

Technische Universität München


TUM School of Engineering and Design

Lehrstuhl für Computergestützte Modellierung und Simulation

Graphen-basierte Repräsentation eines BIM-Modells zur Erstellung von Trainingsdatensätzen für Deep Learning Methoden

Bachelorthesis

für den Bachelor of Science Studiengang Bauingenieurwesen

Autor:	Sebastian Bayer
Matrikelnummer:	
Prüfer:	Prof. Dr.-Ing. André Borrmann
Betreuer:	Jan Clever, Sebastian Esser
Ausgabedatum:	01. Dezember 2022
Abgabedatum:	02. Mai 2023

Abstract

The objectives of this work are the design and creation of a graph-based representation of a building, suitable for training Deep Learning (DL) methods. The informational background for the graph-based representation is a building model created with Building Information Modeling (BIM). In this context, the research question is raised: How can the graph-based representation of a BIM model be developed and created to support training data for Deep Learning methods?

In order to answer this research question, an initial investigation was started to determine which information should be selected in the building representation to make the structure of the graph-based representation as precise as possible. The result of the work is a method that can be easily applied to existing BIM models and generates a network representation from them. Finally, a visual representation of an example model is explained.

Based on the generated network representation training datasets for e.g. DL methods can be created.

Zusammenfassung

Die Ziele dieser Arbeit sind der Entwurf und die Erstellung einer Graphen-basierten Repräsentation eines Gebäudes, welche sich zum Training von Deep Learning (DL) Methoden eignet. Als Informationsgrundlage für die Graphen-basierte Darstellung dient ein Gebäudemodell, das mit Building Information Modeling (BIM) erstellt wurde. In diesem Zusammenhang wird folgende Forschungsfrage gestellt: Wie kann die Graphen-basierte Repräsentation eines BIM-Modells zur Unterstützung von Trainingsdaten für Deep Learning Methoden entwickelt und erstellt werden?

Zur Beantwortung der Forschungsfrage wurde zunächst untersucht welche Informationen in der Gebäuderepräsentation abgebildet werden müssen, um den Aufbau der Graphen-basierten Darstellung so präzise wie möglich zu gestalten. Das Ergebnis der Arbeit ist eine Methode, die leicht an bestehenden BIM-Modellen angewendet werden kann und eine Netzwerkdarstellung aus diesen erzeugt. Final wird eine visuelle Darstellung eines Beispielmodells erläutert.

Auf Grundlage der erstellten Netzwerkdarstellung können beispielsweise Trainingsdaten für DL-Methoden erstellt werden.

Inhaltsverzeichnis

Abstract	II
Zusammenfassung	III
Inhaltsverzeichnis	IV
Abbildungsverzeichnis	VI
1 Einführung und Motivation	1
1.1 Motivation	1
1.2 Ziel der Arbeit.....	2
1.3 Aufbau der Arbeit.....	2
2 Grundlagen und aktueller Stand	3
2.1 Personenstromsimulation	3
2.1.1 Einschränkungen	4
2.2 Künstliche Intelligenz	5
2.2.1 Begriffserklärung und Funktion	5
2.2.2 Training von Deep Learning Methoden.....	7
2.2.3 Anwendung von Deep Learning in der Bauindustrie	8
2.3 Wahl einer geeigneten Darstellungsform für Gebäude	9
2.4 Graphen-basierte Repräsentation.....	11
2.4.1 Aufbau	11
2.4.2 Hierarchische Graphen-basierte Repräsentation	13
2.5 Building Information Modeling.....	15
2.6 Herleitung der Forschungsfrage.....	15
3 Methodik und Vorgehen	16
3.1 Informationsgewinnung.....	16
3.1.1 Benötigte Informationen.....	16
3.1.2 Informationen aus BIM-Modellen auslesen.....	17
3.2 Gestaltung hierarchische Graphen-basierte Repräsentation	18
3.2.1 Aufbau der hierarchischen Ebenen.....	18

3.2.2	Enthaltene Informationen	21
4	Implementierung und technische Umsetzung	24
4.1	Verwendete Programme und Dateiformate	24
4.1.1	Autodesk Revit	24
4.1.2	Revit API	24
4.1.3	Python	25
4.1.4	RevitPythonShell	25
4.1.5	NetworkX und GraphML	28
4.2	Erzeugung der Repräsentation	29
4.2.1	Informationsgewinnung über Revit API – Python in der Revit-Umgebung ..	29
4.2.2	Erstellung des Graphen mit NetworkX – Python in der PC-Umgebung	32
4.2.3	Hierarchische Graphen-basierte Repräsentation – Ergebnis	33
5	Anwendungsbereich und Anwendbarkeit	36
5.1	Vorteile der entwickelten Methode	36
5.2	Hierarchieebenen	36
5.3	Analyse zur technischen Umsetzung	37
6	Zusammenfassung und Fazit	39
	Literaturverzeichnis	40

Abbildungsverzeichnis

Abbildung 2-1: Unterschiede der Begriffe KI (hier: engl. Artificial Intelligence), ML (hier: engl. Machine Learning), DL (K. Aggarwal et al. 2022).....	5
Abbildung 2-2: Neuronales Netzwerk eines DL-Modells (E. Kavlakoglu 2020).....	6
Abbildung 2-3: Grundriss einer Wohnung mit Bemaßung.....	10
Abbildung 2-4: Topologische Netzwerkdarstellung derselbe Wohnung wie in Abbildung 2-3	11
Abbildung 2-5: Darstellung eines Graphen-basierten Netzwerkes (Knoten & Kanten)	12
Abbildung 2-6: Hierarchische Innenraumdarstellung mit mehreren Ebenen (A. Kontarinis et al. 2021)	14
Abbildung 3-1: Die drei hierarchischen Ebenen, mit den jeweiligen Knoten.....	19
Abbildung 3-2: Aufbau der Gebäude-Ebene – Darstellung eines Gebäudes.....	19
Abbildung 3-3: Aufbau der Stockwerk-Ebene – Darstellung mehrerer Räume.....	20
Abbildung 3-4: Aufbau der Raum-Ebene – räumliche Darstellung eines Raumes....	20
Abbildung 3-5: UML-Diagramm - Hierarchische Graphen-basierte Darstellung eines Gebäudes.....	21
Abbildung 4-1: RevitPythonShell innerhalb von Revit.....	26
Abbildung 4-2: Interaktive Python Konsole innerhalb von Revit	26
Abbildung 4-3: Darstellung zur Unterscheidung der Python Revit- und PC-Umgebung	27
Abbildung 4-4: Die vier NetworkX-Klassen inklusive ihrer definierten Eigenschaften (Dokumentation zu NetworkX 3.1 2023).....	28
Abbildung 4-5: Ablauf der Methode zur Erstellung einer Graphen-basierten Repräsentation.....	33
Abbildung 4-6: BIM-Modell eines zweistöckigen Gebäudes – Autodesk Revit	34
Abbildung 4-7: Grundrisse (EG und 1.OG) des Gebäudes aus Abbildung 4-6 mit farblich markierten relevanten Bereichen	34
Abbildung 4-8: Graphen-basierte Darstellung des zweistöckigen Gebäudes aus Abbildung 4-6 und 4-7	35

1 Einführung und Motivation

Für den Entwurf und die Planung eines Gebäudes sind Personenstromsimulationen von Bedeutung, um das Verhalten zukünftiger Nutzer zu analysieren (SimPlan AG 2022). Durch Erkenntnisse solcher Simulationen kann der Entwurf des zu planenden Gebäudes den späteren Nutzern angepasst werden. X. Yang et al. (2022) untersuchten beispielsweise das Verhalten von wartenden Passagieren, wie diese sich auf U-Bahn-Bahnsteigen verteilen. Außerdem können Simulationen dabei helfen, die Sicherheit zu steigern und potenzielle Mängel bereits in frühen Planungsphasen aufzuzeigen. Q. Liu et al. (2022) untersuchten Menschenmengen in komplexen Umgebungen wie Hochhäusern oder Stadien und wie sich diese in solchen Gebäuden im Falle einer Evakuierung verhalten.

Ein Problem von solchen Simulationen ist der große Bedarf an Zeit und Computerleistung (A.-F. Lee und P.-H. Tsai 2023). Durch die Simulation erkannte Konflikte stehen somit nicht sofort zur Verfügung, wodurch die Entwicklung und mögliche Änderungen am Gebäudeplan verzögert werden. Dieser Zeitbedarf könnte bei der Simulierung von Personenströmen durch den Einsatz von Künstlicher Intelligenz (KI) reduziert werden. Die KI könnte bezüglich der zu erwartenden Personenströme innerhalb eines zu planenden Gebäudes schnellere Einschätzungen ermöglichen.

1.1 Motivation

KI wird bereits unter anderem im Gesundheitswesen, in der Finanzindustrie, der Automobilindustrie, dem Schulwesen und dem Rechtswesen eingesetzt. In diesen Bereichen wird KI schon heute unterstützend beispielsweise für Prozessoptimierung, Produktentwicklung und Effizienzsteigerung angewendet (J. Lammers 2023).

Durch die zunehmenden Fähigkeiten von KI ändert sich die Herangehensweise vieler Problemstellungen und deren Lösung. Auch der Bausektor beschäftigt sich immer stärker mit der Möglichkeit, Prozesse und Probleme durch die Unterstützung von KI zu lösen (S. Kristombu Baduge et al. 2022).

Die Arbeitsweise von KI erfordert große Mengen an Daten, die erstellt und formatiert werden müssen, da KI auf Grundlage vorangegangener Berechnungen neue Bewertungen extrapoliert (J. Lammers 2023). Im Bauwesen stehen Planungsdaten oft als

Modell, auf Basis von Building Information Modeling (BIM), zur Verfügung. Damit solche Gebäudemodelle zum Training von KI verwendet werden können, müssen die hierfür benötigten Informationen in ein anderes Format gebracht werden, beispielsweise eine Graphen-basierte Repräsentation.

Die Erstellung von Graphen-basierten Repräsentationen auf Grundlage von BIM-Modellen kann zu präziseren Trainingsdaten für KI führen. Dadurch könnten verbesserte Einschätzung durch KI zum Verhalten von Personenflüssen in Gebäuden getroffen werden.

1.2 Ziel der Arbeit

In dieser Bachelorarbeit wird eine Methode entwickelt, die Graphen-basierte Repräsentationen aus BIM-Modellen erzeugt. Die Ziele der Arbeit sind, eine aussagekräftige Darstellung von Gebäuden für unterschiedliche Anwendungsbereiche zu entwickeln und zu erstellen. Dabei wird das Vorgehen anhand eines konkreten Anwendungsbeispiels beschrieben, der Personenstromsimulation während einer Evakuierung.

1.3 Aufbau der Arbeit

Das zweite Kapitel vermittelt die wichtigsten Grundlagen zu den Themen Personenstromsimulation, KI, Graphen-basierte Repräsentation und BIM.

Im dritten Kapitel werden das Vorgehen und die Methodik bei der Erstellung der Graphen-basierten Netzwerkdarstellung erläutert, nachdem alle erforderlichen Informationen vorgestellt wurden, die aus dem BIM-Modell ausgelesen werden müssen. Zudem wird auf die Gewinnung der einzelnen Informationen eingegangen.

Das vierte Kapitel präsentiert die Implementierung und technische Umsetzung der Erstellung der Graphen-basierten Netzwerkdarstellung mit den jeweils verwendeten Programmen und Dateiformaten. Dabei wird unter anderem die Verwendung benötigter Plugins und Packages näher beschrieben.

Der Anwendungsbereich wird im fünften Kapitel angesprochen inklusive den Vorteilen der Methoden bezüglich der Anwendbarkeit auf bestehende Modelle. Außerdem erfolgt eine kritische Auseinandersetzung mit der entworfenen Netzwerkdarstellung und deren Erstellungsmethode.

2 Grundlagen und aktueller Stand

Die Entwicklung einer geeigneten Graphen-basierten Repräsentation von Gebäuden setzt ein grundlegendes Verständnis über die Themen Personenstromsimulation und Künstliche Intelligenz voraus. Zudem wird auf die Wahl eines geeigneten Modells in Verbindung mit Graphen-basierten Repräsentationen eingegangen.

2.1 Personenstromsimulation

Mit Personenstromsimulationen lassen sich Verhaltensweisen von Einzelpersonen und Menschenmassen nachahmen, die realem Verhalten ähneln, um daraus Erkenntnisse bezüglich deren Entscheidungen zu erhalten (D. H. Biedermann et al. 2021).

Personenstromsimulationen basieren auf Suchalgorithmen, die neben dem Weg auch die Zeit einer Person berücksichtigen, welche für eine Strecke in Anspruch genommen wird (S. Pu und S. Zlatanova 2005). Suchalgorithmen berechnen in diesem Fall die Zeit, den ein virtueller Fußgänger für den kürzesten oder schnellsten Weg benötigt. (D. H. Biedermann et al. 2021)

Bei einer solchen Simulation wird das Modell, das anfänglich nur aus einem geometrischen Grundriss besteht, mithilfe von Variablen wie beispielsweise der Dichte des Fußgängeraufkommens, der Geschwindigkeit und den Bewegungsarten der Personen angepasst. Diese Variablen sind keine statischen Werte, sondern können im Verlauf der Simulation verändert werden. (D. H. Biedermann et al. 2021; H. Dong et al. 2020)

Grundsätzlich wird innerhalb der Simulation die Berechnung für die Zeit einer Strecke mehrmals durchgeführt. Bei jedem Durchgang rechnet der Suchalgorithmus mit veränderten Zufallszahlen der einzelnen Variablen. Dadurch ergibt sich bei jeder Berechnung ein anderes Ergebnis, das auf Basis vorheriger Berechnungen optimiert wird. Bei der Ermittlung des endgültigen Simulationsergebnisses wird von sämtlichen zuvor gewonnenen Ergebnissen der Durchschnitt berechnet. Erst durch die Berechnung des Durchschnitts kann eine Aussage hinsichtlich der Personenströme getroffen werden. Das Ergebnis einer einzelnen Berechnung hat keine Aussagekraft und kann somit auch nicht bewertet werden. Um ein möglichst aussagekräftiges Ergebnis der Perso-

nenstromsimulation zu erhalten, werden die einzelnen Berechnungen so lange wiederholt, bis sich der Durchschnitt nur noch marginal verändert. (D. H. Biedermann et al. 2021)

2.1.1 Einschränkungen

Aufgrund der großen Anzahl an Berechnungen wird für Personenstromsimulationen eine große Menge an Rechenleistung benötigt. Je genauer und aussagekräftiger eine Simulation wird, desto mehr Rechenleistung und Zeit wird für die Berechnung benötigt. (A.-F. Lee und P.-H. Tsai 2023)

Eine weitere Einschränkung ist, dass es aktuell noch keinen Datenstandard für Personenstromsimulations-Software gibt. Falls Personenströme in einem Gebäude simuliert werden sollen, muss vorab eine Speicherung des ursprünglichen Datenformates in das vom Simulationsprogramm benötigte Datenformat erfolgen. Dabei wird auf Datenstandards und Modelle zurückgegriffen, die weniger Daten enthalten als das anfängliche BIM-Modell. (accu:rate - Institute for crowd simulation 2023)

Ergebnisse von Personenstromsimulationen können schon bei kleinen Planänderungen am ursprünglichen Gebäudemodell variieren. Deshalb muss in diesem Fall die Personenstromsimulation erneut durchgeführt werden, um ein aktuelles Simulationsergebnis zu erhalten. Dabei muss sowohl die Umwandlung in das richtige Datenformat als auch die zeitintensive Simulation jedes Mal wiederholt werden.

Für eine präzise Darstellung eines Gebäudes und um die Datenmenge zu reduzieren, könnte auf Datenformate zurückgegriffen werden, die ausschließlich die benötigten Informationen so genau wie möglich wiedergeben.

Um die benötigte Zeit einer Personenstromsimulation zu reduzieren, schlagen D. H. Biedermann und A. Borrmann (2016) einen Ansatz vor, der zunächst die Simulation in einem größeren Maßstab (makroskopisch) durchführt. Nur an kritischen Stellen, an denen das Personenaufkommen erhöht ist, werden weitere Simulationen mit kleineren Maßstäben (mesoskopisch und mikroskopisch) vollzogen. Dadurch wird die Simulation vereinfacht, wodurch sowohl der Zeitaufwand als auch die benötigte Rechenleistung reduziert werden. In der Literatur wird ein weiterer Ansatz verfolgt, um den genannten Nachteilen unter Zuhilfenahme von Künstlicher Intelligenz (KI) entgegenzuwirken (I. Mohammad Abir et al. 2022)

2.2 Künstliche Intelligenz

Künstliche Intelligenz (KI) kommt heutzutage in vielen verschiedenen Anwendungsbereichen zum Einsatz (J. Lammers 2023). Mithilfe von KI können komplexe Muster beispielsweise aus Texten oder Bildern erkannt oder automatisierte Entscheidungen getroffen werden. Vor allem durch den KI-basierten Chatbot ChatGPT von OpenAI (Introducing ChatGPT 2023) hat das Thema KI weltweit Bekanntheit erlangt (O. Bünte 2023). Die folgenden Kapitel behandeln die verschiedenen Typen von KI, sowie deren Funktion und Anwendungsbereiche. Darüber hinaus wird erläutert, wie KI in der Bauindustrie bereits angewendet wird.

2.2.1 Begriffserklärung und Funktion

Zunächst müssen die drei Begrifflichkeiten Künstliche Intelligenz (KI), Maschinelles Lernen (ML) und Deep Learning (DL) erklärt werden.

Künstliche Intelligenz ist ein Überbegriff (s. Abbildung 2-1) und bezieht ML und DL mit ein. Mit KI wird versucht, die menschliche Intelligenz nachzuahmen, um dem Menschen komplexe Arbeiten abnehmen zu können. Wenn Wissen in einem bestimmten Fachgebiet vorhanden ist, kann KI auf dieser Grundlage eine Lösung zu einem gefragten Problem präsentieren. Im Gegensatz zu Simulationen muss KI nicht alle Rechenschritte wiederholen. KI verwendet Informationen aus vorangegangenen Untersuchungen, um das aktuelle Problem zu lösen. (K. Aggarwal et al. 2022)

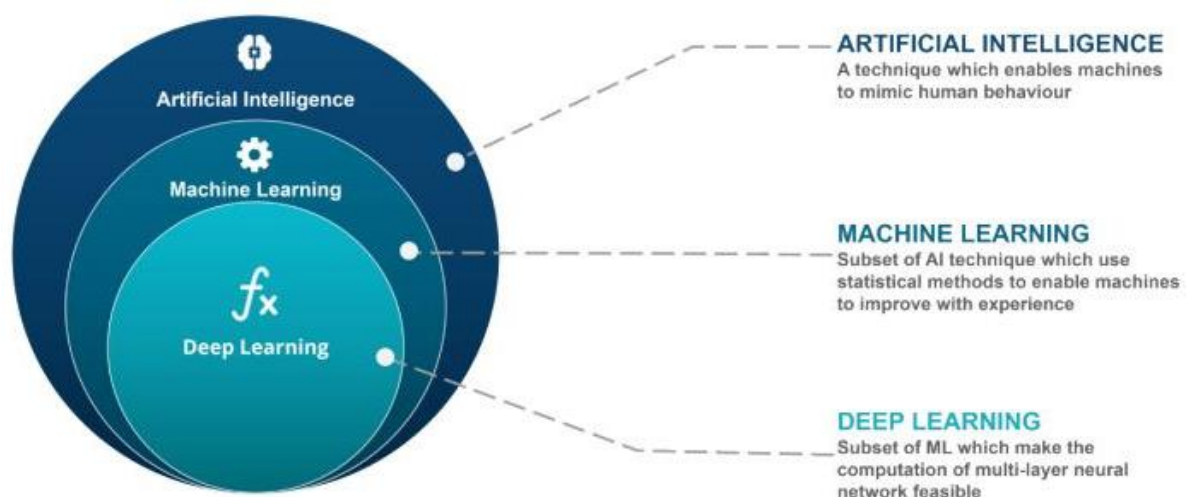


Abbildung 2-1: Unterschiede der Begriffe KI (hier: engl. Artificial Intelligence), ML (hier: engl. Machine Learning), DL (K. Aggarwal et al. 2022)

Maschinelles Lernen bezeichnet alle KIs, die selbstständig ihren Algorithmus anpassen und aus vorangegangenen Ergebnissen lernen können (s. Abbildung 2-1). Bei ML wird der Algorithmus vom Menschen geschrieben, somit können im Nachhinein noch Anpassungen bezüglich des Ergebnisses vorgenommen werden. Ein Anwendungsbereich ist beispielsweise die Bilderkennung oder die Datenanalyse. (K. Aggarwal et al. 2022)

Deep Learning stellt die am weitesten entwickelte Form der KI dar (s. Abbildung 2-1) und basieren auf neuronalen Netzwerken. Diese neuronalen Netzwerke bestehen aus auf verschiedenen Ebenen angeordneten Neuronen, die jeweils mit allen weiteren Neuronen der nächsten Ebene verbunden sind (s. Abbildung 2-2). Unter Neuronen sind Berechnungsknoten zu verstehen, die auf Grund von Eingabewerten der vorherigen Ebene eine Berechnung durchführen und das Ergebnis an die nächste Ebene weitergeben. Die jeweilige Architektur einer DL-Methode sowie die enthaltenen Ebenen werden im Vorhinein durch den Menschen festgelegt. Abbildung 2-2 ist zu entnehmen, dass neben den inneren Neuronen-Ebenen (versteckte Ebenen) auch eine Eingabe- und eine Ausgabe-Ebene vorhanden sind. Um DL-Methoden zu trainieren, findet die Informationsversorgung über diese Eingabe- und Ausgabe-Neuronen statt. Die Datenverarbeitung ist dabei nicht mehr vom Menschen nachvollziehbar, und eine Anpassung der einzelnen Neuronen ist, nach Beginn des Trainings, nicht mehr vorgesehen. (E. Kavlakoglu 2020)

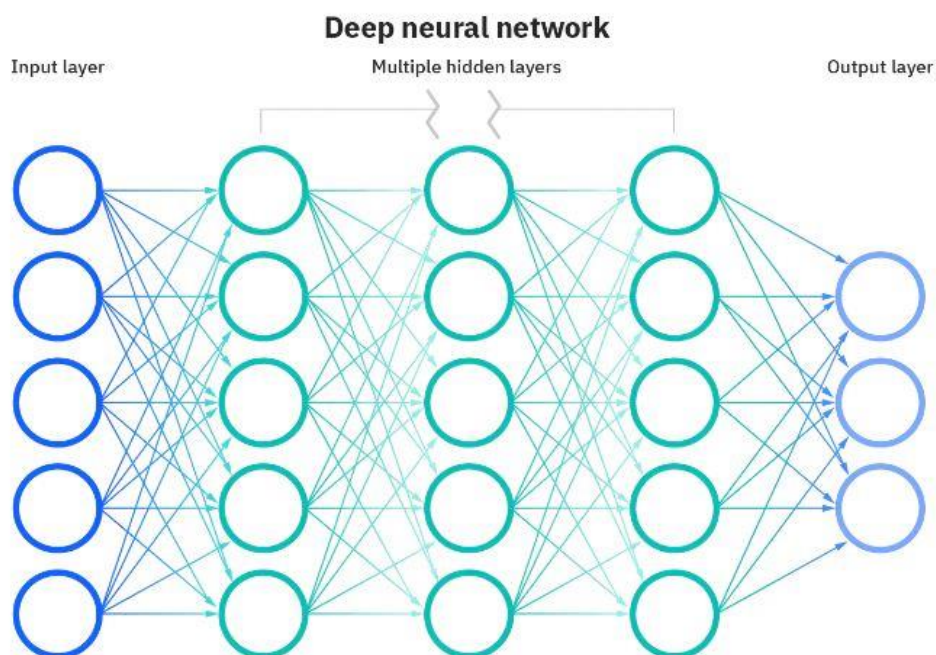


Abbildung 2-2: Neuronales Netzwerk eines DL-Modells (E. Kavlakoglu 2020)

2.2.2 Training von Deep Learning Methoden

Nachdem der Anwendungsfall und die zu lösende Problematik formuliert wurden, kann mit dem Training des neuronalen Netzwerks begonnen werden. Für das Training werden zunächst sowohl die Eingabe- als auch die Ausgabeinformationen benötigt, um der KI beizubringen, welche Ergebnisse richtig sind und in welchem Format sie zurückgegeben werden sollen. Dabei werden im ersten Schritt die auszuwertenden Informationen den Eingabe-Neuronen präsentiert. Daraufhin werden im zweiten Schritt diese vom neuronalen Netzwerk verarbeitet und geben ein auf Wahrscheinlichkeiten beruhendes Ergebnis an die Ausgabe-Neuronen zurück. In einem dritten Schritt wird das berechnete Ergebnis mit dem tatsächlich richtigen Ergebnis verglichen. Dabei wird für jede Ausgabe-Neurone überprüft, wie weit diese vom eigentlichen Wert entfernt ist. Diese Differenz wird daraufhin in einem rückwärts gerichteten Berechnungsprozess dafür genutzt, um die Neuronen der versteckten Ebenen anzupassen (vierter Schritt). Diese Schritte werden so oft wiederholt, bis die durchschnittliche Differenz der Ergebnisvergleiche unterhalb eines bestimmten, festgelegten Grenzwerts liegt. Erst nach einem solchen Training ist die KI in der Lage, verlässliche Ergebnisse zu liefern. (E. Kavlakoglu 2020; H. Larochelle et al. 2009)

Um das Training einer KI erfolgreich durchzuführen, bedarf es verlässliche Trainingsdaten, die den Eingabeneuronen übergeben werden. Diese können beispielsweise als Bild, Text oder Video vorliegen. Zudem muss für jeden Berechnungsschritt während des Trainings ein neues Berechnungsbeispiel vorhanden sein, inklusive der jeweiligen Lösung (alle Berechnungsbeispiele = Trainingsdatensatz). (E. Kavlakoglu 2020; H. Larochelle et al. 2009)

DL-Modelle mit mehreren Neuronen-Ebenen (Anzahl wird im Vorhinein vom Menschen festgelegt) ermöglichen, dass komplexere Trainingsdatensätze von KI verarbeitet werden können.

Training mithilfe von Graphen-basierten Repräsentationen

Es gibt bereits seit mehreren Jahren DL-Modelle, die auf Basis von Netzwerkdarstellungen Informationen verarbeiten. Diese Art neuronale Netzwerke wird *Graph Neural Network (GNN)* genannt. Ein großer Nachteil dieser GNNs ist, dass das zur Verfügung gestellte Netzwerk immer die gleiche Größe und die gleichen Informationen enthalten muss. Bei der Abstraktion eines Gebäudes in die Form eines Graphen führt das zu

einem Problemen, denn die Größe und Anzahl an Zimmern variiert in jedem Gebäude. (I. Mayachita 2020)

Um diesem Problem entgegenzuwirken, wurde eine Unterklasse des GNN entwickelt, das sogenannte *Graph Convolutional Network (GCN)*. In den letzten Jahren wurden Fortschritte bei der Forschung dieser GCNs erzielt, sodass mit deren Hilfe Netzwerke verarbeitet werden können, die nicht immer gleich groß sind. (I. Mayachita 2020)

GCNs ermöglichen die Verwendung unterschiedlich großer Netzwerkdarstellungen als Eingabeinformation beim Training von DL-Modellen.

2.2.3 Anwendung von Deep Learning in der Bauindustrie

Die aktuelle Forschung ist damit beschäftigt, DL-Methoden in der Bauindustrie zu implementieren. Dadurch soll neben der Zeit- und Kosteneffizienz auch die Sicherheit auf Baustellen gesteigert werden. (S. Kristombu Baduge et al. 2022)

Beispielsweise beschäftigen sich M. Yang et al. (2023) mit der Verwendung eines DL-Modells zur Identifizierung von Sicherheitsrisiken auf Baustellen. Dabei werden Überwachungsvideos von Baustellen von einer KI verarbeitet, die in Echtzeit erkennt, ob Arbeiter eine ausreichende Schutzausrüstung tragen. Fehlverhalten werden noch vor Auftreten eines Unfalls gemeldet und mögliche Sicherheitsvorkehrungen vorgeschlagen.

Neben Bilderkennung kann KI, aufgrund von vorherigen Informationen, Daten auswerten und diese in kausale Segmente klassifizieren. Dies ist in der Bauindustrie von Vorteil, da eine Vielzahl von Datenerhebungen wie Temperatur- und Luftfeuchtigkeitsmessungen oder die Messung von Untergrundveränderungen stattfinden. Durch die Verfügbarkeit dieser Informationen und die Klassifizierung durch KI können beispielsweise mechanische Eigenschaften von Baumaterialien verbessert, Material und Kosten gespart werden. (S. Kristombu Baduge et al. 2022)

Aufgrund der Fähigkeit, Muster zu erkennen, kann KI menschliches Verhalten in Menschenmassen verstehen und nachahmen (S. Kristombu Baduge et al. 2022). Diese Fähigkeit nutzen K. Minguk et al. (2023) in einer Studie zu Personenströmen. Sie verwenden ein DL-Modell, das erkennt, wie Personen in der Realität ihren Weg aussuchen. Denn bei der Untersuchung von Verhaltensweisen von Personen kann nicht immer damit gerechnet werden, dass der kürzeste Weg gewählt wird. In der Realität wird

bei der Wahl des Weges zum Beispiel auch auf Sicherheit oder mögliche Kosten geachtet. Mithilfe dieses DL-Modells konnten die realen Wege um 8 % besser simuliert werden als mit herkömmlichen Algorithmen.

Wie in [Kapitel 2.2.2](#) beschrieben, müssen DL-Modelle mithilfe von Datensätzen trainiert werden. Die Genauigkeit der trainierten KI ist stark abhängig von den Trainingsdaten, denn je besser und genauer diese sind, desto detailliertere Ergebnisse werden mit dem DL-Modell erzielt. Deshalb ist die richtige Wahl der Trainingsdaten in einem geeigneten Format ein entscheidender Schritt bei der Erstellung eines DL-Modells. (S. Kristombu Baduge et al. 2022)

2.3 Wahl einer geeigneten Darstellungsform für Gebäude

Grundsätzlich wird zwischen zwei verschiedenen Arten von Informationen unterschieden, die in einem Modell enthalten sein können, den geometrischen und topologischen (S. Pu und S. Zlatanova 2005).

Geometrische Informationen sind beispielsweise Elemente wie Punkte oder Linien, die in einem Koordinatensystem platziert sind und dadurch eine räumliche Beschreibung liefern. Zudem können neben konkreten kartesischen Punkten auch Richtungen in Form von Vektoren durch geometrische Informationen angegeben werden (IFC4.3.1.0 Documentation 2023). Durch geometrische Informationen können in Gebäudemodellen sowohl Abmessungen von Räumen als auch Positionen von Türen und Treppen beschrieben werden. Ein Beispiel für ein solches Modell ist ein Grundriss, siehe Abbildung 2-3. (S. Pu und S. Zlatanova 2005)

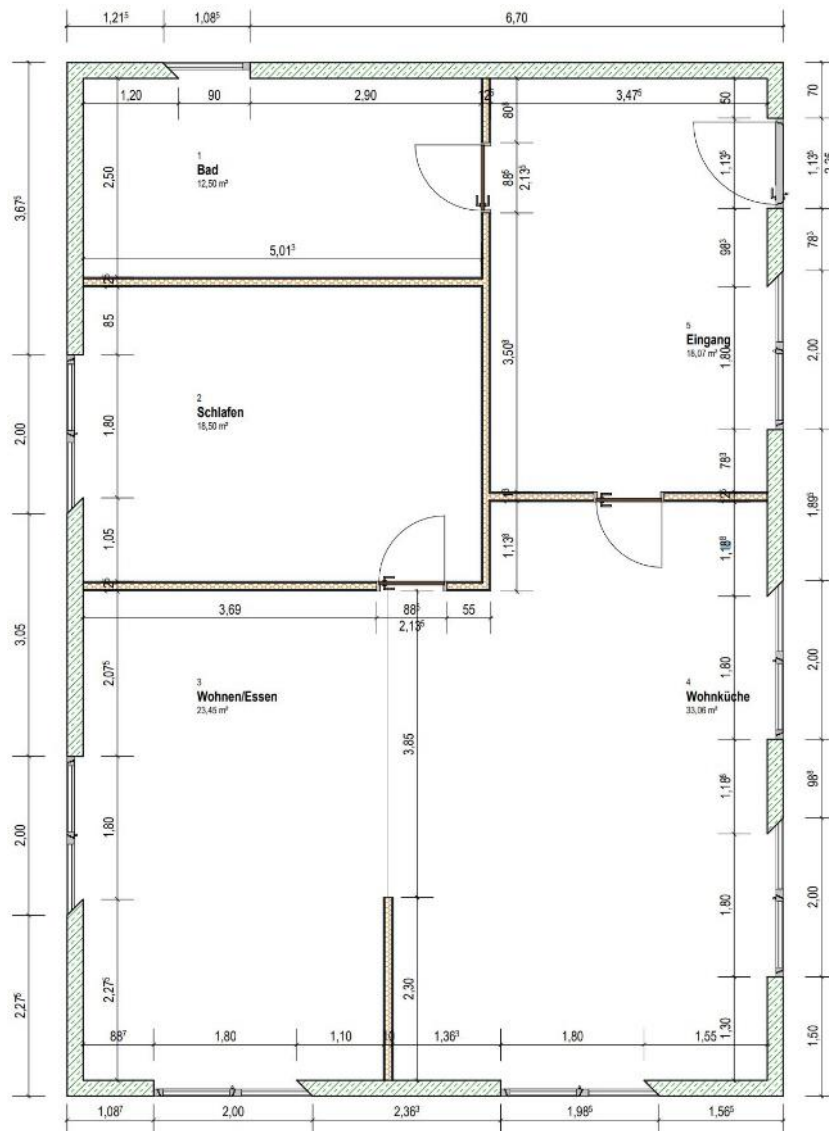


Abbildung 2-3: Grundriss einer Wohnung mit Bemaßung

Topologische Informationen beschreiben die Beziehung zweier Elemente zueinander (IFC4.3.1.0 Documentation 2023). Durch sie können nicht-geometrische Eigenschaften der jeweiligen Elemente beschrieben werden. Topologische Informationen können mithilfe einer Netzwerkdarstellung abgebildet werden. Dabei werden Elemente als Knoten und Beziehungen durch Kanten zwischen den einzelnen Knoten dargestellt (S. Pu und S. Zlatanova 2005). Beispielsweise kann der Grundriss aus Abbildung 2-3 auch als Modell mit reinen topologischen Informationen visualisiert werden (s. Abbildung 2-4). Dabei kann sowohl die Wahl der jeweiligen Knoten als auch Kanten variieren. Je nach Abstraktion kann ein topologisches Modell verschiedene Beziehungen darstellen. Die Netzwerkdarstellung muss somit entsprechend den topologischen Informationen entworfen werden, die abgebildet werden sollen.

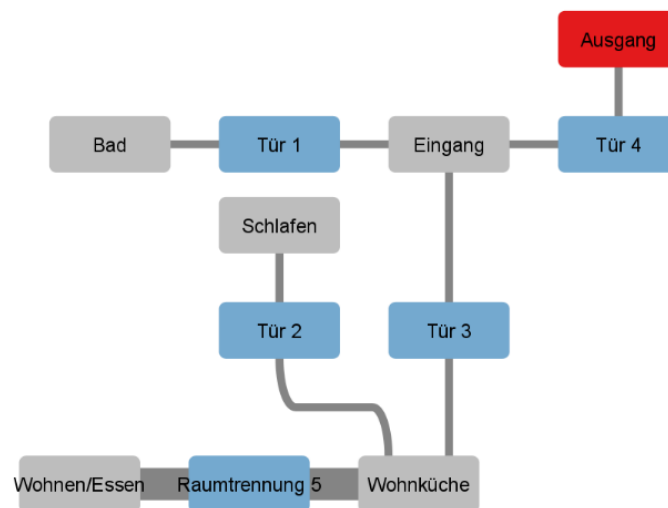


Abbildung 2-4: Topologische Netzwerkdarstellung derselbe Wohnung wie in Abbildung 2-3

Es ist entscheidend, das passende Modell entsprechend dem Anwendungsfall auszuwählen, da geometrische und topologische Informationen unterschiedlich dargestellt werden können. Je nach Anwendung sind nicht immer alle Informationen für ein Modell von Bedeutung. Es gibt Ansätze, bei denen große Datenmodelle zunächst mithilfe von topologischen Informationen in Form einer Netzwerkdarstellung untersucht werden. Falls eine genauere Betrachtung in einzelnen Modellbereichen erforderlich ist, wird zur Konkretisierung der Daten auf geometrische Informationen dieser Bereiche zurückgegriffen. (D. H. Biedermann und A. Borrmann 2016)

In dieser Bachelorarbeit wird eine Abbildung von Gebäuden entwickelt, die sowohl geometrische als auch topologische Informationen zur genauen Darstellung benötigt (D. H. Biedermann und A. Borrmann 2016). Für die Repräsentation eignet sich deshalb eine Graphen-basierte Darstellung, die sowohl geometrische als auch topologische Informationen enthalten kann.

2.4 Graphen-basierte Repräsentation

Das folgende Kapitel geht auf den Aufbau einer Graphen-basierten Repräsentation ein, um ein grundlegendes Verständnis für dieses Thema zu vermitteln.

2.4.1 Aufbau

Ein Modell, dass mit topologischen Beziehungen abgebildet werden kann, ist die Netzwerkdarstellung. Wie im vorherigen Kapitel kurz beschrieben, besteht der Grundge-

danke darin, Zusammenhänge zwischen Objekten mithilfe von Knoten und Kanten darzustellen. Eine Graphen-basierte Repräsentation beinhaltet somit eine Menge an Knoten und Kanten, die in Beziehungen zueinanderstehen. (G. Chartrand et al. 2011)

Knoten

In einem Graphen-basierten Netzwerk werden alle abzubildenden Objekte durch Knoten repräsentiert, die durch Kanten miteinander verbunden sind. Jeder Knoten verfügt zur eindeutigen Identifizierung über einen einzigartigen Namen (s. Abbildung 2-5). Neben dem Namen enthalten die meisten Knoten Informationen über den jeweiligen Knotentyp sowie über zusätzliche Attribute (Labels), die sie näher beschreiben. (S. M. Ciobâ und M. Ram Murty 2022)

Labels werden in einer Netzwerkdarstellung dazu verwendet, einzelnen Knoten und Kanten Eigenschaften zuzuweisen, die die jeweilige Funktionalität unterstützen. Anhand dieser Labels kann das Netzwerk durchsucht, sortiert und klassifiziert werden (Dokumentation zu NetworkX 3.1 2023).

Kanten

Kanten stellen immer eine Verbindung zwischen zwei Knoten dar. Mit dieser Verbindung werden die topologischen Beziehungen zwischen den jeweiligen Knoten beschrieben (s. Abbildung 2-5). Auch Kanten erhalten einen einzigartigen Namen und den Typ der Verbindung als Attribut. Durch weitere Labels werden die topologischen Beziehungen genauer definiert. (S. M. Ciobâ und M. Ram Murty 2022)

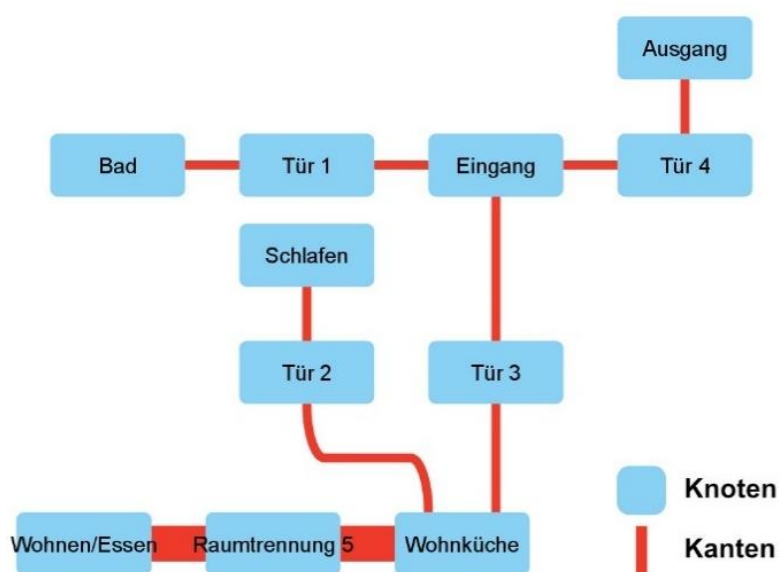


Abbildung 2-5: Darstellung eines Graphen-basierten Netzwerkes (Knoten & Kanten)

Neben der einfachen Verbindung von zwei Knoten kann eine Kante auch eine Verbindung zwischen einem Knoten mit sich selbst sein (Schleife). Zudem wird zwischen gerichteten und ungerichteten Graphen unterschieden. Gerichtete Graphen besitzen ausschließlich richtungsweisende Kanten. Falls eine Beziehung in beide Richtungen dargestellt werden soll, müssen dem Graphen-Netzwerk zwei Kanten hinzugefügt werden. Ein ungerichteter Graph besteht aus Kanten, die keine Richtungsangabe aufweisen. Somit können in ungerichteten Graphen keine gerichteten Beziehungen dargestellt werden. (S. M. Ciobâ und M. Ram Murty 2022)

Um geometrische und topologische Informationen in einem Graphen abspeichern und gezielt die jeweiligen Informationen abfragen zu können, gibt es hierarchisch aufgebaute Netzwerkdarstellungen.

2.4.2 Hierarchische Graphen-basierte Repräsentation

In der Literatur werden bereits seit Jahren hierarchisch aufgebaute Netzwerke beschrieben, die geometrische und topologische Informationen in verschiedenen hierarchischen Ebenen abspeichern (E.-P. Stoffel et al. op. 2008; L. Yang und M. Worboys 2015). Sie eignen sich für Netzwerkdarstellungen, in denen einzelne Knoten hierarchisch über anderen Knoten stehen.

Aufbau

Der Aufbau einer solchen Netzwerkdarstellung ist, wie in Abbildung 2-6 zu sehen, in verschiedene Ebenen unterteilt. Jede hierarchische Ebene enthält unterschiedlich genaue Informationen, wobei die hierarchisch untergeordnete Ebene die übergeordnete näher beschreibt. In Abbildung 2-6 ist auf höchster Ebene (Layer 4) die Informationsdichte geringer als auf der untersten Ebene (Layer 0). In dieser Darstellung werden Ebenen auf der Makroebene durch eine genauere mesoskopische Beschreibung ergänzt, welche wiederum durch eine mikroskopische Verfeinerung weiter detailliert wird. (A. Kontarinis et al. 2021)

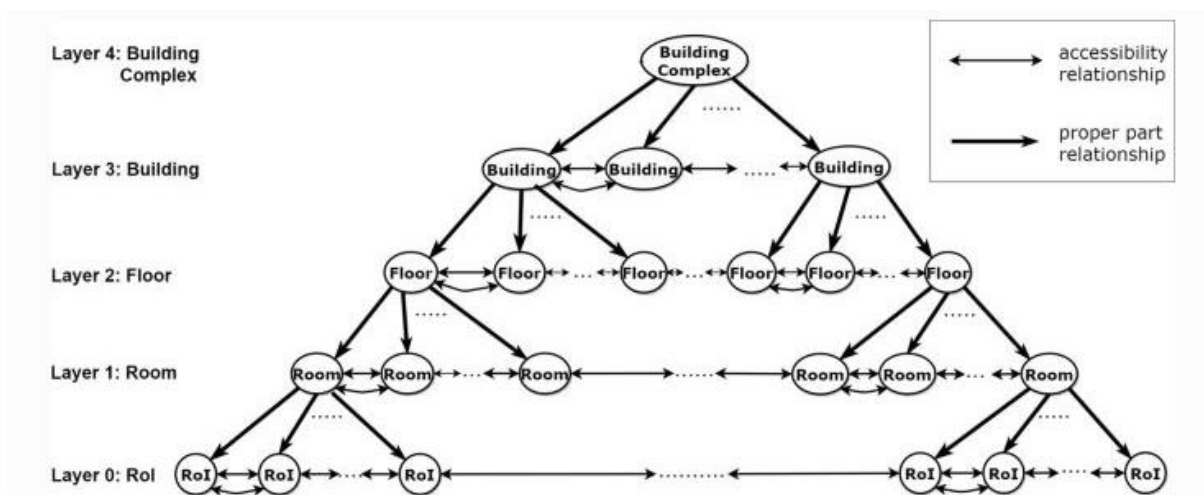


Abbildung 2-6: Hierarchische Innenraumdarstellung mit mehreren Ebenen
(A. Kontarinis et al. 2021)

Technische Darstellung

Der Unterschied zu einer herkömmlichen Netzwerkdarstellung besteht darin, dass die hierarchische Abhängigkeit unter Zuhilfenahme von weiteren Labels der jeweiligen Knoten und Kanten näher beschrieben wird. Ein hierarchisch aufgebautes Netzwerk bleibt somit ein Graphen-basiertes Netzwerk. Dennoch wird im weiteren Verlauf der Arbeit von einzelnen Ebenen in Verbindung mit hierarchischen Graphen-basierten Repräsentationen gesprochen. Mithilfe von Labels lassen sich zwei verschiedene Arten von Kanten beschreiben. Neben Kanten, die topologische Beziehungen innerhalb einer Ebene angeben, gibt es weitere, die für einen hierarchischen Aufbau (hierarchische Kanten) sorgen. Bei einer Entfernung oder Nichtbeachtung dieser hierarchischen Kanten kann jede Ebene als ein eigenes Netzwerk betrachtet werden. Außerdem werden den Knoten durch zusätzliche Labels den einzelnen hierarchischen Ebenen des Netzwerks zugewiesen. (A. Kontarinis et al. 2021)

Mithilfe von hierarchischen Netzwerkdarstellungen können Daten schnell und sortiert verarbeitet werden (B. Perret et al. 2019). Zudem besteht die Möglichkeit je nach Bedarf, die Netzwerkdarstellung in der passenden Ebene zu untersuchen und Daten nur dort abzufragen. Dadurch ist es möglich, bestimmte Informationen auf Mikro-Ebene abzufragen, ohne die komplette Netzwerkdarstellung durchsuchen zu müssen. Zur groben Betrachtung wird das Netzwerk auf makroskopischer Ebene untersucht.

Für die Erstellung der Graphen-basierten Repräsentation von Gebäuden nimmt diese Arbeit als Informationsquelle Modelle, die mit Building Information Modeling erstellt wurden.

2.5 Building Information Modeling

Building Information Modeling (BIM) ermöglicht die Anfertigung einer effizienten, genauen und einfachen Planung eines Gebäudes. Ein BIM-Modell kann alle Informationen über die Planung, die Bauausführung, die Nutzung bis hin zum Rückbau eines Gebäudes enthalten. Dadurch lassen sich alle zeitlichen Abläufe und Fertigungsschritte möglichst realitätsnah planen. Neben den jeweiligen Herstellern der Baustoffe können beispielsweise auch Verbau- und Rückbaumaßnahmen im BIM-Modell mit abgebildet werden. (A. Borrmann und M. König 2021)

Unter anderem ermöglicht BIM, ein zu planendes Gebäude mit allen Planungspartnern und ausführenden Firmen zu teilen. Dadurch wird das gemeinsame Arbeiten an einem aktuell gehaltenen Modell ermöglicht, auf das alle beteiligten Parteien schnell und unkompliziert zugreifen können. Dies führt sowohl zu einer Effizienzsteigerung, da Baupläne nicht händisch aufeinander abgestimmt werden müssen, als auch zu einer Verringerung der Fehleranfälligkeit auf Baustellen, da die Pläne vollständig und aktuell sind. (A. Borrmann und M. König 2021)

2.6 Herleitung der Forschungsfrage

In den vorangegangenen Kapiteln wurden die Grundlagen zu Personenstromsimulation, KI sowie Graphen-basierten Repräsentationen erläutert. Dabei wurden die wichtigsten Theorien, Modelle und Ansätze aufgeführt, die in der Literatur bereits bekannt sind.

Die Recherche des aktuellen Forschungsstands zeigt, dass der Bedarf an Trainingsdaten für DL-Methoden vorhanden ist und der Einsatz in der Baubranche in Zukunft steigen wird (I. Mohammad Abir et al. 2022). Methoden zur Erstellung Graphen-basierter Repräsentationen von BIM-Modellen, die als Trainingsdaten für DL-Methoden eingesetzt werden können, sind noch nicht ausreichend erforscht.

Basierend auf diesen Erkenntnissen lässt sich folgende Forschungsfrage formulieren:

Wie kann die Graphen-basierte Repräsentation eines BIM-Modells zur Unterstützung von Trainingsdaten für Deep Learning Methoden entwickelt und erstellt werden?

Die Beantwortung dieser Forschungsfrage erfordert die Untersuchung der benötigten Informationen sowie die Herleitung und Erstellung der Graphen-basierten Repräsentation des BIM-Modells. Das Vorgehen und der Aufbau dieser Untersuchung werden in den folgenden Kapiteln präsentiert.

3 Methodik und Vorgehen

In dieser Arbeit wird eine Netzwerkdarstellung entwickelt, die den Bewegungsraum von Personen während einer Evakuierung aus einem Gebäude darstellt. Dadurch beschränken sich die benötigten Informationen auf die Bereiche, in denen sich Personen fortbewegen können.

3.1 Informationsgewinnung

In den folgenden Kapiteln werden die Modellinformationen vorgestellt, die benötigt werden, um ein Gebäude realitätsnah darzustellen und wie diese aus einem BIM-Modell ausgelesen werden können.

3.1.1 Benötigte Informationen

Für den beschriebenen Anwendungsfall wird angenommen, dass sich Personen in Gebäuden zwangsweise in Zimmern aufhalten und sich in diesen fortbewegen. Bereiche, die sich außerhalb des Gebäudes befinden, können ebenfalls als Bewegungsraum von Bedeutung sein. Im Folgenden werden die Informationen präsentiert, die für die Netzwerkdarstellung dieser Arbeit benötigt werden.

Neben dem Flächeninhalt und der Form dieser Bewegungsräume muss zur genauen geometrischen Bestimmung das Stockwerk mit ins Netzwerk-Modell aufgenommen werden.

Ein Raumwechsel findet immer über ein horizontales Verbindungselement, wie beispielsweise eine Tür statt. Von diesen werden Informationen über die Breite und die genaue Position benötigt.

Neben den horizontalen gibt es vertikale Verbindungselemente, beispielsweise Treppen. Aufzüge werden in diesem Modell vernachlässigt, da während einer Evakuierung diese nicht immer zur Verfügung stehen. Von diesen vertikalen Verbindungselementen werden neben der Breite und der genauen Startposition auch die Steigung und die Form benötigt. Mit Startposition ist die Position gemeint, an der die Treppe im unteren Stockwerk anfängt.

Diese Informationen müssen alle gezielt aus dem jeweiligen BIM-Modell abgefragt und ausgelesen werden.

3.1.2 Informationen aus BIM-Modellen auslesen

Wie bereits erläutert, beschreibt ein BIM-Modell Gebäude in Form eines digitalen Modells. Solche Modelle können mit vielen Programmen erstellt und bearbeitet werden. Allerdings verwendet jedes dieser Programme zur Abspeicherung eines BIM-Modells ein hauseigenes Datenformat. Um mit anderen Programmen, zum Beispiel Bemessungs- oder Simulationsprogrammen, interagieren und Modellinformationen austauschen zu können, muss bei der Speicherung des BIM-Modells auf ein Format zurückgegriffen werden, welches sich als Standard etabliert hat. Nachteil dieser standardisierten Formate ist, dass die Informationen stark vereinfacht werden, damit jedes Modell mit diesen arbeiten kann. So gehen wichtige topologische Informationen des BIM-Modells bei der Übertragung verloren. (H. Lai und X. Deng 2018)

Um diesem Datenverlust entgegenzuwirken, werden in dieser Arbeit sämtliche Informationen direkt aus BIM-Modellen gewonnen. Ein BIM-Modell teilt alle abgebildeten Objekte in unterschiedliche Elemente ein. Die zugehörigen geometrischen sowie topologischen Informationen werden diesen Elementen als Attribute hinzugefügt (A. Borrmann et al. 2015). Somit müssen im ersten Schritt alle erforderlichen Elemente bestimmt werden, aus denen die benötigten Informationen aus [Kapitel 3.1.1](#) herausgelesen werden können.

Da angenommen wurde, dass der Bewegungsraum der Personen in einem Gebäude auf die Räume beschränkt ist, bietet es sich an, das Element *Raum* aus BIM-Modellen als Datenquelle zu nutzen. Das Raumelement beschreibt eine Fläche, die mithilfe von Wänden und Raumtrennungslinien eingeschränkt wird. Unter anderem sind neben dem Flächeninhalt auch Informationen über das Stockwerk und die Form enthalten. Der Vorteil der Raumelemente ist, dass sie bestehenden BIM-Modellen problemlos hinzugefügt und dadurch die Vorgaben der entwickelten Methode leicht erfüllt werden können.

Die horizontalen Verbindungselemente zwischen den einzelnen Räumen werden durch unterschiedliche Elemente im BIM-Modell dargestellt. Neben dem offensichtlichen Element *Tür* gibt es zwei weitere Möglichkeiten für Personen den Raum zu wechseln. Zum einen kann ein Durchbruch in einer Wand (ohne Tür) die Verbindung zwischen zwei Räumen darstellen und zum anderen können Räume ohne Wand voneinander getrennt sein, wie beispielsweise eine Wohnküche (Wohnzimmer und Küche).

In beiden Fällen werden diese Übergänge mithilfe von Raumtrennungslinien abgebildet. Durch eine Raumtrennungslinie kann der Bereich eines Raumelementes eingegrenzt werden. Zwei Raumelemente, die unmittelbar aneinander liegen, können so voneinander getrennt werden. Ein Raumwechsel kann in diesem Fall über die komplette Länge der Raumtrennungslinie erfolgen, deshalb muss eine solche Raumtrennung ebenfalls mit beachtet werden (als horizontales Verbindungselement).

Als vertikales Verbindungselement muss, wie in [Kapitel 3.1.1](#) erläutert, nur die Treppe beachtet werden. In BIM-Modellen wird diese mithilfe eines Treppenelements dargestellt.

Wenn alle Informationen bekannt sind, die für die Abstraktion des BIM-Modells benötigt werden, kann die Netzwerkdarstellung erstellt werden.

3.2 Gestaltung hierarchische Graphen-basierte Repräsentation

In Netzwerkdarstellungen können sämtliche geometrische Informationen in den Knoten und alle topologischen Informationen in den Kanten abgespeichert werden. Dies ermöglicht eine Darstellung aller Informationen der Elemente des BIM-Modells.

Zunächst wird der Aufbau des Graphen mit den jeweiligen hierarchischen Ebenen beschrieben, bevor auf die jeweiligen, in den Knoten und Kanten enthaltenen Informationen näher eingegangen wird.

3.2.1 Aufbau der hierarchischen Ebenen

Als ersten Schritt müssen diejenigen Elemente bestimmt werden, die sich als Knoten für das Netzwerk eignen und in hierarchischer Beziehung zueinanderstehen (hierarchische Knoten). In diesem Fall bieten sich die folgenden Elemente an: Stockwerke (makroskopisch), Räume (mesoskopisch) und Punkte (mikroskopisch). Jedes dieser Elemente beschreibt das übergeordnete Element genauer. Als Verbindungselemente zwischen den hierarchischen Knoten innerhalb der jeweiligen hierarchischen Ebene, werden horizontale Verbindungselemente und Treppen verwendet. Diese werden auch in Form eines Knotens dargestellt (Verbindungsknoten). Die Kanten des Netzwerks verbinden in den einzelnen hierarchischen Ebenen zwei hierarchische Knoten über einen Verbindungsknoten miteinander.

Die hierarchische Zuordnung findet nur über die hierarchischen Knoten statt. Wie in Abbildung 3-1 zu sehen ist, wird die hierarchische Darstellung nur über die Stockwerks-, Raum- und Punkt-Knoten beschrieben.

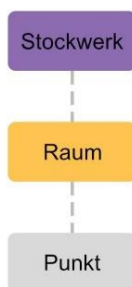


Abbildung 3-1: Die drei hierarchischen Ebenen, mit den jeweiligen Knoten

Die einzelnen hierarchischen Ebenen und der genaue Aufbau des Netzwerks werden in den folgenden Unterkapiteln erläutert.

Gebäude-Ebene (Makroskopisch)

Auf der höchsten hierarchischen Ebene befindet sich die Gebäude-Ebene. Auf dieser Ebene sind die Informationen der einzelnen Stockwerke (hierarchische Knoten) und der Treppen (Verbindungsknoten) zwischen den Stockwerken dargestellt.

Die Kanten in den jeweiligen Gebäude-Ebenen verbinden die einzelnen Stockwerks-Knoten über die jeweiligen Treppe-Knoten miteinander (s. Abbildung 3-2).



Abbildung 3-2: Aufbau der Gebäude-Ebene – Darstellung eines Gebäudes

Zudem führen weitere Kanten von den Stockwerks-Knoten in die hierarchisch niedrigere Ebene zu den Raum-Knoten des entsprechenden Stockwerks.

Stockwerk-Ebene (Mesoskopisch)

Auf der zweiten hierarchischen Ebene werden die einzelnen Stockwerke auf Stockwerk-Ebene näher beschrieben. Die Knoten symbolisieren in diesen Ebenen die einzelnen Räume (hierarchische Knoten) des Stockwerks. Verbunden werden diese über horizontale und vertikale Verbindungsknoten mithilfe von Kanten. Die vertikalen Verbindungen (Treppen) beschreiben die Erreichbarkeit zweier Räume in unterschiedlichen Stockwerken (s. Abbildung 3-3), somit können die Treppen-Knoten sowohl aus der Gebäude- als auch aus der Stockwerksebene erreicht werden.

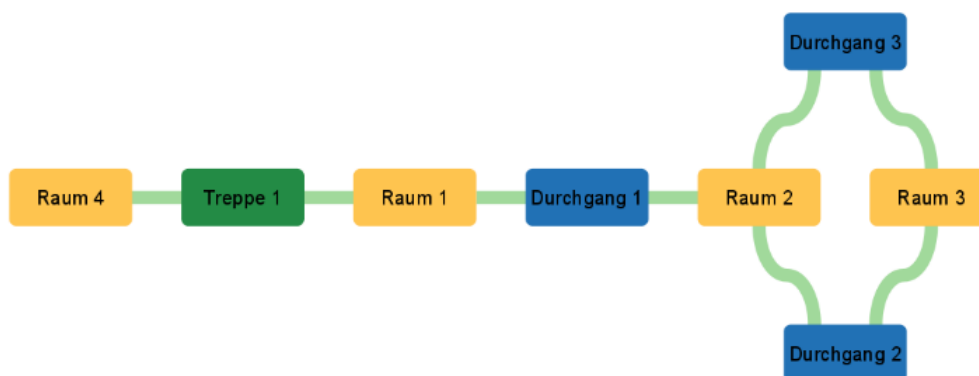


Abbildung 3-3: Aufbau der Stockwerk-Ebene – Darstellung mehrerer Räume

Die einzelnen Raum-Knoten sind außerdem über Kanten mit Punkt-Knoten verbunden, die sich auf der untergeordneten Raum-Ebene befinden.

Raum-Ebene (Mikroskopisch)

Die Raum-Ebene ist die unterste hierarchische Ebene und enthält somit die exaktesten Informationen zur Lage. In dieser Ebene wird die Form der einzelnen Räume mithilfe von Punkten (hierarchische Knoten) genauer beschrieben. Kanten verbinden räumlich verbundene Punkt-Knoten ohne Verbindungsknoten miteinander (s. Abbildung 3-4). Falls eine Begrenzung eines Raumes nicht durch eine Gerade beschrieben werden kann, wird diese mithilfe von 20 Punkt-Knoten approximiert.

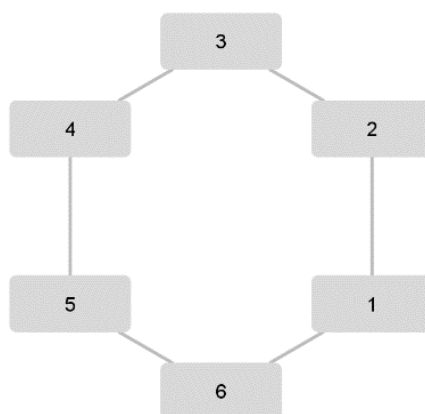


Abbildung 3-4: Aufbau der Raum-Ebene – räumliche Darstellung eines Raumes

Neben der genauen geometrischen Beschreibung der Räume wird in der Raum-Ebene auch der genaue Verlauf und die Position der einzelnen Treppen durch verbundene Punkt-Knoten beschrieben (zweidimensional im jeweiligen Stockwerk).

Nachdem der Aufbau der entworfenen hierarchischen Graphen-basierten Repräsentation beschrieben wurde, geht das nächste Kapitel auf sämtliche Informationen ein, die in den jeweiligen Knoten und Kanten hinterlegt sind.

3.2.2 Enthaltene Informationen

In [Kapitel 3.1.1](#) wurden diejenigen Informationen erarbeitet, die für die Darstellung des Gebäudes benötigt werden. Neben den Informationen, die individuell variieren, besitzt jeder Knoten und jede Kante jeweils einen Namen (*name*), einen Typen (*type*) sowie eine eindeutige ID (*id*). Die ID ist eine generierte fortlaufende Zahl. In Abbildung 3-5 ist ein UML-Diagramm (*Unified Modeling Language*) zu sehen, welches den Aufbau der hierarchischen Netzwerkdarstellung wiedergibt. Im Folgenden werden die jeweiligen Labels jedes Knoten erklärt. Die jeweils verwendete Variable steht kursiv in Klammern und ist im UML-Diagramm an der jeweiligen Stelle zu finden.

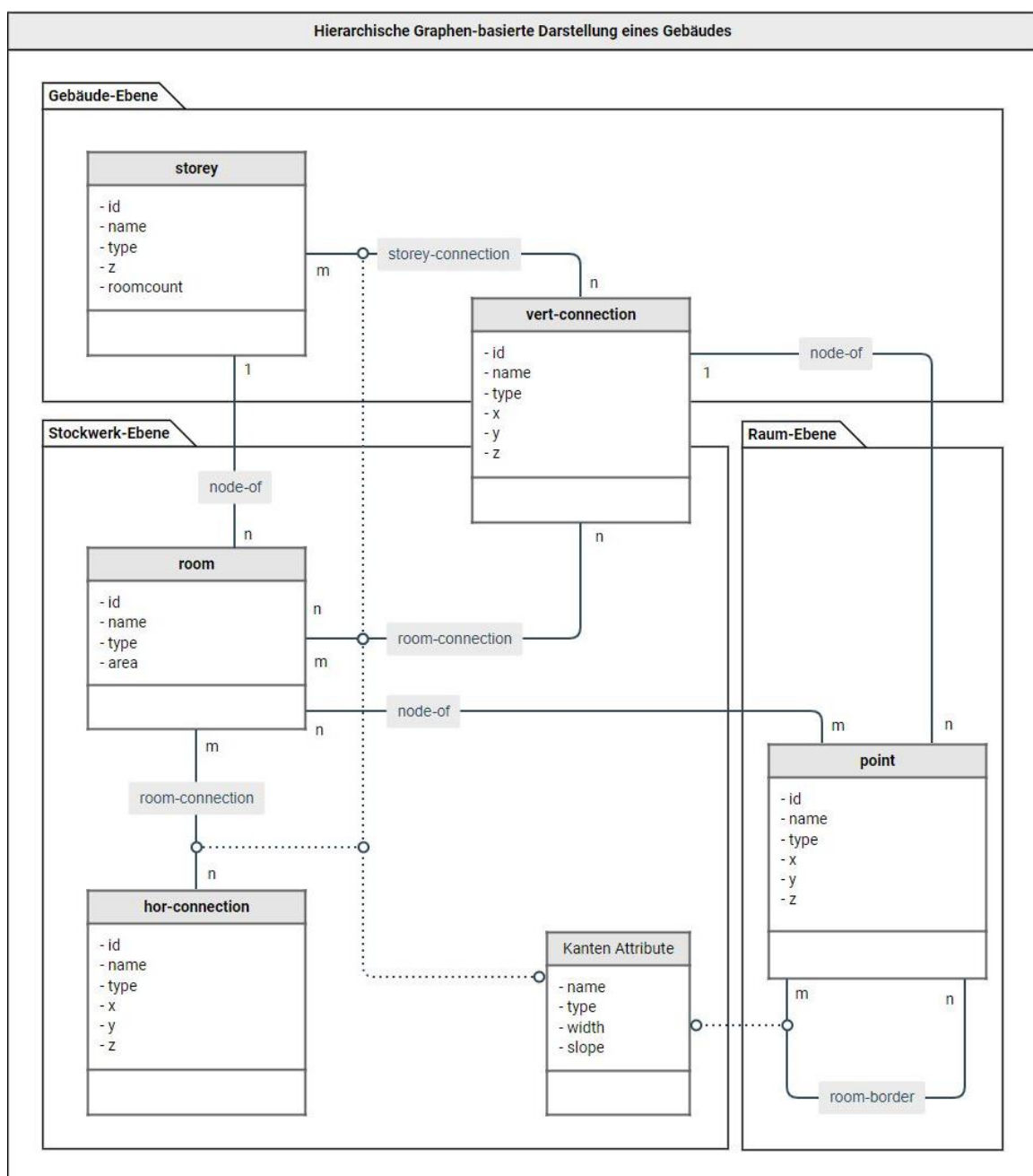


Abbildung 3-5: UML-Diagramm - Hierarchische Graphen-basierte Darstellung eines Gebäudes

Stockwerke

Die Stockwerks-Knoten (*type* = storey) werden nach dem Stockwerksnamen (*name*) benannt, der im BIM-Modell festgelegt wird. Damit die genaue geometrische Lage der Knoten bestimmbar ist, wird die Höhe über Null im BIM-Modell als Label (*z*) mit angegeben. Außerdem wird die Anzahl der sich im Stockwerk befindenden Räume mit abgespeichert (*roomcount*) (s. Abbildung 3-5).

Räume

Für die Stockwerk-Knoten entspricht der Name der Raum-Knoten (*type* = room) den im BIM-Modell festgelegten Raumnamen (*name*). Außerdem ist der Flächeninhalt jedes Raumes mit abgespeichert (*area*) (s. Abbildung 3-5).

Punkte

Punkt-Knoten (*type* = point) beschreiben eine feste Position im Raum mithilfe von dreidimensionalen Koordinaten (*x*, *y*, *z*). Neben einem durchnummerierten Namen (*name*) sind keine weiteren Attribute vorhanden (s. Abbildung 3-5).

Vertikale Verbindungsknoten (Treppen)

Die vertikalen Verbindungsknoten (*type* = vert-connection) besitzen neben dem Namen (*name*) auch die genaue Anfangsposition der Treppe (*x*, *y*, *z*) (s. Abbildung 3-5).

Horizontale Verbindungsknoten

Dem UML-Diagramm (Abbildung 3-5) ist zu entnehmen, dass horizontale Verbindungsknoten (*type* = hor-connection) neben der genauen Position der Verbindung (*x*, *y*, *z*) nur noch den Namen (*name*) als Attribut aufweisen. Mithilfe des Namens kann bei horizontalen Verbindungsknoten zwischen den beiden unterschiedlichen Verbindungsarten auf Stockwerks-Ebene, die in [Kapitel 3.1.2](#) dargestellt wurden, unterschieden werden.

Kanten

Die Kanten des kompletten Graphen, die eine Verbindung auf einer Ebene darstellen, enthalten die gleichen Attribute. Es werden neben der Art der Kante (*type*) auch die Breite (*width*) und Steigung (*slope*) mit angegeben. Im UML-Diagramm sind diese mithilfe einer Assoziationsklasse dargestellt (s. Abbildung 3-5).

Die Steigungen der Kanten, die horizontale Verbindungsknoten mit Räumen und die die Eckpunkte untereinander verbinden, sind Null, da in diesen Fällen keine Steigung vorhanden ist.

Kanten, die eine hierarchische Beziehung darstellen, besitzen neben der Art der Verbindung (*type = node-of*) keine weiteren Labels.

4 Implementierung und technische Umsetzung

Bei der Implementierung und technischen Umsetzung der Methode, aus BIM-Modellen Graphen-basierte Repräsentationen zu erstellen, wurde auf mehrere Programme, Packages und Dateiformate zurückgegriffen.

Die folgenden Kapitel gehen auf deren Funktion und Einsatz genauer ein, bevor die technische Umsetzung der Methode zur Erstellung der Netzwerkdarstellung erläutert wird.

4.1 Verwendete Programme und Dateiformate

Wie im dritten Kapitel beschrieben, gibt es verschiedene Programme, mit denen BIM-Modelle erstellt werden können. In dieser Arbeit wird für die Erstellung sämtlicher BIM-Modelle das BIM-Tool Revit von Autodesk verwendet (Revit-Software | BIM-Software | Autodesk 2023).

4.1.1 Autodesk Revit

Autodesk Revit ist ein weitverbreitetes und oft genutztes BIM-Authoring-Tool in der Baubranche und ermöglicht die Erstellung eines BIM-Modells. Mit vordefinierten Elementen lassen sich nicht nur architektonische und bauliche Pläne erstellen, sondern beispielsweise auch Sanitäranlagen, Stromversorgung und Innenausbau planen. (Revit-Software | BIM-Software | Autodesk 2023)

Autodesk veröffentlicht jährlich eine neue Revit-Version, die neue Funktionen und Änderungen einschließlich der *Application Programming Interface* (API) (auf Deutsch: Programmschnittstelle) einführt. Dies führt jedoch dazu, dass erstellte BIM-Modelle inkompatibel mit früheren Versionen sind und alle beteiligten Personen dieselbe Revit-Version benötigen. In dieser Arbeit wird mit Autodesk Revit und der Revit API in der Version 2022 gearbeitet. (Revit-Software | BIM-Software | Autodesk 2023; G. Talarico 2023)

4.1.2 Revit API

APIs sind Schnittstellen, über die Informationen an Dritte, beispielsweise andere Programme oder Entwickler, weitergegeben werden können. Dabei wird zunächst von der

dritten Partei eine Anfrage über die API gestellt. Daraufhin wird geprüft, ob alle Voraussetzungen für den Erhalt von Informationen erfüllt sind. Falls dem so ist, überträgt die API die angefragten Daten an die Partei. (Was ist ein Application Programming Interface (API)? | IBM 2023)

Im Fall von Revit kann über die API (Revit API) nach Informationen aus BIM-Modellen gefragt werden. Um die Voraussetzungen für eine fehlerfreie Übertragung zu erfüllen, gibt es ausführliche Dokumentationen der Revit API (G. Talarico 2023).

Um mit einer API kommunizieren und Daten austauschen zu können, muss die Anfrage über eine Programmiersprache gestellt werden. Als Programmiersprache, die sowohl einfach zu lesen als auch einfach zu programmieren ist, bietet sich Python an.

4.1.3 Python

Bei Python handelt es sich um eine weltweit verbreitete, objektorientierte Programmiersprache, mit der schnelle und übersichtliche Programme geschrieben werden können (Python.org 2023). Deshalb wurde bei der Umsetzung der Methode dieser Arbeit Python als Programmiersprache gewählt.

Autodesk Revit ermöglicht standardmäßig die Kommunikation mit der API über mehrere Programmiersprachen, allerdings nicht mit Python. Deshalb wurden von D. Thomas et al. (2023) ein Plugin namens RevitPythonShell entwickelt, durch das eine Python-Umgebung in Revit geöffnet werden kann. Eine Python-Umgebung wird benötigt, damit Python-Programme ausgeführt werden können.

4.1.4 RevitPythonShell

RevitPythonShell ermöglicht es Informationen aus Revit mithilfe von Python über die API abzufragen. Die Dokumentation dieses Plugins sowie sämtliche Informationen und Updates sind auf der GitHub-Seite des Projekts zu finden (D. Thomas et al. 2023). Damit Plugins in Revit verfügbar sind, müssen sie installiert werden.

Für die Installation des Plugins muss der benötigte *Installer* heruntergeladen werden (Release 2.0.0 · architecture-building-systems/revitpythonshell · GitHub 2023; D. Thomas et al. 2023). Nachdem die Datei geöffnet und installiert wurde, ist nach einem Neustart von Revit das Plugin unter *Zusatzmodule* -> *RevitPythonShell* zu finden. Unter RevitPythonShell gibt es, wie in Abbildung 4-1 zu sehen, vier Auswahlmöglichkeiten. (D. Thomas et al. 2023)

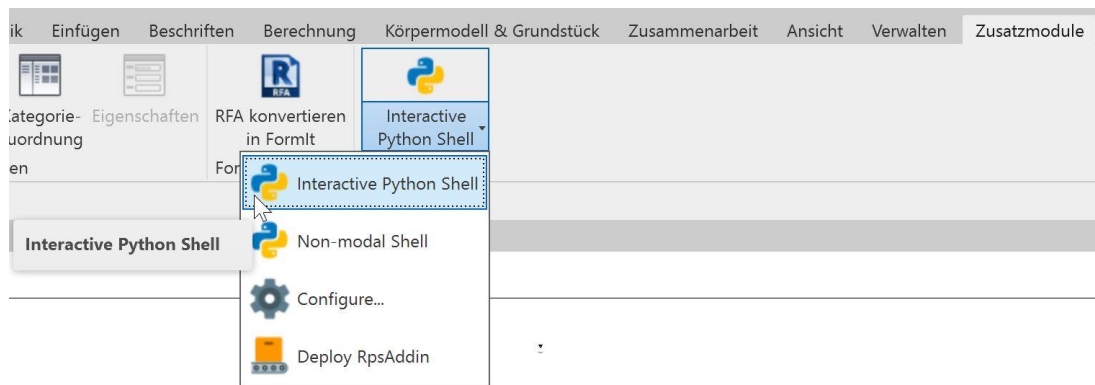


Abbildung 4-1: RevitPythonShell innerhalb von Revit

In dieser Arbeit wird ausschließlich die Option *Interactive Python Shell* benötigt. Durch diese Auswahl öffnet sich eine interaktive Python Konsole, in der Python Programmcode ausgeführt werden kann.

Die in Revit geöffnete Python-Konsole ermöglicht es, Python-Code in einer Umgebung (Python-Umgebung) auszuführen, welche Zugriff auf die Revit API hat. Damit über die interaktive Python Konsole Daten aus BIM-Modellen ausgelesen werden können, muss das entsprechende Gebäudemodell in Revit geöffnet sein. Python-Programme können über die in Abbildung 4-2 zu sehende Option (1) geöffnet und über die Auswahlmöglichkeit (2) ausgeführt werden. Die Ergebnisse werden dann in der Konsole ausgegeben (3).



Abbildung 4-2: Interaktive Python Konsole innerhalb von Revit

Um weitere Funktionen innerhalb einer Python Umgebung nutzen zu können, müssen digitale Bibliotheken (Packages) heruntergeladen und der jeweiligen Umgebung hinzugefügt werden. Da in der Revit-Umgebung benötigte Packages nicht installiert werden können, wird in dieser Arbeit auf eine zweite Umgebung außerhalb von Revit (PC-Umgebung) zurückgegriffen (s. Abbildung 4-3).



Abbildung 4-3: Darstellung zur Unterscheidung der Python Revit- und PC-Umgebung

In der Python-Umgebung wird neben dem Autodesk Package nur das Package Json verwendet. Für die PC-Umgebung werden die Packages Tkinter, NetworkX und Json benötigt und müssen installiert werden.

Autodesk

Das Package Autodesk ist kein installierbares, sondern ein innerhalb von Revit ausführbares Package, welches den Zugriff über die API innerhalb der Python-Umgebung ermöglicht. Dadurch können Elemente, Klassen und Methoden verwendet werden, die von der Revit API verstanden werden. Methoden ermöglichen es einen vordefinierten Programmcode im Python-Skript auszuführen.

Tkinter

Tkinter wird dazu verwendet, den Benutzer aufzufordern, eine Datei auszuwählen, die auf dem Computer vorhanden ist, und speichert den Dateipfad dieser ab.

Json

Um die BIM-Modellinformationen aus der Python-Umgebung in die PC-Umgebung zu transferieren, muss ein Datenformat gewählt werden, das leicht erstellt und wieder ausgelesen werden kann. Außerdem muss es sowohl in der Python-Umgebung erstellbar als auch in der PC-Umgebung lesbar sein (s. Abbildung 4-3). Diese Bedingungen erfüllt das Datenformat *JavaScript Object Notation* (Json).

NetworkX

Das Package NetworkX wird für die schlussendliche Erstellung der Graphen-Datei verwendet. Dieses Package ist in der Python-Umgebung nicht verfügbar. Somit muss der Graph innerhalb der PC-Umgebung erstellt werden.

4.1.5 NetworkX und GraphML

NetworkX ist ein Werkzeug, mit dem Netzwerkdarstellungen sortiert, strukturiert und erstellt werden können. Für diese Arbeit wird die Funktion genutzt, einfach und schnell komplexe Graphen in Python erstellen und abspeichern zu können. (Dokumentation zu NetworkX 3.1 2023)

Der Dokumentation zu NetworkX 3.1 (2023) ist zu entnehmen, dass bei der Erstellung eines Graphen zwischen vier NetworkX-Klassen unterschieden wird (s. Abbildung 4-4).

Networkx-Klasse	Typ	Selbstschleifen erlaubt	Parallele Kanten erlaubt
Graph	ungerichtet	Ja	NEIN
Digraph	gerichtet	Ja	NEIN
MultiGraph	ungerichtet	Ja	Ja
MultiDiGraph	gerichtet	Ja	Ja

Abbildung 4-4: Die vier NetworkX-Klassen inklusive ihrer definierten Eigenschaften (Dokumentation zu NetworkX 3.1 2023)

Das Modell, das im Kontext dieser Arbeit verwendet wird, muss keine parallelen Kanten abbilden können, da für eine Verbindung zwischen zwei Räumen immer ein neuer Knoten erstellt wird. Die einzelnen Kanten des Graphen müssen keine richtungsweisende Eigenschaft aufweisen, da Türen und Treppen in beide Richtungen benutzbar sind.

So ist nach der Dokumentation zu NetworkX 3.1 für diesen Anwendungsfall die NetworkX-Klasse *Graph* zu wählen. Nachdem der Graph erstellt wurde, muss er abgespeichert werden. NetworkX unterstützt mehrere Datenformate, die ein Graphen-basiertes Netzwerk darstellen können. In dieser Arbeit wird das Dateiformat GraphML zur Abspeicherung des in Python erstellten Graphen verwendet. Mit GraphML können ungerichtete und hierarchisch gelabelte Graphen problemlos abgespeichert und dargestellt werden. (Dokumentation zu NetworkX 3.1 2023)

Nachdem alle Programme, Packages und Datenformate vorgestellt wurden, wird im Folgenden auf die technische Umsetzung bei der Erstellung der Graphen-basierten Gebäudedarstellung eingegangen.

4.2 Erzeugung der Repräsentation

In [Kapitel 3.2](#) wurde ausführlich auf den Aufbau und die im Modell enthaltenen Informationen eingegangen. Die folgenden Kapitel stellen die in dieser Arbeit verwendete Methode vor, die Graphen-basierte Gebäudedarstellung zu generieren. Dabei wird auf die Vorgehensweise innerhalb der Python-Programme eingegangen. In dieser Arbeit wurden zwei Python-Programme erstellt: eins für die Revit-Umgebung (*Revit-Skript*) und eins für die PC-Umgebung (*Graph-Skript*).

4.2.1 Informationsgewinnung über Revit API – Python in der Revit-Umgebung

Das *Revit-Skript* enthält den zu öffnenden Code für die interaktive Python Konsole innerhalb von Revit (Abbildung 4-2). Um an die Informationen der Revit API zu gelangen, müssen API-interne Notationen eingehalten werden. Im Folgenden wird auf die Vorgehensweise eingegangen, die jeweiligen Informationen auszulesen. Über die Revit API Docs lassen sich die einzelnen Befehle und Notationen nachlesen (G. Talarico 2023).

Die Informationsabfrage über die API erfolgt über die jeweiligen im BIM-Modell vorhandenen Elemente ([s. Kapitel 3.1.2](#)) immer über die gleiche Art und Weise. Die Abfrage für das erste der vier gewählten Elemente (Räume, Stockwerke, Türen/Raumtrennung, Treppen) wird exemplarisch erklärt.

Zu beachten gilt, dass sämtliche Maßeinheiten, die über die API ausgegeben werden, in Fuß angegeben werden, sodass eine Umrechnung in Meter erfolgen muss.

Liste mit Elementen abfragen

Zunächst muss das zu untersuchende BIM-Modell in Python geladen werden. Daraufhin lassen sich mit den jeweiligen Kategorienamen die jeweiligen Elemente abfragen. Raumelemente werden über den Kategorienamen *OST_Rooms* über die API angefordert. Nach der Rückgabe werden die Ergebnisse sortiert und als eine Liste (Abfrageliste) mit allen angefragten Elementen zurückgegeben. (G. Talarico 2023)

Räume

Nachdem die Abfrage-Liste mit allen Raumelementen zurückgegeben wurde, können aus den einzelnen Elementen der Liste weitere Informationen ausgelesen werden. Damit die jeweilige Form eines Raumelements abgebildet werden kann, werden die Informationen über die Umrandungen der jeweiligen Räume (Raumgrenzen) benötigt.

Um diese zu erhalten, wird über die einzelnen Räume und deren jeweilige Umrandungen iteriert. Dadurch wird es ermöglicht, jede einzelne Raumgrenze zu untersuchen und deren Anfangs- und Endkoordinaten abzuspeichern. In diesem Schritt wird außerdem geprüft, ob es sich bei der jeweiligen Raumgrenze um eine Gerade handelt. Falls dies nicht zutrifft, wird unter Zuhilfenahme von 20 Punkten die Raumgrenze angenähert.

Die Informationen über den Raumnamen, den Flächeninhalt und das Stockwerk können direkt aus dem Raumelement ausgelesen werden.

Für jedes untersuchte Raumelement wird eine *Raum*-Klasse erstellt, in der alle benötigten Informationen abgespeichert werden. Zudem werden die einzelnen *Raum*-Klassen in einer Liste (Raum-Liste) abgespeichert.

Zur Überprüfung werden alle Informationen der jeweiligen Räume auf der interaktiven Python-Konsole in Revit ausgegeben.

Im weiteren Verlauf des Python-Programmes werden die Informationen zu allen Raumtrennungslinien benötigt. Deshalb wird beim Auslesen der Raumelemente eine Liste (Raumtrennung-Liste) erstellt, die alle vorhandenen Raumtrennungslinien enthält.

Stockwerke

Nachdem die Liste (Abfrage-Liste) der Stockwerkelemente abgefragt wurde, werden die einzelnen Elemente weiter untersucht. Die Informationen zum Namen und der Stockwerkserhöhung sind direkt im jeweiligen Stockwerkelement enthalten. Um die Anzahl der Räume einem Stockwerk zu erhalten, wird für jedes Stockwerkelement untersucht, ob sich ein Raum in diesem befindet.

Sämtliche gewonnenen Informationen jedes Stockwerkelements werden durch die Erstellung einer *Stockwerk*-Klasse abgespeichert. Die jeweiligen *Stockwerk*-Klassen werden einer neuen Liste (Stockwerk-Liste) hinzugefügt.

Horizontale Verbindung – Türen

Nach dem Erhalt der Liste (Abfrage-Liste) mit allen Türelementen, kann jedes Element genauer untersucht werden. Bei der Informationsgewinnung der Türelemente treten zwei Besonderheiten auf. Zum einen wird mithilfe einer sogenannten *BoundingBox* - einem Element, das den Bereich beschreibt, den eine Tür einnimmt - nach den beiden Räumen gesucht, die durch die Tür verbunden werden und deren IDs abgespeichert.

Zum anderen wird, ein sogenannter *LookupParameter* benötigt, um die Breite einer Tür auszulesen. Über einen *LookupParameter* kann das jeweilige Attribut über seinen Namen abgerufen werden. Dieser Name variiert je nach verwendeter Sprache in Autodesk Revit. In dieser Arbeit wird eine deutsche Installation von Revit verwendet. Da die Abfrage über die *LookupParameter* auch in einer englischen Version funktionieren soll, wurden sowohl deutsche als auch englische *LookupParameter* bei der Codeimplementierung beachtet.

Die Informationen der jeweiligen Türelemente werden als *Verbindungs*-Klasse abgespeichert und einer Liste (Tür-Liste) mit allen Türen hinzugefügt.

Horizontale Verbindung – Raumtrennungslinien

Um die zwei Räume zu erhalten, die durch eine Raumtrennungslinie miteinander verbunden sind, werden alle vorhandenen Raumtrennungslinien einzeln untersucht. Die Liste (Raumtrennung-Liste) der einzelnen Raumtrennungslinien wurde bereits bei der Untersuchung der Raumelemente ausgelesen und erstellt.

Für jedes Raumtrennungselement wird überprüft, ob sich die IDs der anschließenden Räume überschneiden. Raumtrennungslinien, die zwei Räume voneinander trennen, werden in die Liste der Türen mit abgespeichert.

Sämtliche Informationen der benötigten Raumtrennungslinien werden, wie die vorherigen Türelemente, in die Tür-Liste in Form der *Verbindungs*-Klasse abgelegt.

Treppen

Nach dem Auslesen der Liste aller vorhandenen Treppen, kann der Laufweg jeder Treppe ermittelt werden (zweidimensional im jeweiligen Stockwerk). Bei einer runden Ausbildung des Treppenfades wird, wie bei den Raumgrenzen, die Linie mit 20 Punkten angenähert.

Über den Treppenpfad der einzelnen Treppenelemente können die jeweils angrenzenden Raum- und Stockwerkelemente ermittelt werden.

Die restlichen Informationen zur Steigung, Breite und Position der jeweiligen Treppe können direkt aus dem Treppenelement ausgelesen werden.

Die Informationen der Treppenelemente werden in einer neuen Liste (Treppen-Liste) als einzelne *Treppen*-Klasse abgelegt.

Nachdem die Informationen von allen Elementen ausgelesen und in die jeweiligen Listen (Raum-Liste, Stockwerk-Liste, Tür-Liste, Treppen-Liste) abgespeichert wurden, werden sie in eine Json-Datei umgewandelt. Diese wird im gleichen Ordner wie die ursprüngliche BIM-Datei von Revit automatisch abgespeichert.

Auf diese Json-Datei kann dann mit dem zweiten Python-Skript (*Graph-Skript*) aus der PC-Umgebung zugegriffen werden.

4.2.2 Erstellung des Graphen mit NetworkX – Python in der PC-Umgebung

Durch die Erzeugung der Json-Datei kann aus der PC-Umgebung mithilfe des *Graph-Skripts* auf die Informationen des BIM-Modells zugegriffen werden. Dabei werden die einzelnen Elemente als Knoten und Kanten dargestellt und am Ende als GraphML-Datei abgespeichert.

In Python kann mit dem Package NetworkX ein Graph erstellt werden. Zunächst wird der entsprechende Graph im benötigten Format (*Graph*) erstellt. Daraufhin lassen sich mit Labels zugewiesene Knoten und Kanten dem Netzwerk hinzufügen.

Für die Erstellung des Graphen in dieser Arbeit, werden zwei Anpassungen im Vergleich zum beschriebenen Graph aus [Kapitel 3.2.2](#) vorgenommen. Diese Anpassungen ermöglichen einerseits eine schnellere Fehlersuche innerhalb von Autodesk Revit und andererseits eine Klassifizierung der Räume, die in Evakuierungsszenarien als Ausgang gelten.

Die schnellere Fehlersuche wird durch ein weiteres Label ermöglicht, dieses beinhaltet die jeweilige Revit-internen ID eines Elements. Dadurch lassen sich potenzielle Fehler leichter identifizieren und beheben.

Sämtliche Zonen, in denen sich Personen bewegen können, werden in dieser Arbeit in Revit mithilfe von Raumelementen festgelegt ([s. Kapitel 3.1.2](#)). Aus diesem Grund können auch Bereiche, die sich außerhalb des Gebäudes befinden, als Raumelement

klassifiziert werden, indem der jeweilige Bereich mit Raumtrennungslinien eingegrenzt wird. So können Zonen hinzugefügt werden, die bei einer Evakuierung als Ausgang gelten. Um einen Bereich als Ausgang zu markieren, muss in der Revit-Datei im Namen des Raumelements das Wort *EXIT* enthalten sein.

Bei der Erstellung des Graphen wird die Form der Ausgangsräume nicht mit angegeben, da davon ausgegangen wird, dass es sich dabei um eine Freifläche handelt. Um einen Ausgangsraum in der Netzwerkdarstellung von normalen Räumen abzugrenzen, werden Ausgangsräume als Knoten vom Type *exit* erstellt.

Die Speicherung der fertigen GraphML-Datei erfolgt wie bei der Json-Datei automatisch im gleichen Ordner, in dem sich auch das Revit-Modell befindet.

4.2.3 Hierarchische Graphen-basierte Repräsentation – Ergebnis

Nach der vorgestellten Methode können hierarchische Graphen-basierte Repräsentationen aus Gebäudemodellen erstellt werden, die mit dem BIM-Tool Autodesk Revit entworfen wurden. Im Folgenden wird das Vorgehen beschrieben.

Im ersten Schritt muss das Gebäudemodell (BIM-Modell) innerhalb von Revit geöffnet werden (s. Abbildung 4-5). Daraufhin kann das *Revit-Skript* über die Interactive Python Konsole (Revit-Umgebung) geöffnet und ausgeführt werden (Abbildung 4-2 Option 2). In der Konsole können sämtliche erfasste Informationen und Elemente überprüft werden, die in der Json-Datei abgespeichert wurden. Als letzten Schritt muss das *Graph-Skript* in der PC-Umgebung ausgeführt werden. Dabei wird eine Python Installation auf dem PC vorausgesetzt. Nachdem die erzeugte Json-Datei nach einer Benutzeraufforderung ausgewählt wurde, wird die GraphML-Datei erzeugt.

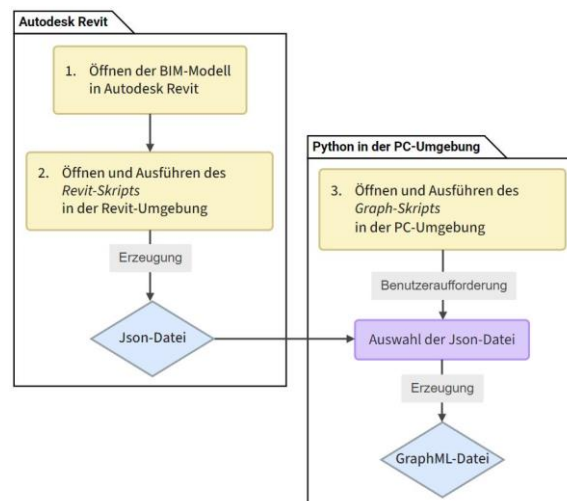


Abbildung 4-5: Ablauf der Methode zur Erstellung einer Graphen-basierten Repräsentation

In Abbildung 4-6 ist ein dreidimensionales Gebäude mit zwei Stockwerken zu sehen. Alle Räume wurden entsprechend benannt, und der Bereich außerhalb des Gebäudes wurde mit Raumtrennungslinien begrenzt und Ausgang-EXIT benannt. Die genaue Raumaufteilung ist in Abbildung 4-7 zu sehen. Zudem sind in Abbildung 4-7 die, für die Graphen-basierte Repräsentation relevanten Bereiche und Übergänge farblich markiert.

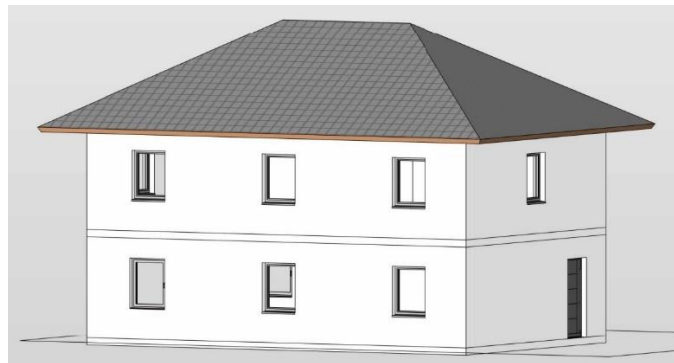


Abbildung 4-6: BIM-Modell eines zweistöckigen Gebäudes – Autodesk Revit

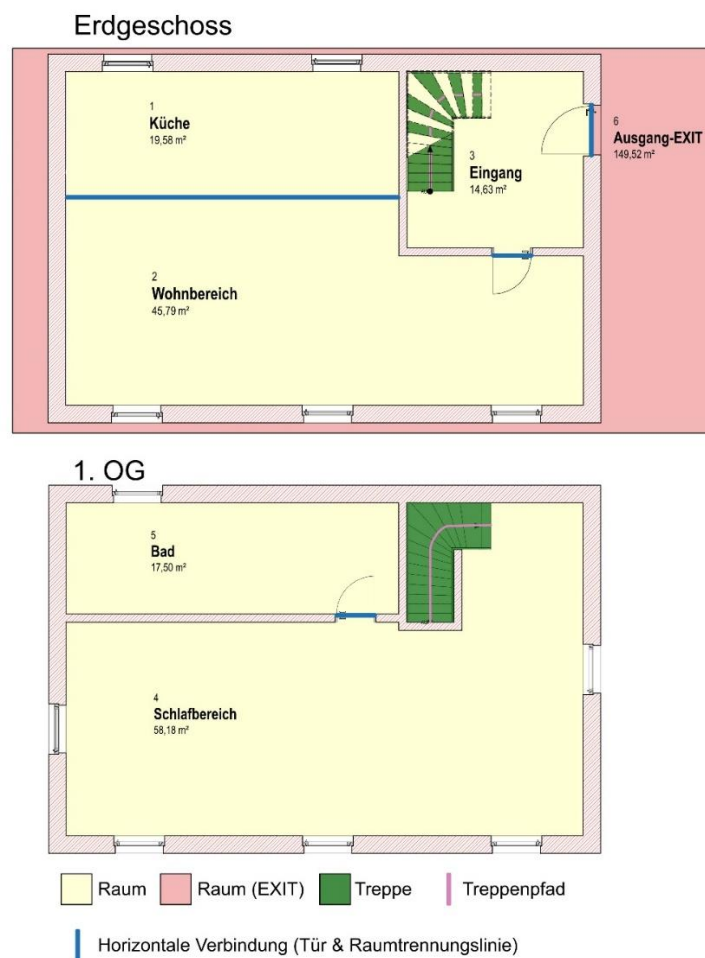


Abbildung 4-7: Grundrisse (EG und 1.OG) des Gebäudes aus Abbildung 4-6 mit farblich markierten relevanten Bereichen

Nachdem die beschriebenen Schritte zur Erstellung der Graphen-basierten Darstellung durchgeführt wurden, kann die erzeugte GraphML-Datei visualisiert werden. Um das Netzwerk in Abbildung 4-8 optisch darzustellen, wurde das Visualisierungsprogramm Cytoscape verwendet (Cytoscape 2023).

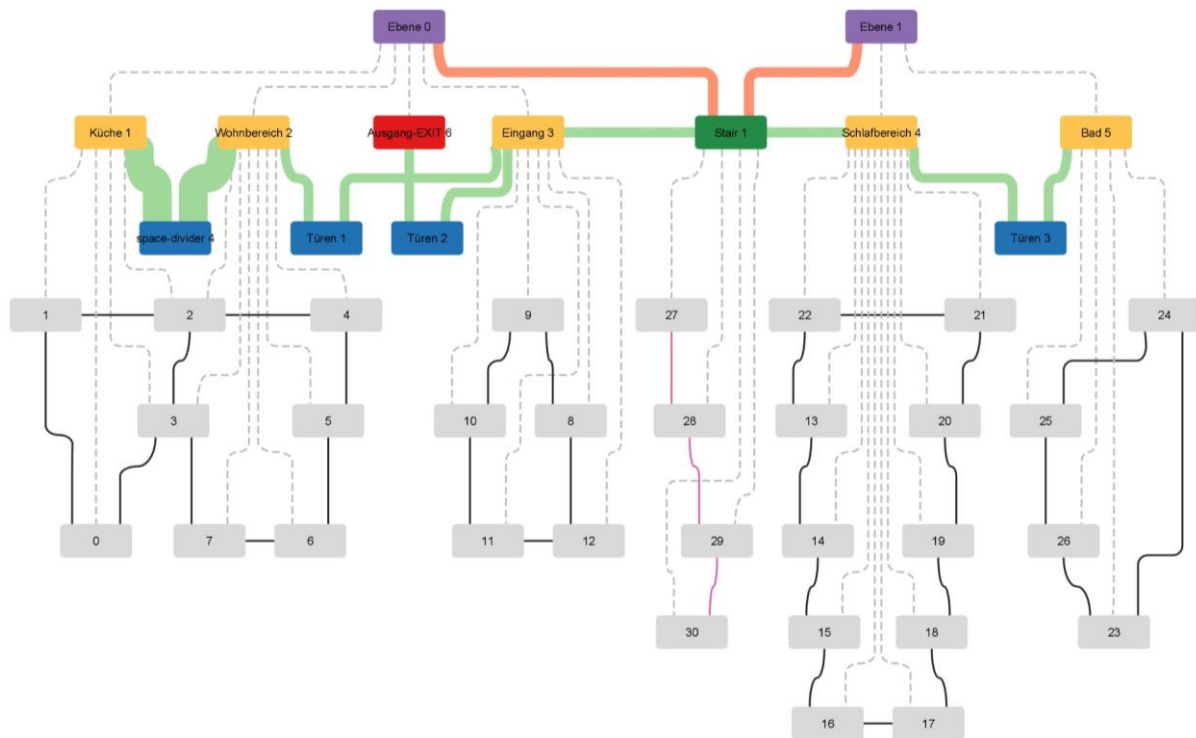


Abbildung 4-8: Graphen-basierte Darstellung des zweistöckigen Gebäudes aus Abbildung 4-6 und 4-7

Abbildung 4-8 zeigt die Graphen-basierte Darstellung des mit BIM erstellten Gebäudes aus Abbildung 4-6 und 4-7. Damit zwischen den einzelnen Knoten und Kanten leicht unterschieden werden kann, wurden die jeweiligen Knoten- und Kantentypen farblich dargestellt.

Die einzelnen Stockwerks-Knoten (lila) sind über Treppen-Knoten (dunkelgrün) mit Kanten (orange) miteinander verbunden. Türen und Raumtrennungslinien (blau) sowie Treppen sorgen für eine Verbindung über Kanten (hellgrün) der einzelnen Raum-Knoten (gelb) miteinander. Wobei eine dickere Kante einen breiten Durchgang symbolisiert.

Die Punkt-Knoten (hellgrau) der einzelnen Räume beschreiben jeweils mit benachbarten Punkt-Knoten die Form des Raumes durch Kanten (dunkelgrau). Die Pfadpunkte der Treppen sind mit dem jeweiligen Folgepunkt verbunden (rosa). Alle hierarchischen Beziehungen sind durch gestrichelte Linien (grau) visualisiert.

5 Anwendungsbereich und Anwendbarkeit

Mithilfe der entworfenen Methode können Graphen-basierte Repräsentationen von Gebäuden direkt aus BIM-Modellen gewonnen werden.

Die in dieser Arbeit entwickelte Methode erzeugt ein Modell, auf dessen Informationsinhalt basierend DL-Methoden trainiert werden können. Durch dieses Modell wird eine Grundlage geschaffen, Einschätzungen hinsichtlich Personenströme mit KI treffen zu können. Allerdings kann die Methode auch für die Erstellung anderer Modellabstraktionen in unterschiedlichen Anwendungsbereichen verwendet werden.

5.1 Vorteile der entwickelten Methode

Ein großer Vorteil der entwickelten Methode besteht darin, dass die meisten Informationen aus den Raumelementen des BIM-Modells gewonnen werden. Diese Raumelemente können leicht eingebaut werden, ohne große Veränderungen am BIM-Modell vornehmen zu müssen. Dadurch ist eine Übertragbarkeit auf bestehende Modelle gegeben. Das BIM-Modell muss nicht nach den Vorgaben der Methodik entworfen werden, sondern die Raumelemente können wie beschrieben im Nachhinein hinzugefügt werden.

Ein weiterer Vorteil ist dadurch gegeben, dass durch die Verwendung von Raumtrennungslinien problemlos Bereiche ausgeschlossen oder hinzugefügt werden können. So kann ein Raumelement mithilfe von Raumtrennungslinien beispielsweise auf einer Dachterrasse als zusätzlicher Bewegungsraum oder vor dem Gebäude als Ausgang (EXIT) eingefügt werden.

Außerdem können, durch das Entfernen von Raumelementen, ganze Zimmer oder Gebäudebereiche für die Erzeugung der Graphen-basierten Repräsentation ausgeschlossen werden. Dadurch wird nicht der komplette Entwurf des Gebäudes zerstört und Raumelemente können dem BIM-Modell leicht wieder hinzugefügt werden.

5.2 Hierarchieebenen

Die einzelnen hierarchischen Ebenen des Graphen bringt viele genannte Vorteile ([s. Kapitel 2.4.2](#)). Es ist ebenso möglich, diese hierarchischen Ebenen anzupassen, zu ändern oder zu erweitern.

Die resultierende Netzwerkdarstellung eines BIM-Modells stellt in der Raumebene die Geometrie der einzelnen Räume dar ([s. Kapitel 3.2.1](#)). Allerdings besitzen diese auf Raumebene keine Öffnungen, Türen oder Ähnliches. Dadurch können bei alleiniger Betrachtung der Raumebene keine Verbindungen zwischen einzelnen Räumen untersucht werden. Alternativ könnte ein Netzwerk mit weniger oder keiner hierarchischen Ebene erstellt werden.

In der erzeugten hierarchischen Graphen-basierten Repräsentation ist es allerdings möglich, weitere Ebenen hinzuzufügen. So können auch Bauprojekte, bei denen mehrere Häuser geplant werden, durch Hinzufügen einer übergeordneten Ebene, die als Knoten die einzelnen Gebäude enthält, mit der Darstellung dieser Arbeit abgebildet werden.

5.3 Analyse zur technischen Umsetzung

Das BIM-Tool Autodesk Revit bietet eine leichte Zugriffsmöglichkeit, um die Informationen des BIM-Modells mithilfe der Revit API auszulesen. Dadurch lässt sich die vorgestellte Methodik leicht implementieren. Potential besteht bei der Implementierung der Methode innerhalb einer Python-Umgebung (entweder Revit- oder PC-Umgebung). Dies würde zu einer vereinfachten Erzeugung der Netzwerkdarstellung führen, da nur ein Python-Programm benötigt wird und so die Graphen-basierte Repräsentation des Gebäudes mit einem Klick erstellbar wäre.

Alternativ gibt es die Möglichkeit, die für das zu erstellende Modell benötigten Informationen aus einer *Industry Foundation Classes*-Datei (IFC) zu gewinnen. Diese können aus BIM-Modellen exportiert werden. In dieser Arbeit wurde der Bereich, in dem sich Personen fortbewegen, mithilfe von Raumelementen definiert. In IFC können sogenannte *Spaces* erstellt und benannt werden, die den Raumelementen ähnlich sind. Allerdings können topologische Beziehungen, wie die Raumtrennungslinien, nicht in IFC abgebildet werden. Zudem bedarf es bei der Abspeicherung als IFC-Datei eines zusätzlichen Arbeitsschritts. Dieser wurde bei der entwickelten Methode durch den direkten Aufruf der Python-Konsole in Autodesk Revit vermieden. (IFC4.3.1.0 Documentation 2023)

Um die Informationen über die einzelnen Räume eines Gebäudes zu erhalten, wäre auch die Eingrenzung der Bereiche durch Wände anstatt durch Raumelementen umsetzbar gewesen. Dabei wird der komplette Bewegungsbereich von Personen durch

die einzelnen Wände des Gebäudes begrenzt. Bei dieser Möglichkeit können allerdings keine Bereiche ausgeschlossen werden, ohne Wände aus dem BIM-Modell zu löschen und Darstellungen von Ausgängen (EXIT) sind nicht möglich.

Zusätzlich kann die Methode bei Bedarf so angepasst werden, dass mehrere BIM-Modelle automatisch im Hintergrund geöffnet und als Netzwerkdarstellung abgespeichert werden.

6 Zusammenfassung und Fazit

Diese Bachelorarbeit befasst sich mit der Entwicklung und Erstellung Graphen-basierter Darstellungen von BIM-Modellen, um unter Einsatz von Künstlicher Intelligenz (KI) Einschätzungen bezüglich Personenströme in Gebäuden treffen zu können.

Hierzu werden zunächst grundlegende Gebäudeinformationen, die für die Fragestellung (Personenströme während einer Evakuierung) benötigt werden, aus den BIM-Modellen extrahiert und im Anschluss aus diesen eine Graphen-basierte Darstellung erzeugt. In dieser Arbeit werden in Autodesk Revit erstellte BIM-Modelle über die Revit API mithilfe von Python in die Form einer Graphen-basierten Darstellung gebracht. Die erzeugte Graphen-basierte Darstellung wird als maschinenlesbare GraphML-Datei ausgegeben und kann zum Beispiel mit Cytoscape visualisiert werden.

Die Anforderungen an diesen Graphen sind eine einfache Erzeugung und die Möglichkeit diesen auf eine Vielzahl von Gebäudemodellen anwenden zu können, um einen aussagekräftigen Datensatz für das Training von Deep Learning (DL) Methoden zu erzeugen. Die Anwendbarkeit auf bestehende BIM-Modelle ist wie beschrieben gegeben, da die Informationsgewinnung über die jeweiligen Raumelemente stattfindet und diese problemlos bestehenden BIM-Modellen hinzugefügt werden können.

Die Erstellung des Graphen wird allerdings durch die Tatsache erschwert, dass auf zwei Python-Umgebungen zurückgegriffen werden muss. Eine technische Lösung dieses Problems konnte im Verlauf der Arbeit nicht gefunden werden. Die Möglichkeiten auf interne Variablen von Autodesk Revit aus der PC-Umgebung zuzugreifen oder das NetworkX Package in der Revit-Umgebung zu installieren, könnten das Problem beheben.

In Zukunft können mit dieser Methode umfangreiche Datenmengen in Form von Graphen-basierten Darstellungen erzeugt werden. Diese können als Grundlage beim Training von KI verwendet werden und dadurch zur Lösung verschiedenster Fragestellungen beitragen.

Literaturverzeichnis

Abir, I. Mohammad; Ibrahim, A. Mohd; Toha, S. Fauziah; Shafie, A. Akramin (2022): A review on the hospital evacuation simulation models. In: *International Journal of Disaster Risk Reduction* 77, S. 103083. DOI: 10.1016/j.ijdr.2022.103083.

accu:rate - Institute for crowd simulation (2023): Kompatible CAD Software -
accu:rate - Institute for crowd simulation. Online verfügbar unter <https://www.accu-rate.de/de/kompatible-cad-software/>, zuletzt aktualisiert am 07.02.2023, zuletzt geprüft am 14.04.2023.

Aggarwal, K.; M. Mijwil, M.; Sonia; Al-Mistarehi, A.-H.; Alomari, S.; Gök, M. et al. (2022): Has the Future Started? The Current Growth of Artificial Intelligence, Machine Learning, and Deep Learning. In: *ijcsm*, S. 115–123. DOI: 10.52866/ijcsm.2022.01.01.013.

Baduge, S. Kristombu; Thilakarathna, S.; Perera, J. Shalitha; Arashpour, M.; Sharafi, P.; Teodosio, B. et al. (2022): Artificial intelligence and smart vision for building and construction 4.0: Machine and deep learning methods and applications. In: *Automation in Construction* 141, S. 104440. DOI: 10.1016/j.autcon.2022.104440.

Biedermann, D. H.; Borrmann, A. (2016): A generic and hybrid approach for pedestrian dynamics to couple cellular automata with network flow models. Online verfügbar unter https://www.researchgate.net/profile/Daniel-Biedermann/publication/309322158_A_generic_and_hybrid_approach_for_pedestrian_dynamics_to_couple_cellular_automata_with_network_flow_models/links/5809ffff08ae45e02c0d6249/A-generic-and-hybrid-approach-for-pedestrian-dynamics-to-couple-cellular-automata-with-network-flow-models.pdf, zuletzt geprüft am 14.04.2023.

Biedermann, D. H.; Clever, J.; Borrmann, A. (2021): A generic and density-sensitive method for multi-scale pedestrian dynamics. In: *Automation in Construction* 122, S. 103489. DOI: 10.1016/j.autcon.2020.103489.

Borrmann, A.; Koch, C.; König, M.; Beetz, J. (2015): Building Information Modeling. Technologische Grundlagen und industrielle Praxis (ISBN 978-3-658-05605-6), zuletzt aktualisiert am 27.03.2023, zuletzt geprüft am 27.03.2023.

Borrmann, A.; König, M. (2021): Building Information Modeling. In: Wendehorst Bau-technische Zahlentafeln: Springer Vieweg, Wiesbaden, S. 1643–1652. Online verfügbar unter https://link.springer.com/chapter/10.1007/978-3-658-32218-2_24.

Bünthe, O. (2023): ChatGPT: Rekord-Wachstum und Abo-Modell in den USA. In: *heise online*, 02.02.2023. Online verfügbar unter <https://www.heise.de/news/Chat-GPT-startet-Abo-Modell-in-den-USA-7480052.html>, zuletzt geprüft am 13.04.2023.

Chartrand, G.; Lesniak, L.; Zhang, P. (2011): Graphs & Digraphs. Fifth Edition. Hg. v. Chapman & Hall/CRC (International Standard Book Number-13: 978-1-4398-2629-4), zuletzt aktualisiert am 02.04.2023, zuletzt geprüft am 02.04.2023.

Ciobâ, S. M.; Murty, M. Ram (2022): A First Course in Graph Theory and Combinatorics. Hg. v. Springer (ISBN 978-981-19-1335-8), zuletzt aktualisiert am 01.04.2023, zuletzt geprüft am 01.04.2023.

Cytoscape (2023): An Open Source Platform for Complex Network Analysis and Visualization. Online verfügbar unter <https://cytoscape.org/>, zuletzt aktualisiert am 10.04.2023, zuletzt geprüft am 17.04.2023.

Dokumentation zu NetworkX 3.1 (2023). Software für komplexe Netzwerke. Online verfügbar unter <https://networkx.org/documentation/stable/index.html>, zuletzt aktualisiert am 14.04.2023, zuletzt geprüft am 15.04.2023.

Dong, Hairong; Zhou, Min; Wang, Qianling; Yang, Xiaoxia; Wang, Fei-Yue (2020): State-of-the-Art Pedestrian and Evacuation Dynamics. Online verfügbar unter <https://ieeexplore.ieee.org/abstract/document/8718399>, zuletzt geprüft am 13.04.2023.

IFC4.3.1.0 Documentation (2023). Online verfügbar unter <https://ifc43-docs.standards.buildingsmart.org/IFC/RELEASE/IFC4x3/HTML/chapter-8/>, zuletzt aktualisiert am 04.04.2023, zuletzt geprüft am 04.04.2023.

Introducing ChatGPT (2023). Online verfügbar unter <https://openai.com/blog/chatgpt>, zuletzt aktualisiert am 13.04.2023, zuletzt geprüft am 13.04.2023.

Kavlakoglu, E. (2020): AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?, 27.05.2020. Online verfügbar unter <https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks>, zuletzt geprüft am 13.04.2023.

Kontarinis, A.; Zeitouni, K.; Marinica, C.; Vodislav, D.; Kotzinos, D. (2021): Towards a semantic indoor trajectory model: application to museum visits. In: *Geoinformatica* 25 (2), S. 311–352. DOI: 10.1007/s10707-020-00430-x.

Lai, H.; Deng, X. (2018): INTEROPERABILITY ANALYSIS OF IFC-BASED DATA EXCHANGE BETWEEN HETEROGENEOUS BIM SOFTWARE. In: 1 24 (7), S. 537–555. DOI: 10.3846/jcem.2018.6132.

Lammers, J. (2023): KI und Maschinelles Lernen – Bedeutung, Anwendungen und Herausforderungen. In: *Gruenderfreunde.de*, 13.02.2023. Online verfügbar unter <https://gruenderfreunde.de/ki-und-maschinelles-lernen-bedeutung-anwendungen-und-herausforderungen/>, zuletzt geprüft am 18.04.2023.

Larochelle, H.; Bengio, Y.; Louradour, J.; Lamblin, P. (2009): Exploring Strategies for Training Deep Neural Networks. Hg. v. Université de Montréal. Montréal, Canada. Online verfügbar unter <https://www.jmlr.org/papers/volume10/larochelle09a/larochelle09a.pdf>, zuletzt aktualisiert am 13.04.2023, zuletzt geprüft am 13.04.2023.

Lee, A.-F.; Tsai, P.-H. (2023): Online Indoor Fire Evacuation System. Online verfügbar unter <https://ieeexplore.ieee.org/abstract/document/10040565>, zuletzt aktualisiert am 13.04.2023, zuletzt geprüft am 13.04.2023.

Liu, Q.; Lu, L.; Zhang, Y.; Hu, M. (2022): Modeling the dynamics of pedestrian evacuation in a complex environment. In: *Physica A: Statistical Mechanics and its Applications* 585, S. 126426. DOI: 10.1016/j.physa.2021.126426.

Mayachita, Inneke (Hg.) (2020): Understanding Graph Convolutional Networks for Node Classification. International Conference on Advanced Machine Learning Technologies and Applications. Online verfügbar unter <https://towardsdatascience.com/>, zuletzt geprüft am 01.04.2023.

Minguk, K.; Youngjib, H.; Choongwan, K.; Tae Wan, K. (2023): Simulating travel paths of construction site workers via deep reinforcement learning considering their spatial cognition and wayfinding behavior. In: *Automation in Construction* 147, S. 104715. DOI: 10.1016/j.autcon.2022.104715.

Perret, B.; Chierchia, G.; Cousty, J.; Guimarães, S.; Kenmochi, Y.; Najman, N. (2019): Higura: Hierarchical Graph Analysis. In: *SoftwareX* 10, S. 100335. DOI: 10.1016/j.softx.2019.100335.

Pu, Shi; Zlatanova, Sisi (2005): Geo-information for disaster management. First International Symposium on Geo-information for Disaster Management. Berlin, Heidelberg: Springer. Online verfügbar unter https://www.researchgate.net/publication/228621962_Evacuation_Route_Calculation_of_Inner_Buildings.

Python.org (2023): Welcome to Python.org. Online verfügbar unter <https://www.python.org/about/>, zuletzt aktualisiert am 15.04.2023, zuletzt geprüft am 15.04.2023.

Release 2.0.0 · architecture-building-systems/revitpythonshell · GitHub (2023). Online verfügbar unter <https://github.com/architecture-building-systems/revitpythonshell/releases/tag/2.0.0>, zuletzt aktualisiert am 27.04.2023, zuletzt geprüft am 27.04.2023.

Revit-Software | BIM-Software | Autodesk (2023). Online verfügbar unter <https://www.autodesk.de/products/revit/overview?term=1-YEAR&tab=subscription>, zuletzt aktualisiert am 02.04.2023, zuletzt geprüft am 02.04.2023.

SimPlan AG (2022): Schnellere Abläufe mit der Personenstromsimulation. Online verfügbar unter <https://www.simplan.de/services/personenstromsimulation/>, zuletzt aktualisiert am 27.09.2022, zuletzt geprüft am 29.04.2023.

Stoffel, E.-P.; Schoder, K.; Ohlbach, H. Jürgen (op. 2008): Applying hierarchical graphs to pedestrian indoor navigation. In: Walid G. Aref (Hg.): Proceedings of the 6th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS 2008), November 5-7, 2008, Irvine, California, USA. GIS '08: 16th International Symposium on Advances in Geographic Information Systems. Irvine California, 05 11 2008 07 11 2008. New York: Association for Computing Machinery, S. 1–4.

Talarico, G. (2023): Revit API Docs 2022. Online Documentation for the Revit API, zuletzt aktualisiert am 14.04.2023, zuletzt geprüft am 14.04.2023.

Thomas, D.; Kron, Z.; Hogge, A.; Moorhouse, J.; et al. (2023): GitHub - architecture-building-systems/revitpythonshell: An IronPython scripting environment for Autodesk Revit and Vasari. Online verfügbar unter <https://github.com/architecture-building-systems/revitpythonshell>, zuletzt aktualisiert am 15.04.2023, zuletzt geprüft am 15.04.2023.

Was ist ein Application Programming Interface (API)? | IBM (2023). Online verfügbar unter <https://www.ibm.com/de-de/topics/api>, zuletzt aktualisiert am 15.04.2023, zuletzt geprüft am 15.04.2023.

Yang, L.; Worboys, M. (2015): Generation of navigation graphs for indoor space. In: *International Journal of Geographical Information Science* 29 (10), S. 1737–1756. DOI: 10.1080/13658816.2015.1041141.

Yang, M.; Wu, C.; Guo, Y.; Jiang, R.; Zhou, F.; Zhang, J.; Yang, Z. (2023): Transformer-based deep learning model and video dataset for unsafe action identification in construction projects. In: *Automation in Construction* 146, S. 104703. DOI: 10.1016/j.autcon.2022.104703.

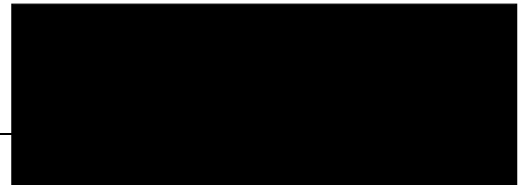
Yang, X.; Zhang, R.; Li, Y.; Yang, Y.; Qu, D.; Liu, T.; Zhao, B. (2022): Fuzzy-theory-based pedestrian dynamics models for studying the waiting passenger distribution at the subway platform. In: *Tunnelling and Underground Space Technology* 129, S. 104680. DOI: 10.1016/j.tust.2022.104680.

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Bachelor-Thesis selbstständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich benannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.

Ich versichere außerdem, dass die vorliegende Arbeit noch nicht einem anderen Prüfungsverfahren zugrunde gelegen hat.

München, 2. Mai 2023



Sebastian Bayer

