

# Computergrafik Praktikum

---

## *Hilfe zum Rahmenprogramm*

### **Inhalt**

|   |   |
|---|---|
| CMake.....  | 2 |
| Rahmenprogramm entpacken und mit CMake erstellen..... | 2 |
| Startprojekt ändern .....                             | 5 |
| Programm für die Abgabe erstellen .....               | 5 |
| Neue Quelltextdateien hinzufügen .....                | 6 |
| Ein neues Projekt hinzufügen .....                    | 7 |

Wenn Sie darüber hinaus Fragen haben oder Hilfe benötigen, können Sie mir gerne eine E-Mail an [ederer@hm.edu](mailto:ederer@hm.edu) schreiben.

## CMake



















Für die Erstellung der Projekte wird CMake (<http://www.cmake.org>) benötigt. CMake ist ein universelles Build-System, dass Visual-Studio Projektmappen oder Makefiles generiert. CMake gibt es für nahezu jede Plattform, allerdings wurden die Praktikumsaufgaben nur auf Windows 7 und Ubuntu Linux getestet.

Das Kernelement von CMake sind die „CMakeLists.txt“ Dateien, die die Build-Konfiguration enthalten. Wenn etwas an der Build-Konfiguration geändert werden muss, dann müssen die CMakeLists.txt angepasst werden, z.B. für das Hinzufügen von neuen Quelltextdateien oder von neuen Projekten. Eine entsprechende Anleitung finden Sie in diesem Dokument.

Für Linux werden zusätzlich die „X11“ Bibliotheken benötigt. Dazu bei Ubuntu das Package „xorg-dev“ installieren.

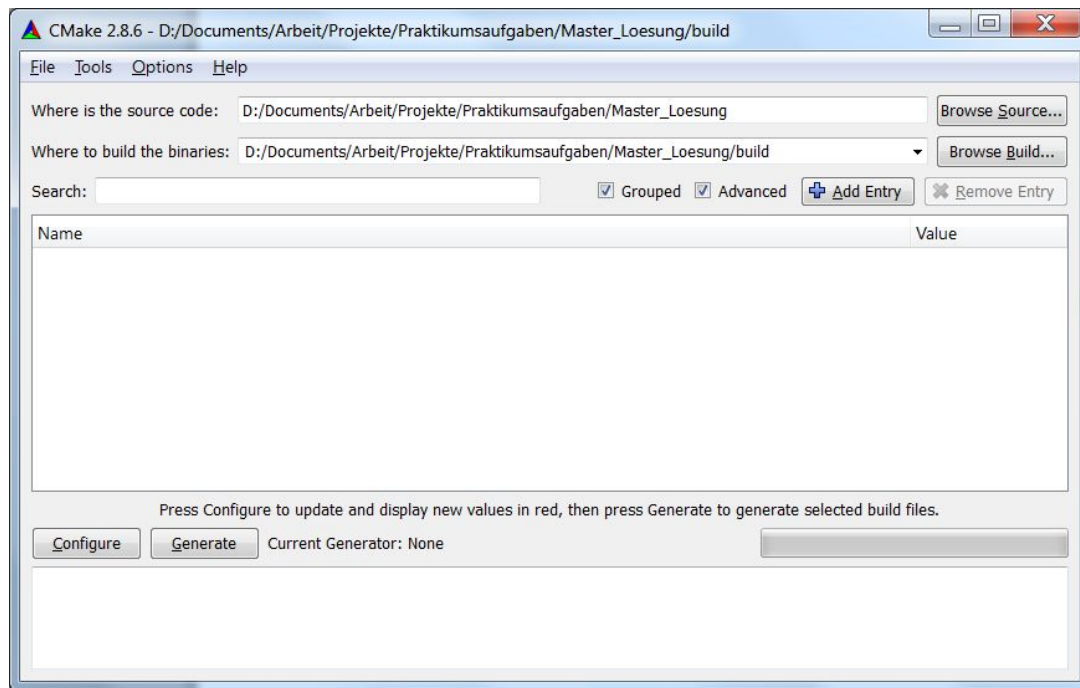
## Rahmenprogramm entpacken und mit CMake erstellen

Nach dem Entpacken der Zip-Datei sollte nun folgende Ordner-Struktur vorhanden sein:

|  |                  |             |
|--|------------------|-------------|
|  A1_FraktaleSzene                 | 17.10.2011 09:00 | Dateiordner |
|  A1_Versuch1a                     | 16.10.2011 13:50 | Dateiordner |
|  A1_Versuch1b                    | 16.10.2011 13:50 | Dateiordner |
|  A1_Versuch1c                   | 16.10.2011 13:50 | Dateiordner |
|  A1_Versuch1d                   | 16.10.2011 13:50 | Dateiordner |
|  A3_Normalenvektoren            | 16.10.2011 13:50 | Dateiordner |
|  A4_CubeMapping                 | 17.10.2011 09:00 | Dateiordner |
|  A4_TextureMapping              | 16.10.2011 13:50 | Dateiordner |
|  A5_BumpMapping                 | 17.10.2011 09:00 | Dateiordner |
|  A5_GeometryShader              | 17.10.2011 09:00 | Dateiordner |
|  A5_PercentageCloserSoftShadows | 16.10.2011 13:50 | Dateiordner |
|  build                          | 17.10.2011 11:03 | Dateiordner |
|  cmake                          | 17.10.2011 10:55 | Dateiordner |
|  Dependencies                   | 16.10.2011 13:50 | Dateiordner |
|  ImageLoader                    | 16.10.2011 13:50 | Dateiordner |
|  Modelle                        | 16.10.2011 13:50 | Dateiordner |
|  ObjLoader                      | 16.10.2011 13:50 | Dateiordner |
|  Texturen                       | 16.10.2011 13:50 | Dateiordner |

Im Folgenden wird nun gezeigt, wie man eine Visual Studio Projektmappe mit CMake erstellt. Für andere Plattformen sind die Schritte nahezu analog.

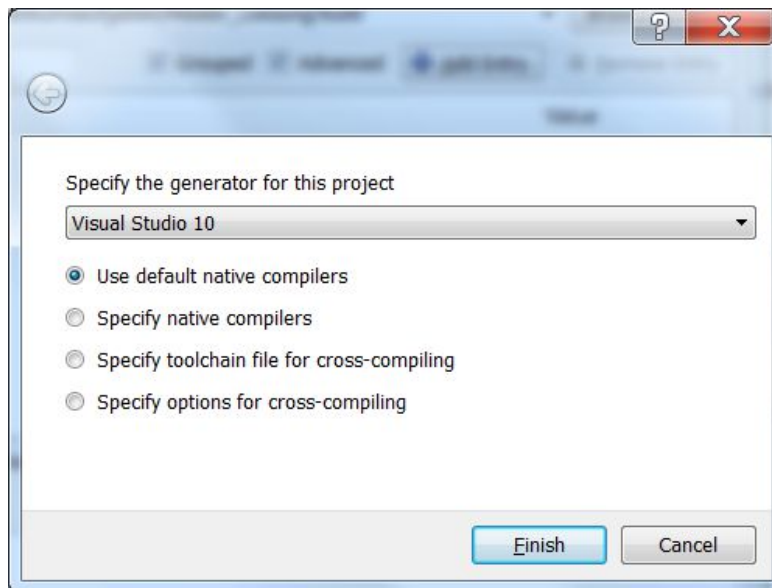
Als erstes muss CMake-GUI aus dem Startmenü gestartet werden (Befehl für Linux: cmake-gui). Es erscheint dann folgende Oberfläche:



Nun muss das Quellcode-Verzeichnis mit „Browse Source...“ ausgewählt werden (= das Verzeichnis mit der oben genannten Ordnerstruktur).

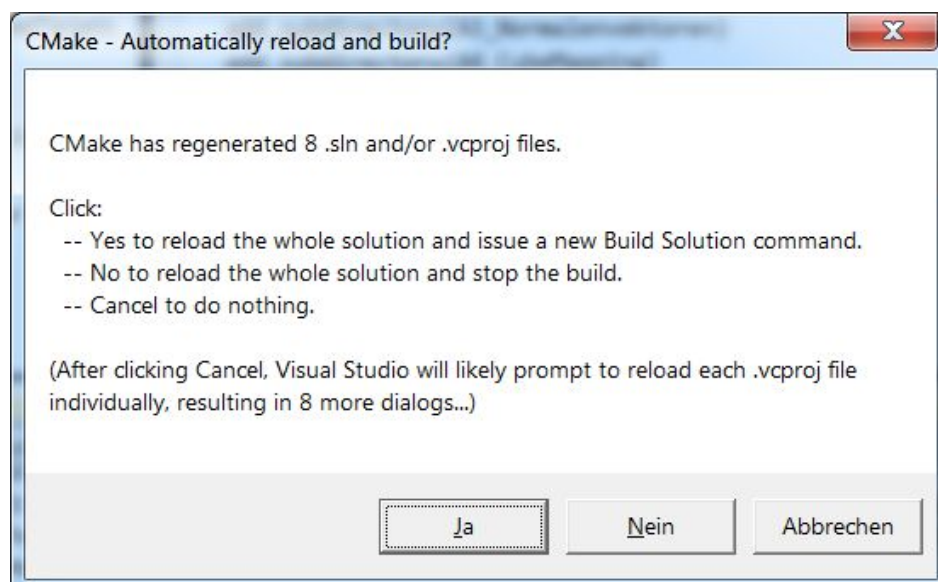
Im zweiten Schritt muss ein Build-Ordner mit „Browse Build...“ ausgewählt werden. CMake erlaubt „out of source“ Builds, bei dem die Binärdateien in separaten Verzeichnissen erstellt werden. Wenn der Quellcode der Praktikumsaufgaben kopiert werden muss, z.B. für die Heimarbeit, kann der Build-Ordner gelöscht werden, wodurch die Dateigröße enorm schrumpft. Daheim können Sie dann die Projektmappen wieder von CMake generieren lassen. Für „out of source“ Builds wird ein extra „Build“ Ordner angegeben, z.B. „build“ wie im Screenshot oben.

Nachdem die beiden Pfade gesetzt sind, erfolgt die Projekt-Konfiguration mit „Configure“. Im darauffolgenden Dialog muss dann die entsprechende Entwicklungsumgebung angegeben werden, z.B. Visual Studio 10 für die Laborrechner. Für Linux können hier entweder Makefiles, KDevelop-Projekte oder Eclipse-Projektdateien erstellt werden. Bei den Optionen noch „Use Native Compiler“ auswählen.



Nun konfiguriert CMake sämtliche benötigte Bibliothek- und Include-Pfade. Wenn eine Bibliothek nicht gefunden wurde, erscheint eine Fehlermeldung. Die Pfade müssen dann im entsprechenden Eintrag manuell hinzugefügt werden bzw. die Bibliothek installiert und „Configure“ erneut gestartet werden. Das sollte allerdings nicht passieren, da alle Bibliotheken als Quellcode mitgeliefert und erstellt werden. Wenn alles geklappt hat, kann die Projektmappe bzw. das Makefile mit „Generate“ erstellt werden. Im „Build“ Ordner befinden sich dann die Visual Studio Projektmappe bzw. das Makefile.

Danach kann die Projektmappe mit Visual Studio geöffnet und das Rahmenprogramm erstellt werden. Jede weitere Änderung an den CMakeLists.txt Dateien wird bei Visual Studio nun über Makros aktualisiert. Bei Änderungen zeigt Visual Studio 2010 dann diesen Dialog.

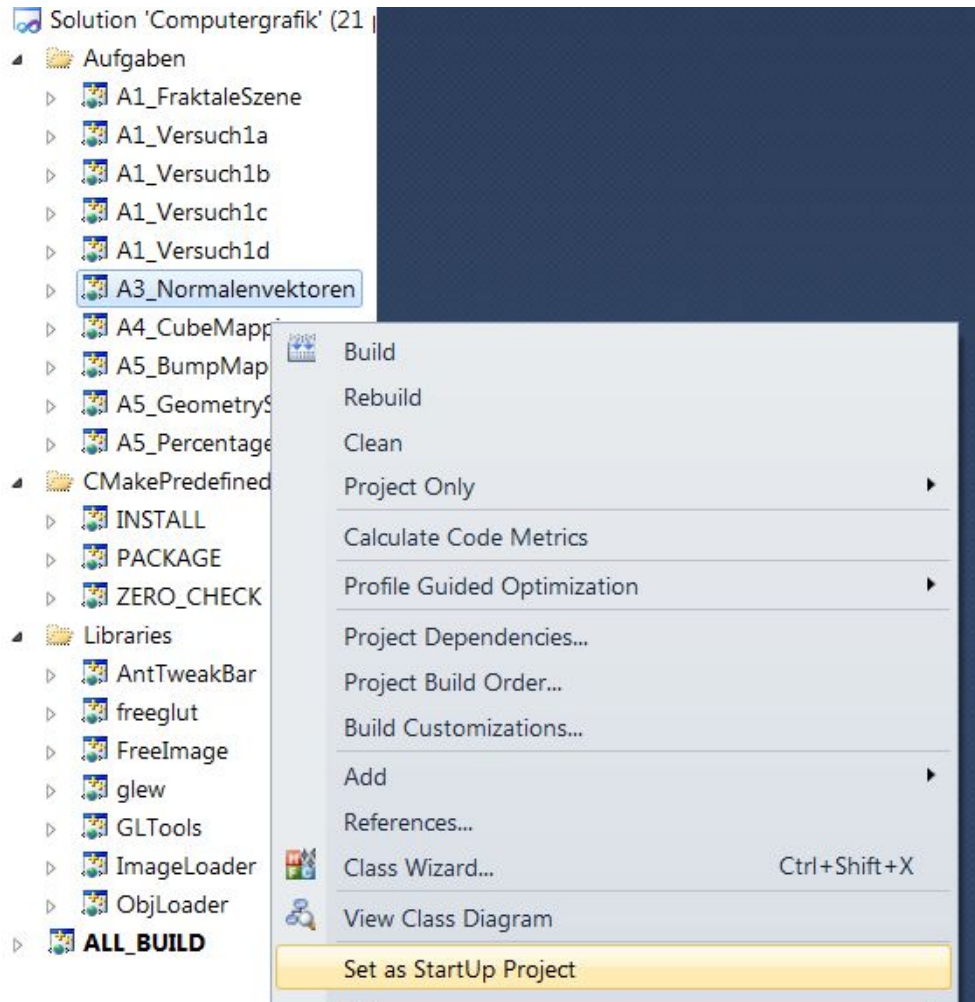


Dieser ist dann mit „Ja“ zu bestätigen und das Projekt auf Nachfrage nachzuladen. Alternativ kann natürlich auch immer die CMake-GUI benutzt werden.

## Startprojekt ändern

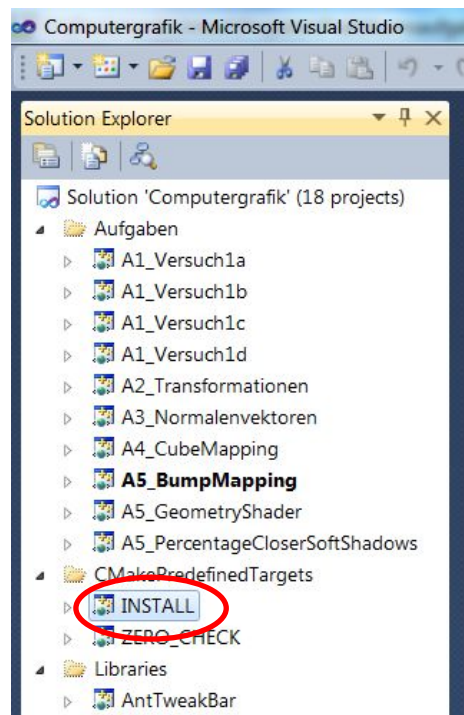
Standardmäßig wird das Projekt „ALL\_BUILD“ als Startprojekt festgelegt. Jedes Mal wenn man „F5“ bzw. der Debugger gestartet wird, wird das Startprojekt aufgerufen. Das Startprojekt wird mit fetter Schrift dargestellt.

Das Startprojekt kann geändert werden, indem in der Projektmappe mit der rechten Maustaste auf das gewünschte Projekt geklickt und „Als Startprojekt festlegen“ ausgewählt wird.

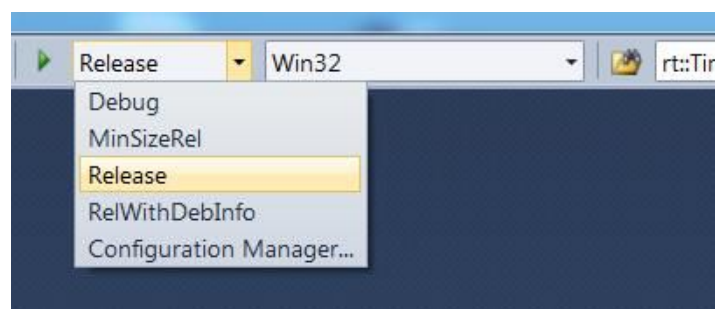


## Programm für die Abgabe erstellen

Im Rahmenprogramm ist eine Installationsroutine enthalten, die die Praktikumsabgabe erleichtern soll. Es kopiert die ausführbaren Dateien und die Quelltextdateien in den „Abgabe“ Ordner im Hauptverzeichnis. Um die Aufgaben zur Abgabe vorzubereiten muss das Projekt „INSTALL“ erstellt werden. Dazu mit der rechten Maustaste auf das Projekt klicken und „Erstellen“ auswählen. Nun werden die Dateien in den „Abgabe“ Ordner kopiert. Für Linux muss entsprechend „make install“ aufgerufen werden. Zusätzlich gibt es die Möglichkeit, den „Abgabe“ Ordner gleich mit Zip zu komprimieren. Dazu das Projekt „PACKAGE“ erstellen. Die Zip-Datei befindet sich dann im Build-Ordner.



Für die Abgabe bitte immer die „Release“ Konfiguration auswählen



## Neue Quelltextdateien hinzufügen

Um neue Quelltextdateien hinzuzufügen, muss die „SOURCES“ Variable der CMakeLists.txt des Projekts angepasst werden, z.B. im Ordner A3\_Normalenvektoren. Hier wird eine neue C++ Datei „MeineNeueCPPDatei.cpp“ hinzugefügt.

```
SET(PROJECT_NAME A3_Normalenvektoren)

set(SOURCES Aufgabe3.cpp
    MeineNeueCPPDatei.cpp
    VertexShader.glsl
    FragmentShader.glsl
)

set(LIBRARIES GLTools ObjLoader AntTweakBar freeglut)
```

Anschließend muss CMake erneut aufgerufen werden (Visual Studio: Projektmappe erstellen, den Rest machen die Makros. Linux: „Generate“ in der CMake-GUI erneut aufrufen).



**Beachten Sie, dass die Quelltextdatei auch wirklich vorhanden ist, sonst liefert CMake einen Fehler.**

Wenn das bei Visual Studio mal passiert ist, finden Sie eine leere Projektmappe vor. Um das zu beheben muss CMake-GUI erneut aufgerufen und „Generate“ ausgeführt werden.

## Ein neues Projekt hinzufügen

Um ein neues Projekt hinzuzufügen, muss zuerst die CMakeLists.txt im Hauptverzeichnis angepasst werden. Am Ende der CMakeLists.txt werden die einzelnen Projekte hinzugefügt. Dort ist ein neuer Eintrag mit „add\_subdirectory(MeinNeuesProjekt)“ hinzuzufügen. Der Projektname ist natürlich beliebig.

```
5  add_subdirectory(A3_Normalenvektoren)
6  add_subdirectory(A4_CubeMapping)
7  add_subdirectory(A5_BumpMapping)
8  add_subdirectory(A5_GeometryShader)
9  add_subdirectory(A5_PercentageCloserSoftShadows)
0
1  #neues Projekt hinzufügen
2  add_subdirectory(MeinNeuesProjekt)
```

Im zweiten Schritt muss ein neuer Ordner mit **gleichen Namen** im Hauptverzeichnis angelegt werden. Nun muss im Ordner eine CMakeLists.txt erstellt werden. Als Basis kann die CMakeLists.txt aus dem „Template“ Ordner kopiert werden. Anschließend sind die mit „TODO“ gekennzeichneten Variablen in der CMakeLists.txt anzupassen.

```
### CMakeLists.txt Template für neue Projekte

#TODO Projektnamen ändern
SET(PROJECT_NAME MeinNeuesProjekt)
#TODO Quelldateien hinzufügen, auch glsl Shader.
#mit Leerzeichen getrennt
set(SOURCES Test.cpp )
#TODO Bibliotheken anpassen
set(LIBRARIES GLTools ObjLoader AntTweakBar freeglut)
```

Anschließend muss CMake erneut aufgerufen werden (bei Visual Studio erledigen das die Makros).