

INSTALACJA OPENCV v 3.2

Kody źródłowe i wersje release dostępne są pod adresem: <http://opencv.org/releases.html>

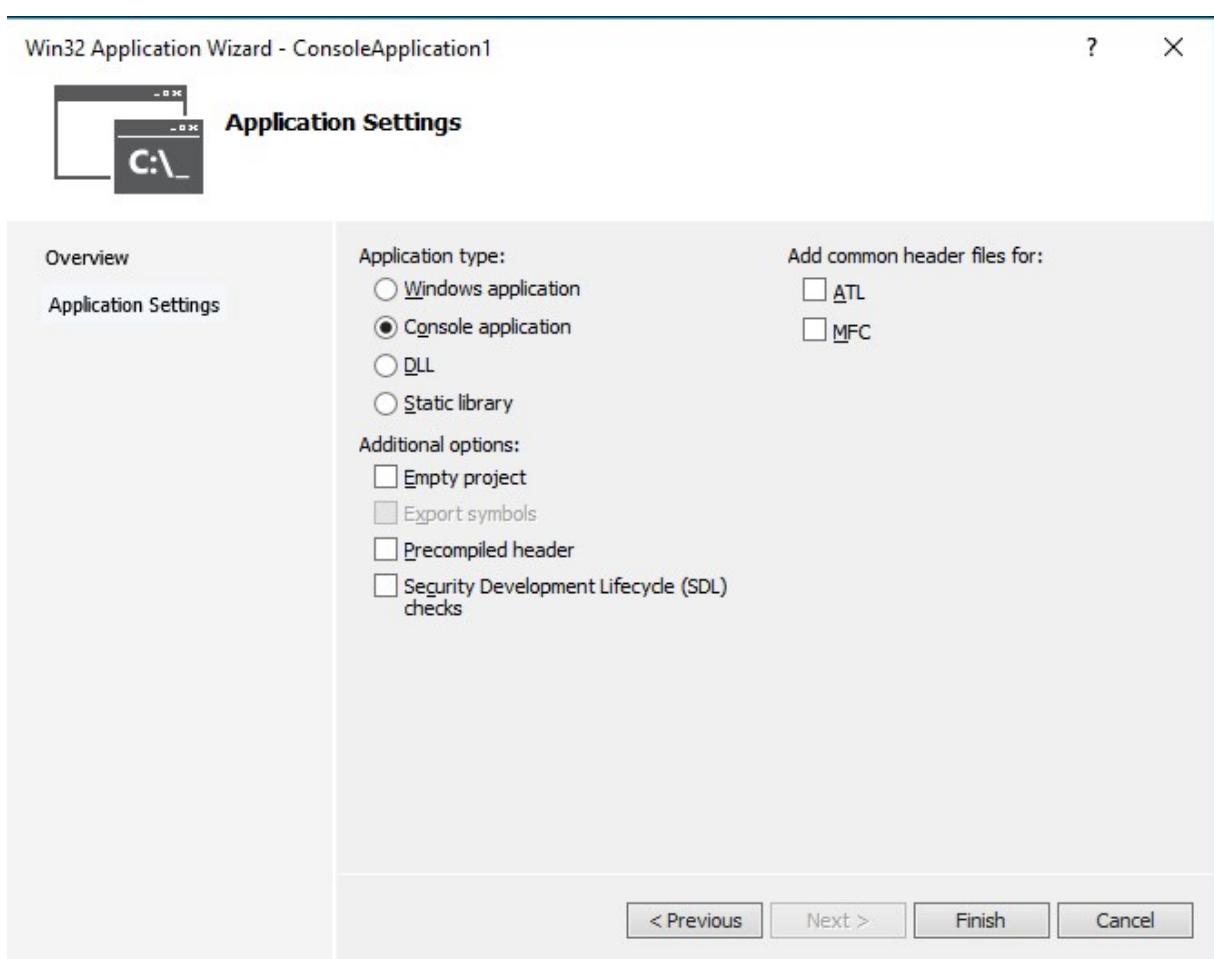
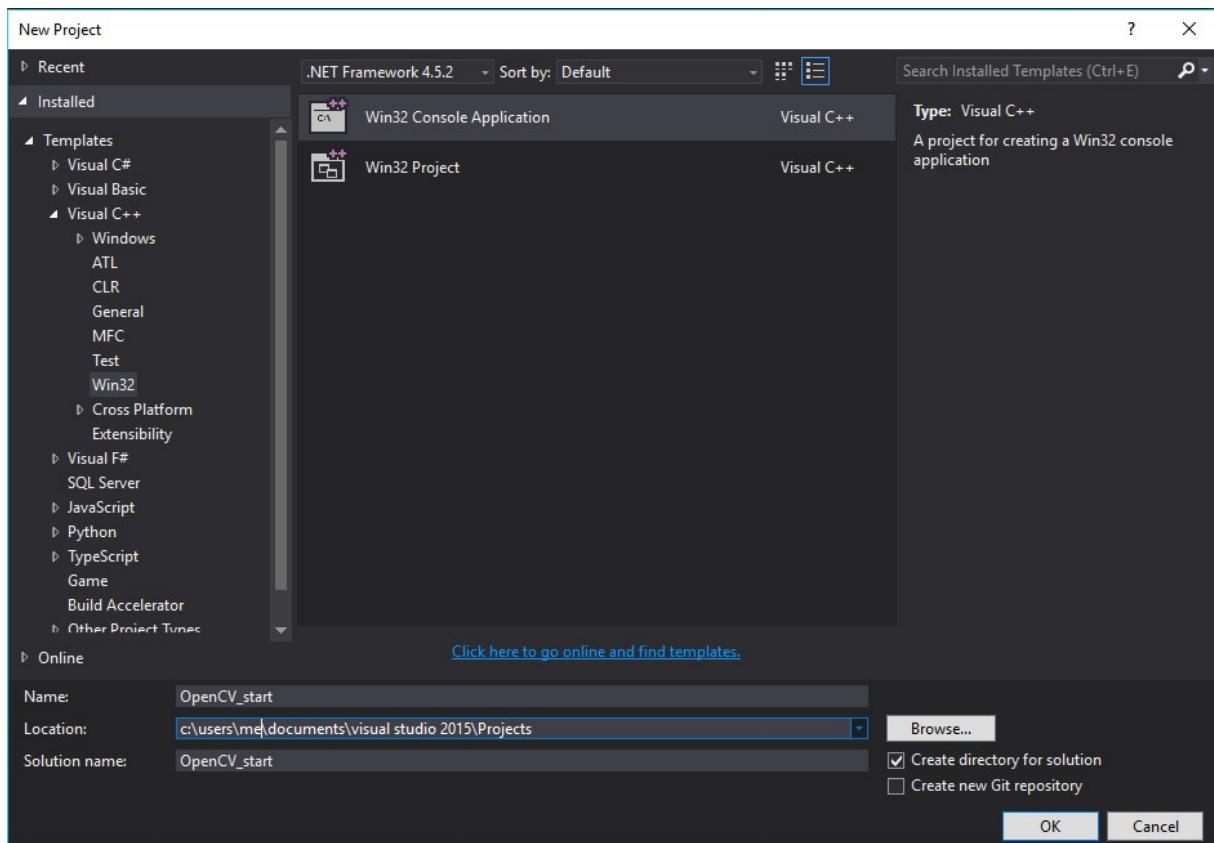
Windows

1. Skorzystanie z wersji release (binarnej)

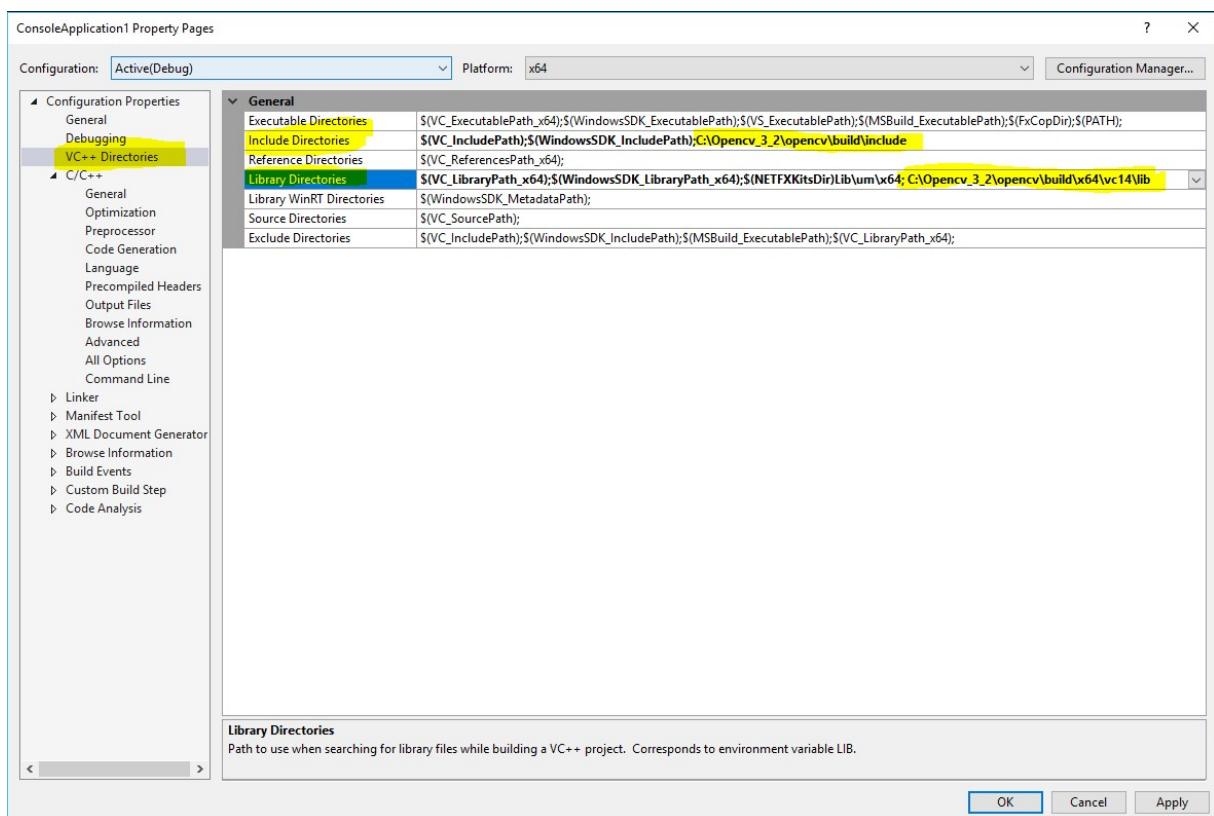
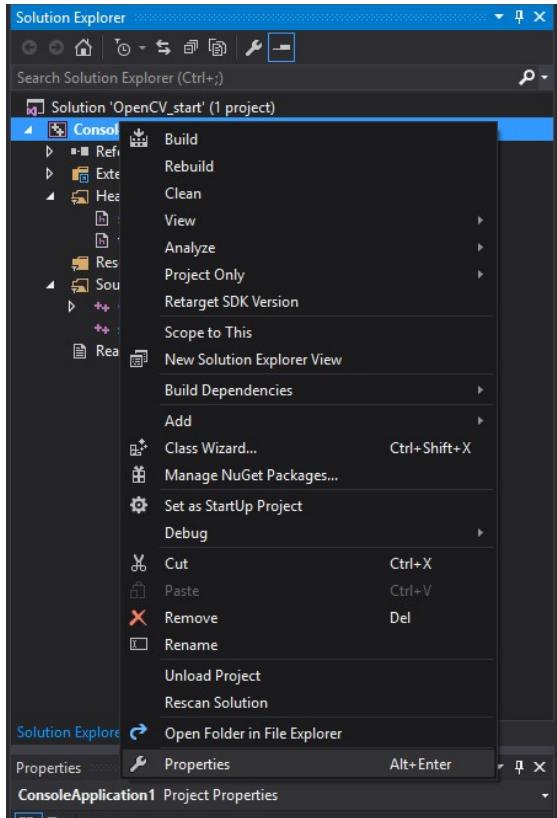
- Ze strony <http://opencv.org/releases.html>ściągnąć wersję Win pack (<https://sourceforge.net/projects/opencvlibrary/files/opencv-win/3.2.0/opencv-3.2.0-vc14.exe/download>) – biblioteki kompatybilne z Visual Studio od wersji 2015 oraz minGW, w wersji 64 bit
- Rozpakować plik do dowolnego katalogu (np. c:\Opencv_3_2)
- W wybranym IDE dodać katalog z bibliotekami jako c:\Opencv_3_2\opencv\build\x64\vc14\lib oraz katalog z plikami nagłówkowymi jako c:\Opencv_3_2\opencv\build\include
- Dodać do projektu linkowanie z biblioteką opencv_world320d.lib (dla wersji debug) lub opencv_world320.lib (dla wersji release)
- Pliki .dll potrzebne do uruchomienia znajdują się w katalogu c:\Opencv_3_2\opencv\build\x64\vc14\bin – należy je przekopiować w miejsce, w którym wybrane IDE generuje pliki wykonywalne

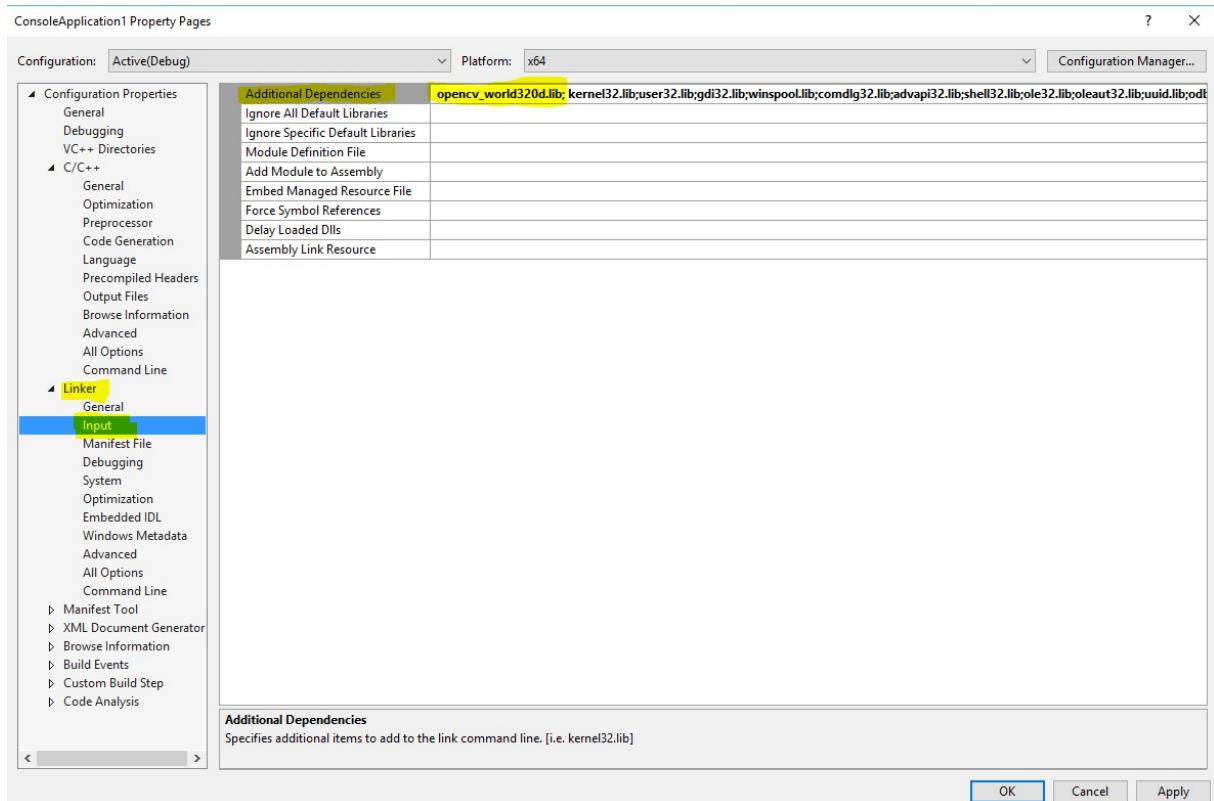
Konfiguracja na przykładzie Visual Studio:

1. Utworzenie nowego projektu



2. Dodanie zależności





Konfiguracja na przykładzie QT Creator:

1. Dodanie bibliotek OpenCV do projektu:

W pliku .pro (właściwości projektu dodać następujące wpisy (TYLKO WINDOWS), a następnie odkomentować właściwą konfigurację.

UWAGA – podać własne ścieżki dostępu do plików

```

INCLUDEPATH += "D:\Lib\opencv\build\include"

#Dla oddzielnych bibliotek dla modułów
LIBS += -L"D:\Lib\opencv\build\x64\vc14\lib"
LIBS += -lopencv_core320d \
        -lopencv_highgui320d \
        -lopencv_imgcodecs320d\
        -lopencv_imgproc320d \
        -lopencv_video320d\
        -lopencv_videoio320d

#Dla zbudowanego opencv_world (ściągnięte binarki ze strony OpenCV)
#LIBS += -L"C:\Opencv_3_2\opencv\build\x64\vc14\lib"
#LIBS += -lopencv_world320d

```

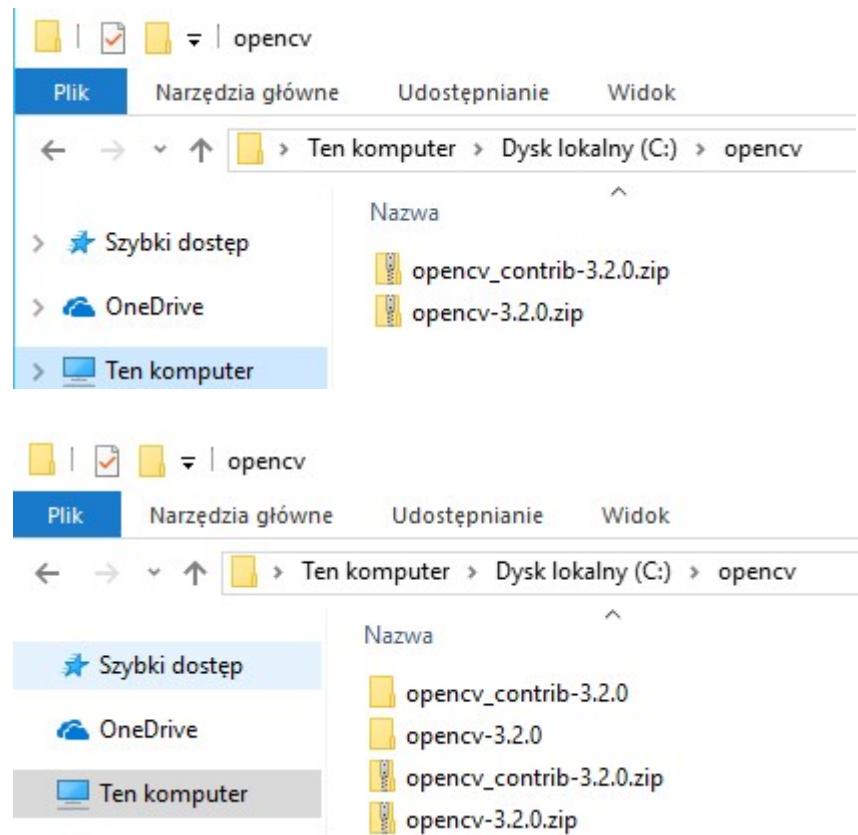
```
1 TEMPLATE = app
2 CONFIG += console c++11
3 CONFIG -= app_bundle
4 CONFIG -= qt
5 #QMAKE_CXXFLAGS += -std=gnu++11
6
7 SOURCES += main.cpp \
8     rs232.c
9
10 include(deployment.pri)
11 qtcAddDeployment()
12
13 HEADERS += \
14     rs232.h \
15     stdafx.h \
16     ArduinoSerialCommunicator.hpp
17
18 INCLUDEPATH += "D:\Lib\opencv\build\include"
19
20 #Dla oddzielnych bibliotek dla modułów
21 LIBS += -L"D:\Lib\opencv\build\x64\vc14\lib"
22 LIBS += -lopencv_core320d \
23         -lopencv_highgui320d \
24         -lopencv_imgcodecs320d\
25         -lopencv_imgproc320d \
26         -lopencv_video320d\
27         -lopencv_videoio320d
28
29 #Dla zbudowanego opencv_world (ściągnięte binarki ze strony OpenCV)
30 #LIBS += -L"C:\0opencv_3_2\opencv\build\x64\vc14\lib"
31 #LIBS += -lopencv_world320d
32
```

2. Budowanie ze źródeł z wykorzystaniem CMAKE

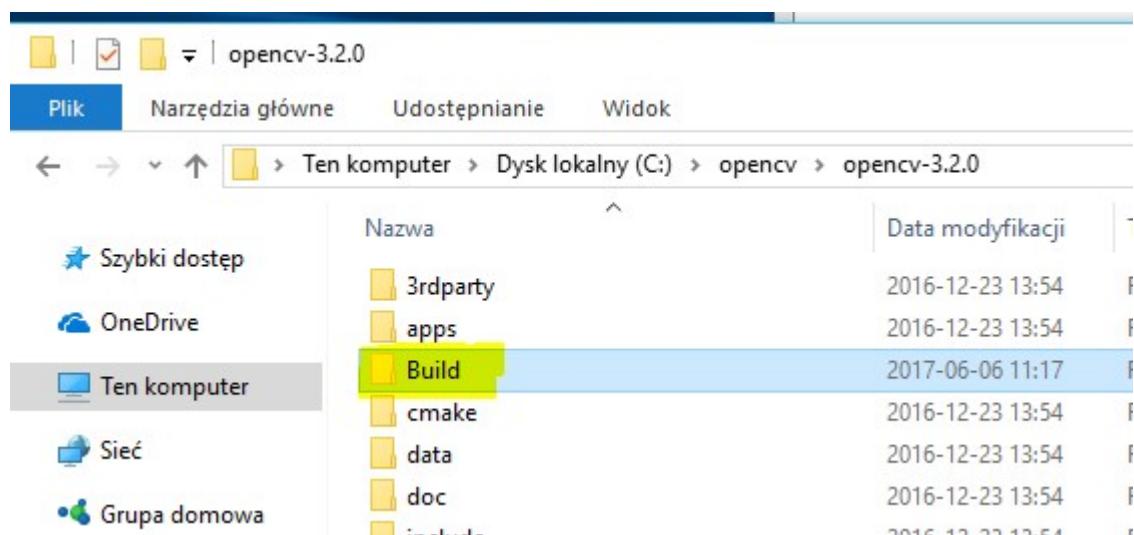
- Ze strony <https://github.com/opencv/opencv/releases>ściągnąć kod źródłowy głównego pakietu OpenCV (<https://github.com/opencv/opencv/archive/3.2.0.zip>)
- Ze strony https://github.com/opencv/opencv_contrib/releases/tag/3.2.0ściągnąć kod źródłowy pakietów dodatkowych (https://github.com/opencv/opencv_contrib/archive/3.2.0.zip)

!!!UWAGA!!! –kod źródłowy można też pobrać bezpośrednio z repozytoriów – dalszy proces instalacji będzie dokładnie ten sam

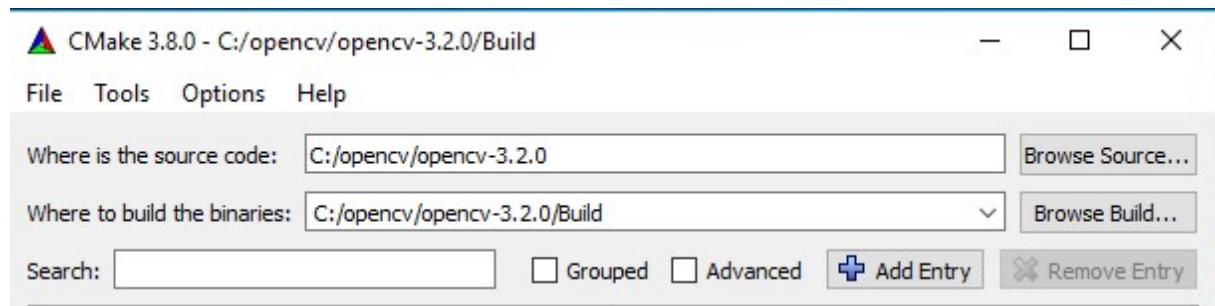
c. Umieścić pliki w odpowiednim katalogu i rozpakować



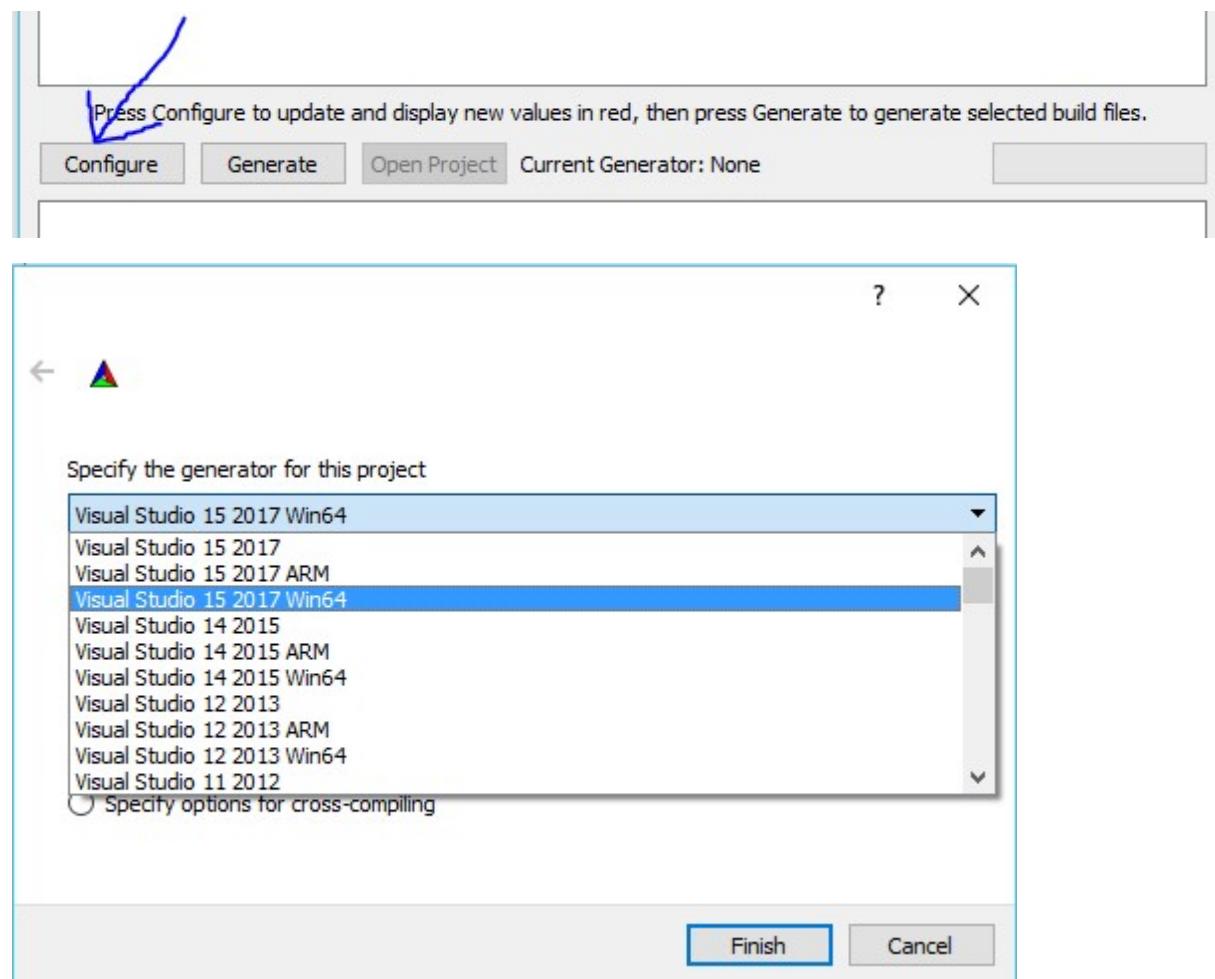
d. Stworzyć katalog Build



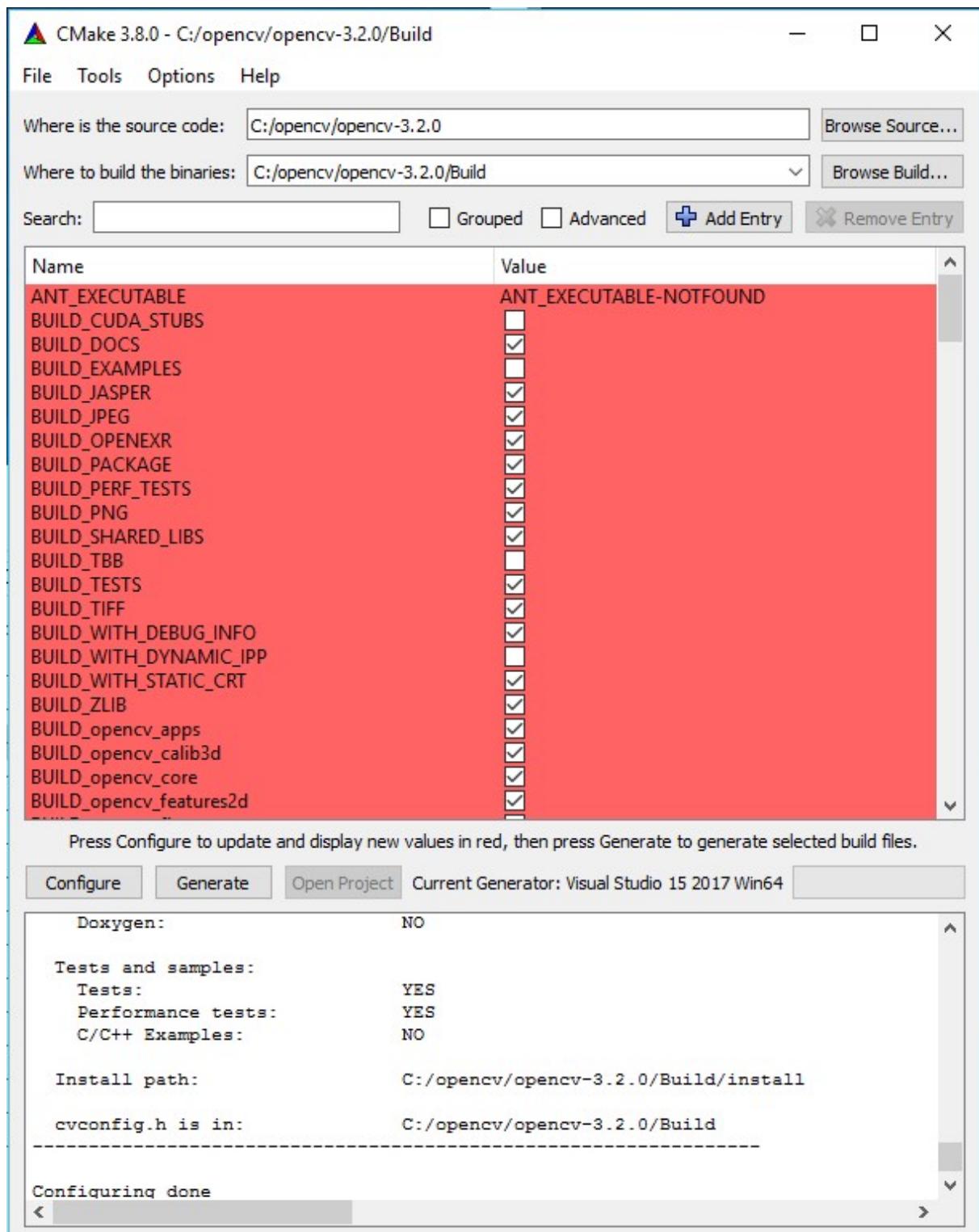
e. Uruchomić program CMake (cmake-gui) i podać ścieżki dostępu do źródeł



- f. Wcisnąć przycisk configure i wybrać kompilator (przykład pokazuje wybór przy zainstalowanym Visual Studio 2017). Począć na zakończenie procesu konfiguracji



(czerwone pola nie oznaczają błędów, tylko opcje nieskonfigurowane lub z wartościami domyślnymi)



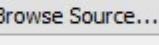
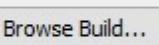
- g. Wybrać opcje konfiguracji (na zrzutach ekranu pokazano przykładową konfigurację, przygotowaną do jak najszybszego budowania bibliotek – wyłączone zostało budowanie testów oraz przykładów).

Search: <input type="text" value="Exa"/>	<input type="checkbox"/> Grouped	<input type="checkbox"/> Advanced	 Add Entry	 Remove Entry
Name	Value			
BUILD_EXAMPLES	<input type="checkbox"/>			
INSTALL_C_EXAMPLES	<input type="checkbox"/>			
INSTALL_PYTHON_EXAMPLES	<input type="checkbox"/>			

Name	Value
BUILD_PERF_TESTS	<input type="checkbox"/>
BUILD_TESTS	<input checked="" type="checkbox"/>
DOWNLOAD_EXTERNAL_TEST_DATA	<input type="checkbox"/>
INSTALL_TESTS	<input type="checkbox"/>

Search: <input type="text" value="cuda"/>	<input type="checkbox"/> Grouped	<input type="checkbox"/> Advanced	 Add Entry	 Remove Entry
Name	Value			
BUILD_CUDA_STUBS	<input type="checkbox"/>			
CUDA_BUILD_CUBIN	<input type="checkbox"/>			
CUDA_BUILD_EMULATION	<input type="checkbox"/>			
CUDA_HOST_COMPILER	\$({VCInstallDir})bin			
CUDA_SDK_ROOT_DIR	CUDA_SDK_ROOT_DIR-NOTFOUND			
CUDA_SEPARABLE_COMPILATION	<input type="checkbox"/>			
CUDA_TOOLKIT_ROOT_DIR	CUDA_TOOLKIT_ROOT_DIR-NOTFOUND			
CUDA_VERBOSE_BUILD	<input type="checkbox"/>			
WITH_CUDA	<input checked="" type="checkbox"/>			

- h. Podać ścieżkę do modułów dodatkowych (katalog modules w rozpakowanym opencv_contrib)

CMake 3.8.0 - C:/opencv/opencv-3.2.0/Build	
File	Tools Options Help
Where is the source code:	<input type="text" value="C:/opencv/opencv-3.2.0"/> 
Where to build the binaries:	<input type="text" value="C:/opencv/opencv-3.2.0/Build"/> 
Search: <input type="text" value="modul"/>	<input type="checkbox"/> Grouped <input type="checkbox"/> Advanced  Add Entry  Remove Entry
Name	Value
OPENCV_EXTRA_MODULES_PATH	C:/opencv/opencv_contrib-3.2.0/modules

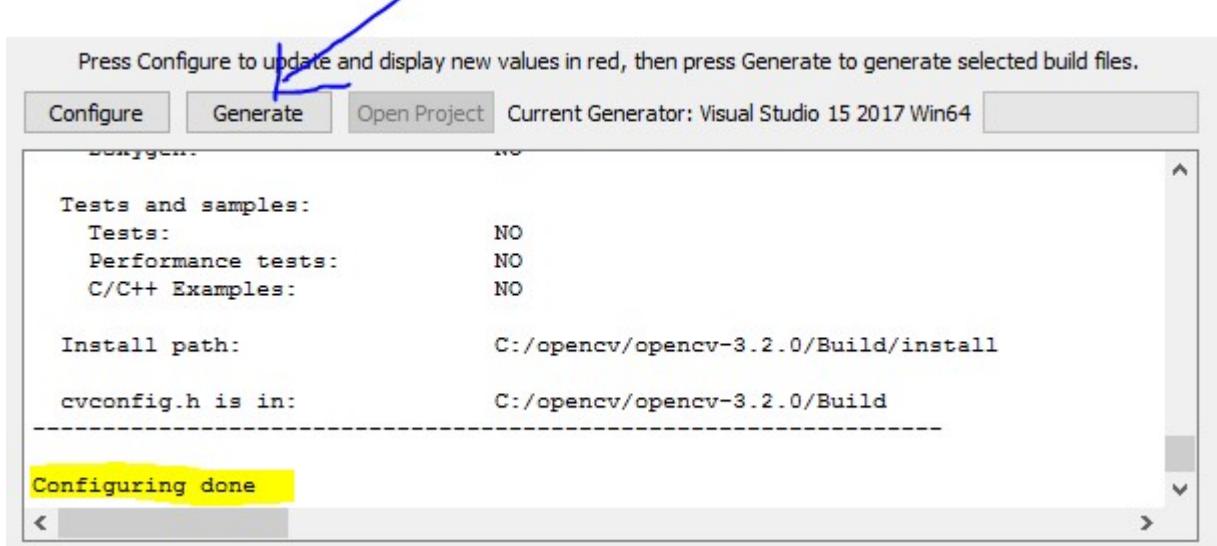
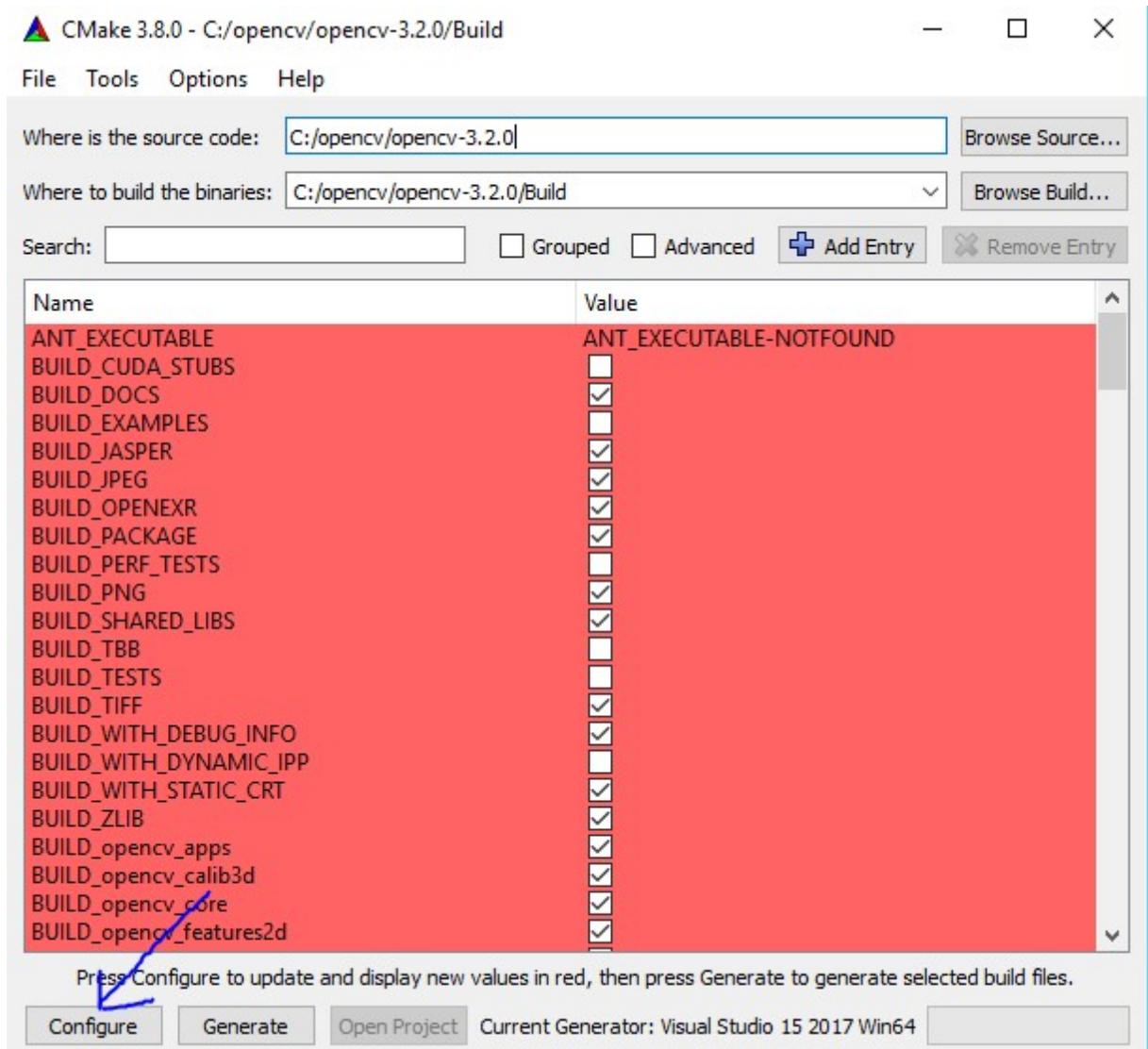
- i. OpenCV można zbudować z tzw. QT backend lub bez – nie powinno to mieć wpływu na działanie

Search: <input type="text" value="qt"/>	<input type="checkbox"/> Grouped	<input type="checkbox"/> Advanced	 Add Entry	 Remove Entry
Name	Value			
WITH_QT	<input type="checkbox"/>			

- j. Opcje zwiększające wydajność – tylko na nowszych komputerach, z procesorami Intel. W przypadku braku pewności pozostawić ustawienia domyślne.

Name	Value
ENABLE_AVX	<input checked="" type="checkbox"/>
ENABLE_AVX2	<input checked="" type="checkbox"/>
ENABLE_CCACHE	<input type="checkbox"/>
ENABLE_FMA3	<input checked="" type="checkbox"/>
ENABLE_IMPL_COLLECTION	<input type="checkbox"/>
ENABLE_INSTRUMENTATION	<input type="checkbox"/>
ENABLE_NOISY_WARNINGS	<input type="checkbox"/>
ENABLE_POPCNT	<input type="checkbox"/>
ENABLE_PRECOMPILED_HEADERS	<input checked="" type="checkbox"/>
ENABLE SOLUTION FOLDERS	<input checked="" type="checkbox"/>
ENABLE_SSE	<input checked="" type="checkbox"/>
ENABLE_SSE2	<input checked="" type="checkbox"/>
ENABLE_SSE3	<input checked="" type="checkbox"/>
ENABLE_SSE41	<input checked="" type="checkbox"/>
ENABLE_SSE42	<input checked="" type="checkbox"/>
ENABLE_SSSE3	<input checked="" type="checkbox"/>
EXECUTABLE_OUTPUT_PATH	C:/opencv/opencv-3.2.0/Build/bin
GLIB_LIBRARY	GLIB_LIBRARY-NOTFOUND
GLIB_gstcdda_LIBRARY	GLIB_gstcdda_LIBRARY-NOTFOUND
GOBJECT_LIBRARY	GOBJECT_LIBRARY-NOTFOUND
GSTREAMER_glib_INCLUDE_DIR	GSTREAMER_glib_INCLUDE_DIR-NOTFOUND
GSTREAMER_glibconfig_INCLUDE_DIR	GSTREAMER_glibconfig_INCLUDE_DIR-NOTFOUND

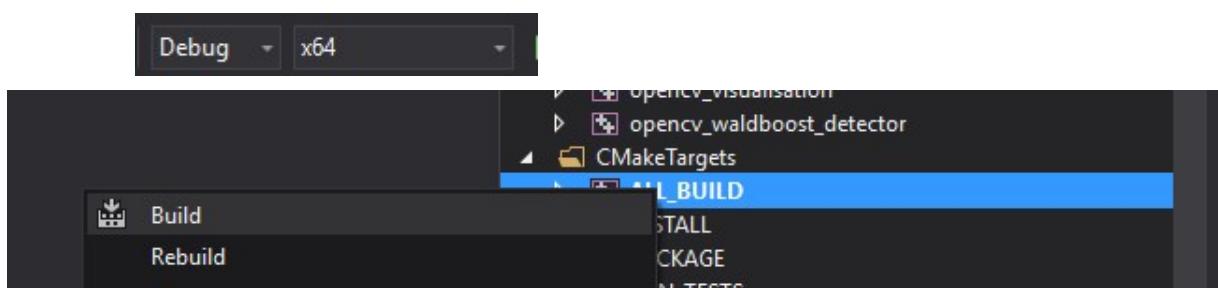
- k. Po ustawieniu parametrów jeszcze raz nacisnąć przycisk Configure, a następnie Generate, co spowoduje wygenerowanie plików potrzebnych do komplikacji i budowania bibliotek



Ten komputer > Dysk lokalny (C:) > opencv > opencv-3.2.0 > Build

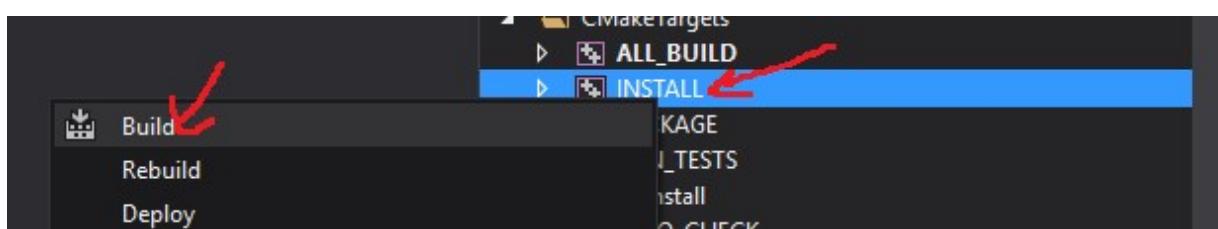
	Nazwa	Data modyfikacji
ep	CMakeCache.txt	2017-06-06 11:27
ter	CMakeVars.txt	2017-06-06 11:27
owa	CPackConfig.cmake	2017-06-06 11:27
	CPackSourceConfig.cmake	2017-06-06 11:27
	CTestTestfile.cmake	2017-06-06 11:28
	custom_hal.hpp	2017-06-06 11:20
	cvconfig.h	2017-06-06 11:20
	INSTALL.vcxproj	2017-06-06 11:28
	INSTALL.vcxproj.filters	2017-06-06 11:28
	OpenCV.sln	2017-06-06 11:29

- I. Wybrać właściwą konfigurację i zbudować projekty w odpowiedniej kolejności



Output

```
Show output from: Build
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/etc/lbpcascades/lbpcascade_profileface.xml
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/etc/lbpcascades/lbpcascade_silverware.xml
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/x64/vc15/bin/opencv_trainscaded.exe
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/x64/vc15/bin/opencv_createsamplesd.exe
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/x64/vc15/bin/opencv_annotationd.exe
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/x64/vc15/bin/opencv_visualisationd.exe
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/x64/vc15/bin/opencv_interactive-calibrationd.exe
1>-- Installing: C:/opencv/opencv-3.2.0/Build/install/x64/vc15/bin/opencv_versiond.exe
===== Build: 1 succeeded, 0 failed, 59 up-to-date, 0 skipped =====
```



Output

```
Show output from: Build
57>opencv_xphoto.vcxproj -> C:\opencv\opencv-3.2.0\Build\bin\Debug\opencv_xphoto320d.dll
58>opencv_videostab.vcxproj -> C:\opencv\opencv-3.2.0\Build\bin\Debug\opencv_videostab320d.dll
59>----- Build started: Project: ALL_BUILD, Configuration: Debug x64 -----
59>Building Custom Rule C:/opencv/opencv-3.2.0/CMakeLists.txt
59>CMake does not need to re-run because C:/opencv/opencv-3.2.0/Build/CMakeFiles/generate.stamp is up-to-date.
===== Build: 59 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
```

m. Do katalogu install zostaną skopiowane pliki wynikowe (biblioteki i pliki nagłówkowe)

Ten komputer > Dysk lokalny (C:) > opencv > opencv-3.2.0 > Build > install

	Nazwa	Data modyfikacji	Typ
łep	bin	2017-06-06 11:40	Folder plików
iter	etc	2017-06-06 11:40	Folder plików
lowa	include	2017-06-06 11:40	Folder plików
	x64	2017-06-06 11:40	Folder plików
	LICENSE	2016-12-23 13:54	Plik
	OpenCVConfig.cmake	2017-06-06 11:20	Plik CMAKE
	OpenCVConfig-version.cmake	2017-06-06 11:20	Plik CMAKE

Ten komputer > Dysk lokalny (C:) > opencv > opencv-3.2.0 > Build > install > include

	Nazwa	Data modyfikacji	Typ	Rozmiar
łep	opencv	2017-06-06 11:40	Folder plików	
iter	opencv2	2017-06-06 11:40	Folder plików	

Ten komputer > Dysk lokalny (C:) > opencv > opencv-3.2.0 > Build > install > x64 > vc15 > lib

	Nazwa	Data modyfikacji	Typ	Rozmiar
łep	opencv_aruco320d.lib	2017-06-06 11:36	Object File Library	267 KB
iter	opencv_bgsegm320d.lib	2017-06-06 11:36	Object File Library	202 KB
lowa	opencv_bioinspired320d.lib	2017-06-06 11:36	Object File Library	179 KB
	opencv_calib3d320d.lib	2017-06-06 11:35	Object File Library	295 KB
	opencv_ccalib320d.lib	2017-06-06 11:36	Object File Library	283 KB
	opencv_core320d.lib	2017-06-06 11:34	Object File Library	684 KB

Mac

OpenCV jest dostępne tylko przez zbudowanie bibliotek z kodów źródłowych z wykorzystaniem CMAKE

Z poziomu terminala:

```
sudo mkdir -p /opt/src
sudo chown $(whoami):staff /opt
sudo chown $(whoami):staff /opt
sudo chown $(whoami):staff /opt/src
cd /opt/src
curl -L https://github.com/opencv/opencv/archive/3.2.0.zip -o opencv32.zip
curl -L https://github.com/opencv/opencv_contrib/archive/3.2.0.zip -o opencv32contrib.zip
unzip opencv32.zip
unzip opencv32contrib.zip
mv -v opencv-3.2.0 /opt/src/opencv32
mv -v opencv_contrib-3.2.0 /opt/src/opencv32_contrib
cd /opt/src/opencv32
mkdir /opt/src/opencv32/release
cd /opt/src/opencv32/release
cmake \
-D CMAKE_INSTALL_PREFIX=/opt/opencv32 \
-D OPENCV_EXTRA_MODULES_PATH=/opt/src/opencv32_contrib/modules \
-D BUILD_TIFF=ON \
-D BUILD_opencv_java=OFF \
-D WITH_CUDA=OFF \
-D ENABLE_FAST_MATH=1 \
-D ENABLE_AVX=ON \
-D WITH_OPENGL=ON \
-D WITH_OPENCL=ON \
-D WITH_IPP=OFF \
-D WITH_TBB=ON \
-D WITH_EIGEN=ON \
-D WITH_VTK=OFF \
-D WITH_V4L=OFF \
-D BUILD_TESTS=OFF \
-D BUILD_PERF_TESTS=OFF \
-D CMAKE_BUILD_TYPE=RELEASE \
(lub DEBUG;RELEASE)

make -j8
make install
```

Graficznie:

COMING SOON

Linux:

Tak jak w przypadku MacOS budowanie tylko ze źródeł, z wykorzystaniem CMAKE

<http://www.pyimagesearch.com/2016/10/24/ubuntu-16-04-how-to-install-opencv/>

http://docs.opencv.org/trunk/d7/d9f/tutorial_linux_install.html

<https://www.linuxhint.com/how-to-install-opencv-on-ubuntu/>