설치 참조 : <https://deepdeepit.tistory.com/19>

$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get install build-essential cmake  
  
$ sudo apt-get install cmake

$ sudo apt-get install pkg-config  
  
$ sudo apt-get install libpng12-dev

$ sudo apt-get install libjpeg-dev

$ sudo apt-get install libtiff5-dev

$ sudo apt-get install libjasper-dev  
  
$ sudo apt-get installlibpng12-dev libjpeg-dev libtiff5-dev libjasper-dev  
  
$ sudo apt-get install libavformat-dev

$ sudo apt-get install libxvidcore-dev

$ sudo apt-get install libswscale-dev

$ sudo apt-get install libxine2-dev

$ sudo apt-get install libavcodec-dev

$ sudo apt-get install libx264-dev  
  
$ sudo apt-get install libv4l-dev

$ sudo apt-get installv4l-utils  
$ sudo apt-get install libgstreamer1.0-dev

$ sudo apt-get install libgstreamer-plugins-base1.0-dev  
$ sudo apt-get install libqt4-dev  
$ sudo apt-get install mesa-utils

$ sudo apt-get install libgl1-mesa-dri

$ sudo apt-get install libqt4-opengl-dev  
$ sudo apt-get install libatlas-base-dev

$ sudo apt-get install gfortran

$ sudo apt-get install libeigen3-dev  
$ sudo apt-get install python2.7-dev

$ sudo apt-get install python3-dev

$ sudo apt-get install python-numpy

$ sudo apt-get install python3-numpy

$ mkdir로 임의로 폴더 하나 만들고 이동

->  
$ wget -O opencv.zip https://github.com/opencv/opencv/archive/3.4.0.zip

$ unzip opencv.zip

-> ls로 설치된지 확인

설치된 opencv 폴더로 이동 후 build 폴더만들고 폴더로 이동 후

cmake -D CMAKE\_BUILD\_TYPE=RELEASE \ 실행 후

make -j3 컴파일 하기

컴파일 완료 후 sudo make install 실행 후

Cat /etc/ld.so.conf.d/\* 명령어 실행하여 /usr/local/lib 설정이 포함되어 있는지 확인

Python, python3 명령어를 입력하여 Opencv 확인

홈으로 간 후 rm –rf opencv 로 삭제

얼굴인식 상황에 따른 오픈소스 (깃허브):

<https://github.com/opencv/opencv/tree/master/data/haarcascades>

얼굴인식 참조 : <https://pinkwink.kr/1124>

얼굴인식 참조 글에서 사용된 버전

* Python 3.6
* anaconda 5.1
* 최신 Opencv 아님

image = cv2.imread('./img\_MiBaRui3.jpg') // Cv2.imread로 인식할 이미지 가져옴

grayImage = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

// 가져온 이미지를 흑백(그레이)화

Matplotlib는 파이썬에서 자료를 차트(chart)나 플롯(plot)으로 시각화하는 패키지

Matplotlib 패키지를 사용할 때는 보통 다음과 같이 주 패키지는 mpl 이라는 별칭(alias)으로 임포트하고 pylab 서브패키지는 plt 라는 다른 별칭으로 임포트하여 사용하는 것이 관례

plt.figure(figsize=(**12**,**12**)) // 인수로 이미지를 띄울 가로,세로 cm크기 조정

plt.imshow(image, cmap='gray') // 이미지를 gray색으로 띄움

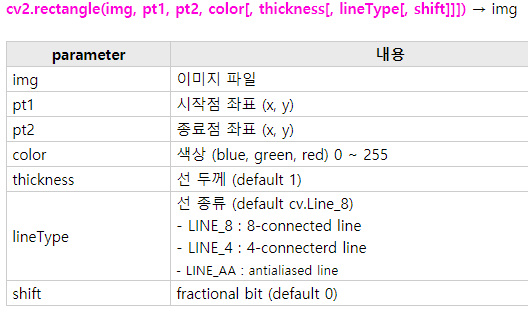
plt.xticks([]), plt.yticks([]) # to hide tick values on X and Y axis

//플롯이나 차트에서 축상의 위치 표시 지점을 틱(tick)이라 함, 표시를 안한다는 뜻

plt.show()

// show 명령은 시각화 명령을 실제로 차트로 렌더링(rendering)하고 마우스 움직임 등의 이벤트를 기다리라는 지시이다. 생성된 모든 figure를 표시해줌

Cv2.rectangle함수를 이용하여 사각형을 그릴 수 있는데, 시작점(좌측 상단)과 종료점(우측 하단) 두곳의 좌표만 기입하여 도형을 그림



face\_cascade = cv2.CascadeClassifier(

'./data/haarcascades/haarcascade\_frontalface\_default.xml')

// 이미지에서 얼굴을 검출하는 코드를 face\_cascade에 대입

image = cv2.imread('./이미지.jpg')

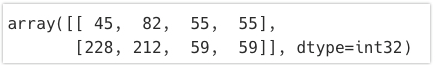
grayImage = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(grayImage, **1.03**, **5**)\

// detectMultiScale 함수 = image에서 object를 검출하는 함수

// 첫 번째 인자인 image는 바로 검출하고자 하는 원본 이미지를 의미합니다.

// 두 번째 인자인 objects에 검출된 이미지가 채워집니다. 그 뒤의 인자부분은 넘어가도 됩니다.

**함수의 결과값인 faces를 입력하면 아래처럼 얼굴을 검출한 수만큼 2차원배열으로 나옴**

**for** (x,y,w,h) **in** faces: // for in 으로 위 배열 순서대로 x,y,w,h에 대입하여 넣는다

cv2.rectangle(image,(x,y),(x+w,y+h),(**0**,**255**,**0**),**2**)

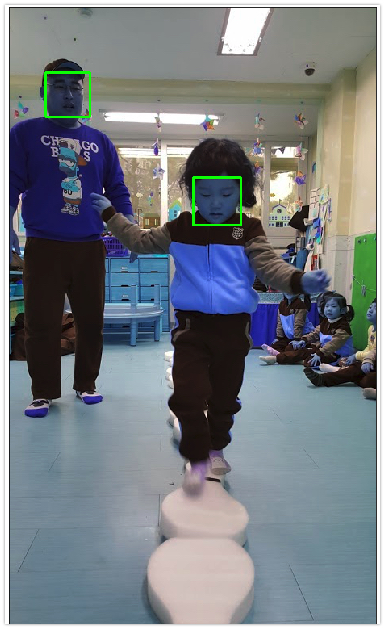
plt.figure(figsize=(**12**,**12**))

plt.imshow(image, cmap='gray')

plt.xticks([]), plt.yticks([]) # to hide tick values on X and Y axis

plt.show()

해당 코드를 실행하면 아래 사진처럼 출력문이 출력된다.



중간 좌표 구하기 수식(현재 시점에서 검출된 얼굴의 중간값으로 가기위해 계산)

x좌표 중간값 : ( (x+w) – x) / 2 ) + x

y좌표 중간값 : ( (y+h) – y) / 2 ) + y

모터 스텝 횟수 구하기 수식(몇번 스텝을 돌릴지 계산)

Middle : 현재 시점

Tox, toy : 검출된 이미지의 중간 x,y 좌표 값

좌,우 모터 스텝 : middle – tox / 스텝각도

상,하 모터 스텝 : middle – toy / 스텝각도

문제점

모터스텝 소스코드 (시간복잡도 해결 가능하다면 해야함)

------------ 소스코드 -----------

Xcount = 0;

Ycount = 0;

If(x > y)

Max = x;

Else

Max = y;

For(int i = 0; i < max; i++){

If(Xcount < x){

For(int j = 0; j < 4; j++)  
 {

For(int z = 0; z < 4; z++)

{

//모터회전

Ex) if(motor[j][z] == j) motor[j][z] = 1;

Else motor[j][z] = 0;

}

}

Xcount++;

}

Else(ycount < y){

For(int j = 0; j < 4; j++)  
 {

For(int z = 0; z < 4; z++)

{

//모터회전

Ex) if(motor[j][z] == j) motor[j][z] = 1;

Else motor[j][z] = 0;

}

}

ycount++;

}

}

-------------- 소스코드 끝 -----------------

전체적인 흐름도

초음파 센서를 통해 사람이 가까이 오는 것을 인식하면 활성화

카메라로 캡처한 이미지를 Opencv를 통해 얼굴인식을 하고 사각틀을 그려 중간값을 구하고

현재 카메라가 인식하고 있는 x,y좌표와 인식한 얼굴의 중간 좌표x,y값을 수식을 통해 계산하여 모터를 구동시킨다.

Opencv를 통해 얼굴인식을 한 최소,최대 x,y 값(4개)로 이미지를 잘라내어 google colud vision으로 감정인식을 실행한다.

Lcd에 감정인식의 결과에 따라 맞는 이모티콘이나 이미지를 출력한다.