HOCHSCHULE
HANNOVER
UNIVERSITY OF
APPLIED SCIENCES
AND ARTS
–
Fakultät IV
Wirtschaft und
Informatik

Übungen zur Vorlesung Computergrafik 1

Soommersemester 2018

Prof. Dr. Ingo Ginkel

Allgemeine Anforderungen:

- Implementierung in gegebenes Framework (Qt,OpenGl) mit C/C++ 1998
- Ihr Code muss lauffähig sein auf den Pool-Pcs. D.h. zum vereinbarten Termin geben Sie eine zip-Datei mit den Sourcen ab, es muss möglich sein diese durch Import der Projektdatei in QtDesigner direkt zu kompilieren.
- Es gibt weiter unten eine Installationsanleitung um sich eine VM mit (L)Ubuntu und den den benötigten Libs erstellen können, damit Sie ihr eigenes System nicht verändern müssen, falls Sie nicht direkt auf den Pool-Pcs entwickeln wollen.
- Teilweise handelt es sich um „offen“ gestellte Aufgaben, die nur die grobe Richtung für die Umsetzung und das Ergebnis vorgeben. Hier sind kreative Lösungen gefragt; es gibt nicht die eine richtige Lösung, sondern Gründe für oder gegen eine bestimmte Umsetzung. Diese Gründe sollen sie im Testat erläutern können.
- Es ist wichtig, dass Sie in ihrer Implementierung im Bereich `Controller` und `Model` keine `Qt`-Klassen, `OpenGl`-Kommandos oder sonstige API-abhängigen Dinge implementieren. In `Controller` und `Model` werden lediglich die Standard-C/C++ Bibliotheken sowie die glm-Library benutzt. So würde ein solches System in der Realität umgesetzt, daher ist dies eine zwingende Voraussetzung für die erfolgreiche Bearbeitung der Übungsaufgaben! (wird beim ersten Aufgabenblatt nochmal genauer erklärt)
- Testat: Jede/r muss alles erklären können.
- Alle Aufgaben müssen mehrfach und nacheinander ausführbar sein. D.h. sie müssen frei von Speicher-Bugs sein und mit funktionierender Reset-Funktionalität versehen sein.
- Dokumentation auf Methoden- und Klassen-Ebene: Wer hat welche Methode/Klasse geschrieben? Welche Vor-Bedingungen an die Übergabe-Parameter sind vorhanden, was ist die Funktionsweise der Methode/Klasse und wie sieht die Rückgabe der Methoden aus (im Sinne von Vor- und Nachbedingung bei Anwendung einer Methode). Dokumentation soll „in-Source“ sein vor jeden Methoden-Kopf bzw. in die Klassen-Definition (Doxygen-Style wenn sie mögen).
- Es wird von jeder Gruppe eine eigene Lösung erwartet. **Kopierte Methoden und Klassen werden als Täuschungsversuch gewertet und haben eine Bewertung mit 5,0** für alle Abgaben die dies beinhalten zur Folge, unabhängig davon wer das „Original“ und wer die „Kopie“ abgegeben hat.
- **Abgabe** der Implementierung als zip-Datei **bis Freitag 15.06. um 12:00** durch hochladen auf Moodle.
- Gruppeneinteilung: Bis Mittwoch 28.03. erhalte ich von jeder Gruppe eine E-Mail mit den Namen der Mitglieder, ich lege eine entsprechende Gruppen-Struktur in Moodle an. Die Gruppen bleiben dann über das Semester unverändert.

Softwareentwicklung allgemein:

- Es gibt immer wieder Aspekte der allgemeinen Softwareentwicklung, die in der Umsetzung der Grafik-Aufgabenstellung genutzt werden. Hauptsächlich sind dies so genannte Design Patterns. Dies sind bewährte Lösungsschablonen für wiederkehrende Entwurfsprobleme in der Softwareentwicklung. Sie stellen eine wiederverwendbare Vorlage zur Problemlösung dar, die in einem bestimmten Zusammenhang einsetzbar ist. Literatur dazu:
  - *Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides* : Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley 1994, Bibliothek `EDV 4640 /62(E)`
  - Deutsche Übersetzung in der Bibliothek vorhanden: `EDV 4640 /62(NA):1`
  - Auch ein Blick in Wikipedia `https://de.wikipedia.org/wiki/Entwurfsmuster` kann nicht schaden.
  - Thema in SE1 Ende Mai, schadet also nicht wenn sie Teile davon jetzt schon mal gesehen haben.

Libraries Computergrafik

Als Hilfsmittel werden die Libraries `glm` und `Qt 5.x` benutzt.

- Die `glm`-Library: „OpenGL Mathematics (GLM)“ ist eine header only C++ Mathematik-Library für Grafik-Software und stellt Funktionalität für die Vektor- und Matrizenrechnung zur Verfügung. Ergänzt wird dies um viele grafik-spezifische Hilfsmethoden, die wir teilweise nutzen und teilweise selbst implementieren werden um die Funktionalität zu verstehen. Diese Library ist auch deswegen sinnvoll, weil sie die gleiche Syntax verwendet wie später die Programmier-API `GLSL` für so genannte Shader-Programme, die direkt auf der Grafik-Karte ausgeführt werden.
  - `http://glm.g-truc.net/glm.pdf`, Datei liegt auch in Moodle.
- `Qt` dient zur graphischen Darstellung, für Buttons, Slider etc. um Nutzer-Interaktion zu steuern und Parameter einzustellen. Ein sehr mächtiges plattformübergreifendes Paket, das für jede Art von GUI geeignet ist. Es beinhaltet ebenso eine eigene OpenGl Implementierung, die dazu dient 3-dimensionale graphische Objekte darzustellen.
  - Ausführliche Dokumentation in `https://doc.qt.io/qt-5.9/index.html`.
  - Falls Sie ein eigenes System mit einer möglicherweise älteren Version nutzen sollte dies auch funktionieren, solange es eine 5.x Version ist.

Kurze Installationsanleitung für ein blankes (l)Ubuntu 17.10 (oder 16.04) System

- Ausgangspunkt ist ein frisch installiertes (L)Ubuntu 17.10 System, an dem noch nichts weiter konfiguriert wurde. Habe ich auch in einer VM getestet, funktioniert ebenfalls. Grafikkarte muss OpenGl 3.3 unterstützen.
- Installieren Sie mit dem Befehl `sudo apt-get install` folgende Libs: `libglm-dev, libqt5opengl5-dev, qtcreator`, da kommen natürlich noch einige Abhängigkeiten mit.

- Entpacken Sie CG1Uebung.zip (liegt in Moodle) in ein Verzeichnis ihrer Wahl.

- Öffnen Sie die Datei CG1Uebung.pro in QtCreator und klicken Sie auf Projekt konfigurieren, dann auf Play. Es sollte sich ein Fester öffnen, in dem zwei Dreiecke zu sehen sind, die mit der Maus gedreht werden können.

Arbeit mit dem zur Verfügung gestellten System

- Für Sie relevant sind vor allem die Klassen `CgQtGui` und `CgSceneControl`. Diese können und sollen sie nach Belieben modifizieren. Ebenso die Beispielklasse `CgExampleTriangle`. Enums können in `CgEnmus.h` hinzugefügt werden, neue Klassen sowieso.
- Bitte führen Sie **keine Änderungen** an allen Klassen mit `Base` im Namen durch also
  - `CgBaseEvent, CgBaseRenderableObject, CgBaseTriangleMesh, CgBase....`

  und auch nicht in den Klassen `CgQtGLRenderWidget` und `CgQtGLBufferObject` .

  Dies ist auch nicht nötig!

  - Diese Klassen werden von mir im Laufe des Semester weiterentwickelt um z.B. Material und Texturen verarbeiten zu können,
  - Das `Base`-Interface wird nach Bedarf für die weiteren Aufgaben erweitert.
  - **D.h. Sie werden gezwungen sein die entsprechenden Dateien in ihrem System im Laufe des Semesters durch neue auszutauschen**, daher keine Änderungen ihrerseits....