컴퓨터비전

윤곽선 개요 및 윤곽선 데이터 구조 생성

contours_generation.py

2024년 1학기

서경대학교 김진헌

Contours : Getting Started

1 https://docs.opencv.org/4.1.0/d4/d73/tutorial_py_contours_begin.html

- 윤곽선이란?
  - Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity.
- 용도는?
  - The contours are a useful tool **for shape analysis and object detection and recognition**.

분석, 탐지, 인식이 목적

- 어떻게 검출하는가?
  - For better accuracy, use **binary images**. So before finding contours, apply threshold or canny edge detection.
- 윤곽선 검출 조건의 가정

검은 배경으로부터 하얀 오브젝트를 찾음

  - In OpenCV, finding contours is like finding white object from black background. So remember, object to be found should be white and background should be black.

전제조건 : 검은 배경, 하얀 물체

OpenCV에서의 Contours 자료

윤곽선 자료형이 따로 정의되어 있지는 않지만 OpenCV 함수들간에 통용되는 윤곽선 자료 형식이 있다. 예: 윤곽선을 그리는 drawContours() 함수

- 1. **1개의 contour(윤곽선)은 여러 개의 꼭지점들의 집합으로 정의**할 수 있다. 꼭지점과 꼭지점 사이는 직선으로 연결한다는 전제이다.
  - 예: 삼각형은 3개의 점으로, 4각형은 4개의 점으로 묘사가능하여 그 이상의 다면체는 그 만큼의 꼭지점 수가 필요하다. 원(circle)의 경우는 충분히 많은 꼭지점으로 구성하면 될 것이다.
- 2. **contour 1 개를 구성하기 위한 점들은 ndarray 구조체로 정의**된다.
  - contour.shape=(점의_개수, 1, 2). <= 1 은 고정(쓰임새 없음). 2는 (x좌표, y좌표)
  - 삼각형 윤곽선의 shape=(3, 1, 2). 사각형 윤곽선의 shape=(4, 1, 2).
- 3. 한 영상에서 검출할 수 있는 물체의 **윤곽선들은 묶어서 list 자료 구조로 표현**된다. Contours = [contour1, contour2, ⋯].
  - 질문: 삼각형 1개와 사각형 1개 있다. 이를 묘사하는 contours 정보는? → contours=[contour_삼각형, contour_사각형]
  - 질문: 1개의 contour가 있다. 이를 contours 정보로 만들어라. → contours=[contour1], Contour1만 있어도 list 자료로 만들어야 한다.

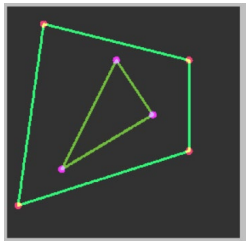
drawContours 는 contours만 그릴 수 있다 - contour 자료형은 안됨 - contours로 만들어야 함

Contour 선언 사례

※ 파이썬 프로그래밍 기술에 관련 되어서는 0_python_practice.py 참조

3

contours_generation.py



프로그램 흐름: 윤곽점 꼭지점을 정의하여 윤곽선 자료를 만든 후, circle()함수로 점을 그리고, drawContours() 함수로 윤곽선을 그린다.

```
contour1 = np.array([[[300, 150]], [[400, 300]], [[150, 450]]], dtype=np.int32)
contour2 = np.array([[[100, 50]], [[500, 150]], [[500, 400]], [[30, 550]]], dtype=np.int32)
contours = [contour1, contour2]   ← 리스트 자료
```

```
    print("\n1.1 여러 개의 contour가 모인 contours의 관찰합니다.")
    print(f"\tcontours: type(contours)={type(contours)}")    #
    print(f"\tcontours: len(contours)={len(contours)}")      #
print("\n1.2 각 contour를 관찰합니다.=> contours 변수로 액세스 합니다.")
for i in range(len(contours)):
    print(f"\n\tcontour num {i}: type={type(contours[i])}, shape={contours[i].shape}")
    num = contours[i].shape[0]      # i번째 contour의 point의 개수
    print(f"\tnumber of points in contours[{i}]={num}")
    for j in range(len(contours[i])):
        print(f"\tpoint {j}:")
        print(f"\t2차원 어레이 반환: contours[{i}][{j}]={contours[i][j]}, "
              f"{contours[i][j].shape}, ndim={contours[i][j].ndim}")
        print(f"\t1차원 어레이 반환: contours[{i}][{j}, 0]={contours[i][j, 0]}, "
              f"{contours[i][j, 0].shape}, ndim={contours[i][j, 0].ndim}")
        print(f"\t\t(x, y) = ", end="")     # (x, y)에 접근하려면 또 한번의 loop가 필요하다
        for k in range(2):      # access to x, y
            print(contours[i][j, 0, k], end=" ")
        print()
```

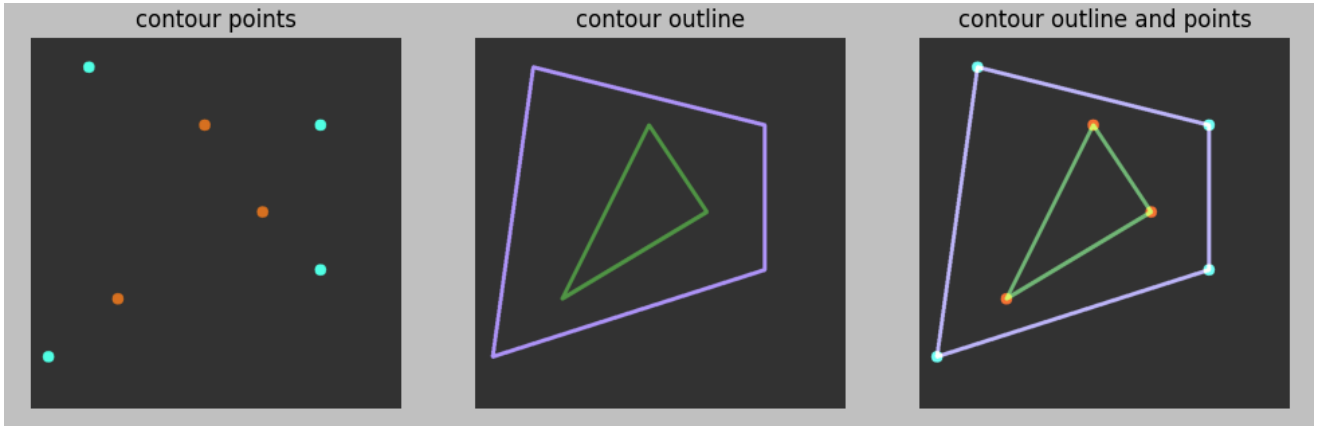
```
1.1 여러 개의 contour가 모인 contours의 관찰합니다.
    contours: type(contours)=<class 'list'>
    contours: len(contours)=2

1.2 각 contour를 관찰합니다.=> contours 변수로 액세스 합니다.

    contour num 0: type=<class 'numpy.ndarray'>, shape=(3, 1, 2)
    number of points in contours[0]=3
    point 0:                                          2차원으로 나옴
    2차원 어레이 반환: contours[0][0]=[[300 150]], (1, 2), ndim=2
    1차원 어레이 반환: contours[0][0, 0]=[300 150], (2,), ndim=1
        (x, y) = 300 150                              1차원으로 나옴
    point 1:
    2차원 어레이 반환: contours[0][1]=[[400 300]], (1, 2), ndim=2
    1차원 어레이 반환: contours[0][1, 0]=[400 300], (2,), ndim=1
        (x, y) = 400 300
    point 2:
    2차원 어레이 반환: contours[0][2]=[[150 450]], (1, 2), ndim=2
    1차원 어레이 반환: contours[0][2, 0]=[150 450], (2,), ndim=1
        (x, y) = 150 450
```

출력 결과와 정리

4



- OpenCV에서는 물체(object)의 형체를 나타낼 때 다음 구조체를 사용한다.
  - 1. contours(윤곽선)은 여러 개의 윤곽선(contour)로 이루어진 list 구조체로 구성한다 . 즉, contours = [contour, contour, ...]
  - 2.contour는 여러 점들의 ndarray 구조체로 구성된다.
- 보통 점은 (x, y), 2축 좌표면 충분한데 여기서는 용도 불명의 좌표축 1개를 더 추가하여 (1, 2)로 구성한다. 즉, contour = np.array([x, y], [x, y], [x, y],...])로 정의해도 될 것을 차원을 1개 늘려서 contour = np.array([[x, y], [x, y], [x, y],...]]) 표시한다는 것이다. 이때문에 (x, y) 좌표 정보를 액세스 하기 위해서는 다음 기법들이 사용된다.
  - 1) contours[0][점의_번호, 0]
  - 2) squeeze = np.squeeze(contour) # (점의_번호, 1, 2) →(점의_번호, 2), squeeze()은 요소의 갯수가 1인 차원을 제거한다.