

# ArchitectsArcade – Game Dokumentation

## Steuerung

Aus der First-Person Sicht steuert man den Protagonisten mit den Tasten **ASDW** in die entsprechenden Richtungen. Dazu ist es möglich, mit der Maus in jene Richtung zu gehen, in die man auch sieht. Mittels **SPACE** ist ein Sprung möglich.

Des Weiteren ist ein weiterer Sprung möglich, in dem man mit den Tasten ASDW zuerst in eine gewünschte Richtung läuft und dann dazu die SPACE Taste drückt. Ein kurzer Sprung ist möglich, in dem man zuerst springt und dann die Tasten ASWD betätigt (es ist hilfreich, falls man vor einer Kante steht, die man ohne Anlauf überspringen kann). Man kann ebenfalls Walljumps durchführen und sich an Wänden hochziehen indem man gegen eine Wand springt und SPACE gedrückt hält.

Mit der Taste **R** ist es möglich das Level neu zu beginnen bzw. zu reloaden.

**ESC** beendet das Spiel

## Gameplay

Man muss mit dem Player Hindernisse überqueren oder darüber springen und man sollte möglichst nirgends hinunterfallen, sonst muss man wieder von Beginn an anfangen.

## Complex Objects

Im Level sind mehrere komplexe Objekte verteilt. Einige werden noch für den GameDay dazukommen.

## Effects

### Omni Directional Shadow Mapping

Beschreibung zu Shadow Mapping

ShadowMapping wird mit F10 Ein- und Ausgeschaltet. Jedes Pointlight im Spiel besitzt eine eigene Depthmap welche die Berechnung der Schatten ermöglicht.

Es wird ein FBO erstellt um die Depthmap für die Lichter zu erzeugen. Hierzu wird eine Cubemap an das FBO gebunden welche die Tiefeninformation für ein Licht enthält. Aus der Position des Lichtes wird Tiefeninformation in alle sechs Richtungen erzeugt. Hierzu verwenden wir einen Geometry Shader weil dann nur ein Shaderpass pro Licht notwendig ist. Außerdem ist es Ressourcenparameter da der Shader die Geometry für die einzelnen Seiten wiederverwendet.

Anschließend werden die Daten in den Hauptshader gebunden um herauszufinden ob eine Stelle schattiert ist. Wir verwenden die PCF Methode mit 4 Passes um die Kanten der Schatten zu glätten.

### Normal mapping

Das NormalMapping kann man mit F6 Ein-und Ausschalten, um den Effekt deutlicher zu sehen. Im Grunde wird an allen Objekten NormalMapping verwendet.

Dazu wird zu jeder Textur eine NormalMap geladen, aus der die Tiefeninformation geladen wird. Zur Berechnung des Lichtes und der NormalMap werden anschließend statt der Normale, die Tangenten verwendet um den Effekt zu erzielen. Dabei ist zu beachten, dass eine Tangent-Space Matrix

gebraucht wird, um diesen Effekt im Tangent-Space zu berechnen.

## Bloom

Den Bloomeffekt kann man mit F7 ein-und ausschalten.

Die Funktionsweise erfolgt durch drei FBOs:

Im ersten FBO erfolgt ein BrightPass auf das gesamte Bild.

Im zweiten FBO erfolgt ein horizontaler Gaußpass, im letzten FBO wird noch ein vertikaler Gaußpass eingefügt und anschließend werden die Wert zusammen mit dem originalen Bild zusammengerechnet, wodurch ein Bloom Effekt entsteht.

Der Effekt ist vor allem auf Lichtquellen zu sehen, die sehr hell sind, oder wenn eine Lichtquelle auf ein Objekt draufstrahlt, sieht man, dass im hellen Bereich alles etwas verschwommen ist.

## Illumination

Es wird ein Phong-Shading Modell verwendet. Im Spiel werden Pointlights verwendet, um das Omnidirectional ShadowMapping zu zeigen.

## LIBRARIES

Verwendete Libraries sind:

AssimpLoader (Zum Importieren von Szenen COLLADA)

Bullet Physics (Physik Engine)

FreeImage (Laden von Texturen)

GLEW

## Animated Objects

Derzeit befindet sich nur eine Y-Rotation des Players als Animation im Spiel, sowie Kisten, die man hinunterschieben kann. Bis zum GameDay ist noch eine hierarchische Animation geplant.

## View Frustum Culling

Das View Frustum Culling wird eingesetzt, um die Anzahl der Draw Aufrufe zu verringern. Das Frustum wird bei jedem Draw Aufruf aufgrund des Seh winkels und der Position neu berechnet und aktualisiert. Für jedes Objekt wird eine Sphere-Bounding Box für das View-Frustum-Culling berechnet.

## Transparency

Derzeit wird die Transparenz nicht unterstützt.

## Experimenting With OpenGL

VBO – für Positions, Normals, UVs, Tangents, Bittangents.

VAO – werden pro Objekt erstellt und kapseln alle VBOs eines Objektes.

FBO – FrameBufferObjects werden für Shadow-Mapping und Bloom verwendet.

Die MipMapping Quality und TextureQuality kann per F4 und F5 gewechselt werden.