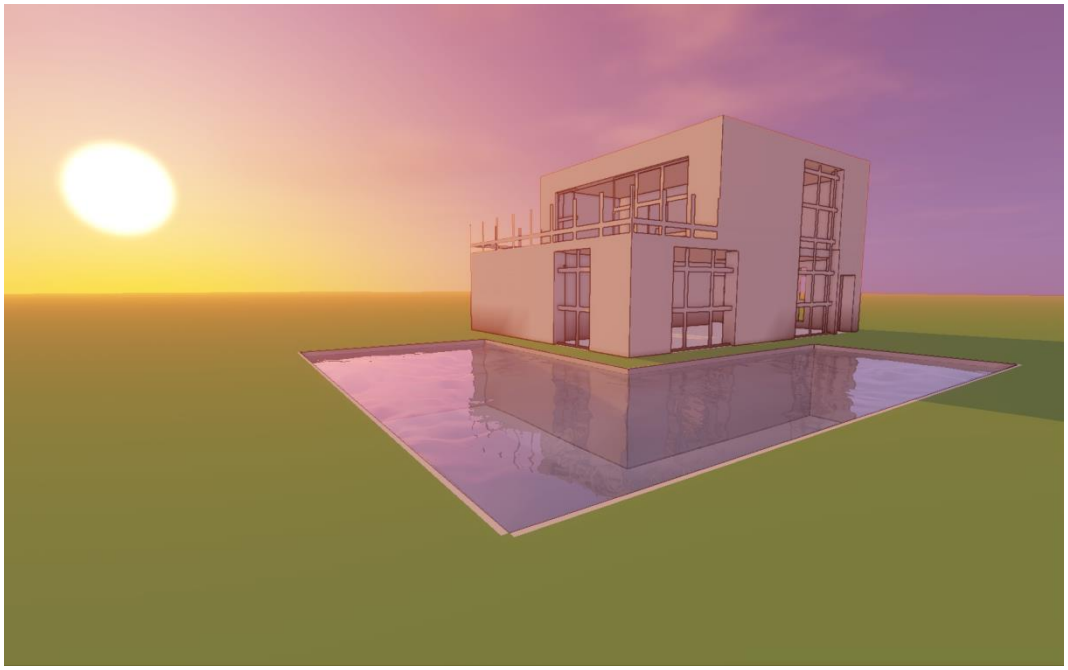
Karlsruher Institut für Technologie

HECTOR SEMINAR

Minecraft für Architekten



Abschlussbericht der Kooperationsphase 2016/2017

In Kooperation mit dem Karlsruher Institut für Technologie (KIT)
Institut Entwerfen und Bautechnik (IEB)
Fachgebiet Building Lifecycle Management (BLM)

Betreut von Dr.-Ing. Volker Koch

Paul Zanner

Christian Kleifges

Inhaltsverzeichnis

Abstract

Architects often work with a lot of different companies and people to build the houses they plan. These people might partially be divided by large distances which complicate the task of planning and building large and/or complicated buildings despite modern communication methods. This project's task is to show that Minecraft may be a way to solve this problem. Therefore the game has been altered to create a virtual space that can be used by architects to create a rough model of the building they plan, enabling them to visualize the ideas they have and also enabling real-time changes to the model by all people involved.

Minecraft is a game made by the Swedish Mojang AB and bought by Microsoft. It uses a grid of cubes with an edge length of one metre to create an open world that can be explored and changed by the player. One of the main aspects of the game is to gather resources with the aim of building a base that protects the player from dangers.

The product of this project is a set of instructions and (third-party) software that can be combined to create a server that is suitable for discussion of architectural problems.

Abb. 1: Screenshot of a "vanilla" minecraft world

1. Einleitung

Architekten müssen mit vielen verschiedenen Menschen, die unterschiedliche Interessen vertreten, arbeiten, um ein, bei ihnen in Auftrag gegebenes Gebäude von der Planung bis zum Bau zu betreuen. Dabei ist die räumliche Nähe aller Beteiligten nicht zwangsläufig gegeben. Gerade bei großen Projekten kann es zu Kooperationen über Landesgrenzen hinweg kommen. Diese räumliche Trennung kann die Planung und Umsetzung vor allem komplexer und/oder großer Gebäude deutlich erschweren.

Das Ziel dieses Projekts ist es, eine Möglichkeit aufzuzeigen, wie dieses Problem durch einen virtuellen Raum gelöst werden kann. Mithilfe des Computerspiels Minecraft soll eine digitale Welt geschaffen werden, in der Architekten einfache Modelle ihrer Gebäude bauen und anderen Beteiligten ihre Ideen zeigen können. Dabei wurde auf eine schnelle und einfache Veränderbarkeit der Gebäude seitens aller Beteiligten Wert gelegt.

Minecraft ist ein Computerspiel, bei dem der Spieler in einer simulierten Welt aus Würfeln mit einer Kantenlänge von einem Meter überleben soll. In der unveränderten Version des Spiels, der sogenannten Vanilla-Welt, bietet das Spiel der schwedischen Mojang AB, die inzwischen von Microsoft gekauft wurde, dem Spieler die Möglichkeit, eine offene Welt zu erkunden und Rohstoffe zu erlangen, um sich eine Basis zum Schutz vor Gefahren zu errichten. Das Konzept des Bauens, auch von großen Gebäuden, ist deshalb ein zentraler Aspekt des Spiels.

Das Ergebnis dieses Projekts ist eine Sammlung an Anleitungen und (Drittanbieter-) Software, die den Prozess des Modellbaus in Minecraft beschleunigen und ästhetisch variabler gestalten können.

Abb. 2: Screenshot aus einer Vanilla-Minecraft-Welt

2. Material und Methoden

In diesem Kapitel wird das in dem Projekt verwendete Material erläutert. Da dieses Projekt eine rein virtuelle Umsetzung hat, werden auch Programme zu diesen Materialien gezählt. Dabei wird hier jedoch bloß ein allgemeiner Überblick über die verschiedenen Arten von verwendeter Software gegeben. Eine genaue Auflistung mit Installationshinweisen folgt in Kapitel 3. Außerdem sollen kurz die Probleme dargestellt werden, die beim Bau eines Hauses im unveränderten Minecraft (man spricht auch von „Vanilla"-Minecraft) entstehen können, und deren Lösung in diesem Projekt gesucht wird.

2.1. Hardware

Für die Umsetzung der Idee wurden hauptsächlich zwei private Rechner mit Windows 10 verwendet.

Ein Rechner ist mit vier Intel® Core™ i5-4690 Prozessoren mit einer 3,50 GHz Taktfrequenz und integrierter Grafik (Intel® HD Graphics 4600), 8 GB RAM sowie einer HDD-Festplatte mit 1 TB Speicher ausgestattet. Dieser Rechner wurde als Server für den finalen Vorschlag verwendet.

Der andere hauptsächlich verwendete Rechner, weist folgende Spezifikationen auf: 4x AMD A8-6600K-Prozessor (APU) mit 3,9 GHz Taktfrequenz und integrierter Grafik (Radeon™ HD Graphics), 16 GB RAM (DDR3), AMD Radeon HD 7800-Grafikkarte, sowie eine 500-GB-SSD- (Spielspeicherort) und eine 1-TB-HDD-Festplatte.

Weiterhin funktionierte das Spiel mit Einschränkungen auf zwei Laptops mit Intel® Core™ i3 Prozessoren, integrierter Grafik und 4 GB Arbeitsspeicher.

Hinsichtlich der Hardware hat die Nutzung von Minecraft den Vorteil, dass die minimalen Systemvoraussetzungen sehr niedrig sind. So kann man Minecraft laut dem Entwickler sogar auf einem Intel® Pentium D Prozessor mit integrierter Grafik und 2 GB RAM spielen. Außerdem wird mit Java 6 Release 45 nur eine relativ alte Version von Java benötigt.[1] Bei der Realisierung des Projekts in Architekturbüros sind also keine Anschaffungskosten für neue PCs zu befürchten, da jeder moderne Computer in der Lage sein müsste, Minecraft auszuführen.

2.2. Ziele

Vor der Vorstellung der Modifikations-Typen sollen an dieser Stelle die Ziele benannt werden, die mit diesem Projekt erreicht werden sollten. Diese Ziele/Kriterien ergeben sich größtenteils aus den Problemen, die beim Bau eines Hauses in Vanilla-Minecraft auftreten können, aber auch aus der in der Einleitung dargestellten Fragestellung. Die Ziele lauten wie folgt:

Mehrspieler-Fähigkeit — Es sollen mehrere Spieler zur selben Zeit ein Gebäude bearbeiten können.

Veränderung zur Laufzeit — Die bereits vorhandene Möglichkeit, das Gebäude zur Laufzeit zu verändern, soll vorhanden bleiben.

Angepasste Blockgröße Das Größenverhältnis zwischen Spieler und Block soll einen größeren Detailgrad beim Bauen ermöglichen. Es muss also möglich sein, entweder mit kleinen Blöcken zu bauen oder den Spieler wachsen zu lassen.

Licht-Simulation Das sehr einfache und ästhetisch ungenügende Licht-Modell in Minecraft soll durch realistischere Simulationen der Beleuchtungssituation vor allem durch Fenster ersetzt werden.

Immersion[I] Während ein vollständiger Realismus als nicht notwendig erachtet wird, soll beim Erschaffen eines Gebäudes die Tatsache, dass es sich bei Minecraft um eine Computersimulation handelt in den Hintergrund treten.

2.3. Software

Dieses Unterkapitel ist eine allgemeine Beschreibung Minecrafts und der Möglichkeiten zur Erweiterung des Spiels. Zusätzlich wird auf die bedeutendsten Archive hingewiesen, in denen solche Erweiterungen gefunden werden können. Weiterhin werden einige Dateiformate erwähnt, die in dem Spiel verwendet werden. Dieses Kapitel ist die Grundlage für das Verständnis von Kapitel 3, in dem dann die in diesem Fall verwendeten Erweiterungen und ihre Funktion aufgelistet werden.

2.3.1. Minecraft

Für dieses Projekt ist Minecraft als „Kernprogramm“ von entscheidender Bedeutung.

Bei dem Spiel handelt es sich um ein sogenanntes „Open-World-Spiel“, was heißt, dass der Spieler frei entscheiden kann, wie er das Spiel innerhalb der Spielwelt spielt. Der Spieler kann die Welt, in der er sich befindet mehr oder weniger beliebig erkunden und verändern. Die Welt besteht aus Blöcken mit einer Kantenlänge von einem Meter und wird dynamisch generiert.

Es wurde im November 2009 erstmals veröffentlicht und wird seitdem kontinuierlich weiterentwickelt. Das Spiel wurde von Markus Persson (bekannt als „Notch“) erfunden und zunächst fast alleine entwickelt. Ende 2010 kam mit Jens Bergensten (bekannt als „jeb“) ein weiterer Entwickler hinzu, der Ende 2011, als Notch seine Arbeit an Minecraft einstellte, die Projektleitung übernahm. Ende 2014 wurde das Spiel und die entwickelnde Firma Mojang von Notch an Microsoft verkauft.[2,3]

Während das Programm Minecraft an sich beliebig oft kostenfrei heruntergeladen werden kann, wird zum Spielen von der Desktop-Version von Minecraft ein Account bei Mojang benötigt. Dieser kostet derzeit 23,95 €[4] und kann auf der Website des Spiels erworben werden. Hier standen jedoch bereits Accounts aus privater Nutzung zur Verfügung.

I Immersion bezeichnet in Videospielen das Gefühl, dass das vom Spiel simulierte geschehen Teil der echten Welt ist. Der Begriff wird oft auch im Zusammenhang mit VR(Virtual-Reality)-Brillen verwendet.

2.3.2. Ressourcen-Pakete

Zu den eher banalen Erweiterungen für Minecraft gehören sogenannte Ressourcen-Pakete. Diese werden von Minecraft ohne weiteres als Plug-Ins unterstützt. Es handelt sich bei diesen Paketen um Sammlungen von Bild- und Tondateien, die im Spiel statt den Standard-Ressourcen verwendet werden können und diese dann ersetzen. So lassen sich das Aussehen von Blöcken und der Klang von beispielsweise Trittgeräuschen sehr einfach und schnell verändern. Gerade zur Veränderung von Texturen, so wird das Aussehen der Blöcke bezeichnet, können viele verschiedene Ressourcen-Pakete kostenfrei aus dem Internet heruntergeladen werden. Dort sind diese oft auch unter der Bezeichnung Texturen-Paket[II] zu finden. Im Gegensatz zu den Modifikationen, bei denen es ein zentrales Archiv gibt, auf dem der Großteil aller Erweiterungen zu finden ist, sind Ressourcen-Pakete auf vielen verschiedenen Webseiten zu finden. Für die Suche nach diesen eignen sich deshalb herkömmliche Internet-Suchmaschinen, wie zum Beispiel Google, sehr gut.

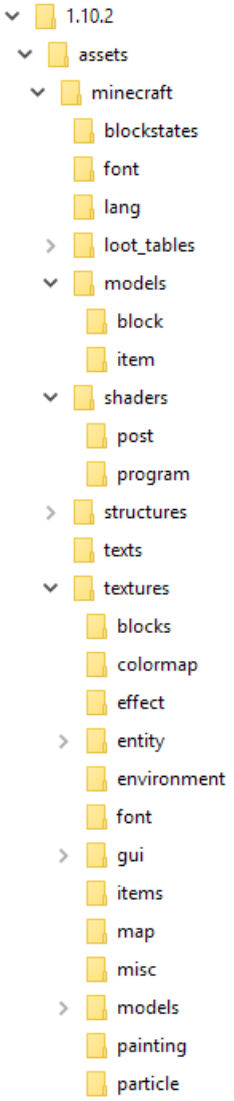


Abb. 3: Ordnerstruktur der 1.10.2.jar

Aufgrund der Eigenschaft von Ressourcen-Paketen, letzten Endes eine Art Archiv zu sein, lassen sich Ressourcen-Pakete relativ einfach auch selber erstellen, was ihre große Vielfalt im Internet erklärt. Es sind lediglich einige Vorschriften beim Anlegen der Verzeichnisstruktur und der Benennung und dem Format der Dateien zu beachten. Der Ordner, dessen Name später auch im Spiel zu sehen sein wird, enthält maximal drei Dateien: Die Datei pack.mcmeta, die Informationen zur unterstützten Minecraft-Version, unterstützten Sprachen und zur Beschreibung des Pakets im JSON-Format enthält, das Bild pack.png, das in Minecraft neben dem Paket angezeigt wird, sowie den assets-Ordner, in dem die neuen Spieldaten zu finden sind. Der assets-Ordner enthält normalerweise nur einen weiteren Ordner mit dem Namen „minecraft“. In diesem sind die eigentlichen Dateien enthalten. Um die Ordnerstruktur in diesem Ordner zu erfahren, öffnet man im .minecraft-Verzeichnis[III] den Ordner „versions“ und wählt den Ordner mit der gewünschte Version aus. In diesem sollte dann eine JAR-Datei mit dem Namen der Version (z.B. 1.10.2.jar) zu finden sein. Diese kann dann wie eine normales ZIP-Archiv entpackt werden. In der entpackten Version kann man ebenfalls einen assets-Ordner finden. Die Struktur von diesem soll in dem Ressourcen-Paket nachgeahmt werden (siehe Abb. 3). Auch das Format oder Vorlagen für die Standardtexturen können diesem Ordner entnommen werden.

Dieser Ordner muss schließlich nur noch zu einer ZIP-Datei komprimiert und im .minecraft-Verzeichnis unter „resourcepacks“ eingefügt werden.

II Bzw. der englischen Version „texture pack“

III Üblicherweise zu finden unter C:/Users/*Benutzername*/AppData/Roaming/.minecraft, wobei *Benutzername* durch den Namen des Benutzerkontos, auf dem Minecraft läuft, zu ersetzen ist. Wenn Windows auf die deutsche Sprache eingestellt ist, heißt der Ordner Users stattdessen Benutzer. Der Ordner AppData ist normalerweise verborgen, weshalb im Explorer die Option „Ausgeblendete Elemente anzeigen“ aktiviert sein muss. Alternativ kann in die Suchleiste von Windows „%appdata%“ eingegeben werden.

2.3.3. Shader-Modifikationen

Abb. 4: Vergleich eines Beispielgebäudes ohne (links) und mit (rechts) Shader

Eine weitere Gruppe von Erweiterungen, die von den Entwicklern vorgesehen sind, implementiert die Simulation von Schatten. Diese sogenannten Shader-Modifikationen, kurz Shader, sind, ebenso wie Ressource-Packs, zahlreich und kostenlos im Internet zu finden. Dabei variiert der Detailgrad der Schatten sehr, da Shader die Grafikkarte des PCs stark belasten können. Generell hat die Wahl des Shaders auch etwas mit persönlichen Präferenzen und ästhetischem Empfinden zu tun. Im Projekt konnte durch den Schattenwurf der Realismus der Simulation erhöht werden. Dies kommt auch der Raumwirkung zugute, das heißt, dass der Raum deutlich tiefer und „dreidimensionaler" wahrgenommen wird. In diesem Projekt war die explizite Suche nach einem Shader hinfällig, da der Schöpfer des im Projekt verwendeten Ressourcen-Pakets einen Shader empfiehlt. Doch auch wenn dies nicht der Fall wäre, reicht eine kurze Internetrecherche, um eine Vielzahl an verschiedenen Shadern zu finden.

2.3.4. Server

Um auf eine beliebige Anwendung von mehreren aus PCs zugreifen zu können, wird ein zentraler Rechner benötigt, der alle Anfragen verarbeitet. Dieser Rechner ist in den meisten Fällen nur für diese Aufgabe bestimmt. In diesem Fall wurde jedoch einer der beiden Computer, auf denen Minecraft lief (man spricht von den Clients), gleichzeitig als Server verwendet.

Die zentrale Frage beim Aufsetzen des Servers ist die, welche Software verwendet wird. Konkret muss man sich, um auf bestehende Modifikationen zurückgreifen zu können, zwischen den beiden größten Projekten, Bukkit und Forge, entscheiden. Beide Projekte beinhalten eine Programmierschnittstelle für Entwickler, denen so das Modifizieren von Minecraft erleichtert werden soll, und bieten eine Software für Server an, die den Betreibern das einfache Verwalten der auf ihrem Server laufenden Modifikationen ermöglichen soll. In diesem Projekt wurden zunächst beide Programme verwendet. Bei Bukkit waren jedoch schon vor Beginn des Projekts einige Urheberrechtsprobleme bekannt, weswegen die zukünftige Entwicklung von Bukkit etwas unklar scheint. Zusätzlich scheint das Angebot an Modifikationen für Forge deutlich größer zu sein. Deshalb wurde in der finalen Version des Servers Forge verwendet.

2.3.5. Modifikationen

Im Gegensatz zu Shadern und Ressourcen-Paketen bestehen „normale" Modifikationen in der Regel aus einer oder mehreren ausführbaren Dateien, meistens aus solchen des Formats JAR, welches für ausführbare Java-Programme steht. Sie verändern die Verhaltensweise des Spiels, beziehungsweise greifen an bestimmten Punkten ins Spiel ein,

wodurch völlig neue Möglichkeiten eröffnet werden, wie zum Beispiel das Hinzufügen neuer Blöcke.

Sie sind von Entwicklerseite her nicht wirklich vorgesehen, und werden vom Spiel auch nicht ohne weiteres unterstützt. Dazu ist für gewöhnlich ein sogenannter „Modloader" (im Großteil aller Fälle, wie auch in diesem Fall, Forge) vonnöten, der dann für gewöhnlich die Hauptspieldatei ersetzt und der die Modifikationen lädt, wenn das Spiel gestartet wird.

Die Breite an Modifikationen im Internet übertrifft vermutlich das Angebot an Shadern und Ressourcen-Paketen noch einmal deutlich. In diesem Projekt ermöglichen Modifikationen viele verschiedene Funktionen, die mit Vanilla-Minecraft nicht möglich sind. Dazu gehören beispielsweise das Einführen kleinerer Blöcke für detailreicheres Bauen, das Verwalten mehrerer „Dimensionen", um verschiedene Bauprojekte zu trennen und die Stapelbearbeitung von Blöcken für einen effizienteren Bauprozess. Für die Suche nach Modifikationen wurde in diesem Projekt hauptsächlich CurseForge[IV] verwendet, das – nach eigenen Angaben – größte Archiv für modifizierte Minecraft-Versionen. Doch auch hier bieten sich durch eine Internet-Recherche einige Alternativen, wobei die Modifikationen von diesen alternativen Ressourcen oft ebenfalls auf CurseForge zu finden sind.

2.3.6. Dateiformate

Im Verlauf des Projektes wurde mit einigen Dateiformaten gearbeitet, die von den Entwicklern Minecrafts erfunden wurden und speziell für Minecraft geschaffen wurden. Da die Kenntnis dieser Formate für den Zweck des Im- und Exports von Dateien von Bedeutung ist, werden diese hier vorgestellt.

NBT

Wenn man das .minecraft-Verzeichnis, in dem die meisten Daten des Spiels gespeichert sind, durchstöbert, so entdeckt man eine große Menge an verschiedenen Datenendungen, von denen die meisten allerdings schon mit einem normalen Texteditor in nützlicher Form betrachtet und bearbeitet werden können. Eine Ausnahme bilden dabei Dateien mit der Namenserweiterung DAT. Sie werden in einem von Notch entwickeltem Format, dem Named Binary Tag (kurz: NBT), gespeichert[5]. Wie der Name bereits impliziert, sind die Daten (teilweise) binär – es werden also nur einzelne Bits verwendet - weshalb ein normaler Texteditor nicht für die Bearbeitung dieser Daten genutzt werden kann. Mit einem Hex-Editor, der am besten auch einzelne Bits auslesen und anzeigen kann, können diese Dateien jedoch interpretiert werden.

Das NBT-Format ist prinzipiell eine Sammlung verschiedener Eigenschaften, den sogenannten Tags, denen Namen zugeordnet werden (deswegen heißt das Format NAMED Binary Tag). Jeder Tag hat dabei die gleiche Struktur. Das erste Byte[V] jedes Tags ist für eine ID reserviert, die angibt, um welchen Datentypen es sich handelt. Darauf folgen dann

[IV] minecraft.curseforge.com (Stand: 05.08.2017)
[V] Also die ersten acht Bit

zwei Bytes, die die Länge des Namens angeben[VI]. Darauf folgt dann der Name in UTF-8-Kodierung.[5]

Schematics Vanilla-Minecraft besitzt keine Möglichkeit, große Blockstrukturen zu speichern und zu bearbeiten, um sie beispielsweise zwischen Welten zu verschieben oder mit anderen Programmen zu bearbeiten. Deswegen entwickelte die Minecraft-Community das SCHEMATIC-Format. Es handelt sich dabei eigentlich nicht um ein Format, sondern um Dateien, deren Dateiendung .schematic ist. Diese Dateien verwenden allerdings das NBT-Format und können als besondere Variante des NBT-Formats betrachtet werden, in der die Dimensionen des gespeicherten Bereichs und die Block-ID[VII] jeder einzelnen Position angegeben sind.

[VI] Die Länge der Tag-Namen ist somit auf $2^{16} = 65.536$ Zeichen begrenzt

[VII] In Minecraft hat jeder Block eine ihm zugeordnete Nummer, die Block-ID

3. Ergebnisse

In diesem Kapitel wird das Produkt des vergangenen Projekts[VIII], eine Liste von Softwares, genauer erläutert. Weiterhin wird auf Erkenntnisse und Entscheidungen aufmerksam gemacht, die während des Projekts getroffen wurden. Die hier beschriebene Vorgehensweise erhebt keinen Anspruch auf alleinige Richtigkeit. Gerade Dank der sehr aktiven Minecraft-Community können viele Wege zum Ziel eines für Architekten nützlichen Servers führen.

Dieses Kapitel beantwortet weder die Frage nach dem Erfolg des Projekts noch die Ausgangsfrage nach der Nutzbarkeit Minecrafts für Architekten. Dies ist Aufgabe des nächsten Kapitels.

3.1. Ressourcen-Pakete/Shader

Im Verlauf des Projekts zeigte sich ziemlich schnell, dass ein Ressourcen-Paket benötigt werden würde, da es vor allem für Wände kein geeignetes Material gab. Der zu Beginn verwendete Stein störte aufgrund seiner grauen Farbe sehr und auch Alternativen wie Quarz-Blöcke störten aufgrund ihrer Textur.

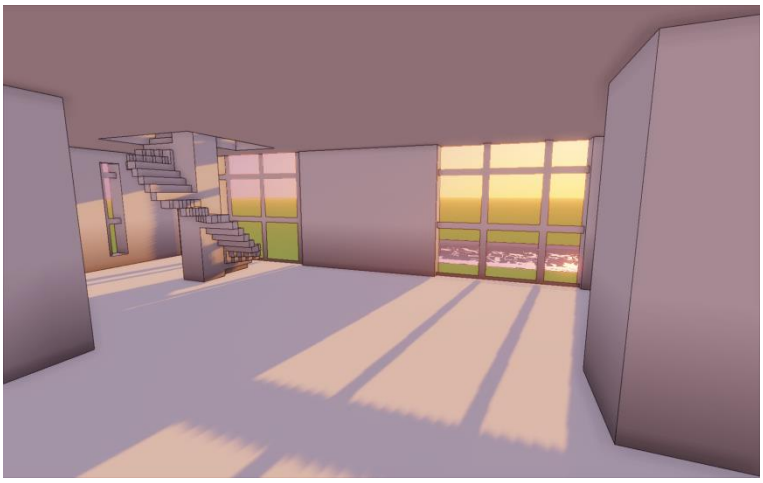


Abb. 5: Beispielgebäude, das mit Chisels & Bits und dem Ressourcen-Paket gebaut wurde

Die Suche nach einem geeigneten Paket gestaltete sich lange Zeit als eine Herausforderung, da zuerst nach einem realistischen und modernen Paket gesucht wurde. Da jedoch keines der untersuchten Pakete zufriedenstellend war, fiel in Absprache die Entscheidung, dass Realismus in diesem Projekt nicht unbedingt notwendig ist. Schließlich wurde dann das FORM Architecture Pack[IX] verwendet. Es wurde nach Angaben des Erschaffers von architektonischen Zeichnungen inspiriert. Die bedeutendste Eigenschaft dieses Paktes ist, dass der Großteil der Blöcke weiß ist, weshalb der Stil des Paktes sehr einfach und elegant wirkt.

Wie bereits in Kapitel 2 erwähnt, schlägt der Erschaffer des Ressourcen-Pakets die Verwendung von Sildur's Vibrant Shaders[X] und die Aktivierung der sogenannten Celshading Option vor. Durch diese werden die Kanten von Blöcken schwarz gefärbt, was die Unterscheidung von Flächen einfacher macht.

Das andere Paket, das von uns erstellt wurde, verstärkt die Trittgeräusche des Spielers, die somit lauter werden und besser wahrnehmbar sind. Dies ist insofern interessant, als

---

VIII Jede Erwähnung eines „Projekts" und dessen „Projektteilnehmer" beziehen sich auf dieses Hector-Kooperationsprojekt von Paul Zanner und Christian Kleifges unter Betreuung von Dr. Volker Koch

IX Ehemals White Architecture Ressource pack; www.planetminecraft.com/texture_pack/white-architecture-pack-111-beta-celshading-highly-recommended/ (Stand: 05.08.2017)

X minecraftforum.net/forums/mapping-and-modding/minecraft-mods/1291396-1-6-4-1-12-sildurs-shaders-pc-mac-intel-vibrant (Stand: 06.08.2017)

dass so durch verschiedene Bodenbeläge auch akustische Unterschiede erzeugt werden können.

3.2. Modifikationen

Wie schon zuvor beschrieben, beinhaltet das Konzept eine Reihe von Modifikationen, die für eine sinnvolle Benutzung Minecrafts für die Zwecke eines Architekten vonnöten sind. Im Nachfolgenden wird beschrieben, was diese Modifikationen im Vergleich zum Hauptspiel können oder verändern.

Chisels & Bits[XI] Chisels & Bits bietet die Möglichkeit, mit Blöcken im Maßstab 1:16 zu bauen.

FlatColoredBlocks[XII] Vom Entwickler von Chisels & Bits stammt auch FlatColoredBlocks, eine Modifikation, die Blöcke in jeder beliebigen Farbe bereitstellt.

WorldEdit[XIII] WorldEdit schafft die Möglichkeit, große Areale der Spielwelt mit ein paar wenigen Klicks und Befehlen zu verändern. Man kann beispielsweise bestehende Strukturen beliebig vervielfachen oder einfach große Bereiche mit einem bestimmten Block füllen. Weiterhin können mit WorldEdit Vorlagen im- und exportiert werden (hierfür dienen Dateien im bereits erwähnten SCHEMATIC-Format). Es wurde außerdem auf den Clients das WorldEdit CUI[XIV] (Client User Interface) installiert, das das Verwenden von WorldEdit erleichtert.

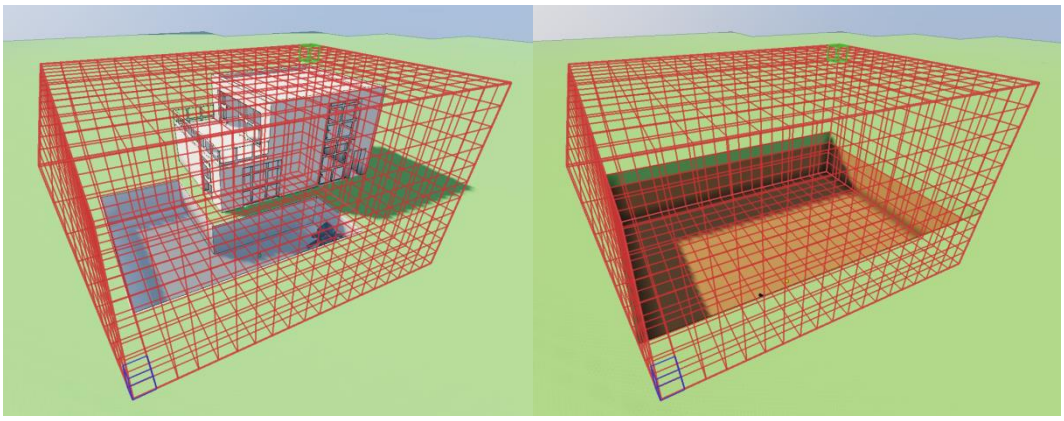


Abb. 6: WorldEdit mit der Benutzeroberfläche (CUI). Links wurde der Bereich eines Gebäudes ausgewählt und ausgeschnitten (rechts).

SpongeForge[XV] Sponge ist eine weitere API, die das Programmieren von Plug-Ins erlaubt. Diese haben gegenüber den normalen Modifikationen den Vorteil, dass sie nur auf dem Server laufen müssen, um zu funkti-

[XI] github.com/AlgorithmX2/Chisels-and-Bits (Stand: 04.08.2017)
[XII] github.com/AlgorithmX2/FlatColoredBlocks (Stand: 04.08.2017)
[XIII] github.com/sk89q/WorldEdit (Stand: 04.08.2017)
[XIV] mods.curse.com/mc-mods/minecraft/242932-worldeditcui-forge-edition
[XV] www.spongepowered.org (Stand: 04.08.2017)

onieren, was das Austauschen/Updaten deutlich einfacher gestaltet.

Optifine[XVI] Optifine optimiert die Grafikverarbeitung von Minecraft, um ein flüssigeres Spielerlebnis zu erreichen. Außerdem werden noch einige andere kleine Optimierungen im grafischen Bereich vorgenommen. Eine vollständige Liste aller Veränderungen kann der Website entnommen werden. An dieser Stelle wird es bei dieser kurzen Zusammenfassung belassen, da nur ein kleiner Teil aller verfügbaren Features genutzt wurde.

Waila[XVII]**/Hwyla**[XVIII] Waila (kurz für: What Am I Looking At) beziehungsweise eine Fork[XIX] davon namens Hwyla (Here's What You're Looking At) fügen ein zusätzliches Element in die Benutzeroberfläche des Spiels ein, das einige Daten über den aktuell betrachteten Block zeigt.

3.3. SpongeForge

Die grundsätzliche Funktion von SpongeForge wurde bereits im vorherigen Abschnitt erwähnt. Hier sollen nur noch einige Plug-Ins aufgelistet werden, die Eingang in dieses Projekt fanden. Hier fand die Recherche ausschließlich auf der Website des Entwicklers statt, da dieser mit dem sogenannten Ore Repository[XX] bzw. mit einer Forumskategorie, die bis zur Fertigstellung von Ore ersatzweise verwendet wurde[XXI], ein Archiv aller mit Sponge erschaffenen Plug-Ins bereitstellt.

Project Worlds[XXII] Project Worlds ermöglicht mehrere „Dimensionen" auf einem Server, wobei der Begriff Dimension hier nahezu gleichzusetzen ist mit Spielwelt. Dadurch können auf einem Server gleichzeitig mehrere verschiedene Projekte bearbeitet werden, ohne dafür Welten auf den Server hoch- oder von ihm herunterladen zu müssen.

Graveyards[XXIII] Graveyards erlaubt es, Orte in der Minecraft-Welt als „graveyards", also „Friedhöfe", zu definieren. Sollte ein Spieler aus irgendwelchen Gründen sterben, so erscheint er wieder beim nächsten Friedhof. So können im Falle eines Todes lange Wege zurück zum Bauprojekt vermieden werden.
Tode können in Minecraft versehentlich bei der falschen Eingabe eines Befehls auftreten, oder sie können absichtlich herbeigeführt werden um beispielsweise lange Wege zu vermeiden oder das eigene Inventar zurückzusetzen.

XVI optifine.net (Stand: 05.08.2017)
XVII minecraft.curseforge.com/projects/waila (Stand: 05.08.2017)
XVIII github.com/TehNut/HWYLA (Stand: 05.08.2017)
XIX So bezeichnen es Entwickler, wenn sie Programmcode von einem anderen Entwickler übernehmen und verändern
XX ore.spongepowered.org (Stand: 06.80.2017)
XXI forums.spongepowered.org/c/plugins/plugin-releases (Stand: 06.08.2017)
XXII forums.spongepowered.org/t/project-worlds-world-manager/10170 (Stand: 06.08.2017)
XXIII ore.spongepowered.org/Zerthick/Graveyards (Stand: 06.08.2017)

3.4. Weitere nützliche Programme

Hier sollen einige weitere Programme vorgestellt werden, die nicht direkt mit dem Server zu tun haben, die aber in verschiedenen Belangen sehr nützlich sein können.

NBTExplorer[XXIV]

Der NBTExplorer ist ein kostenfreies Programm, mit dem Dateien im NBT-Format (Siehe 2.3.6 Dateiformate), leichter bearbeitet werden können. Die Benutzeroberfläche listet alle Daten, die die angezeigte Datei enthält, auf und strukturiert diese entsprechend der Datei. Das bedeutet, dass Werte, die einer bestimmten Gruppe zugeordnet werden, auch als Gruppe angezeigt werden. So kann man in der nebenstehenden Abbildung sehen, dass die Einträge zur Position und Rotation eines Spielers in einer solchen Gruppe eingetragen sind.

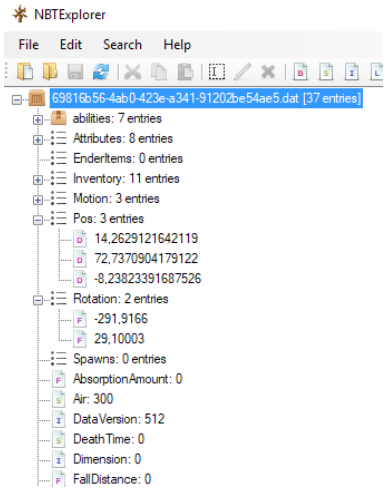


Abb. 7: Screenshot einer Datei im NBTExplorer

MagicaVoxel[XXV]

Der MagicaVoxelViewer ist eine Anwendung, mit der .schematic-Dateien (siehe ebenfalls 2.3.6) betrachtet werden können. Da solche Dateien nur eine Variante des NBT-Formats sind, ist das auch mit dem NBTExplorer möglich. Dieser zeigt jedoch nur die Daten der NBT-Datei an, weshalb es schwierig ist, sich das Gebäude anhand dieser Daten bildlich vorzustellen. Mit dem VoxelViewer wird dieses Problem gelöst, denn dieser stellt die .schematic-

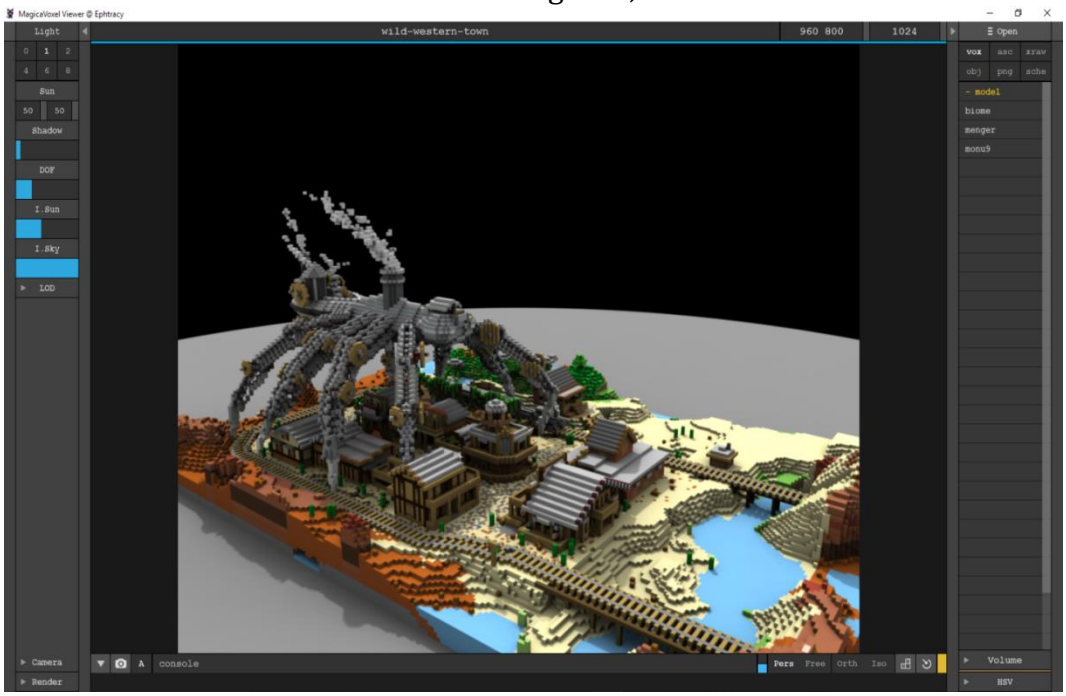


Abb. 8: Screenshot aus dem MagicaVoxelViewer mit einer Beispieldatei im SCHEMATIC-Format

[XXIV] Das zugehörige GitHub-Repository: github.com/jaquadro/NBTExplorer (Stand: 30.07.2017)
[XXV] Auch hier die zugehörige Website: ephtracy.github.io (Stand: 30.07.2017)

Vorlage dreidimensional dar und erlaubt sogar eine Schattensimulation mit veränderbarer Lichtquelle.
MagicaVoxel, das von derselben Person entwickelt wurde, unterstützt das SCHEMATIC-/NBT-Format nicht. Diese Anwendung wurde stattdessen dazu benutzt, um Voxelgrafiken[xxvi] zu erstellen. Diese können dann als Datei im OBJ-Format exportiert werden, welche wiederum mit Tinkercad zu .schematic-Dateien konvertiert werden. Der Vorteil von MagicaVoxel gegenüber Tinkercad ist, dass MagicaVoxel Bilder im PNG-Format importieren kann. So könnten beispielsweise grobe Grundrisse von komplizierteren Gebäuden in .schematic-Dateien konvertiert und in Minecraft importiert werden.

Tinkercad[xxvii]

Tinkercad ist eine Online-CAD-Software, mit der man Modelle in das SCHEMATIC-Format exportieren kann. Bereits vorhandene Modelle können im STL- oder OBJ-Format importiert werden, während Bilder im SVG-Format vorliegen müssen. Exporte sind in den gleichen Formaten und eben dem SCHEMATIC-Format möglich. In diesem Projekt wurde Tinkercad beinahe ausschließlich zum Konvertieren zwischen den Dateiformaten benutzt. Es fanden auch einige Versuche statt, Tinkercad zum Erstellen von Gebäuden zu benutzen und diese nur noch in Minecraft zu importieren. Diese Anstrengungen wurden jedoch wegen der im nächsten Kapitel dargestellten Probleme eingestellt.

[xxvi] Voxel sind das dreidimensionale Äquivalent von Pixeln. Eine Voxelgrafik ist somit ein dreidimensionales Konstrukt aus (großformatigen und) farbigen Würfeln.

[xxvii] www.tinkercad.com (Stand: 30.07.2017)

4. Diskussion

4.1. Alternative Modifikationen

In der bisherigen Dokumentation wurden nicht alle Möglichkeiten erwähnt, die während des Projekts in Betracht gezogen wurden. So gab es beispielsweise einige Alternativen zu Chisels & Bits, einer der zentralen Modifikationen in unserem Konzept.

Little Tiles[xxviii] war dabei eine Alternative, die sehr lange geprüft wurde. Diese Modifikation zeichnet sich vor allem dadurch aus, dass die minimale Größe der Blöcke manuell auf weit über 1000-fache Verkleinerung eingestellt werden kann. Das hätte das maßstäbliche Bauen in Minecraft maßgeblich vereinfacht und hätte den Detailgrad des Spiels deutlich heben können. Allerdings wurde ein Bug festgestellt, bei dem einige Blöcke spontan durchsichtig wurden, wie in Abb. 9 zu sehen. Während die Öffnung am linken unteren Bildrand als Teil eines Türrahmens vorgesehen ist, kann der Fehler rechts oberhalb dieses Türrahmens begutachtet werden. Weiterhin ist das vermeintliche Fenster im Hintergrund auch ein Beispiel für das Problem. Dieser Bug wurde für zu schwerwiegend befunden, um durch die kleineren Blöcke ausgeglichen werden zu können.

Abb. 9: Bug in der Little Tiles-Modifikation

Weiterhin fand auch eine Recherche zu Modifikationen statt, die den Spieler deutlich vergrößern, statt die Blöcke zu verkleinern. Obwohl eine solche Veränderung auf den ersten Blick keinen großen Unterschied macht, hätte es für diese Umsetzung große Vorteile gehabt. Das liegt daran, dass die SCHEMATIC-Dateien in Minecraft nur Standardblöcke speichern können. Es besteht also keine Möglichkeit, die Kreationen mit Chisels & Bits einfach zu transferieren. Es wurde auch keine Option gefunden, Konstruktionen aus normalen Blöcken mit Chisels & Bits oder einer vergleichbaren Mod zu schrumpfen. Somit kann die Im- und Exportfunktion von Minecraft nur für grobe Konstruktionen genutzt werden, deren Detailgrad noch manuell erhöht werden muss. Dieses Problem wäre durch einen wachsenden Spieler behoben worden, da dann normale Blöcke zum Bauen verwendet werden könnten und der Im- und Export mit WorldEdit erfolgen könnte.

Aufgrund des Mangels an solchen Modifikationen wurde also Chisels & Bits zur im Projektzeitraum besten Variante befunden.

4.2. Vorteile gegenüber konventionellen Methoden

Nach der Fertigstellung dieses Projekts ist die wahrscheinlich relevanteste Frage, welche Vorteile Minecraft gegenüber herkömmlichen CAD-Programmen bietet.

[xxviii] minecraft.curseforge.com/projects/littletiles (Stand: 06.08.2017)

Zur Beantwortung dieser Frage hilft es, sich das in der Einleitung umrissene Problem beim Planen von Gebäuden noch einmal ins Gedächtnis zu rufen. Durch die tief im Konzept von Minecraft verankerte Möglichkeit der Veränderung seiner Umwelt zur Laufzeit in Kombination mit der Möglichkeit, ohne große Umstände einen Minecraft-Server aufzusetzen, entsteht eine Plattform, in der Ideen mit minimalem Aufwand visualisiert und präsentiert werden können. So können Architekten zu Beginn der Planungsphase viele Ideen in kürzester Zeit umsetzen und vergleichen. Aber auch im Kundenkontakt kann diese Lösung sinnvoll sein, da der Kunde so eine Idee von der Raumwirkung im Gebäude erhält, ohne gleich zu erwarten, dass das fertige Gebäude exakt so aussieht, was bei CAD-Programmen passieren kann. Der Kunde kann das Gebäude auf eigene Faust „erkunden" und auch sofort Änderungsvorschläge einbringen.

Außerdem kann Minecraft sehr einfach erweitert werden. Es gibt eine sehr aktive Community im Internet und ständig neue Entwickler und Modifikationen, sowie Unterstützung und Foren, wenn man mit dem Entwickeln beginnen will. Daher ist Minecraft prädestiniert, in solchen Projekten wie diesem benutzt zu werden.

Allerdings kann und soll Minecraft konventionelle Programme nicht vollständig ersetzen. Denn diese bieten später in der Planungsphase die nötige Exaktheit, um beim Bau von Gebäuden als Grundriss oder Bauplan verwendet zu werden. Das ist mit Minecraft nicht möglich und war auch nie von den Projektteilnehmern beabsichtigt. Das Spiel wird stattdessen als Ergänzung zu konventionellen Lösungen betrachtet.

4.3. Überlegungen zum Immersionsgrad

Wie in Abschnitt 4.2 angedeutet, sind wir der Meinung, dass Minecraft sich genau in die Schnittstelle zwischen genug Realismus, um einen Eindruck vom Gebäude zu erhalten, und genug Simulation, um nicht mit dem echten Gebäude „verwechselt" zu werden, einfügt. Dabei stellt sich jedoch die Frage, wie viel Realismus wirklich notwendig ist und wie viel Realismus zu viel ist. Direkt damit verbunden ist auch die Immersion in das Spiel. Wie „echt" darf sich das Spielerlebnis anfühlen? Welche Möglichkeiten gibt es, die Immersion zu erhöhen?

Auch im Zusammenhang mit dieser Frage wurden im Projekt einige Recherchen getätigt. So wurde zum Beispiel die Veränderung des Spielsounds durch eine Modifikation in Betracht gezogen. Weitere Möglichkeiten, um Minecraft weiter zu verändern, werden noch später in diesem Kapitel diskutiert. Zur Frage nach dem Immersionsgrad sind wir der Meinung, dass alle Spielelemente in diesem Konzept beliebig realistisch sein könnten, solange zwei Bedingungen erfüllt sind:

a) Es wird das White Architecture Ressourcen-Paket oder ein anderes Paket in einem ähnlichen Stil verwendet.
b) Es ist noch zu erkennen, dass die Welt aus Blöcken besteht.

Wir glauben, dass vor allem diese beiden Elemente dafür verantwortlich sind, dass die Gebäude in Minecraft nicht zu realistisch aussehen.

4.4. Erkenntnisse

Die wohl wichtigste Erkenntnis wird wahrscheinlich schon aus dem bisherigen Verlauf dieser Dokumentation ersichtlich: Minecraft ist für Architekten nutzbar.

Die nächste geht mit dieser einher: so ist Minecraft nur mit einigen Modifikationen sinnvoll in der Lage, einen Architekten zu unterstützen, und, damit man mit mehreren Personen ein geplantes Gebäude besichtigen kann, wird der internetgestützte Mehrspielermodus benötigt, der zusätzlich noch einen Spielserver erfordert, auf dem die Server-Version von Minecraft installiert ist, mitsamt der verwendeten Modifikationen.

Es ist ohne weiteres möglich, einen solchen Server auf seinem Heimrechner aufzusetzen, um ihn jedoch von außen erreichbar zu machen, ist etwas mehr Aufwand erforderlich, da ein Zugang zum Server durch die Firewall des Netzwerks gelegt werden muss. Eine Alternative dazu wäre das Fehlen einer Firewall, was generell nicht zu empfehlen ist.

Minecraft bietet zwar spielseitig einen Chat an. Für eine gute Kommunikation wurde jedoch eine akustische Unterhaltung für sinnvoller befunden. Hiermit kann Minecraft jedoch nicht dienen.

Eine weitere Erkenntnis befasst sich mit dem Beleuchtungsmodell von Minecraft: Es ist nicht wirklich mit der Realität vereinbar. Aus diesem Grund wurden schon die Shader mit in die Lösung aufgenommen, können das Problem jedoch nicht wirklich lösen.

4.5. Fazit

Zunächst lässt sich eindeutig festhalten, dass die Leitfrage und Aufgabenstellung des Projektes nach der Nutzbarkeit des Computerspiels Minecraft für Architekten als Unterstützung für ihre Planungsprozesse, sowie zum gemeinsamen Begehen von geplanten Gebäuden zu einem Grad bewältigt werden konnte, der aus Sicht der Projektteilnehmer als mindestens zufriedenstellend bezeichnet werden kann.

Es konnte gemäß dem aufgestellten Konzept eine Konfiguration und Modifikation von Minecraft erreicht werden, die die vorgegebenen Bedingungen erfüllt.

Lediglich die beiden schon angesprochenen Punkte des Beleuchtungsproblems und eines sogenannten Voice-Chats, also der Möglichkeit der akustischen Kommunikation über den Minecraft-Server konnte nicht realisiert werden, das letztere wurde auch dank ausreichender Möglichkeiten zur Umgehung, zum Beispiel mittels entsprechender Software oder Telefonkonferenzen nicht weiter verfolgt.

Das Beleuchtungsmodell könnte mit einer entsprechenden Modifikation sicherlich angepasst werden, eine solche war jedoch nicht zu finden, und eine eigene zu Programmieren lag außerhalb der Fähigkeiten der Projektteilnehmer und des zeitlichen Rahmens des Projekts.

4.6. Weitere Untersuchungen

Im Weiteren stände zunächst die Untersuchung an, ob man mit einer eigenen Modifikation das Beleuchtungsmodell von Minecraft so verändern kann, dass es realistische Züge annimmt.

Weiterhin kann noch untersucht werden, ob und wie man den Immersionsgrad durch Virtual-Reality-Brillen erhöhen kann. An dieser Stelle ist den Projektteilnehmern lediglich bekannt, dass Minecraft für die Oculus Rift, eine der verbreitetsten VR-Brillen, verfügbar ist. Die Untersuchung des Effekts von VR-Brillen auf die Wirkung des Spiels und den Immersionsgrad ist sicherlich eine weitere interessante Frage.

Des Weiteren können auch noch weitere Möglichkeiten zur Erhöhung des Immersionsgrades, beispielsweise die Anpassung des Spielsounds um realistischeren Hall zu erzeugen, die Aufnahme von Trittgeräuschen auf verschiedenen Untergründen mit dem Ziel eines Ressourcen-Pakets oder die Umsetzung eines noch realistischeren Lichtmodells, untersucht werden.

Eine große Verbesserung wäre es, eine Möglichkeit zu finden, die Modelle mit kleineren Blöcken zu im- und exportieren (vergleiche dazu Abschnitt 4.1).

Die Optionen, Minecraft zu nutzen, sind hier keineswegs ausgeschöpft. Wie bereits erwähnt, ist Minecraft ein sehr variables Spiel, was bedeutet, dass die Möglichkeiten in den meisten Fällen nur durch die eigene Kreativität und Erfahrung im Programmieren begrenzt sind.

5. Danksagung

An dieser Stelle möchten wir, Christian Kleifges und Paul Zanner, uns bei den zahlreichen Personen bedanken, die das Gelingen dieses Projekts ermöglicht haben.

Besonderer Dank gilt dem Fachgebiet Building Lifecycle Management am KIT und unserem Betreuer Herrn Doktor Volker Koch, der uns immer wieder neue Möglichkeiten und Perspektiven eröffnet hat.

Bedanken möchten wir uns außerdem bei allen Kursleitern, die uns in den 6 Jahren beim Hector-Seminar begleitet haben. Besonders herausheben wollen wir dabei unseren langjährigen Kursleiter Herrn Thomas Knecht, der uns seit der ersten Sitzung des Kurses KA 11 betreut, diese Abschlussarbeit mehrfach gelesen und neue Anstöße zu deren Verbesserung gegeben hat. Auch Frau Anke Richert, die uns das Präsentieren und das Schreiben einer wissenschaftlichen Arbeit erläuterte, soll hier Erwähnung finden. Sehr dankbar sind wir außerdem Herrn Paul Bischof, der uns bezüglich der Hardware-Ausstattung beim OsKarl-Wettbewerb unterstützt hat.

Zum Schluss möchten wir uns auch bei Herrn Doktor Hans-Werner Hector und Frau Josephine Hector und der Hector-Stiftung für unsere langjährige und kontinuierliche Förderung bedanken.

6. Quellenverzeichnis

1 Minecraft system requirements. help.mojang.com/customer/en/portal/articles/325948-minecraft-system-requirements, Zugriff: 30.07.2017.

2 Minecraft. minecraft-de.gamepedia.com/Minecraft, Zugriff: 06.08.2017.

3 Jens Bergensten. minecraft-de.gamepedia.com/Jens_Bergensten, Zugriff: 06.08.2017.

4 Minecraft kaufen. www.minecraft.net/store, Zugriff: 30.07.2017.

5 NBT format. minecraft.gamepedia.com/NBT-Format, Zugriff: 30.07.2017.

7. Abbildungsverzeichnis

Alle Abbildungen sind Screenshots, erstellt von Christian Kleifges.

8. Selbstständigkeitserklärung

Hiermit versichern wir, dass wir diese Arbeit unter der Beratung durch Dr.-Ing. Volker Koch selbstständig verfasst haben und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, sowie Zitate kenntlich gemacht haben.

Ort, Datum	Paul Zanner
Ort, Datum	Christian Kleifges