

## Installation von IVT, OpenCV und Qt unter Windows und Linux

*Autor: Lars Pätzold*

Die vorliegende Anleitung beschreibt die Einrichtung einer lauffähigen Umgebung für die Programmierung mit der IVT-Bibliothek unter Windows und Linux. Dazu gehört auch die Einrichtung der von der IVT unterstützten Bibliotheken OpenCV und Qt sowie eines Treibers für Firewire-Kameras.

Um möglichst wenige Vorkenntnisse vorauszusetzen, wurde die Anleitung in vielen Teilen bewusst ausführlich gehalten. Damit soll der Einstieg in die Verwendung der IVT erleichtert werden. Tatsächlich sind eine ganze Menge Schritte notwendig, bis man sich endlich der Programmierung widmen kann. Trotz des Mehraufwandes wird empfohlen, alle aufgeführten Bibliotheken zu installieren. Ist die Verwendung einer Kamera nicht vorgesehen, dann können die entsprechenden Schritte, in denen in irgendeiner Weise „1394“ vorkommt, übersprungen werden.

Die Anleitung gliedert sich in zwei große Abschnitte: *Windows* und *Linux*. Diese beiden Abschnitte sind als voneinander unabhängig zu betrachten. Der Abschnitt für Windows endet mit einer Zusammenfassung. Diese kann mehr oder weniger als Schnellanleitung für Windows dienen. Für das erste Mal wird jedoch die ausführliche Vorgehensweise anhand der Einzelschritte empfohlen, da diese wichtige Informationen rundum die Installation beinhalten.

Es ist zu beachten, dass die hier vorgestellte Software lizenzgemäß nur für private und/oder wissenschaftliche Zwecke verwendet werden darf, nicht jedoch für den gewerblichen Einsatz. Für genauere Informationen bezüglich der Lizenzbestimmungen wird auf die zu jeder Software angegebene Internetseite verwiesen.

## A.1 Windows

### A.1.1 OpenCV

#### 1. Download von OpenCV

Im Internet unter <http://sourceforge.net/projects/opencvlibrary/> ist im Download-Bereich die Datei `OpenCV_b4a.exe` herunterzuladen. Diese befindet sich unter dem Package *opencv-win* und dem weiterführenden Link *View older releases in the opencv-win package*. Dort findet man unter dem Release beta4 den Download `OpenCV_b4a.exe`. Diese Version wird empfohlen, da das Zusammenspiel der IVT mit dieser Version durch zahlreiche Tests sichergestellt wurde.

#### 2. Installation

Mit der heruntergeladenen Datei lässt sich die OpenCV installieren. Während des Setups kann das Zielverzeichnis angegeben werden (z.B. `C:\Programme\OpenCV`).

#### 3. Einstellungen in der Programmierumgebung

In der jeweiligen Programmierumgebung (z.B. Microsoft Visual C++) müssen die Verzeichnispfade für die Include- und Bibliothekdateien gesetzt werden.

Für die Include-Dateien sind die folgenden Pfade zu setzen:

```
C:\Programme\OpenCV\cv\include
C:\Programme\OpenCV\cxcore\include
C:\Programme\OpenCV\otherlibs\highgui
C:\Programme\OpenCV\cvaux\include
C:\Programme\OpenCV\otherlibs\cvcam\include
```

Der Pfad für die Bibliothekdateien lautet:

```
C:\Programme\OpenCV\lib
```

Wurde im vorherigen Schritt ein anderes Zielverzeichnis als `C:\Programme\OpenCV` gewählt, so sind die aufgelisteten Pfade entsprechend anzupassen.

In Microsoft Visual C++ 6.0 lassen sich die Verzeichnispfade in den allgemeinen Optionen wie folgt einstellen: Im Menü *Extras* über den Menüpunkt *Optionen...* gelangt man zur Registerkarte *Verzeichnisse*. Um die Verzeichnisse

für die Include-Dateien einzugeben, wählt man in der rechten Dropdown-Liste *Include-Dateien* und für die Verzeichnisse der Bibliothekdateien entsprechend *Bibliothekdateien* aus (siehe Abbildung A.1). In der unteren Liste lassen sich dann die Verzeichnispfade hinzufügen.

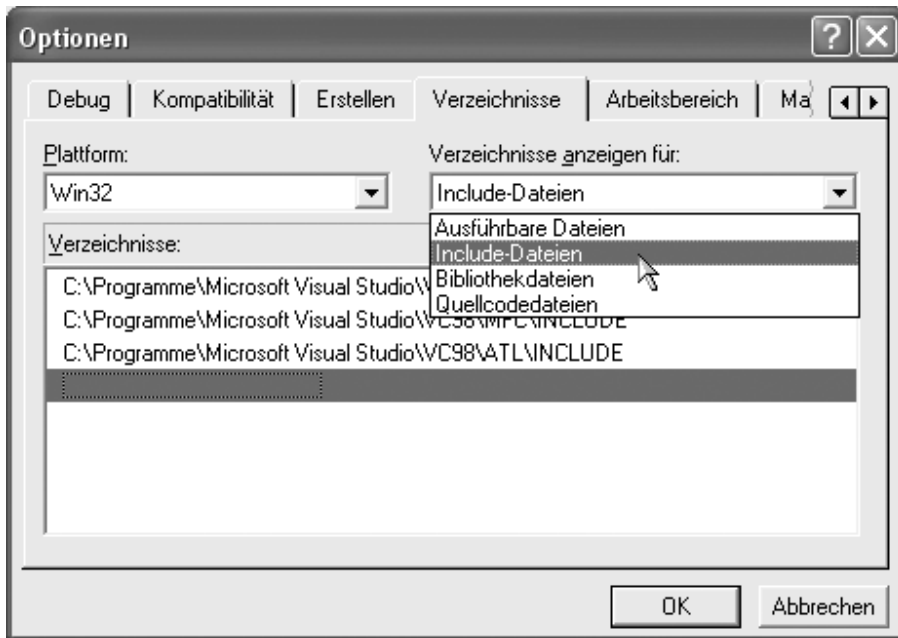
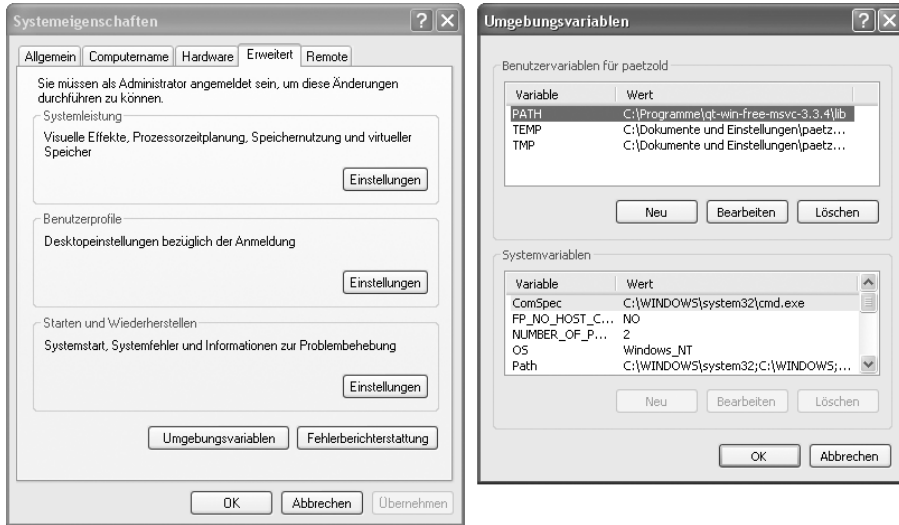


Abbildung A.1. Dialogfenster für die Optionen.

#### 4. Überprüfung der Systemvariablen PATH

Über *Start*, *Systemsteuerung*, *System* öffnet man das Dialogfenster *Systemeigenschaften* (alternativ über die Tastenkombination Windows-Taste + Pause-Taste). Dort gelangt man auf der Registerkarte *Erweitert* über die Schaltfläche *Umgebungsvariablen* zum entsprechenden Dialogfenster (siehe Abbildung A.2). Die PATH-Variable sollte `C:\Programme\OpenCV\bin` enthalten (der Pfad ist entsprechend dem angegebenen Zielverzeichnis während des Setups anzupassen). Ist dies noch nicht der Fall, muss die PATH-Variable entsprechend bearbeitet werden. Es ist darauf zu achten, dass mehrere Pfade in der PATH-Variable durch ein Semikolon getrennt werden müssen.



**Abbildung A.2.** Dialogfenster für die Systemeigenschaften (links) und die Umgebungsvariablen (rechts).

## A.1.2 Qt

### 1. Download von Qt

An dieser Stelle wird eine frei verfügbare Implementierung von Qt3 empfohlen, welche sich auf einfache Art und Weise herunterladen und installieren lässt. Es sei angemerkt, dass diese Implementierung nicht mit der offiziellen Qt-Version der Firma Trolltech in Verbindung steht. Für diesbezüglich genauere Informationen wird ausdrücklich an die folgende Internet-Adresse verwiesen. Die freie Qt3-Version findet man im Internet unter <http://sourceforge.net/projects/qtwin/>. Im Download-Bereich unter dem Package *Unofficial Qtwin* gelangt man über den weiterführenden Link *View older releases in the Unofficial Qtwin package* zum Download der Version qt-win-3.3.4-3. Für Microsoft Visual C++ 6.0 ist die Datei `setup-qt-win-free-msvc-3.3.4-3.exe` und für Microsoft Visual Studio .NET 2003 die Datei `setup-qt-win-free-msvc.net2003-3.3.4-3.exe` herunterzuladen.

### 2. Installation

Mit der heruntergeladenen Datei lässt sich ein Setup ausführen, welches erlaubt, das Zielverzeichnis frei zu wählen.

### 3. Einstellungen in der Programmierungsumgebung

In der Programmierungsumgebung müssen folgende Verzeichnispfade eingestellt werden:

Pfad für die Include-Dateien:

`C:\Programme\qt-win-free-msvc-3.3.4\include`

Pfad für die Bibliothekdateien:

`C:\Programme\qt-win-free-msvc-3.3.4\lib`

Eine Anleitung zum Einstellen der Pfade in Microsoft Visual C++ 6.0 befindet sich in Schritt 3 des vorangegangenen Abschnitts zur OpenCV.

Die Pfade sind entsprechend anzupassen, falls zuvor ein anderes Zielverzeichnis gewählt wurde.

### 4. Überprüfung der Systemvariablen PATH

Die PATH-Variable sollte `C:\Programme\qt-win-free-msvc-3.3.4\bin` (ggf. anzupassen entsprechend dem angegebenen Zielverzeichnis während des Setups) enthalten. Wie sich die Systemvariable überprüfen und ändern lässt, ist in Schritt 4 der Installation der OpenCV beschrieben. Dabei handelt es sich um den Ort der Bibliotheksdatei `qt-mt3.dll`.

#### A.1.3 CMU1394

##### 1. Download

Die Setup-Version des CMU1394-Treibers findet man im Download-Bereich unter <http://www.cs.cmu.edu/~iwan/1394/>.

##### 2. Installation

Das Ausführen der heruntergeladenen Datei startet ein Setup. Während des Setups können die zu installierenden Komponenten sowie das Zielverzeichnis der Installation gewählt werden. Hierbei ist insbesondere darauf zu achten, dass die Komponente *Development Files* mitinstalliert wird (siehe Abbildung A.3). Das Standardverzeichnis für die Installation ist `C:\Programme\CMU\1394Camera`.



Abbildung A.3. Setup des CMU1394-Treibers.

### 3. Einstellungen in der Programmierungsumgebung

Die Verzeichnispfade für die Programmierungsumgebung sind wie folgt:

Pfad für die Include-Dateien:

`C:\Programme\CMU\1394Camera\include`

Pfad für die Bibliothekdateien:

`C:\Programme\CMU\1394Camera\lib`

Eine Anleitung zum Einstellen der Pfade in Microsoft Visual C++ 6.0 ist in Schritt 3 des vorangegangenen Abschnitts zur OpenCV angegeben.

Wieder sind die Pfade entsprechend anzupassen, falls zuvor ein anderes Zielverzeichnis gewählt wurde.

#### A.1.4 IVT

##### 1. Download der IVT

Im Internet unter <http://sourceforge.net/projects/IVT/> ist im Download-Bereich die zip-Datei (z.B. `ivt-1.0.5.zip`) herunterzuladen.

##### 2. Entpacken

Die heruntergeladene zip-Datei kann in ein beliebiges Verzeichnis entpackt werden. Es wird empfohlen, die Datei an den selben Ort zu entpacken,

an dem sich OpenCV und Qt befinden. Dieser Ort entspricht gemäß der Standardkonfiguration der Installationen von OpenCV und Qt dem Verzeichnis `C:\Programme`. Nach dem Entpacken befinden sich unter `C:\Programme\IVT` die Unterverzeichnisse `doc`, `examples`, `files`, `lib`, `src` und `win32`.

### 3. Einstellungen in der Programmierumgebung

In der jeweiligen Programmierumgebung müssen die Verzeichnispfade für die Include- und Bibliothekdateien gesetzt werden. Die Include-Dateien befinden sich im Unterverzeichnis `src` des IVT-Verzeichnisses (z.B. `C:\Programme\IVT\src`). Die Bibliothekdateien befinden sich im Unterverzeichnis `lib\win32` des IVT-Verzeichnisses (z.B. `C:\Programme\IVT\lib\win32`).

Eine Anleitung zum Einstellen der Pfade in Microsoft Visual C++ 6.0 ist in Schritt 3 des Abschnitts zur OpenCV angegeben.

Wieder sind die Pfade entsprechend anzupassen, falls die IVT zuvor in ein anderes Verzeichnis entpackt wurde.

### 4. Erstellen der Bibliotheken

Im Verzeichnis `win32\IVTLib` befindet sich die Arbeitsbereich-Datei `IVTLib.dsw` für Microsoft Visual C++ 6.0. Wird diese Datei in einer neueren Version von Microsoft Visual C++ geöffnet, so findet eine automatische Konvertierung in ein neues Dateiformat statt und der Arbeitsbereich kann dann gleichermaßen verwendet werden. Nach dem Öffnen lässt sich im Menü *Erstellen* über den Menüpunkt *Aktive Konfiguration festlegen...* die Version der zu erstellenden Bibliothek auswählen (vgl. Abbildung A.4).



Abbildung A.4. Dialogfenster für die Konfiguration des aktiven Projektes.

Folgende Versionen stehen zur Auswahl:

### **IVTLib – Win32 Release**

Hierbei handelt es sich um die Standard-Version der IVT-Bibliothek. Nicht enthalten sind Debug-Informationen. Der Dateiname der Bibliotheksdatei lautet `ivt.lib`.

### **IVTLib – Win32 Debug**

Die Debug-Version erlaubt den Einsprung in Funktionen und das schrittweise Ausführen der Funktionen innerhalb der IVT-Bibliothek während des Debuggens. Üblicherweise sind Anwendungen, die mit der Debug-Version kompiliert wurden, in der Ausführung deutlich langsamer. Die zu erstellende Bibliothek trägt den Dateinamen `ivtd.lib`.

### **IVTQTGUIlib – Win32 Release**

Mit dieser Auswahl wird eine weitere Bibliothek erstellt. Dabei handelt es sich lediglich um eine Erweiterung für die Standard-Version der IVT-Bibliothek. Diese enthält die vereinfachte Unterstützung für grafische Benutzerschnittstellen mit Qt. Der Dateiname der Bibliothek ist `ivtguiqt.lib`.

### **IVTQTGUIlib – Win32 Debug**

Die Debug-Version der Erweiterungsbibliothek erlaubt wiederum das Debuggen innerhalb des IVT-Quellcodes. Der Dateiname der Debug-Version ist `ivtguiqtd.lib`.

Die jeweilige Bibliotheksdatei kann anschließend über das Menü *Erstellen* und den entsprechenden Menüpunkt (z.B. *ivt.lib erstellen*) erzeugt werden. Die erstellte Bibliotheksdatei (z.B. `ivt.lib`) befindet sich danach im Unterverzeichnis `IVT\lib\win32`. Um später alle Anwendungen mit IVT fehlerlos kompilieren zu können, sollten alle vier Bibliotheksdateien der IVT erstellt werden.

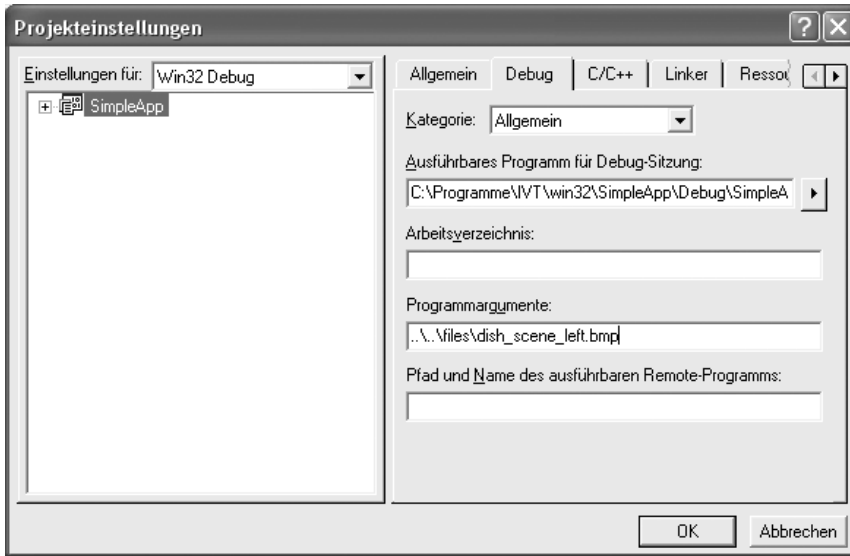
## **5. Beispielanwendung**

Zur Überprüfung der Installation sowie zum Einstieg in die Programmierung mit der IVT eignet sich die Beispielanwendung `SimpleApp`. Diese ist die einfachste der zahlreichen Beispielanwendungen, die in der IVT enthalten sind.

Die Arbeitsbereich-Datei `SimpleApp.dsw` befindet sich im IVT-Unterverzeichnis `win32\SimpleApp`. Nachdem man diese in Microsoft Visual C++ geöffnet hat, sollte sich über das Menü *Erstellen* und den Menüpunkt *SimpleApp.exe erstellen* die Anwendung fehlerlos kompilieren lassen. Bei Problemen ist zuerst zu überprüfen, welche Version im Menü *Erstellen* unter dem Menüpunkt *Aktive Konfiguration festlegen...* ausgewählt ist und ob die entsprechende IVT-Bibliotheksdatei im vorherigen Schritt erstellt wurde. Die Beispielanwendungen sind für die Debug-Version der IVT-Bibliothek vorkonfiguriert, d.h. für `ivtd.lib` und ggf. für `ivtguiqtd.lib`.



Bevor man die Anwendung ausführt, sollte man eine Bilddatei als Programmargument einstellen. Die Programmargumente lassen sich in den *Projekteinstellungen* hinzufügen (siehe Abbildung A.5). Dorthin gelangt man über das Menü *Projekt* und den Menüpunkt *Einstellungen....* Auf der Registerkarte *Debug* gibt man z.B. `..\..\files\dish_scene_left.bmp` als Programmargument an. Dieser Pfad bezieht sich auf eine Bilddatei, welche im Unterverzeichnis `files` der IVT enthalten ist.



**Abbildung A.5.** Dialogfenster für die Projekteinstellungen.

Führt man nun die Anwendung über das Menü *Erstellen* und den Menüpunkt *Ausführen von SimpleApp.exe* aus, erhält man als Ausgabe: *output written to file 'output.bmp'*. Anschließend befindet sich die Datei `output.bmp` im selben Verzeichnis wie die Arbeitsbereich-Datei `SimpleApp.dsw` bzw. `win32\SimpleApp` im IVT-Verzeichnis.

Um zu überprüfen, ob OpenCV und Qt ebenfalls korrekt installiert sind, kann man die Beispielanwendungen `SimpleAppCV` und `ShowImageQT` starten. Beide Anwendungen lassen sich in der gleichen Art und Weise handhaben wie die Anwendung `SimpleApp`.

Wenn bis hierhin alle Schritte durchgeführt wurden und die Beispielanwendungen kompiliert und ausgeführt werden konnten, dann ist eine lauffähige Umgebung für die Programmierung mit der IVT eingerichtet und kann nun verwendet werden.

### A.1.5 Zusammenfassung

Sollten nach Ausführung sämtlicher Installationsschritte Probleme während des Kompilierens auftreten, so dient die folgende Übersicht der Überprüfung aller wichtigen Verzeichnispfade. Die Übersicht fasst auf einen Blick alle Verzeichnispfade zusammen, die in der Programmierumgebung (z.B. Microsoft Visual C++) und in der Systemvariable PATH eingestellt sein sollten. Zusätzlich sind die in den Installationsschritten erwähnten Internetadressen aufgelistet.

Internetadressen:

```
http://sourceforge.net/projects/opencvlibrary/
http://sourceforge.net/projects/qtwin/
http://www.cs.cmu.edu/~iwan/1394/
http://sourceforge.net/projects/IVT/
```

Pfade für die Include-Dateien:

```
C:\Programme\OpenCV\cv\include
C:\Programme\OpenCV\cxcore\include
C:\Programme\OpenCV\otherlibs\highgui
C:\Programme\OpenCV\cvaux\include
C:\Programme\OpenCV\otherlibs\cvcam\include
C:\Programme\qt-win-free-msvc-3.3.4\include
C:\Programme\CMU\1394Camera\include
C:\Programme\IVT\src
```

Pfade für die Bibliothekdateien:

```
C:\Programme\OpenCV\lib
C:\Programme\qt-win-free-msvc-3.3.4\lib
C:\Programme\CMU\1394Camera\lib
C:\Programme\IVT\lib\win32
```

Die Systemvariable PATH sollte folgende Pfade enthalten:

```
C:\Programme\OpenCV\bin; C:\Programme\qt-win-free-msvc-3.3.4\bin;
```

## A.2 Linux

### A.2.1 OpenCV

#### 1. Download von OpenCV

Unter <http://sourceforge.net/projects/opencvlibrary> im Download-Bereich unter dem Package `opencv-linux` folgt man dem weiterführenden Link *View older releases in the opencv-linux package*. Dort lässt sich unter dem Release `beta4` die Datei `opencv-0.9.6.tar.gz` herunterladen.

#### 2. Entpacken

In einer Konsole wechselt man zum Entpacken in das Verzeichnis des heruntergeladenen Archivs und führt anschließend den Befehl `tar xfvz opencv-0.9.6.tar.gz` aus.

#### 3. Erstellen der OpenCV-Bibliotheken

Nach dem Wechseln in das entpackte Verzeichnis `opencv-0.9.6` führt man die folgenden Befehle nacheinander aus:

```
./configure
make
make install
ldconfig
```

**Hinweis:** Für die Befehle `make install` und `ldconfig` sind Root-Rechte erforderlich.

Im Anschluss daran sollten sich im Verzeichnis `/usr/local/include/opencv` die Include-Dateien der OpenCV befinden.

### A.2.2 Qt

Für die Installation wird ein sog. Package-Tool empfohlen. Es ist darauf zu achten, ein Developer-Package von Qt in der Version 3 zu installieren. Unter Debian Linux erfolgt dies z.B. mit: `apt-get install qt3-dev-tools`.

Es sollte anschließend das Verzeichnis `/usr/include/qt3` existieren, in dem sich die Include-Dateien von Qt befinden.

### A.2.3 Firewire und libdc1394/libraw1394

#### 1. Installation von libdc1394/libraw1394

Die beiden Bibliotheken libdc1394 und libraw1394 können ebenfalls mit einem Package-Tool installiert werden. Unter Debian Linux z.B. mit `apt-get install libdc1394` sowie `apt-get install libraw1394`.

#### 2. Einrichten einer Firewire-Kamera (optional)

Um die Kamera unter Linux zu verwenden, müssen die vier Kernel-module namens raw1394, video1394, ohci1394 und ieee1394 aktiviert werden. Dies erfolgt unter Debian Linux mithilfe des Konsolenbefehls `modprobe`. Dabei genügen `modprobe raw1394` und `modprobe video1394`, um alle vier Module zu aktivieren. Ob die vier Module vorhanden sind, lässt sich mit dem Befehl `lsmod | grep 1394` kontrollieren. Dieser Befehl sollte die o.g. vier Module auflisten. Sollte dies nicht der Fall sein, kann versucht werden, jedes Modul einzeln mit dem Befehl `modprobe` hinzuzufügen.

Damit ein Anwender oder eine Anwendung auf die Schnittstelle zur Kamera zugreifen kann, ist der Anwender in die entsprechenden Gruppen der Geräte unter `/dev/raw1394` und `/dev/video1394` einzutragen. Zum Eintragen in die Gruppe – normalerweise handelt es sich um die Gruppe `video` – sind Root-Rechte erforderlich.

### A.2.4 IVT

#### 1. Download der IVT

Hierzu lädt man im Internet unter <http://sourceforge.net/projects/IVT/> im Download-Bereich das Archiv, z.B. `ivt-1.0.5.tar.gz`, herunter.

#### 2. Entpacken

In einer Konsole wechselt man zum Entpacken in das Verzeichnis des heruntergeladenen Archivs und führt den Befehl `tar xfvz ivt-1.0.5.tar.gz` aus. Anschließend sollte sich im aktuellen Verzeichnis das Verzeichnis IVT befinden.

### 3. Konfiguration

Die IVT bietet die Möglichkeit der Konfiguration der Bibliothek, bevor sie erstellt wird. Diese Konfiguration wird in der Datei `IVT/src/Makefile.base` vorgenommen. Dazu muss `Makefile.base` mit einem Text-Editor (z.B. vim) geöffnet und bearbeitet werden. Im Folgenden werden die wichtigsten Konfigurationsmöglichkeiten aufgelistet. Dabei handelt es sich um Konfigurationsvariablen, die entweder auf 1 oder auf 0 gesetzt werden können. Um bestimmte Teile der IVT-Bibliothek einzubinden oder auszuschließen, muss die entsprechende Konfigurationsvariable auf 1 für die Einbindung und auf 0 für den Ausschluss gesetzt werden.

`USE_QT = 1`

Der Umfang der IVT-Bibliothek wird um Klassen erweitert, die es ermöglichen, auf einfache Art und Weise grafische Benutzerschnittstellen mit Qt zu erstellen. Die zugehörigen Quelldateien mit den Dateinamen `QT*` befinden sich in `IVT/src/gui`.

`USE_OPENCV = 1`

Ein Teil der IVT-Bibliothek greift auf die OpenCV-Bibliothek zurück. Diesen Teil bilden die Quelldateien mit den Endungen `CV.h` bzw. `CV.cpp`.

`USE_OPENGL = 1`

OpenGL und GLU werden von der Klasse `COpenGLVisualizer` verwendet, um Kugeln und Zylinder darzustellen. Die Quelldateien für diese Klasse befinden sich unter `IVT/src/Visualizer`.

`USE_HIGHGUI = 1`

HighGUI ist Bestandteil der OpenCV-Bibliothek. Als Alternative zu Qt lassen sich damit Fenster und Bilder grafisch darstellen. Die entsprechenden Quelldateien `OpenCV*`, die HighGUI verwenden, befinden sich in `IVT/src/gui`.

`USE_LIBDC1394 = 1`

Eine Schnittstelle zur Bibliothek `libdc1394` ermöglicht die Ansteuerung von IEEE1394-Kameras (Firewire-Kameras). Die Quelldateien der Schnittstelle heißen `Linux1394Capture.*` und befinden sich in `IVT/src/VideoCapture`.

Neben diesen Variablen lassen sich im unteren Teil der Datei die Verzeichnispfade für Include- und Bibliothekdateien sowie die Dateinamen der Bibliotheken ändern. Die vorkonfigurierten Pfade entsprechen normalerweise den Standard-Installationspfaden. Eine Anpassung der Pfade wird daher nur empfohlen, falls während der bisherigen Installation benutzerdefinierte Pfade verwendet wurden bzw. für den Fall, dass Probleme auftreten.

Für die Anpassung folgt eine kurze Erklärung der entsprechenden Variablen:

#### INCPATHS\_BASE

Enthält alle Verzeichnispfade für Include-Dateien. Das Hinzufügen eines Pfades erfolgt mithilfe des Operators += und des dem Pfad vorangestellten Parameters -I. Beispielsweise `INCPATHS_BASE += -I/usr/include/qt3`

#### LIBPATHS\_BASE

Enthält alle Verzeichnispfade für Bibliothekdateien. Das Hinzufügen eines Pfades erfolgt mithilfe des Operators += und des dem Pfad vorangestellten Parameters -L. Beispielsweise `LIBPATHS_BASE += -L/usr/lib/qt3/lib`

#### LIBS\_BASE

Enthält alle Dateinamen der Bibliotheken. Das Hinzufügen von Dateinamen erfolgt mithilfe des Operators += und des den Dateinamen vorangestellten Parameters -l. Beispielsweise `LIBS_BASE += -lqt-mt -lvtgui`

### 4. Erstellen der IVT-Bibliotheken

Die Bibliotheken werden erstellt, indem der Befehl **make** im Verzeichnis `IVT/src` ausgeführt wird. Vor jedem erneuten Ausführen von **make** sollte der Befehl **make clean** ausgeführt werden, um die IVT für eine vollständige Neu-Kompilierung zurückzusetzen.

### 5. Beispielanwendung

Zur Überprüfung der Installation oder zum Einstieg in die Programmierung mit der IVT eignet sich die Beispielanwendung `SimpleApp` im Verzeichnis `IVT/examples/SimpleApp` im IVT-Verzeichnis. Nachdem Wechseln in das Verzeichnis `SimpleApp` wird mit dem Befehl **make** eine ausführbare Datei mit dem Namen `simpleapp` erstellt. Führt man diese mit `./simpleapp ../../files/dish_scene_left.bmp` aus, so wird bei korrekter Installation aller Bibliotheken eine Datei namens `output.bmp` im selben Verzeichnis erzeugt.

Hiermit ist die Einrichtung und damit auch die Anleitung abgeschlossen. Im Verzeichnis `IVT/examples` befinden sich zahlreiche weitere Beispielanwendungen, mit denen sich die Möglichkeiten der IVT testen und erlernen lassen.