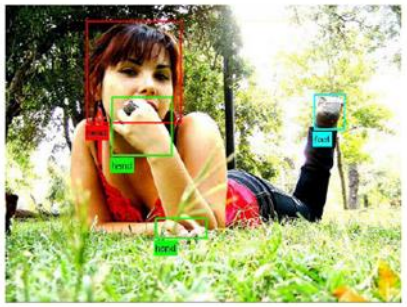
Open CV 개요 및 실습

Open CV overview and practice

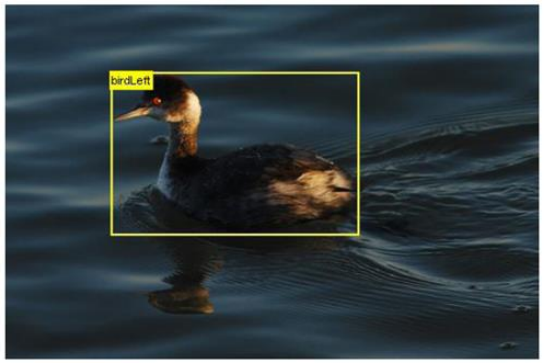
Open CV

Open Source Computer Vision 개요

- 실시간 컴퓨터 비전(Compute Vision) 목적으로 만들어진 프로그래밍 라이브러리
- Windows, Linux, OS X(Mac OS), iOS, Android 플랫폼 지원
- 첫 개발 시 인텔에서 개발 주도 (현재 X)
- 미래 개발을 위해 크라우드 펀딩으로 운영되는 비영리 재단으로 변경 (2012년~)
- 오픈소스 라이선스 채택 (BSD 라이선스)
  > 비상업적/상업적 이용 가능
- (현재 2018년) 2.x 버전과 3.x 버전을 동시에 사용



Computer Vision, 한빛아카데미



Open CV

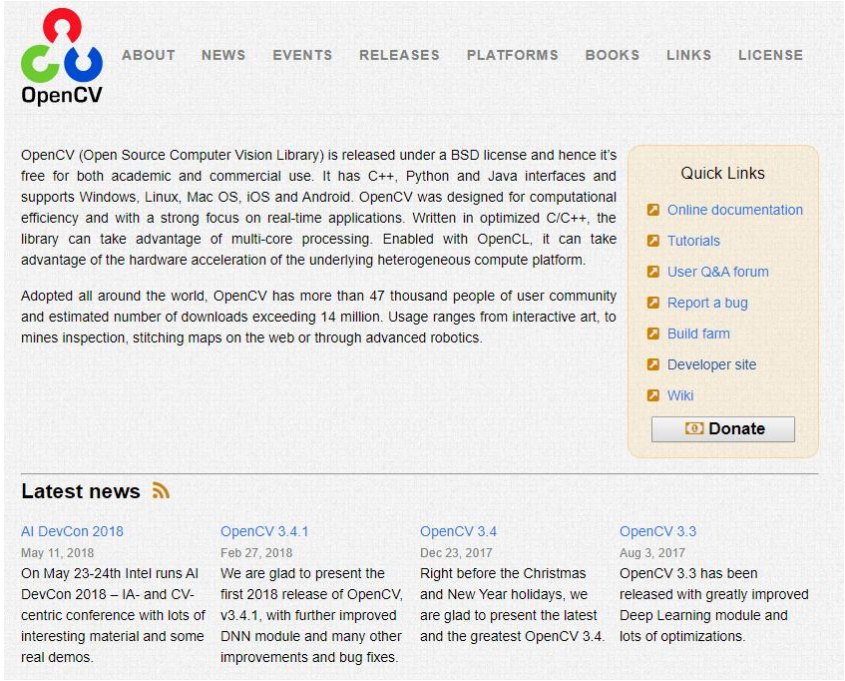
Homepage / Github

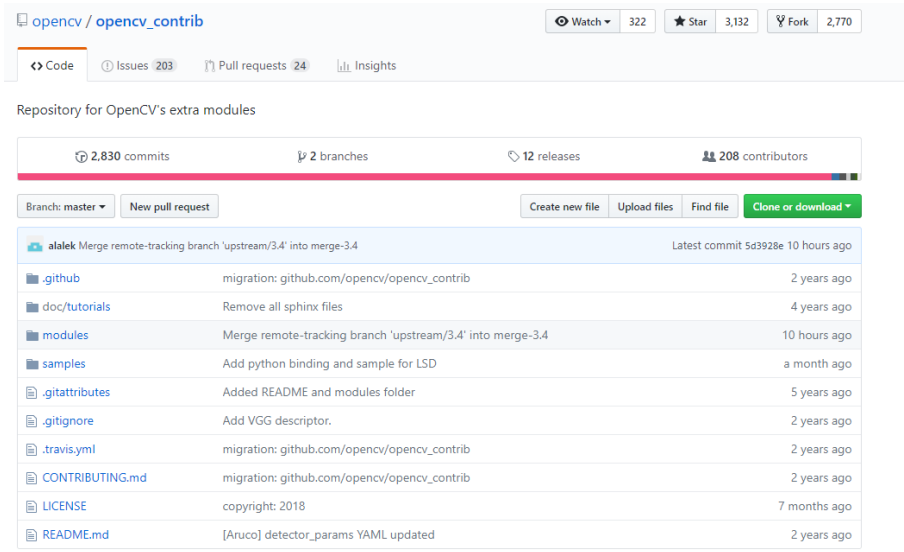
- 공식 홈페이지 : https://opencv.org
- 공식 Github :

  > https://github.com/Itseez/opencv
- **Extra Contribute:**

  > https://github.com/Itseez/opencv_contrib

Extra Contribute
공식적이지는 않지만 최신 기술이 반영된 모듈들이 제공되고 있어 다양한 기능을 더 추가할 수 있음





Open CV

주요 릴리즈 버전

날짜	버전	내용
2006/10	1.0	• 공식 첫 버전 릴리즈 • Python 모듈 포함
2009/10	2.0	• MinGW 사용 MinGW(mingw32) 윈도우 API를 구현할 수 있는 헤더파일을 가지고 있어 윈도우 기반 프로그램을 만들기 편리
2010/04	2.1	• OpenCV 모든 병렬처리 구조가 OpenMP에서 Intel TBB로 전환 • Mac OS X 공식 지원
2012/07	2.4.2	• Android / iOS 공식 지원
2013/03	2.4.4	• 데스크톱 기반 Java 공식 지원 • Android / iOS 버전 업그레이드
2014/04	2.4.9	• VTK 기반 3D 시각화 모듈 추가
2015/06	3.0	• 2 버전과는 별도로 3으로 릴리즈

Open CV

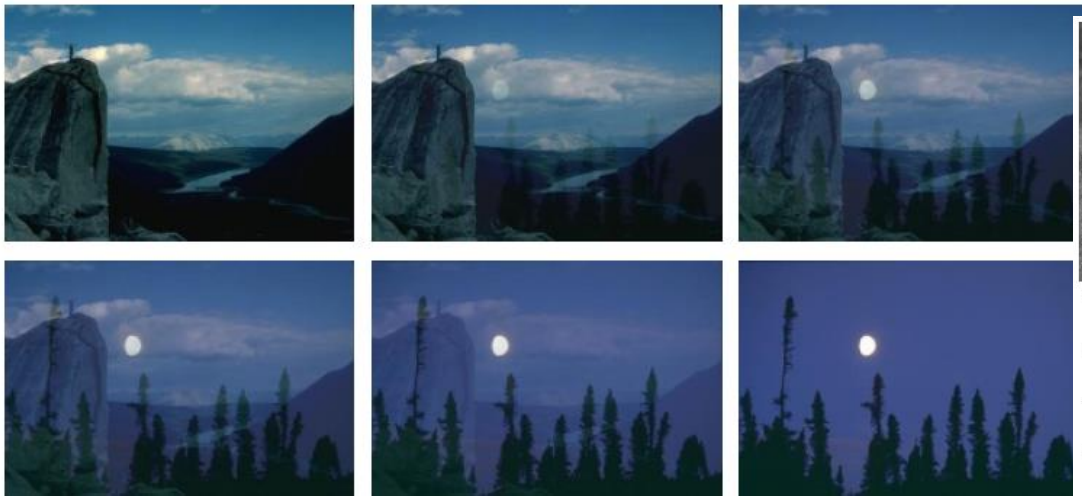
버전별 특징

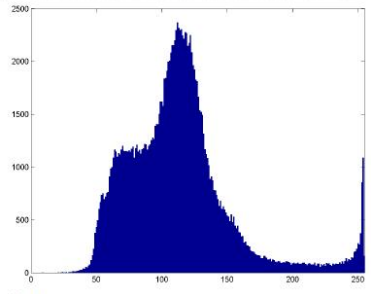
1.0	2.0	2.1
• C언어 기반 API • 구조체 기반 데이터 구조 사용 • 8비트 PNG, JPEG2000 입출력 지원	• C++언어 기반 API • 클래스 기반 데이터 구조 도입 • Cmake를 이용하여 라이브러리 컴파일 후 사용 가능 • 스테레오 카메라 지원	• OpenMP에서 인텔 TBB로 병렬 처리 루프 변경 • Mac OS X에서 빌드 가능
2.2	**2.3**	**2.4.3**
• 안드로이드 지원 • 16비트 LZW 압축 지원(TIFF 영상) • GPU 처리 지원	• 파노라마 지원 • GPU 모듈 CUDA 4.0 지원	• TBB 설치 없이 기본적인 병렬 처리 지원 • OpenCL 컴퓨터 비전 알고리즘인 ocl 모듈 도입 • 안드로이드 카메라 지원 개선
2.4.7	**3.0**	**2.X 와 3.x 가 동시 지원되고 있지만, 새로 만드는 프로젝트일 경우 3.0을 추천**
• 비디오 고해상도 모듈 도입 • GPU 모듈 CUDA 5.0 지원 • 안드로이드 NDK-r9지원 • 안드로이드 4.3 지원	• 기존 C++ API 대폭 개선 • 모바일 CUDA 지원 • IPP, FastCV 같은 저수준 API 지원	

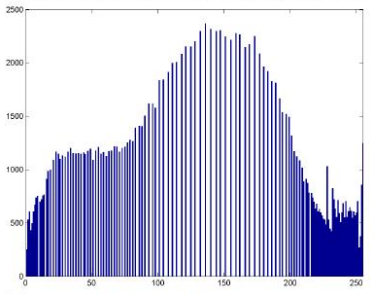
Open CV

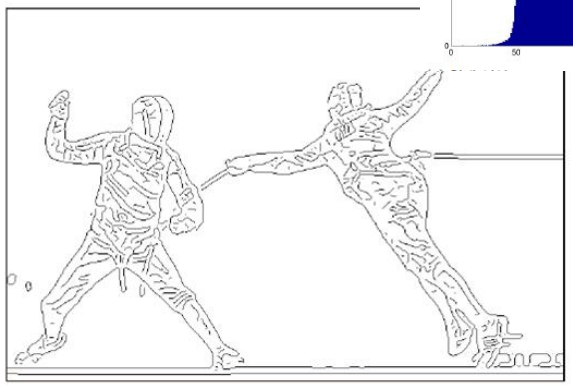
활용 범위

- 이미치 프로세싱 : 영상 추출, 필터, 히스토그램 …





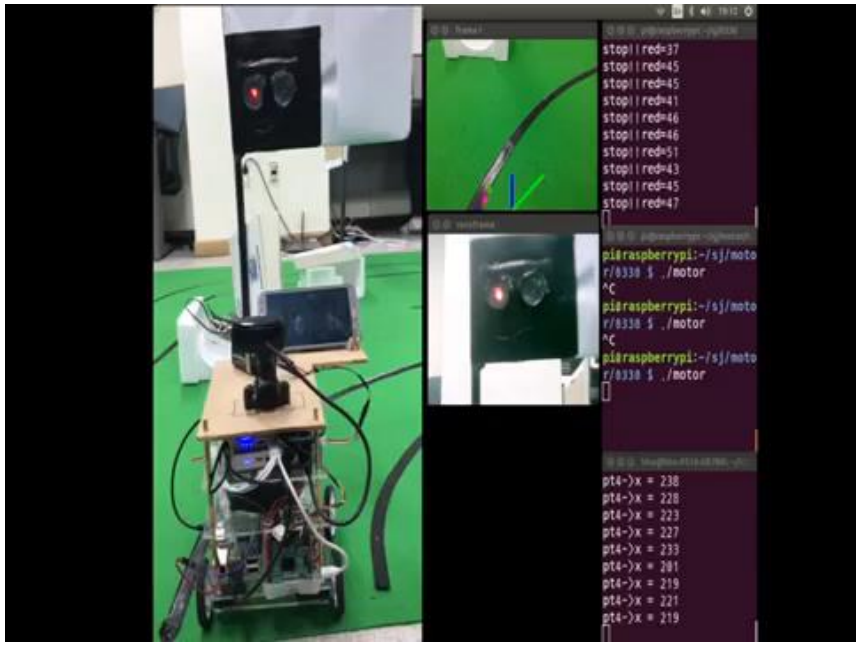




Open CV

활용 범위

- 로봇, 비디오 : 트래킹, 특징 추출

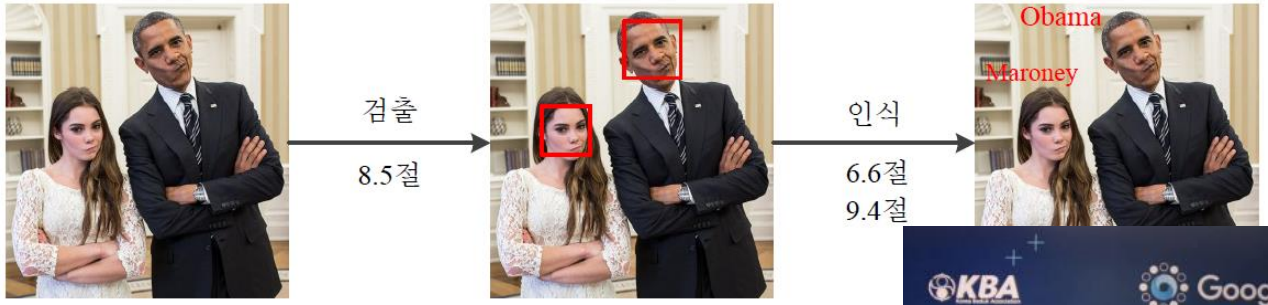


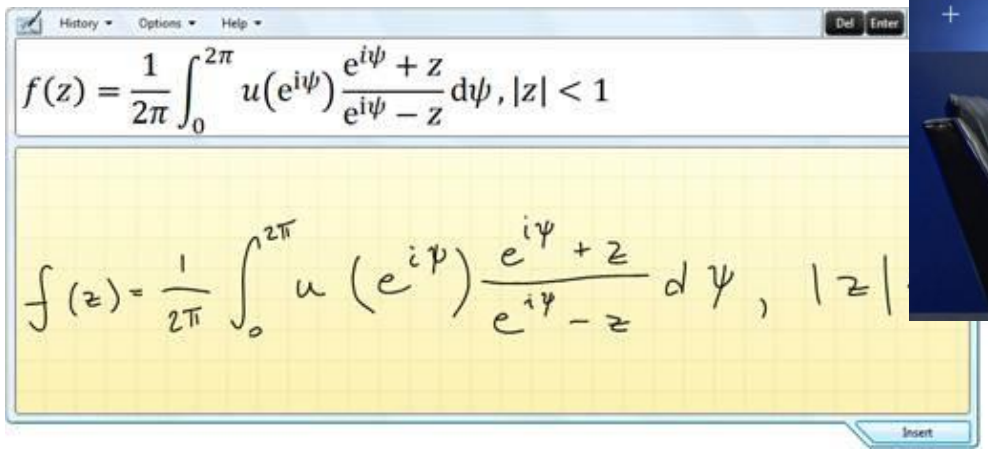
https://www.youtube.com/watch?v=h8BDMiNxmvA&feature=youtu.be
https://www.youtube.com/watch?v=LYpug_bHle8

Open CV

활용 범위

- 인공지능 : 신경망, 딥러닝, 손글씨 인식, …

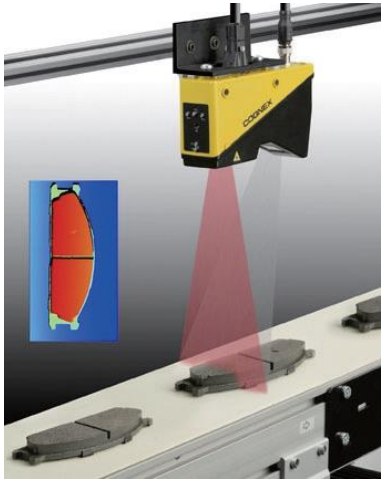


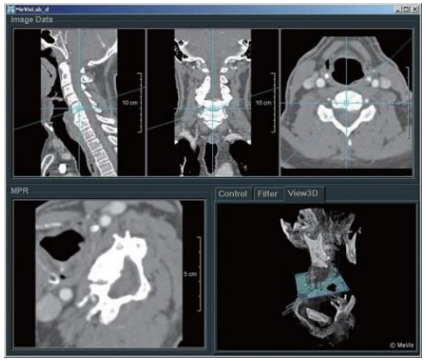


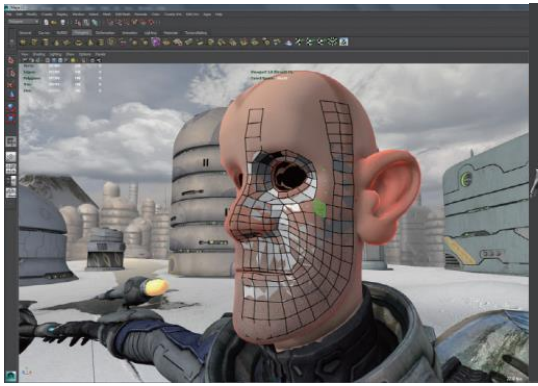
Open CV

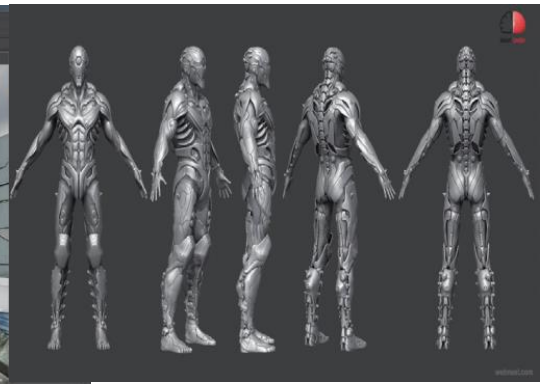
활용 범위

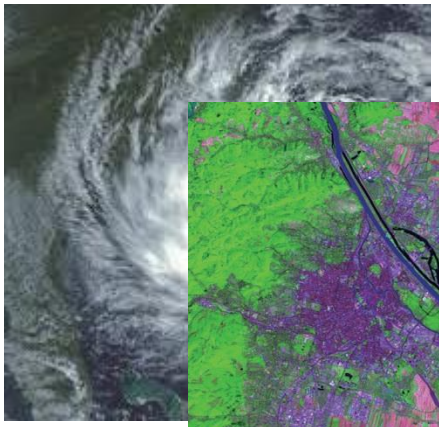
- 의료, 방송통신, 자동화, 게임, 기상 및 지질 탐사

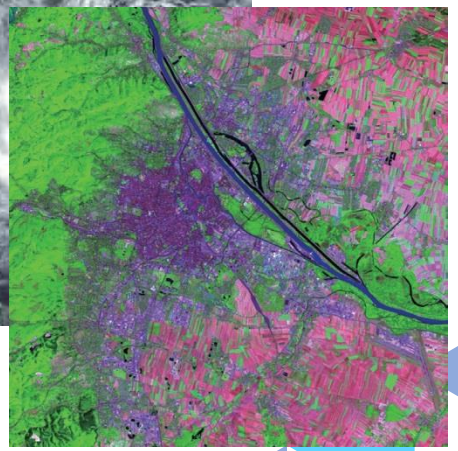






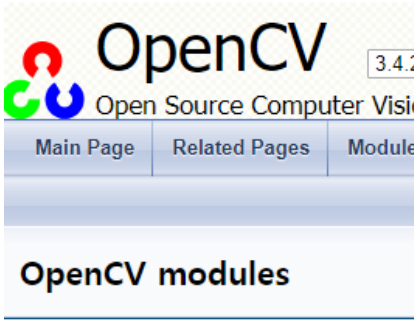






Open CV

Open CV 참고 문서 (https://docs.opencv.org/3.4.2/)



- Main modules:
  - core. Core functionality
  - imgproc. Image processing
  - imgcodecs. Image file reading and writing
  - videoio. Video I/O
  - highgui. High-level GUI
  - video. Video Analysis
  - calib3d. Camera Calibration and 3D Reconstruction
  - features2d. 2D Features Framework
  - objdetect. Object Detection
  - dnn. Deep Neural Network module
  - ml. Machine Learning
  - flann. Clustering and Search in Multi-Dimensional Spaces
  - photo. Computational Photography
  - stitching. Images stitching
  - cudaarithm. Operations on Matrices
  - cudabgsegm. Background Segmentation
  - cudacodec. Video Encoding/Decoding
  - cudafeatures2d. Feature Detection and Description
  - cudafilters. Image Filtering
  - cudaimgproc. Image Processing
  - cudalegacy. Legacy support
  - cudaobjdetect. Object Detection
  - cudaoptflow. Optical Flow
  - cudastereo. Stereo Correspondence
  - cudawarping. Image Warping
  - cudev. Device layer
  - shape. Shape Distance and Matching
  - superres. Super Resolution
  - videostab. Video Stabilization
  - viz. 3D Visualizer
- Extra modules:
  - aruco. ArUco Marker Detection
  - bgsegm. Improved Background-Foreground Segmentation Methods
  - bioinspired. Biologically inspired vision models and derivated tools
  - ccalib. Custom Calibration Pattern for 3D reconstruction
  - cnn_3dobj. 3D object recognition and pose estimation API
  - cvv. GUI for Interactive Visual Debugging of Computer Vision Programs
  - datasets. Framework for working with different datasets
  - dnn_objdetect. DNN used for object detection
  - dpm. Deformable Part-based Models
  - face. Face Analysis
  - freetype. Drawing UTF-8 strings with freetype/harfbuzz
  - fuzzy. Image processing based on fuzzy mathematics
  - hdf. Hierarchical Data Format I/O routines
  - hfs. Hierarchical Feature Selection for Efficient Image Segmentation
  - img_hash. The module brings implementations of different image hashing algorithms.
  - line_descriptor. Binary descriptors for lines extracted from an image
  - matlab. MATLAB Bridge
  - optflow. Optical Flow Algorithms
  - ovis. OGRE 3D Visualiser
  - phase_unwrapping. Phase Unwrapping API
  - plot. Plot function for Mat data
  - reg. Image Registration
  - rgbd. RGB-Depth Processing
  - saliency. Saliency API
  - sfm. Structure From Motion
  - stereo. Stereo Correspondance Algorithms
  - structured_light. Structured Light API
  - surface_matching. Surface Matching
  - text. Scene Text Detection and Recognition
  - tracking. Tracking API
  - xfeatures2d. Extra 2D Features Framework
  - ximgproc. Extended Image Processing
  - xobjdetect. Extended object detection
  - xphoto. Additional photo processing algorithms

Open CV

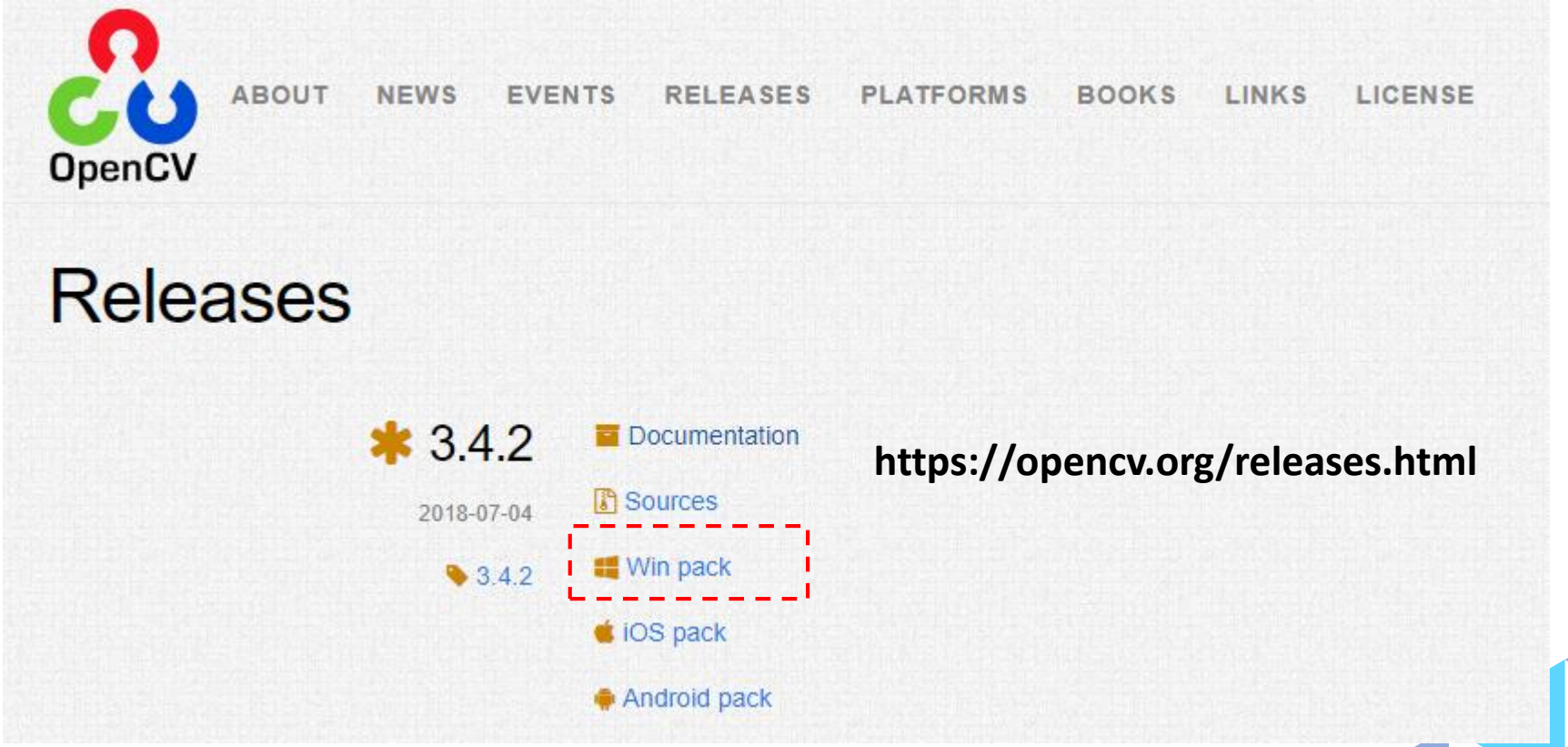
OpenCV JavaScript 버전 (OpenCV.js)

https://www.youtube.com/watch?v=AlOKQFKCCsY
http://soswow.github.io/Various-JS-and-Python/JS/OpenCV.js-receipt/

Open CV

설치 방법

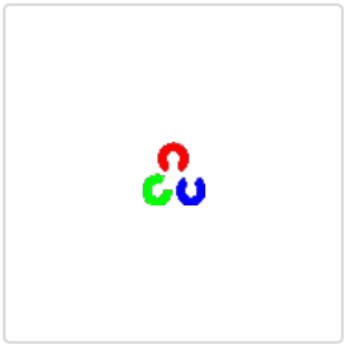
- 라이브러리 형태이므로 다운로드 받은 후 API 형태로 원하는 프로그램 언어를 통해 사용



Open CV

vc14(Visual Studio 2015)
vc15(Visual Studio 2017)

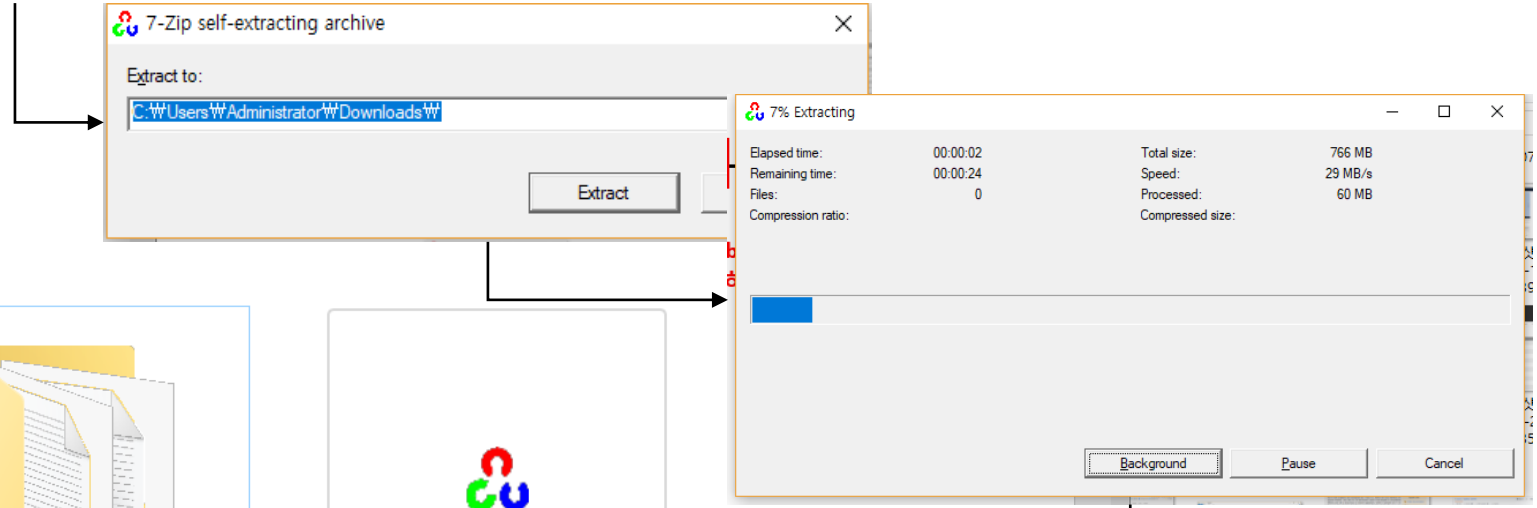
- 다운로드된 파일



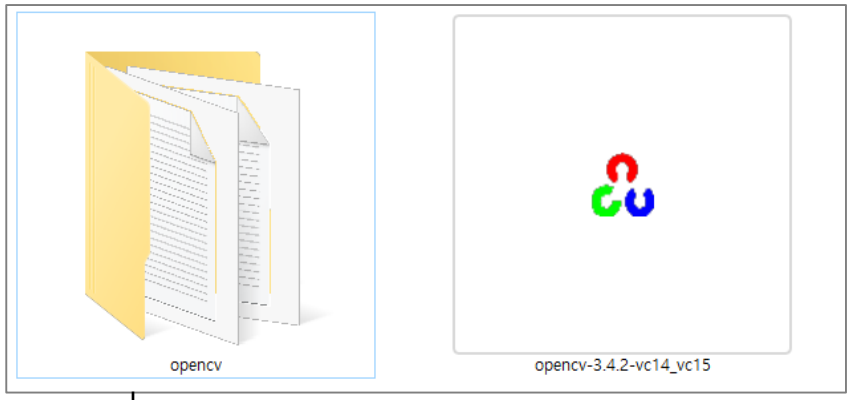
Opencv-3.4.2-vc14_vc15.exe

버전 빌드된 버전

3.X는 Windows 64bit 빌드 버전만 존재
만약 32bit를 사용하고 있다면 별도로 빌드해야 함

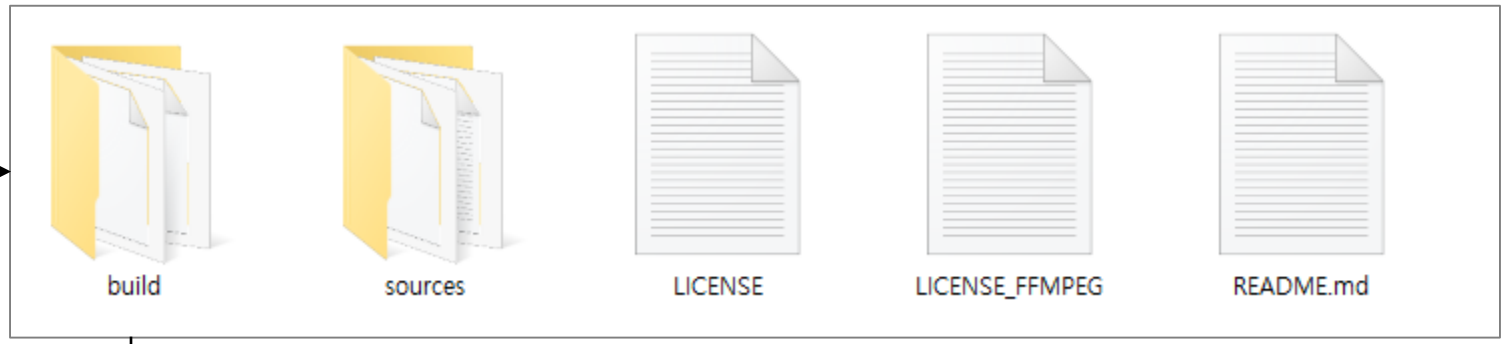


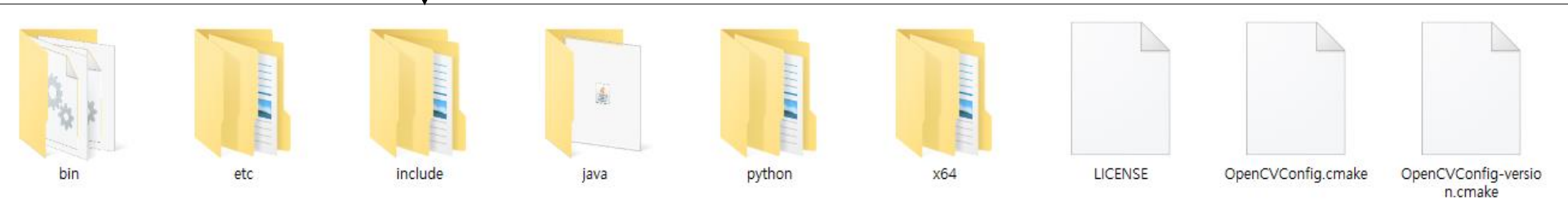
Open CV



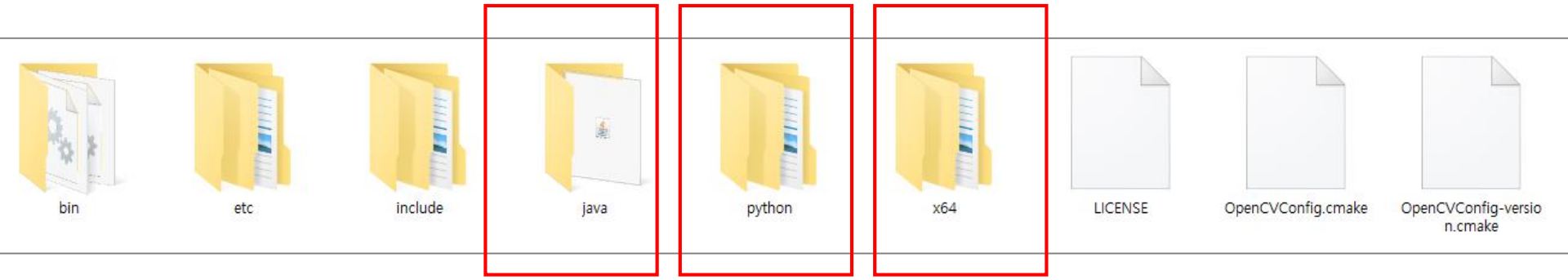
미리 빌드된 파일
Source 코드 모두
포함되어 있음

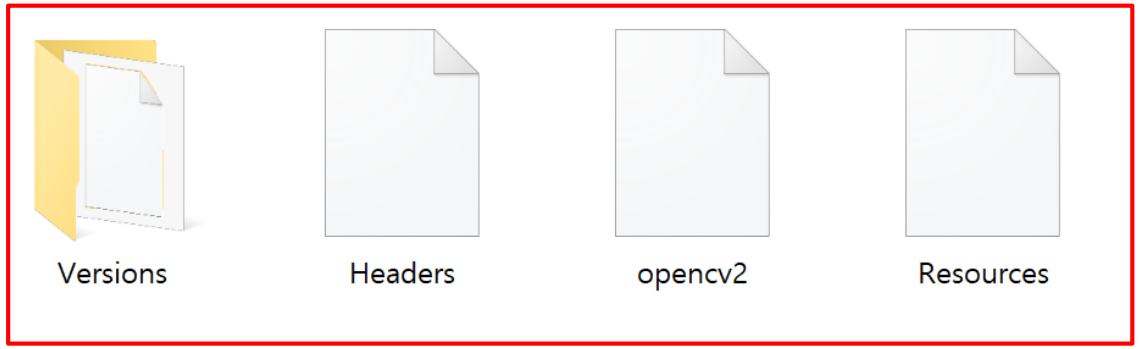
해당 폴더를 원하는 위치로 이동
본 강의에서는 C:\opencv\...





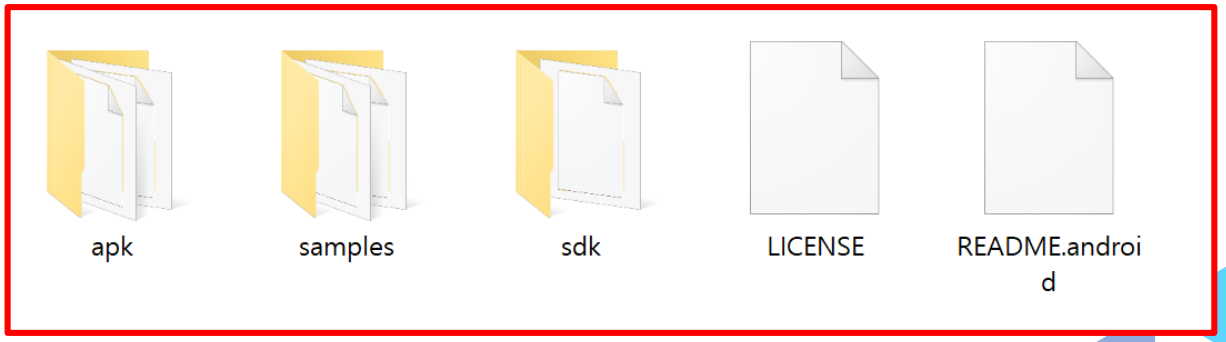
Open CV





iOS

Android

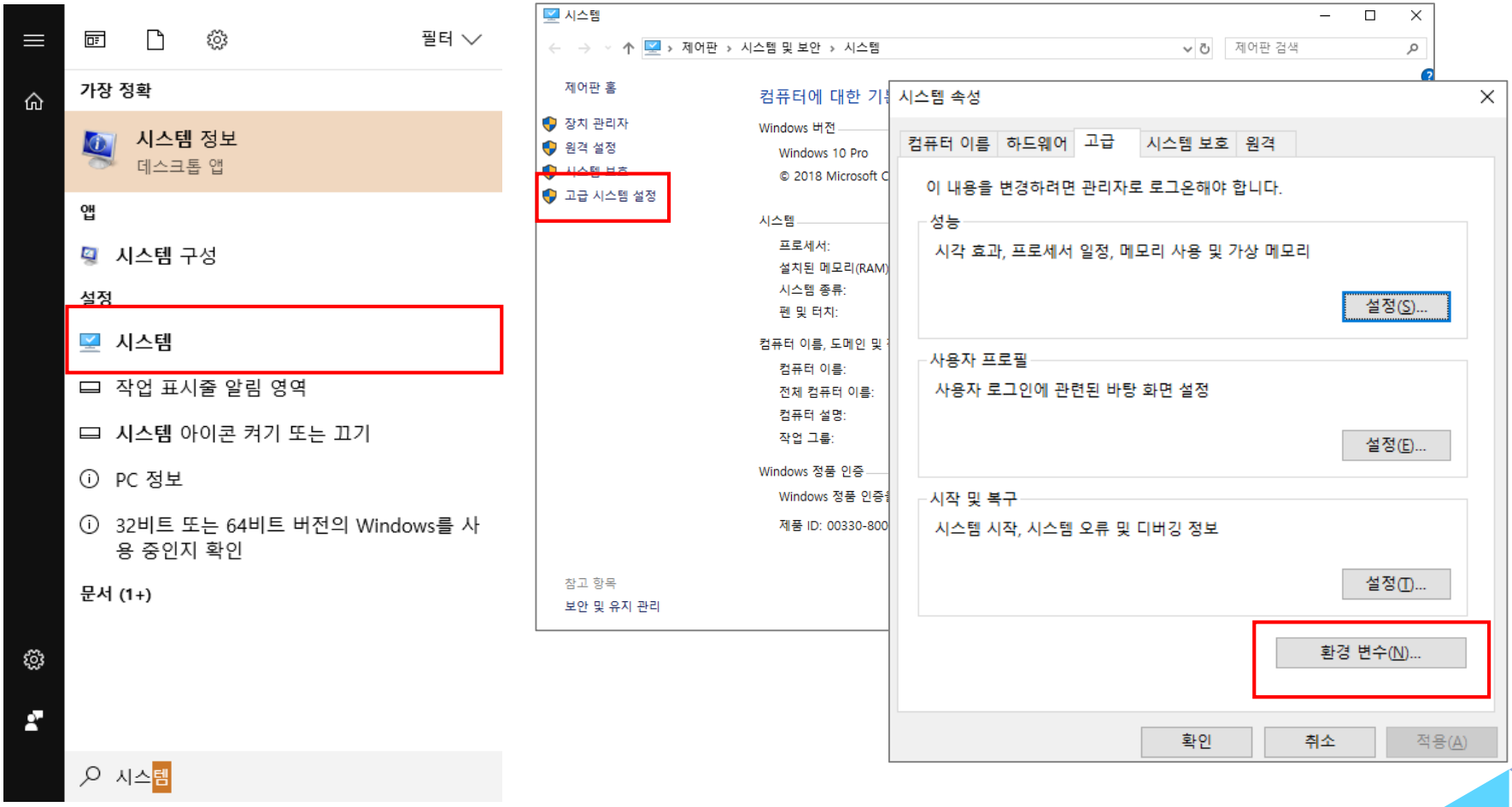


Open CV

Path 환경 변수 경로 추가

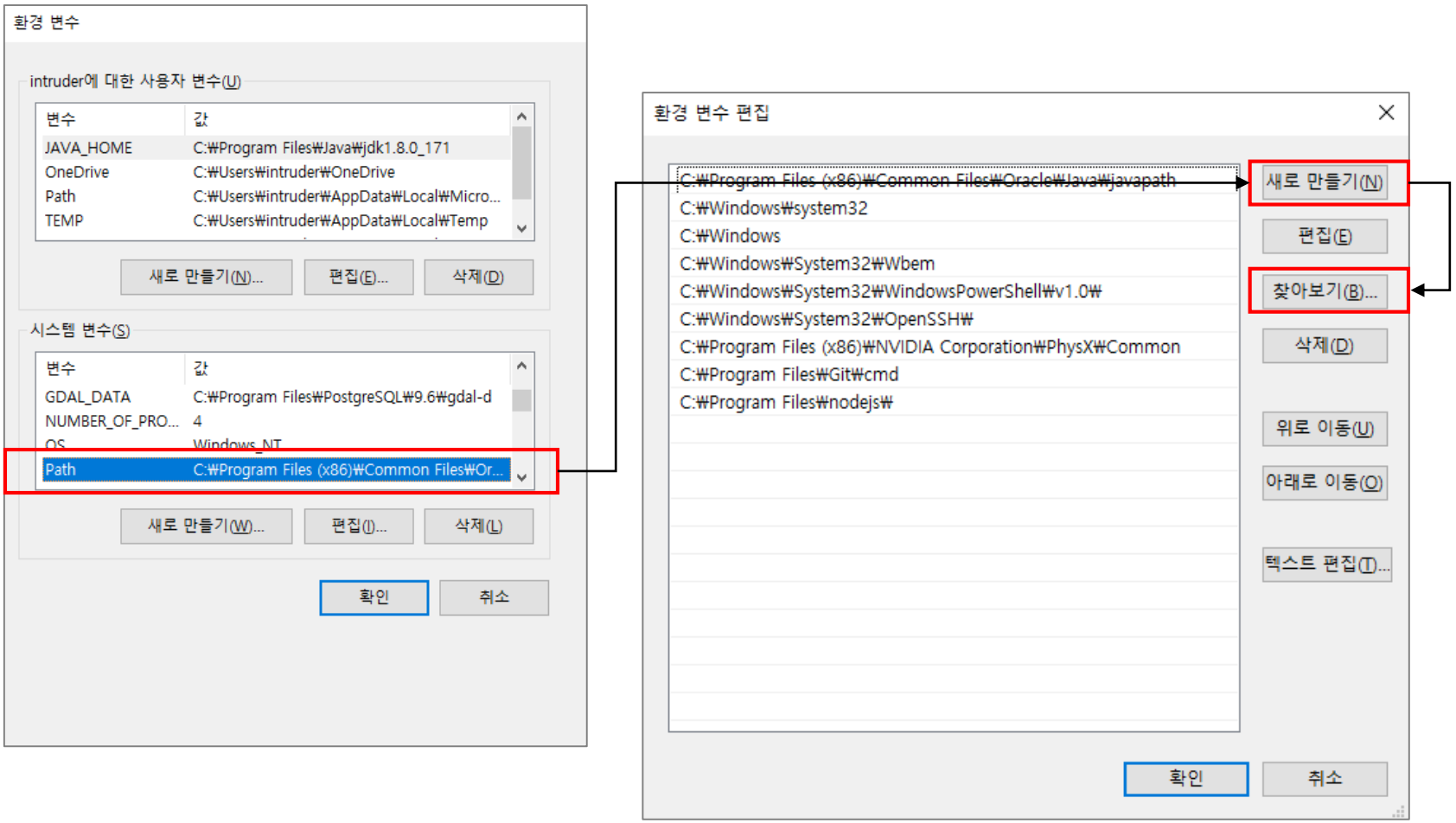
- 오픈 소스로 제공되는 API들은 대부분 동적 연결 라이브러리 파일로 제공
- 동적 연결 라이브러리 역할?
  > 함수 호출 정보만 포함하여 목적 코드를 실행 시간에 호출하여 실행하므로 메모리 적약 / 디스크 공간 줄임
- 동적 연결 라이브러리를 사용하려면?
  > DLL 파일이 프로젝트의 실행 디렉토리로 복사
  > 환경변수에 이미 추가된 Path 디렉토리에 복사
  > 환경변수 Path에 DLL 파일 디렉토리 경로 추가

Path 환경 변수 경로 추가

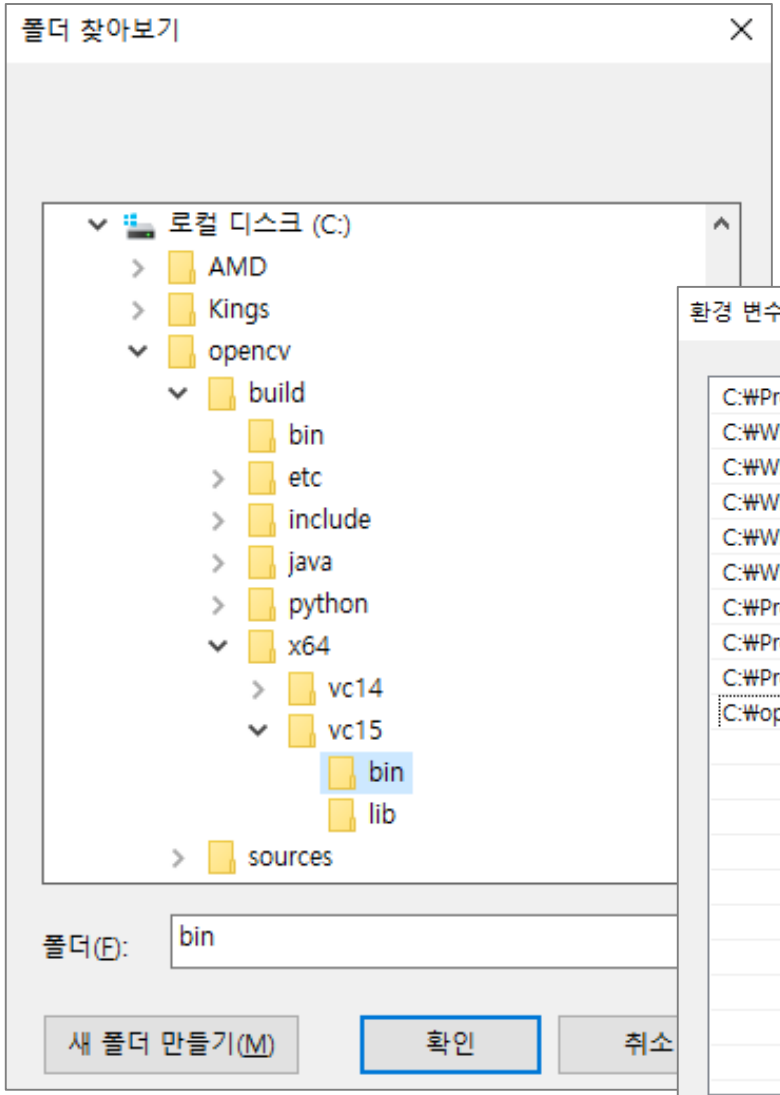


만약 나타나지 않을 경우 “내 컴퓨터” -> 빈공간 오른쪽 클릭 -> 속성

Open CV

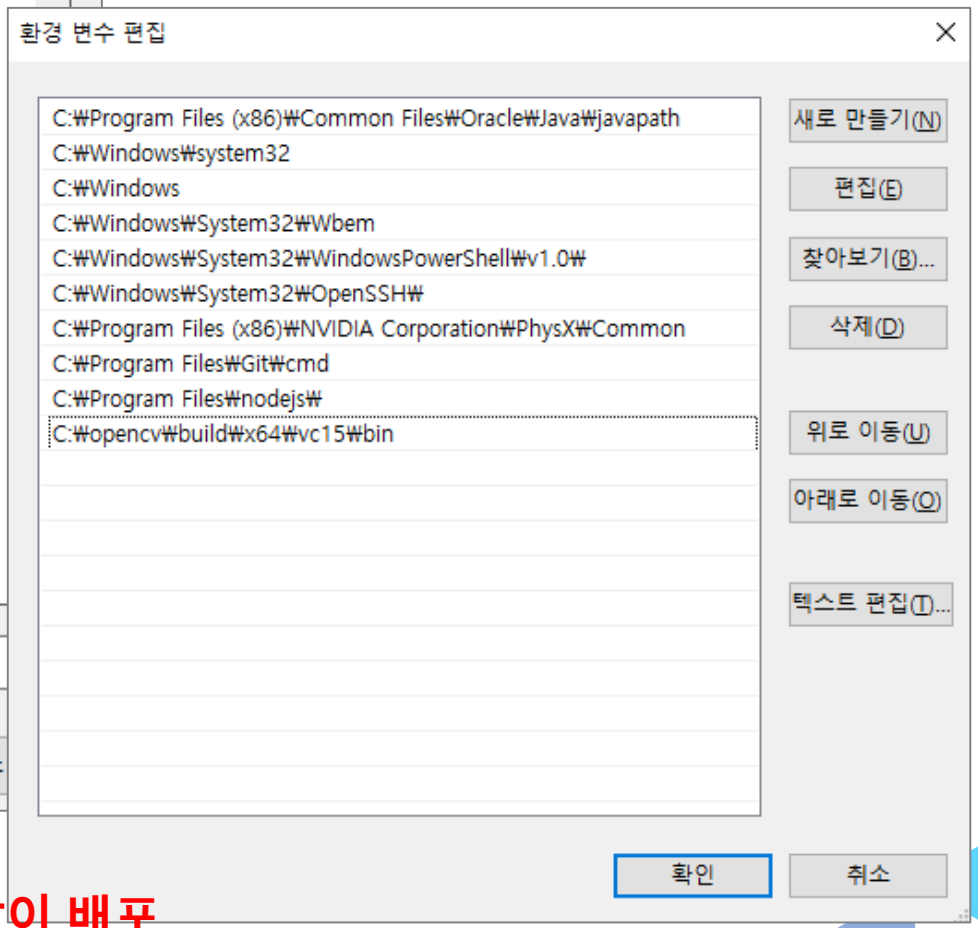

환경 변수
intruder에 대한 사용자 변수(U)
변수
값
JAVA_HOME
C:₩Program Files₩Java₩jdk1.8.0_171
OneDrive
C:₩Users₩intruder₩OneDrive
Path
C:₩Users₩intruder₩AppData₩Local₩Micro...
TEMP
C:₩Users₩intruder₩AppData₩Local₩Temp
새로 만들기(N)...
편집(E)...
삭제(D)
시스템 변수(S)
변수
값
GDAL_DATA
C:₩Program Files₩PostgreSQL₩9.6₩gdal-d
NUMBER_OF_PRO...
4
OS
Windows_NT
Path
C:₩Program Files (x86)₩Common Files₩Or...
새로 만들기(W)...
편집(I)...
삭제(L)
확인
취소
환경 변수 편집
C:₩Program Files (x86)₩Common Files₩Oracle₩Java₩javapath
C:₩Windows₩system32
C:₩Windows
C:₩Windows₩System32₩Wbem
C:₩Windows₩System32₩WindowsPowerShell₩v1.0₩
C:₩Windows₩System32₩OpenSSH₩
C:₩Program Files (x86)₩NVIDIA Corporation₩PhysX₩Common
C:₩Program Files₩Git₩cmd
C:₩Program Files₩nodejs₩
새로 만들기(N)
편집(E)
찾아보기(B)...
삭제(D)
위로 이동(U)
아래로 이동(O)
텍스트 편집(T)...
확인
취소

Open CV



C:\opencv\build\x64\vc15\bin

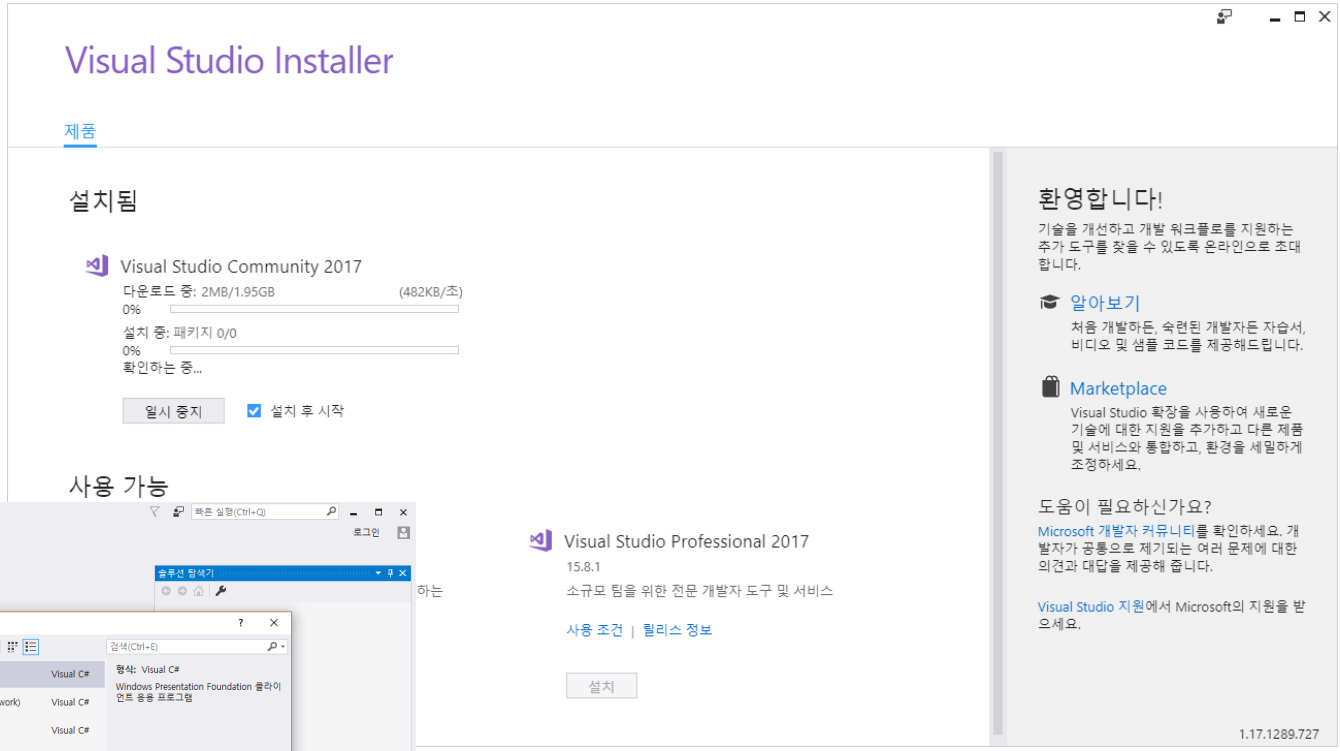
VS community 2017 이므로 vc15 동적 라이브러리 사용

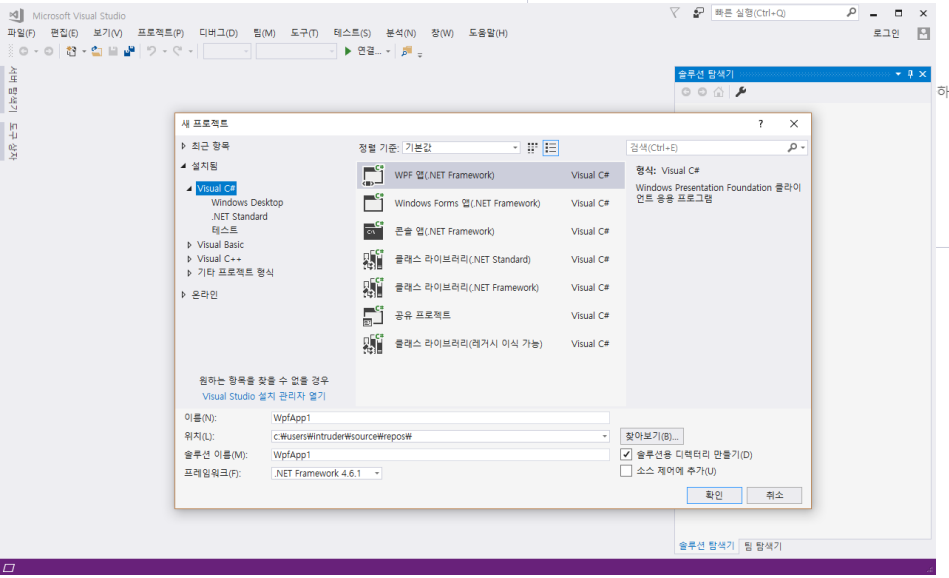


릴리즈 배포시에는???

>> dll 파일을 exe파일과 같이 배포

Open CV





2018년 최신버전 community 2017 사용

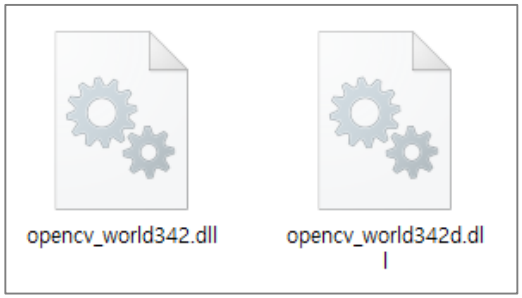
VC15

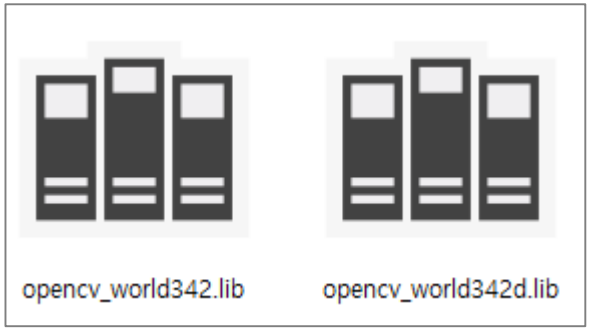
https://visualstudio.microsoft.com/ko/vs/community/?rr=https%3A%2F%2Fwww.google.co.kr%2F

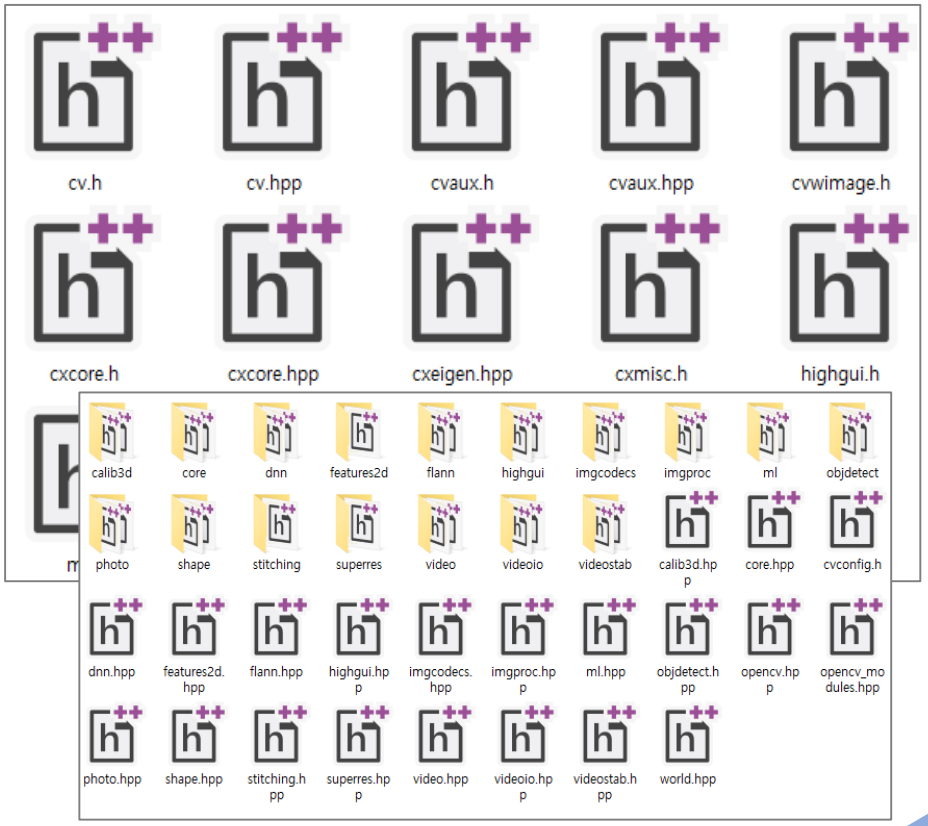
Open CV

Visual Studio 에서 OpenCV 디렉토리 설정

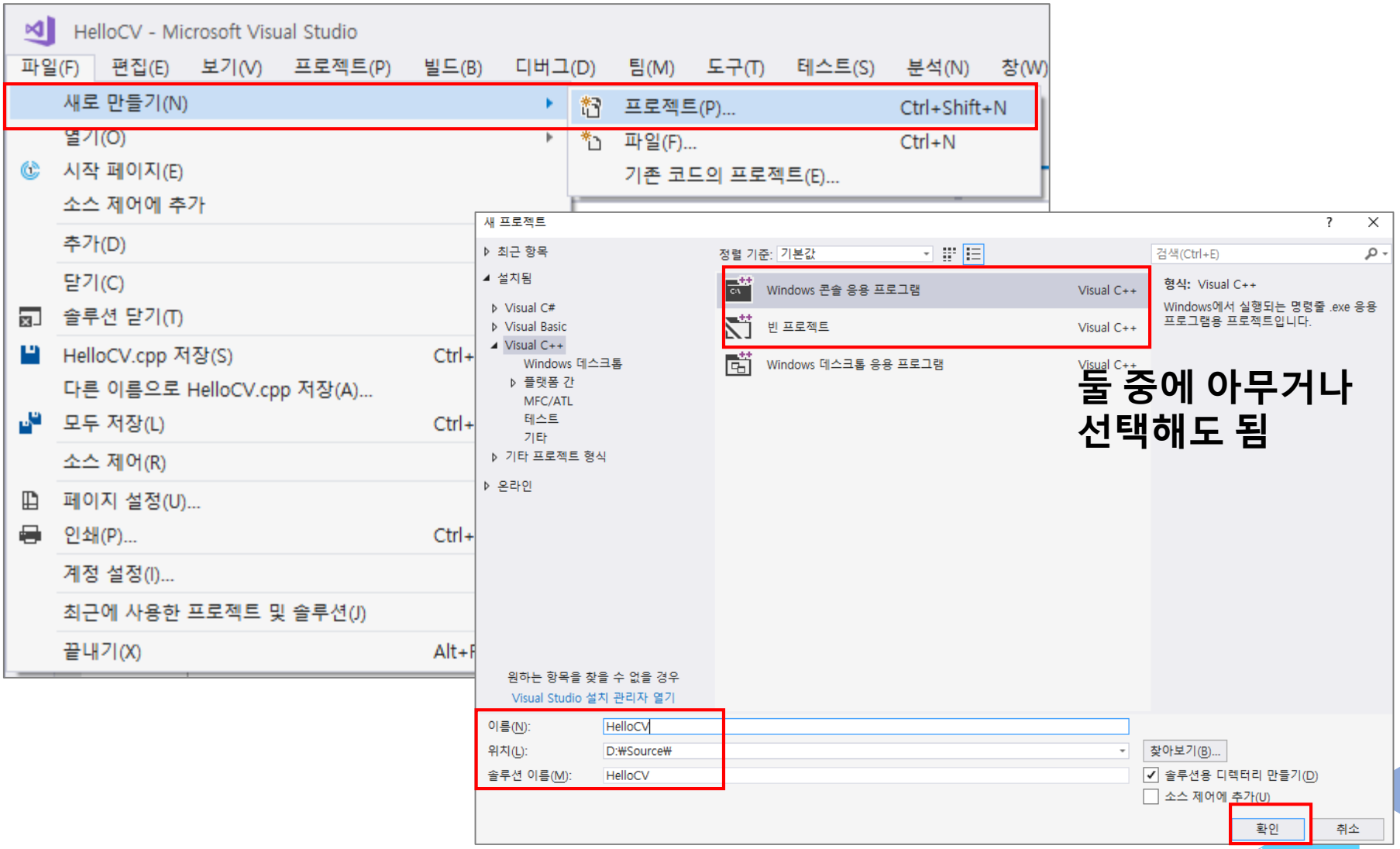
- 총 세 가지 파일 필요
  - > Open API 파일 (.dll)
  - > 헤더 파일 (.hpp)
  - > dll 빌드할 때 함수 정보를 포함하는 라이브러리 (.lib)





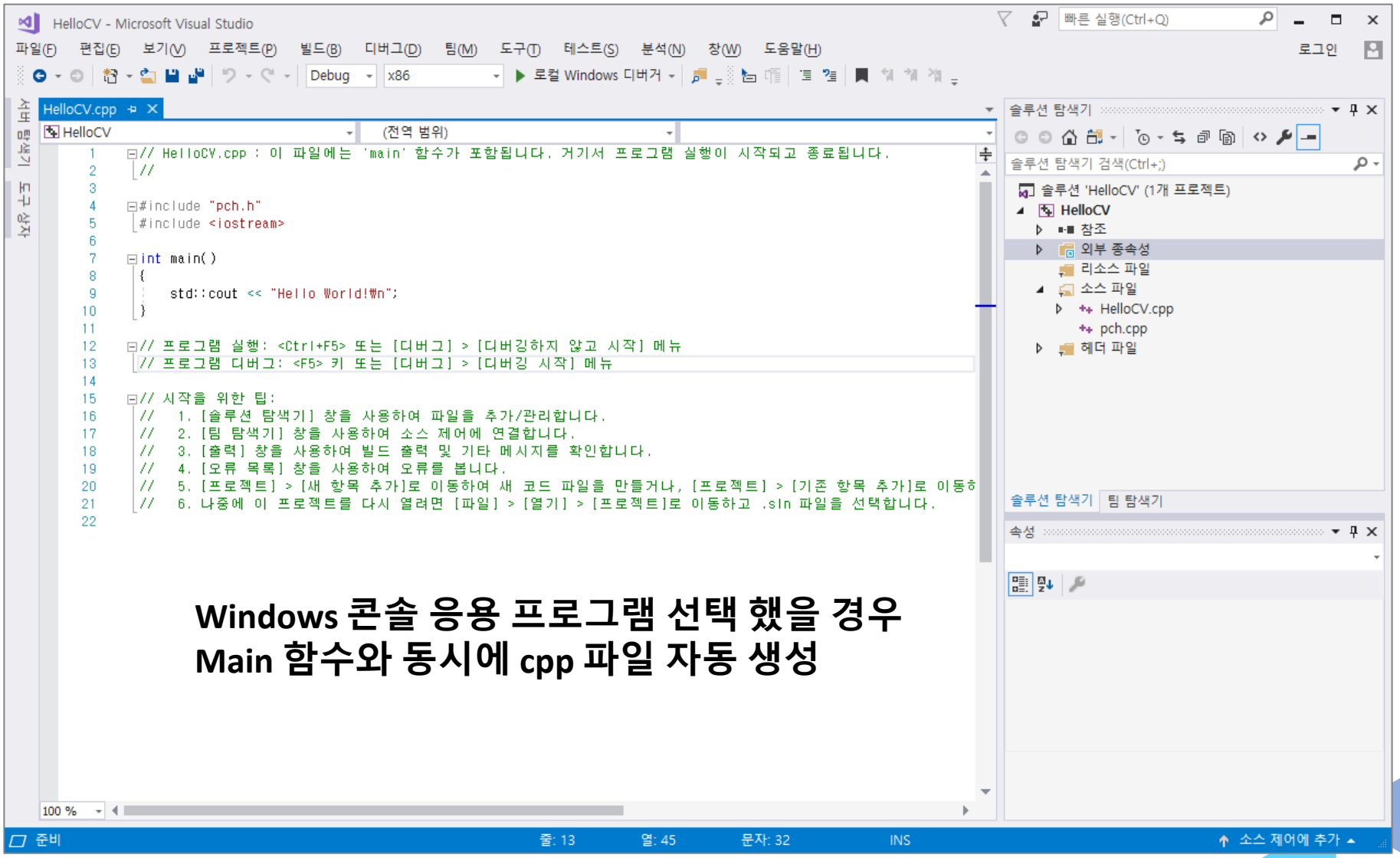


Open CV



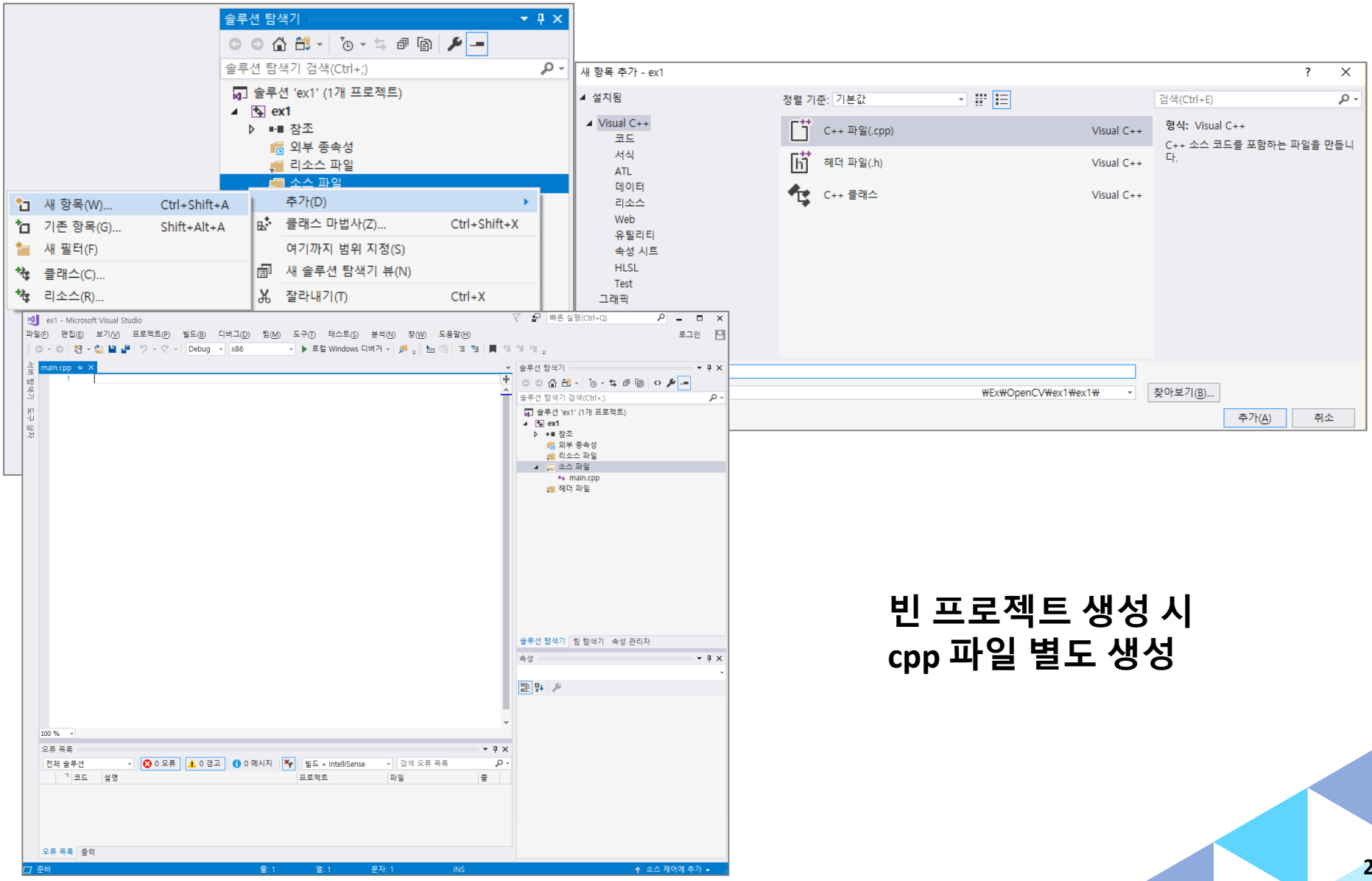
둘 중에 아무거나 선택해도 됨

Open CV



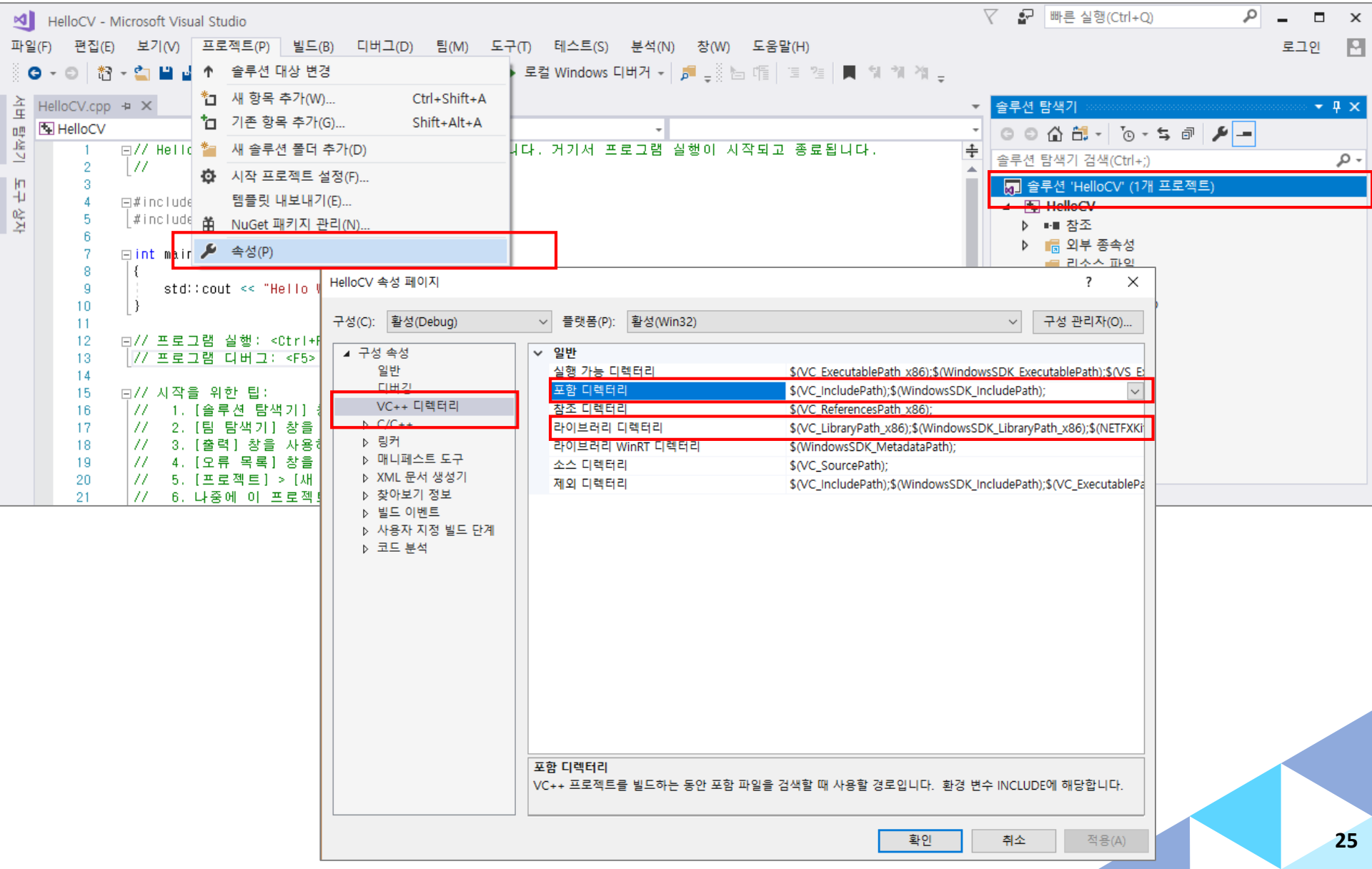
Windows 콘솔 응용 프로그램 선택 했을 경우
Main 함수와 동시에 cpp 파일 자동 생성

Open CV

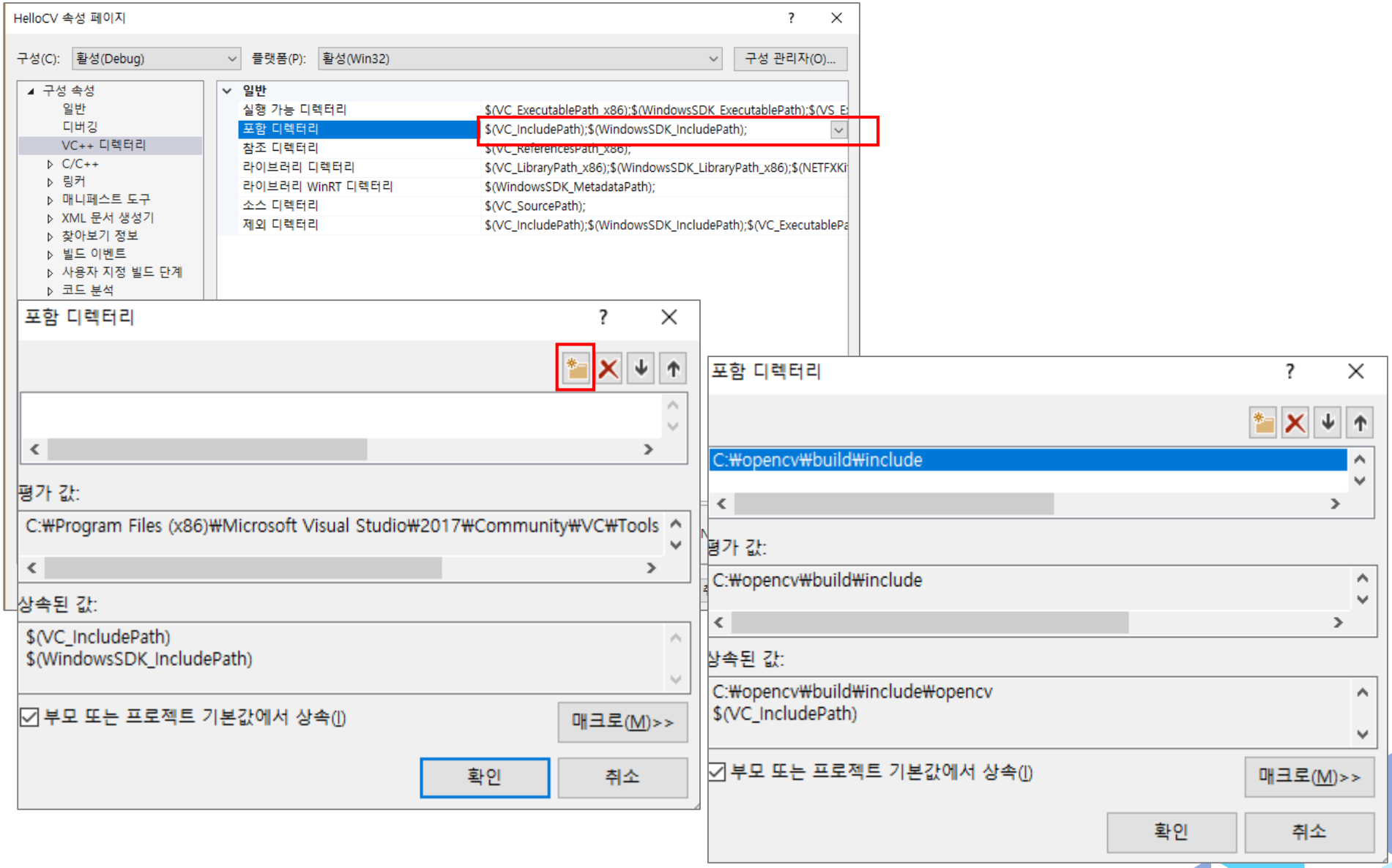


빈 프로젝트 생성 시
cpp 파일 별도 생성

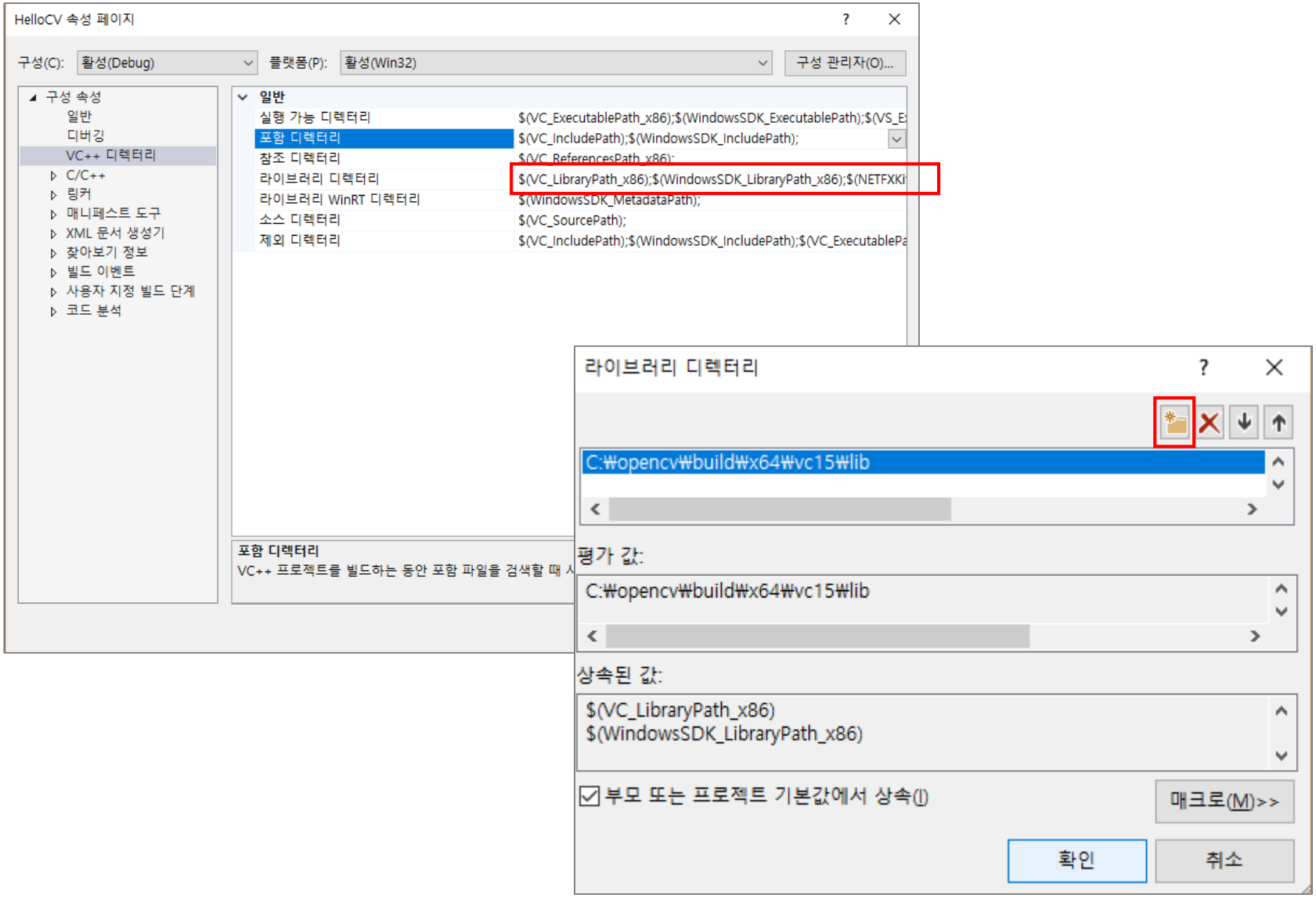
Open CV


HelloCV - Microsoft Visual Studio
파일(F) 편집(E) 보기(V) 프로젝트(P) 빌드(B) 디버그(D) 팀(M) 도구(T) 테스트(S) 분석(N) 창(W) 도움말(H)
로그인
빠른 실행(Ctrl+Q)
로컬 Windows 디버거
솔루션 대상 변경
새 항목 추가(W)... Ctrl+Shift+A
기존 항목 추가(G)... Shift+Alt+A
새 솔루션 폴더 추가(D)
시작 프로젝트 설정(F)...
템플릿 내보내기(E)...
NuGet 패키지 관리(N)...
속성(P)
HelloCV.cpp
HelloCV
솔루션 탐색기
솔루션 탐색기 검색(Ctrl+;)
솔루션 'HelloCV' (1개 프로젝트)
HelloCV
참조
외부 종속성
HelloCV 속성 페이지
구성(C): 활성(Debug)
플랫폼(P): 활성(Win32)
구성 관리자(O)...
구성 속성
일반
디버깅
VC++ 디렉터리
C/C++
링커
매니페스트 도구
XML 문서 생성기
찾아보기 정보
빌드 이벤트
사용자 지정 빌드 단계
코드 분석
일반
실행 가능 디렉터리 $(VC_ExecutablePath_x86);$(WindowsSDK_ExecutablePath);$(VS_E
포함 디렉터리 $(VC_IncludePath);$(WindowsSDK_IncludePath);
참조 디렉터리 $(VC_ReferencesPath_x86);
라이브러리 디렉터리 $(VC_LibraryPath_x86);$(WindowsSDK_LibraryPath_x86);$(NETFXKi
라이브러리 WinRT 디렉터리 $(WindowsSDK_MetadataPath);
소스 디렉터리 $(VC_SourcePath);
제외 디렉터리 $(VC_IncludePath);$(WindowsSDK_IncludePath);$(VC_ExecutablePa
포함 디렉터리
VC++ 프로젝트를 빌드하는 동안 포함 파일을 검색할 때 사용할 경로입니다. 환경 변수 INCLUDE에 해당합니다.
확인
취소
적용(A)

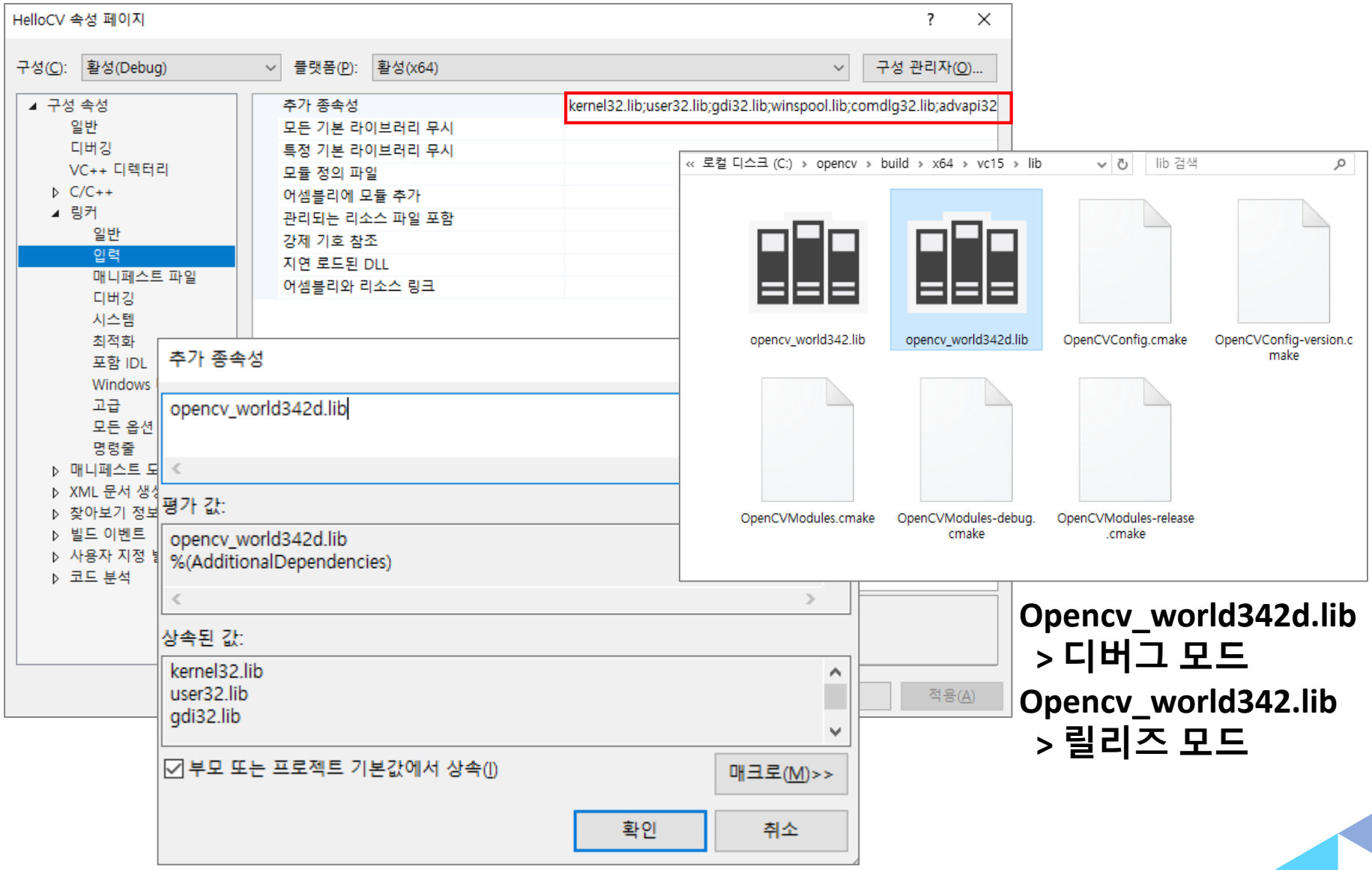
Open CV


HelloCV 속성 페이지
구성(C): 활성(Debug)
플랫폼(P): 활성(Win32)
구성 관리자(O)...
구성 속성
일반
디버깅
VC++ 디렉터리
C/C++
링커
매니페스트 도구
XML 문서 생성기
찾아보기 정보
빌드 이벤트
사용자 지정 빌드 단계
코드 분석
일반
실행 가능 디렉터리
포함 디렉터리
참조 디렉터리
라이브러리 디렉터리
라이브러리 WinRT 디렉터리
소스 디렉터리
제외 디렉터리
$(VC_IncludePath);$(WindowsSDK_IncludePath);
포함 디렉터리
평가 값:
C:₩Program Files (x86)₩Microsoft Visual Studio₩2017₩Community₩VC₩Tools
상속된 값:
$(VC_IncludePath)
$(WindowsSDK_IncludePath)
부모 또는 프로젝트 기본값에서 상속(I)
매크로(M)>>
확인
취소
포함 디렉터리
C:₩opencv₩build₩include
평가 값:
C:₩opencv₩build₩include
상속된 값:
C:₩opencv₩build₩include₩opencv
$(VC_IncludePath)
부모 또는 프로젝트 기본값에서 상속(I)
매크로(M)>>
확인
취소

Open CV


HelloCV 속성 페이지
구성(C): 활성(Debug)
플랫폼(P): 활성(Win32)
구성 관리자(O)...
구성 속성
일반
디버깅
VC++ 디렉터리
C/C++
링커
매니페스트 도구
XML 문서 생성기
찾아보기 정보
빌드 이벤트
사용자 지정 빌드 단계
코드 분석
일반
실행 가능 디렉터리 $(VC_ExecutablePath_x86);$(WindowsSDK_ExecutablePath);$(VS_E
포함 디렉터리 $(VC_IncludePath);$(WindowsSDK_IncludePath);
참조 디렉터리 $(VC_ReferencesPath_x86);
라이브러리 디렉터리 $(VC_LibraryPath_x86);$(WindowsSDK_LibraryPath_x86);$(NETFXKi
라이브러리 WinRT 디렉터리 $(WindowsSDK_MetadataPath);
소스 디렉터리 $(VC_SourcePath);
제외 디렉터리 $(VC_IncludePath);$(WindowsSDK_IncludePath);$(VC_ExecutablePa
포함 디렉터리
VC++ 프로젝트를 빌드하는 동안 포함 파일을 검색할 때
라이브러리 디렉터리
C:₩opencv₩build₩x64₩vc15₩lib
평가 값:
C:₩opencv₩build₩x64₩vc15₩lib
상속된 값:
$(VC_LibraryPath_x86)
$(WindowsSDK_LibraryPath_x86)
부모 또는 프로젝트 기본값에서 상속(I)
매크로(M)>>
확인
취소

Open CV

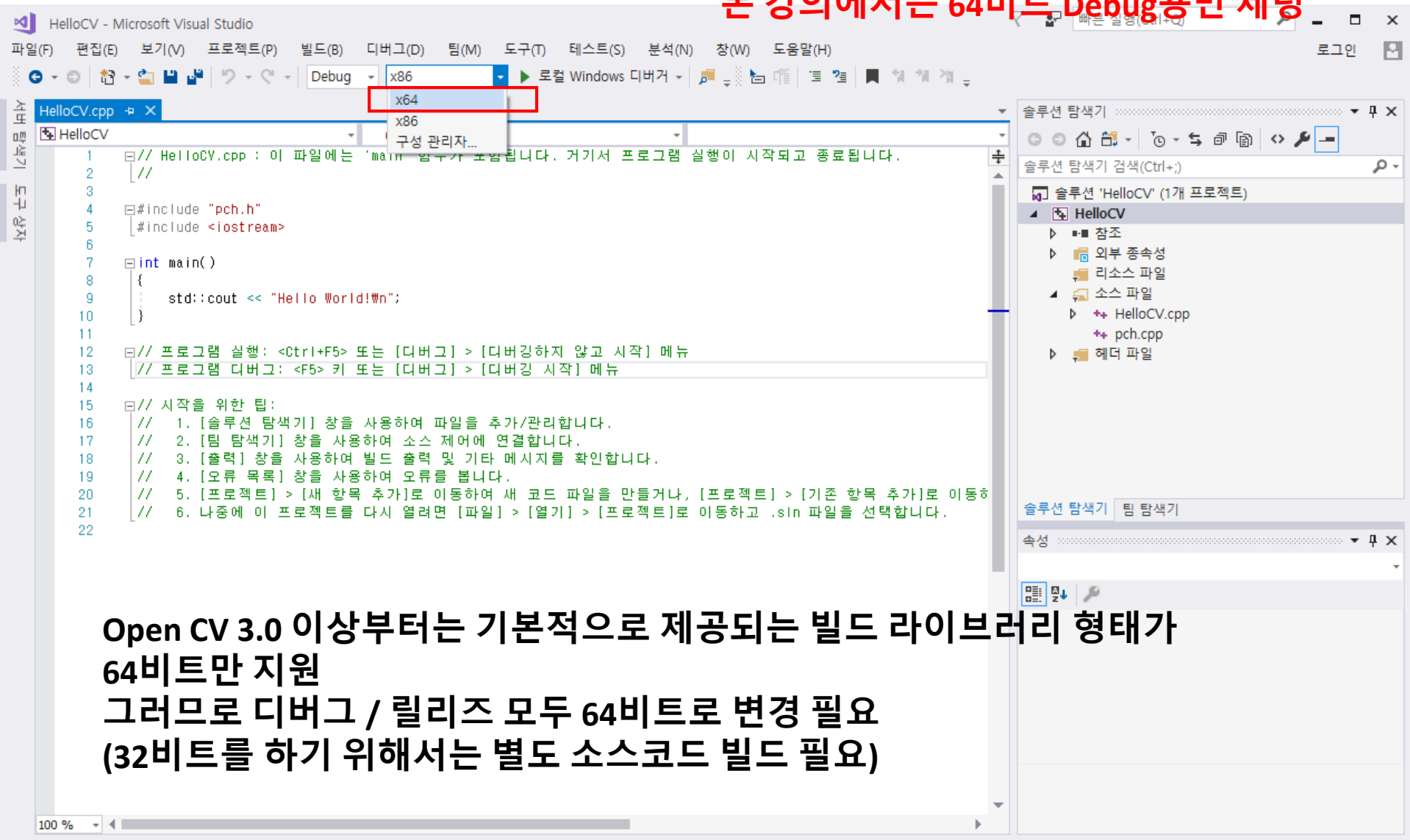


Opencv_world342d.lib
> 디버그 모드

Opencv_world342.lib
> 릴리즈 모드

Open CV

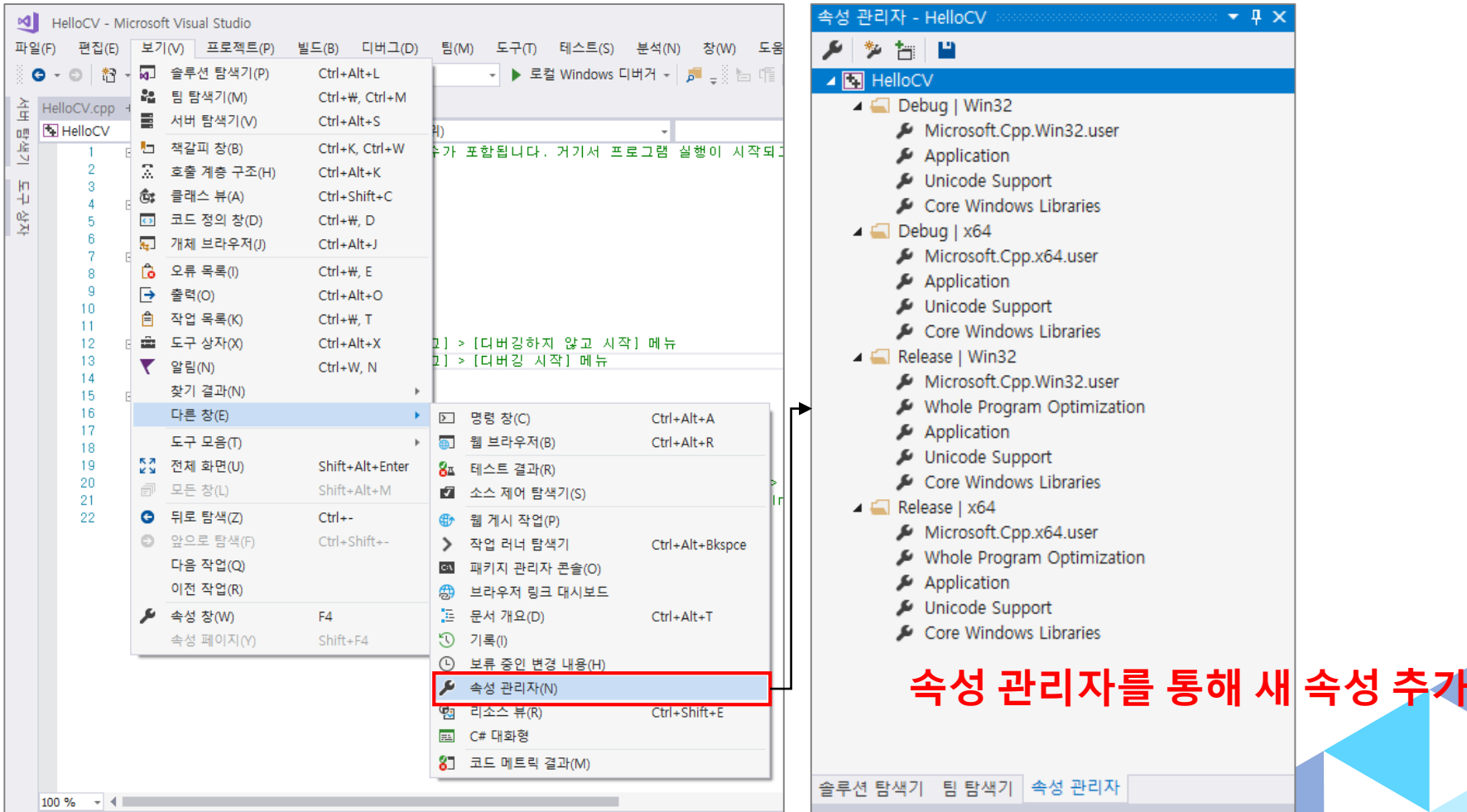
본 강의에서는 64비트 Debug용만 세팅



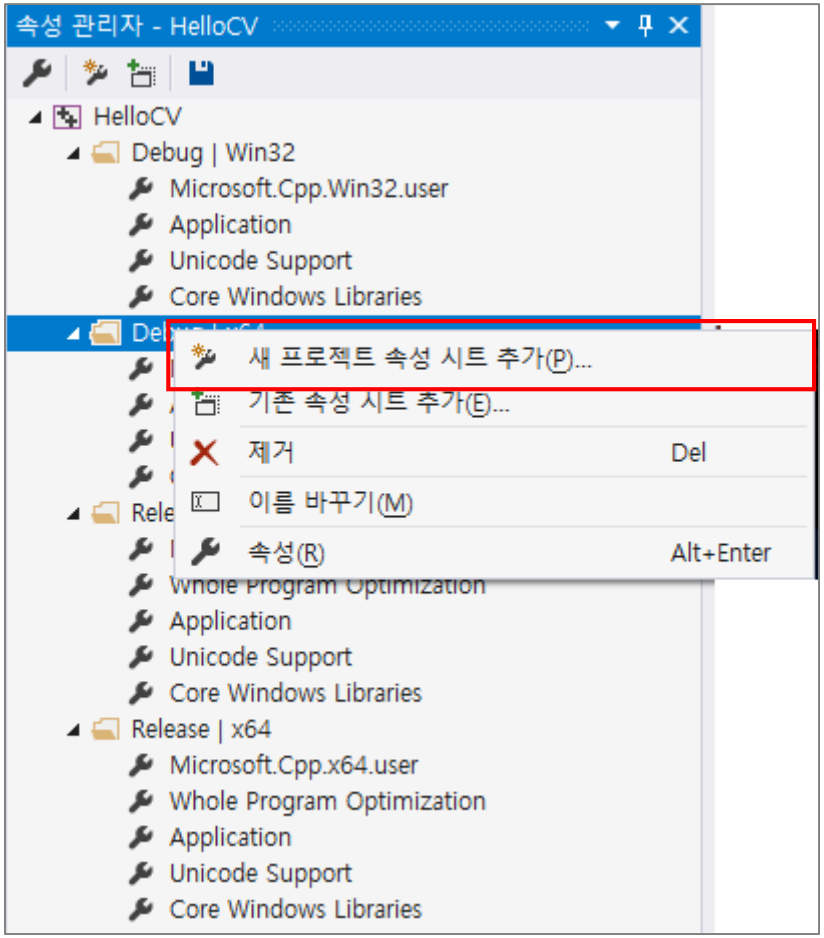
Open CV 3.0 이상부터는 기본적으로 제공되는 빌드 라이브러리 형태가 64비트만 지원
그러므로 디버그 / 릴리즈 모두 64비트로 변경 필요
(32비트를 하기 위해서는 별도 소스코드 빌드 필요)

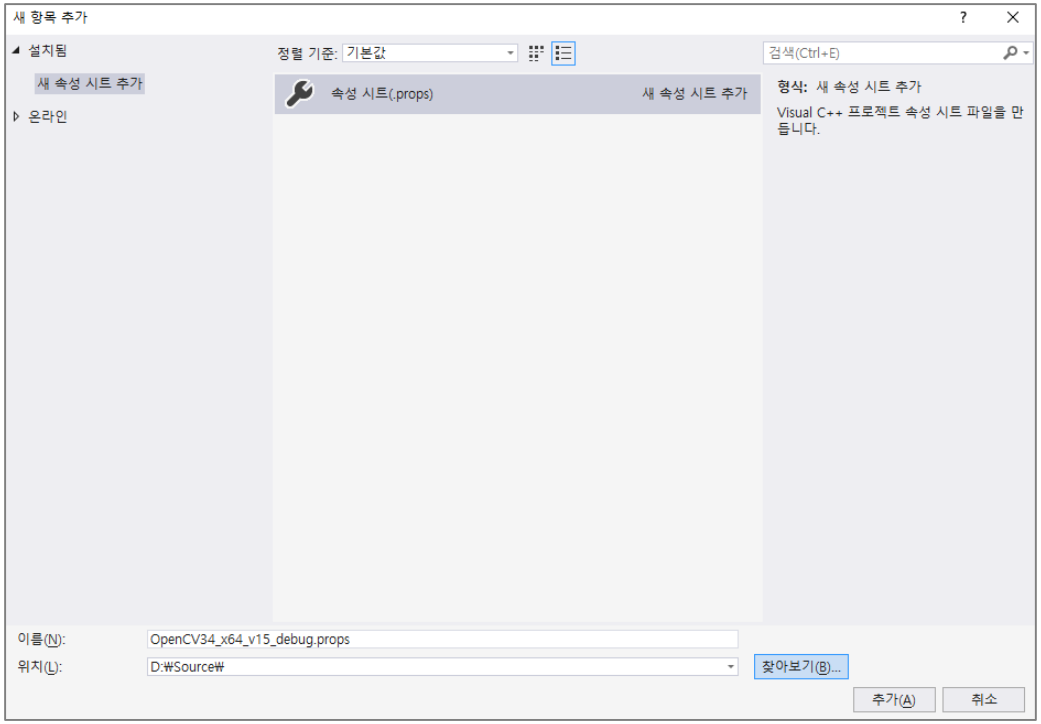
Open CV

- 위 수행한 속성 설정은 프로젝트에만 해당
- 새 프로젝트 생성 시 동일한 작업을 매번 수행
- 그러므로 이를 저장할 수 있는 “속성 시트“ 기능 존재

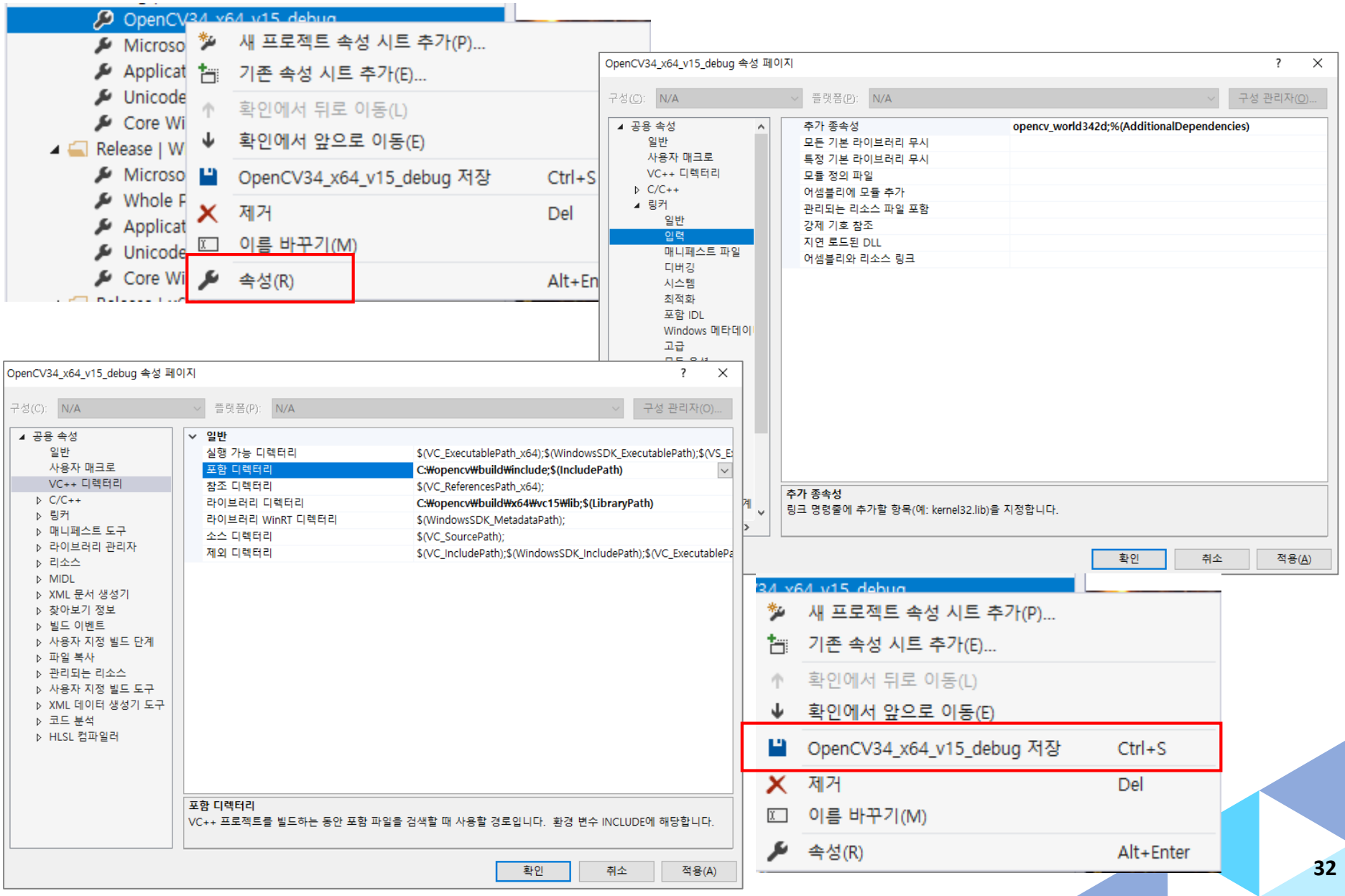


Open CV


속성 관리자 - HelloCV
HelloCV
Debug | Win32
Microsoft.Cpp.Win32.user
Application
Unicode Support
Core Windows Libraries
새 프로젝트 속성 시트 추가(P)...
기존 속성 시트 추가(E)...
제거 Del
이름 바꾸기(M)
속성(R) Alt+Enter
Application
Unicode Support
Core Windows Libraries
Release | x64
Microsoft.Cpp.x64.user
Whole Program Optimization
Application
Unicode Support
Core Windows Libraries


새 항목 추가
설치됨
새 속성 시트 추가
온라인
정렬 기준: 기본값
검색(Ctrl+E)
속성 시트(.props)
새 속성 시트 추가
형식: 새 속성 시트 추가
Visual C++ 프로젝트 속성 시트 파일을 만듭니다.
이름(N): OpenCV34_x64_v15_debug.props
위치(L): D:₩Source₩
찾아보기(B)...
추가(A)
취소

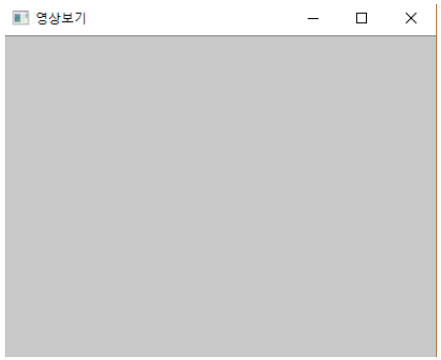
Open CV


새 프로젝트 속성 시트 추가(P)...
기존 속성 시트 추가(E)...
확인에서 뒤로 이동(L)
확인에서 앞으로 이동(E)
OpenCV34_x64_v15_debug 저장 Ctrl+S
제거 Del
이름 바꾸기(M)
속성(R) Alt+En
OpenCV34_x64_v15_debug 속성 페이지
구성(C): N/A
플랫폼(P): N/A
구성 관리자(O)...
공용 속성
일반
사용자 매크로
VC++ 디렉터리
C/C++
링커
일반
입력
매니페스트 파일
디버깅
시스템
최적화
포함 IDL
Windows 메타데이
고급
추가 종속성 opencv_world342d;%(AdditionalDependencies)
모든 기본 라이브러리 무시
특정 기본 라이브러리 무시
모듈 정의 파일
어셈블리에 모듈 추가
관리되는 리소스 파일 포함
강제 기호 참조
지연 로드된 DLL
어셈블리와 리소스 링크
추가 종속성
링크 명령줄에 추가할 항목(예: kernel32.lib)을 지정합니다.
확인
취소
적용(A)
OpenCV34_x64_v15_debug 속성 페이지
구성(C): N/A
플랫폼(P): N/A
구성 관리자(O)...
공용 속성
일반
사용자 매크로
VC++ 디렉터리
C/C++
링커
매니페스트 도구
라이브러리 관리자
리소스
MIDL
XML 문서 생성기
찾아보기 정보
빌드 이벤트
사용자 지정 빌드 단계
파일 복사
관리되는 리소스
사용자 지정 빌드 도구
XML 데이터 생성기 도구
코드 분석
HLSL 컴파일러
일반
실행 가능 디렉터리 $(VC_ExecutablePath_x64);$(WindowsSDK_ExecutablePath);$(VS_E
포함 디렉터리 C:₩opencv₩build₩include;$(IncludePath)
참조 디렉터리 $(VC_ReferencesPath_x64);
라이브러리 디렉터리 C:₩opencv₩build₩x64₩vc15₩lib;$(LibraryPath)
라이브러리 WinRT 디렉터리 $(WindowsSDK_MetadataPath);
소스 디렉터리 $(VC_SourcePath);
제외 디렉터리 $(VC_IncludePath);$(WindowsSDK_IncludePath);$(VC_ExecutablePa
포함 디렉터리
VC++ 프로젝트를 빌드하는 동안 포함 파일을 검색할 때 사용할 경로입니다. 환경 변수 INCLUDE에 해당합니다.
확인
취소
적용(A)
새 프로젝트 속성 시트 추가(P)...
기존 속성 시트 추가(E)...
확인에서 뒤로 이동(L)
확인에서 앞으로 이동(E)
OpenCV34_x64_v15_debug 저장 Ctrl+S
제거 Del
이름 바꾸기(M)
속성(R) Alt+Enter

간단한 창 띄우기

- API를 제대로 사용하는지 확인하기 위해 간단한 창을 띄어봄

```
#include <opencv2/highgui.hpp>

int main()
{
  cv::Mat image(300, 400, CV_8UC1, cv::Scalar(200));
  cv::imshow("영상보기", image);
  cv::waitKey(0);
}
```



Open CV

기본 헤더파일

- opencv.hpp 기본 헤더 파일
- 이를 추가하면 다른 기본 헤더 파일은 별도 정의할 필요 없음

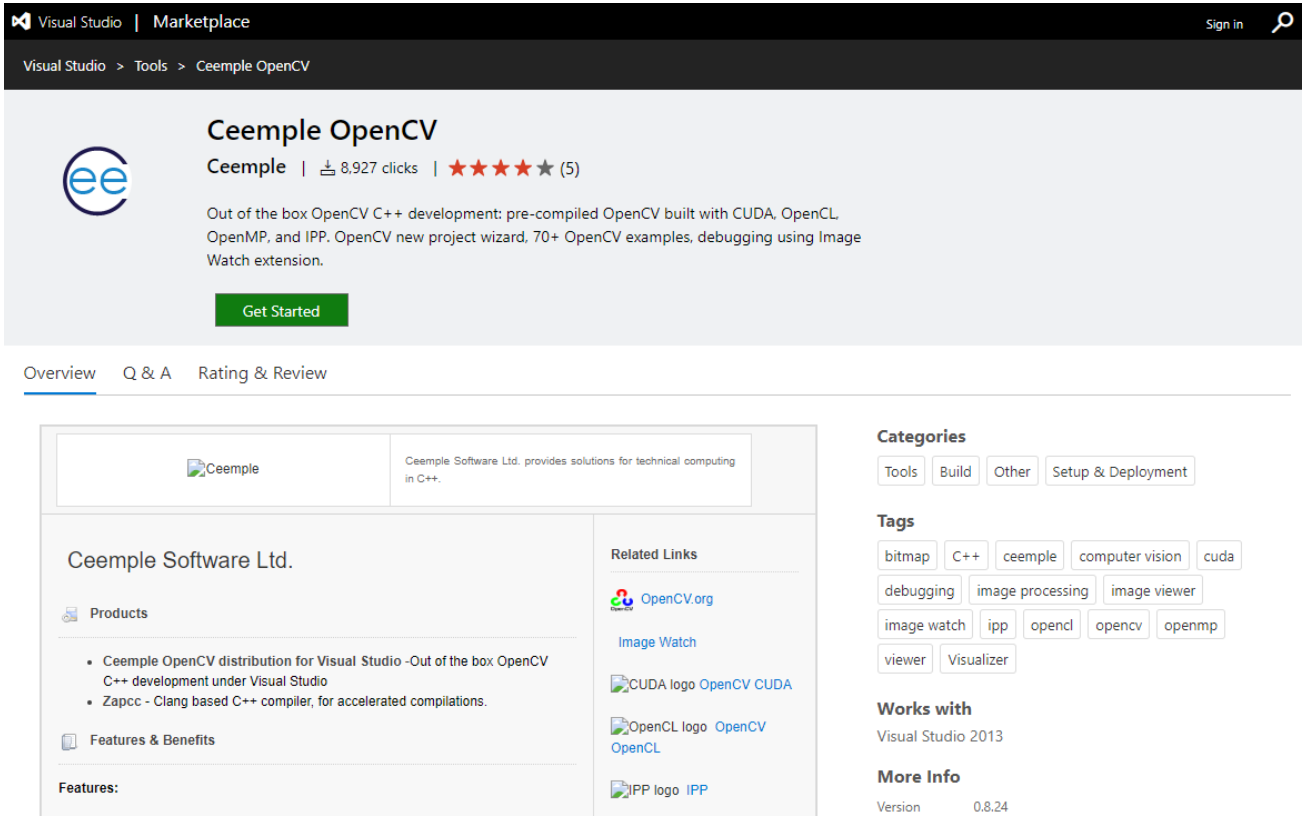
```
#include "opencv2/core.hpp"

// Then the optional modules are checked
#ifdef HAVE_OPENCV_CALIB3D
#include "opencv2/calib3d.hpp"
#endif
#ifdef HAVE_OPENCV_FEATURES2D
#include "opencv2/features2d.hpp"
#endif
#ifdef HAVE_OPENCV_DNN
#include "opencv2/dnn.hpp"
#endif
#ifdef HAVE_OPENCV_FLANN
#include "opencv2/flann.hpp"
#endif
#ifdef HAVE_OPENCV_HIGHGUI
#include "opencv2/highgui.hpp"
#endif
#ifdef HAVE_OPENCV_IMGCODECS
#include "opencv2/imgcodecs.hpp"
#endif
#ifdef HAVE_OPENCV_IMGPROC
#include "opencv2/imgproc.hpp"
#endif
#ifdef HAVE_OPENCV_ML
#include "opencv2/ml.hpp"
#endif
#ifdef HAVE_OPENCV_OBJDETECT
#include "opencv2/objdetect.hpp"
#endif
#ifdef HAVE_OPENCV_PHOTO
#include "opencv2/photo.hpp"
#endif
#ifdef HAVE_OPENCV_SHAPE
#include "opencv2/shape.hpp"
#endif
#ifdef HAVE_OPENCV_STITCHING
#include "opencv2/stitching.hpp"
#endif
#ifdef HAVE_OPENCV_SUPERRES
#include "opencv2/superres.hpp"
#endif
#ifdef HAVE_OPENCV_VIDEO
```

```
// Finally CUDA specific entries are checked and added
#ifdef HAVE_OPENCV_CUDAARITHM
#include "opencv2/cudaarithm.hpp"
#endif
#ifdef HAVE_OPENCV_CUDABGSEGM
#include "opencv2/cudabgsegm.hpp"
#endif
#ifdef HAVE_OPENCV_CUDACODEC
#include "opencv2/cudacodec.hpp"
#endif
#ifdef HAVE_OPENCV_CUDAFEATURES2D
#include "opencv2/cudafeatures2d.hpp"
#endif
#ifdef HAVE_OPENCV_CUDAFILTERS
#include "opencv2/cudafilters.hpp"
#endif
#ifdef HAVE_OPENCV_CUDAIMGPROC
#include "opencv2/cudaimgproc.hpp"
#endif
#ifdef HAVE_OPENCV_CUDAOBJDETECT
#include "opencv2/cudaobjdetect.hpp"
#endif
#ifdef HAVE_OPENCV_CUDAOPTFLOW
#include "opencv2/cudaoptflow.hpp"
#endif
#ifdef HAVE_OPENCV_CUDASTEREO
#include "opencv2/cudastereo.hpp"
#endif
#ifdef HAVE_OPENCV_CUDAWARPING
#include "opencv2/cudawarping.hpp"
#endif

#endif
```

Open CV

Ceemple Open CV

- OpenCV 프로젝트 생성시 바로 사용할 수 있도록 가능하게 해주는 비쥬얼 스튜디오 확장팩
- 하지만 Visual Studio 2013 버전에서 제공



좀 더 많은 기능이 포함되어 빌드된 Open CV 라이브러리로 편리하게 사용할 수 있다는 장점

https://marketplace.visualstudio.com/items?itemName=Ceemple.CeempleOpenCV

Open CV 기본

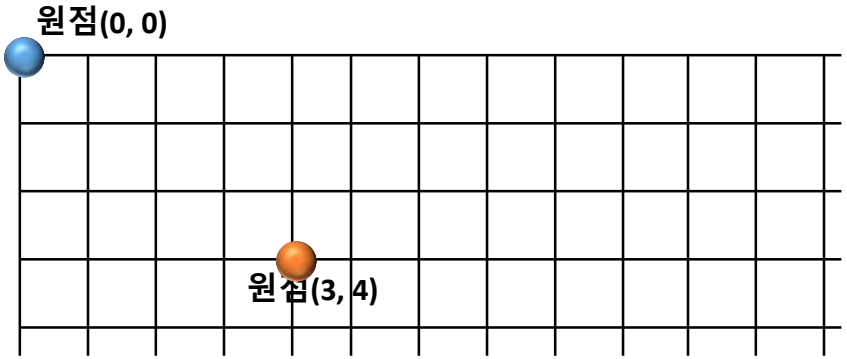
OpenCV 기본 자료 구조

- C++ API 에서는 Point2_, Point3_, Size_, Rect_, Vec_, Scalar_, Mat_ 등 자료구조를 위한 템플릿 클래스 제공

Point_ 클래스

- 가로와 세로의 위치를 2차원 좌표로 나타내기 위한 탬플릿 클래스
- 멤버 변수 : x, y

Point_ <int> pt(3, 4)



https://docs.opencv.org/3.4.2/dc/dd1/structCvPoint.html

Open CV 기본

Point_ 클래스 예제

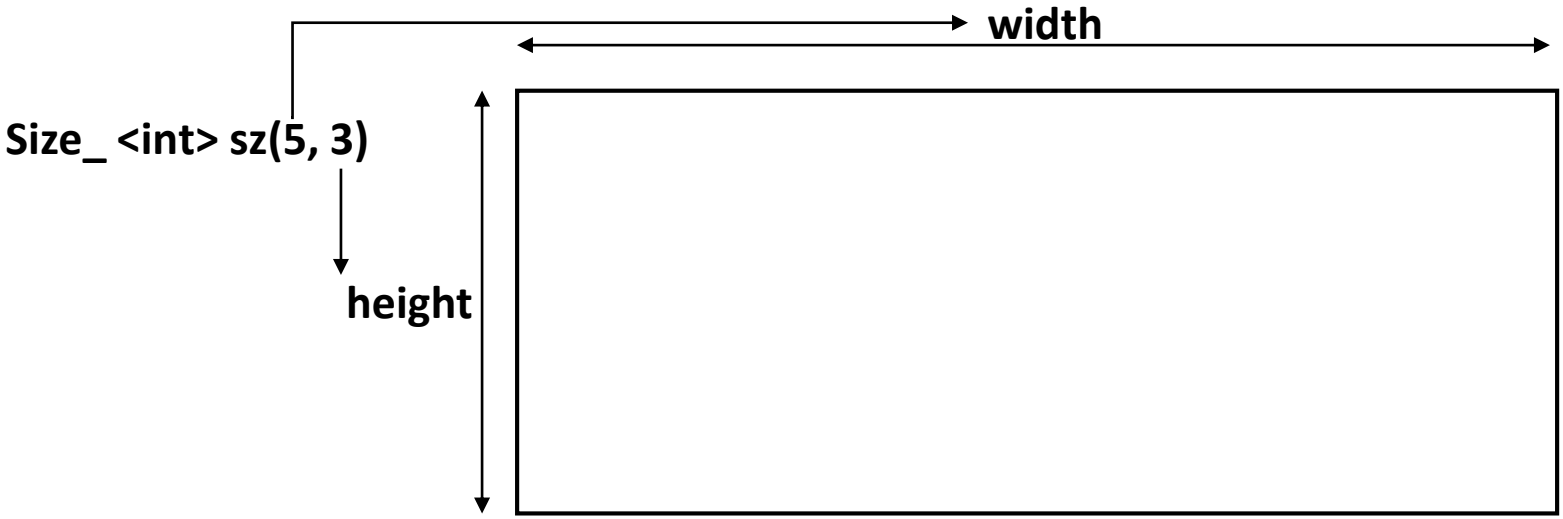
```
#include <opencv2/opencv.hpp>

int  main()
{
  // Point_ 객체 선언 방식
  cv::Point_<int> pt1(100, 200);
  cv::Point_<float> pt2(92.3f, 125.23f);
  cv::Point_<double> pt3(100.2, 300.9);

  // Point_ 객체 간결 선언 방식
  cv::Point2i  pt4(120, 69);
  cv::Point2f  pt5(0.3f, 0.f), pt6(0.f, 0.4f);
  cv::Point2d  pt7(0.25, 0.6);

  // Point_ 객체 연산
  cv::Point    pt8 = pt1 + (cv::Point) pt2;
  cv::Point2f  pt9 = pt6 * 3.14f;
  cv::Point2d  pt10 = (pt3 + (cv::Point2d) pt6) * 10;

  std::cout << "pt8 = " << pt8.x << " , " << pt8.y << std::endl;
  std::cout << "[pt9] = " << pt9 << std::endl;
  std::cout << (pt2 == pt6) << std::endl;
  std::cout << "pt7과 pt8의 내적 : " << pt7.dot(pt8) << std::endl;
  system("pause");
  return 0;
}
```

```
D:₩OpenCV₩ex2_point2₩x64₩Debug₩ex2_point2.exe
pt8 = 192 , 325
[pt9] = [0, 1.256]
0
pt7과 pt8의 내적 : 243
계속하려면 아무 키나 누르십시오 . . .
```

Open CV 기본

Size_ 클래스

- 이미지나 사각형 크기를 규정하는 템플릿 클래스
- 멤버 변수 : width, height



https://docs.opencv.org/3.4.2/da/dcb/structCvSize.html

Open CV 기본

Size_ 클래스 예제

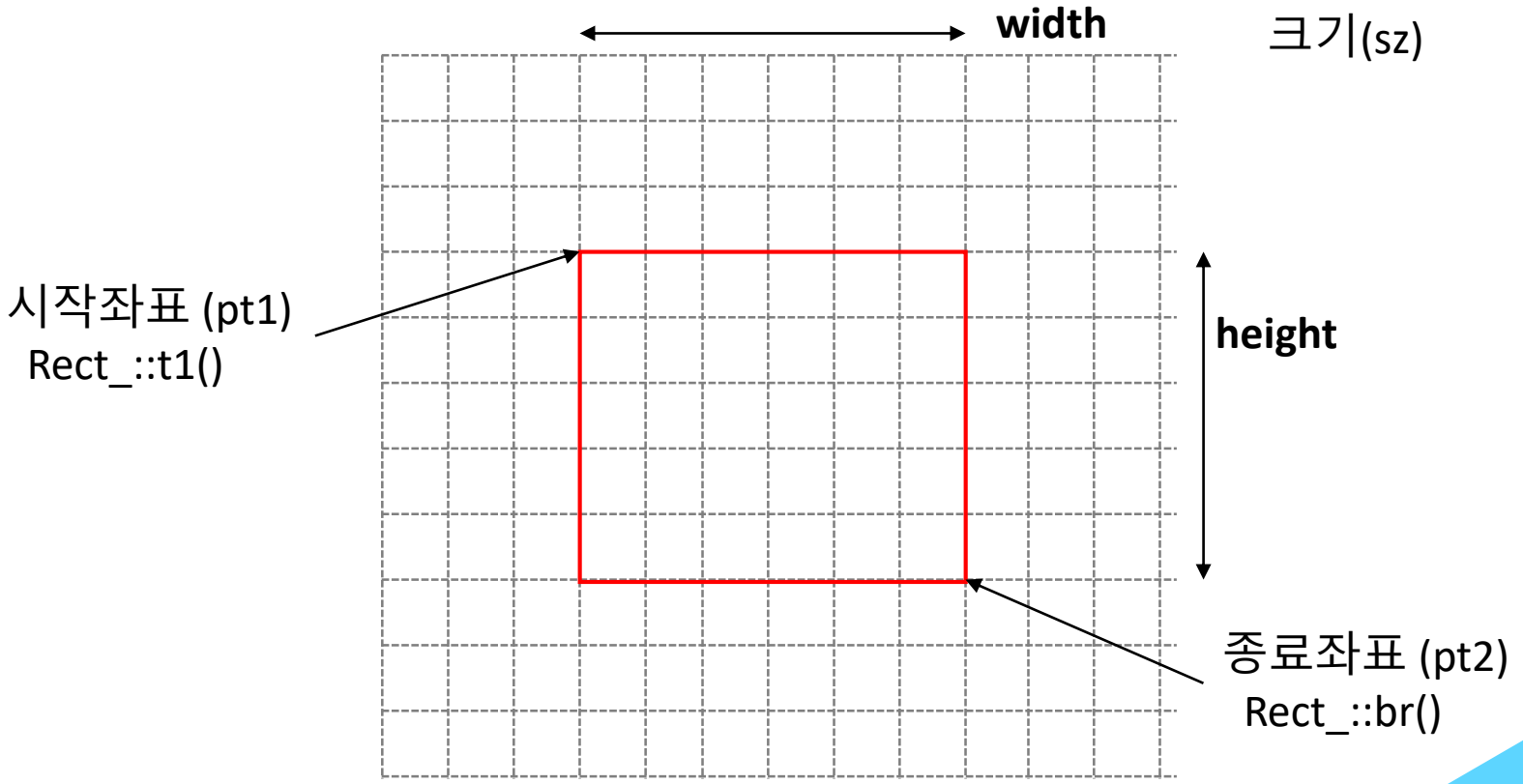
```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int  main()
{
        // Size_ 객체 기본 선언 방식
        Size_<int> sz1(100, 200);
        Size_<float> sz2(192.3f, 25.3f);
        Size_<double> sz3(100.2, 30.9);

        // Size 객체 간결 선언 방식
        Size   sz4(120, 69);
        Size2f  sz5(0.3f, 0.f);
        Size2d  sz6(0.25, 0.6);

        Point2d  pt1(0.25, 0.6);
        Size2i   sz7 = sz1 + (Size2i)sz2;
        Size2d   sz8 = sz3 - (Size2d)sz4;
        Size2d   sz9 = sz5 + (Size2f)pt1;

        cout << "sz1.width = " << sz1.width;
        cout << ",  sz1.height = " << sz1.height << endl;
        cout << "sz1 넓이 " << sz1.area() << endl;
        cout << "[sz7] = " << sz7 << endl;
        cout << "[sz8] = " << sz8 << endl;
        cout << "[sz9] = " << sz9 << endl;
        system("pause");
        return 0;
}
```

```
선택 D:₩OpenCV₩ex3_size₩x64₩Debug₩ex3_size.exe
sz1.width = 100,  sz1.height = 200
sz1 넓이 20000
[sz7] = [292 x 225]
[sz8] = [-19.8 x -38.1]
[sz9] = [0.55 x 0.6]
계속하려면 아무 키나 누르십시오 . . .
```

Open CV 기본

Rect_ 클래스

- 2차원 사각형 정보를 나타내기 위한 탬플릿 클래스
- 멤버 변수: 시작 좌표(x, y), 크기(width, height)



https://docs.opencv.org/3.4.2/d0/d76/structCvRect.html

Open CV 기본

Rect_ 클래스 예제

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int  main()
{
        Size2d  sz(100.5, 60.6);
        Point2f  pt1(20.f, 30.f), pt2(100.f, 200.f);

        // Rect_ 객체 기본 선언 방식
        Rect_<int>    rect1(10, 10, 30, 50);
        Rect_<float>   rect2(pt1, pt2);
        Rect_<double> rect3(Point2d(20.5, 10), sz);

        // 간결 선언 방식 & 연산 적용
        Rect   rect4 = rect1 + (Point)pt1;
        Rect2f rect5 = rect2 + (Size2f)sz;
        Rect2d rect6 = rect1 & (Rect)rect2;

        // 결과 출력
        cout << "rect3 = " << rect3.x << "," << rect3.y << ", ";
        cout << rect3.width << "x" << rect3.height << endl;
        cout << "rect4 = " << rect4.tl() << "  " << rect4.br()
        << endl;
        cout << "rect5 크기 = " << rect5.size() << endl;
        cout << "[rect6] = " << rect6 << endl;
        system("pause");
        return 0;
}
```

```
선택 D:₩OpenCV₩ex4_rect₩x64₩Debug₩ex4_rect.exe
rect3 = 20.5,10, 100.5x60.6
rect4 = [30, 40]  [60, 90]
rect5 크기 = [180.5 x 230.6]
[rect6] = [20 x 30 from (20, 30)]
계속하려면 아무 키나 누르십시오 . . .
```

Open CV 기본

Vec_ 클래스

- 원소 개수가 작은 숫자 벡터를 위한 템플릿 클래스
- Vec<Tp, 2>, Vec<Tp, 3>, Vec<Tp, 4>
  각 각 Point_, Point3_, Scalar_ 클래스로 형변환 가능

Scalar_ 클래스

- Vec<Tp, 4)에서 파생된 템플릿 클래스
- 4개 원소를 갖음
- OpenCV에서 특별히 화소의 값을 지정하기 위한 자료형으로 정의
- 파랑, 초록, 빨강, 투명도 4개 값 저장

https://docs.opencv.org/3.4.2/d6/dcf/classcv_1_1Vec.html
https://docs.opencv.org/3.4.2/d6/db3/structCvScalar.html

Open CV 기본

Mat 클래스

- 1채널 또는 다채널의 실수, 복소수, 행렬, 영상 등 수치 데이터를 표현하는 N 차원 행렬 클래스 (행과 열을 다루는 클래스)
- 1버전에서는 IplImage를 사용하여 화소에 접근(구조체)
- 이미지 데이터는 모두 행과 열로 구성
- 이미지 저장소라고 생각해도 무방

$$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

(예)

...	...	...	...	...	...	...	...	...	...
...	0	0	0	0	0	20	30	40	...
...	0	0	0	0	0	0	0	0	...
...	0	0	0	0	0	0	0	0	...
...	0	0	0	0	20	56	40	0	...
...	20	0	0	0	0	0	0	0	...
...	0	0	0	0	30	20	40	20	...
...	...	...	...	...	...	...	...	...	...

Open CV 기본

Mat 클래스

- IplImage 구조체
  > C 단점을 그대로 갖고 있음
  > 처리 속도 느림
  > 화소 데이터 타입에 따른 배열 인덱싱과 형변환 복잡
  > 메모리 관리 차원에서 사용 후 모두 직접 해제해 주어야하기 때문에 관리가 제대로 되지 않을 경우 메모리 누수 문제 발생

- Mat 클래스
  > 자료형 표현 쉬움
  > 화소에 대한 접근 방법 수월
  > 메모리 해제도 소멸자에서 처리하기 때문에 관리 또한 편리

https://docs.opencv.org/3.4.2/d3/d63/classcv_1_1Mat.html

Open CV 기본

Mat 클래스 예제

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
	float data[] = {
	  1.2f, 2.3f, 3.2f,
	  4.5f, 5.f, 6.5f,
	};
	// Mat 객체 선언 방법
	Mat m1(2, 3, CV_8U);
	Mat m2(2, 3, CV_8U, Scalar(300));
	Mat m3(2, 3, CV_16S, Scalar(300));
	Mat m4(2, 3, CV_32F, data);

	// Size 객체로 Mat 객체 선언 방법
	Size sz(2, 3);
	Mat m5(Size(2, 3), CV_64F);
	Mat m6(sz, CV_32F, data);

	cout << "[m1] =" << endl << m1 << endl;
	cout << "[m2] =" << endl << m2 << endl;
	cout << "[m3] =" << endl << m3 << endl;
	cout << "[m4] =" << endl << m4 << endl << endl;
	cout << "[m5] =" << endl << m5 << endl;
	cout << "[m6] =" << endl << m6 << endl;
	system("pause");
	return 0;
}
```

```
선택 D:₩OpenCV₩ex5_mat₩x64₩Debug₩ex5_mat.exe
[m1] =
[205, 205, 205;
 205, 205, 205]
[m2] =
[255, 255, 255;
 255, 255, 255]
[m3] =
[300, 300, 300;
 300, 300, 300]
[m4] =
[1.2, 2.3, 3.2;
 4.5, 5, 6.5]

[m5] =
[-6.277438562204192e+66, -6.277438562204192e+66;
 -6.277438562204192e+66, -6.277438562204192e+66;
 -6.277438562204192e+66, -6.277438562204192e+66]
[m6] =
[1.2, 2.3;
 3.2, 4.5;
 5, 6.5]
계속하려면 아무 키나 누르십시오 . . .
```

결과에서 확인할 수 있듯이
Size를 통해 행렬을 생성하면
행과 열 순서가 뒤바뀜

Open CV 실습

이미지 파일 처리

- 이미지 파일 읽기 (흑백) (169)

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void  print_matInfo(string name, Mat img)
{
string str;
int depth = img.depth();

if (depth == CV_8U) str = "CV_8U";
else if (depth == CV_8S) str = "CV_8S";
else if (depth == CV_16U) str = "CV_16U";
else if (depth == CV_16S) str = "CV_16S";
else if (depth == CV_32S) str = "CV_32S";
else if (depth == CV_32F) str = "CV_32F";
else if (depth == CV_64F) str = "CV_64F";

cout << name;
cout << format(": depth(%d) channels(%d) -> 자료형: ",
depth, img.channels());
cout << str << "C" << img.channels() << endl;
}
```

```
int main()
{
string filename1 = "image.jpg";
Mat gray2gray = imread(filename1, IMREAD_GRAYSCALE);
Mat gray2color = imread(filename1, IMREAD_COLOR);
CV_Assert(gray2gray.data && gray2color.data);

Rect roi(100, 100, 1, 1);
cout << "행렬 좌표 (100,100) 화소값 " << endl;
cout << "gray2gray " << gray2gray(roi) << endl;
cout << "gray2color " << gray2color(roi) << endl << endl;

print_matInfo("gray2gray", gray2gray);
print_matInfo("gray2color", gray2color);
imshow("gray2gray", gray2gray);
imshow("gray2color", gray2color);
waitKey(0);
return 0;
}
```

Open CV 실습

회색조 사진을 읽었을 때

컬러 사진을 읽었을 때

Open CV 실습

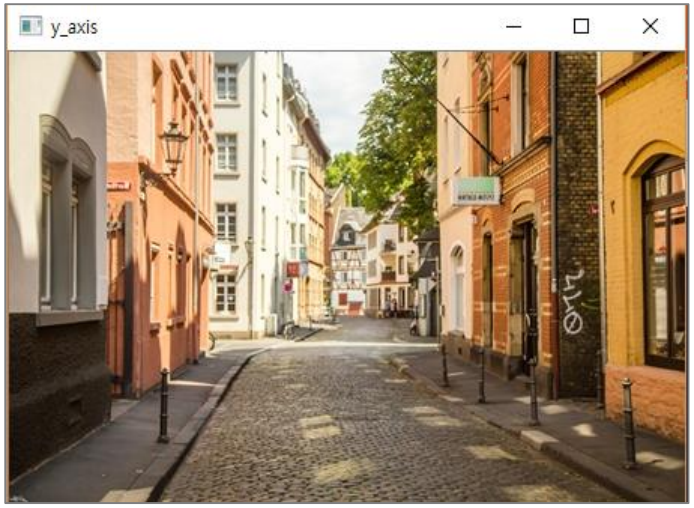
행렬 연산 (기본 배열 처리 함수)

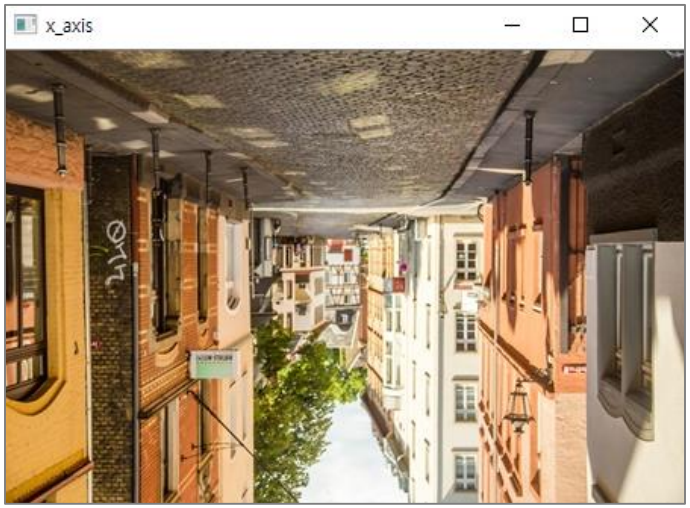
- flip : 입력된 2차원 배열을 수직, 수평, 양축으로 뒤집음
- repeat: 입력 배열의 반복된 복사본으로 출력배열을 채움
- transpose : 입력 행렬의 전치 행렬을 출력 인수로 반환

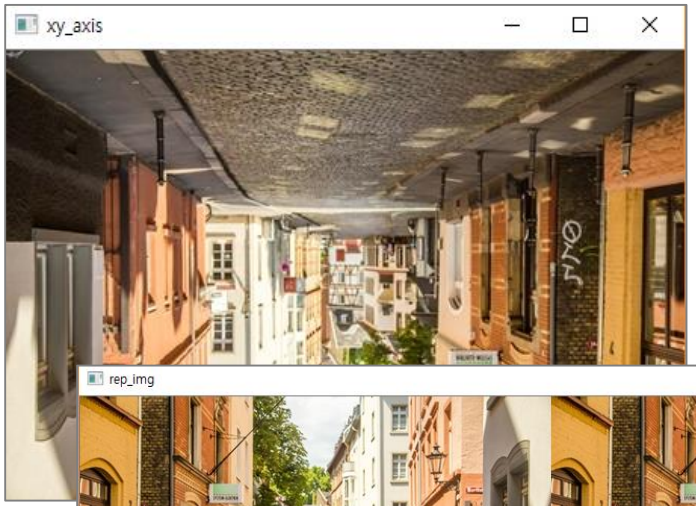
```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
        Mat image = imread("image.jpg",
        IMREAD_COLOR);
        CV_Assert(image.data);

        Mat x_axis, y_axis, xy_axis, rep_img, trans_img;
        flip(image, x_axis, 0);
        flip(image, y_axis, 1);
        flip(image, xy_axis, -1);

        repeat(image, 1, 2, rep_img);
        transpose(image, trans_img);
```

```
        imshow("image", image);
        imshow("x_axis", x_axis);
        imshow("y_axis", y_axis);
        imshow("xy_axis", xy_axis);
        imshow("rep_img", rep_img);
        imshow("trans_img", trans_img);

        waitKey();
        return 0;
}
```

Open CV 실습







Open CV 실습

행렬 연산 (산술 연산 함수)

- 사칙 연산: add, subtract, multiply, divide, addWeighted
- 지수 로그 루트 관련 함수: exp, log, sqrt, pow, magnitude, phase, cartToPolar, polarToChart
- 논리 연산함수: bitwise_and, gitwise_or, bitwise_xor, bitwise_not

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
  Mat m1(3, 6, CV_8UC1, Scalar(10));
  Mat m2(3, 6, CV_8UC1, Scalar(50));
  Mat m_add1, m_add2, m_sub, m_div1, m_div2;
  Mat mask(m1.size(), CV_8UC1);
  // 마스크 행렬 - 8비트 단일채널

  Rect rect(Point(3, 0), Size(3, 3));// 관심영역 지정
  mask(rect).setTo(1);

  add(m1, m2, m_add1);
  add(m1, m2, m_add2, mask);

  divide(m1, m2, m_div1);
  m1.convertTo(m1, CV_32F);
  // 형변환 - 소수점이하 값 보존
```

```
  m2.convertTo(m2, CV_32F);
  divide(m1, m2, m_div2);

  cout << "[m1] = " << endl << m1 << endl;
  cout << "[m2] = " << endl << m2 << endl;
  cout << "[mask] = " << endl << mask << endl << endl;

  cout << "[m_add1] = " << endl << m_add1 << endl;
  cout << "[m_add2] = " << endl << m_add2 << endl;
  cout << "[m_div1] = " << endl << m_div1 << endl;
  cout << "[m_div2] = " << endl << m_div2 << endl;
  system("pause");
  return 0;
}
```

Open CV 실습

행렬 연산 (산술 연산 함수)

```
선택 D:₩OpenCV₩ex8_matOp₩x64₩Debug₩ex8_matOp.exe
[m1] =
[10, 10, 10, 10, 10, 10;
 10, 10, 10, 10, 10, 10;
 10, 10, 10, 10, 10, 10]
[m2] =
[50, 50, 50, 50, 50, 50;
 50, 50, 50, 50, 50, 50;
 50, 50, 50, 50, 50, 50]
[mask] =
[205, 205, 205,   1,   1,   1;
 205, 205, 205,   1,   1,   1;
 205, 205, 205,   1,   1,   1]

[m_add1] =
[ 60,  60,  60,  60,  60,  60;
  60,  60,  60,  60,  60,  60;
  60,  60,  60,  60,  60,  60]
[m_add2] =
[ 60,  60,  60,  60,  60,  60;
  60,  60,  60,  60,  60,  60;
  60,  60,  60,  60,  60,  60]
[m_div1] =
[  0,   0,   0,   0,   0,   0;
   0,   0,   0,   0,   0,   0;
   0,   0,   0,   0,   0,   0]
[m_div2] =
[0.2, 0.2, 0.2, 0.2, 0.2, 0.2;
 0.2, 0.2, 0.2, 0.2, 0.2, 0.2;
 0.2, 0.2, 0.2, 0.2, 0.2, 0.2]
계속하려면 아무 키나 누르십시오 . . .
```

Open CV 실습

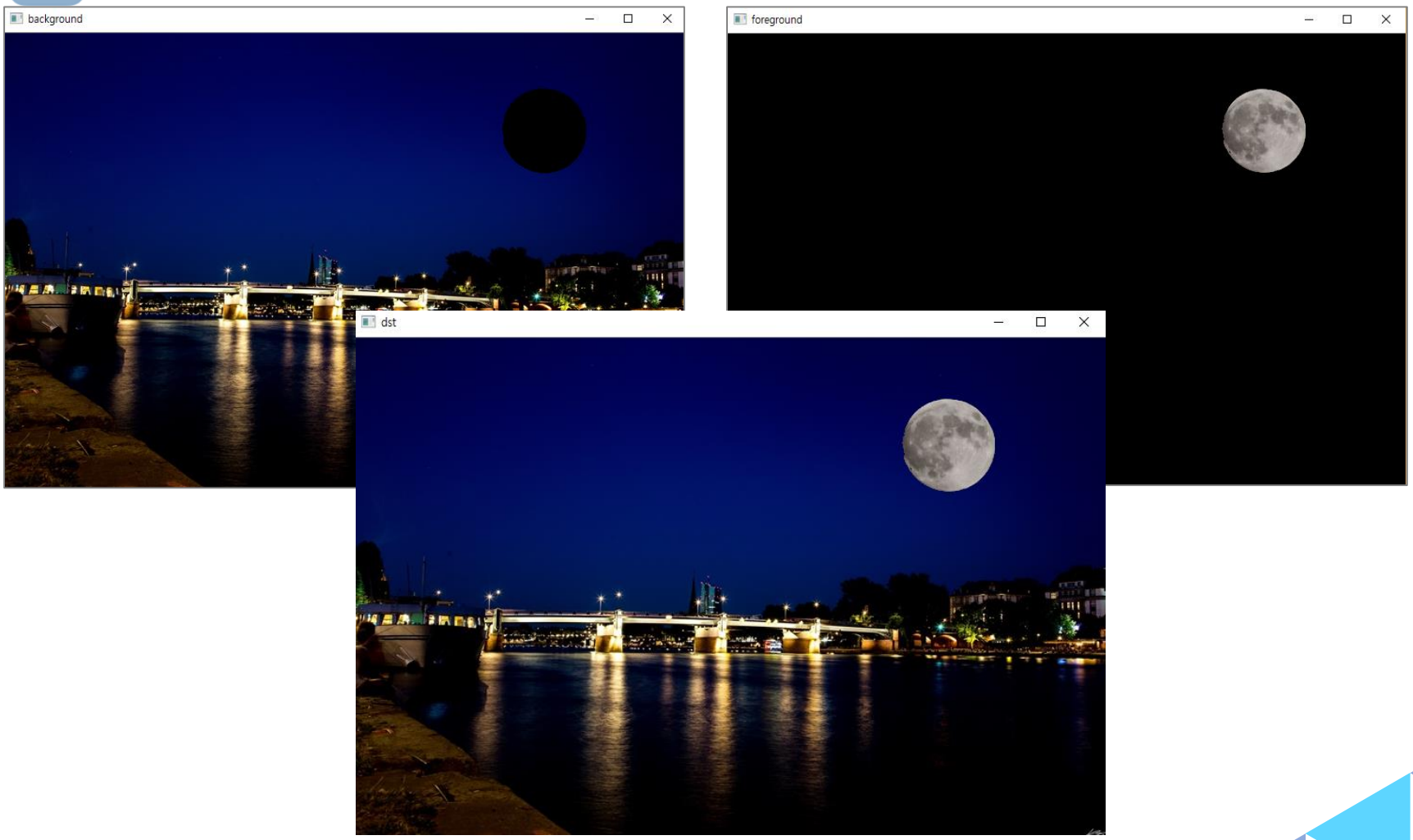
논리 연산 오버랩 예제

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
  Mat image = imread("river.jpg", IMREAD_COLOR);
  Mat logo = imread("moon.jpg", IMREAD_COLOR);
  CV_Assert(image.data && logo.data);// 예외처리
  Mat logo_th, masks[5], background, foreground, dst;
  // 결과 행렬

  threshold(logo, logo_th, 70, 255, THRESH_BINARY);
  // 로고 영상 이진화
  split(logo_th, masks);// 로고영상 채널 분리

  bitwise_or(masks[0], masks[1], masks[3]);// 전경통과 마스크
  bitwise_or(masks[2], masks[3], masks[3]);
  bitwise_not(masks[3], masks[4]);// 배경통과 마스크
```

```
  Point center1 = image.size() / 2;// 영상 중심좌표
  Point center2 = logo.size() / 2;;// 로고 중심좌표
  Point start = center1 - center2;
  Rect roi(start, logo.size());// 로고가 위치할 관심영역

  //비트곱과 마스킹을 이용한 관심 영역의 복사
  bitwise_and(logo, logo, foreground, masks[3]);
  bitwise_and(image(roi), image(roi), background, masks[4]);

  add(background, foreground, dst);// 로고 전경과 배경 합성
  dst.copyTo(image(roi));// 로고 합성 영상을 관심영역에 복사

  imshow("background", background);
  imshow("foreground", foreground);
  imshow("dst", dst);
  imshow("image", image);
  waitKey();
  return 0;
}
```

Open CV 실습

논리 연산 오버랩 예제



Open CV 실습

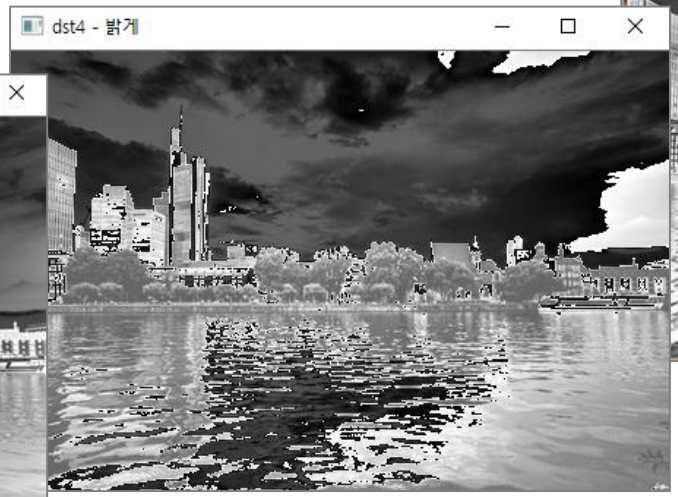
영상 처리 (영상 밝기)

- 화소 값이 영상 밝기를 나타내기 때문에 이 화소 값을 변경하면 영상 밝기를 변경할 수 있음

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
  Mat image = imread("image.jpg", IMREAD_GRAYSCALE);
  CV_Assert(!image.empty());

  Mat dst1 = image + 100;
  Mat dst2 = image - 100;
  Mat dst3 = 255 - image;

  Mat dst4(image.size(), image.type());
  Mat dst5(image.size(), image.type());
```

```
  for (int i = 0; i < image.rows; i++) {
    for (int j = 0; j < image.cols; j++)
    {
      dst4.at<uchar>(i, j) = image.at<uchar>(i, j) + 100;
      //dst4.at<uchar>(i, j) =
saturate_cast<uchar>(image.at<uchar>(i, j) + 100);
      dst5.at<uchar>(i, j) = 255 - image.at<uchar>(i, j);
    }
  }

  imshow("원 영상", image);
  imshow("dst1 - 밝게", dst1);
  imshow("dst2 - 어둡게", dst2);
  imshow("dst3 - 반전", dst3);
  imshow("dst4 - 밝게", dst4);
  imshow("dst5 - 반전", dst5);
  waitKey();
  return 0;
}
```

Open CV 실습



dst4 밝게는 255가 넘기 때문에 밝기가 제대로 되지 않음

Open CV 실습

영상 처리 (명암 대비)

- 명암 대비는 상이한 두 가지 색이 경계에서 서로 영향을 미쳐 그 차이가 강조되어 나타나는 현상

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
  Mat image = imread("image.jpg", 0);// 명암도 타입 읽기
  CV_Assert(image.data);// 예외처리

  Scalar avg = mean(image) / 2.0;// 원본 영상 화소 평균의 절반
  Mat dst1 = image * 0.5;// 명암대비 감소
  Mat dst2 = image * 2.0;// 명암대비 증가
  Mat dst3 = image * 0.5 + avg[0];// 영상 평균 이용 대비 감소
  Mat dst4 = image * 2.0 - avg[0];// 영상 평균 이용 대비 증가

  imshow("image", image);
  imshow("dst1-대비감소", dst1), imshow("dst2-대비증가", dst2);
  imshow("dst3-평균이용 대비감소", dst3), imshow("dst4-평균이용 대비증가", dst4);

  waitKey();
  return 0;
}
```

Open CV 실습

영상 처리 (명암 대비 예제)

Open CV 실습

영상 처리 (히스토그램)

- 화소 접근, 밝기 변환, 히스토그램, 컬러 공간

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void calc_histo(Mat image, Mat &hist, int bins, int range_max = 256)
{
  hist = Mat(bins, 1, CV_32F, Scalar(0));
  float gap = range_max / (float)bins;

  for (int i = 0; i < image.rows; i++) {
    for (int j = 0; j < image.cols; j++)
    {
    int idx = int(image.at<uchar>(i, j) / gap);
    hist.at<float>(idx)++;
    }
  }
}
```

```
int main()
{
  Mat image = imread("image.jpg", IMREAD_GRAYSCALE);
  CV_Assert(!image.empty());

  Mat hist;
  calc_histo(image, hist, 256);// 히스토그램 계산
  cout << hist.t() << endl;

  imshow("image", image);
  waitKey();
  return 0;
}
```

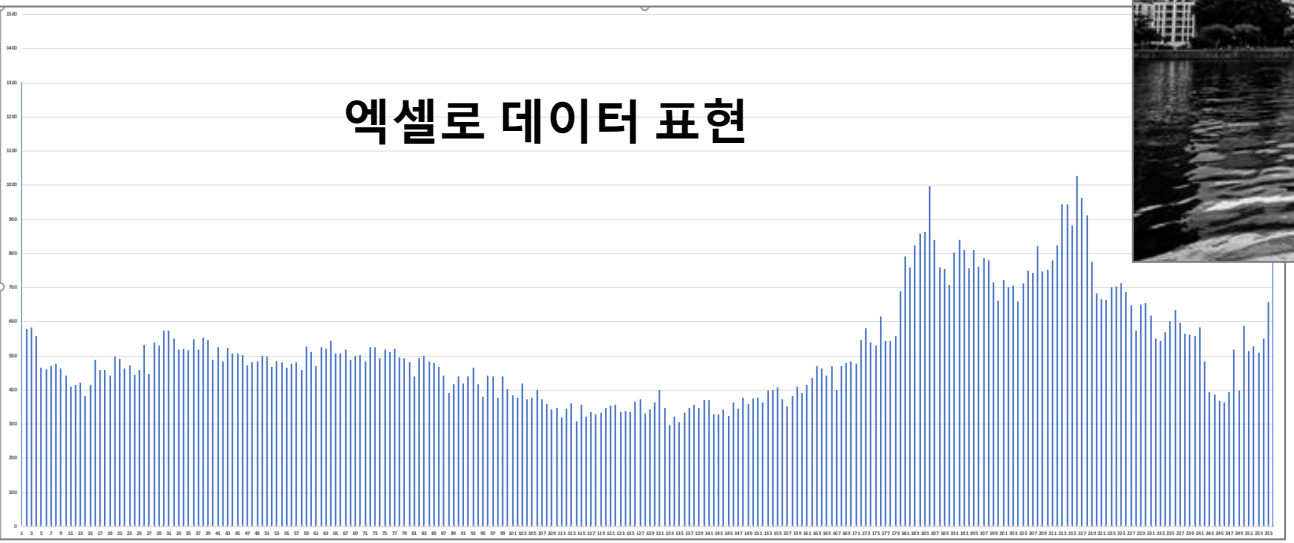
Open CV 실습

D:₩OpenCV₩ex12_histogramCalc₩x64₩Debug₩ex12_histogramCalc.exe

```
[1303, 578, 584, 557, 465, 462, 470, 476, 464, 443, 411, 414, 421, 382, 415, 489, 459, 459, 442, 498, 492, 464, 473, 445
, 458, 533, 447, 539, 531, 575, 575, 550, 519, 521, 517, 548, 519, 553, 547, 489, 526, 485, 524, 508, 507, 502, 472, 481
, 483, 501, 499, 468, 485, 482, 466, 478, 481, 458, 527, 511, 471, 526, 520, 543, 508, 507, 519, 489, 501, 503, 484, 525
, 525, 493, 518, 511, 521, 496, 494, 481, 439, 494, 500, 484, 480, 467, 443, 391, 418, 439, 420, 440, 465, 418, 380, 442
, 440, 377, 439, 402, 385, 377, 420, 372, 378, 401, 373, 359, 342, 348, 320, 345, 362, 308, 357, 323, 336, 328, 333, 348
, 355, 358, 335, 339, 335, 365, 373, 332, 342, 364, 401, 347, 297, 323, 306, 334, 347, 358, 348, 370, 370, 329, 329, 343
, 325, 363, 345, 377, 359, 376, 377, 364, 399, 401, 408, 373, 352, 383, 411, 391, 415, 436, 471, 464, 442, 469, 401, 470
, 479, 484, 476, 546, 580, 539, 530, 616, 544, 545, 559, 689, 792, 759, 824, 858, 863, 997, 841, 758, 755, 708, 803, 839
, 810, 756, 811, 761, 788, 779, 715, 661, 721, 701, 705, 659, 714, 750, 744, 821, 747, 752, 780, 823, 945, 943, 881, 102
8, 962, 911, 776, 684, 666, 664, 702, 704, 712, 687, 649, 575, 651, 656, 619, 552, 543, 570, 601, 634, 598, 565, 563, 55
7, 584, 483, 395, 386, 369, 363, 394, 518, 399, 588, 515, 529, 510, 550, 657, 2549]
```

image

엑셀로 데이터 표현



Open CV 실습

영상 처리 (히스토그램 – 함수 활용)

- OpenCV 에서 제공하는 히스토그램 계산 함수
  cv:calcHist()
- 다채널 행렬에서 다차원의 히스토그램을 구하기 때문에 인수와 구조 복잡

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void  calc_Histo(const Mat& image, Mat& hist, int bins, int   range_max = 256)
{
  inthistSize[] = { bins };// 히스토그램 계급개수
  float   range[] = { 0, (float)range_max };// 채널 화소값 범위
  intchannels[] = { 0 };// 채널 목록
  const float* ranges[] = { range };

  calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges);
}
```

```
int main()
{
  Mat image = imread("image.jpg", IMREAD_GRAYSCALE);
  CV_Assert(!image.empty());

  Mat hist;
  calc_histo(image, hist, 256);// 히스토그램 계산
  cout << hist.t() << endl;

  imshow("image", image);
  waitKey();
  return 0;
}
```

Open CV 실습

영상 처리 (히스토그램 – 함수 활용)


```
D:₩OpenCV₩ex13_histogram2₩x64₩Debug₩ex13_histogram2.exe
[1303, 578, 584, 557, 465, 462, 470, 476, 464, 443, 411, 414, 421, 382, 415, 489, 459, 459, 442, 498, 492, 464, 473, 445
, 458, 533, 447, 539, 531, 575, 575, 550, 519, 521, 517, 548, 519, 553, 547, 489, 526, 485, 524, 508, 507, 502, 472, 481
, 483, 501, 499, 468, 485, 482, 466, 478, 481, 458, 527, 511, 471, 526, 520, 543, 508, 507, 519, 489, 501, 503, 484, 525
, 525, 493, 518, 511, 521, 496, 494, 481, 439, 494, 500, 484, 480, 467, 443, 391, 418, 439, 420, 440, 465, 418, 380, 442
, 440, 377, 439, 402, 385, 377, 420, 372, 378, 401, 373, 359, 342, 348, 320, 345, 362, 308, 357, 323, 336, 328, 333, 348
, 355, 358, 335, 339, 335, 365, 373, 332, 342, 364, 401, 347, 297, 323, 306, 334, 347, 358, 348, 370, 370, 329, 329, 343
, 325, 363, 345, 377, 359, 376, 377, 364, 399, 401, 408, 373, 352, 383, 411, 391, 415, 436, 471, 464, 442, 469, 401, 470
, 479, 484, 476, 546, 580, 539, 530, 616, 544, 545, 559, 689, 792, 759, 824, 858, 863, 997, 841, 758, 755, 708, 803, 839
, 810, 756, 811, 761, 788, 779, 715, 661, 721, 701, 705, 659, 714, 750, 744, 821, 747, 752, 780, 823, 945, 943, 881, 102
8, 962, 911, 776, 684, 666, 664, 702, 704, 712, 687, 649, 575, 651, 656, 619, 552, 543, 570, 601, 634, 598, 565, 563, 55
7, 584, 483, 395, 386, 369, 363, 394, 518, 399, 588, 515, 529, 510, 550, 657, 2549]
```

Open CV 실습

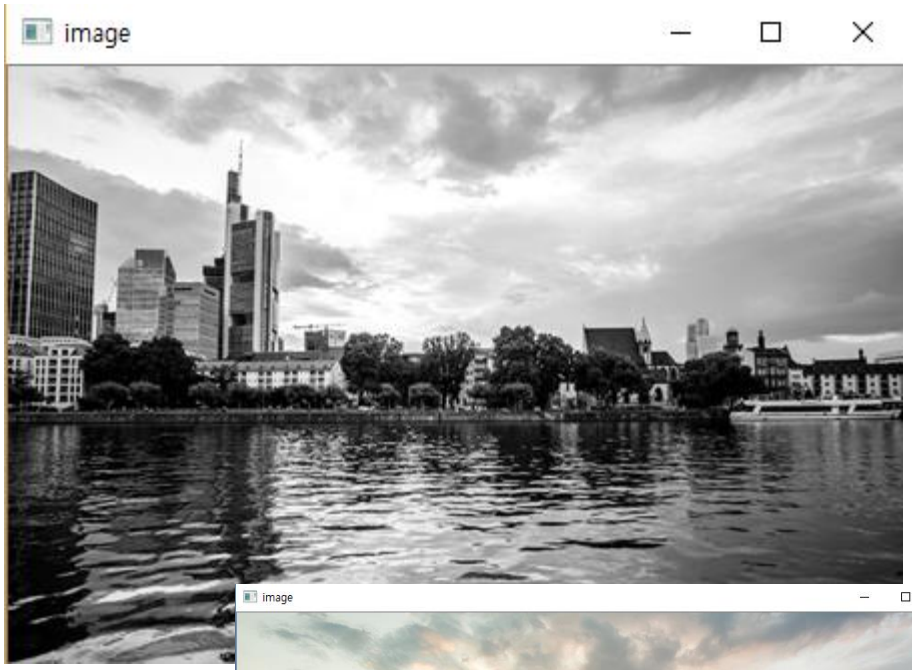
영상 처리 (히스토그램 그리기)

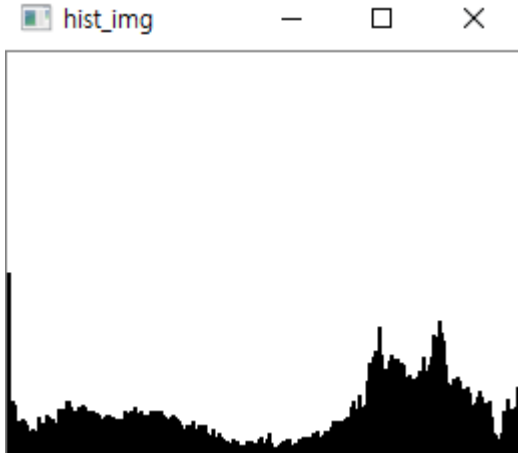
```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void  calc_Histo(const Mat& image, Mat& hist, int bins, int range_max = 256)
{
  int histSize[] = { bins };// 히스토그램 계급개수
  float   range[] = { 0, (float)range_max };// 히스토그램 범위
  int channels[] = { 0 };// 채널 목록
  const float* ranges[] = { range };

  calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges);
}

void draw_histo(Mat hist, Mat &hist_img, Size size = Size(256, 200))
{
  hist_img = Mat(size, CV_8U, Scalar(255));
  float  bin = (float)hist_img.cols / hist.rows;
  normalize(hist, hist, 0, size.height, NORM_MINMAX);

  for (int i = 0; i < hist.rows; i++){
    float  start_x = (i * bin);
    float  end_x = (i + 1) * bin;
    Point2f pt1(start_x, 0);
    Point2f pt2(end_x, hist.at <float>(i));

    if (pt2.y > 0)
      rectangle(hist_img, pt1, pt2, Scalar(0), -1);
  }
  flip(hist_img, hist_img, 0);
}
```

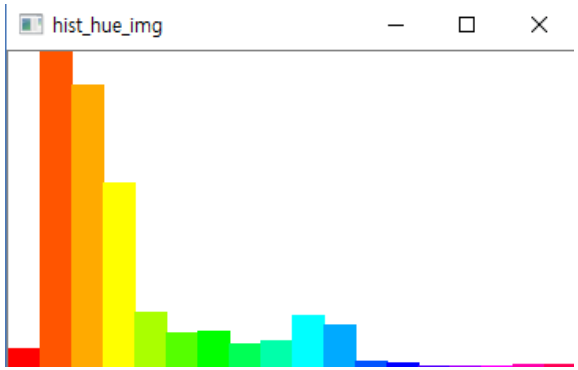
```
int main()
{
Mat image = imread("image.jpg", IMREAD_GRAYSCALE);
CV_Assert(!image.empty());

Mat hist, hist_img;
calc_Histo(image, hist, 256);
draw_histo(hist, hist_img);

imshow("hist_img", hist_img);
imshow("image", image);
waitKey();
return 0;
}
```

Open CV 실습

영상 처리 (히스토그램 그리기)







색상 히스토그램

Open CV 실습

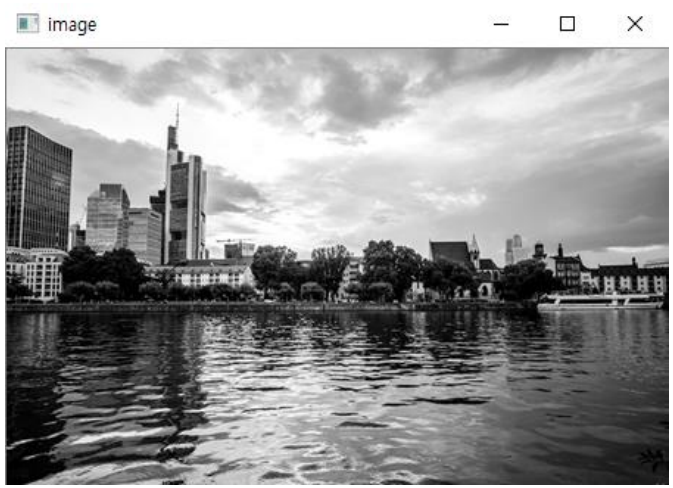
영상 처리 (히스토그램 평활화)

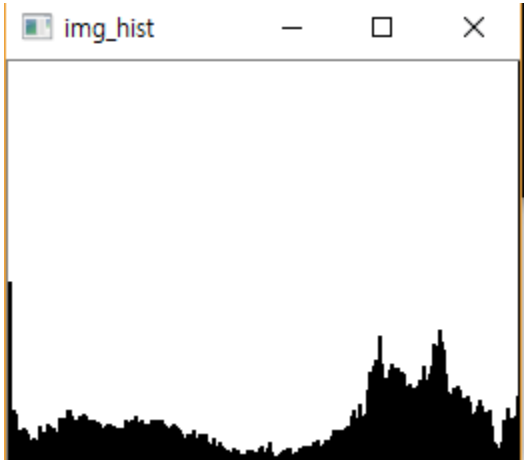
```
void  calc_Histo(const Mat& image, Mat& hist, int bins, int range_max = 256){...}
void draw_histo(Mat hist, Mat &hist_img, Size size = Size(256, 200)){...}
void create_hist(Mat img, Mat &hist, Mat &hist_img){...}
int main(){
  Mat image = imread("image.jpg", 0);// 명암도 영상 읽기
  CV_Assert(!image.empty());// 영상파일 예외처리

  Mat hist, dst1, dst2, hist_img, hist_img1, hist_img2;
  create_hist(image, hist, hist_img);
  // 히스토그램 및 그래프 그리기

  // 히스토그램 누적합 계산
  Mat accum_hist = Mat(hist.size(), hist.type(), Scalar(0));
  accum_hist.at<float>(0) = hist.at<float>(0);
  for (int i = 1; i < hist.rows; i++) {
    accum_hist.at<float>(i) = accum_hist.at<float>(i - 1) + hist.at<float>(i);
  }

  accum_hist /= sum(hist)[0];// 누적합의 정규화
  accum_hist *= 255;
  dst1 = Mat(image.size(), CV_8U);
  for (int i = 0; i < image.rows; i++) {
    for (int j = 0; j < image.cols; j++) {
      int idx = image.at<uchar>(i, j);
      dst1.at<uchar>(i, j) = (uchar)accum_hist.at<float>(idx);
    }
  }
```

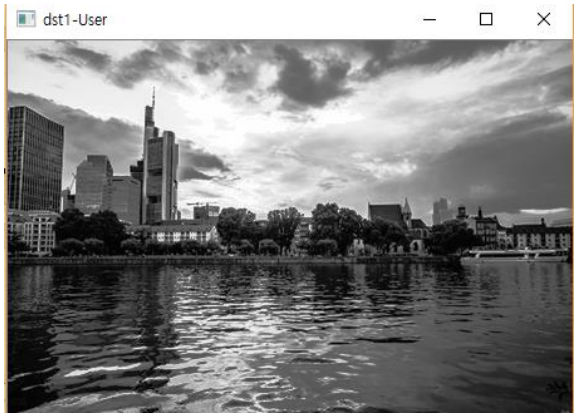
```
//normalize(accum_hist, accum_hist, 0, 255,
NORM_MINMAX);// 누적합의 정규화
//accum_hist.convertTo(accum_hist, CV_8U);
//LUT(image, accum_hist, dst1);// 룩업 테이블 적용

equalizeHist(image, dst2);// 히스토그램 평활화
create_hist(dst1, hist, hist_img1);
// 히스토그램 및 그래프 그리기
create_hist(dst2, hist, hist_img2);

imshow("image", image), imshow("img_hist", hist_img);
// 원본 히스토그램
imshow("dst1-User", dst1), imshow("User_hist", hist_img1);
// 사용자 평활화
imshow("dst2-OpenCV", dst2), imshow("OpenCV_hist",
hist_img2);
// OpenCV 평활화
waitKey();
return 0;
}
```

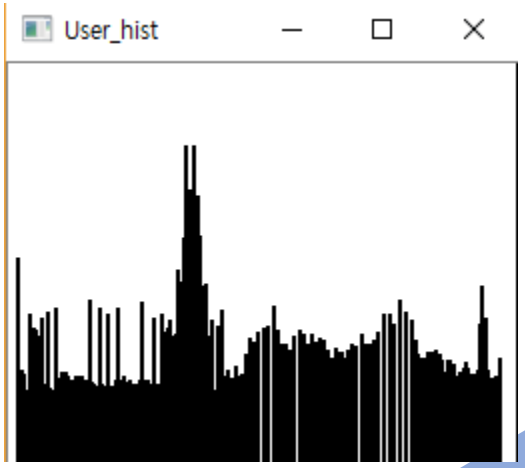
Open CV 실습

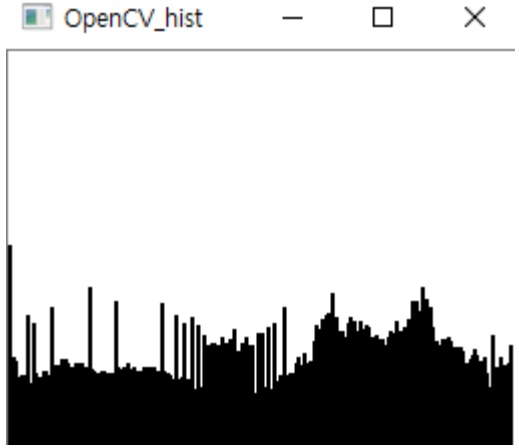
영상 처리 (히스토그램 평활화)











Open CV 실습

영역 처리 (블러링)

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void filter(Mat img, Mat& dst, Mat mask)// 회선 수행 함수
{
  dst = Mat(img.size(), CV_32F, Scalar(0));// 회선 결과 저장 행렬
  Point h_m = mask.size() / 2;// 마스크 중심 좌표

  for (int i = h_m.y; i < img.rows - h_m.y; i++) {// 입력 행렬 반복 순회
    for (int j = h_m.x; j < img.cols - h_m.x; j++)
    {
      float sum = 0;
      for (int u = 0; u < mask.rows; u++) {// 마스크 원소 순회
        for (int v = 0; v < mask.cols; v++)
        {
          int y = i + u - h_m.y;
          int x = j + v - h_m.x;
          sum += mask.at<float>(u, v) * img.at<uchar>(y, x);// 회선 수식
        }
      }
    dst.at<float>(i, j) = sum;// 회선 누적값 출력화소 저장
    }
  }
}
int main()
{
  Mat image = imread("image.jpg", IMREAD_GRAYSCALE);
  CV_Assert(image.data);// 영상파일 예외 처리

  float data[] = {// 블러링 마스크 원소 지정
    1/9.f, 1/9.f, 1 / 9.f,
    1 / 9.f, 1 / 9.f, 1 / 9.f,
    1 / 9.f, 1 / 9.f, 1 / 9.f
  };

  Mat mask1(3, 3, CV_32F, data);// 마스크 행렬 선언
  Mat blur;
  filter(image, blur , mask1);// 회선 수행
  blur.convertTo(blur, CV_8U);// 자료형 변환

  imshow("blur", blur);// 결과 행렬 윈도우에 표시
  imshow("image", image);
  waitKey();
  return 0;
}
```

Open CV 실습

영역 처리 (블러링)

Open CV 실습

영역 처리 (샤프닝)

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void filter(Mat img, Mat& dst, Mat mask)// 회선 수행 함수
{
  dst = Mat(img.size(), CV_32F, Scalar(0));// 회선 결과 저장 행렬
  Point h_m = mask.size() / 2;// 마스크 중심 좌표

  for (int i = h_m.y; i < img.rows - h_m.y; i++) {// 입력 행렬 반복 순회
    for (int j = h_m.x; j < img.cols - h_m.x; j++)
    {
      float sum = 0;
      for (int u = 0; u < mask.rows; u++) {// 마스크 원소 순회
        for (int v = 0; v < mask.cols; v++)
        {
          int y = i + u - h_m.y;
          int x = j + v - h_m.x;
          sum += mask.at<float>(u, v) * img.at<uchar>(y, x);// 회선 수식
        }
      }
    dst.at<float>(i, j) = sum;// 회선 누적값 출력화소 저장
    }
  }
}
int main()
{
  Mat image = imread("image.jpg", IMREAD_GRAYSCALE);
  CV_Assert(image.data);// 영상파일 예외 처리
```

```
float data1[] = {// 사프닝 마스크 원소 지정
0, -1, 0,
-1, 5, -1,
0, -1, 0,
};
float data2[] = {// 사프닝 마스크 원소 지정
-1, -1, -1,
-1, 9, -1,
-1, -1, -1,
};

  Mat mask1(3, 3, CV_32F, data1);// 마스크 행렬 선언
  Mat mask2(3, 3, CV_32F, data2);
  Mat sharpen1, sharpen2;
  filter(image, sharpen1, mask1);// 회선 수행
  filter(image, sharpen2, mask2);
  sharpen1.convertTo(sharpen1, CV_8U);// 자료형 변환
  sharpen2.convertTo(sharpen2, CV_8U);

  imshow("sharpen1", sharpen1);
  // 결과 행렬 윈도우에 표시
  imshow("sharpen2", sharpen2);
  imshow("image", image);
  waitKey();
  return 0;
}
```

Open CV 실습

영역 처리 (샤프닝)

Open CV 실습

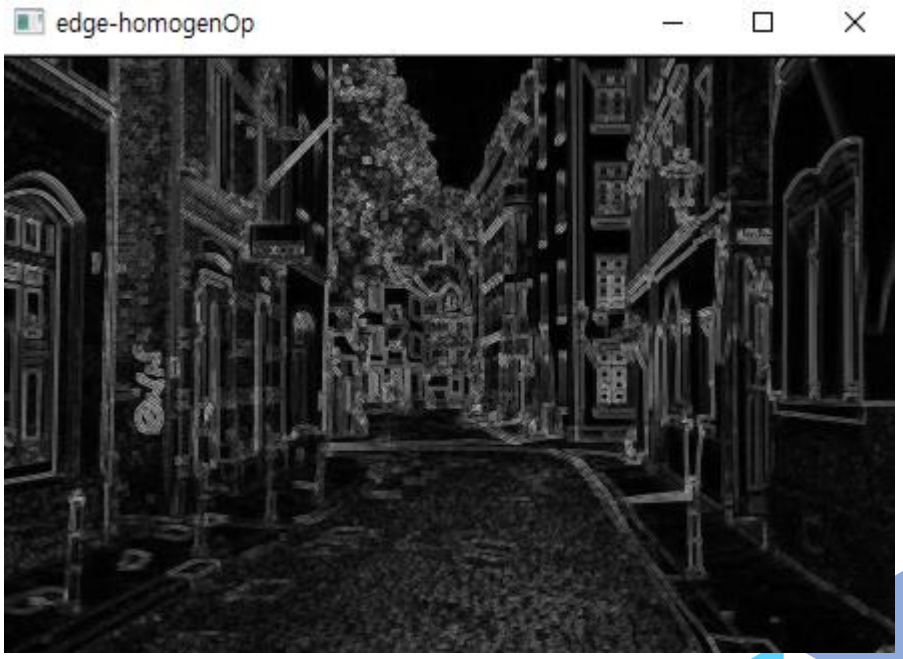
영역 처리 (엣지 검출 – 유사연산자)

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void homogenOp(Mat img, Mat& dst, int mask_size)
{
  dst = Mat(img.size(), CV_8U, Scalar(0));
  Point h_m(mask_size / 2, mask_size / 2);

  for (int i = h_m.y; i < img.rows - h_m.y; i++) {
    for (int j = h_m.x; j < img.cols - h_m.x; j++){
      uchar max = 0;
      for (int u = 0; u < mask_size; u++) {
        for (int v = 0; v < mask_size; v++){
          int y = i + u - h_m.y;
          int x = j + v - h_m.x;
          uchar difference = abs(img.at<uchar>(i, j) - img.at<uchar>(y, x));
          if (difference > max) max = difference;
        }
      }
      dst.at<uchar>(i, j) = max;
    }
  }
}
```

```
int main()
{
  Mat image = imread("image.jpg",
IMREAD_GRAYSCALE);
  CV_Assert(image.data);

  Mat edge;
  homogenOp(image, edge, 3);

  imshow("image", image);
  imshow("edge-homogenOp", edge);
  waitKey();
  return 0;
}
```

Open CV 실습

영역 처리 (엣지 검출 – 유사연산자)



Open CV 실습

영역 처리 (엣지 검출 – 프리윗/소벨)

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void differential(Mat image, Mat& dst, float data1[], float
data2[])
{
Mat dst1, mask1(3, 3, CV_32F, data1);
Mat dst2, mask2(3, 3, CV_32F, data2);

filter2D(image, dst1, CV_32F, mask1);
// OpenCV 제공 회선 함수
filter2D(image, dst2, CV_32F, mask2);
magnitude(dst1, dst2, dst);
dst.convertTo(dst, CV_8U);

convertScaleAbs(dst1, dst1);// 절대값 및 형변환 동시 수행 함수
convertScaleAbs(dst2, dst2);
imshow("dst1 - 수직 마스크", dst1);// 윈도우 행렬 표시
imshow("dst2 - 수평 마스크", dst2);
}


int main()
{
Mat image = imread("../image/edge_test1.jpg",
IMREAD_GRAYSCALE);
CV_Assert(image.data);
```

```
float data1[] = {
-1, 0, 1,
-2, 0, 2,
-1, 0, 1
};
float data2[] = {
-1, -2, -1,
0, 0, 0,
1, 2, 1
};
```

소벨 마스크

```
Mat dst, dst3, dst4;
differential(image, dst, data1, data2);
// 두 방향 소벨 회선 및 크기 계산

// OpenCV 제공 소벨 에지 계산
Sobel(image, dst3, CV_32F, 1, 0, 3);// x방향 미분 - 수직 마스크
Sobel(image, dst4, CV_32F, 0, 1, 3);// y방향 미분 - 수평 마스크
convertScaleAbs(dst3, dst3);// 절대값 및 uchar 형변환
convertScaleAbs(dst4, dst4);

imshow("image", image), imshow("소벨에지", dst);
imshow("dst3-수직_OpenCV", dst3), imshow("dst4-수평
_OpenCV", dst4);

waitKey();
return 0;
}
```

프리윗 마스크

```
float data1[] = {
-1, 0, 1,
-1, 0, 1,
-1, 0, 1
};
float data2[] = {
-1, -1, -1,
0, 0, 0,
1, 1, 1
};
```

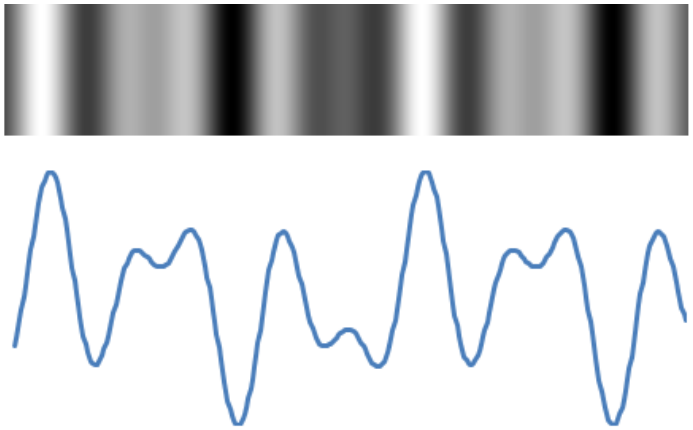
Open CV 실습

변환 영역 처리

- 영상 데이터는 화소값이 직접 표현된 공간영역과 우주 공간과 같은 변환영역 크게 2가지 영역으로 구분
- 변환영역은 직교변환에 의해 얻어진 영상 데이터의 다른 표현

공간 주파수

- 영상 처리에서는 공간 주파수라는 개념 사용
- 예) 화소 밝기의 변화도에 따른 빈도를 주파수로 표현



Open CV 실습

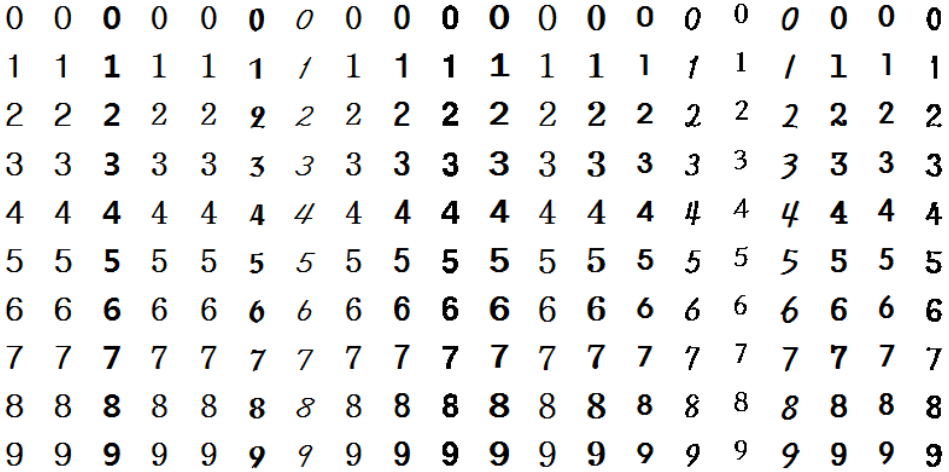
공간 주파수

- 저주파 공간 영역
  > 화소 밝기가 거의 변화가 없거나 점진적으로 변화하는 것
  > 영상에서 배경 또는 객체의 내부
- 고주파 공간 영역
  > 화소 밝기가 급변하는 것
  > 경계부분이나 객체의 모서리 부분

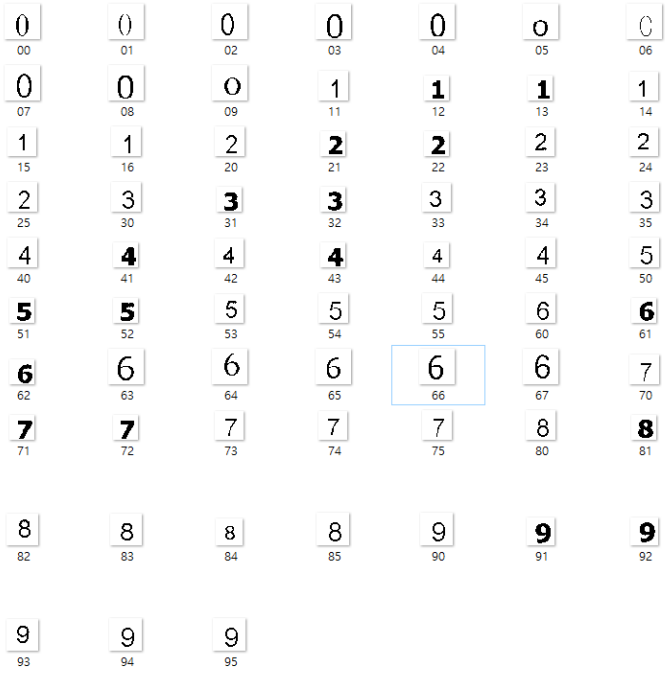
Open CV 실습

영상 분할 (k-최근접 이웃 분류기)

- 기존 가지고 있는 데이터들을 일정한 규칙에 의해 분류된 상태에서 새로운 입력 데이터의 종류를 예측하는 분류 알고리즘



트레이닝 데이터

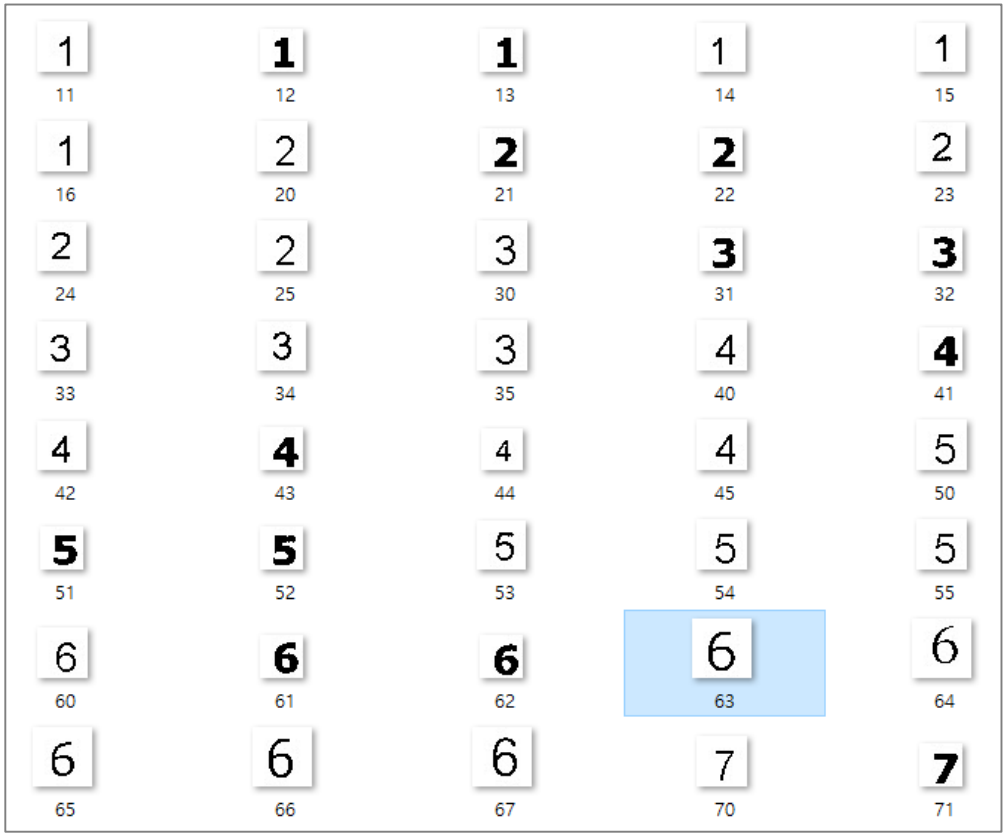


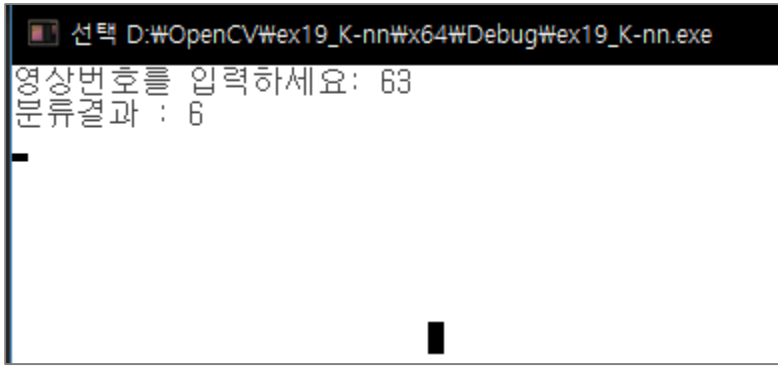
실험 데이터

숫자 인식 예제

Open CV 실습

영상 분할 (k-최근접 이웃 분류기)



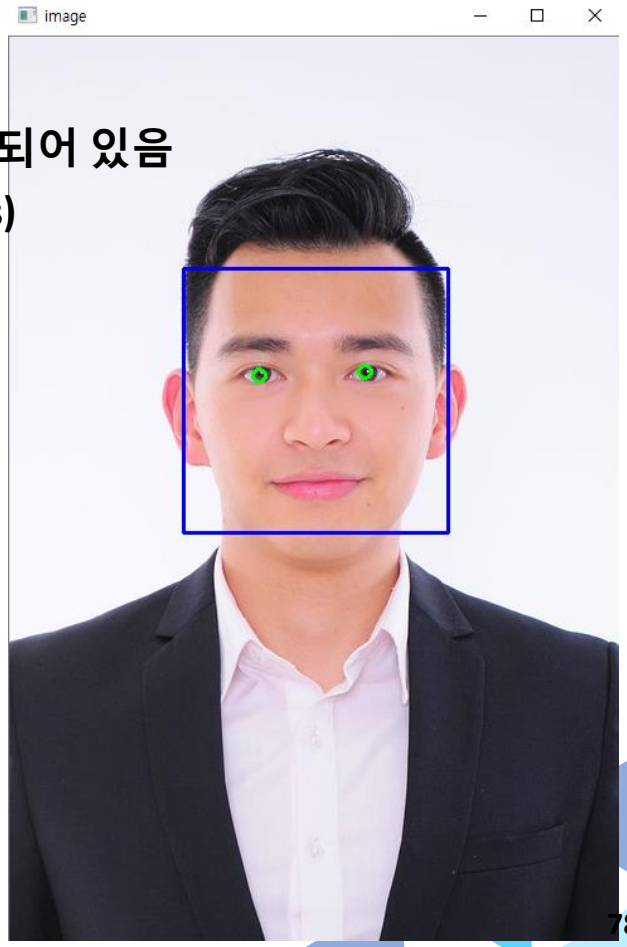


Open CV 실습

응용 사례 (얼굴 검출)

- 하르 기반 검출기 이용 얼굴 / 눈을 검출
- 하르 기반 분류기
  - > 2001년 Viola와 Jones가 제안한 방법
  - > OpenCV 3 버전 디렉토리에 검출기 내용이 이미 포함되어 있음
    (본 강의 기준 : C:\opencv\sources\data\haarcascades)

이름	수정한 날짜	유형	크기
haarcascade_eye	2018-04-11 오전...	XML 문서	334KB
haarcascade_eye_tree_eyeglasses	2018-04-11 오전...	XML 문서	588KB
haarcascade_frontalcatface	2018-04-11 오전...	XML 문서	402KB
haarcascade_frontalcatface_extended	2018-04-11 오전...	XML 문서	374KB
haarcascade_frontalface_alt	2018-04-11 오전...	XML 문서	661KB
haarcascade_frontalface_alt_tree	2018-04-11 오전...	XML 문서	2,627KB
haarcascade_frontalface_alt2	2018-04-11 오전...	XML 문서	528KB
haarcascade_frontalface_default	2018-04-11 오전...	XML 문서	909KB
haarcascade_fullbody	2018-04-11 오전...	XML 문서	466KB
haarcascade_lefteye_2splits	2018-04-11 오전...	XML 문서	191KB
haarcascade_licence_plate_rus_16stages	2018-04-11 오전...	XML 문서	47KB
haarcascade_lowerbody	2018-04-11 오전...	XML 문서	387KB
haarcascade_profileface	2018-04-11 오전...	XML 문서	810KB
haarcascade_righteye_2splits	2018-04-11 오전...	XML 문서	192KB
haarcascade_russian_plate_number	2018-04-11 오전...	XML 문서	74KB
haarcascade_smile	2018-04-11 오전...	XML 문서	185KB
haarcascade_upperbody	2018-04-11 오전...	XML 문서	768KB



Open CV 실습

응용 사례 (성별 검출)

```
D:₩OpenCV₩ex20_1_face₩x64₩Debug₩ex20_1_face.exe
[ INFO:0] Initialize OpenCL runtime...
60.jpg: Man - 유사도 [머리: 0.07  입술: 0.22]
```