Systemspezifikation Vortex-Tunnel

|  |  |  |
| --- | --- | --- |
| **Version** | **Beschreibung** | **Bearbeiter** |
| 1.0 | Erster Entwurf des Berichts | Marc Nussbaumer |
|  |  |  |
|  |  |  |
|  |  |  |

Inhaltsverzeichnis

[Voraussetzungen für den Aufbau 3](#_Toc499716106)

[Mögliche Stolpersteine beim Einrichten 3](#_Toc499716107)

[Einrichtung der VR-Umgebung 4](#_Toc499716108)

[Konfiguration des Tunnels über die XML-Datei 7](#_Toc499716109)

[Standort der XML-Datei 7](#_Toc499716110)

[XML-Struktur 8](#_Toc499716111)

[Erklärung Parameter 9](#_Toc499716112)

[Vortex-Tunnel 9](#_Toc499716113)

[Abschnitte 9](#_Toc499716114)

[Abschnitt 10](#_Toc499716115)

[Wandmuster 10](#_Toc499716116)

[Textur 11](#_Toc499716117)

[Lichter 12](#_Toc499716118)

[Farbe 13](#_Toc499716119)

## Einführung

Im Auftrag des iHomeLab der Hochschule Luzern soll nicht technikaffinen Personen «Virtual Reality» anhand eines sogenannten Vortex-Tunnels nähergebracht werden. Ein Vortex-Tunnel ist eine drehende Röhre, welche von einer Person auf einem Steg durchschritten werden soll. Dabei ist es in der Röhre dunkel, sichtbar sind nur sich drehende Lichtpunkte und Muster.

Mit diesem Aufbau lässt sich der Gleichgewichtssinn der Testpersonen täuschen.

# Anforderungen

Gemäss Aufgabenstellung sind folgende Ergebnisse zu erarbeiten:

* Konzept zur Erreichung des maximalen Effekts auf den Gleichgewichtssinn der Probenden
* Erstellung Softwarekonzept und Testkonzept
* Softwareprototyp mit Unity3d und Oculus Rift oder HTC Vive
* Testresultate mit den Probanden und Diskussion der erarbeiteten Ergebnisse
* Code, Dokumentation und HowTo’s auf GitHub.

Nach einem Workshop mit Herr Biallas (siehe Anhang «20171011\_Protokoll Sitzung Vortex-Tunnel») wurden folgende konkreten Anforderungen für den Tunnel definiert:

Muss:

* Benutzer soll sich durch laufen durch den Tunnel bewegen können
* Es existiert ein gerades Tunnelmodell durch welches gelaufen werden kann
* Der Tunnel soll per XML konfigurierbar sein

Kann:

* Der Tunnel soll Kurven aufweisen

Abgrenzung der Ziele:

* Kurven sollen nur im 90° Winkel konzipiert werden
* Lichtmuster sollen zufällig sein
* Form der Lichtpunkte ist beliebig
* Bodenbeschaffenheit sollen durch 3 Verschiedene Konfigurationen gewählt werden
* Tunnel ist immer ein Kreis / Zylinder
* Minimale Toleranz zu Konfiguration ist mindestens 10cm

Was ist nicht Teil der Ziele (noscope)?

Änderungen während des Projektes:

Basierend auf den Erkenntnissen aus der Versuchsdurchführung (siehe Anhang «XYXXXX») wurde an der Sitzung vom 02.12.2017 folgende Änderungen an der Zielsetzung durchgeführt:

Die Kann-Anforderung der Kurve fällt weg, stattdessen soll die Konfigurierbarkeit des Stegs mit folgenden zusätzlichen Parametern erweitert werden:

* + ***Handgeländer ein/ausschaltbar***
  + ***Breite des Stegs***
  + ***Transparenz-Grad des Stegs***
  + ***Textur***
  + ***Höhe***

-Kurve fällt weg -> Fokus auf Steg

-Ergänzung Steg -> Versuchsresultate

-90° Kurve und Gerader Steg -> Sitzung Klärung Aufgabenstellung

Flexible Parameter -> Sitzung Klärung Aufgabenstellung

Mehrere Abschnitte -> Sitzung Zwischenpräsentation

System

Exe zu SteamVR

Umgebung

Systemübersicht

Komponentendiagramm

Schnittstellen

Steam-VR

# Allgemeine Entwurfsentscheide

## Effektgenerierung der Drehung

Als Entscheidungsbasis für das weitere Vorgehen zur Effekterzeugung wurden drei Prototypen mit unterschiedlichen Ansätzen erstellt und anschliessend in einer Sitzung vorgestellt.

### Prototyp Pointlight

PG -> Insbesondere Drehung um Mittelpunkt genauer beschreiben

### Prototyp Spotlight

PG

### Prototyp Textur

Nebst der Verwendung von vorgefertigten Unity-Lichteffekten wurde zusätzlich eine mögliche Lösung mittels Innen-Textur des Tunnel-Modells untersucht. Dabei wird auf jegliche Beleuchtung ausser dem Ambient-Light verzichtet. Stattdessen werden die Lichtpunkte durch Punkte auf der Grafik für die Textur dargestellt.

Um eine Textur auf einem Modell abzubilden ist eine sogenannte UV-Map notwendig, diese bestimmt welche Bildpunkte auf welche Modell-Punkte abgebildet werden sollen (siehe XXX).

Um die Textur zu drehen wurden folgende Wege untersucht:

* Drehung des Modells
* Drehung der Kamera
* Drehung der Textur

Der Effekt konnte durch eine Rotation an Z-Achse für die Drehung des Modells und der Kamera erzielt werden. Jedoch ist der Nachteil, dass diese Methoden nicht verwendet werden können in Zusammenhang mit Kurven. (Siehe Abbildung XYZ)

Stattdessen wurde der Ansatz in welchem die Textur selbst bewegt wird weiterverfolgt, da dort das Problem einer Kurve gelöst werden kann. (Mittels UV? Wie gelöst -> genauer erläutern)

Die zwei funktionierenden Prototypen wurden an der Sitzung vom 08.11.2017 (siehe Anhang «20171108\_Protokoll Zwischenpräsentation Vortex-Tunnel») präsentiert. Es wurde dabei entschieden beide Lösungsansätze zu verknüpfen und in zukünftigen Versionen zu verwenden.

## Varianten zur Parameter-Eingabe

In den folgenden Abschnitten werden verschiedene Varianten zur Parameter-Eingabe angeschaut und im Anschluss verglichen. Die Parameter-Eingabe muss nicht durch den Endbenutzer erfolgen, sondern durch Vertreter des iHome-Labs. Aus diesem Grund muss die Eingabe nicht zwingend über ein im Programm integriertes GUI erfolgen. Das Verändern der Parameter soll aber das Öffnen von Unity/einer IDE nicht voraussetzen.

### Variante 1: Eingabemaske beim Start der Tunnel-Simulation

Die Parameter können direkt vom Benutzer, der die VR-Brille trägt, vor Start der Simulation gesetzt werden. Unity bietet einfache Steuerelemente an, die dies ermöglichen.

### Variante 2: Parameter in einem XML-File

Die Parameter werden strukturiert in einem XML-File eingegeben. Anschliessend wird die VR-Simulation gestartet und die Werte werden aus der Datei übernommen.

### Variante 3: Ein simpler Editor zum Tunnelbau

Ein simples Tool zum Zeichnen des Tunnels. Die Benutzeroberfläche des Tools könnte den Tunnel aus der Vogelperspektive darstellen und das Setzen von neuen Tunnelabschnitten ermöglichen. Das Tool wird eine Datei erzeugen (xml, txt oder anderes Format), die dann beim Programmstart eingelesen und zur Generierung des Tunnels verwendet wird. Im Vergleich zu den ersten beiden Varianten wird bei dieser Lösung kein zufälliger Tunnel generiert.

### Vergleich der drei Varianten

Zur Auswahl der besten Variante werden alle Konzepte verglichen. Die Kriterien an eine Parametereingabe sind die folgenden:

* Einfachheit der Eingabe: Die Parametereingabe soll möglichst kein technisches Know-How erfordern. (1= Gelernter Informatiker, 3=Technischer Assistent, 5= Microsoft-Office Anwender)
* Komfort: Wie angenehm ist es für den Benutzer, die Parameter einzugeben. (1=Mehrere Zusatzprogramme notwendig, 3=Ein Zusatzprogramm notwendig, 5=Alles in der Applikation)
* Entwicklungsaufwand: Wie aufwändig ist die Entwicklung? (1= sehr aufwändig und grosser Zeitaufwand, 3=tragbarer Aufwand ohne zusätzliche Kenntnisse/Libraries, 5=wenig Aufwand und kann gut in C# umgesetzt werden.)
* Genauigkeit: Wie gezielt und präzise können die Parameter gesetzt werden? (1=grobe Definition ohne Massangabe, 3=Masse in bestimmten Intervallen, 5=freie Massangaben)
* Erweiterbarkeit: Wie einfach können neue Parameter hinzugefügt werden? (1=Entwickler für die Anpassung benötigt, 3=Technischer Assistent, 1=Laie)

Die drei Varianten wurden an den genannten Kriterien gemessen und verglichen. Jedes Kriterium wurde zusätzlich gewichtet.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Gewichtung** | **Variante 1 -GUI** | **Variante 2 -XML** | **Variante 3 -Editor** |
| **Einfachheit** | 3 | 5 | 3 | 5 |
| **Komfort** | 2 | 3 | 3 | 4 |
| **Entwicklungsaufwand** | 3 | 2 | 5 | 1 |
| **Genauigkeit** | 1 | 3 | 5 | 3 |
| **Erweiterbarkeit** | 1 | 1 | 5 | 1 |
| **Gesamt** | 10 | 31 | 40 | 30 |

Aus dem Vergleich geht hervor, dass sich die Parameter-Eingabe über eine XML-Datei am besten eignet. Deshalb wird dieser Ansatz weiterverfolgt.

Gittersteg -> Durchsichtig vor Versuch -> Rückmeldung Zwischenpräsentation

Tunneldrehung -> Textur gewählt beschreiben

Stegkonfiguration -> Aufgrund Rückmeldungen des Versuches (siehe Wissenschaftlicher Bericht)

Mehrere Abschnittskonfigurationen -> Aufgrund Zwischenpräsentation

Tunnelgenerierung -> Mesh setzten wieso entscheidung für SketchUp-Modell

Xml Konfigurationsfile

Renderer Texturen setzen

Stegtransparenz

* PNG-Alpha-Channel
* Alpha-Channel

Rotation der Lichter

* Orientierung wo?

Drehende Textur

* Wie ist Parameter anzusehen?

UV Mapping

SketchUp Modell zu Unity

Transformation Elemente

Vertex-Transformation

XML-Mapping

Best Practices for VR

Ausblick

* Implementation Kurve
* Kurve -> Beliebig (nicht nur 90°)
* Muster der Lichtpunkte
* Physikalischer Steg (wissenschaftlicher bericht)