



**Institut für Software & Systems Engineering**  
Universitätsstraße 6a D-86159 Augsburg



Masterarbeit im Studiengang  
„Informatik“

# **Individualisierbarer Konstruktionsplaner mit automatischer Bauplandeduktion**

Sebastian Rossi





**Institut für Software & Systems Engineering**  
Universitätsstraße 6a D-86135 Augsburg



## **Individualisierbarer Konstruktionsplaner mit automatischer Bauplandeduktion**

Sebastian Rossi

Erstgutachter: Prof. Dr. Wolfgang Reif  
Zweitgutachter: Prof. Dr. Bernhard Bauer  
Matrikelnummer: 1475010  
Abgabe der Arbeit: 8. Januar 2023  
Betreuer: Constantin Wanninger



## **Erklärung**

Hiermit versichere ich, dass ich diese Masterarbeit selbstständig verfasst habe. Ich habe dazu keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Augsburg, den 8. Januar 2023

Sebastian Rossi



## **Zusammenfassung**

The lack of housing is a growing problem in modern states. High cost of skilled labor in building construction intensifies this problem. Use of full automation in this domain is quite uncommon to this day. This paper presents a solution by using a collection of different robots tasked with individual jobs. With this approach the introduced system can tackle a multitude of problems, which arise from the difficult conditions on construction site. Principles of the Semantic Plug&Play concept are used to ensure a simple application and adaptability. As proof of concept different scenarios with increasing levels of difficulty are used.



# Inhaltsverzeichnis

|   |           |
|---|-----------|
| <b>1 Motivation</b>   | <b>1</b>  |
| <b>2 Problemstellung</b>  | <b>3</b>  |
| 2.1 Bausteindeinition . . . . .   | 3         |
| 2.2 Wall-Detailing und Tiling . . . . .   | 3         |
| 2.3 Definition Bauplan . . . . .  | 3         |
| <b>3 Fallstudien</b>  | <b>5</b>  |
| 3.1 Planung und Bauplandeduktion eines LEGO Gebäude mit Einsteinmauerwerk . . . . .   | 5         |
| 3.1.1 Problemstellung . . . . .   | 6         |
| 3.2 Planung und Bauplandeduktion eines Lego Gebäude mit Verbandsmauerwerk . . . . .   | 8         |
| 3.3 Szenario mit veränderbaren Bausteintypen . . . . .  | 8         |
| 3.4 Sternchenaufgabe: Szenario mit "runden" Wänden und arbiträren (nicht rechteckige) Bausteininformen / schräge schnitte . . . . . | 8         |
| <b>4 Related Work</b>   | <b>9</b>  |
| 4.1 3D Druck und Additive Fertigung von Gebäuden . . . . .  | 9         |
| 4.2 Legeroboter . . . . .   | 9         |
| 4.2.1 Digital Plan of Brickwork Layout for Robotic Bricklaying Technology   | 9         |
| 4.3 Materialien . . . . .   | 10        |
| 4.4 Bausteine . . . . .   | 10        |
| 4.5 Wall detailing und das (3D) Bin Packing Problem . . . . .   | 10        |
| 4.6 Ludwigs Dissertation . . . . .  | 11        |
| <b>5 Konzept</b>  | <b>13</b> |
| 5.1 Modellierung . . . . .  | 13        |
| 5.2 Wall Detailing . . . . .  | 14        |
| 5.3 Regelbasierte Bauplangenerierung . . . . .  | 14        |
| <b>6 Realisierung</b>   | <b>15</b> |
| 6.1 Modellierung . . . . .  | 15        |
| 6.2 Wall Detailing . . . . .  | 15        |
| 6.2.1 Konvertieren des IFC zu BREP . . . . .  | 15        |
| 6.2.2 Überprüfen der modellierten Wände . . . . .   | 15        |
| 6.2.3 Lösen der Beziehungen . . . . .   | 20        |
| 6.2.4 Anwenden der Lösung . . . . .   | 20        |
| 6.2.5 Export . . . . .  | 20        |

*Inhaltsverzeichnis*

---

|                             |           |
|-----------------------------|-----------|
| <b>7 Proof of Concept</b>   | <b>21</b> |
| <b>8 Fazit und Ausblick</b> | <b>23</b> |
| <b>Literatur</b>            | <b>25</b> |

# 1 Motivation

Der stetig steigende Fachkräftemangel trifft auch das Baugewerbe. Laut einer Umfrage des Hauptverbandes der Deutschen Bauindustrie e.V. stufen über 75 Prozent aller befragten Unternehmen sowohl den vorherrschenden Fachkräftemangel, als auch die steigenden Energie- und Rohstoffpreise als Risiko für das eigene wirtschaftliche Wachstum ein [5]. Abbildung 1.1 illustriert die Entwicklung dieser Sorge über einen Zeitraum von etwas mehr als zwanzig Jahren. Damit ist es wenig überraschend, dass eine Bewegung weg von menschlichen Arbeitskräften hin zur Automatisierung existiert. Neben dem Fachkräftemangel stellt aber auch die geringe Effizienz von Bauvorhaben ein Problem dar, welche sich über den gesamten Planungs- und Bauprozess erstreckt. Diese Ineffizienz entsteht aufgrund der Vielzahl der an Bauprojekten beteiligten Experten und Unternehmen und ist, als *Fragmentierungsproblem der Bauindustrie* bezeichnet, ein bekanntes Problem [6]. Deshalb etablieren sich derzeit Standards, um Bauprojekte digital zu begleiten. Mit



Abbildung 1.1: Während wirtschaftliches Risiko durch eine potentiell sinkende Nachfrage nach Bauanträgen und eventuell steigender Arbeitskosten unverändert blieben oder sogar als weniger relevant bewertet wurden, ist ein deutlicher Anstieg aufgrund des vorherrschenden Fachkräftemangels und der Energie- und Rohstoffpreise zu erkennen.

diesen soll gleichzeitig die Effizienz gesteigert, die Kommunikation zwischen den einzelnen Expertenteams vereinfacht, der Arbeitsplatz "Baustelle" sicherer gestaltet und ein resourcensparender Bau ermöglicht werden [4] [7]. Gleichzeitig steigen mit der Zunahme an digitalen Informationen zu Bauprojekten, auch die Möglichkeiten diese besser zu analysieren, zu optimieren und an neue Technologien zu knüpfen. Erst dadurch wurde das seit einigen Jahren erforschte Gebiet der Additiven Fertigung von Gebäuden, etwa mit Beton druckenden Roboterarmen oder mobilen Robotern, realisierbar [3]. [TODO

## *1 Motivation*

---

nochmal reinschauen ob das einigermaßen als Quelle passt]. Obwohl es mittlerweile viele Projekte zur Additiven Fertigung von Gebäuden gibt, haben diese oft den Nachteil der Nicht-Parallelisierbarkeit der druckenden Roboter, die durch die Höhe der temporären Stützstrukturen (wie Kräne, Gerüste, Aufhängungen) eingeschränkte Bauhöhe und die vergleichsweise lange Bauzeit [TODO Quelle oder lange bauzeit weg]. [Frage wegen Kommentar: Star Wars roboterschwarm/Marsbesiedelung; soll ich hier sagen dass ein derzeit in der SciFi Szene aufgekommener Trend von organisierten Roboterschwärmen, die was bauen gar nicht so blöd ist und wir das ähnlich angehen?] Diesen Einschränkungen soll nun mithilfe eines Schwarmes bodengebundener automoner Roboter, welche gleichzeitig an dem Bauprojekt arbeiten, entgegengewirkt werden. Dabei sollen sich die Roboter auf den Mauern des Gebäudes selbst bewegen können, während sie dieses errichten. In dieser Arbeit liegt der Schwerpunkt allerdings nicht auf dem Entwickeln der Roboter selbst, sondern das Erstellen eines Bauplans für den genannten Roboterschwarm basierend auf einem 3D Modells des Gebäudes. Das Erstellen des Modells innerhalb eines nutzerfreundlichen 3D Editors stellt nicht nur die geometrischen und physikalischen Eigenschaften des Gebäudes digital bereit, sondern verschlankt auch die Kommunikation zwischen Endnutzer und Architekt oder ersetzt letzteren komplett. Gleichzeitig bildet diese Arbeit damit auch den Trend hin zur sogenannten Massenpersonalisierung ab, welcher als Nachfolgetrend zur Massenproduktion und als "heiliger Gral" der Fertigung angesehen wird . Dieser Trend ist auch für die Bauindustrie interessant, denn auch hier schafft die Möglichkeit sämtliche Kundenwünsche an ein Produkt (oder Gebäude) umzusetzen, ohne dafür spezielles Werkzeug herstellen zu müssen, neue Gewinnmöglichkeiten. Mit der Option der Modellierung des Gebäudes durch den Kunden selbst, ist die Kommunikation mit den Architekten über dessen Wünsche effizienter, da beide Parteien zusammen an dem Modell arbeiten können. Im Anschluss an den Designprozess des 3D Models des Gebäudes, soll dieses in einen Bauplan übersetzt werden, welcher alle notwendigen Informationen für den oben genannten Roboterschwarm enthält, sodass dieser das Gebäude selbstständig errichten kann [TODO ugs]. Dieses Konzept wird anhand nachfolgender Fallstudie(n) getestet.

## 2 Problemstellung

Ziel der Arbeit ist es einen Workflow zu schaffen, welcher es einem Nutzer ermöglicht ein Gebäude in einem 3D Designer zu planen, das im Anschluss in einen durch einen heterogenen Roboterschwarm ausführbaren Bauplan übersetzt wird. Dies erfolgt in mehreren Schritten, welche sich jeweils mit unterschiedlichen Fragestellungen befassen. Anhand der Teilschritte lässt sich diese umfangreiche Problemstellung logisch einteilen und die jeweiligen Kernprobleme und Fragestellungen der einzelnen Schritte werden klar.

Roboter auf Mauer -> Test nicht mit Schaumstoffbricks möglich da gewicht Lücken (Fenster, Türen)

### 2.1 Bausteindeinition

Wie können wir zu einem bestimmten Wandtyp ein Set an Bausteintypen definieren, mit welchen Wände diesen Typs gebaut werden müssen. Dabei sollen die Bausteine nicht auf Quader beschränkt, sondern beliebige Körpern sein können. Ebenfalls relevant sind eventuelle die Bausteinverbindungen, die ebenfalls Einschränkungen haben. Ein Beispiel dafür ist etwa Mörtel bei Ziegelwänden.

### 2.2 Wall-Detailing und Tiling

Wie kann man algorithmisch mit einem Set an Bausteintypen eine Wand, welche als Mesh vorliegt, vollständig erbauen und daraus einen Bauplan herleiten (Stichwort Abhängigkeitsgraph).

### 2.3 Definition Bauplan

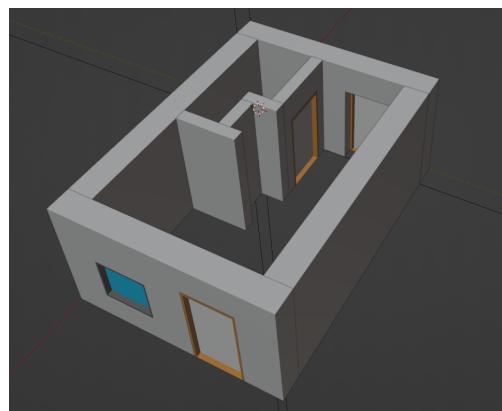
Was muss ein Bauplan konkret beinhalten? LALAB



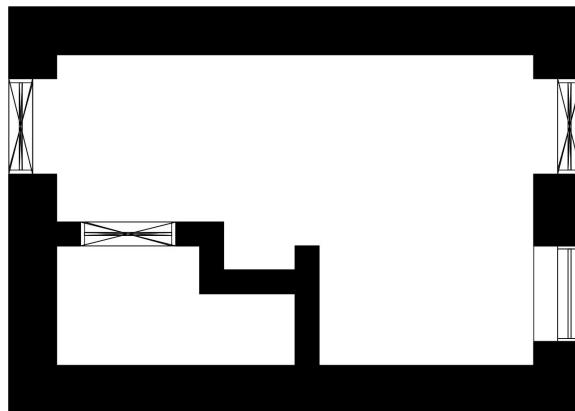
### 3 Fallstudien

#### 3.1 Planung und Bauplandeduktion eines LEGO Gebäude mit Einsteinmauerwerk

Ziel dieses Szenarios ist es, aus dem in Abbildung 3.1a dargestellten 3D Modell ein Bauplan zu generieren. Zu sehen ist der Plan eines einfachen Hauses mit einem Stockwerk.



(a) 3D Modell innerhalb von Blender.



(b) Gebäudeplan des 3D Modells.

Abbildung 3.1: Modell eines Studentenzimmers (verschiedene Darstellungsformen).

Dieses besitzt eine Eingangstür, eine Terrassentür neben einem Fenster und eine Tür, die das Badezimmer vom Hauptraum trennt. Türen und Fenster stellen eine Herausforderung für den Planungsalgorithmus dar, da der Verlauf einer ansonsten durchgängigen Wand dadurch unterbrochen wird und Lücken aufweist. Details wie Duschen, Betten, Toiletten und ähnliche Komponenten, sind für dieses Szenario irrelevant, da diese keinen Effekt auf die Struktur der Wände haben und wurden aus diesem Grund bewusst weggelassen. Das Modell wurde mithilfe der in Kapitel ?? näher behandelten Technologien erstellt und entspricht in seiner Struktur einem verbreiteten Industriestandard. Dafür wurden zwei Wandtypen definiert, die jeweils unterschiedliche Wanddicken vorgeben. So gibt es breite Außen- und dünne Innenwände. Diese entsprechen in ihren Maßen dem Raster, welches das *LEGO System* (siehe Kapitel ??) vorgibt. Für breite Wände gilt, dass diese immer zwei Noppen breit, mindestens eine Noppe lang und einen Stein hoch sein muss. Für dünne Wände hingegen gilt eine feste Breite von einer Noppe, ebenfalls eine Mindestlänge von einer Noppe und eine Mindesthöhe von einem Stein. Beide Wandtypen können nur Höhen beziehungsweise Längen aufweisen, die jeweils einem Vielfachen der Höhe oder Länge des kleinsten Legosteins aufweisen, der zum Bau der Wände verwendet werden soll (hier der 1x1 Stein). Daraus resultiert ein Raster, welches ebenfalls für die

Ausmaße und Positionen der Fenster und Türen einzuhalten ist. Dieses Raster gilt es, abhängig des ausgewählten Wandtyps, in den Editor zu integrieren, um das Modellieren solcher Gebäude nutzerfreundlich zu gestalten und nicht durch ständiges Messen und Eintragen genauer Positionen oder Maße zu unterbrechen. Nicht nur die Abmessungen

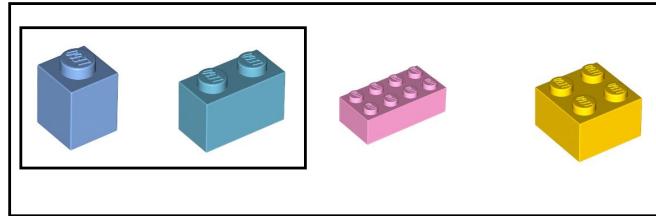


Abbildung 3.2: LEGO Steintypen für die Innen- und Außenwände. (TODO Bild ist sehr hässlich)

der Wände müssen in ein Raster fallen, auch deren Rotation wird in diesem Szenario auf 90° Schritte limitiert. Das stellt in diesem Fall eine vertretbare Enschränkung dar, da es ohnehin dem intuitiven Umgang mit LEGO Steinen und gleichzeitig dem Baustil der meisten einfachen Gebäuden entspricht. Folglich muss ein Format für die Bausteintypen entwickelt werden, aus welchen all diese Informationen abgeleitet werden können. Dieses Format muss sowohl von dem Editor selbst verwendet, als auch zur Berechnung innerhalb des Planungsalgorithmus herangezogen werden. Außerdem werden weitere Regeln benötigt, um den resultierenden Plan näher an das Vorgehen eines realen Baus zu bringen. So werden beim Errichten von Häusern zuerst die Ecken (der Schnittpunkt zweier Wände) um eine Stufe erhöht, um im Anschluss die geraden Wandabschnitte aufzufüllen. Damit wird vermieden, dass in den ohnehin schon komplexeren Eckbereichen auch noch zugeschnittene Ziegel notwenig werden. Stattdessen schneidet man diese erst zurecht, wenn sich dann eher mittig im Wandabschnitt Lücken ergeben, die kleiner sind als die vorhandenen Ziegel. Zwar wird dies im Fall von Legosteinen nie auftreten, aber es ist übersichtlicher das Problem in dem vorliegenden eingeschränkten Szenario zu beschreiben. Zusätzlich müssen Regeln in den Planungsalgorithmus eingeführt werden, die diverse Mauerwerksverbände (vorgestellt in Kapitel ??) erzwingen zu können, um damit die Stabilität und gleichmäßige Kraftverteilung innerhalb einer Wand zu gewährleisten und sich ebenfalls möglichst nah an der Realität des Mauerbaus zu bewegen.

Einschränkungen:

- \* Alle Wände verwenden ein Ziegelset, in welchem alle Ziegel die gleiche Höhe haben
- \* Alle Wände verwenden den selben Mauerwerksverband
- \* Alle anderen sich berührenden Wände stehen in einem 90 Grad Winkel zueinander

#### 3.1.1 Problemstellung

Bevor für das Modell ein passender Bauplan entwickelt werden kann, muss das sogenannte *Wall Detailing* (vgl. Kapitel 4.5) stattfinden. Darunter versteht man das Berechnen einer Bausteinkombination, die eine Wand möglichst gut abbildet. Dabei bedeutet "möglichst gut", dass die Proportionen der Bausteinkombination möglichst den der Ursprungswand entsprechen und keine nicht gewollten Lücken entstehen. In diesem Szenario wird durch

Auflegen des Rasters des LEGO Systems schon während des Modelliervorgangs sicher gestellt, dass das resultierende Modell mit Legosteinen baubar ist. Darum ist das Ziel des *Wall Detailing* Schrittes in diesem Szenario, jede Wand lückenlos mit Bausteinen zu füllen. Zusätzlich existieren folgende Einschränkungen und Eigenschaften, die für das Ergebnis dieses Szenarios gelten sollen.

**Überbindemaß** Obwohl in dem vorliegenden Modell ausschließlich Einsteinmauerwerke vorgesehen sind, gilt es die in Kapitel ?? erläuterte Regel zum Überbindemaß von Bausteinen zu beachten. Da es sich hierbei aber um LEGO Steine handelt, die wesentlich kleiner sind als die genormten Formate für Ziegelsteine, wird der für das Überbindemaß vorgesehene Mindestwert von 45mm ignoriert. Außerdem ist ein Versatz von unter 50% der Dicke eines Legosteines in den meisten Fällen nicht umsetzbar, da diese nicht frei übereinander gesteckt werden können. Darum wird für dieses Szenario ein Überbindemaß von exakt 50% der Steindicke verwendet, was den Einschränkungen des Überbindemaßes entspricht, welches einen Mindestversatz von 40% der Steindicke voraussetzt. Für die in Abbildung 3.2 dargestellten Lego Steine kann mit einem solchen Überbindemaß gleichzeitig auch das vorgesehene Raster des Lego Systems eingehalten werden, da diese alle eine gerade Anzahl an Noppen besitzen. TODO Bild von dumm gestapelter LEGO Wand zu LEGO Wand mit Überbindemaß.

**Anstoßende Wandstücke** Besonders herausfordernd ist die Einhaltung des Überbindemaßes an anstoßenden Wandstücken. Solche Wandstücke entstehen zum Beispiel an Ecken, also 90 Grad zueinanderstehende Wände oder aber an den Übergängen von Außenwänden zu Innenwänden. Wie in Abbildung 3.1a zu erkennen treten in diesem Szenario beide Fälle auf. Zunächst müssen daher alle anstoßenden Wandstücke aus dem Modell gefiltert werden. In Kapitel ?? werden gängige Praktiken zur Lösung von anstoßenden Wandstücken vorgestellt. Mit diesen Informationen muss der Planungsalgorithmus zunächst eine Lösung für solche Wandstücke finden und anschließend die restlichen Flächen mit dem Vorgehen von oben auffüllen.

**Öffnungen** Das Fenster und die drei Türen sind hier die einzigen Besonderheiten in dem Modell. Für den Mauerbau bedeutet dies an den entsprechenden Stellen Öffnungen zu lassen. Die Informationen über diese Öffnungen liegen innerhalb des Formates vor, in welchem das Modell erstellt wurde (siehe ??). So lassen sich Größe und Position leicht herausfinden und das Ergebnis der beiden obigen Schritte dementsprechend abändern.

Nun liegt eine Beschreibung des Modells mit dem zuvor definierten Set an Legosteinen vor. Als letzten Schritt gilt es daraus eine Abfolge an Bauinstruktionen zu finden, die unter Beachtung von diversen Einschränkungen zum gewünschten Ergebnis führen. Mit dieser Problemstellung hat sich Ludwig (TODO) in seiner Dissertation auseinandergesetzt (siehe Kapitel 4.6). Einschränkungen, die beim Bau von Legokonstruktionen gelten sind: Deadlock, Überhang etc. (TODO) Das Vorgehen in seiner Arbeit soll für dieses Szenario evaluiert und gegebenenfalls angepasst werden.

### **3.2 Planung und Bauplandeduktion eines Lego Gebäude mit Verbandsmauerwerk**

Kein Einsteinmauerwerk -> dicke Wände, die mit Läufern und Bindern erstellt werden müssen. Eckfälle sehr komplex (siehe Basics)

### **3.3 Szenario mit veränderbaren Bausteintypen**

Wie können wir die Bausteine veränderbar machen, sprich die Bearbeitungsmöglichkeiten während des Baus (schneiden eines Ziegels) miteinbeziehen in unser Bausteinformat. Erstmal das schneiden nur parallel zu den ebenen eines rechteckigen ziegelsteins betrachten, denn bei  $\beta$ chrägen $\beta$ schnitten entstehen neue komplexe Formen.. Wie generiert man daraus dann Baupläne -> riesiges aber cooles Optimierungsproblem! Vlt geht das richtung constraint programming? Sprich den Bauplan als Lösung für ein beschränktes Problem ansehen. Total viele Fragen, keine Ahnung wo zu beginnen aber klingt cool

### **3.4 Sternchenaufgabe: Szenario mit "runden" Wänden und arbiträren (nicht rechteckige) Bausteinformen / schräge schnitte**

Was machen wir wenn die Wand nicht perfekt mit den vorgegebenen Bausteintypen gebaut werden kann (Beispiel ein runder Turm) Bausteinverbindungen (wie Mörtel) betrachten und als *Verbindungselement* in das Bausteinformat mit aufnehmen? Wie kann man diesen so einschränken, dass nur physisch machbares ausgerechnet wird. Was wenn die Bausteine arbiträre Formen haben und nur in sehr komplexen Mustern eine "dichte" Wand ergeben -> tiling Probleme.

## 4 Related Work

### 4.1 3D Druck und Additive Fertigung von Gebäuden

### 4.2 Legeroboter

#### 4.2.1 Digital Plan of Brickwork Layout for Robotic Bricklaying Technology

In diesem Paper stellen Usmanov et al. ein generelles Vorgehen für das Erstellen eines Ziegel-Legeplans für ein als digitales Modell vorliegendes Gebäude vor [1]. Dieses Vorgehen gliedern sie in sechs Schritte:

1. Das vorliegende IFC Modell (siehe ??) nach Wandelementen durchsuchen und diese in das sogennante BREP-Format (siehe ??) konvertieren.
2. Das Aufteilen des gesamten Modells in Schichten, die der Modulhöhe des verwendeten Ziegelsteinformats entspricht.
3. Verbindungen von getrennt modellierten Wandelementen heraussuchen. Dies ist zum Beispiel an Eckstücken der Fall, da dafür oft zwei einzelne Wandelemente modelliert werden, welche in einem Winkel zueinander stehen und sich berühren. Für die nachfolgenden Schritte sind diese Verbindungen relevante Informationen.
4. Mit den Informationen der vorhergegangenen Schritte können nun für jede Schicht kritische Bereiche identifiziert werden, an welchen später ein komplexer Legevorgang von Ziegeln von Nöten ist.
5. Nun werden zunächst die kritischen Bereiche anhand einer vorher definierten Legeanleitung mit teilweise angepassten Ziegelsteinen bestückt und im Anschluss die restlichen Bereiche aller Wände mit dem ausgewählten Standardziegel aufgefüllt. In diesem Schritt werden zusätzlich Fenster- und Türstürze über deren Öffnungen in den Wänden gelegt.
6. Dieser Schritt fasst das Anzeigen als 3D Modell und Konvertieren des Resultats in eine nicht konkreter definierte Listenform zusammen. Das Ergebnis für das von den Autoren ausgewählte Beispielgebäude ist in Abbildung 4.1 zu sehen.

Besonders detailliert ist ihr Ansatz für das Finden von besagten kritischen Teilbereichen einer Wand mithilfe einiger mathematischer Gleichungen, die alle Wände zueinander in Beziehung stellen. Diese Teilbereiche sind Wanddecken, T-Kreuzungen (wie etwa der Übergang einer Innenwand an eine Außenwand) und Öffnungen innerhalb einer Wand und sind ebenfalls in Kapitel 3.1.1 dieser Arbeit als Problemstellung aufgeführt. Anhand der von ihnen zusammengetragenen Informationen konnten die Autoren erfolgreich die

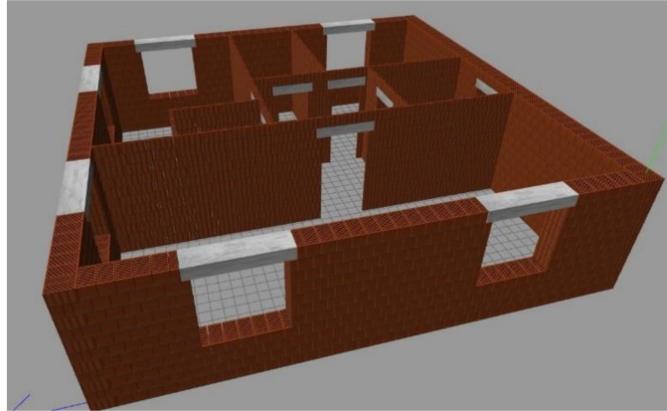


Abbildung 4.1: Ergebnis des Verfahrens zur Erstellung eines Ziegel-Legeplans nach Usmanov et al. [1].

bestmögliche Platzierung von vier Roboterarmen errechnen, die den schnellstmöglichen Bau des Gebäudes ermöglichen. Dennoch werden zum Schluss noch einige Einschränkungen ihres Verfahrens angesprochen. Vor allem die Beschränkung auf 90 Grad Ecken und das Gebunden sein an einen einzigen Standardziegel werden als besonders restriktiv wahrgenommen.

Insbesondere die Verwendung von verschiedenen Ziegelformaten und das algorithmische Finden von Lösungen an den kritischen Teilbereichen einer Wand, anstatt diese als Legeplan vorliegen zu haben, sind Ziele dieser Arbeit. Dennoch sind einige Punkte ihres Vorgehens auch zwangsläufig notwendige Schritte für diese Arbeit und werden in einer ähnlichen Form vorzufinden sein.

### 4.3 Materialien

### 4.4 Bausteine

### 4.5 Wall detailing und das (3D) Bin Packing Problem

TODO hinführen über Bin Packing hin zu Spezialfall "Wall detailing mit arbiträren Bausteinen und Eigenschaften (wie versetzen der ziegel)

TODO über bin packing schreiben, erklären paper suchen, lösungsansätze zu np hartem problem

Xu Chengran et al. haben in ihrem Paper "Optimal brick layout of masonry walls based on intelligent evolutionary algorithm and building information modeling" verschiedene Optimierungsansätze aus dem Bereich des 2D Packaging Problems getestet [2]. Konkret wurden drei Algorithmen verwendet: Differential Evolution, Particle Swarm Optimization und Neighbourhood Field Optimization. Außerdem wird ein dreiphasiges Vorgehen vorgeschlagen: Data collection, Brick layout und Data Output. Dieses Vorgehen eignet sich auch für das Finden von Bausteinkonfigurationen in dieser Arbeit, da zuerst alle

relevanten geometrischen Daten (in diesem Fall Wände, Fenster, Türen usw.) aus dem 3D Modell gesammelt werden müssen, bevor das Detailing stattfinden kann. Nach dem Optimieren der Bausteinkonfiguration muss das Ergebnis ebenfalls in ein Format gebracht werden, das für die folgenden Schritte verwendet und eventuell auch dem Nutzer angezeigt werden kann.

Soft items: <https://arxiv.org/abs/2206.15116>

Irregular Shaped items: <https://link.springer.com/content/pdf/10.1631/FITEE.1400421.pdf>

"Parametric Blockwall-Assembly Algorithms for the Automated Generation of Virtual Wall Mockups Using BIM"

## **4.6 Ludwigs Dissertation**



# 5 Konzept

In diesem Kapitel werden ...

## 5.1 Modellierung

Im Normalfall sind beim Planungsprozess eines Gebäudes oder anderer Infrastruktur eine Vielzahl an Experten aus unterschiedlichen Disziplinen beteiligt. Ein Ziel dieser Arbeit ist es allerdings eine intuitive Konstruktionsplanung zu ermöglichen, sodass eine Einzelperson mit relativ geringer Einlern- und Modellierungszeit in der Lage ist ein Gebäude zu entwerfen. Dieser Entwurf muss dennoch alle notwendigen Informationen für die anschließende Bauplandeduktion enthalten, ohne dass das Einpflegen dieser Daten viel Fachwissen voraussetzt.

Oftmals lässt sich die Komplexität einer Sache oder eines Vorgehens durch die Vorgabe von Einschränkungen reduzieren. Dabei ist es allerdings wichtig diese Einschränkungen so zu wählen, dass die damit erzielten Ergebnisse nach wie vor von Nutzen sind.

### Raster

Im Fall von Gebäude- oder besser Gebilde-Konstruktionen existieren bereits einige Beispiele, die durch Einschränkungen so stark vereinfacht werden, dass sogar Kinder damit umgehen können. Das wohl bekannteste ist das Lego System (siehe Kapitel ??). Neben dessen nützlichen Steckverbindungen, die es ermöglichen ohne Anwenden von Klebstoff oder Schrauben Steine aneinander zu befestigen, ist für diese Arbeit das dadurch vorgegebene Raster ein interessantes Konzept zur Vereinfachung der Modellierung von Gebäuden. In Kapitel ?? wurden auch schon die Begriffe *Baunennmaß* und *Baurichtmaß* eingeführt und das oktametrischen Maßsystem vorgestellt. Dies entspricht im Prinzip ebenfalls einem Raster, das aber in Realität durch die Möglichkeit des Zerschneidens von Bausteinen nicht zwingend eingehalten werden muss. Da das Vorgeben eines Rasters, dem sowohl die Größen der zu verwendenden Bausteine, als auch (im Fall des Lego Systems) deren Platzierung folgen müssen, eine Einschränkung darstellt, die im Einklang mit der Intuition vieler Menschen steht, wurde dies in das Modellierungskonzept dieser Arbeit integriert. Das Vorgeben eines einzigen, fest definierten Rasters stellt allerdings eine zu große Einschränkung dar, weshalb das Definieren verschiedener Rastergrößen möglich sein muss. So können Modelle erstellt werden, die sowohl genau dem Lego System oder dem oktametrischen Maßsystem entsprechen, als auch beliebig anderen Rastern.

Die Modellierungsumgebung kann mithilfe von Rasterinformationen eines Objektes für dessen korrekte Platzierung, Skalierung und Rotierung sorgen, indem eine solche

Transformation auf die nächstliegende Größenordnung des Rasters gerundet wird. Im Beispiel eines Rasters von  $[1.0m, 1.0m, 1.0m]$  würde demnach ein auf  $[0.9m, -0.7m, 0.1m]$  zu translatierendes Objekt an die Position  $[1.0m, -1.0m, 0.0m]$  versetzt werden. Da aber gewünschte valide Rotationen nicht aus einem derart vorgegebenen Raster interpretierbar sind, kann diese zusätzliche Information mithilfe eines kleinstmöglichen Winkelschritts angegeben werden. In den Modellen der Fallstudien aus Kapitel 3 werden zum Beispiel ausschließlich Rotationen eines Vielfachen von  $90^\circ$  verwendet, was neben den Rastern ebenfalls hinterlegt wurde. Damit kann die Modellierungsumgebung auch den Rotationen von Objekten durch Runden eine Art Raster aufzwingen.

## Wandstück

Auch die schiere Menge verschiedener Bestandteile eines Hauses ist für eine Einzelperson ohne Vorkenntnisse nicht überblickbar. Da das Ziel dieser Arbeit das Generieren von Legeplänen für Bausteine innerhalb der Wände eines Gebäudes darstellt, lässt sich diese Menge aber auf zwei wesentliche Objekttypen reduzieren: Wände und Öffnungen in Wänden. Öffnungen werden zum Beispiel für Fenster oder Türen benötigt. Da eine Wand im Prinzip ein arbiträrer geometrischer Körper sein kann, dies abzubilden aber wieder die Komplexität der Modellierung steigert, wird deren Form auf einen beliebig skalierten Quader beschränkt. Ein solcher Quader wird nachfolgend auch als Wandstück bezeichnet. Dieses besitzt demnach eine Länge, eine Breite und eine Höhe. Außerdem eine Rotation und eine Translation im Raum. Während Länge und Höhe dem Raster entsprechend beliebig gewählt werden können, so ergibt sich die Breite aus dem gewählten Bausteinformat (auch als *Modul* bezeichnet) und dem geplanten Mauerwerksverband (siehe Kapitel ??). Der Verband und das Modul können jedem Wandstück als sogenannter Wandtyp zugewiesen werden. Dadurch ist es möglich, verschiedene Arten von Wänden in einem Gebäudemodell zu verwenden.

Dann vlt Ecken und so ansprechen (?) Öffnungen ansprechen (vlt mit kleinem Diagramm?)

## 5.2 Wall Detailing

Cut and Paste Definition Wall Detailing blabla

## 5.3 Regelbasierte Bauplangenerierung

blabla

# 6 Realisierung

## 6.1 Modellierung

Blenderplugin, IFCWallTypes Module die das Raster vorgeben etc.

## 6.2 Wall Detailing

Als „Wall Detailing“ (sprich das „Detailieren von Wänden“) wird in dieser Arbeit der Vorgang ein als geometrischer Körper definiertes Wandstück in ein konkretes Mauerwerk zu überführen, bezeichnet. Innerhalb des IFC Standards werden einige mathematische/-geometrische Repräsentationen der sogenannten *IFCWall* unterstützt, um neben einfachen Boxen auch komplexere Formen abbilden zu können. Beispielsweise ist es möglich in das Modell eines Hauses zunehmend dünner werdende Wandstücke, kurvige Wandstücke oder Wandstücke, welche nur durch ein arbiträres Vieleck beschrieben werden können zu integrieren. Allerdings ist es für die Fallstudien dieser Arbeit zunächst ausreichend lediglich Wandstücke, welche als einfacher geometrischer Quader vorliegen, zu beachten. Das nachfolgende schrittweise Vorgehen weist aufgrund des annähernd gleichen Ergebnisses zwangsläufig Ähnlichkeiten zu dem von Usmanov et al. auf [1].

### 6.2.1 Konvertieren des IFC zu BREP

Den ersten Schritt stellt das Extrahieren aller notwendigen Daten aus dem vorliegenden IFC Modell dar. Für die Fallstudien dieser Arbeit sind sowohl alle Objekte des Typs *IFCWall* als auch die, etwa durch Fenster oder Türen entstehenden, daran angeknüpften Objekte vom Typ *IfcOpeningElement* (siehe ??) von Interesse. Zusätzlich werden aus den, in den *IfcPropertySets* der Wandstücke hinterlegten Daten, Informationen über das zu verwendende Modul ausgelesen. Mithilfe der Werkzeuge der in Kapitel ?? vorgestellten Python Bibliothek *ifcopenshell* (siehe ??) ist dies intuitiv möglich. TODO Sätze zum Code, Code TODO kleines Klassendiagramm von Wall/WallLayerGroup und Opening und BrickInformation

### 6.2.2 Überprüfen der modellierten Wände

#### Filtern

Da sich diese Arbeit zunächst ausschließlich mit quaderförmigen Wandstücken beschäftigt, müssen zunächst alle zuvor aus dem IFC Modell extrahierten Wandstücke auf diese Eigenschaft geprüft werden. Somit ist gewährleistet, dass lediglich passende Wandstücke

an die nachfolgenden Schritte weitergegeben werden. TODO Satz zum Code, Code iscubic

### Anwenden des Moduls

Mit dem zu jedem Wandstück festgelegten Modul werden nun alle Wandstücke in Schichten aufgeteilt. Deren Höhe entspricht im Normalfall der Höhe des jeweiligen Moduls. Lediglich die oberste Schicht kann durch falsch modellierte Wandstücke eine niedrigere Schichthöhe aufweisen. Dies ist der Fall, wenn die Gesamthöhe des Wandstücks nicht exakt einem Vielfachen der Höhe des Moduls entspricht und ein nicht aufzuteilender Rest existiert. Das Aufteilen in Schichten erleichtert es im Anschluss Berechnungen an Wandstücken durchzuführen und Beziehungen zwischen ihnen zu finden.

### Kombinieren passender Wandstücke

Eine solche Beziehung stellen etwa Wandstücke dar, die zwar in dem Modellierungsprozess des Gebäudes durch mehrere einzelne Objekte realisiert wurden, eigentlich aber eine Einheit darstellen. Daher werden in diesem Schritt alle Wandstücke miteinander verglichen und eventuell kombiniert, sodass jeweils ein gefundenes Paar durch ein einzelnes Wandstück representiert wird. Um zwei Wandstücke zu sinnvoll kombinieren zu können, müssen folgende Eigenschaften gelten:

1. Beide Wandstücke verwenden das selbe Modul und sind während der Modellierung mit den gleichen Wandtyp annotiert worden. Dies verhindert vor allem das Kombinieren unterschiedlich dicker Wände.
2. Die lokalen Z-Achsen beider Wandstücke sind parallel. Dies verhindert das Kombinieren unpassend rotierter Wandstücke. Die Überprüfung der Parallelität der Z-Achsen beider Wandstücke wird wie folgt durchgeführt:

Sei  $\vec{v}_z = [0, 0, 1]^T$  die Z-Achse und  $v_z = (v_w, v_x, v_y, v_z) = (0, 0, 0, 1)$  das dazugehörige Quaternion. Außerdem seien  $q_1$  und  $q_2$  die beiden Rotationen der Wandstücke. Dann stellen  $v_{z1} = q_1 * v_z * q_1^{-1}$  und  $v_{z2} = q_2 * v_z * q_2^{-1}$  die jeweiligen „Z-Anteile“ dieser Rotationen dar. Daraus ergeben sich die „Z-Vektoren“ wie folgt:  $\vec{v}_{z1} = [v_{z1x}, v_{z1y}, v_{z1z}]^T$  und  $\vec{v}_{z2} = [v_{z2x}, v_{z2y}, v_{z2z}]^T$ . Ist nun  $|\vec{v}_{z1} \cdot \vec{v}_{z2}| = 1$  gegeben, so sind die lokalen Z-Achsen der Wandstücke gleich oder um exakt  $180^\circ$  verdreht und damit Parallelität erfüllt.

3. Die lokalen X-Achsen beider Wandstücke sind ebenfalls parallel. Das Vorgehen zur Überprüfung entspricht dem von Punkt 2, nur dass  $v_z$  durch  $v_x = (0, 1, 0, 0)$ , also der X-Achse, ersetzt werden muss.
4. Sie stehen auf der selben Höhe oder versetzt um ein Vielfaches der gemeinsamen Modulhöhe.
5. Mindestens eine Schicht des einen Wandstücks berührt, überlappt oder befindet sich exakt eine Modulhöhe ober- oder unterhalb einer Schicht des anderen Wandstücks.

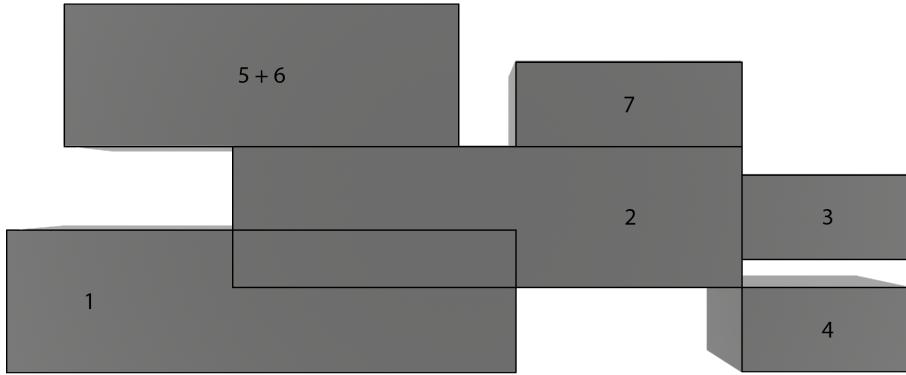


Abbildung 6.1: Eingabe eines Modells mit 7 Wänden des selben Typs, deren Mittelpunkte alle auf einer Ebene liegen.

In Abbildung 6.1 treten verschiedene Konstellationen von Wandstücken auf, die miteinander kombiniert werden müssen. Das aus Wandstück 1 und 2 gebildete Paar erfüllt alle oben genannten Eigenschaften und weist eine Teil-Überlappung auf. Somit müssen beide Wandstücke miteinander kombiniert werden. Den obersten Bereich füllen sowohl Wandstück 5 als auch Wandstück 6, sodass dort eine komplette Überlappung vorliegt. Auch diese beiden Wandstücke werden kombiniert, wobei dabei im Prinzip einfach eines verworfen wird. Zusätzlich muss dieser Bereich, wie auch Wandstück 7, mit Wandstück 2 kombiniert werden, da die beiden Paare sich an Ober- und Unterkante berühren. Ein weiteren Fall stellen seitliche, nicht überlappende Berührungen dar, wie sie zwischen Wandstück 2 und 3 zu sehen ist. Auch für dieses Paar sind alle Voraussetzungen zur Fusion erfüllt. Lediglich Wandstück 4 muss nicht mit dem Rest vereint werden, da dafür kein Paar existiert, das die obrigen Eigenschaften erfüllt. Anhand von Abbildung 6.2 kann man erkennen, dass Wandstück 4 im weiteren Verlauf des Detailings tatsächlich unabhängig betrachtet werden kann.

Während dem Kombinieren von zwei Wänden, wird ein Wandstück schichtweise in das andere überführt. Dabei werden alle Schichten paarweise miteinander verglichen, um diejenigen Paare zu finden, die die Eigenschaften aus Punkt 5 erfüllen. Ein solches Paar wird dann wie folgt miteinander verschmolzen:

```

1 def combine(layer1, layer2):
2     # get the local positions of both edges of layer1 as 3D array
3     left_edge1 = layer1.get_left_edge(relative=True)
4     right_edge1 = layer1.get_right_edge(relative=True)
5
6     # get the position of both edge of layer 2 relative to the coordinate
7     # system of layer1 as 3D array
8     left_edge2 = layer2.get_left_edge(relative=False)
9     left_edge2 = layer1.relative_position_of(left_edge2)
10
11    right_edge2 = layer2.get_right_edge(relative=False)

```

## 6 Realisierung

---

```
11 right_edge2 = layer1.relative_position_of(righ_edge2)
12
13 # get the total length both layers cover by subtracting max - min of the x
14 coordinates
14 max_x = max(left_edge1[0], right_edge1[0], left_edge2[0], right_edge2[0])
15 min_x = min(left_edge1[0], right_edge1[0], left_edge2[0], right_edge2[0])
16 total_length = max_x - min_x
17
18 # calculate the center of the resulting layer
19 left = min([a1, a2, b1, b2], key=lambda x: x[0])
20 local_center = left.copy()
21 local_center[0] += total_length / 2.0
22
23 # convert to global coordinates and return new layer
24 center = layer1.global_position_of(local_center)
25 return Layer(center, total_length)
```

Listing 6.1: Pseudocode zur Vereinigung zweier sich berührender oder überlappender Schichten von zwei nach den Voraussetzungen aus Abschnitt 6.2.2 kombinierbaren Wandstücken.

TODO erklärung Code

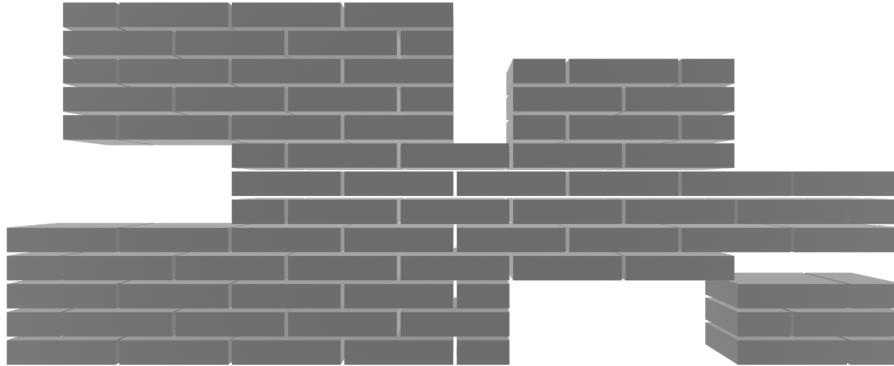


Abbildung 6.2: Ergebnis mit berücksichtigtem  $x\_offset$ .

Steht ein Wandstück in X-Richtung versetzt auf einem anderem, so ist es notwendig diesen Versatz während dem nachfolgenden Detailing zu berücksichtigen. Ignoriert man dies, kann das zu den in Abbildung 6.3 gezeigten Fehlern (zum Beispiel zwischen Wandstück 2 und 7) innerhalb des Mauerwerksverbands und damit ebenfalls zu Verletzungen des vorgeschriebenen Überbindemaßes führen (siehe Abschnitt ??). Dieser, nachfolgend als  $x\_offset$  bezeichnete Versatz ist definiert durch die Differenz zwischen der kleinsten lokalen X-Koordinate aller Schichten eines Wandstückes und der lokalen X-Koordinate der zu betrachtenden Schicht. Der daraus resultierende Wert wird später dazu verwendet den anzuwendenden Mauerwerksverband erst an der passenden Stelle zu beginnen. Dadurch erzielt man einen einheitlichen Verband über das gesamte Wandstück und

verhindert den in Abbildung 6.3 gezeigten Fehlerfall. Eine weitere Eigenschaft, die aus dem Kombinieren mehrerer Wandstücke entstehen kann, ist das vorhandensein unterbrochener Schichten oder, anders ausgedrückt, mehrerer Schichten auf einer Höhe innerhalb des resultierenden Wandstückes. Dies ist ebenfalls in Abbildung 6.1 zu sehen. Zwischen drei Schichten des Bereichs der Wandstücke 5 und 6 und des Wandstücks 7 ist eine Lücke. Durch das Einbeziehen des  $x\_offset$  können derartige Situationen jedoch ebenfalls gelöst werden, da für jedes Teilstück einer Schicht ein eigener  $x\_offset$  berechnet wird.

Nach Vereinigung aller passenden Paare reduziert sich die Zahl der ursprünglichen Wandstücke aus dem Modell in Abbildung 6.1 von 7 auf 2.

### Beziehungen finden

Nun wird die aus dem vorherigen Schritt entstandene neue Menge an Wandstücken auf weitere Beziehungen untersucht. Für das Wall Detailing relevante Beziehungen stellen Ecken, T-Kreuzungen und X-Kreuzungen dar (siehe ??). Der Grund dafür ist die Komplexität in diesen Bereichen die vorgeschriebenen Normen (wie etwa dem in Abschnitt ?? genannten Überbindemaß) einzuhalten. Deswegen existieren für jeden Mauerwerksverband eigene spezielle Eck- und Kreuzungslegepläne, die an diesen Stellen eingefügt werden müssen. Es werden wie oben nur die Wandstücke miteinander verglichen, die mit dem selben Wandtyp annotiert wurden und das gleiche Modul verwenden. Da die drei Beziehungen sich stark ähneln, besitzen sie einige geteilte Eigenschaften:

1. Die lokalen Z-Achsen beider Wandstücke sind parallel. Das Vorgehen zur Überprüfung ist dasselbe wie in Abschnitt 6.2.2.
2. Sie stehen auf der selben Höhe oder versetzt um ein Vielfaches der gemeinsamen Modulhöhe.

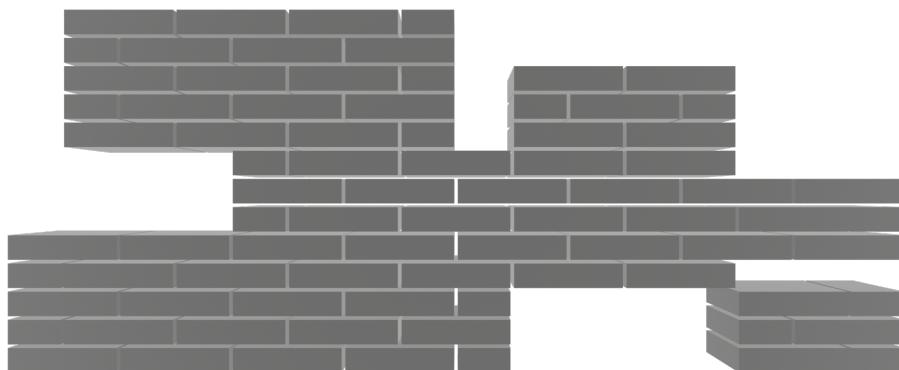


Abbildung 6.3: Ergebnis mit ignoriertem  $x\_offset$ .

3. Mindestens eines der beiden Wandstücke endet auf einem anderen, sodass mindestens eine Schicht das andere Wandstück direkt berührt.

Sind diese Eigenschaften erfüllt, werden für jede durch die einzelnen Schichten der Wandstücke vorgegebene Höhe Schnittpunkte zwischen den beiden Wandstücken errechnet. Im Falle einer einfachen Ecke oder einer T-Kreuzung, existiert nur ein einzelnes Paar an Wandstücken, deren Schichten sich in jedem Höhenschritt des Moduls im Mittelpunkt der Ecke schneiden. Der Unterschied ist lediglich die Stelle des Schnittpunkts relativ zu den beiden Wandstücken. Liegt der errechnete Schnittpunkt bei beiden Wandstücken näher als Wandbreite/2 an einer der beiden Außenkanten handelt es sich um eine Ecke. Falls der Schnittpunkt bei einem der beiden Wandstücke allerdings mindestens Wandbreite/2 innerhalb des Wandstücks liegt, so handelt es sich um eine T-Kreuzung. Werden für zwei unterschiedliche Paare die selben Schnittpunkte errechnet, so kann es sich entweder um eine mit drei Wandstücken modellierte T-Kreuzung oder X-Kreuzung handeln. Teilen sich zwei T-Kreuzungen die selben Schnittpunkte, so sind alle Voraussetzungen für eine X-Kreuzung gegeben. Liegt der Schnittpunkt des Wandstück-Paars mindestens Wandbreite/2 innerhalb beider Wandstücke, so handelt es sich um eine X-Kreuzung - modelliert aus zwei sich schneidenden Wandstücken. TODO Bilder

### 6.2.3 Lösen der Beziehungen

Je nach gewählten Verband müssen z.B. Ecken (deren Baupläne im vornherein definiert wurden) so angeordnet werden, dass die dazwischenliegenden Wandstücke lückenlos eingefüllt werden können. Dafür muss ein sogenannter "plan\_offset" für jede Wand und jede Ecke gefunden werden. Dieser gibt an, an welchem Index der Bauplandefinition das Wandstück (von unten) beginnen muss, um insgesamt einen einheitlichen Wandkörper ohne Lücken zu bilden. Voraussetzung dafür ist natürlich ein passender Eckplan zu dem gewählten Verband.

Schwierigkeiten: Eckpläne insgesamt, Ecken ragen in die sie bildenden Wände hinein. Diese müssen dementsprechend verkleinert werden.

### 6.2.4 Anwenden der Lösung

Ablauen aller Ecken und Wände und einsetzen der Ziegel gemäß den gefundenen Lösungen.

### 6.2.5 Export

Abhängigkeitsgraph und Ontologie für Regelwerk

## **7 Proof of Concept**



## **8 Fazit und Ausblick**



# Literatur

- [1] Vjačeslav Usmanov, Jan Illetško und Rostislav Šulc. „Digital Plan of Brickwork Layout for Robotic Bricklaying Technology“. In: *Sustainability* 13.7 (Apr. 2021), S. 3905. doi: 10.3390/su13073905. url: <https://doi.org/10.3390/su13073905>.
- [2] Chengran Xu u. a. „Optimal brick layout of masonry walls based on intelligent evolutionary algorithm and building information modeling“. In: *Automation in Construction* 129 (Sep. 2021), S. 103824. issn: 0926-5805. doi: 10.1016/J.AUTCON.2021.103824.
- [3] Kathrin Dörfler u. a. „Additive Manufacturing using mobile robots: Opportunities and challenges for building construction“. In: *Cement and Concrete Research* 158 (Aug. 2022), S. 106772. issn: 0008-8846. doi: 10.1016/J.CEMCONRES.2022.106772.
- [4] *BIM for Health and Safety in Construction* | Autodesk University. <https://www.autodesk.com/autodesk-university/article/BIM-Health-and-Safety-Construction-2017>. (Accessed on 04/18/2023).
- [5] *Fachkräftemangel und Rohstoffpreise – Die Deutsche Bauindustrie*. <https://www.bauindustrie.de/zahlen-fakten/bauwirtschaft-im-zahlenbild/fachkraeftemangel-und-rohstoffpreise>. (Accessed on 03/31/2023).
- [6] Mohd Nasrun u. a. „Impact of Fragmentation Issue in Construction Industry: An Overview; Impact of Fragmentation Issue in Construction Industry: An Overview“. In: (). doi: 10.1051/C. URL: <http://dx.doi.org/10.1051/matecconf/20141501009>.
- [7] *Top 10 Benefits of BIM in Construction*. <https://bim360resources.autodesk.com/connect-construct/top-10-benefits-of-bim-in-construction>. (Accessed on 04/18/2023).