용하기: HelloCV

## ❖ OpenCV 개요 (1/4)

- OpenCV는 **오픈 소스**로 개발되고 있는 **컴퓨터 비전 및 머신 러닝 라이브러리**임
- OpenCV는 Open Source Computer Vision Library의 약어 형태로 이름이 지어졌으며, '오픈씨브이'라고 읽음
- OpenCV는 2500개가 넘는 최신 컴퓨터 비전 알고리즘과 머신 러닝 알고리즘을 포함
- 기본적인 영상 파일 입출력, 영상의 화질 향상, 객체 검출과 인식, 추적, 3차원 비전 문제 해결 등 다양한 기능을 제공함

## ❖ OpenCV 개요 (2/4)

- k 최근접 이웃(kNN, k-Nearest Neighbor), 서포트 벡터 머신(SVM, Support Vector Machine) 같은 머신 러닝 알고리즘도 제공함
- 최근에는 딥러닝(deep learning)으로 알려져 있는 심층 신경망(DNN, Deep Neural Network) 모델을 실행하는 기능도 제공함
- OpenCV 라이브러리의 활용도가 더욱 높아지고 있음
- OpenCV는 그 태생부터 **실시간 처리를 고려하여 만들어졌기 때문에 다양한 하드웨어 플랫폼에서 매우 빠르게 동작**

## ❖ OpenCV 개요 (3/4)

- OpenCV는 기본적으로 C/C++ 언어로 작성되었지만, 현재 널리 사용되고 있는 **Python**, Java, Matlab, JavaScript 등 **인터페이스도 제공함**
- OpenCV는 Windows, Linux, MacOS 등 운영 체제를 지원함
- 안드로이드와 iOS 같은 모바일 환경도 지원함
- OpenCV 기능은 대부분 병렬 처리로 동작함
- MMX, SSE, AVX, NEON 등 CPU 특화 명령어도 지원함
- 오래전부터 CUDA와 OpenCL을 통한 GPU 활용을 지원함

## ❖ OpenCV 개요 (4/4)

- OpenCV 라이브러리는 BSD 라이선스를 따르고 있기 때문에 학계 연구용이나 상업적인 용도로 자유롭게 사용할 수 있음
- OpenCV 라이브러리를 이용하여 상용 프로그램을 만들 수도 있음
- OpenCV 소스 코드의 일부를 사용하여 프로그램을 개발하는 것도 허용됨

## ❖ OpenCV 역사 (1/5)

- OpenCV는 1999년 인텔(Intel)에서 개발된 IPL(Image Primitive Library)을 기반으로 만들어지기 시작함
- 이후 2000년 일반에 공개되어 오픈 소스로서 개발이 진행됨
- 2006년에 OpenCV 1.0 버전이 정식으로 배포
- **OpenCV 1.0은 C 언어를 기반으로 구현됨**
- 많은 컴퓨터 비전 알고리즘이 주로 구조체와 함수로 구현됨
- 영상 데이터는 IplImage라는 이름의 구조체를 이용하여 표현함



## ❖ OpenCV 역사 (2/5)

- OpenCV 1.0 버전이 나온 지 3년 후인 2009년에는 OpenCV 2.0이 발표됨
- **OpenCV 2.0은** C 언어 대신 **C++ 인터페이스를 채택함**
- Mat라는 이름의 클래스를 사용하여 영상 데이터를 표현하기 시작함
- C++클래스를 사용함으로써 메모리 관리가 좀 더 수월해지고 소스 코드 작성이 더욱 편리해짐
- 이후 지속적인 소규모 버전업이 진행되면서 새로운 알고리즘 구현 함수, 성능 개선, 병렬 처리 기능 강화 등이 추가됨

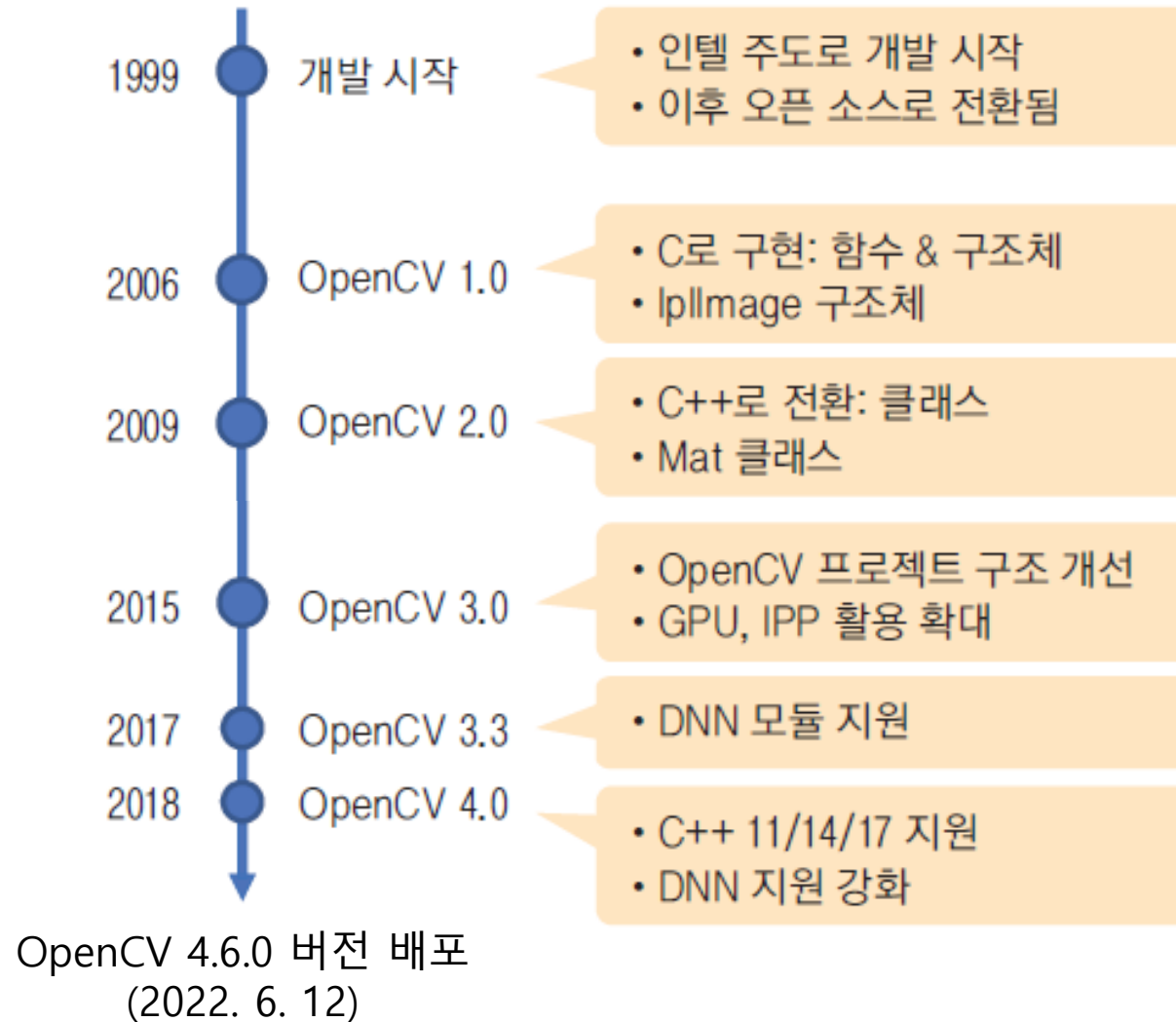
## ❖ OpenCV 역사 (3/5)

- **OpenCV 3.0**은 2015년 6월에 발표됨
- OpenCV 3.0 버전에서는 OpenCV 프로젝트 구조가 크게 개선되었고 전반적인 성능이 향상됨
- OpenCL 사용성을 크게 확대한 T-API(Transparent API)를 지원하기 시작함
- 유료로 사용해야 했던 인텔 IPP(Integrated Performance Primitives) 라이브러리 일부를 OpenCV에서 무료로 사용할 수 있게 됨
- 2017년 8월에 발표된 OpenCV 3.3 버전에서는 최근에 각광받고 있는 심층 신경망을 지원하는 DNN 모듈이 기본 소스에 포함되기 시작함
- AVX/AVX2/SSE4.x 최적화가 추가됨
- 최신 C++11 문법도 지원하기 시작함

### ❖ OpenCV 역사 (4/5)

- **OpenCV 4.0**은 2018년 11월에 발표됨
- OpenCV 4.0의 가장 큰 변화는 C++11의 필수 지원임
- OpenCV 4.0은 C++11을 지원하는 컴파일러 환경에서 사용할 수 있음
- 최신 C++ 문법을 기본적으로 사용할 수 있음
- 함수의 포인터 또는 함수 객체 대신 람다 표현식(lambda expression)을 사용할 수 있게 됨
- Mat 클래스 객체 초기화 시 C++11 초기화 방법을 사용할 수 있게 됨
- **DNN 모듈 기능이 강화**되어 AlexNet, Inception v2, Resnet, VGG 같은 영상 분류기뿐만 아니라 Mask-RCNN, tiny YOLO 같은 **최신 딥러닝 네트워크 구조를 지원함**
- QR 코드를 검출하고 해석하는 기능도 새롭게 제공함
- 참고로 OpenCV 4.0에서는 오래된 **C API 지원이 종료**되어 더 이상 IplImage 구조체 등을 사용할 수 없음

## ❖ OpenCV 역사 (5/5)



◀ 그림 2-1 OpenCV 버전과 주요 특징

## ❖ OpenCV 모듈 (1/4)

- OpenCV 라이브러리는 다수의 모듈(module)로 구성되어 있음
- 모듈은 OpenCV에서 제공하는 다양한 클래스와 함수를 그 기능과 성격에 따라 모아서 만들어 놓은 OpenCV의 부분 라이브러리임

▼ 표 2-1 OpenCV 주요 모듈

모듈 이름	설명
calib3d	카메라 캘리브레이션과 3차원 재구성
core	행렬, 벡터 등 OpenCV 핵심 클래스와 연산 함수
dnn	심층 신경망 기능
features2d	2차원 특징 추출과 특징 벡터 기술, 매칭 방법
flann	다차원 공간에서 빠른 최근방 이웃 검색
highgui	영상의 화면 출력, 마우스 이벤트 처리 등 사용자 인터페이스

### ❖ OpenCV 모듈 (2/4)

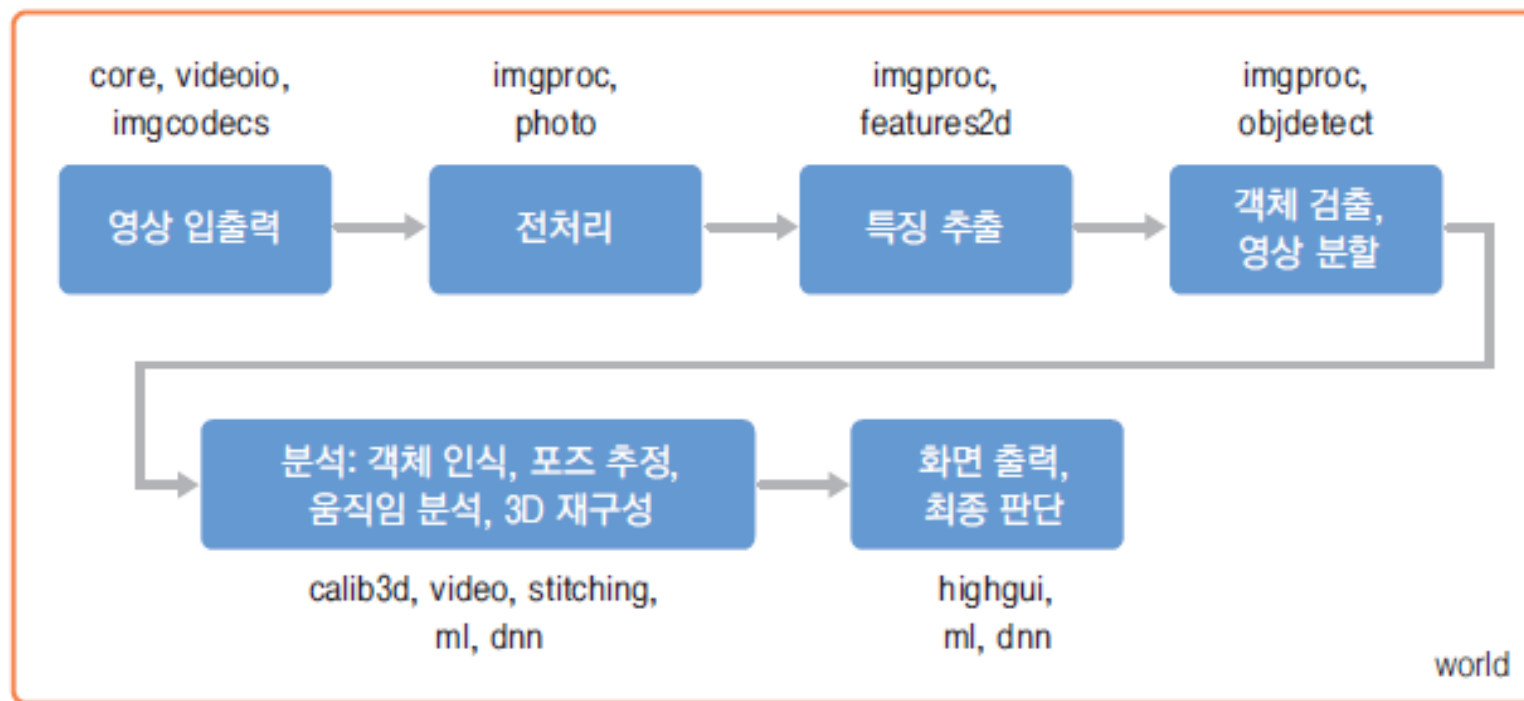
▼ 표 2-1 OpenCV 주요 모듈

모듈 이름	설명
imgcodecs	영상 파일 입출력
imgproc	필터링, 기하학적 변환, 색 공간 변환 등 영상 처리 기능
ml	통계적 분류, 회기 등 머신 러닝 알고리즘
objdetect	얼굴, 보행자 검출 등 객체 검출
photo	HDR, 잡음 제거 등 사진 처리 기능
stitching	영상 이어 붙이기
video	옵티컬 플로우, 배경 차분 등 동영상 처리 기술
videoio	동영상 파일 입출력
world	여러 OpenCV 모듈을 포함하는 하나의 통합 모듈

## ❖ OpenCV 모듈 (3/4)

- 일반적인 컴퓨터 비전 문제 해결 과정에서 사용할 수 있는 OpenCV 모듈을 나타냄

### ▼ 그림 2-2 일반적인 컴퓨터 비전 문제 해결 과정과 관련 OpenCV 모듈



## ❖ OpenCV 모듈 (4/4)

- OpenCV 라이브러리는 현재에도 지속적으로 업데이트되고 있음
- 최신의 컴퓨터 비전 알고리즘은 OpenCV 추가 모듈(extra module) 형태로 함께 개발되고 있음
- 추가 모듈에는 주로 아직 안정화가 되지 않은 최신 알고리즘 구현이 포함됨
- 이외에도 소스 코드는 공개되었지만 알고리즘에 특허가 걸려 있어서 무료로 사용할 수 없는 기능과 CUDA 관련 기능도 추가 모듈로 배포됨
- 추가 모듈은 OpenCV 소스 코드가 배포되는 GitHub 웹 사이트에서 opencv\_contrib라는 이름의 저장소를 통해 따로 배포되고 있음



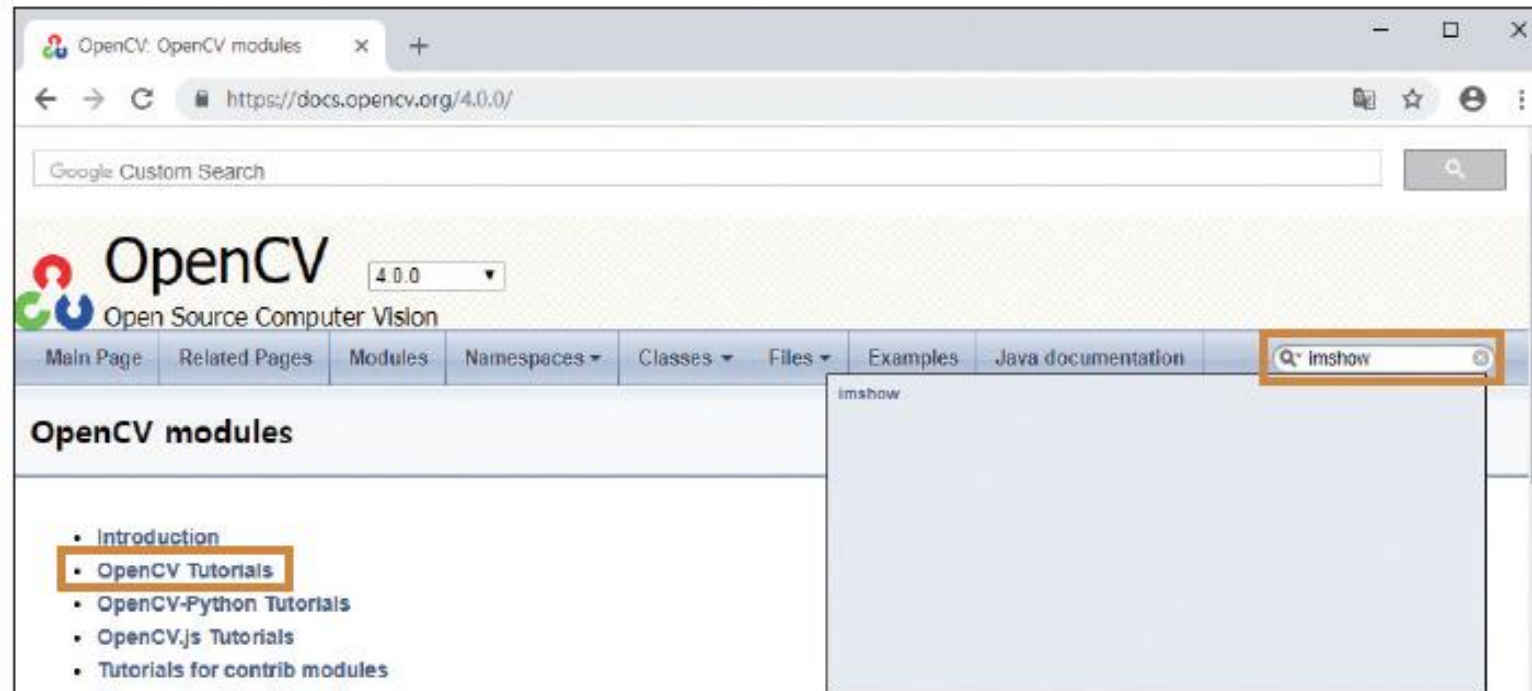
### ❖ OpenCV 관련 사이트 (1/3)

- OpenCV 공식 사이트 주소: <https://opencv.org/>
- OpenCV 공식 웹 사이트에서는 OpenCV에 대한 소개와 최신 소식을 확인할 수 있음
- OpenCV 라이브러리 설치 파일 및 소스파일을 내려받을 수 있음
- OpenCV의 함수 또는 클래스의 자세한 사용방법이 알고 싶다면 OpenCV 문서 사이트를 활용할 수 있음
- OpenCV 문서 사이트 주소: <https://docs.opencv.org/>
- 여기서 OpenCV 버전에 따른 문서 접근이 가능함
- 특히 OpenCV 4.6.0 버전의 문서는 <https://docs.opencv.org/4.6.0/>에서 확인할 수 있음

### ❖ OpenCV 관련 사이트 (2/3)

- OpenCV 문서 페이지의 우측 상단 검색 창에 OpenCV 함수 또는 클래스 이름을 입력하면 해당하는 함수 또는 클래스에 대한 자세한 설명 페이지로 쉽게 이동

#### ▼ 그림 2-3 OpenCV 4.0.0 문서 사이트

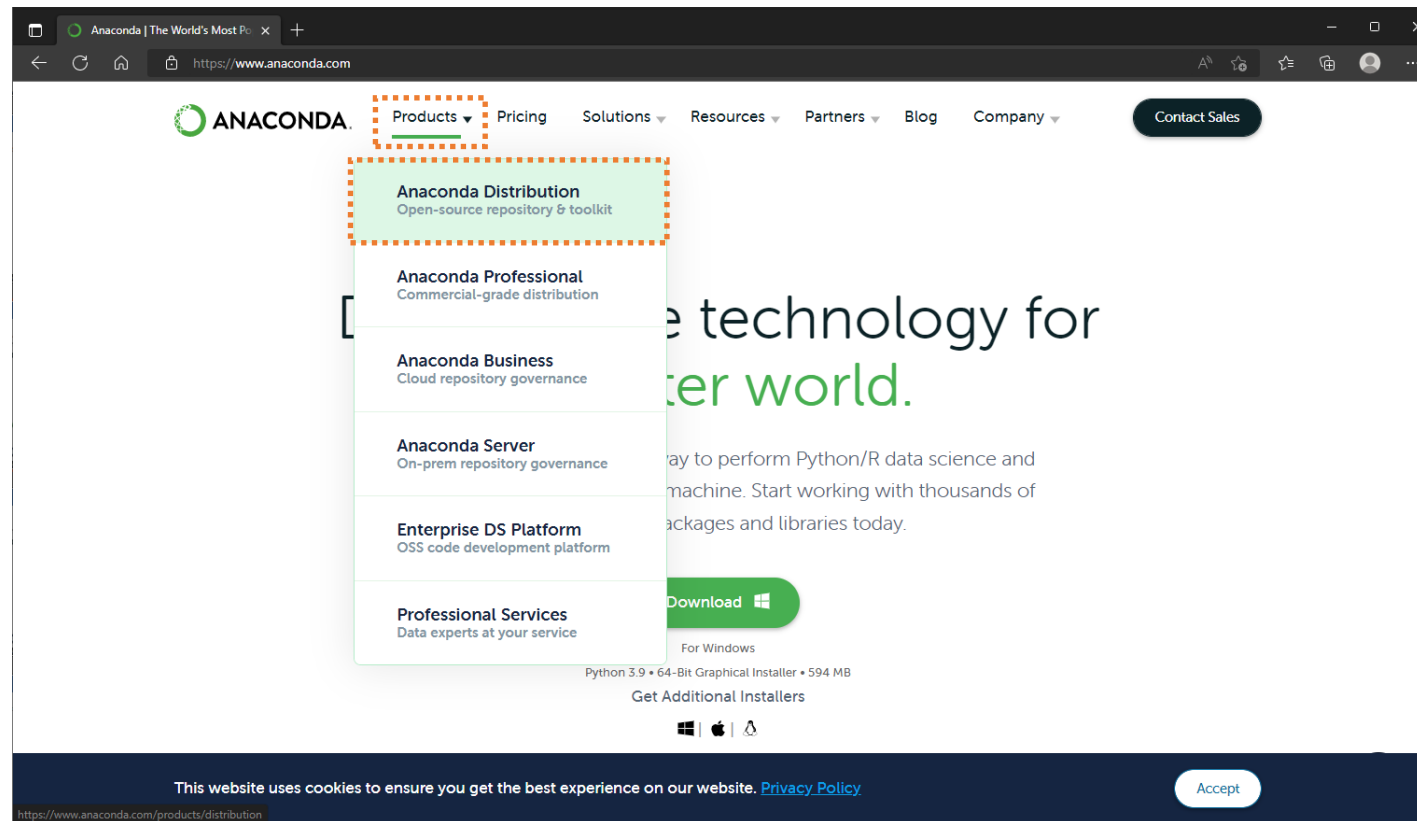


### ❖ OpenCV 관련 사이트 (3/3)

- OpenCV를 이용하다가 궁금한 사항이 생기면 OpenCV 질문/답변 포럼에서 해답을 구할 수 있음
- OpenCV 질문/답변 포럼 주소: <http://answers.opencv.org/questions/>
- 포럼에는 5만 명 이상의 사용자가 가입되어 있으며 다양한 OpenCV 정보를 교환할 수 있음
- 웹사이트에서 궁금한 사항을 검색하거나 또는 새로운 질문을 올려서 해결 방법을 찾을 수 있음
- OpenCV 최신 소스 코드는 GitHub 웹 사이트를 통해 관리됨
- OpenCV 기본 소스 저장소 주소: <https://github.com/opencv/opencv/>
- OpenCV 추가 모듈 소스 저장소 주소: [https://github.com/opencv/opencv\\_contrib/](https://github.com/opencv/opencv_contrib/)
- 이 저장소에서 OpenCV 소스 코드의 업데이트 현황을 확인 및 최신 소스 코드를 내려받을 수 있음

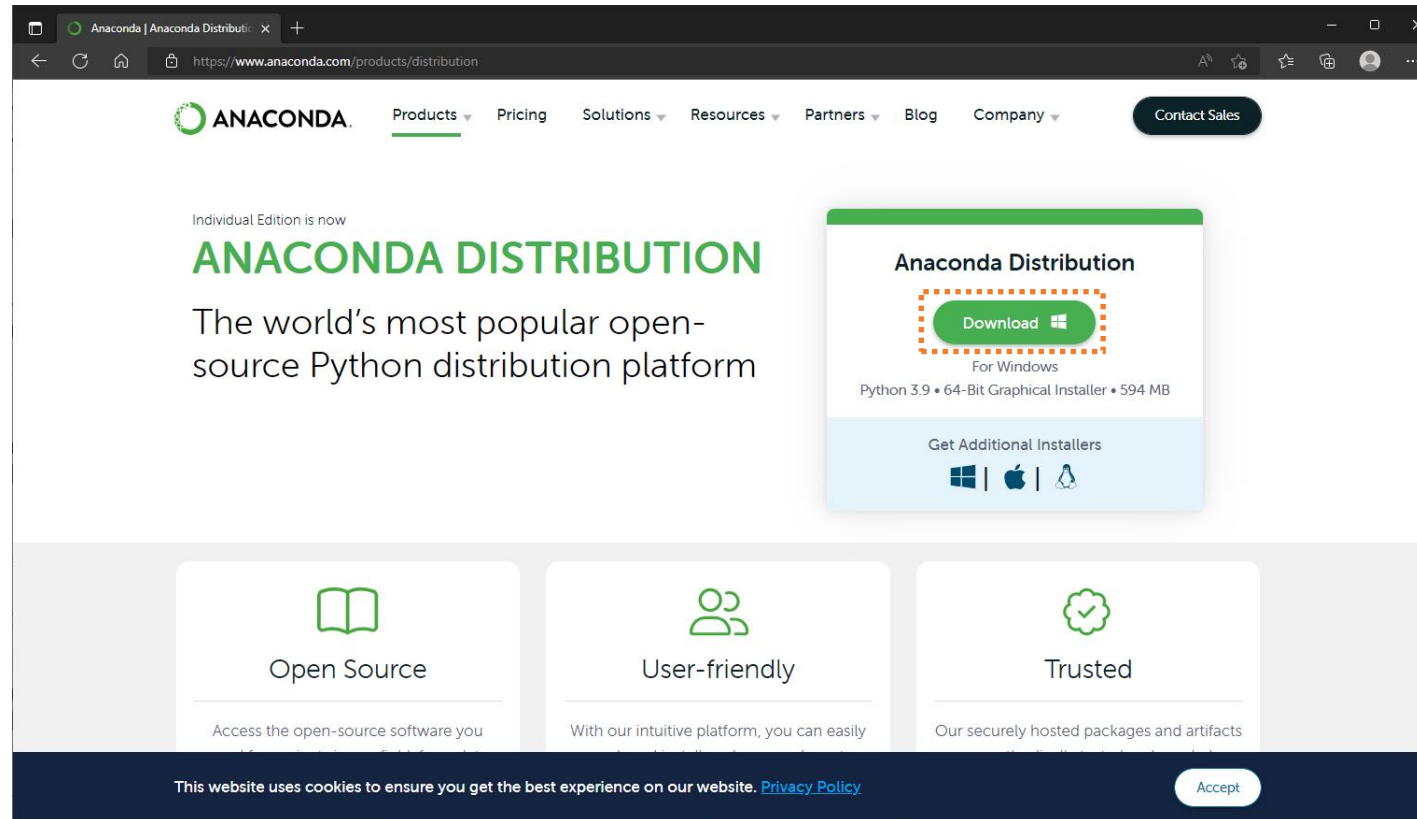
### ❖ OpenCV 설치: ① 아나콘다 설치하기 (1/10)

- 아나콘다 사이트(<https://www.anaconda.com/>)에 접속 후, [Products] → [Anaconda Distribution]을 클릭



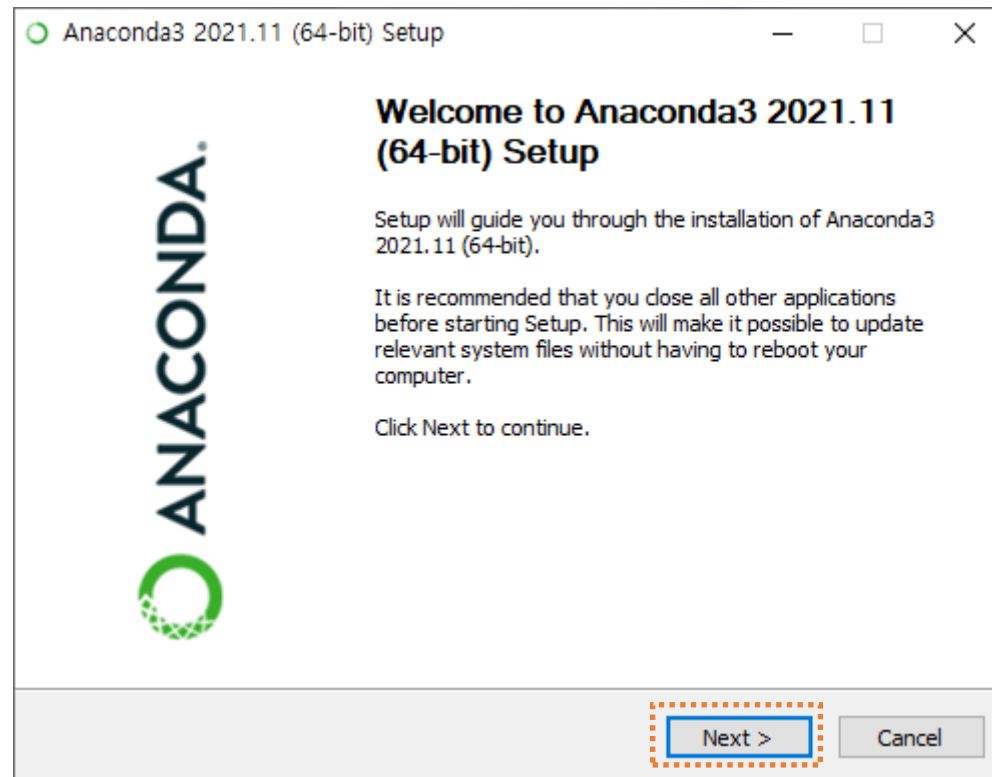
## ❖ OpenCV 설치: ① 아나콘다 설치하기 (2/10)

- [Download] 버튼 클릭



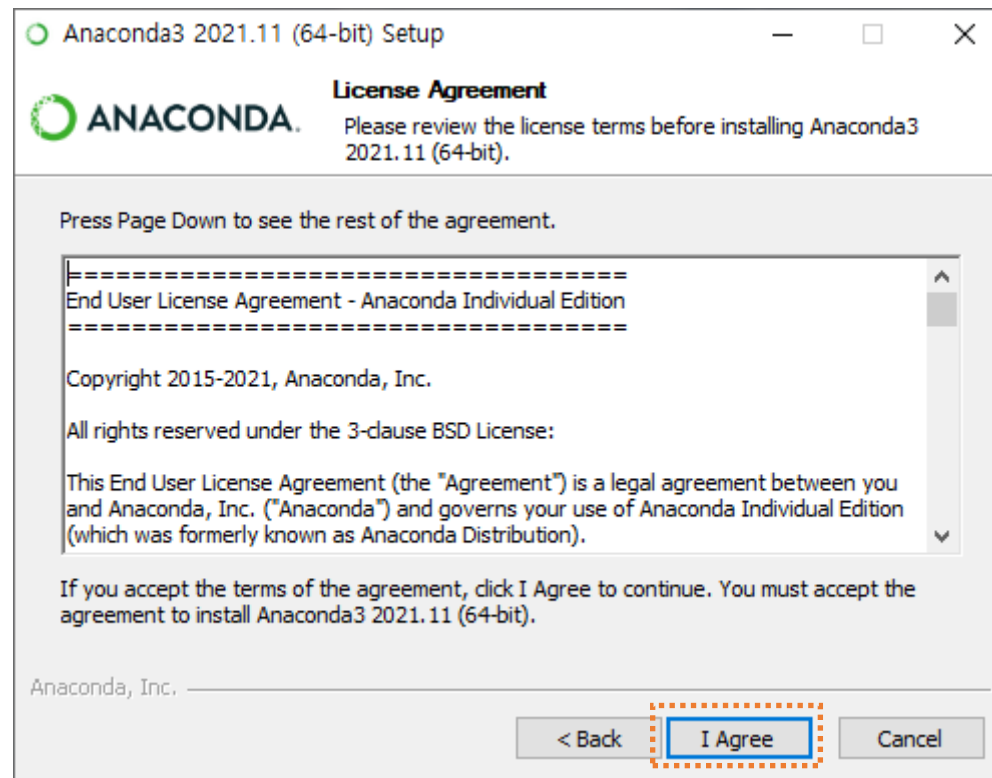
### ❖ OpenCV 설치: ① 아나콘다 설치하기 (3/10)

- 내려받은 파일을 실행해 다음 화면이 나오면 [Next] 버튼을 클릭하여 설치를 진행



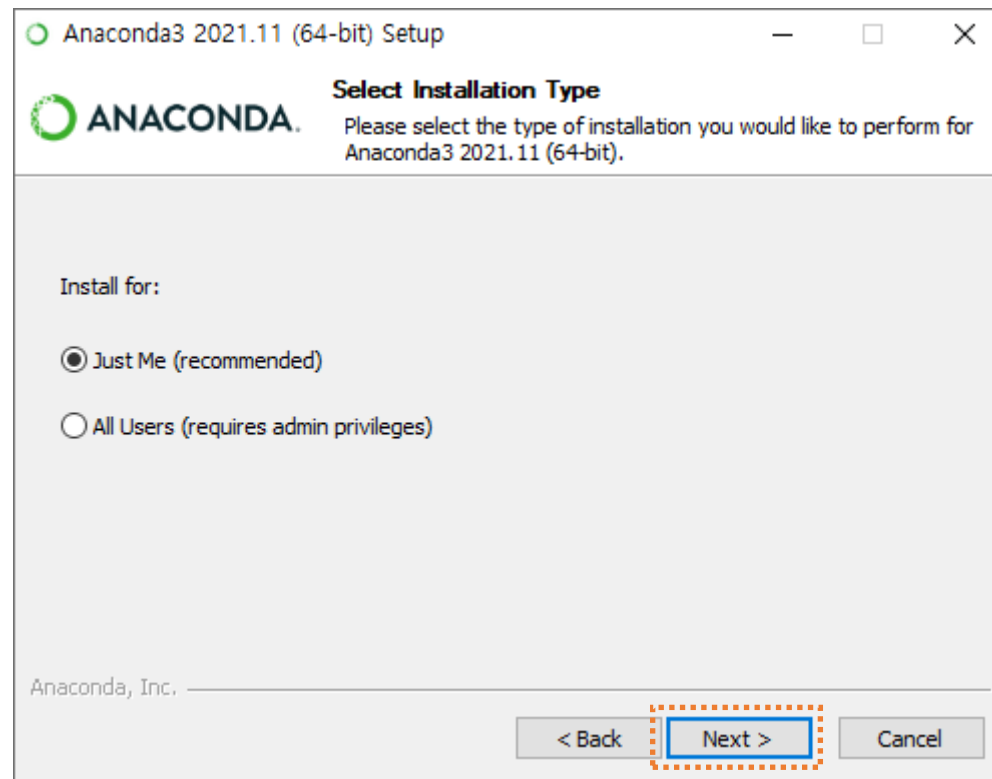
### ❖ OpenCV 설치: ① 아나콘다 설치하기 (4/10)

- 라이선스 동의 화면이 나오면, [I Agree] 버튼을 클릭



### ❖ OpenCV 설치: ① 아나콘다 설치하기 (5/10)

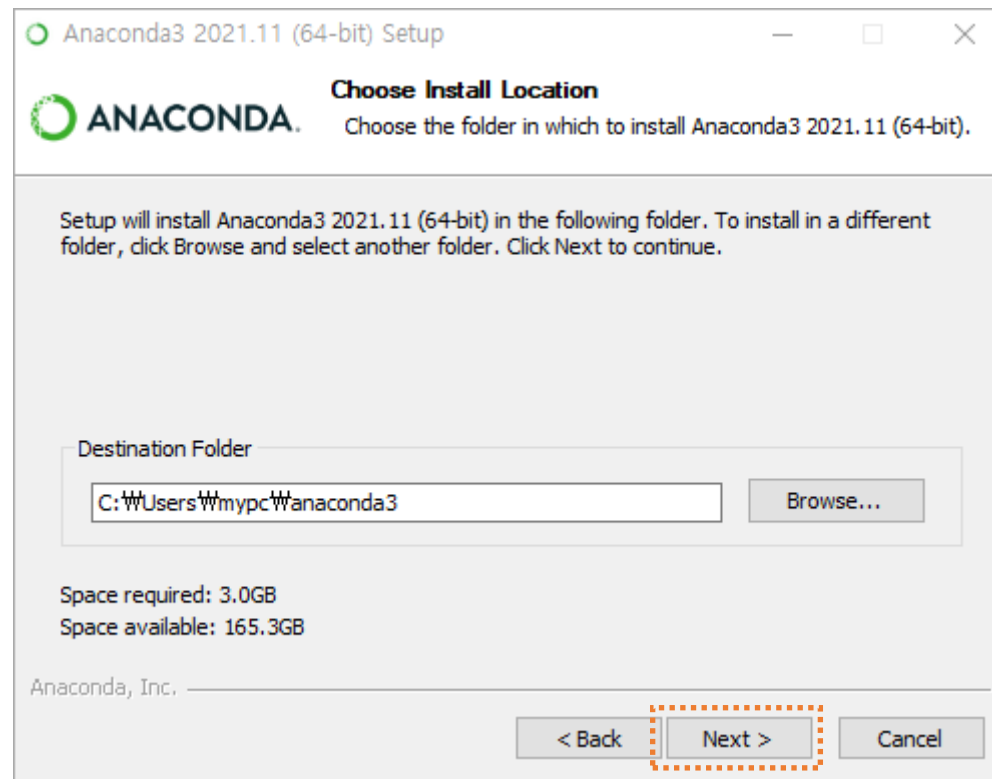
- 설치 타입을 묻는 창이 나오면, [Just Me (recommended)]를 선택하고 [Next] 버튼을 클릭





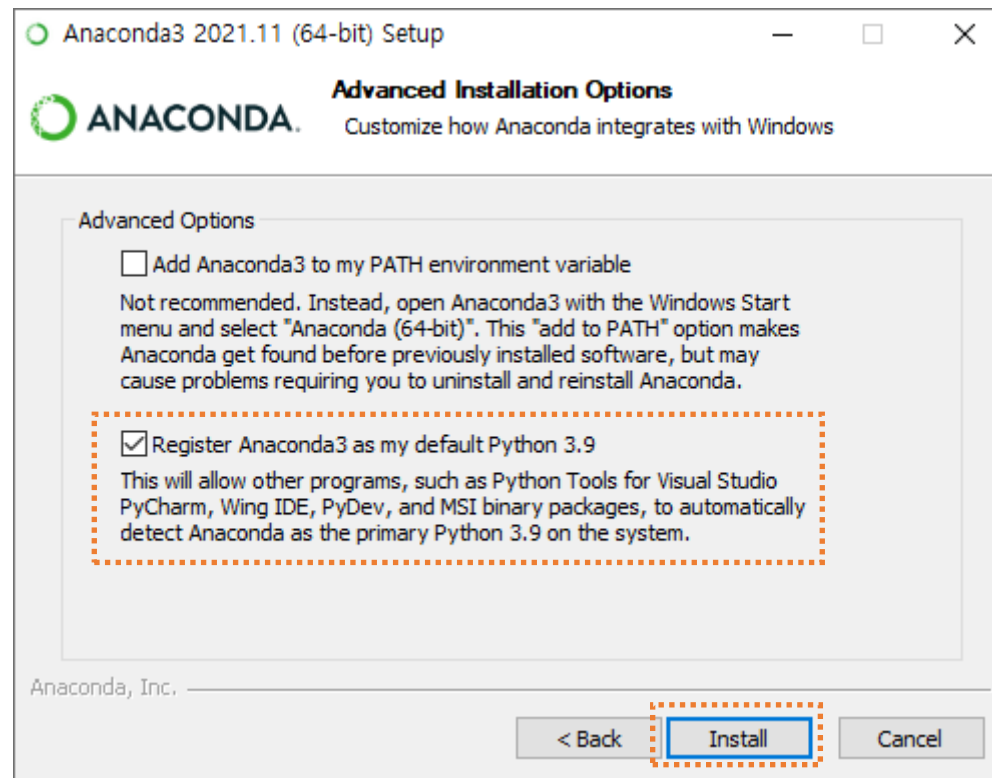
### ❖ OpenCV 설치: ① 아나콘다 설치하기 (6/10)

- 설치 경로를 묻는 창이 나오면 기본값으로 두거나 원하는 경로를 지정하고 [Next] 버튼 클릭



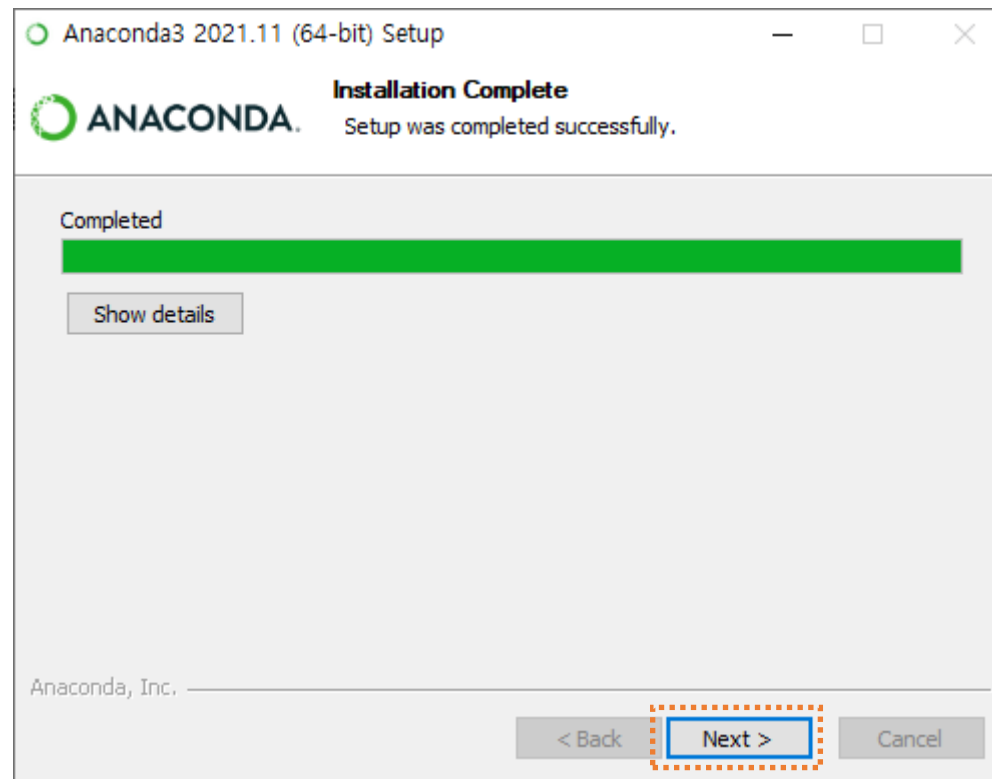
### ❖ OpenCV 설치: ① 아나콘다 설치하기 (7/10)

- 옵션을 설정하는 화면이 나오면 기본값으로 두고 [Install] 버튼을 클릭



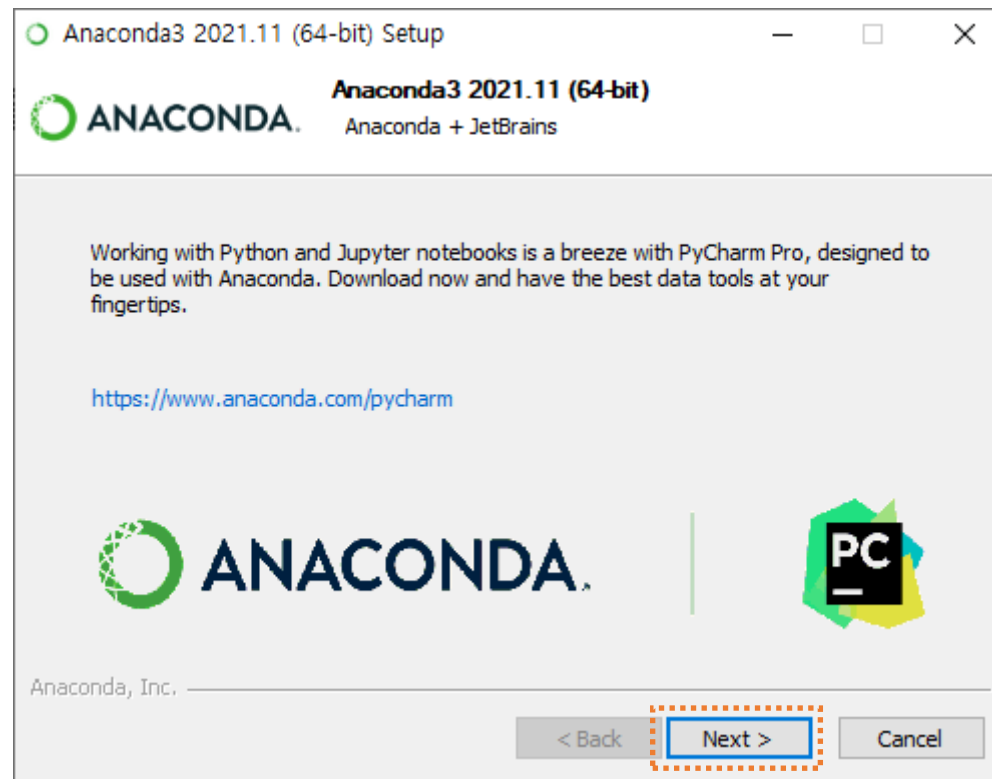
### ❖ OpenCV 설치: ① 아나콘다 설치하기 (8/10)

- 설치에는 몇 분 정도 소요되며, 다음과 같이 설치가 진행되면 [Next] 버튼을 클릭



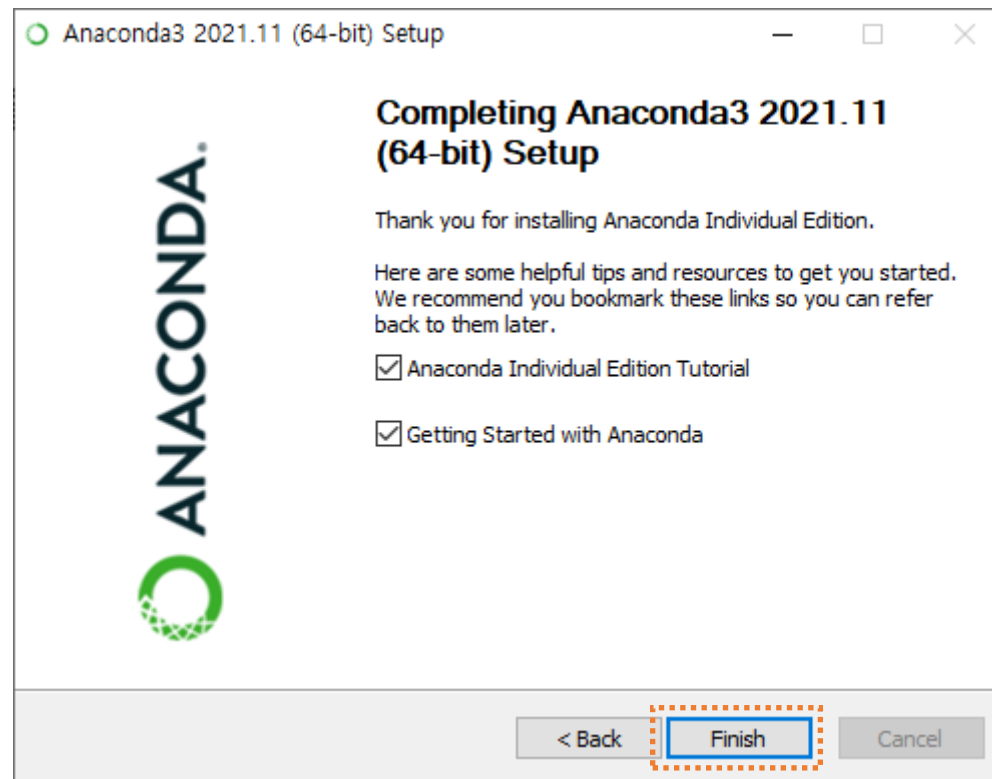
### ❖ OpenCV 설치: ① 아나콘다 설치하기 (9/10)

- [Next] 버튼을 클릭



### ❖ OpenCV 설치: ① 아나콘다 설치하기 (10/10)

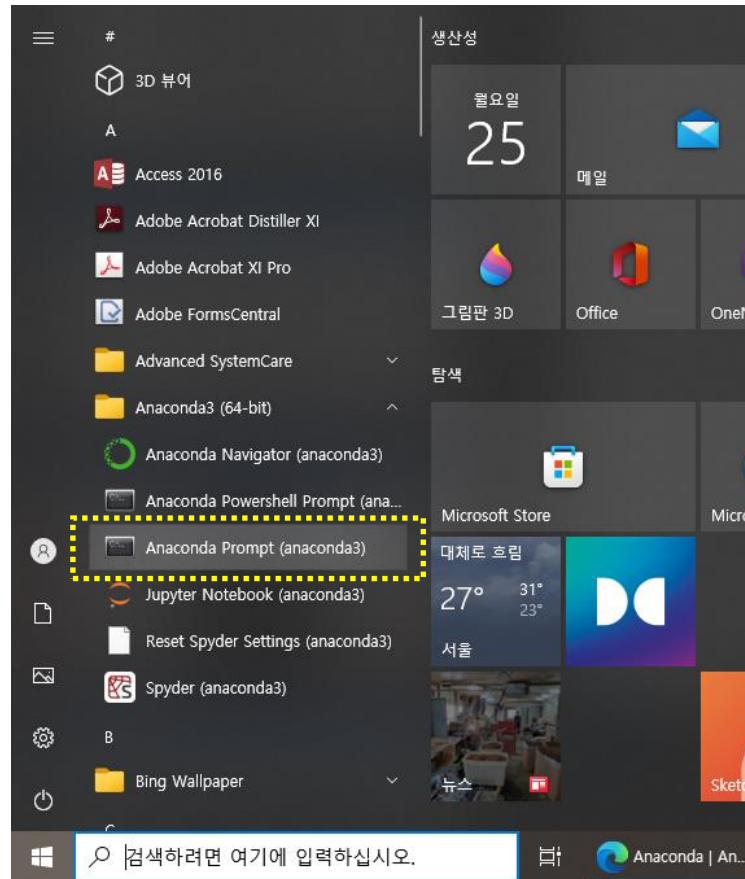
- 설치를 마쳤으면 [Finish] 버튼을 클릭



### ❖ OpenCV 설치: ② 아나콘다 프롬프트 실행하기

- 작업 표시줄의 윈도우 키를 눌러서 Anaconda3 폴더의 [Anaconda Prompt (anaconda3)] 아이콘을 클릭

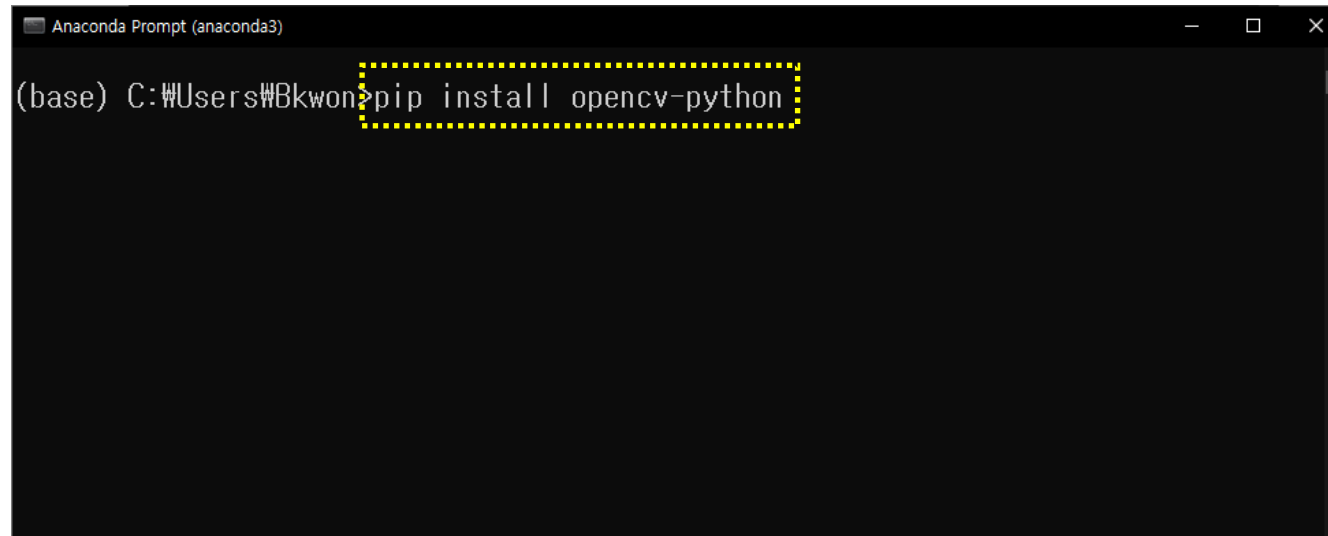
윈도우 키



## 2.1 OpenCV 개요와 설치

### ❖ OpenCV 설치: ③ pip으로 OpenCV 설치하기 (1/2)

- 아래 명령어를 입력하고 엔터키 누르기
  - ◆ pip install opencv-python

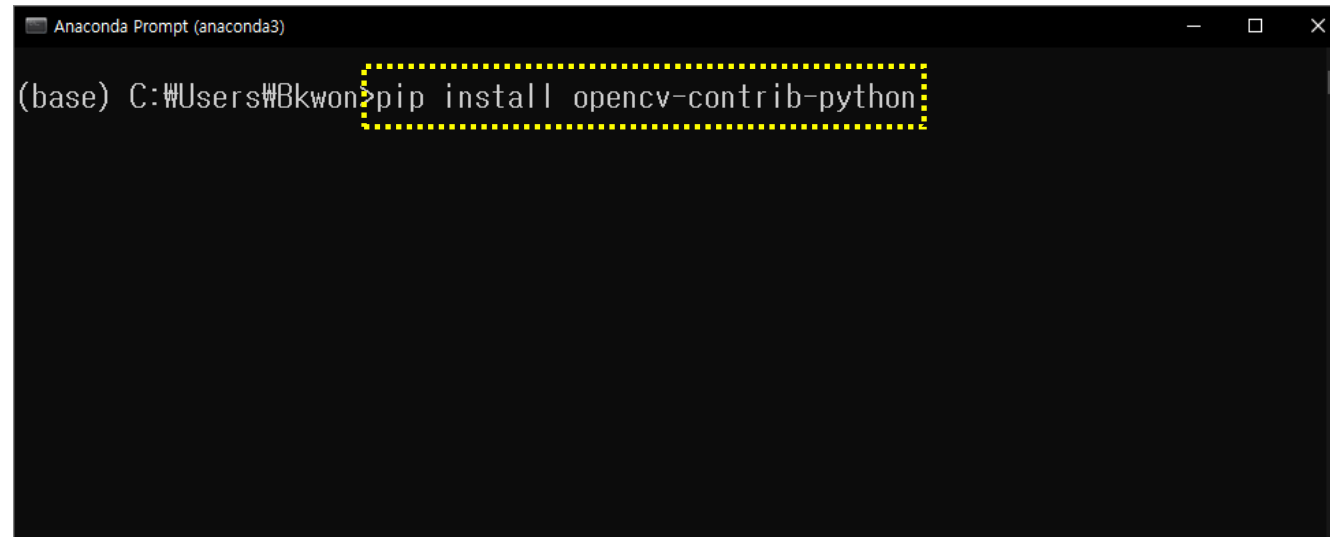


```
Anaconda Prompt (anaconda3)
(base) C:\Users\Bkwon>pip install opencv-python
```

## 2.1 OpenCV 개요와 설치

### ❖ OpenCV 설치: ③ pip으로 OpenCV 설치하기 (2/2)

- 아래 명령어를 입력하고 엔터키 누르기
  - ◆ pip install opencv-contrib-python

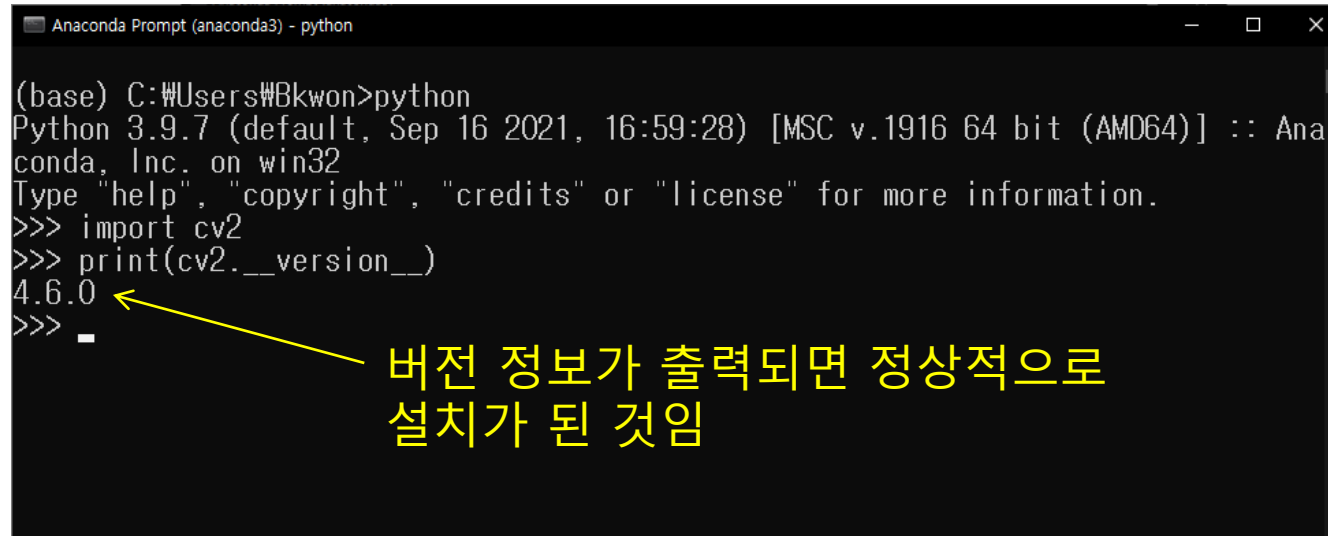


```
Anaconda Prompt (anaconda3)
(base) C:\Users\Bkwon>pip install opencv-contrib-python
```



### ❖ OpenCV 설치: ④ 설치 확인하기

- 아래 명령어를 순서대로 입력하고 엔터키 누르기
  - ◆ python
  - ◆ import cv2
  - ◆ print(cv2.\_\_version\_\_)



```
Anaconda Prompt (anaconda3) - python
(base) C:\Users\WBkwon>python
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)] :: Ana
conda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> print(cv2.__version__)
4.6.0
>>> _
```

버전 정보가 출력되면 정상적으로  
설치가 된 것임

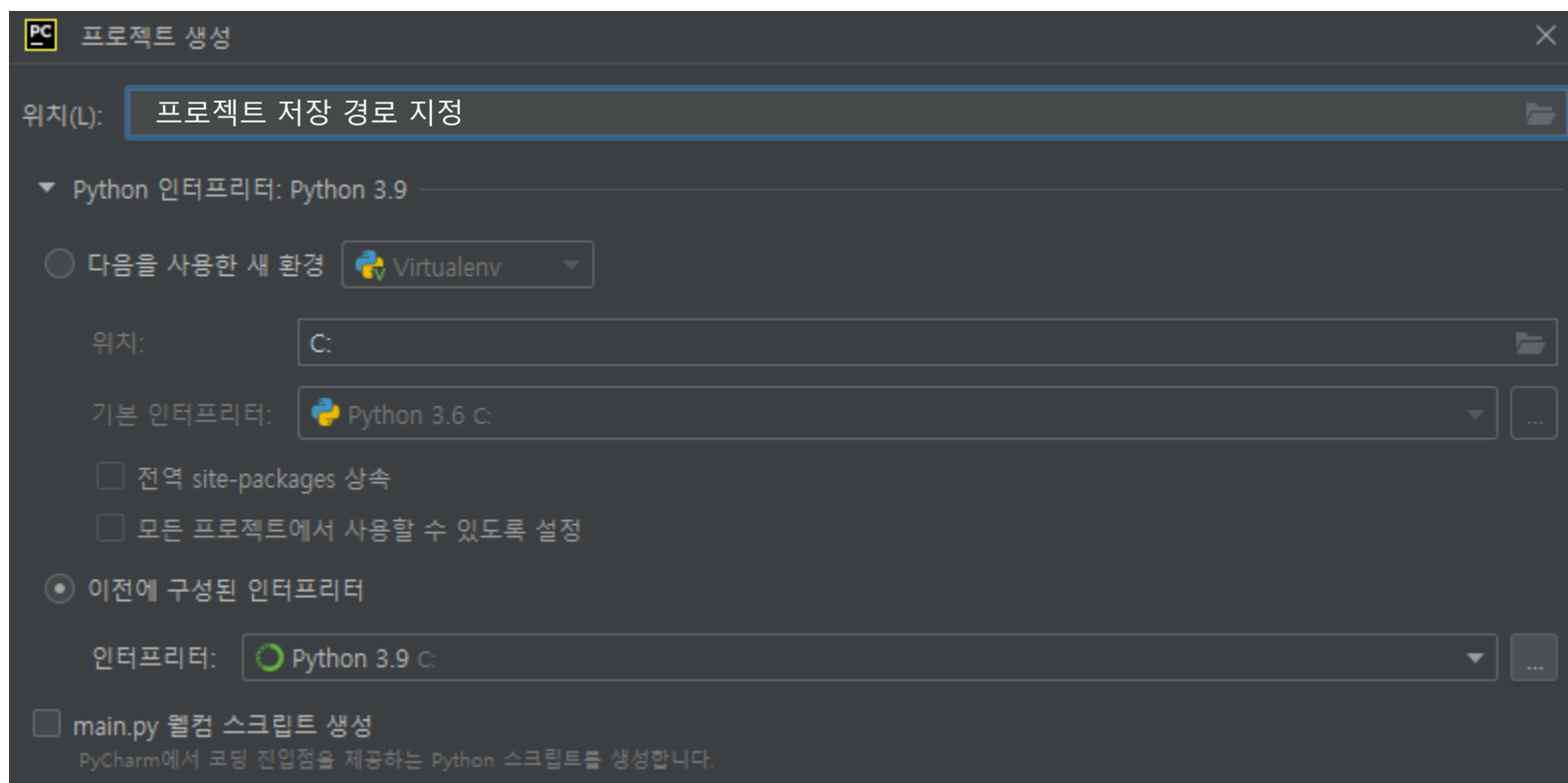
## 2.2 OpenCV 사용하기: HelloCV

### 2.1 OpenCV 개요와 설치

### ❖ OpenCV 프로젝트 만들기

- OpenCV 예제 프로그램을 만들기 위하여 먼저 PyCharm을 실행
- 새로운 프로젝트를 생성하기 위하여 메뉴에서 [파일]→[새 프로젝트]를 클릭

#### ▼ 그림 2-9 프로젝트 생성 창에서 프로젝트 만들기



### ❖ 영상을 화면에 출력하기 (1/6)

**코드 2-2** BMP 파일 영상을 화면에 출력하는 HelloCV 소스 코드 (HelloCV.py)

```
1  import sys
2  import cv2
3
4  print('Hello OpenCV', cv2.__version__)
5
6  img = cv2.imread('lenna.bmp')
7
8  if img is None:
9      print('Image load failed!')
10     sys.exit()
11
12  cv2.namedWindow('image')
13  cv2.imshow('image', img)
14  cv2.waitKey()
15  cv2.destroyAllWindows()
```

### ❖ 영상을 화면에 출력하기 (2/6)

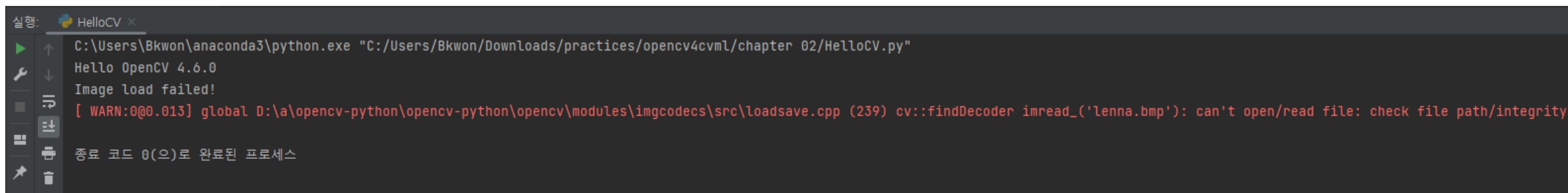
- HelloCV.py 소스 코드 설명

- 6행      OpenCV에서 제공하는 `imread()` 함수를 사용하여 `lenna.bmp` 파일을 불러와서 `img` 변수에 저장합니다. `imread()` 함수 이름은 영상 읽기(image read)를 뜻합니다.
- 8~10행    `lenna.bmp` 파일을 제대로 불러오지 못했을 경우에 대한 예외 처리 코드입니다. 프로그램 실행 디렉터리에 `lenna.bmp` 파일이 존재하지 않거나 혹은 `lenna.bmp` 파일이 손상되어 제대로 불러오지 못한 경우 `if` 문 블록 안으로 진입하여 "Image load failed!"라는 문자열을 출력한 후 프로그램을 종료합니다. 정상적으로 파일을 불러오면 12행 이하의 코드가 실행됩니다.
- 12행      `namedWindow()` 함수를 이용하여 영상을 화면에 나타내기 위한 새로운 창을 생성하고, 그 창에 "image"라는 이름을 부여합니다.
- 13행      `imshow()` 함수를 이용하여 "image"라는 이름의 창에 `img` 객체가 가지고 있는 `lenna.bmp` 영상을 출력합니다. `imshow()` 함수 이름은 영상 보여 주기(image show)를 의미합니다.
- 14~15행    `waitKey()` 함수는 사용자의 키보드 입력을 기다리는 함수이며, 사용자가 키보드를 누르기 전까지 영상을 화면에 나타나게 해 줍니다. 만약 사용자가 키보드에서 아무 키나 누르면 14행을 지나게 되고, 15행의 `destroyAllWindows()` 함수를 만나면 열려있는 모든 창을 닫고 프로그램이 종료됩니다.

### ❖ 영상을 화면에 출력하기 (3/6)

- 프로그램 소스 코드를 제대로 입력하였으면 이제 프로젝트를 실행해 보자
- 실제로 프로그램을 실행해 보면 영상이 화면에 출력되지 않고, 'Image load failed!' 문자열이 출력되고 프로그램이 종료됨

#### ▼ 그림 2-20 영상 불러오기 실패 시 HelloCV.py 실행 화면



```
실행: HelloCV x
C:\Users\Bkwon\anaconda3\python.exe "C:/Users/Bkwon/Downloads/practices/opencv4cvml/chapter 02/HelloCV.py"
Hello OpenCV 4.6.0
Image load failed!
[ WARN:0@0.013] global D:\a\opencv-python\opencv-python\opencv\modules\imgcodecs\src\loadsave.cpp (239) cv::findDecoder imread_('lenna.bmp'): can't open/read file: check file path/integrity
종료 코드 0(으)로 완료된 프로세스
```

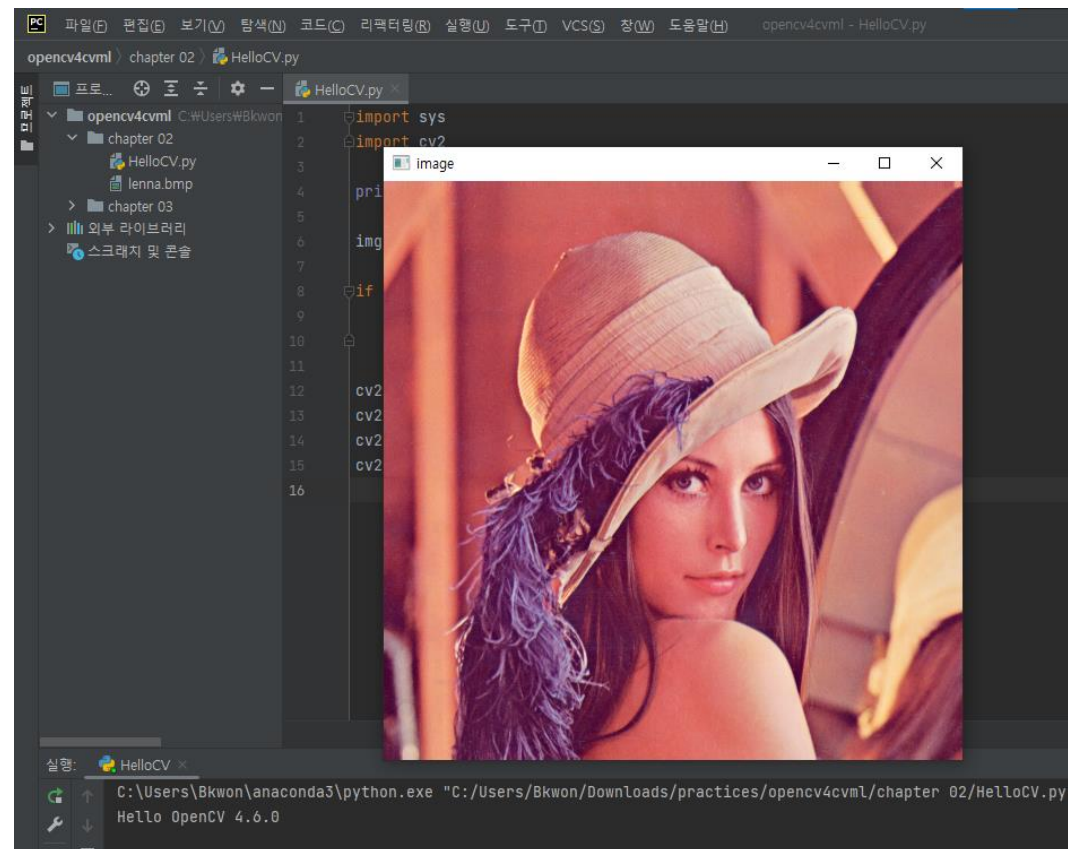
### ❖ 영상을 화면에 출력하기 (4/6)

- 'Image load failed!' 문자열이 출력되었다는 것은 HelloCV.py 코드의 8~10행 if 조건문 블록이 수행되었다는 것을 의미함
- 즉, 프로그램이 실행되면서 lenna.bmp 파일을 불러오는 작업이 실패한 것임
- 새로 만든 프로젝트 폴더에 lenna.bmp 파일이 존재하지 않기 때문임
- lenna.bmp 파일을 여러분이 만든 프로젝트 폴더에 저장 또는 이동

### ❖ 영상을 화면에 출력하기 (5/6)

- 다시 실행해 보면 image라는 이름의 창에 lenna.bmp 영상이 나타나는 것을 확인할 수 있음

#### ▼ 그림 2-21 lenna.bmp 파일 영상을 화면에 출력하기





### ❖ 영상을 화면에 출력하기 (6/6)

- image 창에 나타난 영상이 lenna.bmp 파일에 저장되어 있는 레나(lenna) 영상임
- 레나 영상은 영상 처리 및 컴퓨터 비전 분야에서 테스트용으로 널리 사용되는 영상임
- 레나 영상은 이 책에서 설명하는 다양한 예제 프로그램에서도 테스트 영상으로 자주 사용됨
- HelloCV.py는 Image라는 이름의 새 창에 레나 영상을 출력하고,  
사용자가 키보드의 아무 키나 누르면 창이 사라지면서 프로그램이 종료됨

### ❖ 주요 함수 설명: ① **cv2.imread()** 함수 (1/2)

- 가장 먼저 영상 파일을 불러올 때 사용한 imread() 함수에 대해 알아보자

**cv2.imread(filename, flags)**

filename	불러올 영상 파일 이름/경로
flags	영상 파일 불러오기 옵션 플래그
반환값	불러온 영상 데이터

- imread() 함수는 filename 영상 파일을 불러와 NumPy ndarray 객체로 변환하여 반환함
- filename에 "lenna.bmp"처럼 파일 이름만 지정하면 프로그램 작업 폴더에 위치한 lenna.bmp 파일을 불러옴
- 만약 다른 폴더의 파일을 불러오려면 절대 경로 또는 상대 경로 형식으로 파일 위치를 지정해야 함

### ❖ 주요 함수 설명: ① **cv2.imread()** 함수 (2/2)

- imread() 함수의 두 번째 인자 flags는 영상 파일을 불러올 때 사용할 컬러 모드와 영상 크기를 지정하는 플래그임
- flags 인자에는 세 가지 옵션 중 한 가지를 선택하여 지정할 수 있음
- flags 인자는 기본값으로 cv2.IMREAD\_COLOR가 지정되어 있기 때문에 imread() 함수 호출 시 두 번째 인자를 지정하지 않으면 자동으로 3채널 컬러 영상 형식으로 영상을 불러옴

#### ▼ 표 2-4 flags 인자의 옵션 종류

flags 인자의 옵션 종류	설명
cv2.IMREAD_COLOR	3채널 컬러 영상 형식으로 영상을 불러옵니다. 투명한 부분은 무시되며, 기본값입니다.
cv2.IMREAD_GRAYSCALE	1채널 그레이스케일 영상으로 변환하여 불러옵니다.
cv2.IMREAD_UNCHANGED	입력 파일에 지정된 그대로의 컬러 속성을 사용합니다. 투명한 PNG 또는 TIFF 파일의 경우, 알파 채널까지 이용하여 4채널 영상으로 불러옵니다.

### ❖ 주요 함수 설명: ② `cv2.imwrite()` 함수

- NumPy ndarray 객체에 저장되어 있는 영상 데이터를 파일로 저장하기 위해서는 `imwrite()` 함수를 사용함

`cv2.imwrite(filename, img)`

<code>filename</code>	저장할 영상 파일 이름
<code>img</code>	저장할 영상 데이터(NumPy ndarray 객체)
반환값	정상적으로 저장하면 True, 실패하면 False를 반환

- `imwrite()` 함수는 `img` 변수에 저장되어 있는 영상 데이터를 `filename` 이름의 파일로 저장함
- 영상 파일 형식은 `filename` 문자열에 포함된 파일 확장자에 의해 결정됨

### ❖ 주요 함수 설명: ③ `cv2.namedWindow()` 함수 (1/2)

- NumPy ndarray 객체에 저장되어 있는 영상 데이터를 화면에 나타내기 위해서는 먼저 영상 출력을 위한 빈 창을 생성해야 함
- 이때 사용하는 함수가 `namedWindow()`이며, 이 함수의 원형은 다음과 같음

```
cv2.namedWindow(winname, flags)
```

<code>winname</code>	영상 출력 창 상단에 출력되는 창 고유 이름. 이 문자열로 창을 구분
<code>flags</code>	생성되는 창의 속성을 지정하는 플래그

- Windows 운영 체제에서는 각각의 창을 구분하기 위해 핸들(handle)이라는 숫자 값을 사용하지만, OpenCV에서는 각각의 창에 고유한 문자열을 부여하여 각각의 창을 구분함
- 새로운 창을 만들 때에는 `winname` 인자에 고유한 문자열을 지정해야 함
- `winname`으로 지정한 창의 고유 이름은 실제 생성되는 창의 상단 제목 표시줄에 출력됨

### ❖ 주요 함수 설명: ③ `cv2.namedWindow()` 함수 (2/2)

- `namedWindow()` 함수의 두 번째 인자 `flags`는 새로 생성하는 창의 속성을 지정하는 용도로 사용됨

#### ▼ 표 2-5 flags 인자의 옵션 종류

flags 인자의 옵션 종류	설명
<code>cv2.WINDOW_NORMAL</code>	영상 출력 창의 크기에 맞게 영상 크기가 변경되어 출력됩니다. 사용자가 자유롭게 창 크기를 변경할 수 있습니다.
<code>cv2.WINDOW_AUTOSIZE</code>	출력하는 영상 크기에 맞게 창 크기가 자동으로 변경됩니다. 사용자가 임의로 창 크기를 변경할 수 없습니다.

### ❖ 주요 함수 설명: ④ `cv2.destroyAllWindows()` & `cv2.destroyAllWindows()` 함수 (1/2)

- `namedWindow()` 함수에 의해 생성된 영상 출력 창은 `destroyWindow()` 또는 `destroyAllWindows()` 함수를 이용하여 닫을 수 있음
- `destroyWindow()` 함수는 하나의 창을 닫을 때 사용하고,  
`destroyAllWindows()` 함수는 열려 있는 모든 창을 닫을 때 사용함

```
cv2.destroyAllWindows()
```

```
cv2.destroyAllWindows()
```

winname

소멸시킬 창 이름

### ❖ 주요 함수 설명: ④ `cv2.destroyAllWindows()` & `cv2.destroyAllWindows()` 함수 (2/2)

- 일반적으로 OpenCV 응용 프로그램이 완전히 종료되는 경우에는 운영 체제에 의해 OpenCV 응용 프로그램이 사용하던 모든 자원이 해제됨
- `namedWindow()` 함수에 의해 만들어진 창도 모두 자동으로 닫힘
- HelloCV 프로그램 소스 코드에서도 `destroyWindow()` 또는 `destroyAllWindows()` 함수를 명시적으로 호출하지 않더라도 프로그램 종료 시 영상 출력 창이 자동으로 닫히는 것을 확인할 수 있음
- 프로그램 동작 중에 창을 닫고 싶을 때에는 `destroyWindow()` 또는 `destroyAllWindows()` 함수를 이용해야 함



### ❖ 주요 함수 설명: ⑤ `cv2.moveWindow()` 함수

- OpenCV의 영상 출력 창과 관련된 함수 중에는 창 크기를 바꾸거나 위치를 바꿀 수 있는 함수도 있음
- 먼저 창 위치를 변경하는 함수는 `moveWindow()`임

`cv2.moveWindow(winname, x, y)`

<code>winname</code>	위치를 이동할 창 이름
<code>x</code>	창이 이동할 위치의 x 좌표
<code>y</code>	창이 이동할 위치의 y 좌표

- `moveWindow()` 함수는 `winname` 이름의 창을 (x, y) 좌표 위치로 이동시킴
- 여기서 (x, y) 좌표는 모니터 전체 화면에서의 좌표를 나타내며, 모니터 좌측 상단을 원점으로 간주함

### ❖ 주요 함수 설명: ⑥ `cv2.resizeWindow()` 함수

- 프로그램 동작 중에 영상 출력 창의 크기를 변경하고 싶다면 `resizeWindow()` 함수를 사용함

```
cv2.resizeWindow(winname, width, height)
```

winname	크기를 변경할 창 이름
width	창의 가로 크기
height	창의 세로 크기

- 이때 함수의 인자로 전달하는 `width`와 `height` 크기는 창 전체 크기가 아니라 창의 뷰(view) 영역에 나타나는 영상 크기를 의미함
- `resizeWindow()` 함수에 의해 변경된 창 크기는 창의 제목 표시줄, 경계선 두께로 인해 `width`와 `height` 크기보다 약간 큰 형태로 결정됨
- 다만 `WINDOW_AUTOSIZE` 플래그를 사용하여 만들어진 영상 출력 창은 `resizeWindow()` 함수로 크기를 변경할 수 없음

### ❖ 주요 함수 설명: ⑦ `cv2.imshow()` 함수 (1/2)

- 이번에는 NumPy ndarray 객체에 저장된 영상 데이터를 화면에 출력하는 `imshow()` 함수에 대해 알아보자

```
cv2.imshow(winname, img)
```

<code>winname</code>	영상을 출력할 대상 창 이름
<code>img</code>	출력할 영상 데이터(NumPy ndarray 객체)

- NumPy ndarray 객체에 저장된 영상이 1채널로 구성된 그레이스케일 영상이라면 픽셀 값을 그대로 그레이스케일 밝기 형태로 나타냄
- NumPy ndarray 객체에 저장된 영상이 3채널 컬러 영상이라면 색상 채널이 파란색(Blue), 녹색(Green), 빨간색(Red) 순서로 되어 있다고 간주하여 색상을 표현함

### ❖ 주요 함수 설명: ⑦ `cv2.imshow()` 함수 (2/2)

- 만약 `imshow()` 함수가 호출되는 시점에 `winname`에 해당하는 창이 없으면 `imshow()` 함수는 자동으로 `WINDOW_AUTOSIZE` 속성의 창을 새로 만들어서 영상을 출력함
- 참고로 Windows 운영 체제에서는 [Ctrl] + [C] 키를 눌러 영상 출력 창에 나타난 영상 데이터를 비트맵 형식으로 클립보드로 복사할 수 있으며, [Ctrl] + [S] 키를 눌러서 파일 형태로 저장할 수 있음

### ❖ 주요 함수 설명: ⑧ `cv2.waitKey()` 함수

- HelloCV 예제 프로그램에서 마지막으로 사용한 OpenCV 함수는 `waitKey()` 임
- 이 함수는 사용자로부터 키보드 입력을 받는 용도로 사용됨

```
cv2.waitKey(delay = 0)
```

delay	키 입력을 기다릴 시간(밀리미터초 단위). $\text{delay} \leq 0$ 이면 무한히 기다림
반환값	눌린 키 값. 지정한 시간 동안 키가 눌리지 않았으면 -1을 반환

# THANK YOU!

## Q & A

- Name: 권범
- Office: 동양미래대학교 2호관 704호 (02-2610-5238)
- E-mail: [bkwon@dongyang.ac.kr](mailto:bkwon@dongyang.ac.kr)
- Homepage: <https://sites.google.com/view/beomkwon/home>