MEDIADESIGN HOCHSCHULE FÜR DESIGN UND INFORMATIK
UNIVERSITY OF APPLIED SCIENCES

Generierung und Evaluierung abwechslungsreicher Gebäudekomplexe

Abschlussarbeit zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.) im Fachbereich Gamedesign

vorgelegt von:

Anja Haumann

GD1014

Betreuer:	Jakob Thomsen
	Prof. Henning Janssen
Eingereicht am:	12.02.2018

Zusammenfassung

Diese Arbeit befasst sich mit der prozeduralen Generierung abwechslungsreicher Gebäudekomplexe. Aufgrund der steigenden Ansprüche an Open-World-Games steigen auch die Kosten für manuell erstellte 3D-Inhalte. An diesem Punkt werden prozedurale Generierungsalgorithmen eingesetzt um die Kosten zu verringern. Diese haben jedoch das Problem, dass sie bezüglich der visuellen Abwechslung nicht vergleichbar gute Ergebnisse liefern wie manuell erstellte Inhalte. Im Rahmen der Arbeit soll untersucht werden, welches Potenzial prozedurale Generierungsalgorithmen im Bereich der abwechslungsreichen Gebäude haben. Im Vorfeld werden die Eigenschaften interessanter und monotoner Gebäude an konkreten Beispielen benannt. Anschließend wird ein Voxel-Modell definiert, auf dessen Basis die generierten Gebäude dargestellt werden. Darauf aufbauend wird eine Evaluierungsfunktion erstellt, die die anfangs benannten Eigenschaften an dem Voxel-Modell nachweisen kann. Anschließend wird ein prozeduraler Generierungsalgorithmus auf Basis der Evaluierungsfunktion entwickelt. Ziel der Arbeit ist es herauszufinden, welches Potenzial generierte Gebäude im Vergleich mit manuell erstellten Gebäuden im Hinblick auf den visuellen Abwechslungsreichtum haben. Abgesehen von diesem primären Ziel eignet sich der Ansatz auch als Entwurfshilfe für (Level-) Designer oder als Grundlage für die manuelle Erstellung von Gebäuden.

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen sind, sind als solche einzeln kenntlich gemacht.

Ort, Datum

Unterschrift

Danksagung

Ich danke vor allem meinem Erstbetreuer Jakob Thomsen für seine konstruktive Kritik und seine wertvollen Anregungen. Außerdem möchte ich meinem Zweitbetreuer Henning Janssen danken, welcher mich mit seinem umfangreichen Fachwissen im Bereich der nordischen Architektur zu immer neuen Ideen inspirierte.

Chris danke ich für seine unerschütterliche Ruhe und Geduld mit der er mich in Zeiten von Word-Abstürzen und verlorenen Speicherständen immer wieder zurück auf den Boden der Tatsachen holte.

Ein besonderer Dank gilt auch Julian, der mich jeden Abend daran erinnerte, dass immer Zeit für eine Runde Heroes ist, egal wie stressig es ist.

Abschließend möchte ich meiner Familie danken, die mich stets unterstützte und immer ein offenes Ohr für meine Sorgen hatte.

Inhaltsverzeichnis

Einleitung 1

1. Architekturanalyse 3

1.1 Stabkirchen 3

1.2 Skyrim 6

1.2.1 Weißlauf 7

1.2.2 Rifton 10

1.3 Lake-Town 11

1.4 Definition 15

1.4.1 Aufbau 15

1.4.2 Fassade 17

1.4.3 Innenräume 19

2. Modell und Repräsentation 21

2.1 Ansatz 22

2.2 Erweitertes Voxel-Modell 22

2.3 Datenstruktur 24

3. Evaluierung 29

3.1 Äußere Gestaltung 29

3.1.1 Aufbau 30

3.1.3 Fassade 37

3.2 Innenräume 43

3.2.1 Raumerkennung 44

3.2.2 Erreichbarkeit 45

3.2.3 Existenz von Fenstern 46

3.2.4 Mindestgröße 47

3.3 Bewertungsalgorithmus 47

4. Generierung 51

4.1 Grundriss 51

4.1.1 Erdgeschoss ... 52
4.1.2 Obere Stockwerke ... 53
4.1.3 Stützen ... 55
4.2 Innenräume ... 58
4.2.1 Basierend auf Grundrissrechtecken ... 58
4.2.2 Unabhängige Räume ... 62
4.2.3 Türen und Treppen ... 64
4.3 Fassade ... 65
4.3.1 Dächer ... 65
4.3.2 Fenster ... 67
4.4 Variation ... 69
Zusammenfassung und Fazit ... 71
Literaturverzeichnis ... 73
Spieleverzeichnis ... 75
Filmverzeichnis ... 75
Abbildungsverzeichnis ... 75
Tabellenverzeichnis ... 81
Anhang ... 83

Einleitung

Spielwelten von Open-World-Games werden immer größer und komplexer und somit auch der Bedarf an qualitativ hochwertigen 3D-Inhalten. Ein Beispiel dafür ist das Open-World-Game NO MEN'S SKY (Hello Games, 2016), welches dem Spieler eine theoretisch unendliche Spielwelt zur Verfügung stellt [1].

Diese Menge an 3D-Modellen kann nicht mehr manuell von 3D-Artists erstellt werden, aus diesem Grund hat das Team von NO MEN'S SKY auf die prozedurale Generierung zurückgegriffen um die Spielwelt abwechslungsreich und ansprechend zu gestallten.

Ein ähnliches Ziel wird auch in dieser Arbeit verfolgt. Die vorgestellte Evaluierungsfunktion soll zusammen mit dem Generierungsalgorithmus durch die automatische Generierung abwechslungsreicher und begehbare Architektur dazu beitragen die Entwicklungskosten großer Spielwelten zu verringern.

Es gibt schon viele verschiedene Ansätze um nicht begehbare Gebäudefassaden [2] [3] [4] [5], Innenräume [6] [7] [8] [9] oder Städte [10] [11] [12] zu generieren. Allerdings wenige Ansätze die sich der Generierung eines kompletten Gebäudes [13] mit Fassade und Innenräumen widmen.

Bei prozeduraler Generierung besteht allerdings immer das Problem, dass ein Programm ohne ästhetisches Empfinden ansprechende Objekte generieren soll. Der Algorithmus hat keine Vorstellung was *abwechslungsreich* im Kontext von Gebäuden bedeutet. Aus diesem Grund ist es notwendig den Begriff von Abwechslungsreichtum für den Algorithmus zu definieren.

In dieser Arbeit wird zur Lösung dieses Problems eine Evaluierungsfunktion definiert, welche den Generierungsalgorithmus mit Hilfe einer Bewertung der einzelnen Gebäudebestandteile in Richtung abwechslungsreicher Gebäude leitet.
Die zu generierenden Gebäude werden dabei auf Wohnbauten im Stil des nordischen Mittelalters eingegrenzt, welche in einem einfachen Voxel-Modell dargestellt werden.

Um die angestrebten Gebäude generieren zu können ist es notwendig die Eigenschaften abwechslungsreicher und monotoner Gebäude anhand von Architekturbeispielen zu analysieren. Dabei ist das vorrangige Ziel glaubhaft aussehende Gebäude zu generieren. Es wird nicht angestrebt die komplexen Tragewerke physikalisch und statisch Korrekter Gebäude nachzubilden. Folglich orientieren wir uns ausschließlich an dem groben Aufbau und der Fassade der Architekturbeispiele.

Dabei betrachten wir die Architektur verschiedener Medien, beginnend bei realen Gebäuden über Games bis hin zum Film:

- Mittelalterliche Stabkirchen.
- THE ELDER SCROLLS V: SKYRIM – SPECIAL EDITION
  (Bethesda Softworks, 2016, zukünftig abgekürzt: SKYRIM).
- THE HOBBIT: THE DESOLATION OF SMAUG
  (Peter Jackson, 2013, Zukünftig abgekürzt: THE HOBBIT (FILM)).

Von diesen ausgehend werden in Kapitel *1. Architekturanalyse* die Eigenschaften abwechslungsreicher Gebäude im Hinblick auf die Evaluierungsfunktion zusammengefasst und die für diese Arbeit gültige Definition abwechslungsreicher Gebäude aufgestellt. Bevor diese Eigenschaften in einer Evaluierungsfunktion festgehalten werden können, wird das Modell und die Repräsentation der zu generierenden Gebäude in Form eines erweiterten Voxel-Modells in Kapitel *2. Modell und Repräsentation* erläutert.

Darauffolgend werden in Kapitel *3. Evaluierung* die verschiedenen Methoden zum Nachweis der einzelnen Gebäudeeigenschaften vorgestellt und zu einer Evaluierungsfunktion zusammengefasst. Anschließend wird in Kapitel *4. Generierung* ein simpler Generierungsalgorithmus vorgestellt, der von der Evaluierungsfunktion in Richtung abwechslungsreicher Gebäude geleitet wird. Abschließend wird betrachtet, ob und in wie weit die Evaluierungsfunktion und der Generierungsalgorithmus in der Lage sind abwechslungsreiche Gebäude zu generieren.

1. Architekturanalyse

Zuerst werden die Eigenschaften abwechslungsreicher und monotoner Gebäudekomplexe analysiert. Da eine Trennung der allgemeinen Gebäudeeigenschaften von den Merkmalen der dargestellten Stilepochen nicht immer möglich ist, werden ausschließlich auf nordischer, mittelalterlicher Architektur basierende Gebäude als Beispiele herangezogen und die generierten Gebäude ebenfalls in diesem Stil gehalten.

Dazu werden Stabkirchen als Beispiel für mittelalterliche, nordische Architektur, Gebäude aus dem Spiel SKYRIM und die Stadt Lake-Town aus THE HOBBIT (FILM) herangezogen und deren stilistische Merkmale benannt. Anschließend wird eine Definition abwechslungsreicher Gebäudekomplexe aufgestellt, welche als Leitfaden für die Beurteilung der erstellten Gebäude dient.

1.1 Stabkirchen

Zu Beginn werden die Eigenschaften nordischer Architektur näher betrachtet. Dies dient dazu einen Einblick in die Eigenschaften der realen Vorbilder der in den nächsten zwei Kapiteln betrachteten virtuellen Gebäude zu erhalten. Insbesondere Stabkirchen dienen in diesem Zusammenhang als Paradebeispiel, da sie in SKYRIM als Tempel sowie in THE HOBBIT (FILM) in Form der großen Halle Lake-Towns größtenteils kopiert wurden, vergleiche Abbildung 1. Folglich betrachten wir Stabkirchen stellvertretend für die nordische Architektur. Ein wichtiges Merkmal dieser ist, wie der Name schon andeutet, dass aus hölzernen Stäben bestehende Tragewerk der Kirchen. Dieses wurde nicht eigens für die

1. Stabkirche Gol 2. Große Halle Lake-Towns 3. Tempel in SKYRIM

Abbildung 1: Gegenüberstellung der Stabkirche Gol, der großen Halle Lake-Towns aus THE HOBBIT (FILM) und eines Tempels des Spiels SKYRIM zur Verdeutlichung des maßgeblichen Einflusses der nordischen Stabkirchen auf die Architektur der virtuellen Gebäude. Quelle Große Halle Lake-Towns: Eigens erstellter Screenshot aus THE HOBBIT (FILM). Quelle Tempel in SKYRIM: Eigens erstellter Screenshot des Spiels SKYRIM. Quelle Stabkirche Gol: André Breutel, 2006.

Stabkirchen erfunden, sondern entwickelte sich aus der Bauweise der zuvor in Skandinavien verbreiteten Wikinger-Langhäusern. Als Beispiel für ein solches Langhaus dient eine Rekonstruktion im Freilichtmuseum von Trelleborg, siehe Abbildung 2. An diesem Langhaus kann man bereits die Urform der Holzmasten-Konstruktion einer Stabkirche

Abbildung 2: Rekonstruktion eines Wikinger-Langhauses im Freilichtmuseum von Trelleborg. In seinem Aufbau sind bereits die Grundzüge einer Stabkirche in Form von senkrecht stehenden Holzmasten als Stütze für das Dach des Umganges (A), sowie ein gestaffeltes Dach(B) mit Giebeldekoration (C) zu erkennen. Quelle: Erweiterte Grafik von Schorle 2015.

erkennen. Diese äußert sich in den senkrecht stehenden Stützmasten des das Langhaus umschließenden Umganges und der auf diesen Masten ruhenden Dachkonstruktion mit Giebelschmuck. Diese Merkmale bleiben in der Bauweise der Stabkirchen weiterhin erhalten und werden über die Zeit immer mehr verfeinert. Insbesondere das hölzerne Tragewerk wird weiter entwickelt um den Stabkirchen eine mehr dem Himmel entgegenstrebende Architektur zu ermöglichen, vergleiche Abbildung 3.

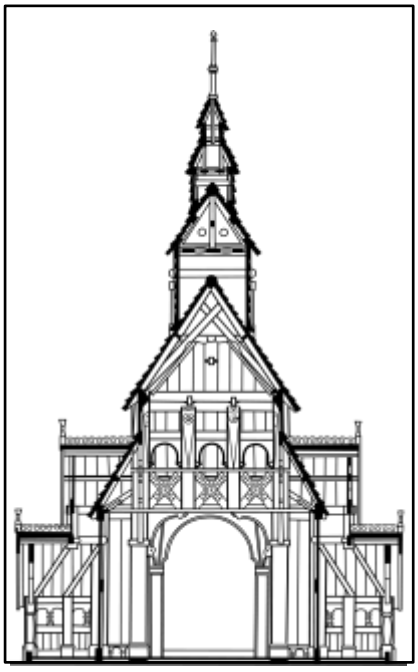


Abbildung 3: Querschnitt der Stabkirche Gol zur Verdeutlichung des komplexen Tragewerks, welches die gestaffelte Dachkonstruktion stützt. Quelle: Micha, 2009.

Stabkirchen sind hölzerne Kirchengebäude, welche seit Beginn des 11. Jahrhunderts bis ins 14. Jahrhundert in Norwegen erbaut wurden [14]. Sie bilden mit ihrer hölzernen Bauweise eine Ausnahme unter den europäischen Sakralbauten [14]. In Europa wurden Kirchen bevorzugt aus Stein und nicht aus Holz errichtet [14]. Da es aber keine Steinbautradition in Norwegen gab, wurde auf Holz als bekanntes Baumaterial zurückgegriffen [14]. Steine wurden nur für die Fundamente der Stabkirchen verwendet um das leicht faulende Holz

vom feuchten Boden zu trennen und so die Haltbarkeit der tragenden Holzmasten zu erhöhen [14]. Diese senkrecht stehenden Holzmasten/Stäbe bilden ein komplexes Tragewerk das die gestaffelte Dachkonstruktion stützt und namensgebend für diese Art von Kirchen ist [14], vergleiche Abbildung 3.

Als konkretes Beispiel dient dabei die Stabkirche Gol im Norsk Folkmuseum Oslo, vergleiche Abbildung 1, Abbildung 3 und Abbildung 4. Die Eigenschaften der Stabkirche werden im weiteren Verlauf in zwei Kategorien unterteilt, die Fassade und den Aufbau. Der Aufbau umfasst dabei alle Merkmale die vorrangig den Grundriss oder die grundlegende Bauweise der Stabkirche Gol betreffen. Die Fassade umfasst die restlichen Merkmale in Form von Dekoration und Oberflächen.

Neben den bereits oben benannten Merkmalen besitzt die Stabkirche als sakraler Bau auch den grundlegenden Aufbau eines Kirchengebäudes. Zu diesem zählt neben einem symmetrischen Grundriss auch ein an den Hauptraum anschließender, kleinerer Altarraum sowie Dachdekoration in Form von Giebelkreuzen. Neben diesen Elementen christlichen Glaubens sind auch heidnische Symbole und Dekorationen in Form von Drachendarstellungen anzutreffen. Diese befinden sich hauptsächlich in Form von Drachenköpfen auf den schwarzen, mit Firstkämmen geschmückten Schuppen-Dächern, vergleiche Abbildung 4. Die in die Höhe gestaffelten Dächer ruhen dabei auf mit zunehmender Anzahl kleiner werdenden Stockwerken. Jedes Stockwerk besitzt dabei eine kleinere Grundfläche als dass unter ihr liegende. Die Wände dieser Stockwerke können aus senkrecht stehenden Brettern bestehen oder mit Schuppen überzogen sein, jedoch enden sie an den Gebäudeecken immer in großen Eckmasten.

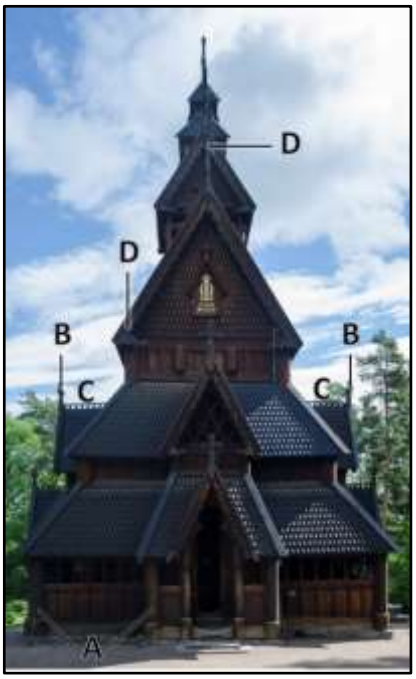


Abbildung 4: Westportal der Stabkirche Gol zur Verdeutlichung der Fassadengestaltung einer Stabkirche. A: Eckmasten. B: Giebelkreuz. C: Firstkamm. D: Drachenköpfe. Quelle: Erweiterte Grafik von Peulle, 2017.

Da wir nicht das Ziel verfolgen Kirchengebäude zu generieren, besitzen alle Eigenschaften sakraler Gebäude keine Relevanz für die Generierung unserer Gebäudekomplexe und werden im weiteren Verlauf nicht beachtet. Alle genannten Eigenschaften sind zusammen mit ihrer Relevanz für diese Arbeit noch einmal in Tabelle 1 aufgeführt.

Eigenschaften nordischer Stabkirchen			
Fassade	*Relevant*	*Aufbau*	*Relevant*
Schwarze Schuppen-Dächer	X	Holzbauweise	X
Dachverzierung - Drachen	X	Tragewerk aus Holzmasten	
Giebelkreuze		Steinfundament	
Firstkämme	X	Symmetrischer Grundriss	
Eckmasten	X	Altarraum	
Verschiedene Wandbeläge • Senkrechte Bretter • Schuppen	X	Mit zunehmender Anzahl kleiner werdende Stockwerke	X
		In die Höhe gestaffelte Dächer	X

Tabelle 1: Zusammenfassung der Eigenschaften nordischer Stabkirchen am Beispiel der Stabkirche Gol und ihrer Relevanz für diese Arbeit. Relevante Eigenschaften sind mit einem X markiert. Quelle: Eigens erstellte Tabelle.

1.2 Skyrim

Skyrim ist ein Open-World-RPG mit nordischem Setting. Folglich haben sich die Entwickler bei dem Design ihrer Gebäude an bereits bestehender nordischer Architektur orientiert. Als Beispiele hierfür dienen die Städte Weißlauf und Rifton. Diese gehören zu den 9 großen Städten SKYRIMS und können somit stellvertretend für die Architektur SKYRIMS gesehen werden. Dabei stehen die Häuser der einfachen Bevölkerung in Form von Läden und Wohnhäusern im Fokus. Diese dienen im weiteren Verlauf der Arbeit als Basis für die Erstellung des Evaluierungs-Algorithmus und als Vergleichsmaterial für die Qualität der generierten Gebäude.

1.2.1 Weißlauf

Zuerst untersuchen wir die Brauerei vor den Toren Weißlaufs. So wie die meisten Gebäude Weißlaufs besitzt auch sie Gemeinsamkeiten mit den norwegischen Stabkirchen, vergleiche Abbildung 6. Neben diesen visuellen Merkmalen besitzt sie einige Eigenschaften, die sie als ein abwechslungsreiches Gebäude charakterisieren. Eines dieser Merkmale ist ihr auf der horizontalen und vertikalen Achse asymmetrischer Grundriss, siehe Abbildung 5. Aus allen 4 Blickwinkeln bietet sich daraus folgend eine einzigartige Ansicht des Gebäudes, vergleiche Abbildung 7.

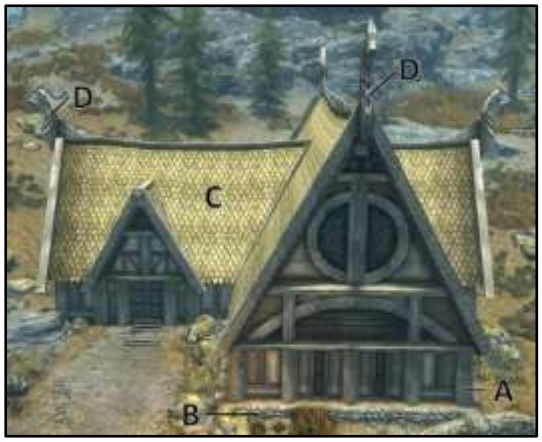


Abbildung 6: Die Brauerei vor den Toren Weißlaufs. Die Parallelen zur zuvor analysierten Stabkirche sind wie folgt markiert. A: Eckmasten. B: Steinfundament. C: Geschuppte Dächer. D: Drachenköpfe. Quelle: Eigens erstellter Screenshot des Spiels SKYRIM.

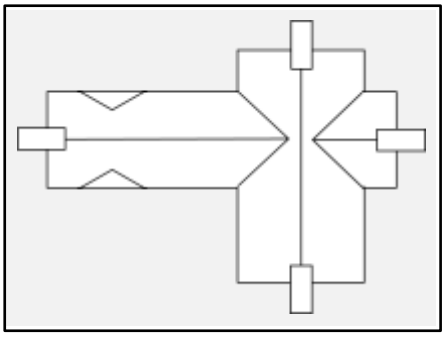


Abbildung 5: Grundriss der Brauerei zur Verdeutlichung des asymmetrischen Grundrisses. Quelle: Eigene Grafik.

Diese Wirkung wird noch zusätzlich von den verschiedenen Richtungen der unterschiedlich hohen Dachteile erhöht, zu welchen auch der kleine Giebel über dem Portal zählt. Zusammen mit dem Dachschmuck, in Form von Drachenköpfen, tragen die lang gezogenen Giebel maßgeblich zu der interessanten Silhouette des Gebäudes bei.

Abgerundet wird das Gesamtbild von der abwechslungsreichen Wandgestaltung der Brauerei. In den Zwischenräumen der für das Gebäude notwendigen Stützbalken sind weitere Balken zur Dekoration eingesetzt. Besonders die Kreisform im oberen Teil des vorderen Giebels ist dabei ein Blickfang, vergleiche Abbildung 6.

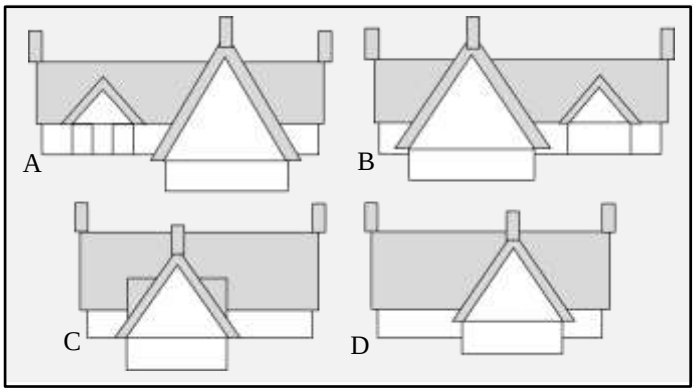


Abbildung 7: Dargestellt sind alle 4 Ansichten der Brauerei. Dies dient zur Veranschaulichung der aus dem asymmetrischen Grundriss resultierenden unterschiedlichen Fassadenseiten. A: Front. B: Rückansicht. C: Linke Seite. D: Rechte Seite. Quelle: Eigene Grafik.

Das in Abbildung 8 dargestellte Stadthaus Weißlaufs besitzt im Gegensatz zu der Brauerei einheitlich hohe Dächer ohne Dachschmuck und einen auf der vertikalen Achse symmetrischen Grundriss, siehe Abbildung 10. Dies resultiert in einer monotoneren Wirkung der Fassaden. Dieser Eindruck wird von den beiden Fenstergiebeln abgeschwächt, welche die aus der Symmetrie des Grundrisses resultierenden, gleichförmigen Fassaden aufbrechen, vergleiche Abbildung 9.

Abbildung 8: Wohnhaus der Stadt Weißlauf zur Verdeutlichung der Eigenschaften abwechslungsreicher und monotoner Gebäude. Quelle: Eigens erstellter Screenshot des Spiels SKYRIM.

Dies sorgt dafür, dass der Abwechslungsreichtum des Hauses gewahrt bleibt. Woraus sich folgern lässt, dass kein vollkommen asymmetrischer Grundriss notwendig ist, um ein abwechslungsreiches Gebäude zu bilden. Die aus einem symmetrischen Grundriss folgende Gleichförmigkeit kann durch die interessante Gestaltung der Fassaden aufgehoben werden. Dies erfolgt im Fall des Stadthauses durch den Einsatz von Fenstergiebeln. Dieser Effekt kann ebenfalls durch den Einsatz von kleineren Anbauten in Form von Erkern erzielt werden. Ein Gebäude, welches ebenfalls versucht seine aus dem Grundriss folgende Monotonie durch die Platzierung von Anbauten aufzuheben ist in Abbildung 11 zu sehen. In diesem Fall wirkt das Haus jedoch nicht so abwechslungsreich wie das zuvor

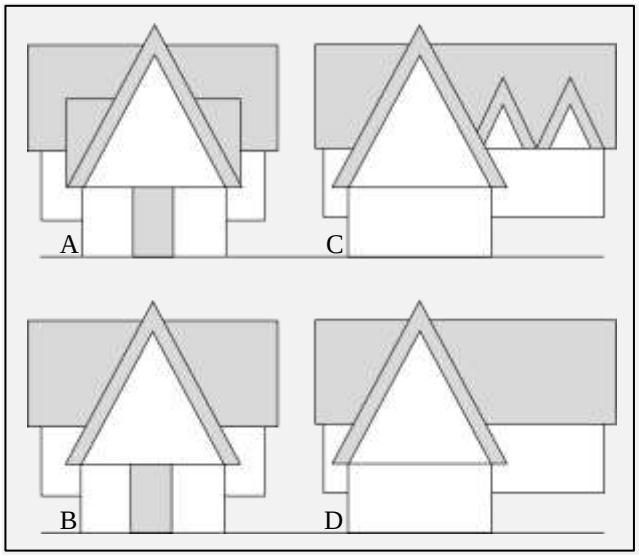


Abbildung 9: Front- und Seitenansicht des Stadthauses einmal mit und ohne Fenstergiebel zur Verdeutlichung ihres positiven Einflusses auf die Fassaden des Hauses. A: Frontansicht mit Giebeln. B: Frontansicht ohne Giebel. C: Seitenansicht mit Giebeln. D: Seitenansicht ohne Giebel. Quelle: Eigene Grafik.

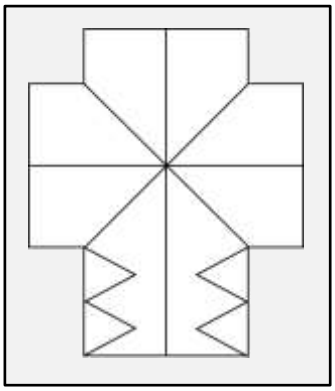


Abbildung 10: Grundriss des Stadthauses zur Verdeutlichung des symmetrischen Grundrisses und der daraus folgenden Monotonie. Quelle: Eigene Grafik.

analysierte Stadthaus. Trotz seiner Fassadenerweiterungen in Form eines kleineren Anbaus auf der rechten Seite des Gebäudes und einem Dachaufbau wirkt es sehr eintönig, vergleiche Abbildung 13. Der einzige wesentliche Unterschied im Aufbau des Gebäudes besteht darin, dass sein Grundriss ohne Anbauten aus einem einzelnen Rechteck besteht und nicht aus zwei Formen, wie bei dem Stadthaus in Abbildung 10.

Abbildung 11: Ein weiters Wohnhaus Weißlaufs als Beispiel für einen gescheiterten Einsatz von Anbauten um die Monotonie des Grundrisses aufzuheben. Quelle: Eigens erstellter Screenshot des Spiels SKYRIM.

Folglich spielt nicht nur die Symmetrie des Grundrisses eine Rolle für den Abwechslungsreichtum des Hauses, sondern auch die Anzahl der Formen, aus denen der Grundriss besteht. Um diese These zu überprüfen wurde dem Grundriss des Wohnhauses ein Rechteck hinzugefügt das seine Achsensymmetrie nicht verändert, siehe Abbildung 12. Nach dieser Veränderung ähnelt der Grundriss mehr dem zuvor analysierten Stadthaus, vergleiche Abbildung 10. Zusammen mit dieser neu entstandenen Ähnlichkeit hat sich ebenfalls der Abwechslungsreichtum des Gebäudes im Vergleich mit den ursprünglichen Ansichten in Abbildung 13 erhöht. Somit kann festgehalten werden, dass neben der Symmetrie des Grundrisses ebenfalls die Anzahl der Grundriss-Formen von Bedeutung für den Abwechslungsreichtum eines Gebäudes ist.

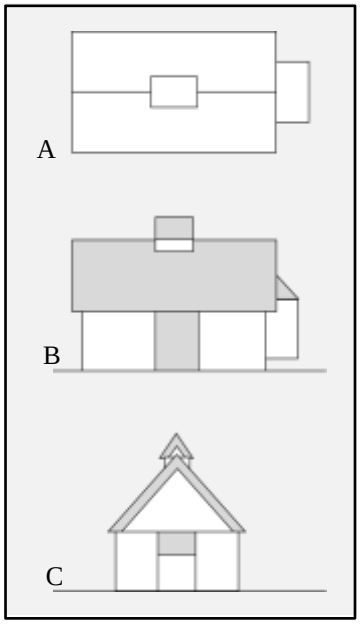


Abbildung 13: Zu sehen sind der Grundriss, die Front- sowie Seitenansicht des Wohnhauses zur Verdeutlichung des Aufbaues. A: Grundriss. B: Frontale Ansicht. C: Rechte Seitenansicht. Quelle: Eigene Grafik.

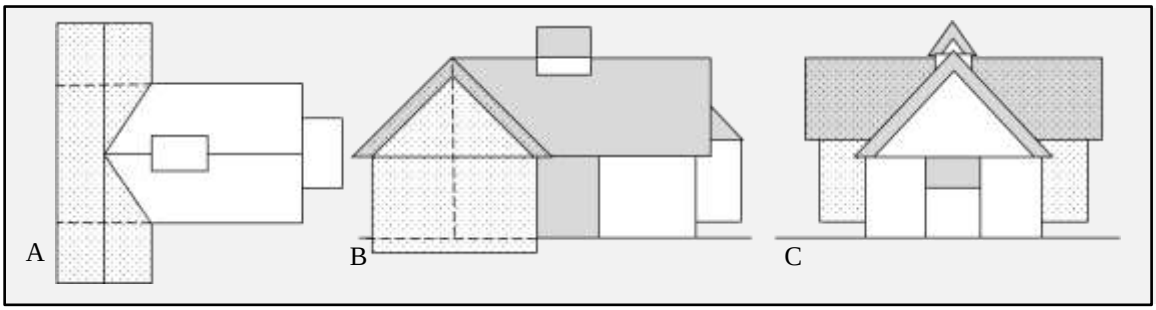


Abbildung 12: Hier wurde das zuvor analysierte Wohnhaus durch das Hinzufügen eines Rechtecks zu seinem Grundriss verändert. Die zuvor bestehende Achsensymmetrie des Grundrisses ist dabei unverändert geblieben. Im Vergleich mit den ursprünglichen Ansichten in Abbildung 13 kann ein gestiegener Abwechslungsreichtum des Gebäudes festgestellt werden. Die neu hinzugefügten Teile der Grafik sind mit einem leichten Punktmuster markiert. A: Neuer Grundriss. B: Daraus folgende Frontalansicht. C. Aus dem Grundriss folgende Seitenansicht. Quelle: Eigene Grafik.

1.2.2 Rifton

Die Architektur Riftons zeichnen sich im Gegensatz zu der Weißlaufs durch mehrere Stockwerke und wesentlich flachere Dächer ohne Verzierungen aus. Wobei die oberen Stockwerke auch über die darunterliegenden hinausragen können und bei Bedarf von Masten gestützt werden, siehe Abbildung 14.

Abbildung 14: Die Taverne der Stadt Rifton, im linken Bereich des Bildes, ist ein Paradebeispiel für ein Gebäude mit einem überhängenden Stockwerk das von Masten Gestützt wird. Quelle: Eigens erstellter Screenshots des Spiels SKYRIM.

Diese Eigenschaft ist im Vergleich mit den zuvor analysierten Stabkirche Gol interessant, da deren Stockwerke nach innen Springen und die der Gebäude Riftons nach außen. Eine Erklärung dafür kann in der Rähmbauweise von Fachwerkhäusern gesucht werden, vergleiche Abbildung 15. Dabei werden aus baulichen Gründen die Stockwerke immer so aufgelegt, dass sie ein Stück über das untere hinausragen [15].

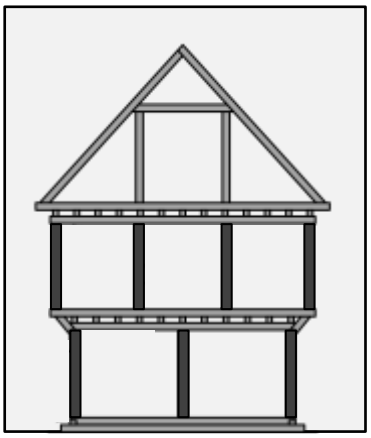


Abbildung 15: Rähmbauweise eines Fachwerkhauses. Quelle: Ulrich J. 2006. - In Anlehnung an Manfred Gerners Buch - Fachwerk: Entwicklung, Gefüge, Instandsetzung [14]

Dies hat den Vorteil, dass die Wohnfläche der oberen Stockwerke vergrößert wird. Eine Stabkirche benötigt jedoch keinen Wohnraum in den oberen Stockwerken, weshalb sich die Architekten entschieden haben könnten die Stockwerke nach innen springen zu lassen um stattdessen die Vertikalität weiter zu betonen.

Somit stehen die Stockwerke der Stadt Rifton nicht nur aus ästhetischen Gründen über die Darunterliegenden, sondern orientieren sich an den praktischen Anforderungen von mittelalterlicher Architektur um ihre Wohnfläche zu vergrößern. Die Stockwerke der Stadt Rifton stehen dabei aber ein ganzes Stück weiter über ihren Grundriss hinaus als die Stockwerke eines Fachwerkhauses. Dies kann im Fall der Taverne in Abbildung 14 allerdings mit den bereits bestehenden Stützbalken des Steges begründet werden. Da die Balken schon bestehen, spricht nichts dagegen, das Stockwerk so weit überhängen zu lassen, das die Balken als Stütze genutzt werden können. Dies lässt das Gebäude abwechslungsreicher und gleichzeitig im Kontext mittelalterlicher Architektur glaubhafter erscheinen.

Abschließend sind alle von den Gebäuden Weißlaufs und Riftons abgeleiteten Eigenschaften abwechslungsreicher Gebäude noch einmal in Tabelle 2 zusammengefasst.

Eigenschaften abwechslungsreicher Gebäude SKYRIMS
Asymmetrische Grundrisse
Aus mehreren Formen bestehende Grundrisse
Mehrere Stockwerke
Unterschiedlich große Stockwerke
Überhängende Stockwerke
Verschieden hohe Dächer
Unterschiedliche Dachrichtungen
Dachfenster mit Giebeln
Verschiedene Wandbeläge

Tabelle 2: Zusammenfassung aller von den Gebäuden Weißlaufs und Riftons abgeleiteten Eigenschaften abwechslungsreicher Gebäude. Quelle: Eigens erstellte Tabelle.

1.3 Lake-Town

Im Vergleich mit der Architektur SKYRIMS orientiert sich Lake-Town stärker an dem Vorbild der nordischen Stabkirche und übernimmt mit besonderer Detailtreue:

- Aufwendiger Dachschmuck
  - Drachenköpfe
  - Firstkämme
- Dunkel gedeckte Dächer
- Verschiedene Wandbeläge
  - Bretter
  - Schuppen

Vergleiche Abbildung 16.

Dabei zeigt Lake-Town neben den skandinavischen Einflüssen auch Merkmale tibetischer Architektur [16, pp. 110, 115].

Abbildung 16: Dachkonstruktion der großen Halle Lake-Towns zur Verdeutlichung der Gemeinsamkeiten mit der Stabkirche Gol. Zu diesen zählen die gestaffelte Dachkonstruktion, die Dekoration in Form von Drachenköpfen und Firstkämmen. Quelle: Eigens erstellter Screenshot aus THE HOBBIT (FILM).

In der Buchvorlage THE HOBBIT, OR THERE AND BACK AGAIN [17] diente Tolkien laut John Howe aber eine wesentlich ältere Architektur als Vorlage:

> *„Tolkien's original inspiration for Lake-town was very likely taken from the Swiss Lake Dwellers, the Celtic people who were thought to have built villages on the many Swiss lakes, on platforms out over the water. "*
> [16, p. 111]

Abbildung 17: Vergleich der Originalzeichnung Tolkiens von Lake-Town (A) mit einer Rekonstruktion eines *Swiss Lake Dweller* Dorfes(B) und einer alten Postkarte des Pfahlbaumuseums Unteruhldingen von 1950. Dies dient zur Verdeutlichung der möglichen Inspiration Tolkiens durch altertümliche Pfahlbausiedlungen Nordeuropas. Quellen: A - [16]. B - [21]. C – [22]

Diese zeichnet sich durch eine wesentlich einfachere, komplett auf Holz basierte Bauweise aus. Die Architektur Lake-Towns im Film hat hingegen mehr Gemeinsamkeiten mit einem aus Holz erbautem Venedig [16, p. 115] und hat sich im Vergleich mit den Pfahlbausiedlungen architektonisch wesentlich weiterentwickelt.

Dabei besteht eine große Ähnlichkeit zwischen der großen Halle Lake-Towns und der zuvor analysierten mittelalterlichen Stabkirche Gol, vergleiche Abbildung 4 und Abbildung 16. Die einfachen Wohngebäude hingegen zeichnen sich im Gegensatz dazu durch eine bodenständige, mehr horizontal orientierte Bauweise aus.

Dabei zeigen sie allerdings auch die bereits erarbeiteten Eigenschaften abwechslungsreicher Gebäude in Form von mehreren, unterschiedlichen Stockwerken mit über den Grundriss überstehenden Teilen. Wobei hier, wie bei den Gebäuden der Stadt Rifton in SKYRIM die Stockwerke teilweise extrem über ihre Grundrisse hinausragen um die Stützpfeiler der darunterliegenden Stege für das Gebäude nutzen zu können, vergleiche Abbildung 20.

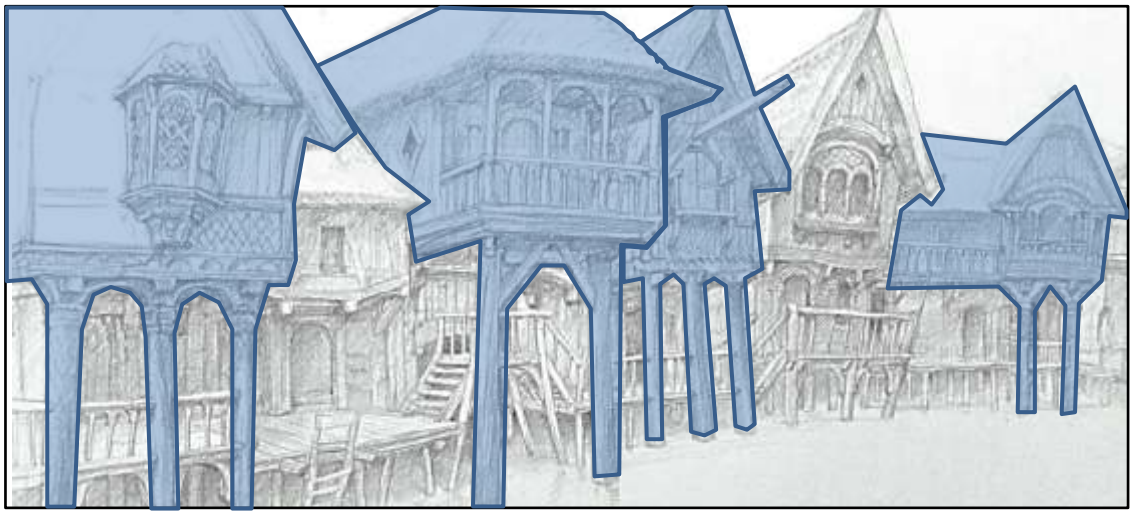


Abbildung 20: Concept einer Häuserfront Lake-Towns von John Howe. Dieses Bild dient zur Verdeutlichung der horizontalen Bauweise einfacher Wohnhäuser und der deutlich über den Grundriss hervorstehenden Stockwerkteile, welche die unter ihnen liegenden Stützpfeiler nutzen. Quelle: Erweiterte Grafik von John Howe [15].

Ein weiters wichtiges Merkmal der Gebäude Lake-Towns ist die Tatsache, dass alle Gebäude auf hölzernen Stegen stehen und mit Brücken verbunden sind. Dies beeinflusst maßgeblich die Wirkung Lake-Towns als Stadt und lässt sie, ähnlich Venedig, in ihrem Aufbau sehr fragmentiert wirken, siehe Abbildung 19. Im Nachfolgenden werden 2 Wohnhäuser Lake-Towns noch einmal im Detail betrachtet um deren besondere Eigenschaften und Stilmerkmale zu verdeutlichen.

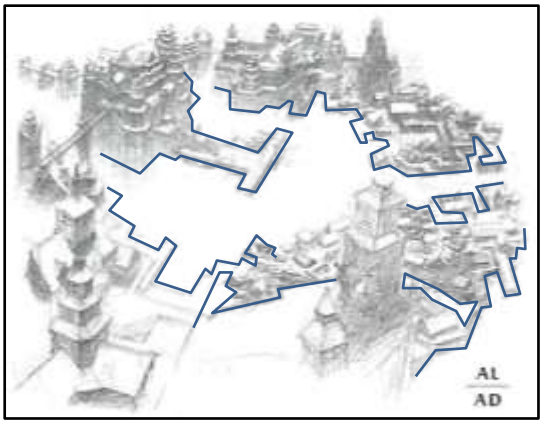


Abbildung 19: Übersicht über einen Bereich Lake-Towns zur Verdeutlichung der Fragmentierung im Stadtbau und deren Wirkung. Quelle: Erweiterte Grafik von Alan Lee [15].

Das erste Gebäude, dass wir näher betrachten ist das Haus Bards, eines der Protagonisten in THE HOBBIT (FILM). Hier sind im Besonderen die windschiefen Wände und Stege/Brücken Lake-Towns gut zu sehen, vergleiche Abbildung 18. Eine besondere Eigenschaft dieses Gebäudes ist es, dass sich die Haustür im Obergeschoss und nicht im Erdgeschoss befindet. Der Grundriss des Gebäudes ist dabei im Gegensatz zu den schiefen Wänden einfach gehalten und auf allen

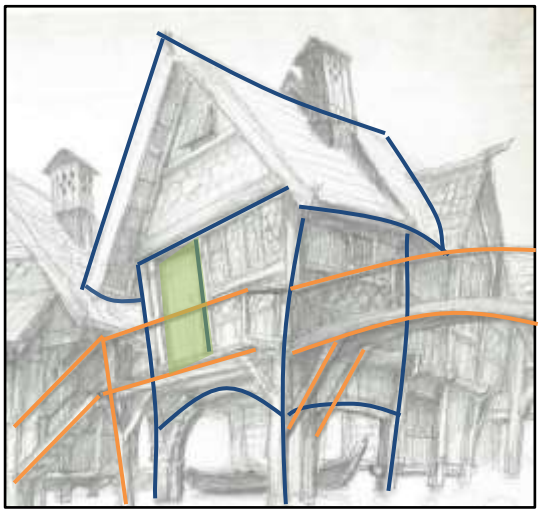


Abbildung 18: Am Haus Bards kann man besonders gut die windschiefen Wände(Blau), Brücken und Stege (Orange)und die Haustür im Obergeschoss (Grün)erkennen, welche die Architektur Lake-Towns auszeichnet. Quelle: Erweiterte Grafik von Alan Lee [15].

Achsen symmetrisch. Dies tut dem Abwechslungsreichtum des Gebäudes jedoch keinen Abbruch, da es durch die schiefen Wände und interessanten Fassaden abwechslungsreich genug wirkt. Ein asymmetrischer Grundriss könnte in diesem Fall auch negative Auswirkungen haben, da das Gebäude dadurch zu unstet oder überladen wirken könnte.

Ein weiteres Beispiel für die gelungene Kombination einer interessanten Fassade und eines einfachen Grundrisses zu einem abwechslungsreichen Gebäude ist in Abbildung 21 zusehen.

Abbildung 21: Concept eines Gebäudes Lake-Towns von John Howe. Hier ist besonders gut die Kombination eines einfachen Grundrisses mit einer interessanten Fassade zur Erzeugung von Abwechslungsreichtum und der Einsatz von Rundbögen (Blau) zu sehen. Quelle: Erweiterte Grafik von John Howe [15].

Die Grundrisse des Gebäudes bestehen jeweils nur aus einem einzigen Rechteck, die Fassaden sind allerdings reich mit verschiedensten Wandbelägen und Fenstern verziert. In diesem Zusammenhang fällt auf, das insbesondere bei den Fenstern und kleineren Stützen gerne Rundbögen als Dekoration genutzt werden. Daraus lässt sich nun folgern, dass die Gebäude Lake-Towns ihren Abwechslungsreichtum vorrangig durch die Kombination einfacher Grundrisse mit komplexen Fassaden erreichen. Zusammengefasst zeigen die Gebäude Lake-Towns zusätzlich zu den bereits erarbeiteten Eigenschaften abwechslungsreicher Gebäude die in Tabelle 3 zusammengefassten Eigenschaften.

Eigenschaften der Gebäude Lake-Towns
Weit überhängende Stockwerke
Einfache, asymmetrische Grundrisse
Mehrere Stockwerke
Aufwendige Fassaden
Holzstege und Brücken
Haustüren in oberen Etagen
Windschiefe Wände
Rundbögen

Tabelle 3: Zusammenfassung aller neuen, von den Gebäuden Lake-Towns abgeleiteten Eigenschaften abwechslungsreicher Gebäude. Quelle: Eigens erstellte Tabelle.

Da Lake-Town einen sehr interessanten und abwechslungsreichen Architekturstil besitzt, gibt es bereits Versuche den Stil der Gebäude prozedural zu reproduzieren. Erwähnenswert ist in diesem Zusammenhang Anastasia Operas Projekt Lake-Village, welches versucht Gebäude im Stil Lake-Towns prozedural in Houdini[1] zu generieren.

Abbildung 22: Beispiele der von Anastasia Opara in Houdini generierten Lake-Village Gebäude, welche versuchen den Stil von Lake-Town nachzustellen. Quelle: Einzelne Bilder - Anastasia Opara 2016.

1.4 Definition

Nachdem wir nun verschiedene Architekturbeispiele aus den Bereichen Realität, Games und Film betrachtet haben, rekapitulieren wir die erarbeiteten Eigenschaften und stellen eine für diese Arbeit gültige Definition abwechslungsreicher Gebäuden auf. Dafür sind besonders die Eigenschaften relevant, die bei mehreren der betrachteten Beispiele Auftreten. Dabei unterscheiden wir in die Bereiche Aufbau, Fassade und Innenraum. Anhand der Beispiele werden die Bereiche Aufbau und Fassade definiert, der Bereich Innenraum wird daran anschließend in einem eigenen Unterpunkt erstmals analysiert und definiert.

1.4.1 Aufbau

In der folgenden Tabelle 4 sind die bereits erarbeiteten, möglichen Eigenschaften abwechslungsreicher Gebäude, welche den Aufbau betreffen noch einmal mit ihrem Auftreten zusammengefasst.

[1] 3D Animationssoftware von Side Effects.

Eigenschaften	**Stabkirchen**	**SKYRIM**	**Lake-Town**
Aufbau			
Holzbauweise	X	X	X
Eckmasten	X	X	X
Holzstege und Brücken		X	X
Rundbögen			X
Grundriss			
Symmetrisch	X	X	X
Asymmetrisch		X	X
Komplexe	X	X	
Einfache		X	X
Stockwerke			
Mehrere	X	X	X
Unterschiedlich große	X		X
Zunehmend kleiner werdend	X		
Zunehmend größer werdend			X
Überhängende		X	X

Tabelle 4: Zusammenfassung aller erarbeiteten Eigenschaften die den Aufbau eines abwechslungsreichen Gebäudes charakterisieren können und deren Auftreten in der Architektur der Beispiele. Je nach Häufigkeit des Auftretens in den analysierten Beispielen sind die Einzelnen Eigenschaften farblich hervorgehoben. Dunkelgrau = Immer. Hellgrau = Mehrheit. Quelle: Eigens erstellte Tabelle.

Daraus kann abgeleitet werden, dass es bis auf die Grundlegenden Eigenschaften der Holzbauweise mit Eckmasten, einem Symmetrischen Grundriss und mehreren Stockwerken keine Eigenschaften gibt, die so allgemeingültig sind, dass sie in allen betrachteten Beispielen vorkommen.

Da wir Gebäude für die Verwendung in Spielen generieren, sind die Eigenschaften SKYRIMS und Lake-Towns von höherer Priorität als die der Stabkirchen. Dabei haben die Eigenschaften der Stadt Lake-Town den Vorrang vor den Gebäuden SKYRIMS. Dies liegt darin begründet, dass in Filmen aufgrund des Renderings[2] ein höherer Grafikstandard gegeben ist als bei Spielen. Dies führt dazu, dass im Besonderen die Eigenschaften, welche in SKYRIM sowie Lake-Town auftreten noch hinzugenommen werden können.

[2] Filme werden im Gegensatz zu Spielen nicht zur Laufzeit auf der Maschine des Nutzers gerendert, sondern während der Entwicklung vorgerendered und später nur wiedergegeben. Dies macht die Qualität des Renderings bei Filmen unabhängig von der Qualität des Wiedergabegerätes.

Somit wird der Aufbau eines abwechslungsreichen Gebäudes durch die folgenden Eigenschaften definiert:

- Das Gebäude ist in einer Holzbauweise mit Eckmasten angelegt.
- Es können Stege und Brücken als Deko-Element hinzugenommen werden.
- Der Grundriss
    - kann aus einer komplexen, asymmetrische Form bestehen.
    - oder aber eine einfache symmetrische Form haben[3].
- Die Stockwerke
    - treten immer in größerer Zahl auf (mindestens 2).
    - besitzen unterschiedliche Größen.
    - stehen über den Grundriss hinaus.

1.4.2 Fassade

In der folgenden Tabelle 5 sind noch einmal die möglichen Eigenschaften der Fassaden abwechslungsreicher Gebäude zusammen mit ihrem Auftreten vermerkt.

Eigenschaften	**Stabkirchen**	**SKYRIM**	**Lake-Town**
Fassade			
Aufwendig gestaltet	X	X	X
Einfach gestaltet		X	X
Stützpfeiler für Überhänge		X	X
Haustüren in oberen Etagen			X
Anbauten und Erker		X	X
Dächer			
Mit Schuppen gedeckt	X	X	X
In die Höhe gestaffelt	X		X
Verschieden hohe Dächer	X	X	X
Unterschiedliche Dachrichtungen	X	X	X
An den Seiten überhängend	X	X	X

[3] Vorausgesetzt sie besitzt eine komplexe Fassade, die die Monotonie des Grundrisses aufhebt.

Dachfenster	X	X	X
Dachverzierung			
Drachenköpfe	X	X	X
Firstkämme	X		X
Wände			
Wandbelag Bretter	X	X	X
Wandbelag Schuppen	X		X
Windschief			X

Tabelle 5: Zusammenfassung der erarbeiteten Eigenschaften, welche die Fassade eines abwechslungsreichen Gebäudes charakterisieren und deren Auftreten in den Beispiele. Je nach Häufigkeit des Auftretens in den analysierten Beispielen sind die Einzelnen Eigenschaften farblich hervorgehoben. Dunkelgrau = Immer. Hellgrau = Mehrheit. Quelle: Eigens erstellte Tabelle.

In der Fassadengestaltung und insbesondere den Dächern gleichen sich die Beispiele sehr stark. Aus diesem Grund können fast alle erarbeiteten Eigenschaften in die finale Definition der Fassade übernommen werden. Lediglich zwei Eigenschaften Lake-Towns treten in keinem anderen Beispiel auf, dies sind die windschiefen Wände und die Möglichkeit eine Haustür in einem der oberen Stockwerke zu platzieren. Da jedoch diese beiden Eigenschaften maßgeblich zu dem Abwechslungsreichtum eines Gebäudes mit einfachem Aufbau beitragen können, werden sie noch mit in die Definition mit aufgenommen. Die Anbauten, Erker und Überhänge wurden bereits als Eigenschaften des Aufbaus definiert, woraus folgend sie nicht mehr als Eigenschaft der Fassade zur Verfügung stehen. Woraufhin die Definition wie folgt lautet:

- Die Fassade
    - sollte aufwendig gestaltet sein, wenn der Aufbau einfach ist.
    - sollte einfach sein, wenn der Aufbau aufwendig ist
    - kann eine Haustür in einem oberen Stockwerk besitzen.
- Die Dächer
    - sind mit schuppen gedeckt und verziert
    - haben unterschiedliche Höhen und Richtungen.
    - stehen an den Seiten des Gebäudes leicht über.
    - können Dachfenster besitzen.
- Die Wände
    - sind mit unterschiedlichen Wandbelägen gestaltet.
    - können windschief sein.

1.4.3 Innenräume

Die Innenraumgestaltung besitzt in der Architekturanalyse eine Sonderstellung, da wir schöne/abwechslungsreiche Räume generieren wollen, aber dieser Begriff im Zusammenhang mit Räumen zu schwammig und ungenau ist um ihn näher zu definieren. Außerdem ist die Einteilung der Innenräume stark von dem späteren Nutzen des Gebäudes abhängt. Besonders in Spielen kann dabei das Game Design maßgeblich beeinflussen, wie viele Räume es geben wird, und wie sie angeordnet sein müssen. Aus diesem Grund beschränken wir uns darauf eine funktionale Innenraumgestaltung anzustreben.

Somit stellen wird keine Definition für einen abwechslungsreichen Lageplan auf, sondern eine Liste funktionaler Merkmale die jeder Raum erfüllen muss. Diese Merkmale garantieren eine glaubhafte Innenraumgestaltung unabhängig von der Anzahl und Platzierung der Räume. Diese lauten wie Folgt:

Jeder Raum

1. ist von der Haustür aus durch angemessen viele Räume zu erreichen.
2. besitzt mindestens ein sinnvoll platziertes Fenster.
3. hat eine angemessene Mindestgröße.

Eines der Grundlegendsten Merkmale eines Lageplans ist, dass jeder Raum erreichbar sein muss. Ein Raum der nicht betreten werden kann besitzt keinen Nutzen und somit keinen Grund zu existieren. Daraus folgend müssen auch alle Stockwerke mit Treppen untereinander verbunden werden um die oberen Räume von der Haustür aus zugänglich zu machen. Ein weiterer Punkt ist die Platzierung der Fenster.

Es gibt nur wenige Räume, wie Flure oder Badezimmer, bei denen der Zugang zu Tageslicht optional ist. Da wir aber während der Generierung keine Aussage über den späteren Nutzen des Raumes treffen können, ist es nötig für jeden Raum den Zugang zu Tageslicht sicher zu stellen. Dies wird durch eine Mindestzahl von Fenstern pro Raum garantiert. Dabei werden die Fenster so platziert, dass sie Zugang zu Licht haben, also nicht auf gegenüberliegende Wände oder Dächer sehen.

Neben dem Zugang zu Licht ist es außerdem noch notwendig eine gewisse Mindestgröße der Räume zu fordern. Dies stellt sicher, dass die Größe eines Raumes seine Existenz rechtfertigt und er nicht als Bestandteil eines anderen Raumes funktionaler wäre.

2. Modell und Repräsentation

Nachdem wir in dem vorrangegangenen Kapitel festgelegt haben, welche Eigenschaften ein abwechslungsreiches Gebäude ausmachen, muss nun eine Datenstruktur gefunden werden, in der die Repräsentation dieses Gebäudes gespeichert werden kann. Dabei müssen alle Features des Gebäudes von der Aufteilung der Räume über Wände und Fenster bis zu den Dächern mit ihrer Dekoration gespeichert werden können. All diese Features sind in der folgenden Liste zusammengefasst und in Abbildung 23 am Beispiel eines Gebäudes visualisiert.

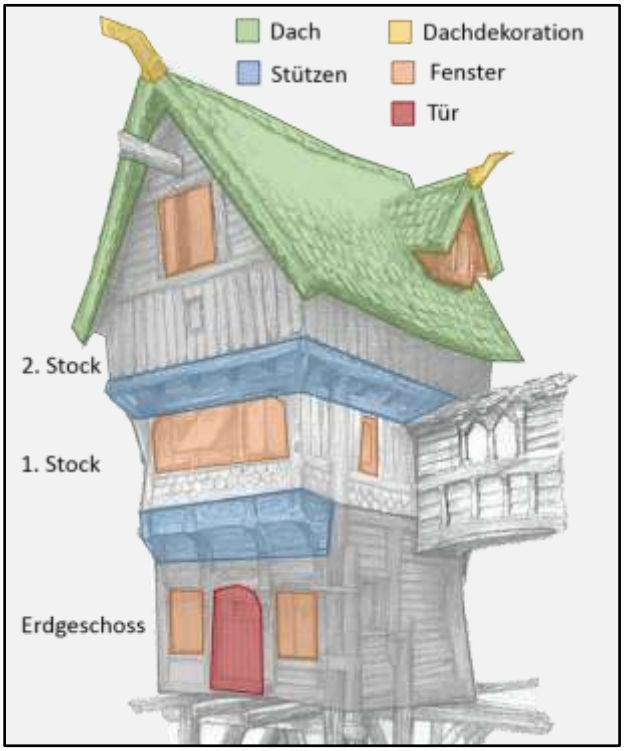


Abbildung 23: Visualisierung einiger in der Datenstruktur speicherbaren Features am Beispiel eines Gebäudes. Quelle: Erweiterte Grafik von John Howe [3].

- Innenraum und Außenraum
- Verschiedene Stockwerke
- Wände
- Stützen
- Dächer
- Dachdekoration
- Fenster und Dachfenster
- Türen
- Treppen

Da die Generierung von Gebäuden bestehend aus beliebigen Formen keine triviale Aufgabe ist und wahrscheinlich die am meisten verbreitetste Art von Grundrissen aus Rechtecken bestehen [13] werden die zu generierenden Gebäude wie folgt eingeschränkt um den Umfang dieser Arbeit nicht zu sprengen:

- Die Grundrisse bestehen ausschließlich aus Rechtecken, welche an den Achsen des Koordinatensystems ausgerichtet sind.

2.1 Ansatz

So wie bereits eine Vielzahl von verschiedensten Ansätzen zur Generierung von Gebäudefassaden [3] [5], Innenräumen [7] [9] [8] oder Städten [12] [4] [2] [10] existiert, wurden auch diverse Ansätze zur Repräsentation der generierten Häuser entwickelt. Die Komplexität der verschiedenen Repräsentationen reichen dabei von Shape Grammars [4] bis zu der Entwicklung einer eigenen Programmiersprache zur Beschreibung der Modelle [18]. Neben diesen sind auch die Dreiecksnetze zu erwähnen, da sie sehr verbreitet sind, aber nur für Visualisierungszwecke geeignet sind. Jedes Modell ist dabei genau auf den konkreten Anwendungsfall zugeschnitten und oft auf die Generierung der Fassade oder des Innenraumes spezialisiert. Diese Arbeit schlägt eine Repräsentation vor, die sowohl die Fassade als auch den Innenraum eines Gebäudes darstellen kann. Dazu wird eine erweiterte Form des Voxel-Modells verwendet.

2.2 Erweitertes Voxel-Modell

Da die Grundrisse der zu generierenden Häuser immer aus Rechtecken bestehen, bietet es sich an ein Modell zu wählen, welches das Gebäude in Würfel zerlegt. Dies ermöglicht die Darstellung des Gebäudevolumens als ein gleichförmiges, dreidimensionales Voxel-Gitter. Dies hat allerdings den Nachteil, dass keine Zylinder oder andere, nicht auf Würfeln basierende Körper, oder Rotationen dargestellt werden können.

Folglich können die Dächer der Gebäude nicht in Form eines Prismas gespeichert werden. Um dieses Problem zu umgehen werden verschiedene Voxel-Typen eingeführt um als Platzhalter für bestimmte Gebäudestrukturen wie Dächer oder Stützen zu fungieren. In diesen werden die zu Laufzeit benötigten Daten für die Rekonstruktion der einzelnen Gebäudestrukturen gespeichert. Ein Beispiel hierfür ist die Darstellung eines Dachs in Form einer Schicht von Dach-Voxeln auf der Oberseite des Gebäudes. In diesen Voxeln sind alle notwendigen Eigenschaften, wie Dachrichtung und Höhe vermerkt um das Dach zur Laufzeit zu rekonstruieren.

Nachdem nun das Volumen des Gebäudes anhand von Voxeln dargestellt werden kann, kommen wir zu den Details der Gebäudefassade in Form von Fenstern und Türen. Um diese repräsentieren zu können, ist es nötig das Voxel-Modell zu erweitern. Dazu wird jeder der 6 Voxel-Seiten zusätzlich ein Wert zugewiesen. Dies ermöglicht es Fenster und Türen, sowie diverse Details der Fassade auf den jeweiligen Seiten des Voxel zu speichern, siehe Abbildung 24.

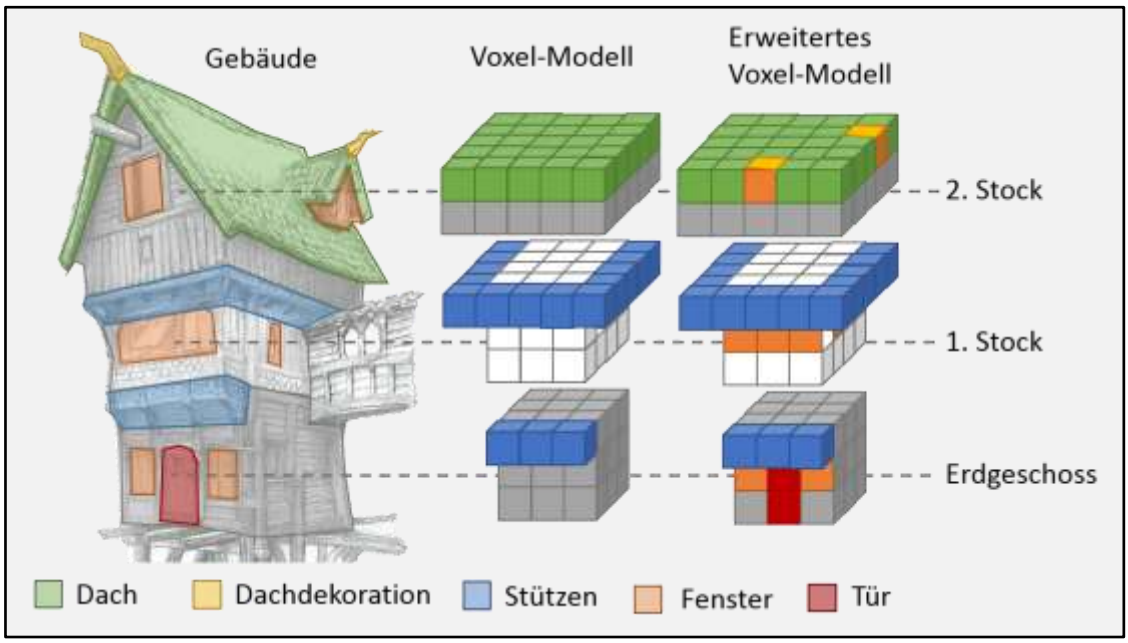


Abbildung 24: Repräsentation eines Gebäudes als Voxel-Modell und erweitertes Voxel-Model. Quelle: Eigens erstellte Grafik auf Basis eines Gebäude-Konzepts von John Howe [3].

Dazu wird auf den Seiten der äußeren Voxel vermerkt, dass es sich bei diesen Seiten um eine Wand, ein Fenster oder ein anderes Detail der Fassade handelt.

Durch die selbe Vorgehensweise werden auch die Innenräume des Gebäudes parallel zur Gebäudefassade dargestellt. Hierzu wird in jedem inneren Voxel, der an eine der Räume angrenzt, auf der entsprechenden Voxel-Seite eine Wand vermerkt, siehe Abbildung 25. Dies bedeutet, das eine innere Wand immer in beiden angrenzenden Voxeln eingetragen wird. Womit die Innenräume indirekt durch die Platzierung der Wände in den jeweils angrenzenden Voxeln definiert werden.

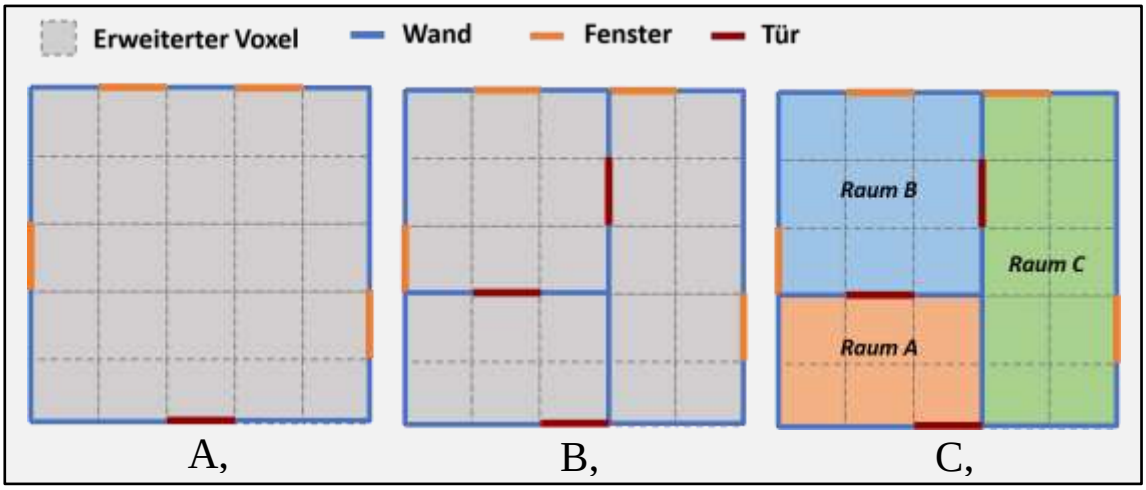


Abbildung 25: Aufsicht auf den Grundriss eines Gebäudes im erweiterten Voxel-Modell zur Veranschaulichung der Innenraum-Repräsentation. In Teil A ist nur die Fassade des Gebäudes in Form der Wände, Fenster und der Tür zu sehen. In Teil B wurden die inneren Wände in das erweiterte Voxel-Modell hinzugefügt, woraus indirekt die Räume in Teil C definiert werden. Quelle: Eigene Grafik.

2.3 Datenstruktur

In den folgenden Kapiteln wird vorausgesetzt, dass der Leser mit dem Konzept der Vererbung in objektorientierten Programmiersprachen vertraut ist, sowie der Anordnung von Objekten in Listen. Um den Datentyp umsetzen zu können wird der Gebrauch einer objektorientierten Programmiersprache empfohlen. Es kann alternativ auch in einer nicht objektorientierten Sprache gearbeitet werden, solange in dieser ein mit der Vererbung in objektorientierten Sprachen vergleichbares Konzept und Listen existieren.

Das erweiterte Voxel-Modell setzt sich aus einer Basisklasse von Voxeln, bestehend aus 6 Voxel-Seiten, zusammen. Von diesen Basis-Voxeln erben die einzelnen Struktur-Voxel, welche einen nicht als Würfel darstellbaren Gebäudeteil repräsentieren und alle zur Rekonstruktion dieser Struktur nötigen Daten enthalten, vergleiche Abbildung 26. Zu diesen Gebäudestrukturen zählen neben den Dächern ebenfalls die Stützen für die über den Grundriss hängenden Gebäudeteile, sowie die Treppen zur Verbindung der einzelnen Stockwerke. Diese werden zur Laufzeit von einem Algorithmus rekonstruiert oder durch vorgefertigte Modelle ersetzt. Dabei ist in dem Struktur-Voxel gespeichert, wie er jeweils zu rekonstruieren ist.

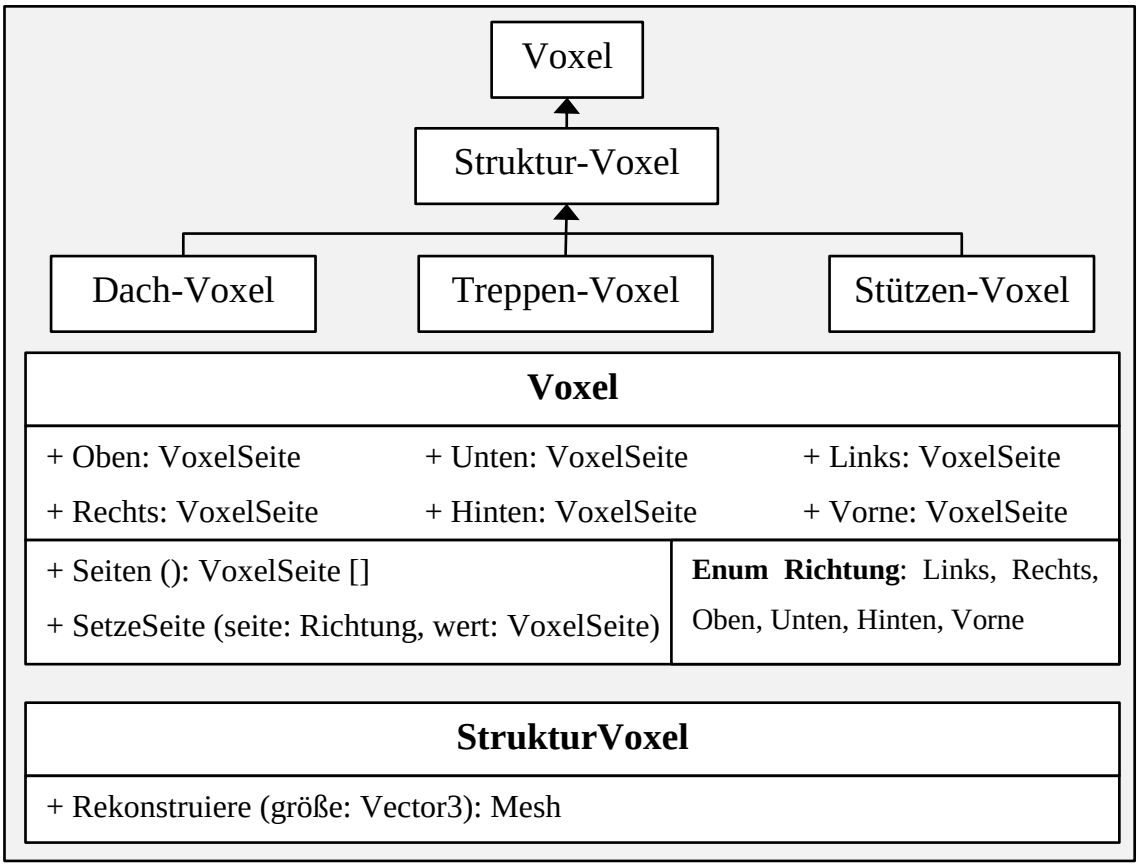


Abbildung 26: Vererbungs-Hierarchie der Voxel und Struktur-Voxel zur Verdeutlichung des Aufbaus. Quelle: Eigene Grafik.

Die einzelnen Voxel-Seiten dienen dabei der Darstellung der Gebäudefassadendetails in Form von Wänden, Fenstern und Türen. Dabei können einzelne Fassadendetails auch mehrere angrenzende Voxel-Seiten belegen um zum Beispiel hohe Fenster darzustellen. Außerdem übernehmen die Voxel-Seiten die Repräsentation der inneren Wände des Gebäudes und definieren so indirekt die Innenräume des Gebäudes. Dabei werden die Innenraumstrukturen, wie Wände und Türen, immer in beiden an die Wand angrenzenden Voxeln eingetragen. Diese redundante Speicherung vermeidet potenzielle Sonderfälle im Umgang mit Innenwänden. Zu diesen zählen:

- **Vertikale, nicht fortlaufende Wände.** Dabei befindet sich der nächste Abschnitt einer vertikal laufenden Wand nicht wie erwartet in dem darüberliegenden Voxel, sondern ist einen Voxel nach links oder rechts verschoben. Vergleiche Abbildung 27 – A.

- **Horizontale, nicht fortlaufende Wände.** Hier tritt ein ähnliches Phänomen wie bei vertikalen Wänden auf. Dabei ist der nächste Wandabschnitt nicht in dem rechts oder links liegenden Voxel eingetragen, sondern ist um eins nach oben oder unten verschoben Voxel. Vergleiche Abbildung 27 – B.

- **Ecken.** Bei einer Ecke müssen bis zu 4 angrenzende Voxel mit all ihren Seiten überprüft werden um eine Aussage über den Verlauf der Ecke treffen zu können. Eine Beispielhafte Konfiguration ist in Abbildung 27 – C zu sehen.

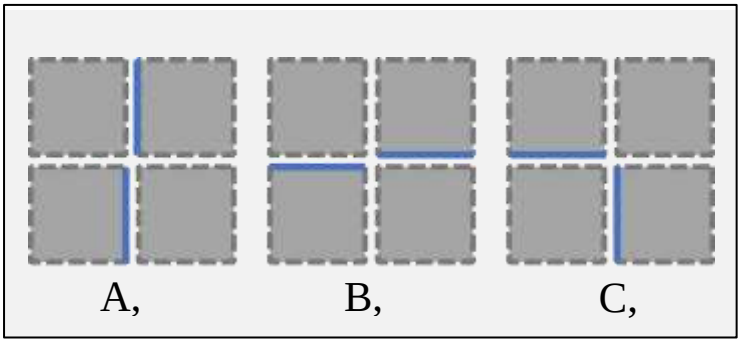


Abbildung 27: Veranschaulichung der verschiedenen Sonderfälle bei Innenwänden, wenn sie nicht in beiden angrenzenden Voxeln gespeichert werden. A. Vertikale, nicht fortlaufende Wand. B: Horizontale, nicht fortlaufende Wand. C: Verschobene Ecke. Quelle: Eigene Grafik.

Ein Gebäude setzt sich dabei aus einer Collection von Stockwerken zusammen, welche wiederum aus einem dreidimensionalen Voxel-Gitter bestehen. Die Stockwerke werden dabei als dreidimensionales Gitter gespeichert um Stockwerke unterschiedlicher Höhen in einem Gebäude zu ermöglichen. So ist durch die 3. Dimension des Gitters immer bekannt, wie hoch das jeweilige Stockwerk ist. Dabei besitzt jedes Stockwerk-Voxel-Gitter eines Gebäudes dieselbe Breite und Tiefe. Das Voxel-Gitter muss dabei allerdings nicht komplett mit Voxeln belegt sein. Die tatsächliche Größe der Stockwerke kann sich weiterhin unterscheiden.

Dieses Vorgehen dient dazu die Anordnung der einzelnen Stockwerke übereinander zu erleichtern, da es ansonsten schwer fallen würde Stockwerke unterschiedlicher Breiten und Tiefen richtig übereinander zu platzieren. Dies erhöht den Speicherbedarf der Repräsentation um 48 Byte pro ungenutztem Voxel[4]. Wobei ein Struktur-Voxel je nach Typ mehr Bytes für seine Rekonstruktions-Variablen benötigt als ein normaler Voxel. Umso weniger Voxel in einem ausladenden Grundriss genutzt werden, umso mehr fällt der zusätzliche Speicherverbrauch ins Gewicht. Im Extremfall eines lichten Gebäudes kann sich der tatsächliche Speicherverbrauch auf einen linearen Faktor verringern, vergleiche Abbildung 28. Dabei wird jedoch auf Grund der Repräsentation das Vielfache des tatsächlich benötigten Speichers reserviert.

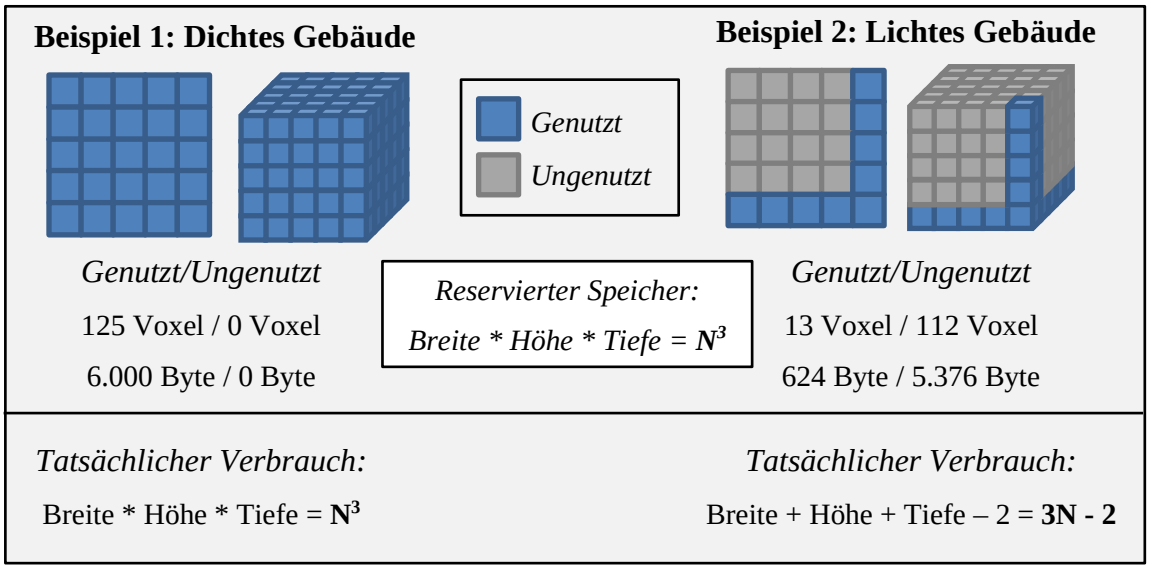


Abbildung 28: Veranschaulichung des tatsächlichen Speicherverbrauchs zweier Gebäude. Zu sehen sind der Grundriss sowie das dreidimensionale Voxel-Gitter der Gebäude. Bei dem Extremfall eines lichten Gebäudes kann der Speicherverbrauch auf einen linearen Faktor sinken. Das Gebäude besitzt in der Repräsentation aber weiterhin einem Speicherverbrauch von N^3. Quelle: Eigene Grafik.

[4] Ein Voxel besitzt 6 Voxel-Seiten welche jeweils 8 Byte für die Referenz beanspruchen. Somit ergibt sich ein Speicherverbrauch von 48 Byte per Voxel.

Da wir abwechslungsreiche Gebäude mit einem aus mehreren Rechtecken bestehenden Grundriss und unterschiedlich großen Stockwerken generieren, ist es zudem sehr unwahrscheinlich, dass wir alle Voxel des Gitters voll belegen. Woraus folgend der Speicher nie voll ausgenutzt wird.

Diese ist in diesem Fall jedoch durch die ansonsten entstehenden, schwer zu behandelnden Sonderfälle der Stockwerkplatzierung gerechtfertigt. Dabei werden folgende Problematiken von vornherein umgangen:

- Die Platzierung eines Stockwerks mit ungerader Breite auf einem mit Gerader Breite. Es ist nicht bekannt, ob das Stockwerk in diesem Fall einen Voxel nach rechts oder links verschoben platziert werden soll.

- Die Platzierung kleinerer Stockwerke auf Größeren. Eine Zentrierung des Stockwerks auf das darunterliegende ist nicht immer gewünscht, da es die Platzierung kleinerer oberer Stockwerke an den Rändern des Gebäudes oder das Erstellen von seitlichen Überhängen unmöglich macht.

Somit wird ein Gebäude durch eine Liste von Stockwerken repräsentiert, welche wiederum aus einer dreidimensionalen Collection erweiterter Voxeln bestehen. Diese besitzen jeweils 6 Voxel-Seiten, auf welchen die Details des Gebäudes platziert werden.

Abschließend ist noch einmal zu bemerken, dass das Voxel-Modell als interne Repräsentation des Gebäudes im Algorithmus dient und die graphische Darstellung auch durch einen zusätzlichen Algorithmus übernommen werden kann, welcher die in den Voxeln gespeicherten Daten interpretiert und in ein 3D-Modell umwandelt.

3. Evaluierung

Ziel dieses Kapitel ist es Methoden vorzustellen, mit welchen die in Kapitel *1. Architekturanalyse* erarbeiteten Eigenschaften abwechslungsreicher Gebäude ermittelt und bewertet werden können. Auf Basis dieser Methoden wird anschließend ein Bewertungs-Algorithmus erarbeitet um die Qualität der generierten Gebäude zu ermitteln.

Dieser Algorithmus ermöglicht es, die Generierung zur Laufzeit durch sein Feedback in Richtung abwechslungsreicherer Gebäude zu beeinflussen. Außerdem legt dies die Grundlagen um den in Kapitel 4 vorgestellten Generierungsalgorithmus in Zukunft durch einen lernfähigen Algorithmus ersetzen zu können. Bei der Evaluierung unterscheiden wir in die Bereiche äußere Gestaltung (Grundrisse, Fassaden) und Innenräume.

3.1 Äußere Gestaltung

Im Bereich der äußeren Gestaltung werden zwei verschiedene Ansätze umgesetzt um ein abwechslungsreiches Gebäude zu erstellen:

1. Ein Gebäude mit einfachem Aufbau und aufwendig gestalteter Fassade.
2. Ein Gebäude mit komplexem Aufbau und einfach gehaltener Fassade.

Dabei wurde die Möglichkeit ein Gebäude mit einem komplexen Aufbau und einer Aufwendig gestalteten Fassade außen vorgelassen. Dieser Ansatz wirkt auf den ersten Blick sehr vielversprechend, ein solches Gebäude besitzt allerdings eine zu hohe visuelle Komplexität. Es wirkt unruhig und überladen auf den Betrachter. Um die Balance zwischen überladen und monoton zu halten, werden hier Ansätze mit einem monotonen und einem abwechslungsreichen Part verfolgt. Dies sorgt dafür, dass die Einfachheit des einen Gebäudebestandteils den Abwechslungsreichtum des Anderen betont und nicht mit ihm um die Aufmerksamkeit des Betrachters kämpft. Bei der Evaluierung der äußeren Gestaltung werden daraus folgend die oben genannten Kombinationen von Aufbau und Fassade als erstrebenswert betrachtet. Somit setzt sich die Bewertungs-Funktion der äußeren Gestaltung aus den einzelnen Ergebnissen der Aufbau- und Fassaden-Evaluierung zusammen, welche im Folgenden näher vorgestellt werden.

3.1.1 Aufbau

Bevor mit der Evaluierung des Aufbaus begonnen wird, werden die grundlegenden Eigenschaften eines korrekten Aufbaus betrachtet. Sind diese nicht gegeben, wird der Gebäudeaufbau als unzureichend bewertet. Das generierte Objekt ist somit nicht als Gebäude zu klassifizieren und in der Evaluierung durchgefallen. Diese Eigenschaften dienen vorrangig dem Erkennen von Generierungsfehlern und lauten wie folgt:

- Es existieren keine schwebenden Stockwerke.
- Alle über den Grundriss ragenden Gebäudeteile besitzen Stützen.
- Alle Gebäudebestandteile sind direkt miteinander verbunden.

3.1.1.1 Schwebende Stockwerke

Überhängende Stockwerke sind erwünscht, jedoch keine komplett schwebenden Stockwerke oder zu weit über den Grundriss ragende Gebäudeteile. Die Ebene eines Stockwerks gilt dabei als schwebend, wenn sie nicht mit mindestens 1/4 ihrer Fläche auf der darunterliegenden Ebene aufliegt. Außerdem müssen alle Gebäudeteile mit mindestens der Hälfte ihrer Fläche auf der unteren Ebene aufliegen, vergleiche Abbildung 29. Dabei sind breite Überhänge oberhalb des ersten Stockwerks verboten um das Gebäude weiterhin statisch korrekt wirken zu lassen.

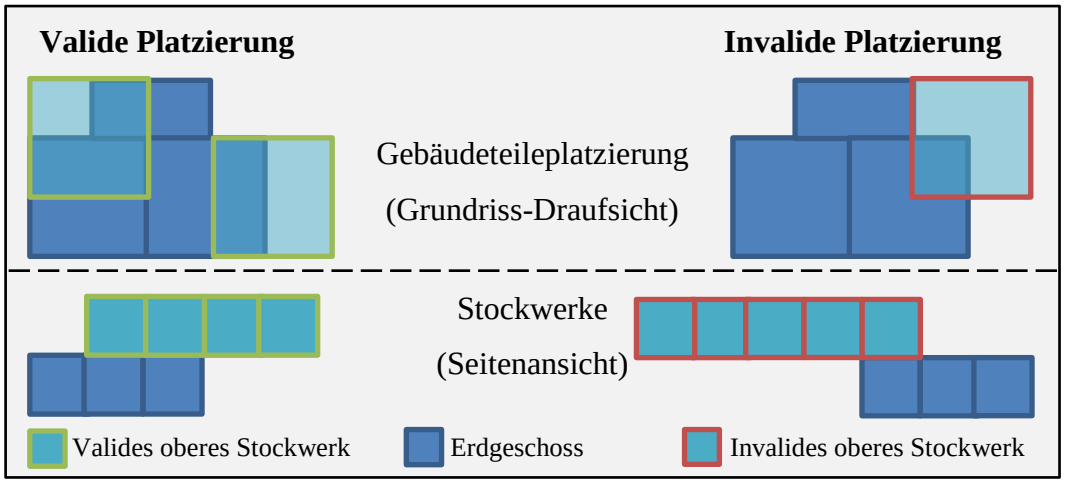


Abbildung 29: Verdeutlichung der validen und invaliden Platzierung von Stockwerken und Gebäudeteilen. Gebäudeteile müssen mit der Hälfte ihrer Fläche aufliegen, Stockwerke mit einem ¼ ihrer Fläche. Gebäudeteil-Ansicht: Draufsicht auf den Grundriss des Gebäudes. Stockwerk-Ansicht: Seitenansicht des Gebäudes. Quelle: Eigene Grafik.

Bei der Evaluierung wird zuerst überprüft, ob das gesamte Stockwerk schwebt, anschließend werden die einzelnen Gebäudeteile getestet. Schwebt das Stockwerk müssen zwangsläufig auch die Gebäudeteile schweben und deren Evaluierung erübrigt sich.

Die für die Evaluierung der Gebäudeteile nötigen Daten, wie Anzahl, Form und Position werden während der Generierung in den Stockwerken vermerkt. Dies ermöglicht es die benötigten Daten während der Evaluierung auszulesen. Dies erübrigt eine zeitintensive Rekonstruktion der einzelnen Gebäudeteile zur Laufzeit.

3.1.1.2 Überhänge ohne Stützen

Auch wenn kein Wert auf statische Korrektheit gelegt wird, ist es wichtig die Glaubwürdigkeit des Gebäudes aufrecht zu halten. Das Gebäude muss nicht statisch stehen können, aber es muss so aussehen, als würde es das.

Aus diesem Grund ist es notwendig zu prüfen, ob alle Überhänge angemessen Stützen stabilisiert wurden. Bei schmalen Überhängen von einem Voxel Breite muss jeder Voxel von einer kleinen Stütze gehalten werden. Bei breiten Überhängen reicht es, wenn die äußere Kontur der überhängenden Fläche durch kleine Stützen gehalten wird. Dabei muss jedoch jede Ecke des Überhanges von einer Säule gestützt werden, siehe Abbildung 30.

Die richtige Platzierung der Stützen wird durch die Kontrolle der schwebenden Voxel evaluiert. Die einzige valide Möglichkeit einen schwebenden Voxel zu platzieren ist, wenn er teil eines breiten Überhanges ist.

Zuerst werden alle schwebenden Voxel des Stockwerks gefunden und ermittelt, ob sie zu einem breiten Überhang gehören. Um dies zu ermitteln muss getestet werden, ob der Rand des Überhanges von Stützen eingeschlossen ist.

Dazu startet man von dem zu testenden Voxel aus eine geradlinige Suche in alle 4 Richtungen der Ebene. Bewegt man sich in jeder dieser Richtungen ausschließlich über

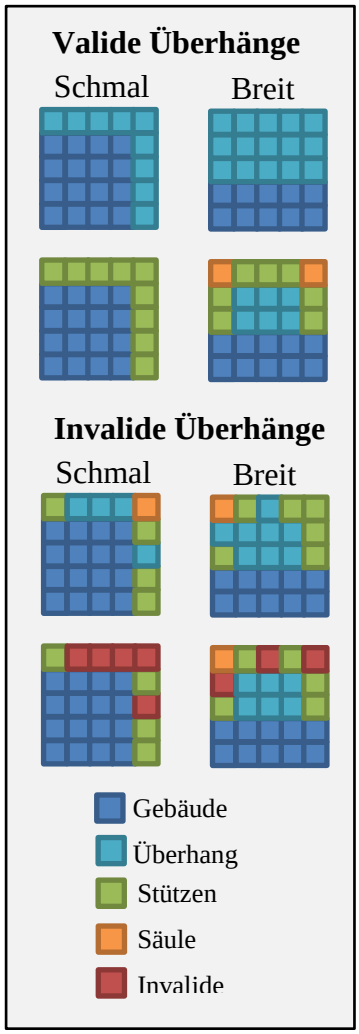


Abbildung 30: Beispiel für valide und invalide Platzierung von kleinen Stützen und Säulen an schmalen und breiten Überhängen. Draufsicht auf die Grundrisse. Quelle: Eigene Grafik.

schwebende Voxel bis man an einem gesicherten Voxel[5] endet, ist der zu testende Voxel Teil eines großen, richtig gesicherten Überhangs. Siehe Abbildung 31 für eine Pseudocode Implementierung.

```
IstVoxelTeilVonÜberhang(voxel, gebäude): Bool
        return ExistierenRänder in allen 4 richtungen(voxel)
-----------------------------------------------------------------
ExistiertRandInRichtung (voxel, richtung, gebäude): Bool
        nächsterVoxel: voxel
        nächsterVoxel = nächster voxel in der angegebenen richtung
        if nächsterVoxel liegt außerhalb der datenstruktur
                return false

        while nächsterVoxel existiert und schwebt
                nächsterVoxel = nächster voxel in der angegebenen richtung
                if nächsterVoxel liegt außerhalb der datenstruktur
                        return false

        return nächsterVoxel existiert und IstGesichert(nächsterVoxel)
-----------------------------------------------------------------
        IstGesichert (voxel): Bool
                return  unterer voxel ist Voxel oder Stütz-Voxel
```

Abbildung 31: Pseudocode zu der vorgestellten Methode zum Erkennen von Überhängen ohne Stützen. Dazu wird nachgewiesen, ob der schwebende Voxel zu einem großen, gesicherten Überhang gehört. Dies ist die einzige, den Regeln entsprechende Möglichkeit eines schwebenden Voxel. Quelle: Eigene Grafik.

Durch dieses Vorgehen werden ebenfalls nicht gesicherte, schwebende Voxel von schmalen Überhängen aufgespürt, da sie nicht an allen Seiten von Stützen umgeben sind.

Für den Nachweis der Säulen werden alle Stütz-Voxel daruf untersucht, ob sie die Ecke zweier aufeinandertreffender Stützreihen darstellen. Durch diesen Test werden die Ecken breiter, sowie schmaler Überhänge gefunden. Um die Ecken der breiten Überhänge von denen der Schmalen zu unterscheiden wird der zwischen den beiden Stützreihen liegende Voxel untersucht, der diagonal an die gefundene Ecke angrenzt, vergleiche Abbildung 32.

Existiert an der Position kein Voxel und ist der darüberliegende Voxel ein Gebäude-Voxel, so handelt es sich um die Ecke eines großen Überhanges. Der diagonal

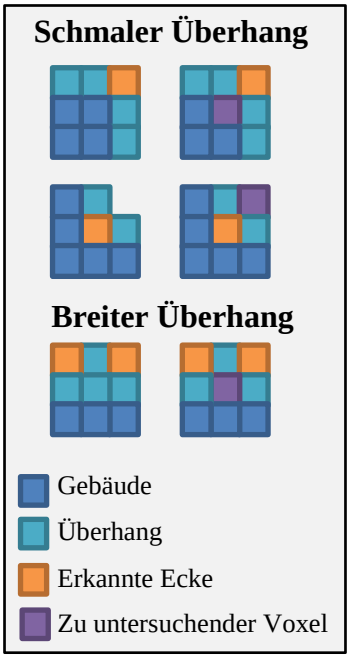


Abbildung 32: Veranschaulichung der gefundenen Ecken bei der Suche nach allen Säulen-Positionen und die daraufhin zu untersuchenden Voxel. Quelle: Eigene Grafik.

[5] Ein gesicherter Voxel liegt entweder auf einem Stütz-Voxel oder einem normalen Voxel der darunterliegenden Ebene auf.

angrenzende Voxel einer nach außen gerichteten Ecke eines schmalen Überhanges ist immer ein Gebäude-Voxel. Bei einer nach innen gerichteten Ecke eines schmalen Überhanges ist diese Position leer sowie die darüber liegende, vergleiche Abbildung 33. Sind die Ecken der breiten Überhänge gefunden, wird überprüft, ob alle unter der Ecke liegenden Voxel Stütz-Voxel sind und somit eine valide Säule bilden.

Zuordnung einer Ecke zu einem schmalen oder breiten Überhang

Untersuchte Voxel:	Diagonal an die Ecke angrenzender Voxel Voxel über dem diagonal angrenzenden Voxel		
Mögliche Zustände:	Gebäude-Voxel Egal	Leer Gebäude-Voxel	Leer Leer
Ergebnis:	Schmaler Überhang Äußere Ecke	Breiter Überhang	Schmaler Überhang Innere Ecke

Abbildung 33: Zusammenfassung der möglichen Zustände der für die Zuordnung der Ecke relevanten Voxel und den jeweiligen Ergebnissen der Zuordnung. Quelle: Eigene Grafik.

3.1.1.3 Losgelöste Gebäudeteile

Losgelöste Gebäudeteile sind im Gegensatz zu den beiden vorrangegangenen Eigenschaften kein Merkmal, welches das generierte Objekt als Gebäudekomplex disqualifizieren würde. Viele größere Gebäudekomplexe besitzen auf ihrem Grundstück kleinere freistehende Gebäude. Genauso sind einzelne Türme in den oberen Etagen eines Gebäudes valide. Da der im weiteren Verlaufe dieser Arbeit vorgestellte Generierungsalgorithmus aber anstrebt einen zusammenhängenden Gebäudekomplex zu generieren, ist dies als Fehler der Generierung zu klassifizieren. Ein zusammenhängendes Gebäude zeichnet sich folglich dadurch aus, dass alle existierenden Voxel miteinander verbunden sind.

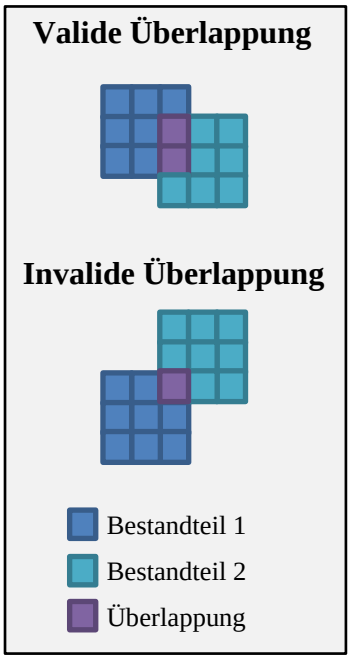


Abbildung 34: Beispiele für eine valide und invalide Verbindung von zwei Gebäudebestandteilen in Form von Überlappungen. Quelle: Eigene Grafik.

Genauso müssen alle Gebäudebestandteile eines Stockwerks miteinander verbunden sein. Diese gelten als verbunden, wenn sie mit mindestens 3 Voxel-Seiten an ein anderes Gebäudbestandeteil angrenzen. Die Beschränkung auf 3

Voxel-Seiten verhindert dabei unschöne Verbindungen, bei denen sich die Bestandteile nur an einer Ecke überlappen, vergleiche Abbildung 34.

Betrachten wir das Voxel-Gitter des Gebäudes als ungerichteten Graph, bei dem die einzelnen Voxel die Knoten bilden und jeder Knoten Kanten zu seinen im Voxel-Gitter direkt angrenzenden 6 Nachbarn besitzt. So zeichnet sich der Graph eines zusammenhängenden Gebäudes dadurch aus, das von jedem beliebigen Knoten des Graphen alle restlichen Knoten erreicht werden können. Dies bedeutet, dass der Graph im Sinne der Graphentheorie ebenfalls zusammenhängend ist. Daraus folgend kann der Zusammenhang der Gebäude-Voxel durch das Prüfen des Gebäude-Graphen auf seinen Zusammenhang bestätigt werden.

In der Graphentheorie gibt es verschiedene Ansätze um den Zusammenhang eines Graphen zu überprüfen. Einer dieser ist es den Graphen beginnend an einem beliebigen Knoten mit einer Breiten- oder Tiefensuche komplett zu durchlaufen und die Anzahl der besuchten Knoten zu speichern. Ist die Anzahl der besuchten Knoten mit der Anzahl der Knoten des Graphen identisch, ist der Graph zusammenhängend [19].

Folglich wird das Voxel-Gitter des Gebäudes in einen Graphen umgewandelt und anschließend auf dessen Zusammenhang mit der genannten Methode überprüft. Ist der

```
IstZusammenhängend(gebäude): Bool
        gefundeneVoxel: int
        startVoxel: Voxel
        warteschlange: Voxel Collection
        besuchteVoxel: Voxel Collection

        startVoxel = ein existierender voxel des gebäudes
        füge startVoxel zu warteschlange  hinzu

        while warteschlange ist nicht leer
                aktuellerVoxel: Voxel
                aktuellerVoxel = entferne voxel vom anfang der warteschlange
                füge aktuellerVoxel zu besuchteVoxel hinzu

                foreach existierenden nachbar voxel von aktuellerVoxel
                        if voxel ist noch nicht besucht oder schon in warteschlange
                                füge voxel zum ende der warteschlange hinzu

                gefundeneVoxel + 1

        return gefundeneVoxel ist genauso viel wie die voxel des gebäudes
```

Abbildung 35: Pseudocode Implementierung der Methode zum Nachweis eines zusammenhängenden Gebäudes. Quelle: Eigene Grafik.

Graph zusammenhängend, besitzt das Gebäude keine losgelösten Voxel. Vergleiche Abbildung 35 für eine beispielhafte Pseudocode Implementierung. Die selbe Methode wird anschließend verwendet um die Bestandteile der einzelnen Stockwerke auf ihre Zusammengehörigkeit zu prüfen. Dabei werden anstelle aller Gebäude-Voxel nur die Voxel des Stockwerkes und deren Verbindungen zu ihren Nachbarn verwendet.

Als letzter Punkt in der Evaluierung der Grundvoraussetzungen des Aufbaus wird der Sonderfall von Löchern in den generierten Stockwerken behandelt. Größere Löcher innerhalb eines Gebäudes bilden valide Innenhöfe. Kleine Löcher von einem Voxel Größe zählen dagegen zu den Generierungsfehlern. Wenn ein leerer Voxel auf seiner Ebene von 4 Voxeln umgeben ist, gilt er als Loch.

Nachdem der Aufbau des Gebäudes als korrekt charakterisiert wurde, folgt die Bestimmung der Komplexität. Die für diese Bewertung der Komplexität wichtigen Faktoren lauten wie folgt:

- Anzahl und Anordnung der Rechtecke, aus welchen die Grundrisse bestehen.
- Anzahl und Größe der Stockwerke.
- Anzahl der Überhänge in den Stockwerken.

3.1.1.4 Grundrissrechtecke

Bei der Betrachtung der Grundrisse ist die Anzahl der verwendeten Rechtecke zur Generierung des Grundrisses direkt proportional zu der Komplexität des Grundrisses. Umso mehr Rechtecke bei der Generierung verwendet wurden, umso komplexer der Grundriss.

Da es die Implementierung erleichtert, sowie die Laufzeit verkürzt, wird die Anzahl der Rechtecke des Grundrisses zur Generierungszeit gespeichert, anstatt sie später zu rekonstruieren. So kann die Anzahl der Rechtecke in der Evaluierung nachgesehen werden.

Ein weiterer Bestandteil der Grundrissrechtecks-Evaluierung ist die Anordnung der Rechtecke im Hinblick auf die Symmetrie des Grundrisses. Ein auf beiden Achsen symmetrischer Grundriss gilt hierbei als monotoner als ein asymmetrischer Grundriss. Um dies zu überprüfen werden zuerst die horizontale und vertikale Achse des Gebäudes ge-

funden und anschließend die Voxel-Anordnung der beiden Achsenseiten miteinander verglichen. Ist die Voxel-Anordnung auf beiden Seiten der Achse gleich, gilt der Grundriss auf dieser Achse als symmetrisch.

Da der Grundriss des Gebäudes im Voxel-Gitter nicht immer zentriert ist, ist es wichtig die möglichen Symmetrieachsen in der tatsächlichen Mitte des Grundrisses zu platzieren. Um zum Beispiel die Position der vertikalen Symmetrieachse zu ermitteln, werden die Koordinaten der am weitesten links und rechts gelegenen Voxel verwendet. Die Mitte der Strecke zwischen diesen Voxeln bildet die vertikale Mitte des Grundrisses und somit die Position der vertikalen Symmetrieachse, vergleiche Abbildung 36.

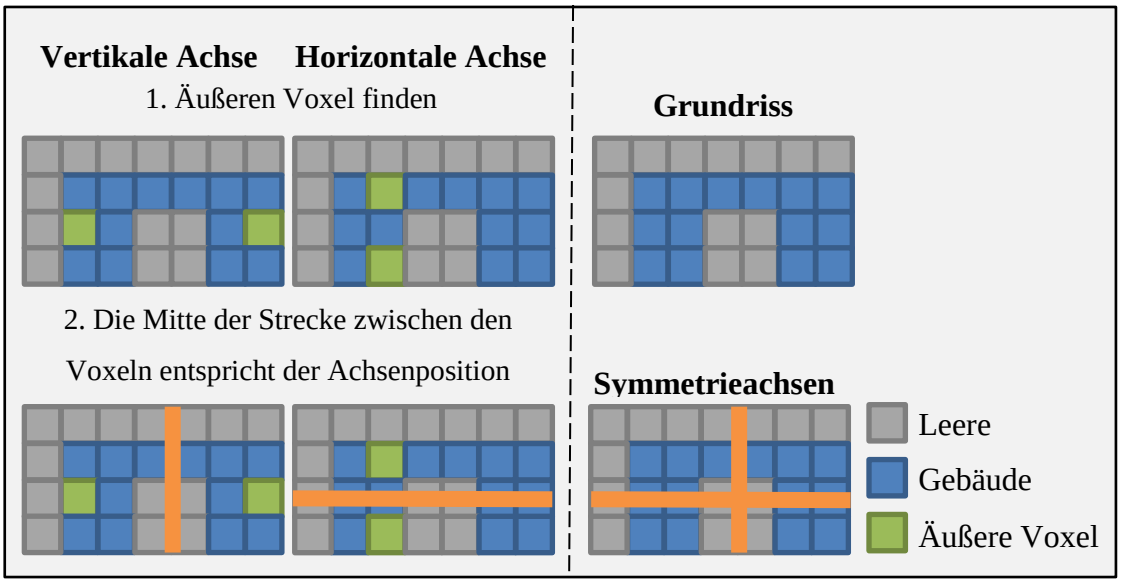


Abbildung 36: Visualisierung des Vorgehens zum Finden der Symmetrieachsen eines Grundrisses. Quelle: Eigene Grafik.

3.1.1.5 Stockwerke

Wie in der Architekturanalyse schon festgestellt, wirken Gebäude mit vielen, unterschiedlich großen Stockwerken komplexer und somit abwechslungsreicher. Dabei gelten Gebäude mit Stockwerken, dessen Größen weit voneinander abweichen als abwechslungsreicher als Stockwerke mit einer geringen Größenabweichung.

Um diese Eigenschaft nachzuweisen werden die Voxel der einzelnen Stockwerkgrundrisse gezählt und verglichen. Weichen die Werte weit voneinander ab, gelten die Stockwerkgrößen als abwechslungsreich. Dabei ist es irrelevant, ob die Stockwerke mit zunehmender Anzahl größer oder kleiner werden.

3.1.1.6 Überhänge

Die Anzahl der Überhänge beschreibt in diesem Fall die Anzahl der Voxel, welche über den Grundriss des darunterliegenden Stockwerkes hinausragen. Da Überhänge immer gestützt werden und die einzige valide Position eines schwebenden Voxel in einem großen Überhang ist, bilden alle schwebenden Voxel zusammen mit allen von einem Stütz-Voxel gesicherten Voxeln die Menge der überhängenden Voxel. Woraus folgend sich die überhängenden Voxel wie in Abbildung 37 beschrieben im Pseudocode nachweisen lassen.

```
ÜberhängendeVoxel(gebäude): Voxel Collection
    überhängendeVoxel: Voxel Collection
    foreach voxel des gebäudes
        if voxel schwebt oder liegt über einem Stütz-Voxel
            überhängendeVoxel + voxel
    return überhängendeVoxel
```

Abbildung 37: Pseudocode Implementierung der vorgestellten Methode zum Finden aller überhängenden Voxel. Quelle: Eigene Grafik.

3.1.3 Fassade

Wie bereits bei der Evaluierung des Aufbaues werden zuerst die Eigenschafften einer korrekten Fassade näher betrachtet, bevor auf die einzelnen Methoden zur Bestimmung der Komplexität der Fassade eingegangen wird. Treffen die folgenden Eigenschaften nicht auf die Bestandteile der Gebäudefassade zu, gilt diese als unzureichend und das Gebäude hat die Evaluierung der Fassade nicht bestanden.

- Alle Fenster sind sinnvoll platziert.
- Die Haustür ist erreichbar.

3.1.3.1 Fensterplatzierung

Bevor die Methode zum Nachweisen der sinnvollen Platzierung eines Fensters aufgezeigt werden kann, ist es notwendig den ungenauen Begriff sinnvoll im Zusammenhang mit der Platzierung eines Fensters zu definieren. Ein Fenster gilt dann als adäquat platziert, wenn es nicht gegen ein angrenzendes Dach blickt und einen direkten Zugang zu Sonnenlicht und Luft-Bewegung hat.

Angrenzende Dächer stellen in der Regel kein Problem für die Platzierung eines Fensters dar. Dies ändert sich, sobald das angrenzende Dach so hoch ist, dass es dem Fenster die Sicht blockiert. Dies kann geschehen, wenn ein Dach direkt an der Voxel-Seite des Fensters angrenzt oder eine Ebene tiefer liegt und höher als ein Voxel ist, vergleiche Abbildung 38. Diese Sonderfälle können durch einen einfachen Vergleich der das Fenster um-

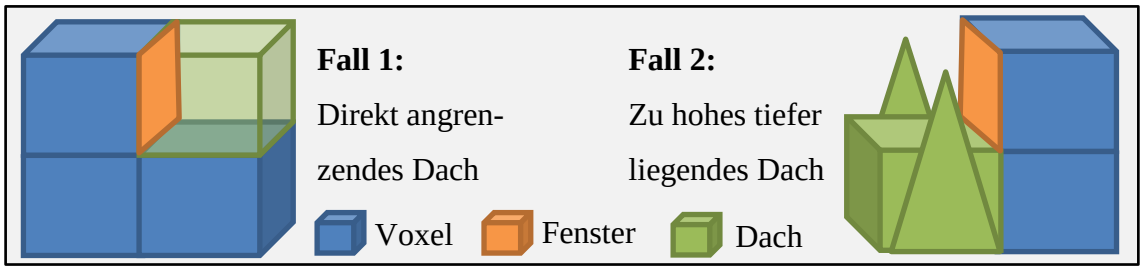


Abbildung 38: Veranschaulichung der beiden Sonderfälle, in denen ein Fenster auf ein angrenzendes Dach blickt. Quelle: Eigene Grafik.

gebenden Voxel festgestellt werden. Ist der an die Voxel-Seite des Fensters angrenzende Block ein Dach-Voxel, ist das Fenster invalide platziert. Dasselbe tritt auf, wenn der auf der unteren Ebene an der Fensterseite angrenzende Voxel ein Dach-Voxel ist und dessen Höhe mehr als ein Voxel beträgt. In jedem Dach-Voxel ist seine jeweilige Höhe verzeichnet und kann nachgesehen werden.

Ein weiterer Aspekt einer korrekten Fensterposition ist der Zugang zu Sonnenlicht. Ein simpler Ansatz wäre, von der Position des Fensters aus eine gradlinige Verbindung aus leeren Voxeln zum oberen Ende des Voxel-Gitters und somit zum Himmel über dem Gebäude zu finden. Dies würde jedoch jede Art von Fenster unter einem noch so kleinen Überhang verbieten.

Gebäude wie das Lake-Town Haus in Abbildung 24 auf Seite 23 wären dadurch nicht möglich. Um Fenster unter kleinen Überhängen zu ermöglichen, sowie Fenster in Stockwerken, die weit unter großen Überhängen liegen muss dieser Ansatz abgeändert werden.

Daraus folgend werden neben geraden Schritten in Richtung Himmel auch horizontale Schritte in Blickrichtung des Fensters erlaubt. Außerdem gelten Stütz-Voxel nicht als Licht-Hindernis, da sie lediglich Stützen sind und somit das Licht nicht komplett blocken. Dies ermöglicht Fenster mit angemessenem Abstand zu einem Überhang, verbietet sie aber weiterhin direkt unter großen Überhängen.

Dabei darf für jeden Schritt in Richtung Himmel ein Schritt in der Horizontalen getan werden. Dies ermöglicht das umgehen kleinerer Überhänge und größerer, solange ein

angemessener Abstand zum Fenster eingehalten ist, vergleiche Abbildung 39. Die erweiterte Methode zur Überprüfung von Sonnenlichtzugang kann dabei wie in Abbildung 40 im Pseudocode nachgewiesen werden.

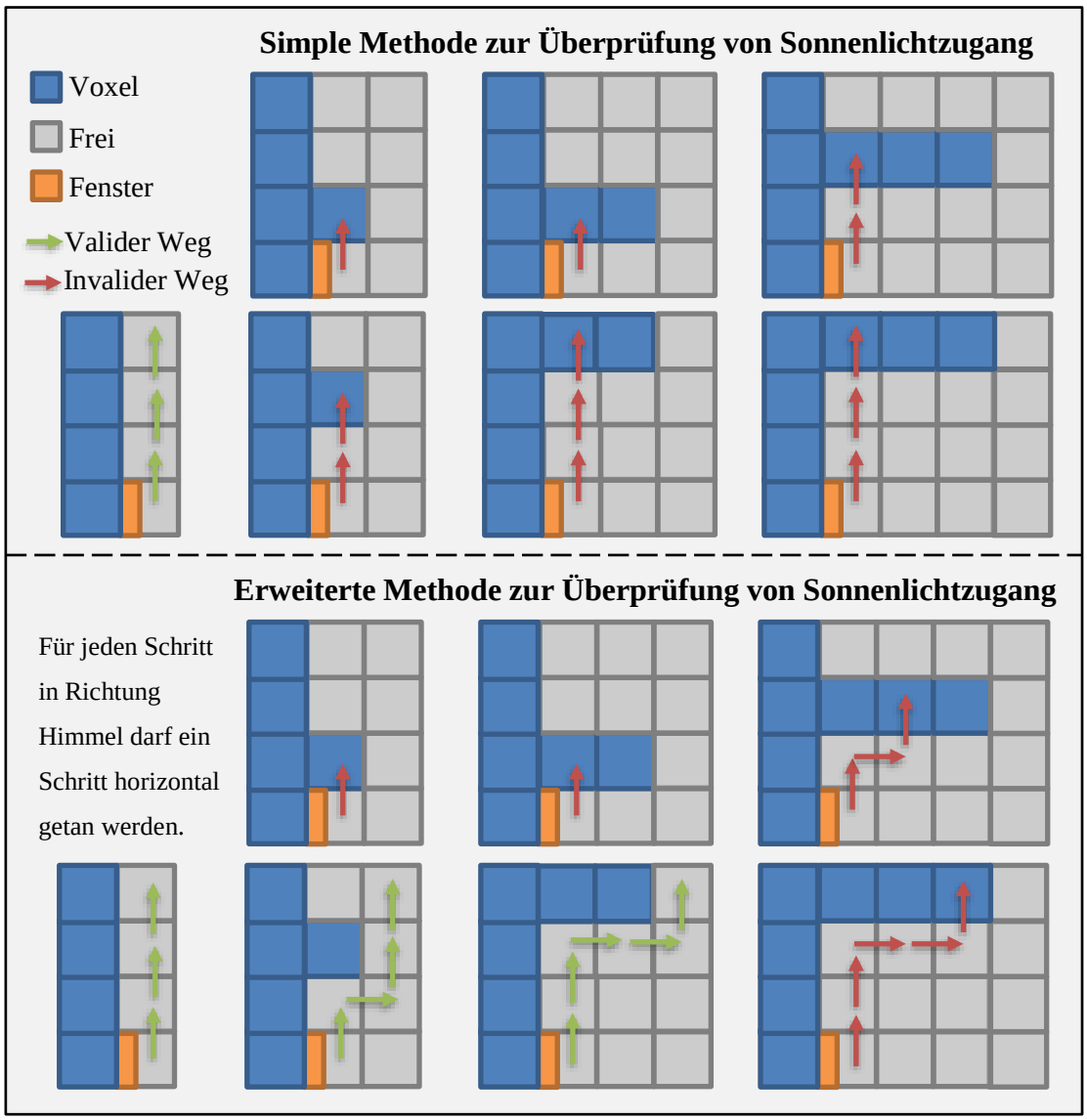


Abbildung 39: Vergleich der Beiden vorgestellten Methoden zum Nachweis von Sonnenlichtzugang an verschiedenen Aufrissen von Gebäudeteilen mit unterschiedlich großen Überhängen über dem zu bewertenden Fenster. Quelle: Eigene Grafik.

```
HatFensterLichtZugang(fenster, gebäude): Bool
        vertikaleSchritte: int
        horizontaleSchritte: int
        aktuellerVoxel: Voxel
        nächsterVoxel: Voxel
        aktuellerVoxel = voxel in blickrichtung des fensters

        while aktuellerVoxel ist nicht in der obersten ebene
                nächsterVoxel = voxel oberhalb von aktuellerVoxel
                if  nächsterVoxel ist  leer oder Stütz-Voxel
                        vertikaleSchritte + 1
                else
                        if  horizontaleSchritte ist kleiner als vertikaleSchritte
                                nächsterVoxel = voxel in fensterBlickrichtung
                                horizontaleSchritte + 1
                        else
                                return false

                        if  nächsterVoxel ist nicht leer oder StützVoxel
                                return false

                aktuellerVoxel = nächsterVoxel
        return true
```

Abbildung 40: Pseudocode Implementierung der erweiterten Methode zum Nachweis von Sonnenlichtzugang. Quelle: Eigene Grafik.

Neben einem Zugang zu Sonnenlicht ist ebenfalls ein Zugang zu frischer Luft wichtig. Der Zugang zu Luft-Bewegung lässt sich ähnlich wie der Zugang zu Sonnenlicht nachweisen. Dabei wird im Gegensatz zu der Suche nach Licht nicht ein Weg in die oberste Ebene des Gebäudes gesucht, sondern beginnend von dem an das Fenster angrenzenden Voxel eine Verbindung zum Rand der jeweiligen Stockwerkebene. Im Fall der Luft-Zirkulation ist es irrelevant, wie der Weg zum Rand des Stockwerks aussieht, der wichtige Punkt ist, das einer existiert. Betrachtet man die Stockwerkebene des Fensters als Graphen, kann diese Verbindung mit dem äußeren Ebenen Rand unter Zuhilfenahme einer Breitensuche ermittelt werden.

3.1.3.2 Haustür

Da die Haustür den einzigen Zugang zum Inneren des Gebäudes darstellt, ist es wichtig, dass diese von außerhalb des Gebäudes erreicht werden kann. Zu diesem Zweck muss sich die Haustür im Erdgeschoss befinden oder durch eine Treppe an der Außenseite der Fassade mit dem Erdgeschoss verbunden sein. Außerdem muss ein Weg von der Haustür zum Rand der Erdgeschossebene existieren, um sicherzustellen, dass sich die Haustür nicht in einem komplett von dem Gebäude umschlossenen Innenhof befindet.

Um dies nachzuweisen stellen wir zuerst fest, ob sich die Tür im Erdgeschoss befindet. Trifft dies zu, wird über eine Breitensuche ein Weg zu den Rändern der Erdgeschossebene gesucht. Liegt die Tür nicht im Erdgeschoss, muss ebenfalls mit Hilfe einer Breitensuche ermittelt werden, ob ein Weg zum Rand des Erdgeschosses existiert. In diesem Fall werden zusätzlich noch die Treppen an der Außenseite der Fassade mit betrachtet um vom Stockwerk der Haustür das Erdgeschoss erreichen zu können.

Nachdem der Aufbau der Fassade als korrekt charakterisiert wurde, folgt die Bestimmung der Komplexität der Fassade. Dafür wird jede Fassadenseite einzeln betrachtet und anschließend mit den anderen drei Ansichten verglichen. Das Dach wird dabei getrennt von den Fassadenansichten bewertet. Die für diese Evaluierung wichtigen Faktoren lauten:

- Unterschiede zwischen den Fassadenseiten.
- Dachgestaltung.

Ein paar der in der Architekturanalyse festgestellten Eigenschaften von Fassaden abwechslungsreicher Gebäude sind Details der visuellen Repräsentation und werden im Voxel-Modell nicht dargestellt. Daraus folgend werden sie nicht mit in die Evaluierung mit einbezogen. Zu diesen Eigenschaften zählen:

- An den Rändern leicht überstehende Dächer.
- Windschiefe Wände.
- Verschiedene Wandbeläge/Texturen.

3.1.3.3 Unterschiede der Fassadenseiten

Die 4 Fassadenansichten eines Gebäudes unterscheiden sich neben ihrer Silhouette hauptsächlich in der Anordnung ihrer Fenster. Folglich ist für die Analyse der Fassadenseiten ausschließlich die Silhouette der jeweiligen Gebäudeseite ausschlaggebend, wobei deren dreidimensionale Eigenschaften nicht beachtet werden. Die Fassadenseiten werden im Hinblick auf die aus der jeweiligen Seite sichtbare Breite der Stockwerke und die Anordnung der Fenster verglichen. Umso mehr Ansichten dieselben Stockwerkbreiten besitzen umso monotoner wird die Fassade bewertet. Das gleiche Prinzip gilt für die Anordnung der Fenster auf den Fassadenseiten. Umso mehr Fenster relativ zur Fassaden-Silhouette

dieselbe Position besitzen wie die Fenster einer anderen Ansicht, umso monotoner wird die Fassade bewertet. Dabei wird die relative Position der Fenster zu ihrer Fassadensilhouette mit ihrem Abstand zu der jeweils näherliegenden linken oder rechten Kante der Fassade bemessen, vergleiche Abbildung 41.

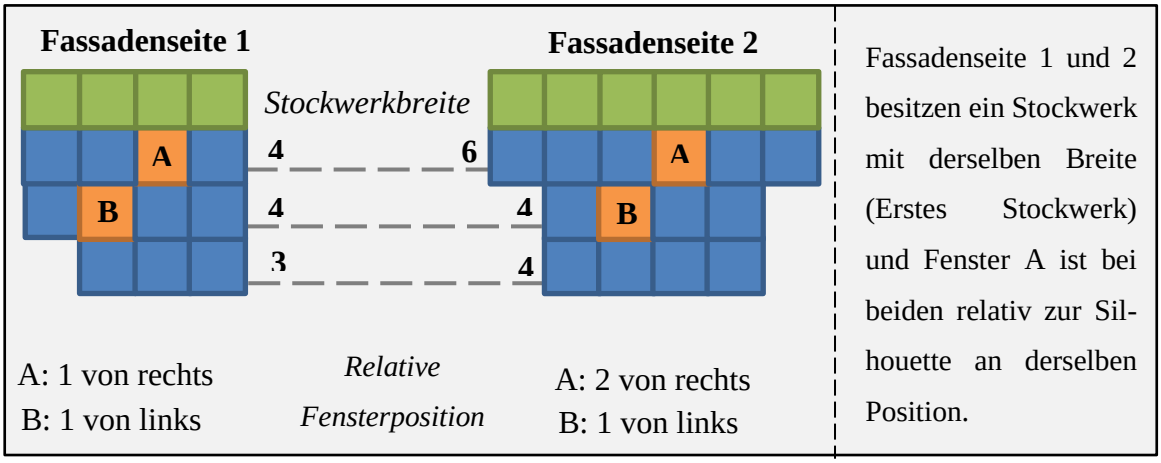


Abbildung 41: Beispielhafte Auswertung von zwei Fassadenseiten eines Gebäudes anhand der vorgestellten Methodik. Quelle: Eigene Grafik.

3.1.3.4 Dachgestaltung

Bei der Dachgestaltung wird bewertet, wie viele verschiedene, zusammenhängende Dachbereiche mit unterschiedlicher Höhe und Richtung existieren. Des Weiteren beeinflusst die Anzahl der Dachdekoration den Abwechslungsreichtum der Dachgestaltung. Dabei gilt viel Dekoration als abwechslungsreicher.

Zuerst werden die verschiedenen Dachbereiche ermittelt. Dach-Voxel bilden einem Bereich, wenn sie direkt aneinander angrenzen und dieselbe Richtung, sowie Höhe besitzen. Ein Dachbereich kann auch von einem einzelnen Dach-Voxel gebildet werden, wenn er ausschließlich von Dach-Voxeln anderer Höhen und Richtungen umgeben ist. In die Bewertung geht dabei die Anzahl der Dachbereiche, sowie die Anzahl der Dachbereiche mit Unterschiedlichen Dachhöhen und Richtungen. Um die einzelnen Dachbereiche ausfindig zu machen, wird das Dach des Gebäudes mittels einer Breitensuche geflutet. Dabei werden nur die Voxel geflutet, die die Selbe Konfiguration an Höhe und Richtung wie der erste Voxel der Suche besitzen. Können unter den angrenzenden Voxeln keine weiteren mit denselben Werten gefunden werden, gilt der Dachbereich als in sich geschlossen, seine Voxel werden als besucht markiert und separat gespeichert. Dieser Vorgang wird so lange wiederholt, bis alle Voxel besucht und somit in einen Dachbereich eingeteilt wurden. Abschließend können die Dachbereiche gezählt und anhand ihrer Höhen und

Richtungen ausgewertet werden. Eine Beispielhafte Implementierung der Einteilung der Dach-Voxel in Dachbereiche ist in Abbildung 42 als Pseudocode zu sehen.

```
ErkenneDachbereiche(gebäude): Collection von Voxel-Collections
        dachbereiche: Collection von Voxel-Collection
        nochNichtBesuchteVoxel: Voxel Collection
        nochNichtBesuchteVoxel = alle Dach-Voxel des gebäudes
        warteschlange: Voxel Collection
        startVoxel: Voxel
        aktuelleCollection: Voxel Collection

        while nochNichtBesuchteVoxel ist nicht leer
                startVoxel = erster Voxel aus nochNichtBesuchteVoxel
                aktuelleCollection = neue Collection mit dem startVoxel
                warteSchlange = neue Collection mit startVoxel

                while warteschlange ist nicht leer
                        aktuellerVoxel: Voxel
                        aktuellerVoxel = entferne voxel vom anfang der warteschlange
                        entferne aktuellerVoxel aus nochNichtBesuchteVoxel

                        foreach angrenzenden voxel an aktuellerVoxel
                                if voxel hat die selben werte wie startVoxel und
                                   ist in nochNichtBesuchteVoxel enthalten und
                                   ist nicht in der warteschlange
                                        füge voxel zum ende der warteschlange hinzu

                        füge aktuellerVoxel in aktuelleCollection hinzu

                füge aktuelleCollection in dachbereiche hinzu

        return dachbereiche
```

Abbildung 42: Pseudocode zur vorgestellten Methode zur Erkennung der einzelnen Dachbereiche eines Gebäudedaches anhand der Höhe, Richtung und Nachbarschaft der einzelnen Dach-Voxel. Quelle: Eigene Grafik.

3.2 Innenräume

Wie bereits in der Architekturanalyse festgestellt, werden keine abwechslungsreichen Räume, sondern funktionale Innenräume generiert. Folglich existieren keine optionalen Eigenschaften für die generierten Räume, welche sie als interessant oder monoton kennzeichnen. Ein Innenraum muss den Vorgaben eines funktionalen Raumes gerecht werden oder wird als unzureichend bewertet, wodurch die Innenraumgestaltung in der Evaluierung durchgefallen ist. Dabei muss jeder Raum:

- von der Haustür aus durch angemessen viele Räume erreichbar sein.
- mindestens ein Fenster besitzen.
- die angegebene Mindestraumgröße überschreiten

Bei jeder der genannten Eigenschaften ist es notwendig zu wissen, aus welchen Voxeln die einzelnen Raum bestehen. Folglich wird bevor auf die Methoden zum Nachweis der genannten Eigenschaften eingegangen wird eine Methode zur Erkennung der einzelnen Innenräume des Gebäudes vorgestellt.

3.2.1 Raumerkennung

Bei der Raumerkennung werden die Grundrisse der einzelnen Stockwerke betrachtet. Dabei repräsentiert die unterste Ebene eines Stockwerks den Grundriss. Die restlichen Ebenen dienen der Fassadengestaltung, sowie der Anpassung der Raumhöhen und sind für die Erkennung der Räume nicht relevant. Bei der Aufteilung des Grundrisses in die jeweiligen Räume gehen wir ähnlich wie bei der Erkennung der Dachbereiche während der Evaluierung der Dachgestaltung vor. Im Gegensatz dazu werden aber nur die Voxel der unteren Stockwerksebene betrachtet, welche den Grundriss des Stockwerks darstellen.

Die Grundrisse der Stockwerke werden mittels einer Breitensuche geflutet bis der untersuchte Bereich von Wänden eingeschlossen ist. Alle erreichten Voxel der Breitensuche sind somit vom Rest des Stockwerks durch Wände abgegrenzt und bilden einen Raum. Die Voxel dieses Raumes werden als besucht markiert und separat gespeichert. Dieser Vorgang wird solange mit einem beliebigen, unbesuchten Voxel der Grundrisse wiederholt, bis alle Voxel in Räume aufgeteilt sind. Eine Beispielhafte Pseudocode Implementierung dieses Vorgehens ist in Abbildung 43 zu sehen.

```
ErkenneRäume(gebäude): Collection von Voxel-Collections
    räume: Collection von Voxel-Collection
    nochNichtBesuchteVoxel: Voxel Collection
    nochNichtBesuchteVoxel = alle Voxel der untersten ebene aller stockwerke
    warteschlange: Voxel Collection
    startVoxel: Voxel
    aktuelleCollection: Voxel Collection

    while nochNichtBesuchteVoxel ist nicht leer
        startVoxel = erster Voxel aus nochNichtBesuchteVoxel
        aktuelleCollection = neue Collection mit dem startVoxel
        warteSchlange = neue Collection mit startVoxel

        while warteschlange ist nicht leer
            aktuellerVoxel: Voxel
            aktuellerVoxel = entferne voxel vom anfang der warteschlange
            entferne aktuellerVoxel aus nochNichtBesuchteVoxel

            foreach angrenzenden voxel an aktuellerVoxel
                if zwischen aktuellerVoxel und voxel liegt keine wand
                und ist in nochNichtBesuchteVoxel enthalten
                und ist nicht in der warteschlange
                    füge voxel zum ende der warteschlange hinzu

            füge aktuellerVoxel in aktuelleCollection hinzu

        füge aktuelleCollection in räume hinzu

    return räume
```

Abbildung 43: Pseudocode zur vorgestellten Methode zur Erkennung der einzelnen Räume eines Gebäudes. Quelle: Eigene Grafik.

3.2.2 Erreichbarkeit

Nachdem die Einzelnen Räume durch die bereits beschriebene Methode identifiziert wurden, wird die Erreichbarkeit der einzelnen Räume untersucht. Dazu wird ein Verbindungs-Graph der einzelnen Räume erstellt.

Dabei entspricht jeder Raum einem Knoten, die Türen zwischen den Räumen und Treppen zwischen den Stockwerken bilden die Kanten des Graphen. Türen werden auf den Voxel-Seiten platziert, somit werden die beiden durch die Tür verbundenen Räume durch die beiden verbundenen Voxel gekennzeichnet. Dies ist jeweils der Voxel auf dem die Tür liegt und der in Richtung der Tür angrenzende Voxel. Die Haustür bildet in diesem Fall eine Ausnahme, da sie das Gebäude mit der Umgebung verbindet und nicht mit einem weiteren Raum. Ähnlich verhält es sich mit den Treppen-Voxeln, welche die Stockwerke untereinander verbinden. Treppen verbinden Stockwerke, wenn in dem darunter oder darüberliegenden Stockwerk an derselben Stelle ebenfalls ein Treppen-Voxel

liegt. Ein Beispiel für die Umwandlung eines Gebäudes in einen Verbindungs-Graphen ist in Abbildung 44 zu sehen. Ist der Graph erstellt, wird vom Raum der Haustür beginnend mittels einer Breitensuche getestet, ob der Graph im Sinne der Graphentheorie zusammenhängend ist. Hierbei gleicht das Vorgehen der bereits in Kapitel 3.1.1.3 beschriebenen Methode zur Überprüfung des Zusammenhanges des Gebäudes.

Nachdem nachgewiesen ist, dass alle Räume erreichbar sind, wird mit Hilfe einer Breitensuche der kürzeste Weg von dem Raum der Haustür zu jedem Raum des Gebäudes ermittelt und festgestellt, ob dieser für das jeweilige Gebäude angemessen ist.

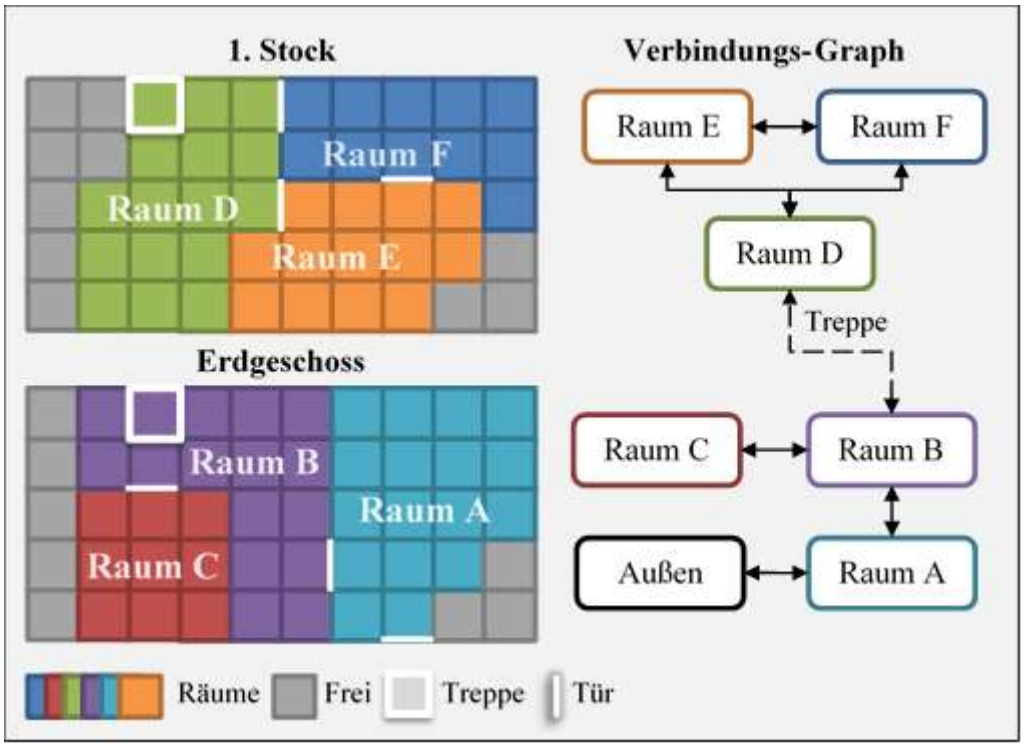


Abbildung 44: Verdeutlichung der Umwandlung eines Gebäudes mit verschiedenen Stockwerken und Räumen in einen Verbindungs-Graphen. Quelle: Eigene Grafik.

3.2.3 Existenz von Fenstern

Es wird vorausgesetzt, dass die Wände eines Raumes vom Grundriss in der untersten Ebene eines Stockwerkes ausgehend durch alle darüberliegenden Ebenen ohne Unterbrechungen oder Ausbuchtungen bis zum Beginn des nächsten Stockwerks laufen. Daraus folgend müssen sich die Fenster eines Raumes in einem der Voxel des Grundrisses oder in den der darüberliegenden Ebenen des Stockwerks befinden. Durch eine Iteration über die Voxel-Seiten der betreffenden Voxel werden alle Fenster des Raumes lokalisiert. Besitzt ein Raum nicht mindestens ein Fenster, wird er als unzureichend bewertet.

3.2.4 Mindestgröße

Die Mindestgröße eines Raumes wird über die Anzahl der Grundriss-Voxel bestimmt. Liegt die Anzahl der Grundriss-Voxel nicht über der Voxel-Mindestgröße eines Raumes, so ist der Raum zu klein und wird als unzureichend bewertet.

3.3 Bewertungsalgorithmus

Nachdem nun Methoden zum Nachweis aller in der Architekturanalyse festgelegten Eigenschaften eines abwechslungsreichen Gebäudes vorgestellt wurden, wird ein Bewertungsalgorithmus erarbeitet, der diese Eigenschaften in einem quantifizierbaren Score umrechnet. Dieser Score ist ein leichter interpretierbares Feedback für den Generierungsalgorithmus, als die Anwesenheit verschiedenster Gebäudeeigenschaften. Der Algorithmus weiß nicht wie er die An- oder Abwesenheit der Verschiedenen Eigenschaften gegeneinander abwiegen soll. Diese Aufgabe nimmt der Bewertungsalgorithmus der Generierung ab und Sorgt somit für eine klare Trennung der Bewertungseigenschaften von der Generierung. Der Generierungsalgorithmus kann ausgetauscht werden, ohne die Evaluierung in Mitleidenschaft zu ziehen.

Dabei werden Gebäude, welche die Grundvoraussetzungen an Aufbau, Fassade und Innenräume erfüllen mit einem positiven Wert beginnend bei 1 versehen. Umso abwechslungsreicher ein Gebäude ist, umso höher ist der Score des Gebäudes. Werden die Grundvoraussetzungen nicht erfüllt gilt das Objekt nicht als Gebäude und wird mit einem Wert von -1 bewertet. Dabei wurde der Score für ein invalides Gebäude in den negativen Bereich verschoben und nicht auf 0 gesetzt, um in Zukunft eine differenzierte Bewertung im negativen Bereich zuzulassen ohne den Sonderfall der 0 beachten zu müssen. So könnte mit der Höhe des negativen Wertes die Schwere der Verstöße gegen die Grundvoraussetzungen beziffert werden, um somit den Aufwand zu quantifizieren, der nötig wäre um das Gebäude zu reparieren. Dies ermöglicht eine einfache Erweiterung des Evaluierungsalgorithmus um einen Generierungsalgorithmus zu unterstützen, der gescheiterte Gebäude nicht verwirft, sondern versucht sie zu reparieren.

Der Bewertungsraum des Evaluierungsalgorithmus erstreckt sich somit von *-1* für invalide Gebäude bis zu *positiv unendlich* für abwechslungsreiche Gebäude, wobei die *0* außenvorgelassen wird.

Zuerst wird bei der Evaluierung geprüft, ob die Grundvoraussetzungen eines Gebäudes erfüllt sind, da ansonsten die restlichen Eigenschaft des Gebäudes irrelevant sind. Dazu werden die Grundvoraussetzungen des Aufbaus, der Fassade und der Innenräume überprüft. Sollte eine der Voraussetzungen nicht erfüllt sein, ist das Gebäude in der Evaluierung durchgefallen und wird mit einem Score von -1 bewertet. Daraus folgend kommt der Evaluierungsalgorithmus zu den in Abbildung 45 zusammengefassten grundlegenden Ergebnissen.

Ergebniss des Evaluierungsalgorithmus basierend auf den Gebäudeeigenschaften

Status: Gegeben = 1, Nicht Gegeben = 0
Eigenschaften: Grundvoraussetzungen/Zusätzliche Eigenschaften

Mögliche Zustände:	1/1	1/0	0/1	0/0
Ergebnis:	+ Wert	+ Wert	-1	-1

Abbildung 45: Ergebnisse des Evaluierungsalgorithmus basierend auf den gegebenen und nicht gegebenen Gebäudeeigenschaften. Quelle: Eigene Grafik.

Nach dem Prüfen der Grundvoraussetzungen beginnt der Algorithmus mit der Bewertung des Aufbaus und geht danach immer weiter auf die Details der Fassade ein. Die Innenräume werden in diesem Schritt nicht betrachtet, da ihre Eigenschaften ausschließlich von Grundvoraussetzungen beschrieben werden, welche an diesem Punkt bereits evaluiert wurden. Die Evaluierungsergebnisse des Aufbaus und der Fassade werden daraufhin miteinander verglichen. Weichen diese weit voneinander ab, ist ein Bestandteil des Gebäudes also als monoton und der andere als abwechslungsreich bewertet worden, wird der Gesamtscore des Gebäudes um die Differenz der beiden Werte erhöht. Dies nimmt die in Kapitel 3.1 bereits beschriebene Erhöhung des Abwechslungsreichtums durch einen Kontrast zu einem monotonen Gebäudebereich mit in die Bewertung auf. Umso höher der Kontrast zwischen den beiden Bereiche, umso mehr wird der Abwechslungsreichtum des einen hervorgehoben und umso höher wird das Gebäude bewertet. Je größer die Differenz zwischen der Fassaden- und der Aufbaubewertung, desto höher die Gesamtbewertung des Gebäudes. Womit der Score = S des Algorithmus bei Aufbaubewertung = A und Fassadenbewertung = F wie folgt berechnet werden kann:

$$S = A + F + |(A - F)|$$

Dieser Effekt ist noch nicht stark genug um ein Gebäude mit zwei stark abwechslungsreichen Gebäudeteilen schlechter zu bewerten als ein Gebäude mit einer großen Differenz der einzelnen Gebäudeteilbewertungen. Um die Differenz stärker ins Geweicht fallen zu

lassen, wird ihr Wert quadriert, wodurch sich der Score = *S* des Algorithmus bei Aufbau = *A* und Fassade = *F* wie folgt berechnen lässt:

$$S = A + F + |(A - F)|^2$$

Dies sorgt dafür, dass die Differenz der Werte mehr ins Gewicht fällt, umso größer sie wird. Woraufhin Gebäude mit stark unterschiedlichen Bestandteilen wesentlich besser bewertet werden als Gebäude mit ähnlich abwechslungsreichen Eigenschaften, vergleiche Tabelle 6.

Formel 1:	$S = A + F + \|(A\text{-}F)\|$					
Formel 2:	$S = A + F + \|(A\text{-}F)\|^2$					
	Beispiel 1		Beispiel 2		Beispiel 3	
	A	F	A	F	A	F
Einzelwerte	10	10	1	5	1	10
Formel 1	20		10		20	
Formel 2	20		22		92	

Tabelle 6: Vergleich der 2 Score-Funktionen anhand von 3 Beispielen zur Verdeutlichung der Bedeutung der Differenz in der Bestandteil-Bewertung und deren Einfluss auf den Score. S = Score. A = Aufbau. F = Fassade. Quelle: Eigene Tabelle.

Um den Mindestscore von 1 für einen monotonen Gebäudebestandteil zu garantieren, wird für das Bestehen der Grundvoraussetzungen jedem Teilbereich ein Score-Punkt gutgeschrieben. Von diesem ausgehend werden dem betreffenden Bereich für jede positiv aufgefallene Eigenschaft Punkte gutgeschrieben. Wieviele Punkte die jeweiligen Methoden für die festgestellten Eigenschaften vergeben ist dabei ein Implementierungsdetail, dass je nach angestrebtem Gebäudetyp variiert werden kann.

Umso mehr eine einzelne Methode in der gesammten Evaluierung gewichtet werden soll, desto mehr Punkte muss sie im Vergleich mit den anderen Methoden vergeben. Um dies zu erreichen erhält jede Methode eine Gewichtung. Die Punkte der Methode werden dabei vor der Rückgabe an die Evaluierungsfunktion mit der Gewichtung multipliziert. Dies ermöglicht das Anpassen der Gewichtung der einzelnen Methoden, sowie das Entfernen aus der Evaluierung, ohne die Evaluierungsfunktion zu verändern. Um eine Methode bei der Evaluierung nicht zu berücksichtigen wird ihre Gewichtung auf Null gesetzt.

4. Generierung

Dieses Kapitel befasst sich mit der Generierung von Gebäuden, bestehend aus den Grundrissen, der Fassade und den Innenräumen. Da die jeweiligen Bereiche aufeinander aufbauen, wird zuerst der Grundriss generiert. Anschließend können die Fassade und die Innenräume in beliebiger Reihenfolge generiert werden.

Dabei ist zu beachten, dass die beiden Bereiche voneinander abhängig sind. Die Räume benötigen jeweils ein korrekt platziertes Fenster. Um dies zu erreichen kann zuerst die Fassade generiert und die Raumpositionen an die bestehenden Fenster angepasst werden. Die zweite Möglichkeit ist die Innenräume zuerst zu generieren und anhand dessen die Position der Fenster zu bestimmen.

Mögliche Generierungsreihenfolgen:

A. Grundriss, Fassade, Innenräume
B. Grundriss, Innenräume, Fassade

In dem hier vorgestellten Ansatz wird Reihenfolge *B.* verwendet. Dies ermöglicht es die Fassade abschließend in einem Schritt passend zu der Innenraumgestaltung zu generieren. Bei Reihenfolge *A.* kann es notwendig werden die Fassade nach der Generierung der Innenräume noch einmal anzupassen. Es kann nicht garantiert werden, dass die Räume so platziert werden können, dass jeder Raum ein Fenster besitzt. Dieser zusätzliche Schritt wird vermieden, wenn die Fassade als letztes generiert wird.

Abschließend werden zusätzlich Dekorations-Objekte platziert um die einfache Form des Gebäudekomplexes aufzuwerten. Zu diesen zählen zum Beispiel Fensterläden, Dachziegel und Türrahmen.

4.1 Grundriss

Die Grundrisse der Stockwerke bilden die Grundlage eines Gebäudes und werden zuerst generiert. Dabei basieren die Grundrisse der oberen Stockwerke auf denen der darunterliegenden um eine einheitliche Wirkung zu erzeugen.

Die Grundrisse werden während der Generierung nicht auf ihren Abwechslungsreichtum untersucht, da monotone so wie abwechslungsreiche Grundrisse valide sind. Nachdem der Grundriss des Gebäudes erstellt wurde, wird sein Abwechslungsreichtum bewertet und darauf basierend eine Fassade mit einer passenden Komplexität generiert.

Die Grundrisse des Gebäudekomplexes werden durch die Benutzereingabe (Grundstücksgröße und Stockwerkanzahl) generiert. Eine der verbreitetsten Methoden zur Generierung von Gebäuden sind *Shape Grammars*, welche von Wonka [5], Müller [4] und Martin [3] [8] erfolgreich verwendet wurden.

Im Gegensatz dazu stellt diese Arbeit eine andere Möglichkeit zur Generierung von Grundrissen vor. Anstelle von *Shape Grammars* wird ein simpler Generierungsalgorithmus verwendet, welcher von der bereits definierten Evaluierungsfunktion geleitet wird. Dies bildet die Vorstufe zu einem neuronalen Netz, wie es bereits erfolgreich von Merrell et. al. zur Generierung von *Residential Building-Layouts* verwendet wurde [9]. Dies bietet die Möglichkeit den Algorithmus in Zukunft durch ein neuronales Netz zu ersetzen. Wodurch der Generierungsalgorithmus nicht mehr auf den gewünschten Gebäudetypus spezialisiert werden muss, sondern alleine von der Evaluierungsfunktion geleitet wird. Dies ermöglicht die Generierung verschiedenster Gebäudetypen ohne den Generierungsalgorithmus austauschen zu müssen. Bei dem vorgestellten, simplen Generierungsalgorithmus wird der Ansatz der additiven Generierung von den *Shape Grammars* übernommen. Dabei werden von einer kleinen Startform ausgehend immer mehr Formen hinzugefügt [11] [6]. Dieser Ansatz ist besser für die Generierung von Wohngebäuden geeignet als der subtraktive Ansatz, welcher für große Gebäude mit einem fast komplett ausgefüllten Grundriss verwendet wird [6].

4.1.1 Erdgeschoss

Angelehnt an den Ansatz der additiven Generierung beginnt der Algorithmus bei der Erstellung des Erdgeschosses mit einer Start-Form, welche in der Mitte des Grundstücks platziert wird. Jede Platzierbare Form ist ein Rechteck mit variabler Breite und Höhe. Dabei wird die maximale Größe der Rechtecke von der Grundstücksgröße beschränkt. Ein Rechteck ist maximal halb so breit und hoch wie das Grundstück. Dies verhindert bis zu einem gewissen Grad, dass Rechtecke generiert werden, die zu groß sind um platziert zu werden. Es ermöglicht aber weiterhin relativ große Gebäudebestandteile.

Dabei werden die Rechtecke immer so platziert, dass sie an den bereits bestehenden Grundriss angrenzen. Die neu erstellten Formen dürfen dabei mit dem bereits bestehenden Grundriss überlappen. Nach der Platzierung einer Form wird der Grundriss durch den Evaluierungsalgorithmus im Hinblick auf die Grundvoraussetzungen geprüft.
Verletzt das platzierte Rechteck eine der Voraussetzungen, wird der Schritt rückgängig gemacht und eine andere Form platziert. Dabei durchläuft die Generierung des Erdgeschosses die folgenden Schritte:

1. Platziere die Start-Form.
2. Platziere eine an den Grundriss angrenzende, zufällige Form.
3. Prüfe die Grundvoraussetzungen eines validen Grundrisses.
4. Passe den Grundriss an. (Basierend auf dem Ergebnis der Evaluierung).
5. Zurück zu Schritt 2.

Dies wiederholt sich so lange, bis einer der folgenden Fälle Eintritt:

- Die vom Nutzer angegebene Anzahl an Formen wurde platziert.
- Es ist keine Platzierung weiterer Formen möglich.
- Der gewünschte Prozentsatz des Grundstücks wird vom Grundriss eingenommen.

4.1.2 Obere Stockwerke

Als nächster Schritt werden die oberen Stockwerke generiert. Dabei unterscheidet sich das Vorgehen von der Generierung des Erdgeschosses. Die Start-Form eines oberen Stockwerks ist immer der Grundriss des darunterliegenden Stockwerks. Dies hat den Effekt, dass die oberen Grundrisse einen Bezug zu den darunterliegenden haben und die einzelnen Stockwerke zusammengehörig wirken. Anschließend werden die Stockwerke vergrößert oder verkleinert. Die Vergrößerung erfolgt dabei auf demselben Weg, wie bei der Generierung der Grundrisse. Es werden zufällige Formen an den Rändern des bestehenden Grundrisses eingefügt.

Die Verkleinerung der Stockwerke kann ebenfalls mit Hilfe von zufällig platzierten Rechtecken erfolgen. Dies hat allerdings den Nachteil, dass zufällig wirkende Einbuchtungen und Löcher in den oberen Stockwerken entstehen. Um die Glaubwürdigkeit des

Gebäudes zu bewahren ist es nötig bei der Verkleinerung der Stockwerke Rücksicht auf den Aufbau des Gebäudes zu nehmen, vergleiche Abbildung 46 A und B.

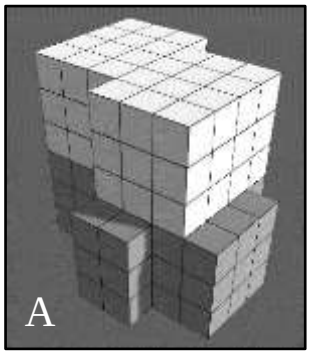
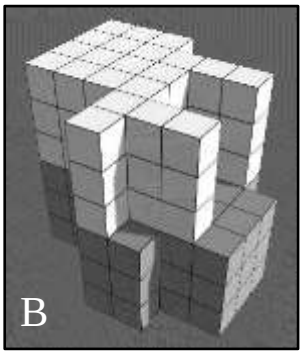


Abbildung 46: Vergleich der zwei Methoden zum Verkleinern eines oberen Stockwerks.

A: Verkleinerung mit Rücksicht auf den Gebäudeaufbau.

B: Zufällige Verkleinerung des Stockwerks.

Quelle: Eigene Grafik.



Der Algorithmus hat keine Kenntnis darüber, welche Voxel entfernt werden können, ohne den Aufbau des Gebäudes zu missachten und es somit unglaubwürdig wirken zu lassen. Um dieses Problem zu umgehen wird auf die Rechtecke zurückgegriffen, die zur Erstellung des Grundrisses verwendet wurden. Diese werden zur Generierungszeit zusammen mit ihrer Position und Größe im jeweiligen Stockwerk gespeichert. Wird eines dieser gespeicherten Rechtecke aus dem Grundriss entfernt, wird immer ein ganzer Gebäudebestandteil gelöscht und kein willkürlicher Teilbereich. Dies hat den Effekt, dass der Aufbau des Gebäudes bei einer Verkleinerung des Stockwerkes intakt bleibt. Somit wirkt die Verkleinerung durch das entfernen eines Grundrissrechteckes mehr wie das Ergebnis, das wir von einem Architekten erwarten würden als ein zufälliges Entfernen von Voxeln.

Die Anzahl der Formen, welche verändert werden sollen, wird zu Beginn der Stockwerkgenerierung zufällig festgelegt. Dabei können nicht mehr Rechtecke hinzugefügt oder entfernt werden, als die Start-Form Rechtecke besitzt. Der Grundriss des Stockwerks muss am Ende immer aus mindestens einem Rechteck bestehen. Daraus folgend geht der Algorithmus bei der Generierung der oberen Stockwerke wie folgt vor:

1. Platziere den Grundriss des unteren Stockwerks als Start-Form.
2. Verändere den Grundriss durch eine der Methoden:
    a. Füge eine zufällige, an den Grundriss angrenzende, Form ein.
    b. Lösche ein bereits bestehendes Grundrissrechteck.
3. Evaluiere den Grundriss.
4. Passe den Grundriss an. (Basierend auf dem Ergebnis der Evaluierung).
5. Zurück zu Schritt 2.

Dies wiederholt sich so lange, bis einer der folgenden Fälle Eintritt:

- Die zu Beginn festgelegte Anzahl an Formen wurde entfernt oder hinzugefügt.
- Es können keine weiteren Formen hinzugefügt/entfernt werden ohne die Grundvoraussetzungen des Aufbaus zu verletzen. (Kapitel *3.1.1 Aufbau*)

4.1.3 Stützen

Nachdem die Grundrisse aller Stockwerke erstellt wurden, werden die benötigten Stützen für die Überhänge platziert. Dazu werden alle schwebenden Voxel des Gebäudes anhand der folgenden 4 Regeln untersucht:

Regel 1 – Äußere Gebäudekanten:

Jeder schwebende Voxel, der an mindestens einen leeren Voxel angrenzt, wird gestützt.

Regel 2 – Innere Ecken schmaler Überhänge:

Alle schwebenden Voxel, die an genau 2 schwebende Voxel angrenzen und deren diagonale Voxel leer ist, werden gestützt.

Regel 3 – Innere Ecken breiter Überhänge:

Alle leeren Voxel unter einem schwebenden Voxel, die weniger als 3 Nachbarn besitzen, von den genau 2 Support-Voxel sind können die innere Ecke eines breiten Überhanges sein. Ist dem so, werden sie gestützt.

Regel 4 – Ecken und Kreuzungen:

Alle Ecken und Kreuzungen an den Rändern breiter Überhänge werden mit Säulen gestützt.

Die Auswirkungen der vier Regeln auf die Voxel breiter und schmaler Überhänge sind in Abbildung 47 einmal visuell aufbereitet.

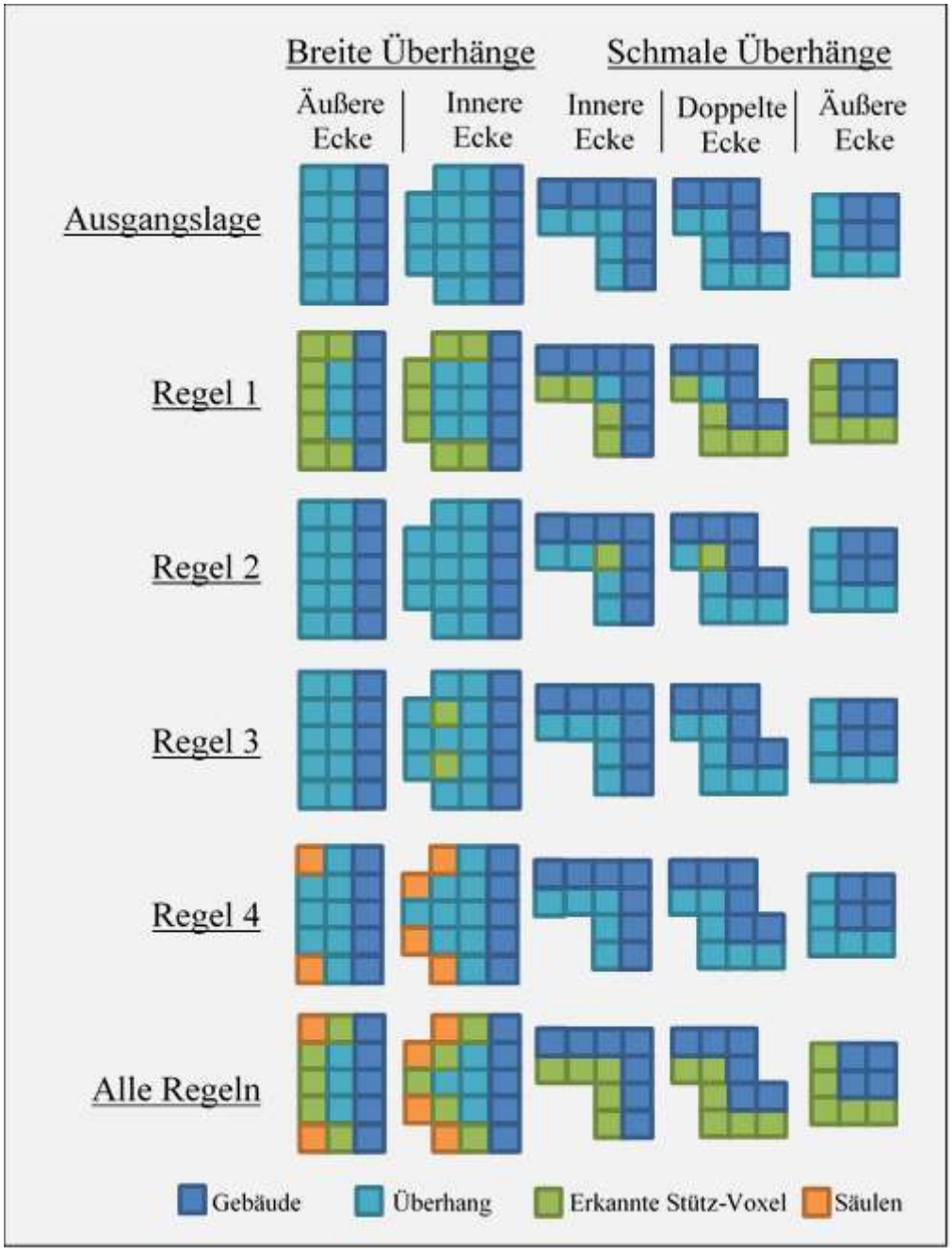


Abbildung 47: Verdeutlichung aller 4 Regeln zur Platzierung von Stützen anhand mehrerer Beispiele. Quelle: Eigene Grafik.

Die erste Regel erkennt alle notwendigen Stütz-Voxel breiter Überhänge, sowie einen Großteil der Voxel schmaler Überhänge. Nur die nach innen gerichteten Ecken schmaler Überhänge werden in dieser Regel nicht berücksichtigt, da sie nicht an leere Voxel angrenzen. Diese Voxel müssen aber genauso gestützt werden, wie die restlichen Voxel schmaler Überhänge.

Aus diesem Grund wird Regel 2 hinzugefügt um die inneren Kanten der schmalen Überhänge mit einzubeziehen. Nachdem Regel 1 und 2 auf alle schwebenden Voxel angewandt wurden, können mit Hilfe von Regel 3 die potenziellen inneren Ecken der breiten Überhänge ausfindig gemacht werden. Dabei wird mittels der Methode aus Kapitel *3.1.1.2 Überhänge ohne Stützen* getestet, ob es sich um eine innere Ecke handelt. Ist dem so, wird der Voxel gestützt. Regel 3 kann dabei erst nach Regel 1 und 2 angewandt werden, da die inneren Ecken ansonsten nicht von den inneren Voxeln eines breiten Überhanges unterschieden werden können.

Nachdem alle notwendigen Stütz-Voxel platziert wurden, werden die in Regel 4 beschriebenen Säulen für breite Überhänge platziert. Wie die Ecken breiter Überhänge erkannt werden können ist in Kapitel *3.1.1.2 Überhänge ohne Stützen* beschrieben. Neben diesen Ecken werden außerdem alle Stütz-Voxel zu Säulen verlängert, die mehr als 2 Nachbarn besitzen, die ebenfalls Stütz-Voxel sind. Dies sind Kreuzungen von Stützreihen.

4.2 Innenräume

Nachdem die Grundrisse der verschiedenen Stockwerke erstellt wurden, werden die Innenräume generiert. Im Folgenden werden zwei Ansätze vorgestellt, mit welchen die Innenräume in einen bereits bestehenden Grundriss eingefügt werden können. Der erste vorgestellte Ansatz verwendet die Rechtecke der Grundrisse als Grundlage für die Generierung der Innenräume. Im zweiten Ansatz hingegen werden die Räume unabhängig von den Rechtecken der Grundrisse erstellt.

4.2.1 Basierend auf Grundrissrechtecken

Ein simpler Ansatz ist die Innenräume des Gebäudes auf den Rechtecken der Grundrisse basieren zu lassen. Dabei entspricht jedes Rechteck einem Raum entsprechen. Dies hat zur Folge, dass immer genauso viele Räume wie Grundrissrechtecke entstehen und diese immer die Form von Rechtecken haben.

Dieser Nachteil wird abgeschwächt, indem Rechtecke, welche eine bestimmte Größe über- oder unterschreiten verändert werden. Überschreitet ein Rechteck die maximale Größe eines Raumes oder sind mehr Räume gewünscht, so kann ein Rechteck durch Teilung in mehrere kleine Räume zerlegt werden. Ähnlich wird mit Räumen verfahren, die eine Mindestgröße unterschreiten. Diese werden mit angrenzenden Rechtecken zusammengelegt.

Dadurch, dass die Räume mittels der Grundrissrechtecke an den Gebäudeaufbau angepasst sind, entsteht eine an die Form des Gebäudes angepasste Innenraumaufteilung, welche die Glaubhaftigkeit des Gebäudes erhöht.

Der Ansatz besitzt jedoch eine Problematik, welche diese Wirkung komplett aufheben kann. Die Raumanzahl ist direkt von der Anzahl der Grundrissrechtecke abhängig. Aus diesem Grund neigt das Vorgehen zu einer starken Fragmentierung der Innenräume, sobald sich mehrere Grundrissrechtecke überlappen, siehe Abbildung 48.

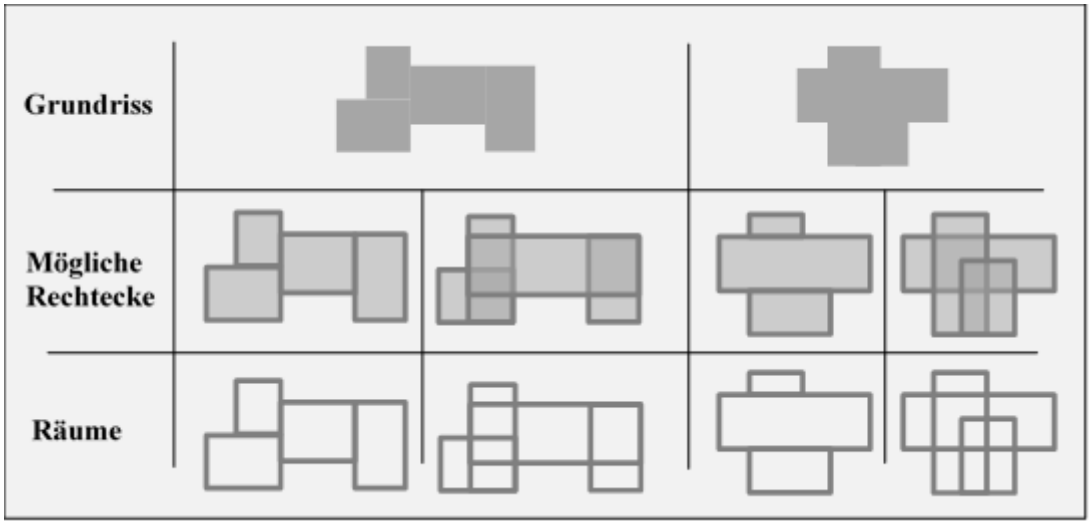


Abbildung 48: Beispiele für Grundrisse mit möglichen Rechteck-Konfigurationen und den daraus folgenden Raumaufteilungen nach dem vorgestellten Ansatz 1. Quelle: Eigene Grafik.

Zusammengefasst besitzt dieser Ansatz die folgenden Vor- und Nachteile:

Vorteile:

- Simpel zu implementierender Ansatz, da die Rechtecke bereits bestehen.
- Glaubhafte Innenraumgestaltung.
- Räume sind immer rechteckig.

Nachteile:

- Raumanzahl ist von der Anzahl der Rechtecke abhängig.
- Starke Fragmentierung der Räume bei überlappenden Rechtecken.

Dabei gibt es zwei verschiedene Ansätze um die Fragmentierung zu unterbinden:

A. **Die Fragmentierung wird verhindert, bevor sie entstehen kann**. Dazu werden Sonderregeln für sich überlappende Rechtecke eingeführt.

B. **Die Fragmentierung wird im Nachhinein korrigiert.** Die Fragmentierung wird nach ihrer Entstehung durch das Zusammenführen einzelner Räume abgeschwächt.

4.2.1.1 Verhindern der Fragmentierung

Um die Fragmentierung zu unterbinden, bevor sie entsteht müssen Sonderregeln für die Behandlung von sich überlappenden Rechtecken aufgestellt werden. Eine simple Sonderregel lautet dabei wie folgt.

Sonderregel:

> Wenn alle Rechtecke in der Reihenfolge ihrer Generierung durchlaufen werden, werden ausschließlich Wände platziert, die nicht in einem bereits bestehenden Rechteck liegen.

Wendet man diese Sonderregel auf die zuvor beschriebenen Beispiele an, so ändert sich die Innenraumgestaltung wie in Abbildung 49 dargestellt. Dies reduziert die Fragmentierung bis zu einem gewissen Grad. Es können aber weiterhin mehr Räume entstehen, als es Rechtecke im Grundriss gibt und es ist nicht mehr garantiert, dass alle erstellten Räume Rechtecke sind. Die Räume können durch andere Rechtecke teilweise angeschnitten werden, vergleiche das zweite Beispiel in Abbildung 49. Dies kann dazu führt, dass manche Räume aufgrund ihrer Ausschnitte nicht der Erwartung eines sinnvoll geschnittenen Raumes entsprechen.

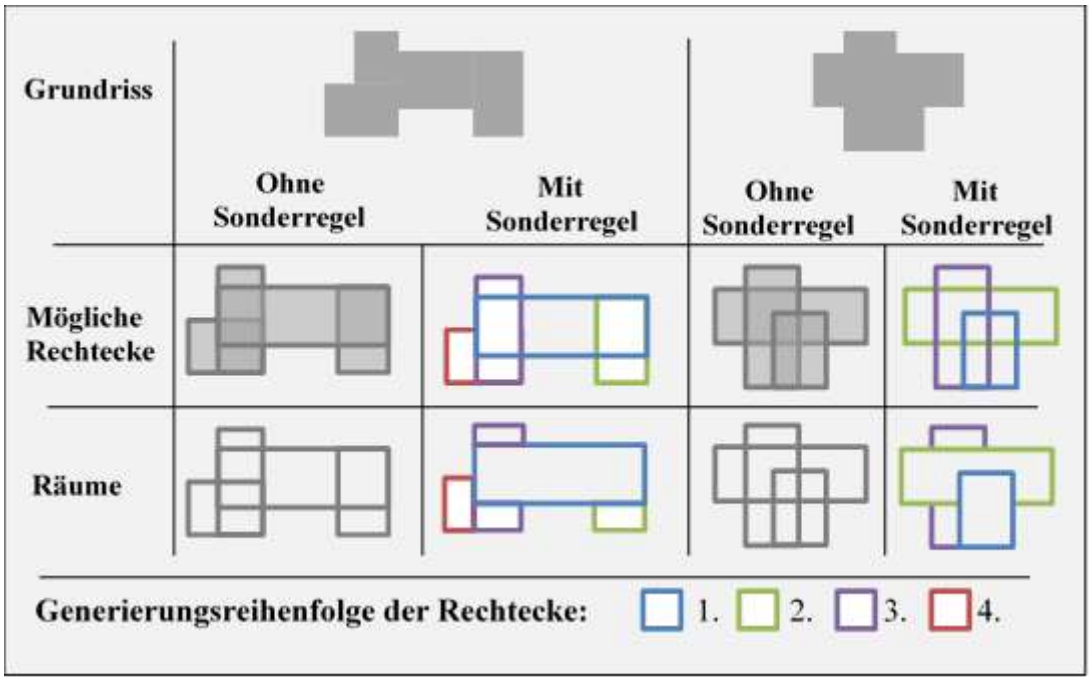


Abbildung 49: Beispiele für die Vermeidung der Fragmentierung mithilfe der vorgestellten Sonderregel. Quelle: Eigene Grafik.

4.2.1.2 Korrektur der Fragmentierung

Ein weiterer Ansatz um mit der Fragmentierung zu reduzieren ist sie im Nachhinein durch die Zusammenführung von Räumen zu korrigieren. Hierfür wird ebenfalls eine Sonderregel benötigt:

Sonderregel:

> Beginnend bei dem kleinsten Raum des Gebäudes werden so lange Räume zusammengeführt, bis die gewünschte Raumanzahl erreicht ist. Dazu werden die kleinen Räume immer mit dem Raum zusammengeführt, mit dem sie sich die größte gemeinsame Wand teilen.

Ein Beispiel für die Anwendung dieser Regel auf eines der bereits beschriebenen Beispiele ist in Abbildung 50 zu sehen. Im Gegensatz zu dem zuvor beschriebenen Ansatz bilden sich keine unerwarteten Ausschnitte, da ein kleiner Raum immer mit dem zusammengeführt wird, mit dem er sich die größte gemeinsame Wand teilt. Allerdings kann auch hier nicht mehr garantiert werden, dass alle erstellten Räume Rechtecke sind.

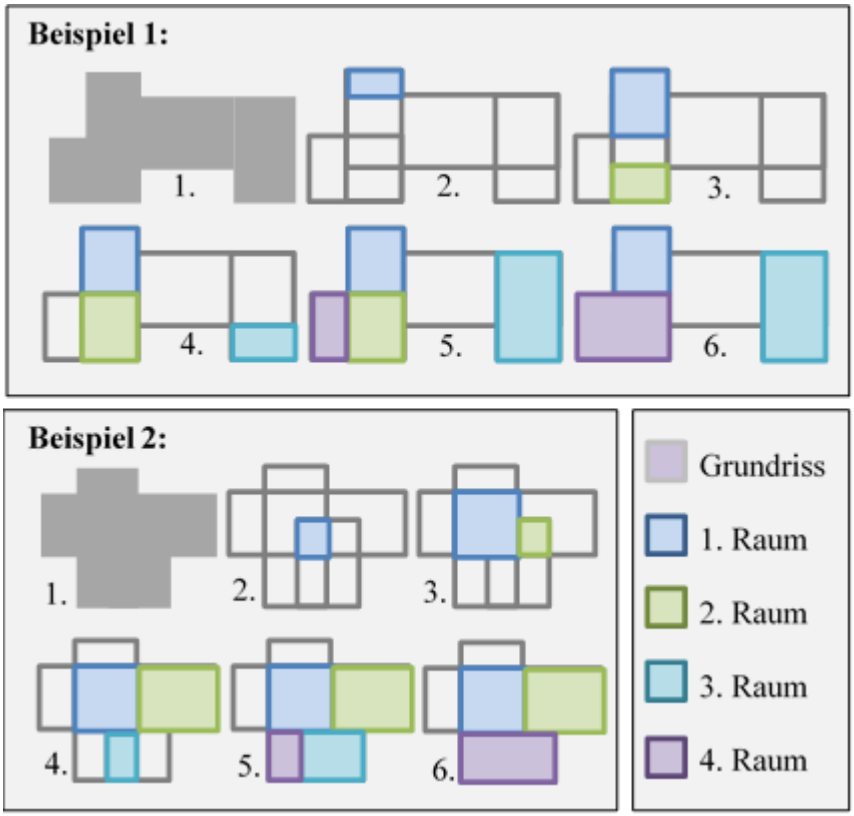


Abbildung 50: Beispiele für die Anwendung der vorgestellten Sonderregel zur Bereinigung der Fragmentierung. Ein Raum wird immer mit dem Nachbarraum zusammengeführt, mit welchem er sich die längste gemeinsame Wand teilt. Quelle: Eigene Grafik.

4.2.2 Unabhängige Räume

Der zweite Ansatz betrifft die Generierung von Innenräumen, welche unabhängig von den Grundrissrechtecken sind. Dies ermöglicht es im Gegensatz zum ersten Ansatz von Anfang an so viele Räume in einem Stockwerk zu platzieren, wie gewünscht sind. Es müssen im Nachhinein keine großen Räume geteilt oder kleinere zusammengefasst werden um die gewünschte Raumanzahl zu erreichen.

Bei diesem Ansatz werden zu Beginn der Generierung so viele zufällige Voxel auf der untersten Ebene eines Stockwerks ausgewählt, wie es später Räume geben soll. Die ausgewählten Voxel fluten von ihrer Position ausgehend den Grundriss des Stockwerks. Dabei entsteht aus jedem der ausgewählten Voxel zusammen mit den von ihm gefluteten Voxeln ein Raum, vergleiche Abbildung 51.

Verläuft das Fluten des Stockwerkes ohne weitere Begrenzungen, führt das zu sehr organisch geformten Räumen. Die Räume erinnern nur zum Teil an Rechtecke und wirken sehr aufgefächert, Siehe Generierungsweg A in Abbildung 51. Somit gestalten sich die Vor- und Nachteile dieses Ansatzes wie folgt:

Vorteil:

- Einfaches Erstellen von beliebig vielen Räumen.

Nachteil:

- Organisch anmutende Räume.

Die organisch wirkenden Räume haben dabei einen Nachteil, der den Ansatz zur Generierung von Innenräumen disqualifiziert. Es werden an Rechtecken orientierte, nicht zu stark fragmentierte Innenraumgestaltungen angestrebt. Folglich muss die Flutung der Grundrisse von Regeln geleitet werden um das organische Wachsen der Räume zu unterbinden. Dies kann durch eine Seitenbeschränkung geschehen.

Seitenbeschränkung:

> Sobald ein wachsender Raum an einer seiner 4 Seiten an einen Voxel eines anderen Raums stößt, gilt diese Kante als fest. Der Raum darf während seines gesamten restlichen Wachstums nicht weiter in die Richtung der festen Kante wachsen.

Die Auswirkung einer solchen Seitenbeschränkung auf das vorgestellte Beispiel ist in Weg B der Abbildung 51 zu sehen. Die noch in Weg A auftretenden organischen Kanten der Räume werden durch die Seitenbeschränkung verhindert.

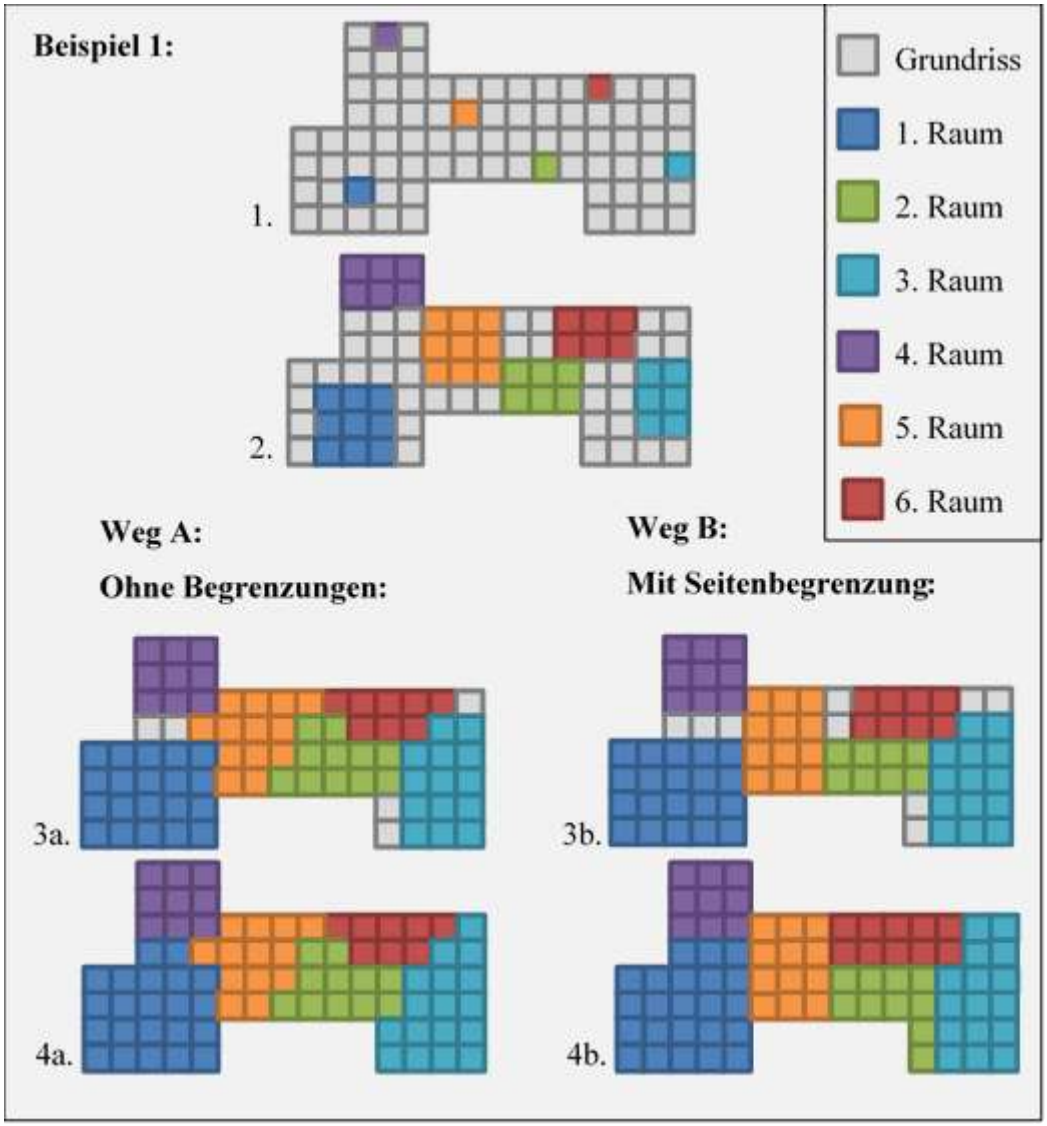


Abbildung 51: Beispiel für das von zufällig ausgewählten Voxeln ausgehende Fluten des Grundrisses zur Definierung von Räumen. Weg A: Fluten des Grundrisses ohne begrenzende Regeln. Weg B: Fluten des Grundrisses unter Beachtung einer Seitenbegrenzung. Quelle: Eigene Grafik.

Im Vergleich mit dem auf den Grundrissrechtecken basierenden Ansätzen ist der in diesem Kapitel vorgestellte Ansatz weniger gut geeignet. Die nach der Beseitigung der Fragmentierung an den Aufbau des Gebäudes angepasste, glaubhaften Raumaufteilung des ersten Ansatzes ist von größerer Bedeutung für das erstellte Gebäude, als das einfache generieren von beliebig vielen Räumen. Aus diesem Grund haben wir uns für den auf den Grundrissrechtecken basierenden Ansatz mit der späteren Bereinigung der Fragmentierung entschieden.

4.2.3 Türen und Treppen

Nachdem alle Räume des Gebäudes definiert wurden, müssen sie noch zugänglich gemacht werden. Zu diesem Zweck werden Türen und Treppen platziert. Die Haustür wird in diesem Schritt ebenfalls platziert, da sie benötigt wird um die Begehbarkeit aller Räume zu überprüfen. Jeder Raum des Gebäudes muss vom Raum der Haustür ausgehend erreicht werden können.

Dazu wird die Haustür in der Außenwand eines beliebigen Raumes platziert. Dieser Raum bildet nun den Eingangsbereich/Flur des Hauses. Dieser bildet den Hauptverbindungspunkt zwischen den einzelnen Räumen und Stockwerken. Dort wird die Treppe ins obere Stockwerk platziert, falls sie benötigt wird. Der Raum, in dem die Treppe im oberen Stockwerk endet ist wiederum ein Hauptverbindungspunkt und potenzieller Raum weiterer Treppen in das höher gelegene Stockwerk.

Nachdem die verschiedenen Stockwerke durch die Treppen verbunden wurden, müssen die einzelnen Räume untereinander verbunden werden. Ein simpler Ansatz wäre, in jedem Raum Türen zu all seinen angrenzenden Räumen zu platzieren. Dies stellt sicher, dass alle Räume untereinander verbunden sind. Allerdings werden auch wesentlich mehr Türen platziert als benötigt werden um alle Räume zu verbinden.

Um die Anzahl der Türen auf das minimal benötigte Maß zu reduzieren werden alle überflüssigen Türen entfernt. Dazu wird jede Tür des Hauses auf ihre Notwendigkeit überprüft, indem sie entfernt wird und anschließend überprüft, ob alle Räume noch miteinander verbunden sind. Sind nichtmehr alle Räume miteinander verbunden, so wird die Tür wieder hinzugefügt. Folglich durchläuft die Platzierung der Türen und Treppen folgende Schritte:

1. Platziere Haustür in der Außenwand eines zufälligen Raums.
2. Platziere Treppen in dem Eingangsbereich des Hauses und den darüberliegenden Stockwerken.
3. Platziere in jedem Raum eine Tür in alle angrenzenden Räume.
4. Entferne so lange Türen, bis keine mehr entfernt werden können ohne die Begehbarkeit der Räume zu verlieren.

4.3 Fassade

Die Generierung der Fassade beinhaltet die Platzierung der Fenster und das Hinzufügen der einzelnen Dachbereiche. Da die Position und Höhe der Dächer für die Platzierung der Fenster relevant ist, werden die Dächer zu Beginn hinzugefügt.

4.3.1 Dächer

Über jedem Voxel des Gebäudes, dessen oberer Voxel leer ist, wird ein Dach-Voxel platziert. Da die oberste Ebene des höchsten Stockwerks mit einer Schicht Voxel abschließt, ist dort kein Platz für eine zusätzliche Schicht Dach-Voxel. Aus diesem Grund wird für die oberste Schicht Dach-Voxel ein neues Stockwerk mit einem Voxel Höhe eingefügt. Dies hat zur Folge, dass das Gebäude um ein Stockwerk in der Höhe wächst.

Dach-Voxel können je nach Aufbau des Gebäudes in allen Stockwerken und Ebenen vorkommen. Aus diesem Grund wird jeder Voxel des Stockwerkes untersucht. Nachdem alle für das Gebäude benötigten Dach-Voxel hinzugefügt wurden, werden die Voxel in einzelne Dachbereiche mit verschiedenen Richtungen und Höhen eingeteilt. Dazu wird jede Ebene des Stockwerks einzeln betrachtet.

Bei der Einteilung der Dachbereiche wird an einer der äußeren Ecken der Dach-Voxel-Ebene ein Rechteck platziert. Dieses wächst, bis es sich nicht mehr vergrößern kann, ohne die Fläche der Dach-Voxel zu verlassen oder die Voxel eines anderen Dachbereichs zu beinhalten, vergleiche Abbildung 52. Das Wachstum des Bereichs beginnt dabei zufällig auf der vertikalen oder horizontalen Achse. Ist das Wachstum auf der ersten Achse abgeschlossen, wächst das Rechteck in Richtung der zweiten Achse. Das Rechteck wächst wiederum, bis es den Dachbereich verlassen würde oder in Konflikt mit einem anderen Dachbereich gerät.

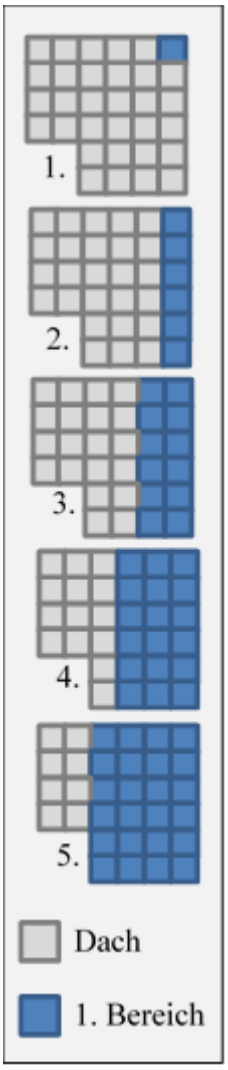


Abbildung 52: Beispielhaftes Wachsen eines Rechteckes zur Einteilung der Dachfläche in einen Dach-Bereich. Quelle: Eigene Grafik.

Dieses Vorgehen wird mit allen Ecken der Dach-Fläche wiederholt, welche bis zu diesem Zeitpunkt noch in keinen Dachbereich eingeteilt sind. Abschließend wird untersucht, ob alle Dach-Voxel Bereichen zugeteilt wurden. Wurde ein Voxel keinem Bereich zugeteilt, wird mit ihm verfahren, wie mit einer nicht zugeteilten Ecke. Es wird ein Rechteck platziert, welches so lange wächst, bis es an andere Dachbereiche anstößt. Dieses Vorgehen wird so lange wiederholt, bis alle Dach-Voxel in Bereiche aufgeteilt wurden. Dieses Vorgehen ist in Abbildung 52 noch einmal verdeutlicht.

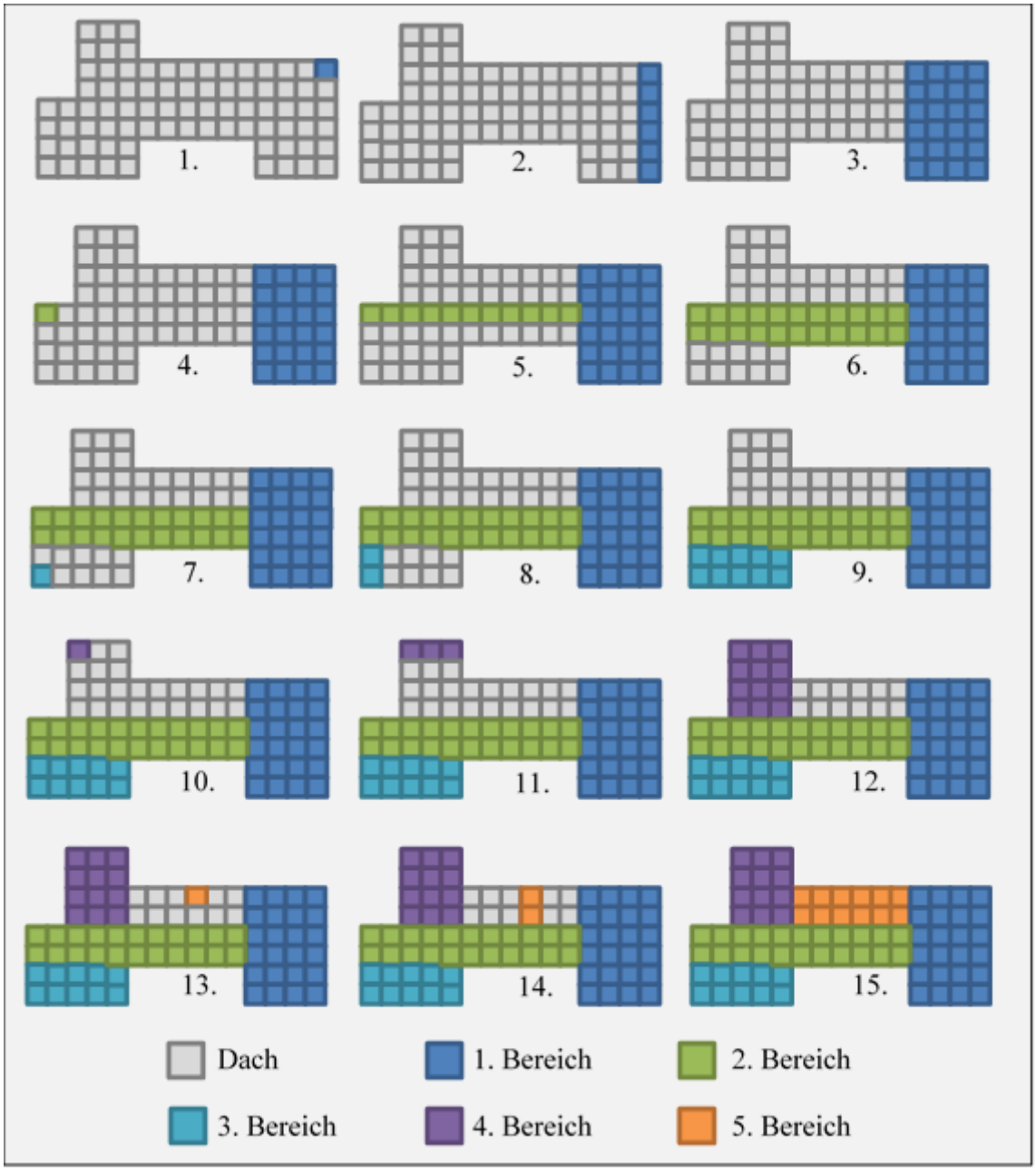


Abbildung 53: Beispielhafte Aufteilung einer Dachfläche in einzelne Bereiche mittels wachsender Rechtecke. Quelle: Eigene Grafik.

Die Richtung und Höhe eines Dachbereichs wird von seiner Form und Größe bestimmt. Die Richtung entspricht dabei der Orientierung der längsten Seite des Dachbereichs und die Höhe basiert auf der Größe des Bereichs. Umso größer die Fläche des Bereichs ist, umso höher ist das Dach. Die Höhe wird dabei in Voxel angegeben. Ein Dach-Voxel kann somit zwischen einem und mehreren Voxeln hoch sein. Dies ist für die spätere Platzierung der Fenster wichtig um eine verdeckte Sichtlinie erkennen zu können.

Abschließend wird auf den oberen Seiten der Dach-Voxel noch Dachdekoration platziert. Bei der Platzierung von First-Dekoration muss darauf geachtet werden, dass sie ausschließlich auf den Dachfirsten platziert werden können. Dies bedeutet, dass sie in der mittleren Voxel-Reihe parallel zur Richtung des Dachbereiches platziert werden müssen.

4.3.2 Fenster

Bei der Platzierung der Fenster ist zu beachten, dass jeder Raum mindestens ein sinnvoll platziertes Fenster besitzen muss. Die Basis der Fensterplatzierung bildet dabei die Evaluierung der Fassadengrundvoraussetzungen. Da die Bewertung der Fensterplatzierung ganz von der Evaluierungsfunktion übernommen wird, können die Fenster zufällig platziert werden.

Dazu wird ein beliebiger Teil der Außenwand eines Raums ausgewählt und ein Fenster platziert und evaluiert. Entspricht die Fensterposition nicht den Grundvoraussetzungen, wird es entfernt und an einer anderen Stelle der Wand platziert.

Dies hat allerdings den Nachteil, dass die Fenster zufällig über der Fassade verstreut wirken können. Um dies zu verhindern erhalten die einzelnen Wandabschnitte folgende Priorisierung:

1. Überhang
2. Ecke eines Überhanges
3. Normale Wand
4. Ecke einer normalen Wand

Dies führt im Gegensatz zu einer rein zufälligen Platzierung der Fenster zu einer Ausnutzung der Überhänge und verhindert direkt an Ecken angrenzende Fenster, wenn sie nicht nötig sind, vergleiche Abbildung 54.

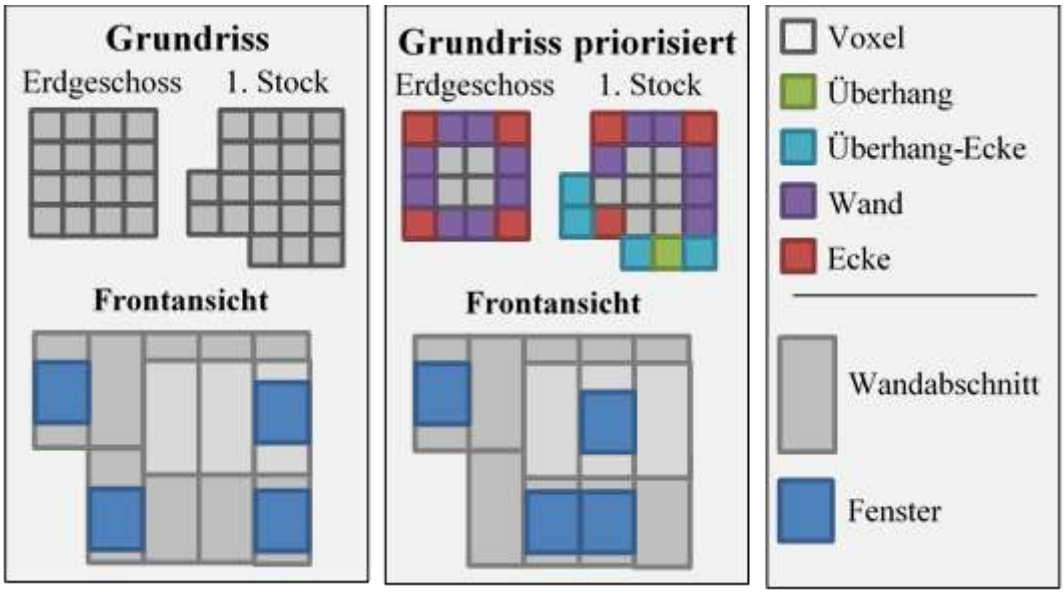


Abbildung 54: Visualisierung einer Fensterplatzierung mit und ohne Priorisierung der Wandabschnitte. Prioritäten: 1. Überhänge, 2. Ecken von Überhängen, 3. Normale Wände, 4. Ecken. Quelle: Eigene Grafik.

Daraus folgend durchläuft der Algorithmus die folgenden Schritte:

1. Platziere ein Fenster auf einem zufälligen Außenwandstück des Raums.
2. Evaluiere die Fensterposition.
3. Passe die Fensterposition an.
   (Basierend auf dem Ergebnis der Evaluierung)
4. Gehe zu Schritt 1.

Dies wiederholt sich so lange, bis entweder eine valide Position für das Fenster gefunden wurde, oder kein sinnvolles Fenster für diesen Raum platziert werden kann. Besitzt jeder Raum mindestens ein valides Fenster, wird eine beliebige Anzahl weiterer Fenstern hinzugefügt um den Abwechslungsreichtum der Fassade anzupassen. Dies ist der Punkt an dem ebenfalls Fenster in den Dach-Voxeln eingefügt werden können. Diese besitzen im derzeitigen Stand der Evaluierung keine Bedeutung für den Lichtzugang eines Raums, sondern sind rein dekorativer Natur.

Kann nicht für jeden Raum mindestens ein valides Fenster platziert werden, wird das Gebäude direkt verworfen und kein Versuch unternommen es zu reparieren. Dies hat den

Grund, dass nicht mit Sicherheit gesagt werden kann, ob die gescheiterte Fensterplatzierung von der Aufteilung der Innenräume oder dem grundlegenden Aufbau des Gebäudes (durch z.B. zu große Überhänge) verursacht wurde.

4.4 Variation

Nachdem die Fenster hinzugefügt wurden ist das Gebäude vollständig. An diesem Punkt kann es durch kleine Veränderungen zusätzlich variiert werden. Es ist möglich weitere Deko-Objekte zu platziert oder bestehende Voxel und deren Seiten durch Variationen dessen auszutauschen. Ein Beispiel hierfür ist das Austauschen der platzierten, generischen Fenster durch verschieden Fenster-Voxel-Seiten, welche jeweils eine andere Art von Fenster darstellen und zu Laufzeit aus den Daten der Voxel-Seite rekonstruiert werden. Das generierte Voxel-Modell stellt dabei die Repräsentation des Gebäudes innerhalb des Algorithmus dar. Die graphische Darstellung wird von einem zusätzlichen Algorithmus übernommen. Dieser interpretiert die in den Voxel gespeicherten Daten und wandelt sie in ein 3D-Modell um.

Zusammenfassung und Fazit

Ziel dieser Arbeit war es abwechslungsreiche Gebäude im Stil SKYRIMS und Lake-Towns aus dem HOBBIT (FILM) zu generieren. Zu diesem Zweck wurden Beispielgebäude der betreffenden Stile analysiert und basierend darauf eine Definition des Begriffs *Abwechslungsreichtum* im Kontext nordischer Architektur aufgestellt. Anschließend wurde das erweiterte Voxel-Modell vorgestellt, welches die generierten Gebäude repräsentiert. Um die erarbeiteten Gebäudeeigenschaften an diesem Modell nachweisen zu können wurden verschiedene Methoden entwickelt und verglichen. Diese Methoden wurden anschließend in einer Evaluierungsfunktion zusammengefasst, welche die einzelnen Bereiche des Gebäudes auf ihren Abwechslungsreichtum hin bewertet. Abschließend wurde ein simpler Generierungsalgorithmus entwickelt, welcher von der Evaluierungsfunktion geleitet abwechslungsreiche Gebäude generiert.

Dabei ist zu beachten, dass die Qualität der generierten Gebäude stark von den Regeln der Evaluierungsfunktion abhängig ist. Umso detaillierter die Regeln der Evaluierungsfunktion, desto gleichförmiger werden die erstellten Gebäude. Aus diesem Grund wurden die Evaluierungsregeln in dieser Arbeit so allgemein wie möglich gehalten um ein hohes Maß an Variation zu ermöglichen. Daraus folgt, dass nicht jedes Gebäude ein hohes Maß an Abwechslungsreichtum aufweist, dafür wird aber eine Vielzahl verschiedenster Gebäude generiert. Abschließend hat die Arbeit festgestellt, dass es möglich ist abwechslungsreiche Gebäude mittels des vorgestellten Ansatzes zu erstellen.

Die Frage, die sich an dieser Stelle stellt, ist ob die generierten Gebäude im Verhältnis zu dem Aufwand ihrer Erstellung stehen. Der vorgestellte Algorithmus ist hoch spezialisiert auf den vorgestellten Gebäudetypus und kann ohne größere Änderungen der Evaluierungsfunktion keine Gebäude eines anderen Stils generieren. Folglich ist es notwendig für jede Gebäudeart eine einzigartige Evaluierungsfunktion zu implementieren. Dies kann sehr Zeit- und Kostenintensiv sein, ist aber ein valider Ansatz zur Erstellung von Gebäuden für große Spielwelten.

In diesem Kontext können auch 3D-Artists die generierten Gebäude als Entwurfshilfe verwenden. Die Gebäude würden als Grundmodelle für die weitere Ausarbeitung dienen

und somit den Workflow des 3D-Artist vereinfachen. Dies könnte auch dazu dienen 3D-Artists die so genannte *Angst vor dem leeren Blatt*[6] zu nehmen. Sie hätten die Möglichkeit anstelle eines Neubeginns mit einem generierten Grundmodell zu beginnen und dieses anzupassen.

In Zukunft könnte man allerdings auch anstelle eines vom Nutzer definierten Generierungsalgorithmus ein neuronales Netz verwenden um die Gebäude zu erstellen. Dieses würde zuvor an Beispielen der zu generierenden Gebäude trainiert werden um ein umfassendes Verständnis des gewünschten Gebäudetypen zu erlernen. Dabei würde die Evaluierungsfunktion genutzt um das neuronale Netz zu trainieren. Hat das Netz die Eigenschaften des gewünschten Gebäudetyps erlernt, wird die Evaluierungsfunktion nicht mehr benötigt.

[6] Ein anderer Begriff für eine Kreativblockade.

Literaturverzeichnis

[1] H. Games, „No Mans Sky - Press Kit,“ Hello Games, [Online]. Available: https://www.nomanssky.com/press/. [Zugriff am 7 Februar 2017].

[2] P. Brandao und A. Coelho, „Journal for Virtual Worlds Research,“ Juli 2011. [Online]. Available: https://journals.tdl.org/jvwr/index.php/jvwr/article/view/2109. [Zugriff am 21 Januar 2018].

[3] J. Martin, „Digital Commons Trinity University - Algorithmic Beauty of Buildings Methods for Procedural Building Generation,“ 1 März 2005. [Online]. Available: https://digitalcommons.trinity.edu /compsci_honors/4/?utm_source=digitalcommons.trinity.edu%2Fcompsci_honors%2F4&utm_medium=PDF&utm_campaign=PDFCoverPages. [Zugriff am 21 Januar 2018].

[4] P. Müller, P. Wonka, S. Haegler, A. Ulmer und L. van Gool, „Procedural modeling of buildings,“ *SIGGRAPH '06 ACM SIGGRAPH Papers,* pp. 614 - 623, 2006.

[5] P. Wonka, M. Wimmer, F. X. Sillion und W. Ribarsky, „Instant Architekture,“ *ACM Transactions on Graphics, Association for Computing Machinery,* pp. 669 - 677, 22 April 2003.

[6] M. Evans, „Blog 3.0: Procedural Generation for Dummies: Building Footprints,“ 7 Mai 2016. [Online]. Available: http://martindevans.me/game-development/2016/05/07/Procedural-Generation-For-Dummies-Footprints/. [Zugriff am 2 Februar 2018].

[7] F. Marson und S. R. Musse, „Automatic real-time generation of floor planes based on squarified treemaps algorithm,“ *International Journal of Computer Games Technology,* p. Artikel No. 7, Januar 2010.

[8] J. Martin, „Procedural House Generation: A method for dynamically generating floor plans,“ *Symposium on Interactive 3D Graphics and Games,* 2006.

[9] P. Merrell, E. Schkufza und V. Koltun, „Computer-generated residental building layouts,“ *ACM Transactions on Graphics,* p. Artikel No. 181, Dezember 2010.

[10] S. Greuter, J. Parker, N. Stewart und G. Leach, „Real-Time generation of 'pseudo infinite' cities,“ *GRAPHITE '03 Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia East Asia,* pp. 87-ff, Februar 2003.

[11] R. Hwang, „CIS700 Procedural Graphics - University of Pennsylvania-Shape Grammars: procedural generation techniques for virtual cities,“ 2016. [Online]. Available: https://cis700-procedural-graphics.github.io/files/ shape_grammar_2_7_17.pdf. [Zugriff am 2 Februar 2018].

[12] Y. . I. H. Parish und P. Müller, „Procedural modeling of cities,“ *SIGGRAPH '01, Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques,* pp. 301 - 308, 2001.

[13] T. Adao, L. Magalhaes und E. Peres, Ontology-based Procedural Modelling of Traversable Buildings Composed by Arbitrary Shapes, Springer Briefs in Computer Science, 2016.

[14] G. Bugge, Stabkirchen. Mittelalterliche Baukunst Norwegens, Regensburg: Friedrich Pustet, 1994.

[15] M. Gerner, Fachwerk: Entwicklung, Gefüge, Instandsetzung, Stuttgart: Deutsche Verlags Anstalt, 1979.

[16] A. Lee, The Hobbit - The desolation of smaug. Chronicles Art & Design, London: Harper Collins Publishers, 2013.

[17] J. R. R. Tolkien, The Hobbit. or There and Back Again, United Kingdom: George Allen & Unwin, 1937.

[18] S. Havemann, „Generative Mesh Modeling,“ Braunschweig, 2005.

[19] J. L. Gross und J. Yellen, Discrete Mathematics and its Applications: Handbook of Graph Theory, New York: CRC Press, 2004.

[20] J. M. Tyler, The new stone age in northern europe, New York: Charles Scribner's Sons, 1922.

[21] H. Kleinstück, „Zehn Kugeln Eis für eine Mark: Erinnerungen an eine Kindheit in Unteruhldingen," Südkurier, 25 August 2016. [Online]. Available: https://www.suedkurier.de/region/bodenseekreis-oberschwaben/uhldingen-muehlhofen/Zehn-Kugeln-Eis-fuer-eine-Mark-Erinnerungen-an-eine-Kindheit-in-Unteruhldingen;art372496,8871363. [Zugriff am 20 Januar 2018].

Spieleverzeichnis

The Elder Scrolls V: Skyrim – Special Edition. Bethesda Game Studios. Bethesda Softworks LLC, 2016.

No Men's Sky. Hello Games. Hello Games, 2016.

Filmverzeichnis

The Hobbit: The Desolation of Smaug. Regie: Peter Jackson. Neuseeland, Vereinigtes Königreich, USA, 2013.

Abbildungsverzeichnis

Abbildung 1: Gegenüberstellung der Stabkirche Gol, der großen Halle Lake-Towns aus The Hobbit (Film) und eines Tempels des Spiels Skyrim zur Verdeutlichung des maßgeblichen Einflusses der nordischen Stabkirchen auf die Architektur der virtuellen Gebäude. Quelle Große Halle Lake-Towns: Eigens erstellter Screenshot aus The Hobbit (Film). Quelle Tempel in Skyrim: Eigens erstellter Screenshot des Spiels Skyrim. Quelle Stabkirche Gol: André Breutel, 2006. http://www.staedte-fotos.de/bild/Bauwerke~Sakrale+Bauten~Norwegen/1454/stabkirche-von-gol-zum-klassischen-erscheinungsbild.html 3

Abbildung 2: Rekonstruktion eines Wikinger-Langhauses im Freilichtmuseum von Trelleborg. In seinem Aufbau sind bereits die Grundzüge einer Stabkirche in Form von senkrecht stehenden Holzmasten als Stütze für das Dach des Umganges (A), sowie ein gestaffeltes Dach(B) mit Giebeldekoration (C) zu erkennen. Quelle: Erweiterte Grafik

von Schorle 2015. - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=46429462 4
Abbildung 3: Querschnitt der Stabkirche Gol zur Verdeutlichung des komplexen Tragewerks, welches die gestaffelte Dachkonstruktion stützt. Quelle: Micha, 2009. https://commons.wikimedia.org/wiki/File:Stabkirche-Gol-Aufriss-Querschnitt.svg ... 4
Abbildung 4: Westportal der Stabkirche Gol zur Verdeutlichung der Fassadengestaltung einer Stabkirche. A: Eckmasten. B: Giebelkreuz. C: Firstkamm. D: Drachenköpfe. Quelle: Erweiterte Grafik von Peulle, 2017. - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=60549959 5
Abbildung 5: Grundriss der Brauerei zur Verdeutlichung des asymmetrischen Grundrisses. Quelle: Eigene Grafik. 7
Abbildung 6: Die Brauerei vor den Toren Weißlaufs. Die Parallelen zur zuvor analysierten Stabkirche sind wie folgt markiert. A: Eckmasten. B: Steinfundament. C: Geschuppte Dächer. D: Drachenköpfe. Quelle: Eigens erstellter Screenshot des Spiels Skyrim. 7
Abbildung 7: Dargestellt sind alle 4 Ansichten der Brauerei. Dies dient zur Veranschaulichung der aus dem asymmetrischen Grundriss resultierenden unterschiedlichen Fassadenseiten. A: Front. B: Rückansicht. C: Linke Seite. D: Rechte Seite. Quelle: Eigene Grafik. 7
Abbildung 8: Wohnhaus der Stadt Weißlauf zur Verdeutlichung der Eigenschaften abwechslungsreicher und monotoner Gebäude. Quelle: Eigens erstellter Screenshot des Spiels SKYRIM. 8
Abbildung 9: Front- und Seitenansicht des Stadthauses einmal mit und ohne Fenstergiebel zur Verdeutlichung ihres positiven Einflusses auf die Fassaden des Hauses. A: Frontansicht mit Giebeln. B: Frontansicht ohne Giebel. C: Seitenansicht mit Giebeln. D: Seitenansicht ohne Giebel. Quelle: Eigene Grafik..... 8
Abbildung 10: Grundriss des Stadthauses zur Verdeutlichung des symmetrischen Grundrisses und der daraus folgenden Monotonie. Quelle: Eigene Grafik. 8
Abbildung 11: Ein weiters Wohnhaus Weißlaufs als Beispiel für einen gescheiterten Einsatz von Anbauten um die Monotonie des Grundrisses aufzuheben. Quelle: Eigens erstellter Screenshot des Spiels Skyrim. 9
Abbildung 12: Hier wurde das zuvor analysierte Wohnhaus durch das Hinzufügen eines Rechtecks zu seinem Grundriss verändert. Die zuvor bestehende

Achsensymmetrie des Grundrisses ist dabei unverändert geblieben. Im Vergleich mit den ursprünglichen Ansichten in Abbildung 13 kann ein gestiegener Abwechslungsreichtum des Gebäudes festgestellt werden. Die neu hinzugefügten Teile der Grafik sind mit einem leichten Punktmuster markiert. A: Neuer Grundriss. B: Daraus folgende Frontalansicht. C. Aus dem Grundriss folgende Seitenansicht. Quelle: Eigene Grafik. 9
Abbildung 13: Zu sehen sind der Grundriss, die Front- sowie Seitenansicht des Wohnhauses zur Verdeutlichung des Aufbaues. A: Grundriss. B: Frontale Ansicht. C: Rechte Seitenansicht. Quelle: Eigene Grafik. 9
Abbildung 14: Die Taverne der Stadt Rifton, im linken Bereich des Bildes, ist ein Paradebeispiel für ein Gebäude mit einem überhängenden Stockwerk das von Masten Gestützt wird. Quelle: Eigens erstellter Screenshots des Spiels Skyrim. 10
Abbildung 15: Rähmbauweise eines Fachwerkhauses. Quelle: Ulrich J. 2006. - In Anlehnung an Manfred Gerners Buch - Fachwerk: Entwicklung, Gefüge, Instandsetzung [14] 10
Abbildung 16: Dachkonstruktion der großen Halle Lake-Towns zur Verdeutlichung der Gemeinsamkeiten mit der Stabkirche Gol. Zu diesen zählen die gestaffelten Dachkonstruktion, die Dekoration in Form von Drachenköpfen und Firstkämmen. Quelle: Eigens erstellter Screenshot aus The Hobbit (Film). 11
Abbildung 17: Vergleich der Originalzeichnung Tolkiens von Lake-Town (A) mit einer Rekonstruktion eines Swiss Lake Dweller Dorfes(B) und einer alten Postkarte des Pfahlbaumuseums Unteruhldingen von 1950. Dies dient zur Verdeutlichung der möglichen Inspiration Tolkiens durch altertümliche Pfahlbausiedlungen Nordeuropas. Quellen: A - [16]. B - [21]. C – [22] 12
Abbildung 18: Am Haus Bards kann man besonders gut die windschiefen Wände(Blau), Brücken und Stege (Orange)und die Haustür im Obergeschoss (Grün)erkennen, welche die Architektur Lake-Towns auszeichnet. Quelle: Erweiterte Grafik von Alan Lee [15]. 13
Abbildung 19: Übersicht über einen Bereich Lake-Towns zur Verdeutlichung der Fragmentierung im Stadtbau und deren Wirkung. Quelle: Erweiterte Grafik von Alan Lee [15]. 13
Abbildung 20: Concept einer Häuserfront Lake-Towns von John Howe. Dieses Bild dient zur Verdeutlichung der horizontalen Bauweise einfacher Wohnhäuser und der

deutlich über den Grundriss hervorstehenden Stockwerkteile, welche die unter ihnen liegenden Stützpfeiler nutzen. Quelle: Erweiterte Grafik von John Howe [15]. 13
Abbildung 21: Concept eines Gebäudes Lake-Towns von John Howe. Hier ist besonders gut die Kombination eines einfachen Grundrisses mit einer interessanten Fassade zur Erzeugung von Abwechslungsreichtum und der Einsatz von Rundbögen (Blau) zu sehen. Quelle: Erweiterte Grafik von John Howe [15]. 14
Abbildung 22: Beispiele der von Anastasia Opara in Houdini generierten Lake-Village Gebäude, welche versuchen den Stil von Lake-Town nachzustellen. Quelle: Einzelne Bilder - Anastasia Opara 2016. http://www.therookies.co/projects/next-gen-gaming/procedural-lake-houses/ 15
Abbildung 23: Visualisierung einiger in der Datenstruktur speicherbaren Features am Beispiel eines Gebäudes. Quelle: Erweiterte Grafik von John Howe [3]. 21
Abbildung 24: Repräsentation eines Gebäudes als Voxel-Modell und erweitertes Voxel-Model. Quelle: Eigens erstellte Grafik auf Basis eines Gebäude-Konzepts von John Howe [3]. 23
Abbildung 25: Aufsicht auf den Grundriss eines Gebäudes im erweiterten Voxel-Modell zur Veranschaulichung der Innenraum-Repräsentation. In Teil A ist nur die Fassade des Gebäudes in Form der Wände, Fenster und der Tür zu sehen. In Teil B wurden die inneren Wände in das erweiterte Voxel-Modell hinzugefügt, woraus indirekt die Räume in Teil C definiert werden. Quelle: Eigene Grafik. 23
Abbildung 26: Vererbungs-Hierarchie der Voxel und Struktur-Voxel zur Verdeutlichung des Aufbaus. Quelle: Eigene Grafik. 24
Abbildung 27: Veranschaulichung der verschiedenen Sonderfälle bei Innenwänden, wenn sie nicht in beiden angrenzenden Voxeln gespeichert werden. A. Vertikale, nicht fortlaufende Wand. B: Horizontale, nicht fortlaufende Wand. C: Verschobene Ecke. Quelle: Eigene Grafik. 25
Abbildung 28: Veranschaulichung des tatsächlichen Speicherverbrauchs zweier Gebäude. Zu sehen sind der Grundriss sowie das dreidimensionale Voxel-Gitter der Gebäude. Bei dem Extremfall eines lichten Gebäudes kann der Speicherverbrauch auf einen linearen Faktor sinken. Das Gebäude besitzt in der Repräsentation aber weiterhin einem Speicherverbrauch von N^3. Quelle: Eigene Grafik. 26
Abbildung 29: Verdeutlichung der validen und invaliden Platzierung von Stockwerken und Gebäudeteilen. Gebäudeteile müssen mit der Hälfte ihrer Fläche aufliegen,

Stockwerke mit einem ¼ ihrer Fläche. Gebäudeteil-Ansicht: Draufsicht auf den Grundriss des Gebäudes. Stockwerk-Ansicht: Seitenansicht des Gebäudes. Quelle: Eigene Grafik. 30
Abbildung 30: Beispiel für valide und invalide Platzierung von kleinen Stützen und Säulen an schmalen und breiten Überhängen. Draufsicht auf die Grundrisse. Quelle: Eigene Grafik. 31
Abbildung 31: Pseudocode zu der vorgestellten Methode zum Erkennen von Überhängen ohne Stützen. Dazu wird nachgewiesen, ob der schwebende Voxel zu einem großen, gesicherten Überhang gehört. Dies ist die einzige, den Regeln entsprechende Möglichkeit eines schwebenden Voxel. Quelle: Eigene Grafik. 32
Abbildung 32: Veranschaulichung der gefundenen Ecken bei der Suche nach allen Säulen-Positionen und die daraufhin zu untersuchenden Voxel. Quelle: Eigene Grafik. 32
Abbildung 33: Zusammenfassung der möglichen Zustände der für die Zuordnung der Ecke relevanten Voxel und den jeweiligen Ergebnissen der Zuordnung. Quelle: Eigene Grafik. 33
Abbildung 34: Beispiele für eine valide und invalide Verbindung von zwei Gebäudebestandteilen in Form von Überlappungen. Quelle: Eigene Grafik. 33
Abbildung 35: Pseudocode Implementierung der Methode zum Nachweis eines zusammenhängenden Gebäudes. Quelle: Eigene Grafik. 34
Abbildung 36: Visualisierung des Vorgehens zum Finden der Symmetrieachsen eines Grundrisses. Quelle: Eigene Grafik. 36
Abbildung 37: Pseudocode Implementierung der vorgestellten Methode zum Finden aller überhängenden Voxel. Quelle: Eigene Grafik. 37
Abbildung 38: Veranschaulichung der beiden Sonderfälle, in denen ein Fenster auf ein angrenzendes Dach blickt. Quelle: Eigene Grafik. 38
Abbildung 39: Vergleich der Beiden vorgestellten Methoden zum Nachweis von Sonnenlichtzugang an verschiedenen Aufrissen von Gebäudeteilen mit unterschiedlich großen Überhängen über dem zu bewertenden Fenster. Quelle: Eigene Grafik. 39
Abbildung 40: Pseudocode Implementierung der erweiterten Methode zum Nachweis von Sonnenlichtzugang. Quelle: Eigene Grafik. 40
Abbildung 41: Beispielhafte Auswertung von zwei Fassadenseiten eines Gebäudes anhand der vorgestellten Methodik. Quelle: Eigene Grafik. 42

Abbildung 42: Pseudocode zur vorgestellten Methode zur Erkennung der einzelnen Dachbereiche eines Gebäudedaches anhand der Höhe, Richtung und Nachbarschaft der einzelnen Dach-Voxel. Quelle: Eigene Grafik. 43
Abbildung 43: Pseudocode zur vorgestellten Methode zur Erkennung der einzelnen Räume eines Gebäudes. Quelle: Eigene Grafik. 45
Abbildung 44: Verdeutlichung der Umwandlung eines Gebäudes mit verschiedenen Stockwerken und Räumen in einen Verbindungs-Graphen. Quelle: Eigene Grafik... 46
Abbildung 45: Ergebnisse des Evaluierungsalgorithmus basierend auf den gegebenen und nicht gegebenen Gebäudeeigenschaften. Quelle: Eigene Grafik. 48
Abbildung 46: Vergleich der zwei Methoden zum Verkleinern eines oberen Stockwerks. 54
Abbildung 47: Verdeutlichung aller 4 Regeln zur Platzierung von Stützen anhand mehrerer Beispiele. Quelle: Eigene Grafik. 56
Abbildung 48: Beispiele für Grundrisse mit möglichen Rechteck-Konfigurationen und den daraus folgenden Raumaufteilungen nach dem vorgestellten Ansatz 1. Quelle: Eigene Grafik. 59
Abbildung 49: Beispiele für die Vermeidung der Fragmentierung mithilfe der vorgestellten Sonderregel. Quelle: Eigene Grafik. 60
Abbildung 50: Beispiele für die Anwendung der vorgestellten Sonderregel zur Bereinigung der Fragmentierung. Ein Raum wird immer mit dem Nachbarraum zusammengeführt, mit welchem er sich die längste gemeinsame Wand teilt. Quelle: Eigene Grafik. 61
Abbildung 51: Beispiel für das von zufällig ausgewählten Voxeln ausgehende Fluten des Grundrisses zur Definierung von Räumen. Weg A: Fluten des Grundrisses ohne begrenzende Regeln. Weg B: Fluten des Grundrisses unter Beachtung einer Seitenbegrenzung. Quelle: Eigene Grafik. 63
Abbildung 52: Beispielhaftes Wachsen eines Rechteckes zur Einteilung der Dachfläche in einen Dach-Bereich. Quelle: Eigene Grafik. 65
Abbildung 53: Beispielhafte Aufteilung einer Dachfläche in einzelne Bereiche mittels wachsender Rechtecke. Quelle: Eigene Grafik. 66
Abbildung 54: Visualisierung einer Fensterplatzierung mit und ohne Priorisierung der Wandabschnitte. Prioritäten: 1. Überhänge, 2. Ecken von Überhängen, 3. Normale Wände, 4. Ecken. Quelle: Eigene Grafik. 68

Tabellenverzeichnis

Tabelle 1: Zusammenfassung der Eigenschaften nordischer Stabkirchen am Beispiel der Stabkirche Gol und ihrer Relevanz für diese Arbeit. Relevante Eigenschaften sind mit einem X markiert. Quelle: Eigens erstellte Tabelle. 6
Tabelle 2: Zusammenfassung aller von den Gebäuden Weißlaufs und Riftons abgeleiteten Eigenschaften abwechslungsreicher Gebäude. Quelle: Eigens erstellte Tabelle. 11
Tabelle 3: Zusammenfassung aller neuen, von den Gebäuden Lake-Towns abgeleiteten Eigenschaften abwechslungsreicher Gebäude. Quelle: Eigens erstellte Tabelle. 14
Tabelle 4: Zusammenfassung aller erarbeiteten Eigenschaften die den Aufbau eines abwechslungsreichen Gebäudes charakterisieren können und deren Auftreten in der Architektur der Beispiele. Je nach Häufigkeit des Auftretens in den analysierten Beispielen sind die Einzelnen Eigenschaften farblich hervorgehoben. Dunkelgrau = Immer. Hellgrau = Mehrheit. Quelle: Eigens erstellte Tabelle. 16
Tabelle 5: Zusammenfassung der erarbeiteten Eigenschaften, welche die Fassade eines abwechslungsreichen Gebäudes charakterisieren und deren Auftreten in den Beispiele. Je nach Häufigkeit des Auftretens in den analysierten Beispielen sind die Einzelnen Eigenschaften farblich hervorgehoben. Dunkelgrau = Immer. Hellgrau = Mehrheit. Quelle: Eigens erstellte Tabelle. 18
Tabelle 6: Vergleich der 2 Score-Funktionen anhand von 3 Beispielen zur Verdeutlichung der Bedeutung der Differenz in der Bestandteil-Bewertung und deren Einfluss auf den Score. S = Score. A = Aufbau. F = Fassade. Quelle: Eigene Tabelle. 49

Anhang

Neben dieser Arbeit wurde ein Teil des vorgestellten Algorithmus beispielhaft in einem Demo-Projekt implementiert. Dabei wurden die folgenden Bestandteile der vorgestellten Evaluierungsfunktion und des Generierungsalgorithmus implementiert:

Evaluierungsfunktion:

- Vollständige Evaluierung des Gebäudeaufbaus, siehe Kapitel *3.1.1 Aufbau*.
    - Identifikation von schwebenden Stockwerken
    - Überprüfung aller Überhänge auf korrekte Stützung.
    - Lokalisieren schwebender Gebäudeteile.
    - Bewertung der Grundrissrechtecke (Anzahl, Symmetrie)
    - Vergleichen der Stockwerkgrößen.
    - Bewertung der Überhanganzahl.
- Teile der Innenraumevaluierung, siehe Kapitel *3.2 Innenräume*.
    - Erkennen von Räumen.
    - Erreichbarkeit aller Räume auf einem Stockwerk.
    - Einhalten der Mindestgröße von Räumen

Generierungsalgorithmus:

- Vollständige Generierung der Grundrisse, siehe Kapitel *4.1 Grundriss.*
    - Erstellen eines Erdgeschosses.
    - Erstellung belieb vieler weiterer Stockwerke.
    - Platzierung aller notwendigen Stützen für Überhänge.
- Ansatz der Innenraumgenerierung basierend auf Grundrissrechtecken, wie in Kapitel *4.2.1.2 Korrektur der Fragmentierung* beschrieben.
- Platzierung von Türen innerhalb eines Stockwerkes.

Das Interface des Demo-Projektes teilt sich dabei in 5 verschiedene Bereiche, siehe Abbildung 55. Dabei dienen Die Bereiche 1, 2, 4, 5 zur Anzeige des generierten Gebäudes und dessen Eigenschaften. In Bereich 3 sind die verschiedenen Optionen zur Beeinflussung der Anwendung. Zu diesen zählen die grundlegenden Generierungsoptionen, die Display- oder Anzeige-Optionen sowie die Parameter zur Grundrissgenerierung und der Knopf zum Start des Generierungsprozesses.

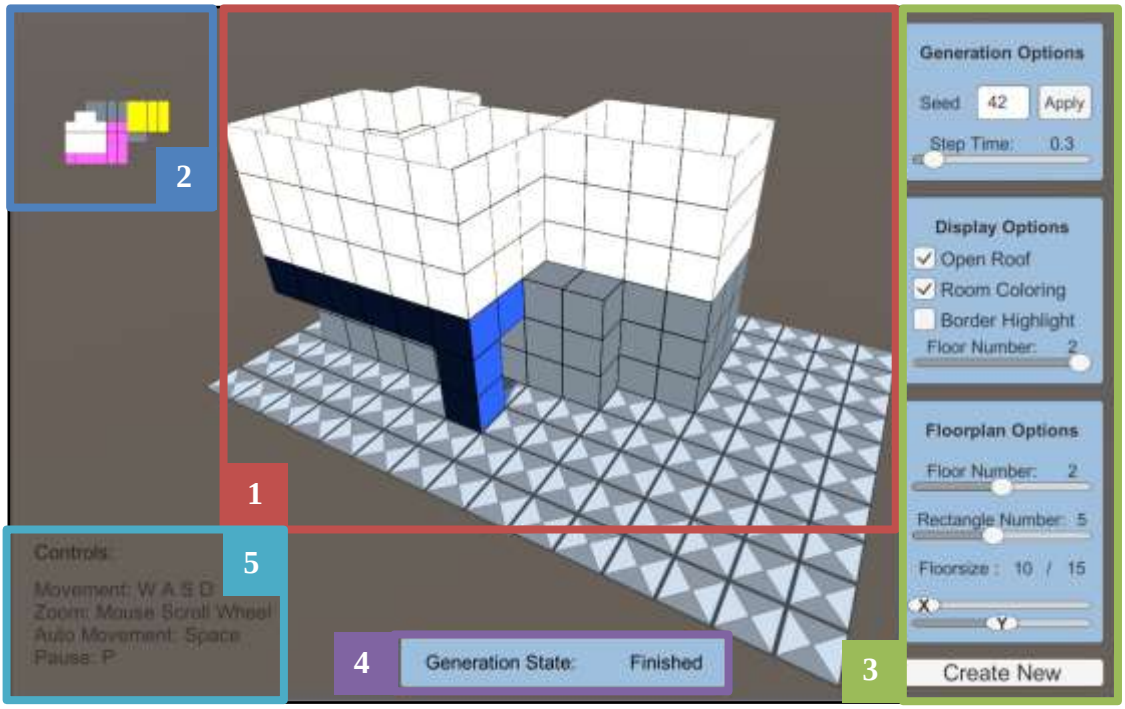


Abbildung 55: Interface des erstellten Demo-Projekts. Bereich 1: Generiertes Gebäude. Bereich 2: Top-Down Ansicht des Gebäudes. Bereich 3: Optionen zur Beeinflussung der Generierung. Bereich 4: Statusanzeige des Generierungsalgorithmus. Bereich 5: Steuerung Quelle: Eigene Grafik.

Die Bedeutung der Voxel und Voxel-Seiten ergibt sich aus ihrer Farbe. Die Aufteilung lautet wie folgt:

- **Weiß und Grau:** Die Wände der Stockwerke wechseln sich in den Farben Grau und Weiß ab um sie besser voneinander unterscheiden zu können.
- **Blau:** Stütz-Voxel.
- **Rot:** Türen.
- **Bunt:** Die Böden der einzelnen Räume können bei Bedarf in verschieden Farben eingefärbt werden. Die Farben haben dabei keine Bedeutung und können auch mehrfach pro Stockwerk vorkommen.

Generierungsoptionen:

- Der Seed zum initialisieren des Zufallszahlengenerators.
- Wartezeit zwischen den einzelnen schritten des Generierungsalgorithmus

Display- oder Anzeige-Optionen:

- Option zum Öffnen eines Stockwerks um die Innenräume betrachten zu können.
- Option zum einfärben der verschiedenen Räume.
- Option zur Anzeige der potenziellen Positionen von zusätzlichen Rechtecken in der Grundrissgenerierung.
- Anzahl der Stockwerke, die angezeigt werden soll.

Grundrissgenerierungsoptionen:

- Anzahl der zu generierenden Stockwerke.
- Anzahl der zu verwendenden Rechtecke zur Erstellung des Erdgeschosses.
- Größe des Grundstücks, auf welchem das Gebäude generiert wird.
- Starten einer neuen Generierung.

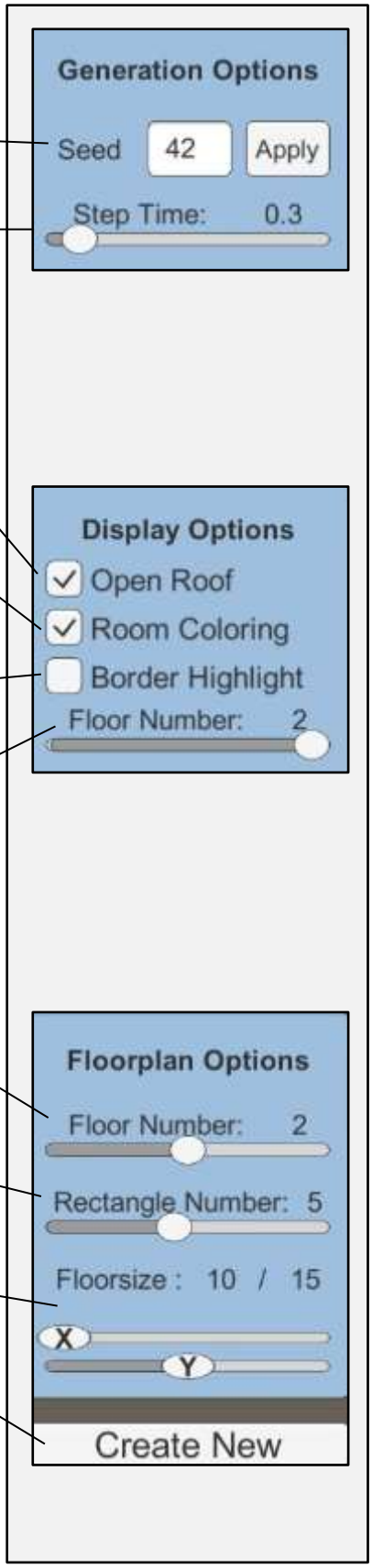


Abbildung 56: Optionen zur Beeinflussung des Demo-Projekts. Quelle: Eigene Grafik.