|  |
| --- |
| http://blog.gfx47.com/wp-content/uploads/2011/02/unity3d1.jpg  Unity3D in CAVE  Pflichtenheft  **Julien Villiger, Daniel Inversini**  **V1.00, 04.03.2015** |
| **Berner Fachhochschule**  Technik und Informatik  Informatik |

Inhaltsverzeichnis

[1 Allgemeines 3](#_Toc413857349)

[1.1 Zweck dieses Dokumentes 3](#_Toc413857350)

[1.2 Lesekreis 3](#_Toc413857351)

[1.3 Ausgangslage 3](#_Toc413857352)

[1.4 Umfang der Projekt 2 Arbeit 3](#_Toc413857353)

[1.5 Ziele der Arbeit 3](#_Toc413857354)

[1.6 Abgrenzungen 4](#_Toc413857355)

[1.6.1 Technische Abgrenzungen 4](#_Toc413857356)

[1.6.2 Weitere Abgrenzungen 4](#_Toc413857357)

[1.7 Voraussetzungen und Ressourcen 4](#_Toc413857358)

[2 Voranalysen 5](#_Toc413857359)

[2.1 Equalizer 5](#_Toc413857360)

[2.2 Chromium Bibliothek 5](#_Toc413857361)

[2.3 MiddleVR 5](#_Toc413857362)

[2.4 Unity Standalone 5](#_Toc413857363)

[3 Prototyping 5](#_Toc413857364)

[4 Funktionale Anforderungen 6](#_Toc413857365)

[4.1 Adaption Unity Anwendung für CAVE 6](#_Toc413857366)

[4.2 Kompatibilität 6](#_Toc413857367)

[4.3 Plattformunabhängigkeit 6](#_Toc413857368)

[5 Nicht Funktionale Anforderungen 6](#_Toc413857369)

[5.1 Presence 6](#_Toc413857370)

[5.2 Wiederverwendbarkeit 6](#_Toc413857371)

[5.3 Ergonomie 6](#_Toc413857372)

[6 Testing 7](#_Toc413857373)

[6.1 Systemtests 7](#_Toc413857374)

[6.2 Usabilitytests 7](#_Toc413857375)

[7 Administratives 8](#_Toc413857376)

[7.1 Projektorganisation 8](#_Toc413857377)

[7.1.1 Projektteam 8](#_Toc413857378)

[7.1.2 Betreuer 8](#_Toc413857379)

[7.2 Projektplan 8](#_Toc413857380)

[7.3 Meilensteine 8](#_Toc413857381)

[7.3.1 Voranalyse 8](#_Toc413857382)

[7.3.2 Prototyp 8](#_Toc413857383)

[7.3.3 Dokumentierte API/Einstellungen der Lösung/Prototyp 8](#_Toc413857384)

[8 Versionskontrolle 10](#_Toc413857385)

# Allgemeines

## Zweck dieses Dokumentes

Mit diesem Pflichtenheft wird der Rahmen, die Vorgehensweise und die Ziele der Projekt 2 – Arbeit dokumentiert.

## Lesekreis

Der Inhalt dieses Dokumentes richtet sich in erster Linie an den Betreuer dieser Arbeit, Prof. Urs Künzler, die BFH-TI Abteilung CPVR und an die Studenten, welche diese Projektarbeit durchführen.

## Ausgangslage

Das cpvrLab besitzt eine CAVE Installation (Cave Automatic Virtual Environment) mit dem virtuelle 3D-Welten in Echtgrösse über drei Projektionswände und eine Bodenprojektion erzeugt können. Alle Projektionsflächen werden dabei mit zwei Projektoren in Stereo projiziert, sodass eine nahezu perfekte Raumwahrnehmung entsteht.

Die Entwicklung von virtuellen 3D-Welten mit Basis-APIs wie OpenGL oder OpenScenegraph ist nach wie vor eine zeitraubende und aufwendige Arbeit und jedes Mal eine Einzelentwicklung.

Es liegt deshalb nahe, eine High-Level Game Engine einzusetzen, mit der die Entwicklungszyklen

vereinfacht und verkürzt werden können. Unity hat sich in den letzten Jahren in diesem Bereich durchgesetzt und ermöglicht es Studenten gratis damit Spiele zu entwickeln. In der professionellen Version bietet Unity viele, z.T. sehr fortgeschrittene Erweiterung um die Realitätsnähe weiter zu steigern.

## Umfang der Projekt 2 Arbeit

Prinzipiell ist geplant, einen lauffähigen Prototypen zu erstellen. Dies kann ein Spiel oder auch eine Techdemo sein.

Wenn der Prototyp/die Prototypen innerhalb der Projektarbeit in den CAVE portiert werden können, ist es möglich, noch weitere VR Aspekte wie Tracking zu überprüfen.

Falls es nicht komplett abgeschlossene Punkte/Features gibt, welche für einen voll funktionsfähigen Prototypen zwingend sind, werden diese aufgenommen.

Im Rahmen einer weiteren Projektarbeit oder einer Bachelorarbeit können fehlende Komponenten/Features realisiert oder wie erwähnt zusätzliche VR Aspekte implementiert werden.

## Ziele der Arbeit

Folgendes sind die Ziele:

1. Einarbeiten in Unity
2. Einarbeiten in die Theorie der 3D Stereo Projektion.
3. Entwicklung eines Demospiels/Anwendung/Demo für den cpvrLab Cave mit Stereoprojektion.
4. Dokumentierte API der schlussendlich verwendeten Lösung
5. Einarbeiten in die Features von Unity Pro Edition

Wobei Punkt 5 sekundär zu betrachten ist, denn u.U. ist eine Pro Edition von Unity nicht nötig ist.

## Abgrenzungen

### Technische Abgrenzungen

Da wir eine saubere, wartbare und moderne Applikation/Lösung bieten wollen, möchten wir uns falls möglich auf Unity, respektive C# begrenzen. Low-Level Implementationen in C, C++ (auch FreeGLut) möchten wir keine vornehmen.

### Weitere Abgrenzungen

Im Rahmen der Projekt 2 Arbeit sind lauffähige Prototypen genügend. Komplette Setups und Schulungen sind nicht vorgesehen.

## Voraussetzungen und Ressourcen

Folgende zwei Voraussetzungen sind zwingend für die Studenten, welche dieses Projekt durchführen:

1. Besuch des Unity Kurses an der BFH (BTI7527a – Game Development)
2. Besuch der Vertiefungsrichtung CPVR (<http://www.cpvrlab.ti.bfh.ch/>)

Desweitern muss der Zutritt zu den Räumlichkeiten der BFH, wo sich die Installation des CAVEs befindet, sichergestellt werden.

# Voranalysen

## Equalizer

Equalizer ist ein Open Source Framework für skalierbares, paralleles Rendering basierend auf OpenGL, welches ein API zur Verfügung stellt um solche graphischen Applikationen zu entwickeln. Es verwendet verschiedenste Wrapperklassen in C++, um die Systemressourcen abstrahiert darzustellen.

Equalizer wird hier analysiert, da es bereits im CAVE der BFH in Verwendung ist.

## Chromium Bibliothek

Der OpenGL Command wird nicht direkt in ein Rasterbild umgewandelt, sondern wird manipuliert und an andere OpenGL Implementationen weitergeschickt. Somit sollte es möglich sein, auf jedem Client den Command entsprechend für die spezifische Leinwand anzupassen. Die Machbarkeit dieser Methode wird geprüft.

## MiddleVR

Der französische Hersteller Creative Valley hat ein Plugin entwickelt, um Unity Anwendungen in einem CAVE stereoskopisch darstellen zu können. Die Machbarkeit dieser Methode wird geprüft.

## Unity Standalone

Möglicherweise bietet kein Hilfsmittel genügend Möglichkeiten und es muss eine eigene Lösung erarbeitet werden.

# Prototyping

Basierend auf den Voranalysen werden ein oder mehrere Prototypen erstellt, welche zu einem späteren Zeitpunkt natürlich auch im CAVE lauffähig installiert werden müssen.

Da der Aufwand der Installation im CAVE schlecht abschätzbar ist, sowie der Prototyp je nach gewähltem Ansatz unterschiedlich sein kann (eigene Lösung, die Verwendung eines Frameworks/API), ist hier die Verteilung der Aufwände nicht gegeben.

Daher werden diese beiden Punkte zusammen im Prototyping behandelt.

# Funktionale Anforderungen

## Adaption Unity Anwendung für CAVE

Beliebige Spiele, Simulationen oder sonstige Anwendungen die mit Unity umgesetzt wurden, sollen so manipuliert werden, dass auf sämtlichen Leinwänden des CAVEs eine stereoskopische Projektion dargestellt wird.

## Kompatibilität

Sämtliche, quelloffene Unity Anwendungen Version 4.6 müssen mit dem umgesetzten System kompatibel sein.

Der Export der Unity Anwendung muss für das spätere Einpflegen in den CAVE für Windows Desktop erfolgen.

## Plattformunabhängigkeit

Durch die plattformunabhängige Architektur von Unity können die Anwendungen im Rahmen der Möglichkeiten von Unity umgesetzt werden.

# Nicht Funktionale Anforderungen

## Presence

Die mentale Immersion wird durch den CAVE im Vergleich zur normalen Ausführung der Unity Anwendungen deutlich gesteigert. Der Benutzer fühlt sich geistig in die virtuelle Welt versetzt und soll möglichst gering durch Input-Devices beeinträchtig werden.

## Wiederverwendbarkeit

Damit zukünftige Entwickler effizient eigene Anwendungen in den CAVE einpflegen können, wird viel Wert auf die Wiederverwendbarkeit gelegt.

Anwender aus verschiedenen Bereichen wie Architektur, Autoindustrie, Game Development usw. können ihre Simulationen in den CAVE einpflegen und ausführen.

## Ergonomie

Im Rahmen eines kleinen Tutorials wird Schritt für Schritt erklärt, wie die eigene Unity Anwendung für den CAVE aufbereitet werden kann.

# Testing

## Systemtests

Während der Prototypingphase werden laufend Tests auf unabhängigen Rechnern sowie im CAVE durchgeführt um sicherzustellen, dass während der Entwicklung mögliche Probleme sofort erkannt werden und dagegengesteuert werden kann.

## Usabilitytests

Abhängig vom Fortschritt der Prototypen werden Tests mit potenziellen Anwendern durchgeführt um die Usability der Lösung abschätzen und optimieren zu können. Sowohl die Inbetriebnahme des CAVEs wie auch die Adaption der eigenen Unity Anwendungen werden berücksichtigt.

# Administratives

## Projektorganisation

Auf eine stark strukturierte Projektorganisation wird bewusst verzichtet. Die Teammitglieder sind gleichberechtigt. Es kann vorkommen, dass verschiedene Teilprojekte, Verantwortungsbereiche zugewiesen werden. Dies bedeutet aber nicht die alleinige Durchführung desselben.

### Projektteam

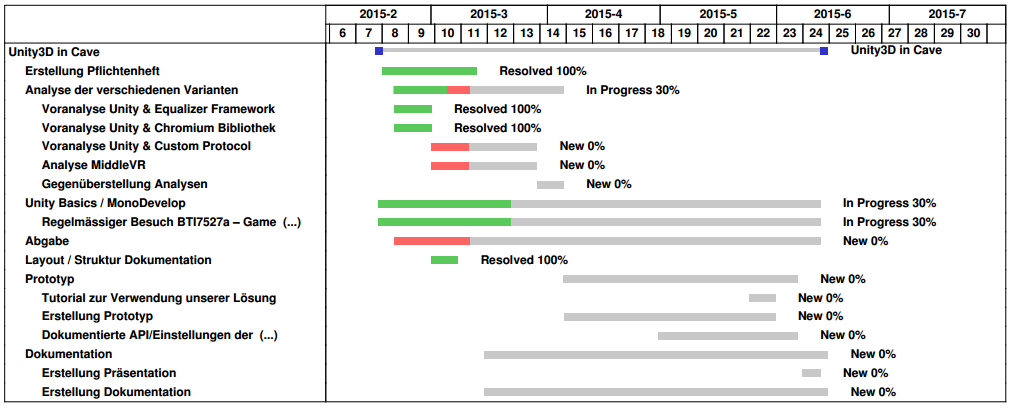
Daniel Inversini [daniel.inversini@students.bfh.ch](mailto:daniel.inversini@students.bfh.ch)

Julien Villiger [julien.villiger@students.bfh.ch](mailto:julien.villiger@students.bfh.ch)

### Betreuer

Prof. Urs Künzler [urs.kuenzler@bfh.ch](mailto:urs.kuenzler@bfh.ch)

## Projektplan



## Meilensteine

Folgende Tasks aus dem Projekt wurden als Meilensteine definiert:

### Voranalyse

Stichtag 29.03.2015

Analyse und Gegenüberstellung der verschiedenen Varianten, Entscheidung für eine dieser Varianten.

### Prototyp

Stichtag 31.05.2015

Lauffähiger Prototyp vorhanden im CAVE der BFH.

### Dokumentierte API/Einstellungen der Lösung/Prototyp

Stichtag 06.06.2015

Der Prototyp, die Machbarkeitsstudie sollte soweit modular und flexibel sein, dass eine nachfolgende Arbeit (Bachelorarbeit oder andere Projektarbeiten) einfach eingebunden werden kann. Dazu dient hier die Dokumentation.

# Versionskontrolle

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Datum** | **Beschreibung** | **Autor** |
| 0.1 | 04.03.2015 | Dokument erstellt / Struktur definiert | Daniel Inversini |
| 0.2 | 04.03.2015 | Struktur überarbeitet | Julien Villiger |
| 1.0 | 10.03.2015 | Update | Julien Villiger |