사용자 인터페이스 및 IO 처리

윈도우 제어

❖ 윈도우 생성과 제어

cv2.namedWindow(winname[, flags]) → None

■ 설명: 윈도우 이름을 설정한 후, 해당 이름으로 윈도우 생성

인수 설명

■ winname(str) 윈도우 이름

■ flags(int) 윈도우의 크기 조정

옵션	값	설명
cv2.WINDOW_NORMAL	0	윈도우 크기 재조정 가능
cv2.WINDOW_AUTOSIZE	1	표시될 행렬의 크기에 맞춰 자동 조정

윈도우 제어

❖ 윈도우 생성과 제어

cv2.imshow(winname, mat) → None

■ 설명: winname 이름의 윈도우에 mat 행렬을 영상으로 표시함. 생성된 윈도우가 없으면, winname 이름으로 윈도우를 생성하고, 영상을 표시한다.

인수 설명	■ mat(numpy.ndarray) 윈도우에 표시되는 영상(행렬이 화소값을 밝기로 표시)

cv2.destroyAllWindows() → None

■ 설명: HighGUI로 생성된 모든 윈도우 파괴

윈도우 제어

❖ 윈도우 생성과 제어

cv2.moveWindow(winname, x, y) → None

■ 설명: winname 이름의 윈도우를 지정된 위치인 (x, y)로 이동. 이동되는 윈도우의 기준 위치는 좌측 상단임

인수 설명	■ x, y	모니터 안에서 이동하려는 위치의 x, y 좌표

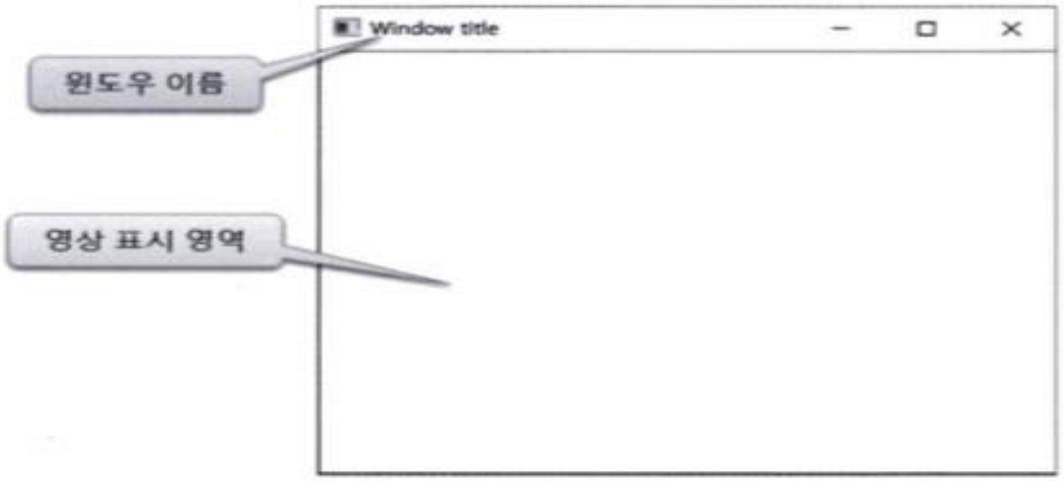
cv2.resizeWindow(winname, width, height) → None

■ 설명: 윈도우의 크기를 재조정한다.

인수 설명	■ width, height	변경 윈도우의 가로, 세로 크기

윈도우 제어

❖ 윈도우 생성 과 제어



✓ cv2.namedWindow() 함수에 윈도우 이름을 지정하여 윈도우를 생성하고 생성된 이름으로 창 크기 변경, 창 닫기, 창 이동 등을 제어할 수 있으며 이벤트 발생도 해당 윈도우 이름으로 제어

이벤트 처리 함수

- 이벤트 처리
  - ✓ 이벤트는 프로그램에 의해 감지되고 처리 될 수 있는 동작이나 사건
  - ✓ 윈도우 운영체제에서는 다양한 이벤트가 발생하며 이러한 이벤트의 처리를 통해 사용하기 편리한 대화형 프로그램을 만들 수 있음
  - ✓ 이벤트를 처리하기 위해 콜백(callback) 함수를 사용하는데 콜백 함수는 개발자가 시스템 함수를 직접 호출하는 방식이 아니라 어떤 이벤트가 발생하거나 특정 시점에 도달했을 때 시스템이 개발자가 등록한 함수를 호출하는 방식
  - ✓ OpenCV에서도 기본적인 이벤트 처리 함수들을 지원하는데 대표적으로 키보드 이벤트, 마우스 이벤트, 트랙바(trackbar) 이벤트를 처리하는 콜백 함수들이 있음

이벤트 처리 함수

- 키보드 이벤트 처리
  - ✓ cv2.waitKey() 함수는 키보드로부터 입력된 1 바이트의 키 값을 리턴
  - ✓ cv2.waitKeyEx() 함수는 2 바이트 키 값을 입력
  - ✓ ms 단위의 delay를 매개변수로 설정할 수 있는데 delay <= 0 인 경우에는 키 이벤트가 발생할 때까지 무한정 대기하고 delay>0 인 경우는 delay 시간(ms)만큼 키보드의 입력을 기다리고 키가 입력되면 해당 키의 코드 값을 리턴하는데 delay 시간 동안 키 이벤트가 발생하지 않으면 -1의 값을 리턴

이벤트 처리 함수

- ❖ 마우스 이벤트 처리
  - ✓ cv2.setMouseCallback(windowName, onMouse[,param]) 함수는 windowName 윈도우에서 발생하는 마우스 이벤트 처리 핸들러 함수를 onMouse 매개변수에 설정
  - ✓ param는 onMouse() 함수로 전달될 추가 정보
  - ✓ onMouse()
    - ❑ onMouse(event, x, y, flags, param=None)
      - ○ event: 발생한 마우스 이벤트의 종류
      - ○ x, y: 이벤트 발생 시 마우스 포인터의 x, y 좌표
      - ○ flags: 조합키 선택 여부

옵션	값	설명
cv2.EVENT_FLAG_LBUTTON	1	왼쪽 버튼 누르기
cv2.EVENT_FLAG_RBUTTON	2	오른쪽 버튼 누르기
cv2.EVENT_FLAG_MBUTTON	4	중간 버튼 누르기
cv2.EVENT_FLAG_CTRLKEY	8	[Ctrl] 키 누르기
cv2.EVENT_FLAG_SHIFTKEY	16	[Shift] 키 누르기
cv2.EVENT_FLAG_ALTKEY	32	[Alt] 키 누르기

      - ○ param콜백 함수로 전달하는 추가적인 사용자 정의 함수

이벤트 처리 함수

❖ 마우스 이벤트 처리

✓ onMouse()

❑ onMouse(event, x, y, flags, param=None)

○ event 종류

옵션	값	설명
cv2.EVENT_MOUSEMOVE	0	마우스 움직임
cv2.EVENT_LBUTTONDOWN	1	왼쪽 버튼 누르기
cv2.EVENT_RBUTTONDOWN	2	오른쪽 버튼 누르기
cv2.EVENT_MBUTTONDOWN	3	중간 버튼 누르기
cv2.EVENT_LBUTTONUP	4	왼쪽 버튼 떼기
cv2.EVENT_RBUTTONUP	5	오른쪽 버튼 떼기
cv2.EVENT_MBUTTONUP	6	중간 버튼 떼기
cv2.EVENT_LBUTTONDBLCLK	7	왼쪽 버튼 더블클릭
cv2.EVENT_RBUTTONDBLCLK	8	오른쪽 버튼 더블클릭
cv2.EVENT_MBUTTONDBLCLK	9	중간 버튼 더블클릭
cv2.EVENT_MOUSEWHEEL	10	마우스 휠
cv2.EVENT_MOUSEHWHEEL	11	마우스 가로 휠

이벤트 처리 함수

❖ 트랙바 이벤트 처리

✓ 트랙바를 생성하여 윈도우에 붙이고 마우스로 트랙바의 슬라이더를 움직여서 정수값을 입력받는데 슬라이더의 최소값은 항상 0

✓ 트랙바 이벤트 처리 함수

함 수	비고
cv.CreateTrackbar(trackbarName, windowName, value, count, onChange) → None	비디오 출력 객체 생성
cv2.setTrackbarPos(trackbarname, winname, pos)	비디오에 이미지 출력
cv2.getTrackbarPos(trackbarname, winname) → retval	비디오 출력 객체 해제

❑ cv.CreateTrackbar() 함수는 윈도우에 트랙바를 생성하고 trackbamame는 트랙바 이름, winname 는 윈도우 이름, value는 트랙바를 생성할 때 슬라이더의 위치이며 count는 슬라이더의 최대값이며 onChange() 함수는 슬라이더 위치가 변경될 때마다 슬라이더 이벤트를 처리를 위해 호출될 함수

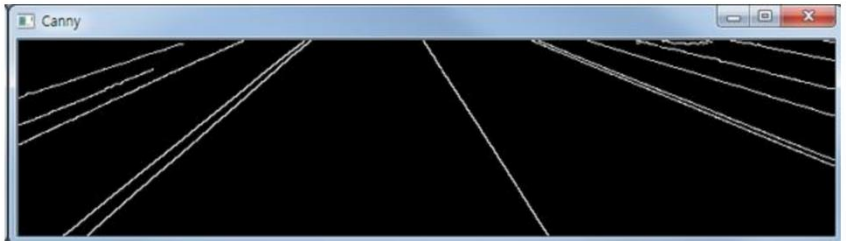
❑ onChange 함수는 매개변수가 1개인데 이 매개변수는 트랙바의 슬라이더 위치에 해당

❑ cv2.setTrackbarPos() 함수는 트랙바의 위치를 pos로 변경

❑ cv2.getTrackbarPos() 함수는 트랙바의 현재 위치를 반환

그리기 함수

- 그리기 함수
  - ✓ 영상처리 프로그래밍 과정에서 해당 알고리즘을 적용했을 때 처리가 정확히 적용되었는지를 출력 영상 위에 사각형이나 원을 그려서 확인하는 경우가 많이 있음
  - ✓ 그림과 같이 얼굴 검출 알고리즘을 적용했을 때 전체 영상 위에 검출한 얼굴 영역을 사각형이나 원으로 표시할 수 있으며 차선을 확인하고자 직선 검출 알고리즘을 적용했을 때 차선을 정확하게 검출했는지 확인하기 위해 도로 영상 위에 선으로 나타낼 수 있음



그리기 함수

❖ 직선 및 사각형 그리기
- ✓ line() 함수는 직선
- ✓ rectangle() 함수는 사각형
- ✓ clipLine() 함수는 사각형과 직선의 교점 좌표를 계산

함 수	비고
cv2.line(img, pt1, pt2, color [, thickness[, lineType[, shift]]])	직선
cv2.rectangle(img, pt1, pt2, color [, thickness[, lineType[, shift]]])	사각형
cv2.clipLine(imgRect, pt1, pt2) → retval, pt1, pt2	사각형-직선 절단 좌표

- ❑ cv2.line()은 img에 좌표 pt1에서 pt2까지 연결하는 직선을 그리는데 color 색상 thickness 두께, lineType은 cv2.LINE_8(디폴트), cv2. LINE_4, cv. LINE__AA 등이 있으며 shift는 pt1과 pt2의 각 좌표에 대한 비트 이동(shift right)을 설정
- ❑ cv2.rectangle()은 img에 좌표 pt1, pt2에 의해 정의되는 직각 사각형을 그리는데 color 색상, thickness 두께로 thickness = -1이면 color 색상으로 채운 사각형을 그림
- ❑ cv2.clipLine()은 좌표 pt1에서 pt2까지의 직선이 imgRect 사각형에 의해 절단되는 좌표점을 계산하여 pt1과 pt2에 반환하는데 직선이 사각 영역 밖에 있으면 retval에 False를 반환

그리기 함수

❖ 원 및 타원 그리기

✓ 원 및 타원 그리기 함수

함 수	비고
cv2.circle(img, center, radius, color [, thickness[, lineType[, shift]]])	원 그리기
cv2.ellipse(img, center, axes, angle, startAngle, endAngle, color [, thickness[, lineType[, shift]]])	타원 그리기
cv2.ellipse(img, box, color[, thickness[, lineType]])	회전 사각형 내접 타원
cv2.ellipse2Poly(center, axes, angle, arcStart, arcEnd, delta) → pts	타원 위 좌표 계산

- cv2.circle()는 영상 img에 중심점 center, 반지름 radius의 원을 색상 color, 두께 thickness로 그리며 thickness = CV_FILLED(-1)이면 color 색상으로 채운 원을 그림
- cv2.ellipae()는 영상 img에 중심점 center, 주축의 크기의 절반 axes, 수평축과의 회전 각도 angle, 호(arc)의 시작과 끝의 각도는 startAngle, endAngle인 타원을 그리며 startAngle = 0, endAngle = 360이면 닫힌 타원을 그리고 thickness = CV_FLLED(-1)이면 color 색상으로 채운 타원을 그림
- cv2.ellipse()는 영상 img에 회전된 사각형 box = (center, size, angle)에 내접하는 타원을 그리며 center는 중심점, size는 크기, angle은 수평축과의 각도
- cv2.ellipse2Poly()는 중심점 center, 축의 크기 axes, 각도 angle, 호(arc)의 시작과 끝의 각도 startAngle, endAngle인 타원 위의 좌표를 delta 각도 간격으로 계산하여 반환

그리기 함수

❖ 다각형 그리기

✓ 다각형 그리기

함수	비고
cv2.polylines(img, pts, isClosed, color [, thickness[, lineType[, shift]]])	다각형 그리기
cv2.fillConvexPoly(img, points, color[, lineType[, shift]])	볼록다각형 채우기
cv2.fillPoly(img, pts, color[, lineType[, shift[, offset]]])	다각형 채우기

- cv2.polylines()는 하나 또는 그 이상의 다각형을 color 색상으로 그리는데 pts는 다각형들의 numpy 배열이고 isClosed = True이면 닫힌 다각형을 그림
- cv2.fillConwxPoly()는 points에 저장된 좌표로 이루어진 볼록 다각형 또는 일반 다각형을 color 색상으로 채웁니다. fillPoly() 함수보다 빠르게 그림
- cv2.fillPoly()는 하나 또는 그 이상의 다각형을 color 색상으루 채우는데 pts는 다각형들의 numpy 배열

그리기 함수

❖ 문자열 출력

✓ 문자열 출력 함수

함 수	비고
cv2.getTextSize(text, fontFace, fontScale, thickness) → retval, baseLine	문자열 출력 크기 반환
cv2.putText(img, text, org, fontFace, fontScale, color[, thickness[, lineType[, bottomLeftOrigin]]])	문자열 출력

- cv2.getTextSize()은 문자열 text의 출력을 위한 크기를 retval에 반환하고 출력될 사각 영역의 하단으로부터의 상대적 기준선(baseLine) y의 위치를 반환
- cv2.putText()는 문자열 text를 사각형 왼쪽 아래 좌표 위치(org)에 폰트(fontFace), 폰트 스케일(fontScale), 색상(color)으로 문자열을 출력하며 thickness는 글자의 굵기이고 lineType은 글자 선의 형태이고 bottomLeftOrigin은 영상의 원점 좌표 설정(True- 좌하단 왼쪽, False- 좌상단)

그리기 함수

- ❖ 문자열 출력
  - ✓ 문자열 출력 함수
    - ❑ 문자열의 fontFace에 대한 옵션

옵션	값	설명
cv2.FONT_HERSHEY_SIMPLEX	0	중간 크기 산세리프 폰트
cv2.FONT_HERSHEY_PLAIN	1	작은 크기 산세리프 폰트
cv2.FONT_HERSHEY_DUPLEX	2	2줄 산세리프 폰트
cv2.FONT_HERSHEY_COMPLEX	3	중간 크기 세리프 폰트
cv2.FONT_HERSHEY_TRIPLEX	4	3줄 세리프 폰트
cv2.FONT_HERSHEY_COMPLEX_SMALL	5	COMPLEX 보다 작은 크기
cv2.FONT_HERSHEY_SCRIPT_SIMPLEX	6	필기체 스타일 폰트
cv2.FONT_HERSHEY_SCRIPT_COMPLEX	7	복잡한 필기체 스타일
cv2.FONT_ITALIC	16	이탤릭체를 위한 플래그

영상 파일 처리

❖ 영상 입출력 과 디스플레이 함수

함 수	비고
cv2.imread(filename[, flags]) → retval	영상 입력
cv2.imwrite(filename, img[, params])→ retval	영상파일 출력

영상 파일 처리

- 영상 입출력 과 디스플레이 함수
  - ✓ 이미지는 imread 메소드를 이용해서 numpy의 ndarray 타입으로 가져오는 것이 가능

    cv2.imread(이미지 파일 경로, 이미지 옵션)
    - ❑ 이미지 옵션
      - cv2.IMREAD_UNCHANGED(-1): Alpha 채널 포함
      - cv2.IMREAD_GRAYSCALE(0): 흑백
      - cv2.IMREAD_COLOR(1): 컬러(기본값)
      - cv2.IMREAD_ANYDEPTH(2): 입력 파일에 정의된 깊이에 따라 16/32 비트 영상으로 변환하고 설정하지 않으면 8비트 영상으로 변환
      - cv2.IMREAD_ANYCOLOR(4): 입력 파일에 정의된 타입의 영상을 반환
  - ✓ 이미지 파일의 출력

    matplotlib.pyplot.imshow(array-like or PIL image Data..)
- 이미지 데이터
  - ✓ 이미지는 2차원 ndarray로 리턴 – shape 속성이 이미지의 해상도
  - ✓ 흑백의 경우는 각 픽셀 값은 검정(0) – 흰색(255) 사이의 값
  - ✓ 컬러의 경우는 BGR(Blue, Green, Red)로 리턴하기 때문에 출력하기 전에 cv2.cvtColor 함수를 이용해서 RGB로 변환해서 출력

영상 파일 처리

❖ 이미지 파일의 저장

cv2.imwrite(파일 경로, 이미지 데이터, params=JPG 파일의 화질이나 PNG 파일의 압축율)

- ✓ 이미지 데이터를 파일 경로에 저장
- ✓ 기존에 파일이 존재하면 파일을 덮어씀
- ✓ 이미지 포맷은 파일의 확장자에 의존
- ✓ 압축 방식에 사용되는 params 인수 튜플

paramId	paramValue (기본값)	설명
cv2.IMWRITE_JPEG_QUALITY	0~100 (95)	JPG 파일 화질, 높은 값일수록 화질 좋음
cv2.IMWRITE_PNG_COMPRESSION	0~9 (3)	PNG 파일 압축 레벨, 높은 값일수록 용량은 적어지고, 압축 시간이 길어짐
cv2.IMWRITE_PXM_BINARY	0 or 1 (1)	PPM, PGM 파일의 이진 포맷 설정

비디오 처리

- 비디오 캡쳐
  - ✓ 비디오 캡처는 아날로그 비디오를 디지털 비디오로 변환하는 과정
  - ✓ 프레임(frame)은 비디오에서 캡처한 한 장의 영상이며 비디오는 이러한 프레임들의 연속 시퀀스
  - ✓ 비디오 파일은 코덱에 의해 해당 포맷에 맞게 압축하여 저장되고 저장된 비디오 파일은 압축을 해제하여 재생
  - ✓ 코덱은 비디오 신호를 압축하는 코더(coder)와 압축을 해제하는 디코더의 합성어
  - ✓ 비디오 파일에서 영상 프레임을 받아서 영상 처리를 하기 위해서는 먼저 파일 포맷에 따른 코덱을 통해서 압축을 해제
  - ✓ 비디오는 압축 알고리즘으로 인코딩되어 있어 코덱으로 디코딩해야 한 장의 프레임을 얻을 수 있는데 VideoCapture()로 비디오 파일 또는 카메라를 개방하고 while 문에 의한 반복문에서 VideoCapture.read() 함수로 프레임을 한 장씩 캡처하여 영상 프레임을 처리하고 윈도우 화면에 표시

비디오 처리

❖ 비디오 캡쳐

함 수	비고
cv2.VideoCapture() → 〈VideoCapture object〉 cv2.VideoCapture(filename) → 〈VideoCapture object〉 cv2.VideoCapture(device) → 〈VideoCapture object〉	비디오 획득 객체 생성

✓ cv2.VideoCapture()는 비디오 파일(filename) 또는 카메라 번호(device)로부터 VideoCapture 객체를 생성하여 반환하는데 장치 번호는 카메라가 한 개면 0이고 두 개이면 0, 1로 번호를 지정하고 파일명 또는 장치 번호 명시 없이 VideoCapture()로 생성한 경우는 VideoCapture.open(filename) 또는 VideoCapture.open(device)로 비디오를 개방하고 VideoCapture.isOpened()를 사용하면 비디오 객체가 개방되었는지를 확인할 수 있음

비디오 처리

❖ 비디오 캡쳐

함수	비고
cv2.VideoCapture.read([image]) → retval, image	프레임 획득
cv2.VideoCapture.grab() → retval	프레임 잡기
cv2.VideoCapture.retrieve([image[, channel]]) → retval, image	프레임 획득
cv2.VideoCapture.release()	비디오 획득 객체 해제
cv2.VideoCapture.get(propId) → retval	비디오 특성 얻기
cv2.VideoCapture.set(propId, value) → retval	비디오 특성 설정

비디오 처리

- 비디오 캡쳐
  - ✓ cv2.VideoCapture.read()는 개방된 VideoCapture 객체로부터 다음 비디오 프레임을 잡아서(grab) 디코딩하고 프레임을 반환
  - ✓ VideoCapture.grab()과 VideoCapture.retrieve()를 모두 수행한 결과로 대부분의 비디오 프레임 캡처를 위해서 VideoCapture.read()를 사용하는데 프레임 캡처에 성공하면 retval는 True이고 실패하면 False
  - ✓ cv2.VideoCapture.grab()은 개방된 VideoCapture 객체에서 다음 프레임을 잡기(grab) 위해 사용하는데 프레임을 캡처하기 위해서는 VideoCapture.retrieve()를 사용하고 여러 대의 카메라(스테레오 카메라, Kinect 등)에서 동기화를 목적으로 사용
  - ✓ cv2.VideoCapture.retrieve()는 VideoCapture.grab()에 의해 잡힌 영상을 디코딩하여 image로 반환하는데 프레임을 캡처하면 retval은 True이고 실패하면 False
  - ✓ cv2.VideoCapture.release()는 개방된 VideoCapture 객체를 해제하여 닫음
  - ✓ cv2.VideoCapture.get()은 개방된 VideoCapture 객체의 특성을 실수(property_id)로 반환하는데 비디오 파일에서 프레임 속도, 총 프레임 수 등이 올바르게 설정되지 않은 값이 있을 수 있음

비디오 처리

- 비디오 캡쳐
  - ✓ cv2.VideoCapture.set()는 개방된 VideoCapture 객체의 propId 특성을 value로 설정하며 설정의 성공 여부를 참 거짓으로 반환
  - ✓ property_id 의 주요 상수

property_id	설명
cv2.CAP_PROP_POS_MSEC	밀리초(milliseconds)로 현재 위치
cv2.CAP_PROP_POS_FRAMES	캡처될 프레임 번호
cv2.CAP_PROP_FRAME_WIDTH	비디오 프레임의 가로 크기
cv2.CAP_PROP_FRAME_HEIGHT	비디오 프레임의 세로 크기
cv2.CAP_PROP_FPS	프레임 속도
cv2.CAP_PROP_FOURCC	코덱의 4 문자
cv2.CAP_PROP_FRAME_COUNT	비디오 파일에서 총 프레임 수
cv2.CAP_PROP_CONVERT_RGB	영상이 RGB로 변환해야 하는지 여부
cv2.CAP_PROP_FORMAT	캡처된 영상 포맷
CV_CAP_PROP_BRIGHTNESS	카메라에서 영상의 밝기
CV_CAP_PROP_CONTRAST	카메라에서 영상의 대비
CV_CAP_PROP_SATURATION	카메라에서 영상의 포화도
CV_CAP_PROP_HUE	카메라에서 영상의 채도
CV_CAP_PROP_GAIN	카메라에서 영상의 Gain
CV_CAP_PROP_EXPOSURE	카메라에서 영상의 노출

비디오 처리

❖ 비디오 파일 녹화

✓ 비디오 파일과 관련된 VideoWriter 객체를 생성하고 영상 프레임을 write() 함수로 비디오 파일에 출력하여 녹화

✓ 비디오 파일에 출력하여 녹화

함 수	비고
cv2.VideoWriter([filename, fourcc, fps, frameSize[, isColor]]) → 〈VideoWriter object〉	비디오 출력 객체 생성
cv2.VideoWriter.write(image)	비디오 파일에 이미지 출력
cv2.VideoWriter.release()	비디오 출력 객체 해제

❑ cv2.VideoWriter()는 VideoWriter 객체를 생성하여 반환하는 함수로 filename은 비디오 파일의 이름이고, fourcc는 비디오 코덱을 위한 4 문자로 코덱 문자는 http://www.fourcc.org/codecs.php에 있으며 VideoVWriter_fajroo(*'DIVX')는 VideoWriter_fourcc('D', 'I', 'V', 'X')와 같으며 fourcc = - 1이면 압축 코덱 선택 대화상자가 나타나며 fps는 프레임 속도, trameSize 는 프레임의 크기, isOolor = True이면 컬러 비디오, isColor = False이면 그레이스케일 비디오로 VideoWriter 객체만 생성된 경우 VideoWriter.open()으로 비디오를 개방할 수 있으며 VideoWriter. isOpened()로 개방 여부를 확인할 수 있음

비디오 처리

❖ 비디오 파일 녹화

✓ 비디오 파일과 관련된 VideoWriter 객체를 생성하고 영상 프레임을 write() 함수로 비디오 파일에 출력하여 녹화

✓ 비디오 파일에 출력하여 녹화하는 함수

❑ fourcc 비디오 코덱 문자

fourcc	코덱
cv2.VideoWriter_fourcc(*'PIM1')	MPEG-1
cv2.VideoWriter_fourcc(*'MJPG')	Motion-JPEG
cv2.VideoWriter_fourcc(*'DIVX')	DIVX 4.0 이후 버전
cv2.VideoWriter_fourcc(*'XVID')	XVID, MPEG-4
cv2.VideoWriter_fourcc(*'MPEG')	MPEG
cv2.VideoWriter_fourcc(*'X264')	H.264/AVC

❑ cv2.VideoWriter.write()는 개방된 VideoWriter 객체에 image를 출력하는데 주의할 것은 image의 크기는 VideoWriter 객체를 생성할 때 명시한 프레임 크기인 frameSize와 같아야 함

❑ cv2.VideoWriter.release()는 개방된 VideoWriter 객체를 해제

matplotlib 패키지 활용

- matplotlib 패키지 활용
  - ✓ matplotlib는 파이썬에서 데이터를 시각화 해주는 라이브러리
  - ✓ matplotlib에 있는 pyplot 모듈은 Matlab의 수치해석 소프트웨어의 시각화 명령을 거의 그대로 사용할 수 있도록 명령어 집합을 제공
  - ✓ numpy에서 사용되는 자료구조를 쉽게 시각화할 수 있기 때문에 주피터 노트북이나 구글의 Colab 사이트에서 간단하게 결과를 확인할 때에는 OpenCV의 cv2.imshow() 함수를 사용하지 않고 Matplotlib 패키지를 사용하는 것이 편리

matplotlib 패키지 활용

- matplotlib 패키지 활용
  - ✓ 이미지 보간