```
import numpy as np
import cv2
# 카메라 열기 / 비디오 캡쳐 객체 생성
cap = cv2.VideoCapture(0)
# 카메라 열기 실패하면 종료
if not cap.isOpened():
    print("Error: camera open failed")
    exit()

while True:
# 비디오의 한 프레임씩 읽음 //프레임을 읽으면 ret 값이 True, 실패하면 False // frame에 읽은 프레임 나옴
    ret, frame = cap.read()
    if ret:
# 입력 영상 frame의 잡음을 제거하기 위해 medianBlur()함수 적용
        blur = cv2.medianBlur(frame, 5)

# 원을 검출하는 HoughCircles함수는 입력 영상으로 그레이스케일 영상을 전달
        gray = cv2.cvtColor(blur, cv2.COLOR_BGR2GRAY)
# 1 : 축적배열 크기는 입력 영상과 같은 크기 // 100 : 두 원의 중심점 거리가 100픽셀보다 작으면 검출 X
# param1 : 캐니 에지 검출기의 높은 임계값 // param2 : 축적배열 원소값이 30보다 크면 원의 중심점으로 선택
        circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1, 100, param1=150, param2=30)
#circles 값들을 반올림/반내림하고 이른 uint16으로 변환
        circles = np.uint16(np.around(circles))

        for i in circles[0,:]:
# (i[0],i[1]) 은 원의 중심 좌표, i[2]는 원의 반지름의 초록색 원을 그림
            cv2.circle(frame, (i[0],i[1]),i[2],(0,255,0),2)
# 원의 중심 좌표는 빨간색으로 표시
            cv2.circle(frame, (i[0],i[1]),2,(0,0,255),3)

# 직선을 검출하는 HoughLines함수는 입력 영상으로 에지 영상을 전달
        edges = cv2.Canny(gray,50,150)
# 확률적 허프 선 변환 함수 사용(허프선변환은 연산량이 많으므로)
        lines = cv2.HoughLinesP(edges, 1, np.pi/180, 30)
        for line in lines:
            x1, y1, x2, y2 = line[0]
            cv2.line(frame, (x1,y1),(x2,y2), (255,0,0),2)

        cv2.imshow("Circles, Lines Detection", frame)
# Esc 키를 누르면 종료
        if cv2.waitKey(1) & 0xFF == 27:
            break

cap.release()
cv2.destroyAllWindows()
```