**OpenCV auf Windows**

**OpenCV OHNE nicht-freie Module:**

Wenn OpenCV ohne Funktionen wie SIFT und SURF installiert werden soll, kann über pip3 einfach das Python3-Modul installiert werden.

pip3 install opencv-python

OpenCV-Funktionen können dann mit dem Import des „cv2“-Moduls abgerufen werden. Um zu testen, ob die Installation erfolgreich war, kann im interaktiven Interpreter das Modul importiert und die Versionsfunktion abgerufen werden.

>>>import cv2

>>>cv2.\_\_version

Danach sollte OpenCV für alle Python-Projekte verfügbar sein.

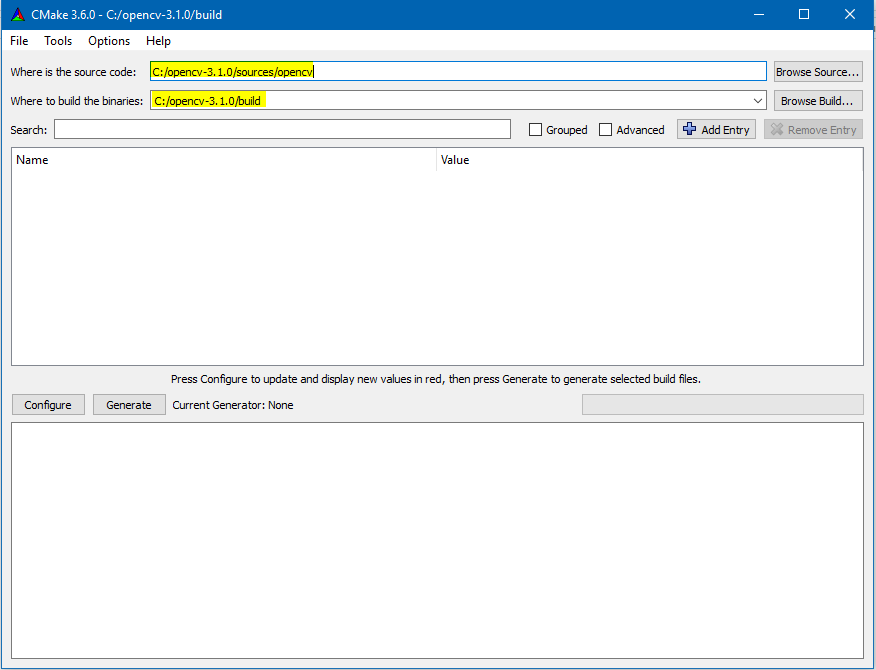
**OpenCV MIT nicht-freien Modulen:**

OpenCV kann außerdem mit Modulen, die auf nicht-freien Algorithmen basieren, kompiliert und installiert werden. Hier sollte man zunächst feststellen, ob diese wirklich benötigt werden, da die Kompilierung erhebliche Zeit in Anspruch nimmt.

Für die Kompilierung werden zunächst „opencv“ und „opencv\_contrib“ benötigt, diese können wie üblich als .zip heruntergeladen werden (<https://github.com/opencv>). Außerdem muss CMake installiert sein (<https://cmake.org/download/>). Benötigt wird auch Visual Studio, hier sollte man sicherstellen, dass ein C++-Compiler installiert und aktiviert ist.

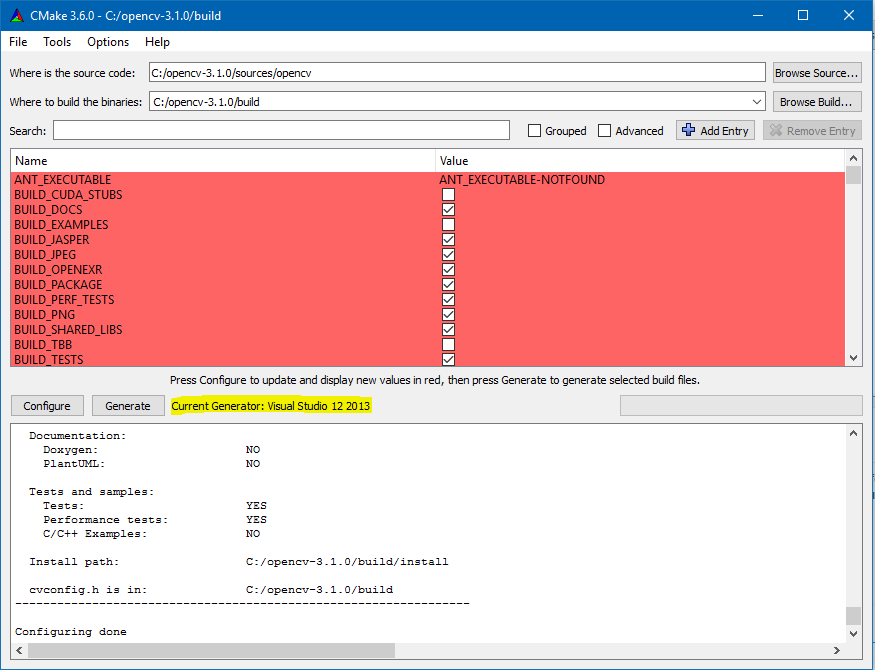
Zunächst muss ein Arbeitsverzeichnis installiert werden (z.B. „opencv-3.2.0-dev“), und in diesem zwei weitere Verzeichnisse „build“ und „source“. Nachdem „openc-master.zip“ und „opencv\_contrib-master.zip“ in ihre respektiven Ordner entpackt wurden, müssen diese Ordner in das „source“-Verzeichnis verschoben werden.

Nun muss CMake gestartet werden. Hier sollte unter dem „source code“-Pfad der Pfad zum „opencv“-Verzeichnis im „source“-Verzeichnis angegeben werden. Der „build“-Pfad entspricht dem vorher erstellten „build“-Verzeichnis.

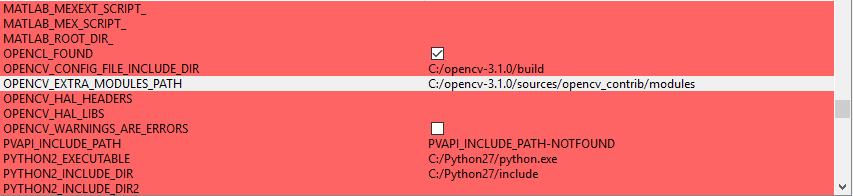


An dieser Stelle sollte nochmals sichergestellt werden, dass in Visual Studio ein C++ Compiler installiert ist.

Nach einem Druck auf „configure“ sollten sehr viele rot markierte Zeilen angezeigt werden.



Unter der NICHT rot markierten Zeile „OPENCV\_EXTRA\_MODULES\_PATH“ muss der Pfad zum „modules“-Verzeichnis im vorher entpackten „opencv\_contrib“-Verzeichnis angegeben werden.



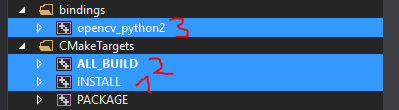
Jetzt muss erneut „configure“ gedrückt werden. Bevor „generate“ ausgewählt wird, muss zwangsweise der Haken bei „BUILD\_opencv\_contrib\_world“ entfernt werden.

Dann wird „generate“ ausgewählt. Der Abschluss des Generierungsprozesses wird mir „generating done“ im Ausgabefenster angezeigt.

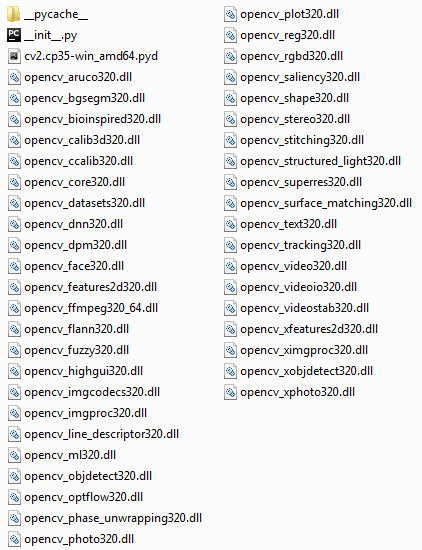
Im „build“-Verzeichnis sollten nun einige generierte Dateien aufwarten. Die .sln-Projektdatei muss nun mit der gleichen Visual Studio-Version, die auch in CMake benutzt wurde, geöffnet werden.

Wenn das Projekt geöffnet wurde, muss zunächst der Modus in den „Release“-Modus geschalten werden.

Die drei angezeigten Dateien müssen nun mit einem Klick auf „Build“ in der unten angezeigten Reihenfolge kompiliert werden.



Die dabei entstandenen .dll-Dateien sowie die .pyd-Datei müssen nun in das [..]\PythonXX\Lib\site-packages\-Verzeichnis verschoben werden (z.B. in einen Unterordner „cv2“). Wenn die .pyd-Datei dabei einen anderen Namen als cv2.pyd hat, muss diese dahingehend umbenannt werden.



Um zu testen, ob die Installation erfolgreich war, kann im interaktiven Interpreter das Modul importiert und die Versionsfunktion abgerufen werden.

>>>import cv2

>>>cv2.\_\_version

Danach sollte OpenCV für alle Python-Projekte verfügbar sein.