메디치소프트 기술연구소

2018.08.07

CH04. OpenCV: 도형 그리기

4. 도형 그리기

- 4.1 테스트 환경

순 번	제 목	설치 버전
1	운영체제	Windows 10 64bit
2	프로그래밍언어	Python3.6.5

4. 도형 그리기

- 4.2 [04-01 example.py] – 이미지 띄우기

  - 다음은 04-01 example.py 예제의 code이다.

```
import numpy as np
import cv2

def drawing():
    img = np.zeros((512,512,3), np.uint8)

    #다양한 색상과 선두께를 가진 도형 그리기
    cv2.line(img,(0,0), (511,511), (255,0,0), 5)
    cv2.rectangle(img, (384, 0), (510, 128), (0, 255,0), 3)
    cv2.circle(img, (477,63), 63, (0,0,255), -1)
    cv2.ellipse(img, (256,256), (100, 50), 0, 0, 180, (255,0,0), -1)
```

[그림] 04-01 example.py의 code (1/2)

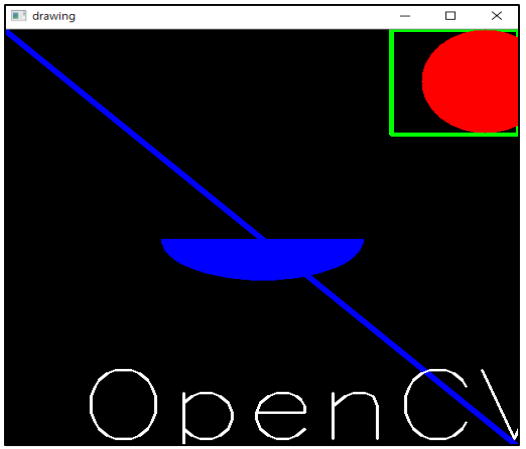
```
    font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.putText(img, ' OpenCV', (10,500), font, 4, (255,255,255), 2)

    cv2.imshow('drawing', img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

drawing()
```

[그림] 04-01 example.py의 code (2/2)

4. 도형 그리기

- 4.2 [04-01 example.py] – 이미지 띄우기

  - 다음 그림은 04-01 example.py를 실행한 결과 생성되는 image 사진이다.
    뒷 장부터 04-01 example.py code의 주요 함수를 line 순서대로 분석해보도록 한다



[그림] 04-01 example.py의 실행 결과

4. 도형 그리기

- 4.2 [04-01 example.py] – 이미지 띄우기

>> LINE 5) img = np.zeros((512,512,3), np.uint8)

- 도형을 그리기 위한 공간을 생성한다.

- 위 코드는 numpy.zeors() 함수를 이용하고 있는데, 이 함수는 numpy 배열을 만들고 모든 값을 0으로 채우는 함수이다.
여기서는 각 멤버가 (0,0,0)인 512 * 512 배열이며, 데이터 타입은 uint8 만드는 예제이다.
이것을 이미지 차원에서 설명하자면, 512 * 512 크기의 검정색 이미지를 생성한 것과 같다. 원소 (0,0,0) 즉 BGR의 값으로 검정색으로 초기화가 되어 있기 때문이다.

4. 도형 그리기

- 4.2 [04-01 example.py] – 이미지 띄우기

>> LINE 8) cv2.line(img, (0,0), (511,511), (255,0, 0), 5)

- 512 * 512 크기의 검정색 판에 좌표 (0,0)에서 (511, 511)까지 파란색의 두께 5인 직선을 그린다.

- 첫 번째 인자 img : 직선을 그릴 그림판
  두번째 인자 (0,0) : 직선의 시작점
  세 번째 인자 (511,511) : 직선의 끝점
  네 번째 인자 (255, 0, 0) : BGR 값으로 선의 색상
  다섯 번째 인자 5 : 선의 굵기

4. 도형 그리기

- 4.2 [04-01 example.py] – 이미지 띄우기

>> LINE 9) cv2.rectangle(img, (384, 0), (510, 128), (0, 255, 0), 3)

- cv2.rectangle() 함수의 인자는 직선 그리기 함수와 동일하다.
  주어진 두 개의 좌표가 좌측 상단, 우측 상단 좌표이며, 이 좌표가 사각형의 대각선 꼭지점이 되는 것이 차이이다.

- 두번째 인자 (384,0) : 좌측 상단의 꼭지점
  세 번째 인자 (510,128) : 우측 하단의 꼭지점

4. 도형 그리기

- 4.2 [04-01 example.py] – 이미지 띄우기

>> LINE 10) cv2.circle(img, (477, 63), 63, (0, 0, 255), -1)

- cv2.rectangle() 함수의 인자는 직선 그리기 함수와 동일하다.
  주어진 두 개의 좌표가 좌측 상단, 우측 상단 좌표이며, 이 좌표가 사각형의 대각선 꼭지점이 되는 것이 차이이다.

- 두번째 인자 (384,0) : 좌측 상단의 꼭지점
  세 번째 인자 (510,128) : 우측 하단의 꼭지점

2. 이미지 Reading과 Writing

- 2.2 [02-01 example.py] – 이미지 띄우기

>> LINE 6) img = cv2.imread(imgfile, cv2.IMREAD_COLOR)

- 두 번째 인자인 이미지 읽기 플래그는 총 3가지가 있다.

순 번	제 목	내 용
1	cv2.IMREAD_COLOR	컬러 이미지 형태로 불러온다. 이미지의 투명한 부분은 모두 무시된다. 정수 값으로는 1이다.
2	cv2.IMREAD_GRAYSCALE	흑백 이미지 형태로 불러온다. 정수 값으로는 0이다.
3	cv2.IMREAD_UNCHANGED	알파채널을 포함하여 이미지를 그대로 불러온다. 정수 값으로는 -1이다.

2. 이미지 Reading과 Writing

- 2.2 [02-01 example.py] – 이미지 띄우기

>> LINE 8) cv2.namedWindow('model', cv2.WINDOW_NORMAL)

- 이미지 크기를 변경시킬 수 있는 창을 만들기 위해서 cv2.namedWindow() 함수를 이용해 이미지가 표시될 창의 성격을 지정한다.

- 해당 code를 추가하면 실행 결과 나오는 창의 크기를 사용자가 마우스를 이용하여 조정할 수 있다.

2. 이미지 Reading과 Writing

- 2.2 [02-01 example.py] – 이미지 띄우기

>> LINE 8) cv2.namedWindow('catl', cv2.WINDOW_NORMAL)

- 첫번째 인자 'cat' : 생성될 이미지 윈도우의 타이틀
  두 번째 인자 cv2.WINDOW_NORMAL : 윈도우 사이즈 플래그

- 두 번째 인자의 플래그는 총 2가지가 있다.

순 번	제 목	내 용
1	cv2.WINDOW_AUTOSIZE	원본 이미지 크기로 고정하여 윈도우 생성
2	cv2.WINDOW_NORMAL	원본 이미지 크기로 윈도우를 생성하여 이미지를 나타내지만 사용자가 마우스를 이용하여 크기를 조정할 수 있다.

2. 이미지 Reading과 Writing

- 2.2 [02-01 example.py] – 이미지 띄우기

>> LINE 9) cv2.imshow('cat', img)

- cv2.imread()에 의해 반환된 이미지 객체 img를 화면에 나타내기 위한 함수이다.

- 첫 번째 인자 'cat' : 윈도우 타이틀
  두 번째 인자 img : 화면에 표시할 이미지 객체

2. 이미지 Reading과 Writing

- 2.2 [02-01 example.py] – 이미지 띄우기

>> LINE 10) cv2.waitKey(0)

- 화면에 이미지를 표시한 후 사용자가 키보드를 누를 때 까지 대기하라는 함수이다.

- cv2.waitKey() 함수는 지정된 시간동안 키보드 입력을 기다린다.
  ex) cv2.waitKey(5)는 5ms 동안 대기하라는 의미이다.

- 이 예제에서 사용한 cv2.waitKey(0)은 키보드 입력이 있을 때 까지 기다리라는 의미이다.
  이 함수의 리턴 값은 사용자가 누른 키보드의 값이다.

2. 이미지 Reading과 Writing

- 2.2 [02-01 example.py] – 이미지 띄우기

>> LINE 11) cv2.destroyAllWindows()

- 생성한 모든 윈도우를 제거한다.

>> LINE 13) showImage()

- 생성한 showImage() 함수를 실행한다.

- 실행 결과 앞서 보았던 cat 이미지가 나오는 것을 확인할 수 있다.

2. 이미지 Reading과 Writing

- 2.3 [02-02 example.py] – 이미지 파일 복사

  - 다음은 02-02 example.py 예제의 code이다.

```python
import numpy as np
import cv2

def showImage():
    imgfile = 'images/cat.jpg'
    img = cv2.imread(imgfile, cv2.IMREAD_COLOR)
    cv2.imshow('cat',img)

    k = cv2.waitKey(0) & 0xFF

    if k == 27:
        cv2.destroyAllWindows()
    elif k == ord('c'):
        cv2.imwrite('images/cat_copy.jpg', img)
        cv2.destroyAllWindows()

showImage()
```

[그림] 02-02 example.py의 code

2. 이미지 Reading과 Writing

- 2.3 [02-02 example.py] – 이미지 파일 복사

  - cat_copy.jpg는 02-02 example.py를 실행한 결과 생성되는 image 사진이다.
    뒷 장부터 02-02 example.py code의 주요 함수를 line 순서대로 분석해보도록 한다.

[그림] 02-02 example.py의 실행 결과

2. 이미지 Reading과 Writing

- 2.3 [02-02 example.py] – 이미지 파일 복사

>>LINE 9) k = cv2.waitKey(0) & 0xFF

- 사용자가 키보드로 입력할 때까지 기다리며 입력이 되면 입력한 값을 k로한다.

- 64bit 컴퓨터에서는 0xFF와 연산해주는데(32bit 컴퓨터에서는 생략해도 된다.)
  이는 byte형에서 int형으로 형변환시 발생할 수 있는 에러를 방지해 준다.

2. 이미지 Reading과 Writing

- 2.3 [02-02 example.py] – 이미지 파일 복사

```
>>LINE 11~12)
    if k == 27:
        cv2.destroyAllWindows()
```

- 아스키 코드값이 27인 ESC키를 누르면 프로그램을 종료한다.

2. 이미지 Reading과 Writing

- 2.3 [02-02 example.py] – 이미지 파일 복사

```
>>LINE 13~15)
    elif k == ord('c'):
        cv2.imwrite('images/cat_copy.jpg', img)
        cv2.destroyAllWindows()
```

- c키를 누르면 images 폴더에 cat_copy.jpg라는 이름의 복사본을 만든다.

- 여기서 ord()함수는 문자를 아스키 값으로 변환하는 함수이다.

- 복사본이 폴더에 생성된 후 프로그램이 종료된다.

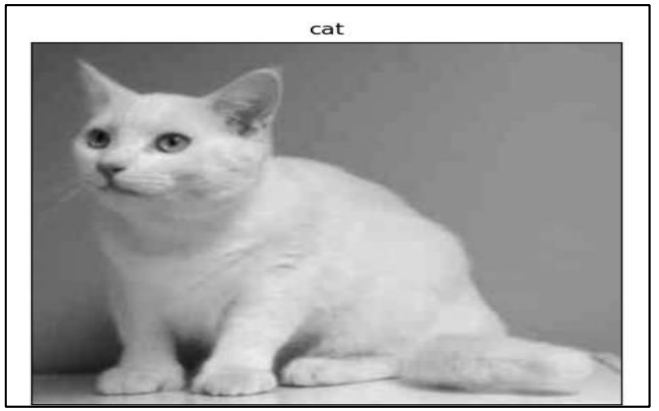
2. 이미지 Reading과 Writing

- 2.4 [02-03 example.py] – Matplotlib를 이용한 이미지 출력

  - 다음은 02-03 example.py 예제의 code와 실행 결과로 나오는 image 사진이다.
    뒷 장부터 02-03 example.py code의 주요 함수를 line 순서대로 분석해보도록 한다.

```
1  import numpy as np
2  import cv2
3  import matplotlib.pyplot as plt
4
5  def showImage():
6      imgfile = 'images/cat.jpg'
7      img = cv2.imread(imgfile, cv2.IMREAD_GRAYSCALE)
8
9      plt.imshow(img, cmap = 'gray', interpolation = 'bicubic')
10     plt.xticks([])
11     plt.yticks([])
12     plt.title('cat')
13     plt.show()
14
15 showImage()
```

[그림] 02-03 example.py의 code



[그림] 02-03 example.py의 실행 결과

2. 이미지 Reading과 Writing

- 2.4 [02-03 example.py] – Matplotlib를 이용한 이미지 출력

>> Matplotlib 정의

- Matplotlib은 파이썬을 위한 플로팅 라이브러리로서 다양한 플로팅 메소드를 제공한다.

- Matplotlib을 이용해 이미지를 화면에 나타낼 수 있으며 이미지 zoom, 저장하기 등과 같은 다양한 기능 등을 활용할 수 있다.

2. 이미지 Reading과 Writing

- 2.4 [02-03 example.py] – Matplotlib를 이용한 이미지 출력

>>LINE 7) img = cv2.imread(imgfile, cv2.IMREAD_GRAYSCALE)

- 이미지를 흑백으로 읽어온다.

- OpenCV는 BGR 모드로 컬러 이미지를 다루는데 비해 Matplotlib은 RGB 모드로 컬러 이미지를 다룬다. 따라서 OpenCV로 읽어 들인 컬러 이미지를 Matplotlib에 그대로 사용하게 될 경우 제대로 된 컬러 색상이 나오지 않는다.

2. 이미지 Reading과 Writing

- 2.4 [02-03 example.py] – Matplotlib를 이용한 이미지 출력

  >>LINE 9) plt.imshow(img, cmap='gray', interpolation='bicubic')

  - Matplotlib의 imshow 함수를 이용하여 화면에 이미지를 디스플레이 하는 방법을 정의한다.

2. 이미지 Reading과 Writing

- 2.4 [02-03 example.py] – Matplotlib를 이용한 이미지 출력

>>LINE 10~11) plt.xticks([]), plt.yticks([])

- Matplotlib은 기본적으로 이미지를 표시할 때 x축 y축으로 눈금 표시를 한다.

- 위 코드는 눈금 표시 없이 이미지를 표시하라는 의미이다.

>>LINE 13) plt.show()

- 화면에 이미지를 출력한다.