CH. 7. OpenCV 기초

Section 01 OpenCV 소개

❑ OpenCV의 개요

- OpenCV는 **오픈 소스 컴퓨터 비전 라이브러리** (Open Source Computer Vision Library)**를 줄여 쓴 말이다**.
- OpenCV는 **영상 처리와 컴퓨터 비전 프로그래밍 분야의 가장 대표적인 라이브러리이다**.
- **예전에는 프로그래밍 영역 중에서도 영상 처리나 컴퓨터 비전 분야는 대학원에서나 본격적으로 접하게 될 만큼 비교적 전문적인 분야였다**.
- **매우 복잡한 수학적 알고리즘을** C/C++ **언어로 구현해야 했으므로 체계적인 수학 지식과 뛰어난 프로그래밍 실력을 갖추고 있지 않으면 이해하기 어려웠기 때문이다**.
- **하지만, 이제는** OpenCV**와 같은 훌륭한 라이브러리가 있어 기초적인 수학 지식만으로도 이미 구현된 알고리즘을 손쉽게 사용할 수 있게 된 데다가, 메모리 주소 하나 하나를 직접 다뤄야 했던 어렵고 복잡한** C **언어가 아닌 비교적 배우기 쉽고 빠르게 구현할 수 있는 파이썬 언어로** OpenCV**를 사용할 수 있게 되면서 영상 처리와 컴퓨터 비전분야의 문턱이 무척 낮아졌다**.

Section 02 이미지 데이터 기초

❑ RGB 데이터

- 이미지 〔 화상 〕 데이터는 컴퓨터에서 숫자로 관리된다.
- 이미지는 픽셀(pixel)이라는작은 입자의 집합으로 표현된다.
- 화면을 구성하는 가장 기본이 되는 단위로, 모습은 대부분 사각형이다.
- 이러한 픽셀에 색을 입혀 이미지를 표현하고 있다.
- 컬러 이미지는 Red, Green, Blue (머릿글자를 따서 RGB로 표현한다)의 3색으로 표현된다.
- 그리고 이 3색의 밝기(농도)를0~255(8비트)의 수치로 나타낸다.
- 수치가 클수록 밝아진다.
- 예를 들어 단순한 빨간색은 (255, 0, 0)으로 나타낸다.
- 진홍색magenta은 (255, 0, 255), 검은색은(0, 0, 0), 흰색은 (255, 255, 255)이다.

Section 02 이미지 데이터 기초

❑ RGB 데이터

- 특수한 경우지만 흑백 사진의 경우 단순히 픽셀의 밝기(0~255)정보만 주어진다.
- 컬러 이미지에 비해 데이터 양이 3분의 1로 줄어든다.
- 이후에 다룰 OpenCV에서는 하나의 픽셀을 나타내기 위한 요소 수를 채널수라고 한다.
- 예를 들어 RGB 이미지는 채널 수가3이며, 흑백 이미지는 채널 수가 1 이다.

Section 02 이미지 데이터 기초

❑ 이미지 데이터 형식

- **이미지 데이터와 특징**

	PNG	JPG	PDF	GIF
특징	무손실 압축이 가능하다. 여러 색상을 재현할 수 있다.	여러 색상을 재현할 수 있다. 작은 용량으로 압축할 수 있다(하지만 질이 떨어짐).	화질이 좋고, 확대해도 거칠어 보이지 않는다. 용량이 크다.	표현할 수 있는 색상은 적지만 그만큼 용량은 작아진다. 애니메이션을 표현할 수 있다.

- **무손실 압축이란 압축된 이미지를 원래대로 완전히 복원할 수 있는 것을 말한다.**
- **손실 압축의 경우에는 완전히 복원할 수 없다.**

Section 02 이미지 데이터 기초

❑ 투명 데이터

- 이미지의 배경을 투명하게 하려면 소프트웨어를 사용해서 수정하거나 이미지 작성 단계에서 투명 효과를 적용하는 등 여러 가지 방법이 있다.
- 투명 처리는 프로그램이 색상을 취급하는 방법에 따라 달라진다.
- 예를 들어 BMP 이미지는 투명 효과를 지원하지 않지만BMP로 만든 아이콘 이미지가 투명하게 보일 경우가 있다.
- 이것은 아이콘을 표시하는 프로그램이 특정 위치의 색을 투명한 색으로 다루었기 때문이다.
- 이미지 자체로 투명 효과를 지원하는 것은 GIF 와 PNG 이다.

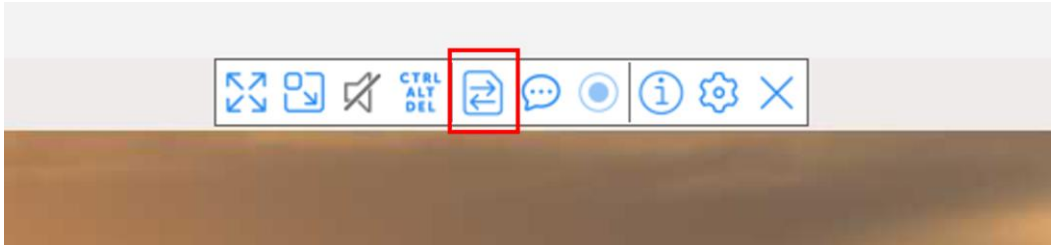
Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

- **앞으로 예제 실행에 필요한 영상 파일들은 다음 링크에서 다운로드할 수 있다.**

  ➢ https://github.com/jjin300/smartcar2

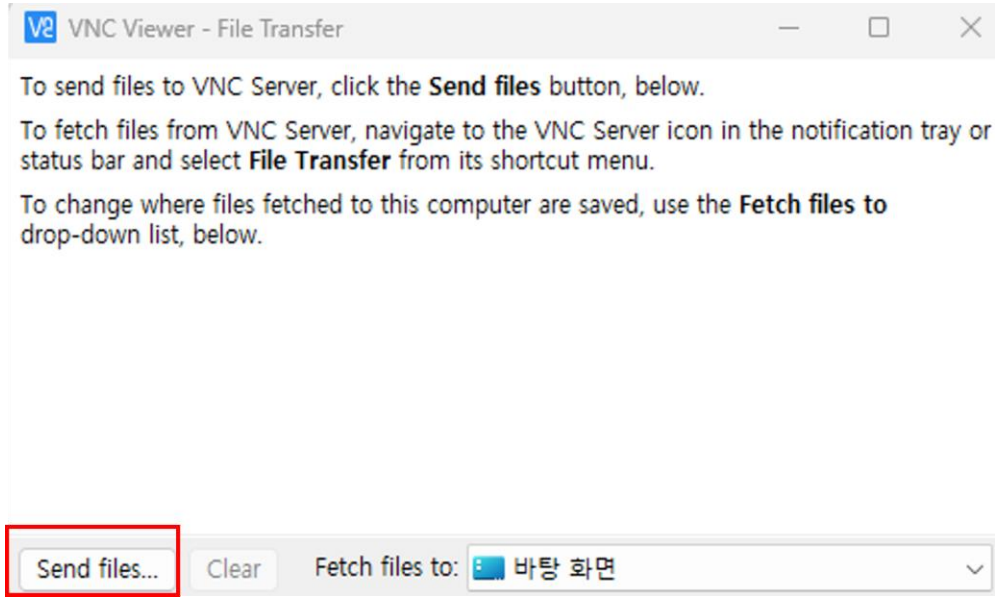
- **다운로드한** img**폴더를 압축을 한 다음** vnc**에서** File Transfer **메뉴를 실행한다.**



Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

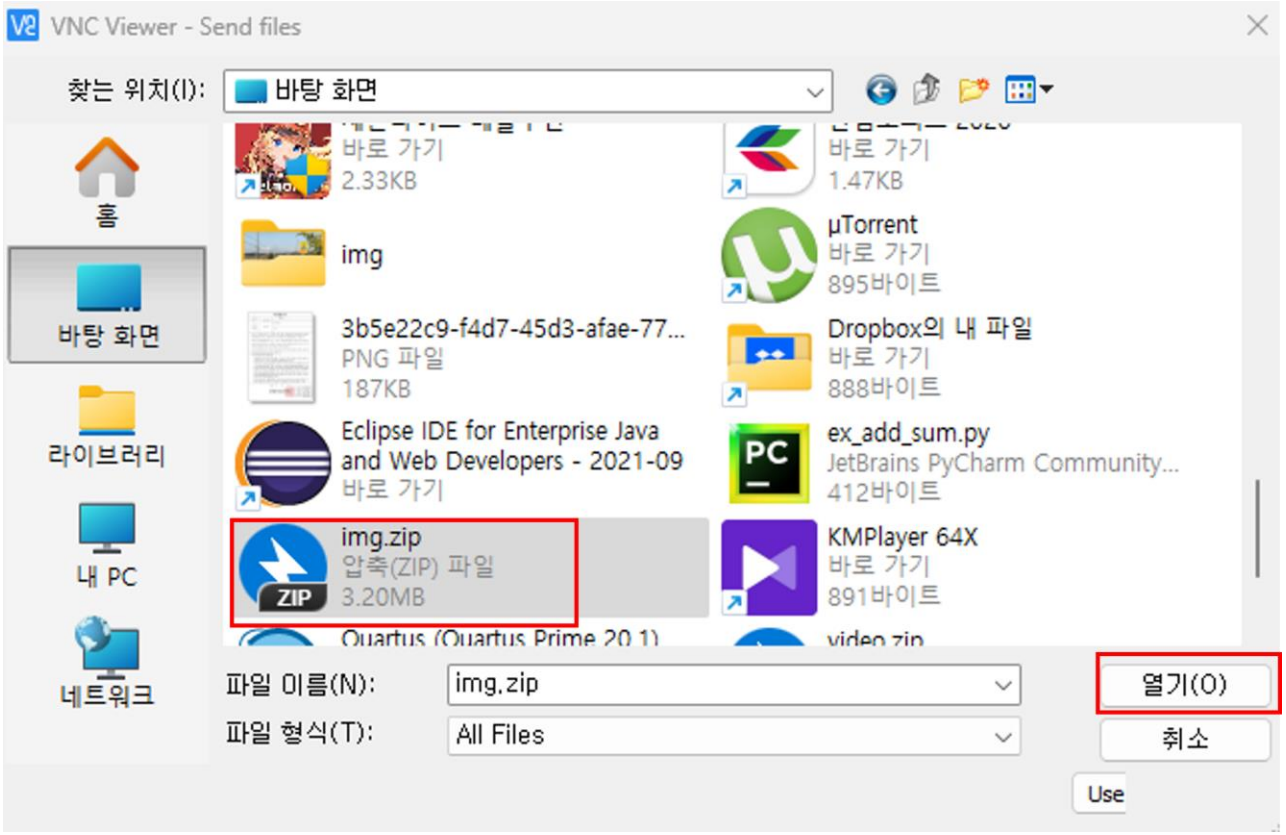
- File Transfer **메뉴에서** Send files **버튼을 클릭한다**.



Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

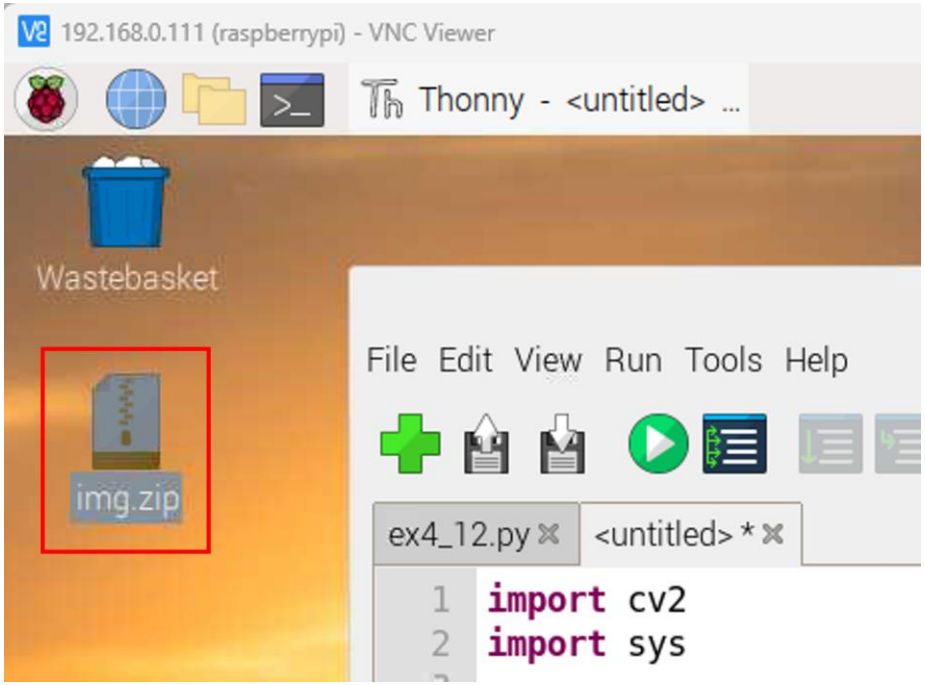
- **앞에서 압축한** img.zip **파일을 선택한다.**



Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

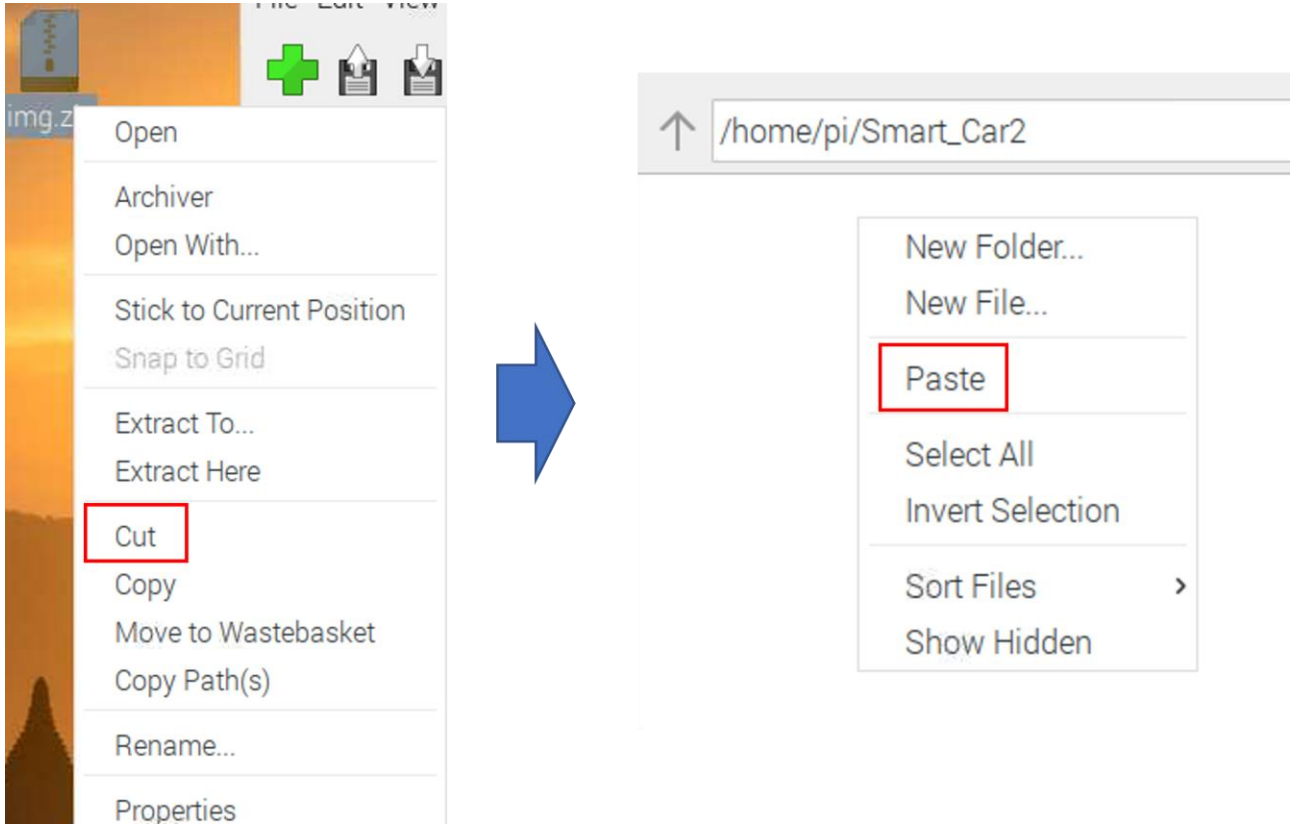
- 열기 버튼을 클릭하면 라즈베리 파이 바탕화면에 압축파일이 생성된 것을 확인할 수 있다.



Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

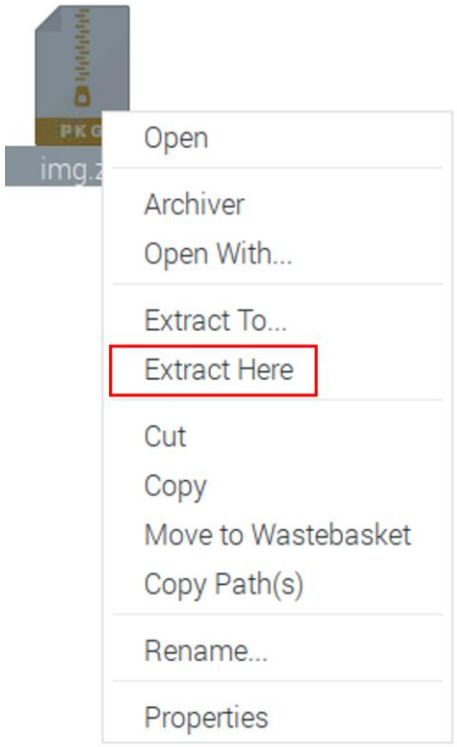
- **압축한 파일을** Smart_Car2 **폴더로 이동시킨다.**



Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

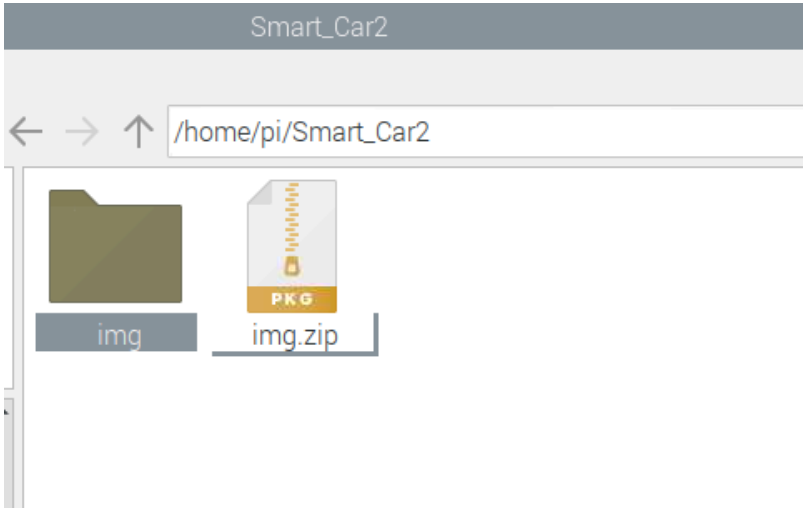
- **이동한** img.zip **파일의 압축을 푼다.**



Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

- **이제 새로운** img **폴더가 생성되었다.**



Section 03 영상을 읽고 표시하기

❑ 처음 해보는 OpenCV 프로그래밍

- **예제 7-1은 영상 파일을 읽고 화면에 표시하는 프로그램이다.**

```
1   import cv2
2   import sys
3
4   img = cv2.imread('./img/lena.jpg')
5
6   if img is not None:
7       sys.exit("파일을 찾을 수 없습니다.")
8
9   cv2.imshow("Image Display", img)
10
11  cv2.waitKey()
12  cv2.destroyAllWindows()
```

Section 03 영상을 읽고 표시하기

❑ OpenCV 에서 영상은 numpy.ndarray 클래스 형의 객체

- type과 dir 명령어로 4행에서 생성된 img 객체의 정체를 확인해보자.

```
1   import cv2
2   import sys
3
4   img = cv2.imread('./img/lena.jpg')
5
6   if img is not None:
7       Sys.exit("파일을 찾을 수 없습니다.")
8
9   cv2.imshow("Image Display", img)
10
11  print(type(img))
12  print(img.shape)
13
14  cv2.waitKey()
15  cv2.destroyAllWindows()
```

Section 03 영상을 읽고 표시하기

❑ OpenCV 에서 영상은 numpy.ndarray 클래스 형의 객체

- Img**의 자료형과 차원**

```
Shell
>>> %Run ex7_1.py
  <class 'numpy.ndarray'>
  (822, 1200, 3)
>>>
```

- type(img) **명령어를 통해** img **객체는** numpy.ndarray **클래스임을 알 수 있다**.
- numpy**는 다차원 배열을 위한 사실상 표준 모듈이다**.
- **이런 이유로** OpenCV**는 영상을 표현하는데** numpy**를 활용한다**.
- img. shape**으로** img**의 배열 모양을 알아본 바에 따르면** 822 x 1200 x 3 **크기의** 3**차원 배열이다**.

Section 03 영상을 읽고 표시하기

❑ OpenCV 에서 영상은 numpy.ndarray 클래스 형의 객체

- img 객체가 표현하는 영상은 822개의 행과 1200개의 열을 가진 채널 3개로 구성된다.
- 3개의 채널은 앞쪽부터 blue, green, red에 해당한다.
- 보통 RGB 순서인데 OpenCV는 기본이 BGR이다.
- 영상을 구성하는 한 점을 화소pixel라 한다.
- pixel은 picture element의 약어다.
- 영상에서 화소의 위치는 행 좌표 r과 열 좌표 c를 이용해 (r, c)로 표기한다.
- 행 좌표는y축, 열 좌표는 x축에 해당한다.
- 수학의 좌표계와 달리 컴퓨터 비전에서는 왼쪽 위를 원점으로 간주하며, 점의 좌표를 쓸 때 (y, x) 또는 (r, c)와 같이 y축을 먼저 쓴다.
- 컬러 영상은 한 화소가 blue, green, red 의 3개 값을 가진다.

Section 03 영상을 읽고 표시하기

❑ OpenCV 에서 영상은 numpy.ndarray 클래스 형의 객체

- (0,0)과 (0,1)에 있는 화소값을 조사해보자.

```
1   import cv2
2   import sys
3
4   img = cv2.imread('./img/lena.jpg')
5
6   if img is not None:
7       Sys.exit("파일을 찾을 수 없습니다.")
8
9   cv2.imshow("Image Display", img)
10
```

Section 03 영상을 읽고 표시하기

❑ OpenCV 에서 영상은 numpy.ndarray 클래스 형의 객체

- (0,0)과 (0,1)에 있는 화소값을 조사해보자.

```
11  print(type(img))
12  print(img.shape)
13  print(img[0, 0, 0], img[0, 0, 1], img[0, 0, 2])
14  print(img[0, 1, 0], img[0, 1, 1], img[0, 1, 2])
15
16  cv2.waitKey()
17  cv2.destroyAllWindows()
```

Shell

```
<class 'numpy.ndarray'>
(822, 1200, 3)
14 45 84
13 44 83

>>>
```

Section 04 영상처리

❑ 영상 크기 변경

- **영상에는 다양한 형태가 있다.**
- ex7_2.py**에서는 채널이** 3**개인 컬러 영상을 채널이** 1**개 뿐인 명암 영상**(grayscale image)**으로 변환하고 영상 크기를 반으로 축소하는 실험을 한다.**

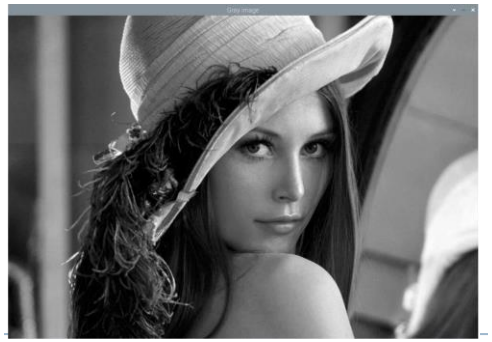
```
1   import cv2
2   import sys
3
4   img = cv2.imread('./img/lena.jpg')
5
6   if img is not None:
7       sys.exit("파일을 찾을 수 없습니다.")
8
9   gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
10  gray_small = cv2.resize(gray, dsize=(0, 0), fx=0.5, fy=0.5)
11
12  cv2.imwrite("lena_gray.jpg", gray)
13  cv2.imwrite("lena_gray_small.jpg", gray_small)
14
```

Section 04 영상처리

❑ 영상 크기 변경

- **영상에는 다양한 형태가 있다.**
- ex7_2.py**에서는 채널이** 3**개인 컬러 영상을 채널이** 1**개 뿐인 명암 영상**(grayscale image)**으로 변환하고 영상 크기를 반으로 축소하는 실험을 한다.**

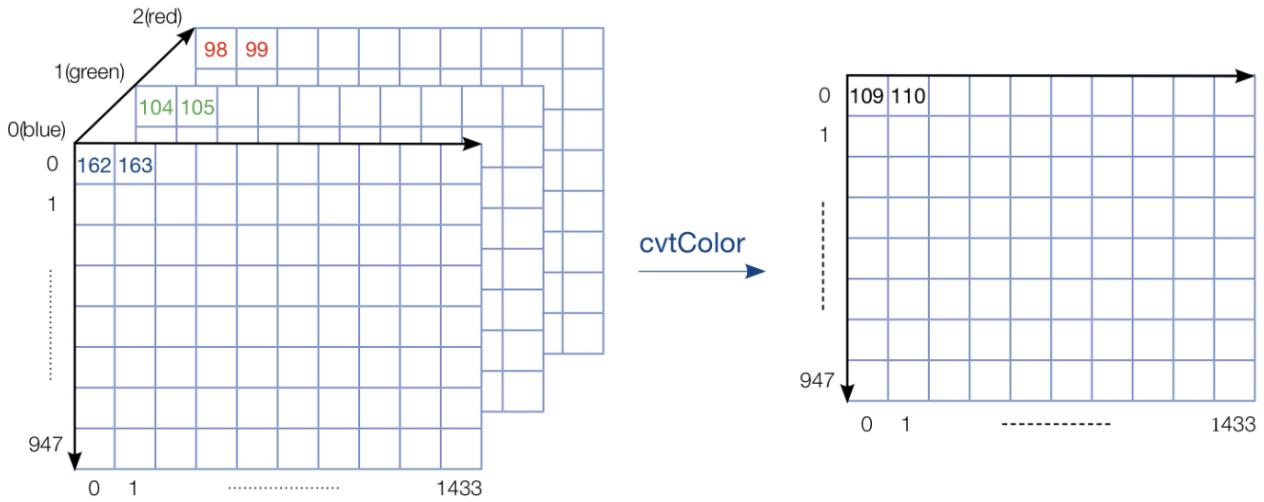
```
15  cv2.imshow("Color image", img)
16  cv2.imshow("Gray image", gray)
17  cv2.imshow("Gray image small", gray_small)
18
19  cv2.waitKey()
20  cv2.destroyAllWindows()
```


Section 04 영상처리

❑ 영상 크기 변경

- OpenCV의 cvtColor 함수는 채널이 세 장인 컬러 영상을 어떻게 채널이 한 장인 명암 영상으로 변환할까?
- OpenCV 공식 사이트는 컬러 이론에 기반해 BGR 영상을 명암 영상으로 변환하는 공식을 식 (2.1)로 제시한다.
- round는 반올림을 뜻한다. $I = round(0.299 \times R + 0.587 \times G + 0.114 \times B)$
- 그림에서 보는 바와 같이 BGR=(162,104,98)은 식 (7.1)에 따라 I=109로 변환된다. 이런 연산을 모든 화소에 적용하면 명암 영상이 만들어진다.



Section 04 영상처리

❑ 회전 및 반전

- **이미지를 회전할 때는** cv2.warpAffine() **함수를 사용한다**.
- **이 함수는 아핀 변환을 한다**.
- **이때 필요한 행렬은**cv2.getRotationMatrix2D**로 얻을 수 있다**.
- **또한 반전에는 '**cv2.flip(**이미지, 대상 축**)**' 함수를 사용한다**.

```
1  import cv2
2  import numpy as np
3
4  img = cv2.imread("./img/sample.jpg")
5
6  re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8  mat = cv2.getRotationMatrix2D(tuple(np.array(re_img.shape[:2]) / 2), 180, 2.0)
9
```

Section 04 영상처리

❑ 회전 및 반전

- **이미지를 회전할 때는** cv2.warpAffine() **함수를 사용한다**.
- **이 함수는 아핀 변환을 한다**.
- **이때 필요한 행렬은**cv2.getRotationMatrix2D**로 얻을 수 있다**.
- **또한 반전에는 '**cv2.flip(**이미지, 대상 축**)**' 함수를 사용한다**.

```
10  my_img = cv2.warpAffine(re_img, mat, re_img.shape[:2])
11
12  cv2.imshow("sample", my_img)
13
14  cv2.waitKey()
15  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 회전 및 반전

- ex7_4.py는 cv2.flip() **함수를 사용하여 이미지를** x**축을 중심으로 반전시킨 프로그램이다**.
- cv2.flip() **함수는 인수가** 0**일 때는** x**축을 중심으로**, **양수일 때는** y**축을 중심으로**, **음수일 때는 두 축을 중심으로 반전한다**.

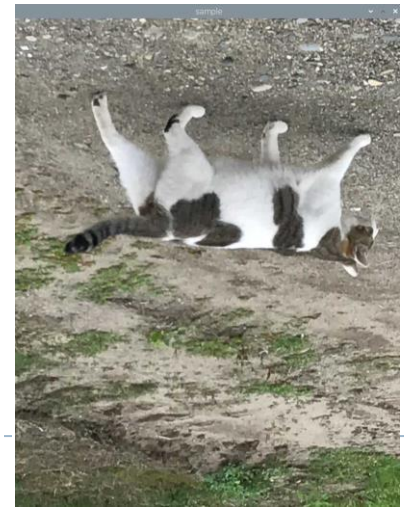
```
1  import cv2
2  import numpy as np
3
4  img = cv2.imread("./img/sample.jpg")
5
6  re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8  my_img = cv2.flip(re_img,0)
9
```

Section 04 영상처리

❑ 회전 및 반전

- ex7_4.py는 cv2.flip() **함수를 사용하여 이미지를** x**축을 중심으로 반전시킨 프로그램이다**.
- cv2.flip() **함수는 인수가** 0**일 때는** x**축을 중심으로**, **양수일 때는** y**축을 중심으로**, **음수일 때는 두 축을 중심으로 반전한다**.

```
10  cv2.imshow("sample", my_img)
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```



Section 04 영상처리

❑ 색조 변환 및 색상 반전

- 앞서 이미지는 RGB로 구성된다고 설명했다.
- 이러한 RGB를 다른 색 공간으로 변환할 수 있다.
- ex7_6.py에서는Lab 색 공간으로 변환한다.
- Lab 색 공간은 인간의 시각에 근접하게 설계되어 있다는 장점이 있다.

```
1  import cv2
2  import numpy as np
3
4  img = cv2.imread(“./img/sample.jpg”)
5
6  re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8  my_img = cv2.cvtColor(re_img, cv2.COLOR_RGB2LAB)
```

Section 04 영상처리

❑ 색조 변환 및 색상 반전

- 앞서 이미지는 RGB로 구성된다고 설명했다.
- 이러한 RGB를 다른 색 공간으로 변환할 수 있다.
- ex7_6.py에서는Lab 색 공간으로 변환한다.
- Lab 색 공간은 인간의 시각에 근접하게 설계되어 있다는 장점이 있다.

```
10  cv2.imshow("sample", my_img)
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 색조 변환 및 색상 반전

- cv2.cvtColor() 함수의 두 번째 인수를COLOR_RGB2GRAY로 하면 흑백 이미지로 변환할 수 있다.
- 또한 이미지의 색상을 반전시키는 것을 네거티브 반전이라고 한다.
- OpenCV에서 네거티브 반전을 하려면 다음처럼 기술한다.

```
img = cv2.bitwise_not(img)
```

- cv2.bitwise() 함수는8bit로 표현된 각 화소의 비트를 조작할 수 있다.
- not은 각 비트를 반전시킨다.

Section 04 영상처리

❑ 임계값 처리(이진화)

- **이미지의 용량을 줄이기 위해 일정 이상으로 밝은 것 혹은 일정 이상으로 어두운 것을 모두 같은 값으로 만들어 버리는 것을 임계값 처리라고 한다.**
- Cv2.threshold() **함수를 사용해서 구현하며, 인수를 설정하여 다양한 임계값 처리를 할 수 있다.**

```
1  import cv2
2  import numpy as np
3
4  img = cv2.imread("./img/sample.jpg")
5
6  re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8  ret, my_img = cv2.threshold(re_img, 75, 255, cv2.THRESH_TOZERO)
9
```

Section 04 영상처리

❑ 임계값 처리(이진화)

- **이미지의 용량을 줄이기 위해 일정 이상으로 밝은 것 혹은 일정 이상으로 어두운 것을 모두 같은 값으로 만들어 버리는 것을 임계값 처리라고 한다.**
- Cv2.threshold() **함수를 사용해서 구현하며, 인수를 설정하여 다양한 임계값 처리를 할 수 있다.**

```
10  cv2.imshow("sample", my_img)
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 임계값 처리(이진화)

- **다음 예제는 임곗값을 100으로 하고, 그 이하는 0으로, 그 이상은 255가 되도록 설정한 프로그램이다.**

```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread(“./img/sample.jpg”)
5
6   re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8   ret, my_img = cv2.threshold(re_img, 100, 255, cv2.THRESH_BINARY)
9
10  cv2.imshow(“sample”, my_img)
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 마스킹

- ex7_7.py는 **이미지의 일부를 추출하는 프로그램이다.**
- **채널 수가 1인 흑백 이미지를 준비한다.**
- **이를 마스크용 이미지라고 하자.**
- **이미지 중에서 마스크용 이미지의 흰 부분만 추출할 수 있다.**

```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread("./img/sample.jpg")
5
6   re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8   ret, my_img = cv2.threshold(re_img, 75, 255, cv2.THRESH_TOZERO)
9
10  cv2.imshow("sample", my_img)
```

Section 04 영상처리

❑ 마스킹

- ex7_7.py는 **이미지의 일부를 추출하는 프로그램이다.**
- **채널 수가 1인 흑백 이미지를 준비한다.**
- **이를 마스크용 이미지라고 하자.**
- **이미지 중에서 마스크용 이미지의 흰 부분만 추출할 수 있다.**

```
10  cv2.imshow("sample", my_img)
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 마스킹

- **다음 예제는** sample.jpg**의 이미지에서** mask.png**의 검은 부분만 꺼내는 프로그램이다**.
- cv2.threshold () **함수로 이미지를 반전시키고**, cv2.bitwise_and() **함수로 마스크 처리를 한다**.

```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread("./img/sample.jpg")
5
6   re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8   ret, my_img = cv2.threshold(re_img, 75, 255, cv2.THRESH_TOZERO)
9
10  cv2.imshow("sample", my_img)
```

Section 04 영상처리

❑ 마스킹

- **다음 예제는** sample.jpg**의 이미지에서** mask.png**의 검은 부분만 꺼내는 프로그램이다**.
- cv2.threshold () **함수로 이미지를 반전시키고**, cv2.bitwise_and() **함수로 마스크 처리를 한다**.

```
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 흐림

- **이미지를 흐리게 하려면 픽셀 주위의** n x n**개(마스크 크기) 픽셀과의 평균을 취한다**.
- **흐림(블러**, blur) **효과는**GaussianBlur() **함수를 사용한다**.

```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread("./img/sample.jpg")
5
6   re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4))
7
8    my_img = cv2.GaussianBlur(re_img, (5, 5), 0)
9
10  cv2.imshow("sample", my_img)
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 노이즈 제거

- **노이즈를 제거할 때는** cv2.fastNlMeansDenoisingColored () **함수를 사용한다.**

```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread(“./img/sample.jpg”)
5
6   re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8    my_img = cv2.fastNlMeansDenoisingColored(re_img)
9
10  cv2.imshow(“sample”, my_img)
11
12  cv2.waitKey()
13  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 팽창 및 침식

- 팽창(dilation)과 침식(erosion)은 주로 이진 이미지로 처리된다.
- 어떤 한 픽셀을 중심으로 두고, 필터 내의 최댓값을 중심값으로 하는 것을 팽창, 반대로 최솟값을 중심값으로 하는 것을 침식이라고 한다.
- 필터는 중심 픽셀의 상하좌우4개 픽셀을 사용하는 방법과 자신을 둘러싼 8개 픽셀을 사용하는 방법의 두 가지가 주를 이룬다.
- 팽창에는cv2.dilate () 함수를, 침식에는cv2.erode() 함수를 사용한다.
- np.uint8은 데이터형을 나타낸다.
- uint8은 8비트로 표현된 부호 없는 정수(unsigned int)를 나타낸다.

Section 04 영상처리

❑ 팽창 및 침식

```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread("./img/sample.jpg")
5
6   re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8   filt = np.array([[0, 1, 0],
                     [1, 0, 1],
                     [0, 1, 0], np.uint8)
9
10   my_img = cv2.dilate(re_img, filt)
11
```

Section 04 영상처리

❑ 팽창 및 침식

```
12  cv2.imshow("sample", my_img)
13
14  cv2.waitKey()
15  cv2.destroyAllWindows()
```

Section 04 영상처리

❑ 팽창 및 침식

- **다음 예제는** ex7_10.py**와 동일한 필터로 침식을 적용한 프로그램이다.**

```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread("./img/sample.jpg")
5
6   re_img = cv2.resize(img, (img.shape[1] // 4, img.shape[0] // 4)
7
8   filt = np.array([[0, 1, 0],
                     [1, 0, 1],
                     [0, 1, 0], np.uint8)
9
10   my_img = cv2.erode(re_img, filt)
11
```

Section 04 영상처리

❑ 팽창 및 침식

- **다음 예제는** ex7_10.py**와 동일한 필터로 침식을 적용한 프로그램이다.**

```
12  cv2.imshow("sample", my_img)
13
14  cv2.waitKey()
15  cv2.destroyAllWindows()
```

Section 05 카메라에서 비디오 읽기

❑ 카메라에서 비디오 읽기

- ex7_12.py**는 카메라를 통해 입력되는 동영상을 윈도우에 디스플레이하는 프로그램이다**.

```
1   import cv2
2   import sys
3
4   cap = cv2.VideoCapture(0)
5   cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
6   cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
7
8   if img is not None:
9       sys.exti("카메라 연결 실패")
10
11  while True:
12      ret, frame = cap.read()
13
14      if not ret:
15          print("프레임 획득에 실패하여 루프를 나갑니다.")
16          break
```

Section 05 카메라에서 비디오 읽기

❑ 카메라에서 비디오 읽기

- ex7_12.py는 **카메라를 통해 입력되는 동영상을 윈도우에 디스플레이하는 프로그램이다**.

```
17
18      cv2.imshow("Video display", frame)
19
20      key = cv2.waitKey(1)
21      if key == ord('q'):
22          break
23
24  cap.release()
25  cv2.destroyAllWindows()
```

Section 05 카메라에서 비디오 읽기

❑ 비디오에서 영상 수집하기

- ex7_13.py**는 비디오에서 영상을 수집할 수 있게 확장한 프로그램으로,** ex7_12.py**의 골격을 그대로 사용한다.**
- **추가된 코드는 노란색으로 표시해 구분한다.**

```
1   import cv2
2   import numpy as np
3   import sys
4
5   cap = cv2.VideoCapture(0)
6   cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
7   cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
8
9   if img is not None:
10      sys.exti("카메라 연결 실패")
11
12  frames = [ ]
13  while True:
14      ret, frame = cap.read()
15
```

Section 05 카메라에서 비디오 읽기

❑ 비디오에서 영상 수집하기

- ex7_13.py**는 비디오에서 영상을 수집할 수 있게 확장한 프로그램으로,** ex7_12.py**의 골격을 그대로 사용한다.**
- **추가된 코드는 노란색으로 표시해 구분한다.**

```
16      if not ret:
17          print("프레임 획득에 실패하여 루프를 나갑니다.")
18          break
19
20      cv2.imshow("Video display", frame)
21
22      key = cv2.waitKey(1)
23      if key == ord('c'):
24          frames.append(frame)
25      if key == ord('q'):
26          break
27
28  cap.release()
29  cv2.destroyAllWindows()
30
```

Section 05 카메라에서 비디오 읽기

❑ 비디오에서 영상 수집하기

- ex7_13.py**는 비디오에서 영상을 수집할 수 있게 확장한 프로그램으로,** ex7_12.py**의 골격을 그대로 사용한다.**
- **추가된 코드는 노란색으로 표시해 구분한다.**

```
31  if len(frames) > 0:
32      imgs = frames[0]
33      for i in range(1, min(3, len(frames))):
34          imgs = np.hstack((imgs, frames[i]))
35
36      cv2.imshow("collected images", imgs)
37
38      cv2.waitKey()
39      cv2.destroyAllWindows()
```

Section 05 카메라에서 비디오 읽기

❑ 비디오에서 영상 수집하기

- **다음 코드를 추가하여** frames **리스트와** imgs **배열을 조사해보자**.

```
1   import cv2
2   import numpy as np
3   import sys
4
5   cap = cv2.VideoCapture(0)
6   cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
7   cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
8
9   if img is not None:
10      sys.exti("카메라 연결 실패")
11
12  frames = [ ]
13  while True:
14      ret, frame = cap.read()
15
16      if not ret:
17          print("프레임 획득에 실패하여 루프를 나갑니다.")
18          break
```

Section 05 카메라에서 비디오 읽기

❑ 비디오에서 영상 수집하기

- **다음 코드를 추가하여** frames **리스트와** imgs **배열을 조사해보자.**

```
19
20      cv2.imshow("Video display", frame)
21
22      key = cv2.waitKey(1)
23      if key == ord('c'):
24          frames.append(frame)
25      if key == ord('q'):
26          break
27
28  cap.release()
29  cv2.destroyAllWindows()
30
31  if len(frames) > 0:
32      imgs = frames[0]
33      for i in range(1, min(3, len(frames))):
34          imgs = np.hstack((imgs, frames[i]))
35
```

Section 05 카메라에서 비디오 읽기

❑ 비디오에서 영상 수집하기

- **다음 코드를 추가하여** frames **리스트와** imgs **배열을 조사해보자**.

```
36    print(len(frames))
37    print(frames[0].shape)
38    print(type(imgs))
39    print(imgs.shape)
40
41    cv2.imshow("collected images", imgs)
42
43    cv2.waitKey()
44    cv2.destroyAllWindows()
```

Shell

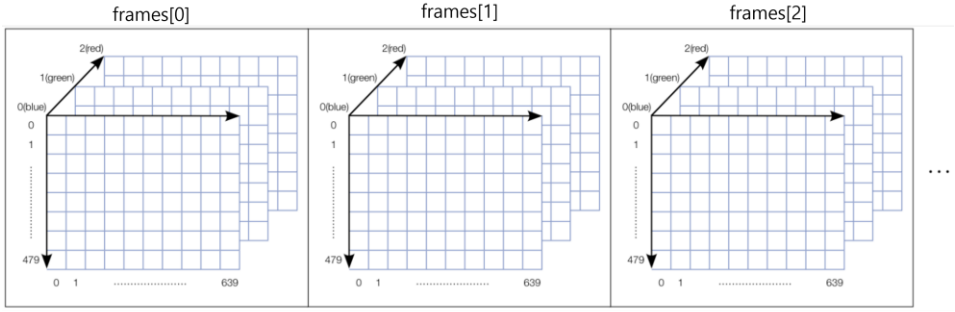
```
3
(480, 640, 3)
<class 'numpy.ndarray'>
(480, 1920, 3)

>>>
```

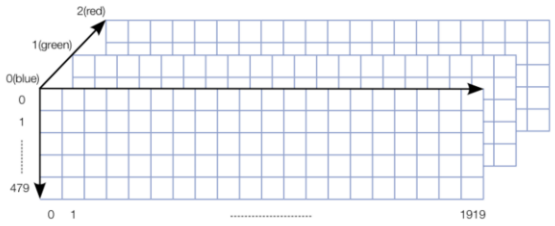
Section 05 카메라에서 비디오 읽기

❑ 비디오에서 영상 수집하기

- **그림은 조사한 내용을 그림으로 보여준다.**
- frames **리스트의 요소는 크기가** 480 x 640 x 3**인 컬러 영상이고** imgs**는** 480 x 1920 x 3 **크기의 배열이라는 사실을 이해할 수 있다.**
- **컴퓨터 비전 프로그래밍을 잘 하는 요령 중의 하나는 프로그램에 등장하는 수많은 객체의 데이터 형과 모양을 정확히 이해하는 것이다.**
- **이때** type**과** shape**은 아주 유용하다.**



(a) frames 리스트



(b) imgs 배열

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 영상에 도형을 그리고 글씨 쓰기

- OpenCV **직선을 그리는** line, **직사각형을 그리는** rectangle, **다각형을 그리는** polylines, **원을 그리는** circle, **타원을 그리는** ellipse, **문자열을 쓰는** putText **함수를 제공한다**.
- ex7_15.py**의** 9**행은** rectangle **함수로 얼굴에 직사각형을 그린다**.

```
1   import cv2
2   import sys
3
4   img = cv2.imread('./img/girl_laughing.jpg')
5
6   if img is None:
7       sys.exit("파일을 찾을 수 없습니다.")
8
9   cv2.rectangle(img, (830, 30), (1000, 200), (0, 0, 255), 2)
10  cv2.putText(img, 'laugh', (830, 24), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2)
11
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 영상에 도형을 그리고 글씨 쓰기

- ex7_15.py**의** 9**행은** rectangle **함수로 얼굴에 직사각형을 그린다**.

```
12  cv2.imshow('Draw', img)
13
14  cv2.waitKey()
15  cv2.destroyAllWindows()
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 마우스를 통한 상호작용

- ex7_15.py는 **고정된 위치에 직사각형을 그린다**.
- **이제 마우스를 이용해 위치를 지정할 수 있도록 확장해보자**.
- ex7_16.py는 **마우스를 클릭한 곳에 직사각형을 그린다**.
- **왼쪽 버튼을 클릭하면 크기가** 200 x 200**인 빨간색 직사각형을 그리고, 오른쪽 버튼을 클릭하면 크기가** 100 x 100**인 파란색 직사각형을 그린다**.

```
1  import cv2
2  import sys
3
4  img = cv2.imread('./img/girl_laughing.jpg')
5
6  if img is None:
7      sys.exit("파일을 찾을 수 없습니다.")
8
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 마우스를 통한 상호작용

- ex7_16.py**는 마우스를 클릭한 곳에 직사각형을 그린다.**

```
9   def draw(event, x, y, flags, param):
10      if event == cv2.EVENT_LBUTTONDOWN:
11          cv2.rectangle(img, (x, y), (x+200, y+200), (0, 0, 255), 2)
12      elif event == cv2.EVENT_RBUTTONDOWN:
13          cv2.rectangle(img, (x, y), (x+100, y+100), (255, 0, 0), 2)
14
15      cv2.imshow( ‘Drawing’, img)
16
17  cv2.setMouseCallback( 'Drawing’, draw)
18  cv2.imshow( ‘Drawing’, draw)
19
20  cv2.setMouseCallback( ‘Drawing’, draw)
21
22  while(True):
23      if  cv2.waitKey(1) == ord( ‘q’ )   :
24          cv2.destroyAllWindows()
25          break
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 마우스를 통한 상호작용

- **마우스를 다루려면 콜백 함수**(callback function)**라는 새로운 프로그래밍 방식이 필요하다**.
- **보통 프로그램은 정해진 순서에 따라 명령어를 실행하는데, 마우스를 다루는 프로그램에서는 클릭이나 커서 이동 같은 이벤트가 언제 발생할지 알 수 없기 때문에 콜백 함수가 필요하다**.
- 17**행은 'Drawing'이라는 이름의 윈도우를 생성하고**, 18**행은 윈도우에** img **영상을 디스플레이한다**.
- 20**행은 'Drawing'이라는 이름의 윈도우에서 마우스 이벤트가 발생하면** draw**라는 콜백 함수를 호출하라고 등록한다**.
- **마우스 이벤트는 버튼 클릭하기, 버튼에서 손 놓기, 커서 이동, 휠 돌리기를 하면 발생한다**.

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 마우스 드래그로 도형 크기 조절하기

- ex7_16.py**프로그램은 직사각형 위치는 선택할 수 있지만 크기는 정해져 있다**.
- **직사각형 크기를 마음대로 조절하려면 마우스 클릭과 드래그를 함께 사용하면 된다**.
- ex7_17.py**프로그램은 사용자가 직사각형 크기를 조절할 수 있게 확장한다**.

```
1  import cv2
2  import sys
3
4  img = cv2.imread('./img/girl_laughing.jpg')
5
6  if img is None:
7      sys.exit("파일을 찾을 수 없습니다.")
8
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 마우스 드래그로 도형 크기 조절하기

- ex7_17.py**프로그램은 사용자가 직사각형 크기를 조절할 수 있게 확장한다.**

```
9   def draw(event, x, y, flags, param):
10      global ix, iy
11
12      if event == cv2.EVENT_LBUTTONDOWN :
13          ix, iy = x, y
14      elif event == cv2.EVENT_LBUTTONUP :
15          cv2.rectangle(img, (ix, iy), (x, y), (0, 0, 255), 2)
16
17      cv2.imshow( ‘Drawing’ , img)
18
19  cv2.namedWindow( ‘Drawing’)
20  cv2.imshow( ‘Drawing’, img)
21
22  cv2.setMouseCallback( 'Drawing’, draw)
23
24  while(True):
25      if  cv2.waitKey(1) == ord( ‘q’ ):
26          cv2.destroyAllWindows()
27          break
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 마우스 드래그로 도형 크기 조절하기

- ex7_17.py**프로그램은 사용자가 직사각형 크기를 조절할 수 있게 확장한다.**

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- 이미 앞서 여러 번 사용한 적이 있는 cv2.waitKey(delay) 함수를 쓰면 키보드의 입력을 알아낼 수 있다.
- 이 함수는 delay 인자에 밀리초(ms, 0.001초) 단위로 숫자를 전달하면 해당 시간 동안 프로그램을 멈추고 대기하다가 키보드의 눌린 키에 대응하는 코드 값을 정수로 반환한다.
- 지정한 시간까지 키보드 입력이 없으면 -1을 반환한다.
- delay 인자에 0을 전달하면 대기 시간을 무한대로 하겠다는 의미이므로 키를 누를 때까지 프로그램은 멈추고 이 때는 -1을 반환할 일은 없다.
- 키보드에서 어떤 키를 눌렀는 지를 알아내려면 cv2.waitKey() 함수의 반환 값을 출력해 보면 된다.

```
key = cv2.waitKey(0)
print(key)
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- 출력되는 키 값을 확인해 보면 ASCII 코드와 같다는 것을 알 수 있다.
- 환경에 따라 한글 모드에서 키를 입력하면 오류가 발생할 수 있으니 키를 입력할 때 한글은 사용하지 않는 것이 좋다.
- 입력된 키를 특정 문자와 비교할 때는 파이썬 기본 함수인 ord() 함수를 사용하면 편리하다.
- 예를 들어 키보드의 'a' 키를 눌렀는지 확인하기 위한 코드는 다음과 같다.

```
if cv2.waitKey(0) == ord(' a' ) :
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- **그런데 몇몇 64 비트 환경에서 cv2.waitKey() 함수는 8비트(ASCII 코드 크기)보다 큰 32비트 정수를 반환해서 그 값을 ord() 함수를 통해 비교하면 서로 다른 값으로 판단할 때가 있다.**
- **그래서 하위 8비트를 제외한 비트를 지워야 하는 경우가 있다.**
- **0xFF는 하위 8비트가 모두 1로 채워진 숫자이므로 이것과 & 연산을 수행하면 하위 8비트보다 높은 비트는 모두 0으로 채울 수 있다.**

```
key = cv2.waitKey (0) & 0xFF
if key == ord ( 'a' ) :
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- **이제 키보드의 입력값을 확인하는 코드를 작성해보자.**
- ex7_18.py

```
1   import cv2
2
3   img = cv2.imread('./img/model3.jpg')
4   while True:
5       cv2.imshow('img', img)
6
7       keyValue = cv2.waitKey(0)
8       print(keyValue)
9
10      if keyValue == ord('q'):
11          break
12
13  cv2.destroyAllWindows()
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- 다음 예제 코드에서는 자동차의 카메라를 켜고 화살표로 자동차를 제어할 것이다.
- 스페이스 바를 누르면 자동차가 정지하도록 코드를 작성하였다.

```
1   import cv2
2   import time
3   from ctypes import *
4   import os
5
6   WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_GLOBAL)
7   swcar = cdll.LoadLibrary('/home/pi/swcar/libswcar.so')
8
9   swcar.SIO_Init(0)
10
11  swcar.SIO_MaxMotorSpeed(50)
12
13  cam = cv2.VideoCapture(0)
14  cam.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
15  cam.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- 다음 예제 코드에서는 자동차의 카메라를 켜고 화살표로 자동차를 제어할 것이다.
- 스페이스 바를 누르면 자동차가 정지하도록 코드를 작성하였다.

```
16
17  while (cam.isOpened()):
18      _, img = cam.read()
19      cv2.imshow('Camera', img)
20
21      key = cv2.waitKey(0)
22      print(key)
23
24      if key == ord('q'):
25          swcar.SIO_WriteServo(0)
26          time.sleep(0.02)
27          swcar.SIO_WriteMotor(0)
28          time.sleep(0.02)
29          break
30
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- 다음 예제 코드에서는 자동차의 카메라를 켜고 화살표로 자동차를 제어할 것이다.
- 스페이스 바를 누르면 자동차가 정지하도록 코드를 작성하였다.

```
31      elif key == 82:
32          print("go")
33          swcar.SIO_WriteServo(100, 50)
34          swcar.SIO_ForwardMotor(20)
35          time.sleep(0.02)
36      elif key == 84:
37          print("back")
38          swcar.SIO_WriteServo(100, 50)
39          swcar.SIO_ReverseMotor(20)
40          time.sleep(0.02)
41      elif key == 81:
42          print("left")
43          swcar.SIO_WriteServo(100, 90)
44          swcar.SIO_ForwardMotor(20)
45          time.sleep(0.02)
```

Section 06 그래픽 기능과 사용자 인터페이스 만들기

❑ 키보드 이벤트

- 다음 예제 코드에서는 자동차의 카메라를 켜고 화살표로 자동차를 제어할 것이다.
- 스페이스 바를 누르면 자동차가 정지하도록 코드를 작성하였다.

```
46      elif key == 83:
47          print("right")
48          swcar.SIO_WriteServo(100, 10)
49          swcar.SIO_ForwardMotor(20)
50          time.sleep(0.02)
51      elif key == 32:
52          print("stop")
53          swcar.SIO_ForwardMotor(0)
54          time.sleep(0.02)
55
56  cam.release()
57  cv2.destroyAllWindows()
```

Section 07 페인팅 기능 만들기

- 때로 마우스가 이동한 궤적을 따라 페인팅하는 기능이 필요하다.
- ex7_21.py는 왼쪽 버튼을 클릭하면 파란색으로 페인팅하고 오른쪽 버튼을 클릭하면 빨간색으로 페인팅하는 프로그램이다.
- 이 프로그램은 ex7_18.py의 골격을 그대로 사용한다.
- 단지 마우스 이벤트가 발생했을 때 수행하는 콜백 함수의 논리만 다르다.

```
1   import cv2
2   import sys
3
4   img = cv2.imread( './img/soccer.jpg' )
5
6   if img is None:
7       sys.exit( "파일을 찾을 수 없습니다." )
8
9   BrushSize = 5
10  LColor, RColor = (255, 0, 0), (0, 0, 255)
11
```

Section 07 페인팅 기능 만들기

❑ ex7_21.py는 왼쪽 버튼을 클릭하면 파란색으로 페인팅하고 오른쪽 버튼을 클릭하면 빨간색으로 페인팅하는 프로그램이다.

```
12  def painting(event, x, y, flags, param) :
13      if event == cv2.EVENT_LBUTTONDOWN :
14          cv2.circle(img, (x, y), BrushSize, LColor, -1)
15      elif event == cv2.EVENT_RBUTTONDOWN :
16          cv2.circle(img, (x, y), BrushSize, RColor, -1)
17      elif event == cv2.EVENT_MOUSEMOVE and flags == cv2.EVENT_FLAG_LBUTTON :
18          cv2.circle(img, (x, y), BrushSize, LColor, -1)
19      elif event == cv2.EVENT_MOUSEMOVE and flags == cv2.EVENT_FLAG_RBUTTON :
20          cv2.circle(img, (x, y), BrushSize, RColor, -1)
21
22      cv2.imshow( 'Painting', img)
23
24  cv2.namedWindow( 'Painting' )
25  cv2.imshow( 'Painting', img)
26
27  cv2.setMouseCallback( 'Painting',  painting)
28
```

Section 07 페인팅 기능 만들기

❑ ex7_21.py는 왼쪽 버튼을 클릭하면 파란색으로 페인팅하고 오른쪽 버튼을 클릭하면 빨간색으로 페인팅하는 프로그램이다.

```
29  while(True):
30      if cv2.waitKey(1) == ord('q'):
31          cv2.destroyAllWindows()
32          break
```

Q&A