OpenCV Tutorial Video

ai-contents

Exported on 06/18/2023

Table of Contents

1 01 OpenCV 기초 3
1.1 01.1 이미지 읽어오기 3
1.2 01.2 이미지 사이즈 조절 3
1.3 01.3 이미지 색상 공간 변환 (Convert Color) 4
1.4 01.4 이미지 저장 4
2 02 영상에서 이미지 추출 5
2.1 02.1 영상 불러오기 5
2.2 02.2 영상 정보 확인 5
2.3 02.3 영상을 프레임으로 나눠 이미지로 저장 6
3 03 직선 검출 8
3.1 03.1 Canny Edge 검출 8
3.2 03.2 허프 직선 검출 8
4 04. 도형 검출 10
4.1 04.1 허프 원 검출 10
4.2 04.2 외곽선 검출 함수 11
4.3 04.3 외곽선 그리는 함수 11
4.4 04.4 도형 검출 예제 12
5 05 색상의 개념 14
5.1 05.1 RGB 14
5.2 05.2 HSL 14
5.3 05.3 HSV 15
6 06 특정 색 검출 16
6.1 06.1 이미지 비트 연산 16
6.2 06.2 inRange 함수를 이용한 색 검출 17
6.3 06.3 영상에서 색 검출 예제 18

1 📌01 OpenCV 기초

1.1 01.1 이미지 읽어오기

```
import cv2

img = cv2.imread('./data/lenna.bmp') # 원하는 경로 지정

cv2.imshow('original', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

특정 키 입력시 창 닫기

```
import cv2

img = cv2.imread('./data/lenna.bmp')
cv2.imshow('original', img)

while True:
    keycode = cv2.waitKey(0)
    if keycode == ord('x'):  # 'x' 키 입력시 종료
        break

cv2.destroyAllWindows()
```

1.2 01.2 이미지 사이즈 조절

```
img = cv2.resize(img,(640,640))
img32 = cv2.resize(img,(320,320))
img16 = cv2.resize(img,(160,160))
imgfx = cv2.resize(img,(0,0), fx=0.5, fy=0.3)
```

1.3 01.3 이미지 색상 공간 변환 (Convert Color)

cv2.cvtColor(): 데이터 타입을 같게 유지하고 채널을 변환

자주 사용되는 파라미터:

- cv2.COLOR_BGR2GRAY
- cv2.COLOR_GRAY2BGR
- cv2.COLOR_BGR2RGB
- cv2.COLOR_BGR2HSV
- cv2..COLOR_BGR2YUV

```
img = cv2.imread('./data/lenna.bmp')
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

1.4 01.4 이미지 저장

cv2.imwrite(filename, image)

```
cv2.imwrite('./data/tmp.jpg', img)
```

---

2 📌02 영상에서 이미지 추출

2.1 02.1 영상 불러오기

```
import cv2

filepath = './data/cat.MOV'
video = cv2.VideoCapture(filepath)

if not video.isOpened():
  print("Video is unavailable :", filepath)
  exit(0)
```

2.2 02.2 영상 정보 확인

cv2.VideoCapture.get(propId) : 카메라, 비디오 장치 속성 값 반환 함수

propId 속성 종류

- **CAP_PROP_FRAME_WIDTH** : 프레임 가로 크기
- **CAP_PROP_FRAME_HEIGHT** : 프레임 세로 크기
- **CAP_PROP_FPS** : 초 당 프레임 수
- **CAP_PROP_FRAME_COUNT** : 비디오 파일의 총 프레임 수
- **CAP_PROP_POS_MSEC** : 밀리 초 단위로 현재 위치
- **CAP_PROP_POS_FRAMES** : 현재 프레임 번호
- **CAP_PROP_EXPOSURE** : 노출

```
import cv2

filepath = './data/cat.MOV'
video = cv2.VideoCapture(filepath)

if not video.isOpened():
  print("Video is unavailable :", filepath)
  exit(0)

length = int(video.get(cv2.CAP_PROP_FRAME_COUNT))
width = int(video.get(cv2.CAP_PROP_FRAME_WIDTH))
```

```
height = int(video.get(cv2.CAP_PROP_FRAME_HEIGHT))
fps = video.get(cv2.CAP_PROP_FPS)

print("length :", length)
print("width :", width)
print("height :", height)
print("fps :", fps)
```

2.3 02.3 영상을 프레임으로 나눠 이미지로 저장

이미지 저장할 폴더 생성

```
import os
import cv2

filepath = './data/cat.MOV'

try:
  if not os.path.exists(filepath[:-4]):
    os.makedirs(filepath[:-4])
except OSError:
  print ('Error: Creating directory. ' + filepath[:-4])
```

영상을 프레임으로 나누어 저장

```
import cv2

filepath = './data/cat.MOV'
video = cv2.VideoCapture(filepath)

if not video.isOpened():
  print("Video is unavailable :", filepath)
  exit(0)

length = int(video.get(cv2.CAP_PROP_FRAME_COUNT))

while(video.isOpened()):
  ret, image = video.read()

  cv2.imwrite(filepath[:-4] + "/frame%d.jpg" % video.get(1), image)
  print('Saved frame number :', str(int(video.get(1))))
```

```
  if int(video.get(1)) == length:
    video.release()
    break
```

1초에 하나의 프레임 저장

```
import cv2

filepath = './data/cat.MOV'
video = cv2.VideoCapture(filepath)

if not video.isOpened():
  print("Video is unavailable :", filepath)
  exit(0)

length = int(video.get(cv2.CAP_PROP_FRAME_COUNT))
fps = video.get(cv2.CAP_PROP_FPS)
count = 0

while(video.isOpened()):
  ret, image = video.read()

  if(int(video.get(1)) % int(fps) == 0):
    cv2.imwrite(filepath[:-4] + "/frame%d.jpg" % count, image)
    print('Saved frame number : ', str(video.get(1)))
    count += 1

  if int(video.get(1)) == length:
    video.release()
    break
```

3 📌03 직선 검출

3.1 03.1 Canny Edge 검출

도로, 건물 등 인위적인 물체에 사용하면 좋음

cv2.Canny(image, threshold1, threshold2, edges=None, apertureSize=None, L2gradient=None) → **edges**

- image: 입력 영상
- threshold1: 하단 임계값
- threshold2: 상단 임계값
    - threshold1:threshold2 = 1:2 또는 1:3
- edges: 에지 영상
- apertureSize: 소벨 연산을 위한 커널 크기. 기본값은 3.
- L2gradient: True이면 L2 norm 사용, False이면 L1 norm 사용. 기본값은 False

```
import sys
import numpy as np
import cv2

src = cv2.imread('./data/building.jpg', cv2.IMREAD_GRAYSCALE)

if src is None:
    print('Image load failed!')
    sys.exit()

dst = cv2.Canny(src, 50, 150)

cv2.imshow('src', src)
cv2.imshow('dst', dst)
cv2.waitKey()

cv2.destroyAllWindows()
```

3.2 03.2 허프 직선 검출

확률적 허프 변환에 의한 직선 검출

cv2.HoughLinesP(image, rho, theta, threshold, lines=None, minLineLength=None, maxLineGap=None) → **lines**

- image: 입력 에지 영상
- rho: 축적 배열에서 rho 값의 간격 (e.g.) 1.0 → 1픽셀 간격
- theta: 축적 배열에서 theta 값의 간격. (e.g.) np.pi/180 → 1° 간격
- threshold: 축적 배열에서 직선으로 판단할 임계값
- lines: 직선 파라미터 (rho, theta) 정보를 담고 있는 numpy.ndarray
- minLineLength: 검출할 선분의 최소 길이
- maxLineGap: 직선으로 간주할 최대 에지 점 간격

```
import sys
import numpy as np
import cv2

src = cv2.imread('../data/building.jpg', cv2.IMREAD_GRAYSCALE)

if src is None:
    print('Image load failed!')
    sys.exit()

edges = cv2.Canny(src, 50, 150)

lines = cv2.HoughLinesP(edges, 1, np.pi/180., 160,
                        minLineLength=50, maxLineGap=5)

dst = cv2.cvtColor(edges, cv2.COLOR_GRAY2BGR)

if lines is not None:
    for i in range(lines.shape[0]):
        pt1 = (lines[i][0][0], lines[i][0][1])  # 시작점 좌표
        pt2 = (lines[i][0][2], lines[i][0][3])  # 끝점 좌표
        cv2.line(dst, pt1, pt2, (0,0,255), 2, cv2.LINE_AA)

cv2.imshow('src', src)
cv2.imshow('dst', dst)
cv2.waitKey()
```

4 📌04. 도형 검출

4.1 04.1 허프 원 검출

cv2.HoughCircles(image, method, dp, minDist, circles=None, param1=None, param2=None, minRadius=None, maxRadius=None) → **circles**

- image: 입력 영상 (에지 영상이 아닌 일반 영상)
- method: OpenCV 4.2 이하에서는 cv2.HOUGH_GRADIENT만 지정 가능
- dp: 입력 영상과 축적 배열의 크기 비율.
    - 1이면 동일 크기
    - 2이면 축적 배열의 가로, 세로 크기가 입력 영상의 반
- minDist: 검출된 원 중심점들의 최소 거리
- circles:(cs, cy, r) 정보를 담은 numpy.ndarray
- param1: 케니 에지 검출기의 높은 임계값
- param2: 축적 배열에서 원 검출을 위한 임계값
- minRadius, maxRadius: 검출할 원의 최소, 최대 반지름

```
import sys
import numpy as np
import cv2

src = cv2.imread('./data/dial.jpg')

if src is None:
    print('Image open failed!')
    sys.exit()

gray = cv2.cvtColor(src, cv2.COLOR_BGR2GRAY)
blr = cv2.GaussianBlur(gray, (0,0), 1.0)

circles = cv2.HoughCircles(blr, cv2.HOUGH_GRADIENT, 1, 50,
                           param1=120, param2=50, minRadius=10, maxRadius=120)

dst = src.copy()
if circles is not None:
    for i in range(circles.shape[1]):
        cx, cy, radius = circles[0][i]
        cv2.circle(dst, (int(cx), int(cy)), int(radius), (0, 0, 255), 2, cv2.LINE_AA)

cv2.imshow('original', src)
cv2.imshow('detected', dst)

cv2.waitKey()

cv2.destroyAllWindows()
```

4.2 04.2 외곽선 검출 함수

cv2.findContours(image, mode, method, contours=None, hierarchy=None, offset=None) → **contours, hierarchy**

- image: 입력 영상. non-zero 픽셀을 객체로 간주함.
- mode: 외곽선 검출 모드. cv2.RETR_로 시작하는 상수.
- method: 외곽선 근사화 방법. cv2.CHAIN_APPROX_로 시작하는 상수.
- contours: 검출된 외곽선 좌표.
- hierarchy: 외곽선 계층 정보.
- offset: 좌표 값 이동 옵셋. 기본값은 (0, 0).

4.3 04.3 외곽선 그리는 함수

cv2.drawContours(image, contours, contourIdx, color, thickness=None, lineType=None, hierarchy=None, maxLevel=None, offset=None) → **image**

- image: 입출력 영상
- contours: (cv2.findContours() 함수로 구한) 외곽선 좌표 정보
- contourIdx: 외곽선 인덱스. 음수(-1)을 지정하면 모든 외곽선을 그린다.
- color: 외곽선 색상
- thickness: 외곽선 두께. thickness < 0 이면 내부를 채운다.
- lineType: LINE_4, LINE_8, LINE_AA 중 하나 지정
- hierarchy: 외곽선 계층 정보
- maxLevel: 그리기를 수행할 최대 외곽선 레벨. maxLevel = 0 이면 contourIdx로 지정된 외곽선만 그린다.

```
import cv2
import sys

img = cv2.imread('./data/polygon.bmp', cv2.IMREAD_GRAYSCALE)
if img is None:
    print('Image load failed!')
    sys.exit()

_, img_bin = cv2.threshold(img, 0, 255, cv2.THRESH_OTSU)

contours, _ = cv2.findContours(img_bin, cv2.RETR_LIST, cv2.CHAIN_APPROX_NONE) # 외곽선
검출
dst = cv2.cvtColor(img, cv2.COLOR_GRAY2BGR)
```

```
for i in range(len(contours)):
    cv2.drawContours(dst, contours, i, (0, 0, 255), 1, cv2.LINE_AA) # 검출된 외곽선 그리기

cv2.imshow('img', img)
cv2.imshow('img_bin', img_bin)
cv2.imshow('dst', dst)
cv2.waitKey()
cv2.destroyAllWindows()
```

4.4 04.4 도형 검출 예제

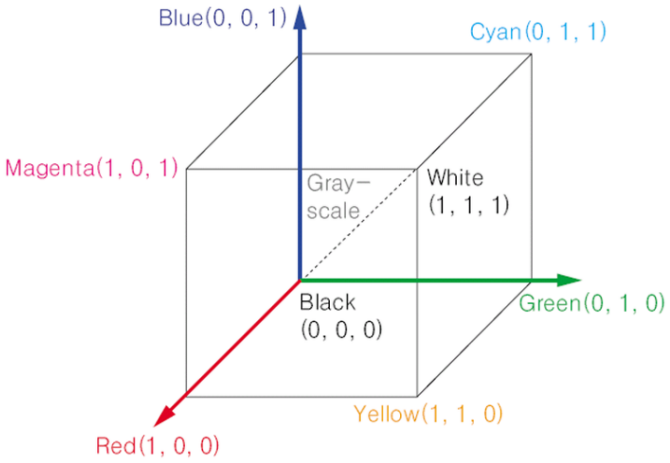
```
import math
import cv2

def setLabel(img, pts, label):
    (x, y, w, h) = cv2.boundingRect(pts)
    pt1 = (x, y)
    pt2 = (x + w, y + h)
    cv2.rectangle(img, pt1, pt2, (0, 0, 255), 1)
    cv2.putText(img, label, pt1, cv2.FONT_HERSHEY_PLAIN, 1, (0, 0, 255))

def main():
    src = cv2.imread('./data/polygon.bmp', cv2.IMREAD_COLOR)
    dst = src.copy()
    if src is None:
        print('Image load failed!')
        return

    gray = cv2.cvtColor(src, cv2.COLOR_BGR2GRAY)
    _, img_bin = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)
    contours, _ = cv2.findContours(img_bin, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)

    for pts in contours:
        if cv2.contourArea(pts) < 400:  # 너무 작으면 무시
            continue

        approx = cv2.approxPolyDP(pts, cv2.arcLength(pts, True)*0.02, True)

        vtc = len(approx)

        if vtc == 3:
            setLabel(dst, pts, 'TRI')
        elif vtc == 4:
            setLabel(dst, pts, 'RECT')
        else:
```

```
            length = cv2.arcLength(pts, True)
            area = cv2.contourArea(pts)
            ratio = 4. * math.pi * area / (length * length)

            if ratio > 0.85:
                setLabel(dst, pts, 'CIR')

    cv2.imshow('src', src)
    cv2.imshow('dst', dst)
    cv2.waitKey()
    cv2.destroyAllWindows()

if __name__ == '__main__':
    main()
```

5 05 색상의 개념

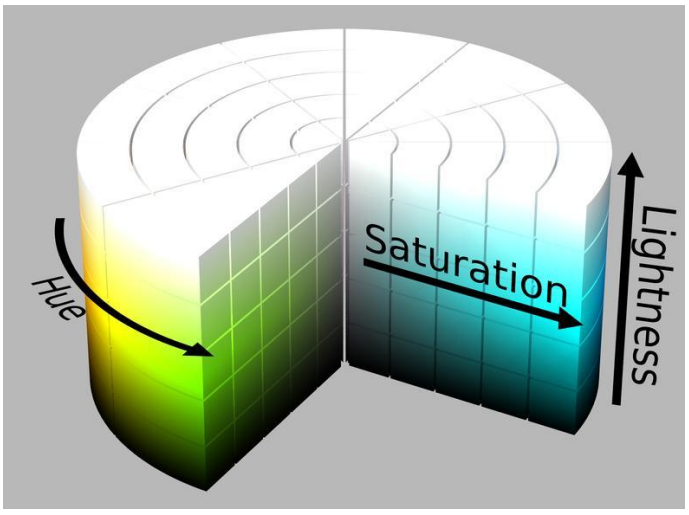
5.1 05.1 RGB

빨강(Red), 녹색(Green), 파랑(Blue) 3원색을 0-255 범위 내에서 조절에 색을 표현



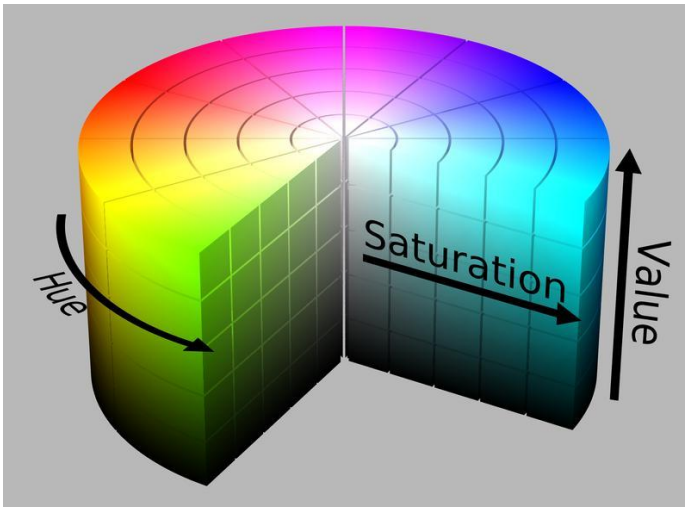
5.2 05.2 HSL

색조(**H**ue), 채도(**S**aturation), 밝기(**L**ightness)



5.3 05.3 HSV

색조(**H**ue), 채도(**S**aturation), 명도(**V**alue)



6 📌06 특정 색 검출

6.1 06.1 이미지 비트 연산

- 이미지의 채널마다 픽셀을 비교하여 판단하는 함수
- src_m에 masking 이미지를 넣으면 masking 됨

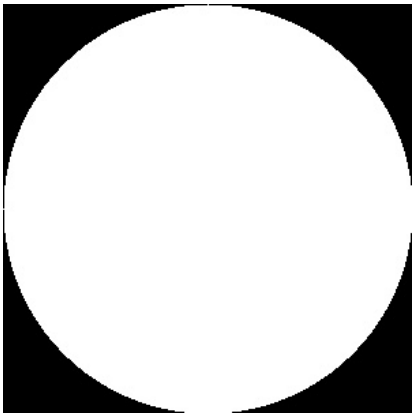
```
cv2.bitwise_and(src1, src2, mask = src_m)
```

```
cv2.bitwise_or(src1, src2, mask = src_m)
```

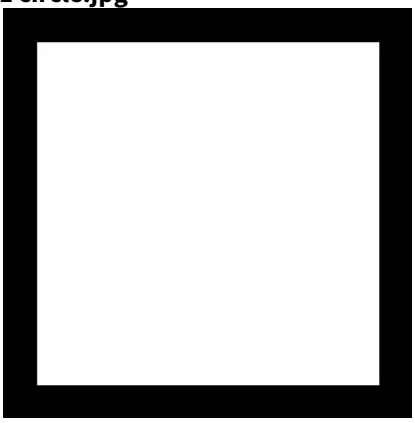
```
cv2.bitwise_xor(src1, src2, mask = src_m)
```

A	B	A and B	A or B	A xor B
1	1	1	1	0
1	0	0	1	1
0	1	0	1	1
0	0	0	0	0

Input image



1 circle.jpg



2 rectangle.jpg

6.2 06.2 inRange 함수를 이용한 색 검출

cv2.inRange(src, lowerb, upperb, dst=None) → dst

- src: 입력 영상
- lowerb: 하한 값 행렬 또는 스칼라
- upperb: 상한 값 행렬 또는 스칼라
- dst: 입력 영상과 같은 크기의 마스크 영상

```
import sys
import numpy as np
import cv2

src = cv2.imread('./data/candies.png')

if src is None:
    print('Image load failed!')
    sys.exit()

src_hsv = cv2.cvtColor(src, cv2.COLOR_BGR2HSV)

dst1 = cv2.inRange(src, (0,128,0), (100,255,100))
dst2 = cv2.inRange(src_hsv, (50,150,0), (80,255,255))

cv2.imshow('src', src)
cv2.imshow('dst1', dst1)
cv2.imshow('dst2', dst2)
cv2.waitKey()

cv2.destroyAllWindows()
```

6.3 06.3 영상에서 색 검출 예제

파란색 검출 예제

```
# 색 검출 예제
import cv2
import numpy as np

## 저장된 동영상 사용
# fpath = './data/chroma_key.mp4'
# video = cv2.VideoCapture(fpath)

## 로봇팔과 함께 제공되는 카메라 사용
video = cv2.VideoCapture(1)

# 추출 여부 플래그
```

```
do_composit = False

while True:
    ret, frame = video.read()

    if not ret:
        break

    if do_composit:

        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

        lower_blue = np.array([120-15, 30, 30])  # 지정할 파란색 범위
        upper_blue = np.array([120+15, 255, 255])

        mask_B = cv2.inRange(hsv, lower_blue, upper_blue) # 파란색만 검출
        res = cv2.bitwise_and(frame, frame, mask=mask_B)

        cv2.imshow('frame', res)
    else:
        cv2.imshow('frame', frame)

    key = cv2.waitKey(30)
    if key == ord(' '):  # Space 누르면 원본 모드 <--> 검출 모드
        do_composit = not do_composit
    elif key == 27:
        break

video.release()
cv2.destroyAllWindows()
```

OpenCV에서 카메라 출력

cv2.VideoCapture(index)

- index: 카메라 장치 번호
- 내장 카메라 또는 외장 카메라 정보를 받아올 수 있다.
- 내장 카메라는 0
- 외장 카메라는 1~n

사용 가능한 카메라 인덱스 확인

```
def returnCameraIndexes():
    # checks the first 10 indexes.
```

```
    index = 0
    arr = []
    i = 10
    while i > 0:
        cap = cv2.VideoCapture(index)
        if cap.read()[0]:
            arr.append(index)
            cap.release()
        index += 1
        i -= 1
    return arr

returnCameraIndexes()
```