der Bundeswehr
Universität München

INSTITUT FÜR MECHANIK UND STATIK
DER UNIVERSITÄT DER BUNDESWEHR MÜNCHEN

Modellunabhängiger Ansatz zur Erstellung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen

Amar Rahimi

Dissertation

2023

Modellunabhängiger Ansatz zur Erstellung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen

Amar Rahimi

Vollständiger Abdruck der von der Fakultät für Bauingenieurwesen und Umweltwissenschaften der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs
- Dr.-Ing. -

genehmigten Dissertation.

Gutachter:

1. Prof. Dr.-Ing. habil. Norbert Gebbeken (EE)
2. Prof. Dr.-Ing. Casimir Katz

Die Dissertation wurde am 20. Dezember 2023 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Bauingenieurwesen und Umweltwissenschaften am 10. April 2024 angenommen. Die mündliche Prüfung fand am 14. Mai 2024 statt.

Abstract

After the complete or partial collapse of a building due to a severe earthquake, immediate search and rescue efforts are required to locate survivors. The structural information of the seismically damaged building plays a crucial role in the safe execution of rescue operations. In this context, Building Information Modeling (BIM) technology provides the possibility to manage collected data of the building in a digital 3D model in an object-oriented manner. The use of a BIM model in the event of a disaster offers a promising approach to optimize the planning of rescue operations and thus increase the safety of rescue forces. However, this requires that the BIM model can be generated quickly and user-friendly.

In this thesis, an innovative, model-independent approach for the generation of BIM models of seismically damaged reinforced concrete frame structures is introduced. The focus is on the algorithmic processing of 3D point clouds which are used as the sole data basis for model creation. The 3D point cloud, obtained by photogrammetry or laser scanning, represents the damaged state of the building in the form of point data. The developed methodology begins with pre-processing and subsequently employs several segmentation algorithms for shape-based structuring of the point data. These include methods for segmenting planar and linear point sets as well as extracting wall point data from frame segments. Based on the segmented point data, the object and material classification for semantic information enrichment, the 3D reconstruction for digital model generation and the conversion of the generated model data into the BIM process are perfomed. The performance of these processing steps was evaluated using real case studies, including a partially collapsed reinforced concrete frame building in Lyon and seismically damaged reinforced concrete frame structures in Jindires and Gölcük.

The results of the present work show that the developed methodology enables the semi-automatic generation of BIM models of seismically damaged reinforced concrete frame structures, achieving a fundamental level of detail based on 3D point clouds. For the case studies, BIM models could be generated within a maximum of 20 minutes using the model-independent approach. This represents a significant advance in the field of *Scan-to-BIM* for the considered use case in which only model-dependent methods have been taken into account to date. In particular, impressive performance was observed in the segmentation of planar point sets, object classification and conversion to the BIM format. However, the segmentation of wall point data from frame segments posed a challenge for irregular geometries of damaged masonry walls. In addition, the developed material classification method revealed potential for improvement in the material prediction of heavily shaded walls. Other challenges identified in this work resulted largely from the limited data collection in the case studies where only the externally visible surfaces of the buildings were scanned using photogrammetry.

Kurzfassung

Nach dem vollständigen oder teilweisen Einsturz eines Gebäudes infolge eines schweren Erdbebens sind sofortige Such- und Rettungsmaßnahmen erforderlich, um Überlebende zu lokalisieren. Die Strukturinformationen des seismisch beschädigten Gebäudes spielen eine entscheidende Rolle bei der sicheren Durchführung von Rettungsmaßnahmen. In diesem Kontext bietet die Building Information Modeling (BIM)-Technologie die Möglichkeit, gesammelte Daten des Gebäudes in einem digitalen 3D-Modell objektorientiert zu verwalten. Die Verwendung eines BIM-Modells im Katastrophenfall stellt einen vielversprechenden Ansatz dar, um die Planung von Rettungseinsätzen zu optimieren und damit die Sicherheit der Rettungskräfte zu erhöhen. Dies setzt jedoch voraus, dass das BIM-Modell schnell und benutzerfreundlich generiert werden kann.
In dieser Arbeit wird ein innovativer, modellunabhängiger Ansatz zur Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen vorgestellt. Der Schwerpunkt liegt auf der algorithmischen Verarbeitung von dreidimensonalen (3D)-Punktwolken, die als alleinige Datengrundlage für die Modellerstellung verwendet werden. Dabei repräsentiert die 3D-Punktwolke, die durch Photogrammetrie oder Laserscanning gewonnen wird, den beschädigten Zustand des Gebäudes in Form von Punktdaten. Die entwickelte Methodik beginnt mit einer Vorverarbeitung und nutzt anschließend mehrere Segmentierungsalgorithmen zur formbasierten Strukturierung der Punktdaten. Hierzu gehören Verfahren zur Segmentierung von ebenen und linearen Punktmengen sowie zur Extraktion von Wandpunktdaten aus Rahmensegmenten. Basierend auf den segmentierten Punktdaten werden die Objekt- und Materialklassifikation zur semantischen Informationsanreicherung, die 3D-Rekonstruktion zur digitalen Modellerzeugung sowie die Umwandlung der generierten Modelldaten in den BIM-Prozess durchgeführt. Die Leistungsfähigkeit dieser Verarbeitungsschritte wurde anhand realer Fallbeispiele evaluiert, darunter ein teilweise eingestürztes Stahlbetonrahmengebäude in Lyon sowie seismisch beschädigte Stahlbetonrahmenkonstruktionen in Jindires und Gölcük.
Die Ergebnisse der vorliegenden Arbeit zeigen, dass der entwickelte Ansatz die halbautomatische Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen ermöglicht und dabei ein grundlegender Detaillierungsgrad auf Basis von 3D-Punktwolken erreicht wird. Für die Fallbeispiele konnten mit dem modellunabhängigen Ansatz BIM-Modelle innerhalb von maximal 20 Minuten erzeugt werden. Dies stellt einen bedeutenden Fortschritt im Bereich des *Scan-to-BIM* für den betrachteten Anwendungsfall dar, in dem bisher ausschließlich modellabhängige Methoden berücksichtigt wurden. Insbesondere bei der Segmentierung von ebenen Punktmengen, der Objektklassifizierung und dem Prozess zur Umwandlung in den BIM-Format wurde eine beeindruckende Leistung festgestellt. Die Segmentierung von Wandpunktdaten aus Rahmensegmenten stellte jedoch bei unregelmäßigen Geometrien von beschädigten Mauerwerkswänden eine Herausforderung dar. Darüber hinaus offenbarte die entwickelte Materialklassifizierungsmethode Verbesserungspotential bei der Materialvorhersage von stark beschatteten Wänden. Weitere Herausforderungen, die in dieser Arbeit identifiziert wurden, resultierten größtenteils aufgrund der eingeschränkten Datenerfassung in den Fallbeispielen, bei denen nur die von außen sichtbaren Oberflächen der Gebäude mit Hilfe der Photogrammetrie gescannt wurden.

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Mechanik und Statik der Universität der Bundeswehr München.
Zu Beginn möchte ich Herrn Prof. Dr.-Ing. habil. Norbert Gebbeken (EE) meinen aufrichtigen Dank für die Betreuung und förderliche Begleitung dieser Arbeit aussprechen. Seine Unterstützung während der gesamten Promotionsphase sowie die großzügige Gewährung von Freiheiten bei der Bearbeitung der Dissertation haben maßgeblich zum Erfolg beigetragen.
Herrn Prof. Dr.-Ing. Casimir Katz vom Lehrstuhl für Computergestützte Modellierung und Simulation an der Technischen Universität München möchte ich für sein Interesse an meiner Arbeit und seine Bereitschaft, die Position des Zweitgutachters zu übernehmen, ganz herzlich danken.
Des Weiteren gilt mein Dank Herrn Prof. Dr.-Ing. Alexander Popp für die Übernahme des Vorsitzes im Promotionsverfahren.
Ein großes Dankeschön geht auch an alle meine Kollegen am Institut für Mechanik und Statik für die angenehme Zusammenarbeit, den regen wissenschaftlichen Austausch und die schöne Zeit, die wir gemeinsam hatten.
Meinen Eltern, Abdul Menan und Zahra, danke ich von ganzem Herzen für ihre unermüdliche Unterstützung und elterliche Fürsorge. Gerade in schwierigen Zeiten konnte ich aus dieser liebevollen Unterstützung immer wieder Kraft und Motivation schöpfen.
Ein besonderer Dank gilt meinem Bruder, Dr. rer. nat. Muslem Rahimi, für seine wertvollen Ratschläge und seine kritische Durchsicht dieser Arbeit.
Mein größter Dank gilt meiner wunderbaren Ehefrau Mariam. Es fällt mir schwer, in Worte zu fassen, wie dankbar ich dir für deine jahrelange Geduld und bedingungslose Unterstützung bin. Ohne dich wäre die Fertigstellung dieser Arbeit nicht möglich gewesen. Ich stehe für immer in deiner Schuld.

München, Dezember 2023 Amar Rahimi

Inhaltsverzeichnis

1 Einleitung 1
1.1 Motivation . . . 1
1.2 Forschungsfragen und Herausforderungen . . . 3
1.2.1 Forschungsfragen . . . 3
1.2.2 Herausforderungen . . . 3
1.3 Aufbau der Arbeit . . . 4

2 Stand der Wissenschaft und Technik 5
2.1 Fernerkundung . . . 5
2.1.1 Photogrammetrie . . . 6
2.1.2 Laserscanning . . . 10
2.2 Verarbeitung von Punktwolken . . . 12
2.2.1 Referenzierung . . . 12
2.2.2 Segmentierung . . . 13
2.2.3 Klassifizierung . . . 15
2.2.4 Oberflächenrekonstruktion . . . 16
2.3 Building Information Modeling . . . 17
2.3.1 Industry Foundation Classes . . . 18
2.3.2 Scan-to-BIM . . . 19
2.4 Numerische Kollapssimulationen . . . 22
2.4.1 Numerische Verfahren . . . 22
2.4.2 Physik-Engine-basierte Starrkörperdynamik . . . 22
2.5 Zusammenfassung . . . 23

3 Methodik 26
3.1 Strategie der algorithmischen Datenverarbeitung . . . 26
3.2 Datenstruktur . . . 28
3.3 Vorverarbeitung . . . 29
3.4 Segmentierung . . . 31
3.4.1 Ebene Punktmengen . . . 32
3.4.2 Randpunkt-Entfernung . . . 35
3.4.3 Linear angeordnete Punktmengen . . . 40
3.5 Objektklassifizierung . . . 44
3.5.1 Ebene Punktmengen . . . 44
3.5.2 Linear angeordnete Punktmengen . . . 47
3.6 Materialklassifizierung . . . 49
3.6.1 Daten . . . 49
3.6.2 Datenvorverarbeitung . . . 51
3.6.3 Deep-Learning-Modell . . . 51
3.6.4 Training und Testen des Modells . . . 52

3.6.5 Arbeitsablauf der Materialklassifizierung . . . 53
3.7 Digitale 3D-Rekonstruktion der sichtbaren Bauteile . . . 54
3.7.1 Vorverarbeitung . . . 54
3.7.2 Oberflächenrekonstruktion . . . 55
3.7.3 Oberflächenextrusion . . . 56
3.7.4 Transformation . . . 57
3.8 Digitale 3D-Rekonstruktion der verdeckten Bauteile . . . 57
3.8.1 Arbeitsablauf . . . 57
3.8.2 Datenbank . . . 59
3.8.3 Matching-Algorithmus . . . 61
3.8.4 CIM-Algorithmus . . . 63
3.9 BIM-Generierung . . . 65
3.9.1 Mesh-to-IFC . . . 65
3.9.2 Informationsanreicherung . . . 67
3.10 Zusammenfassung . . . 69

4 Evaluierung und Diskussion 72
4.1 Datengenerierung und -vorverarbeitung . . . 72
4.1.1 Fallbeispiel Lyon . . . 73
4.1.2 Fallbeispiel Jindires . . . 82
4.1.3 Fallbeispiel Gölcük . . . 86
4.2 Segmentierung . . . 90
4.2.1 Ebene Punktmengen . . . 90
4.2.2 Randpunkt-Entfernung . . . 94
4.2.3 Linear angeordnete Punktmengen . . . 98
4.3 Objektklassifizierung . . . 101
4.4 Materialklassifizierung . . . 108
4.5 Digitale 3D-Rekonstruktion . . . 112
4.6 BIM-Generierung . . . 117
4.7 Zusammenfassung . . . 119

5 Zusammenfassung und Ausblick 122
5.1 Zusammenfassung der Arbeit . . . 122
5.2 Ausblick . . . 125

Literaturverzeichnis 127

Anhang 140
A.1 Pseudocodes . . . 140
A.1.1 Segmentierung . . . 140
A.1.2 Objektklassifizierung . . . 143
A.1.3 Materialklassifizierung . . . 145
A.1.4 3D-Rekonstruktion . . . 146
A.1.5 BIM-Generierung . . . 151
A.2 Stahlbetonrahmentragwerke . . . 152
A.2.1 Fallbeispiel Jindires . . . 152
A.2.2 Fallbeispiel Gölcük . . . 153

Abbildungsverzeichnis

2.1 Bildliche Beschreibung der aktiven und passiven Fernerkundungsmethode. 5
2.2 Schematische Darstellung der photogrammetrischen Zentralprojektion. 6
2.3 Bildhafte Beschreibung der Merkmalsextraktion im Bildraum aus verschiedenen Kamerapositionen mit der SfM-Methode. 7
2.4 Photogrammetrische Rekonstruktion in *VisualSfM*. 8
2.5 Bildliche Beschreibung der Nadir- und Schrägbildaufnahme. 9
2.6 Luftgestützte LiDAR-Punktwolke. 11
2.7 Beschreibung eines Bauteils gemäß dem IFC-Standard. 19

3.1 Schematische Darstellung der Strategie der algorithmischen Datenverarbeitung zur halbautomatischen Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen. 28
3.2 Datenstruktur der Klasse *BuildingComponent*. 29
3.3 Bildliche Beschreibung der *Spatial-Subsample*-Filterfunktion anhand eines vereinfachten Beispiels. 30
3.4 Unterteilung des Segments S_i (links) in die Segmente S_i und S_{i+1} (rechts) mit der *bitmap*-Methode von *Schnabel et. al* [149]. 33
3.5 Bestimmung der Winkeldifferenz $\alpha_{i,j}$ und des orthogonalen Abstandes $t_{i,j}$ zwischen den Segmenten C_i und C_j. 34
3.6 Strukturelle Komponenten eines Stahlbetonrahmens. 35
3.7 Darstellung der automatisch ausgewählten Punktsegmente (in Blau) eines synthetischen Datensatzes in der implementierten GUI, dessen BC-Objekte per Knopfdruck herausgefiltert werden können. 36
3.8 Ausrichtung der Punktdaten P_i (graue Punkte) in der XY-Ebene. Das Ergebnis wird durch $P_{i,xy}$ (rote Punkte) dargestellt. 37
3.9 Bildliche Beschreibung der *Octree*-Datenstruktur: Unterteilte Regionen (links) und Knotenaufbau der Baumdatenstruktur (rechts). 38
3.10 Gefilterte Punktdaten der Wände (b) aus dem Rahmen (a) mit der RE-Methode. 39
3.11 Schemazeichnung zur Erläuterung der RANSAC-Methode mit einem Linienmodell in $\mathbb{R}^2$ 41
3.12 Bildliche Darstellung der Extraktion der linearen Punktdaten von Trägern und Stützen aus einem Rahmensegment unter Verwendung von RANSAC. 43
3.13 Bildliche Darstellung einer *Bounding Box* einer 3D-Punktwolke. 45
3.14 Bildliche Beschreibung der Bestimmung der minimalen *Bounding Box* einer in der XY-Ebene ausgerichteten Punktwolke. 46
3.15 Grafische Darstellung des Winkels α_i zwischen dem Normalenvektor $\mathbf{n_i}$ der an die Punktwolke P_i angepassten Ebene E_i und dem Normalenvektor $\mathbf{n_{xy}}$ der XY-Ebene. 46
3.16 Klassifizierung der linear angeordneten Punktdaten $P_{1,yz}$ bis $P_{3,yz}$ in die Objektklassen *Column*, *Beam* und *Beam* | *Column* basierend auf ihren Abmessungen und ihrer Form in der YZ-Ebene. 48

3.17 Auszug aus den erstellten Bilddatensätzen: Betonoberflächen [117, 187] (oben) und Mauerwerksoberflächen (unten). 50
3.18 Generierte Pixelgrafiken der dominanten Farben aus den Bilddatensätzen für *Concrete* (links) und *Masonry* (rechts). 51
3.19 Schematische Darstellung des erzeugten DNNs mit fünf Schichten. 52
3.20 Ausgabeplots des Lernprozesses unter Verwendung des erstellten DNNs, die die Modellgenauigkeit (links) und den Modellverlust (rechts) zeigen. 53
3.21 Bildliche Beschreibung der Definition des Schnittbereiches von $P_{i,xy}$ mit Hilfe seiner *Bounding Box*. 55
3.22 Bildliche Beschreibung der Oberflächenrekonstruktion mit der α-*Shape*-Methode am Beispiel einer Punktmenge in der XY-Ebene. 56
3.23 Prinzipskizze zur Beschreibung der Oberflächenextrusion mit der Python-Bibliothetk *Trimesh*. 56
3.24 Schematischer Arbeitsablauf der modellunabhängigen 3D-Rekonstruktion von verdeckten Bauteilen in Form eines Flussdiagramms. 58
3.25 Ausgabe des HPR-Algorithmus (mittig und rechts) am Beispiel eines Schadensmodells (links): Draufsicht (oben) und Perspektivenansicht (unten). 61
3.26 Schrittweise Rotation der Eingabepunktwolke P mit dem Winkel φ_i um die z-Achse zum Vergleich mit der HPR-Punktwolke P_i des Schadensmodells. 62
3.27 Bildliche Beschreibung der *Bounding-Box*-Definition (links) und des Ausschneidens von Flächen innerhalb der *Bounding Box* (rechts). 63
3.28 Positionierung der *Bounding Boxes* entlang der Randpunkte in der XY-Ebene. . . 65
3.29 Schematischer Ablauf der *OBJ2IFC*-Methode [119] unter Verwendung der *IfcOpenShell*-Bibliothek [87]. 67
3.30 Datenstruktur der aus OBJ-Daten erstellten IFC-Objekte. 68

4.1 Drohnenaufnahmen eines teilweise eingestürzten Stahlbetongebäudes in Lyon, Frankreich [97, 124]. 73
4.2 Photogrammetrisch erzeugte Punktwolken des teilweise eingestürzten Stahlbetongebäudes in Lyon mit *Agisoft Metashape* [10] (links) und *VisualSfM* [175] (rechts). 75
4.3 Zusammenfassung der Ergebnisse der photogrammetrischen Rekonstruktion für das teilweise eingestürzte Stahlbetongebäude in Lyon, durchgeführt mit *Agisoft Metashape* [10] und *VisualSfM* [175]. 75
4.4 Abschätzung der Kamerapositionen in *Agisoft Metashape* [10] für die Datensätze #2 (links) und #3 (rechts). 76
4.5 Vergleich zwischen der benötigten Berechnungszeit und der erzielten Genauigkeit zur Erzeugung einer dünnen Punktwolke in *Agisoft Metashape* [10] für die Datensätze #1 bis #3. 77
4.6 Vergleich zwischen der benötigten Berechnungszeit und der erzielten Genauigkeit zur Erzeugung einer dichten Punktwolke in *Agisoft Metashape* [10] für die Datensätze #1 bis #3. 78
4.7 Beziehung zwischen den Standardabweichungen σ und den Mittelwerten Ω der mit *CloudCompare* [40] erhaltenen Ergebnissen aus den Datensätzen #2 und #3 verglichen mit Datensatz #1. 80
4.8 Photogrammetrisch erzeugte 3D-Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Lyon aus dem Datensatz #2. 80
4.9 Definierter Schnittbereich in der XY-Ebene, mit der die relevanten Punktdaten für die Datenverarbeitung des teilweise eingestürzten Stahlbetongebäudes in Lyon ausgeschnitten wurden. 81

4.10 Darstellung der identifizierten Ausreißerpunkte (grau) und Nicht-Ausreißerpunkte (blau) der Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Lyon. . 82
4.11 Vorverarbeitete Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Lyon. 82
4.12 Bilder des teilweise eingestürzten Stahlbetongebäudes in Jindires (oben, entnommen aus [129, 173]) und gerenderte Bilder des nachgebildeten 3D-Modells in *Blender* [23] (unten). 83
4.13 Erzeugte Kameras in *Blender* [23] zur photogrammetrischen Aufnahme des teilweise eingestürzten Stahlbetonrahmentragwerkes in Jindires. 84
4.14 Photogrammetrisch erzeugte 3D-Punktwolke der teilweise eingestürzten Stahlbetonrahmenkonstruktion in Jindires. 84
4.15 Definierte Schnittbereiche in der XY- und XZ-Ebene, mit der die relevanten Punktdaten des Stahlbetongebäudes in Jindires für die Datenverarbeitung ausgeschnitten wurden. 85
4.16 Darstellung der identifizierten Ausreißerpunkte (grau) und Nicht-Ausreißerpunkte (blau) der Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Jindires. 85
4.17 Vorverarbeitete Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Jindires. 86
4.18 Bilder des seismisch beschädigten Stahlbetongebäudes in Gölcük (oben, entnommen aus [43]) und gerenderte Bilder des nachgebildeten 3D-Modells in *Blender* [23] (unten). 86
4.19 Erzeugte Kameras in *Blender* [23] zur photogrammetrischen Aufnahme des teilweise eingestürzten Stahlbetonrahmentragwerkes in Gölcük. 87
4.20 Photogrammetrisch erzeugte 3D-Punktwolke des seismisch beschädigten Stahlbetongebäudes in Gölcük. 88
4.21 Definierter Schnittbereich in der XY-Ebene, mit der die relevanten Punktdaten des Stahlbetongebäudes in Gölcük für die Datenverarbeitung ausgeschnitten wurden. 88
4.22 Darstellung der identifizierten Ausreißerpunkte (grau) und Nicht-Ausreißerpunkte (blau) der Punktwolke des seismisch beschädigten Stahlbetongebäudes in Gölcük. 89
4.23 Vorverarbeitete Punktwolke des seismisch beschädigten Stahlbetongebäudes in Gölcük. 89
4.24 Unterteilung der vorverarbeiteten LYN-Punktwolke in intakte (grün) und eingestürzte Bereiche (rot). 91
4.25 Ergebnisse der ebenen Punktsegmentierung für die LYN-Punktwolke nach den Durchläufen #1 und #2, wobei jede Farbe ein identifiziertes Segment repräsentiert. 92
4.26 Störbereich in der LYN-Punktwolke für die ebene Punktsegmentierung im Durchlauf #1. 92
4.27 Ergebnisse der ebenen Punktsegmentierung für die JDS-Punktwolke nach den Durchläufen #1 und #2, wobei jede Farbe ein identifiziertes Segment repräsentiert. 93
4.28 Ergebnisse der ebenen Punktsegmentierung für die GLC-Punktwolke nach den Durchläufen #1 und #2, wobei jede Farbe ein identifiziertes Segment repräsentiert. 94
4.29 Darstellung der *Ground-Truth*-Segmentierung und der RE-Ergebnisse für die Rahmensegmente mit Wänden aus der LYN-Punktwolke. 95
4.30 Darstellung der *Ground-Truth*-Segmentierung und der RE-Ergebnisse für die Rahmensegmente mit Wänden aus der JDS-Punktwolke. Die Farbe Lila steht für die verbleibenden Punktdaten des Rahmensegments und die Farbe Grün für die ausgeschnittenen Wände. 96

4.31 Darstellung der *Ground-Truth*-Segmentierung und der RE-Ergebnisse für die Rahmensegmente mit Wänden aus der GLC-Punktwolke. Die Farbe Lila steht für die verbleibenden Punktdaten des Rahmensegments und die Farbe Grün für die ausgeschnittenen Wände. 97
4.32 Ergebnisse der linearen Punktsegmentierung für die Rahmensegmente der LYN-, JDS- und GLC-Punktwolke. Jede Farbe repräsentiert ein identifiziertes Liniensegment. 100
4.33 Fehlerhafte Segmentierung der linearen Punktdaten aus den Rahmensegmenten der LYN-, JDS- und GLC-Punktwolke. 101
4.34 Darstellung der Ergebnisse der Objektklassifizierung für die LYN-Punktwolke. . . 103
4.35 Darstellung der Ergebnisse der Objektklassifizierung für die JDS-Punktwolke. . . 103
4.36 Darstellung der Ergebnisse der Objektklassifizierung für die GLC-Punktwolke. . . 104
4.37 Veranschaulichung der zugewiesenen Objektklasse *Unclassified* für ein Stützsegment der LYN-Punktwolke. 105
4.38 Darstellung der Gesamtgenauigkeit Ω für die identifizierten Objektklassen aus der LYN-, JDS- und GLC-Punktwolke in Form eines Balkendiagramms. 107
4.39 Darstellung der Ergebnisse der Materialklassifizierung für die LYN-Punktwolke. . 108
4.40 Darstellung der Ergebnisse der Materialklassifizierung für die JDS-Punktwolke. . 109
4.41 Darstellung der Ergebnisse der Materialklassifizierung für die GLC-Punktwolke. . 109
4.42 Darstellung einer richtig positiven und einer falsch positiven Materialklassifizierung von Wandsegmenten aus der JDS-Punktwolke mit ihrer vorherrschenden RGB-Farbe. 111
4.43 Ergebnisse der 3D-Rekonstruktion für die Objekte der LYN-Punktwolke unter Verwendung von $\alpha = 1,0$ und $\alpha = 0,5$. 113
4.44 Ergebnisse der 3D-Rekonstruktion für die Objekte der JDS-Punktwolke unter Verwendung von $\alpha = 1,0$ und $\alpha = 0,5$. 114
4.45 Ergebnisse der 3D-Rekonstruktion für die Objekte der GLC-Punktwolke unter Verwendung von $\alpha = 1,0$ und $\alpha = 0,5$. 114
4.46 Ergebnisse der 3D-Rekonstruktion für die Objekte der LYN-, JDS- und GLC-Punktwolke unter Verwendung von optimalen α-Werten. 116
4.47 Ergebnisse der *Mesh-to-IFC*-Methode für die Objekte der LYN-, JDS- und GLC-Punktwolke. 118

A.1 Grundrissplan eines Stahlbetonrahmentragwerkes in Jindires, Syrien, mit fiktiven Maßangaben. 152
A.2 Grundrissplan eines Stahlbetonrahmentragwerkes in Gölcük, Turkei, (Maßangaben entnommen aus [43]). 153

Tabellenverzeichnis

3.1 Vordefinierte Objektklassen mit Angabe der dazugehörigen *type*- und RGB-Daten im BC-Objekt. 44
3.2 Struktur der gekennzeichneten Daten unter Verwendung einer *One-Hot*-Kodierung. 50
3.3 Kamerapositionen für den HPR-Algorithmus. 60
3.4 Definition von IFC-Klassen in Abhängigkeit von der Objektklasse aus den BC-Objekten. 66

4.1 Versuchsdetails zur photogrammetrischen Aufnahme des teilweise eingestürzten Stahlbetongebäudes in Lyon, Frankreich [97, 124]. 74
4.2 Systemspezifikationen der *Graphic Workstation* und des Standardcomputers. . . . 74
4.3 Mittlerer Abstand Ω und Standardabweichung σ von Datensatz #1 im Vergleich zu den Datensätzen #2 und #3. 79
4.4 Beschreibung der Parameter in der Methode zur Segmentierung von ebenen Punktmengen (Algorithmus 1). 90
4.5 Gewählte Parameterwerte für die ebene Punktsegmentierung der vorverarbeiteten 3D-Punktwolken der Datensätze LYN, JDS und GLC. 91
4.6 Beschreibung der Parameter in der RE-Methode (Algorithmus 2). 94
4.7 Gewählte Parameterwerte für die Anwendung der RE-Methode auf die Rahmensegmente mit Wänden aus den Datensätzen LYN, JDS und GLC. 95
4.8 Beschreibung der Parameter in der Methode zur Segmentierung von linienförmigen Punktmengen (Algorithmus 3). 98
4.9 Gewählte Parameterwerte für die lineare Punktsegmentierung der Rahmensegmente der 3D-Punktwolken der Datensätze LYN, JDS und GLC. 98
4.10 Beschreibung und Definition der geometrischen Parameter für die Objektklassifizierung (Algorithmus 4 und 5). 102
4.11 Ergebnisse der Objektklassifizierung für die LYN-Punktwolke. 106
4.12 Ergebnisse der Objektklassifizierung für die JDS-Punktwolke. 106
4.13 Ergebnisse der Objektklassifizierung für die GLC-Punktwolke. 106
4.14 Ergebnisse der Materialklassifizierung für die LYN-Punktwolke. 110
4.15 Ergebnisse der Materialklassifizierung für die JDS-Punktwolke. 110
4.16 Ergebnisse der Materialklassifizierung für die GLC-Punktwolke. 110

Abkürzungsverzeichnis

AEM	Angewandte-Elemente-Methode
ALS	*Airborne Laserscanning*
API	*Application Programming Interface*
BC	*BuildingComponent*
BCB	*Bullet-Constraint-Builder*
BIM	Building Information Modeling
BPA	*Ball-Pivoting-Algorithmus*
B-rep	*Boundary Representation*
CAD	*Computer-Aided-Design*
C2C	*Cloud-to-Cloud*
CIM	*Crop-Interior-Mesh*
CMVS	*Clustering-View-for-Multi-View-Stereo*
CNN	*Convolutional Neural Network*
DIM	*Damaged Interior Model*
DL	*Deep Learning*
DNN	*Deep Neural Network*
FE	Finite-Elemente
FEM	Finite-Elemente-Methode
FP	*False Positive*
GLC	Gölcük
GNSS	*Global Navigation Satellite System*
GOK	Geländeoberkante
GPS	*Global Positioning System*
GUI	*Graphical User Interface*
GUID	*Globally Unique Identifier*
HPR	*Hidden-Point-Removal*
ICP	*Iterative-Closest-Point*
IFC	*Industry Foundation Classes*
IMU	*Inertial Measurement Unit*
JDS	Jindires
KI	Künstliche Intelligenz
LiDAR	*Light Detection and Ranging*
LoD	*Level of Development*
LGPL	*Lesser General Public License*
LYN	Lyon
ML	*Machine Learning*
NURBS	*Non-Uniform Rational B-splines*

PMVS	*Patch-based-Multi-View-Stereo*
RANSAC	*Random-Sample-Consensus*
RE	Randpunkt-Entfernung
ReLU	*Rectified Linear Unit*
RTK	*Real-Time-Kinetic*
SfM	*Structure-from-Motion*
SIFT	*Scale-Invariant-Feature-Transform*
SOR	*Statistical-Outlier-Removal*
TP	*True Positive*
TLS	*Terrestrial Laser Scanning*
USaR	*Urban Search and Rescue*

Kapitel 1

Einleitung

1.1 Motivation

Erdbeben gehören weltweit zu den häufigsten Ursachen für den vollständigen oder teilweisen Einsturz von Gebäuden [102] und fordern jedes Jahr zahlreiche Menschenleben [7]. Die verheerenden Erdbebenereignisse an der türkisch-syrischen Grenze [164], in Marokko [163] und in Afghanistan [162] im Jahr 2023 resultierten in einem tragischen Verlust von über 60.000 Menschenleben. Diese Ereignisse verdeutlichen die katastrophalen Auswirkungen, die ein starkes Erdbeben in städtischen Gebieten haben kann. Daten der *International Disaster Database* zeigen, dass allein in den letzten 50 Jahren mehr als 1,2 Millionen Menschen durch Erdbeben ums Leben gekommen sind [7]. Die überwiegende Mehrheit der erdbebenbedingten Todesopfer ist auf Gebäudeeinstürze zurückzuführen [41], wobei sich die meisten dieser Vorfälle in Entwicklungsländern ereignen [96]. Einer der Gründe dafür ist, dass Gebäude in Entwicklungsländern meist nicht erdbebensicher gebaut sind und für den Bau der Gebäude häufig minderwertige Materialien verwendet werden [2, 3]. Unter diesem Gesichtspunkt ist auch in Zukunft mit dem Einsturz von Gebäuden aufgrund von Erdbeben zu rechnen. Überlebende in eingestürzten Gebäuden sind in der Regel in Hohlräumen eingeschlossen, umgeben von schweren Trümmern. Da jede Minute für das Überleben der Verschütteten entscheidend sein kann, beginnen die Ersthelfer unmittelbar nach dem Katastrophenereignis mit Such- und Rettungsmaßnahmen. Die Suche nach Überlebenden in den Trümmern stellt für die Einsatzkräfte sowohl wegen der unsicheren Stabilität der Trümmer als auch wegen des Zeitdrucks eine große Herausforderung dar. In solchen Situationen ist ein weiterer Einsturz jederzeit möglich, insbesondere wenn statisch-relevante Trümmerteile entfernt werden. Dies würde nicht nur das Leben der Verschütteten, sondern auch das der Rettungskräfte gefährden [138, 139].
Mehrere Beiträge, wie z. B. [69], [70] und [71], befassen sich mit der Bewertung von Risiken und dem Schutz baulicher Infrastrukturen vor Naturgefahren. Es besteht jedoch ebenfalls die Notwendigkeit, Entscheidungsfindungsprozesse für eingestürzte Gebäude, d. h. für Ereignisse nach Katastrophen, mit modernen Methoden zu erforschen und weiterzuentwickeln. Für Gebäudeeinstürze gibt es in Deutschland im Wesentlichen zwei Richtlinien [169] und [83], die seit 2002 und 2008 den Stand der Technik darstellen. Die Beschaffung von Informationen über das eingestürzte Gebäude ist ein wesentlicher Bestandteil der in den Richtlinien genannten Verfahren zur Beurteilung von Gebäudeschäden. Dies erweist sich jedoch in einer Katastrophensituation als äußerst schwierig und gefährlich, vor allem wenn keine digitalen Modelle, Baupläne oder Unterlagen der betroffenen Gebäude vorliegen. Der Einsatz moderner Methoden oder Technologien zur Erfassung von relevanten Daten über den Katastrophenort wird in den genannten Richtlinien [169] und [83] nicht berücksichtigt. Aus diesem Grund werden beispielsweise Abmessungen oder Volumina von Trümmern bislang mit Hilfe von vereinfachten Formeln abgeschätzt oder zeitaufwändig mit

analogen Messinstrumenten, wie z. B. einem Tachymeter, ermittelt. Diese Vorgehensweise kann zu einer unsicheren und subjektiven Beurteilung der Gefährdungslage vor Ort führen [138, 139]. Durch den Einsatz moderner Messtechniken wie der Photogrammetrie oder dem Laserscanning ist es möglich, ein eingestürztes Gebäude aus der Ferne sicher zu erfassen und in Form einer 3D-Punktwolke präzise zu digitalisieren [139]. Eine 3D-Punktwolke stellt die Oberflächen eines physischen Objektes in Form von Punkten dar, wobei die Darstellungsgenauigkeit von der Punktdichte abhängt. Einsatzkräfte können die 3D-Punktwolke eines eingestürzten Gebäudes nutzen, um sich einen räumlichen Überblick über das Ausmaß der Zerstörung zu verschaffen. Darüber hinaus können Abstände zwischen ausgewählten Punkten manuell aus der Punktwolke bestimmt werden. Da eine Punktwolke jedoch nur aus 3D-Punktkoordinaten $P_i(x_i,y_i,z_i)$ mit $i \in \mathbb{N}$ besteht, die in der Regel unregelmäßig und ungeordnet im Raum liegen, stehen dem Nutzer keine geometrischen oder semantischen Informationen über die erfassten Objekte zur Verfügung. Die BIM-Technologie bietet die notwendige Grundlage, um aus einem digitalen 3D-Modell ein semantisches Bauwerksdatenmodell zu erzeugen. BIM stellt eine fortschrittliche Methode im Bauwesen dar und wird im Allgemeinen für die kosteneffiziente und leistungsorientierte Planung und Ausführung von Bauprojekten eingesetzt. Durch die Verwendung dieser modernen Methode könnten die Einsatzkräfte von den folgenden BIM-Merkmalen profitieren:

- Zugriff auf das digitale 3D-Modell mit der Möglichkeit, gesammelte Daten direkt im Modell zu speichern und zu verwalten.
- Automatische Bestimmung der Geometriedaten und der Massen der Bauteile, wobei für letztere eine Wichte angegeben werden muss.
- Eindeutige Kennzeichnung von Bauteilen mittels *Globally Unique Identifiers* (GUIDs).
- Objektive Abschätzung der Größe und Lage von Hohlräumen in den verdeckten Bereichen.
- Ableitung von Grundriss- oder Schnittplänen aus dem 3D-Modell.
- Einheitlicher Informationsaustausch durch offene Standards.
- Kontinuierliche Anpassung der Daten durch den Nutzer [139].

In den letzten Jahren haben Forscher die Anwendung der BIM-Technologie speziell für die Katastrophenvorsorge und -bewältigung unter verschiedenen Szenarien untersucht und die derzeitigen Möglichkeiten aufgezeigt [98]. Erstmals wurde von *Zeibak et al.* [182] ein modellabhängiger Ansatz vorgestellt, mit dem sogenannte *as-damaged* BIM-Modelle von seismisch beschädigten Stahlbetonrahmentragwerke mit Mauerwerksausfachung automatisch generiert werden können [139]. Zur Anwendung dieses Ansatzes werden das *as-built* BIM-Modell des Bauwerkes, d. h. das erzeugte BIM-Modell des Gebäudes nach Abschluss der Bauphase, und eine 3D-Punktwolke der beschädigten Konstruktion als Eingabe benötigt. Mit dem von den Autoren vorgeschlagenen Ansatz wird die Gebäudehülle des *as-built* BIM-Modells, das als Fassadenmodell bezeichnet wird, an die deformierte Geometrie der Punktwolke angepasst. In diesem Zusammenhang ist für die Anwendung der Methode von *Zeibak et al.* [182] das Vorhandensein eines *as-built* BIM-Modells zwingend erforderlich. Da jedoch für die meisten Gebäude weltweit keine BIM-Modelle verfügbar sind, ist die praktische Anwendbarkeit dieser Methode stark eingeschränkt [139]. Nach dem Kenntnisstand des Autors dieser Arbeit existiert bislang kein modellunabhängiger Ansatz zur Erstellung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen. Ein modellunabhängiger Ansatz bedeutet in diesem Kontext, dass allein die erzeugte 3D-Punktwolke des beschädigten Tragwerkes als Grundlage für die Erstellung des BIM-Modells verwendet wird. Dies macht den Ansatz in realen Situationen wesentlich praktikabler im Vergleich zu modellabhängigen Verfahren.

1.2 Forschungsfragen und Herausforderungen

1.2.1 Forschungsfragen

Die vorliegende Arbeit befasst sich mit der Entwicklung eines modellunabhängigen Ansatzes zur Erstellung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen. Stahlbetonrahmentragwerke sind in vielen Teilen der Welt, einschließlich seismisch aktiver Regionen, die beliebteste Bauart für mehrgeschossige Gebäude [79]. Darüber hinaus sind Erdbeben weltweit die häufigste Ursache für schwere Gebäudeschäden oder Einstürze [102]. Folglich wird in dieser Forschungsarbeit das betrachtete Katastrophenszenario auf seismisch beschädigte Stahlbetonrahmenkonstruktionen beschränkt.

Basierend auf der vorgestellten Einführung in Abschnitt 1.1 werden vier Forschungsfragen formuliert:

F1: Wie können Bauteile einer seismisch beschädigten Stahlbetonrahmenkonstruktion anhand einer 3D-Punktwolke zuverlässig und modellunabhängig digital rekonstruiert werden?

F2: Bis zu welchem Grad können die Arbeitsschritte der digitalen Rekonstruktion automatisiert werden?

F3: Wie kann das rekonstruierte 3D-Modell in den BIM-Prozess übertragen werden, wenn keine BIM-Grundlage vorhanden ist?

F4: Welche Möglichkeiten gibt es, strukturrelevante Informationen auf Basis einer 3D-Punktwolke zu erhalten?

Zur Beantwortung der oben genannten Forschungsfragen **F1** bis **F4** werden in dieser Arbeit neue digitale Methoden entwickelt und evaluiert, die eine einfache und schnelle Erstellung eines BIM-Modells einer seismisch beschädigten Stahlbetonrahmenkonstruktion auf der Grundlage einer 3D-Punktwolke ermöglichen sollen. Es ist wichtig zu beachten, dass sich die Forschungsfrage **F1** auf die in der 3D-Punktwolke dargestellten Bauteile bezieht.

1.2.2 Herausforderungen

Die modellunabhängige Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen erfordert den Einsatz moderner Sensortechnologien und digitaler Datenverarbeitungsmethoden. Die Forschung in diesem Bereich ist jedoch mit mehreren Herausforderungen verbunden. Zu diesen Herausforderungen gehört der Umgang mit Datenlücken in der 3D-Punktwolke aufgrund von Verdeckungen (engl. *occlusions*). Mit Hilfe von optischen Messmethoden, wie z. B. der Photogrammetrie oder das Laserscanning, können nur Daten von reflektierenden Oberflächen erfasst werden, die sich innerhalb des Sichtfeldes des Sensors befinden. Verdeckungen können durch verschiedene Faktoren während der optischen Datenerfassung verursacht werden, z. B. durch Bäume, Autos, Container, Menschen oder Gebäude. In unserem Fall muss davon ausgegangen werden, dass aufgrund der Verdeckung durch die Fassade eine Datenerfassung des Innenraums einer beschädigten Stahlbetonrahmenkonstruktion von außen nicht möglich ist.
Um ein effizientes und objektives Verfahren zur digitalen Rekonstruktion eines seismisch beschädigten Stahlbetonrahmens für *Urban Search and Rescue*-(USaR)-Zwecke zu entwickeln, müssen die dafür benötigten Arbeitsschritte so weit wie möglich automatisiert werden. Aus der Komplexität der Forschungsfragen in dieser Arbeit wird deutlich, dass eine automatische Modellrekonstruktion auf der Grundlage einer 3D-Punktwolke, die nur die Punktdaten der äußerlich

sichtbaren Bauteile enthält, eine besondere Herausforderung darstellt.
Eine weitere Herausforderung dieser Arbeit besteht darin, das erstellte BIM-Modell der beschädigten Konstruktion ausschließlich auf Basis einer 3D-Punktwolke mit semantischen Informationen anzureichern. Obwohl das generierte BIM-Modell in Katastrophensituationen in erster Linie für die digitale Verwaltung der gesammelten Daten durch Einsatzkräfte genutzt werden kann, werden auch Ansätze untersucht, die eine objektive und objektorientierte Informationsanreicherung der Bauteile auf der Grundlage von Punktdaten ermöglichen könnten.

1.3 Aufbau der Arbeit

In Kapitel 2 dieser Dissertation wird ein umfassender Überblick über den aktuellen Stand der Wissenschaft und Technik in den zentralen Forschungsfeldern präsentiert. Die thematischen Schwerpunkte umfassen die Bereiche Fernerkundung mittels Photogrammetrie und Laserscanning (Abschnitt 2.1), Methoden zur Verarbeitung von Punktwolken (Abschnitt 2.2), BIM (Abschnitt 2.3) sowie numerische Kollapssimulationen (Abschnitt 2.4).
In Kapitel 3 wird die entwickelte modellunabhängige Methodik zur halbautomatischen Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmentragwerken detailliert dargestellt. Zu Beginn dieses Kapitels wird in den Abschnitten 3.1 und 3.2 die algorithmische Datenverarbeitungsstrategie beschrieben und die zugrunde liegende Datenstruktur für die Programmierung vorgestellt. Im Anschluss erfolgt eine detaillierte Darstellung der einzelnen Arbeitsschritte und der zugehörigen Methoden der algorithmischen Datenverarbeitung. Diese umfassen die Vorverarbeitung (Abschnitt 3.3), die Segmentierung (Abschnitt 3.4), die Objekt- und Materialklassifizierung (Abschnitt 3.5 und Abschnitt 3.6), die digitale 3D-Rekonstruktion (Abschnitt 3.7 und Abschnitt 3.8) sowie die BIM-Generierung (Abschnitt 3.9).
Kapitel 4 ist der Evaluierung der entwickelten Methodik und der Diskussion der Ergebnisse gewidmet. In Abschnitt 4.1 wird zunächst die Datengenerierung anhand von Fallbeispielen vorgestellt. Anschließend werden die entsprechenden Datensätze vorverarbeitet. Die Evaluierung der einzelnen Arbeitsschritte der algorithmischen Datenverarbeitung erfolgt anhand der bereitgestellten Datensätze aus den Fallbeispielen. In diesem Zusammenhang werden sowohl visuelle als auch quantitative Bewertungsmethoden angewendet, um die Leistungsfähigkeit und die Anwendungsgrenzen der entwickelten Ansätze zu bestimmen. Die Kapitel 2, 3 und 4 werden jeweils mit einer Zusammenfassung der wesentlichen Ergebnisse abgeschlossen.
Abschließend werden in Kapitel 5 die Ergebnisse der durchgeführten Forschung zusammengefasst und ein Ausblick auf potenzielle Weiterentwicklungen und zukünftige Forschungsrichtungen gegeben.

Kapitel 2

Stand der Wissenschaft und Technik

In den letzten Jahren hat die Nutzung optischer Messverfahren aus der Fernerkundung für die digitale Rekonstruktion bestehender Bauwerke zunehmend an Bedeutung gewonnen und neue Möglichkeiten der Digitalisierung im Bauwesen eröffnet. Dieses Kapitel fasst den Stand der Wissenschaft und Technik in Bezug auf moderne Fernerkundungstechniken zur Erzeugung dreidimensionaler Punktwolken (Abschnitt 2.1), Methoden zur Verarbeitung von Punktwolken (Abschnitt 2.2), BIM (Abschnitt 2.3) und numerische Kollapssimulationen (Abschnitt 2.4) zusammen und zeigt damit die aktuellen Möglichkeiten und Anwendungsgrenzen der für diese Dissertation relevanten digitalen Methoden auf.

2.1 Fernerkundung

Fernerkundung ist ein Oberbegriff, der die Gesamtheit aller Messmethoden zur berührungslosen Informations- und Datenerfassung von physischen Objekten aus der Ferne beschreibt. Die entsprechenden Messinstrumente werden entweder als aktive oder passive Fernerkundungssysteme eingestuft. Die jeweiligen Definitionen lauten wie folgt:

- **Aktive Fernerkundungssysteme:** Als aktive Fernerkundungssysteme werden Systeme bezeichnet, die die Strahlung zur Registrierung und Detektion von Objekten künstlich erzeugen und das reflektierte Signal wieder empfangen können. Das Laserscanning ist demnach eine aktive Fernerkundungsmethode.
- **Passive Fernerkundungssysteme:** Systeme, die ausschließlich die Sonnenstrahlung oder künstlich erzeugte Strahlung nutzen, werden als passive Fernerkundungssysteme bezeichnet, wie z. B. die Photogrammetrie.

Der prinzipielle Unterschied zwischen diesen Verfahren ist in der Abbildung 2.1 illustriert.

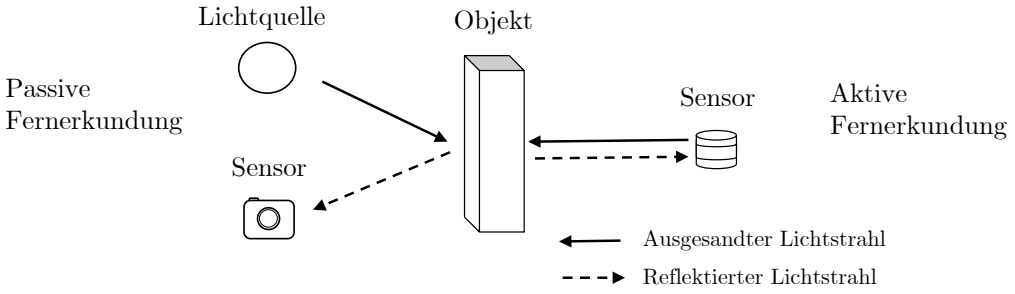


Abbildung 2.1: Bildliche Beschreibung der aktiven und passiven Fernerkundungsmethode.

Die moderne Fernerkundung von Objekten oder Umgebungen führt in der Regel zur Erzeugung von 3D-Punktwolken, die die Oberflächengeometrie der aufgenommenen Szene als Punktmenge $P_i(x_i, y_i, z_i)$ mit $i \in \mathbb{N}$ im 3D-Raum darstellen. Sowohl die aktiven als auch die passiven Fernerkundungsmethoden weisen Vor- und Nachteile auf. Aus diesem Grund werden diese Messverfahren oft in Kombination eingesetzt, um die entsprechenden Anwendungsgrenzen der jeweiligen Messmethode mit Hilfe eines anderen Verfahrens auszugleichen. In den folgenden Abschnitten werden grundlegende Informationen über die Photogrammetrie und das Laserscanning kurz zusammengefasst. Für eine umfassende Darstellung der theoretischen und praktischen Grundlagen der Photogrammetrie und des Laserscannings wird auf die Werke von *Luhmann et al.* [111] und *Heipke et al.* [80] verwiesen.

2.1.1 Photogrammetrie

Die Photogrammetrie ist eine etablierte, passive Fernerkundungsmethode, die seit Mitte des 19. Jahrhunderts kontinuierlich weiterentwickelt wurde. Diese moderne Messtechnik ermöglicht die digitale Rekonstruktion von physischen Objekten in Form einer 3D-Punktwolke aus zweidimensionalen Bildern. Aufgrund der hohen Genauigkeit, der einfachen Anwendung und der geringen Kosten, die mit der photogrammetrischen Analyse verbunden sind, ist diese Technologie in den Bereichen Kartografie, Architektur, Ingenieurwesen, Archäologie und Computergrafik weit verbreitet [80].
Bei der photogrammetrischen Auswertung werden aus den Bilddaten die Position und die Form des erfassten Objektes mit den Grundsätzen der Zentralprojektion bestimmt. In der Abbildung 2.2 ist die photogrammetrische Zentralprojektion schematisch dargestellt. Die geradlinig laufenden Bildstrahlen kreuzen sich im Projektionszentrum **O**, das zwischen der Sensorebene und dem 3D-Objekt liegt. Für die Position des Projektionszentrums **O** kann näherungsweise der Mittelpunkt **M** der bildseitigen optischen Sensorebene angenommen werden. Das photogrammetrische Projektionszentrum **O** liegt jedoch außerhalb der Sensorebene, weshalb die Bildszene spiegelverkehrt in den Bildraum projiziert wird [80].

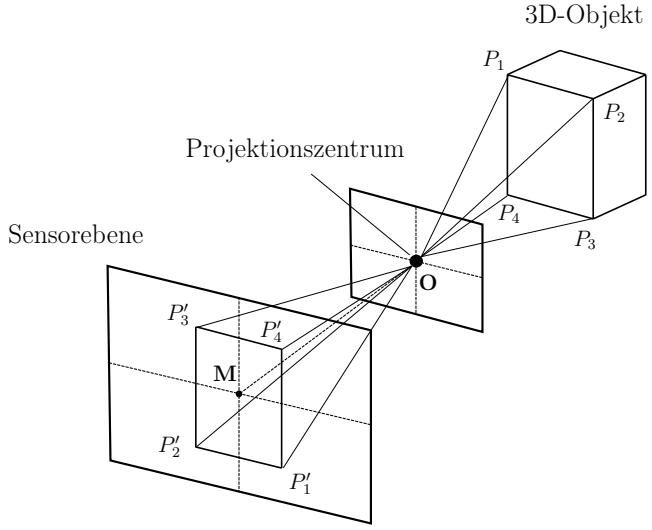


Abbildung 2.2: Schematische Darstellung der photogrammetrischen Zentralprojektion.

Die geometrische Beziehung zwischen dem realen Objekt und den aufgenommenen Bildern des Objektes wird durch die Parameter der inneren und äußeren Orientierung beschrieben. Dabei stellt die innere Orientierung die Lage des Projektionszentrums **O** im Bildkoordinatensystem

dar, während die äußere Orientierung die räumliche Position und Ausrichtung der Kamera zum Zeitpunkt der Aufnahme definiert. Für die konventionelle Photogrammetrie müssen die Parameterwerte der inneren und äußeren Orientierung bekannt sein, um eine photogrammetrische Auswertung durchführen zu können. Die *Structure-from-Motion*-(SfM)-Technologie hingegen ermöglicht die automatische Bestimmung der inneren und äußeren Orientierung aus überlappenden Bildblöcken mittels eines hochredundanten und iterativen Optimierungsverfahrens, das als Bündelblockausgleichung bekannt ist [111, 156]. Die Bündelblockausgleichung kann sowohl mit relativer als auch mit absoluter Orientierung durchgeführt werden. Bei der relativen Orientierung ergeben sich die Parameterwerte als Produkt der Merkmalsextraktion (engl. *feature extraction*), bei dem auffällige und übereinstimmende Verknüpfungspunkte (engl. *keypoints*) aus den Bildern extrahiert werden. Eine höhere Genauigkeit kann mit der absoluten Orientierung erzielt werden, bei der die bereits bekannten Parameterwerte aus der relativen Orientierung mit Hilfe von manuell platzierten Passpunkten korrigiert werden.
Zur Detektion und Beschreibung von Verknüpfungspunkten in Bildern wird üblicherweise der *Scale-Invariant-Feature-Transform*-(SIFT)-Algorithmus von *Lowe* [108] verwendet. Diese Methode nutzt einen Detektor und einen Deskriptor, die im Stande sind, Verknüpfungspunkte aus Bildern automatisch zu extrahieren und sie im Bildraum unter Berücksichtigung der Variation der Bildposition, des Maßstabes und der Ausrichtung zu beschreiben. Für eine eindeutige digitale Rekonstruktion im 3D-Bildraum benötigt die SfM-Photogrammetrie daher mindestens zwei Bilder, die markante Objektpunkte aus unterschiedlichen Kamerapositionen darstellen (Abbildung 2.3).

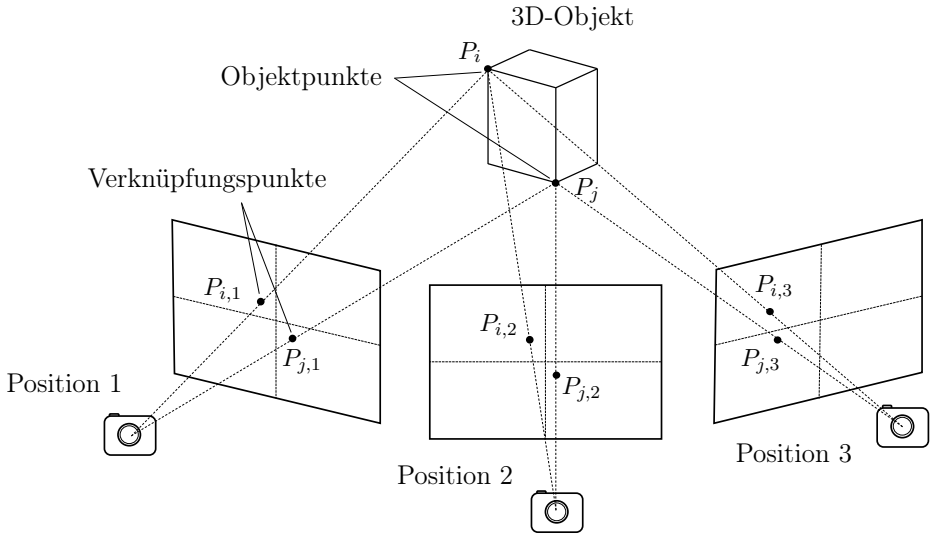


Abbildung 2.3: Bildhafte Beschreibung der Merkmalsextraktion im Bildraum aus verschiedenen Kamerapositionen mit der SfM-Methode.

Das Ergebnis der Merkmalsextraktion und der Bündelblockausgleichung wird als dünne Punktwolke bezeichnet, die die identifizierten Verknüpfungspunkte im 3D-Raum sowie die Position und Ausrichtung der Kameras enthält. Auf der Grundlage der Informationen aus der dünnen Punktwolke wird schließlich die dichte Punktwolke der aufgenommenen Szene erstellt. Die *Open-Source*-SfM-Photogrammetrie-Software *VisualSfM* [175] verwendet zu diesem Zweck die Algorithmen *Clustering-View-for-Multi-View-Stereo* (CMVS) [67, 68] und *Patch-based-Multi-View-Stereo* (PMVS) [67]. Die Abbildung 2.4 zeigt den Arbeitsablauf der SfM-Technik in *VisualSfM* [175] und die resultierenden Ergebnisse anhand eines Beispieldatensatzes.

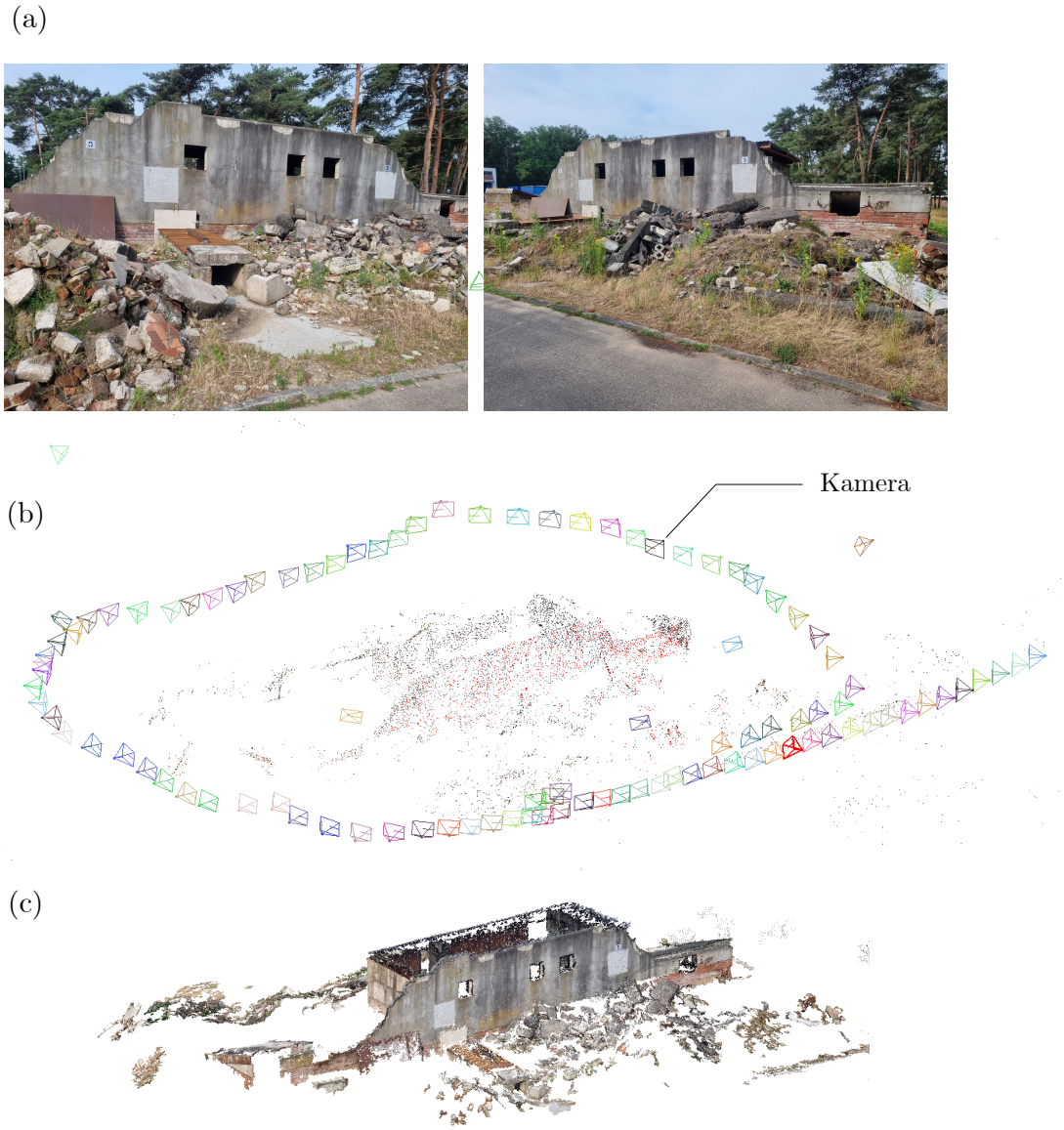


Abbildung 2.4: Photogrammetrische Rekonstruktion in *VisualSfM* [175]: (a) Eingabedaten bestehend aus 106 terrestrischen Bildern der Szene mit seitlicher Bildüberlappung. (b) Dünne Punktwolke mit Lage und Ausrichtung der Kamaras, die mit dem SIFT-Algorithmus [108] und der Bündelblockausgleichung erzeugt wurde. (c) Dichte Punktwolke, die mit den Algorithmen CMVS [67, 68] und PMVS [67] erzeugt wurde.

Die Genauigkeit der photogrammetrischen Rekonstruktion hängt neben der Bildauflösung im Wesentlichen von der Vollständigkeit und der Anzahl der von der Szene aufgenommenen Bilder ab. Um sicherzustellen, dass die mit dem SIFT-Algorithmus extrahierten Verknüpfungspunkte aus den Bildern übereinstimmen, muss eine Bildüberlappung von mindestens 30% gewährleistet sein [168]. Heutzutage kann jede hochauflösende Digital- und Smartphone-Kamera für photogrammetrische Zwecke eingesetzt werden, was die Anwendungsanforderungen erheblich vereinfacht. *Fisheye*-Objektive, wie sie z. B. in der *GoPro*-Kamera verwendet werden, sind für die Photogrammetrie nicht geeignet, da die extremen Verzerrungen im Bildraum die tatsächliche Objektgeometrie stark verfälschen.

In den letzten Jahren haben luftgestützte Systeme, wie z. B. kleine unbemannte Drohnen für den Zivilgebrauch, neue Möglichkeiten für die Fernerkundung geschaffen. Ein Überblick über die Entwicklungen und Anwendungen von unbemannten Drohnen im Bauwesen wurde in [33] veröffentlicht. Drohnen sind flexibel einsetzbar, können mit unterschiedlichen Sensoren ausgestattet werden und lassen sich leicht fernsteuern. Eine Fernerkundung aus der Luft ermöglicht Einblicke in Gebiete, die mit bodengestützten Plattformen nicht zugänglich oder sichtbar wären. Außerdem können die photogrammetrischen Daten sicher aus der Luft erfasst werden, da der Benutzer sich dem zu erfassenden Objekt nicht nähern muss. Die drohnengestützte Photogrammetrie bietet daher optimale Voraussetzungen für eine sichere, großflächige Erfassung von Katastrophengebieten. Der Einsatz von Drohnen für die digitale Aufnahme von Katastrophenszenarien wurde bereits in [9, 136] diskutiert. Darüber hinaus haben mehrere Forscher Drohnenkameras und RGB-Tiefensensoren eingesetzt, um dichte 3D-Punktwolken von Katastrophenszenen für USaR-Maßnahmen zu erstellen [59, 60, 89, 167, 168, 180]. *Yamazaki et al.* [180] haben anhand einer Fallstudie gezeigt, dass auch hochauflösende Drohnenvideos gezielt für die photogrammetrische Rekonstruktion von Katastrophengebieten eingesetzt werden können. In der Literaturübersicht von *Kerle et al.* [97] wird der aktuelle Stand der Technik zur drohnenbasierten Erzeugung von 3D-Punktwolken von beschädigten Gebäuden nach einem Katastrophenereignis vorgestellt. Die in [97] berücksichtigten Beiträge beziehen sich dabei weitgehend auf die Unterstützung von USaR-Operationen.
Die Schrägbildphotogrammetrie ist ein innovativer Ansatz, mit dem eine Szene schnell digital rekonstruiert werden kann. Im Gegensatz zur Nadir-Photogrammetrie, die traditionell zur Erstellung zweidimensionaler Orthophotos verwendet wird, ermöglicht dieses Verfahren eine erhebliche Zeitersparnis bei der Datenerfassung und der photogrammetrischen Auswertung. Der Unterschied zwischen den genannten Aufnahmemethoden ist in der Abbildung 2.5 illustriert. Die Vorteile der Schrägbildphotogrammetrie für die digitale Rekonstruktion von Katastrophenszenen wurde in [168] ausführlich behandelt.

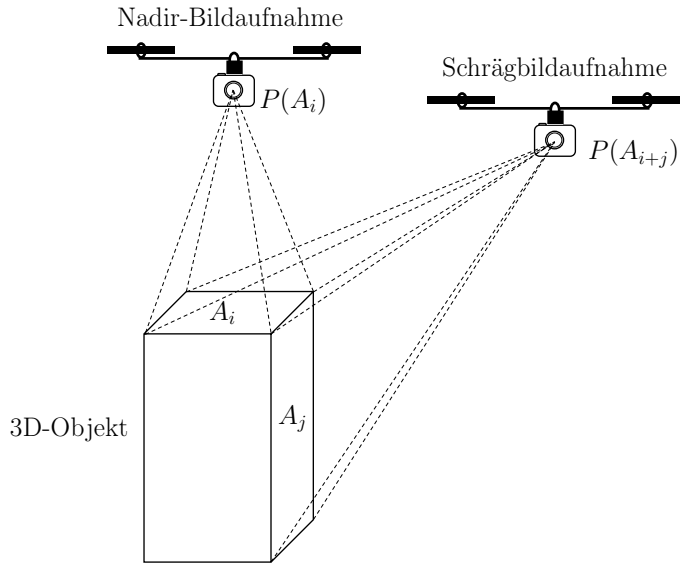


Abbildung 2.5: Nadir-Bildaufnahme $P(A_i)$: Die Drohnenkamera ist nach unten gerichtet und kann daher nur die oberen Oberflächen A_i aufnehmen. Schrägbildaufnahme $P(A_{i+j})$: Die Bilder werden aus einer schrägen Perspektive aufgenommen, wodurch sowohl die Oberseite A_i als auch die Seitenflächen A_j des Objektes abgebildet werden.

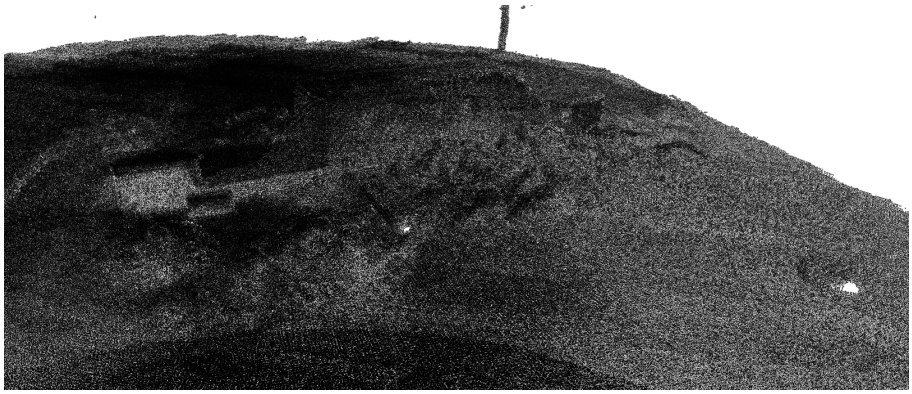
Die Photogrammetrie ist nur in der Lage, geometrische Daten von Objekten zu erfassen, die vom Kameraobjektiv im sichtbaren Spektrum und im nahen Infrarot erkannt werden können. In diesem Zusammenhang sind Verdeckungen während der Datenerfassung ein bekanntes Hindernis in der *Computer Vision*, das in der Literatur oft diskutiert wurde [15, 28, 32]. Ein Nachteil der Photogrammetrie ist, dass diese passive Messmethode Licht und lichtreflektierende Oberflächen benötigt, um eine digitale Rekonstruktion von realen Objekten durchführen zu können. Aus diesem Grund können Bilder von transparenten Oberflächen, wie Glas oder Wasser, nicht photogrammetrisch verarbeitet werden. Gleiches gilt für Oberflächen, die aufgrund der Tageszeit oder der Witterungsverhältnisse schlecht oder gar nicht beleuchtet sind. Dies kann jedoch durch künstlich erzeugtes Licht, z. B. durch LED-Flutlichtstrahler, behoben werden.

2.1.2 Laserscanning

Das Laserscanning ist eine aktive Fernerkundungsmethode, die zur Gewinnung geometrischer Informationen von physischen Objekten oder großflächigen Umgebungen in Form einer 3D-Punktwolke eingesetzt wird. Das zu erfassende Objekt wird mit einem Laserstrahl, in der Regel im nahen Infrarotbereich, abgetastet [80]. Im Gegensatz zur Photogrammetrie ist daher beim Laserscanning keine Belichtung der Oberfläche durch externe Lichtquellen erforderlich. Die emittierte Laserstrahlung wird an der Oberfläche reflektiert und vom Detektor des Sensors empfangen. Über die Laufzeit des Signals wird der Abstand zwischen Sender (Sensor) und Empfänger (Oberfläche) berechnet und die entsprechenden Punkte $P_i(x_i, y_i, z_i)$ im dreidimensionalen Koordinatensystem ausgegeben.
Je nach Anwendung wird die Lasertechnik für 3D-Messungen entweder terrestrisch oder aus der Luft eingesetzt. Terrestrisches Laserscanning (engl. *terrestrial laser scanning*, TLS) zeichnet sich vor allem durch die hohe erreichbare Auflösung aus, die je nach Abstand zum Aufnahmeobjekt zu sehr genauen und dichten Punktwolken führen kann. Während des Scanvorgangs ist der terrestrische Laserscanner stationär. Aus diesem Grund werden bei TLS-Anwendungen in der Regel mehrere Scanpositionen gewählt, um ein vollständiges digitales Abbild der aufgenommen Szene zu erhalten. Die Zusammensetzung der einzelnen Laserscans vom lokalen Koordinatensystem in ein referenziertes Koordinatensystem erfolgt meist mit dem *Iterative-Closest-Point*-(ICP)-Algorithmus [21, 35]. Dieser Vorgang wird als Referenzierung (engl. *registration*) bezeichnet. Üblicherweise erfolgt die Referenzierung mit den Positionsdaten eines globalen Navigationssatellitensystems (engl. *global navigation satellite system*, GNSS), das auf dem terrestrischen Laserscanner montiert ist. Die Referenzierung der Laserscans mit einem GNSS wird als Georeferenzierung bezeichnet. Punktwolken aus TLS-Systemen können auch basierend auf *Global-Positioning-System*-(GPS)-Daten direkt georeferenziert werden [142]. Der Einsatz des terrestrischen Laserscannings für die Überwachung und Messung von Bauteilverformungen wurde vor einigen Jahren behandelt [74]. Mehrere Forscher haben neue Ansätze zur automatischen Erzeugung von Finite-Elemente-(FE)-Modellen aus TLS-Punktwolken vorgeschlagen [44, 82]. Weiterhin wurde das TLS zur Erfassung von Tornoda-induzierten Gebäudeschäden getestet [93]. In [89, 182] wurde die TLS-Technologie bereits für die digitale Rekonstruktion von Katastrophenszenen und zur Unterstützung von USaR-Operationen untersucht und erzielte dabei gute Ergebnisse.
Im Gegensatz zum TLS werden beim luftgestützten Laserscanning (engl. *airborne laserscanning*, ALS) bemannte oder unbemannte Drohnen als Plattform für dynamische Messungen aus der Luft eingesetzt. Für die luftgestützten 3D-Messungen wird vorzugsweise die *Light-Detection-and-Ranging*-(LiDAR)-Methode mit einem geeigneten LiDAR-System verwendet. LiDAR-Systeme sind klein, kompakt und leicht, weshalb sie problemlos auf einer Drohne montiert werden können. Drohnenbasierte LiDAR-Systeme können die Umgebung in einem 360-Grad-Sichtwinkel bestrahlen und die entsprechenden Punktdaten über die Laufzeit des reflektierten Signals ausgeben. Die erzeugten 3D-Punktwolken können wie bei der TLS-Technologie eine sehr hohe Punktdichte auf-

weisen, wobei die erreichbare Auflösung von der Flughöhe abhängt. Da bei der LiDAR-Methode ein fokussierter Laserstrahl verwendet wird, um Oberflächen dynamisch abzutasten, ist der Aufnahmebereich lokal stärker begrenzt als beim TLS [80]. Auf einer dynamischen Plattform, wie z. B. einer unbemannten Drohne, stellt der Aufnahmebereich des LiDAR-Verfahrens jedoch kein Hindernis dar, sofern die zu erfassende Szene gründlich abgeflogen wird. Die erzeugten Punktdaten aus den Laserscans enthalten keine Farbinformationen, können aber mit Hilfe von Bilddaten angereichert werden. Aus Abbildung 2.6 ist ersichtlich, dass Farbinformationen die Interpretation der in einer Punktwolke dargestellten Szene wesentlich erleichtern. Bei der Erfassung der Drohnenbilder sind die Voraussetzungen der passiven Fernerkundung zu beachten, wie z. B. eine ausreichende Beleuchtung der Objekte.

(a)



(b)

Abbildung 2.6: Luftgestützte LiDAR-Punktwolke: (a) Ohne Farbinformationen. (b) Mit Farbinformationen aus Drohnenbildern.

In den letzten Jahren wurde die drohnenbasierte LiDAR-Technik für verschiedene Anwendungen eingesetzt und verbessert. Erste Ansätze für die semantische Segmentierung von LiDAR-Daten wurden von *Filin* und *Pfeifer* [61] vorgeschlagen. *Nagai et al.* [122] entwickelten eine neue Methode zur direkten Georeferenzierung von drohnenbasierten LiDAR-Daten für die großflächige Darstellung von Gebieten nach Naturkatastrophen. In [51] wird ein Überblick über den Einsatz von Fernerkundungsmethoden, einschließlich LiDAR, zur Erfassung von Gebäudeschäden nach Erdbebenereignissen gegeben. *Dong* und *Shan* [51] betonen, dass die LiDAR-Technologie nicht über die gleiche hohe Auflösung wie terrestrische Laserscanner verfügt. Aus diesem Grund werden diese Messmethoden oft in Kombination eingesetzt, um einerseits die hohe Auflösung mit dem

TLS und andererseits die vollständige Abdeckung der Szene durch die flexible ALS-Methode zu erreichen [65, 144]. Der Einsatz von TLS und ALS für USaR-Zwecke wurde in [144] im Rahmen eines EU-Projektes [89] behandelt. Die Ergebnisse in [144] zeigen, dass beide Messmethoden zur Unterstützung von USaR-Maßnahmen geeignet sind.
Wesentliche Nachteile des modernen Laserscannings sind die hohen Kosten und die komplexe Anwendung der Messgeräte. In den meisten Fällen ist zudem eine umfangreiche Nachbearbeitung der Laserscans, z. B. zur Georeferenzierung, erforderlich. Die oben beschriebene Methode der direkten Georeferenzierung erspart zwar eine zeitaufwändige Nachbearbeitung, muss aber im Voraus mit geeigneten Algorithmen und der entsprechenden Ausrüstung (GNSS oder GPS-Sensoren mit einer inertialen Messeinheit (engl. *inertial measurement unit*, IMU) eingerichtet werden, was die Komplexität des Verfahrens erhöht.

2.2 Verarbeitung von Punktwolken

Punktwolken werden aus der 3D-Messung von reflektierenden Oberflächen durch aktive oder passive Fernerkundung gewonnen. Neben der erfassten Oberflächengeometrie in Form von Punktmengen im 3D-Raum können sie auch Farbinformationen aus Bildern enthalten. Da eine Punktwolke in der Regel als Grundlage für eine bestimmte Anwendung verwendet wird, beginnt der Arbeitsablauf mit der Vorverarbeitung (engl. *pre-processing*) der Daten. Dazu gehören die Bereinigung der Punktwolke, z. B. von Ausreißern, die Bestimmung der Oberflächennormalen, die Heruntertaktung (engl. *Downsampling*) der Punktwolke mit einem Voxel-Gitter, das Ausschneiden nicht benötigter Objekte, etc. Zahlreiche fortschrittliche Punktwolken-Algorithmen für die Vorverarbeitung sind in *Open-Source*-Bibliotheken und Programmen wie *Open3D* [186] oder *CloudCompare* [40] implementiert. Im folgenden Abschnitt wird der Stand der Wissenschaft und Technik in Bezug auf die Referenzierung, Segmentierung, Klassifizierung und Oberflächenrekonstruktion von 3D-Punktwolken dargestellt.

2.2.1 Referenzierung

Wie in Abschnitt 2.1 erwähnt, muss eine durch passive oder aktive Fernerkundung gewonnene Punktwolke im ersten Schritt referenziert werden, damit sie im 3D-Raum mit der richtigen Position, Orientierung und dem richtigen Maßstab dargestellt wird. Zu diesem Zweck eignet sich die Verwendung von Positions- und Orientierungsinformationen aus GPS- und IMU-Systemen. Die erreichbare Auflösung von eigenständigen GPS-Sensoren variiert zwischen 1 und 10 m. Die erreichbare Positionsgenauigkeit hängt von den empfangenen GPS-Satelliten zum Zeitpunkt der Positionsbestimmung ab. In der Praxis kann nicht garantiert werden, dass während der Messung in jedem Fall Signale von mehreren GPS-Satelliten empfangen werden, was bei der Positionsbestimmung zu Abweichungen von bis zu ± 10 m führen kann. Aus diesem Grund wurde die sogenannte *Real-Time-Kinetic*-(RTK)-GPS-Technik eingeführt, die die Position üblicherweise mit einer Genauigkeit von bis zu 5 cm bestimmen kann. Die Verwendung des RTK-GPS-Empfängers in Kombination mit luftgestützter Photogrammetrie wurde in [75, 135] diskutiert. In beiden Untersuchungen konnte eine bemerkenswerte Steigerung der Positionsgenauigkeit mittels RTK-Technologie beobachtet werden. Die Nachteile der RTK-Technologie sind die technische Komplexität und der hohe Kostenfaktor.
Im Katastrophenfall muss eine schnelle und zuverlässige Methode zur Georeferenzierung der Punktwolke eingesetzt werden, die möglichst wenig bis gar keine Nachbearbeitung erfordert. Dieses Konzept könnte mit Hilfe von GPS-Sensoren in einem drohnengestützten photogrammetrischen Verfahren umgesetzt werden. Dabei werden Informationen zur Kameraposition und -ausrichtung für jedes während des Scanvorgangs aufgenommene Bild in einem externen Logbuch

gespeichert. Anstelle von Pass- oder Verknüpfungspunkten können die genauen Positions- und Orientierungsdaten der Kamera aus dem Logbuch für die äußere Orientierung genutzt werden. Diese Methode ermöglicht eine erheblich schnellere digitale Rekonstruktion durch den Einsatz der SfM-Technologie. *Hertle et al.* [81] untersuchten die Möglichkeit der direkten Georeferenzierung für die luftgestützte Photogrammetrie mit GPS-Informationen anhand einer alten Betonbrücke in Ebersbach, Deutschland. Die Ergebnisse in [81] zeigen, dass die geometrischen Abmessungen der Brücke mit einer durchschnittlichen Abweichung von 25 cm erfasst werden konnten, wobei nur GPS- und IMU-Daten für die äußere Orientierung verwendet wurden.
Ein alternativer Ansatz zur schnellen Skalierung der erhaltenen nicht georeferenzierten Punktwolke besteht darin, ein Referenzobjekt zu verwenden, dessen geometrische Abmessungen bekannt sind. Die Punktwolke, die auch das Referenzobjekt enthält, kann dann anhand der bekannten Abmessungen des Referenzobjektes skaliert werden. Es ist zu beachten, dass mindestens eine Dimension entlang der X-, Y- und Z-Achse angegeben werden muss, damit die Punktwolke korrekt skaliert werden kann. Eine Kombination aus beiden Referenzierungsmethoden, der direkten Georeferenzierung über GPS- und IMU-Daten und der Verwendung eines Referenzobjektes, kann dazu beitragen, Ungenauigkeiten zu vermeiden und die Ausgabedimensionen der Szene zu überprüfen.

2.2.2 Segmentierung

Punktwolken von komplexen Szenarien wie städtischen Räumen oder Katastrophenszenen bestehen aus einer enormen Anzahl von unstrukturierten und unregelmäßig verteilten Punkten. Zusätzlich zu den fehlenden semantischen Informationen können Rauscheffekte und Datenlücken aufgrund von Verdeckungen die Interpretation einer Punktwolke stark beeinträchtigen. Um ein besseres Verständnis der aufgenommenen Szene zu erhalten, müssen daher die Punkte aus dem Datensatz, die eine erkennbare Form oder einen Körper darstellen, entsprechend segmentiert werden. Insbesondere bei der digitalen Oberflächenrekonstruktion spielt die Segmentierung von Objekten aus der Punktwolke eine wesentliche Rolle. Da die digitale Oberflächenrekonstruktion und die weitere Verarbeitung auf der Grundlage der segmentierten Punkte erfolgt, ist dieser Schritt ein entscheidender Maßstab für die Beurteilung der Genauigkeit des Gesamtergebnisses. Aus diesem Grund muss der Segmentierung von Punktwolken im Rahmen einer automatischen digitalen Modellrekonstruktion große Aufmerksamkeit gewidmet werden.
Die Segmentierung beschreibt den digitalen Prozess der Punktmengenextraktion aus einer Punktwolke basierend auf übereinstimmender Oberflächenmerkmale, die durch verschiedene Eigenschaften der Punkte definiert werden. Die vorhandenen Segmentierungsmethoden aus der Literatur werden von *Nguyen* und *Le* [125] in fünf Kategorien eingeteilt:

(i) Kantenbasierte Methoden,
(ii) Regionenbasierte Methoden,
(iii) Modellbasierte Methoden,
(iv) Graphenbasierte Methoden und
(v) Attributbasierte Methoden.

Wesentliche Beiträge zu den oben genannten Segmentierungsmethoden **(i)** bis **(v)** sind in [76, 103, 125, 146] ausführlich zusammengestellt und werden im Folgenden kurz wiedergegeben.
Kantenbasierte Segmentierungsmethoden **(i)** nutzen Merkmale wie Gradienten oder lokale Krümmungen, um diskontinuierliche Bereiche oder Kanten in Punktwolken zu identifizieren und Übergänge zwischen den Regionen zu erfassen. Die Verwendung von Gradienten zur Erkennung von Kanten in Punktwolken wurde von *Bhanu et al.* [22] vorgeschlagen. Zu diesem Zweck werden bei

dem von den Autoren vorgeschlagenen Ansatz 3D-Linien an eine Punktreihe angepasst, um auf dieser Grundlage Änderungen in der Richtung des Oberflächennormalenvektors zu bestimmen. Im Gegensatz dazu präsentieren *Wang et al.* [172] einen Algorithmus zur Kantenidentifikation in Punktwolken, der auf der Gitterstrukturierung der Punktdaten und der Anwendung von α-*Shape*-Bedingungen basiert.
Bei regionenbasierten Segmentierungsverfahren **(ii)** werden Punkte anhand ähnlicher Merkmale, wie Orientierung oder Krümmung, in homogene Bereiche eingeteilt, was die Ableitung einer zusammenhängenden Struktur innerhalb des Punktedatensatzes ermöglicht. Laut *Ruan* und *Liu* [146] ist diese Methode genauer als kantenbasierte Segmentierungsverfahren. *Xiao et al.* [177] schlugen zwei *Region-Growing*-Algorithmen zur schnellen Ebenensegmentierung in 3D-Punktwolken vor. Diese Algorithmen verwenden inkrementelle Methoden, um die Flächenparameter schrittweise zu berechnen, während die wachsende Region erweitert wird. Dadurch wird im Vergleich zu herkömmlichen Ansätzen eine insgesamt beschleunigte Segmentierung ermöglicht. In [170] präsentieren die Autoren einen innovativen Algorithmus für die schnelle Segmentierung von Oberflächen in TLS-Punktwolken städtischer Umgebungen. Dieser Algorithmus arbeitet in zwei Phasen. In der ersten Phase wird eine *Region-Growing*-Verarbeitung auf einer *Octree*-basierten Voxel-Darstellung der Eingabepunktwolke durchgeführt, um grobe Segmente zu extrahieren. Die erzielten Ergebnisse werden dann in der zweiten Phase einem Verfeinerungsprozess mit Clusterbedingungen unterzogen, um ebene Punktmengen präzise zu identifizieren.
Modellbasierte Segmentierungsmethoden **(iii)** verwenden Formfunktionen einfacher Primitive aus der analytischen Geometrie, wie z. B. eine gerade Linie, eine Ebene oder eine Kugel, um Objekte aus einer 3D-Punktwolke zu segmentieren, die mit diesen Formfunktionen beschrieben werden können. *Fujiwara et al.* [66] untersuchten die Verwendung des *Random-Sample-Consensus*-(RANSAC)-Algorithmus [62] zur Segmentierung von ebenen Punktmengen. Die Ergebnisse in [66] zeigen, dass RANSAC mit den richtigen Parameterwerten in der Lage ist, ebene Punktmengen aus dichten und auch dünnen 3D-Punktwolken zuverlässig zu identifizieren. In der *Computer Vision* ist RANSAC eine beliebte Methode, um falsch identifizierte Verknüpfungspunkte in Bildern zu entfernen, z. B. im Zusammenhang mit dem SIFT-Algorithmus. In den letzten Jahren wurden neue Ansätze zur Verbesserung der Leistung der RANSAC-Methode für den SIFT-Algorithmus vorgestellt [155, 184]. Ein Überblick über die Erweiterungen von RANSAC ist in [137] zusammengefasst. In *CloudCompare* [40] kann der RANSAC-Algorithmus mit den vordefinierten Funktionen einer Ebene, einer Kugel, eines Kegels und eines Torus zur Segmentierung von Punktwolken verwendet werden. Details zur Implementierung von RANSAC mit den genannten Primitiven (Ebene, Kugel, Kegel, Torus) sind in [149] zu finden. Die Segmentierung von ebenen Punktmengen, die z. B. Wände, Decken oder Böden darstellen, wurde in [66, 91, 115, 160] mit der RANSAC-Methode untersucht. Die erzielten Ergebnisse zeigen, dass RANSAC im Stande ist, die genannten Objekte aus einem realen Datensatz zuverlässig zu segmentieren. Eine Erweiterung der RANSAC-Methode zur Segmentierung von Ebenen aus Punktwolken von Kulturerbestätten wurde in [157] präsentiert. Des Weiteren wurde die 3D-Hough-Transformation von *Limberger* und *Oliveira* [105] für die Ebenendetektion [30, 86] sowie für die Identifizierung von Kugeln in 3D-Punktwolken diskutiert [27]. Ein großer Nachteil der modellbasierten Segmentierung ist, dass verrauschte und komplexe Punktmengen im 3D-Raum die Anwendung der Methode stark einschränken können [146].
Graphenbasierte Segmentierungsmethoden **(iv)** betrachten die Punktwolke als einen Graphen, wobei Punkte als Knoten und ihre Verbindungen als Kanten dargestellt werden. Basierend auf den Eigenschaften der Graphenstruktur identifizieren diese Methoden sinnvolle Segmente innerhalb der Punktwolke. Der bekannte FH-Algorithmus für die Bildsegmentierung, der auf einem graphenbasierten Ansatz zur Messung von Grenzen zwischen Regionen basiert, wurde von *Felzenswalb* und *Huttenlocher* [58] vorgeschlagen. Eine wichtige Eigenschaft dieses Algorithmus ist

seine Fähigkeit, bei der Segmentierung Details in Bildbereichen mit geringer Variabilität zu erhalten, während Details in Bereichen mit hoher Variabilität ignoriert werden. *Strom et al.* [158] präsentieren eine innovative Methode zur effizienten und robusten Segmentierung einer Laserpunktewolke, die sowohl geometrische Merkmale als auch Farbinformationen berücksichtigt. Laut den Autoren eignet sich dieser Ansatz aufgrund ihrer Leistungsfähigkeit besonders für die Geländeklassifizierung und Objekterkennung in Echtzeit. Des Weiteren wurde in [34] und [72] die Anwendung einer Segmentierungsmethode basierend auf *k-nearest-neighbours*-Graphen für die automatische Segmentierung von Punktwolken und *Mesh*-Körpern vorgestellt.
Attributbasierte Methoden **(v)** beruhen auf dem Gruppieren ähnlicher Punktwolkenattribute, auch bekannt als *Clustering*. Eine spezielle Methode zur Durchführung dieses *Clustering*-Prozesses ist der *k-means*-Algorithmus, der häufig zur Berechnung solcher Gruppierungen verwendet wird. Diese Methoden repräsentieren einen Ansatz des unüberwachten maschinellen Lernens (engl. *unsupervised machine learning*) und umfassen zwei Hauptphasen: die Extraktion von Merkmalen aus den Daten und das nachfolgende *Clustering* der Punktdaten anhand dieser Merkmale. *Filin* und *Pfeifer* [61] entwickelten eine clusterbasierte Segmentierungsmethode, die geometrische Merkmale, wie den Normalvektor der Punkte im 3D-Raum, als Filterbedingung für die Segmentierung von luftgestützten LiDAR-Punktwolken verwendet. *Shen et al.* [154] schlagen eine neue clusterbasierte Methode zur Segmentierung und Rekonstruktion von Ziegeln aus einem arbiträren Ziegelhaufen vor [154]. Der von den Autoren vorgeschlagene Segmentierungsansatz basiert auf der Verwendung geometrischer Punktattribute und -cluster, bei denen die identifizierten Nachbarpunkte zur Bestimmung der ebenen Oberflächen der Ziegel eingesetzt werden.

2.2.3 Klassifizierung

Um das Szenenverständnis einer 3D-Punktwolke zu verbessern, werden den segmentierten Punktdaten Objektklassen oder Semantiken zugeordnet. Dieser Prozess wird als Punktwolkenklassifizierung oder semantische Segmentierung bezeichnet. Die semantische Segmentierung von Bilddaten auf Pixelebene ist eine bekannte Herausforderung auf dem Gebiet der *Computer Vision*. Diese fortschrittliche Technologie zielt darauf ab, jedem Pixel eines Bildes automatisch semantische Bezeichnungen (engl. *labels*) wie *Building*, *Car* oder *Street* zuzuordnen und sie entsprechend zu segmentieren [181]. Zu diesem Zweck werden spezielle Algorithmen des maschinellen Lernens (engl. *machine learning*, ML) verwendet, die als *Convolutional Neural Networks* (CNNs) bekannt sind [181]. Die Anwendung von CNNs für die semantische Segmentierung von 3D-Punktwolken ist eine neue und noch größere Herausforderung als die semantische Segmentierung von zweidimensionalen Bildszenen. Im Gegensatz zu Bildern weisen 3D-Punktwolken in der Regel eine unregelmäßige und unstrukturierte Verteilung der Punkte im 3D-Raum auf, was die Übertragung konventioneller CNNs, die für die Verarbeitung regelmäßiger Pixel ausgelegt sind, erheblich erschwert. Aus diesem Grund wurden in den letzten Jahren neuronale Netzwerke speziell für die Verarbeitung unstrukturierter Punktwolken entwickelt. Eine Zusammenstellung von leistungsstarken 3D-CNNs findet sich in der Literaturübersicht von *Xie et al.* [178] und *Zhang et al.* [183]. Die semantische Segmentierung von 3D-Punktwolken wurde bereits für die automatische Erzeugung von *as-built* BIM-Modellen untersucht [113, 133]. Der große Vorteil gegenüber herkömmlichen *Scan-to-BIM*-Methoden (Unterabschnitt 2.3.2), bei denen die Semantik von digital rekonstruierten Objekten manuell vergeben wird, liegt in der automatischen Informationsanreicherung der Punktdaten. Für die BIM-Anwendung werden Objektklassen für Bauteile oder Möbel als semantische Informationen verwendet [113, 133].
Die zuverlässige Vorhersage von Objektklassen mit 3D-CNNs hängt weitgehend von der Größe und Qualität des Trainingsdatensatzes ab. Die manuelle Erstellung geeigneter Trainingsdaten stellt dabei eine große Herausforderung dar und ist mit einem hohen Arbeitsaufwand verbunden. Derzeit stellen mehrere *Benchmarks* der Forschungsgemeinschaft vorgefertigte Punktwolkenda-

tensätze mit segmentierten und klassifizierten Punktdaten zur Verfügung, die für die semantische Segmentierung mit einem geeigneten 3D-CNN verwendet werden können. Einige bekannte Beispiele hierzu sind:

- **Semantic3D** [78]: Ein umfassender TLS-basierter Punktwolkendatensatz, bestehend aus natürlichen und städtischen Szenen. Zu den klassifizierten Objektklassen gehören u. a. *Building*, *Terrain*, *Car*, *Scan Artifact* und *Vegetation*.
- **S3DIS** [13]: S3DIS enthält einen umfassenden Punktwolkendatensatz, der Gebäudeinnenräume mit vielen vordefinierten Objektklassen, z. B. *Slab*, *Window*, *Door* und *Table*, darstellt.
- **SemanticKITTI** [18]: Dieser Punktwolkendatensatz wurde speziell zur Unterstützung des autonomen Fahrens im Bereich der Robotikforschung entwickelt. Der Datensatz enthält daher ausschließlich Verkehrsszenen, die mit einem mobilen Laserscanner aufgenommen wurden.

Der Hauptzweck der vorgefertigten Trainingsdatensätze besteht darin, die Leistung neuer 3D-CNNs im Vergleich zu anderen Vorschlägen aus der Literatur zu überprüfen. Laut *Xie et al.* [178] ist derzeit keine CNN-Anwendung in der Lage, konsistente Vorhersagen zu treffen, da die Leistung in hohem Maße von der Eingabe abhängt. Es ist darauf zu achten, dass die eingegebene Punktwolke weitgehend vollständig und ausreichend dicht ist, um eine eindeutige semantische Segmentierung von Objekten auf der Grundlage des vortrainierten Modells zu ermöglichen.

2.2.4 Oberflächenrekonstruktion

Der 3D-Aufnahme eines Objektes oder einer Szene geht eine digitale Oberflächenrekonstruktion aus der Punktwolke voraus. Daraus ergibt sich das *Computer-Aided-Design*-(CAD)-Modell oder Oberflächenmodell, das für eine bestimmte Anwendung, z. B. in der Geodäsie oder im Bauwesen, weiterverarbeitet werden kann. Die Oberflächenrekonstruktion aus unorganisierten Punkten $P_i(x_i, y_i, z_i)$, mit $i \in \mathbb{N}$ im euklidischen $\mathbb{R}^3$, hat in den letzten 25 Jahren einen bemerkenswerten Fortschritt auf dem Gebiet der algorithmischen Geometrie (engl. *Computational Geometry*) und *Computer Vision* verzeichnet [19, 99]. Im Folgenden werden einige weit verbreitete Methoden zur Oberflächenrekonstruktion kurz vorgestellt. Eine ausführliche Literaturübersicht über den Stand der Technik von Oberflächenrekonstruktionsmethoden findet sich in [19, 99, 104].
Für die digitale Rekonstruktion von Oberflächen aus der Punktwolke werden üblicherweise Dreiecksflächen genutzt [99]. Die Kanten der Dreiecksflächen werden anhand von drei benachbarten Punkten, den sogenannten Eckpunkten (engl. *vertices*), definiert. Dementsprechend ist die Anzahl n der Dreiecksflächen, die aus der Punktmenge generiert werden können, eine Funktion der Punktdichte. Das Oberflächenmodell, bestehend aus $n > 1$ endlichen Flächenelementen, wird auch als Polygonnetz (engl. *mesh*) bezeichnet. Bekannte Methoden aus der algorithmischen Geometrie, die Dreiecksnetze im dreidimensionalen euklidischen Raum erzeugen, sind die *Delaunay-Triangulation* [26] und die α-*Shape*-Methode [54, 179]. Gemäß der *Delaunay*-Bedingung liegt eine Triangulation zwischen drei benachbarten Punkten $\{P_1, P_2, P_3\}$ aus der Punktmenge $\mathbf{P}$ vor, wenn keine weiteren Punkte $P_{i>3}$ innerhalb des Umkreises des Dreiecks liegen. Bei der α-*Shape*-Methode hingegen wird die äußere Grenze der Oberflächengeometrie durch den α-Parameter, $\alpha \in \{0, 1\}$, approximiert, und die innerhalb dieser Grenze liegenden Punkte mit der *Delaunay-Triangulation* verarbeitet. Dabei wird mit $\alpha = 1$ eine konvexe Hülle einer Punktmenge $\mathbf{P}$ erzeugt und mit $\alpha < 1$ eine konkave Hülle der Oberfläche approximiert, wobei auch Fehlstellen, d. h. Öffnungen, innerhalb der Grenze berücksichtigt werden können.
Eine weitere Methode zur digitalen Oberflächenrekonstruktion aus Punkten im dreidimensionalen euklidischen Raum ist der *Ball-Pivoting*-Algorithmus (BPA) von *Bernardini et al.* [20]. Der

BPA erzeugt ein Dreiecksnetz unter Verwendung einer gedachten Kugel mit konstantem Radius r. Die Kugel dreht sich entlang der Randkante von einem Punkt zum nächsten benachbarten Punkt und vergleicht anhand der Normalvektoren, ob die Punkte auf der Kugeloberfläche für die Erstellung eines Dreiecks geeignet sind. Aufgrund der schrittweisen Abtastung von Punkten mit einer konstanten Kugelfunktion, ist der BPA robust gegenüber verrauschten Punktmengen. Dies kann jedoch zu zahlreichen Lücken bei der Oberflächenrekonstruktion führen, d. h. zu einer nicht-mannigfaltigen Geometrie, insbesondere wenn aus der unregelmäßigen Punktwolke kein geeigneter Radius für die Kugelfunktion abgeleitet werden kann. *Kazhdan et al.* [95] schlagen vor, die Oberflächenrekonstruktion als ein mathematisches *Poisson*-Problem auszudrücken, bei dem die Oberflächengeometrie der Punktmenge durch eine Indikatorfunktion angenähert und basierend darauf ein Dreiecksnetz erzeugt wird. Die *Poisson*-Rekonstruktionsmethode ist sehr robust gegenüber verrauschten Punktdaten und kann für die Generierung detaillierter und lückenloser Oberflächentopologien aus Punktwolken eingesetzt werden. In [59, 168] verwendeten die Autoren die Implementierung des *Poisson*-Algorithmus in *Meshlab* [39], um ein Oberflächenmodell aus der Punktwolke der erfassten Katastrophenszene zu generieren.
Die automatische Modellerzeugung mit Hilfe von Voxeln stellt die einfachste Form der digitalen Oberflächenrekonstruktion aus Bildinformationen [92, 151], Oberflächen [42] oder Punkten [82] im 3D-Raum dar. Als Voxel werden würfelförmige *Mesh*-Körper in einem 3D-Gitter bezeichnet. Genau wie bei Pixeln in einer Rastergrafik, kann die Auflösung des rekonstruierten Objektes anhand der Voxelgröße gesteuert werden. Je feiner die Voxelgröße gewählt wird, desto besser kann die Oberfläche des realen Objektes digital rekonstruiert werden. Ebene und glatte Oberflächen aus einem Satz von Punkten können nicht mit Voxeln erstellt werden. In jüngster Zeit wurden mehrere lernbasierte Rekonstruktionsmethoden mit vielversprechender Leistung im Bereich des geometrischen *Deep Learnings* (DLs) vorgestellt [36, 140, 153]. Im Gegensatz zu den oben genannten Methoden aus der algorithmischen Geometrie oder der Computergrafik können lernbasierte Rekonstruktionsmethoden die Oberflächengeometrie eines Objektes nicht nur anhand seiner diskreten Punkte aus der 3D-Vermessung, sondern auch anhand seiner semantischen Kennzeichnung rekonstruieren. Dadurch lassen sich unerwünschte Artefakte und Ungenauigkeiten bei der Oberflächenrekonstruktion vermeiden, die insbesondere bei der Verarbeitung dünner Punktwolken mit verdeckten Bereichen auftreten können. *Hoppe et al.* [84] präsentieren eine konturbasierte Rekonstruktionsmethode, die automatisch aus einer Menge unstrukturierter Punkte in $\mathbb{R}^3$ eine Oberfläche generiert. Der entsprechende Algorithmus bestimmt die Nullmenge einer geschätzten Vorzeichenabstandsfunktion, um die topologische Struktur der Punktmenge, einschließlich möglicher Begrenzungskurven, zu erfassen. Darüber hinaus wurde in [85] ein innovativer Ansatz zur digitalen Rekonstruktion von Oberflächen aus unorganisierten Punktwolken vorgestellt, der sowohl topologische als auch geometrische Eigenschaften der Punktdaten berücksichtigt. Dies ermöglicht eine effiziente und automatische Erkennung von Objektgrenzen, selbst bei unvollständigen Daten und komplexen Objektformen.

2.3 Building Information Modeling

Der Begriff BIM beschreibt konventionell einen digitalen, modellgestützten und interdisziplinären Arbeitsprozess während des gesamten Lebenszyklus eines Bauwerkes, d. h. von der Planung, über die Ausführung und Überwachung bis schließlich zum Rückbau [88]. Im Vergleich zu herkömmlichen CAD-Modellen, die das traditionelle Zeichnen von 2D-Plänen nachahmen, unterscheidet sich die BIM-Technologie dadurch, dass sie die reale Welt in Form von 3D-Modellen imitiert. Die BIM-Methode eignet sich daher für die Erstellung digitaler Zwillinge mit einem hohen Entwicklungsgrad (engl. *Level of Development*, LoD). Neben den geometrischen Daten der Bauteile eines Bauwerkes enthält ein BIM-Modell auch semantische und alphanumerische Informationen

sowie Abhängigkeiten zu den einzelnen Bauteilen. In einem CAD-Modell wird eine Wand beispielsweise als eine Reihe unabhängiger ebener Flächen dargestellt, während die Wand in einem BIM-Modell als Volumenkörper mit nicht-geometrischen Daten (z. B. Klassen, Attributen) und objektorientierten Abhängigkeiten zwischen benachbarten Bauteilen definiert ist.
Der Grundstein der BIM-Methode wurde bereits in den 1970er Jahren von *Eastman et al.* [53] gelegt, die in ihrem Beitrag [53] erstmals Ansätze zur Erstellung von digitalen Bauwerksdatenmodellen mit semantischen Informationen diskutiert haben. Der Begriff Building Information Modeling wurde 1992 in der Forschungsarbeit von *Van-Nederveen* und *Tolman* [165] eingeführt. Heutzutage ist die BIM-Technologie im Bausektor weit verbreitet und eine etablierte Methode, was vor allem auf die Entwicklung modernster BIM-Softwaretools, wie *Revit AutoDesk* oder *Allplan Nemetschek*, und die Einführung des offenen, internationalen BIM-Standards IFC (engl. *Industry Foundation Classes*, IFC) [29] zurückzuführen ist. Weitere Informationen zu IFC sind in Unterabschnitt 2.3.1 zusammengefasst.

Laut *Keeffe* und *Bosche* [127] kann ein BIM-Modell einer der folgenden Zustandsbeschreibungen zugeordnet werden:

(i) *as-designed* BIM,
(ii) *as-built* BIM oder
(iii) *as-is* BIM.

Die BIM-Technologie wird üblicherweise für die interdisziplinäre Planung von Neubauten eingesetzt. In der Praxis werden daher überwiegend *as-designed* BIM-Modelle auf der Grundlage von Entwurfs- und Ausführungsplänen erstellt. Seit einigen Jahren werden aber auch zunehmend Bestandsbauwerke in den BIM-Prozess überführt, woraus sich die sogenannten *as-built* und *as-is* BIM-Methoden entwickelt haben [127]. Im Gegensatz zu *as-designed* BIM-Modellen beschreiben *as-built* und *as-is* BIM-Modelle das Bauwerk nach der Bauausführung mit den tatsächlich vorhandenen Informationen, die vom ursprünglichen Entwurf abweichen können. Das *as-built* BIM bezieht sich dabei auf den Zeitpunkt $t = 0$, d. h. unmittelbar nach der Bauausführung, während das *as-is* BIM den Bauwerkszustand zum Zeitpunkt $t = t_s$, d. h. zu einem bestimmten Zeitpunkt während der Nutzungsphase, beschreibt. Die oben genannten Zustandsbeschreibungen von BIM-Modellen werden von *Zeibak et al.* [182] mit *as-damaged* BIM-Modellen erweitert, womit ein Bauwerk in seinem beschädigten Zustand nach einem Katastrophenereignis virtuell dargestellt wird.
Um die Lage und Geometrie der Bauteile eines Bestandsbauwerkes im Ist-Zustand zu erfassen, werden moderne Messverfahren aus der Fernerkundung eingesetzt, wie z. B. das Laserscanning oder die Photogrammetrie (Abschnitt 2.1). Auf der Grundlage der erhaltenen 3D-Punktwolke wird das Bauwerk digital rekonstruiert und schließlich mit Informationen für die BIM-Anwendung angereichert. Diese Vorgehensweise wird im Allgemeinen als *Scan-to-BIM* bezeichnet. Der Stand der Forschung auf diesem Gebiet wird in Unterabschnitt 2.3.2 vorgestellt.

2.3.1 Industry Foundation Classes

IFC sind ein offener Standard im Bauwesen, der die objektorientierte Datenstruktur für den BIM-Datenaustausch definiert. Der IFC-Standard wurde erstmals von buildingSMART e.V. International [29] im Jahr 2000 veröffentlicht und seitdem kontinuierlich weiterentwickelt. IFC werden grundsätzlich zur Referenzierung und Archivierung von BIM-Modellen verwendet, um einen einheitlichen Modell- und Informationsaustausch ohne Datenverlust zwischen allen Projektbeteiligten zu gewährleisten. Seit 2018 ist die aktuelle Version dieses Standards IFC4 ADD2 TC1 [29] mit der anerkannten ISO-Norm 16739-1:2018. Im Gegensatz zur vorherigen IFC-Version

(IFC2x3) ermöglicht das IFC4-Schema unter anderem eine verbesserte Darstellung von komplexen Geometrien, wie z. B. von NURBS (engl. *Non-Uniform Rational B-splines*, NURBS). Die Spezifikationen von Klassen, Attributen und Relationen für die Bauteile werden mit der Datenmodellierungsprache EXPRESS umgesetzt. Diese genormte Sprache (ISO-Standard 10303-11) ist Teil des STEP-Datenformats [143]. Da IFC objektorierentiert sind, werden alle darin enthaltenen Bauteilinformationen als Einheiten (engl. *Entities*) gespeichert. Typische Beispiele für solche Einheiten sind die IFC-Klassen, wie z. B. *IfcWall* für Wände oder *IfcSlab* für Decken, mit denen eine eindeutige Zuordnung der Objekte gewährleistet wird. Die Abbildung 2.7 zeigt beispielhaft, welche Daten einem Bauteil gemäß dem IFC-Standard zugewiesen werden können. Die Informationen des semantischen Modells lassen sich aus der Verknüpfung dieser Daten ableiten. Für die Beschreibung von IFC-Eigenschaften stellt buildingSMART eine Reihe von vordefinierten Parametersätzen zur Verfügung, die mit *Pset_ ** gekennzeichnet sind. Beispielsweise kann mit dem Parameter *Pset_ WallCommon* angegeben werden, ob eine Wand im Modell tragend oder nichttragend ist.
Die meisten CAD-Programme verfügen mittlerweile über eine IFC-Schnittstelle, die den Import oder Export von IFC unterstützen. Ein Großteil der IFC-Implementierungen wird von der *OpenSource*-Gemeinschaft durchgeführt. Zu den beliebtesten offenen IFC-Bibliotheken gehören *xbIM* [176] und *IfcOpenShell* [87], wobei *xbIM* in C# und *IfcOpenShell* in C++ geschrieben ist. Da beide IFC-Toolkits unter der LGPL (*Lesser General Public License*, LGPL) lizenziert sind, können die Quellcodes jederzeit durch den Nutzer modifiziert werden. Die LGPL-Lizenzierung erlaubt sowohl die private als auch die kommerzielle Nutzung der entwickelten IFC-Plug-ins. Weitere Informationen zu IFC sind in [29, 143] zu finden.

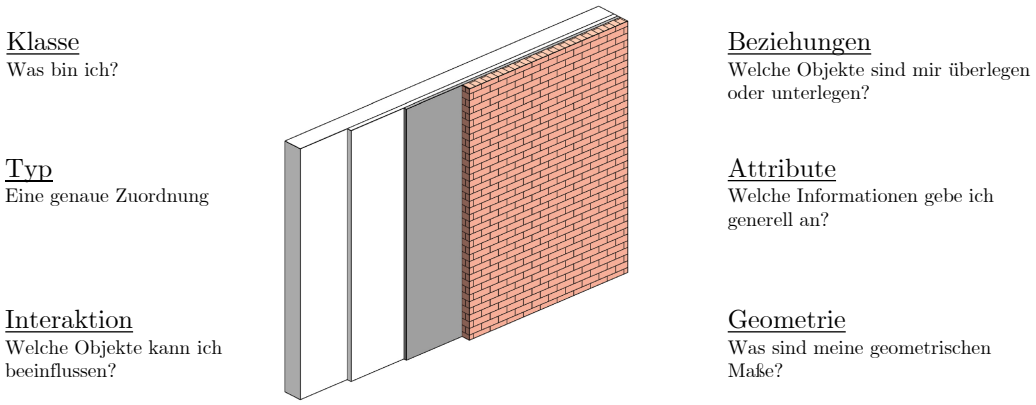


Abbildung 2.7: Beschreibung eines Bauteils gemäß dem IFC-Standard [143].

2.3.2 Scan-to-BIM

Die kontinuierliche Verbesserung und Entwicklung von BIM-Tools ermöglichen heute den Anwendern aus den Fachbereichen Architektur, Ingenieurwesen, Konstruktion und dem Facility Management eine sehr präzise und umfangreiche Arbeit mit der BIM-Technologie. Diese positive Entwicklung betrifft vor allem die Erstellung von *as-designed* BIM-Modellen. Auch für die verbesserte Nutzung des *Scan-to-BIM*-Prozesses wurden in den letzten Jahren auf dem kommerziellen Markt neue Entwicklungen implementiert (z. B. *EdgeWise*, *CloudWorx*, *PolyWorks Modeler*, *RealWorks*) [46]. Kommerzielle *Scan-to-BIM*-Programmsysteme erleichtern insbesondere die manuelle und zeitintensive Verarbeitung der Scan-Daten mittels halbautomatisierter Prozesse bei der digitalen Rekonstruktion des 3D-Modells. Trotz technologischer Fortschritte kommen *Czerniawski* und *Leite* [46] zu dem Schluss, dass noch keine Methode in der Lage ist,

semantisch reichhaltige BIM-Modelle von bestehenden Bauwerken zu erstellen und daher noch großer Entwicklungs- und Forschungsbedarf in diesem Bereich besteht. Derzeit wird bei der *Scan-to-BIM*-Methode noch weitgehend das TLS oder ALS zur berührungslosen Datenerfassung eingesetzt. Jüngste Entwicklungen zeigen, dass ein moderner Laserscanner im Vergleich zu anderen bildgebenden Messverfahren am besten für die *Scan-to-BIM* geeignet ist [109]. Da der Einsatz von LiDAR-Systemen mit Anwendungs- und Leistungsgrenzen verbunden ist, haben mehrere Forscher die halbautomatische *as-built* BIM-Generierung aus Laser- und Bilddaten untersucht [49, 52, 106]. In diesem Zusammenhang ermöglichen moderne 2D-CNNs eine semantische Informationsanreicherung durch Erkennung und Klassifizierung von Objekten aus Bilddaten [46].
Die Erstellung von *as-built* und *as-is* BIM-Modellen erfolgt noch größtenteils manuell, weshalb die damit verbundene Datenverarbeitung in [116, 159] als sehr zeitaufwendig und subjektiv beschrieben wird. In den letzten Jahren haben sich mehrere Forscher mit der Automatisierung der *Scan-to-BIM*-Methode beschäftigt. In der umfassenden Literaturübersicht von *Pătrăucean et al.* [131] werden zahlreiche Beiträge von Autoren, die sich mit der Automatisierung der *Scan-to-BIM*-Methode beschäftigen, zusammengefasst und bewertet. *Pătrăucean et al.* [131] stellen fest, dass die meisten Ansätze in der Literatur auf der Segmentierung der Punktwolke in ebene Oberflächen und der Erstellung von Beziehungen zwischen den rekonstruierten Bauteilen basieren. Eine Generalisierung der Methoden sei schwer umsetzbar und müsse für den jeweiligen Zweck angepasst werden. Die Literaturübersicht von *Pătrăucean et al.* [131] hebt insbesondere die vielen Herausforderungen der *Scan-to-BIM*-Methode hervor, wie z. B. der Umgang mit Verdeckungen während der Datenerfassung, die Zuverlässigkeit von Objekterkennungsverfahren und die Genauigkeit der digitalen Rekonstruktion. Für eine verbesserte Automatisierung der Arbeitsschritte muss laut den Autoren der Fokus auf den Einsatz von Objekterkennungsverfahren gelegt werden.
In [116, 160] werden halbautomatische *Scan-to-BIM*-Methoden für die digitale Rekonstruktion des Innenraums von bestehenden Gebäuden vorgeschlagen. Für die Durchführung der Verfahren wird eine vollständige 3D-Punktwolke des Innenraums als Eingabe vorausgesetzt. Der vorgeschlagene Arbeitsablauf umfasst die Segmentierung der Punktwolke in Wände und Decken mit Hilfe des RANSAC-Algorithmus, die Oberflächenrekonstruktion der Punktsegmente in konvexe Körper, bestehend aus Dreiecksflächen im OBJ-Format [121], und die Umwandlung der Körper in den IFC-Format. Für die Umwandlung der erzeugten Körper in den IFC-Format wird in [116] die *Open-Source*-Software *FreeCAD* [64] verwendet und in [160] das offene *xBIM*-Toolkit [107].
Für die halbautomatische Rekonstruktion von Wänden aus Punktwolken wird in [17] ein Algorithmus präsentiert, mit dem das Wandmodell direkt in *Revit AutoDesk* [143] unter Verwendung von *Rhino+Grasshopper* und *Rhino.Inside* erstellt werden kann.
Die Arbeit von *Barazzetti et al.* [16] stellt eine zweistufige Methode vor, bei der im ersten Schritt ein BIM-Modell eines historischen Gebäudes manuell aus einer TLS-basierten Punktwolke erzeugt wird und im zweiten Schritt die geometrischen und relevanten Informationen des BIM-Modells als Grundlage für die Erstellung eines FE-Modells verwendet werden. Einfache und regelmäßige Bauteilkörper werden in [16] mit den Standardmethoden von *Revit AutoDesk* [143] erstellt, während komplexe Oberflächen mit NURBS modelliert werden. Die manuelle Erstellung des BIM-Modells nach *Barazzetti et al.* [16] erfordert neben einer vollständigen 3D-Punktwolke des Gebäudes von innen und außen auch Strukturdaten der Bauteile, wie z. B. Materialeigenschaften, die in ihrer Fallstudie mit zerstörenden und zerstörungsfreien Prüfmethoden ermittelt wurden. Darüber hinaus stellen die Autoren fest, dass die konventionellen BIM-Programme für die Modellierung komplexer Gebäudegeometrien ungeeignet sind und daher spezifische Lösungsansätze umgesetzt werden müssen. *Rolin et al.* [145] schlagen einen halbautomatischen *Scan-to-BIM-to-FEM*-Arbeitsablauf für Kulturerbestätten vor, wobei sich die automatisierten Arbeitsschritte auf die BIM-Modellierung beschränken. *Tommasi et al.* [161] empfehlen für die digitale Rekonstruk-

tion von Kulturerbestätten einen Ansatz, der mehrere Methoden, Techniken und Programme verbindet, um die jeweiligen Anwendungsgrenzen der eingesetzten Methoden zu überbrücken. Auf diese Weise sei es möglich, ein präzises *as-is* BIM-Modell von Kulturerbestätten zu erstellen.

Zeibak et al. [182] schlagen ein neues Verfahren zur Erzeugung von sogenannten *as-damaged* BIM-Modellen vor. Wie bereits in Abschnitt 2.3 erwähnt, beschreibt ein *as-damaged* BIM-Modell ein Bauwerk in seinem beschädigten Zustand nach einem Katastrophenereignis. Mit der von den Autoren vorgeschlagenen Methode kann eine 3D-Punktwolke, das die Fassade bzw. die Hülle eines beschädigten Stahlbetonrahmens darstellt, in ein semantisches Bauwerksdatenmodell umgewandelt werden. Die 3D-Punktwolke der beschädigten Konstruktion und das *as-built* BIM-Modell werden als Eingabe verwendet. Der in *MATLAB* entwickelte Algorithmus umfasst eine Hohlraumextraktion, Gitterkartierung, Hohlraum-Zellen-Zuordnung, Gitterlinienverformung und die digitale Rekonstruktion von Balken und Stützen auf der Grundlage der Eingabeinformationen. Der Algorithmus wurde anhand von zwei Fallstudien bewertet. In beiden Fällen wurden seismisch beschädigte Stahlbetonrahmentragwerke mit Mauerwerksausfachung aus der EERI-Datenbank [55] entnommen, dessen *as-built* BIM-Modelle nach der vorgeschlagenen Methode von *Ma et al.* [114] manuell erstellt wurden. Die 3D-Punktwolken der beschädigten Konstruktionen wurden unter Verwendung einer Laserscanning-Emulationssoftware künstlich erzeugt. Die erzielten Ergebnisse der Fallstudien zeigen, dass die Versagensform und die Position der Balken und Stützen in beiden Fällen mit einer Genauigkeit von etwa 80% erfolgreich rekonstruiert werden konnten.

Zur Erweiterung des *Scan-to-BIM*-Verfahrens von *Zeibak et al.* [182] schlagen *Bloch et al.* [25] eine neue Methode zur Extraktion von strukturellen und semantischen Informationen aus dem Innenbereich von seismisch beschädigten Stahlbetonrahmenkonstruktionen vor. Wie in [182] wird bei der vorgeschlagenen Methode von *Bloch et al.* [25] das *as-built* BIM-Modell des Tragwerkes als Eingabe verwendet. Vor dem Eintreten des Katastrophenereignisses werden zahlreiche numerische Kollapssimulationen für das Tragwerk unter starker seismischer Belastung durchgeführt und die daraus resultierenden Schadensmodelle in einer Datenbank gespeichert. Da die Kollapssimulationen vor dem Katastrophenereignis durchgeführt werden, stehen die Ergebnisse dieser zeitaufwändigen Arbeit im Katastrophenfall sofort in der Datenbank zur Verfügung. Das generierte *as-damaged* BIM-Modell nach der Methode von *Zeibak et al.* [182] wird mit Hilfe eines *Matching*-Algorithmus mit den Schadensmodellen aus der Datenbank verglichen. Schließlich wird das Schadensmodell mit der höchsten geometrischen Übereinstimmung identifiziert und seine Innenbauteile dem *as-damaged* BIM-Modell hinzugefügt. Zur Bewertung des Verfahrens wurde eine Fallstudie mit einem Schadensmodell aus der Datenbank durchgeführt, das den tatsächlichen Schadenszustand des Tragwerkes repräsentieren sollte. Die erzielten Ergebnisse zeigen, dass mit der vorgeschlagenen Methode von *Bloch et al.* [25] eine genaue Abschätzung des Innenbereiches von seismisch beschädigten Stahlbetonrahmenkonstruktionen möglich sein kann.

Die Literaturübersicht von *Khanmohammadi et al.* [98] bietet einen umfassenden Überblick über den aktuellen Forschungsstand zu BIM-Anwendungen im Bereich der Katastrophenvorsorge und -nachsorge. Daraus wird deutlich, dass die Anwendung der BIM-Methode im Kontext von Katastrophen bisher nur unzureichend erforscht wurde, was auf einen erheblichen Bedarf an weiterführender Forschung in diesem Bereich hinweist. Die vorhandenen Publikationen konzentrieren sich vor allem auf die Anwendung der BIM-Technologie in Brand-, Hochwasser- und Erdbebenszenarien. Hervorzuheben ist dabei die Untersuchung von *Bloch et al.* [25], die sich zuletzt mit dem Einsatz der BIM-Methode zur Unterstützung von Such- und Rettungsmaßnahmen befasst hat.

2.4 Numerische Kollapssimulationen

Die numerische Kollapssimulation ist eine fortschrittliche Methode aus dem Bereich der numerischen Mechanik, die es ermöglicht, das Verhalten von Bauwerken unter extremen Belastungen wie z. B. einem Erdbeben, zu simulieren und zu analysieren. Diese Simulationen basieren auf der Anwendung physikalischer Gesetze, Computermodellen und numerischen Methoden wie der Finite-Elemente-Methode (FEM) oder der Angewandten-Elemente-Methode (AEM). Durch die Simulation von Einsturzereignissen könnten Rettungskräfte die Auswirkungen extremer Belastungen auf Gebäude objektiv beurteilen und geeignete Maßnahmen ergreifen, um Leben zu retten und Schäden zu minimieren. Darüber hinaus können numerische Kollapssimulationen auch dazu beitragen, die Tragstruktur eines Gebäudes zu verbessern, um deren Widerstandsfähigkeit gegenüber extremer Belastung zu erhöhen.
In den letzten Jahren wurden mehrere wissenschaftliche Arbeiten veröffentlicht, die sich mit dem Einsturz von Bauwerken unter seismischen oder außergewöhnlichen Belastungen befassen. Diese Forschung hat zur Entwicklung verschiedener Methoden für die Vorhersage von Trümmerstrukturen geführt. In diesem Abschnitt wird der aktuelle Stand der Forschung auf diesem Gebiet kurz zusammengefasst.

2.4.1 Numerische Verfahren

Sediek et al. [150] haben durch numerische Simulationen mit der AEM ein Verfahren entwickelt, um das Ausmaß der Trümmerstruktur bei Gebäudeeinstürzen abzuschätzen. Dazu wurde ein lineares Regressionsmodell erstellt, welches das Ausmaß des Trümmerfeldes in Abhängigkeit von der Gebäudehöhe vorhersagt. Zusätzlich wurde ein *Deep-Neural-Network* (DNN) eingesetzt, um die Einsturzmode von Stahlbetonrahmentragwerken vorherzusagen. Mit einer Vorhersagegenauigkeit von 92% für die Trainingsdatensätze und 82% für die Testdatensätze zeigt das neuronale Netzwerk der Autoren insgesamt eine solide Leistungsfähigkeit.
Domaneschi et al. [50] stellen eine vereinfachte Methodik zur Bewertung der Trümmerbildung bei eingestürzten Mauerwerksgebäuden vor, die auf validierten Computersimulationen verschiedener Einsturzszenarien basiert. Anhand der Ergebnisse der numerischen Simulationen mit der AEM haben die Autoren eine vereinfachte Formel zur Abschätzung der Trümmerfläche und des -volumens von seismisch beschädigten Mauerwerksbauten entwickelt. Um die Zuverlässigkeit und Genauigkeit der vorgeschlagenen Formel zu verifizieren, wurden im Rahmen einer Fallstudie die simulierten Trümmerstrukturen eines virtuellen Stadtviertels, bestehend aus 27 Gebäuden, analysiert. Die Autoren stellen fest, dass die vorgeschlagene Formel eine hohe Zuverlässigkeit bei der Vorhersage der Trümmerbildung von Mauerwerksgebäuden aufweist.
Die Arbeit von *Grunwald et al.* [77] untersucht die Leistungsfähigkeit und Genauigkeit von der FEM und AEM in Bezug auf Kollapssimulationen von Stahlbetonkonstruktionen. *Grunwald et al.* [77] kommen zu dem Schluss, dass die FEM zwar dazu neigt, erste Einsturzmoden zuverlässig zu bestimmen, jedoch nicht für eine präzise Simulation der endgültigen Einsturzform auf Gebäudeebene geeignet ist. Basierend auf den erzielten Ergebnissen betrachten die Autoren die AEM als eine zuverlässigere und genauere Methode zur Simulation des progressiven Einsturzes von Stahlbetonkonstruktionen. Weitere Forschungsarbeiten, die sich mit der AEM zur Simulation des progressiven Einsturzes von Gebäuden befassen, sind in [11, 56, 112, 118, 148] zu finden.

2.4.2 Physik-Engine-basierte Starrkörperdynamik

Fita et al. [63] schlagen eine vereinfachte Methodik für die Einsturzsimulation historischer Mauerwerksbauten unter Erdbebenbelastung vor, die sich an Anwender ohne Fachkenntnisse im Erdbebeningenieurwesen richtet. Das Tool basiert auf der Starrkörperdynamik und wurde mit der

Physik-Engine *Bullet* [45] entwickelt. Die vorgeschlagene Methode von *Fita et al.* [63] ermöglicht eine einfache und schnelle Simulation der Auswirkungen von Erdbeben auf historische Mauerwerksgebäude. Darüber hinaus kann sie zur Rekonstruktion vergangener Katastrophenereignisse oder zur Vorbeugung künftiger Bauwerksschäden aufgrund von Erdbeben eingesetzt werden.
Oliver und *Kostack* [128] entwickelten das Add-on *Bullet-Constraint-Builder* (BCB) für *Blender* [23] zur Simulation von Gebäudeeinstürzen mit der Methode der Starrkörperdynamik. *Blender* [23] dient dabei als grafische Benutzeroberfläche und interaktive Plattform für die Simulationssteuerung, während die in *Blender* integrierte Physik-Engine *Bullet* [45] für die Simulation von starren und weichen Körpern eingesetzt wird. Der von *Oliver* und *Kostack* [128] entwickelte BCB automatisiert die Generierung von Zwangsbedingungen (engl. *constraints*) zwischen Starrkörperelementen und ermöglicht Kollapssimulationen unter Berücksichtigung der mechanischen Eigenschaften der Materialien. In [128] sind die theoretischen Grundlagen der Starrkörperdynamik, ihre optimierte Herleitung und die Funktionalität des BCBs ausführlich zusammengefasst. Die erzielten Validierungsergebnisse mit dem BCB-Add-on in [171] zeigen vielversprechende Möglichkeiten für die Durchführung kostengünstiger und schneller Kollapssimulationen komplexer Gebäude mit einer *Open-Source*-Software. Wie in [100, 101] dargestellt, ist eine realistische Vorhersage der Einsturzform mit dem BCB-Add-on möglich. Es ist jedoch zu beachten, dass mit dem BCB das Materialverhalten der Bauteile in vereinfachter Form berücksichtigt wird, was einen gewissen Spielraum für Ungenauigkeiten in der Berechnung lässt.
Zheng et al. [185] schlagen eine Kombination aus der FEM- und Physik-Engine-Technologie als neues Rahmenkonzept für die Simulation von Gebäudeeinstürzen vor. Die Autoren stellen fest, dass die FEM zuverlässige Ergebnisse für kleine strukturelle Verformungen liefert, während die Physik-Engine in Kombination mit dem BCB [128] besser für die Simulation großer Verformungen geeignet ist. Laut *Grunwald et al.* [77] ist die FEM nicht in der Lage, eine realistische Einsturzform zu prognostizieren. Aus diesem Grund wird von *Zheng et al.* [185] eine Kombination aus der FEM und der Physik-Engine-basierten Starrkörperdynamik vorgeschlagen, um einen Gebäudeeinsturz sowohl im Bereich kleiner als auch großer Verformungen realistisch zu simulieren. Die Genauigkeit dieses hybriden Ansatzes wurde durch Experimente und Fallstudien validiert. Dieser innovativer Ansatz wurde von *Lu et al.* [110] angewendet, um den Einsturz eines Wohngebäudes in Surfside, Florida, der sich am 24. Juni 2021 ereignete, als Teil einer vorläufigen Analyse zu simulieren. Die erzielten Simulationsergebnisse weisen eine hohe Übereinstimmung mit der tatsächlichen Einsturzform auf. Die Autoren weisen jedoch darauf hin, dass ihre Analysen auf Annahmen und Vereinfachungen beruhen und nicht für forensische oder technische Zwecke verwendet werden sollten.

2.5 Zusammenfassung

In diesem Kapitel wurde der aktuelle Stand der Wissenschaft und Technik in den Bereichen Photogrammetrie, Laserscanning, Punktwolkenverarbeitung, BIM und numerische Kollapssimulation ausführlich dargestellt.

Zusammenfassend zeigt die Literaturrecherche, dass sowohl die Photogrammetrie als auch das Laserscanning in den letzten Jahren erhebliche Fortschritte gemacht haben. Moderne Ansätze wie die SfM ermöglichen eine vollautomatische photogrammetrische Datenverarbeitung zur Generierung einer Punktwolke [111, 156]. Drohnen haben die Fernerkundung aus der Luft dank ihrer Flexibilität und ihrer Fähigkeit, hochauflösende Bilder bereitzustellen, revolutioniert [33]. Mehrere Studien [59, 60, 89, 97, 167, 168, 180] unterstreichen die Vorteile des Einsatzes von Drohnen zur Erstellung von 3D-Punktwolken in Notfallsituationen. Die Schrägbildphotogrammetrie ermöglicht eine schnelle digitale Rekonstruktion von Szenen und spart im Vergleich zur traditio-

nellen Nadir-Photogrammetrie Zeit bei der Datenerfassung und -auswertung. Die Vorteile dieser Methode für die schnelle digitale Rekonstruktion von Katastrophenszenen wurden von *Verykokou et al.* [168] in einer umfassenden Studie hervorgehoben. Die gegenwärtigen Einschränkungen der Photogrammetrie umfassen die Herausforderung, verdeckte Bereiche, transparente Oberflächen oder schlecht beleuchtete Bereiche zu erfassen [15, 28, 32]. Die Literaturrecherche zum Laserscanning zeigt, dass der Einsatz von TLS und ALS mit Drohnen vielseitige Anwendungen findet. In diesem Zusammenhang wurde der Einsatz von TLS und ALS speziell für USaR-Maßnahmen im Rahmen eines Forschungsvorhabens [89, 144] untersucht und für geeignet befunden, Such- und Rettungskräfte zu unterstützen. Trotz ihrer Vorteile sind moderne Laserscanner jedoch mit hohen Kosten verbunden und erfordern komplexe Anwendungen und zeitaufwändige Nachbearbeitungen.
In [76, 103, 125, 146] finden sich verschiedene Segmentierungsmethoden, darunter kantenbasierte, regionenbasierte, modellbasierte, graphenbasierte und attributbasierte Ansätze. Die Auswahl der optimalen Segmentierungsmethode hängt dabei stark von den spezifischen Anforderungen der Szene ab. Kantenbasierte Methoden erweisen sich als effektiv für die Detektion von Übergängen zwischen verschiedenen Regionen innerhalb der Punktwolke [22]. Im Gegensatz dazu sind regionenbasierte Ansätze in der Regel genauer, da sie die Punkte auf der Grundlage ähnlicher Merkmale in homogene Regionen unterteilen [146]. Modellbasierte Methoden, insbesondere der RANSAC-Algorithmus [62], haben sich bei der Identifizierung einfacher geometrischer Strukturen innerhalb der Punktwolke als zuverlässig erwiesen [66]. Graphenbasierte Methoden hingegen zeigen ihre Stärke bei der Identifizierung von Segmenten, indem sie die Punktwolke als einen Graphen betrachten [125]. Attributbasierte Methoden, insbesondere solche, die auf dem *k-means*-Algorithmus beruhen, führen in der Regel erfolgreich Gruppierungen von ähnlichen Attributen durch. Diese Methoden eignen sich daher gut für die automatische Erkennung geometrischer Beziehungen in großen Punktdatensätzen [146].
Um das Verständnis von 3D-Punktwolken zu vertiefen, werden segmentierten Punktdaten Objektklassen bzw. Semantiken zugeordnet. Dieser Prozess ist bekannt als Punktwolkenklassifizierung oder semantische Segmentierung. Zu diesem Zweck wird in [178, 181, 183] die Anwendung von speziell entwickelten CNNs für 3D-Punktwolken vorgeschlagen. Mehrere Forscher [113, 133] haben die Anwendung solcher Netzwerke für die semantische Segmentierung von 3D-Punktwolken untersucht, mit dem Ziel, automatisch *as-built* BIM-Modelle zu erzeugen. Die Genauigkeit von CNN-basierten Vorhersagen hängt nicht nur von der Vollständigkeit und Dichte der Eingabepunktwolke ab, sondern auch in hohem Maße von der Größe und Qualität des Trainingsdatensatzes [178].
Die Literaturrecherche verdeutlicht, dass in den vergangenen 25 Jahren erhebliche Fortschritte im Bereich der algorithmischen Geometrie erzielt wurden, um aus unstrukturierten Punktdaten präzise Oberflächenmodelle zu generieren [19, 99]. In diesem Zusammenhang sind die *Delaunay-Triangulation* [26] und die α-*Shape*-Methode [54, 179] etablierte Verfahren aus der algorithmischen Geometrie, mit denen aus Punktdaten Dreiecksflächen erzeugt und an die konvexe oder konkave Form der Punktmengen angepasst werden. Der BPA [20] hingegen nutzt eine Kugel mit konstantem Radius, die sich entlang der Randkante von einem Punkt zum nächsten bewegt, um Dreiecksflächen zu erstellen. Die *Poisson*-Rekonstruktionsmethode [95] formuliert die Oberflächengeometrie als mathematisches *Poisson*-Problem und erzeugt darauf basierend ein zusammenhängendes Dreiecksnetz. Diese Methode zeichnet sich durch Robustheit gegenüber verrauschten Daten aus und ermöglicht die Erzeugung detaillierter und lückenloser Oberflächentopologien. Voxelbasierte Methoden repräsentieren die einfachste Form der digitalen Oberflächenrekonstruktion aus Punkten [82]. Hierbei werden für jeden Punkt in einem 3D-Gitter würfelförmige *Mesh*-Körper erzeugt, wobei die Voxelgröße manuell festgelegt wird. Darüber hinaus wurden in den letzten Jahren mehrere lernbasierte Rekonstruktionsmethoden entwickelt, die mit Hilfe von DL eine genaue

digitale Rekonstruktion unter Berücksichtigung geometrischer Merkmale und semantischer Informationen ermöglichen [36, 140, 153].
Die Fortschritte, die in den letzten Jahren auf dem Gebiet des *Scan-to-BIM* erzielt wurden, zeigen die vielfältigen Möglichkeiten auf, die derzeit existieren. Trotz dieser Fortschritte besteht weiterhin Forschungsbedarf, da aktuelle Methoden Schwierigkeiten bei der semantisch detaillierten Erstellung von BIM-Modellen für existierende Gebäude aufweisen [46]. Moderne Laserscanner, insbesondere terrestrische und luftgestützte Laserscanner, werden aufgrund ihrer Präzision bei der berührungslosen Datenerfassung vorwiegend im *Scan-to-BIM*-Verfahren eingesetzt [109]. Des Weiteren wurden in [116, 160] halbautomatische Ansätze für die digitale Rekonstruktion von Innenräumen vorgeschlagen. In diesem Zusammenhang konnte eine deutlich positive Entwicklung für die halbautomatische Erstellung von *as-built* BIM-Modellen festgestellt werden. Die Methode von *Zeibak et al.* [182] ermöglicht die halbautomatische Generierung von *as-damaged* BIM-Modellen für Katastrophenszenarien, basierend auf einem vorhandenen *as-built* BIM-Modell und einer 3D-Punktwolke. Des Weiteren geht aus der Literatur hervor, dass Verdeckungen während der Datenerfassung nach wie vor herausfordernde Aspekte in der *Scan-to-BIM* darstellen und sowohl die Objekterkennung als auch die Genauigkeit der digitalen Rekonstruktion beeinträchtigen können. Obwohl die Forschung zu BIM-Anwendungen für die Katastrophenbewältigung begrenzt ist, existieren bereits Ansätze für Brand-, Hochwasser- und Erdbebenszenarien [98]. Trotz der erzielten Fortschritte besteht weiterer Forschungsbedarf, insbesondere im Hinblick auf die semantische Rekonstruktion von bestehenden Gebäuden und die effektive Anwendung der BIM-Methode im Katastrophenfall.
Die Forschungsergebnisse in [50, 77, 150] zeigen, dass die FEM und AEM nicht nur Such- und Rettungskräften eine objektive Beurteilung der Auswirkungen von Einsturzereignissen ermöglichen könnten, sondern auch das Potenzial haben, zur Verbesserung der Tragstruktur bestehender Gebäude beizutragen. Numerische Verfahren, insbesondere die AEM, wurden in mehreren Studien [11, 56, 112, 118, 148] verwendet, um das Ausmaß von Trümmerstrukturen bei Gebäudeeinstürzen abzuschätzen. Aus diesen Studien geht hervor, dass solche Methoden präzise Vorhersagen ermöglichen und die Trümmerbildung von Stahlbeton- und Mauerwerksbauten zuverlässig abschätzen können. Im Gegensatz zur FEM und AEM zeichnet sich die Physik-Engine-basierte Starrkörperdynamik durch eine schnellere Ausführung von Einsturzsimulationen aus. Dies ermöglicht ihre potenzielle Anwendung bei der Abschätzung der Trümmerbildung von eingestürzten Gebäuden zur Unterstützung von USaR-Maßnahmen [128]. Zu den wichtigsten Einschränkungen der Physik-Engine-basierten Starrkörperdynamik im Vergleich zur FEM und AEM gehören die begrenzte Genauigkeit bei kleinen Verformungen und die fehlende Berücksichtigung des nichtlinearen Materialverhaltens. Um diese Anwendungsgrenzen zu überwinden, wird von *Zheng et al.* [185] ein innovativer hybrider Ansatz vorgeschlagen, der die FEM mit der Physik-Engine-basierten Starrkörperdynamik kombiniert. Dabei wird die FEM für die Berechnung von kleinen Bauteilverformungen bis zum Einsturzbeginn und die Physik-Engine-basierte Starrkörperdynamik für die Abschätzung großer Verformungen bis zum eingestürzten Zustand verwendet.

Kapitel 3

Methodik

In diesem Kapitel werden die in dieser Dissertation entwickelten Methoden zur modellunabhängigen Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmentragwerken vorgestellt. Um einen besseren Überblick über die entwickelten Methoden und die damit verbundenen Arbeitsschritte zu erhalten, wird zunächst in Abschnitt 3.1 die Strategie der algorithmischen Datenverarbeitung vorgestellt und kurz erläutert. Abschnitt 3.2 widmet sich der Verwaltung der objektorientierten Daten durch die Definition einer Datenstruktur. Anschließend werden die einzelnen Arbeitsschritte der Strategie aus Abschnitt 3.1 im Detail behandelt, d. h. die Vorverarbeitung (Abschnitt 3.3), die Segmentierung (Abschnitt 3.4), die Objektklassifizierung (Abschnitt 3.5), die Materialklassifizierung (Abschnitt 3.6), die 3D-Rekonstruktion (Abschnitt 3.7 und 3.8) und die BIM-Generierung (Abschnitt 3.9).

3.1 Strategie der algorithmischen Datenverarbeitung

Die Erstellung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmentragwerken erfordert eine digitale 3D-Rekonstruktion der Gebäudekomponenten auf Basis von Punktdaten im euklidischen $\mathbb{R}^3$ und die Übertragung des generierten 3D-Modells in den BIM-Prozess. Unter Berücksichtigung der Erkenntnisse aus der vorgestellten Literaturübersicht in Kapitel 2 wurde eine Strategie zur halbautomatischen Generierung von BIM-Modellen seismisch beschädigter Stahlbetonrahmentragwerke entwickelt, die die folgenden Schritte umfasst:

1. Vorverarbeitung,
2. Segmentierung,
3. Objektklassifizierung,
4. Materialklassifizierung,
5. digitale 3D-Rekonstruktion und
6. BIM-Generierung.

Diese Strategie verfolgt einen modellunabhängigen Ansatz, bei dem zur BIM-Generierung ausschließlich eine 3D-Punktwolke verwendet wird, ohne zusätzliche Informationen über die Tragstruktur des Gebäudes. Der Schwerpunkt der vorgeschlagenen Strategie liegt daher auf der algorithmischen Datenverarbeitung von 3D-Punktwolken und Polygonnetzen. In der Abbildung 3.1 ist die Strategie der algorithmischen Datenverarbeitung mit den dazugehörigen Arbeitsschritten und Methoden schematisch dargestellt. Im Folgenden werden die in Abbildung 3.1 angegebenen Arbeitsschritte kurz erläutert:
Die Rekonstruktionsstrategie beginnt mit der digitalen Erfassung eines beschädigten Stahlbetonrahmens nach einem Erdbebenereignis mit einer ausgewählten Fernerkundungsmethode, wie

z. B. Photogrammetrie oder Laserscanning. Bei der Datenerfassung mittels Photogrammetrie oder Laserscanning muss sichergestellt werden, dass alle sichtbaren Oberflächen des Gebäudes vom Sensor erfasst werden, um eine hochauflösende und weitgehend vollständige Punktwolke der Szene zu erstellen. Wie bereits in Abschnitt 2.1 erwähnt, bieten unbemannte Drohnen eine flexible Plattform für die Datenerfassung aus der Luft. Um die Datenqualität und die Effizienz der Verarbeitung zu verbessern, wird im nächsten Schritt die erzeugte 3D-Punktwolke vorverarbeitet (Abschnitt 3.3). Dabei werden die folgenden Methoden verwendet:

1. (Geo-)Referenzierung in einem gemeinsamen Koordinatensystem,
2. Ausschneiden nicht benötigter Objekte,
3. *Downsampling* mit einer räumlichen Teilstichprobe,
4. Statistische Ausreißerentfernung (Bereinigung).

Für eine objektorientierte Datenanalyse werden anschließend aus der vorverarbeiteten Punktwolke die strukturellen und nicht-strukturellen Komponenten des erfassten Gebäudes segmentiert (Abschnitt 3.4). Zu diesem Zweck wird der RANSAC-Algorithmus [62] mit den Primitiven einer Ebene und einer Gerade als geometrische Anpassungsfunktionen eingesetzt. Eine kurze Zusammenfassung der Grundlagen der RANSAC-Methode ist in Unterabschnitt 3.4.1 angegeben. Im Zuge der Segmentierung mit RANSAC werden Punktdaten, die große ebene Oberflächen darstellen, wie z. B. Decken oder Wände, mit Ebenen in $\mathbb{R}^3$ abgetastet, während linienförmige Punktmengen, wie z. B. Träger oder Stützen, mit Geraden in $\mathbb{R}^2$ überprüft werden. Die erhaltenen Segmente werden dann in Bezug auf Form, Größe und Ausrichtung miteinander verglichen, wobei die Segmente, die zu einer Tragwerkskomponente gehören, zusammengeführt werden. Scheibenförmige Bauteile wie Mauerwerks- oder Stahlbetonwände werden aus den ebenen Rahmensegmenten mit Hilfe eines iterativen Ansatzes basierend auf der α-*Shape*-Methode in $\mathbb{R}^2$ herausgeschnitten (Unterabschnitt 3.4.2).
Nach der Segmentierung der Gebäudekomponenten werden die segmentierten Punktdaten anhand ihrer geometrischen Merkmale klassifiziert und damit strukturellen und nicht-strukturellen Objekttypen zugeordnet (Abschnitt 3.5). Das Ergebnis ist die klassifizierte 3D-Punktwolke eines seismisch beschädigten Stahlbetonrahmens. Mit Hilfe eines DL-Ansatzes werden die Materialien der Objekte klassifiziert. Dabei werden RGB-Daten als Merkmale verwendet, um Vorhersagen über das vorhandene Material zu treffen.
Die digitale 3D-Rekonstruktion der klassifizierten Punktsegmente erfolgt im nächsten Schritt mit der α-*Shape*-Methode, mit der sowohl konvexe als auch konkave Oberflächen als Polygonnetz in $\mathbb{R}^2$ erstellt werden können (Abschnitt 3.7). Um aus den zweidimensionalen Polygonnetzen dreidimensionale Volumenkörper zu erstellen, werden die Polygonflächen entlang ihrer Oberflächennormalen extrudiert ($\mathbb{R}^2 \rightarrow \mathbb{R}^3$). Die Länge der Extrusion entspricht dabei der Dicke des Bauteiles, die entweder aus den zusammengeführten Punktsegmenten bekannt ist oder mit einem geeigneten Standardwert angenommen wird.
Im letzten Arbeitsschritt werden schließlich die digital rekonstruierten Volumenkörper der Bauteile in IFC umgewandelt und mit semantischen Informationen aus der Objekt- und Materialklassifizierung angereichert.

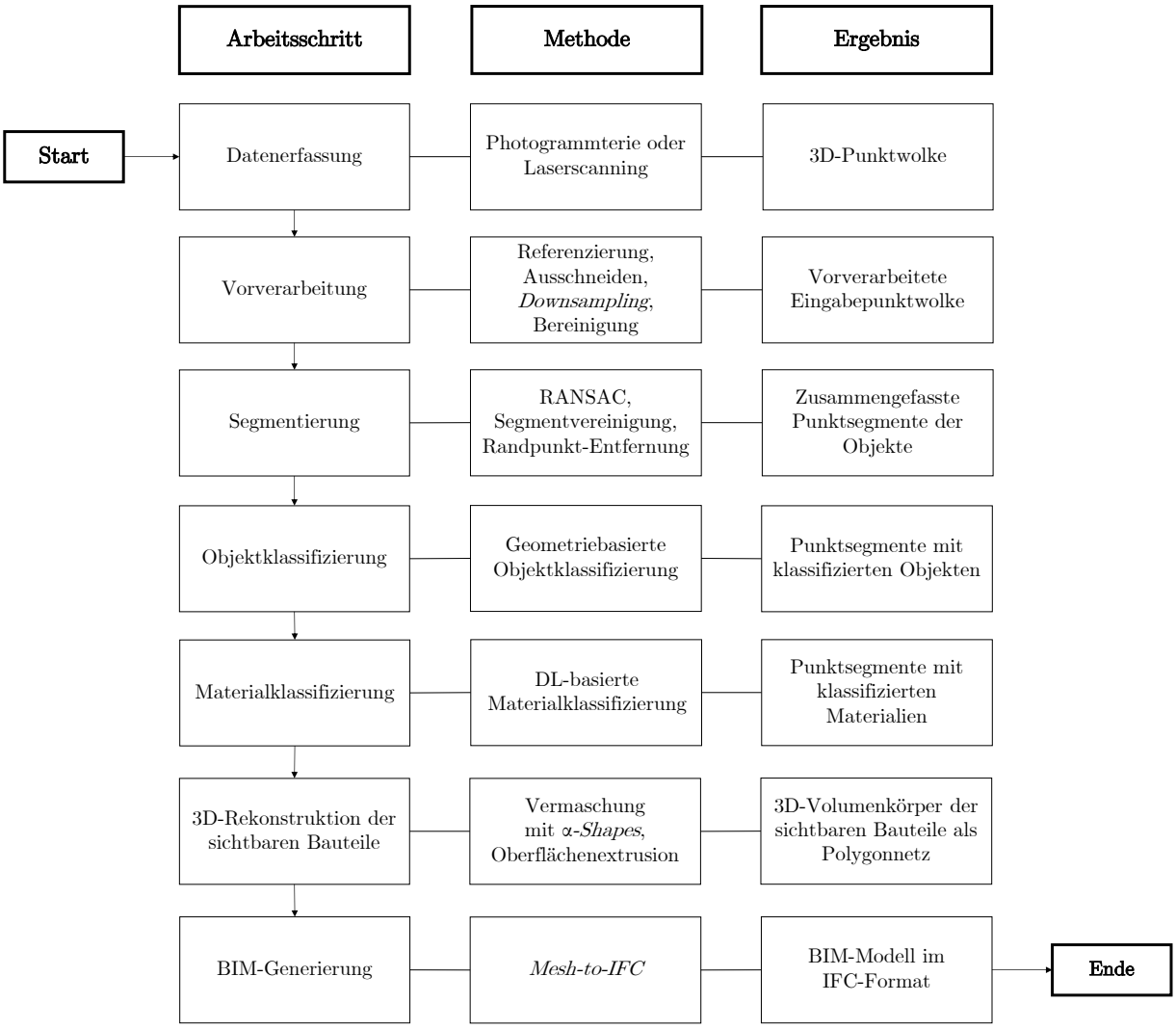


Abbildung 3.1: Schematische Darstellung der Strategie der algorithmischen Datenverarbeitung zur halbautomatischen Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen.

3.2 Datenstruktur

Die automatische Modellgenerierung auf der Grundlage einer Punktwolke erfordert die Verwendung mehrerer Daten mit unterschiedlichen Datentypen. Um auf die Daten der Objekte effizient zugreifen und diese verwalten zu können, wurde die Klasse *BuildingComponent* (BC) in der Programmiersprache Python objektorientiert implementiert. Die Datenstruktur und die Eigenschaften der implementierten Klasse sind in der Abbildung 3.2 angegeben. Im Folgenden werden die Eigenschaften eines BC-Objektes erläutert:

- ID: Eine Kennung für die eindeutige Zuordnung der Komponente im Datentyp *string*.
- *pointData*: Die 3D-Punktdaten der Komponente in der Datenstruktur *array*.
- RGB: RGB-Farbcode $\leftarrow (n, 3)$ mit $n \in \{0, 255\}$ in der Datenstruktur *list*, der zur Einfärbung der Punktdaten und Flächen der Komponente verwendet wird.
- *thickness*: Die Dicke der Komponente im Datentyp *float*.

- *type*: Die zugewiesene Objektklasse der Komponente im Datentyp *string*.
- *material*: Die zugewiesene Materialklasse der Kompoenente im Datentyp *string*.
- *hasParallel*: Der Wahrheitswert im Datentyp *boolean*, der angibt, ob die Komponente eine zugehörige parallele Punktmenge enthält oder nicht.
- *cropSec*: Die Information über den Schnittbereich der Komponente im Datentyp *integer*.

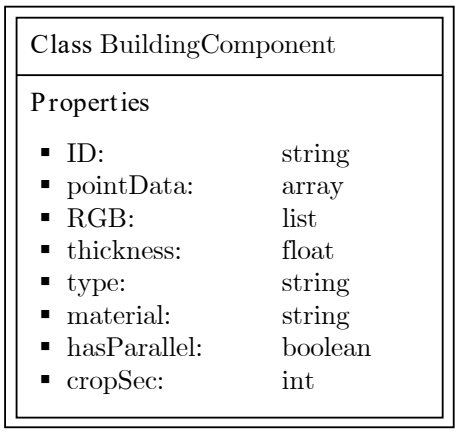


Abbildung 3.2: Datenstruktur der Klasse *BuildingComponent*.

Die oben beschriebene Klasse wird im Zuge der Datenverarbeitung verwendet, um die strukturellen und nicht-strukturellen Komponenten des digitalen Modells in Form eines BC-Objektes darzustellen. Die darin enthaltenen Daten aus der Segmentierung, Objektklassifizierung, Materialklassifizierung und 3D-Rekonstruktion dienen dabei als Grundlage für die BIM-Generierung und semantische Informationsanreicherung. Damit die BC-Objekte für die Weiterverarbeitung in anderen Programmskripten genutzt werden können, werden die Objekte in eine Python-Liste übertragen und als PKL-Datei gespeichert. Eine PKL-Datei ist eine mit der Python-Bibliothek *Pickle* erzeugte Datei, die das schnelle Speichern und Laden von Python-Objekten in einem Binärformat unter Verwendung der Serialisierung und Deserialisierung der Daten auf der Festplatte ermöglicht [5]. Eine detaillierte Beschreibung der Serialisierung von Python-Objekten mit dem *Pickle*-Modul ist in [5] zusammengefasst.

3.3 Vorverarbeitung

Nach der Datenerfassung und Erstellung der 3D-Punktwolke des Gebäudes werden die unstrukturierten Punktdaten zunächst vorverarbeitet, um zum einen die Datenqualität zu verbessern und zum anderen die Datenanalyse in weiteren Schritten effizienter zu gestalten. Wie in der Abbildung 3.1 angegeben, umfasst die Vorverarbeitung die Referenzierung, das Ausschneiden, das *Downsampling* und die Bereinigung der Punktwolke.
Die derzeitigen Möglichkeiten der Referenzierung einer Punktwolke sind in Unterabschnitt 2.2.1 zusammengefasst. Zur schnellen und genauen Referenzierung einer Punktwolke werden in diesem Arbeitsschritt die geografischen Koordinaten „Längengrad“, „Breitengrad“ und „Höhe“, die von einem GPS-Sensor erhoben werden, verwendet. Die Referenzierung von Punktwolken mit geografischen Koordinaten hat neben der Schnelligkeit und Genauigkeit den Vorteil, dass das einheitliche Referenzsystem auf Realweltkoordinaten basiert. Frei nutzbare Geodaten, z. B. von *OpenStreetMap* [130], können auf diese Weise einfach in das Referenzsystem integriert und damit die Umgebung des erfassten Gebäudes visualisiert werden.

Da bei der photogrammetrischen oder LiDAR-basierten Datenerfassung auch Punktdaten erhoben werden, die für die digitale Rekonstruktion eines Gebäudes irrelevant sind, z. B. die Umgebung, müssen diese Daten im Vorfeld entfernt werden. Zum Entfernen der Punktdaten wird die Verwendung der *crop2D*-Funktion von *CloudCompare* [40] vorgeschlagen. Mit Hilfe dieser benutzerfreundlichen Funktion können nicht benötigte Punktdaten innerhalb oder außerhalb eines manuell definierten Schnittbereiches ausgeschnitten bzw. entfernt werden. Der Schnittbereich wird dabei entweder in Form eines Rechteckes oder eines Polygonzuges definiert. Die Folge der Reduktion der Punktdaten ist, eine erhöhte Effizienz bei der Verarbeitung der Punktwolke in den nachfolgenden Schritten.
Je nach geometrischer Komplexität der erfassten Objekte und den Bedingungen, unter denen die Datenerfassung durchgeführt wurde, weisen Punktwolkendatensätze häufig unterschiedliche Punktdichten auf. Da eine hochauflösende Punktwolke nicht immer erforderlich ist, um die Form der in der Punktwolke enthaltenen Objekte eindeutig darzustellen, kann ein gleichmäßiges *Downsampling* der Punktdaten durchgeführt werden. Das *Downsampling* der Punktwolke, d. h. die Reduzierung der Punkte im Datensatz unter Beibehaltung der allgemeinen Form und Struktur der durch die Punktwolke dargestellten Objekte, erfolgt in unserem Fall mit der *Spatial-Subsample*-Filterfunktion von *CloudCompare* [40]. Mit diesem effizienten Filter werden alle Nachbarpunkte in der Punktwolke entfernt, die unter einem vom Benutzer festgelegten Mindestabstand zu ihrem Referenzpunkt liegen. Der angegebene Mindestabstand zwischen dem Nachbarpunkt und dem Referenzpunkt bestimmt dabei die Ausgabeauflösung der Punktwolke, wobei weniger Punkte beibehalten werden, je größer der Mindestabstand gewählt wird. Im ersten Schritt werden die Nachbarpunkte P_j des Referenzpunktes P_i und die Abstände $d_{i,j}$ zwischen dem Referenzpunkt und den Nachbarpunkten bestimmt. Anschließend wird überprüft, ob die ermittelten Punktabstände $d_{i,j}$ den gewählten Mindestabstand $d_{\min}$ des *Spatial-Subsample*-Filters einhalten. Wenn $d_{i,j} \geq d_{\min}$ nicht erfüllt ist, werden die betroffenen Nachbarpunkte aus dem Datensatz entfernt und das Verfahren mit einem neuen Referenzpunkt wiederholt. In der Abbildung 3.3 stellt P_1 beispielhaft den Referenzpunkt der dargestellten Punktemenge dar und die Punkte P_2 bis P_7 die dazugehörigen Nachbarpunkte, von denen nur P_2, P_3 und P_7 den Mindestabstand $d_{\min}$ zu P_1 einhalten und folglich nicht entfernt werden.

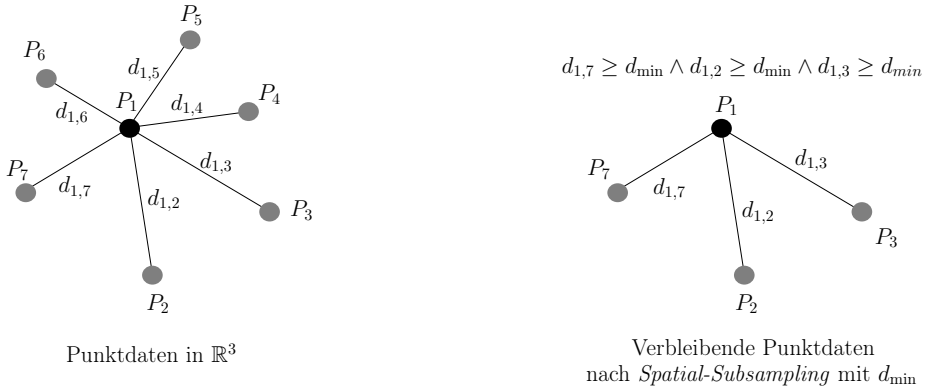


Abbildung 3.3: Bildliche Beschreibung der *Spatial-Subsample*-Filterfunktion [40] anhand eines vereinfachten Beispiels: Punktdaten in $\mathbb{R}^3$ vor (links) und nach dem *Downsampling* (rechts).

Sensorbedingte Messfehler führen in der Regel zu Ausreißern in den Punktdaten, die die tatsächliche Geometrie der Objekte bis zu einem gewissen Grad verfälschen können. LiDAR-Sensoren auf dynamischen Plattformen, wie z. B. einer unbemannten Drohne, sind besonders anfällig für die Erzeugung von Ausreißern. Diese Ausreißer können zu einer fehlerhaften Verarbeitung der

Punktdaten führen und somit die Gesamtqualität der digitalen Rekonstruktion beeinträchtigen. Daher ist es im Rahmen der Vorverarbeitung entscheidend, derartige Messfehler so weit wie möglich zu eliminieren. Zur Bereinigung der Punktwolke von Ausreißern wird der *Statistical-Outlier-Removal*-(SOR)-Filter [40] eingesetzt. Der SOR-Filter berechnet zunächst den durchschnittlichen Abstand $\bar{d}_n$ jedes Referenzpunktes P_i zu seinen Nachbarn P_j anhand der folgenden Formel:

$$\bar{d}_n = \frac{\sum_{j=1}^{k} d_{i,j}}{k}, \qquad \text{mit} \quad k \in \mathbb{N}, \tag{3.1}$$

wobei k die maximale Anzahl der berücksichtigten Nachbarpunkte und $d_{i,j}$ den Abstand der Nachbarpunkte P_j zum Referenzpunkt P_i definiert. Im nächsten Schritt wird die Standardabweichung σ_n der Punktabstände $d_{i,j}$ ermittelt:

$$\sigma_n = \sqrt{\frac{\sum_{j=1}^{k}(d_{i,j} - \bar{d}_n)^2}{k}}, \qquad \text{mit} \quad k \in \mathbb{N}. \tag{3.2}$$

Schließlich wird der statistische Abstand d_{stat} der Nachbarpunkte P_j zu P_i unter Verwendung des durchschnittlichen Abstandes $\bar{d}_n$ (Gleichung 3.1) und der Standardabweichung σ_n (Gleichung 3.2) wie folgt bestimmt:

$$d_{stat} = \bar{d}_n + \sigma_n \cdot x, \qquad \text{mit} \quad x \in \mathbb{R}^+, \tag{3.3}$$

wobei x den Multiplikator der Standardabweichung repräsentiert. Der SOR-Filter entfernt anschließend die Nachbarpunkte P_j aus dem Punktdatensatz $\mathbf{P}$, die den statistischen Abstand d_{stat} zu ihrem Referenzpunkt P_i überschreiten, d. h. für die $d_{i,j} > d_{\text{stat}}$ gilt.
Um eine fehlerhafte Ausreißerentfernung mit der oben beschriebenen Methode zu vermeiden, wird vom Autor dieser Arbeit empfohlen, den SOR-Filter mit $x = 1$ (Gleichung 3.3) anzuwenden, d. h. $d_{\text{stat}} = \bar{d}_n + \sigma_n$. Wenn der SOR-Filter mit $x = 1$ keine ausreichend bereinigte Punktwolke liefert, kann k schrittweise erhöht werden.

3.4 Segmentierung

Nach der Vorverarbeitung der Punktwolke ist die Segmentierung der Punktdaten der nächste Arbeitsschritt gemäß der in Abbildung 3.1 vorgestellten Strategie. Ziel dieses Arbeitsschrittes ist es, die in der Punktwolke enthaltenen Bauteile einer seismisch beschädigten Stahlbetonrahmenkonstruktion zuverlässig zu segmentieren. Zu den Bauteilen, die im Segmentierungsprozess berücksichtigt werden, gehören:

(i) Decken,

(ii) Wände,

(iii) Träger,

(iv) und Stützen.

Im Folgenden werden die digitalen Methoden vorgestellt, die zur Extraktion der oben genannten Bauteile aus einer 3D-Punktwolke entwickelt wurden.

3.4.1 Ebene Punktmengen

Der Prozess der Segmentierung beginnt mit der Extraktion ebener Punktmengen aus der vorverarbeiteten Punktwolke, die die Oberflächen von Decken, Wänden oder Rahmen einer seismisch beschädigten Stahlbetonrahmenkonstruktion repräsentieren können. In diesem Unterabschnitt werden die Grundlagen des gewählten Algorithmus, die Segmentierungsmethode und das Verfahren zur Identifizierung und Vereinigung von Segmenten, die zu einem Bauteil gehören, vorgestellt.

3.4.1.1 Grundlagen des RANSAC-Algorithmus

Aufgrund seiner Einfachheit und Robustheit wird der RANSAC-Algorithmus [62] zur Segmentierung von ebenen Punktdaten ausgewählt. Im Allgemeinen bestimmt der RANSAC-Algorithmus [62] wie gut ein geometrisches Modell M an einen Punktdatensatz $\mathbf{P}$ angepasst werden kann. Dabei besteht der Datensatz $\mathbf{P}$ aus n Punkten im euklidischen $\mathbb{R}^2$ oder $\mathbb{R}^3$. Im ersten Schritt wird das geometrische Modell M_k mit m zufälligen Punkten aus $\mathbf{P}$ erstellt:

$$m \in N_k \rightarrow M_k, \qquad \text{mit} \quad k \in \mathbb{N} \ \text{ und } \ N_k \subseteq \mathbf{P}, \tag{3.4}$$

wobei der Parameter m die Anzahl der Punkte definiert, die für die geometrische Beschreibung von M_k erforderlich sind. Sei M_k eine Ebene, so sind mindestens 3 Punkte ($m \leftarrow 3$) erforderlich, um die Geometrie von M_k mathematisch zu beschreiben. Die übereinstimmenden Punkte der Punktwolke $\mathbf{P}$ mit M_k, d. h. $P_k(M_k) \subseteq \mathbf{P}$, werden als Nicht-Ausreißer (engl. *inlier*) und die nicht übereinstimmenden Punkte als Ausreißer (engl. *outlier*) klassifiziert. Die Prüfung der Anpassungsgüte von M_k an $\mathbf{P}$ mit $N_k \rightarrow M_k$ wird innerhalb einer Iterationsschleife so lange durchgeführt, bis die Wahrscheinlichkeit, eine bessere Übereinstimmung mit $\mathbf{P}$ zu finden, unter einen vordefinierten Schwellenwert fällt. Standardmäßig ist das Abbruchkriterium für RANSAC die Anzahl der Iterationen κ mit $\kappa \in \mathbb{N}$, die definiert, wie oft eine zufällige Teilmenge N_k aus $\mathbf{P}$ abgetastet wird. Die Anzahl der Iterationen κ kann in Abhängigkeit von der Wahrscheinlichkeit ρ, den Ausreißeranteil ϵ und den gewählten Punkten m so festgelegt werden, dass jede der abgetasteten Teilmengen N_k mindestens einen Ausreißer mit der Wahrscheinlichkeit ρ enthält. Hierzu wird von *Fischler* und *Bolles* [62] die folgende Formel vorgeschlagen:

$$\kappa \geq \frac{\log(1-\rho)}{\log(1-(1-\epsilon)^m)}, \qquad \text{mit} \quad \{\rho, \epsilon\} \in \{x \in \mathbb{R}^+ \mid 0 \leq x \leq 1\} \ \text{ und } \ \{\kappa, m\} \in \mathbb{N}. \tag{3.5}$$

Da in der Regel der prozentuale Anteil der Ausreißer ϵ in einem Datensatz nicht bekannt ist, kann κ über einen iterativen Ansatz mit Hilfe der Konsensmenge C (engl. *consensus set*) bestimmt werden. Die Konsensmenge C ist der Anteil der Daten aus $\mathbf{P}$, der mit dem geometrischen Modell M_k übereinstimmt und somit frei von Ausreißern ist. Mathematisch wird die Konsensmenge C wie folgt ausgedrückt:

$$C_k = (1-\epsilon_k) \cdot n, \qquad \text{mit} \quad k \in \mathbb{N} \ \text{ und } \ C_k \subseteq \mathbf{P}, \tag{3.6}$$

wobei n die Gesamtzahl der Datenpunkte in $\mathbf{P}$ darstellt und ϵ_k den Ausreißeranteil in der k-ten Iteration. Um die Anzahl der Iterationen κ anhand der Konsensmenge automatisch zu bestimmen, wird der RANSAC-Algorithmus zunächst mit einer Iteration, d. h. $k = 1$, ausgeführt. Konnte unter der Annahme $k = 1$ keine Konsensmenge gefunden werden, wird RANSAC mit $k \leftarrow k + 1$ so lange ausgeführt, bis eine Menge von Datenpunkten eine Konsensmenge bildet. Anschließend wird durch Umstellen von Gleichung 3.6 aus der Gesamtzahl der Datenpunkte n und der Anzahl der Punkte in C_k der Ausreißeranteil ϵ_k ermittelt:

$$\epsilon_k = 1 - \frac{C_k}{n}. \tag{3.7}$$

Schließlich wird der ermittelte Ausreißeranteil ϵ_k aus Gleichung 3.7 in Gleichung 3.5 eingesetzt und damit die erforderliche Anzahl der Iterationen κ berechnet.

3.4.1.2 Segmentierungsmethode

Für eine effiziente und automatische Formdetektion wird in dieser Arbeit die vorgeschlagene RANSAC-Methode von *Schnabel et. al* [149] verwendet, das als Plugin im Python-Modul *cloudComPy* [40] implementiert ist. Im Vergleich zu herkömmlichen RANSAC-Implementierungen, bei der alle Parameterwerte manuell festgelegt werden müssen, berücksichtigt der Ansatz von *Schnabel et. al* [149] eine automatische Bestimmung der optimalen Parameterwerte auf der Grundlage der geometrischen Eigenschaften der Eingabepunktwolke. Darüber hinaus kann mit der integrierten *bitmap*-Methode von *Schnabel et. al* [149] überprüft werden, ob die Punktmenge des detektierten Segments in Bezug auf die räumliche Punkteverteilung kontinuierlich oder diskontinuierlich ist. Dazu wird der Abstand d_p zwischen zwei Punktgruppen im Bereich der Diskontinuität mit dem Grenzwert d_{bitmap} verglichen, der anhand der Eingabepunktwolke automatisch ermittelt werden kann. Für den Fall: $d_p > d_{\text{bitmap}}$, werden die Punktgruppen in einzelne Segmente aufgeteilt. Die Abbildung 3.4 (links) zeigt beispielsweise das mit RANSAC ermittelte Segment S_i, das in seiner Punktmenge $\mathbf{P}(S_i)$ eine Diskontinuität mit dem Abstand d_p aufweist.

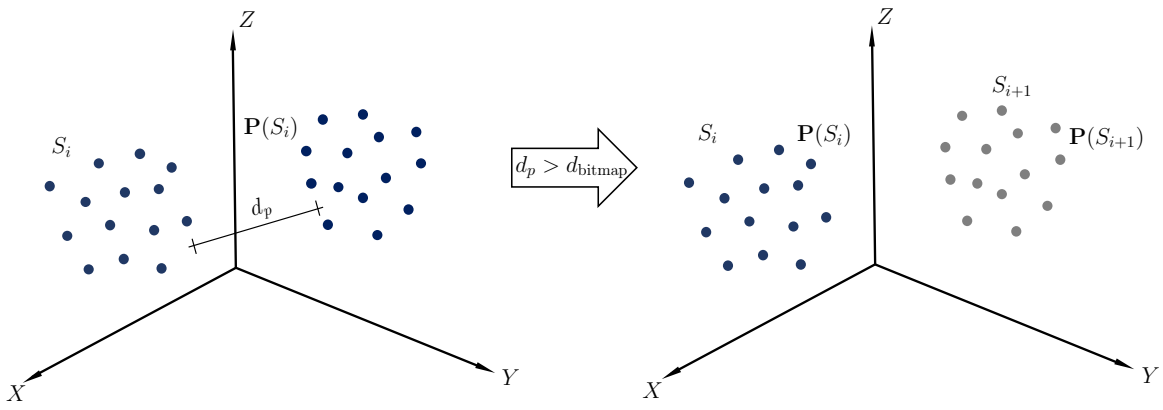


Abbildung 3.4: Unterteilung des Segments S_i (links) in die Segmente S_i und S_{i+1} (rechts) mit der *bitmap*-Methode von *Schnabel et. al* [149].

Da d_p den Grenzwert d_{bitmap} überschreitet, wird S_i in die Segmente S_i und S_{i+1} mit den darin enthaltenen Punktmengen $\mathbf{P}(S_i)$ und $\mathbf{P}(S_{i+1})$ aufgeteilt. Aus der Abbildung 3.4 geht hervor, dass die Verwendung der *bitmap*-Methode in Kombination mit RANSAC besonders nützlich für die individuelle Segmentierung von Objekten ist, deren Oberflächen in einer Ebene liegen, aber nicht miteinander verbunden sind. Für eine detaillierte Beschreibung der in dieser Arbeit verwendeten RANSAC-Methode wird der Leser auf den Beitrag von *Schnabel et. al* [149] verwiesen.

3.4.1.3 Vereinigung von Objektsegmenten

Wie bereits oben erwähnt, wird das Python-Modul *cloudComPy* [40] mit dem implementierten RANSAC-Verfahren von *Schnabel et. al* [149] für die Segmentierung der ebenen Punktmengen in $\mathbb{R}^3$ eingesetzt. Die daraus resultierenden Punktdaten der Segmente werden in dem dreidimensionalen *array* C gespeichert. Um die segmentierten Oberflächen eines Objektes zu identifizieren und zu vereinen, wird im nächsten Schritt geprüft, ob die ebenen Punktmengen in C parallel zueinander ausgerichtet sind und ob der orthogonale Abstand zwischen den Segmenten innerhalb

eines festgelegten Intervalles liegt. Die in diesem Zusammenhang verwendete Methode ist in der Abbildung 3.5 dargestellt und wird im Folgenden erläutert:

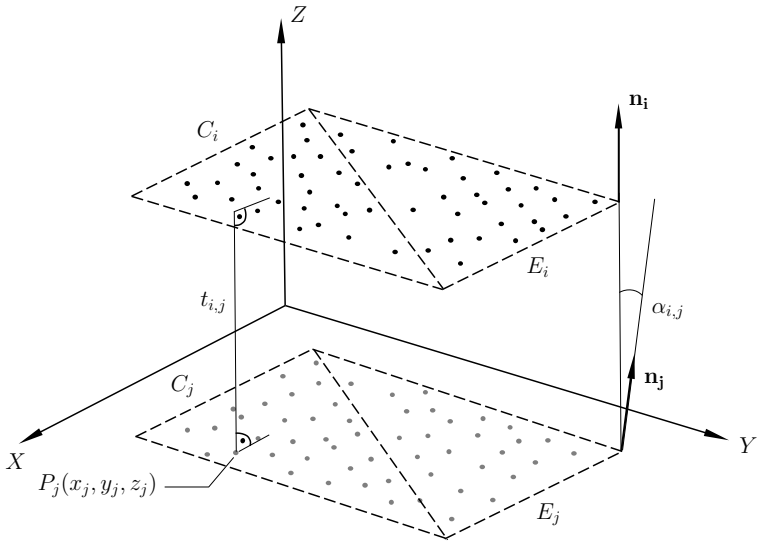


Abbildung 3.5: Bestimmung der Winkeldifferenz $\alpha_{i,j}$ und des orthogonalen Abstandes $t_{i,j}$ zwischen den Segmenten C_i und C_j.

Zwei Segmente, C_i und C_j, sind parallel, wenn die Winkeldifferenz $\alpha_{i,j}$ zwischen den Normalenvektoren der Segmente annähernd Null ist, d. h. $\alpha_{i,j} \approx 0$. Die Winkeldifferenz $\alpha_{i,j}$ wird mit der folgenden Formel bestimmt:

$$\alpha_{i,j} = \arccos\left(\frac{\mathbf{n_i} \cdot \mathbf{n_j}}{|\mathbf{n_i}||\mathbf{n_j}|}\right), \qquad \text{mit} \quad \{i,j\} \in \mathbb{N}, \tag{3.8}$$

wobei $\mathbf{n_i}$ und $\mathbf{n_j}$ die Normalenvektoren der Segmente C_i und C_j sind. Um die Normalenvektoren zu bestimmen, werden die Ebenen E_i und E_j an die jeweiligen Segmente angepasst und der Normalenvektor aus der entsprechenden Ebenengleichung in Koordinatenform extrahiert:

$$a_k \cdot x + b_k \cdot y + c_k \cdot z = d_k \quad \rightarrow \quad \mathbf{n_k} = \begin{pmatrix} a_k \\ b_k \\ c_k \end{pmatrix}, \qquad \text{mit} \quad k = i \vee k = j. \tag{3.9}$$

Da bei der Erstellung einer 3D-Punktwolke mit geometrischen Abweichungen aufgrund von Messungenauigkeiten zu rechnen ist, wird $C_i \parallel C_j$ als erfüllt angesehen, wenn $\alpha_{i,j} \leq \alpha_{\max}$ gilt. Der orthogonale Abstand $t_{i,j}$ zwischen den parallelen Segmenten C_i und C_j wird durch das Einsetzen eines zufälligen Punktes $P_j(x_j, y_j, z_j)$ aus C_j in die Ebenenfunktion von C_i ermittelt:

$$t_{i,j} = a_i \cdot x_j + b_i \cdot y_j + c_i \cdot z_j - d_i, \qquad \text{mit} \quad \{i,j\} \in \mathbb{N}. \tag{3.10}$$

Wenn der Abstand $t_{i,j}$ im Intervall $t_{i,j} \in \{x \in \mathbb{R}^+ \mid t_{\min} \leq x \leq t_{\max}\}$ liegt, werden die Segmente C_i und C_j vereint, d. h. $C_i \cup C_j \rightarrow C_{i,j}$. Die Grenzen des Intervalles, $t_{\min}$ und $t_{\max}$, basieren dabei auf den minimalen und maximalen Bauteildicken, die bei der erfassten Stahlbetonrahmenkonstruktion zu erwarten sind.

Die relevanten Daten jedes verarbeiteten Segments C_i werden in einem BC-Objekt (Abbildung 3.2) gespeichert. Für den Fall:

$$C_i \parallel C_j \quad \text{und} \quad t_{\min} \leq t_{i,j} \leq t_{\max}, \tag{3.11}$$

erhält das BC-Objekt:

(i) den Dateinamen mit dem Laufparameter i als ID,
(ii) die Punktdaten des vereinten Segments $C_{i,j}$,
(iii) die Komponentendicke $t_{i,j}$ und
(iv) den *boolean hasParallel* $\leftarrow$ *True*.

Andernfalls werden der Dateiname, die Punktdaten des Segments C_i, *hasParallel* $\leftarrow$ *False* und $t_{i,j} \leftarrow 0$ in das BC-Objekt hinzugefügt. Der oben beschriebene Arbeitsablauf der ebenen Punktsegmentierung wurde vom Autor dieser Arbeit in Python implementiert. Die grundlegenden Schritte der vorgestellten Methode sind in Algorithmus 1 in Form eines Pseudocodes zusammengefasst.

3.4.2 Randpunkt-Entfernung

Mit der Methode aus Unterabschnitt 3.4.1 können ebene Oberflächen von Decken, Wänden und Rahmen aus der 3D-Punktwolke eines Stahlbetonrahmentragwerkes segmentiert werden. Im Vergleich zu Decken und Wänden setzt sich die Oberfläche eines Rahmens jedoch aus mehreren Bauteilen zusammen:

(i) Träger,
(ii) Stützen
(iii) und Wände.

In der Abbildung 3.6 sind die oben genannten strukturellen Komponenten eines Stahlbetonrahmens in der Vorderansicht dargestellt. Damit die Gebäudekomponenten individuell rekonstruiert werden können, müssen die Punktdaten der Bauteile (i) bis (iii) aus den Rahmensegmenten extrahiert werden. In diesem Unterabschnitt wird die vom Autor dieser Arbeit entwickelte Methode der Randpunkt-Entfernung (RE) vorgestellt, mit der Punktdaten von Wandflächen aus einem Rahmensegment (Abbildung 3.6 (iii)) herausgeschnitten werden können.

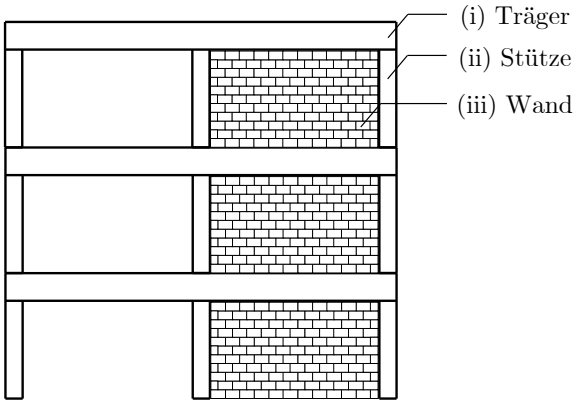


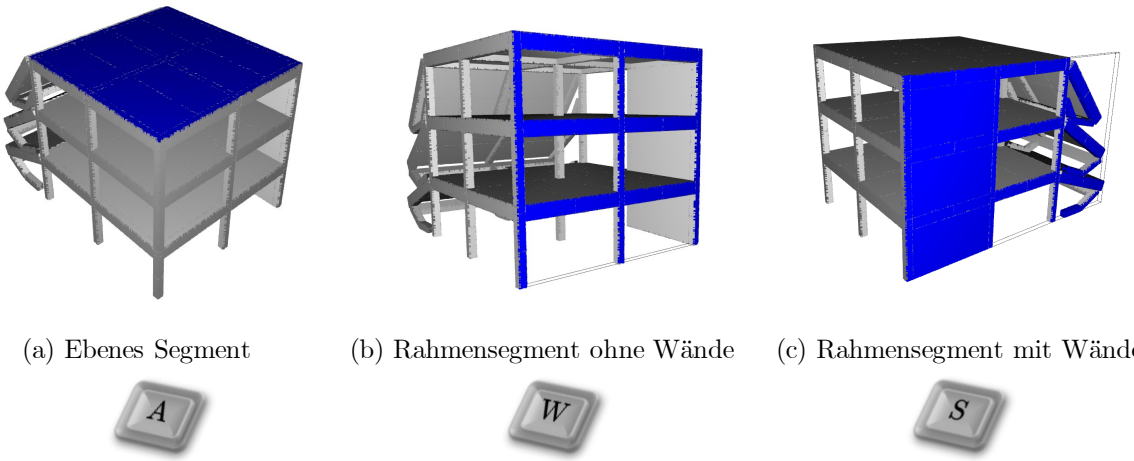
Abbildung 3.6: Strukturelle Komponenten eines Stahlbetonrahmens.

3.4.2.1 Filterung der BC-Objekte

Zunächst müssen die BC-Objekte der Rahmen, die Wandpunktdaten enthalten, in eine separate PKL-Datei übertragen werden, damit sie als Eingabe für die RE-Methode verwendet werden können. Um die Übertragung der Daten zu vereinfachen, wird eine grafische Benutzeroberfläche (engl. *Graphical User Interface*, GUI) verwendet, die vom Autor dieser Arbeit mit der Python-Bibliothek *Open3D* [186] implementiert wurde. Die GUI visualisiert sukzessive die identifizierten Punktsegmente aus Unterabschnitt 3.4.1 in der Gesamtpunktwolke, wobei der Benutzer das BC-Objekt des farblich hervorgehobenen Segments schnell und einfach per Knopfdruck in eine separate PKL-Datei übertragen kann. Dabei gibt es folgende Auswahlmöglichkeiten:

- *Button D*: Löschen des Segments,
- *Button A*: Übertragung des ebenen Segments in die PKL-Datei I,
- *Button W*: Übertragung des Rahmensegments ohne Wände in die PKL-Datei II oder
- *Button S*: Übertragung des Rahmensegments mit Wänden in die PKL-Datei III,

Die Abbildung 3.7 veranschaulicht beispielhaft den Prozess der Filterung mit Hilfe der implementierten GUI. Dabei werden die BC-Objekte der ebenen Segmente, der Rahmensegmente ohne Wände und der Rahmensegmente mit Wänden über die entsprechenden Schaltflächen herausgefiltert.



(a) Ebenes Segment (b) Rahmensegment ohne Wände (c) Rahmensegment mit Wänden

Abbildung 3.7: Darstellung der automatisch ausgewählten Punktsegmente (in Blau) eines synthetischen Datensatzes in der implementierten GUI, dessen BC-Objekte per Knopfdruck herausgefiltert werden können.

3.4.2.2 Rotation der Segmente in die XY-Ebene

Die vorgeschlagene Methode beginnt mit dem automatischen Einlesen der BC-Objekte der Rahmen aus der zuvor erstellten PKL-Datei III in die Variable Ψ. Da bei dieser Methode die Datenverarbeitung in $\mathbb{R}^2$ erfolgt, müssen die 3D-Punktdaten $P_i(x_i, y_i, z_i)$ der Rahmen in der XY-Ebene ausgerichtet werden. Die Ausrichtung der Segmente in der XY-Ebene erfolgt mit Hilfe der *Rodrigues'schen Rotationsformel*. Die *Rodrigues'sche Rotationsformel* gibt die Rotationsmatrix $\mathbf{v_{rot}}$ eines Vektors $\mathbf{v}$ an, der um die Drehachse $\mathbf{k}$ mit dem Winkel θ rotiert wird. Die entsprechende Formel lautet:

$$\mathbf{v_{rot}} = \mathbf{v}\cos\theta + (\mathbf{k} \times \mathbf{v})\sin\theta + \mathbf{k}(\mathbf{k} \cdot \mathbf{v})(1 - \cos\theta), \tag{3.12}$$

wobei $\mathbf{v}$ der ursprüngliche Vektor, $\mathbf{v_{rot}}$ der gedrehte Vektor, $\mathbf{k}$ die Drehachse und θ der Drehwinkel ist. Um den normierten Vektor $\mathbf{a}$ mit dem normierten Vektor $\mathbf{b}$ in Übereinstimmung zu bringen, muss $\mathbf{a}$ um $k = \frac{(\mathbf{a}+\mathbf{b})}{2}$ mit dem Winkel $\theta = \pi$ rotiert werden. Das Einsetzen von $k = \frac{(\mathbf{a}+\mathbf{b})}{2}$ und $\theta = \pi$ in Gleichung 3.12 ergibt die Formel zur Bestimmung der Rotationsmatrix R von zwei normierten Vektoren:

$$R = 2\,\frac{(\mathbf{a}+\mathbf{b})(\mathbf{a}+\mathbf{b})^T}{(\mathbf{a}+\mathbf{b})^T(\mathbf{a}+\mathbf{b})} - \mathbf{I}, \tag{3.13}$$

wobei $\mathbf{I}$ eine Einheitsmatrix mit der Dimension (3×3) repräsentiert. Um die Punktdaten der BC-Objekte $P_i \leftarrow \Psi_i$ in der XY-Ebene auszurichten, werden die folgenden Vektoren in Gleichung 3.13 eingesetzt:

$$\mathbf{a} = \mathbf{n_i} \tag{3.14}$$

und

$$\mathbf{b} = \mathbf{n_{xy}} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \tag{3.15}$$

wobei $\mathbf{n_i}$ der Normalenvektor der an die Punktdaten P_i angepassten Ebene E_i und $\mathbf{n_{xy}}$ der Normalenvektor der XY-Ebene ist (Abbildung 3.8). Schließlich werden die Punktdaten P_i auf der Grundlage der Rotationsmatrix R um den Mittelpunkt P_i^c in die XY-Ebene rotiert. Im Folgenden werden die in der XY-Ebene ausgerichteten Punktdaten als $P_{i,xy}$ bezeichnet. Eine bildliche Beschreibung dieser Methode ist in der Abbildung 3.8 dargestellt.

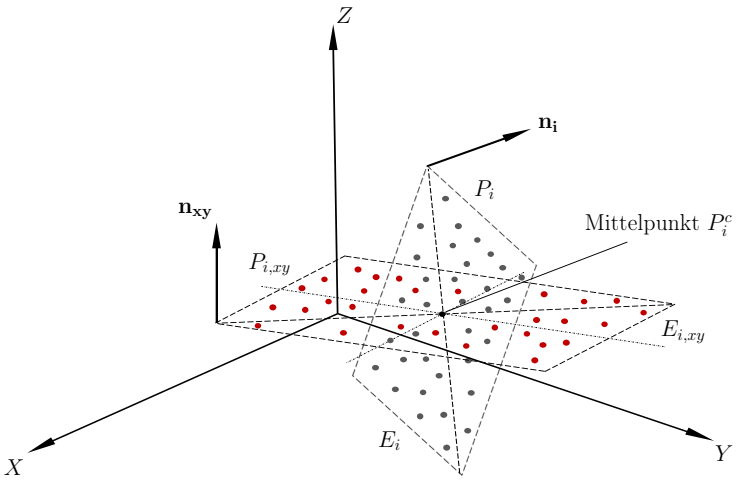


Abbildung 3.8: Ausrichtung der Punktdaten P_i (graue Punkte) in der XY-Ebene. Das Ergebnis wird durch $P_{i,xy}$ (rote Punkte) dargestellt.

3.4.2.3 Bestimmung der Randpunkte mit der α-*shape*-Methode

Die Abbildung 3.6 zeigt, dass Wände in einem Rahmen zwischen Stützen und Trägern angeordnet sind. Eine gezielte Segmentierung von Wänden aus einem Rahmen ist mit RANSAC oder anderen modellbasierten Segmentierungsverfahren (Unterabschnitt A.1.1) aufgrund der Träger und Stützen als geometrische Störfaktoren nicht möglich. Im Rahmen des Segmentierungsprozesses wird daher ein iterativer Ansatz auf der Grundlage der α-*Shape*-Methode [54, 179] zur Bestimmung der Randpunkte E_j von $P_{i,xy}$ im euklidischen $\mathbb{R}^2$ vorgeschlagen. Eine kurze Beschreibung

der α-*Shape*-Methode [54, 179] ist in Unterabschnitt 3.7.2 angegeben. Aufgrund des strukturellen Aufbaus eines Rahmens (Abbildung 3.6) werden nur die Randpunkte der Träger und Stützen mit dem α-*Shape*-Verfahren einbezogen. Folglich können die Punktdaten der Träger und Stützen durch die Wiederholung der folgenden Arbeitsschritte vollständig aus dem Rahmensegment entfernt werden:

1. Bestimmung der Randpunkte E_j von $P_{i,xy}$ mit der α-*Shape*-Methode [54, 179],
2. Suche und Gruppierung der Randpunkte E_j in $P_{i,xy}$,
3. Entfernung der gruppierten Punkte von der Eingabe $P_{i,xy}$.

Die verbleibenden Punktdaten in der Eingabe $P_{i,xy}$ sind die der Wand bzw. der Wände des Rahmens.

3.4.2.4 Suche und Entfernung von Punkten

Um die identifizierten Randpunkte E_j schnell und effizient im Datensatz $P_{i,xy}$ zu finden, wird für $P_{i,xy}$ eine *Octree*-Datenstruktur erstellt. Ein *Octree* ist eine Baumdatenstruktur, die zur Partitionierung eines dreidimensionalen Raumes in kleinere Regionen dient. Jeder Knoten in einem *Octree* stellt dabei eine Region dar und kann bis zu acht Kinder haben (Abbildung 3.9). Im Vergleich zu konventionellen Suchalgorithmen, die den gesamten Datensatz durchsuchen, beschränkt die *Octree*-Datenstruktur die Suche auf die Knoten und ist deshalb im Stande, Punkte in einem großen Punktdatensatz sehr schnell zu finden. Die *Octree*-Datenstruktur wird für die Punktdaten $P_{i,xy}$ unter Berücksichtigung der geometrischen Eigenschaften der Eingabe mit *cloudComPy* [149] erzeugt.

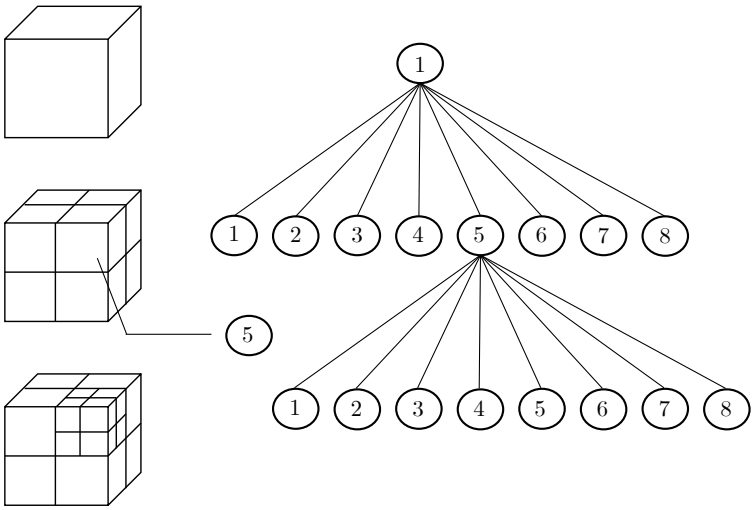


Abbildung 3.9: Bildliche Beschreibung der *Octree*-Datenstruktur: Unterteilte Regionen (links) und Knotenaufbau der Baumdatenstruktur (rechts).

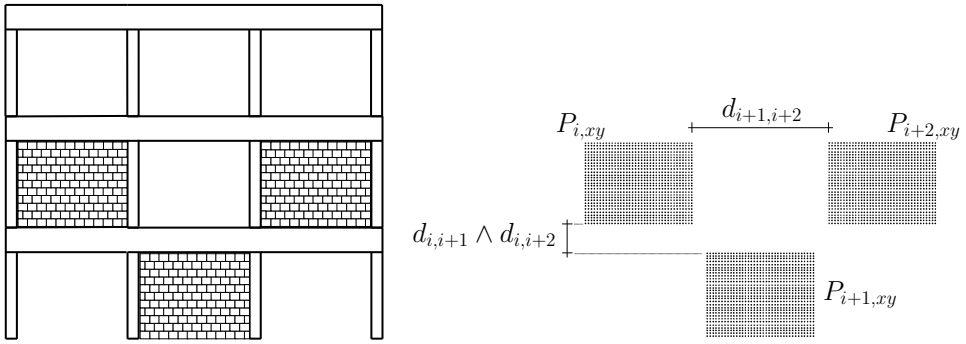
Anschließend wird für die Eingabe $P_{i,xy}$ mit *cloudComPy* [149] ein Skalarfeld generiert, das zur Gruppierung der Punktdaten auf der Grundlage von Skalarwerten verwendet wird. Die gefundenen Randpunkte E_j in $P_{i,xy}$ erhalten im Skalarfeld den Wert 1 und die restlichen Punkte den Wert 0. Die Entfernung der Punktdaten mit dem Skalarwert 1 erfolgt schließlich mit der Filterfunktion *filterbyValue* von *cloudComPy* [149].

3.4.2.5 Trennung nicht verbundener Komponenten

Da die gefilterten Punktdaten der Wände aus unverbundenen Komponenten bestehen können, wie z. B. in der Abbildung 3.10 (a) dargestellt, werden die Daten $P_{i,xy}$ im nächsten Schritt mit der *ExtractConnectedComponents*-Methode von *cloudComPy* [40] verarbeitet. Die *ExtractConnected-Components*-Methode extrahiert verbundene Punktmengen aus der *Octree*-Datenstruktur, die durch einen Mindestabstand $d_{\min}$ getrennt sind. Der Mindestabstand $d_{\min}$ zwischen den Komponenten wird dabei anhand der *Octree*-Datenstruktur der Punktwolke automatisch festgelegt. In der Abbildung 3.10 gelten für die Abstände zwischen den gefilterten Wandkomponenten:

$$d_{i,i+1} < d_{\min} \wedge d_{i,i+2} < d_{\min} \wedge d_{i+1,i+2} < d_{\min}. \tag{3.16}$$

Daher werden die Wandpunktdaten des Rahmens mit der *ExtractConnectedComponents*-Methode in die Punktmengen $P_{i,xy}$, $P_{i+1,xy}$ und $P_{i+2,xy}$ unterteilt.



(a) Rahmen (Vorderansicht) **(b) Punktdaten der Wände**

Abbildung 3.10: Gefilterte Punktdaten der Wände (b) aus dem Rahmen (a) mit der RE-Methode.

3.4.2.6 Herausschneiden von Punktdaten in $\mathbb{R}^3$

Im letzten Schritt werden die in $\mathbb{R}^2$ ermittelten Punktdaten der Wandsegmente in $\mathbb{R}^3$ bestimmt. Zu diesem Zweck wird die Form der Wandsegmente in der *crop2D*-Funktion von *cloudComPy* [40] als Schnittfläche zum Ausschneiden der 3D-Punktdaten verwendet. Für die Festlegung der Schnittfläche, welche einen Polygonzug definiert, wird zunächst eine rechteckige Ebene an das Wandsegment angepasst. Die angepasste Ebene wird anschließend in eine Punktwolke umgewandelt und der durchschnittliche Abstand $\bar{d}_n$ zwischen den Punkten der angepassten Ebene und dem Wandsegment mit der *Cloud-to-Cloud-Distance*-Methode von *cloudComPy* [40] bestimmt. Liegt der durchschnittliche Abstand $\bar{d}_n$ der Punktwolken unter dem festgelegten Grenzwert $\bar{d}_{\max}$ ($\bar{d}_n < \bar{d}_{\max}$), weist das Wandsegment die gleiche Form wie die angepasste Ebene auf, d. h. rechteckig. In diesem Fall definieren die Eckpunkte der angepassten Ebene den Polygonzug des Schnittbereiches. Die Punktmengen innerhalb und außerhalb des Schnittbereiches werden mit Hilfe der inversen Rotationsmatrix R^{-1} in die ursprüngliche Orientierung in $\mathbb{R}^3$ rotiert, wobei erstere das Wandsegment W_n und letztere das Rahmensegment P_n mit ausgeschnittenen Wänden darstellt. Schließlich werden für die Punktdaten W_n und P_n BC-Objekte erstellt und diese jeweils in einer PKL-Datei gespeichert. Dabei werden die BC-Objekte von P_n der PKL-Datei II hinzugefügt, während für die BC-Objekte von W_n die PKL-Datei IV generiert wird. Falls der durchschnittliche Abstand $\bar{d}_n$ über dem festgelegten Grenzwert $\bar{d}_{\max}$ liegt ($\bar{d}_n > \bar{d}_{\max}$), werden

die oben beschriebenen Schritte in gleicher Weise durchgeführt. Die Ausnahme besteht darin, dass anstelle eines Rechteckes ein Polygon an die Form des Wandsegments angepasst wird.

Die beschriebene Methode zur Segmentierung der Punktdaten von Wänden aus einem Rahmen wurde in Python implementiert. Die wesentlichen Schritte der Implementierung sind in Algorithmus 2 als Pseudocode angegeben.

3.4.3 Linear angeordnete Punktmengen

Im letzten Schritt des Segmentierungsprozesses werden die Punktdaten der Träger und Stützen aus den Rahmensegmenten durch Anwendung einer spezifischen Segmentierungsmethode extrahiert. Dieser Unterabschnitt präsentiert die grundlegenden Schritte dieser Methode, die in Algorithmus 3 in Form eines Pseudocodes zusammengefasst dargestellt sind.

3.4.3.1 Segmentierungsmethode

Die Punktdatensätze der Rahmensegmente resultieren aus der Segmentierung der ebenen Punktmengen aus der 3D-Punktwolke (Unterabschnitt 3.4.1). Dementsprechend sind die Punktdaten der Rahmen in einer ebenen Raumkonfiguration angeordnet, weshalb es nicht möglich ist, die einzelnen Komponenten der Rahmensegmente anhand eines Ebenenmodells mit RANSAC zu extrahieren. Da die Punktdaten der Träger und Stützen in einem Rahmensegment eine lineare Struktur aufweisen, wird stattdessen RANSAC mit einem Linienmodell als geeignete Segmentierungsmethode für diesen Anwendungsfall betrachtet. Der gewählte RANSAC-Algorithmus von der *Scikit-Image*-Bibliothek [6] passt dabei eine Linie an $n \geq 2$ zufällige Punkte des Datensatzes an und bestimmt anhand des Modells die Nicht-Ausreißer- bzw. Ausreißerpunkte. Das entsprechende Linienmodell wird im euklidischen $\mathbb{R}^2$ durch die folgende Geradengleichung beschrieben:

$$\bar{y} = m_r \cdot \bar{x} + b_r, \tag{3.17}$$

wobei die Mittelwerte der x- und y-Koordinaten der ausgewählten Punktdaten durch $\bar{x}$ und $\bar{y}$ dargestellt werden, m_r die Steigung des Modells angibt und b_r der y-Achsenabschnitt der Geradengleichung ist. Die Steigung m_r der Linie wird anhand folgender Formel berechnet:

$$m_r = \frac{\sum_{i=1}^{x}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{x}(x_i - \bar{x})^2}, \tag{3.18}$$

wobei x_i und y_i die Koordinaten der ausgewählten Punkte repräsentieren, die zur Anpassung der Linie verwendet werden. Der y-Achsenabschnitt b_r der Linie ergibt sich schließlich aus der Umstellung der Gleichung 3.17:

$$b_r = \bar{y} - m_r \cdot \bar{x}. \tag{3.19}$$

Um die Genauigkeit der Anpassung der Linie an die ausgewählten Punkte zu überprüfen, werden die orthogonalen Abstände zwischen den Punkten und der angepassten Linie berechnet. Punkte, die innerhalb des festgelegten Toleranzbereiches ϵ liegen, werden als Nicht-Ausreißer eingestuft, während alle anderen Punkte als Ausreißer betrachtet werden. Zur Veranschaulichung der beschriebenen RANSAC-Methode mit einem Linienmodell ist in der Abbildung 3.11 eine Schemazeichnung abgebildet. Die Abbildung 3.11 zeigt das Linienmodell (rote Linie), das unter Verwendung von drei zufällig ausgewählten Punkten aus einer gegebenen Punktmenge in $\mathbb{R}^2$ erstellt wurde.

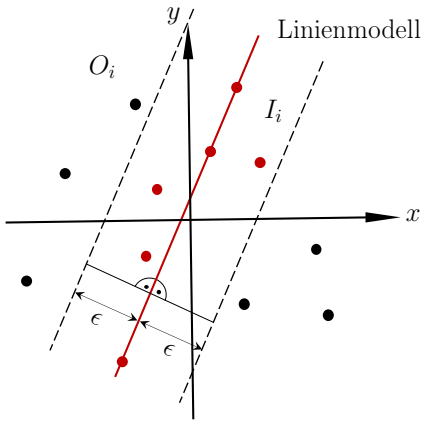


Abbildung 3.11: Schemazeichnung zur Erläuterung der RANSAC-Methode mit einem Linienmodell in $\mathbb{R}^2$.

Die Punkte, die näher als der festgelegte maximale Abstand ϵ orthogonal zur Linie liegen, gehören zur Menge der Nicht-Ausreißerpunkte I_i (rote Punkte). Entsprechend sind die Punkte, die einen orthogonalen Abstand größer als ϵ zur Linie aufweisen, Teil der Ausreißer-Menge O_i des gegebenen Punktdatensatzes (schwarze Punkte).

3.4.3.2 Extraktion von linearen Punktdaten aus einem Rahmensegment

Die vorgeschlagene Methode zur Extraktion von linearen Punktdaten aus Rahmensegmenten beginnt mit dem Einlesen der entsprechenden BC-Objekte aus der PKL-Datei II in die Liste Ψ. Damit die Datenverarbeitung in $\mathbb{R}^2$ durchgeführt werden kann, werden die 3D-Punktdaten P_i der Rahmensegmente mit der in Unterunterabschnitt 3.4.2.2 beschriebenen Methode in der XY-Ebene ausgerichtet. Nachfolgend wird das in der XY-Ebene ausgerichtete Segment, das als Eingabe verwendet wird, als $P_{i,xy}$ bezeichnet. Für das oben beschriebene RANSAC-Verfahren aus der Python-Bibliothek *Scikit-Image* [6] müssen die Anzahl der Iterationen k, der maximale Abstand ϵ und die Anzahl der Punkte n, mit denen das Linienmodell erstellt werden soll, vom Benutzer im Voraus festgelegt werden. Der maximale Abstand ϵ wird wie folgt definiert:

$$\epsilon_b = \frac{h_b}{2} \quad \text{und} \quad \epsilon_c = \frac{b_c}{2}, \tag{3.20}$$

wobei h_b die Höhe der Träger und b_c die Breite der Stützen des betrachteten Rahmensegments repräsentieren. Da die Grenzbereiche des Linienmodells durch den doppelten Wert von ϵ definiert sind (gestrichelte Linien in Abbildung 3.11), stellt ϵ_b die halbe Höhe der Träger und ϵ_c die halbe Breite der Stützen im Segmentierungsprozess dar.
Um zu überprüfen, ob die von RANSAC gefundene Menge der Nicht-Ausreißerpunkte $I_i(x_i, y_i)$ verbunden ist, werden der maximale und durchschnittliche Abstand der Nachbarpunkte mit einem festgelegten Schwellenwert verglichen. Dazu wird zunächst eine rechteckige Ebene an die Form des detektierten Liniensegments $I_i(x_i, y_i, z_i = 0)$ angepasst, wobei die z-Koordinaten der Punkte von I_i in $\mathbb{R}^3$ auf Null gesetzt werden. Die erzeugte Ebene wird in eine äquivalente Punktwolke E_i umgewandelt und anschließend mit der *Cloud-to-Cloud-Distance*-Methode [40] der maximale und durchschnittliche Abstand, $d_{\max}(E_i, I_i)$ und $d_{\text{mean}}(E_i, I_i)$, zwischen den Punkten von E_i und I_i berechnet. Wenn die Bedingungen:

$$d_{\max}(E_i, I_i) \leq d_{\max} \quad \text{mit} \quad d_{\max} \in \{x \in \mathbb{R}^+ \mid 0 \leq x \leq \max(h_b b_c)\} \tag{3.21}$$

und

$$d_{\text{mean}}(E_i, I_i) \leq d_{\text{mean}} \quad \text{mit} \quad d_{\text{mean}} \in \{x \in \mathbb{R}^+ \mid 0 \leq x \leq \max(h_b b_c)\} \tag{3.22}$$

erfüllt sind, wobei $d_{\max}$ und d_{mean} manuell festgelegte Schwellenwerte repräsentieren, werden die Punktdaten des Liniensegments I_i in die Liste I gespeichert. Die Schwellenwerte $d_{\max}$ und d_{mean} sind dabei zwischen Null und der Trägerhöhe h_b bzw. der Stützenbreite b_c des Rahmensegments definiert. Die Überprüfung der Punktabstände zur Bestimmung von Datenlücken ist essentiell, da die RANSAC-Implementierung von *Scikit-Image* [6] darauf abzielt, die Punktmenge mit der höchsten Anzahl von Punkten innerhalb der Grenzbereiche des erstellten Linienmodells zu identifizieren, ohne die Abstände zwischen den gefundenen Komponenten zu berücksichtigen.
Der Prozess zur Extraktion der linearen Punktdaten aller Träger und Stützen wird durch iterative Anwendung auf jedes Rahmensegment $P_{i,xy}$ aus der Menge Ψ innerhalb einer *while*-Schleife realisiert. Dabei wird die Menge der Ausreißerpunkte O_i nach der ersten Iteration der *while*-Schleife als Eingabe für die weitere Datenverarbeitung mit RANSAC definiert, indem die Punktdaten von O_i der Variablen $P_{i,xy}$ zugewiesen werden ($P_{i,xy} \leftarrow O_i$). Die Punktmenge O_i ergibt sich aus der Differenzmenge des detektierten Liniensegments I_i von der Gesamtmenge der Punkte $P_{i,xy}$:

$$O_i = P_{i,xy} \setminus I_i. \tag{3.23}$$

Die Bedingungen in den Gleichungen 3.21 und 3.22 stellen die Kriterien dar, die verwendet werden, um den Wert des Laufparameters a zu definieren. Wenn die genannten Bedingungen erfüllt sind, wird der Wert von a auf Null gesetzt. Andernfalls wird der Wert von a um eins erhöht ($a \leftarrow a + 1$). Die Beendigung der *while*-Schleife erfolgt, wenn der Wert von a größer als der manuell vorgegebene Wert $a_{\max}$ ist, d. h. $a > a_{\max}$. Dadurch wird gewährleistet, dass RANSAC ausreichend Iterationen durchführen kann, um ein Liniensegment mit zusammenhängenden Punktdaten zu identifizieren. Die Schritte des beschriebenen Segmentierungsprozesses, die innerhalb der *while*-Schleife ausgeführt werden, sind in Algorithmus 3 in den Zeilen 7 bis 17 aufgeführt. Nach Abschluss der iterativen Segmentierung des Rahmensegments werden die verbleibenden Punktdaten der Eingabe, d. h. die Ausreißerpunkte der letzten Iteration, der Liste I hinzugefügt (Algorithmus 3, Zeile 18 bis 20).

3.4.3.3 Trennung nicht verbundener Komponenten

Die mit RANSAC extrahierten Punktdaten der Träger und Stützen verlaufen kontinuierlich über die gesamte Länge oder Höhe des Rahmens. Daraus ergeben sich Datenlücken zwischen den Komponenten des Rahmens, deren Ausmaß durch die Breite der Stützen b_c oder die Höhe der Träger h_b definiert ist. Da die Abstände der Nachbarpunkte zwischen den verbundenen Rahmenkomponenten höchstens den Wert der Trägerhöhe h_b bzw. der Stützenbreite b_c erreichen, sind die in den Gleichungen 3.21 und 3.22 angegebenen Abstandsbedingungen erfüllt. Somit werden die entsprechenden Punktdaten der Segmente in die Liste I aufgenommen. Um sicherzustellen, dass die detektierten Punktmengen der Rahmenkomponenten miteinander verbunden sind, werden alle Elemente in der Liste I mit der *ExtractConnectedComponents*-Methode [40] in zusammenhängende Punktgruppen unterteilt. Die resultierenden Segmente werden anschließend in der Liste S gespeichert (Algorithmus 3, Zeile 21 bis 23).
Zur besseren Veranschaulichung dieses Ansatzes zeigt die Abbildung 3.12 beispielhaft die Segmentierung der Punktdaten eines Rahmens (Abb. 3.12 (I)) unter Verwendung von RANSAC und der *ExtractConnectedComponents*-Methode [40]. Da die Träger mehr Punktdaten aufweisen als die Stützen, werden zuerst die Trägerkomponenten mit der RANSAC-Methode segmentiert. Daraus ergeben sich die Mengen der Nicht-Ausreißerpunkte I_1 bis I_3, wobei jede Farbe ein detektiertes Segment der Rahmenkomponente darstellt (Abbildung 3.12 (II)). Aufgrund der fehlenden

Punktdaten der Träger ergeben sich bei den mit RANSAC segmentierten Punkten der Stützen, I_4 bis I_7, Datenlücken zwischen den verbundenen Komponenten im Abstand der Trägerhöhe h_b (Abbildung 3.12 (III)). Abschließend wird die *ExtractConnectedComponents*-Methode auf die detektierten Segmente I_1 bis I_4 angewendet, was zu den verbundenen Rahmenkomponenten S_1 bis S_{15} führt (Abbildung 3.12 (IV)).

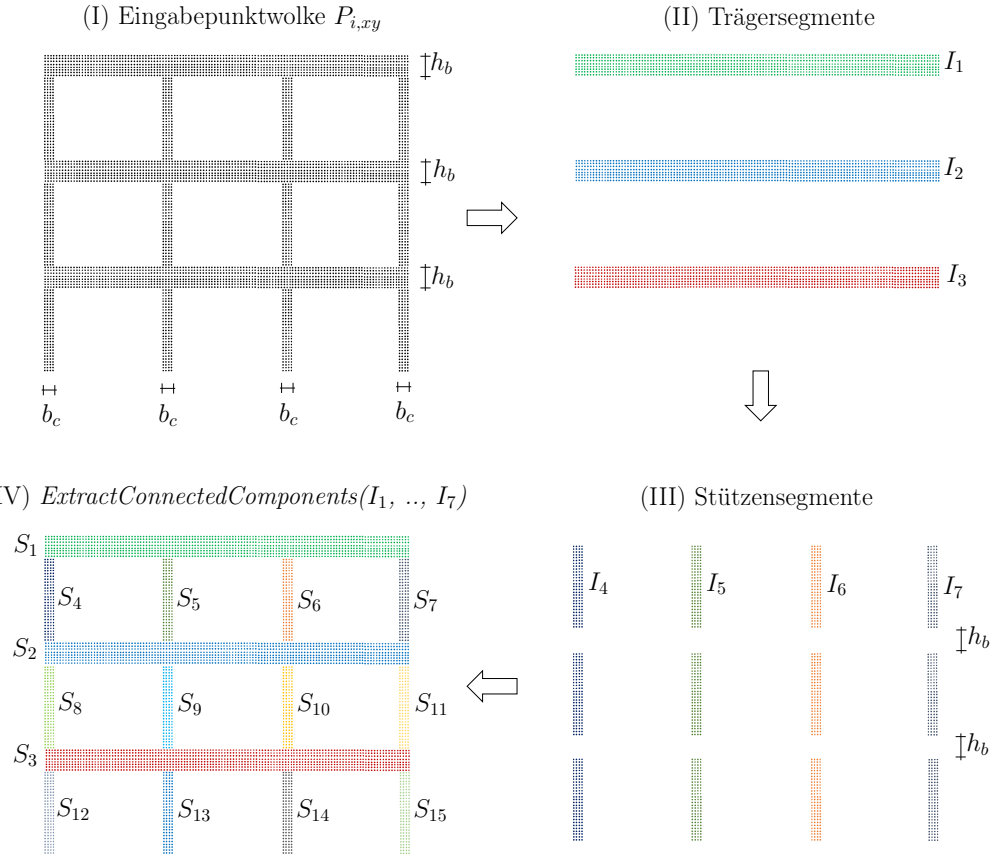


Abbildung 3.12: Bildliche Darstellung der Extraktion der linearen Punktdaten von Trägern und Stützen aus einem Rahmensegment unter Verwendung von RANSAC mit einem Linienmodell und der *ExtractConnectedComponents*-Methode von *cloudComPy* [40].

3.4.3.4 Transformation der segmentierten Punktdaten

Die Komponentendicke t_n des Rahmensegments wird aus der Liste Ψ entnommen ($t_n \leftarrow \Psi_i$). Falls die Komponentendicke t_n kleiner als die vorgegebene minimale Dicke $t_{\min}$ ist, wird der Wert der Komponentendicke t_n auf den Standardwert t_d gesetzt. Zur effizienten Bestimmung der z-Koordinaten der Punktdaten der Träger- und Stützensegmente wird die Ebenengleichung der in der XY-Ebene ausgerichteten Eingabepunktwolke $P_{i,xy}$ verwendet. Die z-Koordinaten errechnen sich aus der Umstellung der Ebenengleichung:

$$z_n = \frac{d_i - a_i \cdot x_n - b_i \cdot y_n}{c_i}, \qquad \text{mit} \quad \{i, n\} \in \mathbb{N}, \tag{3.24}$$

wobei a_i, b_i, c_i und d_i die Koeffizienten der Ebenengleichung des betrachteten Rahmensegments $P_{i,xy}$ und x_n und y_n die Punktkoordinaten der dazugehörigen Stützen- bzw. Trägersegmente, $S_n(x_n, y_n, z = 0)$, sind.

Die angepassten Punktdaten der Segmente $S_n(x_n, y_n, z_n)$ werden anschließend mittels der inversen Rotationsmatrix R^{-1} in ihre ursprüngliche Orientierung transformiert. Damit die transformierten Punktdaten der Stützen- und Trägersegmente eindeutig identifiziert werden können, erhalten sie die Variable P_n mit $n \in \mathbb{N}$. Abschließend werden unter Verwendung der Punktdaten P_n und der zugehörigen Dicke t_n ein BC-Objekt für die Rahmenkomponenten erzeugt und in der PKL-Datei V gespeichert. Der beschriebene Prozess ist in Algorithmus 3 in den Zeilen 24 bis 32 dargestellt.

3.5 Objektklassifizierung

Die Klassifizierung der segmentierten 3D-Punktdaten ist ein wesentlicher Aspekt der vorgeschlagenen Strategie der halbautomatischen BIM-Generierung (Abbildung 3.1), da sie es ermöglicht, der aufgenommenen 3D-Punktwolke eine semantische Bedeutung auf der Ebene der Gebäudekomponenten zuzuweisen. In diesem Abschnitt wird die vorgeschlagene Methode zur Objektklassifizierung der segmentierten Punktdaten auf der Grundlage geometrischer Eigenschaften wie Form, Größe und Orientierung vorgestellt. Dabei werden den Segmenten vordefinierte, strukturelle oder nicht-strukturelle Objektklassen zugewiesen, die in Tabelle 3.1 aufgelistet sind.

Tabelle 3.1: Vordefinierte Objektklassen mit Angabe der dazugehörigen *type*- und RGB-Daten im BC-Objekt.

#	**Objektklasse**	**type**	**RGB**
1	Decke	*Slab*	Blau
2	Wand	*Wall*	Grün
3	Decke\Wand	*Slab|Wall*	Hellblau
4	Träger	*Beam*	Gelb
5	Stütze	*Column*	Rot
6	Träger\Stütze	*Beam|Column*	Orange
7	Bodenplatte	*Base*	Türkis
8	Boden	*Ground*	Grau
9	nicht klassifiziert	*Unclassified*	Beige

3.5.1 Ebene Punktmengen

Zunächst werden die Punktdaten P, die Komponentendicke t und das boolesche *hasParallel* der BC-Objekte der Punktsegmente aus den PKL-Dateien I und IV in die Liste Ψ aufgenommen. Ist die Komponentendicke t_i der Punktmenge kleiner als die vorgegebene Mindestdicke $t_{\min}$, wird der Standardwert t_d als Dicke zugewiesen ($t_i \leftarrow t_d$). Anschließend werden die Höhe $H_{i,\min}$ und die Länge $L_{i,\min}$ der in der XY-Ebene ausgerichteten Punktmenge, $P_i \rightarrow P_{i,xy}$, auf der Grundlage ihrer minimalen *Bounding Box* bestimmt. Die *Bounding Box* ist dabei die kleinstmögliche quadratische Umhüllung, die die Punktdaten entlang der Koordinatenachsen an ihren Datengrenzen umschließt (Abbildung 3.13).
Die Bestimmung der minimalen *Bounding Box* der ausgerichteten Punktmenge $P_{i,xy}$ erfolgt auf der Basis einer inkrementellen Rotation der Punktdaten um die globale z-Achse. Dabei wird nach jeder Rotation um $\varphi_{z,j} \leftarrow \varphi_{z,j>1} + 1$ das Volumen V_j der *Bounding Box* gemäß der folgenden Gleichung berechnet:

$$V_j = H_{j,x} \cdot L_{j,y} \cdot Z_{j,z}, \qquad \text{mit} \quad j \in \{x \in \mathbb{N} \mid 0 \leq x \leq 360\} \tag{3.25}$$

wobei

$$H_{j,x} = \left| x_{\max}(P_{i,xy}) - x_{\min}(P_{i,xy}) \right|, \tag{3.26}$$

$$L_{j,y} = |y_{\max}(P_{i,xy}) - y_{\min}(P_{i,xy})|\,, \tag{3.27}$$

$$Z_{j,z} = |z_{\max}(P_{i,xy}) - z_{\min}(P_{i,xy})|\,, \tag{3.28}$$

die Höhe, Länge und Tiefe der *Bounding Box* repräsentieren.

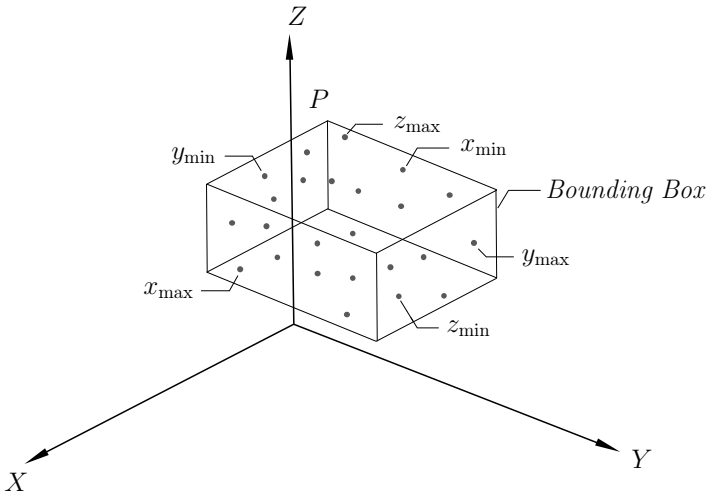


Abbildung 3.13: Bildliche Darstellung einer *Bounding Box* einer 3D-Punktwolke. Die Koordinaten $x_{\max}$, $x_{\min}$, $y_{\max}$, $y_{\min}$, $z_{\max}$ und $z_{\min}$ der Punktwolke P definieren die Grenzen der *Bounding Box* entlang der globalen Raumachsen.

Für die Rotation der Punktdaten um die globale z-Achse wird die folgende Rotationsmatrix verwendet:

$$R_z(\varphi_{z,j}) = \begin{bmatrix} \cos(\varphi_{z,j}) & -\sin(\varphi_{z,j}) & 0 \\ \sin(\varphi_{z,j}) & \cos(\varphi_{z,j}) & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3.29}$$

Das erhaltene Volumen V_j und der zugehörige Rotationswinkel $\varphi_{z,j}$ werden nach jedem inkrementellen Schritt j in der Liste B gespeichert. Nachdem die Punktwolke $P_{i,xy}$ vollständig um 360 Grad gedreht wurde, wird der Rotationswinkel $\varphi_{z,\min}$, bei dem das Volumen der *Bounding Box* minimal ist, wie folgt aus der Liste B ermittelt:

$$V_{\min} = V_j(\varphi_{z,j}) = \min\{B[V_j(\varphi_{z,1}), .., V_{360}(\varphi_{z,360})]\}, \tag{3.30}$$

$$j = B.\text{index}(V_{\min}), \qquad \varphi_{z,\min} = \varphi_{z,j} = B[j]. \tag{3.31}$$

Schließlich wird $\varphi_{z,\min}$ entsprechend der Gleichung 3.29 in eine (3x3)-Rotationsmatrix umgewandelt und damit die Punktwolke $P_{i,xy}$ transfomiert, um die Dimensionen $H_{i,\min}$ und $L_{i,\min}$

der Punktmenge unter Verwendung seiner minimalen *Bounding Box* zu erhalten. Eine bildliche Beschreibung dieses Ansatzes ist in der Abbildung 3.14 dargestellt.

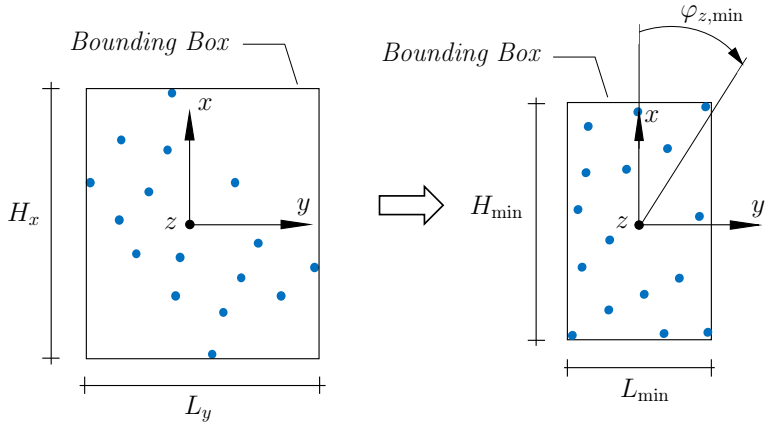


Abbildung 3.14: Bildliche Beschreibung der Bestimmung der minimalen *Bounding Box* einer in der XY-Ebene ausgerichteten Punktwolke (links), bei der die Punktdaten um $\varphi_{z,\min}$ rotiert werden, um das kleinstmögliche Volumen der *Bounding Box* zu erhalten (rechts).

Im nächsten Schritt wird überprüft, ob das ebene Segment P_i in seiner ursprünglichen Orientierung parallel zur XY-Ebene ausgerichtet ist. Zu diesem Zweck wird der Winkel α_i zwischen dem Normalenvektor des Segments $\mathbf{n_i}$ und dem Normalenvektor der XY-Ebene $\mathbf{n_{xy}}$ mit der folgenden Formel berechnet:

$$\alpha_i = \arccos\left(\frac{\mathbf{n_i} \cdot \mathbf{n_{xy}}}{|\mathbf{n_i}||\mathbf{n_{xy}}|}\right) \qquad \text{mit} \quad \mathbf{n_{xy}} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \tag{3.32}$$

wobei $\mathbf{n_i}$ aus der an die Punktdaten P_i angepassten Ebene E_i bestimmt wird (Abbildung 3.15).

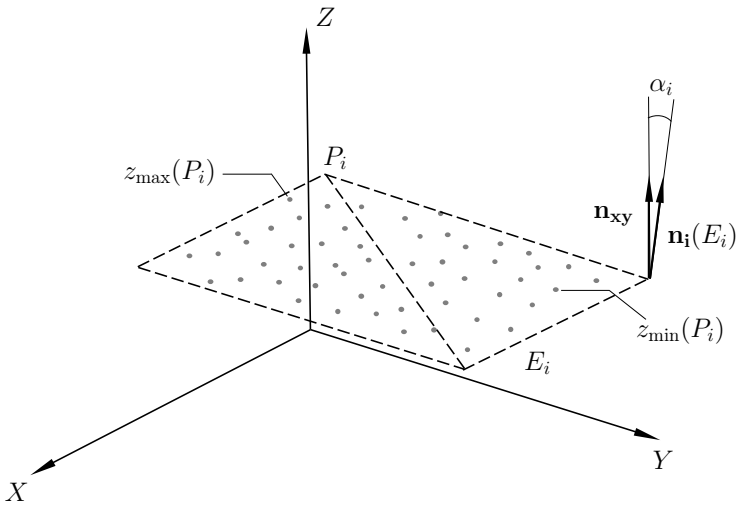


Abbildung 3.15: Grafische Darstellung des Winkels α_i zwischen dem Normalenvektor $\mathbf{n_i}$ der an die Punktwolke P_i angepassten Ebene E_i und dem Normalenvektor $\mathbf{n_{xy}}$ der XY-Ebene.

Wenn der Winkel a_i der Punktmenge P_i kleiner oder gleich dem maximalen Winkel $a_{\max}$ ist, d. h. $a_i \leq a_{\max}$, wird das Segment als parallel in der XY-Ebene betrachtet. In diesem Fall kann die

Objektklasse *Wall* für das Segment ausgeschlossen werden, da Wände orthogonal zur XY-Ebene ausgerichtet sind. Zur Unterscheidung der Objektklassen *Slab*, *Base* und *Ground*, die hinsichtlich ihrer Orientierung identisch sind, wird die Höhe des Objektes über der Geländeoberkante (GOK), d. h. $z > 0$, als Kriterium verwendet. Liegt die minimale z-Koordinate des Segments P_i über dem festgelegten Schwellenwert $z_{\text{min,slab}}$, d. h. $z_{\min}(P_i) > z_{\text{min,slab}}$, wobei $z_{\text{min,slab}}$ die minimale Höhe einer Deckenplatte über GOK definiert, wird dem entsprechenden BC-Objekt die Objektklasse *Slab* zugewiesen. Für den Fall $z_{\max}(P_i) \leq z_{\text{min,base}}$, wobei $z_{\text{min,base}}$ die minimal vorgegebene Höhe der Bodenplatte über GOK ist, wird geprüft, ob das ebene Segment P_i der Objektklasse *Base* oder *Ground* zugeordnet werden kann. Da davon ausgegangen wird, dass Bodenplatten in Stahlbetonrahmenkonstruktionen etwa die gleiche Bauteilfläche wie Geschossdecken aufweisen, wird das Kriterium $A(P_i) \approx A_{i,\text{slab}}$ für die Zuordnung zur Objektklasse *Base* verwendet. Trifft das genannten Flächenkriterium auf die ebene Punktmenge P_i nicht zu, erhält das betreffende BC-Objekt die nicht-strukturelle Objektklasse *Ground*.
Wie bereits oben erwähnt, können ebene Segmente, die orthogonal zur XY-Ebene ausgerichtet sind, Wände darstellen. Die Orthogonalität der Punktmenge P_i zur XY-Ebene liegt vor, wenn der z-Wert des Normalenvektors $\mathbf{n_i}$ aus Gleichung 3.32 annäherend Null ist, d. h. $\mathbf{n_{i,z}} \approx 0$. Die Orthogonalität zur XY-Ebene kann alternativ anhand des Winkels α_i aus Gleichung 3.32 bestimmt werden, der etwa 90 Grad ergeben muss. Wenn die orthogonal zur XY-Ebene ausgerichtete Punktmenge P_i die Mindestabmessung für die Höhe, $H_{i,\min} > H_{\text{min,wall}}$, und die Länge, $L_{i,\min} > L_{\text{min,wall}}$, erfüllt, wird seinem BC-Objekt die Objektklasse *Wall* zugewiesen. Hierbei definieren $H_{\text{min,wall}}$ und $L_{\text{min,wall}}$ die vorgegebenen Mindestabmessungen einer Wand. Wenn das Segment P_i weder parallel noch orthogonal zur XY-Ebene verläuft, wird anhand seiner Abmessungen $H_{i,\min}$ und $L_{i,\min}$ geprüft, ob es sich bei der ebenen Punktmenge um eine beschädigte Wand oder eine beschädigte Decke handelt. Decken und Wände, die eine starke Abweichung von ihrer ursprünglichen räumlichen Orientierung aufweisen, lassen sich in Form von Punktdaten geometrisch nicht voneinander unterscheiden. Aus diesem Grund wird das BC-Objekt des betreffenden Segments, das die Mindestabmessungen einer Wand und einer Decke aufweist, mit der mehrdeutigen Objektklasse *Slab* | *Wall* klassifiziert. Segmente P_i, die keine der oben genannten Bedingungen für die Klassenzuordnung erfüllen, werden der Klasse *Unclassified* zugeordnet, d. h. sie können aufgrund ihrer Form und Orientierung keiner der definierten Objektklassen zugeordnet werden. Die Punktdaten von P_i, die Komponentendicke t_i, die zugeordnete Objektklasse in *type* und die entsprechenden RGB-Daten der jeweilige Klasse werden abschließend in das bestehende BC-Objekt eingefügt und in einer neuen PKL-Datei mit der Nummer VI gespeichert.
Der beschriebene Prozess der geometriebasierten Objektklassifizierung von ebenen Punktmengen ist in Algorithmus 4 in Form eines Pseudocodes zusammengefasst dargestellt.

3.5.2 Linear angeordnete Punktmengen

Nach der Klassifizierung der ebenen Punktmengen wird überprüft, ob die BC-Objekte der linear angeordneten Punktmengen aus Unterabschnitt 3.4.3 den in Tabelle 3.1 angegebenen Objektklassen *Beam*, *Column* oder *Beam* | *Column* zugeordnet werden können. Dazu werden zu Beginn die linearen Punktdaten der BC-Objekte aus der PKL-Datei V in die Liste Ψ eingelesen. Im nächsten Schritt wird die lineare Punktmenge P_i in der YZ-Ebene ausgerichtet ($P_i \rightarrow P_{i,yz}$). Die Höhe $H_{i,z}$ und Länge $L_{i,y}$ des resultierenden Segments in der YZ-Ebene werden mit Hilfe der folgenden Gleichungen berechnet:

$$H_{i,z} = \left| z_{\max}(P_{i,yz}) - z_{\min}(P_{i,yz}) \right|, \tag{3.33}$$

$$L_{i,y} = \left| y_{\max}(P_{i,yz}) - y_{\min}(P_{i,yz}) \right|. \tag{3.34}$$

Anschließend werden die ermittelten Dimensionen des linearen Segments P_i mit den vorgegebenen minimalen und maximalen Abmessungen eines Trägers und einer Stütze verglichen:

$$\begin{bmatrix} H_{\text{min,column}} & < & H_{i,z} & & \\ L_{\text{min,column}} & < & L_{i,y} & < & L_{\text{max,column}} \end{bmatrix}, \tag{3.35}$$

$$\begin{bmatrix} H_{\text{min,beam}} & < & H_{i,z} & < & H_{\text{max,beam}} \\ L_{\text{min,beam}} & < & L_{i,y} & & \end{bmatrix}. \tag{3.36}$$

Erfüllt das Segment P_i die Mindestabmessungen aus Gleichung 3.35 oder Gleichung 3.36, wird untersucht, ob die Punktmenge eine rechteckige Form aufweist. Hierfür wird eine rechteckige Ebene an das Segment P_i angepasst und dann in eine äquivalente Punktwolke E_i transformiert. Im nächsten Schritt wird mit der *Cloud-to-Cloud-Distance*-Methode [40] der durchschnittliche Abstand $\bar{d}(P_i, E_i)$ zwischen den Punkten von P_i und E_i bestimmt. Falls der durchschnittliche Abstand unter dem festgelegten Grenzwert $\bar{d}_{\min}$ liegt, d. h. $\bar{d}(P_i, E_i) < \bar{d}_{\min}$, weist das Segment P_i die gleiche Form wie die angepasste Ebene E_i auf und ist somit rechteckig. Das jeweilige BC-Objekt wird dann basierend auf den vorhandenen Abmessungen seiner Punktdaten P_i entweder als *Beam* oder *Column* klassifiziert.
Wenn das Segment P_i weder die geometrische Bedingung nach Gleichung 3.35 noch nach Gleichung 3.36 erfüllt, wird die minimale *Bounding Box* der Punktdaten gemäß der vorgestellten Methode in Unterabschnitt 3.5.1 bestimmt. Daraus ergeben sich die minimalen Abmessungen $H_{i,\min}$ und $L_{i,\min}$ des Segments P_i. Die weiteren Schritte entsprechen dem zuvor beschriebenen Ansatz, wobei in den Gleichungen 3.35 und 3.36 die Dimensionen $H_{i,\min}$ und $L_{i,\min}$ anstelle von $H_{i,z}$ und $L_{i,y}$ verwendet werden. Da der Bauteiltyp aufgrund der vom intakten Bauteilzustand abweichenden Orientierung nicht eindeutig anhand der Formgeometrie bestimmt werden kann, wird dem BC-Objekt in einem solchen Fall die mehrdeutige Objektklasse *Beam* | *Column* zugewiesen. In Abbildung 3.16 sind zur Veranschaulichung des beschriebenen Klassifizierungsansatzes drei linear angeordnete Punktsegmente, $P_{1,yz}$ bis $P_{3,yz}$, angegeben, die basierend auf ihren Abmessungen und ihrer Form in der XY-Ebene den Objektklassen *Beam*, *Column* und *Beam* | *Column* zugeordnet sind.

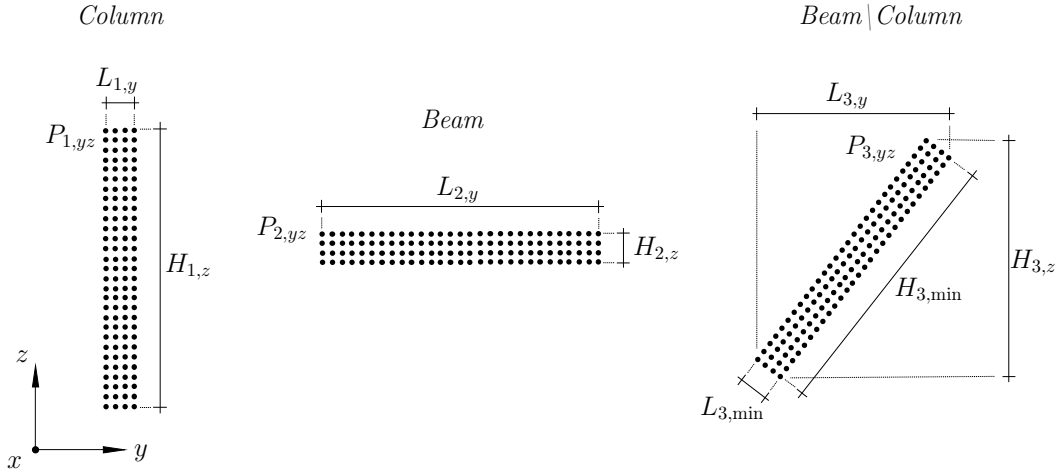


Abbildung 3.16: Klassifizierung der linear angeordneten Punktdaten $P_{1,yz}$ bis $P_{3,yz}$ in die Objektklassen *Column*, *Beam* und *Beam* | *Column* basierend auf ihren Abmessungen und ihrer Form in der YZ-Ebene.

Segmente P_i, die nicht als *Column*, *Beam* oder *Beam\|Column* klassifiziert werden können, erhalten die Klasse *Unclassified*. Die Zeichenfolge in *type* sowie die RGB-Daten der jeweiligen Objektklasse (Tabelle 3.1) werden schließlich in das bestehende BC-Objekt eingefügt und in der PKL-Datei VI gespeichert.
Die Implementierung des beschriebenen Prozesses der Objektklassifizierung von linear angeordneten Punktmengen basierend auf ihrer Geometrie ist als Pseudocde in Algorithmus 5 dargestellt.

3.6 Materialklassifizierung

Die in Abbildung 3.1 dargestellte Strategie der algorithmischen Datenverarbeitung berücksichtigt neben der Objektklassifizierung (Abschnitt 3.5) auch die Materialklassifizierung zur semantischen Informationsanreicherung der BC-Objekte. Im Rahmen dieser Arbeit wird die Materialklassifizierung mit Hilfe eines tiefen neuronalen Netzwerkes (engl. *Deep Neural Networks*, DNNs) auf Basis von DL durchgeführt. In diesem Zusammenhang werden die RGB-Daten aus der erzeugten 3D-Punktwolke des Tragwerkes als Eingabevariablen (engl. *features*) für den DL-Prozess gewählt. Diese Wahl wird damit begründet, dass Materialien wie Beton und Mauerwerk anhand ihrer Farbmerkmale eindeutig unterschieden werden können. Andere Merkmale wie die Orientierung oder die Form der Bauteile werden hierbei aufgrund ihrer Mehrdeutigkeit nicht berücksichtigt. Da in dieser Arbeit nur Stahlbetonrahmentragwerke betrachtet werden, werden für die Zielwerte (engl. *labels*) lediglich *Concrete* und *Masonry* festgelegt. Es ist jedoch wichtig zu beachten, dass die *labels* um weitere Materialklassen, wie z. B. *Steel* oder *Timber*, erweitert werden können. Für eine detaillierte Einführung in die Grundlagen von DL mit DNNs wird der Leser auf [37, 73, 126] verwiesen.
Im Folgenden wird die Vorbereitung der Trainings- und Testdaten sowie der Aufbau des DL-Modells mit einem DNN beschrieben. Darüber hinaus wird die Genauigkeit des trainierten Modells untersucht und die Integration der DL-basierten Materialklassifizierung im Arbeitsablauf der algorithmischen Datenverarbeitung vorgestellt.

3.6.1 Daten

Die Leistungsfähigkeit eines DL-Modells in Bezug auf die Vorhersage von *labels* basierend auf *features* hängt maßgeblich vom Lernprozess des aufgebauten DNNs ab. Entscheidend für den Lernprozess eines DNNs ist die Qualität und Quantität der Trainingsdaten, wobei sich die Quantität nach der Komplexität der *features* richtet. In unserem Anwendungsfall soll ein vortrainiertes Modell verwendet werden, um das Material eines Bauteils auf der Grundlage eines RGB-Wertes vorherzusagen. Aufgrund der moderaten Komplexität dieser *feature-label*-Beziehung werden 1000 Daten pro *label* gesammelt und für den Lernprozess vorbereitet. In diesem Fall entspricht dies 1000 RGB-Daten für *Concrete* und 1000 RGB-Daten für *Masonry*. Um eine objektive Zusammensetzung der Daten zu gewährleisten, werden ausschließlich RGB-Daten aus Bildern berücksichtigt, die reale Oberflächen von Beton und Mauerwerk bei unterschiedlichen Lichtverhältnissen darstellen. Für die Zusammenstellung des Bilddatensatzes von Betonoberflächen wurden teilweise Bilder aus [187] und [117] extrahiert, die von den Autoren speziell für ML-Zwecke zusammengestellt wurden. Da für den beschriebenen Zweck kein geeigneter Bilddatensatz für Mauerwerksoberflächen existiert, wurden vom Autor dieser Arbeit 1000 Bilder von verschiedenen Mauerwerksoberflächen gesammelt und so geschnitten, dass nur die Oberfläche des Materials dargestellt ist. Dabei handelt es sich um lizenzfreie Bilder aus dem Internet und um selbst aufgenommene Bilder. Ein Auszug aus den gesammelten Bilddaten von Beton- und Mauerwerksoberflächen ist in der Abbildung 3.17 dargestellt.

Abbildung 3.17: Auszug aus den erstellten Bilddatensätzen: Betonoberflächen [117, 187] (oben) und Mauerwerksoberflächen (unten).

Da RGB-Daten als *features* für das DL-Modell festgelegt wurden, muss die dominante Farbe aus den gesammelten Bilddaten extrahiert werden, d. h. der RGB-Wert in der Rastergrafik, der am häufigsten auftritt. Zu diesem Zweck wurde ein Python-Skript verwendet, dass in Algorithmus 6 als Pseudocode angegeben ist. Für die Extraktion der dominanten Farbe in einer Rastergrafik wird zunächst die Bilddatei gelesen und die RGB-Werte der Pixelgrafik in einem zweidimensionalen *array* gespeichert. Anschließend wird der *k-means*-Algorithmus aus der ML-Bibliothek *scikit-learn* [132] verwendet, um die RGB-Werte aus dem *array* in Farbgruppen zu clustern. Um den RGB-Wert zu ermitteln, der am häufigsten in der Rastergrafik auftritt, wird der Prozentsatz jeder Farbgruppe berechnet. Dabei wird die Anzahl der Pixel einer Farbgruppe durch die Gesamtzahl der Pixel in der Rastergrafik dividiert und in einer Liste gespeichert. Der Index der Farbgruppe aus der Liste mit dem größten Prozentsatz wird mit der *max*-Funktion von Python ermittelt.
Schließlich wird der dominante RGB-Wert des Bildes in einer CSV-Datei gespeichert. Darüber hinaus erhält der extrahierte RGB-Wert in der CSV-Datei ein *label* zur Kennzeichnung des zugehörigen Materials. Für die Kennzeichnung der Materialien verwenden wir die *One-Hot*-Kodierung, bei der diskrete *label* in binäre *label* umgewandelt werden. Das bedeutet, dass die *label* *Concrete* und *Masonry* einen Wert von 0 oder 1 erhalten, wobei der Wert 1 die zutreffende Materialklasse und der Wert 0 die nicht zutreffende Materialklasse definiert. Zum besseren Verständnis wird in der Tabelle 3.2 die Struktur der gekennzeichneten Daten mit einer *One-Hot*-Kodierung beispielhaft dargestellt. Des Weiteren zeigt die Abbildung 3.18 die dominanten Farben der Bilddatensätze für Beton und Mauerwerk, die mit Hilfe des beschriebenen Verfahrens extrahiert wurden. Aus der Abbildung 3.18 ist ersichtlich, dass eine große Variation der Oberflächenfarben für die Materalien in den Datensätzen berücksichtigt wurde.

Tabelle 3.2: Struktur der gekennzeichneten Daten unter Verwendung einer *One-Hot*-Kodierung.

#	*Feature*			*Label*	
	R	G	B	*Concrete*	*Masonry*
1	117	41	77	0	1
2	169	167	162	1	0
3	189	185	181	1	0
...	...	...	...	...	...
2000	166	107	82	0	1

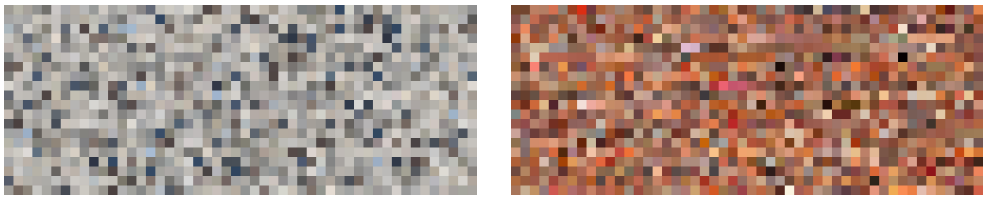


Abbildung 3.18: Generierte Pixelgrafiken der dominanten Farben aus den Bilddatensätzen für *Concrete* (links) und *Masonry* (rechts).

3.6.2 Datenvorverarbeitung

Um die Materialklassen *Concrete* und *Masonry* auf der Grundlage eines RGB-Wertes mit DL zu klassifizieren, müssen die aus Unterabschnitt 3.6.1 gewonnenen CSV-Dateien zusammengeführt und gemischt werden. Die Mischung der Daten ist ein wichtiger Schritt bei der Datenvorverarbeitung, um sicherzustellen, dass das DL-Modell generalisierte Muster erkennt. Die generierten *features* (RGB-Werte) und die zugehörigen *labels* (*Concrete*, *Masonry*) werden anschließend in separaten CSV-Dateien gespeichert. Die Trennung der *features* und *labels* in separate Dateien ist erforderlich, da die *features* die vom DL-Modell verarbeiteten Eingabedaten darstellen, während die *labels* die vom Modell vorhergesagten Ausgaben sind. Zum Trainieren und Testen des DL-Modells werden die Daten in einem Verhältnis von 80/20 aufgeteilt, wobei 80% als Trainingsdaten und 20% als Testdaten dienen. Diese Aufteilung ist im Bereich des maschinellen Lernens weit verbreitet und hat sich als effektiv erwiesen. Nach der beschriebenen Vorverarbeitung der Daten kann im nächsten Schritt das Training des DL-Modells durchgeführt werden.

3.6.3 Deep-Learning-Modell

DNNs sind Algorithmen des maschinellen Lernens, die aus mehreren Schichten (engl. *layer*) bestehen, nämlich einer Eingabeschicht (engl. *input layer*), einer oder mehreren verborgenen Schichten (engl. *hidden layer*) und einer Ausgabeschicht (engl. *output layer*). Die Anzahl der verborgenen Schichten hängt dabei vom Anwendungsfall ab und kann beliebig sein. In jeder Schicht eines neuronalen Netzwerkes existieren miteinander verzweigte Knoten, die auch als Neuronen bezeichnet werden. Jedem Neuron werden Gewichtungen (engl. *weights*) in Form von numerischen Werten zugewiesen, die angeben, wie wichtig eine bestimmte Eingabe für die Ausgabe des Neurons ist. Während der Trainingsphase werden die Gewichtungen der Neuronen so angepasst, dass ein bestimmtes Muster erlernt wird. Die endgültigen Gewichtungen werden dann für die Evaluation und Anwendung des trainierten Modells verwendet.
Für das Training des DL-Modells wird die *Keras*-API [38] aus der *TensorFlow*-Bibliothek [8] verwendet. Das konstruierte DNN besteht aus fünf Schichten, nämlich drei verborgenen Schichten sowie einer Eingabe- und einer Ausgabeschicht. In der Abbildung 3.19 ist eine schematische Darstellung des aufgebauten DNNs dargestellt. Für jede Schicht müssen die Anzahl der Neuronen und eine Aktivierungsfunktion definiert werden. Eine Aktivierungsfunktion ist eine nichtlineare mathematische Funktion, die auf den Ausgang jedes Neurons in einer Schicht eines neuronalen Netzes angewendet wird. Dadurch wird eine Nichtlinearität in das Modell eingeführt, die es ermöglicht, komplexe Beziehungen zwischen Eingabe- und Ausgabedaten zu erlernen [126]. Hierbei wurde die ReLU-Funktion (engl. *Rectified Linear Unit*, ReLU) als geeignete Aktivierungsfunktion für den Aufbau der Schichten im DNN gewählt, da sie sich in vielen DL-Anwendungen bewährt hat und effizient ist. Für die Eingabeschicht sind drei Neuronen definiert. Die Anzahl der Neuronen in den verborgenen Schichten sind wie folgt festgelegt:

- *Hidden Layer* 1: 14 Neuronen,
- *Hidden Layer* 2: 14 Neuronen,
- *Hidden Layer* 3: 6 Neuronen.

Der Ausgabeschicht wird die Aktivierungsfunktion *softmax* zugewiesen und mit zwei Neuronen definiert, wobei in der Ausgabeschicht die Anzahl der Neuronen die Anzahl der *label* im Netzwerk darstellt. Aus diesem Grund darf die Anzahl der Neuronen in der Ausgabeschicht nicht vom Benutzer geändert werden.

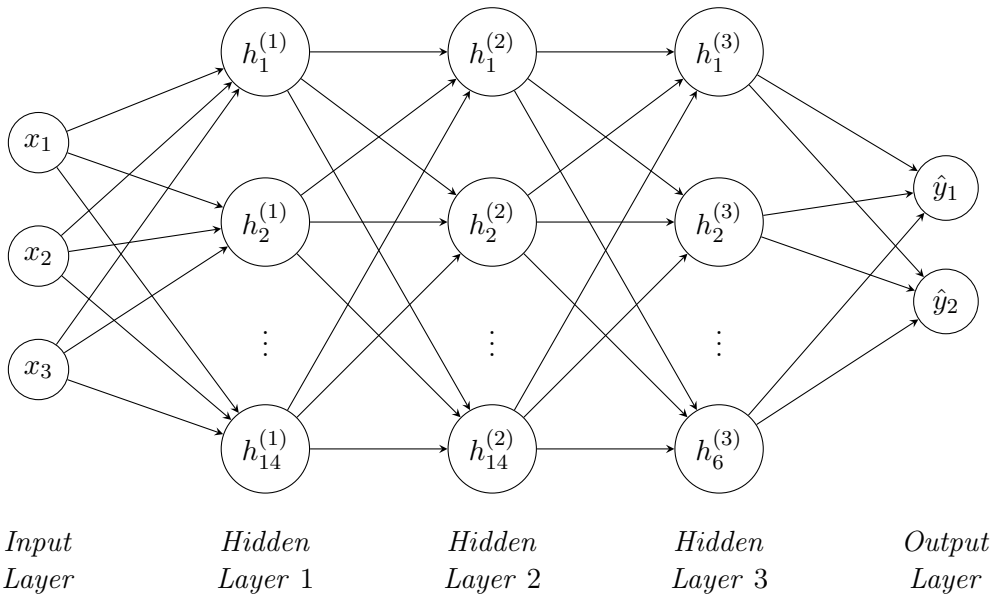


Abbildung 3.19: Schematische Darstellung des erzeugten DNNs mit fünf Schichten.

3.6.4 Training und Testen des Modells

Das Trainieren eines DNNs mit Trainingsdaten erfordert die Wahl eines Optimierungsalgorithmus für den Lernprozess und die Festlegung einer *Loss*-Funktion, um die vorhergesagten Ausgaben des Modells zu bewerten. In Anbetracht seiner Effizienz im Bereich des maschinellen Lernens wurde vom Autor dieser Arbeit der *Adam*-Optimierer von der *Keras*-API [38] als Optimierungsalgorithmus gewählt. Als *Loss*-Funktion für ein DNN wird standardmäßig die bewährte *Categorical-Cross-Entropy*-Funktion in Kombination mit der *softmax*-Aktivierungsfunktion verwendet [123]. Des Weiteren ist die *Categorical-Cross-Entropy*-Funktion für die Verarbeitung von Trainingsdaten im *One-Hot*-Format geeignet [48] und wird daher für diesen Anwendungsfall gewählt. Das aufgebaute DL-Modell wurde anhand der in Unterabschnitt 3.6.1 vorgestellten Trainingsdaten trainiert, wobei 20% dieser Daten zur Validierung verwendet wurden. Das Ergebnis der Validierung wird als Validierungsverlust (engl. *validation loss*) bezeichnet. Dieser Wert wird verwendet, um die Leistungsfähigkeit des Modells zu überwachen und das Risiko einer Überanpassung zu minimieren. Die Anzahl der Epochen wurde auf 400 festgelegt, was bedeutet, dass die intern gespeicherten Gewichtungen während des Trainings 400 Mal aktualisiert wurden. Dabei wurden die Trainingsdaten vor jeder Epoche gemischt, um die Generalisierung der Daten zu gewährleisten. Nach Abschluss des Trainings wurde die Genauigkeit und Leistungsfähigkeit des Modells mit dem Testdatensatz überprüft. Dabei wurde eine durchschnittliche Genauigkeit von 98% erreicht. Dies bedeutet, dass das trainierte DL-Modell in der Lage war, die *labels* der

Testdaten mit einer Genauigkeit von 98% korrekt vorherzusagen. Der Ausgabeplot der Modellgenauigkeit in Abbildung 3.20 (links) zeigt, dass das Modell während des Trainingsprozesses in Bezug auf die Vorhersagegenauigkeit der *labels* konvergierte. Zudem ist aus dem Ausgabeplot des Modellverlustes in Abbildung 3.20 (rechts) ersichtlich, dass sowohl der Trainings- als auch der Validierungsverlust im Laufe des Trainingsprozesses abnahmen, was darauf hinweist, dass das Modell nicht übermäßig an den Trainingsdatensatz angepasst wurde. Schließlich wurde das trainierte DL-Modell lokal als HDF5-Datei [4] gespeichert.

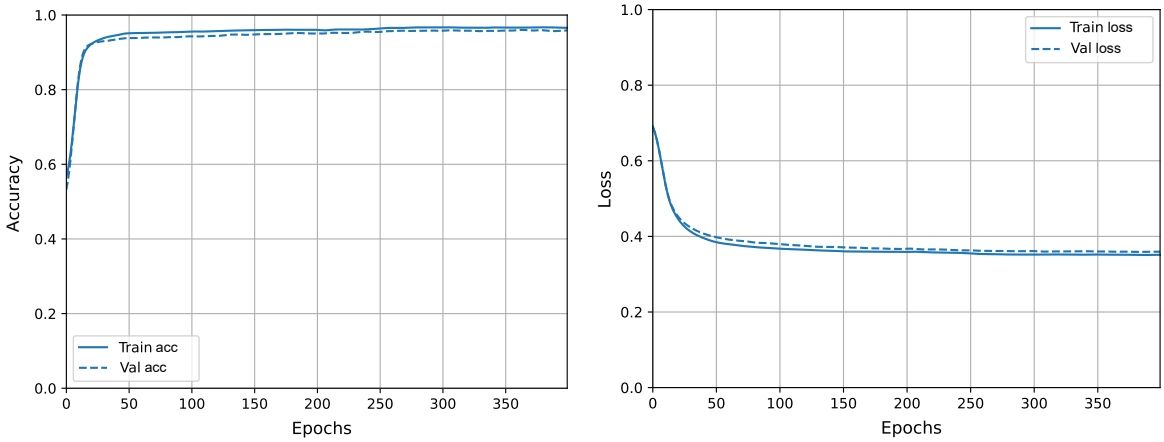


Abbildung 3.20: Ausgabeplots des Lernprozesses unter Verwendung des erstellten DNNs, die die Modellgenauigkeit (links) und den Modellverlust (rechts) zeigen.

3.6.5 Arbeitsablauf der Materialklassifizierung

Im Zuge des Arbeitsablaufes der algorithmischen Datenverarbeitung wird das trainierte DL-Modell verwendet, um die Materialklasse einer Punktmenge P_i auf der Grundlage eines RGB-Wertes vorherzusagen, der in P_i am häufigsten vorkommt. Der allgemeine Arbeitsablauf ist als Pseudocode in Algorithmus 7 zusammengefasst und wird im Folgenden erläutert:
Im ersten Schritt werden die RGB-Daten C_i der Punktmenge P_i und die Objektklasse (Abschnitt 3.5) des BC-Objektes aus der PKL-Datei IV in die Liste Ψ eingelesen. Da im Rahmen dieser Arbeit Stahlbetonrahmenkonstruktionen als Gebäudetyp betrachtet werden, wird den BC-Objekten, die die Objektklasse *Slab*, *Base*, *Beam*, *Column* oder *Beam*|*Column* enthalten, automatisch die Materialklasse *Concrete* zugewiesen, da diese Bauteile immer aus Stahlbeton bestehen. Punktmengen, die mit der vorgestellten Methode in Abschnitt 3.5 keiner Objektklasse zugeordnet werden konnten, erhalten die Materialklasse *Unclassified*. Für BC-Objekte mit der Objektklasse *Wall* oder *Slab*|*Wall*, bei denen es sich um Beton oder Mauerwerk als Material handeln kann, wird der dominante RGB-Wert aus C_i mit der in Algorithmus 6 beschriebenen Methode extrahiert, wobei hier die RGB-Daten C_i als Eingabe dienen und nicht aus einer Rastergrafik ermittelt werden müssen. Der ausgegebene RGB-Wert RGB_{dom}, der am häufigsten in C_i vorkommt, wird anschließend als Eingabe für das trainierte DL-Modell *model* verwendet, um die Materialklasse des BC-Objektes vorherzusagen. Die Ergebnisse der Vorhersage mit *model* sind zum einen die Materialklasse, die entweder *Concrete* oder *Masonry* ist, und zum anderen der *confidence*-Wert, der zwischen 0 und 1 liegt. Der *confidence*-Wert ist in diesem Zusammenhang ein numerischer Wahrscheinlichkeitswert, der angibt, wie zuverlässig die vorhergesagte Materialklasse für den gegebenen RGB-Wert ist. Zur Verdeutlichung: Ein *confidence*-Wert von 100% würde bedeuten, dass das Modell äußerst zuversichtlich in seiner Vorhersage ist. Im Gegensatz

dazu deutet ein *confidence*-Wert von 0% darauf hin, dass das Modell überhaupt keine Zuversicht in seine Vorhersage hat und als äußerst unsicher gilt. Um sicherzustellen, dass nur verlässliche Vorhersagen mit dem trainierten DL-Modell berücksichtigt werden, werden nur die mit *model* vorhergesagten Materialklassen für die BC-Objekte akzeptiert, die einen *confidence*-Wert größer oder gleich 80% aufweisen. Andernfalls wird dem BC-Objekt die Materialklasse *Unclassified* zugewiesen. Nachdem die Materialklassifizierung für alle BC-Objekte in Ψ durchgeführt wurde, werden die aktualisierten BC-Objekte unter Berücksichtigung der Materialklasse in einer neuen PKL-Datei mit der Nummer VII gespeichert.

3.7 Digitale 3D-Rekonstruktion der sichtbaren Bauteile

Eine der Herausforderungen der automatischen Modellgenerierung besteht darin, aus den punktuellen Daten ein kohärentes und sinnvolles 3D-Modell zu erstellen, das die geometrischen Eigenschaften der Objekte richtig wiedergibt. In diesem Abschnitt wird eine automatische Methode zur digitalen Rekonstruktion von segmentierten Punktdaten, die ein Bauteil darstellen, vorgestellt. Dabei werden sowohl Punktmengen mit konvexer und konkaver Geometrie als auch Öffnungen bei der Oberflächenrekonstruktion berücksichtigt.

3.7.1 Vorverarbeitung

Zunächst werden die Punktdaten P_i, die Komponentendicke t_i und der *boolean* Wert *hasParallel* der klassifizierten BC-Objekte aus der PKL-Datei VII gelesen. Um eine zweidimensionale Darstellung der Oberfläche des Bauteils zu erhalten, werden die Punktdaten P_i um ihren Mittelpunkt P_i^c in der XY-Raumebene ausgerichtet. Dadurch ergibt sich die Punktmenge $P_{i,xy}$ (Abbildung 3.8). Ist die Bedingung *hasParallel* erfüllt, d. h. sind die Punktdaten P_i zweier Segmente parallel zueinander, werden die parallelen Segmente mit der *crop2D*-Funktion von *cloudComPy* [40] voneinander getrennt. Zu diesem Zweck wird der Schnittbereich mit Hilfe der *Bounding Box* der Punktmenge $P_{i,xy}$ in der YZ-Ebene als Polygonzug definiert. Der Schnittbereich umfasst dabei die halbe Höhe der *Bounding Box* von $P_{i,xy}$, sodass eine parallele Komponente außerhalb des Schnittbereiches verbleibt. Durch die Anwendung der *crop2D*-Funktion erhalten wir die Punktmengen $P_{i,xy}^{in}$ und $P_{i,xy}^{out}$, wobei $P_{i,xy}^{in}$ die Punkte innerhalb des Schnittbereiches und $P_{i,xy}^{out}$ die Punkte außerhalb des Schnittbereiches enthält. Eine Veranschaulichung dieser Prozedur ist in der Abbildung 3.21 dargestellt. Das Entfernen des parallelen Segments ist ein notwendiger Schritt, um sicherzustellen, dass der Körper des BC-Objektes auf Basis der Oberflächenextrusion (Unterabschnitt 3.7.3) korrekt erzeugt wird. Hierbei wird die Oberflächenrekonstruktion und -extrusion auf die Punktmenge angewendet, die mehr Punktdaten enthält. Wenn die Anzahl der Punktdaten in $P_{i,xy}^{in}$ größer ist als die in $P_{i,xy}^{out}$, werden für $P_{i,xy}$ die Punktdaten von $P_{i,xy}^{in}$ verwendet und die Variable *cropSec* erhält den Wert 1. Andernfalls werden die Daten von $P_{i,xy}^{out}$ für $P_{i,xy}$ angesetzt und *cropSec* erhält den Wert 2. Ist die Bedingung *hasParallel* nicht erfüllt, wird dem Attribut *cropSec* der Wert 0 zugewiesen, da in diesem Fall die Punktdaten P_i des BC-Objektes keine parallele Oberflächenkomponente enthält. Es ist wichtig zu beachten, dass die Variable *cropSec* die Richtung der Oberflächenextrusion bestimmt, die in Unterabschnitt 3.7.3 behandelt wird.

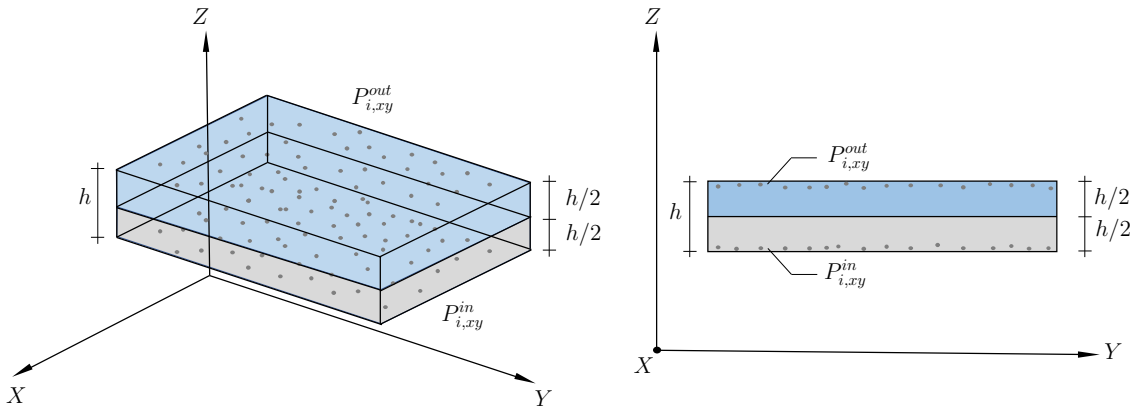


Abbildung 3.21: Bildliche Beschreibung der Definition des Schnittbereiches von $P_{i,xy}$ mit Hilfe seiner *Bounding Box*: Der Polygonzug der grauen Fläche in der YZ-Ebene (rechts) definiert den Schnittbereich von $P_{i,xy}$ und ergibt die Punktmenge $P^{in}_{i,xy}$. Die Punktdaten außerhalb des Schnittbereiches (blauer Bereich) werden als $P^{out}_{i,xy}$ zusammengefasst.

3.7.2 Oberflächenrekonstruktion

Im Rahmen dieser Dissertation wird das α-*Shape*-Verfahren [54, 179] für die digitale Oberflächenrekonstruktion von segmentierten Punktmengen verwendet. Hierbei wird die Oberflächenform durch Auswahl einer Teilmenge von Punkten und eines α-Parameters als Schwellenwert erzeugt. Der α-Parameter, dessen Wert im Bereich von 0 bis 1 liegt ($\alpha \in \{x \in \mathbb{R}^+ \mid 0 \leq x \leq 1\}$), bestimmt dabei die Größe und Form der Oberfläche, die entweder konvex oder konkav sein kann. Ein Wert von $\alpha = 1$ führt zu einer konvexen Oberfläche, während $\alpha < 1$ die konkave Form als Polygonnetz approximiert. Der Rand der Oberfläche wird mit Hilfe eines Umkreises angenähert, dessen Radius durch den α-Parameter berechnet wird. Dieser Umkreis wandert entlang der Randpunkte der Punktmenge und verbindet zwei Punkte, die vom Umkreis berührt werden (Abbildung 3.22). Je kleiner der gewählte α-Wert, desto geringer der Radius des Umkreises und desto präziser wird die konkave Form der Punktmenge approximiert. Die Oberflächenerzeugung aus Punktdaten wird in der α-*Shape*-Methode mit der *Delaunay Triangulation* [26] realisiert. Das berechnete Triangulationsmuster resultiert zu einem geschlossenen Polygonnetz, das die mannigfaltige Geometrie der Punktmenge repräsentiert. In der Abbildung 3.22 ist eine bildliche Beschreibung der Oberflächenrekonstruktion mit der α-*Shape*-Methode dargestellt.

Um α-*Shapes* aus Punktdaten zu erzeugen, wird die in [147] angegebene Funktion für *Wolfram Mathematica* [174] verwendet. Die vollständige Funktion ist im Anhang A.1.4 als Pseudocode aufgeführt. *Wolfram Mathematica* [174] ist ein Programmpaket für mathematisch-naturwissenschaftliche Anwendungen, die eine umfangreiche Sammlung vordefinierter Algorithmen für 2D- und 3D-Berechnungen bietet. Besonders leistungsstark ist der Betriebssystemkern der Software, der als *WolframKernel* bezeichnet wird, welches eine zuverlässige und schnelle Analyse von komplexen Problemen ermöglicht, einschließlich der Verarbeitung von 3D-Daten [174]. Die α-*Shape*-Funktion (Unterabschnitt A.1.4) liest die XY-Punktdaten von $P_{i,xy}$ und erzeugt daraus ein mannigfaltiges Polygonnetz $M_{i,xy}$ basierend auf der *Delaunay Triangulation* [26]. Wie bereits oben erwähnt, kann die Genauigkeit der Triangulation durch den α-Parameter im Bereich von $\alpha \in \{x \in \mathbb{R}^+ \mid 0 \leq x \leq 1\}$ definiert werden, wobei $\alpha = 1$ eine konvexe Triangulation und $\alpha < 1$ eine konkave Approximation ergibt [139]. Es ist wichtig zu beachten, dass der α-Parameter für jedes Objekt individuell angepasst werden muss, um die gewünschte Genauigkeit der Oberflächenrekonstruktion zu erzielen.

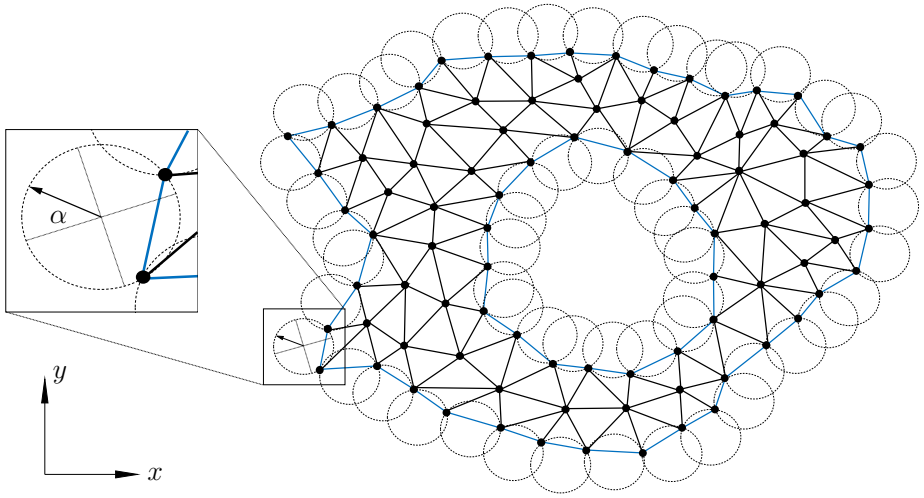


Abbildung 3.22: Bildliche Beschreibung der Oberflächenrekonstruktion mit der *α-Shape*-Methode am Beispiel einer Punktmenge in der XY-Ebene. Die blauen Linien repräsentieren die Ränder der Punktmenge, die mit Hilfe der dargestellten Umkreise erzeugt wurden. Die schwarzen Linien stellen das Polygonnetz dar, das durch Anwendung der *Delaunay Triangulation* auf die Punktdaten erstellt wurde.

3.7.3 Oberflächenextrusion

Die Oberflächenextrusion ist ein Konzept der algorithmischen Geometrie, das die Dimensionserweiterung eines zweidimensionalen Objektes durch die parallele Verschiebung seiner Fläche entlang einer Achse ermöglicht. Dabei entsteht ein neuer Körper in $\mathbb{R}^3$ als Ergebnis der Extrusion. Die Python-Bibliothek *Trimesh* [47] bietet in diesem Zusammenhang eine vordefinierte Funktion für die Extrusion von Flächen in einem Polygonnetz. Der prinzipielle Ablauf einer Oberflächenextrusion mit *Trimesh* [47] wird in der Abbildung 3.23 beschrieben.

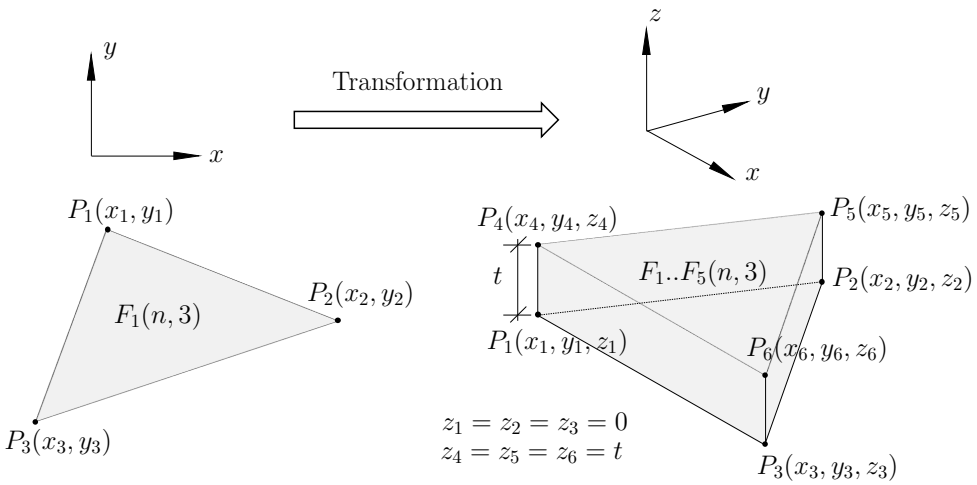


Abbildung 3.23: Prinzipskizze zur Beschreibung der Oberflächenextrusion mit der Python-Bibliothetk *Trimesh* [47] am Beispiel einer zweidimensionalen Dreiecksfläche (links), die entlang der z-Achse zu einem dreidimensionalen Körper (rechts) mit der Dicke t extrudiert wird [138, 139].

Die Polygonfläche wird durch Eckpunkte $P_n(x_n, y_n)$ in der Form $(n, 2)$ und Flächen F_n in der Form $(n, 3)$ definiert. Nach der Oberflächenextrusion werden die Eckpunkte $P_n(x_n, y_n)$ der zweidimensionalen Polygonfläche F_n um die z-Dimension erweitert, indem die Koordinate z_n auf Null gesetzt wird. Daraus resultieren die Eckpunkte $P_n(x_n, y_n, z_n = 0)$ in $\mathbb{R}^3$. Durch die Parallelverschiebung der Eckpunkte von F_n entlang der z-Achse ergeben sich die neuen Koordinaten $P_n(x_n, y_n, z_n = t)$, wobei t den festgelegten Extrusionswert darstellt (Abbildung 3.23) [139]. Die Eckpunkte und Flächen des erzeugten Polygonnetzes $M_{i,xy}$ aus Unterabschnitt 3.7.2 dienen als Eingabedaten zur Anwendung der oben beschriebenen Oberflächenextrusion, um einen geschlossenen Körper für das BC-Objekt zu erstellen. Für die Extrusion wird der Wert der Komponentendicke t_i des BC-Objektes verwendet, welches die Bauteildicke repräsentiert. Die Richtung der Extrusion entlang der z-Achse hängt vom Wert des Attributs *cropSec* ab. Wenn der Wert der Variable *cropSec* entweder 0 oder 2 beträgt, wird die Oberfläche des Polygonnetzes $M_{i,xy}$ in die negative z-Richtung extrudiert. Im Falle eines Attributwertes von 1 erfolgt die Extrusion in positiver z-Richtung. Der resultierende Körper wird als $M^E_{i,xy}$ bezeichnet.

3.7.4 Transformation

Um das BC-Objekt nach der Oberflächenextrusion zurück in seine ursprüngliche Position zu transformieren, sind eine Translation und Rotation des Körpers $M^E_{i,xy}$ erforderlich. Hierbei wird für die Rotation von $M^E_{i,xy}$ die inverse Rotationsmatrix R^{-1} verwendet, welche zuvor zur Ausrichtung der Punktdaten P_i in der XY-Ebene angewendet wurde. Der Verschiebungsvektor für die Translation des extrudierten Körpers $M^E_{i,xy}$ entlang der z-Achse ergibt sich aus der Differenz zwischen dem Mittelpunkt P^c_i von P_i und der Ebene $z = 0$. Schließlich wird der transformierte Körper im OBJ-Format [121] gespeichert.

Die beschriebenen Schritte zur Erzeugung eines geschlossenen Körpers für das BC-Objekt sind in Algorithmus 9 als Pseudocode zusammengefasst.

3.8 Digitale 3D-Rekonstruktion der verdeckten Bauteile

Wie bereits in Abschnitt 3.1 erwähnt wurde, ist derzeit keine Fernerkundungsmethode in der Lage, massive Materialien wie Stahlbeton zu durchdringen. Daher ist es nicht möglich, Bauteilflächen eines Stahlbetonrahmens, die von der Gebäudefassade verdeckt werden, mit aktiven oder passiven Sensoren zu erfassen. Die in Abschnitt 3.7 vorgeschlagene Rekonstruktionsmethode kann deshalb nur auf die von außen sichtbaren Bauteile eines Stahlbetonrahmens angewendet werden, die in der 3D-Punktwolke dargestellt sind. Die vorliegende Arbeit konzentriert sich primär auf die Entwicklung einer Methode zur digitalen 3D-Rekonstruktion der sichtbaren Bauteile von seismisch beschädigten Stahlbetonrahmen anhand von 3D-Punktwolken (Abschnitt 1.2). Dennoch wird in diesem Zusammenhang eine ergänzende Methode zur digitalen 3D-Rekonstruktion der verdeckten Bauteile vorgeschlagen. Dies soll als Anregung und möglicher Ausgangspunkt für weitere Forschungsbemühungen dienen, ohne den primären Fokus dieser Arbeit zu verändern.

3.8.1 Arbeitsablauf

In Anlehnung an die in Unterabschnitt 2.3.2 zusammengefasste Methode von *Bloch et al.* [25], basiert die vorgeschlagene Modellierungsstrategie für die verdeckten Bauteile auf der Extraktion von Informationen aus Schadensmodellen, die durch numerische Kollapssimulationen erzeugt werden. Während *Bloch et al.* [25] das *as-built* BIM-Modell als Grundlage für die Vorhersage des beschädigten Innenraummodells voraussetzen, konzentriert sich unser Vorschlag auf einen modellunabhängigen Ansatz, d. h. ohne die Verwendung eines digitalen 3D-Modells des Gebäudes

in seinem intakten Zustand. Die Abbildung 3.24 fasst den vorgeschlagenen Arbeitsablauf der modellunabhängigen 3D-Rekonstruktion von verdeckten Bauteilen schematisch zusammen.

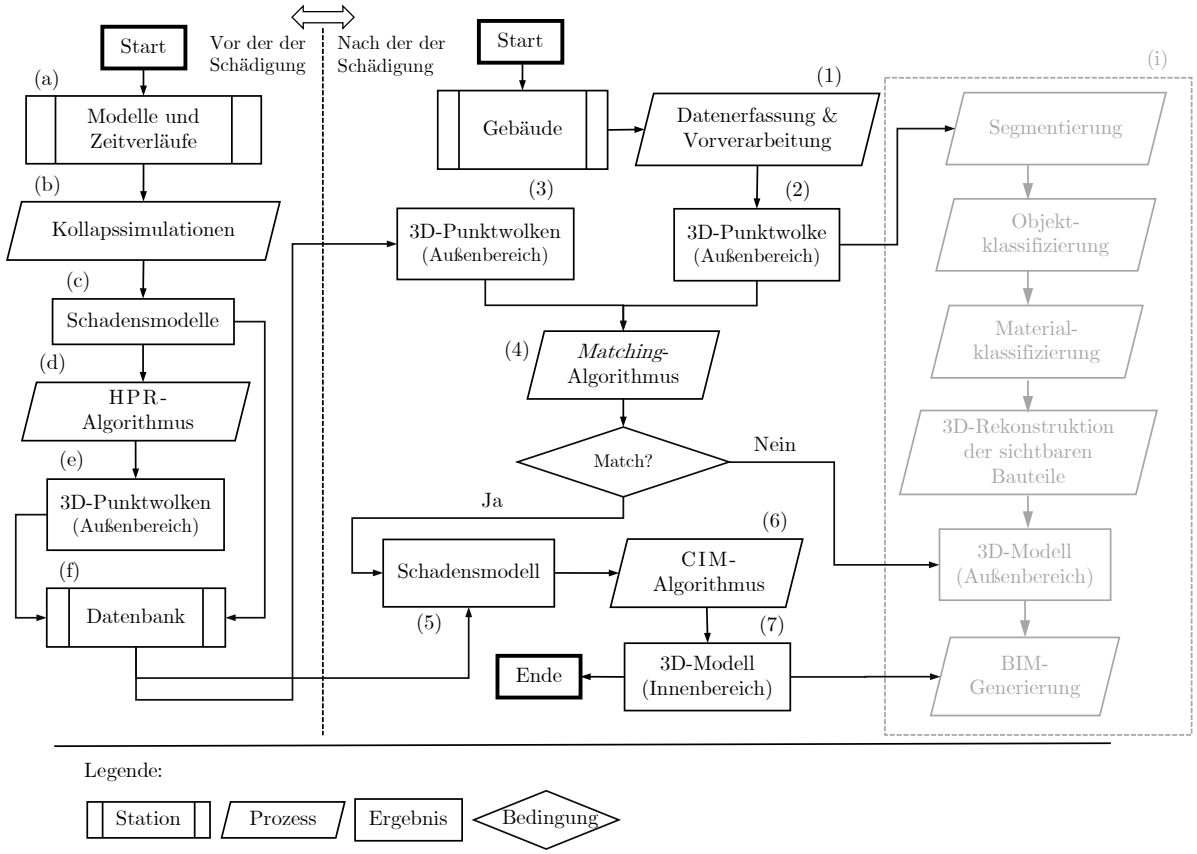


Abbildung 3.24: Schematischer Arbeitsablauf der modellunabhängigen 3D-Rekonstruktion von verdeckten Bauteilen in Form eines Flussdiagramms.

Vor dem Auftreten von seismisch induzierten Gebäudeschäden steht eine Datenbank (Abbildung 3.24 (f)) zur Verfügung, die Schadensmodelle (Abbildung 3.24 (c)) von verschiedenen Stahlbetonrahmentragwerken (Abbildung 3.24 (a)) enthält. Die Schadensmodelle werden dabei durch numerische Kollapssimulationen (Abbildung 3.24 (b)) erzeugt. Neben den 3D-Modellen enthält die Datenbank auch die zugehörigen 3D-Punktwolken der Schadensmodelle (Abbildung 3.24 (e)). Diese werden mit Hilfe des HPR-Algorithmus (engl. *Hidden-Point-Removal*, HPR) von *Katz et al.* [94] synthentisch erzeugt. Dieser Algorithmus berücksichtigt wie bei der SfM-Photogrammetrie nur die Punktdaten, die sich im sichtbaren Blickwinkel der Kamera befinden. Da sich die Datenerfassung (Abbildung 3.24 (1)) nach der seismisch induzierten Schädigung der Stahlbetonrahmenkonstruktion mittels aktiver oder passiver Fernerkundung auf die von außen sichtbaren Bauteiloberflächen beschränkt, werden nur die von außen erfassten Punktdaten der Schadensmodelle (Abbildung 3.24 (3)) zum Vergleich mit der generierten 3D-Punktwolke (Abbildung 3.24(2)) des beschädigten Gebäudes herangezogen. Der Vergleich der Punkdaten (Abbildung 3.24 (2) und (3)) erfolgt mit dem vom Autor erstellten *Matching*-Algorithmus. Der *Matching*-Algorithmus sucht nach der höchsten geometrischen Übereinstimmung zwischen einer Basispunktwolke und einer Zielpunktwolke, indem der kleinstmögliche durchschnittliche Abstand zwischen den XYZ-Punktdaten iterativ ermittelt wird. Ist die Auswahlbedingung für die Datenübernahme erfüllt, wird das entsprechende Schadensmodell (Abbildung 3.24 (5)) aus der Datenbank extrahiert und

für den weiteren Arbeitsablauf verwendet. Schließlich werden die verdeckten Bauteile (Abbildung 3.24 (7)) des ausgewählten Schadensmodells basierend auf den gesammelten Punktdaten des Gebäudes mit Hilfe des CIM-Algorithmus (Abbildung 3.24 (6)) (engl. *Crop-Interior-Mesh*, CIM) ausgeschnitten und dem 3D-Modell des Außenbereiches (Abbildung 3.24 (i)) hinzugefügt. Der CIM-Algorithmus wurde vom Autor dieser Arbeit entwickelt und verwendet eine iterative Methode, um die Eckpunkte und Flächen der Strukturelemente des Schadensmodells innerhalb der Hülle des beschädigten Gebäudes zusammenzustellen. Zur Vollständigkeit wird der Prozess der algorithmischen Datenverarbeitung aus Abbildung 3.1 in Abbildung 3.24 (i) dargestellt, wobei die potenzielle Verknüpfung mit der oben beschriebenen Methode verdeutlicht wird.

3.8.2 Datenbank

Die genaue Vorhersage des Verhaltens von Stahlbetonrahmentragwerken unter starker seismischer Belastung ist eine komplexe Aufgabe aus dem Bereich des Erdbebeningenieurwesens. Abweichungen im Materialverhalten oder in der Belastung können das Ergebnis einer Simulation stark beeinflussen, was es schwierig macht, das Einsturzverhalten mit nur einer Kollapssimulation korrekt vorherzusagen. In Notfallsituationen, wie bei USaR-Maßnahmen, ist es in der Regel nicht möglich, ein digitales 3D-Modell des betroffenen Gebäudes oder eine Aufzeichnung des Erdbeben-Zeit-Verlaufes für numerische Kollapssimulationen zu nutzen. Um dieses Problem zu überbrücken, wird vom Autor dieser Arbeit die Erstellung einer Datenbank vorgeschlagen, die verschiedene Schadensmodelle aus Kollapssimulationen enthält. Durch die Verwendung dieser Datenbank kann die Wahrscheinlichkeit erhöht werden, dass ein Modell aus der Datenbank dem tatsächlichen Schadensbild der beschädigten Stahlbetonrahmenkonstruktion entspricht.

3.8.2.1 Kollapssimulation

Die Literaturübersicht in Abschnitt 2.4 zeigt, dass die auf der *Bullet*-Physik-Engine basierende Starrkörperdynamik in *Blender* [23] in Kombination mit dem BCB [128] zu einer zuverlässigen Vorhersage der Einsturzform von Gebäuden in Kollapssimulationen führen kann. In diesem Zusammenhang generiert der BCB [128] automatisch die Zwangsbedingungen zwischen den Starrkörperelementen unter Verwendung vereinfachter Materialgesetze, die in Form von Senk- und Drehfedern modelliert werden. Auf diese Weise ermöglicht die Physik-Engine-basierte Starrkörperdynamik eine einfache und schnelle Simulation des Verhaltens eines Tragwerkes unter extremer Belastungen, wie z. B. einem Erdbeben. Im Gegensatz dazu erfordern herkömmliche Simulationsmethoden wie die FEM oder die AEM eine aufwendige Vorbereitung des Berechnungsmodells und beanspruchen aufgrund der inkrementellen Lösung des Gleichgewichts für jeden Knoten in den Elementen erheblich mehr Zeit für die Simulationen. Für die Erzeugung der Schadensmodelle unter seismischer Belastung bietet sich die Physik-Engine-basierte Starrkörperdynamik mit dem BCB [128] in *Blender* [23] aus folgenden Gründen an:

- *Open-Source*-Software mit Unterstützung diverser CAD- und *Mesh*-Formate für den Import und Export.
- Unterstützung der BIM-Anwendung mit IFC-Schnittstelle.
- Einfache und schnelle Erzeugung des mechanischen Berechnungsmodells mit dem BCB-Add-on [128].
- Eingabemöglichkeit für drei gleichzeitig wirkende Erdbeben-Zeit-Verläufe in x-, y- und z-Richtung.
- Kurze Simulationsdauer im Vergleich zu FEM oder AEM.
- Trennung der Starrkörperelemente durch Versagen der mechanischen Zwangsbedingungen (Senk- und Drehfedern).

Aufgrund der oben genannten Vorteile wird vom Autor dieser Arbeit die Physik-Engine-basierte Methode der Starrkörperdynamik mit dem BCB [128] in *Blender* [23] für die Simulation der Stahlbetonrahmentragwerke unter Erdbebenbelastung vorgeschlagen, um eine Datenbank mit Schadensmodellen zu generieren.

3.8.2.2 HPR-Algorithmus

Die in Abbildung 3.24 vorgestellte Methode beinhaltet einen Vergleich zwischen der aus der Datenerfassung generierten 3D-Punktwolke und den Schadensmodellen in der Datenbank. Um diesen Vergleich durchzuführen, werden die Schadensmodelle in äquivalente 3D-Punktwolken umgewandelt. Dabei werden nur die Bauteile berücksichtigt, die von außen sichtbar sind. Dies erfolgt durch die digitale Simulation der auf Laserscanning oder Photogrammetrie basierenden Datenerfassung für die Schadensmodelle. Zu diesem Zweck wird der HPR-Algorithmus von *Katz et al.* [94] verwendet. Die vom Autor dieser Arbeit entwickelte Methode zur Erzeugung einer Punktwolke aus der Hülle eines 3D-Modells ist in Algorithmus 10 als Pseudocode zusammengefasst und wird im Folgenden beschrieben:
Zunächst werden alle Oberflächen des Schadensmodells mit dem Python-Modul *Open3D* [186] in 3D-Punktdaten umgewandelt. Dabei kann der Benutzer die gewünschte Punktdichte festlegen. Anschließend wird der in *Open3D* [186] implementierte HPR-Algorithmus von *Katz et al.* [94] auf die erzeugte Punktwolke an verschiedenen Positionen angewendet. Der HPR-Algorithmus speichert dabei nur die Punkte der Punktwolke, die innerhalb des Blickwinkels der *Open3D*-Kamera liegen, d. h. die Oberflächen der sichtbaren Bauteile. Um sicherzustellen, dass die Punktwolke P_i des Schadensmodells vollständig von Außen erfasst wird, werden die *Open3D*-Kameras auf vier verschiedenen Ebenen in z-Richtung platziert, die auf die Höhe der *Bounding Box* von P_i ausgerichtet sind. Eine Beschreibung der Kamerapositionen ist in der Tabelle 3.3 aufgelistet. Hierbei steht bb_h für die Höhe der umgebenden *Bounding Box* der Punktwolke P_i.

Tabelle 3.3: Kamerapositionen für den HPR-Algorithmus.

Ebene i	Höhe (z)	Anzahl Kameras #	Winkelschritt (φ_i)	Kamerapositionen
1	0	36	10	Kreis um Punktwolke auf Bodenhöhe
2	$bb_h \cdot 0.5$	36	10	Kreis um Punktwolke in mittlerer Höhe
3	$bb_h \cdot 1.5$	12	30	Kreis um Punktwolke in größerer Höhe
4	$bb_h \cdot 2.0$	1	-	Zentrisch über Punktwolke

In jeder der in Tabelle 3.3 aufgelisteten Ebenen wird eine *for*-Schleife ausgeführt, die dazu dient, die HPR-Punktdaten an den jeweiligen Kamerapositionen in einer Liste zu speichern. Die x- und y-Koordinaten der Kameras ergeben sich aus den folgenden Formeln:

$$x_{\text{cam}} = r \cdot \cos(\varphi_i), \tag{3.37}$$

$$y_{\text{cam}} = r \cdot \sin(\varphi_i), \tag{3.38}$$

wobei r den Radius des Kreises und φ_i den Winkelschritt definiert. Schließlich werden alle HPR-Punktdaten aus der Liste im XYZ-Format gespeichert. In der Abbildung 3.25 ist die Anwendung des HPR-Algorithmus am Beispiel eines Schadensmodells dargestellt. Die Koordinatensysteme (Abbildung 3.25 rechts) geben dabei die Positionen der Kameras an, die für den HPR-Algorithmus in den Ebenen 1 bis 4 verwendet wurden. Ziel ist es, dass die Datenbank für jedes Schadensmodell eine entsprechende 3D-Punktwolke enthält, die mit der in Algorithmus 10 vorgeschlagenen Methode schnell und einfach erzeugt werden kann.

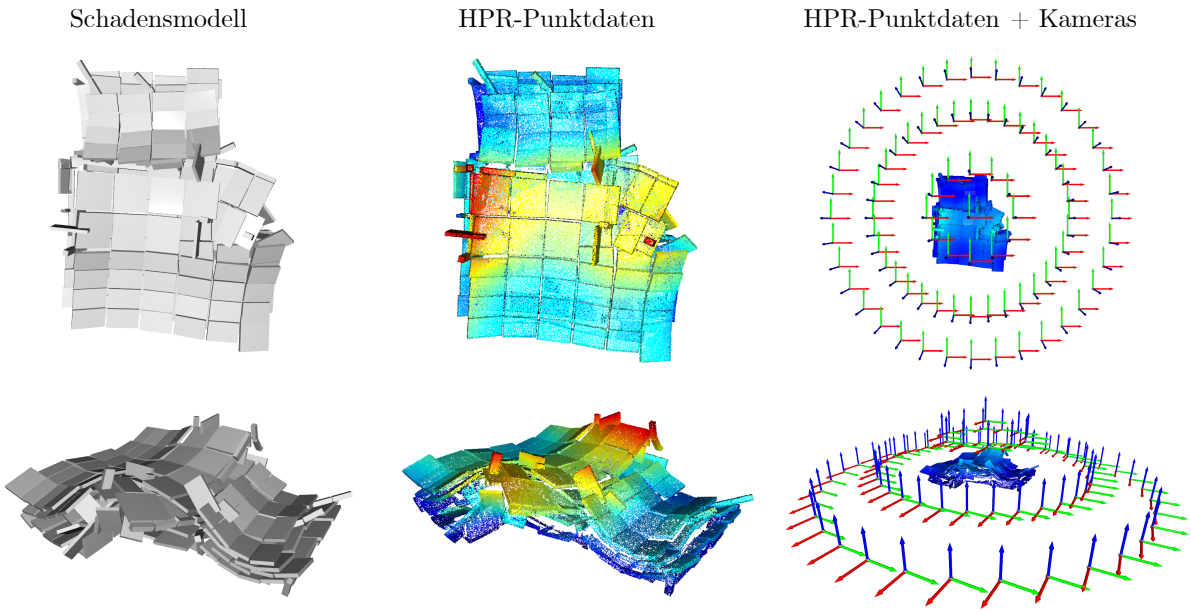


Abbildung 3.25: Ausgabe des HPR-Algorithmus (mittig und rechts) am Beispiel eines Schadensmodells (links): Draufsicht (oben) und Perspektivenansicht (unten).

3.8.3 Matching-Algorithmus

Wir nehmen an, dass die geometrische Übereinstimmung zwischen zwei Modellen durch ein prozentuales Ähnlichkeitsmaß gemessen wird. Dieses Maß kann durch eine Funktion S berechnet werden, die als Eingabe zwei Punktwolken P und P_i erhält. Dabei steht P für die erzeugte Punktwolke des beschädigten Tragwerkes und P_i für die HPR-Punktdaten des i-ten Modells in der Datenbank mit $i \in \mathbb{N}$. Die Funktion S gibt eine Zahl im Intervall $[0, 1]$ zurück:

$$S(P, P_i) \in [0, 1]. \tag{3.39}$$

Die Übereinstimmung wird als gegeben definiert, wenn das Ähnlichkeitsmaß S zwischen P und P_i größer oder gleich einem vorgegebenen Schwellenwert R ist:

$$S(P, P_i) \geq R. \tag{3.40}$$

Das Ergebnis der Funktion S gibt an, wie ähnlich die Modelle P und P_i sind. Dabei bedeutet $S(P, P_i) = 1$, dass die Modelle vollständig übereinstimmen und $S(P, P_i) = 0$, dass sie keine geometrischen Ähnlichkeiten aufweisen. Die zu diesem Zweck entwickelte Methode, die als Pseudocode in Algorithmus 11 zusammengefasst ist, wird im Folgenden beschrieben:
Zu Beginn des Prozesses wird die 3D-Punktwolke P des digital rekonstruierten Tragwerkes eingelesen. Anschließend wird die Höhe der *Bounding Box* bb_h von P entlang der globalen z-Achse bestimmt. Diese Höhe dient als Filterkriterium, um nur Schadensmodelle aus der Datenbank zu berücksichtigen, die eine maximale Abweichung von 15% von der Modellhöhe bb_h aufweisen. Da die Breite und Tiefe einer *Bounding Box* von der Ausrichtung des Objektes im 3D-Raum abhängen, wird nur die Höhe der *Bounding Box*, die von der Orientierung der Punktdaten in der XY-Ebene unabhängig ist, als zuverlässiges Maß für die Filterbedingung gewählt. Wenn die HPR-Punktdaten P_i eines Schadensmodells aus der Datenbank das oben genannte Kriterium erfüllt, wird die Auflösung der entsprechenden HPR-Punktwolke P_i und der Eingabepunktwolke P auf 10 cm reduziert, um den Rechenaufwand zu reduzieren.

Die Punktwolken P und P_i werden bezüglich ihres Schwerpunktes im Ursprung des Koordinatensystems verschoben, wobei die z-Koordinaten unverändert bleiben. Anschließend wird die Eingabepunktwolke P iterativ mit einem Winkelschritt φ_i von 10 Grad einmal vollständig um die z-Achse gedreht, um die Orientierung mit der höchsten Übereinstimmung zwischen P und P_i zu ermitteln. Dieser Ansatz wird in der Abbildung 3.26 veranschaulicht. Nach jeder Iteration wird die Funktion *compute_point_distance* von *Open3D* [186] verwendet, um die Abstände zwischen den Punktdaten von P und P_i zu berechnen. Dabei wird der Mittelwert der Punktabstände $\Omega_{n,i}$ mit $n \in \{1, 2, 3\}$ und $i \in \mathbb{N}$ ermittelt und in einer Liste gespeichert. Für die iterative Rotation der Punktwolke P um die z-Achse wird die folgende Rotationsmatrix verwendet:

$$R_z(\varphi_i) = \begin{bmatrix} \cos(\varphi_i) & -\sin(\varphi_i) & 0 \\ \sin(\varphi_i) & \cos(\varphi_i) & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3.41}$$

Um die korrekte Zuordnung zwischen Basis- und Zielpunktwolke zu gewährleisten, wird sowohl die HPR-Punktwolke P_i des Schadensmodells mit der Eingabepunktwolke P verglichen als auch umgekehrt. Hierbei wird die Zuordnung anhand des größeren Mittelwertes $\Omega_{n,i}$ der Punktabstände festgelegt, da dieser den ungünstigeren Fall darstellt.
Eine exakte Übereinstimmung der Punktwolken, d. h. $\Omega_{n,i} = 0$, ist selbst bei identischen Aufnahmen derselben Szene, die zu unterschiedlichen Zeitpunkten aufgenommen wurden, nicht erreichbar. Aus diesem Grund kann ein Mittelwert der Punktabstände $\Omega_{n,i}$, der kleiner oder gleich $\Omega_{\max}$ ist, als ausreichende Übereinstimmung der Modelle angesehen werden. Ein größerer Mittelwert deutet hingegen auf eine geringere Übereinstimmung der Punktwolken hin.

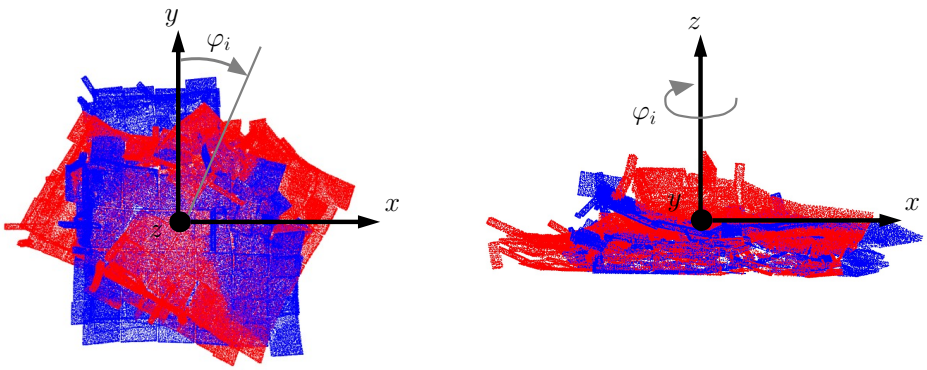


Abbildung 3.26: Schrittweise Rotation der Eingabepunktwolke P (rot) mit dem Winkel φ_i um die z-Achse zum Vergleich mit der HPR-Punktwolke P_i des Schadensmodells (blau): Draufsicht (links) und Seitenansicht (rechts).

Um den Zielwinkel φ_t, der die höchste Übereinstimmung zwischen der Basis- und Zielpunktwolke ergibt, so genau wie möglich zu bestimmen und gleichzeitig eine möglichst kurze Berechnungsdauer zu gewährleisten, wird die Iteration des Zielwinkels φ_t in drei Stufen durchgeführt:

(1) **1. Iteration:** Die Basispunktwolke wird einmal vollständig um die z-Achse mit einem Winkelschritt von $\varphi_{1,i} = 10$ rotiert und dabei der Mittelwert der Punktabstände $\Omega_{1,i}$ berechnet.

(2) **2. Iteration:** Der Winkelwert $\varphi_1(\Omega_{1,i,\min})$, der den kleinsten Mittelwert der Punktabstände aus (1) ergibt, wird in der zweiten Iteration als Startwinkel angenommen. Hierbei wird die Basispunktwolke im Bereich von $\varphi_{\Delta_1} = \varphi_1(\Omega_{1,i,\min})$ - 10 und $\varphi_{\Delta_2} = \varphi_1(\Omega_{1,i,\min}) + 10$ mit einer Winkelschrittweite von $\varphi_{2,i} = 1$ rotiert und dabei der Durchschnittswert der Punktabstände $\Omega_{2,i}$ berechnet.

(3) **3. Iteration:** Der Winkelwert $\varphi_2(\Omega_{2,i,\min})$, der den kleinsten Mittelwert der Punktabstände aus (2) ergibt, wird in der dritten Iteration als Startwinkel angenommen. Die Basispunktwolke wird im Bereich von $\varphi_{\Delta_1} = \varphi_2(\Omega_{2,i,\min})$ - 1 und $\varphi_{\Delta_2} = \varphi_2(\Omega_{2,i,\min}) + 1$ mit einer Winkelschrittweite von $\varphi_{3,i} = 0{,}1$ rotiert und dabei der Durchschnittswert der Punktabstände $\Omega_{3,i}$ berechnet. Die Iteration gibt schließlich den finalen Winkelwert des Zielwinkels φ_t und den dazugehörigen minimalen Mittelwert der Punktabstände $\Omega_{3,i,\min}$ aus, die in einer Liste gespeichert werden.

Im letzten Schritt wird der kleinste Wert in der Liste aller minimalen Durchschnittswerte der Punktabstände $\Omega_{3,i,\min}$ ermittelt. Dieser Wert wird als $\Omega_{\kappa,\min}$ bezeichnet, wobei κ den Index des entsprechenden Schadensmodells darstellt. Wenn $\Omega_{\kappa,\min} \leq \Omega_{\max}$ erfüllt ist, wird abschließend das Schadensmodell mit der Indexnummer κ ausgewählt und als Eingabe für den CIM-Algorithmus (Unterabschnitt 3.8.4) verwendet.

3.8.4 CIM-Algorithmus

Der CIM-Algorithmus ist eine vom Autor dieser Arbeit vorgeschlagene Methode, um die inneren Komponenten eines ausgewählten Schadensmodells auf der Grundlage von *Bounding Boxes* im geometrischen Bereich einer 3D-Punktwolke auszuschneiden. Die ausgeschnittenen Innenbauteile werden dem digital rekonstruierten Modell hinzugefügt, welches die sichtbaren Bauteile enthält. Für die Anwendung des Algorithmus wird vorausgesetzt, dass sich der Schwerpunkt der Eingabepunktwolke P und des Schadensmodells im Ursprung des Koordinatensystems befindet und dass die Modelle in der XY-Koordinatenebene ausgerichtet sind. Der Algorithmus 12 enthält den entsprechenden Pseudocode dieser Methode. Um sicherzustellen, dass nur die inneren Bauteile des Schadensmodells ausgeschnitten werden, muss für jedes Schadensmodell aus der Datenbank ein 3D-Modell enthalten sein, das ausschließlich die Innenbauteile enthält. Dieses Modell wird in Algorithmus 12 als beschädigtes Innenraummodell (engl. *damaged interior model*, DIM) bezeichnet.
Zu Beginn des Arbeitsablaufes werden das DIM des ausgewählten Schadensmodells M_i und die Eingabepunktwolke P eingelesen. Anschließend wird iterativ für jeden Punkt aus P eine *Bounding Box* erstellt. Die Erstellung einer *Bounding Box* erfolgt hierbei mit Hilfe von zwei Koordinaten, *minBB* und *maxBB*, die diagonal zueinanderstehende Eckpunkte des quadratischen Hüllkörpers mit den minimalen und maximalen Koordinatenwerten definieren (Abbildung 3.27, links).

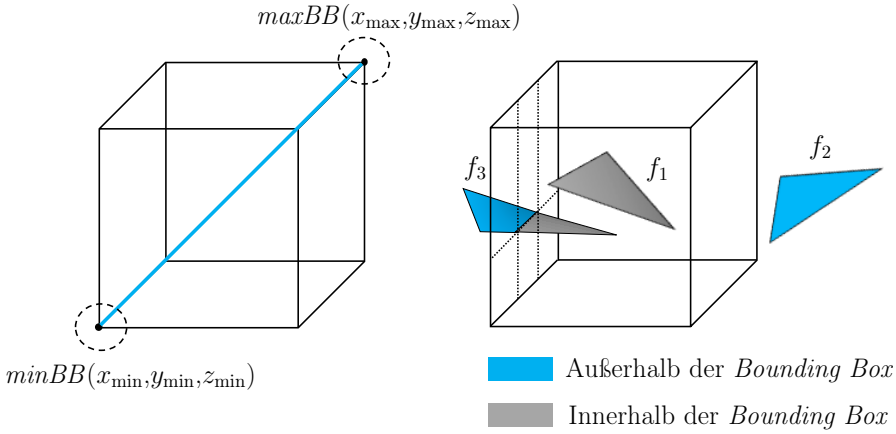


Abbildung 3.27: Bildliche Beschreibung der *Bounding-Box*-Definition (links) und des Ausschneidens von Flächen innerhalb der *Bounding Box* (rechts).

Im nächsten Schritt werden die Dreiecksflächen des DIMs, die sich innerhalb einer *Bounding Box* befinden, ausgeschnitten und zu einer leeren Netzeinheit hinzugefügt. Durch die iterative Erstellung einer *Bounding Box* an allen Punkten von P werden alle Flächen des DIMs innerhalb der geometrischen Hülle der Eingabepunktwolke P ausgeschnitten und zu einem Gesamtmodell zusammengesetzt. Dieser Vorgang wird in der Abbildung 3.27 (rechts) schematisch dargestellt. Es ist zu beachten, dass nur Flächen, die vollständig von der *Bounding Box* umschlossen werden, wie die Fläche f_1 in der Abbildung 3.27 (rechts), ausgeschnitten werden können. Flächen, die teilweise innerhalb der *Bounding Box* liegen, wie die Fläche f_3 in der Abbildung 3.27 (rechts), können nicht ausgeschnitten werden. Um alle Flächen des DIMs innerhalb der geometrischen Hülle von P möglichst vollständig zu erfassen, ist daher für den CIM-Algorithmus eine Überschneidung von *Bounding Boxes* vorgesehen.
Der Benutzer kann die Länge und Breite der *Bounding Boxes* mit Hilfe der Parameter dx und dy festlegen. Die Position und die Höhe der *Bounding Box* ergeben sich aus den Punktkoordinaten der Eingabepunktwolke. Um sicherzustellen, dass die *Bounding Boxes* in der XY-Ebene nicht über die Randpunkte der Eingabepunktwolke P hinausragen, werden für die Ermittlung der *Bounding-Box*-Koordinaten, *minBB* und *maxBB*, vier Fälle unterschieden:

(1) **Bereich $+x/+y$:** Wenn der Eingabepunkt einen positiven x- und y-Wert hat, übernimmt *maxBB* die Punktkoordinate (x, y, z) aus der Eingabe P. *minBB* ergibt sich durch Subtraktion von dx und dy mit den jeweiligen x- und y-Punktwerten, wobei der z-Wert auf Null gesetzt wird.

(2) **Bereich $-x/-y$:** Wenn der Eingabepunkt einen negativen x- und y-Wert hat, übernimmt *minBB* die Punktkoordinate (x, y, z) aus der Eingabe P. *maxBB* ergibt sich durch Addition von dx und dy mit den jeweiligen x- und y-Punktwerten, wobei der z-Wert auf Null gesetzt wird.

(3) **Bereich $+x/-y$:** Wenn der Eingabepunkt einen positiven x-Wert und einen negativen y-Wert hat, wird das gleiche Verfahren wie in Fall (2) angewendet, jedoch wird die erstellte *Bounding Box* um -dx in x-Richtung verschoben.

(4) **Bereich $-x/+y$:** Wenn der Eingabepunkt einen negativen x-Wert und einen positiven y-Wert hat, wird das gleiche Verfahren wie in Fall (2) angewendet, jedoch wird die erstellte *Bounding Box* um -dy in y-Richtung verschoben.

Die oben beschriebene Fallunterscheidung in den Bereichen (1) bis (4) ist in der Abbildung 3.28 dargestellt. Nach der Erstellung der *Bounding Box* für einen Punkt aus P werden die Flächen des DIMs, die innerhalb der *Bounding Box* liegen, mit der *crop*-Funktion von *Open3D* [186] ausgeschnitten und in einer Netzeinheit gespeichert. In den folgenden Iterationen werden alle weiteren ausgeschnittenen Flächen zu dieser Netzeinheit hinzugefügt. Da sich die erzeugten *Bounding Boxes* überschneiden können, wird das Endergebnis von doppelten Eckpunkten und Flächen bereinigt und schließlich mit dem digital rekonstruierten Modell der sichtbaren Bauteile (Abschnitt 3.7) zusammengeführt. Das Endergebnis dieses Prozesses ist ein 3D-Modell eines Gebäudes, das sowohl die äußeren als auch die inneren Komponenten enthält, wobei die Form und Anordnung der inneren Komponenten objektiv abgeschätzt werden.

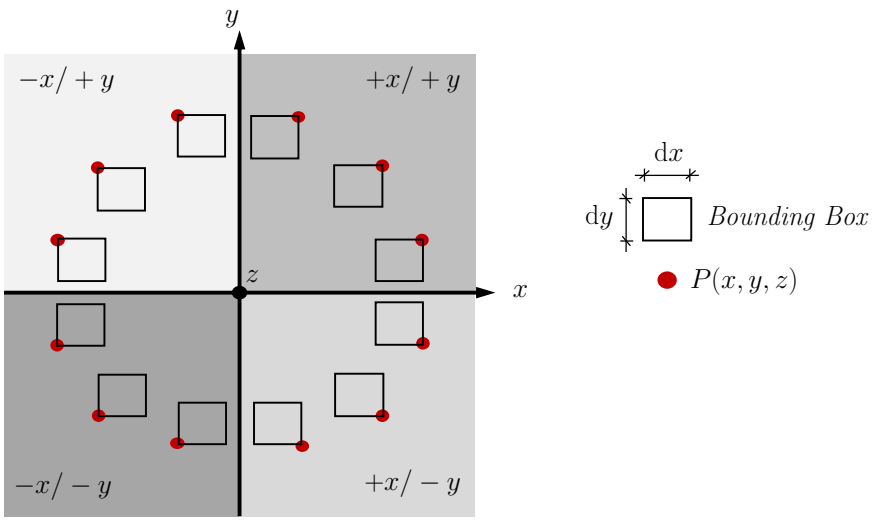


Abbildung 3.28: Positionierung der *Bounding Boxes* entlang der Randpunkte in der XY-Ebene.

3.9 BIM-Generierung

Die Integration der digital rekonstruierten Bauteile in den BIM-Prozess ist von entscheidender Bedeutung für die Erzeugung eines konsistenten Bauwerksdatenmodells, das bei der Planung, Koordination und Durchführung von Such- und Rettungseinsätzen effizient genutzt werden kann. In diesem Zusammenhang dient das BIM-Modell als Informationsquelle und digitales Verwaltungsinstrument, das die Überwachung des Zustandes und der Eigenschaften von Bauteilen ermöglicht und eine objektive Entscheidungsfindung bei Rettungseinsätzen erlaubt. In diesem Abschnitt wird der letzte Schritt der in Abbildung 3.1 dargestellten Strategie der algorithmischen Datenverarbeitung erörtert, bei dem die digital rekonstruierten Körper der Gebäudekomponenten in den BIM-Prozess überführt werden. Hierbei erfolgt die Umwandlung der *Mesh*-Körper in das IFC-Format [29], das als Standard für den Datenaustausch in der BIM-Methode gilt. Die in den BC-Objekten gespeicherten Daten werden genutzt, um die erzeugten IFC-Objekte mit semantischen und alphanumerischen Informationen anzureichern und somit eine präzise Darstellung der Gebäudekomponenten im BIM-Modell zu gewährleisten.

3.9.1 Mesh-to-IFC

Die vorgeschlagene Methode zur automatischen Konvertierung von digital rekonstruierten Bauteilen im OBJ-Format [121] in das IFC-Format [29] ist in Algorithmus 13 als Pseudocode zusammengefasst und wird im Folgenden erläuert:
Da es sich bei den digital rekonstruierten Bauteilkörpern um Polygonnetze handelt, die aus Dreiecksflächen bestehen, wird die vom Autor dieser Arbeit entwickelte Methode der IFC-basierten Datentransformation als *Mesh-to-IFC* bezeichnet. Die *Mesh-to-IFC*-Methode beginnt mit der Erstellung einer standardisierten IFC-Projektstruktur für das BIM-Modell unter Verwendung der *IfcOpenShell*-Bibliothek [87]. Diese Projektstruktur umfasst das Wurzelelement *IfcProject* mit seinen verwandten Elementen *IfcSite*, *IfcBuilding* und *IfcBuildingStorey*, die nur einmal im Projekt verwendet werden. Die genannten IFC-Elemente ermöglichen die einheitliche Verwaltung von IFC-Daten, erleichtern den Informationsaustausch zwischen verschiedenen Softwareanwendungen und unterstützen die geschossweise Strukturierung von Bauteilen des Gebäudemodells. Im nächsten Schritt werden die gespeicherten OBJ-Dateien der digital rekonstruierten Bauteile sowie die Objekteigenschaften *type*, *material* und *thickness* aus den dazugehörigen BC-Objekten

eingelesen. Auf der Grundlage der Objektklasse, *type*, des BC-Objektes wird die IFC-Klasse, *IfcClass*, für das rekonstruierte Bauteil definiert. In der Tabelle 3.4 sind die entsprechenden IFC-Klassen in Abhängigkeit von der Objektklasse aus den BC-Objekten aufgeführt.

Tabelle 3.4: Definition von IFC-Klassen in Abhängigkeit von der Objektklasse aus den BC-Objekten.

#	**Objektklasse**	**type**	***IfcClass***
1	Decke	*Slab*	*IfcSlab*
2	Wand	*Wall*	*IfcWall*
3	Decke\Wand	*Slab | Wall*	*IfcBuildingElement*
4	Träger	*Beam*	*IfcBeam*
5	Stütze	*Column*	*IfcColumn*
6	Träger\Stütze	*Beam | Column*	*IfcBuildingElement*
7	Bodenplatte	*Base*	*IfcFooting*
8	nicht klassifiziert	*Unclassified*	*IfcBuildingElement*

Für die mehrdeutigen Objektklassen *Slab | Wall* und *Beam | Column* wird die Entität *IfcBuilding-Element* zugewiesen, wobei die Zeichenfolge in *type* als Beschreibung in der Entität verwendet wird, um die Mehrdeutigkeit der Objektklasse zu berücksichtigen. Dasselbe gilt analog für Punktmengen, denen im Zuge der Objektklassifizierung die Klasse *Unclassified* zugewiesen wurde.
Nach der Definition der IFC-Klasse in Abhängigkeit von der Objektklasse des BC-Objektes erfolgt die Umwandlung der OBJ-Daten in IFC. Zu diesem Zweck wird die implementierte *OBJ2IFC*-Methode von *Dion Moult* [119] verwendet, die auf den Python-Bibliotheken *PyMeshLab* [120] und *IfcOpenShell* [87] basiert. Der allgemeine Prozess der *OBJ2IFC*-Methode ist in der Abbildung 3.29 schematisch dargestellt und wird im Folgenden erläutert:
Zunächst wird mit Hilfe von *PyMeshLab* [120] eine *MeshSet*-Instanz M_i aus der OBJ-Datei erstellt, die die geometrischen Daten des rekonstruierten Bauteils enthält. Anschließend werden aus M_i die Eckpunkte (engl. *vertices*) und Flächen (engl. *faces*) extrahiert und jeweils im Datentyp *array* ausgegeben (Abbildung 3.29 (1)). Die Eckpunkte der Fläche $F_i(n, 3)$ mit $i \in \mathbb{N}$ werden als *IfcCartesianPoint* dargestellt (Abbildung 3.29 (2)), d. h. als Punkte in einem dreidimensionalen kartesischen Koordinatensystem. Diese Punkte werden zur Definition des Polygons *IfcPolyLoop* (Abbildung 3.29 (3)) verwendet, das wiederum die äußere Begrenzung der IFC-Fläche *IfcFace* (Abbildung 3.29 (5)) bildet, die mittels *IfcFaceOuterBound* (Abbildung 3.29 (4)) festgelegt wird. Das erzeugte *IfcFace*-Objekt von $F_i(n, 3)$ wird dann in der Liste IfcFaceList gespeichert (Abbildung 3.29 (6)). Die in Abbildung 3.29 dargestellten Schritte (1) bis (6) werden innerhalb einer Schleife für alle Flächen $F_i(n, 3)$ aus der OBJ-Datei ausgeführt. Anschließend wird aus den IFC-Flächen in IfcFaceList ein *IfcClosedShell*-Objekt generiert (Abbildung 3.29 (7)), welches eine geschlossene Hülle zur Begrenzung des Körpers in $\mathbb{R}^3$ repräsentiert. Da es sich bei den OBJ-Daten um begrenzte Polygone mit ebenen Flächen handelt, wird das *Boundary Representation* (B-rep) als geeignete Darstellungsform gewählt. Zu diesem Zweck wird aus dem *IfcClosedShell* ein *IfcFacetedBrep* erzeugt (Abbildung 3.29 (8)), das die IFC-Flächen in Form von facettierten B-reps darstellt. Gemäß der Definition des Standards ISO/CD 10303-42:1992 [90] handelt es sich bei einem facettierten B-rep um eine vereinfachte Form des B-reps, bei der alle Flächen planar und alle Kanten geradlinig sind. Im Vergleich zu einem herkömmlichen B-rep-Modell werden Kanten und Eckpunkte im Objekt *IfcFacetedBrep* nicht explizit dargestellt, sondern implizit durch die Verwendung der Entität *IfcPolyLoop*. Die Definition von *IfcFacetedBrep* aus *IfcClosedShell* erfolgt im Zusammenhang mit einem geometrischen Darstellungskontext und der Art der Darstellung, die in *IfcShapeRepresentation* wiedergegeben werden (Abbildung 3.29 (9)). Der Darstellungskontext bezieht sich dabei auf ein leeres IFC-Modell gemäß dem ICF4-Standard [29], wobei der

Darstellungstyp für Volumenelemente (engl. *body*) spezifiziert ist. Das resultierende *IfcProductDefinitionShape*-Objekt (Abbildung 3.29 (10)) stellt die geometrischen Daten der OBJ-Datei im IFC-Format dar und kann nun als Produktdefinition in IFC-Entitäten verwendet werden.

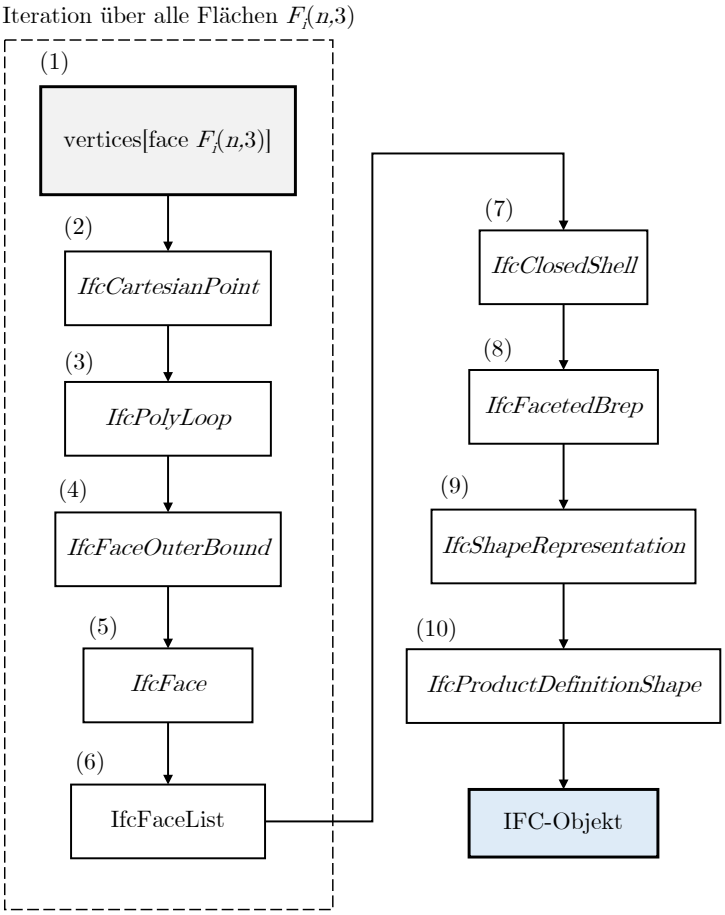


Abbildung 3.29: Schematischer Ablauf der *OBJ2IFC*-Methode [119] unter Verwendung der *IfcOpenShell*-Bibliothek [87]. Die Schritte (1) bis (6) innerhalb der gestrichelten Box werden für alle Flächen $F_i(n, 3)$ aus der OBJ-Datei wiederholt.

3.9.2 Informationsanreicherung

Nachdem die OBJ-Datei im IFC-Format umgewandelt wurde, wird das IFC-Objekt verwendet, um eine IFC-Entität zu erstellen und diese mit semantischen Informationen anzureichern. Die Abbildung 3.30 zeigt die Datenstruktur der semantischen Informationen einer IFC-Entität, die im Rahmen dieser Arbeit erstellt wurde. Die Informationsanreicherung beginnt mit der Zuweisung der entsprechenden IFC-Klasse, die, wie bereits in Unterabschnitt 3.9.1 erläutert, von der Objektklasse des zugehörigen BC-Objektes abhängt. In Tabelle 3.4 sind die IFC-Klassen in Abhängigkeit von der Objektklasse der BC-Objekte aufgeführt. Zusätzlich zur IFC-Klasse erhält die IFC-Entität die von buildingSMART [29] definierten Standardattribute gemäß IFC4 zur eindeutigen Identifizierung der Entität. Diese Attribute umfassen eine automatisch generierte GUID, einen Namen, der aus dem Namen der OBJ-Datei übernommen wird, eine optional verwendete Beschreibung (nur bei den Objektklassen *Slab* | *Wall* oder *Beam* | *Column*) sowie eine Kennzeichnung im Format „type:thickness:ObjectTypeID“, wie z. B. „Wall:20cm:03“. Zur Be-

schreibung der Eigenschaften der IFC-Entität wurden vom Autor dieser Arbeit benutzerdefinierte IFC-Eigenschaftssätze mit der Bezeichnung *IfcPropertySet* definiert, die in der Abbildung 3.30 dargestellt sind.

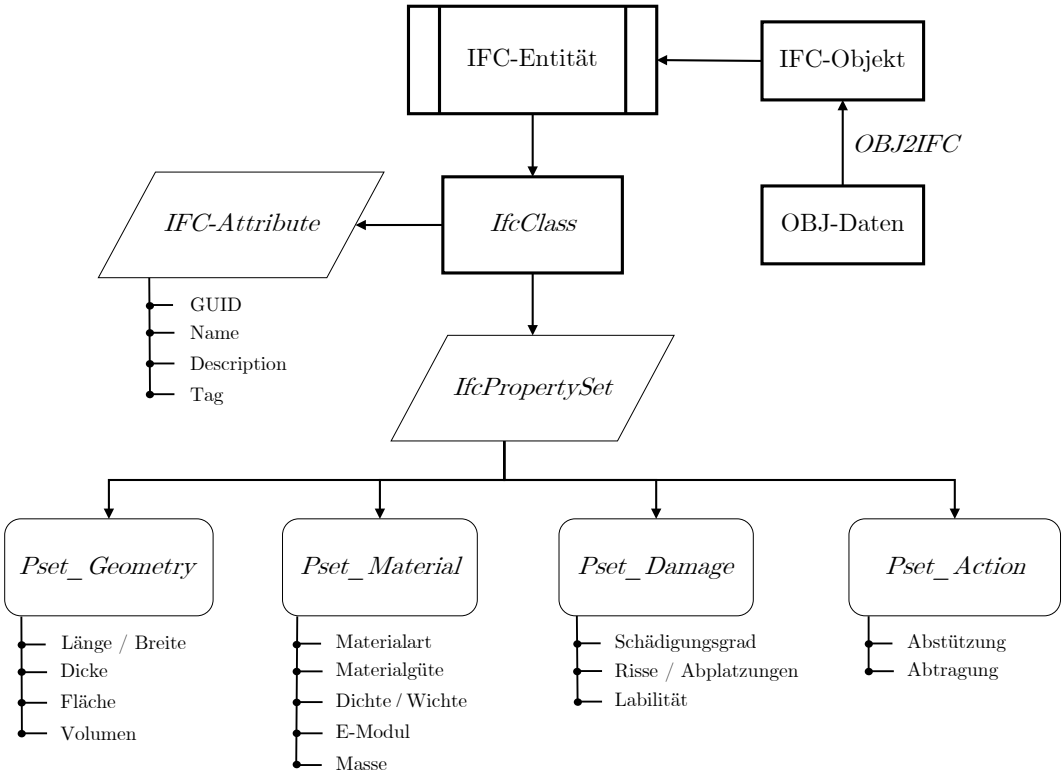


Abbildung 3.30: Datenstruktur der aus OBJ-Daten erstellten IFC-Objekte.

Die Eigenschaftssätze der IFC-Entität enthalten Informationen zur Geometrie, Material, Schäden und Maßnahmen. Die geometrischen Daten wie Fläche, Volumen oder Dicke des Bauteils werden automatisch berechnet und können im *IfcPropertySet Pset_ Geometry* für die entsprechende IFC-Entität abgerufen werden. Das Material der Komponente ergibt sich aus der vorhergesagten Materialklasse im BC-Objekt. Diese beeinflusst den Wert der anderen Materialparameter im *IfcPropertySet Pset_ Material* (Abbildung 3.30) wie folgt:

(1) Wenn das Material als *Concrete* identifiziert wird, werden für die Dichte 2.400 kg/m^3 und für den E-Modul 30.000 MPa angenommen. Diese Materialkennwerte entsprechen Durchschnittswerten für Normalbeton [12]. Die Masse wird dann durch Multiplikation von Volumen, Dichte und einem Umrechnungsfaktor von 10^{-3} berechnet, um die Einheit von Kilogramm auf Tonnen umzuwandeln.

(2) Wenn das Material als *Masonry* identifiziert wird, ist das Verfahren analog zu (1), wobei für die Dichte 1.950 kg/m^3 und für den E-Modul 8.000 MPa als Durchschnittswerte angenommen werden [12].

(3) Für ein BC-Objekt mit der Materialklasse *Unclassified* wird die Dichte und der E-Modul der IFC-Entität auf Null gesetzt, was bedeutet, dass keine Berechnungen durchgeführt werden können, da das Material nicht klassifiziert ist.

Im Gegensatz zu den Eigenschaftssätzen *Pset_ Geometry* und *Pset_ Material*, bei denen die Daten ihrer Elemente automatisch ermittelt werden, müssen die Werte der Elemente in den Ei-

genschaftssätzen *Pset_ Damage* und *Pset_ Action* manuell vom Benutzer festgelegt werden. Dies ist darauf zurückzuführen, dass im Rahmen dieser Arbeit keine digitalen Methoden zur automatischen Auswertung der Schadensanalyse berücksichtigt werden und somit keine objektive Bewertung hinsichtlich der Maßnahme, sei es Abstützung oder Abtragung von Bauteilen, vorgenommen werden kann.
Die IFC-basierte Datentransformation (Unterabschnitt 3.9.1) und die Informationsanreicherung der erzeugten IFC-Entität werden für alle OBJ-Dateien der digital rekonstruierten Bauteile in einer Schleife durchgeführt und das resultierende Gesamtmodell im IFC-Format gespeichert. Dieses Format kann von verschiedenen Softwareprogrammen geöffnet werden, die IFC unterstützen, wie z. B. *Blender* [23], *BIMvision* [1] oder *Revit Autodesk* [14].

3.10 Zusammenfassung

In diesem Kapitel wurde die vom Autor dieser Arbeit entwickelte Methodik zur modellunabhängigen Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmentragwerken umfassend vorgestellt. Wie in Abschnitt 3.1 beschrieben, besteht die vorgeschlagene Methodik aus sechs aufeinanderfolgenden Arbeitsschritten, die jeweils spezifische Methoden aus der algorithmischen Datenverarbeitung beinhalten:

1. Vorverarbeitung (Abschnitt 3.3),
2. Segmentierung (Abschnitt 3.4),
3. Objektklassifizierung (Abschnitt 3.5),
4. Materialklassifizierung (Abschnitt 3.6),
5. digitale 3D-Rekonstruktion (Abschnitt 3.7 und Abschnitt 3.8) und
6. BIM-Generierung (Abschnitt 3.9).

Die Implementierung der Arbeitsschritte erfolgte mit der objektorientierten Programmiersprache Python. Im Abschnitt A.1 sind die implementierten Verfahren in Form von Pseudocodes zusammengefasst. Um im Zuge der algorithmischen Datenverarbeitung der 3D-Punktwolke effizient auf die generierten Daten, die unterschiedliche Datentypen aufweisen, zugreifen und diese verwalten zu können, wurde die Klasse BC objektorientiert in Python implementiert (Abschnitt 3.2). Die in der BC-Klasse enthaltenen Daten aus den oben genannten Arbeitsschritten bilden dabei die Grundlage für die automatische Generierung des BIM-Modells und die semantische Informationsanreicherung. Im ersten Schritt der vorgeschlagenen Methodik werden die unstrukturierten Punktdaten der 3D-Punktwolke vorverarbeitet, um einerseits die Datenqualität zu verbessern und andererseits die Effizienz der Datenanalyse zu erhöhen (Abschnitt 3.3). Im Rahmen der Vorverarbeitung werden die Punktdaten anhand von GPS-Koordinaten referenziert, nicht benötigte Bereiche aus der Punktwolke ausgeschnitten, ein *Downsampling* durchgeführt und Ausreißerpunkte entfernt. Hierfür wird die Verwendung der benutzerfreundlichen Software *CloudCompare* [40] vorgeschlagen, die effiziente vordefinierte Funktionen für diese Aufgaben bereitstellt.
Für die Segmentierung der Punktwolke (Abschnitt 3.4) werden zwei Szenarien betrachtet: (1) die Segmentierung von ebenen Punktmengen in $\mathbb{R}^3$ und (2) die Segmentierung von linienförmigen Punktmengen in $\mathbb{R}^2$. Ziel ist es, die flächenförmigen Bauteile wie Decken und Wände sowie die stabförmigen Bauteile wie Träger und Stützen aus der 3D-Punktwolke der seismisch beschädigten Stahlbetonrahmenkonstruktion basierend auf ihrer Form und Orientierung zuverlässig zu segmentieren. Für die automatische Extraktion von ebenen Punktmengen in $\mathbb{R}^3$ wird der RANSAC-Algorithmus von *Schnabel et al.* [149] vorgeschlagen, der im Python-Modul *cloudComPy* [40] implementiert ist. Wenn die ebenen Punktmengen parallele Komponenten mit ähnlicher Form und Größe aufweisen, werden diese Segmente miteinander vereinigt, um die Dicke

der Komponente abzuleiten. Für die Segmentierung von Wänden in einem Stahlbetonrahmen wird ein iterativer Ansatz basierend auf der α-*Shape*-Methode [54, 179] vorgeschlagen. Dabei werden Randpunkte von Trägern und Stützen mittels der α-*Shape*-Methode [54, 179] in einem Rahmensegment iterativ identifiziert und entfernt, um eine präzise Segmentierung der Punktdaten von Wänden zwischen Trägern und Stützen zu ermöglichen. Die Methode zur Segmentierung von linienförmigen Punktdaten aus Rahmensegmenten, d. h. Träger und Stützen, basiert auf dem RANSAC-Algorithmus [6] mit einem Linienmodell in $\mathbb{R}^2$. Hierbei wird nach zusammenhängenden Punktmengen gesucht, die sich entlang einer Linie innerhalb eines benutzerdefinierten Intervalles befinden, wobei die Größe des Intervalles in Abhängigkeit von der Höhe der Träger und der Breite der Stützen definiert wird. Durch die Anwendung der *ExtractConnectedComponents*-Methode von *cloudComPy* [40] werden aus den identifizierten RANSAC-Liniensegmenten zusammenhängende Träger- und Stützensegmente generiert.
Im nächsten Schritt des Arbeitsablaufes werden die segmentierten Punktdaten anhand ihrer Form, Abmessungen und räumlichen Orientierung einer strukturellen oder nicht-strukturellen Objektklasse zugeordnet (Abschnitt 3.5). Die strukturellen Objektklassen umfassen dabei die Komponenten einer Stahlbetonrahmenkonstruktion (*Slab*, *Wall*, *Beam*, *Column*, *Base*), während die nicht-strukturellen Klassen den Boden (*Ground*) oder nicht-klassifizierbare (*Unclassified*) Punktsegmente berücksichtigen. Die Abmessungen der Punktsegmente werden anhand ihrer minimalen *Bounding Box* bestimmt. Um die geometrischen Eigenschaften der Bauteile zu berücksichtigen, sind für die strukturellen Objektklassen Mindest- und Maximalabmessungen definiert, die eingehalten werden müssen. Sofern die Orientierung dem intakten Zustand entspricht, können Träger und Stützen eindeutig anhand ihrer Abmessungen voneinander unterschieden werden. Wenn dies nicht der Fall ist, wird das betreffende Segment der mehrdeutigen Objektklasse *Beam* | *Column* zugeordnet, da es sich um einen Träger oder eine Stütze handeln kann. Decken und Wände werden auf Basis ihrer räumlichen Orientierung im intakten Zustand klassifiziert, die durch den Normalenvektor ihrer Ebene definiert wird. Falls die ebene Punktmenge auf der Grundlage ihrer Orientierung nicht eindeutig der Objektklasse *Slab* oder *Wall* zugeordnet werden kann, wird die mehrdeutige Objektklasse *Slab* | *Wall* vergeben. Zur Unterscheidung der Objektklassen *Ground* und *Base* werden die Orientierung, die Fläche und die minimale z-Koordinate der ebenen Punktmenge berücksichtigt.
In der vorgeschlagenen Methodik wird neben der Objektklassifizierung auch die automatische Materialklassifizierung der BC-Objekte berücksichtigt (Abschnitt 3.6). Zu diesem Zweck wurde ein DNN mittels DL trainiert, das in der Lage ist, eine passende Materialklasse auf der Grundlage eines RGB-Wertes vorherzusagen. Dabei wurden für den Trainingsdatensatz 2000 RGB-Daten von realen Oberflächen von Beton und Mauerwerk berücksichtigt. Das trainierte DL-Modell erreichte auf dem Testdatensatz eine durchschnittliche Genauigkeit von 98% und kann somit die Materialklassen *Concrete* und *Masonry* unter Berücksichtigung ihrer dominanten RGB-Farbe mit hoher Präzision unterscheiden. Um die Materialklasse eines Punktsegments anhand des trainierten DL-Modells vorherzusagen, wird der am häufigsten auftretende RGB-Wert aus den Punktdaten verwendet, der mit Hilfe des k-*means*-Algorithmus [132] bestimmt wird.
Nach der Klassifizierung der Gebäudekomponenten wird die digitale 3D-Rekonstruktion für die sichtbaren Bauteile durchgeführt (Abschnitt 3.7), d. h. die Bauteile, die in der erfassten 3D-Punktwolke dargestellt sind. Hierbei wird die α-*Shape*-Methode [54, 179] zur Oberflächenrekonstruktion verwendet, mit der sowohl konvexe als auch konkave Formen mit Öffnungen erzeugt werden können. Die rekonstruierten Oberflächen werden durch Anwendung einer Oberflächenextrusion zu einem geschlossenen Körper erweitert ($\mathbb{R}^2 \rightarrow \mathbb{R}^3$), wobei die Komponentendicke der BC-Objekte als Wert für die Extrusion der Oberflächen entlang ihrer Normalen verwendet wird. Zur digitalen Rekonstruktion von Bauteilen im verdeckten Innenbereich eines Gebäudes, wurde vom Autor dieser Arbeit eine objektive, modellunabhängige Methode für weitere For-

schungsbemühungen vorgestellt (Abschnitt 3.8). Bei dieser Methode werden bei ausreichender geometrischer Übereinstimmung zwischen der Eingabepunktwolke des rekonstruierten Gebäudes (Basismodell) und einer synthetischen Punktwolke eines Schadensmodells (Zielmodell) Strukturinformationen aus dem Schadensmodell extrahiert und dem digital rekonstruierten Gebäudemodell hinzugefügt. Um die Wahrscheinlichkeit einer hohen geometrischen Überstimmung mit dem tatsächlichen Schadensbild einer seismisch beschädigten Stahlbetonrahmenkonstruktion zu erhöhen, ist die Verwendung von $n > 1$ Schadensmodellen mit unterschiedlichen Schadensbildern in einer Datenbank vorgesehen. Zur schnellen und einfachen Generierung von Schadensmodellen seismisch beschädigter Stahlbetonrahmenkonstruktionen wird vom Autor dieser Arbeit die numerische Kollapssimulation unter Verwendung der auf der *Bullet*-Physik-Engine basierenden Starrkörperdynamik in *Blender* [23] und dem BCB [128] vorgeschlagen.
Im letzten Schritt der entwickelten Methodik werden die digital rekonstruierten Bauteile des Gebäudes vom OBJ-Format in das IFC-Format konvertiert und auf diese Weise in den BIM-Prozess überführt (Abschnitt 3.9). Die dabei verwendete *OBJ2IFC*-Methode [119] beinhaltet die Extraktion von Eckpunkten und Flächen aus den OBJ-Daten und darauf aufbauend die Erzeugung von *IfcFacetedBrep*-Objekten, aus denen schließlich ein IFC-Objekt generiert wird. Nach der Konvertierung der OBJ-Daten in IFC wird aus dem Objekt eine IFC-Entität mit semantischen und alphanumerischen Informationen erstellt. Die Informationsanreicherung beginnt mit der Zuweisung der entsprechenden IFC-Klasse auf der Grundlage der Objektklasse des zugehörigen BC-Objektes. Der IFC-Entität wird außerdem eine GUID, ein Name und eine eindeutige Kennzeichnung gemäß der Definition von buildingSMART [29] zugewiesen. Zur Beschreibung der Eigenschaften einer IFC-Entität wurden vom Autor dieser Arbeit benutzerdefinierte IFC-Eigenschaftssätze definiert, die objektorientierte Informationen über Geometrie, Material, Schäden und Maßnahmen berücksichtigen. Die Geometrie- und Materialdaten werden automatisch anhand der in den BC-Objekten enthaltenen Informationen festgelegt. Im Gegensatz dazu müssen die Angaben für die IFC-Eigenschaftssätze, die Schäden und Maßnahmen betreffen, manuell durch den Benutzer definiert werden. Der beschriebene Prozess der Umwandlung von OBJ-Daten in IFC und der Informationsanreicherung wird auf alle Gebäudekomponenten im Datensatz angewendet. Das Endergebnis der vorgeschlagenen Methodik ist der digitale Zwilling des erfassten Gebäudes im standardisierten IFC-Format.

Kapitel 4

Evaluierung und Diskussion

Dieses Kapitel widmet sich der Evaluierung und Diskussion der in Kapitel 3 vorgestellten Methodik zur modellunabhängigen Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen. Ziel dieser Evaluierung ist es, die Leistungsfähigkeit und Genauigkeit der einzelnen Arbeitsschritte der entwickelten Methodik zu bewerten und ihre Anwendungsgrenzen zu bestimmen. Zunächst werden in Abschnitt 4.1 die für die Bewertung verwendeten Daten vorgestellt und vorverarbeitet. Anschließend werden die einzelnen Arbeitsschritte der entwickelten Methodik separat evaluiert. Dies umfasst die Evaluierung der Segmentierung in Abschnitt 4.2, der Objektklassifizierung in Abschnitt 4.3, der Materialklassifizierung in Abschnitt 4.4, der 3D-Rekonstruktion der sichtbaren Bauteile in Abschnitt 4.5 und der BIM-Generierung in Abschnitt 4.6. Die erzielten Ergebnisse werden ausführlich diskutiert, um einerseits die Leistungsfähigkeit der entwickelten Methodik hervorzuheben und andererseits ihre Anwendungsgrenzen und möglichen Herausforderungen zu beleuchten. Abschließend werden in Abschnitt 4.7 die durchgeführte Evaluierung und die daraus gewonnenen Erkenntnisse zusammengefasst.

4.1 Datengenerierung und -vorverarbeitung

Für eine aussagekräftige Evaluierung der in Kapitel 3 vorgestellten Methodik werden 3D-Punktwolken von seismisch beschädigten Stahlbetonrahmenkonstruktionen benötigt. Wie bereits in Abschnitt 2.1 erwähnt, können solche Daten mit Hilfe von aktiven oder passiven Fernerkundungsmethoden erhoben werden. Beispiele hierfür sind das Laserscanning oder die Photogrammetrie. In diesem Zusammenhang erfordert die Datenerfassung eine spezielle technische Ausrüstung und Zugang zu den betroffenen Gebäuden in den Katastrophengebieten. Im Rahmen dieser Arbeit war es aufgrund von Sicherheits- und Haftungsbedenken nicht möglich, geeignete Daten von seismisch beschädigten Stahlbetonrahmentragwerken zu erfassen. Darüber hinaus stehen derzeit keine vorhandenen Punktwolkendatensätze von solchen Gebäuden frei zur Verfügung. Um dennoch eine fundierte Evaluierung der entwickelten Methodik vornehmen zu können, werden in diesem Abschnitt die Generierung und Vorverarbeitung synthetischer Daten von seismisch beschädigten Stahlbetonrahmenkonstruktionen behandelt. Die Generierung von synthetischen Daten ermöglicht es, gezielt verschiedene Schadensbilder von seismisch beschädigten Stahlbetonrahmenkonstruktionen zu simulieren und darauf basierend die Leistungsfähigkeit der entwickelten Methodik umfassend zu bewerten. Die Erstellung dieser Schadensbilder erfolgt anhand digitaler 3D-Modelle, die auf realen Fallbeispielen basieren. Mit Hilfe der Photogrammetrie werden auf der Grundlage dieser Modelle 3D-Punktwolken erzeugt. Als zusätzliches Fallbeispiel wird die Verarbeitung und Auswertung eines photogrammetrischen Datensatzes einer nachgestellten Szene eines teilweise eingestürzten Stahlbetongebäudes in Lyon vorgestellt. Die Einbindung dieses

realen Datensatzes, der dem Autor dieser Arbeit zur Verfügung gestellt wurde, ermöglicht eine praxisnahe Bewertung der Leistungsfähigkeit der entwickelten Methodik. Um die Leistungsfähigkeit und Genauigkeit der verwendeten Photogrammetrie-Software anhand von realen Daten zu bewerten, beginnt dieser Abschnitt mit der Vorstellung des Fallbeispiels aus Lyon. Die daraus gewonnenen Erkenntnisse werden als Grundlage für die photogrammetrische Rekonstruktion der synthetischen Datensätze genutzt.

4.1.1 Fallbeispiel Lyon

Neben den synthetisch erzeugten Punktwolkendatensätzen wird im Rahmen dieser Arbeit eine nachgestellte Katastrophenszene eines teilweise eingestürzten Stahlbetongebäudes in Lyon, Frankreich, photogrammetrisch rekonstruiert. Die entsprechenden Bilddaten wurden dem Autor dieser Arbeit von den Urheberrechtsinhabern [97, 124] zu Forschungszwecken zur Verfügung gestellt. Bei dem teilweise eingestürzten Gebäude (Abbildung 4.1) handelt es sich um eine dreigeschossige Stahlbetonrahmenkonstruktion mit einem rechteckigen Grundriss. Die Gebäudeabmessungen entsprechen etwa 15 x 28 m in Länge und Breite und etwa 10 m in der Höhe. Die Stahlbetonstützen sind rechteckig geformt. An der Eckseite des Gebäudes befindet sich ein Treppenhaus mit Stahlbetonwänden, das sich vom Erdgeschoss bis zum Dachgeschoss erstreckt. Des Weiteren ist das Tragwerk mit einem Nachbargebäude verbunden, das über eine Übergangsbrücke im ersten und zweiten Geschoss auf der Rückseite erreichbar ist. Es ist wichtig zu beachten, dass keine Mauerwerksausfachungen zwischen den Außenstützen vorhanden sind und für die nachgestellte Szene als vollständig zerstört betrachtet werden können. Wie in der Abbildung 4.1 ersichtlich ist, sind die Deckenplatten des ersten und zweiten Obergeschosses mit einer Fläche von etwa 15 x 10 m teilweise auf die Decke des Erdgeschosses eingestürzt und dabei erheblich beschädigt worden. In [134] kann der gezielt herbeigeführte Teileinsturz in Form eines Videos beobachtet werden.

Abbildung 4.1: Drohnenaufnahmen eines teilweise eingestürzten Stahlbetongebäudes in Lyon, Frankreich [97, 124].

Das Stahlbetongebäude befindet sich auf einem Abbruchgelände in Lyon und war Gegenstand der von der Europäischen Union finanzierten Forschungsprojekte INACHUS [89] und RECONASS [141]. Mit insgesamt 312 Drohnenaufnahmen, die sich aus 200 Nadir- und 112 Schrägbildaufnahmen zusammensetzen, wurde die nachgestellte Katastrophenszene von *Kerle et al.* [97] und *Nex et al.* [124] zu photogrammetrischen Zwecken umfangreich erfasst. Die aufgenommene Fläche erstreckt sich über etwa 18.000 m^2. Die relevanten Daten des photogrammetrischen Datensatzes sind in der Tabelle 4.1 aufgelistet.
Die digitale Rekonstruktion von Katastrophenszenen zur Unterstützung von Such- und Ret-

tungsmaßnahmen stellt einen spezifischen Anwendungsbereich dar, die eine umfassende Untersuchung im Zusammenhang mit der Photogrammetrie erfordert. Aus diesem Grund liegt der Fokus der vorliegenden Untersuchung auf der Bewertung der Leistungsfähigkeit ausgewählter Photogrammetrie-Software. Dabei werden insbesondere die Möglichkeiten und Grenzen der Software im Hinblick auf Genauigkeit und Rechenzeit analysiert. Zur Erstellung einer dichten 3D-Punktwolke aus dem gegebenen Bilddatensatz werden zwei Programme verwendet: die kommerzielle Software *Agisoft Metashape* [10] und das *Open-Source*-Programm *VisualSfM* [175].

Tabelle 4.1: Versuchsdetails zur photogrammetrischen Aufnahme des teilweise eingestürzten Stahlbetongebäudes in Lyon, Frankreich [97, 124].

Teilweise eingestürztes Stahlbetongebäude	
Ort	Lyon, Frankreich
Aufnahmetechnik	Nadir- und Schrägbildaufnahme
Kamerasystem	DJI FC330
Bildauflösung	4000x3000 = 12 Megapixel
Brennweite	4 mm
GPS-Daten	Ja
Wetter	Sonnig
# der Bilder	312
Bildüberlappung	$>$ 60% (seitlich) $>$ 80% (vorwärts)
Aufnahmefläche	$\approx$ 18.000 m^2

Im Rahmen dieser Untersuchung werden die Punktwolken, die mit *VisualSfM* [175] erzeugt wurden, mit den Ergebnissen von *Agisoft Metashape* [10] verglichen. Es ist wichtig zu beachten, dass *Agisoft Metashape* [10] auf einer leistungsstarken *Graphic Workstation* über eine Remote-Verbindung ausgeführt wurde, während *VisualSfM* [175] auf einem Standardcomputer getestet wurde. Der Grund dafür ist, dass der in *VisualSfM* [175] implementierte SIFT-Algorithmus [108] keinen Remote-Zugriff unterstützt. Da die Programme auf unterschiedlichen Computern verwendet wurden, wird die Berechnungszeit für den Vergleich nicht berücksichtigt. Die technischen Spezifikationen der verwendeten Computer sind in der Tabelle 4.2 zusammengefasst.

Tabelle 4.2: Systemspezifikationen der *Graphic Workstation* und des Standardcomputers.

Graphic Workstation UniBw M		**Standardcomputer**
CPU	Intel® Xeon® CPU E5-2680 v4 @ 2.40GHz	Intel® Core™ i7-8565U CPU @ 1.80GHz
RAM	128 GB	32 GB
GPU	Nvidia Quadro M5000 with 8 GB GDDR5 Speicher	Intel® UHD Graphics 620

Die gesamte Berechnungszeit bis zur Fertigstellung der Punktwolke betrug in *Agisoft Metashape* [10] bei mittlerer Qualitätseinstellung für die Ausrichtung der Fotos und die dichte Rekonstruktion 75 Minuten. In *VisualSfM* [175] dauerte der photogrammetrische Arbeitsablauf für alle 312 Bilder insgesamt 115 Minuten. Die Ermittlung der Verknüpfungspunkte nahm dabei 21 Minuten in Anspruch, während für die Schätzung der Kameraposition und die Bündelblockausgleichung 2,7 Minuten benötigt wurden. Schließlich dauerte die Erstellung der dichten Punktwolke unter Verwendung des integrierten CMVS-Algorithmus [68] 91 Minuten. Die mit *Agisoft Metashape* [10] und *VisualSfM* [175] erzeugten Punktwolken sind in der Abbildung 4.2 dargestellt. Des Weiteren zeigt die Abbildung 4.3 die Anzahl der erzeugten Punktdaten und die dafür benötigte Berechnungszeit in Form eines Balkendiagramms.
Da der verwendete Datensatz viele Schrägbildaufnahmen enthält, die weite Bereiche des Standortes darstellen, enthält die generierte Punktwolke von *VisualSfM* [175] eine erhebliche Anzahl

von Ausreißern. Im Gegensatz dazu bietet *Agisoft Metashape* [10] die Möglichkeit, Berechnungsgrenzen festzulegen und die Größe der Berechnungsbox vorab einzustellen, was zu einer schnelleren Verarbeitung der Bilddaten führt.

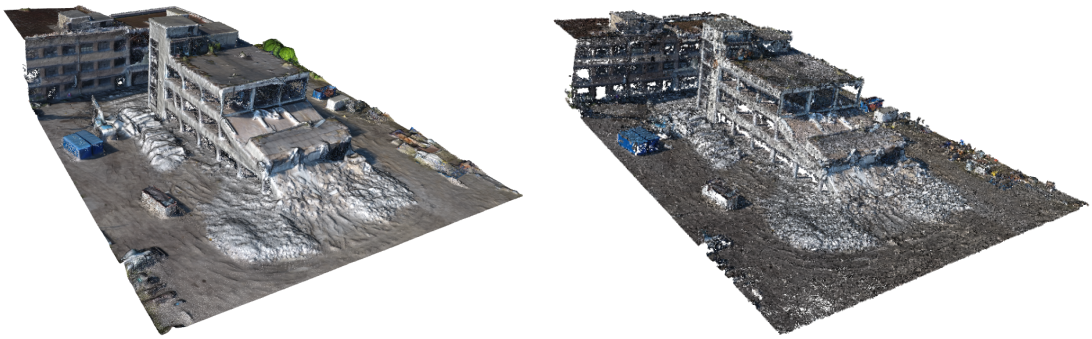


Abbildung 4.2: Photogrammetrisch erzeugte Punktwolken des teilweise eingestürzten Stahlbetongebäudes in Lyon mit *Agisoft Metashape* [10] (links) und *VisualSfM* [175] (rechts).

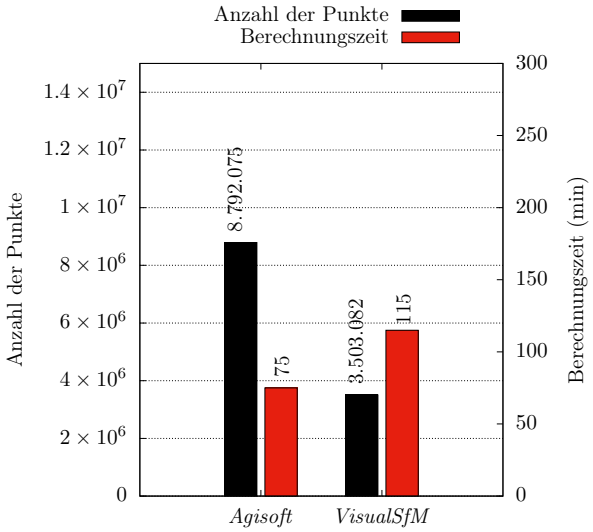


Abbildung 4.3: Zusammenfassung der Ergebnisse der photogrammetrischen Rekonstruktion für das teilweise eingestürzte Stahlbetongebäude in Lyon, durchgeführt mit *Agisoft Metashape* [10] und *VisualSfM* [175].

Die mit *VisualSfM* [175] erzeugte Punktwolke (Abbildung 4.2 (rechts)) weist an vielen Stellen entweder eine niedrige Punktdichte oder größere Datenlücken auf. Angesichts der dafür benötigten Rechenzeit von 115 Minuten ist das erhaltene Ergebnis für den betrachteten Anwendungsfall als mangelhaft zu bewerten. Mit *Agisoft Metashape* [10] wurden hingegen 8.792.075 Punkte ermittelt, was das 1,6-fache der Punktdaten aus *VisualSfM* [175] entspricht, wobei für die Fotoausrichtung und die dichte Rekonstruktion eine mittlere Qualitätseinstellung mit einer milden Filteroption festgelegt wurde. Die Modellkoordinaten wurden mit Hilfe der GPS-Informationen aus den EXIF-Dateien in das Weltkoordinatensystem transformiert. In der erzeugten 3D-Punktwolke (Abbildung 4.2 (links)) können sowohl intakte als auch kollabierte Bauteile eindeutig identifiziert werden. Die hohe Genauigkeit der Punktwolke erlaubt eine visuelle Erfassung von Bauschäden,

insbesondere in gut beleuchteten Bereichen. Außerdem ist in den beschatteten Bereichen kaum ein Rückgang in der Punktdichte festzustellen. Die erforderliche Berechnungszeit von 75 Minuten, ohne Berücksichtigung der Zeit für die Vorbereitung der technischen Ausrüstung und die Datenerfassung, stellen für Ersthelfer einen langen Zeitraum dar. Aus diesem Grund wurde vom Autor dieser Arbeit untersucht, wie die Berechnungsdauer zur Verarbeitung der Bilddaten in *Agisoft Metashape* [10] optimiert werden kann.
In Anlehnung an die in [166] durchgeführte Untersuchung zur Leistungsfähigkeit von *Agisoft Metashape* [10], wurden drei unterschiedliche Bilddatensätze zur photogrammetrischen Rekonstruktion des teilweise eingestürzten Stahlbetongebäudes in Lyon verwendet. Die Datensätze setzen sich wie folgt zusammen:

- #**1:** 312 drohnenbasierte Nadir- und Schrägbilder,
- #**2:** 112 drohnenbasierte Schrägbilder und
- #**3:** 200 drohnenbasierte Nadirbilder

Im ersten Schritt wird der Einfluss der Qualitätseinstellungen in *Agisoft Metashape* [10] sowohl in Bezug auf die Genauigkeit als auch auf die Berechnungszeit analysiert. Dabei wird die Anzahl der erzeugten Punktdaten als Indikator für die Genauigkeit herangezogen. In *Agisoft Metashape* [10] hat der Benutzer die Möglichkeit, zwischen vier Qualitätseinstellungen für die Datenverarbeitung zu wählen: hoch (H), mittel (M), niedrig (L) und sehr niedrig (LL). Jede Einstellung führt zu einer spezifischen Reduzierung der Bildauflösung, wobei der Auflösungsfaktor von hoch (H) auf sehr niedrig (LL) um den Faktor 2 zunimmt. Der Prozess der Fotoausrichtung wurde für die Datensätze #1 bis #3 unter Verwendung der niedrigsten (ALL), niedrigen (AL) und mittleren (AM) Genauigkeitseinstellungen durchgeführt. Die Abschätzung der Kamerapositionen für die Datensätze #2 und #3 ist in der Abbildung 4.4 dargestellt. Es ist wichtig zu beachten, dass der Datensatz #1 die Kamerapositionen der Datensätze #2 und #3 enthält.

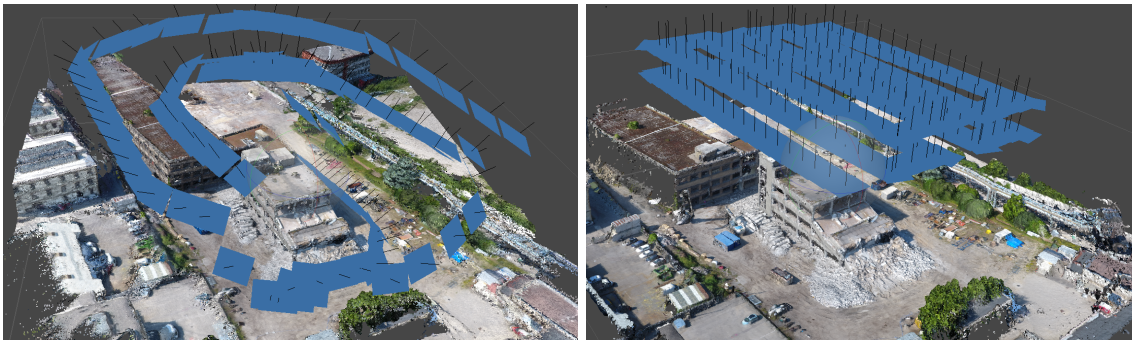


Abbildung 4.4: Abschätzung der Kamerapositionen in *Agisoft Metashape* [10] für die Datensätze #2 (links) und #3 (rechts).

Bei der mittleren Genauigkeitseinstellung (M) wird die Bildauflösung um den Faktor vier reduziert, während bei der niedrigen Genauigkeit (L) eine Verringerung um den Faktor acht erfolgt. Für den niedrigsten Wert (LL) wird die Auflösung um den Faktor 16 reduziert, wobei jede Bildseite halbiert wird. Um den Ausrichtungsprozess zu beschleunigen, wurde für alle Genauigkeitseinstellungen der generische Paar-Vorwahlmodus verwendet. Dieser Modus ermöglicht die Auswahl überlappender Bildpaare durch ein *Matching*-Verfahren mit niedrigeren Qualitätseinstellungen. Auf diese Weise werden redundante Bilder in den Datensätzen bei der Datenverarbeitung vermieden. Die maximale Anzahl von *keypoints* pro Bild wurde auf 4.000.000 begrenzt, während die maximale Anzahl von *tie points* auf 500.000 festgelegt wurde. Für die Erzeugung

dichter Punktwolken wurden die niedrigsten (DLL), niedrigen (DL) und mittleren (DM) Qualitätseinstellungen angewendet, was zu einer Reduzierung der ursprünglichen Bildauflösung von 4000x3000 Pixeln um den Faktor 256, 64 und 16 führte. Die oben genannten Qualitätseinstellungen wurden wie folgt miteinander kombiniert:

(i) ALL+DLL,
(ii) AL+DLL,
(iii) AM+DLL,
(iv) AM+DL,
(v) AL+DL und
(vi) AM+DM.

Insgesamt wurden 18 Berechnungen auf der zuvor genannten *Graphic Workstation* (Tabelle 4.2) durchgeführt, davon sechs für jeden Datensatz unter Berücksichtigung der in (i) bis (vi) genannten Qualitätseinstellungen. Zunächst wurde untersucht, wie lange es dauert, die dünne Punktwolke mit den Qualitätseinstellungen ALL, AL und AM zu erstellen. Die erhaltenen Ergebnisse zur Erstellung der dünnen Punktwolke in *Agisoft Metashape* [10] sind in der Abbildung 4.5 dargestellt.

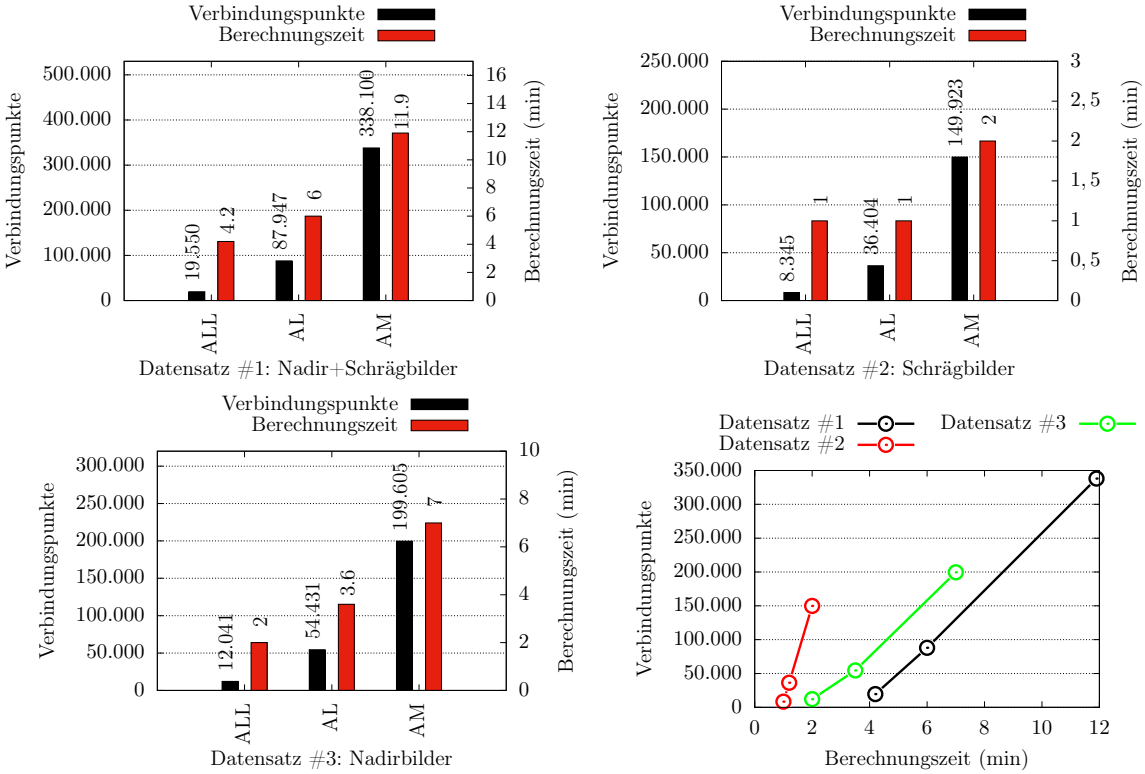


Abbildung 4.5: Vergleich zwischen der benötigten Berechnungszeit und der erzielten Genauigkeit zur Erzeugung einer dünnen Punktwolke in *Agisoft Metashape* [10] für die Datensätze #1 bis #3.

Aus der Abbildung 4.5 ist zu entnehmen, dass sowohl die Genauigkeit als auch die Berechnungszeit mit zunehmender Qualitätseinstellung steigt. Darüber hinaus kann für die verwendeten Datensätze eine annähernd lineare Beziehung zwischen der Genauigkeit und der Berechnungszeit

festgestellt werden. Der Datensatz #2 erzielte die effizientesten Ergebnisse in Bezug auf die Berechnungszeit und die Genauigkeit, wie aus der steilen Steigung in der Abbildung 4.5 (unten rechts) ersichtlich ist. Dies ist auf die vorteilhafte Eigenschaft der Schrägbildaufnahme zurückzuführen, bei der sowohl die Ober- als auch die Seitenflächen von Objekten abgebildet werden. Dadurch werden weniger Bilder benötigt, um eine präzise photogrammetrische Rekonstruktion dreidimensionaler Objekte durchzuführen. Mit einer maximalen Berechnungszeit von zwei Minuten konnten die dünnen Punktwolken für den Datensatz #2 nahezu in Echtzeit erzeugt werden. Die längste Berechnungszeit von etwa 12 Minuten wurde für den Datensatz #1 bei mittlerer Qualitätseinstellung (AM) verzeichnet.
Die Ergebnisse der Fotoausrichtung und dichten Rekonstruktion für die Datensätze #1 bis #3 unter Verwendung der oben genannten Qualitätseinstellungen sind in der Abbildung 4.6 dargestellt. Für die Datensätze #2 und #3 wurden mit der mittleren Qualitätseinstellung (AM+DM) Berechnungszeiten von 15,5 bzw. 47 Minuten gemessen. Die dichte Punktwolke wurde für den Datensatz #2 mit der niedrigsten Qualitätseinstellung (ALL+DLL) in 2,5 Minuten, d. h. nahezu in Echtzeit, erzeugt. Das Diagramm unten rechts in der Abbildung 4.6 zeigt die mit den Datensätzen erzielte Leistungsfähigkeit von *Agisoft Metashape* [10] in Bezug auf die Berechnungszeit und die Genauigkeit.

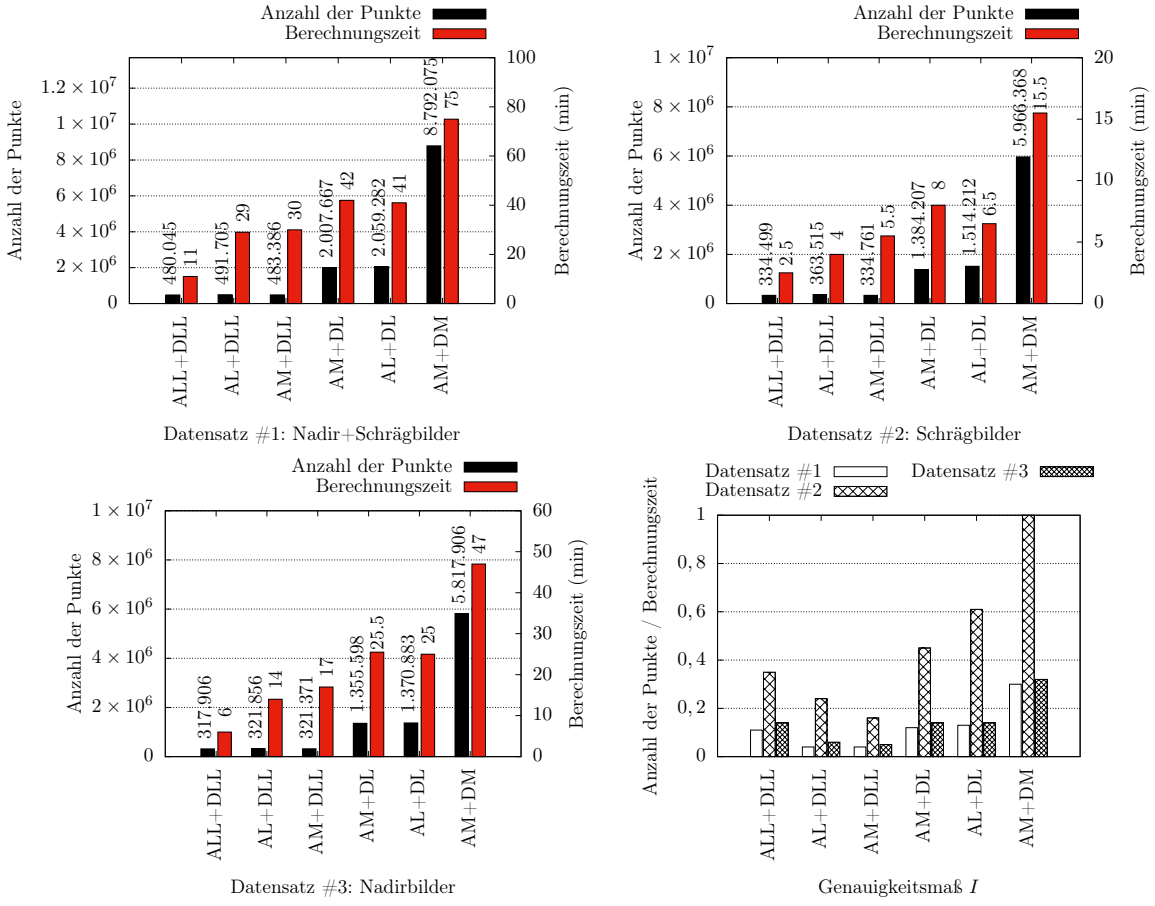


Abbildung 4.6: Vergleich zwischen der benötigten Berechnungszeit und der erzielten Genauigkeit zur Erzeugung einer dichten Punktwolke in *Agisoft Metashape* [10] für die Datensätze #1 bis #3.

Zur besseren Veranschaulichung der Leistungsfähigkeit wurde das Verhältnis zwischen der Anzahl der Punkte und den Berechnungszeiten bestimmt. Dieses Verhältnis beschreibt das Genauigkeitsmaß I. Die Ergebnisse dieses Maßes wurden anschließend im Verhältnis zu den mit der Einstellung AM+DM erzielten Werten normiert. Dabei lässt ein hoher Wert für I auf eine hohe Leistungsfähigkeit der Software bei der jeweiligen Qualitätseinstellung schließen. Aus der Abbildung 4.6 (unten rechts) ist ersichtlich, dass der Datensatz #2 in allen Qualitätsbereichen die höchste Leistung erzielte. Insbesondere bei der Einstellung AM+DM ist das Verhältnis von Genauigkeit zu Berechnungszeit im Vergleich zu den Datensätzen #1 und #3 besonders herausragend. Auf Grundlage dieses Ergebnisses kann eindeutig festgestellt werden, dass die SfM-Photogrammetrie mit Schrägbildaufnahmen nicht nur eine hohe Genauigkeit bietet, sondern auch im Vergleich zu Nadirbilder (Datensatz #2) und Nadir+Schrägbilder (Datensatz #1) innerhalb einer erheblich kürzeren Zeitspanne durchgeführt werden kann.
Im nächsten Schritt wurde eine *Cloud-to-Cloud*-(C2C)-Analyse für die generierten Punktwolken durchgeführt, wobei die Ergebnisse des Datensatzes #1 als Referenzobjekt verwendet wurden. In der Tabelle 4.3 sind die berechneten mittleren Punktabstände Ω und Standardabweichungen σ aus der C2C-Analyse aufgeführt. Wie bereits oben erwähnt, wurden die Ergebnisse des Datensatzes #1 als Referenz verwendet und mit den Punktwolken der Datensätze #2 und #3 verglichen. In der Abbildung 4.7 ist die Beziehung zwischen den berechneten mittleren Punktabständen Ω und den Standardabweichungen σ aus Tabelle 4.3 dargestellt. Dabei lässt sich erkennen, dass bei höheren Qualitätseinstellungen eine annähernd lineare Abnahme beider statistischer Parameterwerte zu beobachten ist. Daraus kann geschlossen werden, dass die Genauigkeit der photogrammetrischen Rekonstruktion bei Verwendung des Datensatzes #1 mit niedriger Qualität (ALL+DLL) herausragend ist, während die Verwendung von Nadir- oder Schrägbildaufnahmen als Eingangsdaten bei höherer Qualität, insbesondere AM+DM, vergleichsweise gute Ergebnisse mit nahezu keinem Unterschied in Bezug auf die Präzision liefert.

Tabelle 4.3: Mittlerer Abstand Ω und Standardabweichung σ von Datensatz #1 im Vergleich zu den Datensätzen #2 und #3.

#	**Qualitätseinstellung**	**C2C-Analyse**	Ω (m)	σ (m)
C2C1	ALL+DLL	#1 Vs #2	0,49	0,39
C2C2	AL+DLL	#1 Vs #2	0,33	0,29
C2C3	AM+DLL	#1 Vs #2	0,33	0,29
C2C4	AL+DL	#1 Vs #2	0,24	0,20
C2C5	AM+DL	#1 Vs #2	0,22	0,17
C2C6	AM+DM	#1 Vs #2	0,17	0,13
C2C7	ALL+DLL	#1 Vs #3	0,53	0,32
C2C8	AL+DLL	#1 Vs #3	0,28	0,19
C2C9	AM+DLL	#1 Vs #3	0,26	0,17
C2C10	AL+DL	#1 Vs #3	0,21	0,15
C2C11	AM+DL	#1 Vs #3	0,27	0,19
C2C12	AM+DM	#1 Vs #3	0,18	0,13

Die durchgeführte Untersuchung der Leistungsfähigkeit von *Agisoft Metashape* [10] zeigt, dass die photogrammetrische Verarbeitung von Schrägbildern innerhalb kurzer Zeit zu hochauflösenden Punktwolken führen. Selbst bei der Verwendung der mittleren Qualitätseinstellung (AM+DM) gelang es, innerhalb von 15,5 Minuten eine 3D-Punktwolke der dargestellten Szene mit 5.966.368 Punkten zu erzeugen. Dies stellt eine bemerkenswerte Leistung im Bereich der SfM-Photogrammetrie dar. Auf der Grundlage dieser Ergebnisse kann der Einsatz der SfM-Photogrammetrie mit Schrägbildern insbesondere für die schnelle Rekonstruktion einer Katastrophenszene emp-

fohlen werden, bei der die Zeit eine entscheidende Rolle spielt. Aus diesem Grund wird die 3D-Punktwolke (Abbildung 4.8), die mit der Qualitätseinstellung AM+DM aus dem Datensatz #2 generiert wurde, zur Evaluierung der entwickelten Methodik ausgewählt.

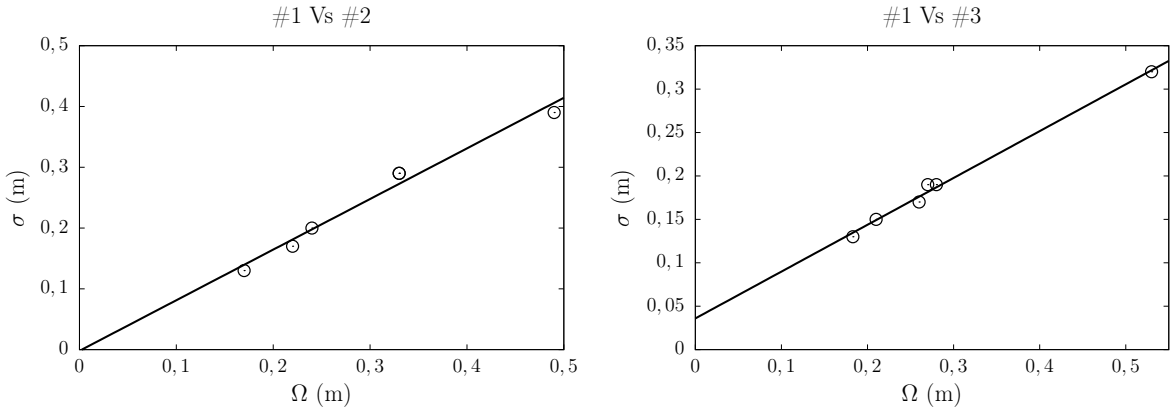


Abbildung 4.7: Beziehung zwischen den Standardabweichungen σ und den Mittelwerten Ω der mit *CloudCompare* [40] erhaltenen Ergebnissen aus den Datensätzen #2 und #3 verglichen mit Datensatz #1.

Abbildung 4.8: Photogrammetrisch erzeugte 3D-Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Lyon aus dem Datensatz #2.

Die ausgewählte Punktwolke wurde mit den in Abschnitt 3.3 beschriebenen Methoden vorverarbeitet. Zunächst wurden die GPS-Daten aus den EXIF-Dateien der Bilder genutzt, um die Punktwolke mit geografischen Koordinaten zu referenzieren. Dieser Schritt ermöglicht eine genaue Zuordnung der photogrammetrisch erzeugten Punkte zu den entsprechenden Aufnahmepositionen und damit eine korrekte Skalierung der dargestellten Szene. Anschließend wurde die Punktwolke mit Hilfe der *crop2D*-Funktion des Python-Moduls *cloudComPy* [40] manuell geschnitten, um den relevanten Bereich des teilweise eingestürzten Stahlbetongebäudes von den nicht benötigten Punktdaten abzugrenzen. Dabei wurde ein rechteckiger Schnittbereich in der XY-Ebene definiert, der in der Abbildung 4.9 zu sehen ist. Durch diesen Verarbeitungsschritt

wurden 46% der Punktdaten entfernt, wodurch in den folgenden Arbeitsschritten die Rechenressourcen effizienter genutzt werden können.

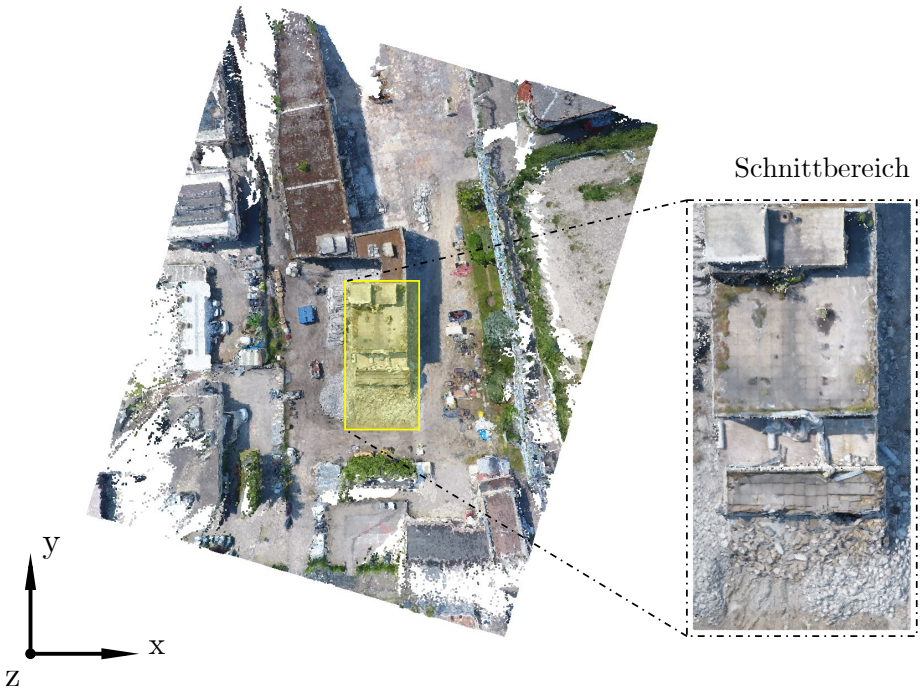


Abbildung 4.9: Definierter Schnittbereich in der XY-Ebene, mit der die relevanten Punktdaten für die Datenverarbeitung des teilweise eingestürzten Stahlbetongebäudes in Lyon ausgeschnitten wurden.

Die Bereinigung der Punktwolke von Ausreißern wurde mit Hilfe des SOR-Filters von *cloudComPy* [40] durchgeführt. Es ist wichtig zu beachten, dass die Entfernung der Ausreißerpunkte mit dem SOR-Filter ein iterativer Prozess ist, bei dem die Parameter k und x aus der statistischen Abstandsformel (Gleichung 3.3) schrittweise angepasst werden, bis die gewünschte Qualität erreicht ist. Dabei definiert k die Anzahl der Nachbarpunkte und x den Multiplikator der Standardabweichung σ_n. Bei der Anpassung der oben genannten Parameter ist die visuelle Begutachtung der identifizierten Ausreißerpunkte von großer Bedeutung, um sicherzustellen, dass keine relevanten Punkte des Tragwerkes verloren gehen. Im Zuge der Anwendung des SOR-Filters mit den Parametern $k = \{100, 200, 300, 400, 500\}$ und $x = \{0,2; 0,4; 0,6; 0,8; 1\}$ wurde festgestellt, dass ein Wert von $x < 1$ zu einer übermäßig konservativen Bereinigung der Punktwolke von Ausreißern führt. Dies ist darauf zurückzuführen, dass bei $x < 1$ der Wert der Standardabweichung σ_n abnimmt, wodurch der zulässige Abstand für einen Nicht-Ausreißerpunkt kleiner wird. Dies führt dazu, dass mehr Punkte als Ausreißer identifiziert und entfernt werden, selbst wenn sie keine tatsächlichen Ausreißer sind. Um dieses Phänomen zu vermeiden, empfiehlt der Autor dieser Arbeit den Multiplikator der Standardabweichung x auf 1 festzulegen. Vor diesem Hintergrund konnte eine visuell angemessene Identifizierung der Ausreißerpunkte mit den Parametern $k = 500$ und $x = 1,0$ erreicht werden. Die identifizierten Ausreißerpunkte entsprechen dabei 23,5% der Datenpunkte der geschnittenen Punktwolke. In der Abbildung 4.10 sind die identifizierten Ausreißerpunkte farblich gekennzeichnet. Wie in der Abbildung 4.10 zu sehen ist, befinden sich diese Punkte aufgrund fehlender Bildinformationen überwiegend innerhalb des Gebäudes.

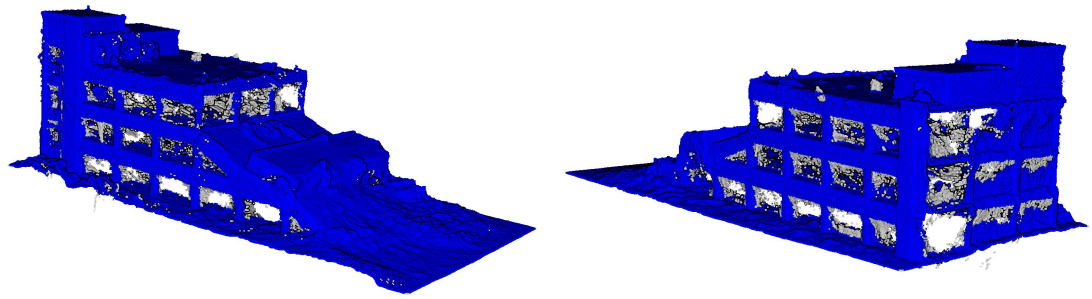


Abbildung 4.10: Darstellung der identifizierten Ausreißerpunkte (grau) und Nicht-Ausreißerpunkte (blau) der Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Lyon.

Im letzten Schritt wurde der *Spatial-Subsample*-Filter von *cloudComPy* [40] angewendet, um eine gleichmäßige Verteilung der Punkte zu gewährleisten. Dabei wurde ein Mindestabstand von 1 cm zum Nachbarpunkt gewählt, wodurch 6,5% der Punktdaten entfernt wurden. Der geringe Prozentsatz der entfernten Punktdaten zeigt, dass die photogrammetrisch erzeugte Punktwolke größtenteils eine hohe Punktdichte aufweist. Das Endergebnis der Vorverarbeitung der Punktwolke des teilweise eingestürzten Stahlbetonrahmens in Lyon ist in der Abbildung 4.11 dargestellt. Die gesamte Vorverarbeitung der Punktdaten dauerte etwa drei Minuten. Dies verdeutlicht, dass die Vorverarbeitung einer Punktwolke dieser Größenordnung innerhalb einer sehr kurzen Zeitspanne durchgeführt werden kann, ohne den Zeitfaktor wesentlich zu beeinträchtigen. Der Datensatz des Fallbeispiels aus Lyon wird im Folgenden als LYN bezeichnet.

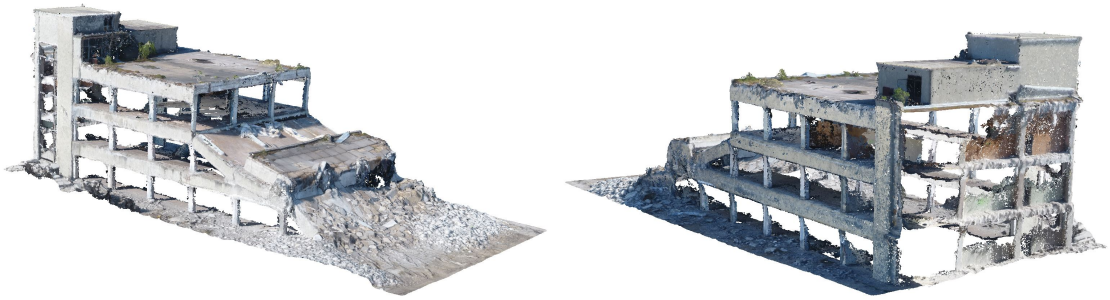


Abbildung 4.11: Vorverarbeitete Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Lyon.

4.1.2 Fallbeispiel Jindires

Im Rahmen der Evaluierung wird ein Fallbeispiel eines teilweise eingestürzten Stahlbetonrahmentragwerkes in Jindires, Syrien, behandelt. Das Gebäude war von dem verheerenden Erdbeben betroffen, das sich am 6. Februar 2023 in der Türkei und in Syrien mit einer Magnitude von 7,8 M_w ereignete [31]. In der Abbildung 4.12 (oben) sind Bilder des zerstörten Gebäudes dargestellt, die aus [129, 173] entnommen wurden. Es handelt sich um ein viergeschossiges Stahlbetonrahmentragwerk mit Mauerwerksausfachung und einer unregelmäßigen, L-förmigen Grundrissform. Das Gebäude ist durch außenliegende Stahlbetonwandscheiben eines Treppenhauses ausgesteift. Bei den sichtbaren Stahlbetonstützen handelt es sich um Rechteckstützen. Im Zuge der heftigen Bodenbewegungen wurden die Mauerwerkswände in den oberen Geschossen nahezu vollständig zerstört. Zusätzlich stürzte die Decke des ersten Obergeschosses teilweise auf die Decke des Erdgeschosses, was zu einer erheblichen Schiefstellung des Gebäudes ab dem ersten Obergeschoss

führte. Auf dem Dach des Gebäudes befinden sich mehrere Behälter und zwei Solarmodule. Da keine Baupläne oder sonstige Maßangaben des Gebäudes gefunden wurden, wurden für die Nachmodellierung dieses Fallbeispiels realistische Annahmen bezüglich der Maße getroffen. Diese angenommen Maßangaben für das Gebäude sind im Anhang in der Abbildung A.1 dargestellt.

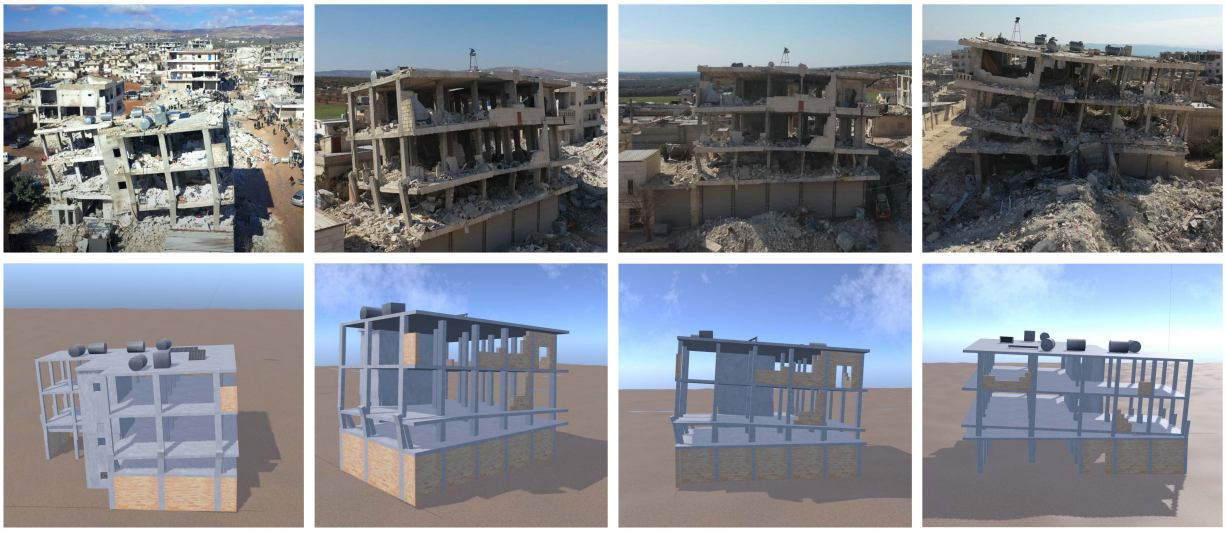


Abbildung 4.12: Bilder des teilweise eingestürzten Stahlbetongebäudes in Jindires (oben, entnommen aus [129, 173]) und gerenderte Bilder des nachgebildeten 3D-Modells in *Blender* [23] (unten).

Die Nachmodellierung des Gebäudes wurde mit Hilfe von *Blender* [23] durchgeführt. Zunächst wurde das Gebäude in seinem intakten Zustand basierend auf den in Abbildung A.1 angegebenen Abmessungen modelliert. Für die Beschreibung der Bauteile wurden IFC unter Verwendung des *BlenderBIM*-Add-ons [24] genutzt. Anschließend wurden die seismisch induzierten Gebäudeschäden so weit wie möglich auf Basis der in [129, 173] dargestellten Schadensbilder nachgebildet. Es ist wichtig zu beachten, dass Trümmer oder Möbel im Inneren des Gebäudes bei der Nachmodellierung nicht berücksichtigt wurden. Um eine möglichst realistische digitale Darstellung des Gebäudes zu erreichen, wurden für die Oberflächen von Beton und Mauerwerk lizenzfreie Texturen verwendet. Darüber hinaus wurden ähnliche Lichtverhältnisse wie in den Aufnahmen aus [129, 173] geschaffen. Dies wurde in *Blender* [23] durch die Verwendung einer synthetischen Sonne und eines bewölkten Himmels im Rendering-Modus umgesetzt. Zur Vereinfachung des Katastrophenschauplatzes wurde die unmittelbare Umgebung nicht nachmodelliert. Das Ergebnis der digitalen Nachmodellierung des teilweise eingestürzten Stahlbetonrahmentragwerkes in Jindires ist in der Abbildung 4.12 (unten) dargestellt.
Die Ergebnisse der Auswertung in Unterabschnitt 4.1.1 haben gezeigt, dass die Schrägbildphotogrammetrie besonders effizient ist. Aus diesem Grund wurde dieser Ansatz zur Erstellung des synthetischen Bilddatensatzes für die photogrammetrische Rekonstruktion des Gebäudes gewählt. Um eine drohnenbasierte Aufnahme des Gebäudes für photogrammetrische Zwecke zu simulieren, wurde die Kamerafunktion in *Blender* [23] genutzt. Die Erzeugung der Kameras für die Aufnahme der Schrägbilder erfolgte mit Hilfe der Python-API von *Blender* [23]. Dabei wurden 40 Kameras auf jeweils zwei verschiedenen Höhenlagen über der GOK definiert. Diese wurden entlang eines Kreisumfangs mit einem Radius von 30 m gleichmäßig verteilt und in einem Neigungswinkel von 70 Grad bzw. 45 Grad auf das Gebäude ausgerichtet. Die erzeugten Kameras sind in der Abbildung 4.13 dargestellt.

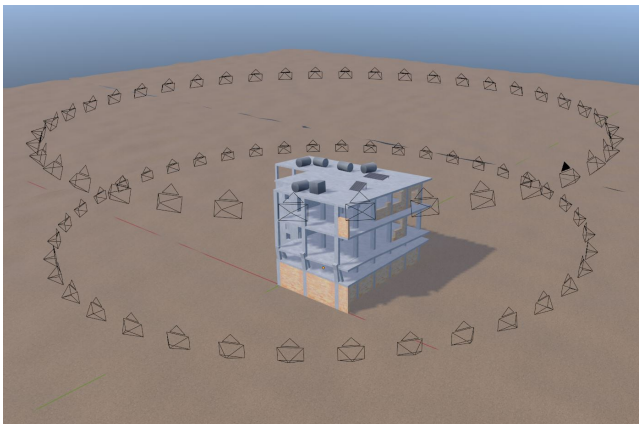


Abbildung 4.13: Erzeugte Kameras in *Blender* [23] zur photogrammetrischen Aufnahme des teilweise eingestürzten Stahlbetonrahmentragwerkes in Jindires.

Um den technischen Spezifikationen herkömmlicher Drohnenkameras zu entsprechen, wurden die von den Kameras erzeugten Schrägbilder mit einer Brennweite von 20 mm und einer Bildauflösung von 1920x1080 Pixeln aufgenommen. Die photogrammetrische Datenverarbeitung der Bilder wurde anschließend in *Agisoft Metashape* [10] mit der mittleren Qualitätseinstellung (AM) für die Fotoausrichtung und der hohen (DH) für die dichte Rekonstruktion durchgeführt. Die gesamte Verarbeitungszeit bis zur Erstellung der dichten Punktwolke betrug 2,73 Minuten, wobei 10.170.424 Punktdaten rekonstruiert wurden. In der Abbildung 4.14 ist die erzeugte 3D-Punktwolke der Katastrophenszene dargestellt. Wie aus der Abbildung 4.14 ersichtlich ist, konnte aus dem synthetisch erzeugten Bilddatensatz des teilweise eingestürzten Stahlbetongebäudes in Jindires eine realitätsnahe 3D-Punktwolke der Szene generiert werden. Durch die Anwendung der SfM-Photogrammetrie für den synthetischen Bilddatensatz war es außerdem möglich, realistische Störfaktoren wie Rauscheffekte und Datenlücken im Inneren des Gebäudes zu erzeugen.

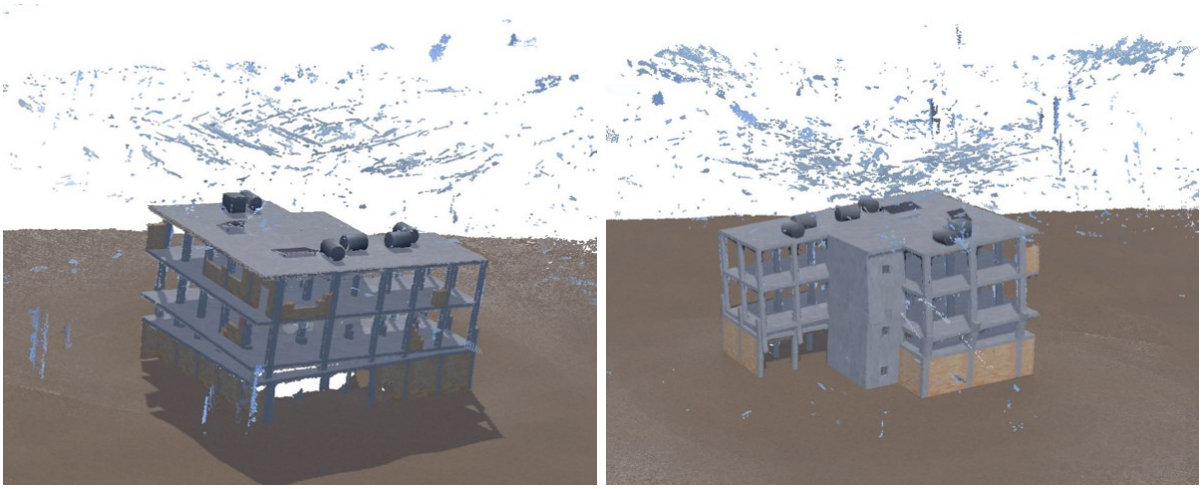


Abbildung 4.14: Photogrammetrisch erzeugte 3D-Punktwolke der teilweise eingestürzten Stahlbetonrahmenkonstruktion in Jindires.

Für das vorliegende Fallbeispiel des teilweise eingestürzten Stahlbetongebäudes in Jindires wurden die gleichen Methoden wie in Unterabschnitt 4.1.1 zur Vorverarbeitung der photogrammetrisch erzeugten 3D-Punktwolke angewendet. Im ersten Schritt wurde die *crop2D*-Funktion von *cloudComPy* [40] angewendet, um nicht relevante Punktdaten auszuschneiden. Die festgelegten

Schnittbereiche sind in der Abbildung 4.15 dargestellt. Durch diesen Schritt wurde die Anzahl der Punkte in der Punktwolke um 75% reduziert. Dieser hohe Prozentsatz verdeutlicht, dass ein erheblicher Teil der erzeugten Punktwolke nicht relevante Punktdaten der Umgebung enthält, insbesondere der Boden und grobe Rauscheffekte durch den Himmel. Darüber hinaus ist die signifikant reduzierte Datenmenge mit einer effizienteren und kürzeren Datenverarbeitung verbunden.

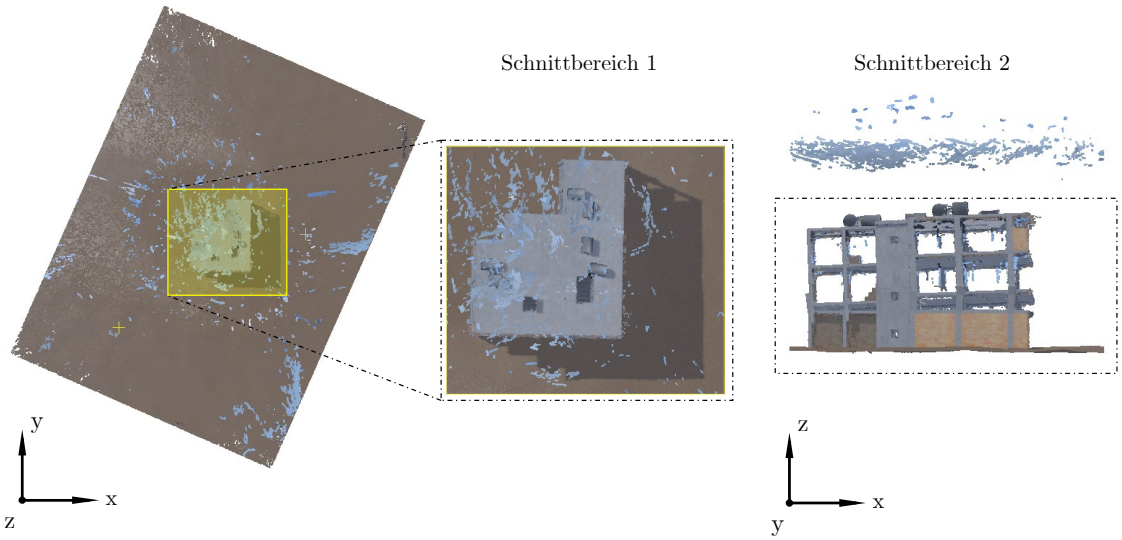


Abbildung 4.15: Definierte Schnittbereiche in der XY- und XZ-Ebene, mit der die relevanten Punktdaten des Stahlbetongebäudes in Jindires für die Datenverarbeitung ausgeschnitten wurden.

Die Bereinigung der Punktwolke von Ausreißern wurde mit dem SOR-Filter von *cloudComPy* [40] durchgeführt. Dabei wurden die Parameter $k = 500$ und $x = 1,0$ verwendet. Dieser Schritt führte zu einer Reduzierung der Datenpunkte der geschnittenen Punktwolke um 5%. Die mit dem SOR-Filter identifizierten Ausreißerpunkte für das Stahlbetongebäude in Jindires sind in der Abbildung 4.16 zu sehen. Wie erwartet, befinden sich die meisten der identifizierten Ausreißerpunkte im Inneren des Gebäudes, insbesondere im Bereich der Stützen, die aufgrund fehlender Bildinformationen nicht vollständig photogrammetrisch rekonstruiert werden konnten.

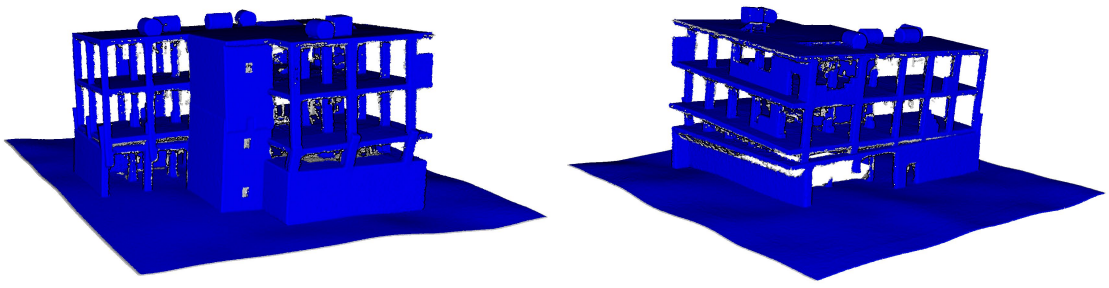


Abbildung 4.16: Darstellung der identifizierten Ausreißerpunkte (grau) und Nicht-Ausreißerpunkte (blau) der Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Jindires.

Zur weiteren Datenbereinigung wurde das *Downsampling* mit der *Spatial-Subsample*-Funktion von *cloudComPy* [40] durchgeführt. Dabei wurde ein Mindestabstand von 1 cm zwischen den

Punkten festgelegt, um die gleiche Punktdichte zu erhalten, wie die LYN-Punktwolke (Unterabschnitt 4.1.1). Als Ergebnis des *Downsamplings* wurden 5% der Punkte aus der bereinigten Punktwolke entfernt. Die Abbildung 4.17 zeigt das Endergebnis der vorverarbeiteten Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Jindires, das im Rahmen der Evaluierung der entwickelten Methodik verwendet wird. Die gesamte Vorverarbeitung des Datensatzes dauerte insgesamt zwei Minuten. Im weiteren Verlauf wird der Datensatz des Fallbeispiels aus Jindires als JDS bezeichnet.

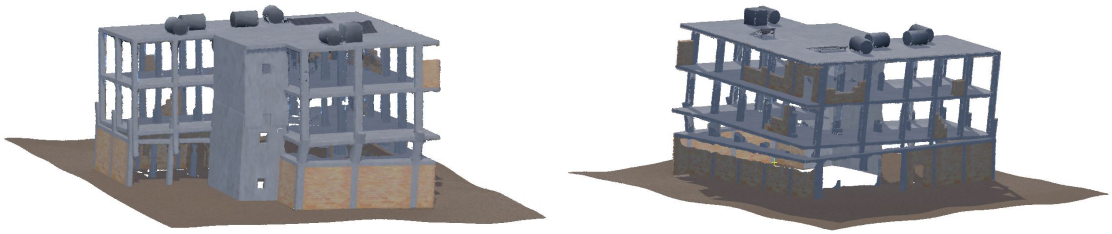


Abbildung 4.17: Vorverarbeitete Punktwolke des teilweise eingestürzten Stahlbetongebäudes in Jindires.

4.1.3 Fallbeispiel Gölcük

Das letzte Fallbeispiel, das zur Evaluierung der entwickelten Methodik herangezogen wird, ist ein Wohngebäude in Gölcük, Türkei. Dieses Gebäude wurde infolge eines starken Erdbebens, das sich am 17. August 1999 in der türkischen Provinz Kocaeli ereignete [57], beschädigt. In der Abbildung 4.18 (oben) sind Bilder des Wohngebäudes dargestellt, die nach dem Erdbebenereignis aufgenommen wurden. Wie in der Abbildung 4.18 (oben) zu sehen ist, handelt es sich um ein sechsgeschossiges Stahlbetonrahmentragwerk mit Mauerwerksausfachung, das im Grundriss regelmäßig ist.

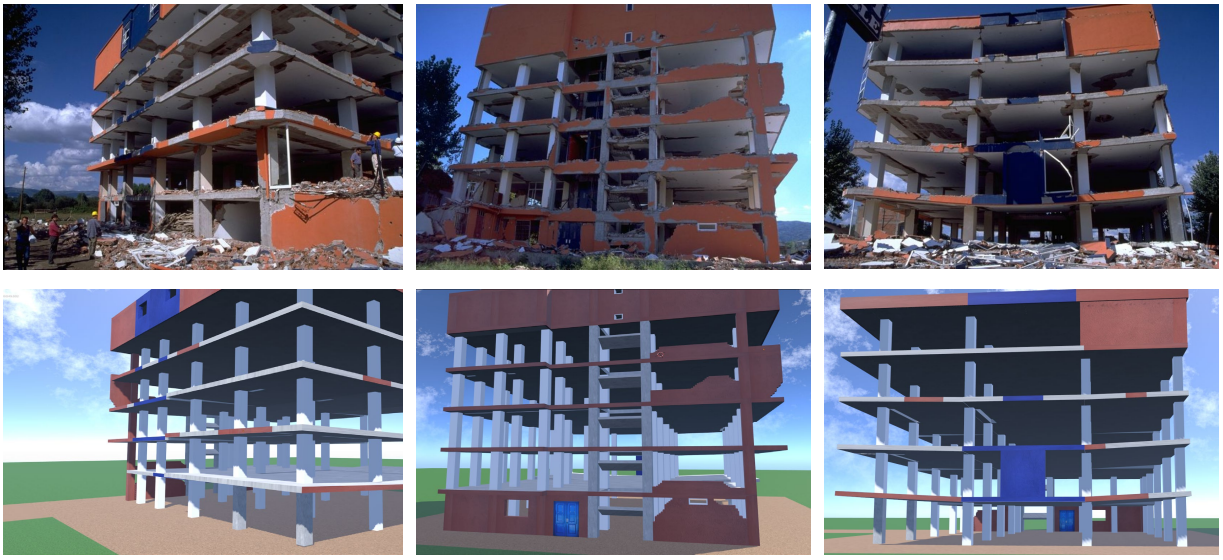


Abbildung 4.18: Bilder des seismisch beschädigten Stahlbetongebäudes in Gölcük (oben, entnommen aus [43]) und gerenderte Bilder des nachgebildeten 3D-Modells in *Blender* [23] (unten).

Die Grundrissabmessungen des Gebäudes sind aus [43] entnommen und betragen etwa 19,4 m in der Breite, 23,2 m in der Länge und 18 m in der Höhe (Abbildung A.2). Um eine ausreichende

Aussteifung gegen horizontale Einwirkungen zu gewährleisten, wurden Stahlbetonrahmensysteme sowohl in Längs- als auch in Querrichtung vorgesehen. An der Rückseite des Gebäudes befindet sich ein Treppenhaus, das mit den äußeren Rahmenstützen verbunden ist. Die Treppenpodeste liegen dabei etwa 1 m unterhalb der Riegel-Stützen-Verbindungen. Die Ausfachungswände des Gebäudes wurden aus Hohltonziegeln hergestellt und gezielt angeordnet, um die Steifigkeit des Gebäudes zu erhöhen [43, 152]. Nach dem Erdbeben wies eine beträchtliche Anzahl der Stützen im ersten Stock erhebliche strukturelle Schäden auf und versagte entweder aufgrund von Schubbelastung oder axialem Druck. Insbesondere die Stützen im Treppenhaus brachen direkt über den Treppenpodesten aufgrund von Schubversagen ein. Es wurden wiederholt Betonabplatzungen und Schubversagen an den Verbindungsstellen zwischen Trägern und Stützen festgestellt. Aufgrund der Verkürzung der Stützen erfuhren die Deckenplatten eine übermäßige Durchbiegung. Des Weiteren wurden die Mauerwerkswände in den ersten fünf Stockwerken größtenteils zerstört, während sie im obersten Stockwerk nahezu unbeschädigt blieben [43, 152].
Wie das vorherige Fallbeispiel JDS (Unterabschnitt 4.1.2) wurde die Nachmodellierung des beschädigten Stahlbetonrahmengebäudes in Gölcük mit Hilfe von *Blender* [23] und des *BlenderBIM*-Add-Ons [24] durchgeführt. Als Grundlage für die manuelle Nachmodellierung diente der Grundrissplan, der aus [43] entnommen wurde, sowie die Schadensbilder, die in der Abbildung 4.18 (oben) dargestellt sind. Es ist wichtig zu beachten, dass aufgrund fehlender Angaben zu den genauen Abmessungen der Stützen, Mauerwerkswände und Träger vom Autor dieser Arbeit realistische Annahmen für die Modellierung dieser Bauteile getroffen wurden. Das in *Blender* [23] nachgebildete 3D-Modell des beschädigten Stahlbetonrahmengebäudes in Gölcük ist in der Abbildung 4.18 (unten) dargestellt. Wie in der Abbildung 4.18 (unten) zu sehen ist, stellt das erstellte Gebäudemodell den beschädigten Zustand und die vorhandenen Texturen des realen Gebäudes äußerst realistisch dar. Für die digitale Simulation der Drohnenaufnahme des Gebäudes wurden insgesamt 120 Kameras mit Hilfe der Python-API von *Blender* [23] erzeugt. Dabei wurden drei verschiedene Höhenlagen berücksichtigt, wobei in jeder Höhenlage 40 Kameras entlang eines Kreisumfangs mit einem Radius von 30 m angeordnet wurden. Die in *Blender* [23] erzeugten Kameras für die Schrägaufnahme des Gebäudemodells mit einer Brennweite von 20 mm und einer Auflösung von 1920x1080 Pixeln sind in der Abbildung 4.19 dargestellt. Die photogrammetrische Rekonstruktion des Gebäudemodells sowie die nachfolgende Vorverarbeitung der erzeugten 3D-Punktwolke erfolgten analog zum JDS-Fallbeispiel (Unterabschnitt 4.1.2).

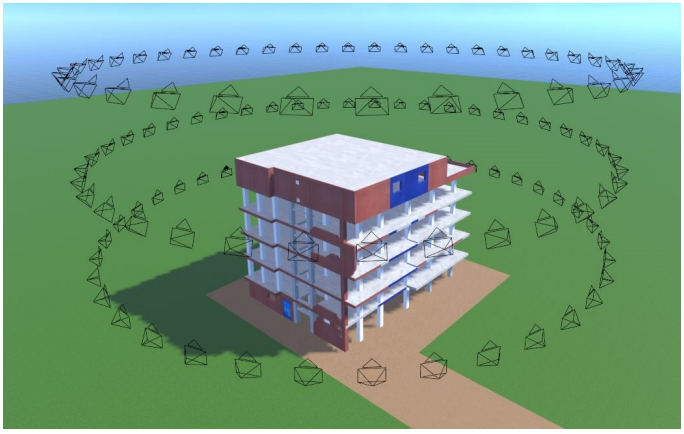


Abbildung 4.19: Erzeugte Kameras in *Blender* [23] zur photogrammetrischen Aufnahme des teilweise eingestürzten Stahlbetonrahmentragwerkes in Gölcük.

Die Abbildung 4.20 zeigt die photogrammetrisch erzeugte 3D-Punktwolke der beschädigten Stahlbetonrahmenkonstruktion in Gölcük. Die Punktwolke wurde mit *Agisoft Metashape* [10] unter

Verwendung der Qualitätseinstellung AM+DH erstellt, wobei insgesamt fünf Minuten für die Erstellung von 10.240.724 Punktdaten benötigt wurden. Während die Punktwolke nahezu vollständige Darstellungen der Rahmensysteme an der Gebäudefassade aufweist, sind im Innenbereich des Gebäudes große Datenlücken zu verzeichnen. Wie bereits im vorherigen Fallbeispiel erwähnt, sind diese Datenlücken auf unzureichende Bildinformationen aus dem synthetischen Bilddatensatz der von außen aufgenommenen Schrägbilder zurückzuführen. Ähnliche Datenlücken wurden auch in der LYN-Punktwolke festgestellt (Abbildung 4.11), das aus realen Schrägbildaufnahmen generiert wurde. In diesem Zusammenhang kann gefolgert werden, dass der synthetische Bilddatensatz reale Störfaktoren und Datenlücken erfolgreich reproduzieren konnte.
Der definierte Schnittbereich in der XY-Ebene zur Beschränkung des Punktdatensatzes auf den relevanten Bereich ist in der Abbildung 4.21 dargestellt. Durch diesen Schnitt wurde die Punktwolke um 52% ihrer ursprünglichen Datenmenge reduziert.

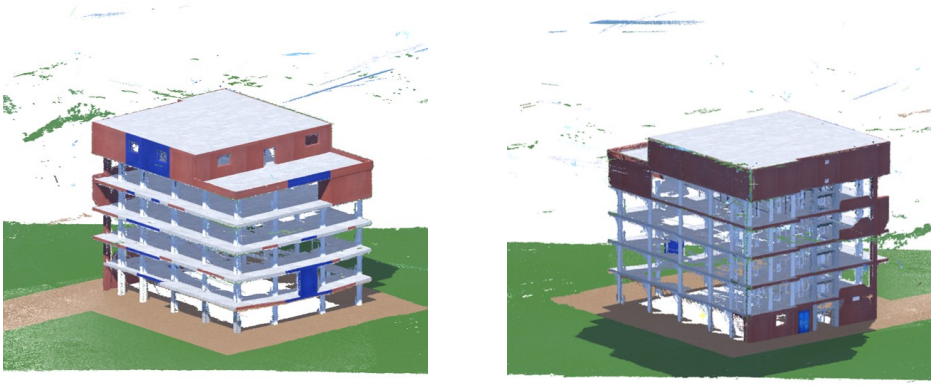


Abbildung 4.20: Photogrammetrisch erzeugte 3D-Punktwolke des seismisch beschädigten Stahlbetongebäudes in Gölcük.

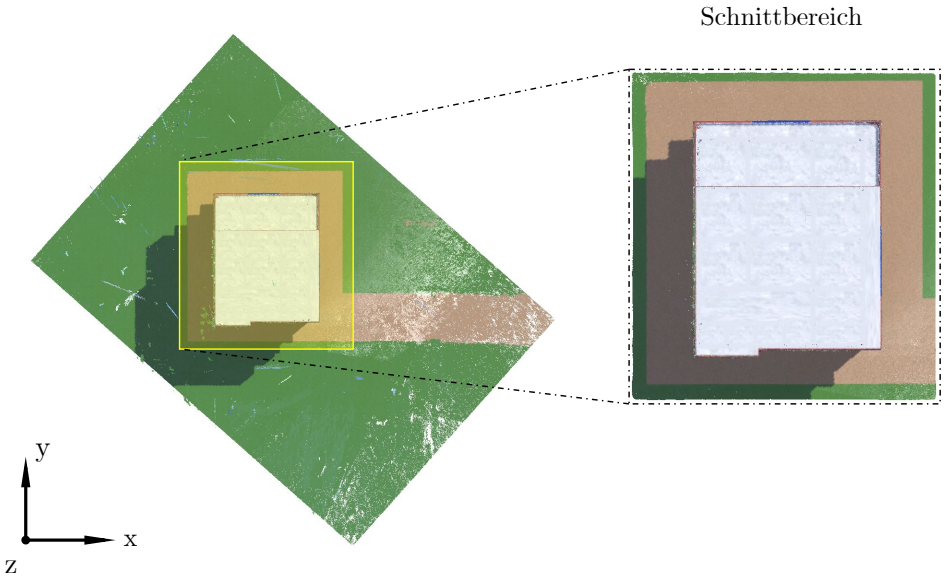


Abbildung 4.21: Definierter Schnittbereich in der XY-Ebene, mit der die relevanten Punktdaten des Stahlbetongebäudes in Gölcük für die Datenverarbeitung ausgeschnitten wurden.

Durch die Anwendung des SOR-Filters [40] mit den Parametern $k = 500$ und $x = 1,0$ konnten insgesamt 347.293 Ausreißer in den Punktdaten identifiziert werden, was 7,1% der geschnittenen Punktwolke entspricht (Abbildung 4.22). Schließlich führte der *Spatial-Subsample*-Filter [40] mit einem festgelegten Mindestabstand von 1 cm zu den Nachbarpunkten zu einer weiteren Reduzierung der Datenmenge um 7%. Die gesamte Vorverarbeitung der Punktwolke des Fallbeispiels aus Gölcük wurde innerhalb von drei Minuten abgeschlossen. Die Abbildung 4.23 zeigt das Endergebnis der vorverarbeiteten Punktwolke des seismisch beschädigten Stahlbetongebäudes in Gölcük. Zur Vereinfachung wird dieses Fallbeispiel im Folgenden als GLC abgekürzt.

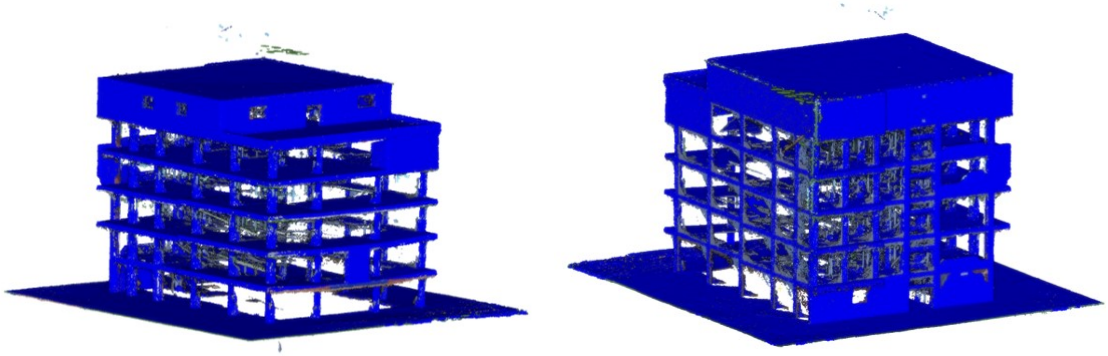


Abbildung 4.22: Darstellung der identifizierten Ausreißerpunkte (grau) und Nicht-Ausreißerpunkte (blau) der Punktwolke des seismisch beschädigten Stahlbetongebäudes in Gölcük.

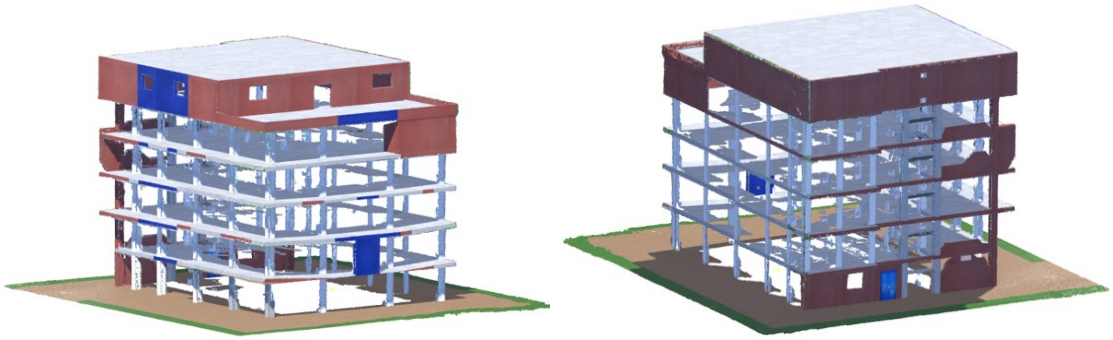


Abbildung 4.23: Vorverarbeitete Punktwolke des seismisch beschädigten Stahlbetongebäudes in Gölcük.

4.2 Segmentierung

Die Evaluation der entwickelten Methodik beginnt mit dem Arbeitsschritt der Segmentierung von Punktdaten aus einer 3D-Punktwolke (Abschnitt 3.4). Hierbei wird untersucht, inwieweit die dafür vorgesehenen digitalen Methoden in der Lage sind, die sichtbaren Komponenten einer seismisch beschädigten Stahlbetonrahmenkonstruktion, d. h. Decken, Wände, Träger und Stützen, aus einer Punktwolke zu segmentieren. Die erzielten Ergebnisse werden sowohl quantitativ als auch visuell ausgewertet. Besonderes Augenmerk wird auf die korrekte Identifizierung der Oberflächen der Bauteile sowie die dafür benötigte Rechenzeit im Rahmen der halbautomatischen Datenverarbeitung gelegt. Darüber hinaus wird auch die Robustheit der Segmentierung gegenüber Variationen der Eingabeparameter sowie ihre Empfindlichkeit gegenüber Störungen (Rauschen, Datenlücken) untersucht, um eine umfassende Analyse der Zuverlässigkeit und Konsistenz der Segmentierungsmethoden in verschiedenen Szenarien zu ermöglichen.

4.2.1 Ebene Punktmengen

Das Ziel der in Unterabschnitt 3.4.1 beschriebenen Methode ist die automatische Extraktion und Gruppierung von ebenen Punktmengen basierend auf dem RANSAC-Algorithmus [149]. Die identifizierten Punktsegmente repräsentieren dabei die ebenen Oberflächen der Komponenten einer Stahlbetonrahmenkonstruktion. Zusätzlich dazu wird überprüft, ob den identifizierten Punktmengen ein parallel verlaufendes Segment zugeordnet werden kann. Dieses zugeordnete Segment ermöglicht die Bestimmung der Dicke des Bauteils, das im Rahmen der digitalen 3D-Rekonstruktion genutzt wird. Die Anwendung dieser Methode erfordert die Angabe von mehreren Parametern, die in der Tabelle 4.4 aufgelistet sind.

Tabelle 4.4: Beschreibung der Parameter in der Methode zur Segmentierung von ebenen Punktmengen (Algorithmus 1).

Parameter	Einheit	Beschreibung	Bedingung
m	[-]	Mindestpunktanzahl zur Identifizierung einer Ebene	$m \geq 3$
ϵ	[m]	Maximaler orthogonaler Abstand von der Ebene	$\epsilon \in \mathbb{R}^+$
d_{bitmap}	[m]	Maximaler Abstand zwischen Punktgruppen	$d_{\text{bitmap}} \in \mathbb{R}^+$
a	[deg]	Maximale Normalabweichung der Punktdaten von der Ebene	$a \in \mathbb{N}^+$
t_{min}	[m]	Minimale Komponentendicke	$t_{\text{min}} \in \mathbb{R}^+$
t_{max}	[m]	Maximale Komponentendicke	$t_{\text{max}} \in \mathbb{R}^+$
α_{max}	[deg]	Maximale Normalabweichung zweier Segmente (Parallelität)	$\alpha_{\text{max}} \in \mathbb{N}^+$

Die Evaluierung der Segmentierungsmethode erfolgte durch die Analyse der vorverarbeiteten Punktwolken von beschädigten Stahlbetonrahmengebäuden aus den Städten Lyon, Jindires und Gölcük. Dabei wurden jeweils zwei Durchläufe (#1 und #2) für die Auswertung herangezogen. Die gewählten Parameterwerte für die oben genannten Eingabedaten sind in der Tabelle 4.5 aufgeführt. Es ist wichtig zu beachten, dass die Parameterwerte für ϵ und d_{bitmap} im ersten Durchlauf (#1) nach der Methode von *Schnabel et al.* [149] automatisch ermittelt wurden. Im Gegensatz dazu wurden die Werte für m und a programmintern standardmäßig auf 500 bzw. 25 Grad festgelegt. Im zweiten Durchlauf (#2) wurden diese Parameter auf Basis der Ergebnisse des ersten Durchlaufes angepasst, um die Qualität der Ergebnisse zu optimieren. Die Parameter für die Identifizierung von parallelen Segmenten, d. h. t_{min}, t_{max} und α_{max}, wurden in allen Durchläufen konstant gehalten.

Tabelle 4.5: Gewählte Parameterwerte für die ebene Punktsegmentierung der vorverarbeiteten 3D-Punktwolken der Datensätze LYN, JDS und GLC.

Parameter	Einheit	LYN		JDS		GLC	
		#1	#2	#1	#2	#1	#2
m	[-]	500	1000	500	1000	500	100
ϵ	[m]	0,207	0,2	0,154	0,13	0,177	0,18
d_{bitmap}	[m]	0,414	0,2	0,308	0,3	0,355	0,35
a	[deg]	25	20	25	20	25	20
t_{min}	[m]	0,18	0,18	0,18	0,18	0,18	0,18
t_{max}	[m]	0,45	0,45	0,45	0,45	0,45	0,45
α_{max}	[deg]	5	5	5	5	5	5

Um die Präzision der Bauteilsegmentierung zu erhöhen und somit die Genauigkeit der nachfolgenden Arbeitsschritte zu verbessern, wurde die vorverarbeitete LYN-Punktwolke in einen intakten und einen eingestürzten Bereich unterteilt (Abbildung 4.24). Die Ergebnisse der ebenen Punktsegmentierung für LYN, JDS und GLC unter Verwendung der in Tabelle 4.5 zusammengefassten Parameterwerte sind in den Abbildungen 4.25 bis 4.28 dargestellt.

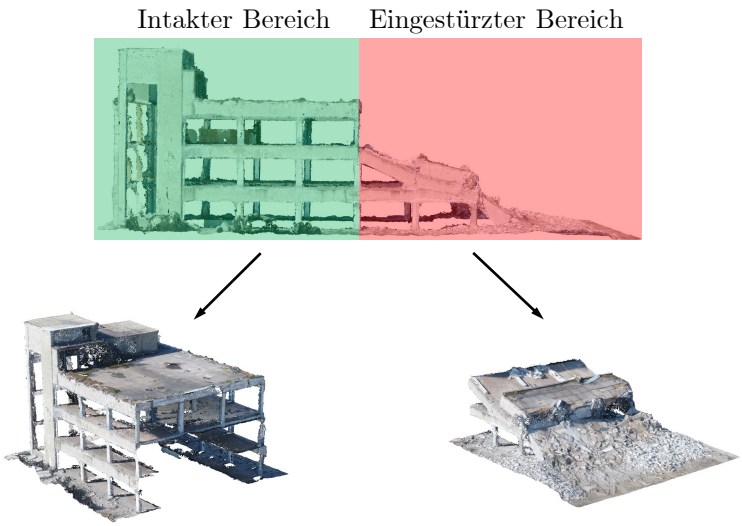
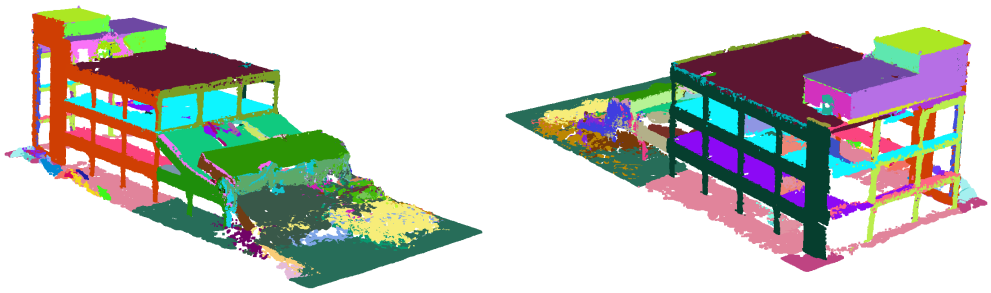


Abbildung 4.24: Unterteilung der vorverarbeiteten LYN-Punktwolke in intakte (grün) und eingestürzte Bereiche (rot).

In Durchlauf #1 wurden für die LYN-Punktwolke insgesamt 104 ebene Segmente ermittelt (Abbildung 4.25). Davon konnten neun Segmente einem parallelen Segment zugeordnet werden. Die Berechnungszeit für diesen Durchlauf betrug 1,42 Minuten. Im anschließenden Durchlauf (#2) reduzierte sich sowohl die Anzahl der identifizierten ebenen Segmente auf 67 als auch die Anzahl der parallelen Segmente auf zwei. Gleichzeitig verkürzte sich die Berechnungszeit auf 0,96 Minuten. Die Veränderungen in Bezug auf die Anzahl der identifizierten Segmente deuten darauf hin, dass im ersten Durchlauf eine Übersegmentierung (engl. *oversegmentation*) der Punktdaten auftrat, welche im zweiten Durchlauf durch Anpassung der Parameterwerte erfolgreich reduziert wurde. Die Ergebnisse des ersten Durchlaufes dienten dabei als Leitfaden zur Optimierung der Parameter im zweiten Durchlauf. Die Übersegmentierung in der LYN-Punktwolke ist insbesondere auf den eingestürzten Bereich zurückzuführen (Abbildung 4.26). In diesem Bereich war der RANSAC-Algorithmus aufgrund der Oberflächengeometrie der Trümmer in der Lage, eine große Anzahl kleiner, getrennter Punktsegmente zu finden. Dies verdeutlicht den Einfluss von Trüm-

merhaufen und ihre individuelle geometrische Anordnung auf die Segmentierungsergebnisse und unterstreicht die Bedeutung der Parameteranpassung, um eine Übersegmentierung in solchen Bereichen zu minimieren.

LYN #1: 104 Segmente, t = 1,42 min



LYN #2: 67 Segmente, t = 0,96 min

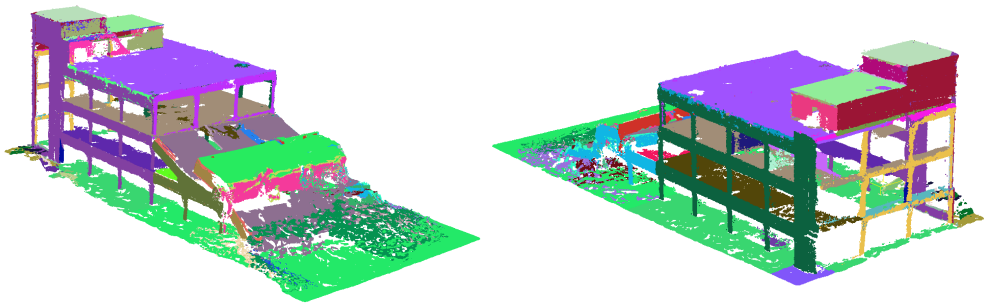


Abbildung 4.25: Ergebnisse der ebenen Punktsegmentierung für die LYN-Punktwolke nach den Durchläufen #1 und #2, wobei jede Farbe ein identifiziertes Segment repräsentiert.

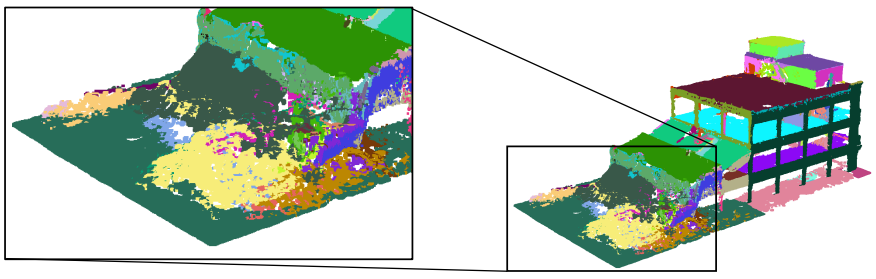
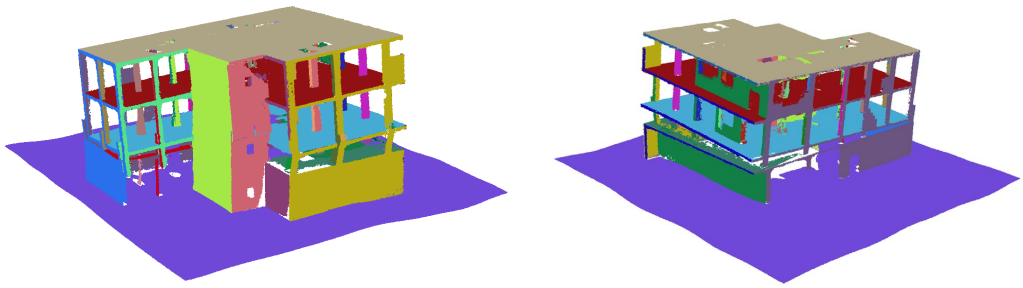


Abbildung 4.26: Störbereich in der LYN-Punktwolke für die ebene Punktsegmentierung im Durchlauf #1.

Die Anwendung der ebenen Punktsegmentierung auf die JDS-Punktwolke führte im ersten Durchlauf (#1) zur Identifikation von 34 ebenen Punktmengen und sechs parallelen Segmenten innerhalb einer Berechnungszeit von 1,2 Minuten (Abbildung 4.27). Im zweiten Durchlauf (#2) stieg die Anzahl der identifizierten Punktmengen leicht auf 42 an, während die Anzahl der parallelen Segmente auf drei reduziert wurde. Hierbei blieb die Berechnungszeit mit 1,23 Minuten nahezu konstant. Die geringfügigen Veränderungen bei der Segmentierung und der Berechnungszeit im zweiten Durchlauf lassen darauf schließen, dass die automatische Parametereinstellung im ersten Durchlauf bereits zu angemessenen Ergebnissen geführt hat.

JDS #1: 40 Segmente, t = 1,20 min



JDS #2: 42 Segmente, t = 1,23 min

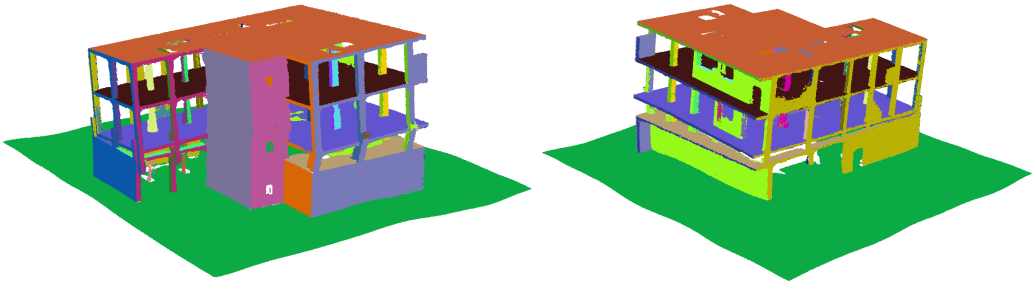


Abbildung 4.27: Ergebnisse der ebenen Punktsegmentierung für die JDS-Punktwolke nach den Durchläufen #1 und #2, wobei jede Farbe ein identifiziertes Segment repräsentiert.

Die Segmentierung der GLC-Punktwolke im ersten Durchlauf (#1) ergab eine Aufteilung in 34 ebene Segmente, von denen zwei als parallele Segmente identifiziert wurden (Abbildung 4.28). Die Berechnungszeit für diesen Durchlauf betrug 0,6 Minuten. Der zweite Durchlauf (#2) führte dagegen zu einem deutlichen Anstieg der identifizierten Segmente auf 83 ebene Punktmengen und neun parallele Segmente. Die Berechnung dauerte 0,7 Minuten. Diese deutliche Steigerung der Segmentanzahl im zweiten Durchlauf ergab sich vorwiegend aus der Reduzierung des Parameters m auf den Wert 100. Dadurch wurde ermöglicht, dass eine geringe Anzahl von Punkten ausreichte, um eine ebene Punktmenge zu definieren. Die Anpassung des Parameters m erfolgte auf der Grundlage einer sorgfältigen Überprüfung der Segmentierungsergebnisse des ersten Durchlaufes (#1). Dabei zeigte sich, dass die beschädigten Treppenpodeste aufgrund der für m gewählten Punktzahl ($m = 500$) nicht als ebene Segmente identifiziert werden konnten. Um dieses Problem zu beheben, wurde der Wert von m im zweiten Durchlauf auf 100 verringert, wodurch die Identifizierung der Oberflächen der Treppenpodeste verbessert wurde.
Zusammenfassend zeigen die in den Abbildungen 4.25 bis 4.28 dargestellten Segmentierungsergebnisse die bemerkenswerte Leistungsfähigkeit des angewandten Segmentierungsalgorithmus. Dieser konnte die planaren Decken-, Wand- und Rahmenflächen aus den Eingabepunktwolken mit hoher Präzision innerhalb kurzer Zeit extrahieren. Die gründliche Analyse der beiden Durchläufe im Segmentierungsprozess verdeutlicht die signifikante Verbesserung der Segmentierungsgenauigkeit durch manuelle Parameteranpassung im zweiten Durchlauf. Diese Erkenntnisse unterstreichen die essentielle Bedeutung der Parametereinstellungen für die Qualität der erzielten Segmentierungsergebnisse. Dabei erwies sich die automatische Parametereinstellung im ersten Durchlauf als wertvoller Ausgangspunkt für die gezielte Feinjustierung im zweiten Durchlauf.

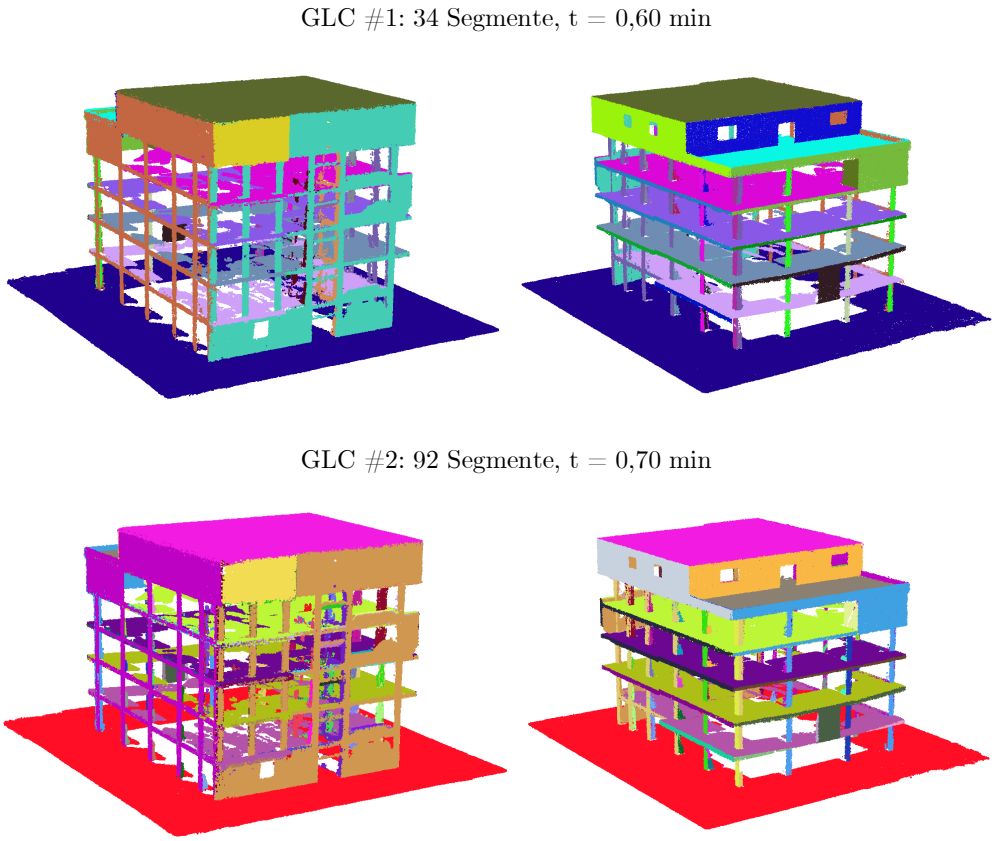


Abbildung 4.28: Ergebnisse der ebenen Punktsegmentierung für die GLC-Punktwolke nach den Durchläufen #1 und #2, wobei jede Farbe ein identifiziertes Segment repräsentiert.

4.2.2 Randpunkt-Entfernung

Nachdem die ebenen Punktmengen extrahiert wurden, erfolgte für die zugehörigen BC-Objekte der Punktsegmente eine manuelle Filterung mit Hilfe der implementierten GUI, das in Unterabschnitt 3.4.2 beschrieben ist. Dieser Prozess dauerte jeweils vier Minuten für LYN und JDS und 4,9 Minuten für GLC. Da die gefilterten Rahmensegmente mit Wänden sowohl lineare als auch ebene Punktmengen enthalten, werden die ebenen Punktmengen, d. h. die Punktdaten der Wände, aus diesen Rahmensegmenten durch Anwendung der in Unterabschnitt 3.4.2 beschriebenen RE-Methode herausgeschnitten. Diese Methode basiert auf der iterativen Entfernung von Randpunkten, wobei die Randpunkte mit Hilfe der α-*Shape*-Methode [54, 179] bestimmt werden. Die Anwendung des entsprechenden Algorithmus zur Entfernung von Randpunkten (Algorithmus 2) erfordert die Festlegung von drei Parametern: α, $numIter$ und $\bar{d}_{\max}$. In der Tabelle 4.6 werden diese Parameter näher erläutert. Die ausgewählten Parameterwerte für die Verarbeitung der Rahmensegmente mit Wänden aus den Datensätzen LYN, JDS und GLC sind in der Tabelle 4.7 aufgeführt.

Tabelle 4.6: Beschreibung der Parameter in der RE-Methode (Algorithmus 2).

Parameter	Einheit	Beschreibung	Bedingung
α	[-]	Beeinflusst die Form der extrahierten α-*Shape*-Kontur	$\alpha \in \{0, 1\} \wedge \alpha \in \mathbb{R}^+$
$numIter$	[-]	Anzahl der Iterationen der Randpunkt-Entfernung	$numIter \in \mathbb{N}^+$
$\bar{d}_{\max}$	[m]	Maximaler durchschnittlicher Punktabstand	$\bar{d}_{\max} \in \mathbb{R}^+$

Tabelle 4.7: Gewählte Parameterwerte für die Anwendung der RE-Methode auf die Rahmensegmente mit Wänden aus den Datensätzen LYN, JDS und GLC.

Parameter	Einheit	LYN		JDS		GLC	
		#1	#2	#1	#2	#1	#2
α	[-]	0,15	0,10	0,13	0,10	0,10	0,3
$numIter$	[-]	25	22	30	22	20	15
$\bar{d}_{\max}$	[m]	0,3	0,3	0,1	0,5	0,3	0,3

Die Evaluierung der Methode erfolgt durch den Vergleich der resultierenden Segmentierungen mit einem manuell erstellten *Ground Truth* für die Rahmensegmente mit Wänden. Wie im vorherigen Schritt wurden zwei Durchläufe (#1 und #2) zur Extraktion der Wände aus den Rahmensegmenten durchgeführt, wobei die Parameterwerte im zweiten Durchlauf angepasst wurden.
In den Abbildungen 4.29 bis 4.31 sind die *Ground-Truth*-Segmentierungen und die RE-Ergebnisse für die Rahmensegmente mit Wänden aus der LYN-, JDS- und GLC-Punktwolke dargestellt. Der visuelle Vergleich der Segmentierungsergebnisse des LYN-Datensatzes mit den manuell erstellten *Ground-Truth*-Segmentierungen (Abbildung 4.29) zeigt die präzise Extraktion der Wandpunktdaten innerhalb der Rahmensegmente sowohl im ersten als auch im zweiten Durchlauf. Dabei konnte eine hohe Genauigkeit erreicht werden. Die benötigte Berechnungszeit betrug 1,3 Minuten für den ersten Durchlauf und 1,23 Minuten für den zweiten Durchlauf. Die visuellen Abweichungen zwischen den *Ground-Truth*-Daten und den Ergebnissen aus den Durchläufen #1 und #2 in Abbildung 4.29 erweisen sich als gering bis unauffällig. Dies unterstreicht die Fähigkeit der angewandten Methode, konsistente und verlässliche Ergebnisse in diesem Datensatz zu erzielen.

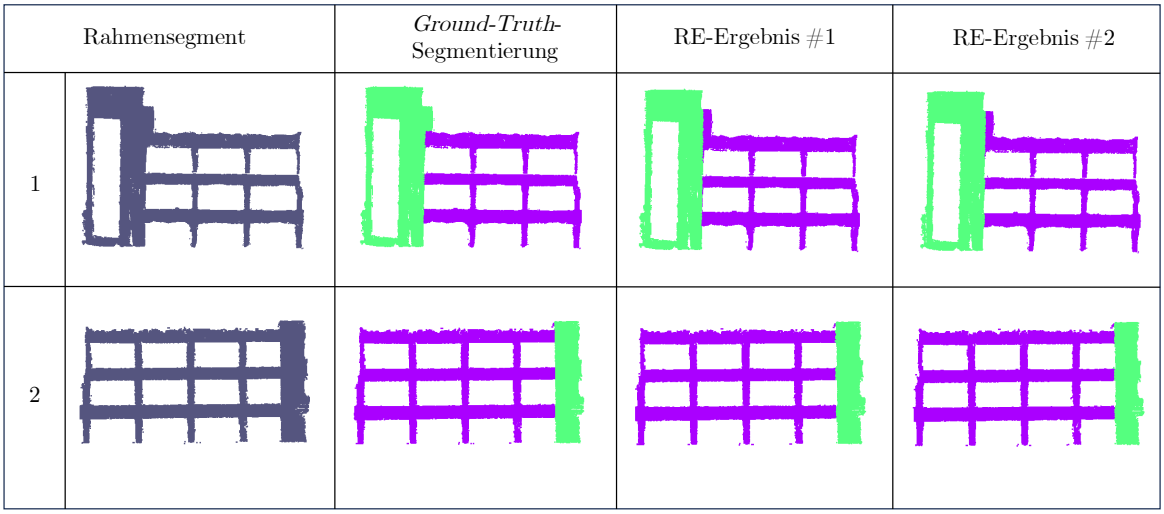


Abbildung 4.29: Darstellung der *Ground-Truth*-Segmentierung und der RE-Ergebnisse für die Rahmensegmente mit Wänden aus der LYN-Punktwolke. Die Farbe Lila steht für die verbleibenden Punktdaten des Rahmensegments und die Farbe Grün für die ausgeschnittenen Wände.

Aus den Ergebnissen der ebenen Punktsegmentierung der JDS-Punktwolke wurden gezielt fünf Rahmensegmente ausgewählt, um sie mit der RE-Methode zu verarbeiten (Abbildung 4.30 links). Diese Rahmensegmente enthalten sowohl Punktdaten von teilweise zerstörten als auch von intakten Mauerwerksausfachungen. Die entsprechenden Segmentierungsergebnisse und manuell erstellten *Ground-Truth*-Daten sind in der Abbildung 4.30 dargestellt. Die Ergebnisdarstellung in Abbildung 4.30 verdeutlicht die effektive Wirkung der Parameteranpassung während des zweiten Durchlaufes, was zu einer genaueren Extraktion der Wände innerhalb der Rahmensegmente

führte. Darüber hinaus wird anhand der Resultate für die Rahmensegmente 2 und 4 (Abbildung 4.30) deutlich, dass eine exakte Segmentierung von teilweise beschädigten Wänden mit unregelmäßigen Geometrien eine anspruchsvolle Aufgabe für die RE-Methode darstellt. Dies ist darauf zurückzuführen, dass die Punktdaten der teilweise zerstörten Wände nicht vollständig zwischen Stützen und Trägern liegen, sodass sie im Zuge der Randpunkt-Entfernung mit der α-*Shape*-Methode beeinflusst werden. Dieser Effekt führt dazu, dass die Form dieser Wände mit der *crop2D*-Funktion [40] nicht exakt ausgeschnitten werden kann.

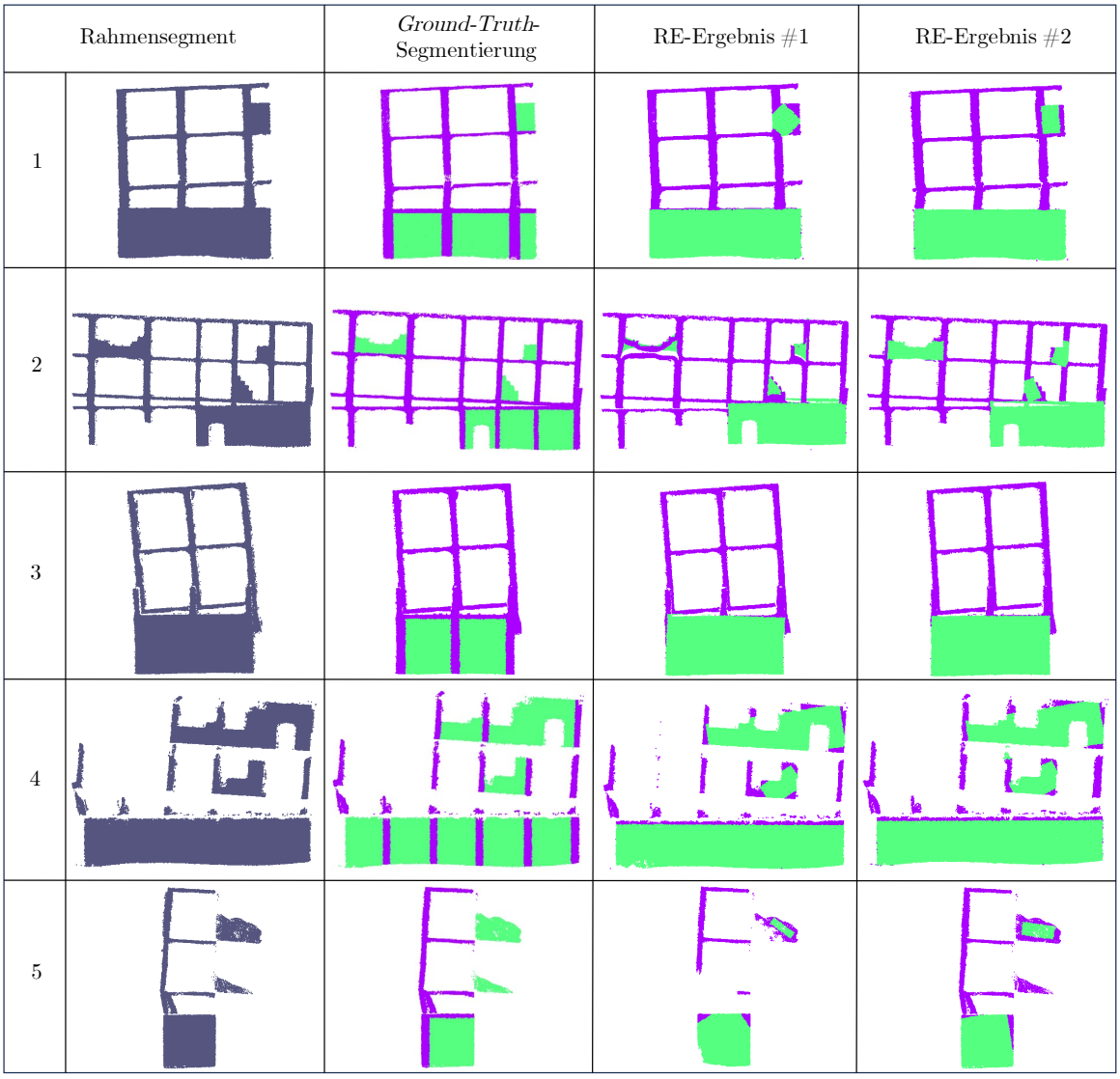


Abbildung 4.30: Darstellung der *Ground-Truth*-Segmentierung und der RE-Ergebnisse für die Rahmensegmente mit Wänden aus der JDS-Punktwolke. Die Farbe Lila steht für die verbleibenden Punktdaten des Rahmensegments und die Farbe Grün für die ausgeschnittenen Wände.

Des Weiteren ist aus der Abbildung 4.30 ersichtlich, dass die Punktdaten der Stützen zwischen den Mauerwerkswänden bei der Segmentierung der Wände größtenteils nicht berücksichtigt werden konnten. Der Grund dafür ist, dass die α-*Shape*-Methode [54, 179] nur Randpunkte entfernen kann, während Punktdaten innerhalb einer Punktmenge nicht berücksichtigt werden können. Aus diesem Grund wird bei der RE-Methode die Punktsegmentierung von Wänden, die sich über mehr als ein Feld erstrecken, als vollständige Wand durchgeführt. Diese Erkenntnisse verdeutlichen die

Grenzen der RE-Methode bei der exakten Segmentierung von teilweise zerstörten Wänden mit unregelmäßigen Geometrien sowie die Herausforderung, die sich aus der Behandlung von Punktdaten zwischen Stützen und Mauerwerkswänden ergibt. Zusätzlich ist anzumerken, dass die Verarbeitungszeit für diesen Prozess im ersten Durchlauf 7,3 Minuten und im zweiten Durchlauf 7,07 Minuten betrug.
Die Abbildung 4.31 zeigt die Ergebnisse der Randpunkt-Entfernung für die ausgewählten Rahmensegmente der GLC-Punktwolke. Anhand dieser Abbildung wird deutlich, dass die extrahierten Punktdaten der Wände in den Rahmensegmenten 1 bis 3 einerseits auf lokaler Ebene korrekt erfasst wurden, andererseits auch Anzeichen von ungenauer Segmentierung im Vergleich zu den *Ground-Truth*-Daten aufweisen. Die fehlerhaften Segmentierungen der Wände lassen sich ähnlich wie im JDS-Datensatz auf die iterative Randpunkt-Entfernung mit Hilfe von α-*Shapes* zurückführen. In diesem Prozess werden die Randpunkte der bestehenden Wände im Verlauf der Iterationen entfernt, welche bei der Extraktion der verbleibenden Punktdaten mittels der *crop2D*-Funktion nicht berücksichtigt werden. Dies kann zur Folge haben, dass entweder zu wenige oder zu viele Punktdaten dem Rahmensegment als Teil des Wandsystems zugeordnet werden. Aus diesem Grund konnten z. B. die Punktdaten der Stützen zwischen den Mauerwerkswänden in den Rahmensegmenten 1 und 2, wie im JDS-Datensatz, bei der Segmentierung größtenteils nicht berücksichtigt werden.

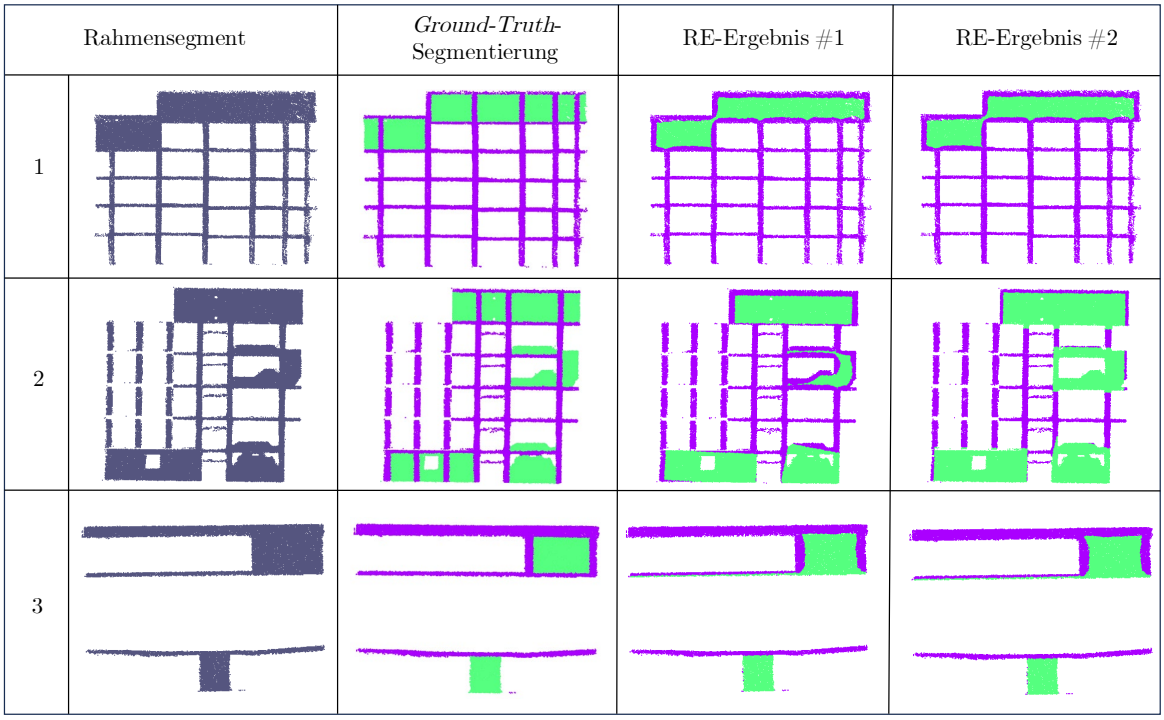


Abbildung 4.31: Darstellung der *Ground-Truth*-Segmentierung und der RE-Ergebnisse für die Rahmensegmente mit Wänden aus der GLC-Punktwolke. Die Farbe Lila steht für die verbleibenden Punktdaten des Rahmensegments und die Farbe Grün für die ausgeschnittenen Wände.

In Bezug auf die Parameteranpassung zeigt sich, dass aufgrund der geometrischen Einfachheit der Wände in den Rahmensegmenten 1 und 3 die Anpassung der Parameterwerte in diesem Datensatz nur im Fall des Rahmensystems 2 zu einer erkennbaren Verbesserung der Segmentierung geführt hat. Ein ähnliches Verhalten wurde auch bei den Rahmensegmenten der JDS-Punktwolke beobachtet. Dies unterstreicht die Tatsache, dass die Anpassung der Parameter in der RE-Methode vor allem bei Wänden mit komplexen geometrischen Formen zu einer Verbesserung der Ergeb-

nisse führen kann. Die Berechnungszeit für den ersten Durchlauf betrug 1,63 Minuten und für den zweiten Durchlauf 1,73 Minuten. Die längere Verarbeitungszeit der Rahmensegmente von JDS im Vergleich zu LYN und GLC lässt sich auf die größere Anzahl verarbeiteter Segmente und die damit einhergehende Datengröße zurückführen.

4.2.3 Linear angeordnete Punktmengen

Nachdem die Wände aus den Rahmensegmenten entfernt wurden, konzentrieren wir uns auf die Segmentierung der linear angeordneten Punktmengen aus den Rahmen. Ziel ist es, die Punktdaten der Stützen und Träger in einem Rahmen vollständig zu erfassen und voneinander zu trennen, damit sie in den nachfolgenden Schritten semantisch klassifiziert werden können. Zu diesem Zweck wird der vom Autor dieser Arbeit entwickelte Algorithmus (Algorithmus 3) verwendet, bei dem die linearen Punktmengen eines Rahmens mittels eines iterativen Ansatzes auf Basis der RANSAC-Methode in $\mathbb{R}^2$ segmentiert werden (Unterabschnitt 3.5.2).
In der Tabelle 4.8 ist eine Beschreibung der Parameter aufgelistet, die zur Anwendung des Algorithmus 3 benötigt werden. Die Tabelle 4.9 fasst die gewählten Werte dieser Parameter für die einzelnen Datensätze und ihre Durchläufe für die Evaluierung zusammen. Um bei der Suche nach Liniensegmenten Träger und Stützen mit unterschiedlichen Breiten zu berücksichtigen, wurden die Parameter ϵ, d_{mean} und d_{max} jeweils zweifach definiert, die durch die Indizes a und b gekennzeichnet sind. Der Wert des Parameters ϵ wurde auf der Grundlage der vorhandenen Abmessungen der Rahmenkomponenten definiert.

Tabelle 4.8: Beschreibung der Parameter in der Methode zur Segmentierung von linienförmigen Punktmengen (Algorithmus 3).

Parameter	Einheit	Beschreibung	Bedingung
n	[-]	Mindestpunktanzahl zur Identifizierung einer Linie	$n \in \mathbb{N}^+$
k	[-]	Anzahl der RANSAC-Iterationen	$k \in \mathbb{N}^+$
ϵ	[m]	Maximaler orthogonaler Abstand von der Linie	$\epsilon_b \in \mathbb{R}^+$
d_{mean}	[m]	Durchschnittlicher Punktabstand	$d_{\text{mean}} \in \mathbb{R}^+$
d_{max}	[m]	Maximaler Punktabstand	$d_{\text{max}} \in \mathbb{R}^+$
a_{max}	[-]	Maximale Anzahl der Iterationen für die Liniensuche	$a_{\text{max}} \in \mathbb{N}^+$
t_{min}	[m]	Mindestdicke von parallelen Liniensegmenten	$t_{\text{min}} \in \mathbb{R}^+$
t_d	[m]	Standarddicke von parallelen Liniensegmenten	$t_d \in \mathbb{R}^+$

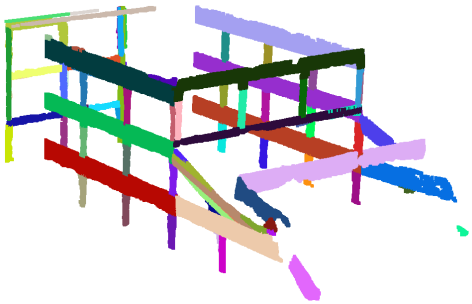
Tabelle 4.9: Gewählte Parameterwerte für die lineare Punktsegmentierung der Rahmensegmente der 3D-Punktwolken der Datensätze LYN, JDS und GLC.

Parameter	Einheit	LYN		JDS		GLC	
		#1	#2	#1	#2	#1	#2
n	[-]	2	2	2	2	2	2
k	[-]	1000	1000	1000	1000	1000	1000
ϵ_a	[m]	0,6	0.55	0.25	0.25	0.3	0.3
$d_{a,\text{mean}}$	[m]	0.3	0.1	0.1	0.1	0.2	0.2
$d_{a,\text{max}}$	[m]	0.6	0.4	0.4	0.4	0.5	0.5
ϵ_b	[m]	0.2	0.17	0.25	0.25	0.3	0.3
$d_{b,\text{mean}}$	[m]	0.1	0.1	0.1	0.1	0.2	0.2
$d_{b,\text{max}}$	[m]	0.4	0.4	0.4	0.4	0.5	0.5
a_{max}	[-]	10	10	10	10	10	10
t_{min}	[m]	0.1	0.1	0.1	0.1	0.1	0.1
t_d	[m]	0.25	0.25	0.25	0.25	0.25	0.25

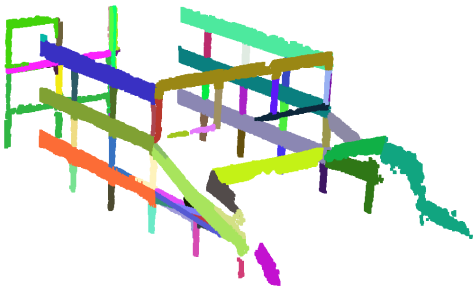
Weiterhin ist es wichtig zu erwähnen, dass die Parameter k und $a_{\max}$, welche die Anzahl der Iterationen für die Modellsuche festlegen, in allen Durchläufen konstant gehalten wurden, um eine Vergleichbarkeit hinsichtlich der benötigten Rechenzeit zu gewährleisten. Die Abbildung 4.32 zeigt eine zusammenfassende Darstellung der Ergebnisse der linearen Punktsegmentierung für die Rahmensegmente der LYN-, JDS- und GLC-Punktwolke. Wie aus dieser Abbildung ersichtlich ist, wurden die Punktdaten der Träger und Stützen aus den Rahmensegmenten der LYN-Punktwolke im ersten Durchgang (#1) mit hoher Genauigkeit segmentiert. Insgesamt wurden dabei 79 Liniensegmente innerhalb von 2,5 Minuten gefunden. Die Anpassung der Parameter ϵ_a und ϵ_b führte im zweiten Durchgang (#2) zu geringfügig weniger Liniensegmenten, wobei keine wesentlichen Unterschiede in der Genauigkeit der identifizierten Segmente festgestellt werden können. Im teilweise eingestürzten Bereich eines LYN-Rahmensegments, der in Abbildung 4.33 (oben) dargestellt ist, wurde festgestellt, dass sowohl im ersten als auch im zweiten Durchgang eine Übersegmentierung auftrat. Dies resultierte aus der Tatsache, dass die linearen Punktdaten in den gekennzeichneten Bereichen der Abbildung 4.33 (oben) keine rechteckige Form aufweisen, wodurch die Kriterien der Formüberprüfung mit $d_{a,\text{mean}}$ und $d_{a,\max}$ nicht erfüllt wurden. Nur bei den mit ϵ_b identifizierten Liniensegmenten konnten diese Kriterien erfüllt werden, was in der Folge zu einer Übersegmentierung führte.
Die Abbildung 4.32 präsentiert die Ergebnisse der Liniensegmentierung für die Rahmensegmente der JDS- und GLC-Punktwolke. Für die JDS-Punktwolke konnten im ersten Durchlauf (#1) insgesamt 132 Liniensegmente innerhalb einer Zeitspanne von 2,53 Minuten identifiziert werden. Im zweiten Durchlauf (#2) wurden innerhalb von 2,05 Minuten 140 Liniensegmente erkannt. Es ist wichtig zu erwähnen, dass im zweiten Durchlauf keine manuelle Anpassungen der Parameter vorgenommen wurde, da die in Durchlauf #1 gewählten Parameter bereits zu zufriedenstellenden Ergebnissen führten. Für die GLC-Punktwolke wurden im ersten Durchlauf (#1) 126 Liniensegmente in einer Zeit von 1,36 Minuten gefunden. Im zweiten Durchlauf (#2) konnten 131 Liniensegmente innerhalb von 1,22 Minuten identifiziert werden. Auch hier wurden die Parameterwerte in Durchlauf #2 aus dem oben genannten Grund nicht verändert. Angesichts der Tatsache, dass in den manuell erstellten Modellen von JDS und GLC die Höhe der Träger und die Breite der Stützen aus Vereinfachungsgründen gleich gewählt wurden, wurden die Parameter ϵ_a und ϵ_b für die Segmentierung der linearen Punktmengen entsprechend mit dem gleichen Abstand gewählt. Es ist wichtig zu beachten, dass die Berücksichtigung mehrerer unterschiedlicher Abmessungen von Stützen und Trägern im Prozess der linearen Punktsegmentierung nicht implementiert ist.
Die Ergebnisse der Liniensegmentierung zeigen eine präzise und umfassende Segmentierung der linearen Punktdaten in den meisten Rahmensegmenten, insbesondere in den Abschnitten, die nahezu vollständige Punktdaten der Träger und Stützen enthalten. Andererseits gestaltet sich die Identifizierung von Liniensegmenten in Bereichen, in denen die Punktdaten der Träger und Stützen aufgrund von Verdeckungen und den Anwendungsgrenzen der Photogrammetrie unvollständig sind, als herausfordernd. Weiterhin ist festzustellen, dass der entwickelte Ansatz je nach Größe der Punktmenge entweder zu einer kontinuierlichen Segmentierung der linearen Punktdaten eines Trägers oder einer Stütze führt. Dies hat zur Folge, dass die Punktdaten der Stützen nicht konsequent nach Geschossen getrennt werden können, wie es insbesondere in den Ergebnissen der JDS- und GLC-Punktwolke deutlich wird (Abbildung 4.33). In Anbetracht der Tatsache, dass die BIM-Methode eine geschossweise Informationsanreicherung der Bauteile vorsieht, stellt dies eine Einschränkung des entwickelten Ansatzes dar.
In der Abbildung 4.33 ist beispielhaft für den JDS- und GLC-Datensatz je ein Rahmensegment dargestellt, aus dem im vorherigen Schritt die Punktdaten der Wände iterativ entfernt wurden. Die ungenaue Entfernung der Wandpunkte im ersten Durchlauf führte zu einer linearen Segmentierung der verbleibenden Punktdaten der Wände. Da diese Segmente weder Träger noch Stützen darstellen, können sie im nachfolgenden Arbeitsschritt nicht zielführend klassifiziert

werden. Dieser Mangel zeigte im zweiten Durchlauf eine geringere Ausprägung, da im vorherigen Arbeitsschritt (Unterabschnitt 4.2.2) die Wandpunkte der Rahmensegmente mittels der RE-Methode präziser entfernt wurden als im ersten Durchlauf. Die Optimierung im Prozess der Randpunkt-Entfernung trug dazu bei, die Genauigkeit der Segmentierung zu verbessern und die Wahrscheinlichkeit zu erhöhen, dass es sich bei den identifizierten Liniensegmenten tatsächlich um Träger und Stützen handelt.

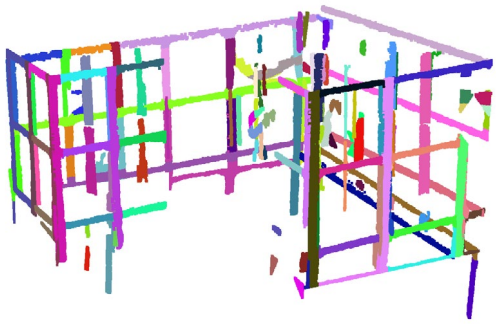
LYN #1: 79 Segmente, t = 2,5 min



LYN #2: 70 Segmente, t = 2,1 min

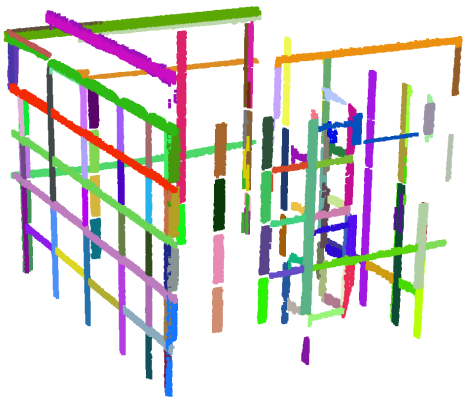


JDS #1: 132 Segmente, t = 2,53 min



JDS #2: 140 Segmente, t = 2,05 min

GLC #1: 126 Segmente, t = 1,36 min



GLC #2: 131 Segmente, t = 1,22 min

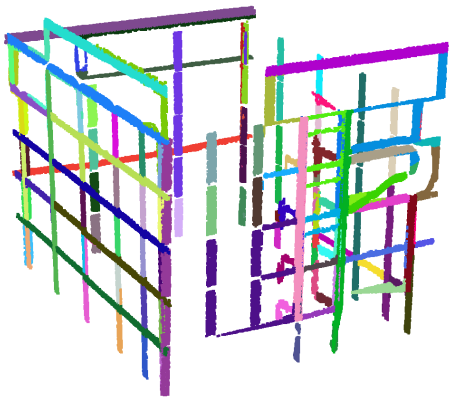


Abbildung 4.32: Ergebnisse der linearen Punktsegmentierung für die Rahmensegmente der LYN-, JDS- und GLC-Punktwolke. Jede Farbe repräsentiert ein identifiziertes Liniensegment.

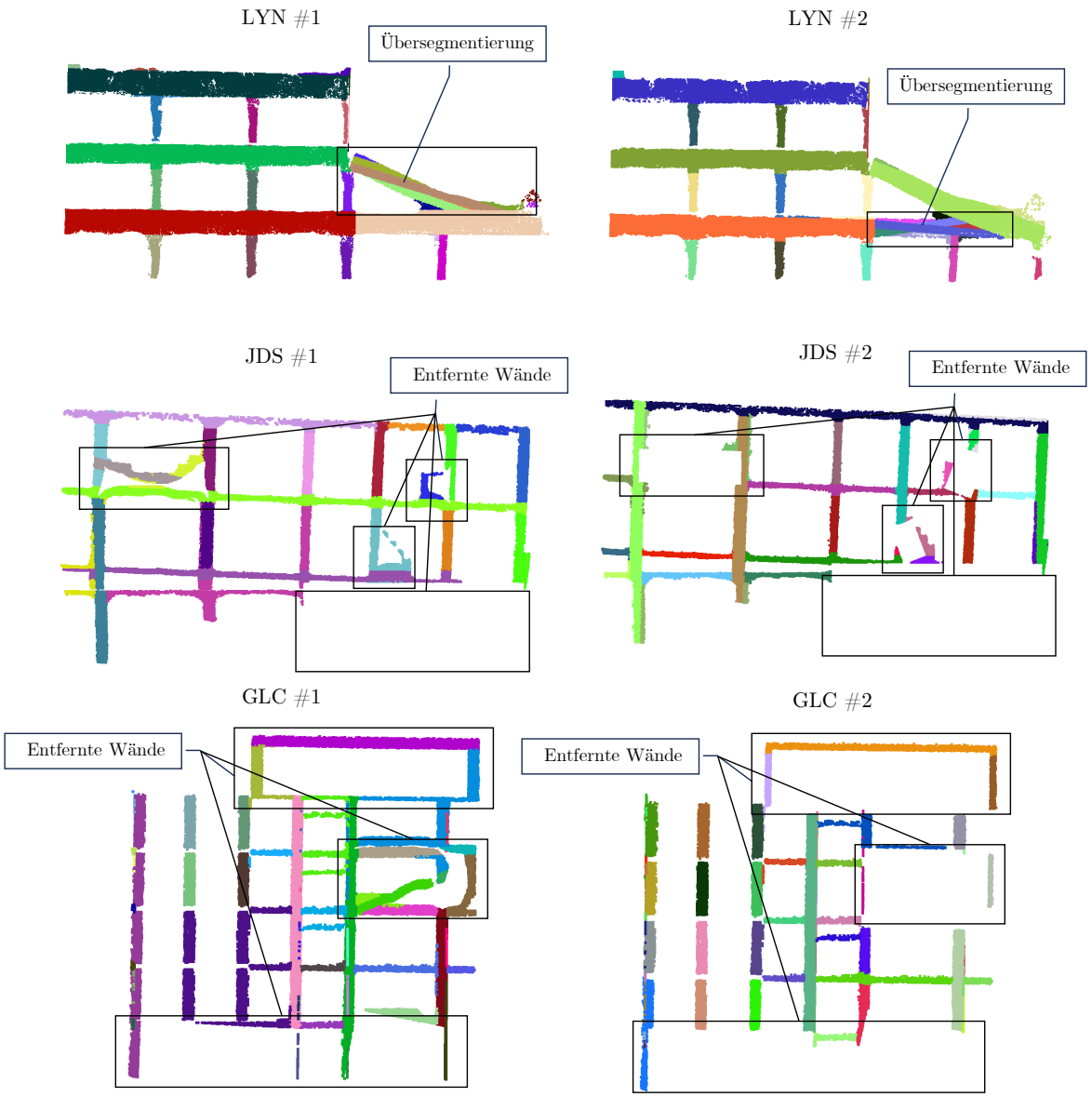


Abbildung 4.33: Fehlerhafte Segmentierung der linearen Punktdaten aus den Rahmensegmenten der LYN-, JDS- und GLC-Punktwolke.

4.3 Objektklassifizierung

Nachdem die Datensätze in ebene und lineare Punktmengen segmentiert wurden, werden sie strukturellen oder nicht-strukturellen Objektklassen zugeordnet. Zu diesem Zweck wird die in Abschnitt 3.5 vorgestellte geometriebasierte Methode der Objektklassifizierung verwendet. Die festgelegten geometrischen Parameter zur Anwendung der automatischen Objektklassifizierung sind in der Tabelle 4.10 zusammengefasst. Um die Konsistenz und Vergleichbarkeit der Ergebnisse zwischen den Datensätzen zu gewährleisten, wurden die in Tabelle 4.10 angegebenen Parameterwerte für die Objektklassifizierung der segmentierten Punktdaten nicht variiert. Die klassifizierten Punktmengen der LYN-, JDS- und GLC-Punktwolke sind in den Abbildungen 4.34 bis 4.36 dargestellt, wobei jede Objektklasse durch eine entsprechende Farbkodierung gekennzeichnet ist.

Tabelle 4.10: Beschreibung und Definition der geometrischen Parameter für die Objektklassifizierung (Algorithmus 4 und 5).

Parameter	Einheit	Beschreibung	Gewählt
$z_{\text{min,slab}}$	[m]	Minimale Höhe einer Deckenplatte über GOK	2,3
$z_{\text{min,base}}$	[m]	Minimale Höhe einer Bodenplatte über GOK	1,0
$H_{\text{min,wall}}$	[m]	Mindesthöhe einer Wand	1,0
$L_{\text{min,wall}}$	[m]	Mindestlänge einer Wand	$L_{\text{min,wall}} > 4 \cdot t_i$
$H_{\text{min,beam}}$	[m]	Mindesthöhe eines Trägers	0,2
$H_{\text{max,beam}}$	[m]	Maximale Höhe eines Trägers	1,2
$L_{\text{min,beam}}$	[m]	Mindestlänge eines Trägers	$L_{\text{min,beam}} > 3 \cdot H_{i,z}$
$H_{\text{min,column}}$	[m]	Mindestlänge einer Stütze	$H_{\text{min,column}} > 3 \cdot L_{i,y}$
$L_{\text{min,column}}$	[m]	Mindestbreite einer Stütze	0,1
$L_{\text{max,column}}$	[m]	Maximale Breite einer Stütze	$H_{\text{min,beam}} \leq 4 \cdot t_i$
$H_{\text{min,Slab\textbackslash Wall}}$	[m]	Mindestlänge einer Decke\Wand	3,0
$L_{\text{min,Slab\textbackslash Wall}}$	[m]	Mindestbreite einer Decke\Wand	3,0
d_{mean}	[m]	Durchschnittlicher Punktabstand	0,1
d_{max}	[m]	Maximaler Punktabstand	0,6
t_{min}	[m]	Minimale Komponentendicke	0,1
t_{d}	[m]	Standardmäßige Komponentendicke	0,25

Um die Methode der Objektklassifizierung objektiv zu bewerten, werden verschiedene quantitative Kennzahlen verwendet, die im Folgenden stichpunktartig erläutert werden:

- SOLL: Hierbei handelt es sich um die erwartete Gesamtanzahl von Objekten, die gemäß den *Ground-Truth*-Daten einer strukturellen oder nicht-strukturellen Objektklasse zugeordnet werden sollten.
- IST: Diese Kennzahl gibt die tatsächliche Anzahl der Objekte an, die nach Anwendung der Klassifizierungsmethode einer Objektklasse zugeordnet wurden.
- TP: TP (*True Positive*) gibt an, wie viele Objekte in IST mit der richtigen Objektklasse identifiziert wurden.
- FP: FP (*False Positive*) gibt dagegen an, wie viele Objekte in IST einer falschen Objektklasse zugeordnet wurden.

Um diese Kennzahlen in den richtigen Kontext zu setzen, wurden auch die folgenden Prozentsatzberechnungen vorgenommen:

- TP [%]: Diese Kennzahl gibt an, welcher Prozentsatz der zugeordneten Objektklassen in IST korrekt identifiziert wurde. Folglich beschreibt dieser Wert die Leistungsfähigkeit der Objektklassifizierungsmethode und wird wie folgt ermittelt:

$$\text{TP } [\%] = \frac{\text{TP}}{\text{IST}} \times 100. \tag{4.1}$$

- FP [%]: Dies ist der Prozentsatz der falsch zugewiesenen Objektklassen in IST, der sich aus der Differenz zwischen 100% und TP [%] ergibt:

$$\text{FP } [\%] = 100 - \text{TP } [\%]. \tag{4.2}$$

- Ω [%]: Ω repräsentiert die Gesamtgenauigkeit der Objektklassifikation, d. h. den Prozentsatz der korrekt zugeordneten Objektklassen im Vergleich zum *Ground Truth*. Dieser Parameter wird wie folgt berechnet:

$$\Omega \ [\%] = \begin{cases} \frac{\text{TP}}{\text{SOLL}} \times 100 & \text{SOLL} \geq \text{IST} \\ \frac{\text{TP}}{\text{IST}} \times 100 & \text{SOLL} < \text{IST} \end{cases}. \tag{4.3}$$

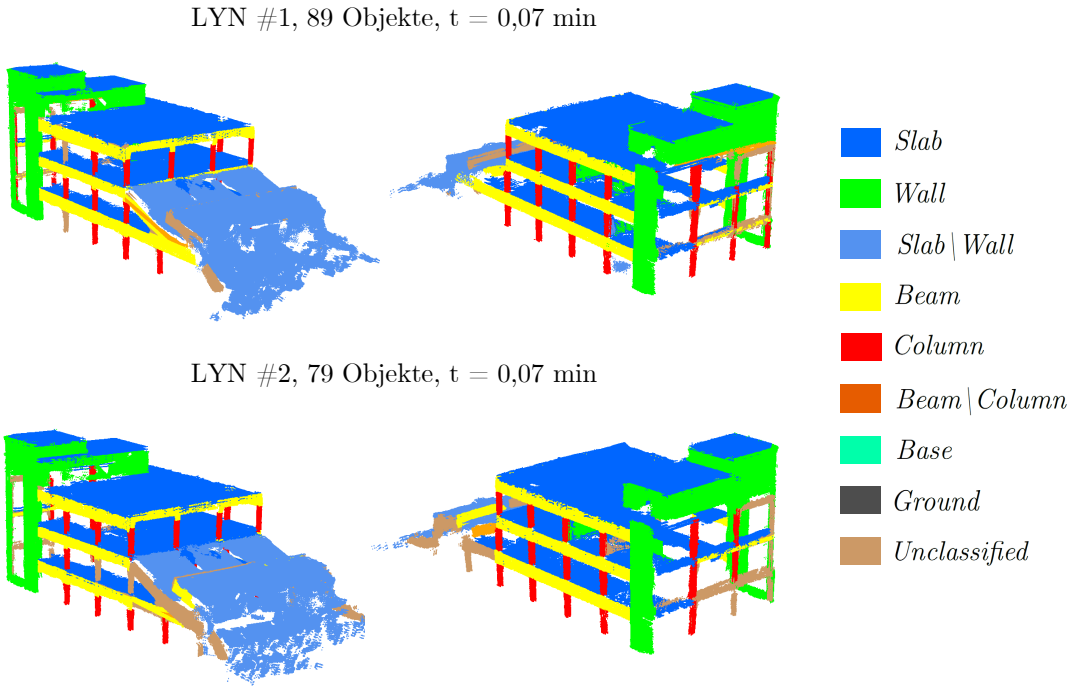


Abbildung 4.34: Darstellung der Ergebnisse der Objektklassifizierung für die LYN-Punktwolke.

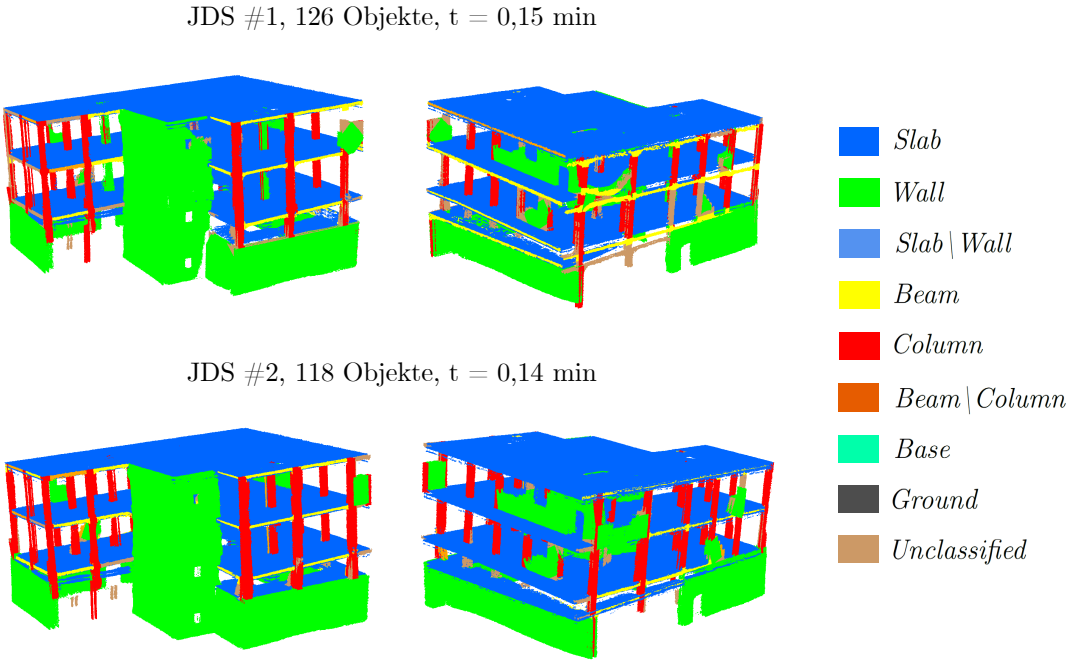


Abbildung 4.35: Darstellung der Ergebnisse der Objektklassifizierung für die JDS-Punktwolke.

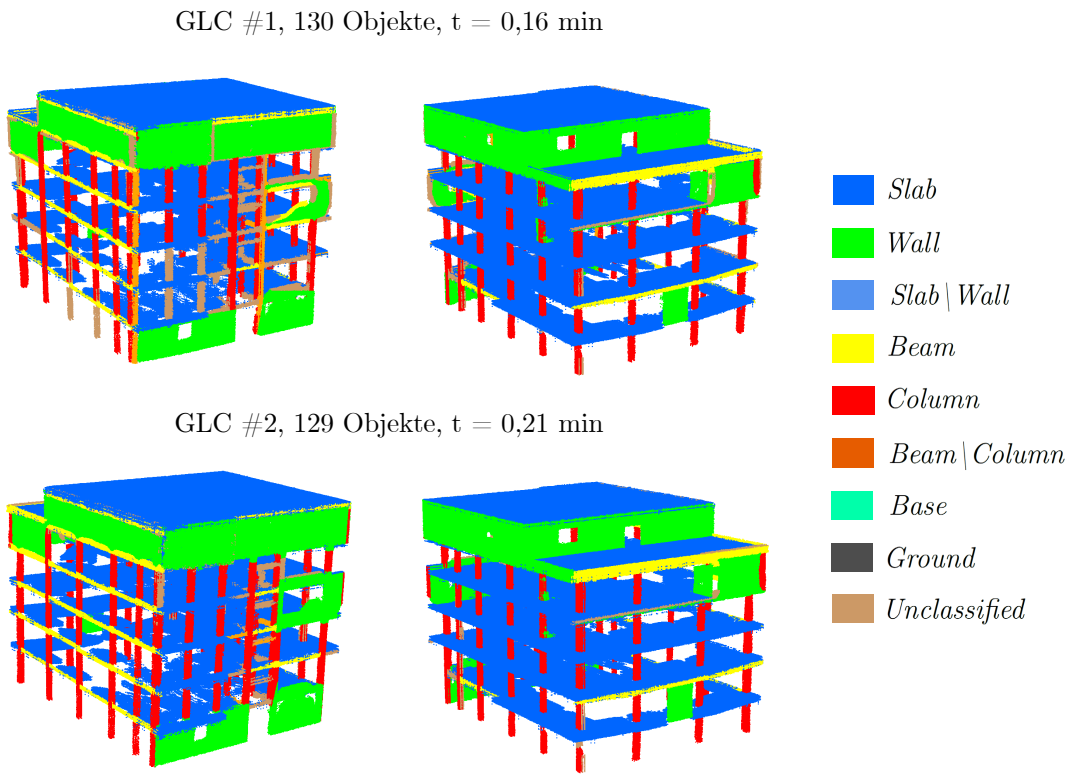


Abbildung 4.36: Darstellung der Ergebnisse der Objektklassifizierung für die GLC-Punktwolke.

Die oben genannten Kennzahlen zur Bewertung der Methode der Objektklassifizierung sind in den Tabellen 4.11 bis 4.13 für die jeweilige Objektklasse der LYN-, JDS- und GLC-Punktwolke übersichtlich aufgeführt. Zusätzlich zu diesen Kennzahlen finden sich dort auch die Gesamtsummen für SOLL, IST, TP und FP sowie der arithmetische Mittelwert μ für TP [%], FP [%] und Ω [%]. Diese Werte dienen zur quantitativen Darstellung der Gesamtgenauigkeit der Objektklassifizierung.
Aus der Tabelle 4.11 ist zu entnehmen, dass die durchschnittliche Gesamtgenauigkeit Ω für die LYN-Punktwolke im ersten Durchlauf (#1) bei 85% liegt. Von den insgesamt 89 Punktsegmenten konnten mit der Klassifizierungsmethode beachtliche 94% der richtigen Objektklasse zugeordnet werden. Im zweiten Durchlauf (#2) konnte die durchschnittliche Gesamtgenauigkeit der Objektklassifikation nicht erhöht werden (78%), da mehrere Punktdaten von Stützen aufgrund ihrer ungenauen Segmentform der Klasse *Unclassified* zugewiesen wurden. Um dies zu veranschaulichen, ist in der Abbildung 4.37 ein Stützensegment aus der LYN-Punktwolke dargestellt, dem die Objektklasse *Unclassified* zugewiesen wurde. Die *Bounding Box* dieses Segments, die zur Ableitung der Abmessungen verwendet wird, zeigt, dass sie weder den definierten Abmessungen einer Stütze noch eines Trägers entspricht. Folglich wurde das dargestellte Segment der Objektklasse *Unclassified* zugewiesen. Die Abbildung 4.38 (oben links) veranschaulicht, dass die Gesamtgenauigkeit Ω für die verschiedenen Objektklassen der LYN-Punktwolke generell sehr hoch ist. Die Identifikation von Decken und Wänden entspricht dem *Ground Truth* mit einer Genauigkeit von 100%. Eine Ausnahme bildet die Objektklasse *Beam | Column*, bei der in beiden Durchläufen eine Gesamtgenauigkeit von 50% erreicht wurde. Es ist jedoch zu beachten, dass das zugrunde liegende SOLL für diese Klasse nur zwei Segmente umfasst, wodurch eine umfassende Beurteilung der Objektklassifikation in diesem Fall eingeschränkt ist. Die Klassifizierung der Punktsegmente der LYN-Punktwolke dauerte insgesamt 0,07 Minuten.

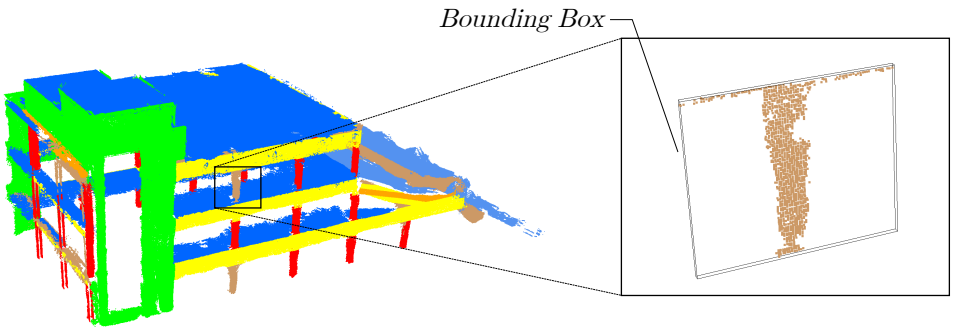


Abbildung 4.37: Veranschaulichung der zugewiesenen Objektklasse *Unclassified* für ein Stützsegment der LYN-Punktwolke.

Die Tabelle 4.12 zeigt, dass die durchschnittliche Gesamtgenauigkeit Ω für die JDS-Punktwolke im ersten Durchlauf bei 77% liegt und im zweiten Durchlauf auf 86% gesteigert werden konnte. Die Steigerung der Gesamtgenauigkeit auf 86% im zweiten Durchlauf ist hauptsächlich auf die verbesserte Objektklassifizierung der Träger zurückzuführen, deren prozentualer TP-Anteil von 69% auf 100% erhöht wurde. Die in IST enthaltenen Objekte wurden im Durchlauf #1 zu 89% und im Durchlauf #2 zu 99% richtig klassifiziert, was auf eine hohe Leistungsfähigkeit der Klassifizierungsmethode hinweist. Die Abbildung 4.38 (oben rechts) illustriert, dass die Decken und Wände in beiden Durchläufen nahezu vollständig identifiziert werden konnten. Die Identifizierung von 76% der Träger des Gebäudes kann als angemessene Genauigkeit betrachtet werden. Andererseits konnte für die Objektklasse *Column* mit 60% und 67% keine hohe Genauigkeit erzielt werden. Es ist jedoch wichtig zu beachten, dass der Mittelwert von Ω vom erwarteten SOLL-Wert des Bauteils abhängt, wodurch die Genauigkeit der vorherigen Arbeitsschritte diesen Wert beeinflusst. In Unterabschnitt 4.2.2 wurde erläutert, dass die RE-Methode nicht in der Lage ist, Punktdaten von Stützen, die sich zwischen Mauerwerkswänden befinden, vollständig zu berücksichtigen. Aufgrund dieser Anwendungsgrenze war es in 14 Fällen bei der JDS-Punktwolke nicht möglich, diese als Stützen zu klassifizieren. Infolgedessen fällt die Gesamtgenauigkeit Ω für die Objektklasse *Column* vergleichsweise gering aus. Werden diese 14 Stützen nicht berücksichtigt, liegen die Ω-Werte für *Column* in diesem Fall bei 80% und 89%. Die Objektklassifizierung der 126 Segmente der JDS-Punktwolke nahm in beiden Durchläufen rund 0,15 Minuten in Anspruch.
Aus der Tabelle 4.13 geht hervor, dass durchschnittlich 90% der identifizierten Segmente in der GLC-Punktwolke (IST) im ersten Durchlauf und 92% im zweiten Durchlauf der jeweiligen Objektklasse korrekt zugeordnet werden konnten. Dies unterstreicht die Tatsache, dass die Objektklassifizierungsmethode in allen drei Gebäuden eine durchschnittliche Genauigkeit von mehr als 90% bei den TPs erzielen konnte, was auf ihre hohe Leistungsfähigkeit hinweist. Die durchschnittliche Gesamtgenauigkeit Ω für die GLC-Punktwolke beträgt im ersten Durchlauf 80% und 88% im zweiten Durchlauf. Bemerkenswert ist, dass wie bei den LYN- und JDS-Punktsegmenten in beiden Durchläufen nahezu alle Decken und Wände vollständig identifiziert wurden (Abbildung 4.38 (unten)). Dies lässt darauf schließen, dass die angewandte Klassifizierungsmethode insbesondere bei der Identifizierung von Decken und Wänden in verschiedenen Datensätzen eine hohe Zuverlässigkeit aufweist. Die Ursache für die vergleichsweise niedrige Genauigkeit bei der Klassifizierung der Objektklasse *Column* liegt, wie im Fall der JDS-Punktwolke, darin begründet, dass sich 18 Stützen zwischen den Mauerwerkswänden befinden und daher mit Hilfe der RE-Methode nicht vollständig von den Wandpunktdaten getrennt werden konnten. Wären diese 18 Stützen aus der Berechnung der Gesamtgenauigkeit ausgeschlossen, läge der Ω-Wert bei der Klassifizierung der Stützen bei 78% in Durchlauf #1 und 86% in Durchlauf #2.

Tabelle 4.11: Ergebnisse der Objektklassifizierung für die LYN-Punktwolke.

Objektklasse	SOLL	IST [-]		TP [-]		FP [-]		TP [%]		FP [%]		Ω [%]	
	[-]	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2
Slab	5	5	5	5	5	0	0	100	100	0	0	100	100
Wall	11	11	9	11	9	0	0	100	100	0	0	100	82
Slab \ Wall	5	6	4	5	4	1	0	83	100	17	0	83	80
Beam	12	12	10	10	9	2	1	83	90	17	10	83	75
Column	33	30	26	30	26	0	0	100	100	0	0	91	79
Beam \ Column	2	1	1	1	1	0	0	100	100	0	0	50	50
Base	0	0	0	-	-	-	-	-	-	-	-	-	-
Ground	0	0	0	-	-	-	-	-	-	-	-	-	-
Unclassified	-	24	24	-	-	-	-	-	-	-	-	-	-
	$\sum$ [-]							μ [%]					
	68	89	79	62	54	3	1	**94**	**98**	**6**	**2**	**85**	**78**

Tabelle 4.12: Ergebnisse der Objektklassifizierung für die JDS-Punktwolke.

Objektklasse	SOLL	IST [-]		TP [-]		FP [-]		TP [%]		FP [%]		Ω [%]	
	[-]	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2
Slab	4	4	4	4	4	0	0	100	100	0	0	100	100
Wall	14	16	14	15	14	2	0	94	100	6	0	94	100
Slab \ Wall	0	0	0	-	-	-	-	-	-	-	-	-	-
Beam	21	16	16	11	16	4	0	69	100	31	0	52	76
Column	58	37	40	35	39	2	1	95	98	5	2	60	67
Beam \ Column	0	0	0	-	-	-	-	-	-	-	-	-	-
Base	0	0	0	-	-	-	-	-	-	-	-	-	-
Ground	0	0	0	-	-	-	-	-	-	-	-	-	-
Unclassified	-	53	44	-	-	-	-	-	-	-	-	-	-
	$\sum$ [-]							μ [%]					
	97	126	118	65	73	8	1	**89**	**99**	**11**	**1**	**77**	**86**

Tabelle 4.13: Ergebnisse der Objektklassifizierung für die GLC-Punktwolke.

Objektklasse	SOLL	IST [-]		TP [-]		FP [-]		TP [%]		FP [%]		Ω [%]	
	[-]	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2
Slab	6	6	6	6	6	0	0	100	100	0	0	100	100
Wall	13	11	13	11	13	0	0	100	100	0	0	85	100
Slab \ Wall	0	0	0	-	-	-	-	-	-	-	-	-	-
Beam	13	13	13	8	9	5	4	62	69	38	31	62	69
Column	99	63	70	63	70	0	0	100	100	0	0	64	71
Beam \ Column	0	0	0	-	-	-	-	-	-	-	-	-	-
Base	0	0	0	-	-	-	-	-	-	-	-	-	-
Ground	0	0	0	-	-	-	-	-	-	-	-	-	-
Unclassified	-	37	27	-	-	-	-	-	-	-	-	-	-
	$\sum$ [-]							μ [%]					
	131	130	129	88	98	5	4	**90**	**92**	**10**	**8**	**77**	**85**

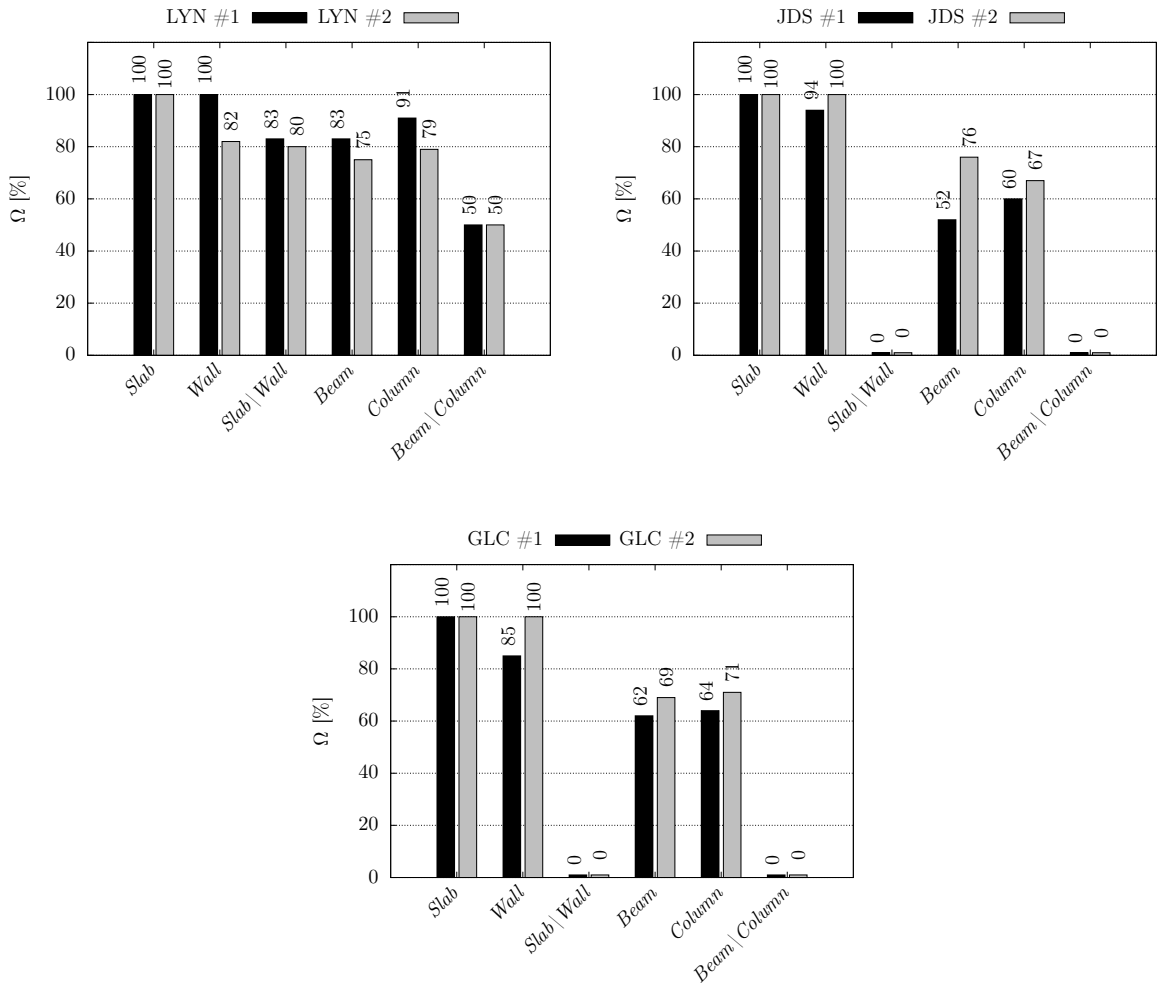


Abbildung 4.38: Darstellung der Gesamtgenauigkeit Ω für die identifizierten Objektklassen aus der LYN-, JDS- und GLC-Punktwolke in Form eines Balkendiagramms.

Die geringe Gesamtgenauigkeit von 62% und 69% bei der Klassifizierung der Träger (Abbildung 4.38 (unten)) kann auf drei Hauptursachen zurückgeführt werden:

(1) In der vorliegenden GLC-Punktwolke, die aus von außen aufgenommenen Schrägbildern erzeugt wurde, fehlen Punktdaten der Träger in den nicht sichtbaren Bereichen, was die korrekte Klassifizierung im geometriebasierten Ansatz erschwert.

(2) In einigen Fällen wurden Punktdaten von Deckenränder fälschlicherweise als Träger klassifiziert, was zu einer weiteren Verringerung der Genauigkeit beigetragen hat.

(3) In mehreren Fällen führte die Anwendung der RE-Methode dazu, dass die Punktdaten der Träger den Wänden zugeordnet wurden, was die Klassifizierung dieser Träger verhinderte.

Die Objektklassifizierung der Punktsegmente der GLC-Punktwolke dauerte insgesamt 0,16 bzw. 0,21 Minuten. Zusammenfassend zeigt die Ergebnisauswertung der Objektklassifizierung, dass die angewandte Methode äußerst präzise und zuverlässig bei der Klassifizierung der Objekte ist, wobei der durchschnittliche Anteil der korrekt klassifizierten Objekte (TP) bei mehr als 90% liegt. Trotz einiger Herausforderungen bei der Klassifizierung von Stützen und Träger, insbesondere aufgrund der Anwendungsgrenzen der RE-Methode und fehlender Punktdaten, zeigt die

Gesamtbetrachtung, dass die angewendete Methode im Bereich der semantischen Informationsanreicherung für die automatische BIM-Erzeugung effektiv ist.

4.4 Materialklassifizierung

Zusätzlich zur Objektklasse wird die Materialklasse der Gebäudekomponenten mit Hilfe eines DL-basierten Klassifikationsverfahrens bestimmt und damit die semantischen Informationen der Objekte erweitert. In diesem Zusammenhang wird der vorherrschende RGB-Wert eines Punktsegments als *feature* zur Vorhersage der Materialklasse verwendet. Das trainierte DL-Modell umfasst die Materialklassen *Concrete* und *Masonry* sowie die nicht-materialbezogene Klasse *Unclassified*. Weitere Einzelheiten zum Aufbau des DNNs sowie zum Training und Testen des DL-Modells sind in Abschnitt 3.6 zu finden.
Die Bewertung der Ergebnisse der Materialklassifizierung für die Segmente der LYN-, JDS- und GLC-Punktwolke erfolgt nach dem gleichen Schema wie bei der Objektklassifizierung. Eine detaillierte Erläuterung des entsprechenden Bewertungsverfahrens kann in Abschnitt 4.3 nachgelesen werden. Die Abbildungen 4.39 bis 4.41 zeigen die farblich gekennzeichneten Materialklassen der LYN-, JDS- und GLC-Punktwolke, die mit dem vortrainierten DL-Modell vorhergesagt wurden. Darüber hinaus sind in den Tabellen 4.14 bis 4.16 die Ergebnisse der Materialklassifizierung für die genannten Punktwolken quantitativ zusammengefasst. Anhand dieser Tabellen wird nachfolgend die Leistungsfähigkeit der DL-basierten Klassifizierungsmethode bewertet. Zum besseren Verständnis werden die visuellen Darstellungen der Ergebnisse in den Abbildungen 4.39 bis 4.41 als Hilfsmittel für die Bewertung herangezogen.

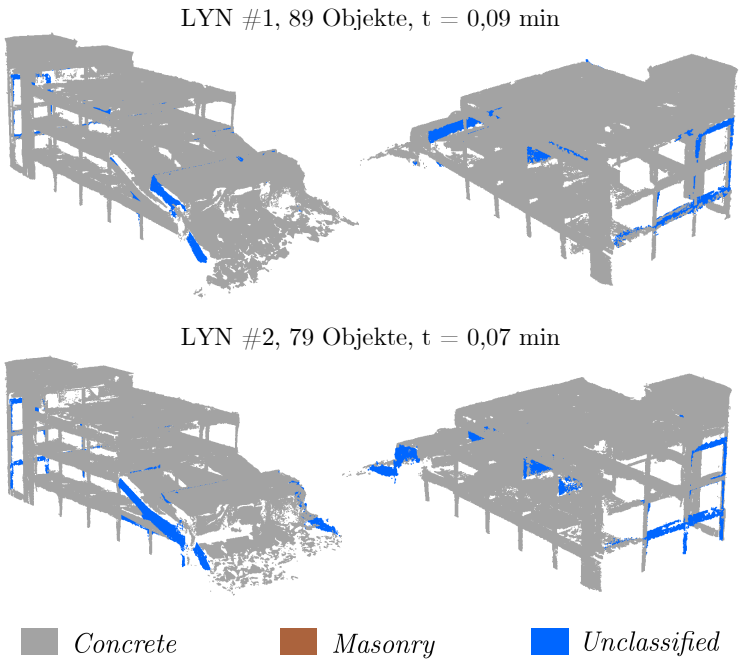


Abbildung 4.39: Darstellung der Ergebnisse der Materialklassifizierung für die LYN-Punktwolke.

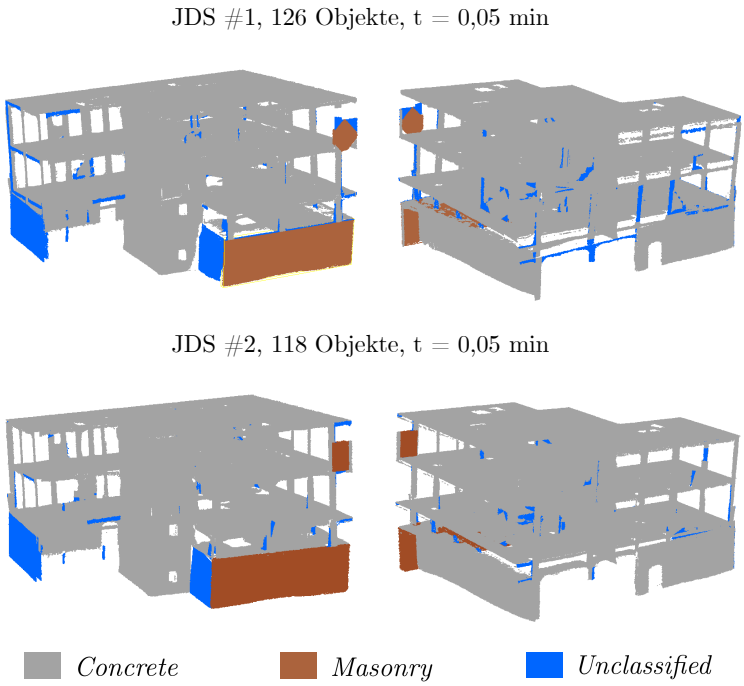


Abbildung 4.40: Darstellung der Ergebnisse der Materialklassifizierung für die JDS-Punktwolke.

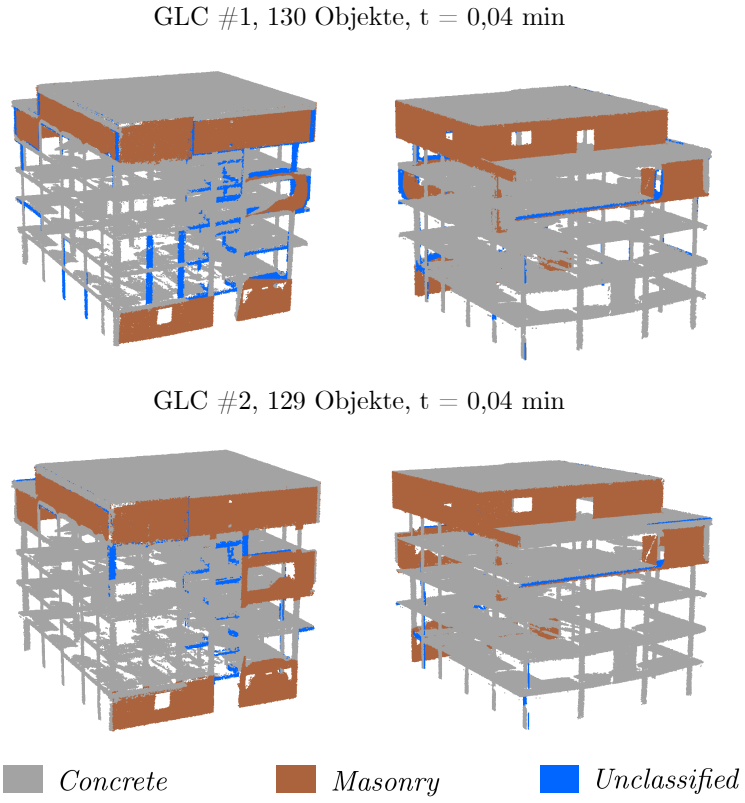


Abbildung 4.41: Darstellung der Ergebnisse der Materialklassifizierung für die GLC-Punktwolke.

Tabelle 4.14: Ergebnisse der Materialklassifizierung für die LYN-Punktwolke.

Materialklasse	SOLL	IST [-]		TP [-]		FP [-]		TP [%]		FP [%]		Ω [%]	
	[-]	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2
Concrete	68	59	54	59	54	0	0	100	100	0	0	87	79
Masonry	0	0	0	0	0	0	0	-	-	-	-	-	-
Unclassified	-	30	25	-	-	-	-	-	-	-	-	-	-
	$\sum$ [-]							μ [%]					
	68	89	79	59	54	0	0	**100**	**100**	**0**	**0**	**87**	**79**

Tabelle 4.15: Ergebnisse der Materialklassifizierung für die JDS-Punktwolke.

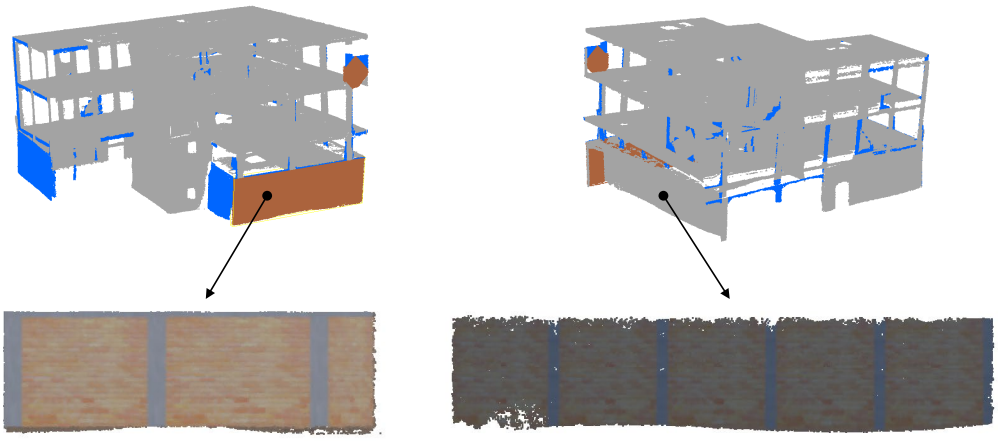
Materialklasse	SOLL	IST [-]		TP [-]		FP [-]		TP [%]		FP [%]		Ω [%]	
	[-]	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2
Concrete	86	69	70	62	63	7	7	90	90	10	10	72	73
Masonry	11	2	2	2	2	0	0	100	100	0	0	18	18
Unclassified	-	55	46	-	-	-	-	-	-	-	-	-	-
	$\sum$ [-]							μ [%]					
	97	126	118	64	65	7	7	**95**	**95**	**5**	**5**	**45**	**46**

Tabelle 4.16: Ergebnisse der Materialklassifizierung für die GLC-Punktwolke.

Materialklasse	SOLL	IST [-]		TP [-]		FP [-]		TP [%]		FP [%]		Ω [%]	
	[-]	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2	#1	#2
Concrete	117	79	102	77	101	2	1	97	99	3	1	66	86
Masonry	14	13	13	13	13	0	0	100	100	0	0	93	93
Unclassified	-	37	27	-	-	-	-	-	-	-	-	-	-
	$\sum$ [-]							μ [%]					
	131	129	142	90	114	2	1	**99**	**100**	**1**	**0**	**79**	**90**

Aus der Tabelle 4.14 ist zu entnehmen, dass die Materialklassifizierung der LYN-Punktwolke auf der Grundlage des vorherrschenden RGB-Wertes des Punktsegments in beiden Durchläufen zu 100% richtig durchgeführt wurde. Auf diese Weise konnten insgesamt 87% bzw. 79% der tatsächlich vorhandenen Materialien, wobei in diesem Fall nur die Materialklasse *Concrete* existiert, korrekt für das jeweilige Bauteil prognostiziert werden. Die visuelle Darstellung der Materialklassifizierung für die LYN-Punktwolke in Abbildung 4.39 zeigt, dass einige Segmente nicht klassifiziert werden konnten. Dies ist darauf zurückzuführen, dass der Vertrauenswert (*confidence*-Wert) in den blau markierten Segmenten weniger als 80% beträgt, was eine zuverlässige Materialklassenvorhersage verhindert. In beiden Durchläufen betreffen diese nicht-klassifizierten Segmente der LYN-Punktwolke weitgehend die gleichen Bereiche. Dies unterstreicht die konsistente Leistung des DL-Modells, was als positiver Aspekt anzusehen ist. Die DL-basierte Materialklassifizierung der LYN-Punktwolke dauerte im ersten Durchlauf 0,09 Minuten und im zweiten Durchlauf 0,07 Minuten.
Aus den Ergebnissen der Materialklassifizierung für die JDS-Punktwolke in Tabelle 4.15 ist eine unzureichende Genauigkeit des vortrainierten DL-Modells zu entnehmen. Hier liegt die durchschnittliche Gesamtgenauigkeit μ bei lediglich 45% bzw. 46%. Diese niedrige Gesamtgenauigkeit

resultiert hauptsächlich aus der falschen Klassifizierung der Mauerwerkswände, von denen sieben als *Concrete* klassifiziert wurden. Nur zwei der insgesamt neun Mauerwerkswände konnten in beiden Durchläufen der richtigen Materialklasse zugeordnet werden. Zur Veranschaulichung zeigt die Abbildung 4.42 zwei Segmente von Mauerwerkswänden mit ihrer vorherrschenden RGB-Farbe aus der JDS-Punktwolke, von denen das Material in einem Fall richtig und im anderen Fall falsch klassifiziert wurde. In dieser Abbildung ist zu sehen, dass die Mauerwerkswände unterschiedliche Farben aufweisen, da eine von ihnen dem direkten Sonnenlicht ausgesetzt ist, während die andere im Schatten liegt. Dies führt dazu, dass die Mauerwerkswand im Schatten einen vorherrschenden RGB-Farbton aufweist, der innerhalb des Farbspektrums für *Concrete* liegt. Die übrigen sechs fehlerhaft klassifizierten Mauerwerkwände befinden sich ebenfalls im Schatten und wurden daher irrtümlich vom DL-Modell als *Concrete* klassifiziert.



Richtig Positive (TP)
Materialklassifizierung

Falsch Positive (FP)
Materialklassifizierung

Vorherrschende RGB-Farbe

Vorherrschende RGB-Farbe

Abbildung 4.42: Darstellung einer richtig positiven und einer falsch positiven Materialklassifizierung von Wandsegmenten aus der JDS-Punktwolke mit ihrer vorherrschenden RGB-Farbe.

Dieses Ergebnis legt eine Schwäche in der vorgeschlagenen Methodik der Materialklassifizierung offen. Diese Schwäche, die mit einer erheblichen Unzuverlässigkeit bei der Vorhersage der Materialklasse einhergeht, tritt auf, wenn eine der folgenden Bedingungen nicht erfüllt ist:

- Die tatsächliche Materialfarbe muss deutlich erkennbar sein und darf nicht durch Putz oder andere Farbüberlagerungen verdeckt werden.
- Es muss ausreichend Beleuchtung vorhanden sein, sei es durch künstliche Lichtquellen oder natürliches Sonnenlicht, damit die tatsächliche Farbe sichtbar wird.
- Die tatsächliche Materialfarbe muss in ausreichender Menge vorhanden sein, um zuverlässig für die Vorhersage genutzt werden zu können.

Die oben genannten Bedingungen unterstreichen die Anwendungsgrenzen der entwickelten Methodik der Materialklassifizierung. Des Weiteren legt die hohe Anzahl nicht-klassifizierbarer Segmente in der JDS-Punktwolke nahe, dass das DL-Modell mit zusätzlichen Trainingsdaten verbessert werden muss, um eine eindeutige und zuverlässige Vorhersage der Materialklasse auf der

Grundlage eines einzelnen RGB-Wertes zu ermöglichen. Derzeit sind die Trainingsdaten auf jeweils 1000 RGB-Datensätze für die Zielwerte *Masonry* und *Concrete* begrenzt. Die DL-gestützte Bestimmung der Materialklasse für die Komponenten der JDS-Punktwolke dauerte in beiden Durchläufen insgesamt 0,05 Minuten. Mit der DL-gestützten Materialklassifizierung konnten für die GLC-Punktwolke eine durchschnittliche Gesamtgenauigkeit von 79% im ersten Durchlauf und 90% im zweiten Durchlauf erreicht werden (Tabelle 4.16). In beiden Durchgängen wurden 93% der tatsächlich vorhandenen Mauerwerkswände korrekt klassifiziert. Des Weiteren konnte im zweiten Durchlauf insgesamt 86% der tatsächlich vorhandenen Betonbauteile mit der richtigen Materialklasse vorhergesagt werden. Die Anzahl der falsch positiven Materialklassifizierungen der Klasse *Concrete* kann wie im Fall der JDS-Punktwolke auf die fehlerhafte Interpretation der Oberflächenfarbe durch das DL-Modell zurückgeführt werden, da die Oberflächenfarbe der Bauteile des GLC-Gebäudes nicht immer der tatsächlichen Materialfarbe entspricht. Dennoch ist die Anzahl der falsch positiven Vorhersagen für den GLC-Datensatz mit 3% im ersten Durchlauf und 1% im zweiten Durchlauf im Verhältnis zu den korrekten positiven Vorhersagen äußerst gering. Die hohe Anzahl nicht-klassifizierbarer Punktsegmente in der GLC-Punktwolke (Abbildung 4.41 und Tabelle 4.16) zeigt, wie in den vorherigen Fällen, dass das DL-Modell mit zusätzlichen Trainingsdaten verbessert werden muss, um einen *confidence*-Wert von über 80% für die Vorhersage einer Materialklasse auf Basis eines RGB-Wertes zu erreichen. Die Materialklassifizierung für die Punktsegmente der GLC-Punktwolke dauerte in beiden Durchläufen jeweils 0,04 Minuten. In Anbetracht der Tatsache, dass die Materialklassifizierungen für die LYN-, JDS- und GLC-Punktwolken jeweils maximal 0,09 Minuten in Anspruch nahmen, kann gefolgert werden, dass die DL-gestützte Klassifizierungsmethode eine hohe Effizient bei der schnellen Verarbeitung großer Punktdaten aufweist.
Zusammenfassend zeigt die Evaluierung der DL-gestützten Materialklassifizierung für verschiedene Punktwolken von seismisch beschädigten Stahlbetonrahmenkonstruktionen die Potenziale und Herausforderungen dieser Methode auf. Die Ergebnisse verdeutlichen, dass die Materialklassifizierungsmethode zuverlässige Vorhersagen ermöglicht, sofern die wahre Materialfarbe im Punktsegment klar erkennbar ist. Diese Bedingungen wurden weitgehend für die LYN- und GLC-Punktwolken eingehalten, während die Klassifizierung der Mauerwerkswände in der JDS-Punktwolke aufgrund von Schattenverhältnissen eine Herausforderung darstellten. Dies betont die Wichtigkeit, die Methode durch die Integration weiterer Merkmale (*features*) für die Materialklassifizierung zu erweitern, um ihre Leistungsfähigkeit zu steigern. Darüber hinaus verdeutlichen die Ergebnisse die Notwendigkeit, das DL-Modell durch zusätzliche Trainingsdaten zu verbessern, um eine zuverlässigere Vorhersage der Materialklasse auf der Grundlage eines RGB-Wertes zu gewährleisten. Schließlich unterstreicht die Effizienz der Methode bei der schnellen Verarbeitung großer Datensätze, dass die DL-basierte Materialklassifizierung eine vielversprechende Technologie für zukünftige Anwendungen im Bereich der Punktwolkenanalyse darstellt.

4.5 Digitale 3D-Rekonstruktion

In diesem Abschnitt wird der vom Autor dieser Arbeit vorgeschlagene Ansatz zur Erstellung von geschlossenen *Mesh*-Körpern aus dreidimensionalen Punktsegmenten im Kontext der digitalen 3D-Rekonstruktion evaluiert. Dieser Schritt ist von entscheidender Bedeutung, um ein BIM-Modell basierend auf den *Mesh*-Körpern der BC-Objekte zu generieren. Die in Abschnitt 3.7 vorgestellte Methode zur digitalen 3D-Rekonstruktion der sichtbaren Bauteile basiert auf der Verwendung der α-*Shape*-Methode [54, 179] zur präzisen Oberflächenrekonstruktion. Dabei wird die digital rekonstruierte *Mesh*-Oberfläche entlang ihrer Oberflächennormalen um die im BC-Objekt gespeicherte Bauteildicke t_i extrudiert. Auf diese Weise entsteht für das jeweilige BC-Objekt ein geschlossener Volumenkörper, der aus Dreiecksflächen besteht. Der detaillierte Ablauf

dieses Prozesses ist im Algorithmus 9 in Form eines Pseudocodes beschrieben.
Es ist wichtig zu beachten, dass die digitale Rekonstruktion der Punktsegmente ausschließlich für die Datensätze durchgeführt wurde, bei denen entweder im Durchlauf #1 oder #2 bessere Ergebnisse im Zuge der algorithmischen Verarbeitung der Punktdaten erzielt wurden. Im Einzelnen sind dies die Resultate aus Durchlauf #1 für LYN, Durchlauf #2 für JDS und ebenfalls Durchlauf #2 für GLC. Diese Auswahl stellt sicher, dass der abschließende Schritt der algorithmischen Datenverarbeitung, d. h. die automatische BIM-Generierung, auf den optimal verfügbaren Daten basiert. Des Weiteren ermöglicht dies eine präzisere Bewertung der Leistungsfähigkeit der Rekonstruktionsmethode sowie eine bessere Identifizierung von möglichen Optimierungsmöglichkeiten.
Wie bereits in Unterabschnitt 3.7.2 erläutert, erfordert die Oberflächenrekonstruktion mittels der α-*Shape*-Methode die Festlegung eines Wertes für den α-Parameter. Dieser Parameter liegt im Bereich zwischen 0 und 1 und wird wie folgt definiert: $\alpha \in \{x \in \mathbb{R}^+ \mid 0 \leq x \leq 1\}$.
Dabei ist zu beachten, dass ein α-Wert von 1 die Erzeugung einer konvexen Oberfläche zur Folge hat, während für $\alpha < 1$ die Generierung einer konkaven Oberflächenform approximiert wird. Um eine vollautomatische Oberflächenrekonstruktion für Punktsegmente zu realisieren, setzt das vorgeschlagene Verfahren einen konstanten Wert für den α-Parameter voraus. In diesem Kontext wurde die Oberflächenrekonstruktion der Punktsegmente in den BC-Objekten von LYN, JDS und GLC mit Hilfe von zwei konstanten α-Werten, nämlich 1,0 und 0,5, durchgeführt. Die daraus resultierenden *Mesh*-Körper der Komponenten der beschädigten Stahlbetonrahmengebäude sind in den Abbildungen 4.43 bis 4.45 abgebildet.

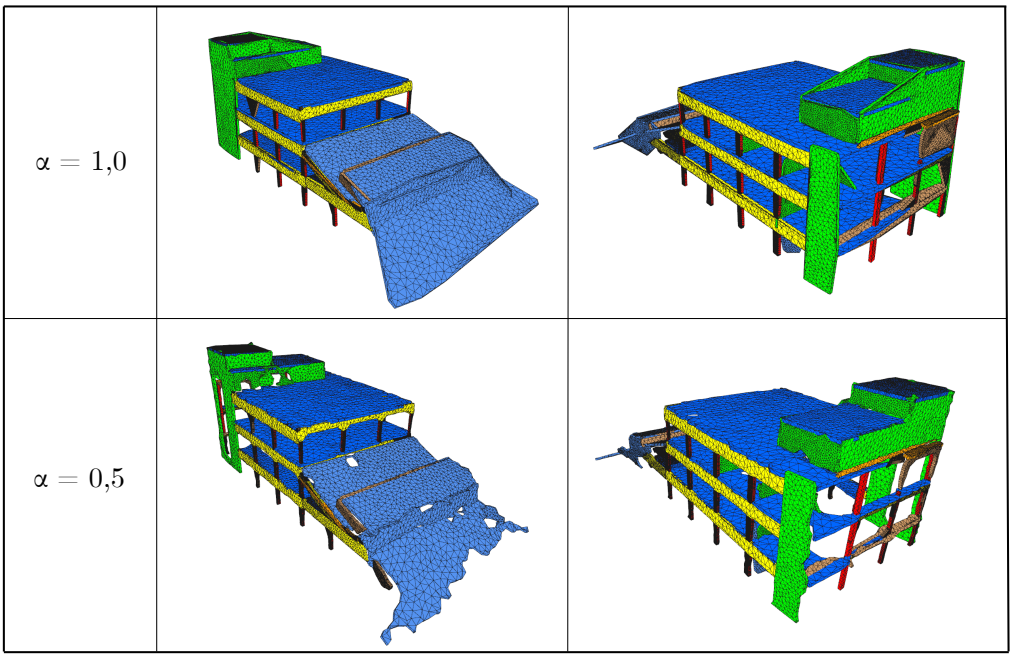


Abbildung 4.43: Ergebnisse der 3D-Rekonstruktion für die Objekte der LYN-Punktwolke unter Verwendung von $\alpha = 1,0$ und $\alpha = 0,5$.

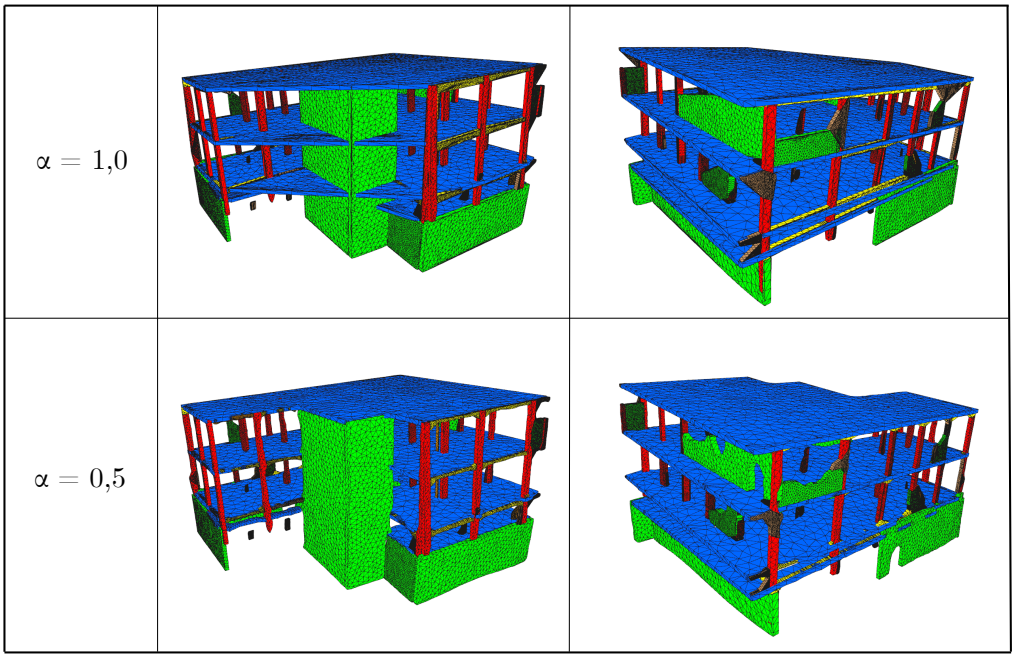


Abbildung 4.44: Ergebnisse der 3D-Rekonstruktion für die Objekte der JDS-Punktwolke unter Verwendung von $\alpha = 1,0$ und $\alpha = 0,5$.

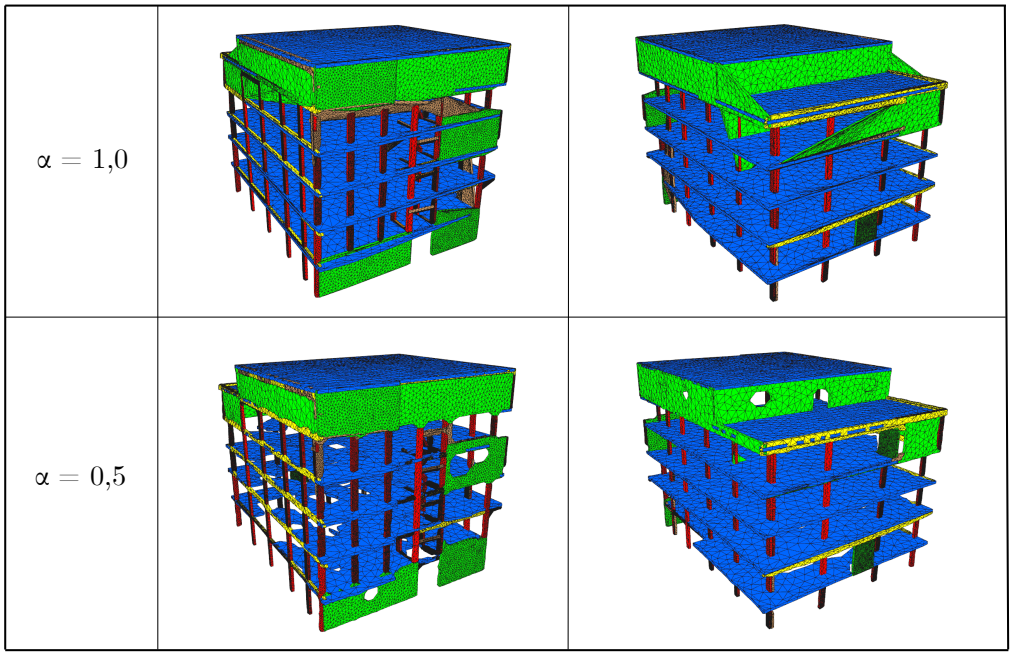


Abbildung 4.45: Ergebnisse der 3D-Rekonstruktion für die Objekte der GLC-Punktwolke unter Verwendung von $\alpha = 1,0$ und $\alpha = 0,5$.

Die Ergebnisse in den Abbildungen 4.43 bis 4.45 verdeutlichen, dass die Verwendung eines konstanten α-Wertes in vielen Fällen zu ungenauen Ergebnissen bei der Oberflächenrekonstruktion der Punktsegmente führt. Dies liegt daran, dass die Punktsegmente der Gebäudekomponenten unterschiedliche Formen und Vollständigkeitsgrade aufweisen. Mit einem α-Wert von 1,0 wurde eine konvexe Oberflächenrekonstruktion durchgeführt, was bei Oberflächen mit konkaver Form

zu fehlerhaften Oberflächengenerierungen und somit zu erheblichen Ungenauigkeiten bei vielen Komponenten führte.
Durch die Verwendung eines α-Wertes von 0,5 erfolgte eine Oberflächengenerierung mit einer moderaten Konkavität. Im Vergleich zur konvexen Oberflächenrekonstruktion näherte sich die generierte konkave Netzstruktur insgesamt besser der tatsächlichen Form der Objekte an. Dennoch war es auch unter Verwendung des festgelegten α-Wertes von 0,5 nicht möglich, die Oberfläche aller Punktsegmente optimal zu rekonstruieren. Dies ergibt sich aus der Tatsache, dass einige Punktsegmente, die z. B. Wandöffnungen aufweisen, einen α-Wert kleiner als 0,5 benötigen, um die Öffnungen bei der Oberflächenrekonstruktion angemessen zu berücksichtigen. Andererseits erfordern andere Segmente, die aufgrund von Verdeckungen unvollständig sind, einen α-Wert größer als 0,5, um die durch Verdeckungen entstandenen Lücken bei der Generierung der Oberfläche zu schließen. In Bezug auf die Berechnungszeit ergab sich für die 3D-Rekonstruktion folgendes: Für LYN dauerte die digitale 3D-Rekonstruktion 3,2 Minuten, für JDS 3,1 Minuten und für GLC 1,7 Minuten. Es ist dabei beachtenswert, dass der α-Wert von 0,5 im Vergleich zu 1,0 keine Auswirkungen auf die erforderliche Berechnungszeit hatte.
In Anbetracht der signifikanten Anwendungen, die auf den Ergebnissen der digitalen 3D-Rekonstruktion basieren, wird die Bedeutung der Präzision in der Oberflächenrekonstruktion deutlich. Insbesondere im Rahmen von Rettungseinsätzen spielen diese Ergebnisse eine entscheidende Rolle. Mit einem genauen digitalen Modell könnten Rettungskräfte fundierte Entscheidungen treffen, um den effizientesten Zugang zu einem Gebäude zu gewährleisten und die strukturelle Stabilität von Bauteilen objektiv abzuschätzen. Dabei ist es unerlässlich, die Herausforderungen bei der algorithmischen Verarbeitung unvollständiger Punktwolken aufgrund von Verdeckungen während der Datenerfassung zu berücksichtigen. In solchen Fällen ist es von großer Bedeutung, geeignete Methoden anzuwenden, um den verdeckten Bereich des Gebäudes möglichst präzise und objektiv abzuschätzen. Eine gründliche Behandlung dieser Problematik kann dazu beitragen, die Genauigkeit und Zuverlässigkeit der digitalen Rekonstruktion erheblich zu verbessern. In diesem Zusammenhang kann die vom Autor vorgeschlagene Methode in Abschnitt 3.8 als Leitfaden dienen, um diese Herausforderung zu bewältigen und letztlich die Qualität der Rekonstruktionsergebnisse zu optimieren.
Die aus den Abbildungen 4.43 bis 4.45 ersichtlichen Rekonstruktionsergebnisse zeigen auf, dass die Verwendung eines konstanten α-Wertes keine ausreichende Genauigkeit bei der digitalen 3D-Rekonstruktion gewährleisten kann. Um die Ungenauigkeiten in der Oberflächenrekonstruktion zu beheben, ist es erforderlich, den α-Wert automatisch und individuell für jedes Punktsegment anzupassen. Eine mögliche Umsetzung dieses Ansatzes könnte die Verwendung von Algorithmen des maschinellen Lernens und der Mustererkennung beinhalten, um die Form der einzelnen Punktsegmente zu analysieren und den entsprechenden α-Wert dynamisch anzupassen. Dies würde eine präzisere und genauere Oberflächenrekonstruktion ermöglichen, unabhängig von den variierenden geometrischen Eigenschaften der einzelnen Punktsegmente.
Um den Effekt der optimalen Anpassung des α-Wertes zu illustrieren, wurde für jedes Punktsegment von LYN, JDS und GLC der optimale α-Wert für die Oberflächenrekonstruktion manuell ermittelt. Dazu wurde der α-Wert zunächst auf 1,0 gesetzt und dann schrittweise in 0,1-Schritten verringert, bis die Form nicht mehr genauer angenähert werden konnte. Dabei wurde die erzeugte Netzstruktur nach jedem Iterationsschritt zur Kontrolle visualisiert. Dieser manuelle Aufwand wurde unternommen, um zu veranschaulichen, wie das optimale Endergebnis der digitalen 3D-Rekonstruktion unter Verwendung der α-*Shape*-Methode aussehen könnte, wenn der α-Wert automatisch optimiert werden könnte, z. B. durch maschinelles Lernen. Die Ergebnisse sind in der Abbildung 4.46 dargestellt. Die digital rekonstruierten Bauteile der beschädigten Stahlbetonrahmengebäude in der Abbildung 4.46 zeigen im Vergleich zu den Ergebnissen in den Abbildungen 4.43 bis 4.45 eine deutliche Verbesserung der Genauigkeit hinsichtlich der Annäherung an die

tatsächliche Oberflächenform und der Berücksichtigung von Öffnungen in der Netzstruktur. Die erhaltenen Rekonstruktionsergebnisse mit variierenden α-Werten illustrieren den positiven Effekt einer optimalen Parameteranpassung zur Verbesserung der Genauigkeit bei der Oberflächenrekonstruktion von Punktdaten. In diesem Zusammenhang kann die Integration eines automatisierten Verfahrens zur formabhängigen Anpassung des α-Wertes die Leistungsfähigkeit der digitalen 3D-Rekonstruktion zielführend optimieren.

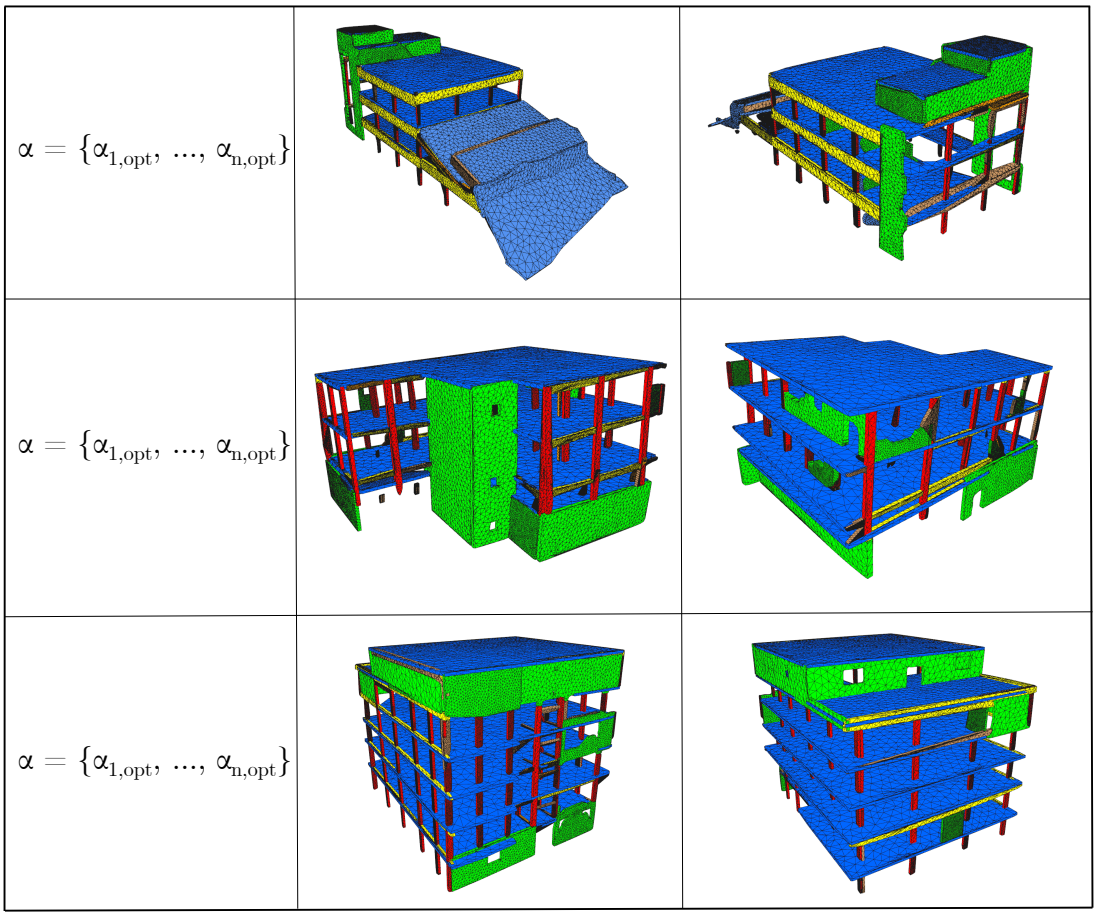


Abbildung 4.46: Ergebnisse der 3D-Rekonstruktion für die Objekte der LYN-, JDS- und GLC-Punktwolke unter Verwendung von optimalen α-Werten.

Gemäß der Beschreibung der digitalen Rekonstruktionsmethode in Abschnitt 3.7 wird die im BC-Objekt gespeicherte Dicke t_i verwendet, um die erzeugte Oberfläche entlang ihrer Flächennormalen zu extrudieren. In den Fällen, in denen das entsprechende Punktsegment kein paralleles Segment aufweist, von dem die Dicke t_i abgeleitet werden kann, wird dem Segment die Standarddicke t_d zugewiesen. Im Rahmen der Evaluierung wurde die Standarddicke t_d auf 25 cm festgelegt. Eine Analyse der Daten in den BC-Objekten zeigt, dass nur 3% der Segmente in der LYN-Punktwolke parallele Segmente aufweisen, während dieser Prozentsatz in JDS bei 18% und in der GLC-Punktwolke bei 11% liegt. Diese niedrigen Prozentsätze deuten darauf hin, dass die Identifizierung paralleler Punktsegmente durch das Fehlen von Punktdaten erschwert wurde, was mit den bereits erwähnten Einschränkungen bei der Datenerfassung zusammenhängt. Dementsprechend weisen die meisten rekonstruierten Bauteile der ausgewählten Gebäude eine Dicke von 25 cm auf, was eine große Ungenauigkeit bei der Modellrekonstruktion darstellt.
Eine mögliche Lösung für diese Anwendungsgrenze besteht darin, bei der Bestimmung der Bauteildicke t_i nicht nur das parallele Segment zu berücksichtigen, sondern auch die benachbarten

Punktsegmente des Bauteils einzubeziehen. Die genaue Identifizierung eines solchen benachbarten Punktsegments stellt jedoch eine Herausforderung dar, da verschiedene Faktoren, wie z. B. eine unzureichende Punktdichte oder eine komplexe Oberflächengeometrie aufgrund von seismischen Schäden, eine zuverlässige Zuordnung eines Segments zu einer Komponente erschweren können. Darüber hinaus könnte die Dicke auf diese Weise ungenau bestimmt werden, wenn das benachbarte Punktsegment die Oberfläche der zugehörigen Komponente nicht vollständig bedeckt.
Zusammenfassend zeigt die Auswertung der Ergebnisse der digitalen 3D-Rekonstruktion die vielversprechende Leistungsfähigkeit der angewandten Methodik, verdeutlicht aber gleichzeitig die damit verbundenen Anwendungsgrenzen und die Bedeutung weiterer Forschungsanstrengungen auf diesem Gebiet, um die Genauigkeit und Zuverlässigkeit des vorgeschlagenen Rekonstruktionsverfahrens weiter zu verbessern. Besonders kritisch ist in diesem Zusammenhang die Verbesserung der Genauigkeit der Oberflächenrekonstruktion durch eine formabhängige Anpassung des α-Wertes im α-*Shape*-Algorithmus.

4.6 BIM-Generierung

Abschließend wird der letzte Schritt der algorithmischen Datenverarbeitung für die betrachteten Gebäudemodelle evaluiert, bei dem die in den BC-Objekten enthaltenen Daten zur Beschreibung der digitalen Gebäudekomponenten in den BIM-Prozess übertragen werden. Hierfür wird der in Unterabschnitt 3.9.1 vorgeschlagene *Mesh-to-IFC*-Ansatz angewendet. Dabei werden die digital rekonstruierten Gebäudekomponenten im OBJ-*Mesh*-Format in IFC umgewandelt und die daraus resultierenden IFC-Objekte mit semantischen und alphanumerischen Informationen angereichert. Der Pseudocode der in Python implementierten *Mesh-to-IFC*-Methode ist in Algorithmus 13 aufgeführt.
Im Gegensatz zu den vorangegangenen Arbeitsschritten erfordert die Anwendung des *Mesh-to-IFC*-Algorithmus keine manuelle Festlegung von Parametern und kann somit vollautomatisch ausgeführt werden. Dies liegt daran, dass die triangulierten *Mesh*-Daten aus der digitalen 3D-Rekonstruktion als geometrische Grundlage für die Erzeugung der IFC-Objekte dienen und die in den BC-Objekten enthaltenen Daten, wie beispielsweise die Objekt- und Materialklasse, zur Anreicherung der semantischen Informationen genutzt werden. Die umgewandelten Gebäudemodelle der LYN-, JDS- und GLC-Datensätze im IFC-Format sowie die für diesen Prozess benötigte Rechenzeit sind in der Abbildung 4.47 dargestellt.
Wie aus der Abbildung 4.47 ersichtlich ist, wurden alle digital rekonstruierten Komponenten der LYN-, JDS- und GLC-Datensätze mit der *Mesh-to-IFC*-Methode erfolgreich in das IFC-Format konvertiert. In diesem Zusammenhang ist die Effizienz des Prozesses besonders hervorzuheben, da die Datenkonvertierung und Informationsanreicherung für alle Datensätze weniger als eine Minute in Anspruch nahm. Um die Funktionalität der erzeugten IFC-Dateien zu überprüfen, wurde sie in verschiedenen BIM-Programmen, darunter *BIMVision* [1], *FreeCAD* [64] und *Revit* [14], importiert und überprüft. Die IFC-Dateien konnten in den genannten Programmen erfolgreich importiert und bearbeitet werden, wobei alle erzeugten IFC-Objekte und die zugehörigen Attribute und Eigenschaftssätze vollständig enthalten waren. Diese erfolgreiche Integration und Überprüfung bekräftigen die Interoperabilität und Anpassungsfähigkeit der generierten BIM-Daten sowohl für den Bereich des Katastrophenmanagements als auch für die Bauindustrie. Eine eingehende Prüfung der zugewiesenen IFC-Klasse und der Eigenschaftssätze (*IfcPropertySets*) für die Objekte ergab, dass die Zuordnung der IFC-Klasse und der Parameterwerte in den Eigenschaftssätzen, wie der E-Modul, die Dichte und die Masse der Bauteile, basierend auf den im BC-Objekt enthaltenen Daten korrekt erfolgte.

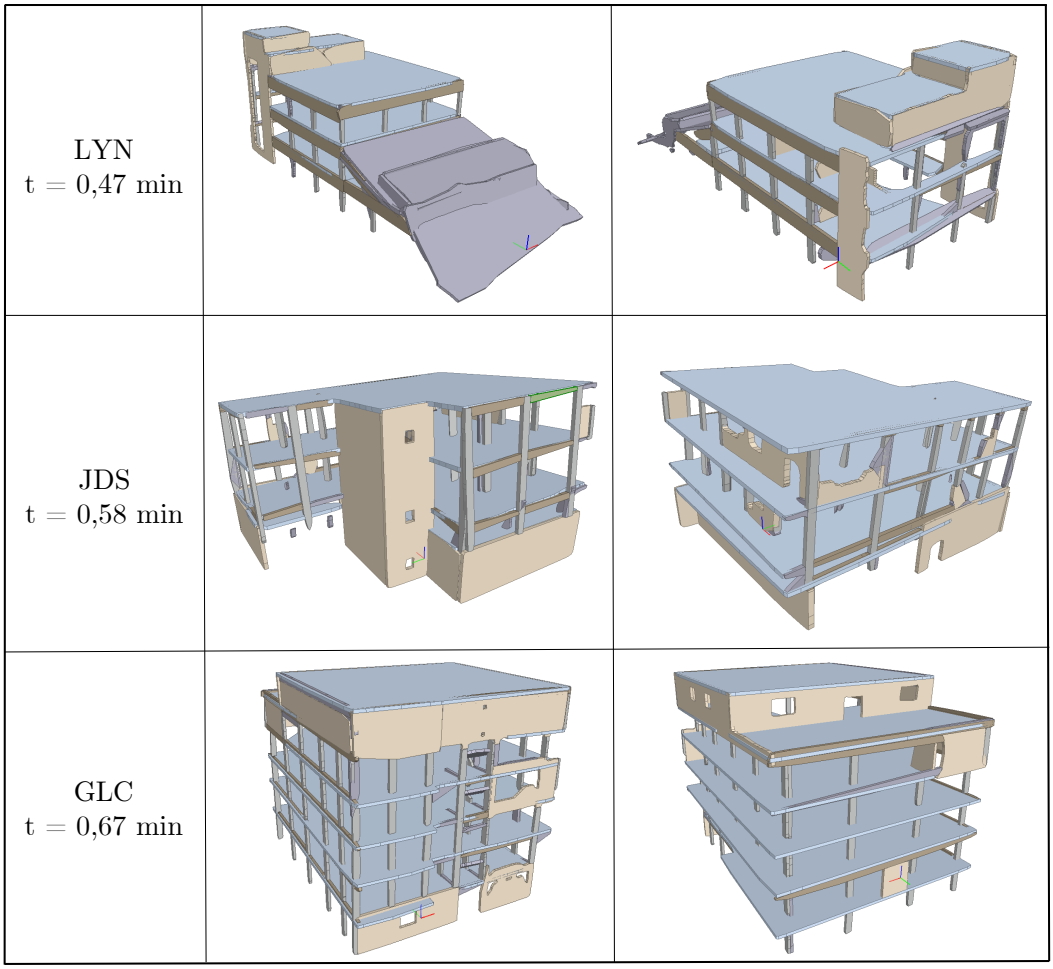


Abbildung 4.47: Ergebnisse der *Mesh-to-IFC*-Methode für die Objekte der LYN-, JDS- und GLC-Punktwolke.

Wie bereits erwähnt, entsprechen die geometrischen Daten der IFC-Objekte denen der *Mesh*-Objekte aus der digitalen Rekonstruktion. Eine Verbesserung der Genauigkeit in Bezug auf die Bauteilgeometrie kann daher nur durch eine entsprechende Optimierung der vorangegangenen Arbeitsschritte erreicht werden. Die generierten BIM-Modelle der Fallbeispiele (Abbildung 4.47) weisen ein LoD von 200 auf, was die zweitniedrigste Stufe darstellt, sowohl in Bezug auf die geometrische Darstellung der Gebäudekomponenten als auch bezüglich der semantischen Informationen und Attribute. Dies ist darauf zurückzuführen, dass die generierten Bauwerksdatenmodelle nur grundlegende Informationen über die Bauteile enthalten. Es ist jedoch wichtig zu betonen, dass die halbautomatische Erzeugung eines BIM-Modells mit einem LoD von 200 aus lediglich einer unvollständigen 3D-Punktwolke einer seismisch beschädigten Stahlbetonrahmenkonstruktion einen bedeutenden Fortschritt im Hinblick auf den aktuellen Stand der Wissenschaft und Technik darstellt. Die erzielten Ergebnisse bilden somit eine solide Grundlage für zukünftige Optimierungen, um die Genauigkeit des Detaillierungsgrades weiter zu verbessern und die Anwendungsmöglichkeiten dieser Methode zu erweitern. Der gesamte Prozess von der Vorverarbeitung der 3D-Punktwolken bis zu den endgültigen IFC-Modellen dauerte 16 Minuten für LYN, 20 Minuten für JDS und 14 Minuten für GLC. Diese vergleichsweise kurzen Verarbeitungszeiten unterstreichen die Effizienz der entwickelten modellunabhängigen Methodik, insbesondere in Bezug auf ihre Relevanz im Kontext der Katastrophenbewältigung.

4.7 Zusammenfassung

In diesem Kapitel wurde eine umfassende Evaluierung und Diskussion der entwickelten modellunabhängigen Methodik zur Erstellung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen auf der Grundlage einer 3D-Punktwolke durchgeführt. Ziel dieser Evaluierung war es, die Leistungsfähigkeit und Genauigkeit der einzelnen Schritte der in Kapitel 3 vorgestellten Methodik zu beurteilen und ihre Anwendungsgrenzen zu bestimmen. Die wesentlichen Ergebnisse und Erkenntnisse der durchgeführten Evaluierung sind im Folgenden zusammengefasst:
Als Datengrundlage für die Evaluierung wurden drei Fallbeispiele verwendet, die teilweise eingestürzte Stahlbetonrahmengebäude in Lyon (LYN), Jindires (JDS) und Gölcük (GLC) darstellen (Abschnitt 4.1). Der LYN-Datensatz enthält reale photogrammetrische Daten, während die JDS- und GLC-Bilddatensätze vom Autor dieser Arbeit synthetisch erzeugt wurden. Eine Parameterstudie mit *Agisoft Metashape* [10] ergab, dass die SfM-Photogrammetrie mit Schrägbildern im Vergleich zu Nadirbildern nicht nur äußerst genaue Punktwolken erzeugt, sondern auch deutlich zeitsparender ist. Bei Verwendung der mittleren Qualitätseinstellung von *Agisoft Metashape* konnte mit Schrägbildern innerhalb von 15,5 Minuten die 3D-Punktwolke für LYN erzeugt werden. Unter Verwendung der synthetisch erzeugten Bilddatensätze für JDS und GLC, die ausschließlich Schrägbilder aus der Luft enthalten, wurden die entsprechenden 3D-Punktwolken in *Agisoft Metashape* bei mittlerer Qualitätseinstellung nach maximal fünf Minuten generiert. Die erzeugten Punktwolken von LYN, JDS und GLC wurden jeweils einer Vorverarbeitung unterzogen, bei der nicht benötigte Bereiche aus den Punktdatensätzen manuell ausgeschnitten wurden. Darüber hinaus wurde der SOR-Filter [40] zur Entfernung von Ausreißern und der *Spatial-Subsample*-Filter [40] zur Vereinheitlichung der Punktwolkenauflösung auf 1 cm angewendet. Die Gesamtdauer der Vorverarbeitung betrug in allen Fallbeispielen nicht mehr als drei Minuten.
Im ersten Schritt der algorithmischen Datenverarbeitung wurden die ebenen Punktmengen aus den 3D-Punktdaten von LYN, JDS und GLC mit Hilfe des RANSAC-Algorithmus [149] segmentiert (Unterabschnitt 4.2.1). Da die Genauigkeit dieses Verfahrens von verschiedenen Parametern abhängt, wurden jeweils zwei Durchläufe (#1 und #2) durchgeführt. Im ersten Durchlauf wurden automatisch ermittelte Parameterwerte verwendet, während im zweiten Durchlauf manuelle Parameteranpassungen basierend auf den Ergebnissen des ersten Durchlaufes vorgenommen wurden. Die Ergebnisse der ebenen Punktsegmentierung zeigten eine bemerkenswerte Präzision und Effizienz des verwendeten Algorithmus. Es ist jedoch auch wichtig zu betonen, dass die automatische Festlegung der Parameterwerte für alle drei Fallbeispiele zu angemessenen Ergebnissen bei der Segmentierung der ebenen Punktmengen führte. Die ebene Punktsegmentierung wurde in allen Fällen in weniger als 1,5 Minuten abgeschlossen.
Anschließend wurden die Punktdaten der Wände durch Anwendung der RE-Methode (Unterabschnitt 4.2.2) aus den Rahmensegmenten ausgeschnitten. Die Evaluation der RE-Methode erfolgte durch den Vergleich der resultierenden Segmentierungen mit einem manuell erstellten *Ground Truth* für die Rahmensegmente mit Wänden. In diesem Kontext wurden zwei Durchläufe (#1 und #2) zur Extraktion der Wandpunktdaten aus den Rahmensegmenten durchgeführt. Dabei wurden im zweiten Durchlauf die oben genannten Parameter angepasst, um eine höhere Genauigkeit zu erzielen. Für die Rahmensegmente des LYN-Datensatzes wurde in beiden Durchläufen eine präzise Extraktion der Wandpunktdaten innerhalb von maximal 1,3 Minuten erreicht. Jedoch stellten insbesondere bei den Rahmensegmenten der JDS- und GLC-Punktwolken unregelmäßige Geometrien und komplexe Strukturen eine Herausforderung für den angewandten Ansatz dar. Dies resultierte daraus, dass die Punktdaten der teilweise beschädigten Wände nicht vollständig zwischen Stützen und Trägern lagen, wodurch ihre Form nicht präzise mit der α-*Shape*-Methode

extrahiert werden konnte. Des Weiteren wurde festgestellt, dass die RE-Methode die Segmentierung von ebenen Punktmengen, die sowohl Punktdaten von Wänden als auch von Stützen enthalten, als vollständige Wand durchführt. Dabei werden die Punktdaten der Stützen fälschlicherweise als Teil des Wandsegments betrachtet. Dies liegt daran, dass die α-*Shape*-Methode nur in der Lage ist, Randpunkte zu entfernen, während Punktdaten innerhalb einer Punktmenge nicht berücksichtigt werden können. Die gezielte Anpassung der Parameter für die JDS- und GLC-Rahmensegmente auf der Grundlage der Ergebnisse des ersten Durchlaufes führte zu einer erkennbaren Verbesserung der Segmentierungsgenauigkeit. Jedoch konnten trotz dieser Anpassungen die oben genannten Ungenauigkeiten nicht vollständig überwunden werden. Die Verarbeitungszeit für die Rahmensegmente der JDS-Punktwolke belief sich auf etwa 7 Minuten, während für die GLC-Rahmensegmente weniger als zwei Minuten benötigt wurden.
Im Zuge der Evaluierung des Arbeitsschrittes zur Segmentierung linearer Punktmengen (Unterabschnitt 4.2.3) wurde festgestellt, dass der entwickelte Ansatz in der Lage ist, die Punktdaten von Trägern und Stützen in den Rahmensegmenten überwiegend präzise zu segmentieren. Dies zeigte sich insbesondere bei den Ergebnissen für die LYN-Punktwolke, bei der im ersten Durchlauf eine genaue Segmentierung der linearen Punktmengen erreicht wurde. Im teileingestürzten Bereich eines LYN-Rahmensegments kam es jedoch aufgrund der nicht rechteckigen Form der linearen Punktdaten zu einer Übersegmentierung in beiden Durchläufen. Bei den JDS- und GLC-Punktwolken stellten sich Herausforderungen heraus, die auf eine ungenaue Entfernung der Wandpunkte im vorherigen Schritt zurückzuführen waren. Dies führte zu inkorrekten Segmentierungen, die weder Träger noch Stützen darstellten. Die Optimierung der Randpunkt-Entfernung trug jedoch dazu bei, die Genauigkeit der Segmentierung im zweiten Durchlauf zu verbessern und die Verlässlichkeit der identifizierten Liniensegmente als Träger und Stützen zu erhöhen. Je nach Größe der Punktmenge führt der entwickelte Ansatz entweder zu einer kontinuierlichen Segmentierung der linearen Punktdaten eines Trägers oder einer Stütze. Dies erschwert eine konsistente Trennung der Punktdaten der Stützen nach Geschossen, was in den Ergebnissen der JDS- und GLC-Punktwolken besonders deutlich wurde. Darüber hinaus wurde festgestellt, dass die Segmentierung linearer Punktmengen in Bereichen, in denen die Punktdaten aufgrund von Verdeckungen unvollständig waren, zu ungenauen Ergebnissen führte. Insgesamt beanspruchte der angewendete Ansatz der linearen Punktsegmentierung in allen Durchläufen der Datensätze nicht mehr als 2,5 Minuten.
Im nächsten Schritt wurde die Objektklassifizierungsmethode evaluiert (Abschnitt 4.3). Insgesamt führte die entwickelte Methode zu einer präzisen Identifizierung der berücksichtigten Objekte in weniger als 0,21 Minuten. Von den verarbeiteten Segmenten wurden durchschnittlich 94% in der LYN-Punktwolke, 99% in der JDS-Punktwolke und 92% in der GLC-Punktwolke korrekt klassifiziert. Dies gilt insbesondere für die Klassifizierung von Decken und Wänden. Bei der Klassifizierung von Trägern und Stützen wurden aufgrund von geometrischen Ungenauigkeiten geringere Genauigkeitswerte erzielt. In diesem Zusammenhang wurden einige Punktsegmente von Trägern auf Basis ihrer geometrischen Merkmalen fälschlicherweise Wänden zugeordnet. In anderen Fällen wurden Punktsegmente von Stützen aufgrund fehlender Punktdaten oder Ungenauigkeiten bei der Anwendung der RE-Methode entweder der falschen Objektklasse zugewiesen oder als *Unclassified* gekennzeichnet. Weiterhin wurden Deckenränder in der GLC-Punktwolke fälschlicherweise als Träger klassifiziert, was sich negativ auf die Genauigkeit auswirkte. Verglichen mit den *Ground Truths* konnte die Objektklassifizierungsmethode für LYN, JDS und GLC im Durchschnitt mehr als 80% der Objekte richtig klassifizieren. Dieses Ergebnis zeigt, dass die Methode eine hohe Leistungsfähigkeit bei der Klassifizierung von Objekten einer seismisch beschädigten Stahlbetonrahmenkonstruktion aufweist.
Die Ergebnisse aus dem Arbeitsschritt der Materialklassifizierung wurden nach dem gleichen Konzept wie bei der Objektklassifizierung bewertet (Abschnitt 4.4). Hierbei wurden die Materialklas-

sen der Segmente in weniger als 0,10 Minuten mit dem vortrainierten DL-Modell vorhergesagt. Für die Segmente der LYN-Punktwolke konnte die Materialklassifizierung mit hoher Genauigkeit durchgeführt werden, wobei im Durchlauf #1 eine durchschnittliche Gesamtgenauigkeit von 87% und im Durchlauf #2 von 79% erzielt wurde. Aufgrund falsch klassifizierter Mauerwerkswände wies die Materialklassifizierung der Segmente der JDS-Punktwolke eine geringe durchschnittliche Gesamtgenauigkeit von 45% bzw. 46% auf. Diese Ergebnisse offenbarten eine Einschränkung in der Zuverlässigkeit der Materialklassifizierungsmethode, die in stark beschatteten Bereichen auftrat, in denen die Materialfarbe der Mauerwerkswände aufgrund unzureichender Beleuchtung nicht deutlich sichtbar war. Im Fall von GLC wurden in beiden Durchläufen durchschnittlich 79% bzw. 90% der Baumateralien richtig vorhergesagt.
Die Evaluierung des vorgeschlagenen Ansatzes zur digitalen 3D-Rekonstruktion (Abschnitt 4.5) verdeutlichte, dass ein konstanter α-Wert nicht ausreicht, um die Oberflächen aller Bauteile anhand ihrer Punktdaten genau zu rekonstruieren. Darüber hinaus stellen unvollständige Punktdaten im Kontext der digitalen Oberflächenrekonstruktion eine zusätzliche Herausforderung dar, die mit einer konvexen Oberflächenerzeugung nur begrenzt gelöst werden kann. Die Berechnungszeit für die digitale 3D-Rekonstruktion betrug 3,2 Minuten für LYN, 3,1 Minuten für JDS und 1,7 Minuten für GLC. Eine potentielle Lösung zur Verbesserung der digitalen 3D-Rekonstruktionsmethode besteht darin, den α-Wert automatisch für jedes Punktsegment in Abhängigkeit von der Segmentform anzupassen.
Im letzten Schritt der Evaluierung wurde die Leistungsfähigkeit des vorgeschlagenen *Mesh-to-IFC*-Ansatzes zur Generierung von BIM-Modellen für die gewählten Stahlbetonrahmentragwerke untersucht (Abschnitt 4.6). Die Methode erwies sich als äußerst effizient und war in der Lage, alle digital rekonstruierten Gebäudekomponenten aus den LYN-, JDS- und GLC-Datensätzen in jeweils weniger als einer Minute in das IFC-Format zu konvertieren. Die generierten IFC-Dateien wurden erfolgreich in verschiedenen BIM-Programmen, darunter *BIMVision* [1], *FreeCAD* [64] und *Revit* [14], importiert und validiert. Dabei konnten alle erzeugten IFC-Objekte und die zugehörigen Attribute und Eigenschaftssätze vollständig erfasst werden. Die Analyse der zugewiesenen IFC-Klassen und Eigenschaftssätze ergab eine korrekte Zuordnung der Parameterwerte gemäß den im BC-Objekt enthaltenen Daten.
Von der Vorverarbeitung der 3D-Punktwolken bis zur Fertigstellung der IFC-Modelle dauerte der gesamte Prozess 16 Minuten für LYN, 20 Minuten für JDS und 14 Minuten für GLC.

Kapitel 5

Zusammenfassung und Ausblick

Die vorliegende Dissertation befasste sich mit der Entwicklung eines modellunabhängigen Ansatzes zur Erstellung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen. Dabei wurden die in Kapitel 1 genannten Forschungsfragen im Detail untersucht. In diesem abschließenden Kapitel werden die wesentlichen Ergebnisse dieser Arbeit zusammengefasst und ein Ausblick auf mögliche Weiterentwicklungen in diesem Forschungsbereich gegeben.

5.1 Zusammenfassung der Arbeit

Die entwickelte modellunabhängige Methodik zur Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmentragwerken basiert auf der algorithmischen Verarbeitung einer 3D-Punktwolke des Tragwerkes, die von außen mittels Photogrammetrie oder Laserscanning erstellt wurde. Die Datenverarbeitung besteht aus sechs aufeinanderfolgenden Arbeitsschritten, wobei jeder Schritt spezifische Methoden beinhaltet. Diese Schritte sind im Folgenden kurz zusammengefasst:

1. Vorverarbeitung: Der erste Schritt befasst sich mit der Vorverarbeitung der unstrukturierten Punktdaten der 3D-Punktwolke. Dabei werden verschiedene Maßnahmen ergriffen, darunter die Referenzierung der Punktdaten anhand von GPS-Koordinaten, das Ausschneiden nicht benötigter Bereiche, das *Downsampling* und die Entfernung von Ausreißerpunkten. Aufgrund der benutzerfreundlichen grafischen Oberfläche und der Programmierschnittstelle ist die *Open-Source*-Software *CloudCompare* [40] für die Durchführung der Vorverarbeitung besonders gut geeignet.
2. Segmentierung: Dieser Arbeitsschritt zielt darauf ab, ebene (Decken, Wände) und linienförmige (Träger, Stützen) Bauteile innerhalb der 3D-Punktwolke der Stahlbetonrahmenkonstruktion zu segmentieren. Dabei kommt der etablierte RANSAC-Algorithmus [149] zur automatischen Segmentierung von ebenen Punktmengen zum Einsatz. Die Dicke der Bauteile wird durch die Analyse vorhandener paralleler Segmentpaare bestimmt. Für die Segmentierung der Punktdaten von Stützen und Trägern aus den Rahmensegmenten wird ein eigens entwickelter Ansatz genutzt, der auf dem RANSAC-Algorithmus in $\mathbb{R}^2$ basiert. Darüber hinaus wird ein iterativer Ansatz unter Verwendung der α-*Shape*-Methode [54, 179] vorgeschlagen, um Wandpunktdaten aus Rahmensegmenten zu segmentieren. Dieses Verfahren wird als Randpunkt-Entfernung (RE) bezeichnet.
3. Objektklassifizierung: In diesem Schritt werden die segmentierten Punktdaten auf der Grundlage ihrer Form und Orientierung strukturellen oder nicht-strukturellen Objektklassen zugeordnet. Hierbei werden die Punktsegmente als *Slab*, *Wall*, *Beam*, *Column*, *Base*, *Ground* oder *Unclassified* klassifiziert. In Fällen, in denen eine eindeutige Klassifizierung

anhand der geometrischen Merkmale nicht möglich ist, werden mehrdeutige Objektklassen wie *Slab | Wall* oder *Beam | Column* zugewiesen.

4. Materialklassifizierung: Neben der Objektklassifizierung wurde auch eine Materialklassifizierung für die semantische Informationsanreicherung implementiert. Ein trainiertes DNN, das auf DL basiert, sagt die Materialklasse (*Concrete* oder *Masonry*) eines Punktsegments auf der Grundlage ihres vorherrschenden RGB-Wertes voraus. Das trainierte DL-Modell erzielte auf dem Testdatensatz eine bemerkenswerte durchschnittliche Vorhersagegenauigkeit von 98%.
5. Digitale 3D-Rekonstruktion: Für die digitale Oberflächenrekonstruktion wird die *α-Shape*-Methode [54, 179] verwendet, um aus den segmentierten Punktdaten konvexe und konkave *Meshes*, bestehend aus Dreiecksflächen, zu erzeugen. Die rekonstruierten Oberflächen der klassifizierten Punktsegmente werden anschließend zu geschlossenen Körpern erweitert. Dies wird durch die Oberflächenextrusion entlang der Normalen unter Berücksichtigung der gespeicherten Komponentendicke des Segments durchgeführt.
6. BIM-Generierung: Schließlich werden im letzten Schritt die digital rekonstruierten Gebäudekomponenten vom OBJ- ins IFC-Format umgewandelt. Hierbei wird der *Mesh-to-IFC*-Prozess angewendet, mit der *IfcFacetedBrep*-Objekte basierend auf OBJ-Daten generiert werden. Diese bilden die Grundlage für IFC-Entitäten, denen semantische Informationen, wie die Objekt- und Materialklasse, automatisch zugewiesen werden. IFC-Klassen, GUIDs und eindeutige Kennzeichnungen werden entsprechend den buildingSMART-Definitionen [29] hinzugefügt. Die benutzerdefinierten IFC-Eigenschaftssätze *Geometry*, *Material*, *Damage* und *Actions* ermöglichen eine umfassende Beschreibung der Bauteile im Kontext von USaR-Maßnahmen.

Die Schritte der entwickelten Methodik wurden mit der Programmiersprache Python umgesetzt. Zu diesem Zweck wurde die Klasse BC objektorientiert implementiert. Diese Klasse dient als strukturierte Datenbank, in der die Ergebnisse der einzelnen Arbeitsschritte systematisch abgelegt werden. Die Implementierung der BC-Klasse ermöglicht somit eine flexible und übersichtliche Verwaltung der aus der 3D-Punktwolke extrahierten Daten und Informationen.
Die oben genannten Schritte 2 bis 5 wurden als eigenständige Algorithmen implementiert, wobei ausschließlich *Open-Source*-Python-Bibliotheken zum Einsatz kamen. Die einzige Ausnahme bildet die digitale Oberflächenrekonstruktion, für die *Wolfram Mathematica* [174] aufgrund seiner Leistungsfähigkeit und Effizienz verwendet wurde. Die entsprechenden Pseudocodes der erstellten Algorithmen sind im Anhang aufgeführt. Von den implementierten Algorithmen können der RANSAC-Algorithmus für die Segmentierung ebener Punktmengen, die Materialklassifizierung und der *Mesh-to-IFC*-Prozess vollautomatisch ausgeführt werden. Alle anderen Algorithmen, d. h. die Segmentierung linearer Punktmengen, die RE-Methode, die Objektklassifizierung und die 3D-Rekonstruktion, sind parameterabhängig.

Die entwickelte Methodik wurde anhand von drei realen Fallbeispielen evaluiert: einem teilweise eingestürzten Stahlbetonrahmengebäude in Lyon (LYN) sowie seismisch beschädigten Stahlbetonrahmenkonstruktionen in Jindires (JDS) und Gölcük (GLC). Die Evaluierung der entwickelten Methodik auf der Grundlage dieser Fallbeispiele ergab die folgenden Erkenntnisse:

Um die Genauigkeit der Segmentierung zu erhöhen, schlägt der Autor dieser Arbeit vor, ein teilweise eingestürztes Gebäude nach Möglichkeit in einen intakten und einen eingestürzten Bereich aufzuteilen und die Ergebnisse am Ende zusammenzuführen. Die Anwendung des RANSAC-Algorithmus zur Segmentierung von ebenen Punktmengen erwies sich als präzise und effizient. Dennoch wurden manuelle Parameteranpassungen vorgenommen, um die Segmentierungsgenauigkeit zu verbessern, insbesondere in komplexen Bereichen wie Trümmerhaufen, in denen eine

Übersegmentierung festgestellt wurde. Es wird empfohlen, den RANSAC-Algorithmus zunächst mit automatischer Parametereinstellung auszuführen und bei Bedarf auf Grundlage der erhaltenen Ergebnisse manuelle Anpassungen vorzunehmen.
Die entwickelte RE-Methode zur Segmentierung von Wandpunktdaten aus Rahmensegmenten erzielte vielversprechende Ergebnisse, zeigte jedoch Ungenauigkeiten bei unregelmäßigen Geometrien von beschädigten Mauerwerkswänden. In diesem Zusammenhang führten unvollständige Wandpunktdaten zwischen Stützen und Trägern zu einer ungenauen Formextraktion mit der α-*Shape*-Methode. Darüber hinaus ist es mit der RE-Methode nicht möglich, Punktdaten von Stützen innerhalb eines ausgeschnittenen Wandsegments zu berücksichtigen. Dies hat zur Folge, dass Punktdaten von Stützen fälschlicherweise einem Wandsegment zugeordnet werden können.
Der entwickelte Ansatz für die Segmentierung linearer Punktmengen aus Rahmensegmenten lieferte angemessene Resultate. Die Genauigkeit variierte je nach Datensatz, wobei die linearen Punktdaten in der LYN-Punktwolke aufgrund ihrer Einfachheit und Vollständigkeit besonders präzise segmentiert wurden. In einigen Fällen kam es jedoch zu einer Übersegmentierung und zu Problemen bei der geschossweisen Trennung von Stützenpunktdaten. Eine Optimierung der Randpunkt-Entfernung mittels Parameteranpassung verbesserte insgesamt die Genauigkeit der linearen Punktsegmentierung. Schwierigkeiten traten in Bereichen mit unvollständigen Punktdaten aufgrund von Verdeckungen auf. Die festgestellten Anwendungsgrenzen des Verfahrens beziehen sich insbesondere auf die konsistente, geschossweise Trennung von Stützenpunktdaten, die präzise Segmentierung von Stützen und Trägern mit unterschiedlichen Abmessungen und den Umgang mit unvollständigen Punktdaten.
Die geometriebasierte Objektklassifizierungsmethode ermöglichte eine effiziente und genaue Identifizierung von Objekten, wobei in allen drei Fallbeispielen durchschnittlich mehr als 80% der Gebäudekomponenten korrekt klassifiziert wurden. Allerdings kam es aufgrund geometrischer Ungenauigkeiten zu Fehlklassifizierungen von Trägern und Stützen. Insgesamt zeigte die Methode eine herausragende Leistung bei der Identifizierung von Komponenten einer beschädigten Stahlbetonrahmenstruktur. Insbesondere Decken und Wände wurden äußerst zuverlässig klassifiziert. Dennoch sind Verbesserungen der RE-Methode und der Segmentierung von linearen Punktmengen erforderlich, um die präzise Klassifizierung von Trägern und Stützen zu optimieren.
Die Leistung des DL-Modells für die Materialklassifizierung erwies sich als konsistent, da sich die Vorhersagen in zwei aufeinanderfolgenden Durchläufen kaum unterschieden. Mit dem DL-Modell wurden die Materialklassen der LYN- und JDS-Komponenten im Durchschnitt zu 87% bzw. 90% richtig vorhergesagt. Im Gegensatz dazu konnten für den JDS-Datensatz durchschnittlich nur 46% der Materialien richtig bestimmt werden, was auf falsch klassifizierte Mauerwerkswände zurückzuführen war. Dabei wurde festgestellt, dass das DL-Modell besonders gut bei der Identifizierung von Materialien funktioniert, wenn die tatsächliche Materialfarbe deutlich erkennbar ist. Dies war für mehrere Mauerwerkswände im JDS-Datensatz aufgrund von stärkeren Beschattungen nicht gegeben.
Die Leistungsfähigkeit des vorgeschlagenen Ansatzes für die digitale 3D-Rekonstruktion erwies sich als unzureichend, da die Implementierung einen konstanten α-Wert für die Oberflächenrekonstruktion von Punktsegmenten mit unterschiedlichen Formen vorsieht. Zur Verbesserung der Genauigkeit wurde vom Autor dieser Arbeit vorgeschlagen, den α-Wert für jedes Punktsegment automatisch und individuell einzustellen. Zu diesem Zweck wurde der Effekt der optimalen Parametereinstellung für die 3D-Rekonstruktion durch die manuelle Einstellung des α-Wertes für die Punktsegmente demonstriert, wobei eine hohe Genauigkeitssteigerung zu beobachten war. Die Evaluierung unterstreicht das vielversprechende Potenzial der 3D-Rekonstruktionsmethode und betont gleichzeitig die Notwendigkeit weiterer Forschungsbemühungen zur Verbesserung der Genauigkeit, insbesondere durch formabhängige Anpassung des α-Wertes für jedes Punktsegment.

Die BIM-Generierung erfolgte mit Hilfe des *Mesh-to-IFC*-Ansatzes, der eine effiziente Datenumwandlung ermöglichte. Die generierten IFC-Dateien wurden erfolgreich in verschiedene BIM-Programme importiert und validiert, was die Effektivität des angewandten Prozesses und die Interoperabilität der generierten BIM-Daten unterstreicht. Allerdings wiesen die Modelle aufgrund der unvollständigen 3D-Punktwolken einen begrenzten Detaillierungsgrad (LoD 200) auf. Die entwickelte Methodik ermöglicht daher nur die halbautomatische Generierung von BIM-Modellen von seismisch beschädigten Stahlbetonrahmenkonstruktionen mit grundlegenden Bauteilinformationen. Die gesamte Rechenzeit von der Vorverarbeitung der 3D-Punktwolke bis zur Umwandlung in das IFC-Format betrug 16 Minuten für LYN, 20 Minuten für JDS und 14 Minuten für GLC.

Zusammenfassend wurde im Rahmen dieser Dissertation ein vielversprechender, modellunabhängiger Ansatz entwickelt, der unter Berücksichtigung bestimmter Anwendungsgrenzen die halbautomatische Generierung eines BIM-Modells (LoD 200) von seismisch beschädigten Stahlbetonrahmenkonstruktionen auf Basis einer 3D-Punktwolke ermöglicht. Dieser Beitrag markiert einen bedeutenden Fortschritt auf dem Gebiet der *Scan-to-BIM* und bietet entscheidende Impulse für die automatisierte Erstellung von BIM-Modellen in Katastrophensituationen.

5.2 Ausblick

Die vorliegende Dissertation legt den Grundstein für die modellunabhängige Generierung eines BIM-Modells einer seismisch beschädigten Stahlbetonrahmenkonstruktion auf Basis einer 3D-Punktwolke. Trotz des erzielten Fortschrittes zeigen die Ergebnisse dieser Arbeit mehrere Aspekte auf, die für die zukünftige Forschung und Weiterentwicklung dieses Ansatzes von entscheidender Bedeutung sind.
Eine Forschungsperspektive liegt in der Optimierung der Segmentierung und Klassifizierung von Punktdaten einer seismisch beschädigten Stahlbetonrahmenkonstruktion im Hinblick auf unregelmäßige Geometrien. Zukünftige Forschungsbemühungen könnten darauf abzielen, die Segmentierung und Klassifizierung der Gebäudekomponenten aus einer 3D-Punktwolke mit Hilfe von maschinellem Lernen zu verbessern. Hierbei bietet sich der Einsatz der semantischen Segmentierung mit etablierten 3D-CNNs an. Die Umsetzung dieser Methode erfordert jedoch die Erstellung eines geeigneten Trainingsdatensatzes, was die bestehende Herausforderung in diesem Bereich unterstreicht.
Um die Zuverlässigkeit der DL-basierten Materialklassifizierungsmethode zu erhöhen, insbesondere bei stark beschatteten Bauteilen, könnte ein vielversprechender Ansatz darin bestehen, alle RGB-Werte im Punktsegment in die Materialklassenvorhersage einzubeziehen. Zusätzlich könnte die Berücksichtigung von Texturinformationen aus Bilddaten als weiteres Merkmal erfolgen.
Ein weiterer Aspekt zur Verbesserung der Methodik könnte die Integration einer formabhängigen α-Wert-Einstellung im Kontext der digitalen Oberflächenrekonstruktion mittels der α-*Shape*-Methode sein. Darüber hinaus kann die Robustheit des modellunabhängigen Ansatzes anhand zusätzlicher realer Fallbeispiele untersucht werden. In diesem Zusammenhang könnte die Anpassung der Methodik auf andere Gebäudearten und Materialien die Anwendbarkeit auf unterschiedliche Konstruktionen erhöhen und somit einen breiteren Einsatz ermöglichen.
Durch die Implementierung von Algorithmen der künstlichen Intelligenz (KI) zur Schadensklassifizierung, insbesondere zur Identifizierung von Rissen oder Verformungen, könnte das BIM-Modell mit statisch-relevanten Informationen angereichert werden. Dazu kann der Einsatz von KI-gestützter Bildklassifikation, z. B. aus Drohnenbildern, in Kombination mit den entsprechenden Punktsegmenten untersucht werden. Die Genauigkeit der Schadensanalyse könnte durch die Einbindung verschiedener Sensoren, wie z. B. eines Beschleunigungsmessers, RADAR oder akus-

tischer Sensoren, erhöht werden.
Ein weiterer vielversprechender Forschungsausblick wäre die Entwicklung einer Methode, die automatisch das generierte BIM-Modell eines beschädigten Stahlbetonrahmentragwerkes in ein berechenbares FE-Modell umwandelt. Dieser Schritt würde es ermöglichen, lokale Instabilitäten der Tragstruktur numerisch und objektiv abzuschätzen. Der Einsatz einer solchen Methode könnte dazu beitragen, die Sicherheit von Rettungskräften zu erhöhen und damit Such- und Rettungsaktionen effektiver und sicherer zu gestalten.
Eine zentrale Herausforderung bei der automatischen Generierung von BIM-Modellen aus 3D-Punktwolken ist der Umgang mit Verdeckungen. Insbesondere nach Naturkatastrophen können Teile der Struktur durch Fassaden, Trümmer oder andere Hindernisse verdeckt sein, was die digitale Erfassung von außen erheblich beeinträchtigt. Die Vollständigkeit der Punktwolke beeinflusst dabei die Gesamtgenauigkeit der digitalen Rekonstruktion eines Gebäudes. Zukünftige Entwicklungen könnten sich auf die Implementierung generativer KI-Techniken wie *Generative Adversarial Networks* oder *Variational Autoencoders* konzentrieren, um fehlende oder unvollständige Daten in der Punktwolke zu ergänzen und so eine detailliertere Rekonstruktion in verdeckten Bereichen zu ermöglichen.
Eine alternative Methode für die digitale Rekonstruktion von verdeckten Bauteilen ohne den Einsatz komplexer KI-Methoden wurde in dieser Arbeit vorgeschlagen. Dabei wird die geometrische Übereinstimmung zwischen der digital rekonstruierten Stahlbetonrahmenkonstruktion und Schadensmodellen aus einer vordefinierten Datenbank überprüft. Um die Leistungsfähigkeit dieser Methode zu bewerten, bedarf es einer umfassenden Evaluation. In diesem Zusammenhang ist die Erstellung einer geeigneten Datenbank mit repräsentativen Schadensmodellen essentiell.
Die kontinuierliche Zusammenarbeit zwischen Forschern, Praktikern und Entwicklern ist entscheidend, um die Herausforderungen im Bereich der *Scan-to-BIM* weiter anzugehen und innovative Lösungen für die Katastrophenbewältigung zu schaffen. Mit fortschreitender Sensortechnologie und KI-Entwicklung ist zu erwarten, dass in Zukunft die automatische Erstellung noch detaillierterer BIM-Modelle ermöglicht wird.

Literaturverzeichnis

[1] BIMvision. https://bimvision.eu/, zuletzt aufgerufen am 17.04.2023

[2] Center for Disaster Philanthropy: 2022 Afghanistan Earthquake, https://disasterphilanthropy.org/disasters/2022-afghanistan-earthquake/, aufgerufen am: 13.07.2022

[3] Center for Disaster Philanthropy: Haiti Earthquake and Tropical Storm Grace, https://disasterphilanthropy.org/disaster/2021-haiti-earthquake-and-tropical-storm-grace/, aufgerufen am: 12.07.2022

[4] HDF5: Hierarchical data format 5. https://www.hdfgroup.org/solutions/hdf5/, zuletzt aufgerufen am 30.03.2023

[5] Python documentation: pickle - python object serialization, https://docs.python.org/3/library/pickle.html, aufgerufen am: 21.12.2022

[6] Python documentation: scikit-image - image processing in python, https://scikit-image.org/, aufgerufen am: 08.01.2023

[7] Université catholique de Louvain: The International Disaster Database: Centre for Research on the Epidemiology of Disasters, https://public.emdat.be/, aufgerufen am: 12.07.2022

[8] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: TensorFlow. https://www.tensorflow.org/ (2015), zuletzt aufgerufen am 30.03.2023

[9] Adams, S.M., Levitan, M.L., Friedland, C.J.: High resolution imagery collection for post-disaster studies utilizing unmanned aircraft systems (UAS). Photogrammetric Engineering & Remote Sensing **80**(12), 1161–1168 (2014)

[10] AgiSoft: Agisoft metashape pro 64 version 1.5.5. (2019), http://www.agisoft.com/downloads/installer/

[11] Alanani, M., Ehab, M., Salem, H.: Progressive collapse assessment of precast prestressed reinforced concrete beams using applied element method. Case Studies in Construction Materials **13**, e00457 (2020)

[12] Albert, A., Schneider, K., Verlag, B.: Bautabellen für Ingenieure: mit Berechnungshinweisen und Beispielen. Bundesanzeiger Verlag (2018), https://books.google.de/books?id=GskytAEACAAJ

[13] Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S.: 3d semantic parsing of large-scale indoor spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1534–1543 (2016)

[14] Autodesk Inc.: Autodesk Revit (2023), https://www.autodesk.com/products/revit/, zuletzt aufgerufen am 17.04.2023

[15] Barazzetti, L.: Point cloud occlusion recovery with shallow feedforward neural networks. Advanced Engineering Informatics **38**, 605–619 (2018)

[16] Barazzetti, L., Banfi, F., Brumana, R., Gusmeroli, G., Previtali, M., Schiantarelli, G.: Cloud-to-BIM-to-FEM: Structural simulation with accurate historic BIM from laser scans. Simulation Modelling Practice and Theory **57**, 71–87 (2015)

[17] Bassier, M., Mattheuwsen, L., Vergauwen, M.: BIM Reconstruction: Automated Procedural Modeling from Point Cloud Data (December 2019)

[18] Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J.: SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In: Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV) (2019)

[19] Berger, M., Tagliasacchi, A., Seversky, L.M., Alliez, P., Guennebaud, G., Levine, J.A., Sharf, A., Silva, C.T.: A survey of surface reconstruction from point clouds. In: Computer Graphics Forum. vol. 36, pp. 301–329. Wiley Online Library (2017)

[20] Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., Taubin, G.: The ball-pivoting algorithm for surface reconstruction. IEEE transactions on visualization and computer graphics **5**(4), 349–359 (1999)

[21] Besl, P.J., McKay, N.D.: Method for registration of 3-d shapes. In: Sensor fusion IV: control paradigms and data structures. vol. 1611, pp. 586–606. Spie (1992)

[22] Bhanu, B., Lee, S., Ho, C.C., Henderson, T.: Range data processing: Representation of surfaces by edges. In: Proceedings of the eighth international conference on pattern recognition. pp. 236–238. IEEE Computer Society Press Piscataway, NJ, USA (1986)

[23] Blender: Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam (2021), http://www.blender.org

[24] BlenderBIM: Blenderbim documentation. https://blenderbim.org/docs/blenderbim/scene.html (2021)

[25] Bloch, T., Sacks, R., Rabinovitch, O.: Interior models of earthquake damaged buildings for search and rescue. Advanced Engineering Informatics **30**(1), 65–76 (2016)

[26] Boissonnat, J.D.: Geometric structures for three-dimensional shape representation. ACM Transactions on Graphics (TOG) **3**(4), 266–286 (1984)

[27] Borrmann, D., Elseberg, J., Lingemann, K., Nüchter, A.: The 3d hough transform for plane detection in point clouds: A review and a new accumulator design. 3D Research **2**(2), 1–13 (2011)

[28] Brito, J.: Occlusion detection in digital images through bayesian networks. International Archives of Photogrammetry and Remote Sensing **33**(B3/1; PART 3), 101–108 (2000)

[29] buildingSMART: buildingSMART Documentation for IFC4. https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/ (2020)

[30] Camurri, M., Vezzani, R., Cucchiara, R.: 3d hough transform for sphere recognition on point clouds. Machine vision and applications **25**(7), 1877–1891 (2014)

[31] Center for Disaster Philanthropy: 2023 turkey-syria earthquake (2023), https://disasterphilanthropy.org/disasters/2023-turkey-syria-earthquake/, zugriff am 22. Juni 2023

[32] Chandel, H., Vatta, S.: Occlusion detection and handling: a review. International Journal of Computer Applications **120**(10) (2015)

[33] Chen, A.Y., Huang, Y.N., Han, J.Y., Kang, S.C.J.: A review of rotorcraft unmanned aerial vehicle (UAV) developments and applications in civil engineering. Smart Struct. Syst **13**(6), 1065–1094 (2014)

[34] Chen, X., Golovinskiy, A., Funkhouser, T.: A benchmark for 3d mesh segmentation. Acm transactions on graphics (tog) **28**(3), 1–12 (2009)

[35] Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. Image and vision computing **10**(3), 145–155 (1992)

[36] Choe, J., Joung, B., Rameau, F., Park, J., Kweon, I.S.: Deep point cloud reconstruction. arXiv preprint arXiv:2111.11704 (2021)

[37] Chollet, F.: Deep Learning with Python. Manning Publications (2018)

[38] Chollet, F., et al.: Keras. https://keras.io/ (2015), zuletzt aufgerufen am 30.03.2023

[39] Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: MeshLab: an Open-Source Mesh Processing Tool. In: Scarano, V., Chiara, R.D., Erra, U. (eds.) Eurographics Italian Chapter Conference. The Eurographics Association (2008). https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136

[40] CloudCompare: Cloudcompare (version 2.11.3) [gpl software] (2020), http://www.cloudcompare.org/

[41] Coburn, A.W., Spence, R.J., Pomonis, A.: Factors determining human casualty levels in earthquakes: mortality prediction in building collapse. In: Proceedings of the tenth world conference on earthquake engineering. vol. 10, pp. 5989–5994. Balkema Rotterdam (1992)

[42] Cohen-Or, D., Kaufman, A.: Fundamentals of surface voxelization. Graphical models and image processing **57**(6), 453–461 (1995)

[43] Concrete Buildings Damaged in Earthquakes: A Collection of Case Studies: Residential Concrete Building in Gölcük, Turkey, damaged by the Kocaeli earthquake in 1999 (2022), http://db.concretecoalition.org/building/135, zugriff am 28. Juni 2023

[44] Conde-Carnero, B., Riveiro, B., Arias, P., Caamaño, J.: Exploitation of geometric data provided by laser scanning to create FEM structural models of bridges. Journal of Performance of Constructed Facilities **30**(3), 04015053 (2016)

[45] Coumans, E., Bai, Y.: Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org (2016–2021)

[46] Czerniawski, T., Leite, F.: Automated digital modeling of existing buildings: A review of visual object recognition methods. Automation in Construction (2020)

[47] Dawson-Haggerty et al.: trimesh, https://trimsh.org/

[48] DeepNotes.io: Softmax Cross-Entropy Loss: Why use it & how it helps? (2021), https://deepnotes.io/softmax-crossentropy

[49] Díaz-Vilariño, L., Martínez-Sánchez, J., Lagüela, S., Armesto, J., Khoshelham, K.: Door recognition in cluttered building interiors using imagery and lidar data. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci **40**(5), 203–209 (2014)

[50] Domaneschi, M., Cimellaro, G.P., Scutiero, G.: A simplified method to assess generation of seismic debris for masonry structures. Engineering Structures **186**, 306–320 (2019)

[51] Dong, L., Shan, J.: A comprehensive review of earthquake-induced building damage detection with remote sensing techniques. ISPRS Journal of Photogrammetry and Remote Sensing **84**, 85–99 (2013)

[52] Dore, C., Murphy, M.: Semi-automatic generation of as-built bim facade geometry from laser and image data. Electronic Journal of Information Technology in Construction (2014)

[53] Eastman, C., Fisher, D., Lafue, G., Lividini, J., Stoker, D., Yessios, C.: And outline of the building description system. Tech. rep., Institute of Physical Planning (1974)

[54] Edelsbrunner, H., Kirkpatrick, D., Seidel, R.: On the shape of a set of points in the plane. IEEE Transactions on information theory **29**(4), 551–559 (1983)

[55] EERI: Earthquake engineering research institute: Concrete buildings damaged in earthquakes (2022), http://db.concretecoalition.org/

[56] Elshaer, A., Mostafa, H., Salem, H.: Progressive collapse assessment of multistory reinforced concrete structures subjected to seismic actions. KSCE Journal of Civil Engineering **21**(1), 184–194 (2017)

[57] F. Bendimerad and L. Johnson and A. Coburn and M. Rahnama and G. Morrow: Event Report: Kocaeli, Turkey Earthquake. Tech. rep., Risk Management Solutions, Inc. (2000), https://www.preventionweb.net/files/2667_TurkeyEvent.pdf

[58] Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. International journal of computer vision **59**, 167–181 (2004)

[59] Ferworn, A., Herman, S., Tran, J., Ufkes, A., Mcdonald, R.: Disaster scene reconstruction: Modeling and simulating urban building collapse rubble within a game engine. In: Proceedings of the 2013 Summer Computer Simulation Conference. pp. 1–6 (2013)

[60] Ferworn, A., Tran, J., Ufkes, A., D'Souza, A.: Initial experiments on 3D modeling of complex disaster environments using unmanned aerial vehicles. In: 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics. pp. 167–171. IEEE (2011)

[61] Filin, S., Pfeifer, N.: Segmentation of airborne laser scanning data using a slope adaptive neighborhood. ISPRS journal of Photogrammetry and Remote Sensing **60**(2), 71–80 (2006)

[62] Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)

[63] Fita, J.L., Besuievsky, G., Patow, G.: Earthquake Simulation on Ancient Masonry Buildings. Journal on Computing and Cultural Heritage (JOCCH) **13**(2), 1–18 (2020)

[64] FreeCAD: Freecad documentation. https://wiki.freecadweb.org/Main_Page (2021)

[65] Fryskowska, A., Walczykowski, P., Delis, P., Wojtkowska, M.: ALS and TLS data fusion in cultural heritage documentation and modeling. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences **40**(5), 147 (2015)

[66] Fujiwara, T., Kamegawa, T., Gofuku, A.: Evaluation of plane detection with ransac according to density of 3d point clouds. arXiv preprint arXiv:1312.5033 (2013)

[67] Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R.: Towards internet-scale multi-view stereo. In: 2010 IEEE computer society conference on computer vision and pattern recognition. pp. 1434–1441. IEEE (2010)

[68] Furukawa, Y., Ponce, J.: Accurate, dense, and robust multi-view stereopsis (pmvs). In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. vol. 2, p. 3 (2007)

[69] Gebbeken, N., Krauthammer, T.: Natural Threat–Tornadoes. Bautechnik **93**(4), 243–253 (2016)

[70] Gebbeken, N., Videkhina, I., Pfeiffer, E., Garsch, M., Rüdiger, L.: Risikobewertung und Schutz von baulichen Infrastrukturen bei Hochwassergefahr. Bautechnik **93**(4), 199–213 (2016)

[71] Gebbeken, N., Braun, M., Hachmann, T., Yilmaz, H.: Earthquake Engineering–Reconnaissance and Assessment of Existing Buildings. International Journal of Protective Structures **3**(4), 375–388 (2012)

[72] Golovinskiy, A., Funkhouser, T.: Min-cut based segmentation of point clouds. In: 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops. pp. 39–46. IEEE (2009)

[73] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)

[74] Gordon, S.J., Lichti, D.D.: Modeling terrestrial laser scanner data for precise structural deformation measurement. Journal of surveying engineering **133**(2), 72–80 (2007)

[75] Grayson, B., Penna, N.T., Mills, J.P., Grant, D.S.: GPS precise point positioning for UAV photogrammetry. The Remote Sensing and Photogrammetry Society and John Wiley & Sons Ltd (2018). https://doi.org/https://doi.org/10.1111/phor.12259

[76] Grilli, E., Menna, F., Remondino, F.: A review of point clouds segmentation and classification algorithms. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences **42**, 339 (2017)

[77] Grunwald, C., Khalil, A.A., Schaufelberger, B., Ricciardi, E.M., Pellecchia, C., De Iuliis, E., Riedel, W.: Reliability of collapse simulation–comparing finite and applied element method at different levels. Engineering Structures **176**, 265–278 (2018)

[78] Hackel, T., Savinov, N., Ladicky, L., Wegner, J.D., Schindler, K., Pollefeys, M.: SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. vol. IV-1-W1, pp. 91–98 (2017)

[79] Haldar, P., Singh, Y., Paul, D.: Identification of seismic failure modes of URM infilled RC frame buildings. Engineering Failure Analysis **33**, 97–118 (2013)

[80] Heipke, C., Freeden, W., Rummel, R.: Photogrammetrie und Fernerkundung. Springer (2017)

[81] Hertle, T., Garsch, M., Gebbeken, N.: Die aktuellen Vorgehensweisen bei der Bauwerkspruefung und der Dokumentation sind heute nicht mehr zeitgemäss. Drohnen und Smartphones sind die neuen Werkzeuge für Bauwerksüberwachungen und die BIM-Modellierung. Prüfingenieur (53) (2018)

[82] Hinks, T., Carr, H., Truong-Hong, L., et al.: Point cloud data conversion into solid models via point-based voxelization. Journal of Surveying Engineering **139**(2), 72–83 (2013)

[83] Hohage, H.: Einsatztaktik bei Gebäudeschäden (2008)

[84] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. In: Proceedings of the 19th annual conference on computer graphics and interactive techniques. pp. 71–78 (1992)

[85] Huang, J., Menq, C.H.: Combinatorial manifold mesh reconstruction and optimization from unorganized points with arbitrary topology. Computer-Aided Design **34**(2), 149–165 (2002)

[86] Hulik, R., Spanel, M., Smrz, P., Materna, Z.: Continuous plane detection in point-cloud data based on 3d hough transform. Journal of visual communication and image representation **25**(1), 86–97 (2014)

[87] IfcOpenShell: IfcOpenShell: The open source IFC toolkit and geometry engine. http://ifcopenshell.org/ (2022)

[88] Ignatova, E., Kirschke, H., Tauscher, E., Smarsly, K.: Parametric geometric modeling construction planning using industry foundation classes (20-22 July 2015)

[89] INACHUS: Technological and Methodological Solutions for Integrated Wide Area Situation Awareness and Survivor Localisation to Support Search and Rescue Teams - European 7th Framework Project under grant agreement no. 607522. Tech. rep., European Union (2019), https://www.inachus.eu/

[90] International Organization for Standardization: ISO/CD 10303-42:1992: Industrial automation systems and integration – product data representation and exchange – part 42: Integrated generic resources: Geometric and topological representation. ISO/CD Standard 10303-42, ISO (1992)

[91] Isa, S.M., Shukor, S.A., Rahim, N., Maarof, I., Yahya, Z., Zakaria, A., Abdullah, A., Wong, R.: Point cloud data segmentation using ransac and localization. In: IOP Conference Series: Materials Science and Engineering. vol. 705, p. 012004. IOP Publishing (2019)

[92] Karabassi, E.A., Papaioannou, G., Theoharis, T.: A fast depth-buffer-based voxelization algorithm. Journal of graphics tools **4**(4), 5–10 (1999)

[93] Kashani, A.G., Crawford, P.S., Biswas, S.K., Graettinger, A.J., Grau, D.: Automated tornado damage assessment and wind speed estimation based on terrestrial laser scanning. Journal of Computing in Civil Engineering **29**(3), 04014051 (2015)

[94] Katz, S., Tal, A., Basri, R.: Direct visibility of point sets. In: ACM SIGGRAPH 2007 papers, pp. 24–es (2007)

[95] Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the fourth Eurographics symposium on Geometry processing. vol. 7 (2006)

[96] Kenny, C.: Why do people die in earthquakes? The costs, benefits and institutions of disaster risk reduction in developing countries. The Costs, Benefits and Institutions of Disaster Risk Reduction in Developing Countries (January 1, 2009). World Bank Policy Research Working Paper (4823) (2009)

[97] Kerle, N., Nex, F., Gerke, M., Duarte, D., Vetrivel, A.: Uav-based structural damage mapping: A review. ISPRS international journal of geo-information **9**(1), 14 (2019)

[98] Khanmohammadi, S., Arashpour, M., Bai, Y.: Applications of building information modeling (BIM) in disaster resilience: present status and future trends. In: ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction. vol. 37, pp. 1380–1387. IAARC Publications (2020)

[99] Khatamian, A., Arabnia, H.R.: Survey on 3d surface reconstruction. Journal of Information Processing Systems **12**(3), 338–357 (2016)

[100] Kostak, K.: Christchurch Earthquake Simulation and Pyne Gould Building Collapse - INACHUS (May 2020), https://www.youtube.com/watch?v=TCJq_2-q34k&ab_channel=KostackStudio

[101] Kostak, K.: Miami Condo Collapse Simulation - Champlain Towers South Surfside Florida (Jul 2021), https://www.youtube.com/watch?v=HPwJ0JvTcg8&ab_channel=KostackStudio

[102] Landsberg, L., Braun, A., Mudimu, O., Buttgen, K.D.: Considering end user needs when developing new technologies - a new plug and play sensor technology for locating trapped victims. Proceedings of the 18th ISCRAM Conference (2021)

[103] Lee, H., Jung, J.: Clustering-based plane segmentation neural network for urban scene modeling. Sensors **21**(24), 8382 (2021)

[104] Lim, S.P., Haron, H.: Surface reconstruction techniques: a review. Artificial Intelligence Review **42**(1), 59–78 (2014)

[105] Limberger, F.A., Oliveira, M.M.: Real-time detection of planar regions in unorganized point clouds. Pattern Recognition **48**(6), 2043–2053 (2015)

[106] Liu, X., Eybpoosh, M., Akinci, B.: Developing as-built building information model using construction process history captured by a laser scanner and a camera. In: Construction Research Congress 2012: Construction Challenges in a Flat World. pp. 1232–1241 (2012)

[107] Lockley, S., Benghi, C., Černý, M.: Xbim.Essentials: A Library for Interoperable Building Information Applications. The Journal of Open Source Software **2**(20), 473 (2017). https://doi.org/10.21105/joss.00473, http://joss.theoj.org/papers/b23bed93a0377b4f4317d0583b4d2c5e

[108] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2), 91–110 (2004)

[109] Lu, Q., Lee, S.: Image-based technologies for constructing as-is building information models for existing buildings. J. Comput. Civ. Eng. (2017)

[110] Lu, X., Guan, H., Sun, H., Li, Y., Zheng, Z., FEI, Y., Yang, Z., Zuo, L.: A preliminary analysis and discussion of the condominium building collapse in surfside, florida, us, june 24, 2021 (2021)

[111] Luhmann, T., Robson, S., Kyle, S., Harley, I.: Close Range Photogrammetry: Principles, Techniques and Applications. Whittles publishing Dunbeath (2011)

[112] Lupoae, M., Baciu, C., Constantin, D.: Theoretical and experimental research on progressive collapse of RC frame buildings. Urbanism. Arhitectura. Constructii **4**(3), 71 (2013)

[113] Ma, J.W., Czerniawski, T., Leite, F.: Semantic segmentation of point clouds of building interiors with deep learning: Augmenting training datasets with synthetic bim-based point clouds. Automation in Construction **113**, 103144 (2020)

[114] Ma, L., Sacks, R., Zeibak-Shini, R., Aryal, A., Filin, S.: Preparation of synthetic as-damaged models for post-earthquake bim reconstruction research. Journal of computing in civil engineering **30**(3), 04015032 (2016)

[115] Macher, H., Landes, T., Grussenmeyer, P.: Point clouds segmentation as base for as-built BIM creation. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences **2**(5), 191 (2015)

[116] Macher, H., Landes, T., Grussenmeyer, P.: From point clouds to building information models: 3d semi-automatic reconstruction of indoors of existing buildings. Applied Sciences **7**(10), 1030 (2017)

[117] Maguire, M., Dorafshan, S., Thomas, R.J.: SDNET2018: A concrete crack image dataset for machine learning applications (2018). https://doi.org/10.15142/T3TD19

[118] Marginean, I., Dinu, F., Dubina, D.: Simulation of the dynamic response of steel moment frames following sudden column loss. experimental calibration of the numerical model and application. Steel Construction **11**(1), 57–64 (2018)

[119] Moult, D.: BlenderBIM Add-on: obj2ifc-meshlab.py. https://github.com/IfcOpenShell/IfcOpenShell/blob/v0.7.0/src/blenderbim/scripts/obj2ifc-meshlab.py (2021), zuletzt aufgerufen am 10.04.2023

[120] Muntoni, A., Cignoni, P.: PyMeshLab (Jan 2021). https://doi.org/10.5281/zenodo.4438750

[121] Murray, J.D., VanRyper, W.: Encyclopedia of graphics file formats. egff (1996)

[122] Nagai, M., Chen, T., Shibasaki, R., Kumagai, H., Ahmed, A.: UAV-borne 3-D mapping system by multisensor integration. IEEE Transactions on Geoscience and Remote Sensing **47**(3), 701–708 (2009)

[123] Neptune.ai: Keras Loss Functions: A Practical Guide to Selection (2021), https://neptune.ai/blog/keras-loss-functions

[124] Nex, F., Duarte, D., Tonolo, F.G., Kerle, N.: Structural building damage detection with deep learning: Assessment of a state-of-the-art cnn in operational conditions. Remote sensing **11**(23), 2765 (2019)

[125] Nguyen, A., Le, B.: 3d point cloud segmentation: A survey. In: 2013 6th IEEE conference on robotics, automation and mechatronics (RAM). pp. 225–230. IEEE (2013)

[126] Nielsen, M.: Neural Networks and Deep Learning. Determination Press (2015)

[127] O'Keeffe, S., Bosche, F.: The need for convergence of bim and 3d imaging in the open world. CITA BIM Gatherings (November 2015)

[128] Oliver, W., Kostack, K.: Final Relase of the Blender and Bullet Physics Engine based on fast on-site assessment tool. Tech. rep., INACHUS (2017), https://github.com/KaiKostack/bullet-constraints-builder

[129] Omar Haj Kadour: Aerial Shots of quake destruction, search operations in Syria's Jandairis. Manila Bulletin Online (February 2023), https://www.youtube.com/watch?v=-XwFq2b7zPc&ab_channel=ManilaBulletinOnline

[130] OpenStreetMap contributors: Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org (2017)

[131] Pătrăucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I., Haas, C.: State of research in automatic as-built modelling. Advanced Engineering Informatics **29**(2), 162–171 (2015)

[132] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: scikit-learn: Machine learning in Python. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html (2011), zuletzt aufgerufen am 30.03.2023

[133] Pierdicca, R., Paolanti, M., Matrone, F., Martini, M., Morbidoni, C., Malinverni, E.S., Frontoni, E., Lingua, A.M.: Point cloud semantic segmentation using a deep learning framework for cultural heritage. Remote Sensing **12**(6), 1005 (2020)

[134] Project, I.U.: Pilot 2: Building partial collapse modelisation (2017), https://www.youtube.com/watch?v=4VF8gb3VYKg&ab_channel=INACHUSUSaRProject

[135] Przybilla, H.J., Reuber, C., Bäumker, M., Gerke, M.: Untersuchungen zur Genauigkeitssteigerung von UAV-Bildflügen. Publikationen der Deutschen Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation e. V **24**, 45–54 (2015)

[136] Quaritsch, M., Kuschnig, R., Hellwagner, H., Rinner, B., Adria, A., Klagenfurt, U.: Fast aerial image acquisition and mosaicking for emergency response operations by collaborative UAVs. In: ISCRAM (2011)

[137] Raguram, R., Frahm, J.M., Pollefeys, M.: A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In: European conference on computer vision. pp. 500–513. Springer (2008)

[138] Rahimi, Amar; Gebbeken, N.: Towards digital twin generation of collapsed buildings – use of new sensors and digital methods to support search and rescue efforts. In: Wohlgemuth, Volker; Naumann, S.A.H.K.B.G. (ed.) EnviroInfo 2021 : Environmental Informatics - a bogeyman or saviour to achieve the UN Sustainable Development Goals? Adjunct Proceedings of the 35th edition of the EnviroInfo – the long standing and established international and interdisciplinary conference series on leading environmental information and communication technologies. pp. 173–180. Shaker Verlag, Düren (2021). https://doi.org/10.2370/9783844083293

[139] Rahimi, Amar; Gebbeken, N.: Erstellung von Building Information Models von Trümmerfeldern basierend auf 3D-Punktwolken. In: Sensoren und Messsysteme : Beiträge der 21. ITG/GMA-Fachtagung 10. – 11. Mai 2022 in Nürnberg. pp. 420–425. VDE Verlag, Berlin ; Offenbach (2022), https://www.ama-science.org/direct/tagungsband-sensoren-und-messsysteme-2022

[140] Rakotosaona, M.J., Guerrero, P., Aigerman, N., Mitra, N.J., Ovsjanikov, M.: Learning delaunay surface elements for mesh reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22–31 (2021)

[141] RECONASS: Reconstruction and REcovery Planning: Rapid and Continuously Updated COnstruction Damage, and Related Needs ASSessment - European 7th Framework Project under grant agreement no. 312718. Tech. rep., European Union (2012), http://reconass.eu/

[142] Reshetyuk, Y.: Direct georeferencing with GPS in terrestrial laser scanning. Zeitschrift für Geodäsie, Geoinformation und Landmanagement **135**(3), 151–159 (2010)

[143] Revit AutoDesk, Inc.: Revit IFC manual: Detailed instructions for handling IFC files. Revit AutoDesk, Inc. (2020), https://damassets.autodesk.net/content/dam/autodesk/draftr/2528/180213_IFC_Handbuch.pdf

[144] Riviere, N., Amditis, A., Amiez, A., Athanasiou, G., Berggren, J., Boulch, A., Bozabalian, N., Duarte, D., Dupouy, P.E., Escalas, P., et al.: 3D laser imaging techniques to improve USaR operations for wide-area surveillance and monitoring of collapsed buildings (2017)

[145] Rolin, R., Antaluca, E., Batoz, J.L., Lamarque, F., Lejeune, M.: From point cloud data to structural analysis through a geometrical hbim-oriented model. Journal on Computing and Cultural Heritage (JOCCH) **12**(2), 1–26 (2019)

[146] Ruan, X., Liu, B.: Review of 3d point cloud data segmentation methods. International Journal of Advanced Network, Monitoring and Controls **5**(1), 66–71 (2020)

[147] RunnyKine, U.: Delaunay Mesh in a specified closed region: creating a concave hull from a set of points in Mathematica (2015), https://mathematica.stackexchange.com/questions/88752/delaunaymesh-in-a-specified-closed-region-creating-a-concave-hull-from-a-set-o, zuletzt aufgerufen am 06.02.2023

[148] Salem, H., Mohssen, S., Nishikiori, Y., Hosoda, A.: Numerical collapse analysis of Tsuyagawa bridge damaged by Tohoku tsunami. Journal of Performance of Constructed Facilities **30**(6), 04016065 (2016)

[149] Schnabel, R., Wahl, R., Klein, R.: Efficient ransac for point-cloud shape detection. In: Computer graphics forum. vol. 26, pp. 214–226. Wiley Online Library (2007)

[150] Sediek, O.A., El-Tawil, S., McCormick, J.: Seismic Debris Field for Collapsed RC Moment Resisting Frame Buildings. Journal of Structural Engineering **147**(5), 04021045 (2021)

[151] Seitz, S.M., Dyer, C.R.: Photorealistic scene reconstruction by voxel coloring. International Journal of Computer Vision **35**(2), 151–173 (1999)

[152] Sezen, H., Elwood, K.J., Whittaker, A.S., Mosalam, K.M., Wallace, J.W., Stanton, J.F.: Pacific earthquake engineering research center. University of California, Berkeley (2000)

[153] Sharp, N., Ovsjanikov, M.: Pointtrinet: Learned triangulation of 3d point sets. In: European Conference on Computer Vision. pp. 762–778. Springer (2020)

[154] Shen, Y., Lindenbergh, R., Wang, J., G. Ferreira, V.: Extracting individual bricks from a laser scan point cloud of an unorganized pile of bricks. Remote Sensing **10**(11), 1709 (2018)

[155] Shi, G., Xu, X., Dai, Y.: SIFT feature point matching based on improved RANSAC algorithm. In: 2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics. vol. 1, pp. 474–477. IEEE (2013)

[156] Snavely, N., Seitz, S.M., Szeliski, R.: Modeling the world from internet photo collections. International journal of computer vision **80**(2), 189–210 (2008)

[157] Spina, S., Debattista, K., Bugeja, K., Chalmers, A.: Point cloud segmentation for cultural heritage sites. In: Proceedings of the 12th International conference on Virtual Reality, Archaeology and Cultural Heritage. pp. 41–48 (2011)

[158] Strom, J., Richardson, A., Olson, E.: Graph-based segmentation for colored 3d laser point clouds. In: 2010 IEEE/RSJ international conference on intelligent robots and systems. pp. 2131–2136. IEEE (2010)

[159] Tang, P., Huber, D., Akinci, B., Lipman, R., Lytle, A.: Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. Automation in Construction (2010)

[160] Thomson, C., Boehm, J.: Automatic geometry generation from point clouds for BIM. Remote Sensing **7**(9), 11753–11775 (2015)

[161] Tommasi, C., Achille, C., Fassi, F.: From Point Cloud to BIM: A Modelling Challenge in the Cultural Heritage Field (2016)

[162] U.S. Geological Survey: Earthquake, Herat, Afghanistan 2023 (2023), https://earthquake.usgs.gov/earthquakes/eventpage/us6000lfn5/executive, [Online; zuletzt abgerufen am 31.10.2023]

[163] U.S. Geological Survey: Earthquake, Marocco 2023 (2023), https://earthquake.usgs.gov/earthquakes/eventpage/us7000kufc/executive, [Online; zuletzt abgerufen am 31.10.2023]

[164] U.S. Geological Survey: Earthquake, Turkey and Syria 2023 (2023), https://www.usgs.gov/news/featured-story/m78-and-m75-kahramanmaras-earthquake-sequence-near-nurdagi-turkey-turkiye, [Online; zuletzt abgerufen am 31.10.2023]

[165] Van-Nederveen, G., Tolman, F.: Modelling multipleviews on buildings. Automation in Construction (1992)

[166] Verykokou, S., ioannidis, C., Athanasiou, G., Doulamis, N., Amditis, A.: 3d reconstruction of distaster scenes for urban search and rescue. Springer Science+Business Media, LLC, part of Springer Nature 2017 (2017)

[167] Verykokou, S., Doulamis, A., Athanasiou, G., Ioannidis, C., Amditis, A.: UAV-based 3D modelling of disaster scenes for Urban Search and Rescue. In: 2016 IEEE International Conference on Imaging Systems and Techniques (IST). pp. 106–111. IEEE (2016)

[168] Verykokou, S., Ioannidis, C., Athanasiou, G., Doulamis, N., Amditis, A.: 3d reconstruction of disaster scenes for urban search and rescue. Multimedia tools and applications **77**(8), 9691–9717 (2018)

[169] Vfdb-Richtlinie: Hinweise für Maßnahmen der Feuerwehr und anderer Hilfskräfte nach Gebäudeinstürzen (2002)

[170] Vo, A.V., Truong-Hong, L., Laefer, D.F., Bertolotto, M.: Octree-based region growing for point cloud segmentation. ISPRS Journal of Photogrammetry and Remote Sensing **104**, 88–100 (2015)

[171] Walter, O., Kostack, K.: INACHUS, Report on Model Enhancement and Validation Cases. Tech. rep., Laurea University of Applied Sciences (2016)

[172] WANG, Z., MA, H., XU, H., PENG, J.: A method for extracting water contour lines from lidar point clouds data. Journal of Information Science **35**(4), 432–435 (2010)

[173] White Helmets: Thousands of lives are still trapped, waiting for help from under the rubble of destroyed buildings (February 2023), https://www.youtube.com/watch?v=kPdE8NeKNXk&ab_channel=TheSyriaCivilDefence-TheWhiteHelmets

[174] Wolfram Research Inc.: Mathematica 12.0 (2019), http://www.wolfram.com

[175] Wu, C.: Towards linear-time incremental structure from motion. In: 2013 International Conference on 3D Vision-3DV 2013. pp. 127–134. IEEE (2013)

[176] xbim2022: xbim tookit: making building information flow. https://docs.xbim.net/ (2022)

[177] Xiao, J., Zhang, J., Adler, B., Zhang, H., Zhang, J.: Three-dimensional point cloud plane segmentation in both structured and unstructured environments. Robotics and Autonomous Systems **61**(12), 1641–1652 (2013)

[178] Xie, Y., Tian, J., Zhu, X.X.: Linking points with labels in 3D: A review of point cloud semantic segmentation. IEEE Geoscience and Remote Sensing Magazine **8**(4), 38–59 (2020)

[179] Xu, X., Harada, K.: Automatic surface reconstruction with alpha-shape method. The visual computer **19**(7), 431–443 (2003)

[180] Yamazaki, F., Matsuda, T., Denda, S., Liu, W.: Construction of 3D models of buildings damaged by earthquakes using UAV aerial images. In: Proceedings of the tenth pacific conference earthquake engineering building an earthquake-resilient pacific. vol. 204 (2015)

[181] Yu, H., Yang, Z., Tan, L., Wang, Y., Sun, W., Sun, M., Tang, Y.: Methods and datasets on semantic segmentation: A review. Neurocomputing **304**, 82–103 (2018)

[182] Zeibak-Shini, R., Sacks, R., Ma, L., Filin, S.: Towards generation of as-damaged BIM models using laser-scanning and as-built BIM: First estimate of as-damaged locations of reinforced concrete frame members in masonry infill structures. Advanced Engineering Informatics **30**(3), 312–326 (2016)

[183] Zhang, J., Zhao, X., Chen, Z., Lu, Z.: A review of deep learning-based semantic segmentation for point cloud. IEEE Access **7**, 179118–179133 (2019)

[184] Zhao, M., Chen, H., Song, T., Deng, S.: Research on image matching based on improved RANSAC-SIFT algorithm. In: 2017 16th International Conference on Optical Communications and Networks (ICOCN). pp. 1–3. IEEE (2017)

[185] Zheng, Z., Tian, Y., Yang, Z., Lu, X.: Hybrid Framework for Simulating Building Collapse and Ruin Scenarios Using Finite Element Method and Physics Engine. Applied Sciences **10**(12), 4408 (2020)

[186] Zhou, Q.Y., Park, J., Koltun, V.: Open3D: A modern library for 3D data processing. arXiv:1801.09847 (2018)

[187] Özgenel, C.F.: Concrete crack images for classification (2019). https://doi.org/10.17632/5y9wdsg2zt.2

Anhang

A.1 Pseudocodes

A.1.1 Segmentierung

Algorithmus 1 : Segmentierung von ebenen Punktmengen mit RANSAC.

Input: P : Pre-processed point cloud;
m, ϵ, d_{bitmap}, $a \in$ params;
Output: BC objects of all plane segments in PKL file;
$C \leftarrow$ RANSAC_PLANE(**P**, params);
for i *in* C **do**
  for $j \leftarrow i+1$ *in* C **do**
    $\alpha_{i,j} \leftarrow \arccos\left(\frac{\mathbf{n_i} \cdot \mathbf{n_j}}{|\mathbf{n_i}||\mathbf{n_j}|}\right)$;
    if $\alpha_{i,j} \leq \alpha_{max}$ **then**
      $t_{i,j} \leftarrow a_i \cdot x_j + b_i \cdot y_j + c_i \cdot z_j - d_i$;
      if $t_{min} \leq t_{i,j} \leq t_{max}$ **then**
        hasParallel $\leftarrow$ *True*;
        $C_i \cup C_j \rightarrow C_{i,j}$;
        ID $\wedge$ $C_{i,j}$ $\wedge$ $t_{i,j}$ $\wedge$ *hasParallel* $\rightarrow$ BC;
      else
        $t_{i,j} \leftarrow 0$;
        hasParallel $\leftarrow$ *False*;
        ID $\wedge$ C_i $\wedge$ $t_{i,j}$ $\wedge$ *hasParallel* $\rightarrow$ BC;
      end
    else
      $t_{i,j} \leftarrow 0$;
      hasParallel $\leftarrow$ *False*;
      ID $\wedge$ C_i $\wedge$ $t_{i,j}$ $\wedge$ *hasParallel* $\rightarrow$ BC;
    end
  end
end
Save all BC objects in PKL file;

Algorithmus 2 : Randpunkt-Entfernung mit der α-*shape*-Methode.

Input: Ψ : PKL file III containing BC objects of frames with walls;
Output: BC objects of all frames and wall segments in PKL files;
for *i in* Ψ **do**
    $P_i \leftarrow \Psi_i$;
    $P_{i,xy} \leftarrow$ rotate P_i around its center of gravity to the XY plane;
    for *j in numIter* $\leftarrow$ *int* **do**
        $E_j = \alpha$-shape($P_{i,xy}$, $\alpha \leftarrow$ float);
        Search $P_{i,xy}(E_j)$ using octree;
        $P_{i,xy} \leftarrow$ Filter $P_{i,xy}(E_j)$ from $P_{i,xy}$;
    end
    $S \leftarrow$ ExtractConnectedComponents($P_{i,xy}$);
    for *n in S* **do**
        if S_n *hasRectangleShape* **then**
            $P_{n,xy}$, $W_{n,xy} \leftarrow$ crop(P_i, plane(S_n));
            $P_n \wedge W_n \leftarrow$ rotate $P_{n,xy}$ and $W_{n,xy}$ back to their original orientation;
            $P_n \wedge W_n \rightarrow$ BC;
        else
            $P_{n,xy}$, $W_{n,xy} \leftarrow$ crop(P_i, polygon(S_n));
            $P_n \wedge W_n \leftarrow$ rotate $P_{n,xy}$ and $W_{n,xy}$ back to their original orientation;
            $P_n \wedge W_n \rightarrow$ BC;
        end
    end
    Save all frame BC objects in PKL file II;
    Save all wall BC objects in PKL file IV;
end

Algorithmus 3 : Segmentierung von linearen Punktmengen mit RANSAC.

Input: Ψ : PKL file II containing BC objects of frames without walls;
Output: BC objects of line segments in PKL file V;
for *i in* Ψ **do**
    $P_i \leftarrow \Psi_i$;
    $P_{i,xy} \leftarrow$ rotate P_i around its center of gravity to the XY plane;
    $a \leftarrow 0$;
    while $a \leq a_{max}$ **do**
        $I_i(x_i, y_i) \leftarrow$ RANSAC_LINE($P_{i,xy}(x_i, y_i)$, params);
        $\mathbb{R}^2 \rightarrow \mathbb{R}^3 \Rightarrow I_i(x_i, y_i, z_i = 0)$;
        if $d_{max}(I_i) \leq d_{max}$ *and* $d_{mean}(I_i) \leq d_{mean}$ **then**
            $I_i \rightarrow I$;
            $P_{i,xy} \leftarrow O_i \leftarrow P_{i,xy} \setminus I_i$;
            $a \leftarrow 0$;
        else
            $a \leftarrow a + 1$;
        end
    end
    if $P_{i,xy}$ *is not* $\emptyset$ **then**
        $P_{i,xy} \rightarrow I$;
    end
    for *j in I* **do**
        $S \leftarrow$ ExtractConnectedComponents(I_j);
    end
    for *n in S* **do**
        $t_n \leftarrow \Psi_i$;
        if $t_n \leq t_{min}$ **then**
            $t_n \leftarrow t_d$;
        end
        $P_n \leftarrow$ rotate point segments in S back to its original orientation;
        $P_n \wedge t_n \rightarrow$ BC;
    end
    Save all BC objects in PKL file V;
end

A.1.2 Objektklassifizierung

Algorithmus 4 : Objektklassifizierung von ebenen Punktmengen.

Input: Ψ : PKL file I $\wedge$ PKL file IV;
Output: Classified BC objects in PKL file VI;
for *i in* Ψ **do**
  $P_i \wedge t_i \leftarrow \Psi_i$;
  $P_{i,xy} \leftarrow$ rotate P_i around its center of gravity to the XY plane;
  $H_{i,min} \wedge L_{i,min} \leftarrow$ MinimalBoundingBox($P_{i,xy}$);
  $\alpha_i \leftarrow \arccos\left(\frac{\mathbf{n_i} \cdot \mathbf{n_{xy}}}{|\mathbf{n_i}||\mathbf{n_{xy}}|}\right)$ with $\mathbf{n_{xy}} \leftarrow (0, 0, 1)$;
  if $t_i < t_{min}$ **then**
    $t_i \leftarrow t_d$;
  end
  if $\alpha_i \leq \alpha_{max}$ **then**
    if $z_{min}(P_i) > z_{min,slab}$ **then**
      $P_i \wedge t_i \wedge$ type $\leftarrow$ *Slab* $\wedge$ RGB $\rightarrow$ BC;
    else if $z_{max}(P_i) \leq z_{min,base}$ **then**
      if $A(P_i) \approx A_{i,slab}$ **then**
        $P_i \wedge t_i \wedge$ *type* $\leftarrow$ *Base* $\wedge$ RGB $\rightarrow$ BC;
      else
        $P_i \wedge t_i \wedge$ *type* $\leftarrow$ *Ground* $\wedge$ RGB $\rightarrow$ BC;
      end
    else
      $P_i \wedge t_i \wedge$ *type* $\leftarrow$ *Unclassified* $\wedge$ RGB $\rightarrow$ BC;
    end
  else if $\mathbf{n_{i,z}} \approx 0$ **then**
    if $H_{i,min} > H_{min,wall}$ *and* $L_{i,min} > L_{min,wall}$ **then**
      $P_i \wedge t_i \wedge$ *type* $\leftarrow$ *Wall* $\wedge$ RGB $\rightarrow$ BC;
    else
      $P_i \wedge t_i \wedge$ *type* $\leftarrow$ *Unclassified* $\wedge$ RGB $\rightarrow$ BC;
    end
  else
    if $H_{i,min} > H_{min,Slab \setminus Wall}$ *and* $L_{i,min} > L_{min,Slab \setminus Wall}$ **then**
      $P_i \wedge t_i \wedge$ *type* $\leftarrow$ *Slab* $\setminus$ *Wall* $\wedge$ RGB $\rightarrow$ BC;
    else
      $P_i \wedge t_i \wedge$ *type* $\leftarrow$ *Unclassified* $\wedge$ RGB $\rightarrow$ BC;
    end
  end
  Save all BC objects in PKL file VI;
end

Algorithmus 5 : Objektklassifizierung von linear angeordnete Punktmengen.

Input: Ψ : PKL file V;
Output: Classified BC objects in PKL file VI;
for *i in* Ψ **do**
    $P_i \leftarrow \Psi_i$;
    $P_{i,yz} \leftarrow$ rotate P_i around its center of gravity to the YZ plane;
    $H_{i,z} \wedge L_{i,y} \leftarrow$ BoundingBox($P_{i,yz}$);
    if ($H_{i,z} > H_{min,column}$ *and* $L_{min,column} < L_{i,y} < L_{max,column}$) $\leftarrow$ $SizeCondition_{Column}$ **then**
        if P_i *hasRectangleShape* **then**
            $P_i \wedge type \leftarrow Column \wedge$ RGB $\rightarrow$ BC;
        else
            $P_i \wedge type \leftarrow Unclassified \wedge$ RGB $\rightarrow$ BC;
        end
    else if ($H_{min,beam} < H_{i,z} < H_{max,beam}$ *and* $L_{i,y} > L_{min,beam}$) $\leftarrow$ $SizeCondition_{Beam}$ **then**
        if P_i *hasRectangleShape* **then**
            $P_i \wedge type \leftarrow Beam \wedge$ RGB $\rightarrow$ BC;
        else
            $P_i \wedge type \leftarrow Unclassified \wedge$ RGB $\rightarrow$ BC;
        end
    else
        $P_{i,minBB} \leftarrow$ rotate P_i around X axis until its bounding box becomes minimal;
        $H_{i,min} \wedge L_{i,min} \leftarrow$ MinimalBoundingBox($P_{i,minBB}$);
        if $SizeCondition_{Beam}$ *or* $SizeCondition_{Column}$ **then**
            if $P_{i,minBB}$ *hasRectangleShape* **then**
                $P_i \wedge type \leftarrow Beam \,|\, Column \wedge$ RGB $\rightarrow$ BC;
            else
                $P_i \wedge type \leftarrow Unclassified \wedge$ RGB $\rightarrow$ BC;
            end
        else
            $P_i \wedge type \leftarrow Unclassified \wedge$ RGB $\rightarrow$ BC;
        end
    end
    Save all BC objects in PKL file VI;
end

A.1.3 Materialklassifizierung

Algorithmus 6 : Extraktion der dominanten Farbe aus Bildern.

Input: Path to folder containing images;
Output: CSV file with RGB values and corresponding labels;
labels ← [*Masonry*, *Concrete*];
for *i in labels* **do**
    Get imgList that contains all filenames in the image folder;
    Create a csv file;
    for *img in imgList* **do**
        Load img;
        RGB_list ← reshape img into a 2D array;
        RGB_cluster ← Kmeans($k = 5$, RGB_list);
        P ← Calculate the percentage of each RGB_cluster in RGB_list;
        P_{max} ← max(P);
        RGB_{dom} ← RGB_cluster[P_{max}];
        Save RGB values and binary labels in the CSV file;
    end
end

Algorithmus 7 : *Deep-Learning*-basierte Materialklassifizierung.

Input: Ψ : PKL file IV containing BC objects of building components;
Output: Classified BC objects in PKL file VII;
model ← load_model(model.HDF5);
for *i in* Ψ **do**
    $C_i \wedge type \leftarrow \Psi_i$;
    if *type = Slab or Base or Beam or Column or Beam|Column* **then**
        material ← *Concrete*;
    else if *type = Unclassified* **then**
        material ← *Unclassified*;
    else
        RGB_{dom} ← extractDominantColor(C_i);
        matClass, confidence = model.predict(RGB_{dom});
        if *matClass = Concrete and confidence* ≥ 0.80 **then**
            material ← matClass;
        else if *matClass = Masonry and confidence* ≥ 0.80 **then**
            material ← matClass;
        else
            material ← *Unclassified*;
        end
        material → BC;
    end
    Save all BC objects in PKL file VII;
end

A.1.4 3D-Rekonstruktion

Algorithmus 8 : α-*shape*-Implementierung in *Wolfram Mathematica* [147].

```
1  Function alphaShapes(points, crit)
2      Function alphacriteria(tetrahedra, radii, rmax)
3          return Pick(tetrahedra, UnitStep(rmax - radii), 1)
4      end
5      Function selectExternalFaces(facets)
6          return MeshRegion[points, facets]
7      end
8      del ← Quiet(DelaunayMesh(points));
9      if Head(del) = EmptyRegion then
10         return del;
11     else
12         tetras ← MeshCells(del, 2);
13         tetcoords ← MeshPrimitives(del, 2)[All, 1];
14         tetradii ← Quiet(Thread(Circumsphere(tetcoords)))[All, 2];
15         for i in tetradii do
16             if i < 1 then
17                 i ← 1;
18                 i → tetradii 2;
19             else
20                 i → tetradii 2;
21             end
22         end
23         return selectExternalFaces (alphacriteria(tetras, tetradii2, crit));
24     end
25 end
```

Algorithmus 9 : 3D-Rekonstruktion der sichtbaren Bauteile.

Input: Ψ : PKL file VII containing classified BC objects;
Output: Reconstructed 3D mesh model of the structure with volume mesh components in OBJ format;
for *i in* Ψ **do**
    $P_i \wedge t_i \wedge hasParallel \leftarrow \Psi_i$;
    $P_{i,xy} \leftarrow$ rotate P_i around its center of gravity to the XY plane;
    if *hasParallel* = *True* **then**
        $P^{in}_{i,xy}$, $P^{out}_{i,xy}$ = cropParallelSegment($P_{i,xy}$);
        if $P^{in}_{i,xy} \geq P^{out}_{i,xy}$ **then**
            $P_{i,xy} = P^{in}_{i,xy}$;
            cropSec = 1;
        else
            $P_{i,xy} = P^{out}_{i,xy}$;
            cropSec = 2;
        end
    else
        $P_{i,xy}$ has no parallel segment;
        cropSec = 0;
    end
    distZ $\leftarrow a_0 \cdot x_{i,c} + b_0 \cdot y_{i,c} + c_0 \cdot z_{i,c} - d_0$;
    $M_{i,xy} \leftarrow \alpha$-shape($P_{i,xy}$, $\alpha \leftarrow$ float);
    if *cropSec* = 0 *or cropSec* = 2 **then**
        $M^E_{i,xy} \leftarrow$ extrudeMesh($M_{i,xy}$, $-t_i$);
    else
        $M^E_{i,xy} \leftarrow$ extrudeMesh($M_{i,xy}$, t_i);
    end
    $M^T_{i,xy} \leftarrow$ translate($M^E_{i,xy}$, distZ);
    $M^{TR}_{i,xy} \leftarrow$ rotate $M^T_{i,xy}$ back to its original orientation;
    Save $M^{TR}_{i,xy}$ as mesh object in OBJ format;
end

Algorithmus 10 : Hidden-Point-Removal-Algorithmus.

Input: M : Damage model in OBJ format;
Output: P_e : Exterior point cloud of the damage model in xyz format;
pointlist = [];
Mesh to Point Cloud: $M \rightarrow P$;
BoundingBox(P) $\rightarrow bb_h$;
$\varphi = 0$;
$z_1 = 0$;
for *i in range(0,37)* **do**
    $\Pi_i = [x_c = r \cdot cos(\varphi_i),\ y_c = r \cdot sin(\varphi_i),\ z_1]$;
    Create camera object at Π_i;
    φ_i += 10°;
    $P_{h,i} = HPR(P, \Pi_i)$;
    pointlist += $P_{h,i}$;
end
$z_2 = bb_h \cdot 0.5$;
for *i in range(0,37)* **do**
    $\Pi_i = [x_c = r \cdot cos(\varphi_i),\ y_c = r \cdot sin(\varphi_i),\ z_2]$;
    Create camera object at Π_i;
    φ_i += 10°;
    $P_{h,i} = HPR(P, \Pi_i)$;
    pointlist += $P_{h,i}$;
end
$z_3 = bb_h \cdot 1.5$;
for *i in range(0,13)* **do**
    $\Pi_i = [x_c = r \cdot cos(\varphi_i),\ y_c = r \cdot sin(\varphi_i),\ z_3]$;
    Create camera object at Π_i;
    φ_i += 30°;
    $P_{h,i} = HPR(P, \Pi_i)$;
    pointlist += $P_{h,i}$;
end
$z_4 = bb_h \cdot 2$;
$\Pi = [0,\ 0,\ z_4]$;
Create camera object at Π;
$P_h = HPR(P, \Pi)$;
pointlist += P_h;
$P_e \leftarrow P -$ pointlist;
Save P_e as point cloud in xyz format;

Algorithmus 11 : Matching-Algorithmus.

Input: P = input point cloud;
Output: P in the same orientation as the matching P_i;
Read N = numbers of models in database;
Read $bb_{h,i}$: boundingbox height values of the damage models P_i in database;
BoundingBox(P) $\rightarrow$ bb_h boundingbox height of P;
for *i in N* **do**
  if $bb_h \approx bb_{h,i}$ **then**
    Read P_i = HPR point cloud of damage model from database ;
    Downsample P and P_i;
    for *i in range(0,37)* **do**
      Compute point cloud distance: P as source and P_i as target;
      Compute point cloud distance: P_i as source and P as target;
      Compute $\Omega_{1,i}$ = average of point cloud distance;
      Rotate P by $\varphi_{1,i} = 10$;
    end
    Determine source and target point cloud;
    Determine $\varphi_1(\Omega_{1,i,\min})$;
    Rotate P by $\varphi_1(\Omega_{1,i,\min})$;
    for *i in range(0,21)* **do**
      Compute point cloud distance;
      Compute $\Omega_{2,i}$ = average of point cloud distance;
      Rotate P by $\varphi_{2,i} = 1$;
    end
    Determine $\varphi_2(\Omega_{2,i,\min})$;
    Rotate P by $\varphi_2(\Omega_{2,i,\min})$;
    for *i in range(0,21)* **do**
      Compute point cloud distance;
      Compute $\Omega_{3,i}$ = average of point cloud distance;
      Rotate P by $\varphi_{3,i} = 0.1$;
    end
    Determine $\varphi_3(\Omega_{3,i,\min}) = \varphi_t$;
    Rotate P by φ_t;
  else
    Pi does not satisfy dimension conditions;
  end
end
κ = index(min(list($\Omega_{3,i,\min}$)));
$\Omega_{\kappa,\min} \leftarrow \Omega_{3,i\leftarrow\kappa,\min}$;
if $\Omega_{\kappa,min} \leq \Omega_{max} \leftarrow$ *float* **then**
  Match found. Damage model with index number κ selected for further processing;
else
  There is no matching damage model from the database for this input;
end

Algorithmus 12 : Crop-Interior-Mesh-Algorithmus.

Input: Read damaged interior model from database (mesh_DB) and input point cloud P;
Output: mesh_interior + mesh_exterior;
Define bounding box parameters dx, dy;
Initialise empty mesh = *mesh_entity*;
$N \leftarrow size(P)$;
for i *in* N **do**
    if $P(x_i) > 0$ *and* $P(y_i) > 0$ **then**
        $minBB = [P(x_i) - \mathrm{d}x,\ P(y_i) - \mathrm{d}y,\ 0]$;
        $maxBB = [P(x_i),\ P(y_i),\ P(z_i)]$;
        Create bounding box with *minBB* and *maxBB* ;
    end
    if $P(x_i) < 0$ *and* $P(y_i) < 0$ **then**
        $minBB = [P(x_i),\ P(y_i),\ 0]$;
        $maxBB = [P(x_i) + \mathrm{d}x,\ P(y_i) + \mathrm{d}y,\ P(z_i)]$;
        Create bounding box with *minBB* and *maxBB* ;
    end
    if $P(x_i) > 0$ *and* $P(y_i) < 0$ **then**
        $minBB = [P(x_i),\ P(y_i),\ 0]$;
        $maxBB = [P(x_i) + \mathrm{d}x,\ P(y_i) + \mathrm{d}y,\ P(z_i)]$;
        Create bounding box with *minBB* and *maxBB* ;
        Translate bounding box with -dx in x-direction;
    end
    if $P(x_i) < 0$ *and* $P(y_i) > 0$ **then**
        $minBB = [P(x_i),\ P(y_i),\ 0]$;
        $maxBB = [P(x_i) + \mathrm{d}x,\ P(y_i) + \mathrm{d}y,\ P(z_i)]$;
        Create bounding box with *minBB* and *maxBB* ;
        Translate bounding box with -dy in y-direction;
    end
    mesh_interior = crop.*mesh_DB*(bounding_box) and save result in *mesh_entity*;
end
Clean mesh_interior by removing duplicate vertices and faces;
Write mesh_interior + mesh_exterior;

A.1.5 BIM-Generierung

Algorithmus 13 : *Mesh-to-IFC*-Algorithmus.

Input: Read OBJ files of the digital reconstructed building components;
Output: BIM model containing the digital reconstructed building components in IFC format;
model ← Create blank IFC model using IFC4 standard;
Create standard IFC project structure for model;
objList ← Read obj filenames from folder;
for *objfile in objList* **do**
    Read objfile;
    $type \wedge material \wedge t \leftarrow$ Read BC object data of corresponding objfile;
    Define ifcClass depending on *type* (see Table 3.4);
    vertices, *faces* ← objfile;
    for *face in faces* **do**
        model.createIfcCartesianPoint using vertices of face;
        model.createIfcPolyLoop using IfcCartesianPoint;
        model.createIfcFaceOuterBound using IfcPolyLoop;
        model.createIfcFace using IfcFaceOuterBound;
        ifcFaceList ← IfcFace
    end
    model.createIfcClosedShell using all items in ifcFaceList;
    model.createIfcFacetedBrep using IfcClosedShell;
    model.createIfcShapeRepresentation using IfcFacetedBrep;
    model.createIfcProductDefinitionShape using IfcShapeRepresentation;
    IFC-Object ← IfcProductDefinitionShape of model
    IfcEntity ← model.IFC-Object;
    IfcClass of model.IfcEntity ← ifcClass;
    Define user-defined psets: *Pset_ Geometry*, *Pset_ Material*, *Pset_ Damage* and *Pset_ Actions*;
    $t, V, A \rightarrow$ pset_Geometry;
    if *material = Concrete* **then**
        $\rho \leftarrow 2.400$ kg/m^3;
        $E \leftarrow 30.000$ MPa;
        $m \leftarrow V \cdot \rho \cdot 10^{-3}$;
    end
    if *material = Masonry* **then**
        $\rho \leftarrow 1.950$ kg/m^3;
        $E \leftarrow 8.000$ MPa;
        $m \leftarrow V \cdot \rho \cdot 10^{-3}$;
    end
    if *material = Unclassified* **then**
        $\rho \leftarrow 0$;
        $E \leftarrow 0$;
    end
    material, $\rho, E, m \rightarrow$ *Pset_ Material*;
    Manual input → *Pset_ Damage*;
    Manual input → *Pset_ Actions*;
    psets → model.IfcEntity;
end
Write model in IFC format;

A.2 Stahlbetonrahmentragwerke

A.2.1 Fallbeispiel Jindires

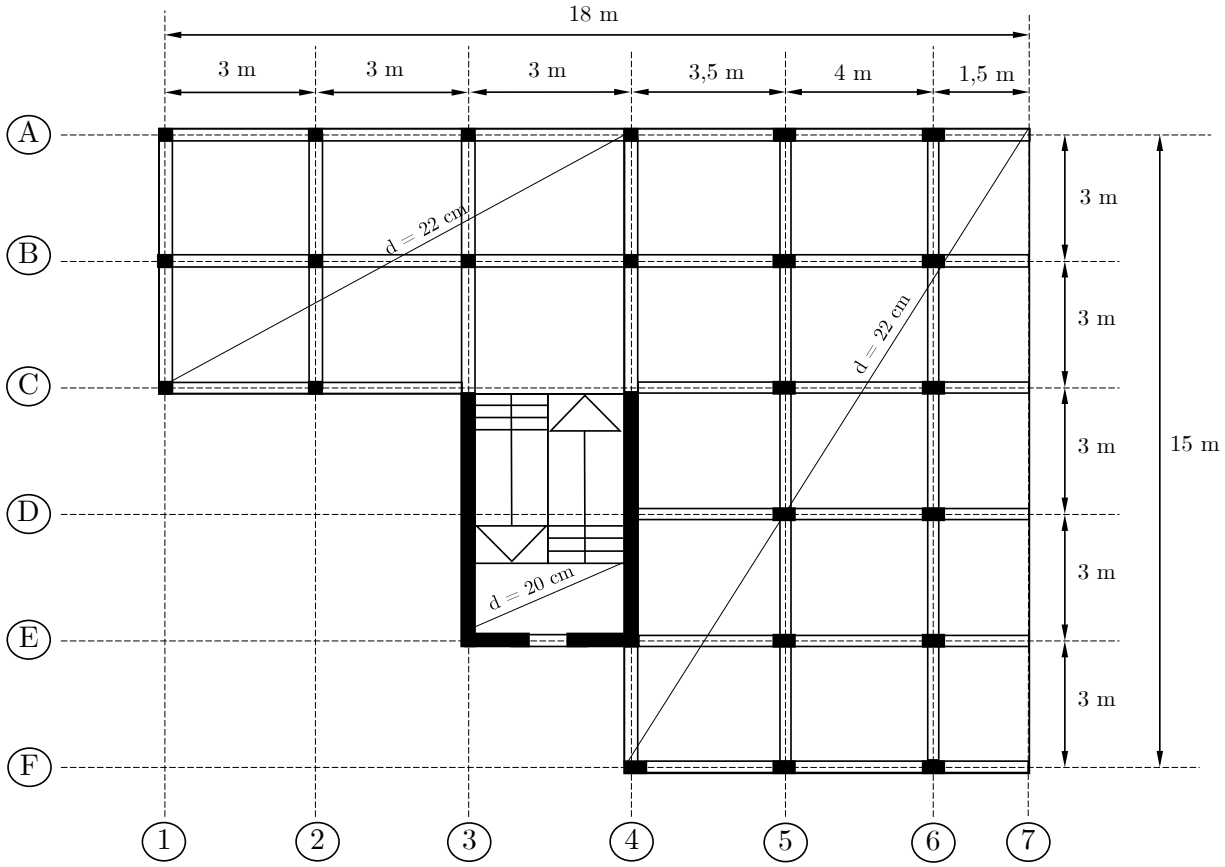


Abbildung A.1: Grundrissplan eines Stahlbetonrahmentragwerkes in Jindires, Syrien, mit fiktiven Maßangaben.

A.2.2 Fallbeispiel Gölcük

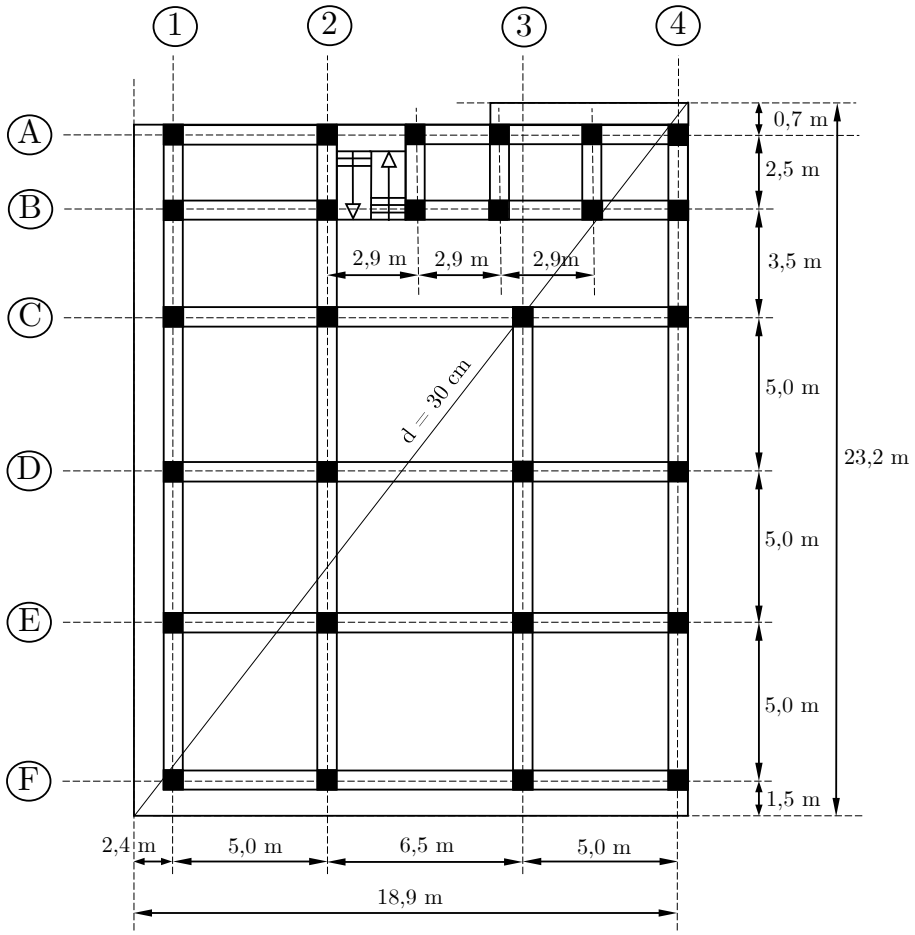


Abbildung A.2: Grundrissplan eines Stahlbetonrahmentragwerkes in Gölcük, Turkei, (Maßangaben entnommen aus [43]).