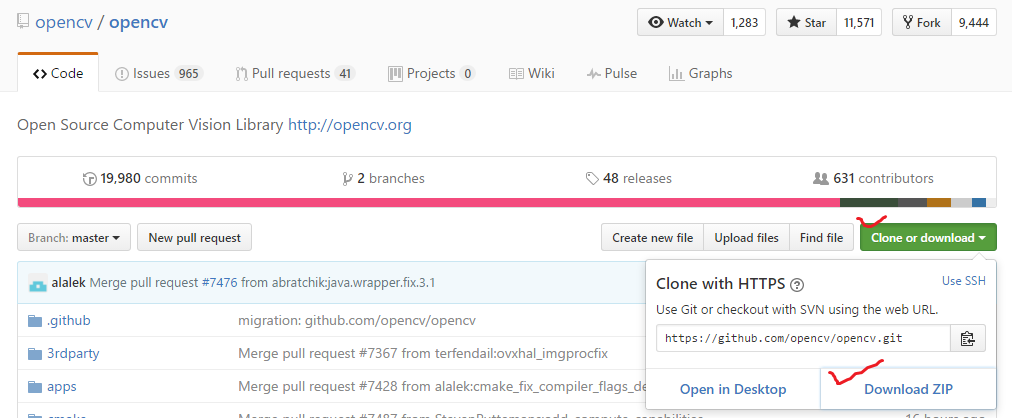
2016년 10월 16일 (일요일)

Open MPI 를 c++ 표준에 도입 시도하고 있다.

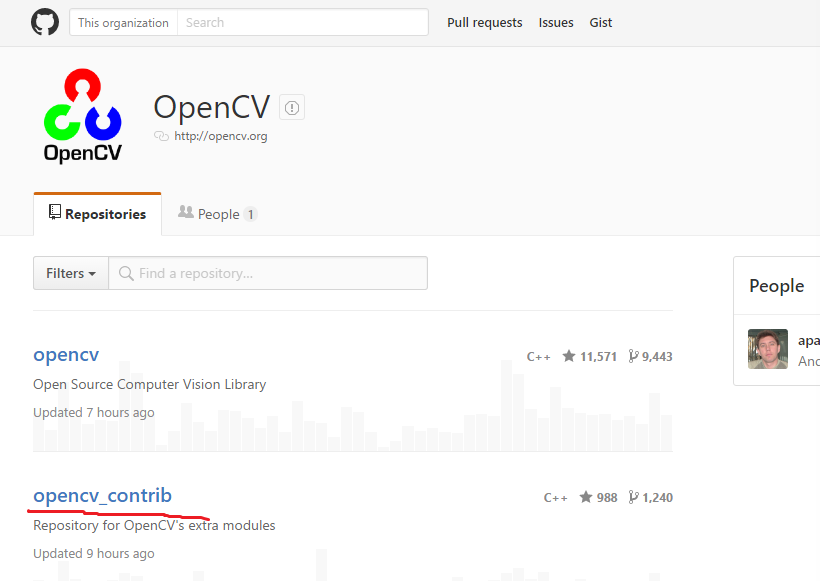
<https://www.open-mpi.org/>

openCV를 빌드해보자

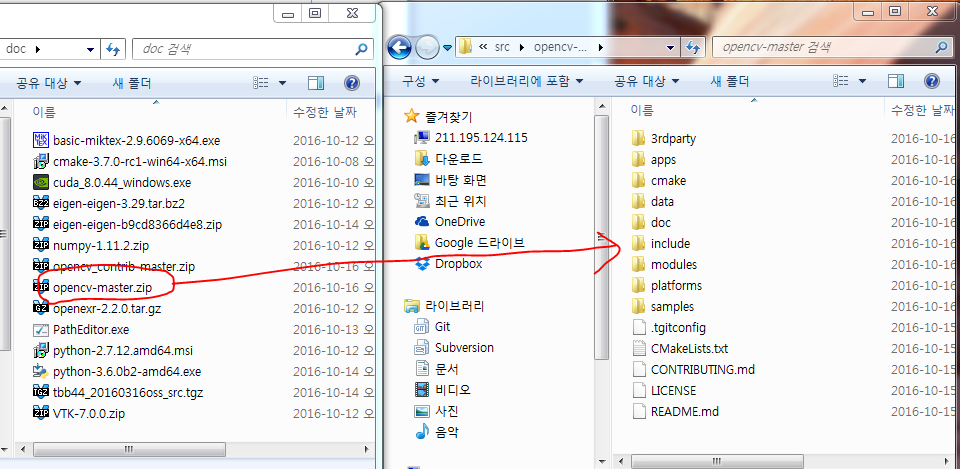
<https://github.com/opencv> 에 접속한다. opencv/opencv\_contrib는 확장 버전인데 일부는 라이선스가 있다. Download ZIP 버튼을 눌러서 다운로드 받는다.



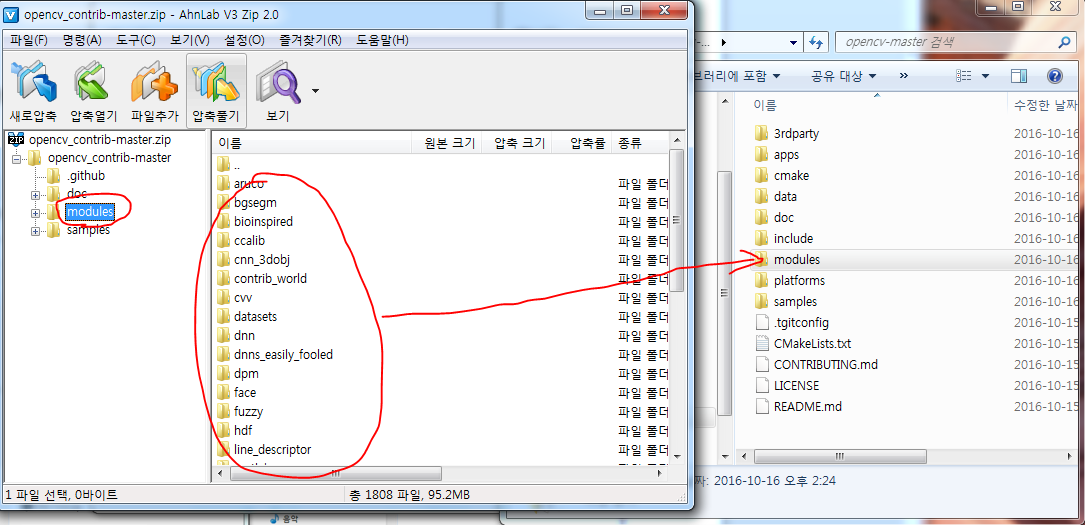
확장 기능 설치를 위해 Contrib로 들어가서 contrib도 다운 받는다. 다운로드 받는 방식은 동일하다.



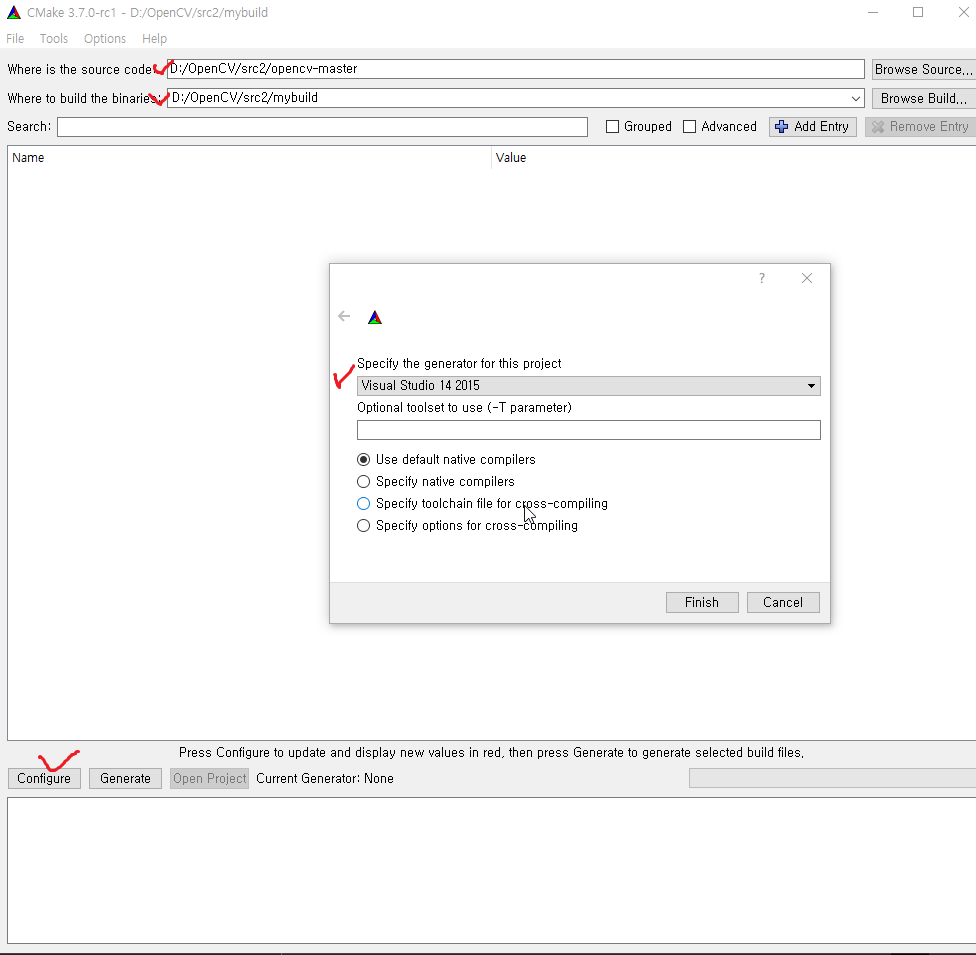
Opencv-master.zip 파일의 압축을 푼다.



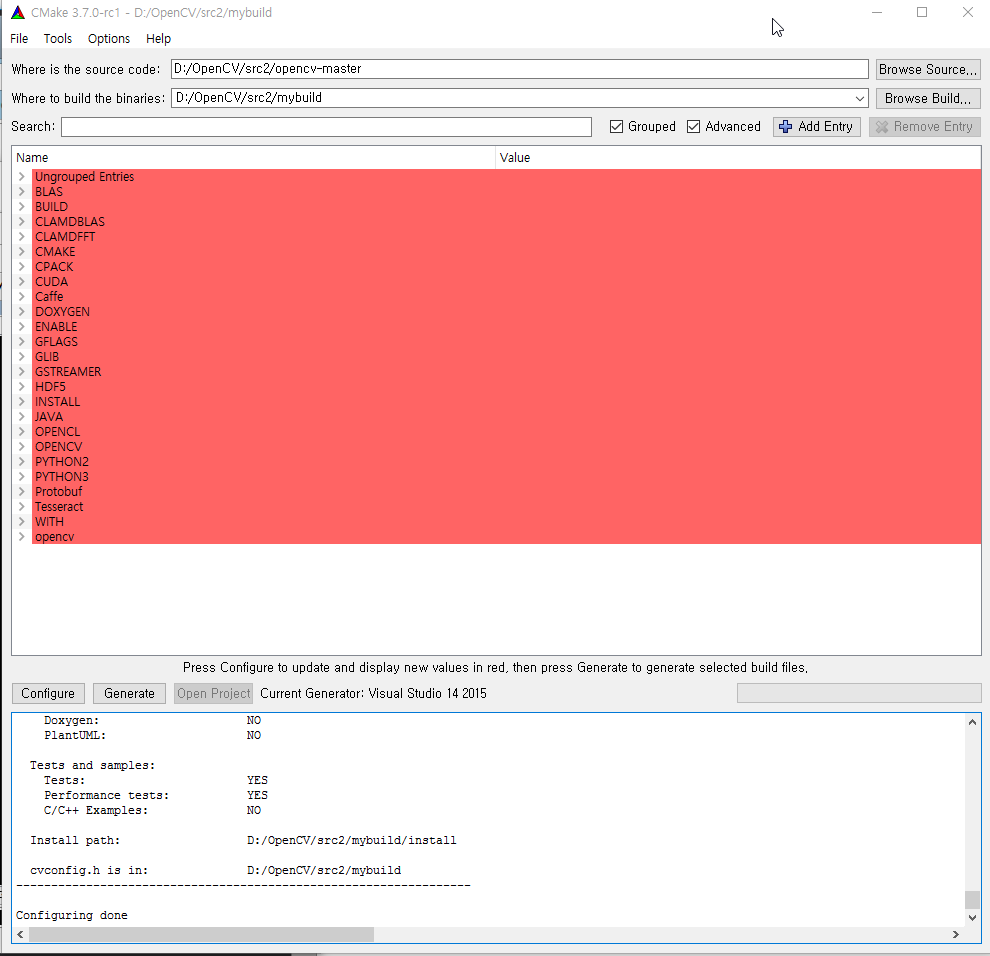
Opencv\_contib-master.zip 파일에 압축되어 있는 modules 폴더에 있는 파일들을 방금 압축을 푼 opncv-master의 modules 폴더에 압출을 푼다.



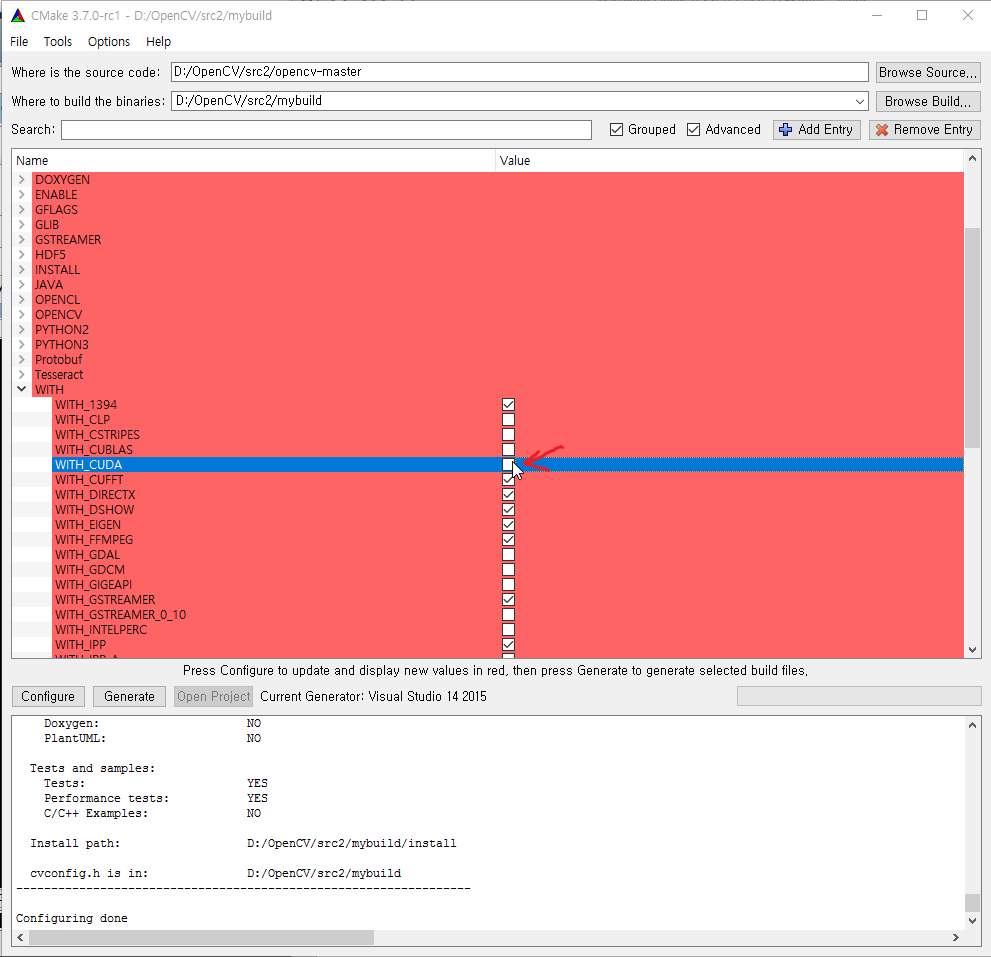
CMake를 설치하고 실행 시킨다. 소스가 있는 폴더를 지정하고 build 결과 파일을 저장 할 위치를 지정하고 Configure 버튼을 클릭 한다. VS2015 버전으로 만들 32 bit 버전으로 만들기 위해 Visual Studio 14 2015를 선택 한다.



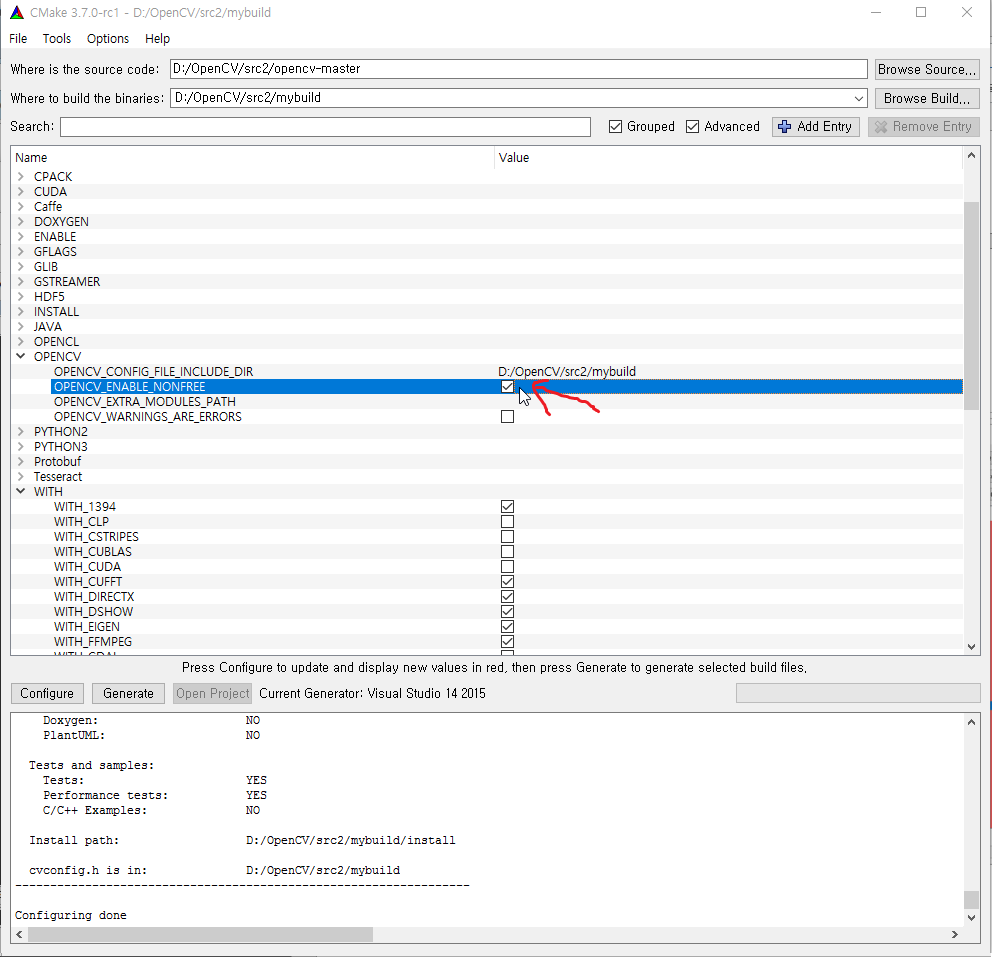
Grouped과 Advanced 체크 박스를 체크하면 보기 편하다.



CUDA를 사용하지 않을 것이라면 CUDA를 제외 시켜 주고 Configure를 다시 눌러 준다.



OPENCV의 OPENCV\_ENABLE\_ NONFREE를 해줘야 Contrib를 포함 시킨다. 라이브러리의 연관성 때문에 차례대로 해줘야 한다.



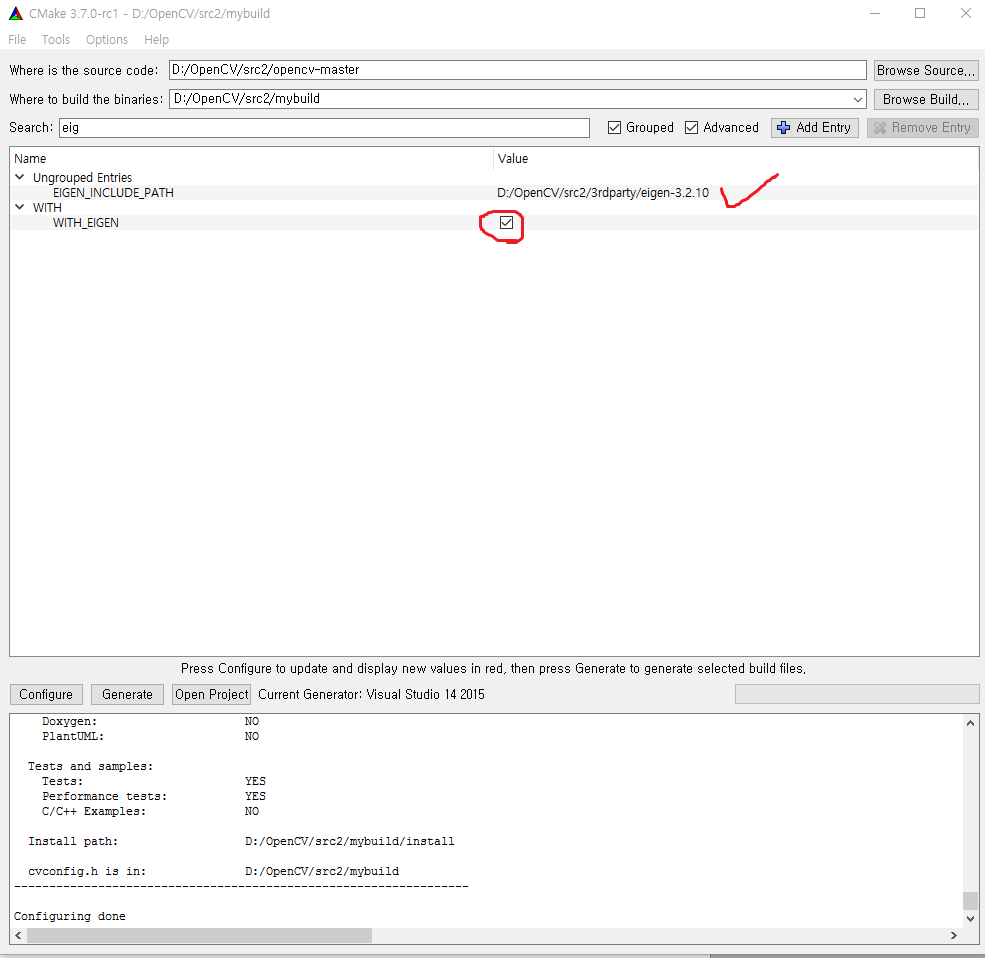
<http://eigen.tuxfamily.org/index.php?title=Main_Page> 방문해서 eigen 다운 받는다.

다른 3rdParty들도 컴파일 하는 방법이 모두 동일하므로 eigen만 다룬다.

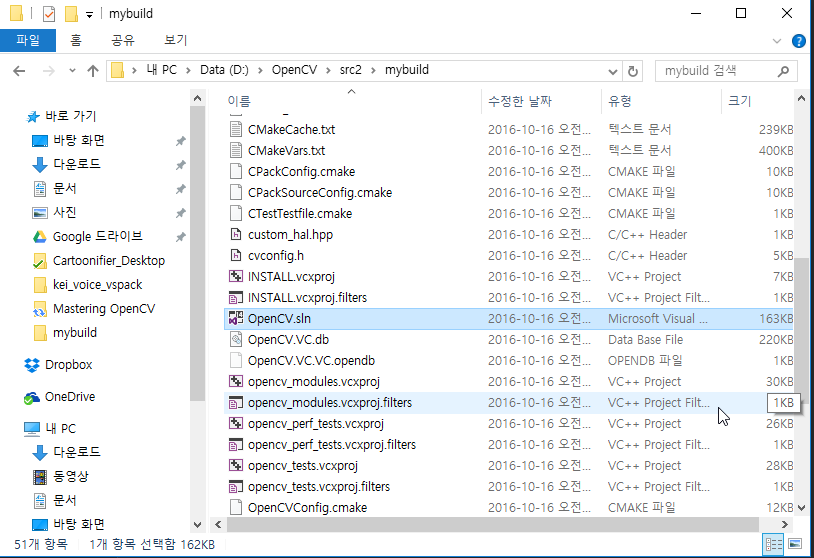


Eigen 을 사용 할 것을 체크하고 소스파일 위치를 지정한 후 Configure를 누르면 eigeh도 포함 시킬 수 있다. 이런 방법으로 3rdparty를 포함 시킨다.

Generate 버튼을 누르면 지정된 위치에 binary 파일들이 빌드 된다.



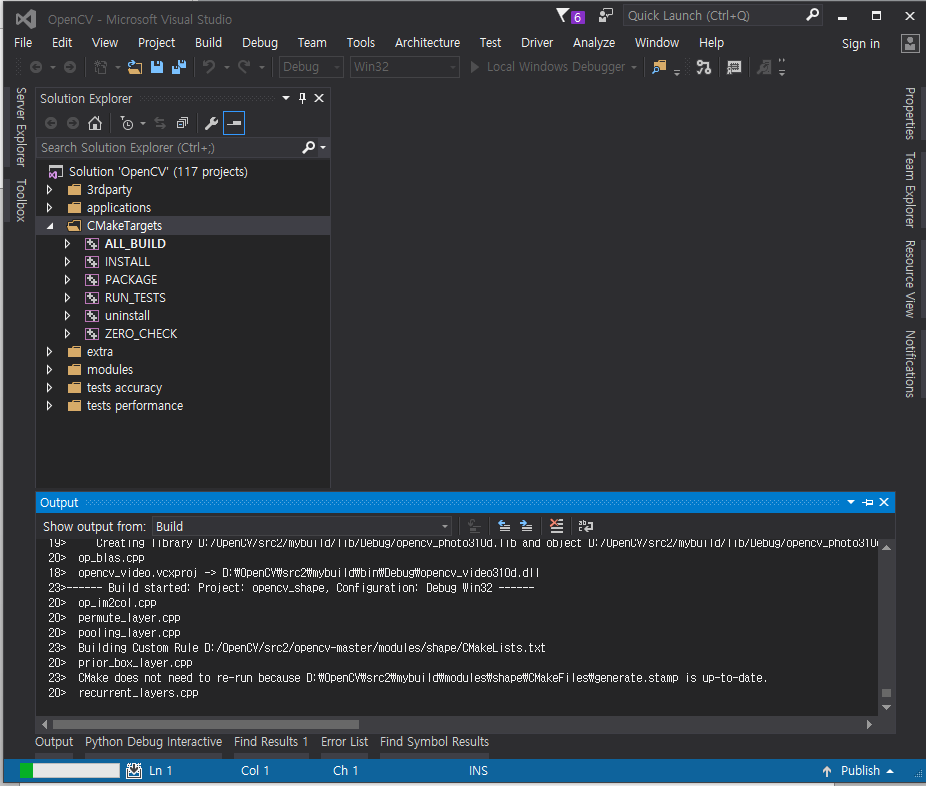
솔루션 파일을 선택하여 프로젝트를 비쥬얼 스튜디오로 오픈한다.



Build solution으로 빌드를 한다. Debug Build가 완료되었다면 Release도 Build 해준다.

Sample 파일까지 포함 시키면 컴파일 시간만 40분 이상 소요 된다.

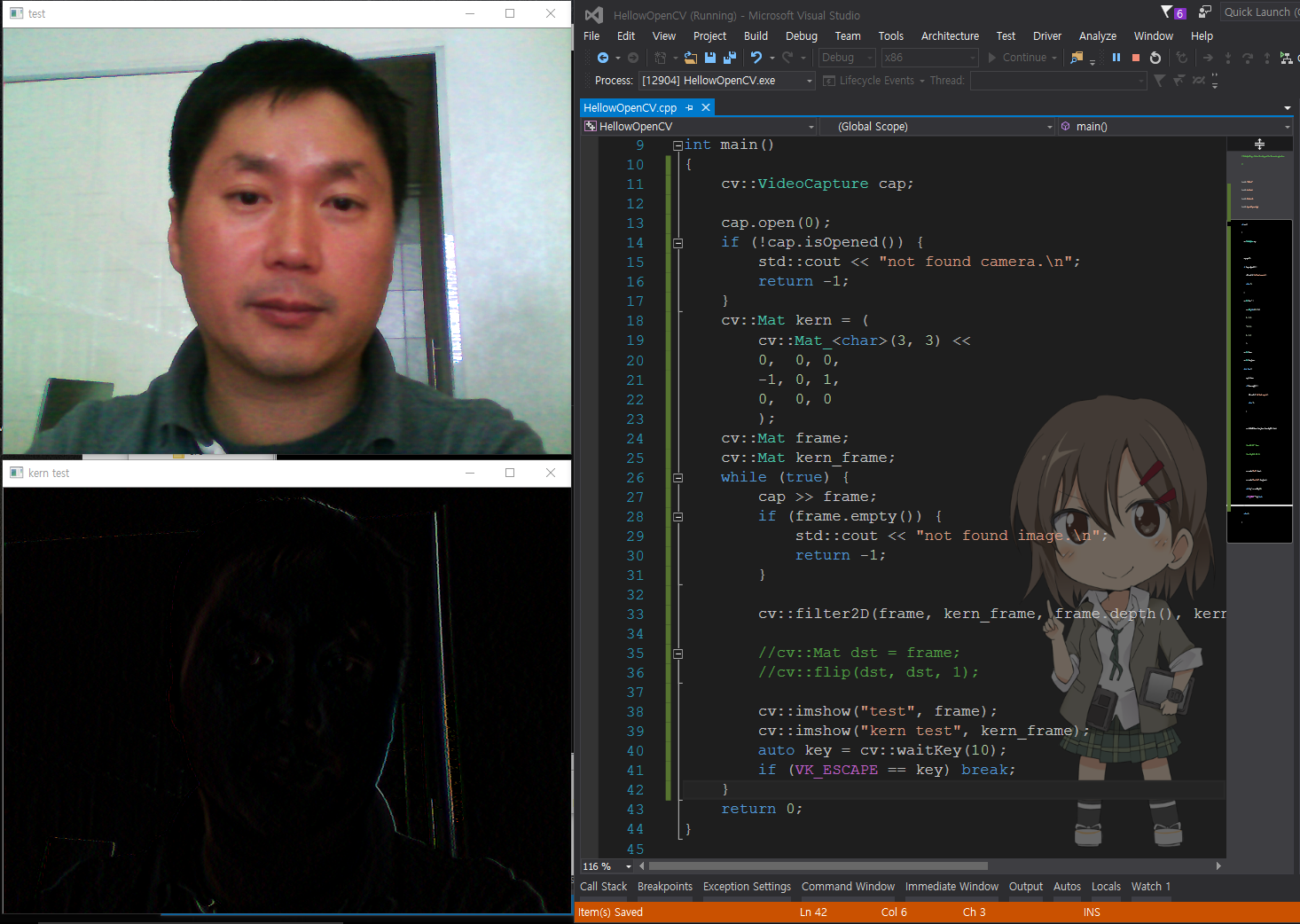
INSTALL 프로젝트를 컴파일 하면 Install 폴더가 만들어지고 개발 환경을 만들어 준다. Install 폴더에 Demo 폴더를 또는 프로젝트 폴더를 만들고 작업하면 작업이 용이하다.



HellowOpenCV demo 작성.

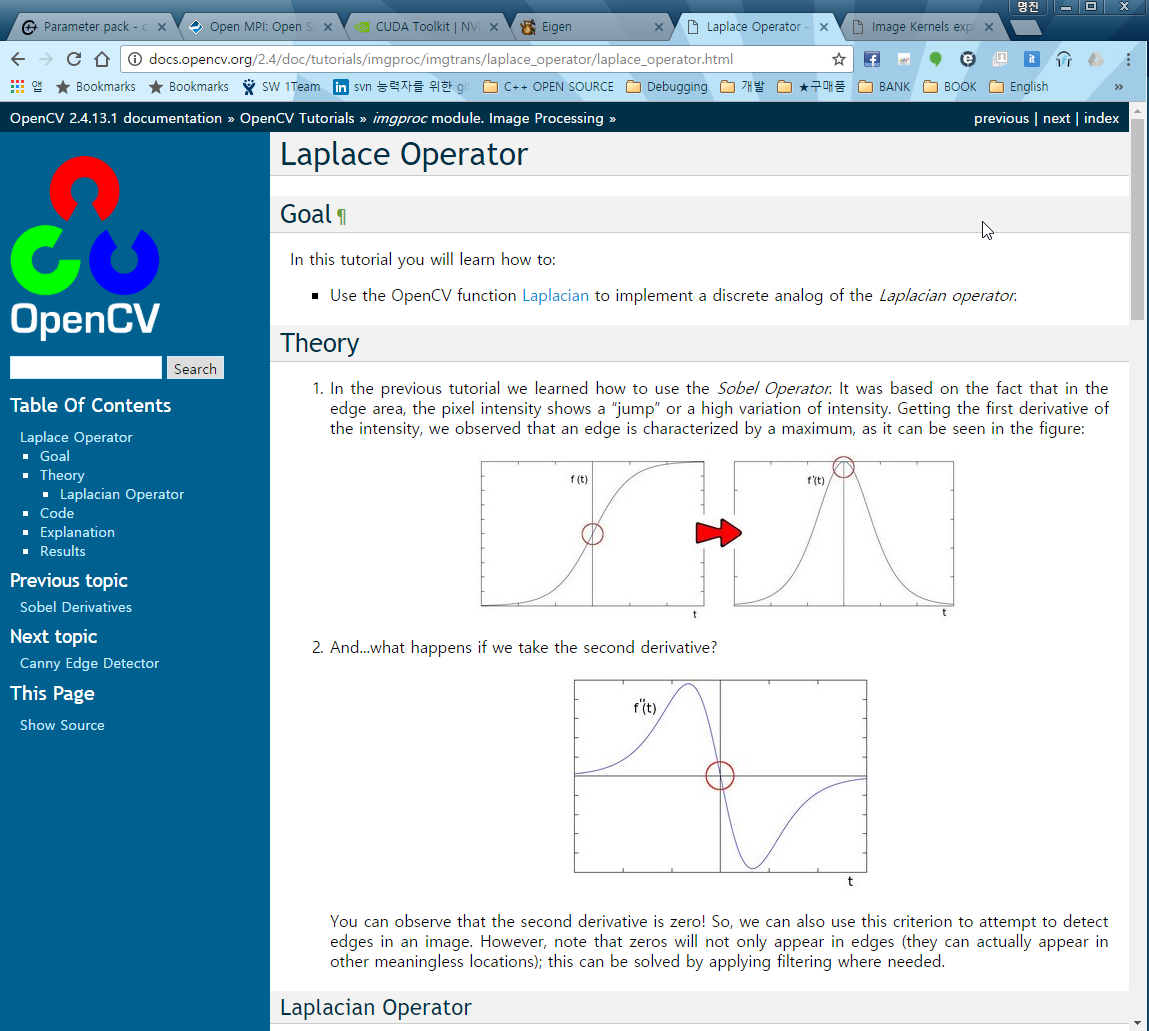
웹캠을 통해 영상을 획득한 후 filter2D 함수를 이용하여 마스킹을 수행한 영상을 얻어낸다.

해당 소스코드는 외우고 숙달 시켜야 한다.



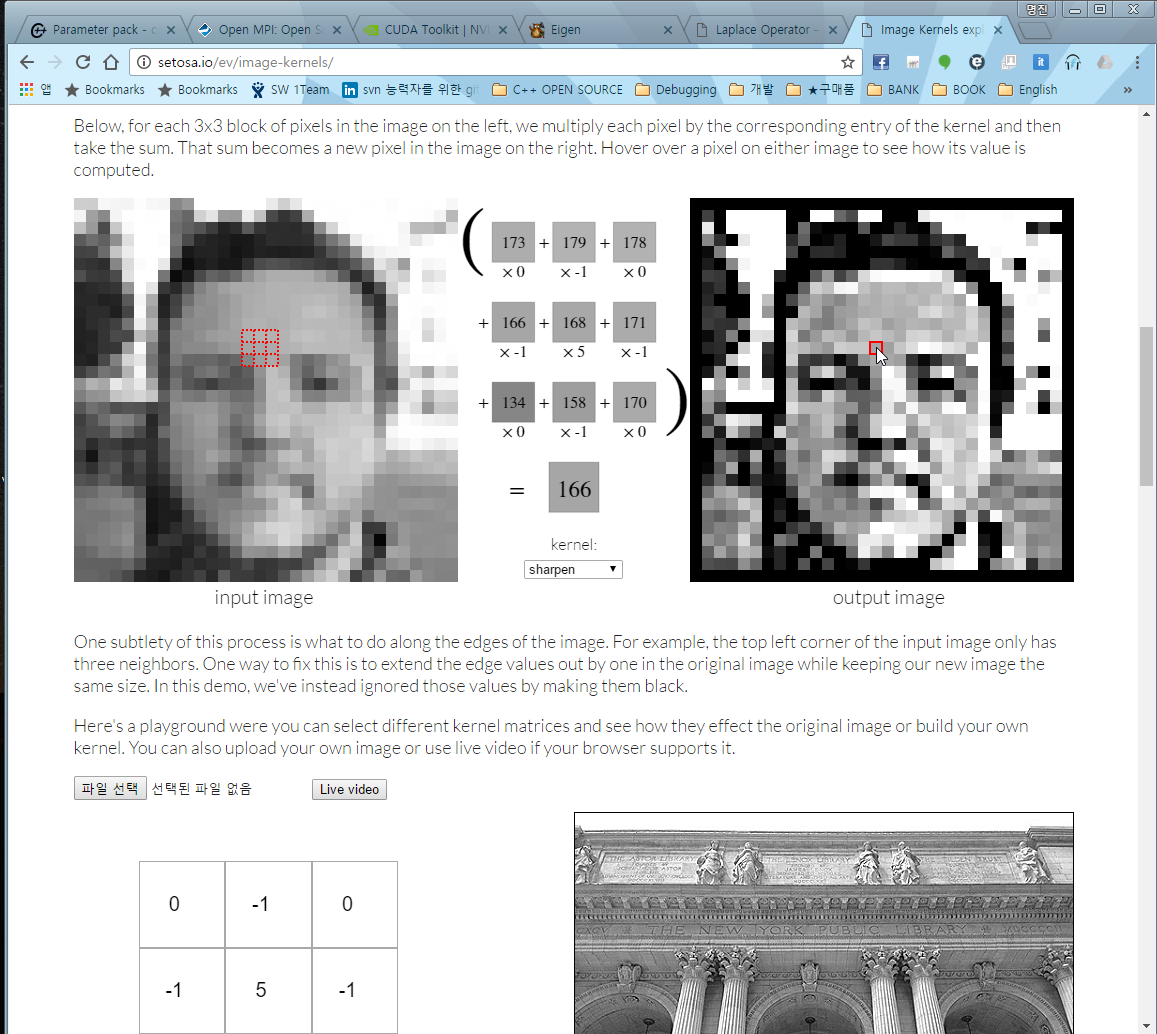
<http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/laplace_operator/laplace_operator.html>

라플라시안에 대한 개념 확인



Kernels에 대한 개념 확인

<http://setosa.io/ev/image-kernels/>



2016년 10월 23일(일요일)