Anleitung zur Einrichtung von *OpenCV*

Dieses Dokument beschreibt die Einrichtung und Funktionsüberprüfung von *OpenCV* im Kontext der Lehrveranstaltung *Medieninformatik LB*.

Einrichtung

Dieser Abschnitt beschreibt die Einrichtung von *OpenCV* unter einem für die Lehrveranstaltung *Medientechnologie* eingerichteten *Ubuntu* Linux.

Anmerkung zu möglicherweise bereits installierten Paketen

Falls Sie Zusatzsoftware installiert oder nachinstalliert haben, müssen Sie jene, die *OpenCV* über Paketverwaltungssoftware benötigt, möglicherweise deinstallieren. Dies liegt darin begründet, dass die meisten über herkömmliche Paketverwaltungssoftware erhältlichen *OpenCV*-Pakete, z.B. `libopencv-*` über *apt*, zwar den Großteil der im Rahmen des Laboratoriums benötigten *OpenCV*-Funktionalität enthalten, zumeist aber nicht das für das Laboratorium essenzielle *xfeatures2d*-Modul. Wenn Sie sicherstellen können, dass *OpenCV* mitsamt diesem Modul auf Ihrem System installiert ist, können Sie den nachfolgenden Installationsabschnitt überspringen (falls nicht, lesen Sie bitte unbedingt im nächsten Absatz weiter). Stellen Sie in diesem Fall lediglich sicher, dass die auf Ihrem System installierte *OpenCV*-Version exakt 3.2.0 ist. Die Versionsnummer können Sie u.a. mit dem Aufruf

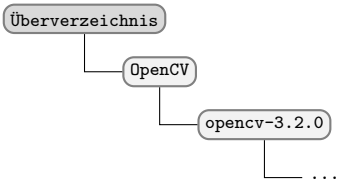
```
pkg-config --modversion opencv
```

überprüfen.
Wenn Sie nicht sicherstellen können, dass die auf Ihrem System installierte *OpenCV*-Version zur Nutzung im Rahmen des Laboratoriums geeignet ist (oder Sie sich dessen nicht sicher sind), deinstallieren Sie alle *OpenCV*-Pakete von Ihrem System über die entsprechende Paketverwaltungssoftware. Dies erfordert möglicherweise die Deinstallation aller Anwendungen, die diese Pakete benötigen. Parallelinstallationen im System führen oft zu Konflikten und schwer zu reproduzierenden Problemen, weswegen von ersteren abzuraten ist.

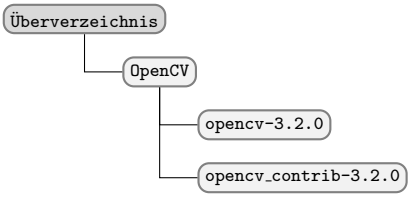
Konfiguration

Stellen Sie sicher, dass auf dem System die Pakete `build-essential` (enthält *g++*) sowie die Pakete `cmake`, `qt5-default` und `libvtk6-dev` (inkl. der Abhängigkeiten `libavcodec-dev`, `libavformat-dev`, `libswscale-dev`, `libjpeg-dev` und `libpng-dev`) installiert sind, bevor Sie fortsetzen.
Zuerst muss der Quelltext von *OpenCV* von (Stand: 11.7.2017) `https://github.com/opencv/opencv/archive/3.2.0.zip` herunterladen und entpackt werden.

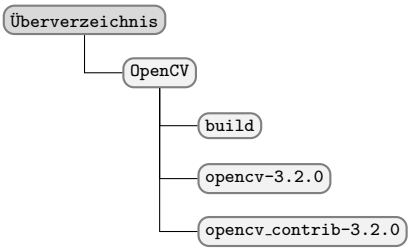
Es wird empfohlen, die nachfolgende Ordnerstruktur in einem von Ihnen gewählten Überverzeichnis[1] zu verwenden:



Der Inhalt des Ordners `opencv-3.2.0` entspricht dabei jenem des gleichnamigen Ordners aus der ZIP-Datei in entpackter Form.
Laden Sie außerdem die Zusatzmodule zu *OpenCV* von (Stand: 11.7.2017) `https://github.com/opencv/opencv_contrib/archive/3.2.0.zip` herunter und entpacken Sie diese in einen Ordner `opencv_contrib-3.2.0`. Es wird empfohlen, dass dieser Ordner in derselben Hiearchieebene der Ordnerstruktur wie der Ordner `opencv-3.2.0` angelegt wird. Die Ordnerstruktur sieht dann wie folgt aus:



Optional: Wenden Sie den Patch zur Behebung des `waitKey`-Fehlers durch Ausführung des zur Verfügung gestellten Shellskriptes im `OpenCV`-Verzeichnis an.
Erstellen Sie für den Build-Prozess einen Ordner `build`. Es wird empfohlen, dass dieser Ordner in derselben Hierarchieebene der Ordnerstruktur wie der Ordner `opencv-3.2.0` angelegt wird. Die Ordnerstruktur sieht dann wie folgt aus:



Wechseln Sie anschließend in das erstellte Verzeichnis `build`. Rufen Sie *cmake* wie unten mit dem (relativen) Pfad des `opencv-3.2.0`-Ordners zur Konfiguration von *OpenCV* auf. Geben Sie dabei außerdem mittels `-D`

[1] Bitte geben Sie diesem Verzeichnis einen anderen Namen als `Überverzeichnis` – vermeiden Sie in jedem Fall Leer- und Sonderzeichen (insbesondere Umlaute) im **gesamten** Dateipfad des Überverzeichnisses

`OPENCV_EXTRA_MODULES_PATH=<Pfad>/modules` den Pfad für die Zusatzmodule an. Deaktivieren Sie außerdem die Verwendung vorkompilierter Header mittels `-D ENABLE_PRECOMPILED_HEADERS=OFF` und aktivieren Sie die Verwendung von *QT* zur grafischen Visualisierung mittels `-D WITH_QT=ON` (**ohne** Zeilenumbrüche):

```
cmake -D ENABLE_PRECOMPILED_HEADERS=OFF -D WITH_QT=ON
-D OPENCV_EXTRA_MODULES_PATH=../opencv_contrib-3.2.0/modules
../opencv-3.2.0/
```

Zur Minimierung der Kompilierungszeit wird empfohlen, alle nicht benötigten Komponenten (im **selben** Aufruf, **nicht** separat!) zu deaktivieren, z.B. wie in

```
cmake -D WITH_1394=OFF -D WITH_CUDA=OFF -D WITH_CUFFT=OFF -D
WITH_EIGEN=OFF -D WITH_GSTREAMER=OFF -D WITH_IPP=OFF -D
WITH_JASPER=OFF -D WITH_WEBP=OFF -D WITH_OPENEXR=OFF -D
WITH_PVAPI=OFF -D WITH_GIGEAPI=OFF -D WITH_TIFF=OFF -D
WITH_V4L=OFF -D WITH_LIBV4L=OFF -D WITH_OPENCL=OFF -D
WITH_OPENCLAMDFFT=OFF -D WITH_OPENCLAMDBLAS=OFF -D
WITH_GPHOTO2=OFF -D BUILD_opencv_apps=OFF -D
BUILD_ANDROID_EXAMPLES=OFF -D BUILD_DOCS=OFF -D BUILD_PACKAGE=OFF
-D BUILD_PERF_TESTS=OFF -D BUILD_TESTS=OFF -D
BUILD_FAT_JAVA_LIB=OFF -D ENABLE_PRECOMPILED_HEADERS=OFF -D
CMAKE_BUILD_TYPE=RELEASE -D BUILD_opencv_photo=OFF -D
BUILD_opencv_photo=OFF -D BUILD_opencv_stitching=OFF -D
BUILD_opencv_superres=OFF -D BUILD_opencv_ts=OFF -D
BUILD_opencv_videostab=OFF -D
OPENCV_EXTRA_MODULES_PATH=../opencv_contrib-3.2.0/modules -D
BUILD_opencv_aruco=OFF -D BUILD_opencv_bgsegm=OFF -D
BUILD_opencv_bioinspired=OFF -D BUILD_opencv_ccalib=OFF -D
BUILD_opencv_contrib_world=OFF -D BUILD_opencv_cvv=OFF -D
BUILD_opencv_datasets=OFF -D BUILD_opencv_dnn=OFF -D
BUILD_opencv_dpm=OFF -D BUILD_opencv_fuzzy=OFF -D
BUILD_opencv_hdf=OFF -D BUILD_opencv_line_descriptor=OFF -D
BUILD_opencv_matlab=OFF -D BUILD_opencv_optflow=OFF -D
BUILD_opencv_phase_unwrapping=OFF -D BUILD_opencv_plot=OFF -D
BUILD_opencv_reg=OFF -D BUILD_opencv_rgbd=OFF -D
BUILD_opencv_saliency=OFF -D BUILD_opencv_sfm=OFF -D
BUILD_opencv_structured_light=OFF -D
BUILD_opencv_surface_matching=OFF -D BUILD_opencv_text=OFF -D
BUILD_opencv_tracking=OFF -D BUILD_opencv_ximgproc=OFF -D
BUILD_opencv_xobjdetect=OFF -D BUILD_opencv_xphoto=OFF
../opencv-3.2.0/
```

Die Ausgabe von `cmake` **muss** Folgendes enthalten (möglicherweise mit leicht unterschiedlichen Versionsnummern):

```
[..]
```

```
-- General configuration for OpenCV 3.2.0
    ↪ =====================================
[..]
--   OpenCV modules:
--     To be built:                 core flann imgproc ml
    ↪ video viz imgcodecs shape videoio highgui objdetect
    ↪ face features2d calib3d stereo xfeatures2d
[..]
--   GUI:
--     QT 5.x:                      YES (ver 5.5.1)
[..]
--     VTK support:                 YES (ver 6.2.0)
[..]
--   Media I/O:
[..]
--     JPEG:                        /usr/lib/x86_64-linux-
    ↪ gnu/libjpeg.so (ver )
[..]
--     PNG:                         /usr/lib/x86_64-linux-
    ↪ gnu/libpng.so (ver 1.2.54)
[..]
--   Video I/O:
[..]
--     FFMPEG:                      YES
--       avcodec:                   YES (ver 56.60.100)
--       avformat:                  YES (ver 56.40.101)
--       avutil:                    YES (ver 54.31.100)
--       swscale:                   YES (ver 3.1.101)
[..]
--   Install path:                  /usr/local
[..]
```

Installation

OpenCV kann unter Verwendung des im vorherigen Abschnitt konfigurierten `build`-Ordners mittels

```
make
```

kompiliert und mittels

```
sudo make install
```

installiert werden. Beachten Sie, dass für die Installation root-Rechte erforderlich sind, da standardmäßig global in `/usr/local/` installiert wird.
Bei Platzmangel können sowohl der `build`- als auch der `opencv-3.2.0`- und der `opencv_contrib-3.2.0`-Ordner gelöscht werden. Beachten Sie allerdings, dass dadurch eine Änderung der Installation (z.B. Aktivierung oder Deaktivierung von Modulen oder Funktionen) nicht mehr möglich ist und eine erneute Durcharbeitung dieses Abschnittes notwendig ist.

Funktionsüberprüfung

Dieser Abschnitt beschreibt, wie die Funktionalität einer wie im vorigen Abschnitt beschrieben eingerichteten *OpenCV*-Installation überprüft werden kann.

Minimal-Testprogramm

Erstellen Sie eine *cpp*-Datei in einem neuen Ordner (z.B. in Ihrem Projektverzeichnis für die erste Aufgabe zum ersten Laboratorium) an. Achten Sie darauf, dass weder im Dateinamen noch im -pfad Leer- oder Sonderzeichen (insbesondere Umlaute) enthalten sind[2]. Die *cpp*-Datei soll folgenden Inhalt haben:

```
#include <iostream>
#include "opencv2/core.hpp"

using namespace std;
using namespace cv;

int main(const int argc, const char * const argv[])
{
  cout << getBuildInformation() << endl;
  return 0;
}
```

Die Kernbestandteile (Klassen und Funktionen) von *OpenCV* werden dabei über `core.hpp` eingebunden; `cv::getBuildInformation` ist ein Beispiel für einen dieser Bestandteile – eine Hilfsfunktion, die Build-Informationen zurückliefert, die über `std::cout` auf der Standardausgabe, gefolgt von einem Zeilenumbruch (`std::endl`), ausgegeben werden.

Makefile

Die im vorigen Abschnitt erstellte `cpp`-Datei muss nun kompiliert und gelinkt werden. Dazu wird folgendes Makefile als Muster empfohlen:

```
CXX ?= g++

CXXFLAGS += -c -Wall `pkg-config --cflags opencv`
LDFLAGS += `pkg-config --libs opencv`
    ↪ -Wl,-rpath,/usr/local/lib

SRC = $(wildcard *.cpp)
OBJ = $(SRC:.cpp=.o)

NAME = EL00a
```

[2] Zwar können nahezu alle Programme mit Leer- und Sonderzeichen umgehen, doch das im nachfolgenden Abschnitt verwendete *pkg-config* stellt in diesem Kontext eine Ausnahme dar

```
all: $(NAME)

$(NAME): $(OBJ)
	$(CXX) $^ -o $@ $(LDFLAGS)

%.o: %.cpp
	$(CXX) $< -o $@ $(CXXFLAGS)

clean:
	rm -f $(OBJ) $(NAME)
```

Beachten Sie die Tabulatoren innerhalb der Targets sowie die Zeilenumbrüche! Als C++-Compiler wird *g++* (gesetzt über die Variable `CXX`) verwendet. Die entsprechenden Kommandozeilenparameter zur Einbindung der *OpenCV*-Pfade und -Bibliotheken in der richtigen Reihenfolge werden durch *pkg-config* generiert, das auf das bei der Installation erzeugte Verzeichnis mit Konfigurationsinformationen zu *OpenCV* zurückgreift. Zum Auffinden der dynamisch gelinkten Bibliotheken zur Laufzeit muss zusätzlich der Bibliothekspfad mittels `-rpath` an den Linker (Ergänzung `-Wl`) übergeben werden.
Durch den Aufruf von

```
make
```

im Verzeichnis der *cpp*-Datei wird nun das Programm `EL00a` erzeugt.

Überprüfung

Durch Ausführen des im vorigen Abschnitt erzeugten Beispielprogrammes über

```
./EL00a
```

kann überprüft werden, ob die *OpenCV*-Installation funktionsfähig (und das Beispielprogramm samt Makefile korrekt) ist – erscheinen ohne weitere Fehler die Build-Informationen aus der `cmake`-Ausgabe (vom Einrichtungsschritt oben), war die Installation erfolgreich und das Beispielprogramm funktioniert.