Entwicklung eines Interaktionskonzeptes zur Modellieren in Videospielen

Andreas Jagel

MASTERARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

Interactive Media

in Hagenberg

im September 2013

© Copyright 2013 Andreas Jagel

Diese Arbeit wird unter den Bedingungen der *Creative Commons Lizenz Namensnennung–NichtKommerziell–KeineBearbeitung Österreich* (CC BY-NC-ND) veröffentlicht – siehe http://creativecommons.org/licenses/by-nc-nd/3.0/at/.

Erklärung

Ich erkläre eidesstattlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt und die den benutzten Quellen entnommenen Stellen als solche gekennzeichnet habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Hagenberg, am 30. September 2013

Andreas Jagel

Inhaltsverzeichnis

Erklärung iii

Abstract vi

Kurzfassung vii

1 Einleitung 1
- 1.1 Motivation . . . 1
- 1.2 Problemstellung . . . 2
- 1.3 Zielsetzung . . . 3
- 1.4 Gliederung . . . 3

2 Stand der Entwicklung 4
- 2.1 Bestehende Anwendungen . . . 4
  - 2.1.1 Verwandte Spiele . . . 4
  - 2.1.2 3D-Modellierung . . . 6
  - 2.1.3 Schlussfolgerung . . . 9
- 2.2 Bestehende 3D Interaktionstechniken . . . 9
  - 2.2.1 Erzeugen von Polygonen . . . 9
  - 2.2.2 Selektionstechniken . . . 11
  - 2.2.3 Manipulation . . . 14
  - 2.2.4 Löschen von Polygonen . . . 16
  - 2.2.5 Sonstiges . . . 16

3 Interaktionskonzepte 19
- 3.1 Anforderungen . . . 19
- 3.2 Implementierung . . . 20
- 3.3 Konzeptunabhängige Eigenschaften . . . 20
  - 3.3.1 Ausgangsszene . . . 20
  - 3.3.2 Navigation . . . 20
  - 3.3.3 Modellierungsmodus . . . 22
  - 3.3.4 Tiefenebene . . . 23
- 3.4 Konzept 1 . . . 26
  - 3.4.1 Weltmodus . . . 27

3.4.2 Objektmodus . . . 32
3.5 Konzept 2 . . . 37
3.5.1 Weltmodus . . . 37
3.5.2 Objektmodus . . . 39
3.6 Fazit . . . 41

4 Evaluation **43**
4.1 Vorstudie . . . 43
4.2 Teilnehmer . . . 44
4.3 Testsystem . . . 44
4.4 Benutzerstudie . . . 44
4.4.1 Ziel . . . 45
4.4.2 Ablauf . . . 45
4.4.3 Einführung . . . 46
4.4.4 Aufgaben . . . 46
4.4.5 Daten . . . 48
4.5 Ergebnisse . . . 50
4.5.1 Bewertungen . . . 51
4.5.2 Feedback . . . 52
4.5.3 Benutzungsdaten . . . 55
4.5.4 Zeiten . . . 56
4.5.5 Verhalten . . . 56
4.6 Experten-Vergleich mit *Minecraft* . . . 58

5 Resultate und Diskussion **62**
5.1 Analyse . . . 62
5.1.1 Bewertungen . . . 62
5.1.2 Feedback . . . 64
5.1.3 Benutzungsdaten . . . 66
5.1.4 Zeiten . . . 67
5.1.5 Verhalten . . . 68
5.2 Ergebnis . . . 69

6 Ausblick **70**
6.1 Verbesserungsvorschläge . . . 70
6.1.1 Konzept 3 . . . 70
6.1.2 Alte Konzepte . . . 73
6.2 Weitere Literatur . . . 73

A Fragebogen **75**

Quellenverzeichnis **78**
Literatur . . . 78
Online-Quellen . . . 79

Abstract

This work introduces a 3D modelling concept that significantly differs from previously applied approaches. The successful interactive game Minecraft served as both inspiration for the developed software. Some years ago a very successful video game was published, in which players can completely manipulate their virtual world—Minecraft. The virtual world, consisting of one meter cubes, can be altered by removing or attaching new blocks to existing ones. This interaction concept is a mix of modeling and gaming and didn't exist to this extent before. But it made the game one of the most successful games worldwide. Still, the simplified world and the intuitive interaction concept have some serious restrictions. The present work therefore describes a new interaction concept to overcome these restrictions and help the players to express their artistic ideas. Techniques from existing 3d modeling software are used and an evaluation of a developed prototype was performed to determine possible usability problems.

Kurzfassung

Vor einigen Jahren erschien das Videospiel Minecraft, in welchem der Spieler seine Umwelt selbst gestalten kann. Es hat ein sehr einfaches Interaktionskonzept und eine Welt bestehend aus kubikmetergroßen Blöcken. Diese Blöcke können nach Belieben manipuliert werden. Das Spielkonzept ist eine Mischung aus Modellierung und Spiel, welche in der Art noch nicht existierte und nun zu den erfolgreichsten Spielen weltweit gehört. Diese vereinfachte Welt und das ebenso einfache Interaktionskonzept haben aber starke Einschränkungen. In dieser Arbeit wurde ein neues Konzept mit dem Ziel, diese Einschränkungen zu minimieren und dem Spieler möglichst viel kreative Freiheit zu geben, entwickelt. Zur Umsetzung wurde sich an Techniken aus bestehenden 3D-Modellierungsanwendungen bedient. Der entwickelte Prototyp wurde einer Evaluierung unterzogen, um ihn auf Benutzerfreundlichkeit und Handhabung zu überprüfen. Basierend darauf werden Möglichkeiten zur Verbesserung in zukünftigen Iterationen präsentiert.

Kapitel 1

Einleitung

1.1 Motivation

Die Branche der Videospiele, zu der Computer-, Konsolen- und Browserspiele zählen, ist nun seit einigen Jahren weltweit wirtschaftlich signifikanter denn je und aus dem Alltag nicht mehr weg zu denken. Laut [20] macht sie sogar der Musikbranche Konkurrenz:

> *Im vergangenen Jahr hat die Videospielindustrie weltweit 56 Mrd. Dollar umgesetzt. Das ist mehr als doppelt so viel wie die Musikindustrie erwirtschaftet hat.*

Eine solide Branche mit stetigem Wachstum, in der immer noch Innovationen hervortreten. Allerdings ist das die Ausnahme, denn die meisten Spiele übernehmen nur Konzepte anderer erfolgreicher Spiele. Besonders die großen Konzerne, die eigentlich die Geldmittel hätten, um in dieser Branche zu forschen, handeln laut [17] nach diesem Ansatz:

> *Es sind privatwirtschaftliche, börsennotierte Konzerne, die Umsatz- und Gewinnziele erreichen müssen. Für Experimentelles sei da kein Platz.*

Auch [2, S. 7] bekräftigt diese Tatsache:

> *Viele Spielentwickler können sich daher keinen „Flop“ leisten, sodass innovative Spielideen häufig nicht umgesetzt, sondern bewährte Konzepte übernommen und weiterentwickelt werden.*

Neue Ideen und innovative Spielkonzepte werden meist von freien Entwicklern[1] hervorgebracht. *Minecraft* ist bisher das weltweit erfolgreichste *Indie Game* für den Desktop-Computer und macht selbst den erfolgreichsten Titeln Konkurrenz, da es laut [19] mit 12 Millionen verkauften Exemplaren,

[1] *Indie Game*-Entwickler, ohne finanzielle Unterstützung und Abhängigkeit von anderen Unternehmen.

Abbildung 1.1: Ein Haus am Meer, das in Minecraft erstellt wurde.

auf Platz drei der meistverkauften Videospiele für den Desktop-Computer steht.

Das Interessante an dem Spiel ist, dass die Spieler die virtuelle Welt in der ihr Charakter existiert, manipulieren können. Die Welt besteht aus kubikmetergroßen Blöcken, die an einem globalen dreidimensionalem Raster ausgerichtet sind. Der Spieler kann diese Blöcke entfernen und neue Blöcke neben vorhandene Blöcke setzen. Dadurch können Objekte approximiert, dargestellt werden, wie in Abbildung 1.1 ersichtlich ist. Da die Blöcke relativ groß sind, müssen sehr viele Blöcke platziert werden, um Runde oder sehr komplexe Objekte mit ihnen darzustellen. Das Interaktionskonzept des Spiels ist sehr einfach und kann daher in unter einer Minute erlernt werden. Im Prinzip ist es eine Modellierungsanwendung mit zusätzlichen Spielinhalten.

1.2 Problemstellung

Die Vereinfachung der virtuellen Welt als Blöcke und dazu das einfache Interaktionskonzept, haben aber gravierende Einschränkungen. Die Blockgröße ist immer gleich, sodass keine kleinen Objekte vom Spieler modelliert werden können. Es gibt keine Definition, für durch Blöcke approximierte Objekte. Am Beispiel eines Hauses illustriert, kann dieses niemals als ganzes bewegt werden. Stattdessen, müsste jeder Block einzeln abtransportiert werden. Demzufolge können Objekte auch nicht rotiert werden. Selbst wenn dies möglich wäre, gäbe es die Einschränkung des dreidimensionalen Rasters, an das die Blöcke ausgerichtet sein müssen. Rotationen wären nur in 90 Grad Schritten möglich. Eine Skalierung ist ebenso eingeschränkt und könnte nur um den Faktor zwei erfolgen. Es wäre daher auch nur eine Vergrößererung, aber keine Verkleinerung möglich.

1.3 Zielsetzung

Auf Basis dieser Einschränkungen wird ein Interaktionskonzept definiert, mit dem der Spieler nicht davon abgehalten wird, freiere, dem künstlerischen nähere Modellierung zu tätigen. Dazu wird die Welt aus Polygonen bestehen, ohne ein einschränkendes Raster. Dies ist nun seit fast 20 Jahren das standardmäßige grafische Primitiv für jegliche Objekte in Videospielen. Wie auch in aktuellen 3D-Modellierungsanwendungen, wird der Spieler Objekte aus Polygonen zusammensetzen. Diese Anwendungen sind allerdings sehr komplex und benötigen eine lange Einarbeitungszeit, bis sie beherrscht werden. Da das Interaktionskonzept für ein Spiel ist, sollte es einfach zu benutzen sein. Daher werden bei der Entwicklung entsprechende Kompromisse getroffen.

Um entscheiden zu können, ob der erdachte Ansatz von möglichen Nutzern akzeptiert wird, wird eine Evaluation durchgeführt. Dazu werden unterschiedliche Interaktionskonzepte entwickelt, um sie gegeneinander zu evaluieren, und so das bessere Konzept, anhand von Bewertungen und Messungen, zu bestimmen.

1.4 Gliederung

Zuerst werden in Kapitel 2 verwandte Arbeiten, allen voran das Spiel *Minecraft*, sowie Techniken für die Interaktion mit Objekten und Polygonen beleuchtet. Kapitel 3 stellt die entwickelten Interaktionskonzepte vor, die sich teilweise auf die in Kapitel 2 präsentierten Techniken referenzieren lassen. Kapitel 4 folgt mit der Evaluation des entwickelten Prototypen und die signifikanten Ergebnisse werden im darauffolgenden Kapitel 5 diskutiert. Diese Arbeit schließt mit Kapitel 6, mit der Vorstellung von identifizierten Verbesserungsmöglichkeiten und potenziellen Erweiterungen, ab.

Kapitel 2

Stand der Entwicklung

2.1 Bestehende Anwendungen

Die Kombination von Modellierung und Spiel erfordert eine Analyse beider Aspekte. Es werden Spiele, die mit der 3D-Modellierung verwandt sind, und Techniken aus 3D-Modellierungsanwendungen, die in Spielen eingesetzt werden können, untersucht. Diese Anwendungen neigen zu massiv erhöhter Komplexität, was zu einer äquivalent langen Einarbeitungszeit und teilweise sehr steilen Lernkurven führt.

In Spielen gibt es häufig *Tutorials*, kurze begleitende Spielabschnitte, die dem Spieler das Spiel erklären, sodass das Gelernte sofort umgesetzt werden kann. Spielern wird also eine kürzere Einarbeitungszeit zugemutet, als Nutzern von 3D-Modellierungsanwendungen. Daher kann nicht einfach der Funktionsumfang der Modellierungsanwendungen in ein Spiel verbaut werden. Es müssen Kompromisse eingegangen werden um diese beiden Aspekte zu kombinieren. In diesem Kapitel wird der Begriff „Spiel“ mit „Videospiel“ assoziiert.

2.1.1 Verwandte Spiele

Spiele die etwas mit der Modellierung zu tun haben, kamen bisher nur ansatzweise vor. Schon im Jahr 1989 begann die Spielserie *Populous* von *Bullfrog*, in der die Landschaft mit verschiedenen Methoden angepasst werden konnte. Diese Serie endete aber 1997 mit *Populous: The Beginning*.

Der Spieleentwickler *Blizzard* schlug 1995 mit dem Strategiespiel *Warcraft II: Tides of Darkness* einen anderen Weg ein, indem neben dem Spiel, auch ein Editor-Programm beigefügt wurde, mit dem weitere Szenarios erstellt und gespielt werden konnten, zusätzlich zu den schon vorhandenen Szenarios. Dies machte das Spiel so erfolgreich, dass es auch 2002 noch gekauft und gespielt wurde, während der Nachfolger *Warcraft 3: Reign of Chaos* erschien [30].

Der Editor in *Warcraft 3: Reign of Chaos* ist noch umfangreicher und er-

laubt es vorhandene Ressourcen zu verbinden (3D-Modelle, Sounds, Musik) und diese in ein Level zu setzen. Logik wird mit einer C ähnlichen Sprache eingefügt, die alternativ auch mit GUI-Bausteinen zusammensetzt werden kann. Die Levels die damit erstellt werden, können als eigenständige kleine Spiele betrachtet werden. Diese Mini-Spiele waren so erfolgreich, dass sogar ganze Genres damit definiert wurden.

Das *Multiplayer Online Battle Arena* (MOBA) Genre, welches erst seit 2009 kommerziell entwickelt wird (siehe dazu [27, S. 2]) und vorher nur als Modifikation in diversen Spielen aufzufinden war, startete mit dem von Spielern erstellten Level *Aeon of Strife* [27, S. 1]. Dieses wurde mit dem Editor von *Blizzard's StarCraft* erstellt (1998). Weitere Levels folgten mit dem Editor aus *Warcraft 3: Reign of Chaos* und dessen Add-on. Mittlerweile sind schon 10 eigenständige Spiele aus diesem Spielkonzept veröffentlicht worden, darunter auch *League of Legends*, welches mit 32 Millionen aktiven Spielern weltweit, der aktuelle eSports-Hit ist [27, S. 1].

Das *Tower Defense* Genre kam als erstes in *Dune 2* vor und wurde dann auch im *Warcraft 3*-Editor nachgebaut und hat sich mit der Zeit weiterentwickelt. Es gibt aber auch kommerzielle Veröffentlichungen: *Defense Grid: The Awakening* (2008) und *Plants vs. Zombies* (2009).

Im Jahr 2008 wurde die Simulation *Spore* veröffentlicht. Diese ermöglichte es dem Spieler Kreaturen zu erstellen, indem vorgegebene Körperteile angepasst und aneinander gesetzt werden konnten. 2011 erschien das Action-Rollenspiel *Darkspore*, welches einen ähnlichen Kreaturen-Editor mitlieferte, allerdings keine Neuheiten hinsichtlich der Modellierung.

Das Spiel welches der Modellierung am nächsten kommt, ist *Minecraft*, welches 2009 erschienen ist. Dessen Spielwelt besteht aus kubikmetergroßen Würfeln, die entfernt und hinzugefügt werden können. Dadurch können Berge, Häuser oder auch ganze Städte errichtet werden. Die Steuerung ist sehr einfach und schnell erlernbar. Seit der Veröffentlichung wurden unzählige virtuelle Welten erschaffen, die an Bücher, Filme, Fernsehserien oder auch an anderen Computerspielen angelehnt sind. Ein Beispiel aus der Fernsehserie *Game of Thrones* ist in Abbildung 2.1 gezeigt. Nach [19] ist *Minecraft*, mit über 12 Millionen Spielbesitzern, das dritt meist verkaufte Desktop Videospiel weltweit.

Obwohl dieses simple Interaktionskonzept dazu führt, dass Nutzer eine signifikant kurze Einarbeitungsphase haben, birgt es folgende Nachteile:

1. Wichtige 3D-Modellierungsfunktionen fehlen: Das Verschieben, Drehen und Skalieren von Objekten. Diese Funktionen können auch nur beschränkt umgesetzt werden, da die Welt an einem globalen Raster für die Blöcke ausgelegt ist. Drehen wäre nur in 90 Grad Schritten möglich und Skalieren nur um ein vielfaches von zwei, und auch nur Vergrößern.
2. Die Blöcke könnten auch nur passend zum Raster verschoben werden.

Abbildung 2.1: Nachbildung der Hauptstadt *King's Landing* des Königreiches aus der Fernsehgserie *Game of Thrones* in *Minecraft*. 100 Spieler erstellten dies innerhalb von 4 Monaten. Quelle: [23].

Außerdem wäre der Detailgrad des 3D-Modells, abhängig von der festgelegten Größe der Blöcke, also des Rasters. Dadurch können Objekte die kleiner sein sollen als einige Blöcke, nicht repräsentativ genug modelliert werden.

Die Beispiele der entstandenen Genres zeigt, dass die Spieler gewillt sind sich selbst zu helfen und Spielinhalte zu erstellen. Wenn dies unterstützt wird, wird Innovation gefördert. Die nachgebildeten Welten aus Büchern und Serien, zeigen wie ehrgeizig, motiviert und ausdauernd die Spieler sein können und welche Masse an Leuten am Modellieren im Spiel interessiert ist.

2.1.2 3D-Modellierung

Die Modellierung von 3D-Welten basiert auf der Erstellung von Gitternetzen, die wiederum aus Polygonen bestehen. Diese Gitternetze sind feingranulare Repräsentationen von Objekten, den sogenannten 3D-Modellen. Die bekanntesten Anwendungen für diese Aufgabe sind *3ds Max*, *Maya*, *Blender* und *Cinema 4D* [16]. Alle vier Anwendungen ähneln sich sehr stark, vor allem in ihrer Komplexität und Vielfalt an Funktionalität, da ihr Umfang an Funktionen sehr groß ist: Erstellen, Texturieren, Rigging[1] und Animieren von 3D-Modellen. Jede einzelne dieser Aufgaben allein ist sehr komplex und benötigt stundenlange Einweisung durch eine zweite Person und noch

[1] Erstellung eines Skelettmodells, zum Anwenden von Animationen.

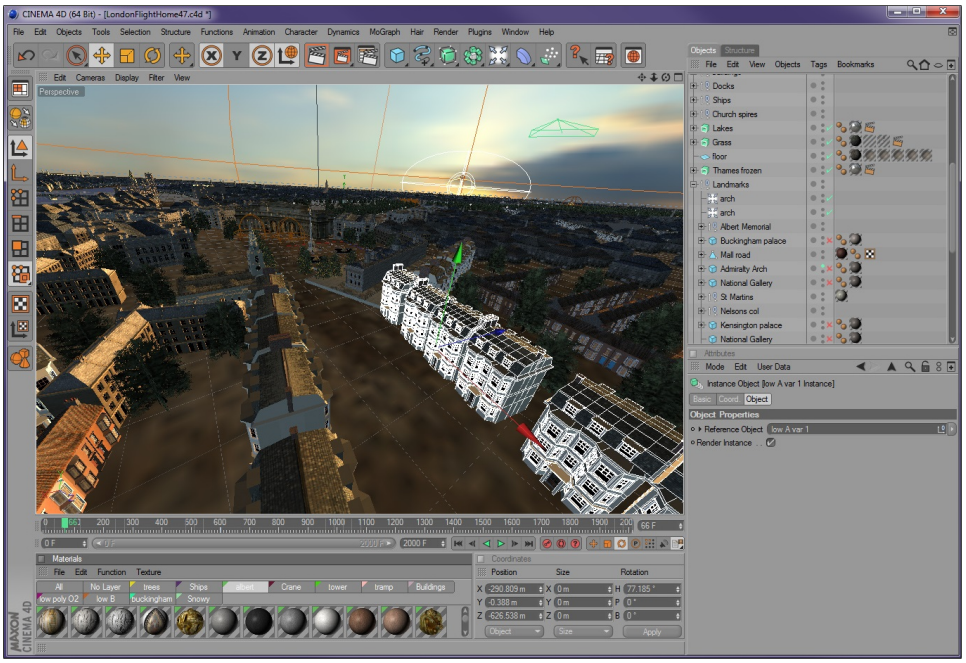


Abbildung 2.2: Fenster der *Cinema 4D* Modellierungsanwendung. Quelle: [15].

längere Übung. Die Komplexität und Umfang von Cinema4D kann in Abbildung 2.2 betrachtet werden. In allen vier Anwendungen wird generell mit vier Ansichten gearbeitet: Jeweils eine Ansicht für jede Achse in der dritten Dimension und eine freie Ansicht zum Anschauen der Szene. Die drei extra Ansichten der Achsen dienen dabei zur Reduktion der Komplexität von 3D auf 2D, was die Arbeit erleichtert und in [3, S. 29] bestätigt wird. Spieler könnten mit vier Ansichten überfordert sein, da generell in Spielen nur eine Kameraansicht gleichzeitig aktiv ist.

Als Beispiel wird nun der Umfang an Modellierungsfunktionen von *Maya* angegeben. Die Funktionen zur Polygonmodellierung werden ausführlicher angegeben. Die nachfolgende Liste wurde auf Basis der Onlinedokumentation von *Maya 2013* [21] erstellt:

- Modellierung von Polygonnetzen:
  - Selektieren von Polygonkomponenten (Flächen, Kanten und Vertices),
  - Erzeugen vordefinierter Objekten (Flächen, Würfel, Kugeln, Zylinder, Kegel, etc.),
  - Kombinieren, Trennen und Zerschneiden von Polygonen,
  - verschiedene Anpassungen von Polygonnetzen,
  - Verschieben, Drehen und Skalieren von Polygonkomponenten,

  - Glätten von Polygonnetzen,
  - Vertexnormalen anpassen,
  - Löschen von Polygonkomponenten,
  - verschiedene Korrekturfunktionen,
  - Verformungsfunktionen[2].
- Modellierung von NURBS[3],
- Texturkoordinaten setzen,
- Animation,
- Rigging,
- Physikeffekte (Haare, Flüssigkeiten, etc.).

Die grundlegendsten Funktionen auf Polygonen sind:

1. Das Erzeugen,
2. das Selektieren,
3. die Manipulation,
4. und das Löschen.

Alle weiteren Funktionen in der Polygonmodellierung, bis auf das Anpassen der Vertexnormalen, erweitern nur diese vier Basisfunktionen.

Die verbleibenden Punkte werden nun kurz diskutiert. Das Erzeugen und Manipulieren von Vertexnormalen ist wichtig für eine schönere Lichtberechnung, was für diese Arbeit nicht wichtig ist und daher vernachlässigt wird. NURBS sind mathematisch definierte 3D-Modelle und werden z.B. in Animationsfilmen benutzt, da damit die unschönen Kanten der Polygone vermieden werden können, was realistischere Modelle hervorbringt. Sie sind komplizierter zu berechnen und darzustellen als Polygonnetze und würden den Rahmen dieser Arbeit sprengen. Daher werden die Polygonnetze als Basis der 3D-Repräsentation gewählt. Texturen sollten also ein Teilaspekt der Modellierung sein, aber nicht zwangsläufig eine Aufgabe für den Spieler sein. Nach [13, S. 11] erleichtern texturierte Objekte aber die Tiefenwahrnehmung, was dem Spieler hilfreich sein würde. Texturen sollten schon beim Modellieren vorhanden, aber nicht vom Spieler zu modellieren sein. Ein Ansatz zur automatisierten Texturierung sollte benutzt werden. Die Animation sollte für den Anfang ignoriert werden, da sie auf schon vorhandenen Modellen aufbaut und vorher erst der Modellierungsprozess ordentlich definiert und evaluiert werden sollte. Damit fällt auch das Rigging weg, da es die Vorstufe für die Animation ist. Physikeffekte sollen die Modelle realistischer wirken lassen, was aber nicht das Ziel ist und somit die Modellierung nur unnötig verkomplizieren würde.

[2] Sculpt Geometry Tool
[3] Non-Uniform Rational B-Splines

Neben den vier Basisfunktionen und der Texturierung, sind an dieser Stelle alle weiteren Funktionen vernachlässigbar, da sie nicht das vorrangige Ziel dieser Arbeit hervorheben.

2.1.3 Schlussfolgerung

Die in Abschnitt 2.1.1 genannten Spiele bieten nur eingeschränkte Möglichkeiten zur Modellierung. Es können keine vorhandenen Ansätze übernommen werden. Es wird daher auf vorhandene Techniken aus existierenden Modellierungsanwendungen zurückgegriffen. Die in Abschnitt 2.1.2 festgelegten vier Basisfunktionen sind ein solides Fundament, um ein erstes Modellierungskonzept zu entwerfen. Allerdings besteht noch das Problem der vier Ansichten, die das Arbeiten in 3D vereinfachen. Es wird, als Resultat der obigen Diskussion entschieden, das Ansichtsmodell (vier Ansichten) nicht zu übernehmen. Da die Umsetzung der benutzten Techniken in den Anwendungen nicht öffentlich zugänglich und teilweise Firmengeheimnisse sind, wird versucht diese so ähnlich wie möglich nachzubilden. Dazu werden ähnliche Ansätze aus der Literatur untersucht.

2.2 Bestehende 3D Interaktionstechniken

Zuerst ist es wichtig die Umsetzung der vier Basisfunktionen zu kennen – oder zu definieren, wenn sie nicht zugänglich ist. Im Anschluss folgen eine Reihe von Tipps für die Interaktion in virtuellen Umgebungen.

2.2.1 Erzeugen von Polygonen

Bevor mit Polygonnetzen interagiert werden kann, müssen diese erzeugt werden. In den Modellierungsanwendungen in Abschnitt 2.1.2 werden dazu vordefinierte Objekte und einige Funktionen, um vorhandene Polygone zu erweitern, angeboten.

Vordefinierte Objekte

In *Maya* wird der Modellierungsvorgang begonnen, indem eine geometrische Form gewählt wird und die Maße angegeben werden, in denen das Objekt erstellt wird. Siehe Abbildung 2.3 für ein Beispiel mit einem Würfel und Abbildung 2.4 für eine Auswahl an Formen in *Maya*. Zusätzliche Eigenschaften der Formen können in Textfeldern gesetzt werden, wie z.B. die Aufteilung des Würfels in mehrere Teile oder die Genauigkeit mit der eine Kugel durch Polygone approximiert wird.

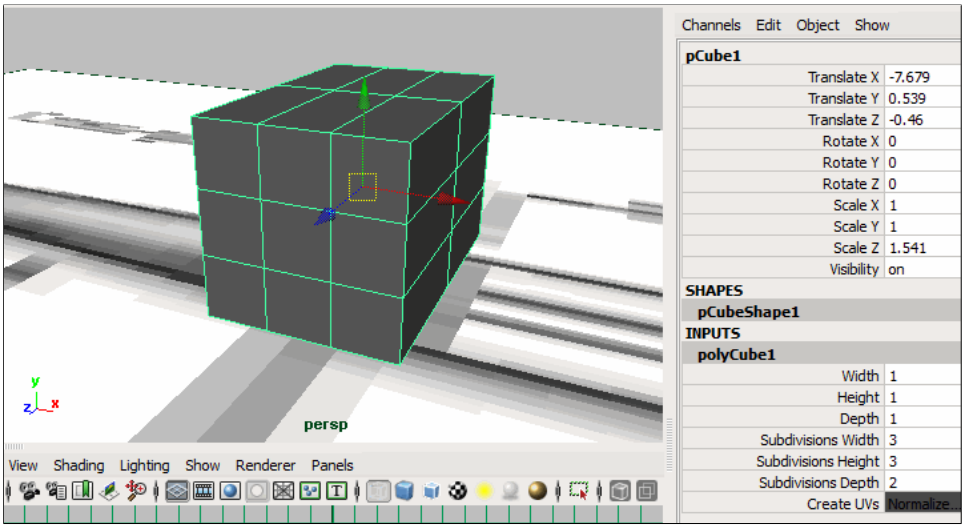


Abbildung 2.3: Erzeugen eines mehrteiligen Würfels in *Maya*. Quelle: [22].

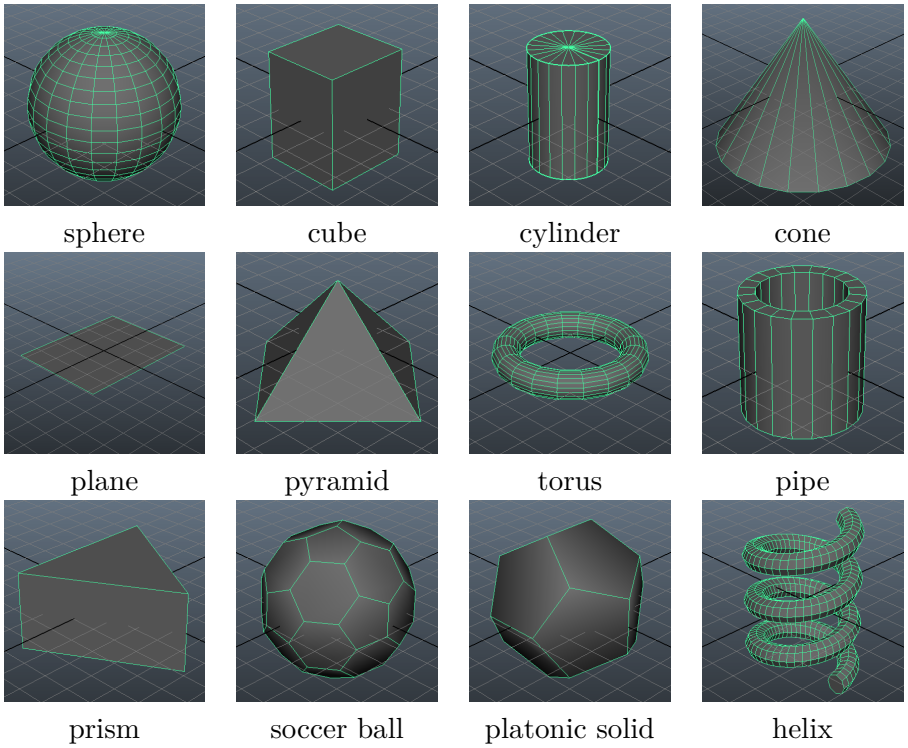


Abbildung 2.4: Mögliche Formen, die in *Maya* erzeugt werden können.

Erweitern von Polygonen

Alternativ zu den vordefinierten Objekten, können Polygone auch einfach einzeln erstellt werden. Dazu werden der Reihe nach die Eckpunkte gesetzt und das Ende des Vorgangs bestätigt. Vorhandene Polygone können außerdem dupliziert oder extrudiert werden. Bei der Extrusion wird die Fläche des Polygons aus seiner Ebene herausgezogen und dadurch ein Volumen hinzugefügt. Aus einem Dreieck würde eine Prisma entstehen.

2.2.2 Selektionstechniken

Nachdem Polygone erzeugt werden, müssen diese auch manipuliert werden können. Dazu muss der Spieler aber die Möglichkeit haben mitzuteilen, welche Komponenten an welchen Objekten manipuliert werden sollen. Dies wird durch das Auswählen der Komponente oder des Objektes durch den Spieler vollbracht. Die Umsetzung der Selektion ist nur für *Blender* frei zugänglich, jedoch nur als Programm-Code ohne zugehörige Dokumentation. Eine Untersuchung des Programm-Codes führt zum Schluss, dass die Selektion von Polygonen mit Hilfe des *Back Buffer Picking* (siehe Unten) durchgeführt wird. Zusätzlich werden verschiedene Techniken der Literatur in Betracht gezogen, um mehr Möglichkeiten offen zu halten.

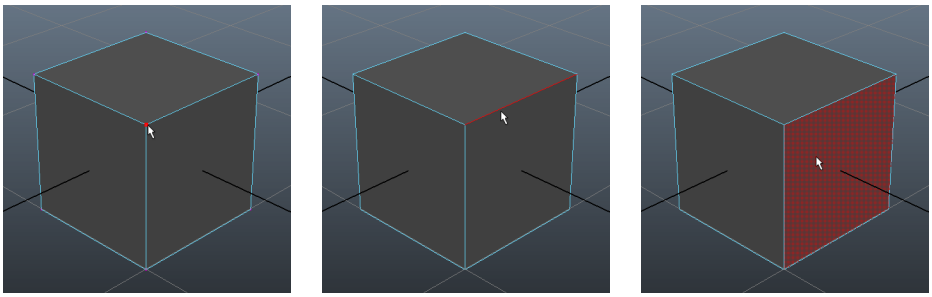
Die Selektion in *Blender* ist sehr umfangreich und bietet sehr unterschiedliche Methoden, kann aber auf zwei Kategorien reduziert werden: Einfache und erweiterte Selektion. Die Dokumentation der Selektion kann [18] entnommen werden. Die einfache Selektion wird generell mit dem Mauszeiger durchgeführt und erfordert keine bereits vorhandene Selektion. Die erweiterte Selektion erfordert eine vorhandene Selektion und fügt weitere Komponenten zur Selektion hinzu oder entfernt welche, anhand bestimmter Kriterien: Ähnlichkeit zur vorhandenen Selektion, Nachbarbeziehungen, Zufall, etc.. Die erweiterte Selektion ist weitaus komplexer als die einfache, daher sollte die einfache Selektion für den neuen Ansatz herangezogen werden.

Zur einfachen Selektion gehören: Ein einfacher Klick, die Auswahl mit einem Rechteck und einem Kreis. Generell wird immer der einfache Klick verwendet, außer etwas anderes wird aus dem Menü aktiviert. Die Komponenten die selektiert werden können, sind: Eckpunkte, Kanten und Flächen der Polygone (siehe Abbildung 2.5 für Beispiele).

Back Buffer Picking/Color Picking

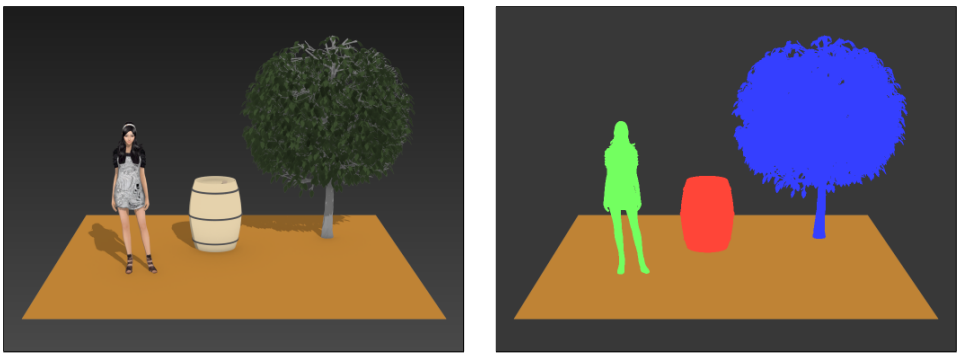
Beim *Back Buffer Picking* (auch *Color Picking* genannt), geht es darum die Objekte einer Szene mit einer individuellen Farbe zu versehen und mit dieser in einen *Frame Buffer*[4] zu zeichnen. Dabei wird *Z-Buffering*[5] aktiviert

[4] Speicherbereich mit der Größe der Auflösung des gezeichneten Bildes.
[5] Funktion zum Zeichnen der sichtbaren Objekte in einer Szene.



(a) (b) (c)

Abbildung 2.5: Selektion der drei Komponentenarten eines Polygons in *Blender*. Eckpunkt (a), Kante (b) und die Fläche (c).



(a) (b)

Abbildung 2.6: Eine einfache Szene mit verschiedenen Modellen in *3ds Max*. (a) Eine gezeichnete Szene aus Spielersicht. (b) Der *Back Buffer* zu (a), gefüllt mit den individuellen Farben der Objekte, um diese identifizieren zu können.

und die Objekte die für den Spieler sichtbar sind, stehen in diesem Speicherbereich. Im *Frame Buffer* können dann beliebige Pixel ausgelesen und die Farbe des Objektes erhalten werden, dadurch kann das Objekt identifiziert werden, welches selektiert werden soll. Ein Beispiel für eine einfache Szene ist in Abbildung 2.6 zu sehen. Beschreibungen, teilweise mit Beispiel-Code für *OpenGL*, sind in [24, 26, 29, 31] zu finden. Die Anzahl an möglich zu selektierenden Objekten entspricht der Bitanzahl die für die Farbe ausgelesen werden. Meist sind das 32 Bit, was 4.294.967.296 Objekten entspricht. Abzüglich einem Wert, der für nicht selektierbare Objekte, wie den Hintergrund, reserviert bleiben sollte.

Alternativ kann auch der *Stencil Buffer*[6] in *OpenGL* angewendet wer-

[6]Vielseitiger kleiner Speicherbereich mit Zusatzinformationen zu jedem Pixel.

den. Der Vorteil besteht im Auslassen eines Zeichenvorgangs. Der Nachteil liegt in der kleinen Größe, wodurch keine ganzen Szenen selektiert werden können, sondern nur bestimmte Objekte in einer Szene. Meist sind es nur 8 Bit, was 256 Objekten entspricht. Beispiel-Code für *OpenGL* findet sich in [25]. Diese Technik kann sehr einfach in eine vorhandene Anwendung integriert werden, falls der *Stencil Buffer* nicht schon genutzt wird.

Falls selbst 32 Bit nicht reichen, oder nicht genug Bits vorhanden sind, um die Objekte zu kodieren, dann findet sich in [24] eine Lösung. Dabei werden die Objekte erst in Gruppen und Untergruppen eingeteilt, und dann schrittweise solange die Gruppe und dessen Untergruppen identifiziert, bis die Bits ausreichen um die Objekte der zuletzt identifizierten Gruppe erfassen zu können. Jeder Identifikationsschritt entspricht dabei einem Selektionsdurchlauf im *Back Buffer*.

Vor- und Nachteile die wichtig für die Umsetzung sind, werden in [31] angeführt – der wichtigste Vorteil ist die Flexibilität:

> *Color Picking funktioniert mit jeder Grafik-Schnittstelle und jeder Art von Geometrie – das schließt 2D, 3D, animiert, dynamisch und alphamaskiert mit ein.*

Diese Art von Selektion ist pixelgenau, was Situationsabhängig positiv und negativ sein kann. Kleine Objekte sind meist nur wenige Pixel groß und können daher eher schlecht selektiert werden. Das gleiche gilt für Objekte, die weit entfernt sind, da die Pixelanzahl mit zunehmender Entfernung abnimmt.

Ray Picking

Bei dieser Technik werden Kollisionsberechnungen angewendet. Ein Strahl wird aus der Kamera „geschossen“ – dessen Kollisionen mit der Szene werden berechnet. Die Kollision am nächsten zur Kamera, ist das Objekt das selektiert wird. Wenn kein Objekt getroffen wurde, gibt es auch keine Selektion. Die Szene ist generell zusammengesetzt aus 3D-Modellen bestehend aus Polygonnetzen, die normalerweise auf Dreiecke normiert sind. Dann muss nur eine Kollisionsfunktion für Strahl und Dreieck implementiert werden. Andere 3D-Modelle sind möglich, finden aber keine Anwendung in Spielen, sondern eher in Animationsfilmen.

Die Kollisionsberechnungen mit allen Objekten in der Szene ist sehr aufwändig. Die wichtigsten Optimierungsarten sind: Räumliche Aufteilung der Szene, z. B. *Object Bounding Volumes*, und Parallelisierung der Berechnungen. Auch diese Technik ist ansatzweise pixelgenau, kann aber keine Objekte treffen, die keine Fläche oder Volumen haben. Punkte und Linien könnten damit nicht selektiert werden, was aber der nächste Ansatz kann.

Cone/Spotlight Selection

Ähnelt dem *Ray Picking*, allerdings wird statt einem Strahl ein Kegel zur Kollisionsberechnung benutzt. [7, 9, 11] beschreiben solch eine Technik. Eine Beschreibung der *Spotlight*-Technik findet sich in [10], allerdings keine Angaben zur Umsetzung. Eine eigene Idee zur Umsetzung ist in Kapitel 3 angegeben.

Vorteile sind, dass auch Objekte ohne Fläche und Volumen, wie Punkte und Linien, selektiert werden können. Außerdem erleichtert es allgemein die Selektion, da der Mauszeiger nur in der Nähe eines Objekts zu sein braucht und nicht pixelgenau auf dem Objekt, wie bei den vorherigen Techniken. Das Problem dabei ist aber, dass sich meist mehrere Objekte innerhalb des Kegels befinden. Es muss also entschieden werden, welches davon selektiert wird, wenn nicht alle Objekte innerhalb des Kegels selektiert werden sollen.

Highlighting

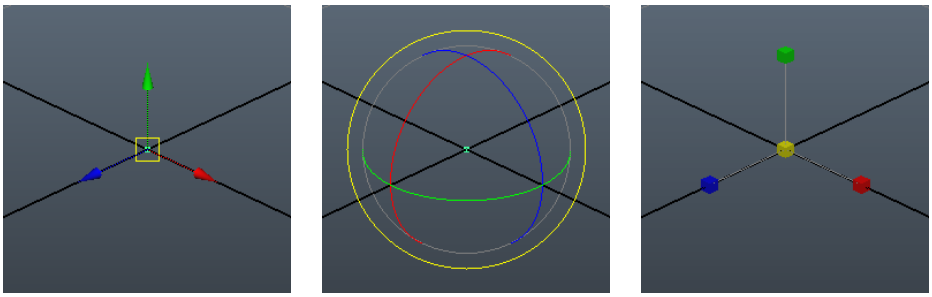
In *Maya 2013* werden Objekte/Komponenten hervorgehoben, wenn sie selektiert werden können. In [10, S. 4] wird stattdessen der Kegel hervorgehoben, dabei besteht allerdings die Ungewissheit, welches Objekt selektiert werden würde. Die Hervorhebung findet durch einen Farbwechsel statt. Solch eine Reaktion von der Selektionstechnik hilft dem Nutzer seine eigenen Aktionen schneller durchzuführen, da nicht erst mehrmals versucht werden muss etwas auszuwählen, um zu erkennen ob die Selektion auch gelungen ist.

2.2.3 Manipulation

Da nun Objekte erzeugt und selektiert werden, sollte die Selektion nun manipuliert werden können. Dazu gibt es drei wichtige Methoden, die auch in den vier Modellierungsanwendungen wiedergefunden werden: Translation, Rotation und Skalierung.

Virtual Trackball

Ein virtueller Ball, der durch Klicken und Ziehen (*Drag and Drop*) mit dem Mauszeiger gedreht werden kann. Die Orientierung des Balls entspricht dabei der Orientierung des ausgewählten Objektes. Die Umsetzung basiert auf Rotationsmatrizen und kann [4, S. 7f] entnommen werden. Der Nachteil dieser Technik, ist der Nachteil von Rotationsmatrizen: Der Verlust an Freiheitsgraden bei komplexen Rotationen, was zu einem *Gimbal Lock* führen kann. Dann kann nur noch um zwei Achsen gedreht werden. Laut [6, S. 2] wird diese Art von Technik auch in *Maya* benutzt, wie in Abbildung 2.7 gesehen werden kann.



(a) (b) (c)

Abbildung 2.7: Die drei Manipulationswerkzeuge in *Maya*: Translation (a), Rotation (b) und Skalierung (c).

ARCBALL

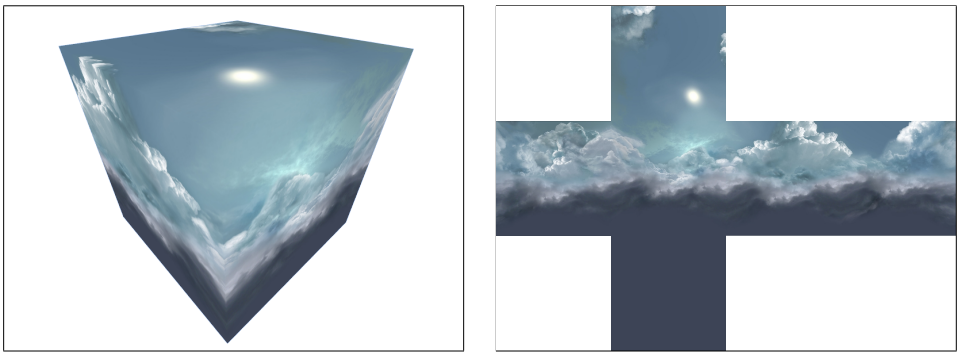
Den Nachteil des *Gimbal Locks* hat diese Technik nicht, da sie auf Quaternionen basiert. Der *Arcball* ist dem *Virtual Trackball* ähnlich, hat aber eine andere Umsetzung, die in [12, S. 3f] eingesehen werden kann. Laut einer Studie in [8] wurde der *Arcball* gegenüber dem *Virtual Trackball* leicht bevorzugt, der Unterschied ist aber nicht signifikant. Die Implementierung soll außerdem einfacher sein, als bei den bisherigen *Virtual Trackballs*. Eine Analyse der *Virtual Trackball* Techniken nach über 10 Jahren, wird in [6] beschrieben und teilweise auch Verbesserungen vorgeschlagen.

Translation und Skalierung

Die Techniken zum Bewegen und Skalieren von Objekten ist in Abbildung 2.7 gezeigt. Beide bilden ein Koordinatensystem, und an den Enden der einzelnen Achsen sind Pfeile (Translation) oder Würfel (Skalierung), um das Objekt entlang dieser Achse zu manipulieren. In dem Koordinatenursprung ist eine zusätzliche Funktion eingebaut und beide Techniken unterscheiden sich darin. Die Translation hat dort ein kleines Rechteck, bei dem das Objekt entlang einer Fläche orthogonal zum Sichtvektor der Kamera bewegt wird. Die Skalierung hat dort einen weiteren Würfel, der das Objekt entlang allen Achsen homogen skaliert. Alle diese Funktionen werden durch *Drag and Drop* mit den jeweiligen Steuerungselementen benutzt.

Snapping

Beim Verschieben von Objekten kann häufig eine Funktion genutzt werden, um Objekte an anderen Objekten oder an einem Raster ausgerichtet zu erstellen. Dabei wird beim Verschieben eine mögliche Position angeboten, die z. B. auf anderen Objekten ist oder an einem wichtigen Punkt in der Szene. Beim Verschieben über diese Position, bleibt das zu verschiebende Objekt in



(a) (b)

Abbildung 2.8: (a) Eine gezeichnete Szene umgeben von einer *Skybox*. (b) Die benutzte Textur dieser *Skybox*. Quelle für (b): [28].

der Position in einem gewissen Abstand stecken und ermöglicht so schnelle und gleichzeitig präzise Verschiebungen. Meistens werden einzelne Punkte von Polygonen beim Modellieren, in andere Punkte gelegt (ge-*snapped*), um eine geschlossene Oberfläche zu garantieren. Die Funktion wird *Snapping* genannt und wurde schon sehr früh in [1] entwickelt und gehört zu den Standardfunktionen der Modellierungsanwendungen.

2.2.4 Löschen von Polygonen

In *Maya* können Eckpunkte, Kanten und Flächen durch Selektieren dieser und einem Tastendruck entfernt werden. Allerdings wird immer versucht ein geschlossenes Modell beizubehalten, was beim Löschen dazu führt, dass oft neue Kanten und Flächen entstehen. Außerdem gibt es unterschiedliche Situationen in denen die Komponenten nicht auf diese Weise gelöscht werden können, sondern nur durch eine gesonderte Funktion.

2.2.5 Sonstiges

Es gibt noch verschiedenste Elemente zu beachten, wenn ein 3D Interaktionskonzept in eine virtuelle Umgebung eingebaut wird.

Skybox

In einer leeren Szene ohne Umwelt, kann es schwer fallen, sich zu orientieren. Dies ist ein Problem, welches auch während der Implementierung des Prototypen für diese Arbeit, erkannt wurde. Abhilfe kann durch eine simulierte Umwelt geschaffen werden, wie sie in [5] beschrieben wird. Die sogenannte *Skybox* die dazu genutzt wird, besteht aus sechs Texturen die als Würfel zusammenstellt werden, wie in Abbildung 2.8 gesehen werden kann. Die ganze

Szene spielt sich im Würfel ab, wobei sich der Spieler immer im Zentrum befindet. Dies soll dem Spieler helfen nicht die Orientierung zu verlieren, da er sich an den texturierten Innenseiten des Würfels orientieren kann.

Allgemeine Tipps

Die folgende Liste besteht aus hilfreichen Evaluationsergebnissen aus anderen Publikationen:

1. Extra Freiheitsgrade vermeiden [7, S. 4f],
2. Reduktion der Dimensionen eines Problems, vereinfacht das Problem [3, S. 29],
3. Transparenz kann sehr hilfreich sein [7, S. 5],
4. die Geschwindigkeit der Anwendung ist wichtig für fließende Interaktion [14, S. 2],
5. Schatten sind wichtig für die Tiefenwahrnehmung [14, S. 3],
6. 2D-Eingabegeräte sind vorteilhafter als 3D-Eingabegeräte [3, S. 29],
7. Texturen sind wichtig zur Tiefenwahrnehmung [13, S. 11].

Diese Tipps werden in nachfolgenden Kapiteln als „Tipp Nr. X“ referenziert. Wobei X der Nummerierung des Tipps in der Aufzählung entspricht.

Kapitel 3

Interaktionskonzepte

Aus den vorgestellten Funktionen aus Kapitel 2 werden nun zwei Konzepte zusammengestellt. Beide sind einfach gehalten und bieten die vier grundlegenden Funktionen aus Abschnitt 2.1.2, mit denen alle möglichen Gitternetzmodelle modelliert werden können. Es werden zwei verschiedene Ansätze vorgestellt, da diese in Kapitel 4 gegeneinander evaluiert werden, um Defizite und Vorteile der einzelnen Konzepte herauszufinden und eine Entwicklungsrichtung vorgeben zu können.

Im weiteren Verlauf dieser Arbeit ist der Begriff „Konzept“, auf ein Interaktionskonzept zum Modellieren von 3D-Modellen zu beziehen. In diesem Kapitel werden häufig die Begriffe Polygon und Vertex[1] benutzt. Ein Vertex ist ein Teil eines Polygons und somit schließt der Begriff Polygon, die Vertices des Polygons mit ein. Diese Arbeit ist beim Begriff Polygon auf reine Dreiecke beschränkt, ist in der Computergrafik aber geläufig.

3.1 Anforderungen

In Kapitel 1 wurde als Ziel festgelegt, ein *einfaches* Konzept zu erschaffen, dass dem Spieler ermöglicht seine virtuelle Umwelt *uneingeschränkt* zu gestalten. Damit stehen die qualitativen Anforderungen an die Konzepte fest:

1. Einfach,
2. möglichst uneingeschränkt.

In Kapitel 2 wurden die notwendigen vier Basisfunktionen herausgearbeitet, die dazu benötigt werden. Womit die funktionalen Anforderungen feststehen, mit denen der Spieler direkt interagiert:

1. Erstellung,
2. Selektion,
3. Manipulation,

[1] Eckpunkt – eine der drei Komponenten des Polygons. Meist mit zusätzlichen Informationen verbunden, z. B. Texturkoordinaten oder einer Farbe.

4. Löschung.

3.2 Implementierung

Im Prototyp[2] sind beide Konzepte, wie in diesem Kapitel beschrieben umgesetzt. Während der Entwicklung des Prototyps wurden Kompromisse getroffen, einige Konzeptteile anders zu implementieren. Diese Entscheidungen wurden in diesem Kapitel dokumentiert und dabei explizit der Prototyp erwähnt.

Als Basis für das Projekt wurden *C++* und *OpenGL* gewählt, um keine Einschränkungen hinsichtlich der Leistung oder der Umsetzungsmöglichkeiten zu haben. Es wurde Tipp Nr. 4 befolgt, sodass der Spieler keine Einbußen, durch schlechte Leistung, in seinen Interaktionen hat. Zusätzlich wurde *Simple and Fast Multimedia Library* (SFML) benutzt, um das Fenster der Anwendung zu erstellen und Eingabegeräte abzufragen. Zum Einlesen der Texturen wurde die *FreeImage Library* benutzt.

3.3 Konzeptunabhängige Eigenschaften

Einige Eigenschaften die in beiden Konzepten vorkommen, werden hier erklärt. Diese Eigenschaften gehen meist aus den Regeln in Abschnitt 2.2.5 hervor oder sind Basis eines Spiels.

3.3.1 Ausgangsszene

Die Szene in der sich der Spieler befindet, besteht aus einem texturierten ebenen Boden und einer üblichen *Skybox*. Im Prototyp wird eine Wiese und ein leicht bewölkter Himmel mit Sonnenschein vorgetäuscht (siehe Abbildung 3.1). Die Skybox ist nach der Beschreibung in Abschnitt 2.2.5 implementiert worden. Die Skybox und der Boden sollen die Orientierung des Spielers unterstützen, da dies ein Problem in der Anfangsphase des Prototyps war, in der diese beiden Objekte nicht in die Szene integriert waren. Die Textur des Bodens soll außerdem noch die Tiefenwahrnehmung erleichtern (Tipp Nr. 7). Es sind keine Objekte vorplatziert und die Kamera befindet sich in der Mitte der Szene.

3.3.2 Navigation

Der Spieler steuert die Kamera, die der Einfachheit halber fliegen kann und somit nicht direkt eine Charakterfigur repräsentiert, was in einem Spiel der Fall wäre. Es ist anzumerken, dass es in *Minecraft* einen Baumodus gibt, in

[2] Anwendung in der die Konzepte implementiert sind, um diese evaluieren zu können.

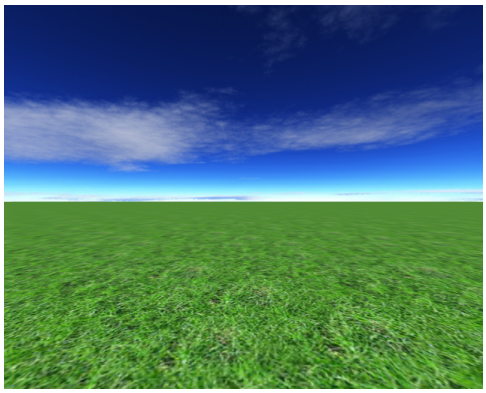


Abbildung 3.1: Skybox und texturierter Boden als Ausgangsbasis für die Szene im Prototyp. GUI[3]-Elemente sind ausgeblendet.

dem der Spieler ebenfalls fliegen kann. Dieser wird zum schnelleren Bauen verwendet wird.

Drehung

Die Drehung wird mit der Mausbewegung gesteuert. Die Kamera ist in der vertikalen Drehrichtung beschränkt, um die Orientierung des Spielers im Raum nicht zu verletzen. Daraus würde eine spiegelverkehrte Steuerung und ein verwirrter Spieler resultieren. Dieser Umstand wurde im Anfangsstadium der Entwicklung erkannt. Durch die Einschränkung kann der Spieler nur 89 Grad nach oben und unten sehen, da bei 90 Grad zusätzlich ein mathematisches Problem entsteht und die Sicht „springt“. In horizontaler Richtung gibt es keine Beschränkung.

Die Drehung wird mit der Mausbewegung gesteuert. Die Umsetzung der Bewegung zur Drehung ist festgelegt durch

$$\theta_y = \frac{\Delta x}{\upsilon}, \tag{3.1}$$

$$\theta_x = \frac{\Delta y}{\upsilon}. \tag{3.2}$$

θ_y ist die Drehung in Radianten um die y-Achse. Δx ist dabei die Veränderung der Mausposition in Pixel auf der x-Achse. Die zweite Gleichung ist dazu analog, nur mit vertauschten Achsen. υ bestimmt die Umwandlung von Pixel zu Radianten und wurde mit 125 im Prototyp festgelegt. Diese Einstellung wurde als angenehm empfunden, hängt aber wahrscheinlich von der eingestellten Mausgeschwindigkeit des Betriebssystem und dem Spielerempfinden ab. Die Rotationen müssen, abhängig vom gewählten Koordinatensystem, negativ interpretiert werden.

[3] engl.: Graphical User Interface.

Taste	Kamerabewegung
W	Vorwärts
A	Links
S	Rückwärts
D	Rechts

Tabelle 3.1: WASD-Steuerung. Wird bei Spielen in der Egoperspektive angewandt.

Bewegung

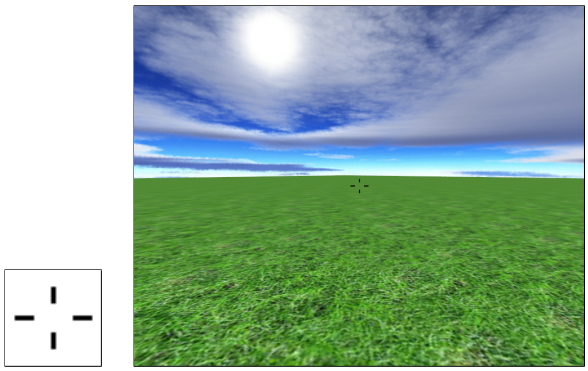
Die Bewegung der Kamera wird mit der Tastatur gesteuert. Es wurde die Standardsteuerung für Spiele in der Egoperspektive gewählt, die sich in den letzten 15 Jahren etabliert hat – die WASD-Steuerung (siehe Tabelle 3.1). Zusätzlich, um die Höhe zu ändern, wurden die Tasten Q und E gewählt, da sie direkt daneben liegen. Die Rotation um die y-Achse beeinflusst die Bewegung nach vorn, sodass sich der Spieler immer in seine Blickrichtung vorwärts bewegt. Diese Drehung kann als Drehung des Körpers interpretiert werden. Die Rotation um die x-Achse beeinflusst die Bewegung allerdings nicht, sodass beim Hoch- und Herunterschauen, nicht die Position auf der y-Achse geändert wird. Diese Drehung kann als Drehung des Kopfes interpretiert werden. Die Blickrichtung die im späteren Verlauf häufig referenziert wird, ist die Kombination aus beiden Rotationen, begonnen mit der x-Achse.

Fadenkreuz

In der Mitte des Fensters der Anwendung befindet sich ein Fadenkreuz, wie in Abbildung 3.2 gesehen werden kann. Die Normale der Blickrichtung führt durch das Fadenkreuz. Diese Normale wird für die Selektionsmethoden benutzt und somit gibt das Fadenkreuz eine entscheidende Hilfestellung beim Selektieren. Diese Technik wird in den meisten Spielen mit der Egoperspektive angewandt, um das Anvisieren zu erleichtern und somit auch die Interaktion mit der Umwelt simpler zu gestalten.

3.3.3 Modellierungsmodus

Beide Konzepte bestehen aus zwei unterschiedlichen Modi: Dem Weltmodus und dem Objektmodus, in denen unterschiedliche Arten von Funktionen angewendet werden können. Diese Trennung der Funktionen ist hilfreich, da im Weltmodus nur Funktionen auf ganzen Objekten stattfinden und im Objektmodus nur Funktionen auf Polygon-Ebene. Dies vereinfacht die Steuerung im jeweiligen Modus, da ansonsten ein Modus, alle Funktionen beinhalten müsste. Die Komplexität würde steigen, was das Gegenteil der qualitativen Anforderungen wäre.



(a) (b)

Abbildung 3.2: Das Fadenkreuz, das dem Spieler das Selektieren erleichtern soll. (a) Das reine Fadenkreuz. (b) Fadenkreuz eingebunden in der Szene.

Der Spieler startet im Weltmodus. In diesem werden Objekte erstellt und zwischen den Objekten das ausgewählt, welches manipuliert werden soll. Die möglichen Manipulationen sind das Rotieren, Skalieren und Verschieben. Außerdem kann nur mit einem ausgewählten Objekt in den Objektmodus gewechselt werden.

Im Objektmodus wird die Struktur des ausgewählten Objektes modelliert. Es können Polygone erzeugt werden, meist indem Komponenten erzeugt und verbunden werden. Diese Komponenten können ausgewählt, manipuliert und gelöscht werden. Die Manipulation beschränkt sich bei den Komponenten auf das Verschieben dieser.

Beide Modi decken alle vier funktionalen Anforderungen ab.

3.3.4 Tiefenebene

Im Objektmodus werden Polygone und deren Komponenten im dreidimensionalen Raum platziert. Dies ist eine schwierige Aufgabe, da nur eine zweidimensionale Sichtfläche durch das Display vorhanden ist. Tipp Nr. 2 weist darauf hin, dass die Reduktion von Dimensionen die Komplexität eines Problems reduziert. Eine Reduktion von 3D auf 2D könnte demzufolge den Modellierungsprozess vereinfachen. Daraus ist die Tiefenebene entstanden, die der Tiefe entspricht, in der der Spieler die Polygone bearbeitet. Es ist eine zweidimensionale Ebene im dreidimensionalen Raum, auf der Modellierungsarbeiten verrichtet und die beeinflusst werden kann (siehe Abbildung 3.3). Gesteuert wird die Ebene durch die aktuelle Selektion der Vertices. Dies erscheint einfacher als die direkte Manipulation der Ebene, da ansonsten die Ebene platziert und rotiert werden müsste. Eine einfache Selektion hingegen, ist mit ein oder zwei Mausklicks erledigt. Der Nachteil für den Spieler ist, das nicht direkt bekannte Verhalten der Ebene, bei der Selektion der

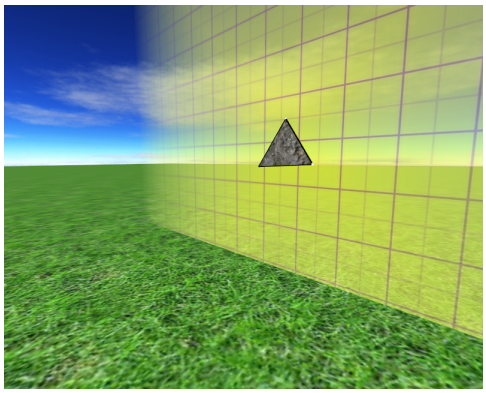


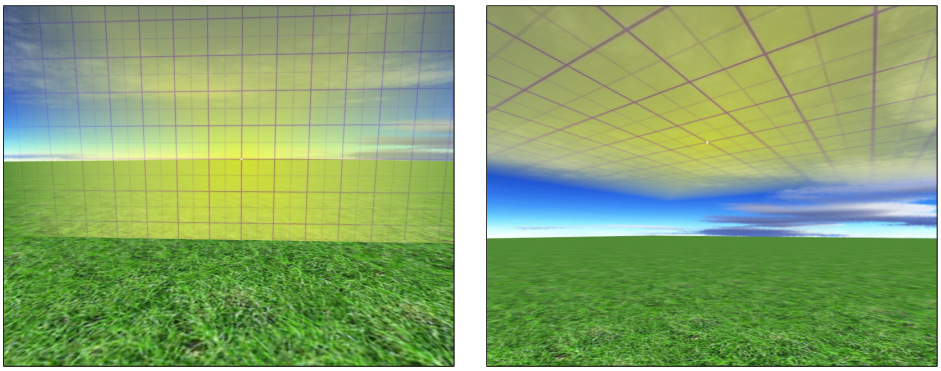
Abbildung 3.3: Die Tiefenebene (weiße gerasterte Ebene) im Prototyp. GUI-Elemente sind ausgeblendet.

Vertices, welches erst gelernt werden müsste. Dadurch entsteht eine Lernkurve, welche aber auch in Spielen wiederzufinden ist. Zusätzlich wurde Tipp Nr. 3 berücksichtigt und die Ebene transparent gezeichnet, sodass dahinterliegende Polygone trotzdem noch gesehen und bearbeitet werden können. Außerdem wird die Tiefenwahrnehmung durch ein Raster auf der Ebene verstärkt, sodass die Ausrichtung der Ebene einfacher erkannt wird (Tipp Nr. 7). Das Raster hilft auch beim Modellieren von rechtwinkligen Objekten, da sich der Spieler an den Linien orientieren kann. Im Prototyp ist es als transparente Textur in die Ebene gelegt worden. Nach der Vorstudie wurde die Auflösung der Textur erhöht, um die Linien des Rasters schmaler zu machen. Die Farben der Ebene wurden so gewählt, dass sie einen Kontrast zu den Vertices, Kanten und Highlights bieten, die schwarz und weiß sind.

Abhängig von der Anzahl an selektierten Vertices, wird die Ebene unterschiedlich berechnet. Die Ebene wird so berechnet, dass die selektierten Vertices in der Ebene liegen. Durch diesen Ansatz, wird dem Spieler auf einfache Weise, genügend Kontrolle gegeben.

Eine Selektion

Dieser Ebenenmodus ist gedacht um beliebige Objekte zu modellieren. Während ein Vertex selektiert ist, kann die Ebene dabei zwei Ausrichtungen haben: Vertikal und Horizontal (siehe Abbildung 3.4). Im vertikalen Fall, ist die Ebene auch noch abhängig von der Kameraposition, denn sie ist immer in Kamerarichtung ausgerichtet. Es soll dem Spieler zusätzliche Kontrolle geben und löst gleichzeitig das Problem der Ausrichtung der Ebene. Der Spieler kann sowohl die Tiefe, als auch die Höhe der Vertices, signifikant einfach. Diese Ausrichtung der Ebene ist für alle möglichen Objekte gedacht, da jede beliebige Position erreicht werden kann. Die Normale der Ebene zeigt vom Mittelpunkt der Ebene immer in Richtung der Kameraposition,



(a) (b)

Abbildung 3.4: Die beiden möglichen Ausrichtungen der Tiefenebene, bei nur einem selektierten Vertex. (a) Die vertikale Ausrichtung, die immer zur Kamera ausgerichtet ist. (b) Die horizontale Ausrichtung der Ebene.

aber projiziert auf die Höhe des Mittelpunktes, sodass eine Achse der Ebene immer die y-Achse bleibt und somit die Ebene vertikal ist. Mit folgenden Formeln kann die Normale der Ebene bestimmt werden:

$$\vec{c}_p = \begin{pmatrix} \vec{c}.x \\ \vec{pl}.y \\ \vec{c}.z \end{pmatrix}, \tag{3.3}$$

$$\hat{n} = \frac{\vec{c}_p - \vec{pl}}{\|\vec{c}_p - \vec{pl}\|}. \tag{3.4}$$

$\vec{c}$ ist die Kameraposition, $\vec{pl}$ der Mittelpunkt der Ebene, der dem selektierten Vertex entspricht und $\hat{n}$ ist die Normale der Ebene. Mit der y-Achse als zusätzliche Normale, kann mit dem Kreuzprodukt eine Orthonormalbasis berechnet werden, mit der eine Rotationsmatrix erstellt werden kann, mit der die Ebene gezeichnet werden kann (Falls die *Render-Engine*[4] auf Rotationsmatrizen basiert, wie der Prototyp).

Bei der horizontalen Ausrichtung zeigt die Normale der Ebene immer in den Himmel. Die Normale ist somit gleich der y-Achse. Zum Erstellen einer Rotationsmatrix, können die verbleibenden Achsen der Szene genutzt werden. Diese Ausrichtung ist hilfreich, um Vertices auf die selbe Höhe zu setzen oder das Fundament eines Objektes zu errichten.

Zweifache (n-fache) Selektion

Für gerade Objekte und 90 Grad Ecken ist die Selektion von zwei Vertices gedacht. Es gibt drei Ausrichtungen der Ebene in diesem Ebenenmodus die

[4]Interner Anwendungsteil, der zum Zeichnen der 3D-Szene zuständig ist.

in Abbildung 3.5 illustriert sind. Für den Fall, dass mehr als zwei Vertices selektiert sind, werden nur die zwei Vertices zur Berechnung benutzt, die als erste selektiert wurden.

Bei der ersten Ausrichtung führt die Ebene durch beide Vertices und liegt parallel zur y-Achse. Dazu wird eine Normale $\hat{n}_{points}$ vom ersten Vertex $\vec{p}_1$ zum zweiten Vertex $\vec{p}_2$ gezogen. Damit die Ebene parallel zur y-Achse steht, wird nun das Kreuzprodukt von $\hat{n}_{points}$ und der y-Achse $= (0, 1, 0)^T$ errechnet. Das Kreuzprodukt wird normalisiert und $\hat{n}_{cross}$ genannt. $\hat{n}_{cross}$ ist die Normale der Ebene für die erste Ausrichtung und der Mittelpunkt der Ebene liegt im Zentrum der beiden Vertices $\vec{p}_1$ und $\vec{p}_2$.

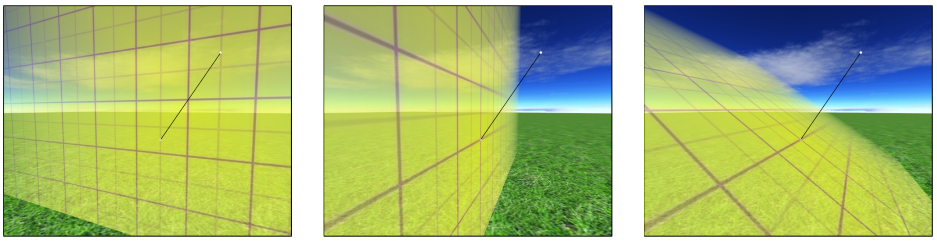
Die zweite Ausrichtung entspricht einer Ebene orthogonal zur ersten Ausrichtung, auch parallel zur y-Achse. Es reicht also $\hat{n}_{cross}$ um 90 Grad um die y-Achse zu drehen. Oder es wird ein weiteres Kreuzprodukt zwischen der y-Achse und $\hat{n}_{cross}$ errechnet. Eine Normalisierung ist nicht erforderlich, da beide Normalen orthogonal zueinander stehen. Dieses Kreuzprodukt entspricht der Normalen der Ebene. Als Position wird der Vertex $\vec{p}_1$ genutzt, da diese Ebene nicht mehr durch beide Vertices führt und somit nicht im Zentrum der beiden Vertices liegen sollte. Diese Ausrichtung ist gedacht, um 90 Grad Ecken zu modellieren und daher sollte die Ebene zumindest in einem der beiden Vertices liegen, da ansonsten keine 90 Grad Ecke entsteht.

Die dritte Ausrichtung führt auch nur durch einen der beiden Vertices und soll eine andere Ebene bieten, mit der 90 Grad Ecken modelliert werden können. Die Normale entspricht $\hat{n}_{points}$ und die Position entspricht wieder dem ersten der beiden Vertices, $\vec{p}_1$.

Wenn $\hat{n}_{points}$ der y-Achse entspricht, kann kein Kreuzprodukt der beiden errechnet werden und somit keine Normalen für die ersten beiden Ausrichtungen. Es gibt dann keine logische Ausrichtung für die beiden Ebenen und es muss eine Ausnahme definiert werden. Im Prototyp ist die erste Ausrichtung in Richtung der z-Achse und die zweite Ausrichtung in Richtung der x-Achse. So bleiben die Ebenen parallel zur y-Achse und orthogonal zueinander.

3.4 Konzept 1

Im ersten Konzept wird nicht von einem vorhandenen Modell ausgegangen, wie in den vorgestellten Modellierungsanwendungen (siehe Abschnitt 2.2.1), sondern ist es dem Spieler überlassen, wie das Modell angefangen wird. Dies soll dem Spieler keine Formen aufzwingen, sodass seine Phantasie nicht beeinflusst wird. Die Manipulation richten sich nach den Techniken, die in den 3D-Modellierungsanwendungen verwendet werden.



(a) (b) (c)

Abbildung 3.5: Die drei möglichen Ausrichtungen der Tiefenebene, bei zwei oder mehr selektierten Vertices. (a) Die Ebene führt durch beide Vertices und ist parallel zur y-Achse. (b) Die Ebene ist orthogonal zu der in (a) gezeigten und parallel zur y-Achse. (c) Die Ebene entspricht der Normalen zwischen den beiden selektierten Vertices.

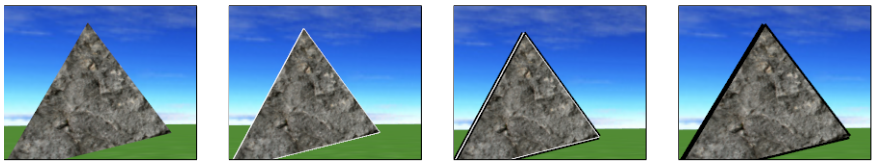
Wechsel des Modellierungsmodus

Solange ein Objekt selektiert ist, wird die Distanz vom Spieler zu diesem Objekt berechnet. Ab einem Grenzwert, wird der Modellierungsmodus gewechselt. Unter dem Grenzwert befindet sich der Spieler im Objektmodus, da er beim Modellieren des Objekts Präzisionsarbeit leisten muss. Und über dem Grenzwert, befindet sich der Spieler im Weltmodus. Aus der Distanz hat der Spieler bessere Übersicht, was ihm beim Verschieben und Selektieren der Objekte hilfreich sein sollte. Der Grenzwert ist abhängig von der *Bounding Box*[5] des Objekts festgelegt. Da das Objekt bei der Erzeugung kein Volumen besitzt und nur aus einem Vertex besteht, könnte der Spieler nicht in den Objektmodus wechseln, um dieses zu erweitern. Daher ist ein Mindestgrenzwert erforderlich, welcher im Prototyp mit etwa einem Meter festgelegt wurde. Dem Spieler wird im Prototyp permanent angezeigt, in welchem der beiden Modi er sich befindet, sodass diese Information ersichtlich ist.

3.4.1 Weltmodus

Zu Beginn kann im Weltmodus nur ein Objekt erstellt werden, da alle anderen Funktionen auf vorhandene Objekte angewiesen sind. Nachdem ein Objekt erstellt wurde, wird automatisch in den Objektmodus gewechselt, da die Funktionen des Weltmodus eher uninteressant sind, solange kein komplexeres Objekt erstellt wurde. Dies wird erreicht, indem das Objekt immer in einer Entfernung unter dem Mindestgrenzwert gesetzt wird, in der es zum Wechsel in den Objektmodus kommt. Beim Erstellen eines neuen Objektes, wird dessen erster Vertex in einem festen Abstand entlang des Richtungsvektors der Kamera gesetzt, welcher auch der Position des ganzen Objektes

[5] Eine Box die das Objekt komplett umschließt.



(a) (b) (c) (d)

Abbildung 3.6: Ein Objekt in unterschiedlichen Selektionsphasen: (a) Ohne Highlight, (b) mit Highlight, (c) mit Highlight und selektiert und (d) nur selektiert.

entspricht. Also in einem festen Abstand vor dem Spieler.

Selektion

Vorhandene Objekte können selektiert werden, um diese dann zu manipulieren. Die letzte Selektion wird von der aktuellen Selektion ersetzt, sodass immer nur ein Objekt selektiert sein kann. Dies soll den Wechsel den Wechsel zwischen den Objekten vereinfachen. Durch eine einzige Selektion, werden dann auch nur die Polygone des selektierten Objektes manipuliert und Verbindungen zwischen den Objekten ausgeschlossen. Diese Restriktion reduziert die Komplexität des Objektmodus, da ansonsten die Verbindungen der Objekte verwaltet werden müssten. Die Selektion kann auch aufgehoben werden, sodass kein Objekt mehr selektiert ist. Nochmaliges Selektieren eines bereits selektierten Objekts hebt dessen Selektion auf.

Die Highlight-Methode aus Abschnitt 2.2.2 wird angewandt, um dem Spieler mitzuteilen, dass eine Selektion getätigt werden kann. Im Prototyp werden die Kanten der Polygone weiß gezeichnet, wie in Abbildung 3.6 gesehen werden kann. Wenn ein Objekt ein Highlight hat, kann dieses durch einen Linksklick mit der Maus ausgewählt werden. Die Kanten eines selektierten Objekts werden dick und schwarz gezeichnet. Ein selektiertes Objekt kann auch wieder mit einem Linksklick abgewählt werden, falls dieses oder keines der Objekte ein Highlight hat.

Die Selektion von Objekten wird im Prototyp mit der *Cone Selection* aus Abschnitt 2.2.2 durchgeführt. Da keine mathematische Berechnung in den Referenzen angegeben wurde, wurde eine eigene entwickelt. Für jedes Objekt in der Szene wird von der Kameraposition $\vec{cp}$ zum Objektmittelpunkt $\vec{c_i}$ ein normierter Vektor

$$\hat{n}_i = \vec{c}_i - \vec{c}_{pos} \tag{3.5}$$

berechnet, der mit dem Richtungsvektor der Kamera $\hat{c}$ verglichen wird, indem zwischen beiden das Skalarprodukt berechnet wird. Der Wert

$$f_i = \hat{n}_i \cdot \hat{c}_{dir} \tag{3.6}$$

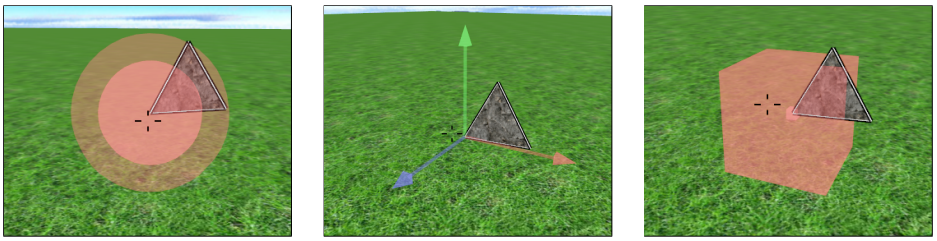
gibt an, wie nah das Objekt am Blickzentrum der Kamera liegt. Der Wertebereich von f_i ist $[-1, 1]$. Umso höher der Wert, desto näher liegt das Objektzentrum am Blickzentrum. Ein negativer Wert bedeutet, dass das Objekt hinter dem Spieler liegt. Das Objekt mit dem höchsten f_i-Wert, erhält das Highlight. Dies würde aber bedeuten, dass immer ein Objekt ein Highlight erhält und die Kegel-Eigenschaft der Technik ignoriert werden würde. Dies kann mit einem Grenzwert erreicht werden, der einem Winkel entspricht, in dem sich die zu untersuchenden Objekte befinden müssen. Da der Wert von f_i allerdings nicht linear ist, können die Werte nicht direkt einem Winkel zugeordnet werden. Ein Winkel kann allerdings mit der $\cos(x)$-Funktion in den selben Wertebereich umgerechnet werden, da dies auch beim Skalarprodukt zweier Vektoren indirekt passiert. Der resultierende Wert wird w genannt und jeder f_i-Wert, der niedriger ist als w, wird aussortiert. Zwischen den verbleibenden Objekten, die sich im Kegel befinden, wird der höchste f_i-Wert gesucht und das Objekt i erhält das Highlight. Falls alle Objekte aussortiert wurden, gibt es auch kein Highlight.

Die *Cone Selection* wurde schon für den Objektmodus implementiert, sodass der Fokus beim Prototyp auf wichtigere Funktionen gelegt werden konnte. Das *Back Buffer Picking* aus Abschnitt 2.2.2 wäre ansonsten gewählt worden, da damit wirklich das Objekt selektiert werden würde, das vom Spieler mit dem Fadenkreuz anvisiert wird.

Manipulation

Mit dem selektierten Objekt können drei verschiedene Manipulationen durchgeführt werden: Die Rotation, die Translation und die Skalierung (siehe Abbildung 3.7). Dies sind die gleichen Manipulationsarten, die in den Modellierungsanwendungen zum Einsatz kommen. Alle drei Manipulationen orientieren sich an den Manipulationen in diesen Anwendungen. Um die Komplexität geringer zu halten, ist nur eine Manipulationsfunktion gleichzeitig aktiv. Der Spieler wählt die gewünschte Funktion, wie auch in den Modellierungsanwendungen. Mit dem Mausrad, wird zwischen den drei Funktionen durchrotiert.

Bei diesen Techniken ist ebenfalls eine Selektion nötig, um zu erfassen welcher Bereich des Steuerelements manipuliert wird. Bei der Rotation gibt es den äußeren hellen Bereich und den dunkleren Bereich. Beide Bereiche manipulieren das Objekt auf unterschiedliche Weise, daher ist es wichtig, dies zu unterscheiden. Die Translation besteht aus drei 3D-Pfeilen, die das Objekt in unterschiedliche Richtungen verschieben. Jeder Pfeil entspricht einer Achse des Objektes. Bei der Skalierung gibt es eine große Box um – und eine kleine Box im Objekt. Bei der Skalierung ist es nur wichtig, dass eine der beiden Boxen anvisiert wird. Da diese Steuerungselemente nur einmal vorkommen, wurde ein einfaches *Back Buffer Picking* mit Hilfe des *Stencil Buffer* implementiert. Dadurch können die Bereiche pixelgenau



(a) (b) (c)

Abbildung 3.7: Die drei verschiedenen Manipulationsmethoden im Prototyp: (a) Rotation, (b) Translation und (c) Skalierung.

selektiert werden. Es wird nur der Pixel des *Stencil Buffers* ausgelesen, der sich in der Mitte des Fadenkreuzes befindet.

Mit einem Linksklick und Halten auf eines der Steuerelemente und einer Mausbewegung, wird die Funktion des Steuerelementes aktiviert. Das Objekt reagiert bei jeder Mausbewegung mit der getätigten Manipulation, sodass der Spieler sofort ein Feedback darüber erhält, was seine Aktionen bewirken. Mit dem wieder loslassen der Taste wird die Funktion beendet und das Objekt verbleibt in dem manipulierten Zustand. Alternativ kann bevor die Taste losgelassen wird, die rechte Maustaste betätigt werden, um das Objekt in den ursprünglichen Zustand zu versetzen und die Manipulation abzubrechen.

Für die Rotation wurde der ARCBALL aus Abschnitt 2.2.3 gewählt, da die Technik, laut [12], robuster und einfacher zu implementieren ist, als die alternativen *Virtual Trackball*-Techniken und keine signifikanten Unterschiede in den *Usability* Studien der *Virtual Trackballs* zu erkennen waren. Implementierungsdetails können [12] entnommen werden. Der ARCBALL sollte mit seinem Mittelpunkt im Objektmittelpunkt liegen, da um diesen Punkt gedreht werden soll, dies soll das Verständnis erleichtern. Abbildung 3.7 (a) kann entnommen werden, dass es zwei Bereiche gibt, die das Objekt um drei unterschiedliche Achsen rotieren lassen. Das Koordinatensystem der Kamera wird zum rotieren verwendet. Die Position der Rotation bleibt allerdings die Objektposition. Es spielt also für den Spieler eine Rolle, aus welcher Perspektive das Objekt angeschaut wird. Der äußere Bereich rotiert das Objekt um die Normale der Blickrichtung der Kamera. Der innere Bereich rotiert um die beiden übrigen Achsen. Bei einer horizontalen Bewegung in der vertikalen Mitte des Kreises wird das Objekt um die Achse der Kamera gedreht, die nach oben gerichtet ist (meist $\vec{up}$ genannt). Bei einer vertikalen Bewegung in der horizontalen Mitte des Kreises, wird um den seitliche Achse der Kamera gedreht (meist $\vec{side}$ genannt).

Die Translation wurde ähnlich der Translationstechnik aus *Maya* in Abschnitt 2.2.3 gewählt. Dabei wurde aber die freie Bewegung auf allen Achsen

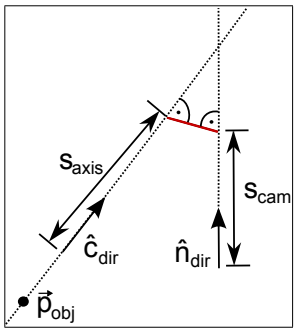


Abbildung 3.8: Der kürzeste Abstand zwischen zwei Linien und die dazugehörigen Variablen aus der Berechnung in Abschnitt 3.4.1.

weggelassen, um Tipp Nr. 1 zu folgen und zusätzliche Freiheitsgrade zu vermeiden. Daher wurden nur Bewegungen auf einzelnen Achsen zugelassen. In Abbildung 3.7 (b) können drei verschiedenfarbige Pfeile wahrgenommen werden, welche den drei Achsen des Objektes entsprechen. Da in der Literatur keine Implementierungsdetails verfügbar waren, wurde auf einfache mathematische Methoden zurückgegriffen und das *Back Buffer Picking* als Selektion für die Manipulationselemente verwendet.

Beim Linksklick auf einen der Pfeile wird von der Normalen des Sichtvektors $\hat{c}_{dir}$ der Kamera zur Normalen $\hat{n}_{axis}$ der ausgewählten Achse, die Skalare s_{cam} und s_{axis} errechnet, mit denen die beiden Normalen multipliziert werden müssen, um die beiden Punkte der beiden fortgeführten Linien der Normalen zu erhalten, bei denen der kürzeste Abstand zwischen den beiden Linien besteht (. Die Normalen werden dabei als Linien interpretiert, die durch die Kameraposition bzw. den Koordinatenursprung führen. Die Relationen sind in Abbildung 3.8 illustriert. Skalar s_{axis}, die benutzte Achse $\hat{n}_{axis}$ und die Position $\vec{p}_{obj}$ des Objektes müssen für die Dauer der Manipulation gespeichert werden. Bei jeder Bewegung des Spielers muss s_{axis} erneut berechnet werden und der alte Wert vom neuen Wert abgezogen werden, dadurch wird die Distanz $dist_{axis}$ berechnet, um die das Objekt von seiner ursprünglichen Position $\vec{p}_{obj}$, entlang der gespeicherten Achse $\hat{n}_{axis}$, verschoben werden muss. Beim Abbruch der Manipulation, wird die Position des Objekts mit der gespeicherten Position $\vec{p}_{obj}$ ersetzt und somit die Manipulation rückgängig gemacht.

Bei der Skalierung wird im Gegensatz zur Translation, auf allen Achsen gleichzeitig die Manipulation angewandt. Dies soll den Vorteil haben, die Objekte nicht asymmetrisch zu skalieren und so zu verzerren (auch *uniform scaling* genannt). Es wird von der Annahme ausgegangen, dass der Spieler nur diese Art der Skalierung benötigt. Abbildung 3.7 (c) können zwei transparente Würfel entnommen werden, die in das Objekt gezeichnet sind. Um die Skalierung zu beginnen, ist es nötig auf einen der beiden Würfel zu Kli-

cken. Die Berechnungen sehen ähnlich wie bei der Translation aus. Es wird der kürzeste Abstand *dist* zur Sichtlinie der Kamera berechnet. *dist* und die aktuelle Skalierung $scaling_{obj}$ des Objekts werden gespeichert. Bei jeder Bewegung des Spielers muss *dist* erneut berechnet werden und der neue Wert wird durch den alten Wert dividiert. Das Ergebnis ist ein Faktor der, mit $scaling_{obj}$ multipliziert, die neue Skalierung des Objekts liefert. Beim Abbruch der Manipulation, wird die Skalierung des Objekts mit dem gespeicherten $scaling_{obj}$-Wert ersetzt und somit die Manipulation rückgängig gemacht.

Steuerung

Es folgt eine Zusammenfassung der Steuerung im Weltmodus, wie sie im Prototyp implementiert wurde:

Maus

Linke Maustaste	Selektieren, Verschieben, Rotieren, Skalieren
Rechte Maustaste	Objekt erstellen
Mausrad vor	Manipulations-Methode ändern (vor)
Mausrad zurück	Manipulations-Methode ändern (zurück)

Aus der Zusammenfassung ist gut ersichtlich, welche funktionalen Anforderungen aus Abschnitt 3.1 der Weltmodus bisher erfüllt. Es können Objekte erzeugt werden, diese allerdings nicht mit zusätzlichen Polygonen erweitert werden. Die Objekte können selektiert und sehr unterschiedlich manipuliert werden. Allerdings werden dabei alle Polygone des Objektes damit selektiert und manipuliert. Feine Manipulationen an einzelnen Polygonen sind nicht möglich. Drei von vier Anforderungspunkten sind halb erfüllt, aber es mangelt bei den existierenden Funktionen an Feinheit, um einzelne Polygone selektieren, manipulieren und erstellen zu können. Im Prototyp wurde auf eine Löschfunktion für ganze Objekte verzichtet, um versehentliche Löschungen bei der späteren Evaluation zu vermeiden. Eine einfache Möglichkeit wäre es bei einem Objekt mit Highlight die Entf-Taste oder die rechte Maustaste zu betätigen. Dann wäre einer der vier Anforderungspunkte erfüllt. Im nächsten Abschnitt werden die fehlenden Anforderungspunkte erfüllt.

3.4.2 Objektmodus

Nachdem ein Objekt erstellt oder selektiert wurde, ist es möglich in den Objektmodus zu wechseln. In diesem Modus kann die Struktur des Objekts durch Polygone erweitert werden. Dabei wird die Tiefenebene aus Abschnitt 3.3.4 genutzt, um die Vertices der Polygone in die Szene zu setzen. Um Polygone zu erstellen, werden Verbindungen zwischen den Vertices gelegt.

(a) (b) (c) (d)

Abbildung 3.9: Ein Vertex in unterschiedlichen Selektionsphasen: (a) Ohne Highlight, (b) mit Highlight, (c) mit Highlight und selektiert, (d) nur selektiert.

Selektion

Die Technik, die für die Selektion von Vertices am besten erscheint, ist die *Cone Selection*, wie sie auch im Weltmodus in Abschnitt 3.4.1 angewandt wurde. Die Vertices sind klein, daher pixelgenau schwer zu selektieren und haben kein Volumen, wodurch nur die *Cone Selection* von den Selektionstechniken in Abschnitt 2.2.2 in Frage kommt. Es wird die Umsetzung aus Abschnitt 3.4.1 verwendet. Es wird über alle Vertices des selektierten Objektes iteriert und analoge Berechnungen durchgeführt, nur dass $\vec{c_i}$ nun die Vertexposition ist, anstatt dem Objektmittelpunkt.

Der Vertex den die Selektion vorschlägt wird auch hier mit einem Highlight versehen, sodass der Spieler weiß, dass dieser selektiert werden kann (hier sollte zwischen der Objektselektion im Weltmodus und der Vertexselektion im Objektmodus unterschieden werden). Abbildung 3.9 zeigt alle vier Phasen der Selektion. Ein nicht selektierter Vertex ist schwarz und ein Vertex mit Highlight, ist ein großer schwarzer Punkt. Dieser Größenunterschied soll die Aufmerksamkeit des Spielers verstärkt auf sich ziehen. Ein selektierter Vertex ist weiß, was den Kontrast zu den anderen Vertices verstärken soll, die nicht selektiert sind. Ein selektierter Vertex mit Highlight ist ein großer schwarzer Punkt, mit einem weißen Kern.

Es wird auch hier mit dem Linksklick selektiert und auch die Selektion aufgehoben, äquivalent zu den Objekten im Weltmodus. Die Maus sollte dabei nicht bewegt werden, ansonsten wird eine Manipulation ausgeführt. Im Gegensatz zum Selektionsprozess im Weltmodus, können im Objektmodus mehrere Selektionen gleichzeitig existieren, da dies notwendig ist, um die Verbindungen der Polygone zu setzen. Die alten Selektionen werden nicht durch neue ersetzt. Falls kein Highlight vorhanden ist, können alle Selektionen der Vertices durch einen Linksklick entfernt werden. Dadurch würde die Tiefenebene vorerst verschwinden und der Spieler kann sich einen Überblick über das Modell verschaffen.

Manipulation

Zur Manipulation der Vertices wird die Tiefenebene aus Abschnitt 3.3.4 genutzt, um die Tiefe festzulegen, in der die Manipulationen stattfinden und dies auch dem Spieler ersichtlich zu machen. Erstellte Vertices landen direkt in der Ebene und das Verschieben der Vertices findet auch in dieser statt. Dieser Ansatz richtet sich nach dem Erstellen einzelner Polygone in Abschnitt 2.2.1 beim Erweitern von Polygonen. Zur Erleichterung der Erstellung von aneinander geketteten Polygonen, wird nach dem Erstellen eines Vertex die Selektion angepasst, um sofort das nächste Polygon erstellen zu können.

Das Erstellen eines Vertex wird mit einem Rechtsklick durchgeführt. Dabei sollte das Fadenkreuz auf den sichtbaren Bereich der Tiefenebene zeigen. Falls das nicht der Fall ist, passiert nichts. Die Position des erstellten Vertex wird durch eine Schnittberechnung zwischen der Tiefenebene und der Normalen der Blickrichtung der Kamera bestimmt. Die Normale wird dabei als Linie interpretiert, die durch die Kameraposition führt, wie auch bei der Umsetzung der Translation und Rotation in Abschnitt 3.4.1. Die errechnete Position des Vertex soll als ungültig markiert werden, wenn der Abstand zwischen dem Zentrum der Tiefenebene und dem Vertex größer ist als der Abstand zwischen dem Zentrum der Tiefenebene und einem der Eckpunkte. Im Grunde also, wenn sich der Vertex nicht auf der sichtbaren Tiefenebene befindet (Eine Ebene ist im mathematischen Sinne immer unendlich). Es ist nicht vollkommen korrekt und lässt dem Spieler Spielraum beim Modellieren am Rand der Ebene, was in Spielen häufig angewandt wird, um Frust bei Fehlern zu vermeiden.

Beim Erstellen von geschlossenen Netzen aus Polygonen, ist es unabdingbar die Funktionalität von *Snapping* aus Abschnitt 2.2.3 zu nutzen. Nur dadurch kann garantiert werden, dass zwei aneinander liegende Polygone auch wirklich die gleichen Kanten haben und somit keine Spalten zwischen ihnen beim Modellieren entstehen. Außerdem soll das *Snapping* den Modellierungsprozess beschleunigen, da die endgültige Position eines Vertex schneller gefunden wird. Um Probleme mit dem *Snapping* zu vermeiden, falls zu viele Vertices im Blickfeld sind, kann dieses durch Drücken und Halten einer Taste deaktiviert werden. Dieses Problem wurde in der Vorstudie in Abschnitt 4.1 erkannt.

Abhängig von der Anzahl selektierter Vertices, werden unterschiedliche Verbindungen zwischen den selektierten Vertices und dem neuen Vertex hergestellt. Falls kein Vertex selektiert ist, wird keine Tiefenebene angezeigt und somit kann auch kein Vertex gesetzt werden. Bei einem selektierten Vertex, wird eine Linie zwischen dem selektierten und dem neuen Vertex gezogen (siehe Abbildung 3.10 (a,b)). Diese Linie erweitert die Struktur nicht, soll dem Spieler aber die Information vermitteln, dass zwischen den Vertices eine Verbindung existiert und das Polygon nicht aus dem Nichts erscheint,

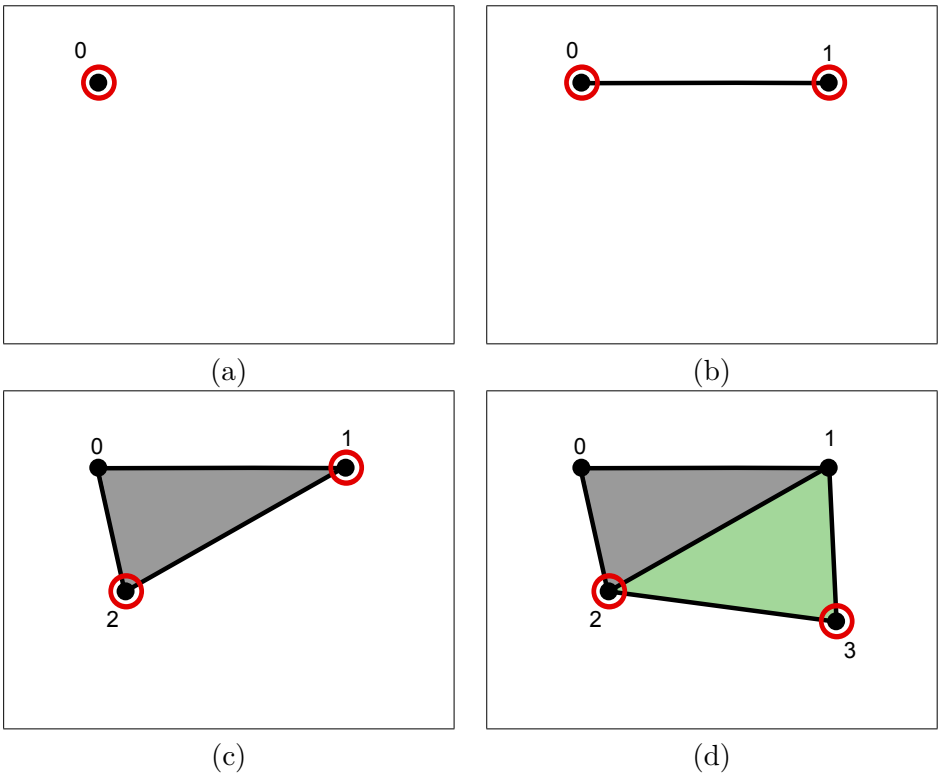


Abbildung 3.10: Die unterschiedlichen Phasen während der Modellierung von Polygonen: (a) Ausgangssituation mit einem selektierten Vertex, (b) nachdem ein Vertex erstellt wurde, (c) nachdem ein zweiter Vertex erstellt wurde und somit ein Polygon, (d) ein weiterer Vertex wurde gesetzt, da die Selektion passend ist. Während des Prozesses verändert sich automatisch die Selektion. Legende: Schwarzer Punkt – Vertex, roter Kreis – Selektion.

sondern aus der Kante zwischen den beiden Vertices. Nachdem die Kante erstellt wurde, wird der neue Vertex zur aktuellen Selektion hinzugefügt, damit als nächstes ein Polygon erstellt werden kann.

Bei zwei selektierten Vertices, wird zwischen diesen zwei Vertices und dem neuen Vertex ein Polygon erstellt, womit die Struktur des Objekts erweitert wird (siehe Abbildung 3.10 (b, c)). Nachdem das Polygon erstellt wurde, wird der neue Vertex zur Selektion hinzugefügt. Die Selektion, des zuerst selektierten Vertex, wird aufgehoben. Dies ermöglicht dem Spieler ohne zusätzlichen Aufwand, weitere Polygone zu setzen (siehe Abbildung 3.10 (c, d)). Der Spieler muss allerdings die ersten beiden Selektionen in korrekter Reihenfolge durchführen, sonst ist dieser „Selektionsvorschlag“ evtl. nicht nützlich. Diese Funktion wird im weiteren Verlauf der Arbeit „Vor-Selektion“ genannt.

Da Vertices selten beim Erstellen an ihrer endgültigen Position sind,

kann der Spieler die Vertices verschieben. Dazu wird der bekannte *Drag and Drop*-Ansatz benutzt. Mit einem Linksklick auf einen Vertex mit Highlight und Halten der Maustaste, kann der Spieler den Vertex in der Tiefenebene verschieben. Dabei wird dieselbe Kollisionsberechnung zwischen der Blickrichtung und der Tiefenebene durchgeführt, um die anvisierte Position auf der Ebene zu erlangen (siehe Oben). Während der Verschiebung kann wieder die *Snapping*-Funktion verwendet werden. Mit einem Rechtsklick während der Translation, kann die Manipulation abgebrochen werden und der Vertex wird an seine ursprüngliche Position, vor der Manipulation, zurückgesetzt. Dazu sollte die Position vor jeder Manipulation gespeichert werden. Mit dem Loslassen der linken Maustaste, wird die Manipulation beendet und der Vertex verbleibt an seiner neuen Position. Durch die Tiefenebene sollte der Spieler die Vertexposition auf der Ebene besser erahnen können. Vertices die vor der Manipulation nicht in der Ebene liegen, werden in die Ebene gelegt. Sie werden nicht parallel zur Ebene verschoben.

Um grobe Fehler zu beheben oder Objekte zu löschen, kann der Spieler Vertices löschen. Beim Löschen eines Vertex werden auch all seine Verbindungen zu anderen Vertices gelöscht. Dies führt zum Löschen existierender Polygone. Ein Vertex mit Highlight wird durch einen Rechtsklick gelöscht. Ein Objekt wird gelöscht, indem alle Vertices des Objekts gelöscht werden.

Steuerung

Es folgt eine Zusammenfassung der Steuerung im Objektmodus, wie sie im Prototyp implementiert wurde:

Maus

Linke Maustaste	Vertex selektieren, Vertex verschieben
Rechte Maustaste	Vertex erstellen, Vertex löschen
Mausrad vor	Ebenen-Modus ändern (vor)
Mausrad zurück	Ebenen-Modus ändern (zurück)

Tastatur

Leertaste	Deaktivierung des *Snapping*

Im Objektmodus können Manipulationen auf einzelnen Vertices angewandt werden, wodurch auch die Polygone manipuliert werden. Einzelne Vertices können erstellt, selektiert, manipuliert und gelöscht werden. Dies vervollständigt die Erfüllung der Anforderungen an ein Konzept, in Bezug auf die schon erfüllten Anforderungen im Weltmodus. Somit erfüllt Konzept 1 alle funktionalen Anforderungen, während bei der Entwicklung des Konzepts die qualitativen Anforderungen stets beachtet wurden.

3.5 Konzept 2

Im zweiten Konzept wird im Gegensatz zum ersten Konzept von einem vordefinierten Objekt ausgegangen (siehe Abschnitt 2.2.1), wie in den vorgestellten Modellierungsanwendungen, das vom Spieler erweitert werden kann. Dies soll dem Spieler eine Basis bieten, auf der aufgebaut werden kann und den Modellierungsprozess am Anfang beschleunigen.

Wechsel des Modellierungsmodus

Der Wechsel zwischen den Modi, wird in diesem Konzept vom Spieler direkt bestimmt. In der Vorstudie in Abschnitt 4.1 wurde erkannt, dass sich die Manipulationstechnik im Weltmodus, aus großen Entfernungen, schwierig bedienen lässt. Diese Entfernung zum Objekt ist in Konzept 1 nötig, um den Wechsel des Modus auszulösen. Daher wird in diesem Konzept der Wechsel durch betätigen einer Taste durchgeführt, um dieses Problem zu umgehen.

3.5.1 Weltmodus

Der Weltmodus ist, bis auf die Manipulationen, dem ersten Konzept sehr ähnlich und daher werden nur die Änderungen genannt. Beim Erstellen eines Objekts, wird statt nur einem Vertex, ein vordefiniertes Objekt erstellt.

Selektion

Die Selektion der Objekte funktioniert analog. Da in Konzept 1 bereits eine rudimentäre Selektion im Prototyp implementiert wurde, um den Fokus auf andere Funktionen zu legen, wurde dieser Teil übernommen. Auch hier wäre eine Selektion basierend auf *Back Buffer Picking* die gewünschte Technik gewesen.

Manipulation

Statt jede Manipulationstechnik in einem eigenen Modus anzuwenden, wird alternativ eine Technik genutzt, die alle drei Manipulationstechniken in einer kombiniert. Dies erspart den Wechsel zwischen den Manipulationsarten und damit auch die zu merkenden Steuerungstasten. Die *Bounding Box* des Objektes wird benutzt, um die Manipulationen durchzuführen. Diese ist ein Quader, der aus drei verschiedenen Formen zusammengesetzt werden kann: Die sechs Flächen des Quaders, die zwölf Eckkanten und die acht Eckpunkte (siehe Abbildung 3.11). Jede Form entspricht einer der drei Manipulationstechniken: Kante – Rotation, Fläche – Translation und Eckpunkt – Skalierung. Wie auch in Konzept 1, werden die Manipulationen per *Drag and Drop* mit der linken Maustaste benutzt. Die Manipulation kann mit einem Rechtsklick abgebrochen und rückgängig gemacht werden.

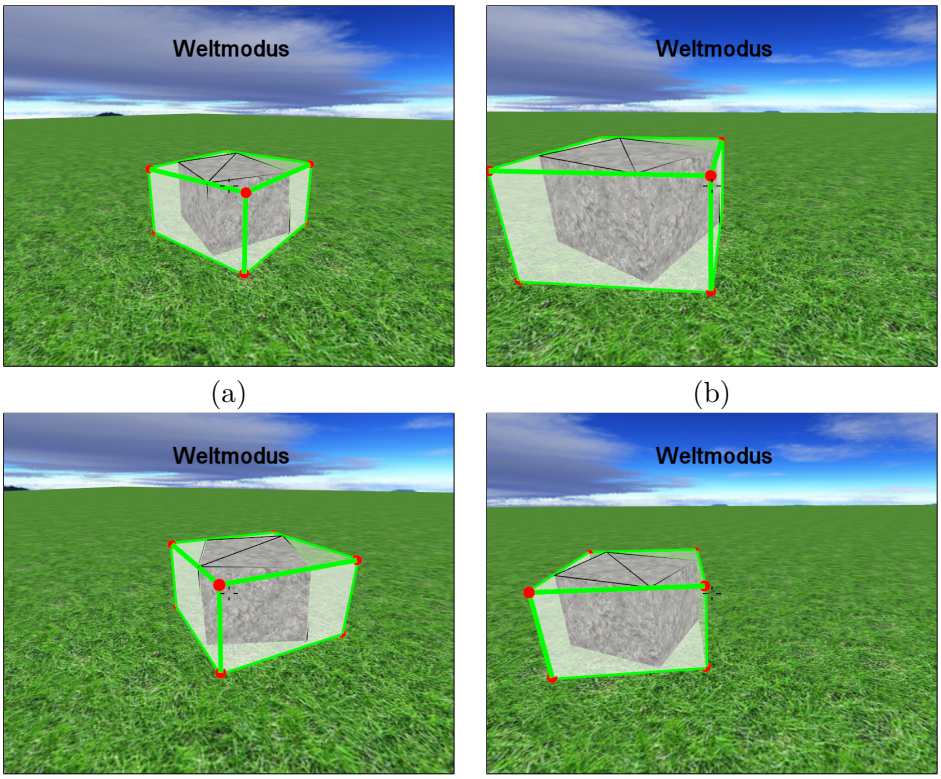


Abbildung 3.11: Die unterschiedlichen Manipulationsfunktionen des Weltmodus in Konzept 2: (a) Ausgangssituation mit einem selektierten Objekt, (b) nach einer Skalierung mit dem anvisierten Eckpunk, (c) nach einer Rotation mit der anvisierten Kante, (d) nach einer Translation mit der rechten sichtbaren Fläche. Alle drei Manipulationen gehen von der Ausgangssituation aus. Die Steuerungselemente der Manipulationen sind die Flächen (weiß und transparent), Kanten (grün) und Eckpunkte (rot) der *Bounding Box*.

Die Rotation ist eine beschränkte Rotation um eine festgelegte Achse, um Tipp Nr. 1 zu befolgen. Im Prototyp wurde die gleiche ARCBALL-Implementierung benutzt, wie in Konzept 1, aber mit beschränkter Rotation um eine vorher festgelegte Achse. Diese Erweiterung ist eine geringe Modifikation und kann [12] entnommen werden. Die Achse ist abhängig von der Kante, die für die Manipulation verwendet wird. Die Kante entspricht einem Vektor, welcher normalisiert, der Achse entspricht um die gedreht wird.

Die Translation basiert auf den selben Berechnungen, wie auch die Translation in Konzept 1. Die Normale $\hat{n}_{axis}$, die der Achse entspricht, entlang der das Objekt verschoben wird, entspricht in diesem Fall der Normalen der Fläche, die zum Manipulieren ausgewählt wird.

Die Skalierung folgt der, in Konzept 1 beschriebenen, nur dass sie erst

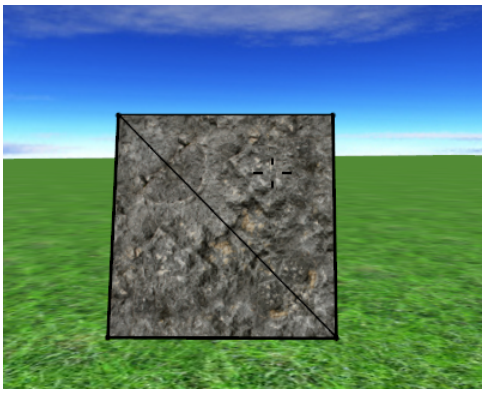


Abbildung 3.12: Das vordefinierte Objekt das im Weltmodus in Konzept 2 platziert wird. Es ist eine Fläche, bestehend aus zwei Polygonen.

aktiviert wird, wenn ein Eckpunkt zur Manipulation ausgewählt wird.

Beim Erstellen eines Objektes platziert der Spieler eine Fläche, die aus zwei Polygonen besteht (siehe Abbildung 3.12). In Abschnitt 2.2.1 werden zu den Objekten zusätzliche Werte angegeben, die aber den Erstellungsprozess verkomplizieren würde, daher wird nur eine einfache Fläche mit vordefinierten Werten erstellt. Eine Auswahl an vordefinierten Objekten konnte nicht mehr in den Prototypen implementiert werden, wäre aber die bessere Alternative gewesen.

Steuerung

Es folgt eine Zusammenfassung der Steuerung im Weltmodus, wie sie im Prototyp implementiert wurde:

Maus

Linke Maustaste	Objekt selektieren, verschieben, rotieren und skalieren
Rechte Maustaste	Objekt erstellen

Tastatur

Leertaste	Wechsel zum Objektmodus

Ähnlich wie in Konzept 1, sind die Anforderungen nur teilweise erfüllt, da die selben Funktionalitäten nur in einer anderen Form angeboten werden.

3.5.2 Objektmodus

Der Objektmodus ähnelt sehr dem Objektmodus aus Konzept 1, allerdings funktioniert das Erstellen der Polygone auf eine andere Weise. Es wird von

einem bestehenden Modell ausgegangen, da ein vordefiniertes Objekt benutzt wird. Aus diesem Grund wurde ein Ansatz entwickelt, bei dem neue Polygone, aus vorhandenen Kanten anderer Polygone, herausgezogen werden.

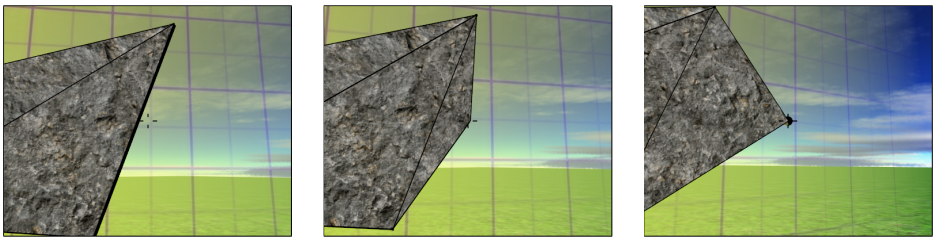
Selektion

Die Selektion von Vertices funktioniert analog zu Konzept 1. Hinzu kommt die Möglichkeit der Selektion von Kanten, bei der gleich beide Vertices der Kante selektiert werden. Die Reihenfolge in der die beiden Vertices zur Selektion hinzugefügt werden, ist abhängig vom Abstand zu diesen Punkten entlang der Kante. Generell wird der Vertex selektiert, der am nächsten zur Position des Fadenkreuzes auf der Kante ist. Die Reihenfolge ist wichtig, da sie darüber entscheidet, an welcher Position die Tiefenebene platziert wird. Im Gegensatz zum Konzept 1, wird die Selektion, wie im Weltmodus, nur ersetzt, wodurch nur noch maximal zwei Vertices gleichzeitig selektiert sein können. Dies soll das Anpassen der Tiefenebene beschleunigen. Bei der Erstellung von Polygonen in diesem Konzept wird die Selektion nicht mehr angepasst, um das Setzen des nächsten Polygons zu erleichtern. Dies ist nicht mehr nötig, da das Erstellen nicht mehr abhängig von den selektierten Vertices ist, sondern nur von einem Highlight einer Kante eines vorhandenen Polygons.

Die Selektion der Linien wurde im Prototyp nur rudimentär implementiert und basiert auch auf der *Cone Selection*, wie es bei der Vertex-Selektion schon der Fall ist. Die Kollisionsberechnung mit dem Kegel wird mit regelmäßig gesetzten Punkten entlang der Linie approximiert. Dies ist nicht korrekt, aber konnte im Prototyp schnell implementiert werden. Es funktioniert gut und wurde daher so belassen. Im Prinzip werden zu den vorhanden Vertices, noch zusätzliche Punkte aus den Kanten errechnet und diese bei der Berechnung des besten Punktes mit eingeschlossen. Falls der beste Punkt ein errechneter Punkt einer Kante ist, wird stattdessen die Kante ein Highlight erhalten.

Manipulation

Das Löschen und Verschieben von Vertices funktioniert genauso wie in Konzept 1. Diese Techniken sind bereits so einfach, dass keine anderen Techniken entwickelt werden konnten, die diesen Techniken Konkurrenz bieten könnten. Das Erstellen von Polygonen ist nicht mehr abhängig von der Selektion, sondern nur von der Kante, die ein Highlight hat (siehe Abbildung 3.13). Es wird wieder das *Drag and Drop*-Konzept benutzt. Die rechte Maustaste muss gedrückt und gehalten werden, während ein Kanten-Highlight aktiv ist. Dann kann die Maus bewegt werden und das neue Polygon erscheint schon während des Erstellungsprozesses und kann angepasst werden. Falls



(a) (b) (c)

Abbildung 3.13: Der Erstellungsprozess eines Polygons: (a) Kante hat ein Highlight, (b) Erstellungsprozess ist gestartet und der dritte Vertex wird positioniert, (c) Erstellungsprozess ist beendet und das Polygon ist erstellt.

der Erstellungsprozess ungewollt war oder es einen anderen Grund gibt, kann dieser durch einen Linksklick abgebrochen werden, und dadurch die Erstellung annulliert werden. Mit dem Loslassen der rechten Maustaste wird der Vorgang erfolgreich abgeschlossen und das Polygon wurde erstellt. Das *Snapping* ist auch in diesem Konzept beim Erstellen und Verschieben der Vertices verfügbar.

Steuerung

Es folgt eine Zusammenfassung der Steuerung im Objektmodus, wie sie im Prototyp implementiert wurde:

Maus

Linke Maustaste	Vertex selektieren, Vertex verschieben
Rechte Maustaste	Polygon erstellen, Vertex löschen
Mausrad vor	Ebenen-Modus ändern (vor)
Mausrad zurück	Ebenen-Modus ändern (zurück)

Tastatur

Leertaste	Wechsel zum Weltmodus

Das Selektieren, Manipulieren und Löschen von einzelnen Vertices wurde aus Konzept 1 übernommen. Der Erstellungsprozess hat sich gewandelt, allerdings erfüllt auch dieser die Anforderung, dass Polygone erstellt werden können. Die ausstehenden Anforderungen wurden somit erfüllt. Die qualitativen Anforderungen wurden auch bei diesem Konzept stets beachtet.

3.6 Fazit

Zwei Konzepte wurden entwickelt, welche die zu Anfang definierten Anforderungen erfüllen. Beide Konzepte unterscheiden sich signifikant, um diese

gegeneinander zu evaluieren und das Beste aus beiden zu vereinen. Die Modi sind so entwickelt worden, dass diese auswechselbar sind und die jeweils bessere Modi zum Schluss, zu einem Konzept verschmolzen werden kann. Während der Evaluation sind außerdem weitere Vorschläge der Versuchspersonen zu erwarten, die auch mit einbezogen werden.

Kapitel 4

Evaluation

Eine Evaluation beider Konzepte wurde mit acht Teilnehmern durchgeführt. Die Teilnehmer mussten Aufgaben bewältigen und bewerteten im Anschluss mit Fragebögen, die jeweilig während der Aufgaben eingesetzten Techniken. Die Vergleichsdaten bestehen aus den Nutzungsdaten, die während der Experimente gesammelt werden und den Antworten und Bewertungen der Teilnehmer aus den Fragebögen. Anhand dieser Daten wird das bessere Konzept bestimmt und in den folgenden Kapiteln ein Ausblick über weitere Schritte gegeben, die gegangen werden können, um die Konzepte weiterzuentwickeln. Die leitende Person für die Benutzerstudie ist ein Experte für beide Konzepte, weist die Teilnehmer ein und notiert sich besondere Ereignisse und Meinung der Teilnehmer während dessen.

4.1 Vorstudie

Bevor die Benutzerstudie begonnen wurde, die in Abschnitt 4.4 beschrieben ist, wurde eine Vorstudie mit zwei Teilnehmern durchgeführt, die dieser sehr ähnlich ist. Diese Vorstudie diente dazu, den Verlauf der Benutzerstudie zu testen und Defizite in der Planung zu eliminieren. Dabei wurde erkannt, dass die Reihenfolge der Aufgaben und der Zeitpunkt der Beantwortung der Fragebögen Probleme erzeugten. Die Teilnehmer verwechselten die zu bewertenden Techniken. Um dem entgegenzuwirken, wurden die Fragebögen in der Benutzerstudie schrittweise ausgefüllt. Nach jeder getesteten Technik, wurde diese im Anschluss bewertet.

Außerdem wurden zwei Defizite in den beiden Konzepten erkannt, welche für die Benutzerstudie korrigiert wurden. Der Mangel an einer Funktion zur Deaktivierung des *Snapping*. Und ein spielergesteuerter Wechsel zwischen den beiden Modi, der Konzepte. Zusätzlich wurde eine höher aufgelöste Textur für die Tiefenebene verwendet, um die Linien des Rasters schmaler zu machen und so präzisere Manipulationen zu ermöglichen. Diese Korrekturen wurden in Abschnitt 3.3.4, Abschnitt 3.4.2 und Abschnitt 3.5 erklärt und

gekennzeichnet.

4.2 Teilnehmer

Über alle acht Teilnehmer wurden Daten über bisherige Erfahrungen mit Videospielen und der Modellierung erhoben, da dies die Ergebnisse beeinflussen kann. Dieser Einfluss sollte erkannt werden, um die Gründe für Probleme und Vorteile erkennen zu können. Diese Gründe können in der Zukunft bei der Entwicklung weiterer Konzepte berücksichtigt werden.

Zwei Teilnehmer sind zwischen 19 und 24 und sechs zwischen 25 und 30 Jahren alt. Alle acht spielen Videospiele am Desktop-Computer oder Notebook, vier auch an Konsolen und drei an Mobiltelefonen. Keiner spielt an Tablets oder sonstigen Geräten. Drei Teilnehmer spielen täglich, vier wöchentlich und einer zumindest monatlich. Drei haben Erfahrung mit Blender, einer mit Google Sketchup, einer mit der CAD-Anwendung Visi und drei haben gar keine Erfahrung mit Modellierungsanwendungen. Alle sind Studenten. Davon sind drei weiblich und fünf männlich. Diese Daten wurden mit der erste Seite des Formulares erhoben (siehe Anhang A).

4.3 Testsystem

Das Testsystem auf dem die Benutzerstudie durchgeführt wurde, ist ein *Compal HL90* Notebook mit einer *Logitech M100* Maus. Der Prototyp wurde auf die Hardwarespezifikationen dieses Modells angepasst, da er darauf entwickelt wurde. Folgende Spezifikationen hat das Notebook:

- OS: Windows 7 Professional 64 Bit,
- CPU: Intel® Core™2 Duo Processor P8600 (2x2,4 GHz),
- RAM: 4 GB DDR2,
- GPU: NVIDIA® GeForce® 9600M GT (OpenGL 3.3),
- Display: 15.4 Zoll, WXGA Glare Type (1280x800 Pixel).

Der entwickelte Prototyp beinhaltet beide implementierten Konzepte. Die Anwendung wurde mit einer Auflösung von 800×600 Pixel gestartet und in der Mitte des Bildschirms platziert.

4.4 Benutzerstudie

Die Studie besteht aus dem Test der Konzepte durch die Teilnehmer, der Bewertung derselben und den aufgezeichneten Nutzungsdaten während des Tests. Der Test besteht aus einigen Aufgaben, die die Teilnehmer mit beiden Konzepten durchführen.

4.4.1 Ziel

In Kapitel 3 wurden zwei Konzepte entwickelt, welche in dieser Studie evaluiert werden sollen. In diesem Kapitel wird das bessere von beiden, anhand der Evaluationsergebnisse bestimmt, um dieses Ziel zu erreichen. Zusätzliche Verbesserungsvorschläge wurden von den Teilnehmern erfragt, um einen Überblick zu verschaffen, ob wichtige Funktionen fehlen.

4.4.2 Ablauf

Dies ist ein vollständiger Verlauf der Benutzerstudie mit einem Teilnehmer, welcher in zwei Durchläufe unterteilt ist:

1. Nutzerdaten im Formular ausfüllen,
2. Einführung in das Konzept im Prototyp,
3. Erledigen der Aufgaben eins bis drei im Prototyp,
4. Bewerten des Objektmodus im Formular,
5. Erledigen der Aufgaben vier und fünf im Prototyp,
6. Bewerten des Weltmodus im Formular (Ende des ersten Durchlaufs),
7. kurze Pause,
8. Wiederholung von Schritt 2 bis 6 mit anderem Konzept (zweiter Durchlauf).

Beide Konzepte werden evaluiert. Wenn aber immer das gleiche Konzept als erstes evaluiert wird, nutzen die Teilnehmer ihre Erfahrungen im zweiten Durchlauf, was zu statistischen Werten, mit unterschiedlichen Ausgangssituationen, zwischen den beiden Konzepten führen würde. Dies verfälscht das Ergebnis. Daher wird das erste Konzept im Verlauf abgewechselt. Bei der Auswertung der Daten, werden in Kapitel 5 die drei Unterscheidungen zwischen Sammlungen von Durchläufen mit bestimmten Konzepten gemacht: Die gesamten Daten, Daten aus dem ersten Durchlauf und Daten aus dem zweiten Durchlauf. Die gesamten Daten entsprechen beiden Datensammlungen aus beiden Durchläufen. Daten aus dem ersten Durchlauf, entsprechen den Daten des ersten Konzeptes, das von einem Teilnehmer im Ablauf getestet wurde. Dies verringert die Datenmenge auf vier pro Konzept. Daten aus dem zweiten Durchlauf, entsprechen den Daten des zweiten Konzeptes, das von einem Teilnehmer im Verlauf getestet wurde. Auch hier ist die Datenmenge pro Konzept halbiert.

Die Aufgaben werden aufgeteilt, da ansonsten Verwechslungsgefahr bei der Bewertung besteht, was in der Vorstudie erkannt wurde. Die Bewertung wird daher direkt nach einer Aufgabenreihe durchgeführt, zu der die Bewertung gehört. Zwischen den beiden Konzepten ist eine kurze Pause nötig, da in der Vorstudie der gesamte Durchlauf fast zwei Stunden dauerte. Dies soll die Konzentration aufrechterhalten. Während dem Ausführen der Aufgaben

werden Notizen über das Verhalten der Teilnehmer gemacht. Dazu zählen häufige Probleme, genutzte Funktionen und Vorgehensweise der Teilnehmer.

4.4.3 Einführung

Den Teilnehmern wurde das jeweilige Konzept erklärt, während sie das Erklärte sofort ausprobieren konnten. Es wurde darauf geachtet, dass alle Funktionen des jeweiligen Konzepts und die konzept-unabhängigen Eigenschaften in Abschnitt 3.3 einige Male angewendet und verstanden wurden. Die Funktionen wurden nicht in derselben Tiefe, wie in Kapitel 3 erklärt, aber in genügender Tiefe, um sie verwenden zu können. Die möglichen Ausrichtungen der Tiefenebene wurden besonders sorgfältig erklärt. Um das Gelernte zu prüfen, sollte versucht werden die erste Aufgabe zu erledigen. Nachdem erkannt wurde, dass die Teilnehmer mit den Funktionen umgehen können, wurde der Prototyp neugestartet und mit Schritt drei im Ablauf begonnen.

4.4.4 Aufgaben

Fünf Aufgaben wurden pro Konzept erledigt, davon drei im Objektmodus und zwei im Weltmodus. Der Fokus der Aufgaben liegt im Objektmodus, da dieser der innovative Teil dieser Arbeit ist und auch der Teil, in dem der Spieler die meiste seiner Zeit verbringen würde. Die Aufgaben eins bis drei werden im Objektmodus durchgeführt und benötigten in der Vorstudie etwa 90 Prozent der Zeit. Zu diesen Aufgaben wird den Teilnehmern die Abbildung 4.1 gezeigt, damit sich alle an diesen Objekten orientieren können und ähnliche Objekte bauen. Die Aufgaben vier und fünf finden im Weltmodus statt und benötigten in der Vorstudie nur einige Minuten. Für die ersten drei Aufgaben wurde die Zeit gemessen, die zum Erfüllen der Aufgabe gebraucht wurde. Dies soll ein zusätzlicher statistischer Wert sein, um die Konzepte zu vergleichen. Die Aufgaben vier und fünf wurden gemeinsam gemessen. Die Rotation wurde nicht in den Aufgaben benutzt, da in Konzept 1 im Prototyp die Rotation nicht voll funktionsfähig ist.

Aufgabe 1

Die erste Aufgabe besteht aus der Modellierung einer Box. Das zu modellierende Objekt ist in Abbildung 4.1 (a) gezeigt. Ob Quader oder Würfel ist nicht signifikant, aber die Winkel der Eckkanten sollten ungefähr 90 Grad betragen. Dies kann mit der Ausrichtung der Tiefenebene mit zwei selektierten Vertices erreicht werden, welche damit auch getestet werden soll. Außerdem kann das Raster auf der Tiefenebene verwendet werden, um die Eckpunkte der Flächen zu setzen. Daran kann erkannt werden, wie wichtig und hilfreich den Teilnehmern das Raster ist. Zusätzlich kann die horizontale Ausrichtung der Tiefenebene bei nur einem selektierten Vertex für die

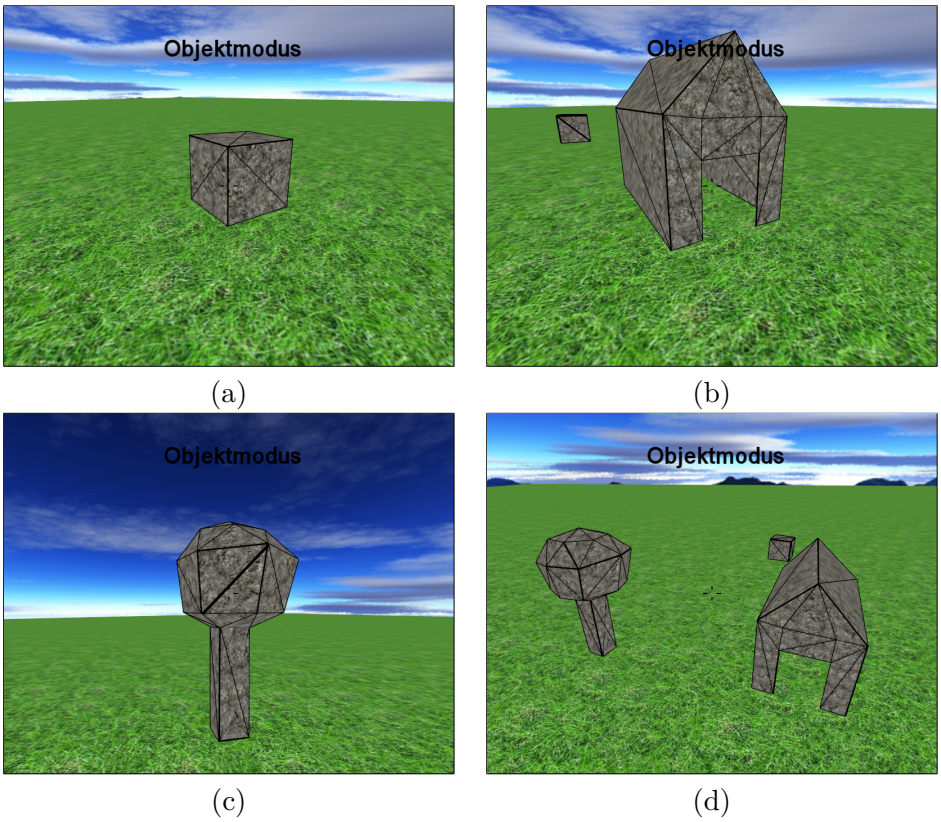


Abbildung 4.1: Die vorgegebenen Objekte zu den Aufgaben eins bis drei: (a) rechteckige Box, (b) Haus mit Türloch und Satteldach, (c) Baum mit vereinfachter Krone, (d) alle drei Objekte zusammen.

Unter- und Oberseite genutzt werden. Mit dem Raster auf dieser Ausrichtung, ist es besonders einfach das Fundament für eine rechtwinklige Box zu setzen.

Aufgabe 2

Das Ziel ist ein einfaches Haus mit vier Wänden, einem Türloch und einem Satteldach[1]. Das zu modellierende Objekt ist in Abbildung 4.1 (b) gezeigt. Ein Boden ist nicht erwünscht, da die Teilnehmer ohne die horizontal ausgerichtete Tiefenebene bei nur einem selektierten Vertex auskommen sollen. Das Haus ist ein Objekt, das sehr häufig von den Spielern in *Minecraft* gebaut wird. Daher sollte dieser Modellierungsprozess evaluiert werden. Dies wurde durch Begutachtung verschiedener virtueller *Minecraft*-Welten erkannt.

[1] Auch Giebeldach genannt.

Aufgabe 3

Ein baumähnliches Objekt soll in dieser Aufgabe modelliert werden. Ein viereckiger Stamm und eine kugelförmige Baumkrone, sind die Vorgaben. Dies ist ein komplexes Objekt, das außerdem in der Krone nicht mit der herkömmlichen Vorgehensweise modelliert werden kann. Die rechtwinkligen Beziehungen im Objekt sind, zumindest in der Baumkrone, nicht mehr gegeben und müssen durch eine Kombination verschiedener Ausrichtungen der Tiefenebene gelöst werden. Teilnehmer und Konzept sollen durch die komplexere Vorgehensweise besonders beansprucht werden, um die qualitativen Anforderungen zu testen.

Aufgabe 4

Der zuvor erstellte Baum soll auf die Box gestellt und verkleinert werden. Dies soll die Verschiebung und Skalierung im Allgemeinen testen. Da keine komplexen Szene zur Verfügung stehen, können nur einfache Manipulationsaufgaben getestet werden. Dies ist akzeptabel, da der Fokus auf dem Objektmodus liegt.

Aufgabe 5

Das Haus soll um die beiden anderen Objekte herum gestellt werden. Anstatt die beiden Objekte ins Haus zu stellen, ist es einfacher das Haus drumherum zu stellen. Dies wurde in der Vorstudie erkannt. Durch die Sichtbehinderung des Hauses, können die Objekte nur schwierig darin Manipuliert werden. Dies scheint ein allgemeines Problem zu sein und nicht konzeptabhängig.

4.4.5 Daten

Neben den Fragebögen, die ausgefüllt wurden, wurden auch Daten mit dem Prototyp aufgezeichnet und die Zeiten gemessen, die zum erfüllen der Aufgaben benötigt wurden.

Fragebogen

Die Nutzerdaten in Abschnitt 4.2 wurden mit dem Fragebogen in Anhang A, zu Beginn der Nutzerstudie erhoben. Diese Daten ermöglichten es dem Leiter der Benutzerstudie, sich an die Erfahrung der Teilnehmer anzupassen. Konzepte konnten einfacher erklärt werden, da auf ein gemeinsames Basiswissen zurückgegriffen werden könnte, wenn die Teilnehmer Erfahrung mit der Modellierung hatten.

Der restliche Fragebogen besteht aus den Bewertungen der beiden Konzepte. Die Bewertung pro Konzept ist in zwei Teile geteilt. Einen Teil für den Objektmodus, der nach Aufgabe 3 bewertet wird. Einen weiteren Teil für

den Weltmodus, nach Aufgabe 5 bewertet wird. Beide Modi werden getrennt bewertet, da sie sehr unterschiedlich sind. Dies hat den Vorteil, dass ein guter Modus, nicht von einem schlechteren Modus negativ beeinflusst wird. Sie sind austauschbar, somit kann das bessere Paar am Ende als Ergebnis präsentiert werden.

Für beide Modi wird die Benutzerfreundlichkeit und Intuitivität bewertet. Beide Eigenschaften sollen die Schwierigkeit und die Erlernbarkeit der Modi widerspiegeln und bieten eine Möglichkeit die qualitativen Anforderungen in Abschnitt 3.1 zu untersuchen. Außerdem bewertet der Teilnehmer, die eigene Zufriedenheit mit den Objekten, die modelliert wurden. Dies ist ein wichtiges Indiz für einen guten Modus und gibt Rückschluss darüber, ob die Teilnehmer den Modus wieder verwenden würden. Da beide Konzepte nicht für eine produktive Modellierungsanwendung, sondern für ein Videospiel entwickelt werden, wird von der Annahme ausgegangen, dass der Spieler kein Spiel spielt, mit dem er nicht zufrieden ist.

Eine zusätzliche Bewertung der Tiefenebene ist nur für den Objektmodus vorgesehen. Die Teilnehmer wurden aufgefordert zu bewerten, wie gut sie mit der Tiefenebene zurecht kamen. Da die Ebene ein fundamentaler Teil beider Konzepte ist, wurde sie gesondert bewertet. Damit soll nicht nur die allgemeine Idee der Tiefenebene, als Ersatz für die vier verschiedenen Ansichten in den Modellierungsanwendungen, bewertet werden, sondern auch die Nützlichkeit der möglichen Ausrichtungen der Tiefenebene.

Alle Bewertungen sind mit sieben Bewertungsfeldern versehen, die eine Bewertung von eins bis sieben ermöglichen. Eins wurde mit „sehr gut“, vier mit „ok“ und sieben mit „sehr schlecht“ beschriftet. Die anderen Felder hatten keine Beschriftung.

Zwei Bereiche für Probleme und Verbesserungsvorschläge wurden bereitgestellt, die den Teilnehmern ermöglichten besondere Ereignisse, die während der Aufgaben passierten und eigene Ideen zu notieren. Diese Bereiche sind besonders wichtig für die Entwicklung zukünftiger Konzepte und die Ergebnisse dieser Bereiche werden unter anderem in Kapitel 6 diskutiert.

Nutzungsdaten

Mit dem Prototyp wurden Daten während der Benutzung gespeichert. Diese sollen den Aufwand eines Spielers widerspiegeln. Mit diesen Daten soll ein statistischer Unterschied bei der Benutzung beider Konzepte untersucht werden. Es werden Daten im Objektmodus, Weltmodus und unabhängig vom Modus gesammelt.

Im Objektmodus wurden folgende Modellierungsaktionen gezählt:

- Erstellen eines Vertex,
- Löschen eines Vertex,
- Verschieben eines Vertex (auch beim Erstellen in Konzept 2),

- Abbruch einer Verschiebung eines Vertex,
- Verschiebungsdistanz des verschobenen Vertex (aufsummiert),
- *Snapping* eines Vertex in einen anderen,
- Selektieren eines Vertex (pro Vertex),
- Aufheben einer Selektion eines Vertex,
- Aufheben aller Selektionen.

Die Distanz der Verschiebung eines Vertex wurde nicht gezählt, sondern alle Werte aufsummiert.

Im Weltmodus wurden die folgenden Manipulationsaktionen gezählt:

- Verschiebung eines Objektes,
- Verschiebungsdistanz des verschobenen Objektes (aufsummiert),
- Abbruch einer Verschiebung eines Objektes,
- Skalierung eines Objektes,
- Skalierungsänderung des skalierten Objektes (aufsummiert),
- Abbruch einer Skalierung eines Objektes.

Die Verschiebungsdistanz und die Skalierungsänderung wurden beide nicht gezählt, sondern wie auch im Objektmodus alle Werte aufsummiert.

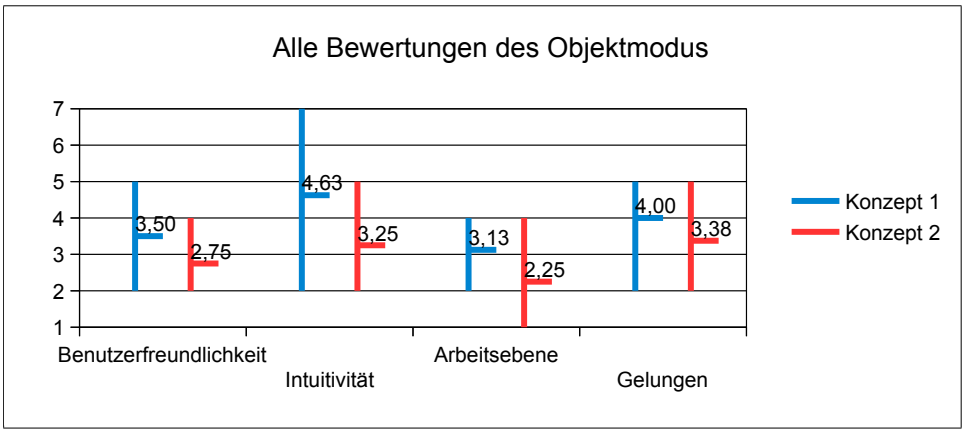
Unabhängig vom aktiven Modus, wurden die direkten Spielerbewegungen gespeichert – die zurückgelegte Distanz der Kamera und dessen Winkeländerungen.

Zeiten

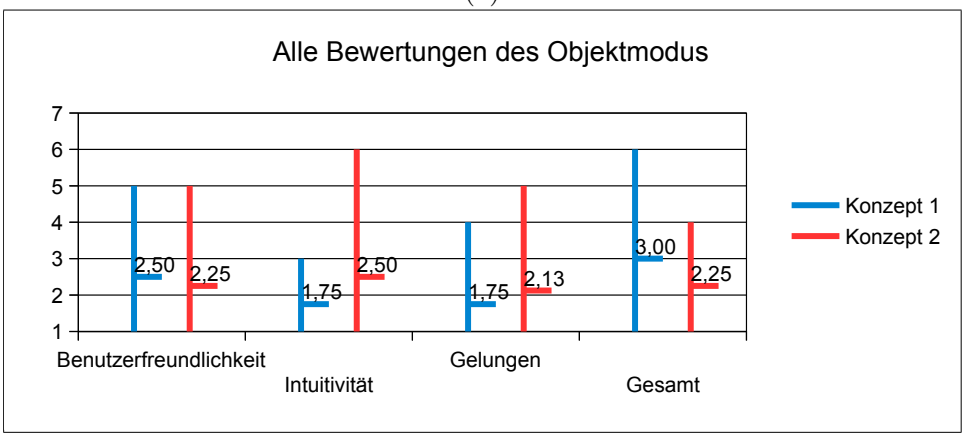
Für jede Aufgabe wurde die Zeit gemessen, die zur Erfüllung dieser benötigt wurde. Mit Ausnahme von Aufgabe vier und fünf, welche zusammen gemessen wurden.

4.5 Ergebnisse

Alle Werte sind als Durchschnittswerte über alle Teilnehmer angegeben, die an der Benutzerstudie teilgenommen haben. Während bei den Schritten zwei bis sechs beim ersten Durchlauf Erfahrung gesammelt wurde, ist der zweite Durchlauf mit dem anderen Konzept nicht mehr unter den selben Bedingungen durchgeführt worden. Diese zusätzliche Erfahrung kann aber dazu genutzt werden, um zu erkennen, wie schnell die Konzepte erlernt werden und ob überhaupt Leistungssteigerungen eintreten. Aus diesem Grund werden neben den gesamten Daten auch die Daten präsentiert, die nur im ersten Durchlauf und die im zweiten Durchlauf ermittelt wurden. Diese Daten wurden allerdings aus einer halb so großen Datenmenge errechnet, da nur jeweils 4 Personen pro Konzept diese Anforderungen erfüllen. Die Konzepte wurden im Vorfeld abwechselnd als erste und zweite Konzepte eingesetzt, wie in Abschnitt 4.4.2 schon erklärt wurde und zu dieser Reduzierung führte.



(a)

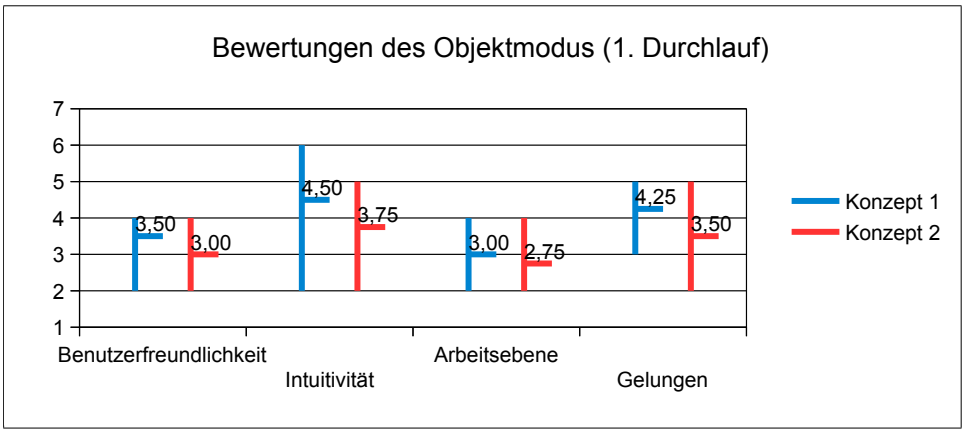


(b)

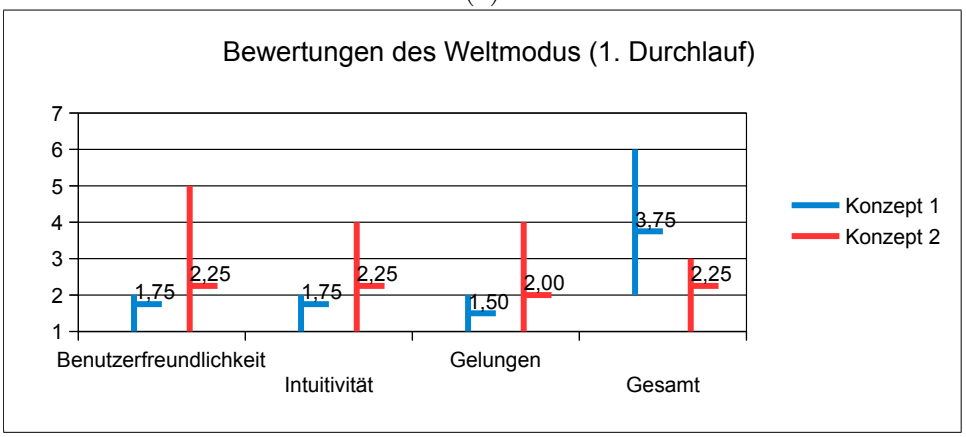
Abbildung 4.2: Die Bewertungen aus beiden Durchläufen der beiden Konzepte der Teilnehmer. (a) Bewertung des Objektmodus. (b) Bewertung des Weltmodus und Gesamtwertung beider Modi. Ein kleinerer Wert ist besser. Eine eins bedeutet „sehr gut“, eine vier bedeutet „ok“ und eine 7 bedeutet „sehr schlecht“.

4.5.1 Bewertungen

In Abbildung 4.2 werden die gesamten Bewertungen, in Abbildung 4.3 die Bewertungen für den ersten Durchlauf und in Abbildung 4.4 die Bewertungen für den zweiten Durchlauf mit dem Fragebogen gezeigt. Alle abgefragten Eigenschaften wurden in Abschnitt 4.4.5 mit dem selben Wortlaut als Wertbeschriftung erklärt.



(a)



(b)

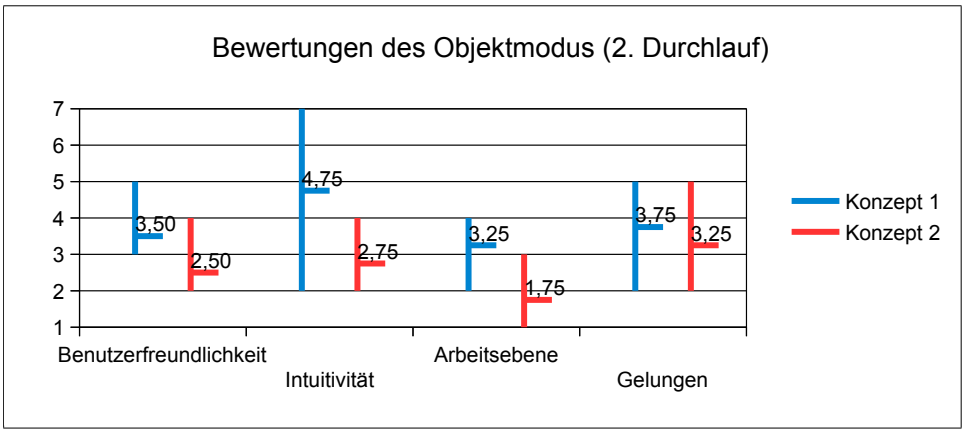
Abbildung 4.3: Die Bewertungen aus den ersten Durchläufen der beiden Konzepte der Teilnehmer. (a) Bewertung des Objektmodus. (b) Bewertung des Weltmodus und Gesamtwertung beider Modi. Ein kleinerer Wert ist besser. Eine eins bedeutet „sehr gut“, eine vier bedeutet „ok“ und eine 7 bedeutet „sehr schlecht“.

4.5.2 Feedback

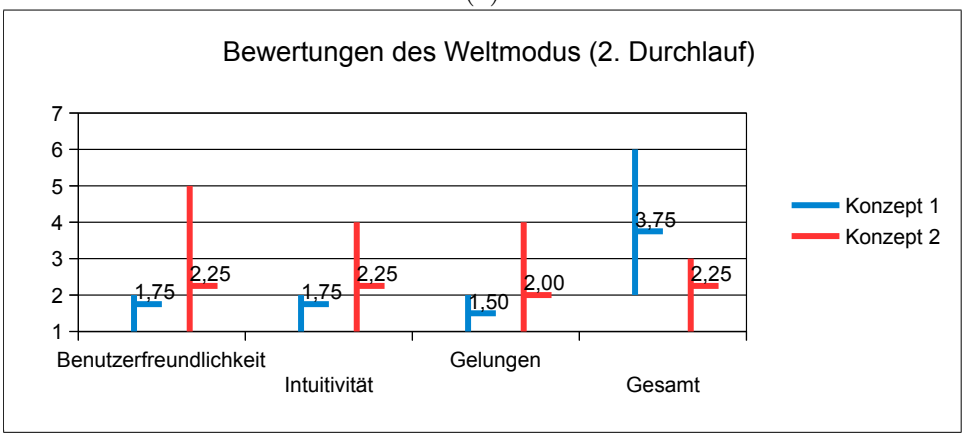
Neben den Bewertungen konnten Probleme und Verbesserungsvorschläge im Formular angegeben werden. Diese Daten und deren Häufigkeit (in Klammern) werden hier aufgelistet. Wenn keine Häufigkeit angegeben ist, ist diese mit eins zu werten.

Konzept 1

Folgende Probleme wurden von den Teilnehmern für den Objektmodus in Konzept 1 angemerkt:



(a)



(b)

Abbildung 4.4: Die Bewertungen aus den zweiten Durchläufen der beiden Konzepte der Teilnehmer. (a) Bewertung des Objektmodus. (b) Bewertung des Weltmodus und Gesamtwertung beider Modi. Ein kleinerer Wert ist besser. Eine eins bedeutet „sehr gut“, eine vier bedeutet „ok“ und eine 7 bedeutet „sehr schlecht“.

- Ungewolltes *Snapping* (4 Mal),
- Löschen und Erstellen von Vertices auf der selben Maustaste,
- manchmal zu großer Abstand, in dem der Moduswechsel stattfindet.

Zu einigen Problemen gab es Verbesserungsvorschläge, aber auch ohne ein direktes Problem anzugeben, gab es Ideen für Verbesserungen für den Objektmodus in Konzept 1:

- Eine Funktion zum Rückgängig machen der letzten Aktion, so wie die Strg-Z-Funktion in den meisten Anwendungen (5 Mal),
- andere Tasten für Höhenänderung der Kamera (2 Mal),

- *Snapping* per Taste aktivieren, anstatt es zu deaktivieren (2 Mal).

Probleme für den Weltmodus:

- Die Steuerungselemente zum Verschieben sind zu groß, dadurch ist das Objekt nicht mehr sichtbar (2 Mal),
- Mindestabstand zum Objekt ist einschränkend.

Und Verbesserungsvorschläge für den Weltmodus:

- Steuerungselemente zum Verschieben nicht zu klein machen,
- Steuerungselemente zum Verschieben besser greifbar machen, evtl. dickere Linien.

Konzept 2

Folgende Probleme wurden für den Objektmodus in Konzept 2 angegeben:

- Highlight durch Ebene manchmal schwierig zu erkennen,
- ungewolltes *Snapping*.

Und Verbesserungsvorschläge für den Objektmodus:

- Eine Funktion zum Rückgängig machen der letzten Aktion, so wie die Strg-Z-Funktion in den meisten Anwendungen (3 Mal),
- Ebene sollte horizontal auf dem Boden starten,
- Setzen einzelner Vertices ermöglichen,
- aus mehreren Linien Polygone herausziehen können.

Folgende Probleme wurden für den Weltmodus in Konzept 2 angegeben:

- Vordefiniertes Objekt ist immer schief,
- Eckpunkte schwer zu treffen beim Skalieren.

Und Verbesserungsvorschläge für den Weltmodus:

- Gleichzeitige Manipulation von mehreren Objekten (2 Mal),
- Verschieben der Objekte in allen drei Achsen gleichzeitig ermöglichen,
- Eckpunkte der *Bounding Box* größer machen,
- anstatt entlang der Normalen der Fläche zu verschieben, in der Ebene der Normalen verschieben,
- Objekt-*Snapping* ermöglichen.

Konzeptunabhängig

Einige der angemerkten Probleme und Vorschläge haben sich nicht direkt auf das jeweilige Konzept bezogen, sondern auf eine Funktion, die von beiden Konzepten angeboten wird. Dass ein Problem oder ein Vorschlag nicht in beiden Konzepten angemerkt wurde, könnte verschiedene Gründe haben. Unter anderem, reine Vergesslichkeit oder aber der Gedanke, dass schon darauf aufmerksam gemacht wurde und nochmaliges Wiederholen nicht nötig

sei. Diese Probleme und Vorschläge werden in diesem Abschnitt präsentiert. Folgende Probleme wurde angemerkt:

- Ungewolltes *Snapping* (5 Mal),
- Löschen und Erstellen von Vertices auf der selben Maustaste,
- Highlight durch Ebene manchmal schwierig zu erkennen.

Und folgende Verbesserungsvoschläge:

- Eine Funktion zum Rückgängig machen der letzten Aktion, so wie die Strg-Z-Funktion in den meisten Anwendungen (8 Mal),
- Löschen von Dreiecken ermöglichen (4 Mal),
- Drehen in 90 Grad-Schritten (3 Mal),
- *Snapping* per Taste aktivieren (3 Mal),
- andere Tasten für Höhenänderung der Kamera (2 Mal),
- *Snapping* an Rasterlinien ermöglichen (2 Mal),
- *Snapping* per Taste aktivieren, anstatt es zu deaktivieren (2 Mal),
- Erstellen per Linksklick (2 Mal),
- Belegung der Maustasten wählen können (2 Mal),
- die Selektion ist schwierig zu erkennen,
- global ausgerichtete Tiefenebene, orientiert an globalen Achsen,
- schnellere Bewegung per Taste aktivieren,
- andere Texturen für Objekte,
- Schattenberechnung für die Szene einfügen,
- Löschen von Linien ermöglichen,
- Kopieren von Objekten ermöglichen,
- Verschmelzen von Objekten ermöglichen,
- andere Farbe für Highlight nutzen.

4.5.3 Benutzungsdaten

In Abbildung 4.5 werden die gesamten Benutzungsdaten, in Abbildung 4.6 die Benutzungsdaten für den ersten Durchlauf und in Abbildung 4.7 die Benutzungsdaten für den zweiten Durchlauf, die während der Aufgaben aufgezeichnet wurden, gezeigt. Die Benutzungsdaten sind in der gleichen Reihenfolge gezeigt, wie sie in Abschnitt 4.4.5 aufgeführt sind. Generell ist ein kleinerer Wert vorteilhafter, da dies weniger Aufwand für die Teilnehmer, bei gleichem Ergebnis bedeuten würde.

Die Werte für die Rotation wurden ausgelassen, da diese aus den Aufgaben entfernt wurde. Außerdem ist die Zählung des *Snapping* fehlerhaft und kann nicht für die Analyse in Kapitel 5 genutzt werden.

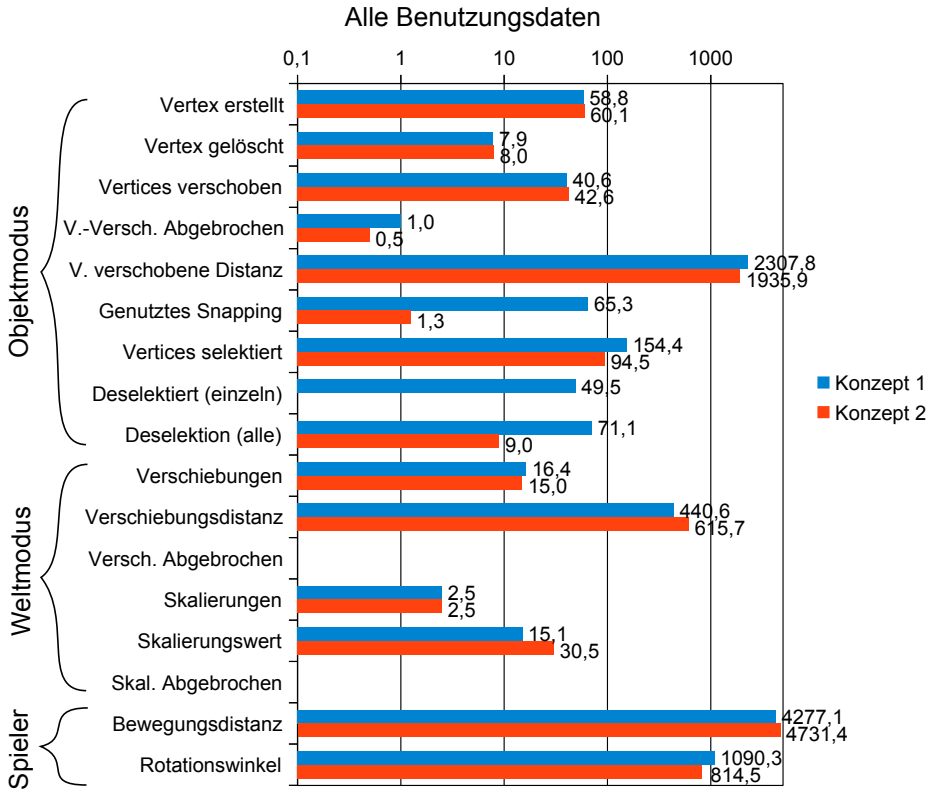


Abbildung 4.5: Die gesamten aufgezeichneten Benutzungsdaten der beiden Konzepte der Teilnehmer, während dem lösen der Aufgaben. Kleinere Werte sind generell besser.

4.5.4 Zeiten

In Abbildung 4.8 werden die gesamten Zeiten, in Abbildung 4.9 die Zeiten für den ersten Durchlauf und in Abbildung 4.10 die Zeiten für den zweiten Durchlauf, die während der Aufgaben gemessen wurden, gezeigt. Die Zeiten sind pro Aufgabe gemessen worden und die Beschriftung der Zeiten entspricht dem zu modellierten Objekt. Nur die Aufgaben vier und fünf wurden zur Weltmodus-Zeit zusammen gefasst.

4.5.5 Verhalten

Die Teilnehmer haben versucht sehr präzise zu modellieren und nutzten dabei alle sehr intensiv das Raster der Tiefenebene. Sechs Teilnehmer nutzten das Raster der Tiefenebene besonders intensiv. Alle waren sehr vorsichtig, was zu längeren Zeiten zum Erfüllen der Aufgaben geführt hat, im Gegen-

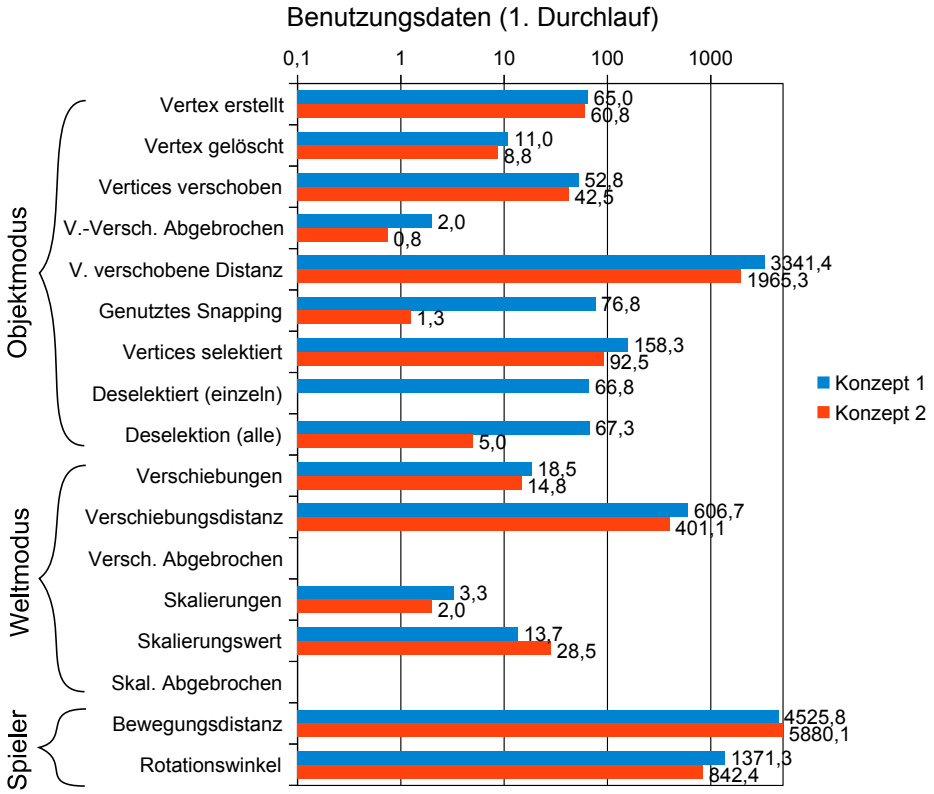


Abbildung 4.6: Die aufgezeichneten Benutzungsdaten der beiden Konzepte aus dem ersten Durchlauf der Teilnehmer, während dem lösen der Aufgaben. Kleinere Werte sind generell besser.

satz zu den Zeiten eines selbstsicheren Experten (siehe Abschnitt 4.6). Beim zweiten Durchlauf waren die Teilnehmer selbstsicherer, trotz eines anderen Konzeptes. Dies liegt wohl an den konzeptübergreifenden Eigenschaften, die nach dem ersten Durchlauf erlernt wurden. Trotz vorsichtigem Verhalten, wurden die Funktionen der linken und rechten Maustasten immer wieder vertauscht. Dies reduzierte sich jedoch im weiteren Verlauf. Sechs von acht Teilnehmern vertauschten häufig die Maustasten. Die Verwechslung bestand darin, dass sie versuchten mit der linken Maustaste Vertices zu erstellen oder aber mit der rechten Maustaste Vertices zu selektieren. Vier von acht hatten häufige *Snapping*-Probleme, bei denen ungewolltes *Snapping* auftrat. Nur drei nutzten häufiger die Abbruch-Funktion, während eine *Drag-and-Drop*-Aktion durchgeführt wurde. Die gewünschte Ausrichtung der Tiefenebene war für die Teilnehmer nicht immer schnell zu finden, auch weil es zwei unterschiedliche Typen gab. Einige vergaßen immer wieder, dass die Se-

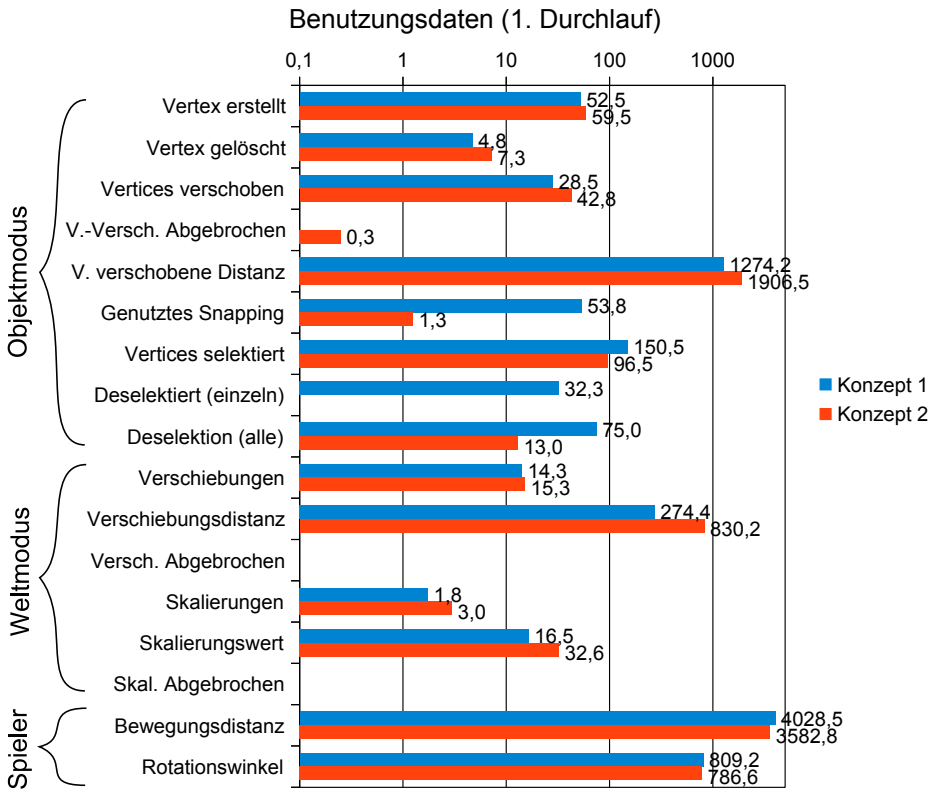


Abbildung 4.7: Die aufgezeichneten Benutzungsdaten der beiden Konzepte aus dem zweiten Durchlauf der Teilnehmer, während dem lösen der Aufgaben. Kleinere Werte sind generell besser.

lektion von nur einem Vertex, ganz andere Ausrichtungen ermöglicht. Mit der Vor-Selektion hatten die meisten Teilnehmer zu Beginn Schwierigkeiten. Sie mussten sehr konzentriert modellieren, um diese Funktion erfolgreich zu nutzen.

Obwohl die Benutzerstudie oft zwei Stunden dauerte, empfanden die Teilnehmer es nicht als Arbeit oder anstrengend. Im Gegenteil, sie hatten die meiste Zeit Spaß am Modellieren. Die meisten waren sehr begeistert von der Mischung aus Spiel und Modellierung, was wohl auch ein Grund war, dass sie an der Studie teilnahmen.

4.6 Experten-Vergleich mit *Minecraft*

Ein zusätzlicher kleinerer Test wurde durchgeführt, um zu verifizieren, ob die beiden Konzepte konkurrenztauglich sind. Dazu hat ein Experte die Aufga-

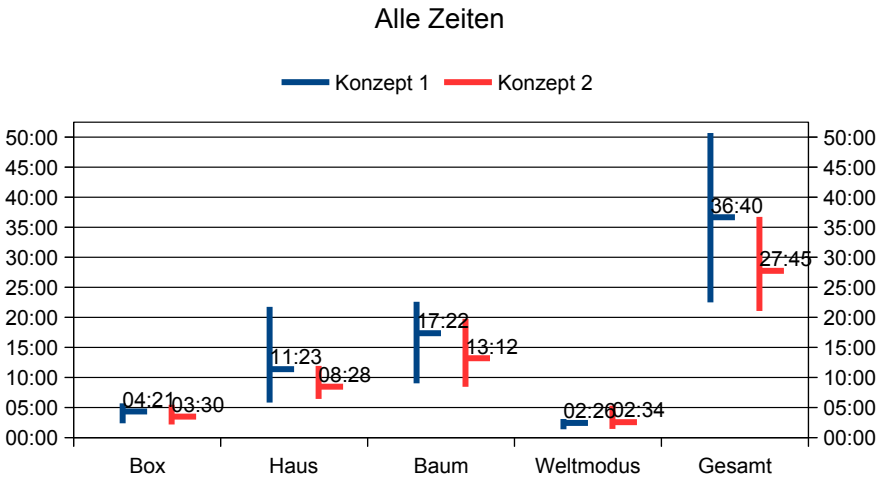


Abbildung 4.8: Die gemessenen Zeiten aus beiden Durchläufen der beiden Konzepte der Teilnehmer, bis zum Erfüllen der Aufgaben. Kürzere Zeiten sind besser. Die Zeiten sind in Minuten und Sekunden angegeben (mm:ss). Standardabweichungen (von links nach rechts): Konzept 1 – 00:49, 04:23, 04:31, 00:24, 08:17 und Konzept 2 – 00:54, 01:17, 04:05, 00:50, 04:38.

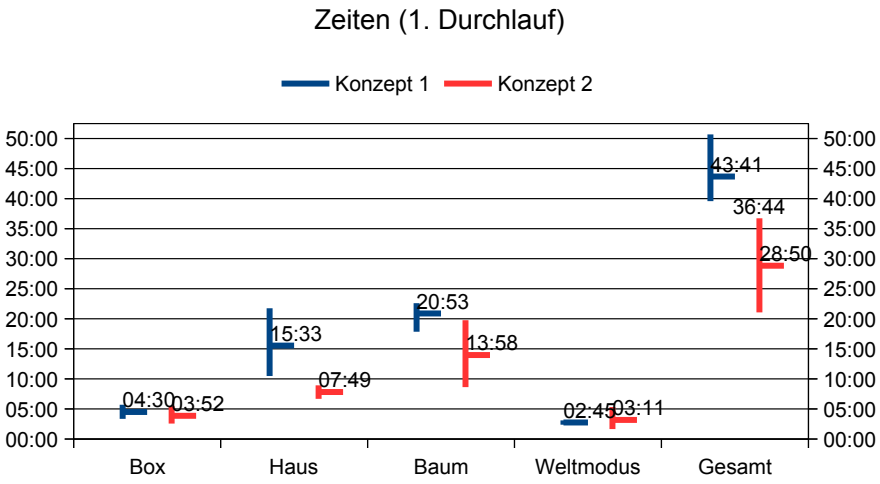


Abbildung 4.9: Die gemessenen Zeiten aus den ersten Durchläufen der beiden Konzepte der Teilnehmer, bis zum Erfüllen der Aufgaben. Kürzere Zeiten sind besser. Die Zeiten sind in Minuten und Sekunden angegeben (mm:ss). Standardabweichungen (von links nach rechts): Konzept 1 – 00:49, 03:52, 01:41, 00:18, 03:31 und Konzept 2 – 01:02, 00:38, 04:25, 01:04, 07:03.

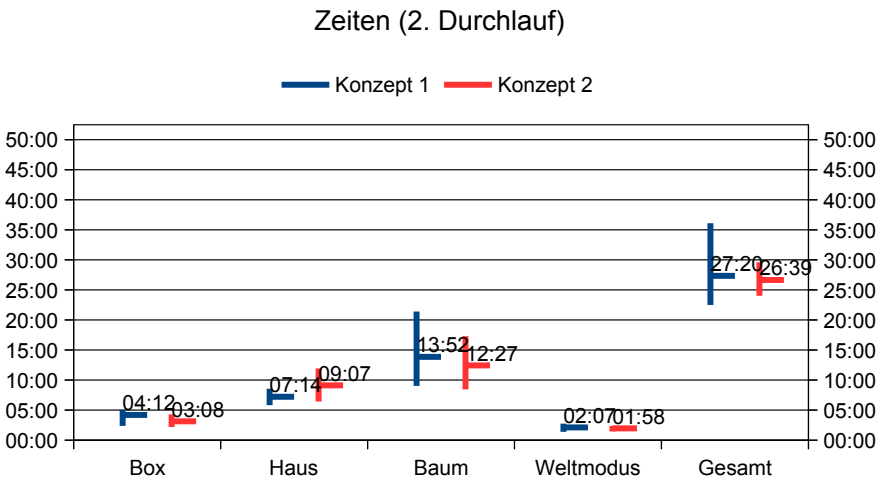


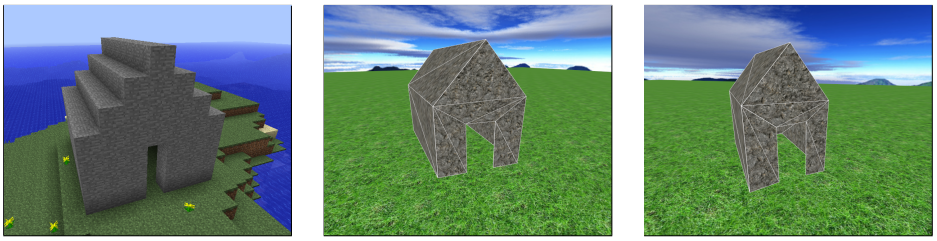
Abbildung 4.10: Die gemessenen Zeiten aus den ersten Durchläufen der beiden Konzepte der Teilnehmer, bis zum Erfüllen der Aufgaben. Kürzere Zeiten sind besser. Die Zeiten sind in Minuten und Sekunden angegeben (mm:ss). Standardabweichungen (von links nach rechts): Konzept 1 – 00:54, 00:42, 03:46, 00:30, 05:50 und Konzept 2 – 00:35, 01:41, 03:45, 00:21, 02:14.

Konzept	Zeit
Minecraft	2:01
Konzept 1	2:41
Konzept 2	1:43

Tabelle 4.1: Die Zeiten, die zur Erstellung eines Hauses, im jeweils angegebenen Konzept von einem Experten der drei Konzepte benötigt wurden.

be zwei in *Minecraft* und im Prototyp mit beiden Konzepten durchgeführt. Der Experte ist ein sehr erfahrener *Minecraft*-Spieler und verfügt über ausgereifte Fähigkeiten mit beiden Konzepten.

Das Haus wurde als typisches Objekt ausgesucht, das von den meisten Spielern gebaut wird. Für *Minecraft* können keine Benutzungsdaten aufgezeichnet werden, daher wurde nur die Zeit gemessen, die zur Erfüllung der Aufgabe benötigt wurde (siehe Tabelle 4.1). In Abbildung 4.11 sind die Objekte gezeigt, die während der Aufgabe erstellt wurden.



(a) (b) (c)

Abbildung 4.11: Die drei Häuser die beim Experten-Vergleich mit *Minecraft* erstellt wurden. (a) *Minecraft*, (b) Konzept 1 und (c) Konzept 2.

Kapitel 5

Resultate und Diskussion

Die Ergebnisse der Evaluation werden in diesem Kapitel analysiert und ausgewertet. Dieser Schritt resultiert in der Fusion der jeweils als besser empfundenen Konzepte. Zum Schluss wird validiert, ob das ursprüngliche Ziel dieser Arbeit erfüllt werden konnte.

5.1 Analyse

Die Ergebnisse in Kapitel 4 werden in diesem Abschnitt in der gleichen Gliederung ausgewertet. Es wird versucht, die Ursache der Ergebnisse zu begründen und eine Wahl zwischen den Konzepten zu geben. Als Unterstützung werden die wichtigsten Werte nochmals in Tabellen mit Vergleichswerten angegeben, auf welche referenziert wird. Die Tabelleninhalte sind auf die Gesamtwerte reduziert, also der Durchschnitt aus beiden Durchläufen. Oft wird ein Vergleich zwischen dem ersten und zweiten Durchlauf durchgeführt, um den Faktor der Erfahrung miteinzubeziehen. Diese Werte sind nicht in den Tabellen vorhanden und müssten aus den Basisdaten in Abschnitt 4.5 errechnet werden.

5.1.1 Bewertungen

Die Analyse ist in Objekt- und Weltmodus unterteilt. Hauptsächlich wird die Differenz der Bewertungen verglichen.

Objektmodus

Der Objektmodus des zweiten Konzepts ist eindeutig besser bewertet worden, als der Objektmodus des ersten Konzepts. Der Durchschnittswert für die gesamten Bewertungen des Objektmodus, ist um 0,81 besser bewertet worden, wie in Tabelle 5.1 nochmal zusammengefasst wurde. Im ersten Durchlauf ist der Durchschnittswert um 0,38 und im zweiten Durchlauf sogar

Bewertungspunkt	Bewertung (K1)	Bewertung (K2)	$K1 - K2$
Benutzerfreundlichkeit	3,50	2,75	0,75
Intuitivität	4,63	3,25	1,375
Tiefenebene	3,13	2,25	0,875
Zufriedenheit	3,63	3,38	0,25
Durchschnitt	3,72	2,91	0,81

Tabelle 5.1: Die Gesamten Bewertungen der Objektmodi beider Konzepte, deren Durchschnitt und Unterschied zueinander. Niedrigere Bewertungen sind besser. Die Bewertung geht von 1 bis 7.

um 1,25 besser (nicht in der Tabelle vorhanden). Die Differenz aller Bewertungspunkte des Objektmodus waren in beiden Durchläufen nicht negativ. Bis auf die Zufriedenheit im ersten Durchlauf, waren alle Bewertungspunkte des zweiten Konzepts besser als die des ersten Konzepts.

Die Bewertungen sprechen eindeutig für den Objektmodus des zweiten Konzeptes, allerdings sind die Werte meist näher am Mittelwert der möglichen Werte, als am besten Wert. Besonders die Intuitivität und Zufriedenheit liegen bei lediglich 3,25 und 3,38 und somit nicht optimal. Als qualitative Anforderung in Abschnitt 3.1 wurde ein *einfaches* Konzept gefordert. Die beiden Werte sind allerdings sehr viel näher am „ok“, als am „sehr gut“, was das Optimum wäre. Daher sollte an der Erfüllung dieser Anforderung noch gearbeitet werden.

Weltmodus

Der Weltmodus im ersten Konzept wurde knapp besser bewertet, als der Weltmodus des zweiten Konzepts. Bis auf die Benutzerfreundlichkeit, sind die gesamten Bewertungen des ersten Konzeptes für den Weltmodus besser. Im ersten Durchlauf war die Durchschnittswertung sogar um 0,5 Punkte besser, als die Wertung des zweiten Konzeptes. Alle drei Bewertungspunkte waren besser. Im zweiten Durchlauf wurde die Benutzerfreundlichkeit um 1,0 Punkte schlechter Bewertet, was zur schlechten Wertung für die gesamten Bewertungen führte. Es scheint als hätten einige Teilnehmer, nachdem sie das zweite Konzept kennengelernt haben, den Weltmodus des ersten Konzeptes, bei der Benutzerfreundlichkeit deutlich schlechter bewertet.

Die Durchschnittswertung und die generell besseren Wertungen der Intuitivität und der Zufriedenheit sprechen für den Weltmodus des ersten Konzepts. Die Bewertungen der Weltmodi sind deutlich besser als die der Objektmodi. Die Werte sind generell fast um 1,0 Punkte besser und auch näher an der „sehr gut“-Bewertung. Allerdings wurden die Weltmodis nicht intensiv getestet. Intensivere Tests hätten möglicherweise mehr Probleme erkennen lassen und somit zu schlechteren Bewertungen führen können.

Bewertungspunkt	Bewertung (K1)	Bewertung (K2)	K1 − K2
Benutzerfreundlichkeit	2,50	2,25	0,25
Intuitivität	1,75	2,50	-0,75
Zufriedenheit	1,75	2,13	-0,38
Durchschnitt	2,00	2,29	-0,29

Tabelle 5.2: Die gesamten Bewertungen der Weltmodi beider Konzepte, deren Durchschnitt und Unterschied zueinander. Niedrigere Bewertungen sind besser.

5.1.2 Feedback

Für beide Konzepte wurden zahlreiche Probleme und Vorschläge angemerkt, die hilfreich sind, um die Konzepte weiter zu verbessern. In Kapitel 6 wird auf die Verbesserungsvorschläge, genauer eingegangen. Die Probleme werden in diesem Abschnitt behandelt.

Konzept 1

Trotz der Funktion das *Snapping* per Taste deaktivieren zu können, gab es große Probleme mit ungewolltem *Snapping*. Diese Funktion wurde erst nach der Vorstudie hinzugefügt, um das Problem umgehen zu können. Allerdings scheint das schwierige Erkennen der Vertex-*Highlights* dazu zu führen, dass der Teilnehmer, sich nicht bewusst war, dass das *Snapping* verwendet wird. Die Funktion das *Snapping* zu deaktivieren wurde nicht in Anspruch genommen, bis das Problem öfter passierte. Dies könnte jedoch als relativ simpler Lerneffekt angenommen werden. Eine andere Auffälligkeit ist, dass beim zweiten Konzept, das gleiche Problem nur ein Mal gemeldet wurde, aber bei diesem Konzept vier Mal, obwohl in diesem Konzept das *Snapping* deaktivieren werden kann. Beim zweiten Konzept ist das neue Polygon schon während der Erstellung zu sehen, dadurch können Probleme durch *Snapping* vor dem endgültigen Erstellen erkannt werden. Dies ist in diesem Konzept nicht der Fall, da neue Polygone bei einem Rechtsklick direkt endgültig erstellt werden. Diese Visualisierung des Erstellungsprozesses scheint ein wichtiger Vorteil zu diesem Konzept zu sein. Eine andere Sichtweise des Problems ist im späteren Konzeptunabhängigen Abschnitt zu finden.

Der automatische Moduswechsel in Abhängigkeit der Entfernung zum Objekt, war für die meisten mehr störend und verwirrend als hilfreich. Wenn das Objekt sehr groß wurde, musste eine relativ weite Strecke zurückgelegt werden, um den Modus zu wechseln. Es war für die Teilnehmer nicht abzuschätzen, in welcher Distanz der Wechsel eintritt. Das Problem der beiden Funktionen auf der selben Maustaste ist in einem späteren Abschnitt erklärt.

Die Steuerungselemente zum Verschieben sind sehr groß, wenn das zu manipulierende Objekt sehr groß ist. Dies führt ab einer bestimmten Größe

dazu, dass die Pfeilenden, die besser selektiert werden können, so weit weg vom Objekt sind, dass das Objekt nicht mehr sichtbar ist beim Manipulieren. Zusätzlich zu dem Größenproblem, kommt eine Einschränkung mit dem Wechsel des Modus. Der Abstand zum Objekt der beibehalten werden muss, um im Weltmodus zu bleiben, ist zu groß, wenn das Objekt zu groß ist. Dieser Minimalabstand beeinträchtigt die Präzision der Manipulationen, besonders die Verschiebung.

Konzept 2

Das ungewollte *Snapping* ist auch in Konzept 2 ein Problem, allerdings wurde es nur von einem Teilnehmer angemerkt, im Gegensatz zu vier bei Konzept 1. Eine Erklärung ist im vorherigen Abschnitt des ersten Konzepts und im nächsten Abschnitt zu finden.

Die Ausrichtung des vordefinierten Objektes das beim Erstellen eines neuen Objektes gesetzt wird, wurde von einem Teilnehmer als störend empfunden. Es wurde gewünscht, das die Fläche parallel zur y-Achse ausgerichtet sein soll. Die Ausrichtung der Fläche zu bestimmen, war für die meisten nicht einfach. Die Ebene war immer leicht schief, obwohl sie vertikal platziert wurde. Sie musste immer wieder auf die Tiefenebene gesetzt werden, durch kurzes Verschieben aller Vertices. Dies ist eine unnötige Aktion, wenn die meisten Spieler eine Fläche parallel zur y-Achse haben wollen.

Beim Skalieren der Objekte, ist aufgefallen, dass die Eckpunkte schwierig zu selektieren sind, im Gegensatz zu den Flächen oder den Kanten. Die Eckpunkte sind häufig sehr klein, um sie ordentlich benutzen zu können. Meist wird eine andere Manipulationsmethode aus versehen angewählt.

Konzeptunabhängig

Ungewolltes *Snapping* geschieht zu häufig und scheint meistens mit Vertices hinter der Tiefenebene zu passieren, da diese schwieriger zu erkennen sind. Entweder ist ein unpassender Transparenzwert der Tiefenebene das Problem oder aber die Sichtbarkeit des Vertex-Highlight, wenn der Vertex verdeckt ist und somit eigentlich nicht gesehen wird. Die aktuelle Selektionsmethode nimmt keine Rücksicht auf die Sichtbarkeit des untersuchten Vertex.

Das Löschen und Erstellen von Vertices auf der selben Maustaste, scheint für die meisten Teilnehmer verwirrend zu sein. Eine Taste sollte wohl nicht mehr als eine Funktion haben. Allerdings schlugen einige Teilnehmer vor, das Erstellen auf die linke Maustaste zu verlegen, obwohl diese schon mit der Selektion und Verschiebung belegt ist. Außerdem reduzierte sich das Problem mit zunehmender Erfahrung der Teilnehmer und könnte auf die Dauer ganz verschwinden, durch Gewöhnung an diese Tastenbelegung.

Das selektierte Objekt ist schwierig zu erkennen, da der Unterschied zwischen selektiertem und nicht selektiertem Objekt nur die zusätzlich gezeich-

neten schwarzen Linien sind. Dies scheint keine ausreichende Markierung zu sein.

5.1.3 Benutzungsdaten

Die Daten sind unschlüssig, da sie sehr unterschiedlich sind. Zwischen dem ersten und zweiten Durchlauf konnten bei den meisten Werten keine direkten Verbesserungen der Werte erkannt werden. Eine Verschlechterung der Werte kann nicht erklärt werden, da die Teilnehmer an Erfahrung dazugewinnen. Die meisten Werte sind auch nur Zählungen von Aktionen und bestimmen nicht direkt die Leistung des Konzeptes. Die Zeiten sind repräsentativere Werte für die Leistung eines Konzeptes.

Die einzigen Werte, die auch zwischen dem ersten und zweiten Durchlauf als sinnvoll erscheinen, sind die Kamerawerte und das Erstellen und Löschen eines Vertex. Es ist in beiden Konzepten eine Verbesserung dieser Werte und somit ist eine Leistungssteigerung durch Erfahrung erkennbar.

Konzept 1

Beim ersten Konzept verringert sich der Rotationswert der Kamera um 40,99 Prozent und der Bewegungswert um 10,99 Prozent, vom ersten zum zweiten Durchlauf. Dies bedeutet, dass die Teilnehmer effizientere Bewegungen ausführten. Der stark verringerte Rotationswert kann dadurch erklärt werden, dass die aktiven Selektionen überprüft wurden, nachdem ein neuer Vertex gesetzt wurde. Im zweiten Durchlauf konnte durch vorhandene Erfahrung, die Vor-Selektion leichter erlernt werden, da weniger neue Funktionen zu lernen waren und dadurch die Teilnehmer fokussierter waren.

Das Erstellen neuer Vertices wurde im zweiten Durchlauf um 19,23 Prozent gesenkt und es wurden 56,82 Prozent weniger Vertices gelöscht. Es scheint als wurden weniger unnötige Vertices gesetzt, die im Nachhinein, doch nur gelöscht worden wären. Wenn die Anzahl gelöschter Vertices von der Anzahl erstellter Vertices subtrahiert wird, wurden 11,57 Prozent weniger Vertices gesetzt, die zum endgültigen Objekt gehören. Die selben Formen, konnten also mit weniger Vertices realisiert werden.

Konzept 2

Beim zweiten Konzept hat sich der Bewegungswert der Kamera vom ersten zum zweiten Durchlauf um 39,07 Prozent verringert und der Rotationswert um nur 6,63 Prozent. Im ersten Konzept war die Leistungssteigerung umgekehrt. Auch in diesem Konzept wurden die Bewegungen mit zunehmender Erfahrung effizienter ausgeführt. Für den Objektmodus kann diese hohe Leistungssteigerung nicht erklärt werden, aber für den Weltmodus. Die Verschiebung im Weltmodus erfordert es, die eigene Position stark zu verändern. Diese Manipulation konnte wohl effizienter ausgeführt werden.

Die Anzahl gesetzter Vertices hat sich nur marginal zum zweiten Durchlauf um 2,06 Prozent geändert. Dies könnte eine statistische Schwankung sein und wird daher ignoriert. Die Anzahl gelöschter Vertices hat sich, trotz gleich viel gesetzter Vertices, um 17,14 Prozent verringert. Es scheint, als wurden weniger unnötiger Vertices gesetzt. Wenn die Anzahl gelöschter Vertices wieder subtrahiert, wurden sogar 0,48 Prozent mehr Vertices gesetzt. Diese Zunahme kann nicht kausal erklärt werden.

Konzept-Vergleich

In Konzept 2 haben sich die Teilnehmer 10,62 Prozent weiter bewegt als in Konzept 1, was durch die zusätzlich notwendige Bewegung beim Verschieben der Objekte erklärbar ist. Der Rotationswert für Konzept 2 ist um 25,29 Prozent niedriger als für Konzept 1, was durch die notwendigen Drehungen der Kamera erklärt werden kann, um die aktuelle Selektionen zu erkennen. Das erste Konzept benötigt also mehr physischen Aufwand mit der Handbewegung als das zweite Konzept. Im zweiten Durchlauf beider Konzepte unterscheiden sich die Rotationswerte nur noch um 2,80 Prozent, damit sind beide Konzepte fast gleichwertig und der Nachteil des ersten Konzepts wird mit zunehmender Erfahrung aufgehoben.

Die Anzahl erstellter und gelöschter Vertices sind insignifikant verschieden. Nur 2,34 Prozent und 1,59 Prozent. Dies kann durch eine statistische Schwankung entstanden sein und wird daher vernachlässigt. Beide Konzepte haben somit sehr ähnliche Werte. Es werden gleich viele unnötige Vertices in beiden Konzepten gesetzt, ob gewollt oder ungewollt ist unklar.

5.1.4 Zeiten

Die benötigten Zeiten zur Erfüllung der Aufgaben sind die einzigen Leistungskriterien, neben den Benutzungsdaten. Diese repräsentieren die Leistung eines Konzeptes besser, da viele Benutzungsdaten trotz besserer Zeit, keine signifikante Veränderung aufweisen.

Objektmodus

Bis auf eine Zeit im zweiten Durchlauf, haben die Teilnehmer die kürzesten Zeiten mit dem zweiten Konzept erreicht. Die Zeit die aus dem Rahmen fällt, wurde bei der zweiten Aufgabe mit dem ersten Konzept erreicht. Im Gesamtergebnis in Tabelle 5.3 ist diese Schwankung allerdings nicht zu erkennen, da die Zeiten für das erste Konzept im ersten Durchlauf, sehr viel schlechter waren, als die Zeiten des zweiten Konzepts im ersten Durchlauf. An den gesamten Zeiten gemessen, ist das zweite Konzept mit 7:56 Unterschied zum ersten Konzept, sehr viel besser.

Beide Konzepte zeigen eine Leistungssteigerung vom ersten zum zweiten Durchlauf, für die erste und dritte Aufgabe, an. Die Leistungssteigerung für

Aufgabe	Zeit (K1)	Zeit (K2)	K1 – K2
Box	4:21	3:30	0:51
Haus	11:23	8:28	2:55
Baum	17:22	13:12	4:10
Gesamtzeit	33:06	25:10	7:56

Tabelle 5.3: Die gesamten Zeiten der Aufgaben eins bis drei beider Konzepte, deren Gesamtzeit und Unterschied zueinander. Kürzere Zeiten sind besser. Die Zeiten sind in Minuten und Sekunden.

Aufgabe	Zeit (K1)	Zeit (K2)	K1 – K2
Beide Durchläufe	02:26	02:34	-00:08
1. Durchlauf	02:45	03:11	-00:26
2. Durchlauf	02:07	01:58	00:10

Tabelle 5.4: Alle Zeiten der Aufgaben vier und fünf beider Konzepte, deren Gesamtzeit und Unterschied zueinander. Kürzere Zeiten sind besser. Die Zeiten sind in Minuten und Sekunden.

das erste Konzept beträgt 38,23 Prozent und für das zweite Konzept nur 3,74 Prozent. Beim zweiten Konzept ist eine unerklärliche Leistungsverringerung für Aufgabe zwei zu erkennen, welche etwas mehr als eine Minute ausmacht. Beim zweiten Konzept ist keine signifikante Leistungssteigerung zu verzeichnen, was zum Schluss führt, dass die Lernkurve entweder sehr flach ist und somit mehr Zeit benötigt wird um besser zu werden oder aber das Konzept schon sehr einfach ist und die Zeiten sehr gut sind im Vergleich zum ersten Konzept. Immerhin ist das zweite Konzept auch im zweiten Durchlauf noch besser als das erste (ausgenommen Aufgabe 2). Das zweite Konzept hat eine gleichbleibende Leistung, gemessen an der Zeit und profitiert nicht von der gewonnen Erfahrung aus dem vorherigen Durchlauf. Das erste Konzept hat eine steile Lernkurve, beginnt aber mit viel schlechteren Werten.

Weltmodus

Durchschnittlich ist das erste Konzept besser, als das zweite. Allerdings sind die Unterschiede nicht sehr groß, wie in Tabelle 5.4 gesehen werden kann. Eine Leistungssteigerung ist in beiden Konzepten zu erkennen.

5.1.5 Verhalten

Die Teilnehmer modellierten präzise und nutzten alle ihre Möglichkeiten, wie z.B. das Raster auf der Tiefenebene und das Verschieben von Vertices, um rechtwinklige Ecken zu erstellen. Allerdings waren sie nicht sehr selbstsicher und dies führte zu langen Zeiten zum erfüllen der Aufgaben. Trotz

vorhandener Abbruch-Funktionen für *Drag-and-Drop*-Aktionen, wünschten sich die Teilnehmer eine Funktion, um die letzte Aktion rückgängig zu machen. Die *Snapping*-Probleme traten im ersten Konzept zu häufig vor, trotz möglicher Deaktivierung. Diese Funktion scheint nicht optimal zu sein. Die Vor-Selektion ist in einigen Situationen vorteilhaft, aber die meiste Zeit hat sie wohl nur verwirrt, da einige Aktionen nicht so einfach nachvollzogen werden konnten, da auf viele Dinge gleichzeitig geachtet werden musste. Dies könnte den Lernprozess zu Beginn beim ersten Konzept gehemmt haben. Die Ausrichtungen der Tiefenebene sind zu wenig nachvollziehbar gewesen, obwohl die Erklärungen dazu verstanden wurden. Die Ausrichtungen sollten vereinfacht werden oder in der Zahl reduziert werden.

5.2 Ergebnis

Das endgültige Konzept besteht aus dem Objektmodus vom zweiten Konzept und dem Weltmodus des ersten Konzepts. Diese haben die besten Bewertungen der Teilnehmer, als auch besten Leistungsergebnisse gemessen an der Zeit. Die Benutzungsdaten waren nicht sehr hilfreich zum Bewerten der Konzepte und sollten überdacht oder ganz weggelassen werden, um unnötigen Aufwand zu vermeiden. Aus dem Feedback sind viele Probleme bekannt geworden, welche erst noch behoben oder umgangen werden sollten. In Kapitel 6 wird versucht Ansätze zu liefern, die diese Probleme lösen. Zusätzlich werden nützliche Verbesserungsvorschläge aus Abschnitt 4.5.2 diskutiert und weiterführende Literatur genannt. Es werden also Ratschläge zur weiteren Entwicklung des Konzepts gegeben. Das neue Konzept wird nun Konzept 3 genannt. Der Moduswechsel in beiden Konzepten gehört weder zum Objektmodus noch zum Weltmodus, daher muss einer der beiden Ansätze gewählt werden. Da der Moduswechsel in Konzept 2 keine Probleme bereitet hat, im Gegensatz zu Konzept 1, wird der Moduswechsel in Konzept 3 manuell per Taste aktiviert.

Wie in Abschnitt 5.1.1 erklärt, ist die Intuitivität und Zufriedenheit für den Objektmodus nicht akzeptabel. Allerdings waren die Teilnehmer begeistert vom Prototyp und hatten sehr viel Spaß beim Modellieren. Anhand der festgelegten Anforderungen in Abschnitt 3.1 ist dieses Konzept nicht vollständig akzeptabel, da die qualitative Anforderung eines *einfachen* Konzepts, nicht erfüllt wurde. Es würde aber wohl jetzt schon in seinem frühen Stadium eine Menge Spieler anlocken, da es ein innovatives Spielkonzept ist. Es sollte jedoch bedacht werden, dass die Teilnehmer während der Benutzerstudie, eine Einweisung in beide Konzepte von einem Experten bekommen haben, was in der Regel nicht möglich ist. Dies dürfte außerdem die Lernzeit erheblich verkürzt haben. Konzept 3 ist zwar nicht vollständig akzeptabel, aber eine gute Basis, um Weiterentwicklungen durchzuführen.

Kapitel 6

Ausblick

6.1 Verbesserungsvorschläge

Die Ideen der Teilnehmer aus Abschnitt 4.5.2 und die Feststellungen aus Kapitel 5, werden nun gesammelt und Umsetzungsmöglichkeiten vorgeschlagen. Es wird unterteilt in Verbesserungen für das neue Konzept und für die übrigen Modi. Einige Verbesserungsvorschläge, deren Umsetzung nicht entwickelt werden konnte, werden nicht präsentiert.

6.1.1 Konzept 3

Die Verbesserungen werden in Objektmodus, Weltmodus und Konzeptunabhängig eingeteilt.

Objektmodus

Eines der größten Probleme, ist das ungewollte *Snapping*. Nach der Vorstudie, wurde schon eine Funktion zum kurzzeitigen Deaktivieren des *Snapping* hinzugefügt. Dieser Ansatz scheint nicht gut genug zu sein, da während der Benutzerstudie, das Problem bestehen blieb. Zwei Verbesserungen werden vorgeschlagen: Anstatt das *Snapping* nur kurzzeitig zu deaktivieren, wäre es besser, dass Snapping entweder nur kurzzeitig zu aktivieren oder aber zwischen aktiv und inaktiv zu wechseln. Zusätzlich sollte auch das *Highlighting* überarbeitet werden, da das *Snapping* darauf aufbaut. Ein Vertex mit Highlight ist hinter der Tiefenebene schwierig zu erkennen und wenn dieser verdeckt ist, gar nicht erst zu sehen. Die Transparenz der Ebene sollte so angepasst werden, dass die dahinterliegenden Objekte gut sichtbar sind und verdeckte Vertices sollten vom *Highlighting* ausgeschlossen werden. Dazu kann die Sichtbarkeit z. B. per *Ray-Tracing* bestimmt werden.

Das *Snapping* sollte auf das Raster der Tiefenebene erweitert werden. Die Teilnehmer in der Benutzerstudie haben generell versucht die Vertices auf die Linien oder Ecken des Rasters zu setzen, um gerade Objekte zu

modellieren. Dies hat immer sehr viel Zeit beansprucht, da sie sehr präzise sein wollten. Das *Snapping* sollte auf die Linien des Rasters erweitert werden, oder zumindest auf die Positionen, an denen sich die Linien auf dem Raster kreuzen. Dies würde den Modellierungsprozess sehr beschleunigen. Da die Teilnehmer so intensiv mit der Tiefenebene modelliert haben, ist ein Problem beim Positionieren der Ebene aufgefallen. Wenn zwei Selektionen bestehen, wird die Ebene in den Mittelpunkt der Selektionen gesetzt. Dies führt zu einem lästigen Verhalten beim Verschieben der selektierten Vertices. Nach jeder Verschiebung dieser, wird die Ebene versetzt und die bestehenden Vertices, die am Raster ausgerichtet wurden, sind nicht mehr auf den Linien, da sich die Position der Ebene und somit der Ursprung der Textur geändert hat. Entweder die Position der Ebene wird nur noch von einem Vertex abhängig gemacht oder aber das Raster der Ebene orientiert sich nicht mehr an der Position der Ebene, sondern global. Ersteres wäre einfacher und gibt gleichzeitig dem Spieler mehr Kontrolle.

Eine häufig vermisste Funktion, ist das Löschen von Dreiecken und Linien. Wenn ein Vertex ungewollt ge-*snapped* ist und das erstellte Polygon wieder entfernt werden soll, musste einer der Vertices gelöscht werden, die zu diesem Polygon gehören. Dies führt meist zu Löschungen von mehreren Polygonen und zusätzlichem Aufwand, das zerstörte wieder zu rekonstruieren. Eine zusätzliche Löschfunktion würde die Effizienz steigern und solche Probleme vermeiden. Um allerdings ungewollten Löschungen vorzubeugen, sollte die Löschfunktion nicht mehr auf die Maustasten gelegt sein. Viele Teilnehmer haben die Löschfunktion häufig unabsichtlich benutzt, was zu Verzögerungen in der Modellierung führte. Es sollte eine Taste auf der Tastatur belegt werden, die nicht ungewollt während dem Spielen betätigt werden kann. Damit wäre auch das Problem geklärt, dass Löschen und Erstellen auf der selben Taste ist.

Im Prototyp waren die Highlights manchmal schwierig zu erkennen, da die Tiefenebene sie leicht verdeckt hat. Linien und Punkte waren nur Flache Objekte. Die Linien wurden als Linien-Primitive gezeichnet, was wahrscheinlich generell nicht benutzt wird, weil die Darstellung sehr vom Treiber abhängig ist. Das gleiche Problem existiert mit Punkt-Primitiven. Anstatt diese Primitive zu nutzen, sollten Vertices und Linien als 3D-Objekte gezeichnet werden, wodurch sie auch ein Volumen erhalten und aus der Tiefenebene herausragen würden. Vertices als Kugeln und Linien als Zylinder zeichnen. Dadurch sollte die Sichtbarkeit verbessert werden.

Die Teilnehmer hatten meistens Schwierigkeiten, die für sie passende Ausrichtung der Tiefenebene zu finden. Häufig wurde vergessen, dass die Ebene abhängig von der Anzahl selektierter Vertices ist. Fünf verschiedene Ausrichtungen sind möglicherweise zu viele. Ein Teilnehmer, der Erfahrung mit der Modellierung hat, hatte vorgeschlagen drei Ebenen, ausgerichtet an den globalen Achsen, festzulegen. Das Objekt könnte im Nachhinein noch gedreht werden.

Weltmodus

Die Größe der Verschiebungspfeile ist abhängig von der Größe des Objekts. Wenn das Objekt allerdings sehr klein oder sehr groß ist, ist es schwierig die Manipulation ordentlich durchzuführen. Wenn es sehr klein ist, sind die Pfeile zu klein um sie zu selektieren. Wenn es sehr groß ist, sind die Pfeile so weit weg, dass das Objekt beim Manipulieren am Pfeilkopf nicht mehr sichtbar ist, und nicht mehr ersichtlich ist, was die Manipulation am Objekt bewirkt. Eine Mindest- und Maximalgröße könnte diesen Problemen entgegenwirken. Außerdem sind die Linien der Pfeile so dünn, dass sie nur schwierig selektiert werden können. Die Linien sollten sehr viel dicker gemacht werden, sodass sie auch zum Manipulieren selektiert werden können.

Obwohl es nicht zu den Aufgaben gehörte, haben die Teilnehmer die Rotation ausprobieren können und wollten keine freie Rotation, sondern eine schrittweise. 90 Grad pro Schritt wäre für sie ausreichend gewesen.

Konzeptunabhängig

Was am häufigsten gewünscht wurde, ist die Funktion zum rückgängig machen der zuletzt ausgeführten Aktion, wie sie in den meisten Anwendungen zu finden ist. Auch in den Modellierungsanwendungen gehört sie zum Standard. Zusätzlich gibt es auch eine Funktion, um das rückgängig gemachte wieder rückgängig zu machen. Also zum Zustand zurückzukehren, bevor die letzte Aktion rückgängig gemacht wurde. Beide Funktionen wären eine starke Bereicherung und würden schon bekannte Funktionen bieten, die zum Standard gehören und nicht gelernt werden müssen. Normalerweise sind diese Funktionen mit Strg+Z und Strg+Y belegt, was auch übernommen werden sollte.

Einige Teilnehmer wünschten sich andere Tasten um die Höhe der Kamera zu ändern. Q und E scheinen nicht optimal zu sein, da sich bei dieser Belegung, nicht auf allen drei Achsen gleichzeitig bewegt werden kann. In *Minecraft* wird die Leertaste zum Erhöhen und Shift zum Senken der Höhe benutzt. Alternativ gibt es eine Kamerasteuerung, die keine direkte Höhenänderung erlaubt, sondern sich relativ zur Blickrichtung bewegt. Durch hinauf sehen und nach vorn bewegen, kann so die Höhe geändert werden. Diese Steuerung wurde in früheren Ego-Shootern als Beobachtungskamera benutzt.

Andere Teilnehmer wünschten sich wählbare Texturen. Diese Überlegung kam schon sehr früh bei der Entwicklung des Prototypen in den Sinn. Allerdings liegt der Fokus dieser Arbeit zuallererst in der Entwicklung eines Interaktionskonzeptes. Diese Funktion sollte aber in einem zukünftigen Konzept nicht fehlen, um die künstlerische Gestaltung der Spieler nicht einzuschränken. Die Textur für das nächste zu erstellende Polygon könnte aus einem Sortiment aus Texturen gewählt werden. So wird die Art des nächsten

Blocks in *Minecraft* gewählt.

6.1.2 Alte Konzepte

Viele der Verbesserungen für Konzept 3, sind auch für die übrigen Modi anwendbar. Die verbliebenen Verbesserungen sind auf den Objektmodus des ersten Konzeptes und den Weltmodus des zweiten Konzeptes verteilt. Obwohl das dritte Konzept gewählt wurde, wurden mit den verbliebenen Modi Erfahrungen gesammelt, die weitergegeben werden können.

Objektmodus Konzept 1

Der Erstellungsprozess im zweiten Konzept hatte den Vorteil der Visualisierung während des Prozesses. Diese kann auch für dieses Konzept übernommen werden. Diese Funktion hat wahrscheinlich dazu beigetragen, das ungewollte *Snapping* zu reduzieren, da diese erkannt werden konnten, bevor das Polygon endgültig platziert wurde.

Die Vor-Selektion war sehr schwierig für die meisten Teilnehmer umzusetzen und sollte nicht von Anfang an aktiv sein, da das ständige Ändern der Selektion, die Teilnehmer beim Erlernen anderer Funktionen behindern könnte.

Weltmodus Konzept 2

Beim erstellen eines neuen Objektes, wurde eine kleine Fläche vor dem Spieler erstellt. Diese wird abhängig von der Blickrichtung des Spielers ausgerichtet. Besser wäre es, die Fläche parallel zur y-Achse zu setzen, sodass sie nicht jedes mal angepasst werden müsste. Nur die Höhe würde dann noch vom Spieler bestimmt werden.

Beim Skalieren mit der *Bounding Box*, ist es schwierig die Eckpunkte zu selektieren. Diese sollten definitiv größer sein. Ebenso die Seitenkanten, um die Rotation einfacher vollziehen zu können. Bei der Verschiebung könnte es intuitiver sein, anstatt entlang der Normalen der selektierten Fläche, in der Fläche selbst zu verschieben. Dies hatte ein Teilnehmer vorgeschlagen.

6.2 Weitere Literatur

Im Verlauf dieser Arbeit, gab es einige Ansätze in der Literatur, die nicht direkt Fokus dieser Arbeit waren, aber als nützlich eingestuft werden konnten. Oder aber sie waren zu Aufwändig zu implementieren und mussten wichtigeren Funktionen im Prototyp weichen. Der *Gravity Fields*-Ansatz aus [14, S. 5] gehört zu diesen Funktionen. Es ist ein Ansatz um das *Snapping* zu erweitern. Durch diesen Ansatz wäre es möglich, das *Snapping* visuell und funktional zu ändern. Anstatt, dass der aktuelle Vertex am Fadenkreuz, an

eine andere Position direkt gesetzt wird, wird dieser näher zu der *Snapping*-Position gezogen, umso näher das Fadenkreuz an dieser ist. Dies ermöglicht es, Vertices direkt neben andere Vertices zu setzen. Dies ist durch das aktuelle *Snapping* nicht ohne Umstände möglich.

Der Tipp Nr. 5 könnte in den Prototyp implementiert werden und in einer erneuten Evaluation getestet werden. Obwohl im Konzept die Schattenberechnung vorgeschlagen ist, konnte diese nicht mehr in den Prototyp eingebaut werden. Ein Teilnehmer hatte die Schatten vermisst und argumentiert, dass diese bei der räumlichen Wahrnehmung unterstützen.

In [14, S. 5f] werden Geräusche als unterstützendes Element bei der Interaktion benutzt. Diese lassen den Nutzer den Zustand seiner Aktionen erfahren. Es wird argumentiert, dass der Nutzer dadurch Echtzeit-Feedback erhält. Ähnlich der Visualisierung des Erstellungsprozesses in Konzept 3.

Anhang A

Fragebogen

Zur Person

1. Wie alt sind Sie? *(Nur Einfachantwort)*

12-14 Jahre ..... ☐
15-18 Jahre ..... ☐
19-24 Jahre ..... ☐
25-30 Jahre ..... ☐
31-39 Jahre ..... ☐
40+ Jahre ..... ☐

2. Wie häufig spielen Sie Computerspiele? *(Nur Einfachantwort)*

Täglich ..... ☐
Wöchentlich ..... ☐
Monatlich ..... ☐
Jährlich ..... ☐
Noch nie ..... ☐

3. Mit welchen Geräten spielen Sie? *(Mehrfachantwort möglich)*

Computer/Laptop ..... ☐
Tablet ..... ☐
Handy ..... ☐
Konsole ..... ☐
Andere ..... _____
Bitte eintragen

4. Haben Sie Erfahrung mit 3D-Modellierungssoftware? *(Mehrfachantwort möglich)*

Ja, und zwar ...
Maya (von Autodesk) ..... ☐
3ds Max (von Autodesk) ..... ☐
Blender ..... ☐
Hexagon (von DAZ 3D) ..... ☐
Google Sketchup ..... ☐
Cinema 4D (von MAXON) .... ☐

Ja, eine andere Software: ..... _____
Bitte eintragen

Nein ..... ☐

Objekt-Modus

Nun bewerten Sie bitte den Objekt-Modus, in dem man die Objekte aus Punkten und Dreiecken zusammenbaut.

	sehr gut			ok			sehr schlecht
Benutzerfreundlichkeit	☐	☐	☐	☐	☐	☐	☐
Intuitivität	☐	☐	☐	☐	☐	☐	☐

Wie gut kommen Sie mit der transparenten Arbeitsebene zurecht?

sehr gut			ok			sehr schlecht
☐	☐	☐	☐	☐	☐	☐

Ist Ihnen alles so gelungen, so wie Sie es wollten?

sehr gut			ok			sehr schlecht
☐	☐	☐	☐	☐	☐	☐

Haben Sie Vorschläge, was man verbessern könnte? Oder eine andere Idee?

_____

_____

_____

Hat Sie etwas besonders gestört? Gab es ein besonderes Problem?

_____

_____

_____

Welt-Modus

Bewerten Sie bitte den Welt-Modus, in dem man die erstellten Objekte drehen, bewegen und skalieren kann.

	sehr gut			ok			sehr schlecht
Benutzerfreundlichkeit	☐	☐	☐	☐	☐	☐	☐
Intuitivität	☐	☐	☐	☐	☐	☐	☐

Ist Ihnen alles so gelungen, so wie Sie es wollten?

sehr gut			ok			sehr schlecht
☐	☐	☐	☐	☐	☐	☐

Haben Sie Vorschläge, was man verbessern könnte? Oder eine andere Idee?

Hat Sie etwas besonders gestört? Gab es ein besonderes Problem?

Gesamtwertung

Wie fanden Sie das Konzept im großen und ganzen?

sehr gut			ok			sehr schlecht
☐	☐	☐	☐	☐	☐	☐

Quellenverzeichnis

Literatur

[1] Eric A. Bier und Maureen C. Stone. „Snap-dragging“. In: *SIGGRAPH Comput. Graph.* 20.4 (Aug. 1986), S. 233–240 (siehe S. 16).

[2] Ralf Armin Böttcher. „Flow in Computerspielen“. Diplomarbeit. Fakultät für Informatik der Otto-von-Guericke-Universität Magdeburg, 2005 (siehe S. 1).

[3] D.A. Bowman u. a. „3D User Interfaces: New Directions and Perspectives“. In: *Computer Graphics and Applications, IEEE* 28.6 (2008), S. 20–36 (siehe S. 7, 18).

[4] Michael Chen, S. Joy Mountford und Abigail Sellen. „A study in interactive 3-D rotation using 2-D control devices“. In: *SIGGRAPH Comput. Graph.* 22.4 (Juni 1988), S. 121–129 (siehe S. 14).

[5] Mark A. DeLoura. *Game Programming: Gems 2*. Cengage Learning, 2001 (siehe S. 16).

[6] Knud Henriksen, Jon Sporring und Kasper Hornbk. „Virtual Trackballs Revisited“. In: *IEEE Trans. Vis. Comput. Graph.* 10.2 (3. Sep. 2004), S. 206–216 (siehe S. 14, 15).

[7] Ken Hinckley u. a. „A survey of design issues in spatial input“. In: *Proceedings of the 7th annual ACM symposium on User interface software and technology*. UIST '94. Marina del Rey, California, USA: ACM, 1994, S. 213–222 (siehe S. 14, 18).

[8] Ken Hinckley u. a. „Usability Analysis of 3D Rotation Techniques“. In: *Proceedings of the 10th annual ACM symposium on User interface software and technology*. UIST '97. Banff, Alberta, Canada: ACM, 1997, S. 1–10 (siehe S. 15).

[9] Jiandong Liang und Mark Green. „JDCAD: A Highly Interactive 3D Modeling System“. In: *Computer and Graphics*. Bd. 18. 4. 1994, S. 499–506 (siehe S. 14).

[10] Mark Mine. „Working in a virtual world: Interaction techniques used in the chapel hill immersive modeling program“. In: *University of North Carolina* (1996) (siehe S. 14).

[11] Jeffrey S. Pierce u. a. „Image plane interaction techniques in 3D immersive environments“. In: *Proceedings of the 1997 symposium on Interactive 3D graphics*. I3D '97. Providence, Rhode Island, USA: ACM, 1997, 39–ff. (Siehe S. 14).

[12] Ken Shoemake. „ARCBALL: a user interface for specifying three-dimensional orientation using a mouse“. In: *Proceedings of the conference on Graphics interface '92*. Vancouver, British Columbia, Canada: Morgan Kaufmann Publishers Inc., 1992, S. 151–156 (siehe S. 15, 30, 38).

[13] Wolfgang Stuerzlinger und Chadwick A. Wingrave. „The Value of Constraints for 3D User Interfaces“. In: *Virtual Realities*. Hrsg. von Guido Brunnett, Sabine Coquillart und Greg Welch. Springer Vienna, 2011, S. 203–223 (siehe S. 8, 18).

[14] Dan Venolia. „Facile 3D direct manipulation“. In: *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*. CHI '93. Amsterdam, The Netherlands: ACM, 1993, S. 31–36 (siehe S. 18, 73, 74).

Online-Quellen

[15] URL: http://www.maxon.net/uploads/pics/CINEMA_4D_Screenshot4_04.jpg (besucht am 13. 07. 2013) (siehe S. 7).

[16] *3D-Grafiksoftware*. Juni 2013. URL: http://de.wikipedia.org/w/index.php?title=3D-Grafiksoftware&oldid=118469282 (besucht am 06. 07. 2013) (siehe S. 6).

[17] Benjamin Sterbenz. *"Österreich muss Computerspiele mehr fördern"*. Feb. 2012. URL: http://futurezone.at/b2b/7154-oesterreich-muss-computerspiele-mehr-foerdern.php (besucht am 19. 08. 2013) (siehe S. 1).

[18] *Blender's Documentation 2.6: Manual - Mesh Selecting*. URL: http://wiki.blender.org/index.php/Doc:2.6/Manual/Modeling/Meshes/Selecting (besucht am 14. 07. 2013) (siehe S. 11).

[19] *List of best-selling PC games*. Aug. 2013. URL: http://en.wikipedia.org/w/index.php?title=List_of_best-selling_PC_games&oldid=570341202 (besucht am 31. 08. 2013) (siehe S. 1, 5).

[20] Markus Keler. *Videospiele: Weltweit 56 Mrd. Dollar Umsatz*. Dez. 2011. URL: http://www.pressetext.com/news/20111209014 (besucht am 19. 08. 2013) (siehe S. 1).

[21] *Maya User's Guide: Autodesk Maya 2013*. Sep. 2012. URL: http://download.autodesk.com/global/docs/maya2013/en_us/ (besucht am 08. 07. 2013) (siehe S. 7).

[22] *Medieval Sword Modeling*. URL: http://www.cgarena.com/freestuff/tutorials/maya/sword/sword.html (besucht am 14. 07. 2013) (siehe S. 10).

[23] *Minecraft-Wahnsinnige haben King's Landing aus Game of Thrones nachgebaut*. Jan. 2013. URL: http://www.bitblokes.de/2013/01/minecraft-wahnsinnige-haben-kings-landing-von-game-of-thrones-nachgebaut/ (besucht am 13. 07. 2013) (siehe S. 6).

[24] *OpenGL Programming Guide 1.1 - Chapter 14: Now That You know - Object Selection Using the Back Buffer*. URL: http://www.glprogramming.com/red/chapter14.html#name6 (besucht am 15. 07. 2013) (siehe S. 12, 13).

[25] *OpenGL Programming/Object selection*. URL: http://en.wikibooks.org/wiki/OpenGL_Programming/Object_selection#Using_the_stencil_buffer_to_identify_objects (besucht am 15. 07. 2013) (siehe S. 13).

[26] *OpenGL Selection Using Unique Color IDs*. URL: http://content.gpwiki.org/index.php/OpenGL_Selection_Using_Unique_Color_IDs (siehe S. 12).

[27] PC Games. *MOBA auf dem Vormarsch: Von der Mod zum Erfolgs-Genre - Große Historie und Spiele-Überblick*. März 2013. URL: http://www.pcgames.de/Spiele-Thema-239104/Specials/MOBA-auf-dem-Vormarsch-Von-der-Mod-zum-Erfolgs-Genre-1061841/ (besucht am 03. 07. 2013) (siehe S. 5).

[28] *Rendering a skybox using a cube map with OpenGL and GLSL*. URL: http://www.keithlantz.net/2011/10/rendering-a-skybox-using-a-cube-map-with-opengl-and-glsl (besucht am 06. 09. 2013) (siehe S. 16).

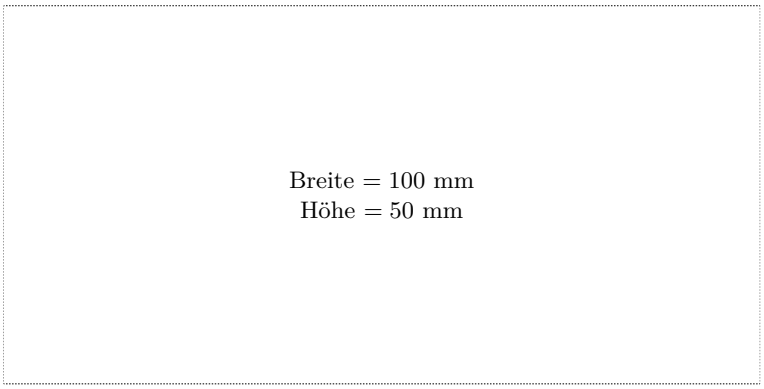
[29] *Selection Tutorial (3d programming)*. URL: http://www.3dkingdoms.com/selection.html#backbuffer (besucht am 15. 07. 2013) (siehe S. 12).

[30] I. G. N. Staff. *PC Retroview: WarCraft II*. Jan. 2002. URL: http://www.ign.com/articles/2002/01/31/pc-retroview-warcraft-ii (besucht am 03. 07. 2013) (siehe S. 4).

[31] *Tutorial ColorPicking Shader*. URL: http://wiki.delphigl.com/index.php/Tutorial_ColorPicking_Shader (besucht am 15. 07. 2013) (siehe S. 12, 13).

Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —



— Diese Seite nach dem Druck entfernen! —