Labyrinth - Dokumentation

Projekt in Computergrafik 1

Von Laszlo Garab, Simorg Ajun Jöller und Oliver Hörner

## Inhalt

[Das Projekt 3](#_Toc360302238)

[Plattform und System 3](#_Toc360302239)

[Team und Aufwand 3](#_Toc360302240)

[Features 4](#_Toc360302241)

[Ausführen des Programms 5](#_Toc360302242)

[Buildanweisungen 5](#_Toc360302243)

# Das Projekt

Ziel unseres Projekts ist es ein Programm zur Verfügung zu stellen welches ein simples Labyrinth aus Zeichen in einem Textfile in eine 3D Visualisierung umwandelt.

Jeder User kann selbstständig Labyrinthe entwerfen und ohne weitere spezielle Kenntnisse oder komplizierte Schritte mit Hilfe unseres Programms darin „herumirren“.

## Plattform und System

Labyrinth ist eine Win32 Applikation und kann daher auf entsprechenden Windows-Systemen ausgeführt werden.

Als Entwicklungsumgebung wurde *Visual Studio 2012* verwendet und das Programm ist in C++ geschrieben.

Des Weiteren wurde natürlich openGL mit GLUT sowie der in der Lehrveranstaltung zur Verfügung gestellte TGA-Loader verwendet.

Alle verwendeten Texturen sind ein Gratis-Download von [www.turbosquid.com](http://www.turbosquid.com).

## Team und Aufwand

**Laszlo Garab**, zuständig für:

Texturen laden und darstellen

2D Textdarstellung

**Simorg Ajun Jöller**, zuständig für:

Userinteraktion Logik

Zeitmessung

**Oliver Hörner**, zuständig für:

Textdatei überprüfen und Konvertieren

Display Logik

Collision Detection

Dokumentation

Zusätzlich wurde von allen Personen Licht gleichermaßen bearbeitet, allerdings konnten wir es leider nicht in einem brauchbaren Zustand ins fertige Projekt implementieren ohne andere Teile zu vernachlässigen.

Gesamt wurden in etwa 90-100 Stunden in dieses Projekt investiert, wobei ein Großteil davon Einarbeiten und Verstehen der verwendeten Materie war.

# Features

**Ein komplett dynamisch erstelltes Labyrinth** basierend auf einer Textdatei als Vorlage. Die Textdatei muss dabei nur einigen Regeln entsprechen:

* Mauern werden durch Raute („#“) und Gänge durch Punkte („.“) gesetzt
* Mehrere Punkte „.“ nebeneinander ergeben Kammern
* Das Labyrinth muss rechteckig und durch einen Rahmen aus Raute „#“ begrenzt sein.
* Das Labyrinth muss einen Startpunkt „S“ und Endpunkt „E“ haben. Diese Punkte verhalten sich wie normale Gänge „.“ Und können beliebig gesetzt werden
* Die maximale Größe des Labyrinths ist 100x100 Zeichen, inklusive Rahmen
* Die Textdatei darf keine leeren Zeilen enthalten

C:\Users\Oliver\Desktop\bsp.jpgEin einfaches Beispiel wäre:

**Intuitive Interaktion** ähnlich alter „Dungeon Crawler“. Mit [W] und [S] kann man sich durch das Labyrinth bewegen, mit [A] und [D] wird die Sicht gedreht. Per [Esc] wird das Programm beendet.

Bewegt wird immer um genau einen „Block“ bzw. 90°. Wird während der Drehung/Bewegung eine Taste gedrückt wird diese ausgeführt sobald die derzeitige Bewegung zu Ende ist.

Dies führt trotz der etwas limitierten Art der Interaktion zu einem sehr flüssigen Bewegungsablauf.

**Collision Detection**, es wäre kein echtes Labyrinth wenn man durch Wände gehen könnte.

**„Dungeon Texturen“**, die dem Spieler ein gewisses Gefühl der Enge und „Eingesperrtheit“ übermitteln sollen.

**Zeitmessung**, es wird angezeigt wie lange der Spieler gebraucht hat ans Ende zu kommen.

Ursprünglich geplant war eine Lichtquelle zu implementieren die sich mit dem Spieler mitbewegt, ähnlich einer Art Fackel die man halten würde. Leider haben wir den Aufwand hierfür unterschätzt (Stichwort: Normalvektorberechnung) und uns letztendlich lieber auf andere Teile konzentriert.

# Ausführen des Programms

Vor dem Starten des Programms sicherstellen das die Ordner „**Files**“ und „**Textures**“ sowie die „**glut32.dll**“ sich im selben Ordner wie die Anwendung „**Labyrinth.exe**“ befinden.

Beim Start des Programms wird man aufgefordert den Namen des Textfiles anzugeben, welches geladen werden soll. Die Anwendung wird in Folge den eingegebenen Namen mit „.txt“ verknüpfen und in „Files“ danach suchen.

Alle eventuell auftretenden Fehler und weitere Instruktionen können der Konsole entnommen werden.

# Buildanweisungen

Alle Sourcen und andere Abhängigkeiten sind in der Visual Studio Solution inkludiert. Es sollte daher keine Hindernisse geben die Anwendung zu builden.

Eine Expressversion kann unter <http://www.microsoft.com/visualstudio/> kostenfrei heruntergeladen werden.

Es ist noch zu erwähnen, dass wenn die Anwendung aus Visual Studio heraus gestartet wird die Ordner „Files“ und „Textures“ sowie „glut32.dll“ aus demselben Ordner herangezogen werden in dem die Source-Dateien liegen.