Entwicklung und Implementierung einer Methode zum Identifizieren und Extrahieren von Kabinen- und Systemkomponenten aus gemessenen 3D-Datensätzen realer Kabinengeometrien

Bachelorarbeit

des Studiengangs Informationstechnik

an der Dualen Hochschule Baden-Württemberg Mannheim

von

Luca Julian Tiedemann

29. September 2023

Bearbeitungszeitraum	06.06.2023 - 29.09.2023
Matrikelnummer	6739231
Kurs	TINF20IT1
Ausbildungsfirma	Deutsches Zentrum für Luft- und Raumfahrt (DLR) e.V.
Betreuer	Prof. Dr. Jörn Biedermann
Gutachter	Prof. Dr. Reinhold Hübl

Plagiatserklärung

Ich erkläre hiermit ehrenwörtlich:

1. dass ich meine Bachelorarbeit mit dem Thema *Entwicklung und Implementierung einer Methode zum Identifizieren und Extrahieren von Kabinen- und Systemkomponenten aus gemessenen 3D-Datensätzen realer Kabinengeometrien* ohne fremde Hilfe angefertigt habe;
2. dass ich die Übernahme wörtlicher Zitate aus der Literatur sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe;
3. dass ich meine Bachelorarbeit bei keiner anderen Prüfung vorgelegt habe;
4. dass die eingereichte elektronische Fassung exakt mit der eingereichten schriftlichen Fassung übereinstimmt.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Hamburg, 29. September 2023

Luca Julian Tiedemann

Abstract

Um eine detaillierte digitale Kabine zu erstellen, werden physische Komponenten durch digitale Modelle, welche die Real-Geometrien wiedergeben, verknüpft. Diese digitale Version der Kabine ermöglicht das Ermitteln von Optimierungspotenziale und das Testen neuer Komponenten in der digitalen Kabinenumgebung. Die konstante Aufrechterhaltung eines Digitalen Zwillings erfordert automatisch generierbare und abfragbare Daten wie 3D-Modelle. Um Real-Geometrien in diesen aktuellen 3D-Modellen wiederzufinden, werden Scan-Daten von den jeweiligen Komponenten benötigt. Die Scan-Daten liegen zunächst in der Form von mehreren Punktewolken vor und spiegeln eine gesamte Szene wider. In dieser Arbeit wird eine Methode evaluiert, die in der Szene Kabinenkomponenten erkennt und die zugehörigen Punkte extrahiert. Zudem soll die Möglichkeit bestehen, wichtige Daten zu der Komponente abzurufen und die Position der Komponente zu bestimmen. Dazu wird aus einer Auswahl an Algorithmen der Spin-Image-Algorithmus gewählt, wichtige Parameter des Algorithmus identifiziert und im Sinne des Anwendungsfalls Kabine getestet und ausgewertet. Zudem werden unterschiedliche Parameter für einen automatisierbaren Scanner identifiziert und beurteilt, um dessen Gebrauch in der Kabinenumgebung in Kombination mit dem Spin-Image-Algorithmus zu evaluieren. Hier wurde festgestellt, dass der verwendete automatisierbare Scanner deutlich größeres Rauschen aufweist als vom Hersteller angegeben. Dieses Rauschen wurde als ein großer negativer Einfluss auf das Erkennen der Objekte mithilfe des Spin-Image-Algorithmus erkannt.

Abstract

To create a detailed digital cabin, physical components are linked by digital models that reflect real-world geometries. This digital version of the cabin enables the determination of optimization potentials and the testing of new components in the digital cabin environment. The constant maintenance of a digital twin requires automatically generated and retrievable data such as 3D models. In order to retrieve real geometries in these current 3D models, scan data from the respective components is required. Initially, the scan data is in the form of multiple point clouds and reflects an entire scene. In this work, a method is evaluated that detects cabin components in the scene and extracts the corresponding points. In addition, it should be possible to retrieve important data about the component and determine the position of the component. For this purpose, the spin-image algorithm is chosen from a selection of algorithms, important parameters of the algorithm are identified and tested and evaluated in terms of the cabin use case. In addition, different parameters for an automatable scanner are identified and evaluated to evaluate its use in the cabin environment in combination with the spin image algorithm. Here, it was found that the automatable scanner used had significantly greater noise than that specified by the manufacturer. This noise was found to have a large negative impact on the detection of objects using the spin image algorithm.

Inhaltsverzeichnis

Abkürzungsverzeichnis **V**

Abbildungsverzeichnis **VI**

Tabellenverzeichnis **VII**

1 Einleitung **1**

2 Grundlagen der 3D-Darstellung von Objekten **6**
2.1 Möglichkeiten der virtuellen Darstellung von Objekten 6
2.2 Objektmodellierung für CAx 8

3 Grundlagen der Objekterkennung **10**
3.1 Spin-Image Algorithmus für die Objekterkennung 11
3.2 RANSAC Algorithmus mit Merkmalserkennung für Objekterkennung . . . 18
3.3 KI für Objekterkennung 19

4 Anforderungsanalyse des Algorithmus **20**
4.1 Erarbeitung der Parameter für den Algorithmus 20
4.2 Validierung des Algorithmus mit den erarbeiteten Parametern 25

5 Algorithmus Implementierung in einem Versuchsaufbau zur Analyse von Parametern **27**
5.1 Beschreibung des Versuchsaufbaus 27
5.2 Versuchsdurchführung 29
5.3 Extraktion der Punktewolke aus CAD-Daten 30
5.4 Zusammensetzung der Punktewolken aus den Scan-Daten 31
5.5 Extraktion des Objektes aus der Punktewolke 33

6 Ergebnisse des Versuchsaufbaus **35**
6.1 Einfluss des Abstandes auf den Algorithmus 35
6.2 Einfluss der Beleuchtung auf den Algorithmus 37
6.3 Einfluss der Anzahl an Aufnahmen auf den Algorithmus 38

7 Auswertung der Analyse des Algorithmus und Ergebnisse des Versuchsaufbaus **40**
7.1 3D-Data Extraktion 40
7.2 Objekterkennung 41

8 Zusammenfassung und Ausblick **44**

Literatur **45**

Abkürzungsverzeichnis

DLR	Deutsche Zentrum für Luft- und Raumfahrt
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CAx	Computer-Aided x
VTK	The Visualisation Toolkit
OCC	Open CASCADE
RANSAC	RANdom SAmple Consensus
STEP	Standard for the Exchange of Product model data
KI	Künstlichen Intelligenz
ROIs	Region Of Interest
KNN	Nächste-Nachbarn-Klassifikation
KDTree	K-dimensionalen Baums
STL	Standard Transformation Language
ICP	Iterative Closest Point
DigECAT	Digital Twin for Engine, Components and Aircraft Technologies

Abbildungsverzeichnis

2.1 Beispiele für Oberflächennetze 7

3.1 Beispiel zum Generieren eines Spin-Image von der Sitzreihe 12
3.2 Aufbau des Spin-Image anhand der (α, β) Koordinaten [9] 13
3.3 Überblick zu dem Spin-Image-Algorithmus 17

4.1 Beispiel zum Vergleich zweier Spin-Images von der Sitzreihe 21
4.2 Beispiel zum Vergleich zweier Spin-Images von der Sitzreihe 22
4.3 Auswertung des Einflusses von Störungen auf die Korrelation 23
4.4 Auswertung des Einflusses von Störungen auf die Korrelation beim Modell des ISTAR 24
4.5 Validierung des Algorithmus mit den Parametern aus Kapitel 4.1 25

5.1 Versuchsaufbau zum Scannen der Sitzreihe 27
5.2 Verwendeter Scanner: rc_visard 28
5.3 Aufbau des Versuches(Top down) 30
5.4 Sitzreihe als Computer-Aided Design (CAD)-Modell(links) und Punktewolke(rechts) 31
5.5 Aufgenommene Sitzreihe als Bild(links) und 3D-Scan(rechts) 32
5.6 Besipiel einer Extraktion der Komponente aus der Szene 34

6.1 Auswertung des Abstands zum Objekt 36
6.2 Visualisierung der aufgenommenen Daten zur Auswertung der Beleuchtungsstufen 37
6.3 Fehlerhafte Auswertung des Einflusses der Anzahl an Scans auf die Erkennungsrate 38
6.4 Darstellung der fehlerhaft ausgeschnittenen Sitzreihe 39

Tabellenverzeichnis

5.1 Aufführung der betrachteten Abstände zu der aufgenommenen Sitzbank . . 29
5.2 Vorstellung der verschiedenen Aufnahmewinkel 29
5.3 Darstellung der unterschiedlichen Helligkeiten bei Aufnahme der Punktewolke 29

1 Einleitung

Seit der Erfindung von Flugzeugen werden immer weitere technologische Errungenschaften für Flugzeuge entwickelt oder an diesen geforscht. Beispiele für Forschungen sind unter anderem unbemannte Flugzeuge, Verwendung von alternativen Treibstoffen und das generelle Verbessern von Arbeitsprozessen für das Produzieren von Flugzeugen. Diese Forschungsbereiche sind durch den ökologischen Wandel der Gesellschaft und dem fortschreitenden Klimawandel eine Herausforderung, um die Luftfahrt klimaneutraler und nachhaltiger zu gestalten [33]. Zum Evaluieren, Vorstellen und Fördern von neuen Technologien und Errungenschaften in diesen Gebieten wird letztendlich eine physische Nachbildung gebaut. Die Nachbildung besteht aus einem Modell des Designs in seiner eigentlichen Größe. In der Entwicklung eines Flugzeuges entstehen mehrere Konzepte. Diese können allerdings aufgrund beschränkter Ressourcen, die zum Herstellen benötigt werden, wie Zeit und Geld, nicht alle gebaut werden, um sie zu validieren. Deshalb werden Methoden gebraucht, um frühzeitig in der Konzeptphase die Konzepte auf eine bestmögliche Auswahl zuzuschneiden. Mithilfe der Digitalisierung soll eine ganzheitliche visualisierte Beurteilung der unterschiedlichen Konzepte ermöglicht werden und Änderungen an neuen Konzepten schnellstmöglich integrierbar sein.

Das Deutsche Zentrum für Luft- und Raumfahrt (DLR) forscht an der durchgängigen Digitalisierung der Luftfahrt [10]. Der wesentliche Aspekt, auf dem diese Arbeit beruht, ist dabei die Automatisierung, angefangen mit der Produktionsentwicklung bis zur Analyse-basierten Zulassung, Herstellung und Wartung [10]. Damit wird eine vollständige digitale Durchgängigkeit, Daten- und Modellkonsistenz und -kompabilität erreicht. Das Institut für Systemarchitekturen in der Luftfahrt beschäftigt sich im Sinne des Ziels der durchgängigen Digitalisierung mit unterschiedlichen Themen in den Bereichen der Kabinen und Nutzlastsysteme, den Konzepten und der Evaluierung von Flugsystemen und der Systemarchitektur, um eine ganzheitliche Betrachtung des Systems Flugzeug vorzunehmen. Die Abteilung für Kabinen und Nutzlastsysteme erarbeitet Möglichkeiten der Digitalisierung in Bezug auf Kabinen und Nutzlastsysteme, um ungenutztes Potenzial zu identifizieren und den gesamten Entwicklungsprozess zu beschleunigen. Dazu gehört die Forschung an alternativen Kabinenkonzepten für den Passagiertransport und dem Erstellen von Interface- und Feedback-Schleifen zwischen dem realen und virtuellen Produkt- und Produktions-System, um einen Digitalen Zwilling zu ermöglichen. Durch die Unterstützung des Entwicklungs- und Produktionsprozesses wird die Forschung an alternativen Kabinenkonzepten beschleunigt. Das Ziel liegt in dem Erstellen von Simulationsumgebungen für

die Produktbeurteilung und Planung und deren Bereitstellung für Experten, die mit der Entwicklung von Systemen und Kabinen involviert sind oder mit deren Versorgungskette.

Computer-Aided x (CAx) zeigt bereits die Effizienz, die durch die Digitalisierung von Prozessen ermöglicht wird. Es ist möglich, viele der benötigten Berechnungen zum Evaluieren und Validieren von Konzepten von Computern durchführen zu lassen. McKinsey & Company [6] führt auf, wie physische Tests an den Nachbildungen durch digitale ersetzt bzw. mithilfe der digitalen Tests die benötigten physischen Nachbildungen reduziert werden. Das digitale Testen und Validieren der erstellten Nachbildung vor deren Produktion ermöglicht deutliche Kostenersparnisse und Qualitätsverbesserung [6]. Um das digitale Testen und Visualisierungen zu ermöglichen, werden alle Informationen über das Konzept gesammelt und in dem sogenannte Digitalen Faden übertragen. Der Digitale Faden vernetzt alle Informationen über den gesammten Lebenszyklus des Flugzeuges sinnvoll. Ein Ziel des Digitalen Fadens ist neue Inovationen zu beschleunigen, indem die in ihm bereitgestellten Informationen von allen Teilnehmern der Produktions und Herstellungskette eines Flugzeuges eingesehen werden können. Der Digitale Zwilling hingegen beschreibt als Teil des Digitalen Fadens den aktuellen Zustand eines Produktes oder Gegenstandes und bietet eine Grundlage, um Simulationen von echten Prozessen laufen zu lassen [19]. Das Fehlen einer solchen Schnittstelle erlaubt, aufgrund der Komplexität der Flugzeugkabine, Fehlern die Fortpflanzung durch mehrere Prozesse. Dadurch muss eine kostenintensive Rückführung und Behebung dieser Fehler vorgenommen werden. Als Beispiel hierfür dient Airbus, die das Produzieren von modernen Flugzeugen mit dem Leiten eines Orchesters vergleichen [2]. Airbus gibt an, dass viele Prozesse über den gesamten Globus verteilt stattfinden, wodurch Fehler nicht nur den normal größeren Arbeitsaufwand erfordern, sondern ebenfalls über Zeitzonen hinweg gelöst werden müssen.

Der Digitale Zwilling kann zum Validieren und Optimieren von Komponenten oder Architekturen und einer Instandhaltungsanalyse benutzt werden. Demnach können Teile der Evaluierung und Vorstellung eines Konzeptes, die normalerweise in einer Nachbildung getätigt werden, in den Digitalen Zwilling verlagert und dadurch Geld und Zeit gespart werden. Die Anforderungen an den Digitalen Zwilling sind je nach Anwendungsbereich unterschiedlich. Für die Produktion, den Entwurf und die Instandhaltung sind seine Datenintegrität, Skalierbarkeit, Interoperabilität und Echtzeitfähigkeit interessant. Für Produktdesigner sind zum Beispiel die Details in 3D-Modellen, wie Fixpunkte, deutlich wichtiger als für den zuständigen Simulationstechniker, der sich eher auf Material und Kräfteauswirkung bezieht. Der Digitale Zwilling beinhaltet im Allgemeinen den Zustand seines realen Gegenstücks zum Zeitpunkt der Aufnahme. Somit wird es ermöglicht den gesamten Lebenszyklus eines Gegenstücks in einer Ansammlung von Digitalen Zwillingen darzustellen. Um die Echtzeitfähigkeiten des Digitalen Zwillings zu benutzen, braucht dieser einen konstanten Zufluss von Daten, wie 3D-Modellen und Sensorinformationen,

um so aktuell wie möglich zu sein. Die 3D-Modelle werden benötigt, um den tatsächlichen Zustand des Flugzeugs darzustellen. Aufgrund von Abweichungen in der Produktion und Annahmen bzw. Vereinfachungen in der Modellierung sind CAD-Daten und andere vor der Produktion erstellte Modelle nicht akkurat genug und weisen Abweichungen zu der realen Geometrie auf. Diese undokumentierten Abweichungen entstehen ebenfalls beim Umbauen und retrofitting von Flugzeugkabinen und sind dadurch schwer nachzuverfolgen. Zudem sind für ältere Flugzeugmodelle und zugekaufte Bauteile keine CAD- und 3D-Daten vorhanden. Vorhandene CAD-Modelle sind zusätzlich nicht für jeden zugänglich, sodass diese für einzelne Bauteile fehlen.

Deshalb werden manuell die benötigten Komponenten gescannt, die zugehörigen Punkte der entstehenden Punktewolke extrahiert und ein 3D-Modell aus diesen gebildet. Dieser manuelle Prozess ist besonders bei großen und detalierten Objekten sehr Zeitaufwendig [33]. Pu et al. [31] beschreibt in seiner Arbeit, dass das manuelle Erstellen von 3D-Modellen auch in anderen Bereichen, wie das erstellen von Stadt-Modellen, sehr zeitaufwendig ist. Um den Zeitaufwand zu verringern, wiederholende Arbeitsschritte zu vereinheitlichen und deren Effizienz zu steigern, wird das Scannen der Systeme, die Objekterkennung und die Extraktion automatisiert. Dadurch wird die Modellkonsistenz für einen echtzeitbetriebenen Digitalen Zwilling ermöglicht. Mithilfe eines Roboters kann die Aufnahme der Punktewolken automatisiert werden, indem der Roboter den Scanner an unterschiedliche Stellen bewegt und die Scann-Daten mit den Positions-Daten verknüpft [35]. Das Aufnehmen der Punktewolken mithilfe des Roboters automatisiert allerdings nur das Ansammeln von Punktewolken. Der nächste Schritt besteht darin, die erfassten Punktewolken auszuwerten. Für die Integration in den Digitalen Zwilling ist es wichtig, Objekte in der Punktewolke vom Rest der Szene zu trennen. Die erkannten Objekte werden anschließend in ein 3D-Modell umgewandelt. Dafür wird die Punktewolke zu einem Oberflächennetz verarbeitet, um ein Oberflächenmodell zu bilden. Die extrahierte Punktewolke stell bereits die akkuraten Geometrien, welches für neue Entwürfe wiederverwendet werden können, dar und bietet die Möglichkeit automatische Vergleiche bestehenden 3D- oder CAD-Modellen zu ziehen.

Die Zuweisung von bestimmten Punkten innerhalb einer Punktewolke zu einem Objekt bietet mehrere Vorteile. In der eigentlichen Szene sind aufgrund der großen Anzahl der einfarbigen Punkte keine Details erkennbar. Deshalb wird eine Objekterkennung benötigt, die das Extrahieren und betrachten einzelner Objekte ermöglicht. Ein erkanntes Objekt kann aus Szenen entfernt werden, um dies durch ein anderes Modell zu ersetzen. Dadurch besteht die Möglichkeit schnell Kabinenumgebung der Realität virtuell anzupassen. Zudem besteht das Bedürfnis, gescannte Objekte für den Digitalen Zwilling zu erkennen und aus ihrer Umgebung für detaillierte Analysen zu extrahieren [14]. Mithilfe dieser Objekte und ihrer Positionen kann zum Beispiel das Layout einer Flugzeugkabine schnell erkannt und nachgestellt werden. Ein solches Layout wird unter anderem für die Datenaufarbeitung

benötigt, weil zum Zeitpunkt des Baues vieler Flugzeuge noch nicht die Möglichkeit bestand ein digitales Modell dieser anzufertigen oder bei Umbauten nicht alle Änderung dokumentiert werden. Das Objekt kann außerdem mit seinem Soll-Zustand verglichen und bereits erstellte Konzepte mit den Echtdaten angepasst werden. Zudem kann der Lebenszyklus eines Objektes aufgenommen werden, indem die veralteten Digitalen Zwillinge im Digitalen Faden gespeichert werden, um zum Beispiel dessen Änderungen zu beurteilen. Um eine Versionierung der Digitalen Zwilling zu ermöglichen und Modifikationen an der Flugzeugkabine festzustellen, werden akkurate und aktuelle 3D-Modelle der einzelnen Komponenten und detaillierte Scans der Flugzeugkabine benötigt. Die Automatisierung wird aufgrund der Tatsache benötigt, dass ein manuelles Scannen in eine sich bewegenden Arbeitsumgebung nicht möglich ist und das manuelle Scannen ein deutlich höheren Zeitaufwand beansprucht. Wenn zum Beispiel Änderungen in einem Flugzeug erfasst werden, muss dies schnellstmöglich geschehen, weil die Bodenzeit der Flugzeuge so gering wie möglich gehalten wird. Während dieser Bodenzeit werden bereits Wartungen und die Installation von Änderungen durchgeführt, sodass der Zeitraum, der zum Scannen verwendet wird, deutlich eingeschränkt ist. Zudem besteht in der Robotik ein Drang dazu, objektspezifische Informationen über das vorliegende Objekt zu besitzen. Um die objektspezifischen Informationen zu erfassen, besteht die Anforderung, das Objekt zu identifizieren. Letztlich ist für einige Anwendungen und die Überwachung von Prozessen eine Szenenanalyse nötig, die z. B. die Anzahl bestimmter Objekte erfasst.

Ein Teil dieser Forschungen findet im Sinne des Projektes Digital Twin for Engine, Components and Aircraft Technologies (DigECAT) statt. DigECAT ist das Folgeprojekt von DigTwin (01.2019-12.2021) und beschäftigt sich unter anderem mit der Entwicklung des Digitalen Faden und Digitalen Zwilling für Flugzeuge. Dafür ist die Datenaufbereitung von 3D-Aufnahmen, wie das Erstellen von Oberflächenmodellen, dem Export von Modellen und dem Separieren von Komponenten in der Aufnahme zu automatisieren, um die Effizienz zu steigern und diesen sich wiederholenden Arbeitsschritt zu vereinheitlichen. Der erstellte konsistente Datensatz wird zB. für FEM-Simulationen, Strukturanalysen oder Versionierung genutzt. Demnach fleißen die Ergebnissen zurück in die CAD-Umgebung. [33]

Diese Arbeit beschäftigt sich damit, die Erkennung von Objekten in den 3D-Scans zu gewährleisten und aus dem Scan ein 3D-Modell zu bilden oder heraus zu lösen. Deshalb werden zunächst Grundlagen für die virtuelle Darstellung von Objekten in Kapitel 2 erläutert. In Kapitel 3 werden Grundlagen zur Objekterkennung mit dem Fokus auf die Objekterkennung im 3D-Raum erklärt. An den erläuterten Algorithmus und die Untersuchung der benötigten Bedingungen zum Erkennen von Objekten in einem 3D-Scan werden in Kapitel 4 dargestellt. Zudem werden Voraussetzung für den zuvor erläuterten Objekterkennungsalgorithmus im folgenden Kapitel 5 dargelegt. Im folgenden Kapitel 6 werden die

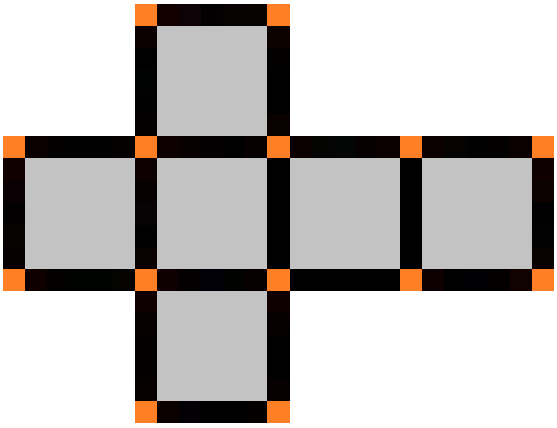
Daten aus den vorherigen Kapiteln vorgestellt und ausgewertet, um Aussagen bezüglich der verschiedenen Parameter zu tätigen. Diese Aussagen werden im Anschluss in Kapitel 8 in Bezug auf die Motivation gesetzt und ein Ausblick in weitere Forschungsmöglichkeiten gegeben.

2 Grundlagen der 3D-Darstellung von Objekten

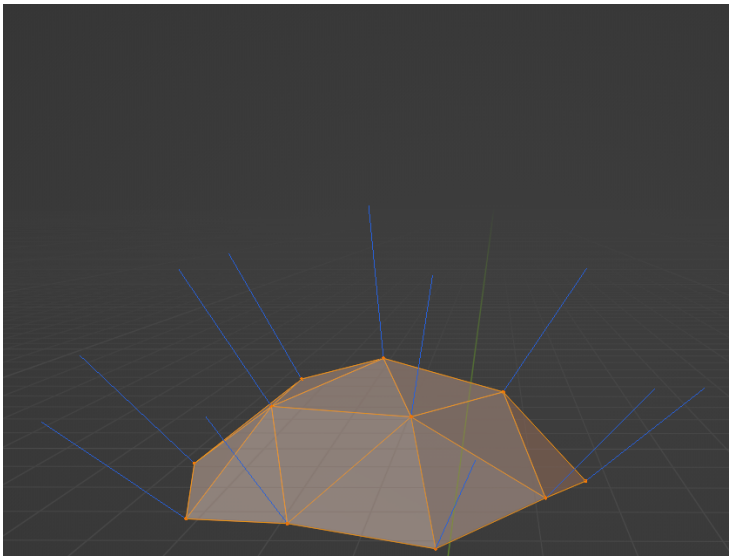
Besonders bei der Herstellung und der Überarbeitung, dem Retrofitting, von Flugzeugen ist eine Digitalisierung der Komponenten sehr wichtig. Diese ermöglicht es einen besseren Überblick über einzelne Systeme und das gesamte Flugzeug zu gewinnen. Dadurch kann zum Beispiel ungenutztes Potenzial in den Entwürfen, der Produktion und anderen Teilen des Lebenszyklus eines Flugzeuges identifiziert werden. Die Digitalisierung der Komponenten kann zudem dafür genutzt werden neue Pläne von Bauteilen zu erstellen, die zuvor nicht dokumentiert wurden oder deren Dokumentation nicht zugänglich ist. Aufgrund der Komplexität des Flugzeugkabinesystems, ist es schwierig einen Überblick über dieses zu behalten. Mithilfe von digitalen Versionen der Flugzeugkabinen wird es ermöglicht, leichter einen Überblick über die Flugzeugkabine und deren Systeme zu gewinnen. Aus diesem Grund werden in diesem Kapitel Möglichkeiten der digitalen Darstellung von Objekten erläutert.

2.1 Möglichkeiten der virtuellen Darstellung von Objekten

In der letzten Zeit sind leistungsfähigere Computer entwickelt worden und Fortschritte in Machine Learning haben eine Überarbeitung der klassischen Modellierung von 3D-Modellen angeregt [30][46]. Objekte können als Datensatz in unterschiedlichen Weisen dargestellt werden. Häufig werden die Objekte mit einem sogenannten Oberflächennetz modelliert. Dieses besteht, wie der Name bereits andeutet, aus einem Netz, welches durch Eckpunkte (Vertices), Kanten (Edges) und Oberflächen (Faces) beschrieben wird. Die Eckpunkte sind dabei durch ihre Postion in dem Koordinatensystem des 3D-Modells und ihrer Ausrichtung definiert. Die Ausrichtung beschreibt dabei einen Normalenvektor der Senkrecht von der Fläche, in dem der Punkt sich befindet, weg führt. Diese Fläche wird auch als Tangenten-Ebene bezeichnet. Kanten werden jeweils durch die zwei Eckpunkte beschrieben, die sie verbindet und Oberflächen werden durch mindestens drei Kanten beschrieben, die jeweils einen Punkt teilen. Es gibt unterschiedliche Arten von Netzen, die sich lediglich durch die Anzahl an Kanten, die eine Oberfläche formen, unterscheiden. Am häufigsten wird dabei die Dreiecksvariante, bei der jede Oberfläche durch genau drei



(a) Oberflächennetz eines Würfels (2D)



(b) Oberflächennetz einer Halbkugel mit Visualisierung von Normalvektoren an den Punkten

Abbildung 2.1: Beispiele für Oberflächennetze

Kanten beschrieben wird, verwendet. Diese Variante ist besonders nützlich, weil sich Kreise und Rundungen leichter darstellen lassen.

In Abbildung 2.1a wird ein Oberflächenmodel eines Würfels abgebildet. Die Vertices, Edges und Faces werden durch unterschiedliche Farben dargestellt. Die grauen Flächen stellen die Faces dar, die schwarzen Linien symbolisieren die Edges und die Vertices werden durch die orangen Pixel dargestellt. Der Würfle ist in diesem Fall im 2D-Raum dargestellt und besteht aus einem Viereckснetz.

Ein weiteres Beispiel für ein Oberflächenmodell ist in Abbildung 2.1b zu erkennen. Dabei handelt es sich um die Oberseite einer sogenannten 'Ico Sphere', einer Kugel die mit einem

Dreiecksnetz gebildet wurde. In der Abbildung sind die Vertices, Edges, Faces und die den Vertices zugehörigen Normalen zu erkennen. Die orangen Punkte stellen die Vertices dar, die orangen Linien zwischen den Vertices sind die Edges und die grauen Flächen entsprechen den Faces. Die Normalen sind in der Form von blauen Linien dargestellt. Wie in der Abbildung zu erkennen, spiegelt die Normale die Ausrichtung der anliegenden Flächen wider. Das liegt daran, dass Normalen den orthogonalen Vektor einer Tangenten-Ebene zu dem zugehörigen Punkt darstellen. Diese Tangenten-Ebene wird entweder durch die umliegenden Flächen beschrieben oder durch die umliegenden Punkte.

Normalerweise werden durch den Scan selbst keine Verbindung zwischen den Vertices festgestellt, deshalb wird bei 3D-Scans grundsätzlich eine Punktewolke erstellt. Zudem werden nicht von jedem Scanner die Normalen mitberechnet. Diese können im Nachhinein mithilfe der Relationen der einzelnen Punkte bestimmt werden. Dafür wird eine bestimmte Anzahl an nahen Punkte gewählt und eine Abschätzung zur Normalen getätigt. Diese ist meistens besser, wenn mehr Punkte benutzt werden, allerdings ist der Rechenaufwand pro Punkt ebenfalls größer.

Die Normalen sind für die Darstellung von Objekten besonders wichtig, weil unterandererem Oberflächennetze mithilfe der Normalen gebildet werden. In dieser Arbeit sind die Normalvektoren besonders wichtig, weil mit ihnen das erstellen von objektspezifischen Deskriptoren möglich ist.

2.2 Objektmodellierung für CAx

Die Darstellung in einer Punktewolke oder anderen herkömmlichen 3D-Modellen reicht für die Modellierung von komplexen Systemen nicht aus. Dort werden zusätzlich zu der Geometrie auch technische Details zum Material und Mechanismen benötigt, um diese mithilfe von CAx auszuwerten. CAx beschreibt seine Bestandteile, wie CAD und Computer-Aided Engineering (CAE). CAD wurde erstellt, um die traditionell getrennten Prozesse Design und Produktion besser zu integrieren. Demnach beschreibt die für CAD verwendeten Modelle den Soll-Zustand, der für das Modell erstellt wurde, während Punktewolken und andere gescannte Daten hingegen den Ist-Zustand beschreiben. Somit legt CAD einen Grundstein für Computerintegrierte Produktion. CAD beschreibt im Allgemeinen das Nutzen von Softwaresystemen zum Berechnen von unter anderem Stress- und Belastungsanalysen, der aerodynamischen Reaktion des Modells, sowie anderen für das Design relevanten Daten. [34] CAE hingegen beschreibt die Verwendung von Computern in dem Bereich des Engineering, um zum Beispiel Kollisions- und Baubarkeitsprüfungen durchzuführen.

Für die Verwendung eines Modells für diese Berechnungen und einem Austausch von Informationen zwischen den verschiedenen CAx-Systemen wird ein neues Dateiformat benötigt, welches Informationen zur Geometrie, Material und der Zusammensetzung und Aufbau des Modells enthält. Ein solches Dateiformat ist das Standard for the Exchange of Product model data (STEP) Format. Dabei handelt es sich, wie der Name bereits zu erkennen gibt, um den Standard für den Austausch von Produktmodelldaten. Dieser ist plattformunabhängig und kann von modernen CAD Systemen ausgegeben und gelesen werden [17] [1]. Allerdings unterstützt der Standard nicht die Übertragung von allen Funktionen von CAD [22]. Zum Beispiel ist es nicht möglich den bisherigen Entwurfszyklus in dem Format zu übertragen [25]. Eine weitere Möglichkeit in der Übertragung von CAD-Daten ist das in 1988 von Stereolithographie-Software entwickelte Standard Transformation Language (STL)-Format. Dieses Format kann ebenfalls von vielen Programmen genutzt werden. Allerdings überträgt es nur Geometrie-Daten in der Form von unstrukturierten triangulierten Oberflächen, basierend auf Normal-Vektoren und Dreieckspunkten in einem kartesischen Koordinatensystem [41] [38].

Für diese Arbeit wird aus der CAD-Datei die Geometrie in der Form einer Punktewolke extrahiert und benutzt, um das Modell in einer Szene wiederzufinden. Das Erkennen eines Objektes ermöglicht es den Ist-Zustand mit dem CAD-Modell zu vergleichen und weitere Informationen aus dem CAD-Modell dem realen Objekt zuzuordnen. Die in den STEP oder STL gespeicherten Daten werden in dieser Arbeit als CAD-Daten benannt.

3 Grundlagen der Objekterkennung

Die Objekterkennung ist ein Teilgebiet der Bildverarbeitung und beschäftigt sich mit der Erkennung von Objekten in jeglichen Daten. Viele der bekannteren und funktionierenden Algorithmen beschäftigen sich mit der Objekterkennung in 2D-Daten wie Bildern.

Es gibt viele unterschiedliche Methoden zur Objekterkennung. Eine Möglichkeit besteht darin, sogenannte Merkmale aus Bildern zu extrahieren, ähnliche Merkmale zu gruppieren und dieser Gruppierung einen Namen zuzuordnen [4] [21]. Andere Möglichkeiten beinhalten das Finden von Regionen von Interesse (Region Of Interest (ROIs)) mit denen ebenfalls eine Gruppierung und somit Klassifizierung durchgeführt wird [15]. Häufig wird der Erkennungsalgorithmus nach einer Segmentierung des Bildes durchgeführt. Dadurch sind die einzelnen Objekte bereits voneinander getrennt und werden nur noch durch den Algorithmus erkannt [4] [35]. Dieser Aufbau ist teilweise auch in Objekterkennungsalgorithmen für den 3D-Raum zu erkennen. Andere Algorithmen reduzieren das 3D-Problem auf einen Vergleich von 2D-Daten.

Beim Segmentieren von Szenen besteht die Gefahr von Unter- und Übersegmentierung. Untersegmentierung benennt den Zustand, wenn eine Szene in zu wenig Segmente unterteilt wird, wodurch das Objekt nicht komplett von anderen Teilen der Szene getrennt werden kann. Übersegmentierung beschreibt das Gegenteil, wenn eine Szene in zu viele Segmente unterteilt wird. Dadurch wird selbst das Objekt in kleinere Teile unterteilt. In beiden Fällen besteht die Möglichkeit, dass der Objekterkennungsalgorithmus das Objekt nicht mehr erkennt, Teile fehlen bzw. zusätzlich vorhanden sind. Die genauen mathematischen Definitionen von Unter- und Übersegmentierung werden von Koster K. et al. [26] erläutert.

Diese Arbeit widmet sich der Untersuchung von Algorithmen zur Objekterkennung in dreidimensionalen Räumen, weil die Erstellung eines 3D-Modells des erkannten Objekts von essenzieller Bedeutung ist und dies nur mit 3D-Daten möglich ist. Ein 3D-Modell eines Objektes bietet weitaus mehr Möglichkeiten zur Datenverarbeitung als 2D-Bilder. Zudem liefert die dreidimensionale Objekterkennung deutlich genauere Daten bezüglich der Position des Objektes als Methoden die auf zweidimensionalen Daten basieren [28]. Es ist möglich, dass selbst mit einer 2D-Objekterkennung die CAD-basiert ist, ein benutzbares 3D-Modell verfügbar ist. Allerdings wird häufig die tatsächliche Geometrie eines Objektes benötigt, welche in einer Punktewolke bzw. 3D-Scan wiedergespiegelt wird. Zudem bietet ein 3D-Scan weitaus mehr Details.

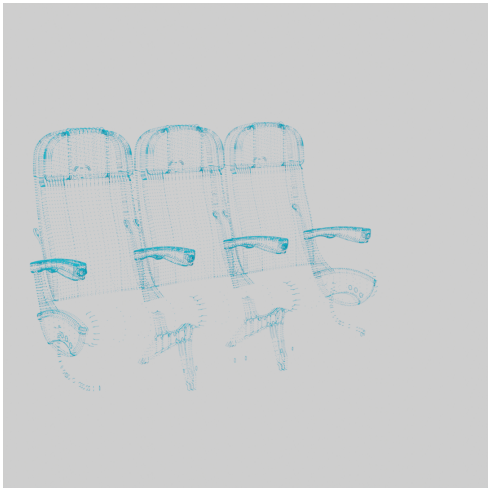
A.E. Johnson et al. [23] zeigen bereits mit dem Spin-Image-Algorithmus, wie das Erkennen von Objekten in einer Szene möglich ist. Des Weiteren zeigen K. Fu et al. [13], wie der RANdom SAmple Consensus (RANSAC) Algorithmus verwendet werden kann, um eine Transformation für eine Punktewolke zu finden. Aufgrund der Fortschritte in dem Bereich maschinelles Lernen werden ebenfalls Algorithmen in diesem Bereich der Künstlichen Intelligenz (KI) betrachtet. CAD-Modelle werden bisher nur für 2D Objekterkennung durch KIs genutzt. Zudem werden häufig manuelle Prozesse, wie das Erstellen eines Gitters für die Vereinfachung der CAD-Modelle eingebunden. Das CAD-Modell wird aus unterschiedlichen Winkeln aufgenommen, um Trainingsbilder für den Algorithmus zu generieren. Beispiele für eine Anwendung von KI für die Objekterkennung mit CAD-Modellen sind [43] [18] [37] [8] [44].

In dieser Arbeit wird lediglich der Spin-Image-Algorithmus ausgewertet. Die KI und der RANSAC-Algorithmus werden in dieser Arbeit kurz erläutert, aber nicht weiter angewendet. Der Umfang dieser Bachelorarbeit ermöglicht es nicht, alle drei Algorithmen ausführlich zu besprechen und zu vergleichen. Von den drei erläuterten Methoden bietet sich die genauere Betrachtung des Spin-Image Algorithmus an. Dieser hat die Vorteile, nicht auf Zufallswerte zu beruhen und keine große Menge an Trainingsdate zu benötigen. In diesem Kapitel werden die Algorithmen erläutert, wobei der Spin-Image-Algorithmus aufgrund seiner Implementation und Auswertung in dieser Arbeit deutlich ausführlicher beschrieben wird. Die KI und der RANSAC-Algorithmus werden nur kurz erläutert, um Vor- und Nachteile auszuführen die zu dem Fokus auf den Spin-Image-Algorithmus geführt haben.

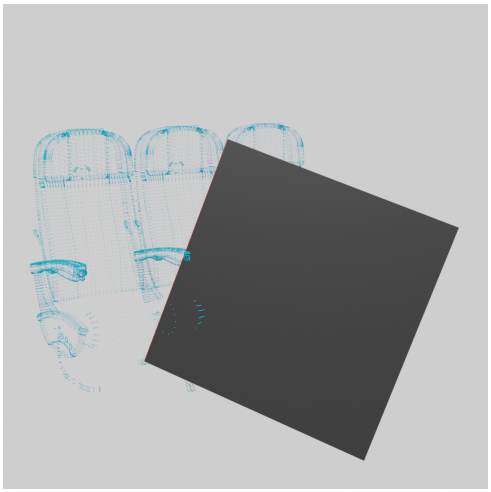
3.1 Spin-Image Algorithmus für die Objekterkennung

Der Spin-Image-Algorithmus ist eine Methode um 3D-Objekte zu beschreiben und wird in der Bildverarbeitung und Computergrafik verwendet. Der Deskriptor, der durch den Algorithmus generiert wird, fasst die geometrischen Eigenschaften einer Punktewolke zu einem 2D-Bild zusammen. Dafür wird eine zusammengesetzte Punktewolke und ein 3D-Modell von dem zu erkennenden Objekt benötigt. Der Algorithmus erstellt für beide 3D-Modelle ein sogenanntes Spin-Image, von dem dieser auch seinen Namen hat. Dabei handelt es sich um eine 2D-Aufnahme von einer zuvor bestimmten Seite des Objektes, welches die Intensität von Punkten darstellt. Die Darstellung für den Menschen erfolgt in einer Graustufung oder Wärmekarte. Die Seiten des verwendeten Farbspektrums beschreiben in diesem Fall eine Ansammlung von vielen bzw. wenig Punkten.

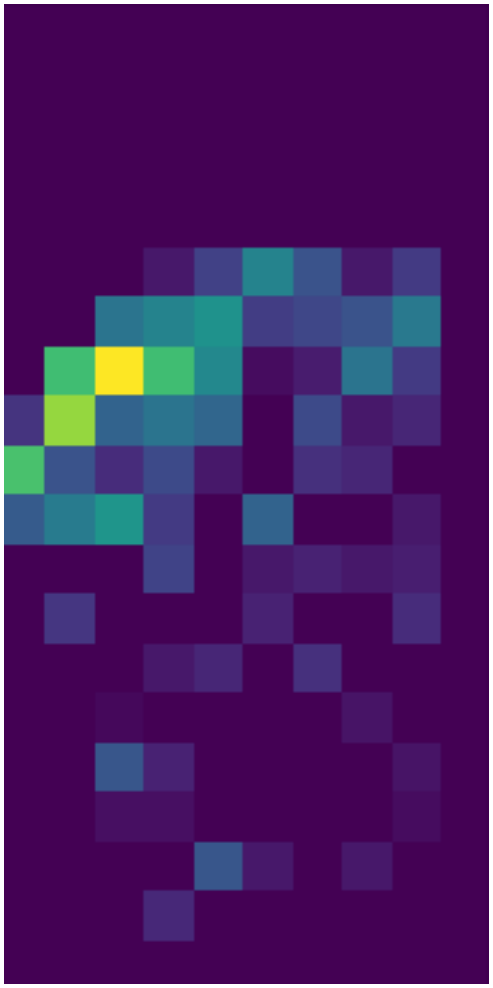
In der Abbildung 3.1a bis 3.1c sind die Schritte zum Erstellen eines Spin-Image zu erkennen. Diese wurden anhand einer Sitzbank aus einer Flugzeugkabine dargestellt. Die Sitzbank wird in einem in Kapitel 5 erläuterten Versuch benutzt, weshalb bereits an dieser Stelle die



(a) Punktewolke einer Sitzbank



(b) Generierung eines Spin-Image in einem Punkt von (a)



(c) Das generierte Spin-Image von (b)

Abbildung 3.1: Beispiel zum Generieren eines Spin-Image von der Sitzreihe

Funktionalität des Algorithmus mit ihr gezeigt wird. Im Folgenden werden die einzelnen Schritte zum Erstellen eines Spin-Image erklärt. Für das Bild 3.1c ist wichtig, dass in diesem Fall die Pixel heller werden, je mehr Punkte an der Stelle des Spin-Image notiert wurden.

Um das Spin-Image zu erstellen wird zunächst das 3D-CAD-Modell in eine Punktewolke umgewandelt, sofern es noch nicht in diesem Format vorliegt. Dafür wird die Oberfläche des Modells in diskrete Punkte unterteilt. Die Punktewolke ist in Abbildung 3.1a zu erkennen. Im Folgenden wird ein Punkt mit einer Ausrichtung ausgewählt. Die Punkteauswahl kann durch unterschiedliche Bedingungen getroffen werden. Wichtig ist jedoch, dass es zu dem benutzten Punkt eine Orientierung bzw. eine Normale gibt. Diese beschreibt die Ausrichtung des Punktes im Raum. Für die Auswahl der Punkte, von denen das Spin-Image aufgenommen wird, wird häufig die Referenz zu benachbarten Punkten gezogen.

Für diese Arbeit ist es ausschlaggebend die gewünschten Punkte mit einem Algorithmus deterministisch auszuwählen. Ansonsten ist eine Automatisierung der Objekterkennung nicht möglich.

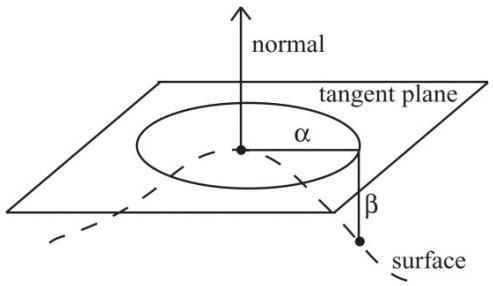


Abbildung 3.2: Aufbau des Spin-Image anhand der (α, β) Koordinaten [9]

Die Normale eines Punktes wird normalerweise mithilfe der Tangenten-Ebene, die den Punkt schneidet gebildet, wie in Abbildung 3.2 zu erkennen ist. In dem Fall der Punktewolke liegt allerdings keine Tangenten-Ebene vor. Diese wird mithilfe einer bestimmten Anzahl an Nachbarpunkten gebildet. Die Berechnung wird mithilfe des Nächste-Nachbarn-Klassifikation (KNN)-Algorithmus getätigt. Im Allgemeinen wird der Algorithmus dazu genutzt, die k nächsten Punkte von einem bestimmten Punkt zu berechnen. In diesem Fall legt der Algorithmus durch die nächsten k Punkte eine Tangenten-Ebene, die möglichst den gleichen durchschnittlichen Abstand zu allen Punkten hat, und berechnet deren Normalvektor. Mithilfe der Normalen lassen sich im Anschluss die Koordinaten anderer Punkte der Punktewolke auf das Koordinatensystem des Ausgangspunktes beziehen. Dafür wird auf der Tangenten-Ebene ein Kreis mit dem Ausgangspunkt als Mittelpunkt gelegt und dieser in Richtung seiner Normalen skaliert. Dadurch bildet sich ein Zylinder. Das ist in Abbildung 3.1b für einen Punkt dargestellt. Dieser Zylinder wird radial α und horizontal β in gleichmäßige Segmente unterteilt. Eine Skizze zum Ermitteln der (α, β) Koordinaten ist in Abbildung 3.2 wiederzufinden. Durch die radiale und horizontale Anordnung von den Segmenten ergibt sich ein 2D-Bild. Dieses ist in Abbildung 3.1c wiederzufinden. In jedem Segment wird die Anzahl an Punkten oder eine Kombination an Normalvektoren berechnet. Diese Werte werden in den Spin Images als Helligkeit der Pixel ausgegeben. [12]

Sobald ein Punkt ausgewählt wurde und dessen Position p^*, Ausrichtung n^* und alle anderen Punkte der Punktewolke v bekannt sind, kann das Spin-Image berechnet werden.

$$v = [v_1, v_2, \ldots, v_n]$$

$$v \in Punktewolke, v_{1-n} \in \mathbb{R}^3$$

$$p^* \in Puntewolke, p^* = StartPunkt, p^* \in \mathbb{R}^3$$

$$n^* \in \mathbb{R}^3, |n| = 1$$

n^* legt die Orientierung der Fläche, die durch p^* gelegt wird, fest. Daraus wird die Differenz d zwischen den Punkten v und der Position p^* gebildet. Diese Berechnung ist in Gleichung (3.1) zu erkennen.

$$d_i = v_i - p^* \tag{3.1}$$

Die Differenz d_i wird dazu verwendet, um den horizontalen Abstand β des Punktes v_i mithilfe der Ausrichtung n^* zu bilden. Dafür wird das Skalarprodukt von der Richtung n^* und Differenz d berechnet. Das Skalarprodukt von zwei Vektoren u und s ist in der Gleichung (3.2) definiert.

$$u \cdot s = u_1 * s_1 + u_2 * s_2 + u_3 * s_3 \tag{3.2}$$

Demnach wird das Skalarprodukt in Gleichung 3.2 der Richtung n^* und Differenz d nach der Gleichung (3.3) berechnet. Das Skalarprodukt aus Gleichung (3.3) bildet den Wert für den horizontalen Abstand β [9] [24] [39] [3] [27].

$$\beta_i = n^* \cdot d_i = n_1^* * d_{i_1} + n_2^* * d_{i_2} + n_3^* * d_{i_3} \tag{3.3}$$

Infolge wird die euklidische Norm n_i jedes Punktes auf der x-Achse mit der Gleichung (3.4) berechnet.

$$n_i = \|v_i - p^*\| \tag{3.4}$$

Mithilfe von dem Satz des Pythagoras aus Gleichung 3.5 ergibt sich die Gleichung (3.6) für den Abstand α zum ausgewählten Punkt p^* [9] [24] [39] [3] [27]. β_i^2 wird als Projektion bezeichnet.

$$a^2 + b^2 = c^2 \tag{3.5}$$

$$\alpha_i = \sqrt{n_i^2 - \beta_i^2} \tag{3.6}$$

Zusammengefasst ergeben die Gleichungen 3.1 bis 3.6 die Gleichung (3.8) und erfüllt somit die Bedingung 3.7 [9] [24] [39] [3] [27]. 0 beschreibt in diesem Fall den Punkt p^* mit der Ausrichtung n^*.

$$S_0(v) : \mathbb{R}^3 \to \mathbb{R}^2 \tag{3.7}$$

$$S_0(v) \to (\alpha, \beta) = (\sqrt{\|v - p^*\|^2 - (n^* \cdot (v - p^*))^2}, n^* \cdot (v - p^*)) \tag{3.8}$$

Die hier festgelegten Werte für α und β stellen allerdings noch nicht die Koordinate für das Spin-Image dar. Bisher definieren diese nur den radialen und horizontalen Abstand (α_i, β_i) eines Punktes v_i zu dem Punkt p^*.

Diese werden nun in das Spin-Image umgewandelt. Dafür wird eine Größe für das Bild *width* und Verhältnis *scale* definiert. Die Größe *width* gibt dabei die Pixelbreite und Höhe des Bildes an. Das Verhältnis *scale* definiert wie viele Punkte von der Punktewolke in das Spin-Image übertragen wird.

$$width \geq 1, scale \geq 1 \mid width \in \mathbb{R}, scale \in \mathbb{R}$$

Daraus lässt sich die Matrix (3.9) für das Bild $Image$ bilden. Das Bild wird auf die Breite *width* und die doppelte Höhe zugeschnitten, um Punkte über und unter dem Referenzpunkt zu erfassen.

$$Image_{m \times n} = \begin{bmatrix} h_{m,0} & h_{m,1} & \cdots & h_{m,n} \\ \vdots & \vdots & \iddots & \vdots \\ h_{1,0} & h_{1,1} & \cdots & h_{1,n} \\ h_{0,0} & h_{0,1} & \cdots & h_{0,n} \\ h_{-1,0} & h_{-1,1} & \cdots & h_{-1,n} \\ \vdots & \vdots & \ddots & \vdots \\ h_{-m,0} & h_{-m,1} & \cdots & h_{-m,n} \end{bmatrix} \mid m = width{-}1, n = width{-}1, h_{m,n} = 0, h_{m,n} \in \mathbb{N} \tag{3.9}$$

Alle Werte der Matrix $Image$ werden zunächst auf null gesetzt. Im Anschluss werden die radialen und horizontalen Abstände α und β mithilfe des Verhältnis *scale* skaliert und zusammen in der Liste an neuen Punkten *normalized_points* wie in Gleichung (3.10) gesammelt.

$$normalized\_points_i = (\frac{\alpha_i}{scale}, \frac{\beta_i}{scale}) \tag{3.10}$$

Mit den Punkten aus Gleichung (3.11) werden die Punkte $normalized\_points_i$, die außerhalb des Bildes $Images$ liegen, herausgefiltert. Weil die Werte in der Gleichung (3.10) bereits mit *scale* skaliert wurden, werden in (3.11) alle Koordinaten, die außerhalb dieser Skalierung liegen, entfernt. Die verbleibenden Punkte $settable\_points_j$ werden in der Gleichung (3.12) definiert.

$$settable\_indices = \{i \in 1, \ldots, n \mid \left|normalized\_points_{i,k}\right| < 1 \mid k \in 0, 1\} \tag{3.11}$$

$$settable\_points_j = \{normalized\_points_j * (width - 1) \mid j \in settable\_indices\} \tag{3.12}$$

Die Koordinaten der Punkte $settable\_points_j$ werden in der Gleichung (3.13) abgerundet, um für das Bild jede Koordinate, an der ein Punkt vorhanden ist, um 1 zu erhöhen, wie in Gleichung (3.14) zu erkennen.

$$indices_j = (\lfloor settable\_points_{j,0} \rfloor, \lfloor settable\_points_{j,1} \rfloor) \tag{3.13}$$

$$h_{m,n} = \#indices_j \mid indices_j = (m, n), j \in settable\_indices \tag{3.14}$$

Das berechnete Spin-Image $Image$ ist nun dem Punkt p^* zuzuordnen.

Sobald das Spin-Image für sowohl die Szene als auch das zu findende Objekt erstellt wurde, kann mit einem 2D-Bildvergleichsalgorithmus festgestellt werden, ob das Objekt im Bild ist. In diesem Fall wird für den Bildvergleich die Korrelation der beiden zu vergleichenden Spin-Images berechnet. Sofern die Korrelation über $p = 0.95$ liegt, wird davon ausgegangen, dass die Spin-Images das gleiche Objekt darstellen. Der Wert wurde in dem Bereich gewählt, um zu verhindern, dass lange Laufzeiten des Algorithmus entstehen. Zudem weist das CAD-Modell deutliche Unterschiede zu dem tatsächlichen Sitzt auf. Mit dem Wert von $p = 0.95$ wird dem etwas entgegengesteuert. Wenn ein passendes Bild gefunden wurde, wird mithilfe der benutzen Punkte die Position des Objektes berechnet. Aus den beiden Referenzpunkten wird ein Translations-Vektor und eine Rotations-Matrix mithilfe der Gleichungen 3.15, 3.16 und 3.17 berechnet.

$$T(p, n_p, c, n_c) \to t, R \mid t \in \mathbb{R}^3, R \in \mathbb{R}^{3x3} \tag{3.15}$$

$$t_i = c_i - p_i \mid i \in [0, 1, 2] \tag{3.16}$$

$$R_{3\times3} = \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix} \tag{3.17}$$

Mithilfe der Transformation ist es möglich, das CAD-Modell mit der Szene zu überlagern. Das Ergebnis des Spin-Image-Algorithmus lässt zwei Arten von Fehlerzuständen zu. Diese Fehlerzustände bestehen aus dem Erkennen eines nicht vorhandenen Objektes und dem nicht erkennen eines vorhanden Objektes. Entgegengesetzt dazu gibt es natürlich auch zwei Arten von richtigen Zuständen. Diese beinhalten ein erkanntes vorhandenes Objekt und ein nicht vorhandenes nicht erkanntes Objekt.

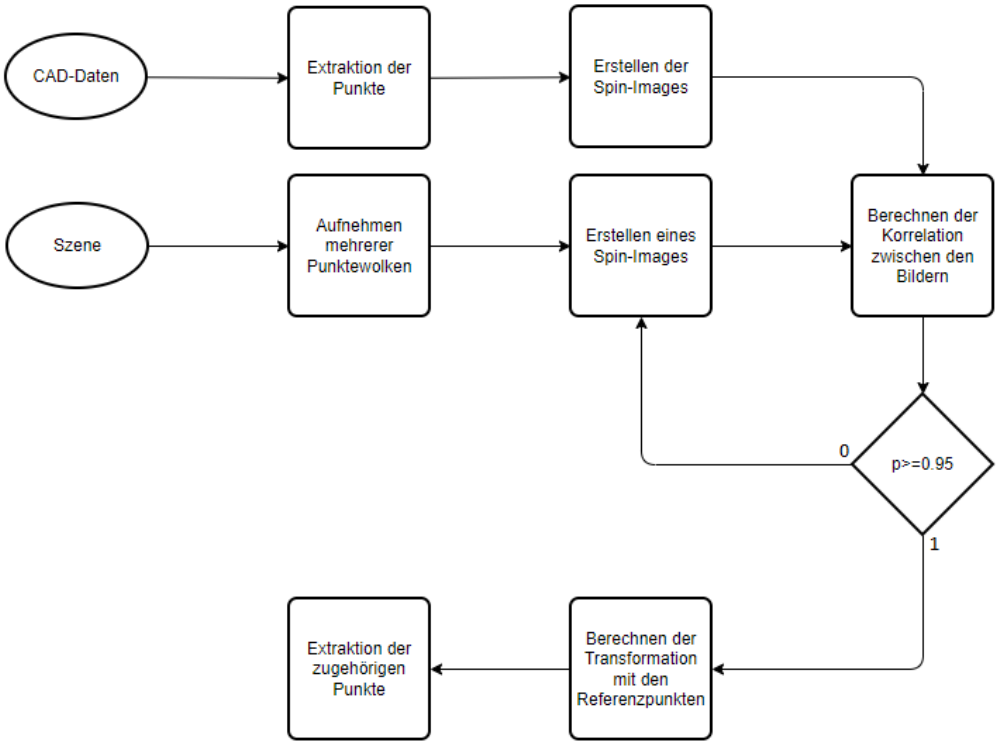


Abbildung 3.3: Überblick zu dem Spin-Image-Algorithmus

Die zuvor erläuterten Schritte sind in Abbildung 3.3 zusammengefasst. Die Abbildung zeigt die Schritte zum Erstellen des Spin-Images, dem Vergleich dieser und überlappen der Punktewolken. Dafür wird zunächst die Punktewolke der CAD-Modelle extrahiert und alle Spin-Images dazu berechnet. Im Anschluss wird von der Punktewolke der Szene ein Spin-Image erstellt und die Korrelationskoeffizienten zu den der CAD-Modelle gebildet. Sofern einer der Korrelationskoeffizienten größer als $p > 0.95$ ist, wird die Transformation zwischen den Punktewolken mithilfe ihrer Referenzpunkte berechnet. Das CAD-Modell wird auf die Szene transformiert und die zugehörigen Punkte extrahiert. Falls keiner der Werte über 0.95 liegt, wird das nächste Spin-Image der Szene berechnet. Sobald alle möglichen Spin-Images berechnet und kein Objekt erkannt wurde, ist das Objekt nicht in der Szene vorhanden.

Aus dieser Darstellung des Spin-Image-Algorithmus wird bereits ein großer Nachteil, dem linearen Verlauf für jedes weitere zu erkennende Objekt, offensichtlich. Das wird in dieser Arbeit nicht weiter besprochen, allerdings ist zu erwähnen, dass A.E. Johnson et al. [23] bereits einen Lösungsansatz vorstellt. Diese Lösung hat ebenfalls der Entscheidung des genaueren Ausarbeitens dieses Algorithmus beigetragen.

3.2 RANSAC Algorithmus mit Merkmalserkennung für Objekterkennung

Der RANSAC Algorithmus ist ein Algorithmus, der aus zufälligen Schätzungen besteht. Dafür werden wie in Kapitel 3.1 Punktewolken der Objekte benötigt. Um den RANSAC Algorithmus zu benutzen wird die Szene segmentiert. Im Anschluss wird jedes Segment mit dem Vergleichsobjekt verglichen. Aus diesen werden Merkmale der Punktewolke mithilfe eines K-dimensionalen Baums (KDTree) mit KNN bestimmt. Durch die Segmentierung der Szene und das einzelne Vergleichen der einzelnen Segmente mit der zu erkennenden Komponente entsteht ein großer Zeitaufwand.

Der RANSAC Algorithmus bestimmt zunächst zufällig die minimale Anzahl an Punkten, um die Modell-Parameter zu bestimmen. Daraus lässt sich bestimmen, wie viele Punkte von allen Punkten der zuvor bestimmten Toleranz ϵ zu den Parametern passen. Sofern die Anzahl an Punkten einem bestimmten Grenzwert τ entspricht oder überschreitet, werden die Modell-Parameter neu bestimmt und die Schleife beendet. Ansonsten werden die zuvor genannten Schritte maximal N-mal wiederholt. [11][13][7]

Daraus ergibt sich eine Transformation, mit der die Punktewolken übereinander gelegt werden. Die Segmentierung der Szene ermöglicht es bereits nur die Punkte des gewünschten Objektes zu extrahieren.

Aufgrund der Zufällig generierten Parameter für dieses verfahren und des zusätzlichen Zeitaufwandes der durch die Segmentierung und das einzelne Betrachten der Elemente im Nachhinein entsteht, wird dieser Algorithmus in dieser Arbeit nicht weiter betrachtet. Zudem sorgt eine Skalierung des Algorithmus zu deutlich höheren Zeitaufwänden. Dafür würde jedes Segment auf alle zu suchenden Komponenten untersucht werden. Zusätzlich besteht die zuvor erwähnte Gefahr von Über- und Untersegmentierung.

3.3 KI für Objekterkennung

Wie bereits erläutert wird die KI hauptsächlich genutzt, um mithilfe von CAD-Modellen Bilderkennung zu ermöglichen. Das lässt sich natürlich auch auf die Erkennung von Objekten in Szenen mithilfe von Punktewolken anwenden. Dafür würden Bilder von den Punktewolken gemacht werden, in welcher die KI die gewünschten CAD-Modelle erkennen kann. Bei einer solchen Methode müsste die Punktewolke allerdings zuvor eingefärbt werden, um es der KI zu ermöglichen etwas in den Bildern zu erkennen. Eine andere Möglichkeit bestände darin eine KI mit anderen Algorithmen zu verbinden. Ein Beispiel wäre den Bildvergleich in dem Spin-Image Algorithmus 3.1 durch eine KI zu ersetzen.

Die Vorteile des Nutzens einer KI bestehen hauptsächlich aus der Laufzeit des Algorithmus. Sobald die KI auf die gewünschten CAD-Modelle trainiert ist, werden diese schnell und mit einer gewissen Wahrscheinlichkeit, die in der Regel bei 95 % liegt, erkannt [5]. Daraus ergeben sich allerdings auch direkt die Nachteile beim Nutzen von KIs. Zum einen gibt es immer eine Wahrscheinlichkeit, dass ein Objekt nicht erkannt wird. Weiteres Trainieren kann zu einer Verschlechterung der Wahrscheinlichkeit führen. Zudem muss die KI für jedes neue Objekt trainiert werden. Dadurch dauert es eine Weile neue Objekte zu erkennen und das Hinzufügen neuer Objekte kann die Konfidenz für andere bereits trainierte Objekte verschlechtern. Das Trainieren erfordert große Datensätze, die derzeit noch nicht zur Verfügung stehen und dessen sammeln in diesem Bereich einen großen Zeitaufwand haben. Zudem ist das manuelle zuordnen von CAD-Modellen in einer Szene äußerst Zeitaufwändig [26].

Aus diesen Gründen werden KIs, die dem Deep Learning Prinzip folgen, in dieser Arbeit nicht weiter betrachtet. Stattdessen konzentriert sich diese Arbeit auf den Spin-Image-Algorithmus, mit dessen Hilfe eine Möglichkeit zu einer späteren Betrachtung des Einsatzes von KIs ermöglicht wird.

4 Anforderungsanalyse des Algorithmus

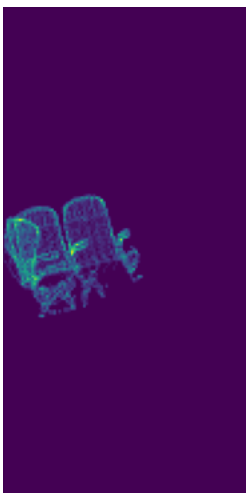
Im Folgenden werden Anforderungen beschrieben, um den Algorithmus zu bewerten. Zunächst wird der Algorithmus untersucht und ausgewertet. Dafür werden die änderbaren Parameter des Algorithmus, wie die Größe des Bildes *width*, das Verhältnis *scale* und unterschiedliche Bedingungen zum Aufnehmen von Punkten in das Spin-Image, untersucht. Diese werden systematisch geändert um Abhängigkeiten zu den Werten zu beleuchten. Die systematischen Änderungen werden in Graphen eingetragen und die Erfolgsrate des Algorithmus auf diese projiziert. Die Erfolgsrate wird in diesem Fall durch den jeweils gebildeten Korrelationskoeffizient p dargestellt, um den genauen Einfluss der jeweiligen Faktoren zu betrachten. Dadurch ergibt sich die jeweilige Abhängigkeit zu diesen Parametern bzw. deren Idealwerte.

4.1 Erarbeitung der Parameter für den Algorithmus

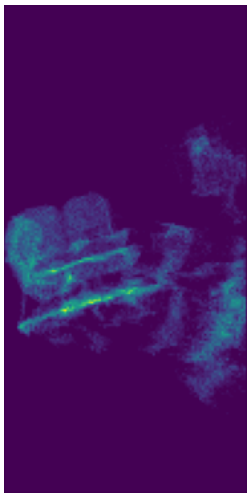
Zunächst wird die Auswirkung der Bildgröße *width*, des Verhältnisses *scale* erarbeitet. Zudem wird die Auswirkung von Rauschen auf den Korrelationskoeffizient ausgewertet. Zusätzlich besteht die Möglichkeit die Bedingung, eines maximalen Winkels *angel* zwischen den Normalen des Referenzpunktes von dem Spin-Image und dem Normalen des Punktes der hinzugefügt wird, auszuwerten. Dieser Wert entfernt alle Punkte, die nicht in eine ähnliche Richtung orientiert sind, wie der Ausgangspunkt. Dieser Wert sollte verändert werden, je nach Aufnahme der Szene. In diesem Fall werden die unterschiedlichen Werte anhand zweier Punktewolken mit dem gesamten Objekt in der Szene betrachtet. Deshalb wird der Winkel in diesem Fall bei $angle = 180°$ gehalten, wodurch alle in der Szene enthaltenen Punkte für das Spin-Image benutzt werden.

In der Abbildung 4.1 sind die Spin-Images des CAD-Modells in Bild 4.1a und 4.1d und der aufgenommenen Szene in Bild 4.1b und 4.1e zu erkennen. Die Spin-Images wurden mit jeweils unterschiedlichen Werten für die Bildgröße *width* und das Verhältnis *scale* gebildet. Die maximalen Winkel *angel* zwischen der Normalen der Punkte wurde aufgrund der Art der Aufnahme nicht betrachtet. Der Winkel eignet sich dazu die Punkte des 3D-Modell an die Szene anzupassen, sofern nur Aufnahmen aus einer Richtung getätigt wurden. In Anwendung auf die hier gezeigten Spin-Images sorgen geringere Winkel dafür, dass weniger von dem Objekt erkennbar ist und somit eine geringere Korrelation zwischen den Bildern entsteht. In Abbildung 4.1 die Streudiagramme 4.1c und 4.1f dargestellt, damit zwischen

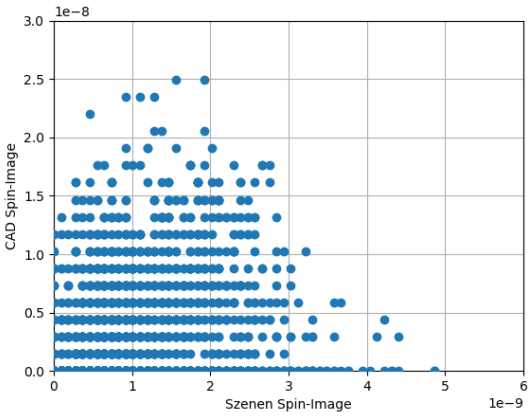
Aufnahme des Spin-Image mit $width = 100$, $scale = 2.5$ und $angle = 180°$



(a) Spin-Image CAD

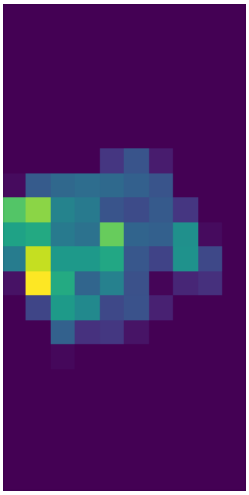


(b) Spin-Image Szene

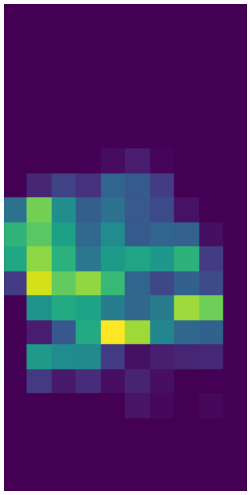


(c) Streudiagramm als Visualisierung der Korrelation ($p = 0.4805$) zwischen 4.1a und 4.1b

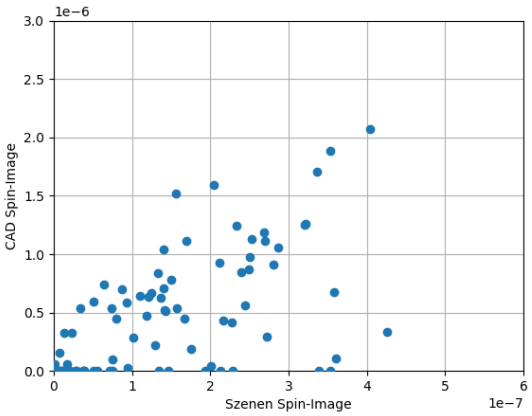
Aufnahme des Spin-Image mit $width = 10$, $scale = 1.5$ und $angle = 180°$



(d) Spin-Image CAD



(e) Spin-Image Szene



(f) Streudiagramm als Visualisierung der Korrelation ($p = 0.7451$) zwischen 4.1d und 4.1e

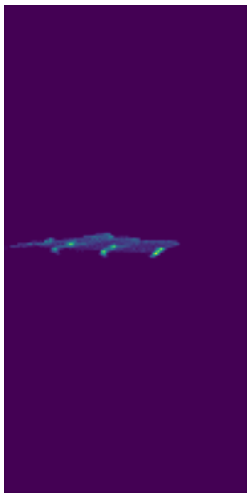
Abbildung 4.1: Beispiel zum Vergleich zweier Spin-Images von der Sitzreihe

diesen ein Vergleich der Korrelation gezogen werden kann. Die Streudiagramme stellen die relative Punktedichte der aufgenommenen Punkte aus der Punktewolke respektive für jeden Punkt in den Spin-Images, von dem CAD-Modell und der Szene, dar.

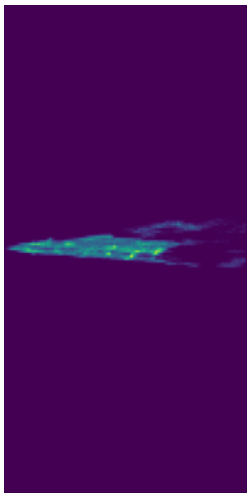
Aufgrund der größeren Werte für die Bildgröße *width* zeigen die Spin-Images 4.1a, 4.1b, 4.2a und 4.2b mehr Details als die Spin-Images 4.1d, 4.1e, 4.2d und 4.2e. Das Ändern des Verhältnisses *scale* hat Einfluss auf die Größe des Anteils der Punktewolke, die in dem Spin-Image abgebildet ist.

Diese Veränderungen in den Spin-Images haben Auswirkungen auf die Korrelation zweier Bilder, welche genutzt wird, um das Objekt in der Szene zu finden. Die Abbildung 4.1 zeigt, dass mit niedrigeren Werten eine höhere linear Korrelation vorliegt. Das bedeutet, dass die

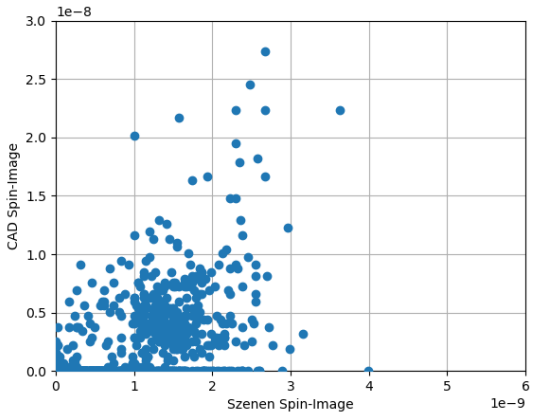
Aufnahme des Spin-Image mit $width = 100$, $scale = 2.5$ und $angle = 180°$



(a) Spin-Image CAD



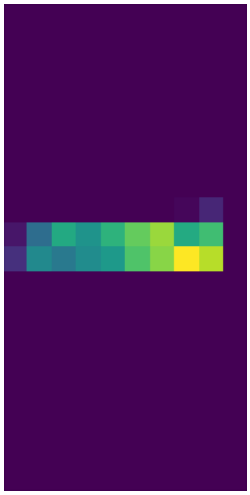
(b) Spin-Image Szene



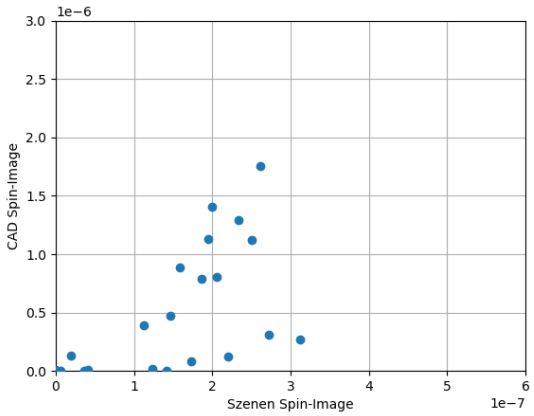
(c) Streudiagramm als Visualisierung der Korrelation ($p = 0.6824$) zwischen 4.2a und 4.2b

Aufnahme des Spin-Image mit $width = 10$, $scale = 1.5$ und $angle = 180°$

(d) Spin-Image CAD



(e) Spin-Image Szene



(f) Streudiagramm als Visualisierung der Korrelation ($p = 0.8001$) zwischen 4.2d und 4.2e

Abbildung 4.2: Beispiel zum Vergleich zweier Spin-Images von der Sitzreihe

Bilder sich ähnlicher sind und zueinander zugeordnet werden können. Dieser Schluss kann aus der Linearisierung der Punkte gezogen werde. In Abbildung 4.1c sind die einzelnen Korrelationen weit verteilt. Die Abbildung 4.1f hingegen zeigt, dass mit den niedrigeren Werten die Punkte weitaus linearer verteilt sind. Das bedeutet, dass ein proportionaler Anstieg zwischen den Helligkeiten in den Spin-Images bei beiden Spin-Images gleich ist. Daraus lässt sich auf eine gleiche Verteilung von Punkten in der Punktewolke schließen und somit auf ein identisches Objekt.

Die Abbildung 4.2 zeigt das gleiche Vorgehen wie in Abbildung 4.1. Der einzige Unterschied ist der Aufnahmepunkt der beiden Spin-Images. Hier ist zu erkennen, dass die unterschiedlichen Aufnahmepunkte ebenfalls einen Einfluss auf die Korrelation haben und

dass die Änderung der Werte zur Bildgröße *width* und Verhältnis *scale* bei unterschiedlichen Spin-Images unterschiedlich große Auswirkungen haben. Der Unterschied in der Korrelation zwischen Abbildung 4.2c und 4.2f ist ca. 0.1 während zwischen Abbildung 4.1c und 4.1f ein Unterschied von ca. 0.25 liegt. Für den Verlauf der Arbeit ist zu beachten, dass die Abbildung 4.2 zwar einen größeren Wert für die Korrelation der Spin-Images aufweist, aber das nicht unbedingt bedeutet, dass das gleiche Objekt zu erkennen ist.

Im Anschluss wird die Auswirkung einer Störung auf den Korrelationskoeffizient überprüft, um zu erfassen, wie viel Störung in Form von Rauschen in den Punktewolken vorhanden sein darf. Dafür wird der Punktewolke künstlich immer mehr Rauschen hinzugefügt. Das wird mithilfe eines Programmes getan, welches jeden Punkt um einen zufälligen Wert mit einer Amplitude verschiebt. Um die Auswirkungen darzustellen wurde wie in Abbildung 4.3 die Amplitude des Rauschens gegenüber des Korrelationskoeffizienten in einem Graphen aufgetragen.

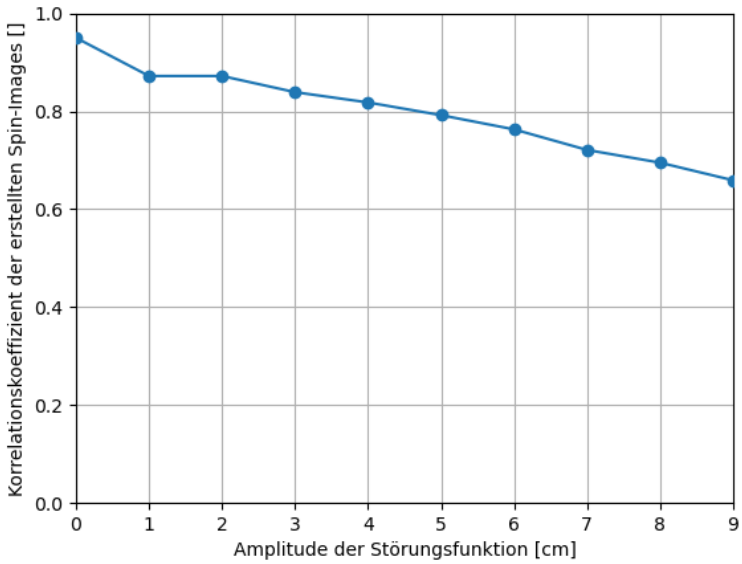


Abbildung 4.3: Auswertung des Einflusses von Störungen auf die Korrelation

In Abbildung 4.3 ist eben diese Amplitude in Relation zu der Größe der in der Punktewolke abgebildeten Sitzbank gewählt. Dafür wurden die Werte von null bis neun jeweils in dem Graphen aufgetragen. Hier ist zu erkennen, dass sobald das Rauschen auftritt der Korrelationskoeffizient sinkt. In dieser Arbeit wurde wie bereits in Kapitel 3.1 erwähnt ein Korrelationskoeffizient von $p = 0.95$ gewählt. Dadurch werden selbst Werte mit einem geringen Rauschen, wie $1cm$ nicht erkannt. Es wurden in dieser Abbildung keine kleineren Werte für das Rauschen betrachtet, weil durch Filter, die hinterher auf das Objekt gelegt werden das Rauschen unter $1cm$ herausgefiltert wird.

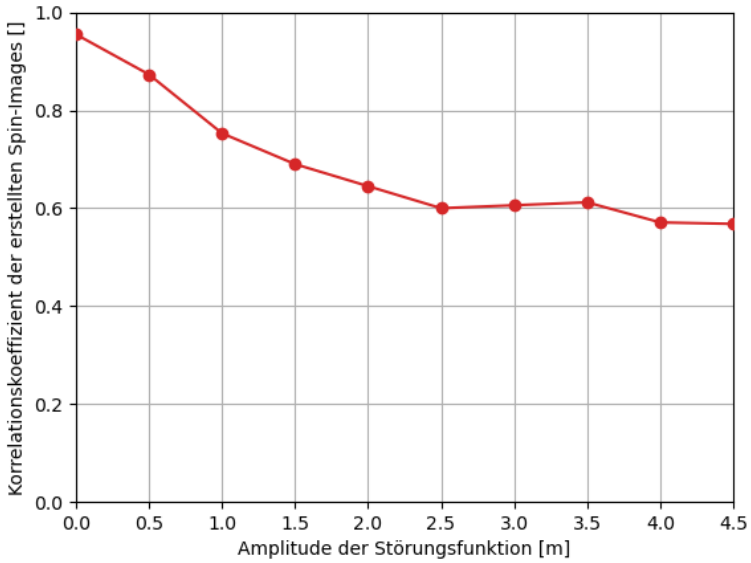
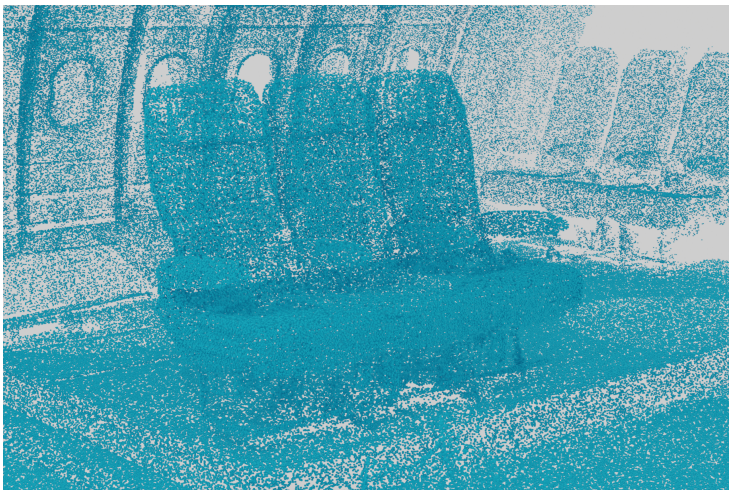


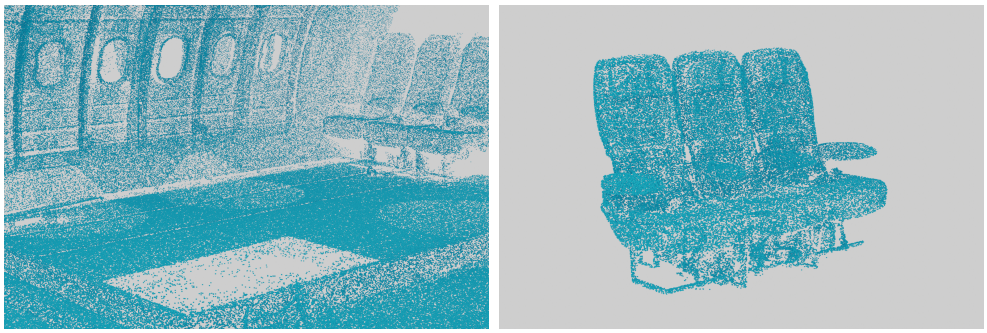
Abbildung 4.4: Auswertung des Einflusses von Störungen auf die Korrelation beim Modell des ISTAR

In Abbildung 4.4 werden dieselben Werte für das Model eines Flugzeuges gebildet. Dabei ist zu erkennen, dass bei dem Flugzeug aufgrund seiner Größe im Vergleich zu der Sitzbank das Rauschen eine deutlich höhere Amplitude benötigt, um den Korrelationskoeffizient ausreichend zu verringern, sodass das Flugzeug nicht mehr erkannt wird. Das bedeutet, dass die Größe des Objektes den Einfluss des Rauschens bestimmt und dass bei kleineren Objekten eine geringere Toleranz gegenüber von Scan Ungenauigkeiten wie Rauschen vorliegt.

Aus den hier dargestellten Werten ergeben sich die Bildgröße $width = 10$, das Größenverhältnis $scale = 1.5$ und der Winkel $angle = 180°$. Der Winkel ist hierbei für diese Arbeit ein temporärer Wert und muss, sobald Grundlagen zu dem Algorithmus stehen weiter untersucht werden. Dafür bietet es sich an, eine Ausarbeitung auf alleine diesen Faktor in Bezug auf die Anzahl an Aufnahmen und Winkel zu tätigen. Dies erfordert allerdings einen großen Rechenaufwand und ist somit für diese Arbeit nicht plausibel. Die Fortsetzung würden untersuchen, ob eine niedrigere Anzahl an Aufnahmen oder kleinere Durchschnittswinkel möglich sind, sofern der Winkel *angle* verringert wird.



(a) Visualisierung der Szene vor der Extraktion des Sitzes



(b) Visualisierung der getrennten Komponenten

Abbildung 4.5: Validierung des Algorithmus mit den Parametern aus Kapitel 4.1

4.2 Validierung des Algorithmus mit den erarbeiteten Parametern

Die zuvor erarbeitete Parameter von einer Bildgröße $width = 10$, Größenverhältnis $scale = 1.5$ und Winkel $angle = 180°$ wurden im Anschluss für die CAD-Daten des Sitzes und eine Szene, aufgenommen mit einem Hochleistungsscanner getestet, um die Funktionalität des Algorithmus unter den beschriebenen Parametern zu gewährleisten.

In Abbildung 4.5a ist dabei zunächst die gesammte Szene einheitlich eingefärbt zu erkennen. Der verwendete Sensor hat die Möglichkeit, Farben aufzunehmen. Allerdings spielt diese für den Algorithmus keine Rolle, weshalb sie hier ebenfalls nicht dargestellt wurde. Im Vordergrund ist die Sitzreihe zu erkennen, während der Hintergrund eine weitere Sitzbank zeigt und die Innenseite einer Flugzeugkabine darstellt. In den darunterliegenden Abbildungen 4.5b sind das extrahierte Modell des Sitzes und die Szene, aus dem das Modell extrhiert wurde zu erkennen. Dabei ist zunächst festzustellen, dass um den Sitz etwas mehr Punkte fehlen, als in der Sitzpunktewolke zu erkennen sind. Das ist auf die Methode der Extraktion zurückzuführen, bei der der Sitz zunächst grob ausgeschnitten wird. Des

Weiteren fällt auf, dass die hintere Sitzreihe nicht von dem Algorithmus erkannt wird. Das liegt daran, dass für das Erstellen der Spin-Images hier ein Winkel von $angle = 180°$ genutzt wird. Die hintere Sitzreihe wird nur aus einer Richtung aufgenommen, weshalb in den Spin-Images keine ausreichende Korrelation erkannt wird. In der erkannten Sitzreihe ist zudem immernoch ein Teil an nicht zughörigen Punkten zu erkennen. Demnach muss die Extraktion weiter angepasst werden, um ein besseres Model erstellen zu können. Für diese Arbeit wird dies jedoch nicht mehr getan, weil sich dazu ebenfalls eine Versuchsreihe anbieten würde.

5 Algorithmus Implementierung in einem Versuchsaufbau zur Analyse von Parametern

Diese Arbeit beschäftigt sich damit, Objekte mithilfe von CAD-Modellen in einer Szene zu erkennen. Für den zuvor vorgestellten Spin-Image-Algorithmus in Kapitel 3 werden aus dem erklärten CAD-Format bestimmte Daten, wie Punkte und Normalen, gebraucht. Zudem werden Daten benötigt, mit denen der erläuterte Algorithmus ein Objekt erkennen kann. Dafür werden in der Laborumgebung kontrolliert Scans durchgeführt.

5.1 Beschreibung des Versuchsaufbaus

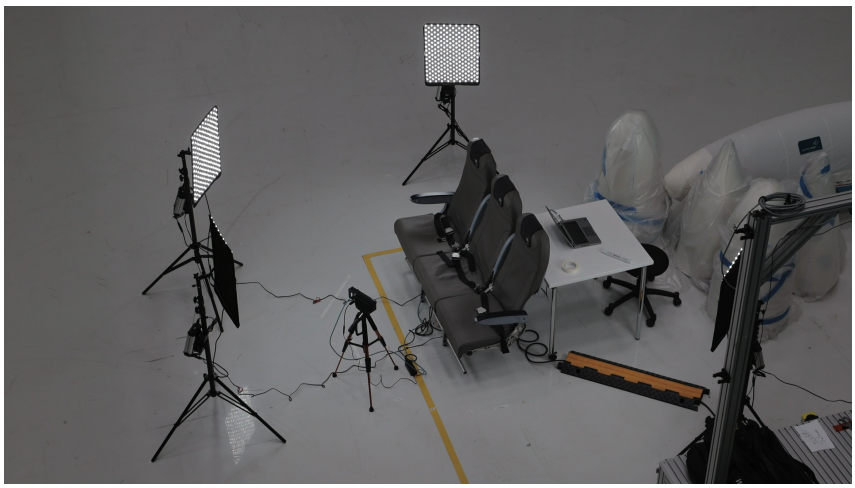


Abbildung 5.1: Versuchsaufbau zum Scannen der Sitzreihe

Für das Messen des Datensatzes wurde die Sitzbank gescannt. Dafür wurde ein Aufbau, wie in Abbildung 5.1, aufgebaut. Die zu scannende Sitzbank ist dabei in der Mitte des Aufbaus platziert. Um diese herum sind vier Dauerlichter aufgestellt, die so platziert sind, dass Licht auf die zu scannenden Flächen fällt. Der Scanner wird für die unterschiedlichen Scans im Kreis um den Mittelpunkt der Sitzreihe gedreht. Dabei wird die Höhe des Scans nicht verändert. Die Entfernung zu dem Objekt wurde für den Versuch auf unterschiedliche Werte angepasst. Wie auf dem Bild zu erkennen, wurde hinter die Sitzbank teil zeitig ein Tisch gestellt, um von dort aus die Scans zu steuern. Zum Aufnehmen der Bilder wurde

das Programm der Projektarbeit von Luca Tiedemann [40] verwendet. Allerdings konnten die Programmabschnitte zum Zusammensetzen der Punktewolken nicht benutzt werden, weil die Aufnahmen nicht alle direkt zusammengesetzt werden mussten.

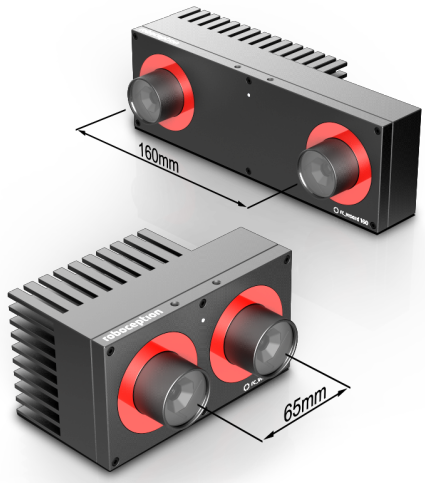


Abbildung 5.2: Verwendeter Scanner: rc_visard

Bei dem verwendeten Scanner handelt es sich um die Roboception rc_visard. Die rc_visard ist eine intelligente 3D-Kamera, die speziell für Roboteranwendungen entwickelt wurde. Sie vereint hochauflösende Stereo-Bildumgebungen mit Tiefeninformationen und ermöglicht so eine umfassende und präzise Wahrnehmung der Umgebung. Wie in Abbildung 5.2 zu erkennen, nutzt die Kamera zwei Objektive mit einem fest definierten Abstand, um die Tiefen- und Stereo-Bilder zu ermöglichen. Es sind in der Abbildung 5.2 zwei unterschiedliche Versionen der Kamera zu erkennen, die sich nur in der Entfernung der Objektive unterscheiden. Das hat Auswirkungen auf verschiedene Werte, wie den Sichtbereich, die Tiefenauflösung und das Gewicht. Für diese Arbeit wurde die 160-er Version des Scanners verwendet. Diese Kamera hat einen Sichtbereich von 0,5 Metern bis 3 Metern Entfernung und bis zu ca. 3,5 Metern Breite. Die Kamera kann mit einer Genaueigkeit von 1,5 Millimetern in 2 Metern entfernung messen. [16]

Die Kamera wurde für die Scans ausgewählt, weil sie speziell für Roboter entworfen wurde und somit sehr gut zu der Anwendung von Roboter durchgeführten Scans passt. Zudem ist die Kamera im Vergleich zu anderen Sensoren zum 3D-Scannen eine der kostengünstigeren Versionen. Das ermöglicht eine breitere Anwendung der Ergebnisse dieser Arbeit.

5.2 Versuchsdurchführung

Die Scans wurden aus acht verschiedenen Richtungen aufgenommen. Zudem wurde mithilfe der Dauerlichter die Beleuchtungsstufe verändert. Für die Anzahl an Beleuchtungsstufen wurden drei gewählt. Alle Aufnahmen wurden in fünf Abstandsstufen getätigt. Dadurch ergeben sich die Tabellen 5.1, 5.2 und 5.3.

Tabelle 5.1: Aufführung der betrachteten Abstände zu der aufgenommenen Sitzbank

Abstand zur Sitzbank					
Abstand [m]	1	1.5	2	2.5	3

Tabelle 5.2: Vorstellung der verschiedenen Aufnahmewinkel

verfügbare Aufnahmewinkel								
Position []	1	2	3	4	5	6	7	8
Winkel [°]	0	45	90	135	180	225	270	315

Tabelle 5.3: Darstellung der unterschiedlichen Helligkeiten bei Aufnahme der Punktewolke

Helligkeitseinstellung der Standleuchten			
Beleuchtungsgrad [%]	0	50	100
Helligkeit [Lux/2m]	+0	+475	+950

In der ersten Tabelle 5.1 sind fünf Abstände zu erkennen, mit denen die Punktewolken aufgenommen wurden. Die Tabelle 5.2 enthält die acht für die Messungen verwendeten Winkel und deren zugehörige Positionsbenennung. In der dritten Tabelle 5.3 sind die drei Beleuchtungsstufen mit der zugehörigen Helligkeit aufgezeichnet.

Die Aufnahme-Winkel aus Tabelle 5.2 sind in der Abbildung 5.3 wiederzufinden. Die Abbildung zeigt die Sitzbank von oben und die verschiedenen Aufnahmepositionen sind mit Zahlen von 1 bis 8 beschrieben. Am unteren Ende des Bildes ist eine zwei Meter lange Markierung als Maßstab vorhanden. Die Messungen aus unterschiedlichen Richtungen und die Evaluierung dieser Messergebnisse ist auschlaggebend für den Späteren gebraucht, weil in einer Flugzeugkabine nicht ausreichen Platz ist um die Komponenten aus allen Winkeln zu betrachten. Zudem besteht die Limitierung, dass für das erfassen der Realgeometrien nicht die Möglichkeit besteht einzelteile auszubauen, weil das Zeitaufwendig ist und die Position der Komponent in relation zu der Kabine benötigt wird.

Aus den Punktewolken mit unterschiedlichen Faktoren werden für jede Helligkeitsstufe Kombinationen mit unterschiedlicher Anzahl und Kombination an Winkeln gebildet. Aus

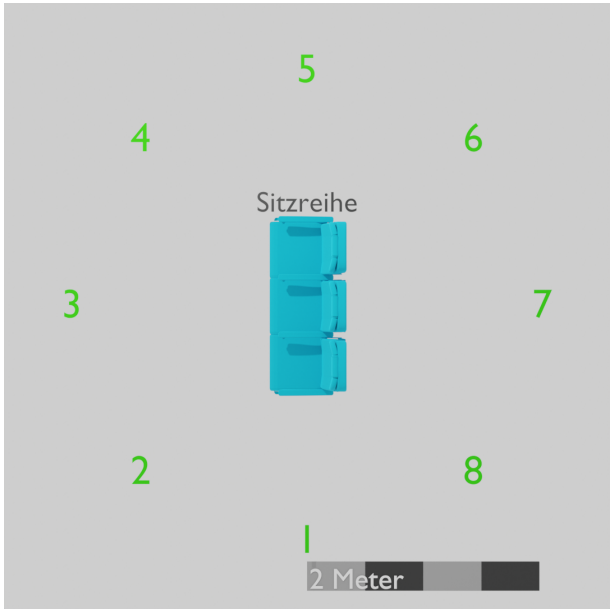


Abbildung 5.3: Aufbau des Versuches(Top down)

den 8 Winkeln ergeben sich 255 unterschiedliche Kombinationen. Mit den drei Helligkeitsstufen ergeben sich 765 unterschiedliche Punktewolken mit 30.200 bis ca. 2.500.000 Punkten. Sofern pro Punktewolke mit einer Berechnungszeit von 10 bis 15 Minuten ausgegangen werden kann, dauert die Berechnung der Werte von allen Punktewolken bereits mehr als 6 Tage. Daraus ergibt sich, dass die Auswahl an Zusammensetzung der Punktewolken eingeschränkt werden muss.

5.3 Extraktion der Punktewolke aus CAD-Daten

Wie zuvor erklärt, beinhalten CAD-Dateien unterschiedliche Informationen zu einem Objekt. Besonders wichtig zum Erkennen von Objekten ist hierbei die Geometrie. Deshalb wird im Folgenden erklärt, wie aus der CAD-Datei die gewünschten Daten extrahiert werden. Dazu wird das DLR-interne Python-Paket occhelper, welches auf Open CASCADE (OCC) beruht, benutzt.

OCC ist eine quelloffene Sorftwareentwicklungsplatform für 3D-Modellierung, Visualisierung und Simulationen. Diese hat den Vorteil, dass dem Nutzer bereits viele Werkzeuge zum Umgang mit unter anderem CAD-Daten bereitstehen. Besonders wichtig für die Extraktion der benötigten Punkte und Normalen ist das unterstützte STEP-Format. [36]

Zum Extrahieren der benötigten Daten aus dem STEP-Format wird zunächst die CAD-Datei in Python als Form geladen. Diese bietet keine direkte Möglichkeit die enthaltenen Daten zu extrahieren. Deshalb wird die Form in das The Visualisation Toolkit (VTK)-Format übertragen. Darüber besteht die Möglichkeit die Punkte und Normalen als numpy

Array zu extrahieren. Das bedeutet, dass alle Punkte und Normalen des dargestellten Objektes in einer Liste vorliegen, die an ein weiteres Programm weitergegeben werden kann. Zusätzlich liegen einige CAD-Daten im STL-Format bei. Diese werden direkt mit dem Python-Paket open3d [42] geladen, welches die Möglichkeit bietet, die benötigten Daten direkt zu extrahieren.

Abbildung 5.4: Sitzreihe als CAD-Modell(links) und Punktewolke(rechts)

In Abbildung 5.4 ist auf der linken Seite das CAD-Modell und auf der rechten Seite die extrahierte Punktewolke zu erkennen. Bei dem Model handelt es sich um eine Dreier-Sitzreihe aus einer Flugzeugkabine. An Stellen des Models, an denen mehr Details abgebildet werden, wie die Armlehnen, sind mehr Punkte in der Punktewolke zu erkennen. Zudem ist die Punktewolke ziemlich groß.

Deshalb werden zunächst Punkte auf der Oberfläche des CAD-Modells gebildet und diese anschließend in eine Punktewolke extrahiert. Dadurch entsteht eine gleichmäßige Verteilung der Punkte auf gesamten Oberfläche des Modells. Im Anschluss werden die Punkte mit dem Voxel-Grid-Filter reduziert. Dieser ermöglicht es die gesamte Anzahl an Punkten zu reduzieren und die vorhandenen Punkte gleichmäßig zu verteilen. Der Filter bildet ein Netz aus s großen Quadraten. Sofern diese Quadrate Punkte der Punktewolke enthalten, wird der Mittelpunkt in die Punktewolke eingefügt. Das ermöglicht es, ein gleichmäßiges Netz an Punkten zu bilden. [27]

5.4 Zusammensetzung der Punktewolken aus den Scan-Daten

Eine Punktewolke wurde mithilfe eines Scanners aus einer Position aufgenommen. Aus diesem einen Scan kann allerdings keine komplette Szene gebildet werden. Um eine

komplette Szene abzubilden, werden mehrere Scans aus unterschiedlichen Positionen benötigt [45][21]. Die Anzahl an benötigten Scans hängt sowohl von der gescannten Szene als auch dem Scanner ab. Um ein Objekt in einer Szene erkennen zu können, muss diese zunächst zusammengesetzt werden [20].

Mithilfe der Relationen der unterschiedlichen Positionen des Scanners können die verschiedenen Punktewolken registriert und zusammengesetzt werden [20]. Eine andere Möglichkeit besteht darin, die Relation mithilfe eines Orientierungspunktes in der Szene zu erfassen. Das erfordert durchgängige Sichtbarkeit des Orientierungspunktes in den Punktewolken. Der Orientierungspunkt kann allerdings nicht immer in der Szene platziert werden [45]. Die Transformation zur Registrierung kann ebenfalls mithilfe von Algorithmen, wie Iterative Closest Point (ICP)[20] [32] [29] [35], ermittelt werden.

Die Punktewolke wird immer in Abhängigkeit zur Position des Scanners aufgenommen. Für diese Arbeit wird die relative Position und Rotation des Scanners verwendet, um die Transformation zu ermöglichen. Die Punkte einer Wolke werden zur Registrierung mit der zugehörigen Position und Rotation transformiert. Zusätzlich kann ein Referenzpunkt in der Punktewolke markiert werden, der dazu genutzt wird, die Position der Objekte in der Messumgebung anzugeben. Dazu bietet es sich an, eine der Positionen des Scanners zu benutzen.

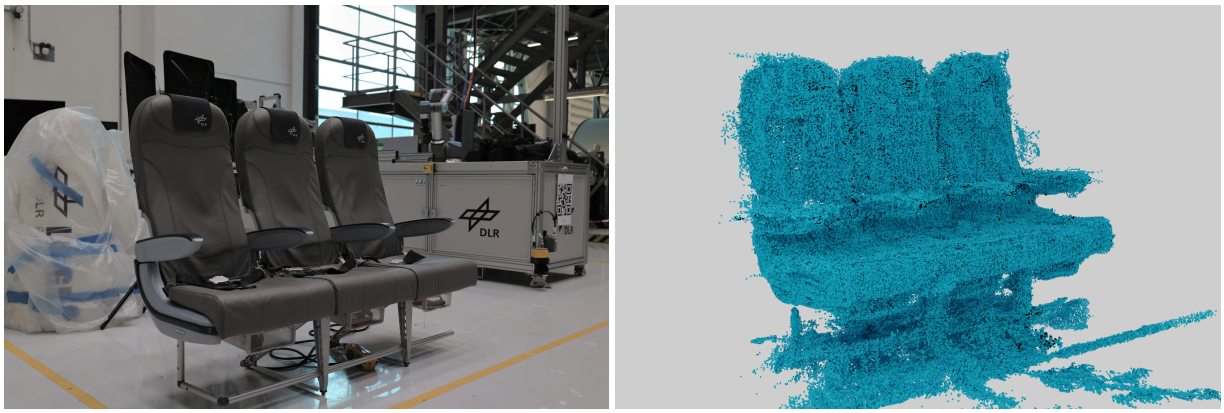


Abbildung 5.5: Aufgenommene Sitzreihe als Bild(links) und 3D-Scan(rechts)

Für diese Arbeit wurden mehrere Punktewolken aufgenommen. Diese werden benötigt, um die Qualität des Algorithmus auf unterschiedliche Faktoren zu untersuchen. Als Modell für diese Aufnahmen wurde eine Sitzreihe aus einer Flugzeugkabine gewählt. Zu der Sitzreihe ist das in Abbildung 5.4 abgebildete CAD-Modell vorhanden.

In Abbildung 5.5 ist auf der linken Seite ein Foto der physischen Sitzreihe und auf der rechten Seite einer der aufgenommenen Scans zu erkennen. Beide Bilder bilden eine identische Sitzreihe ab, weshalb hier beide Bilder zusammen beschrieben werden. In der Mitte des Bildes ist die Sitzbank mit drei Sitzen leicht nach rechts gedreht. Dadurch ist ein

Frontal- und Seitenprofil zu erahnen. Dabei sind die einzelnen Sitze durch eine Armlehne getrennt und die äußeren Sitze durch eine weitere abgegrenzt. In der Abbildung 5.5 ist bereits zu erkennen, dass das Rauschen in der Aufnahme größer ist als erwartet. Das kann dafür sorgen, dass es dem Algorithmus nicht gelingt die Punktewolke dem CAD-Modell zuzuordnen. Die abgebildete Punktewolke wurde bei Beleuchtung der Sitzbank und mit zwei Metern Abstand aufgenommen und mithilfe von Visualisierungs-Software visualisiert. Für dieses Bild wurde die Punktewolke zugeschnitten und ein Großteil der nicht benötigten Punkte entfernt.

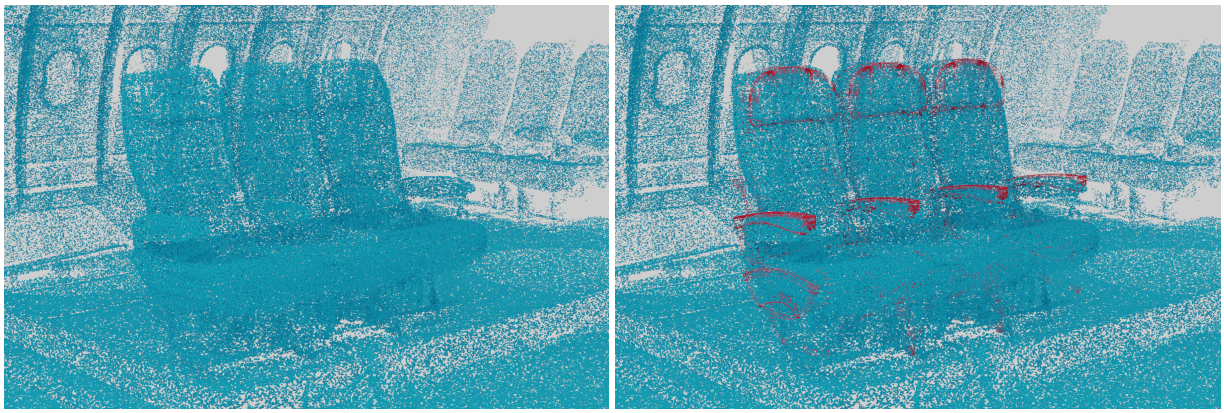
Die Algorithmen werden auf unterschiedliche Faktoren untersucht, die einzeln bei den Scans verändert wurden. Diese Faktoren bestehen aus der Belichtung des Modells, die Entfernung zum Objekt und dem Winkel der Aufnahme. Die Belichtung des Models verändert die Qualität der Punktewolke dadurch, dass mehr Punkte in dem beleuchteten Bereich erkannt werden. Das Verändern des Winkels von der Aufnahme ermöglicht es zu überprüfen, wie viel Prozent der Oberfläche des Objektes sichtbar sein muss, damit die Algorithmen das Objekt erkennen. Zudem wird ermittelt, welche Winkel ideal sind, um ein Objekt zu erkennen bzw. welche Auswirkung das Nutzen von unterschiedlichen Winkeln hat. Das wird damit erreicht, dass eine unterschiedliche Anzahl an Punktewolken zusammengesetzt werden. Die Entfernung zum Objekt verändert die Punktedichte der Punktewolke und kann das Sichtfeld einschränken. Aufgrund der Einflüsse dieser Faktoren wird die Qualität der resultierenden Punktewolke beeinflusst. Das kann zu Fehlern beim Erkennen von Objekten führen [28].

5.5 Extraktion des Objektes aus der Punktewolke

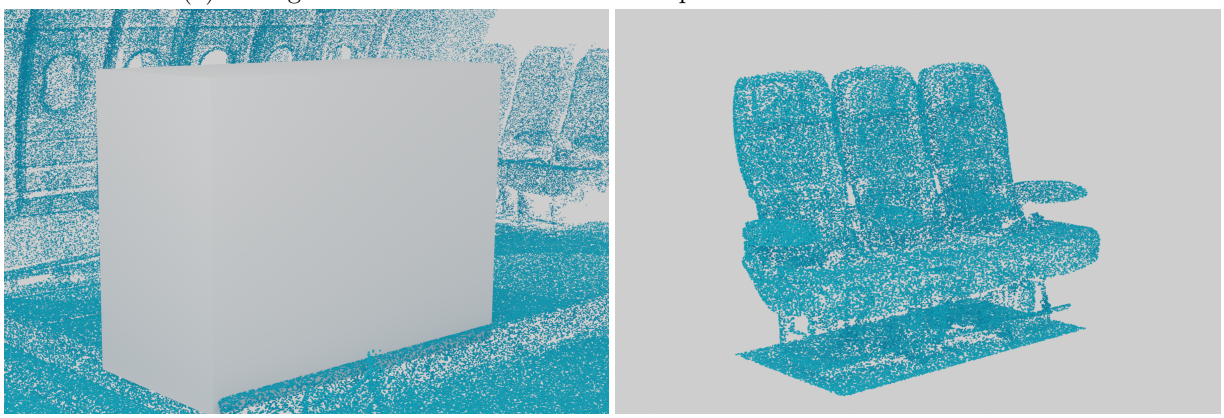
Wenn das CAD-Modell mit der Szene überlagert, stehen unterschiedliche Möglichkeiten zur Verfügung, um die zugehörigen Punkte aus der Szene zu extrahieren. Zunächst kann mithilfe eines Begrenzungsrahmens, eine das CAD-Modell umfassenden rechteckigen Box, ein Großteil der Szene entfernt und dadurch folgender Rechenaufwand reduziert werden. Durch die verbleibenden Punkte kann eine Schleife iterieren, die alle Punkte entfernt, die zu weit von dem CAD Modell entfernt sind.

Zudem stellt das open3d-Paket die Möglichkeit bereit, alle Punkte zu entfernen, die nicht eine bestimmte Anzahl an weiteren Punkten in einem bestimmten Radius hat [42]. Eine weitere Möglichkeit besteht darin, die kumulative Varianz der Punkte auszuwerten und Stellen mit einer geringeren Punktedichte zu entfernen.

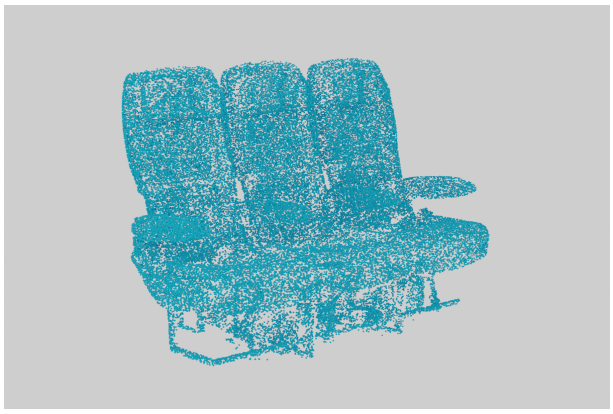
Für diese Arbeit wird das CAD-Modell mithilfe des Translations-Vektors und einer Rotations-Matrix, deren Berechnung in Kapitel 3.1 erläutert wird, mit der Szene über-



(a) Einfügen des CAD-Modells an der entsprechenden Stelle in der Szene



(b) Einfügen des Begrenzungsrahmens des CAD-Modells und entfernen der außen liegenden Punkte



(c) Darstellung der extrahierten Punktewolke

Abbildung 5.6: Besipiel einer Extraktion der Komponente aus der Szene

lappt. Dieser Teil wird in Abbildung 5.6a dargestellt. Im Anschluss wird mithilfe des CAD-Modells der Begrenzungsrahmen erstellt und, wie in Abbildung 5.6b zu erkennen, die außerhalb liegenden Punkte der Szene entfernt. Um nicht zu viele Punkte zu entfernen, wird der Rahmen um 110 % der Länge, Breite und Höhe des CAD-Modells gespannt. Die verbleibende Punktewolke wird mithilfe des open3d-Algorithmus [42] zum Entfernen der Punkte, die zu wenig Nachbarpunkte in einem bestimmten Radius haben, aufgeräumt. Dabei entsteht eine Punktewolke, die nur noch den Ist-Zustand des Objektes widerspiegelt. Diese wird in Abbildung 5.6c Beispielsweise abgebildet.

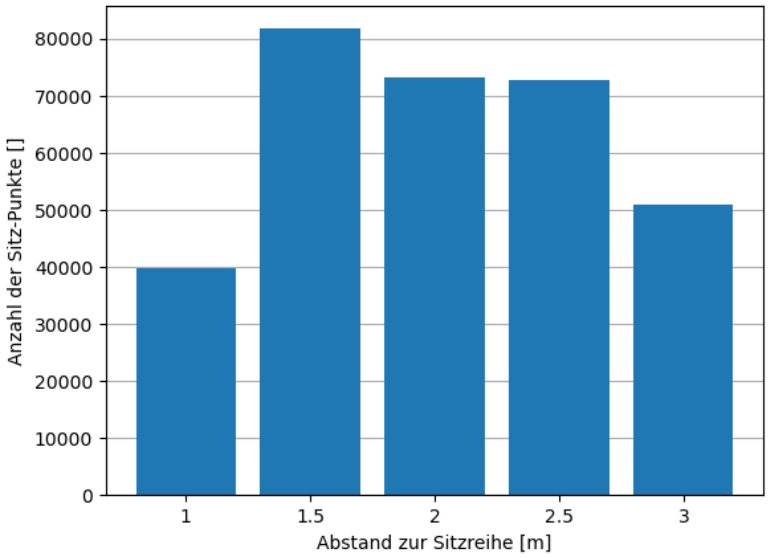
6 Ergebnisse des Versuchsaufbaus

Um die Aufnahme-Situation bewerten zu können, ist eine Betrachtung unterschiedlicher Einflüsse auf die Erkennungswahrscheinlichkeit nötig. Zunächst wird der Einfluss von Beleuchtung des Objektes und Abstand zu dem Objekt auf das Erkennen von diesem getestet. Dafür wird der jeweilige Abstand und Beleuchtungsgrad in Relation zu der Qualität der aufgenommenen Punktewolke betrachtet. Die Qualität der Punktewolke wird in diesem Fall an der Anzahl an Punkten, die dem erkannten Objekt zugeordnet wurden, gemessen. Für die Auswertung des Einflusses durch die Beleuchtung wird die Anzahl der Punkte in Relation zu der Gesamtanzahl an Punkten in der Punktewolke gesetzt.

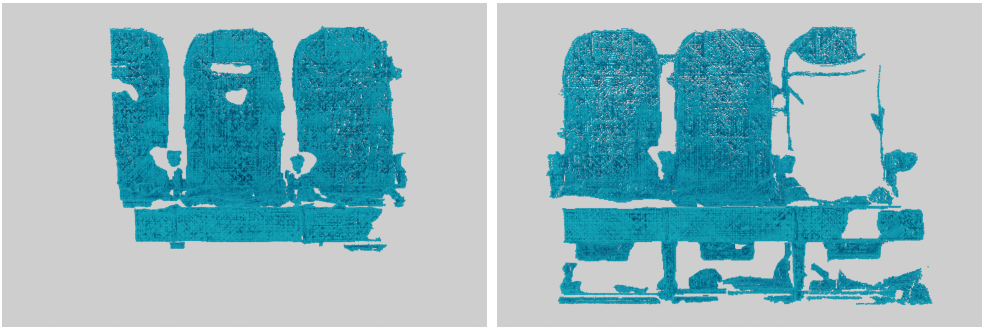
In Folge wird die ideale Anzahl und Position von Aufnahmen des Objektes betrachtet. Dafür werden die Beleuchtung des Objektes und der Abstand zu dem Objekt konstant gehalten. Diese beiden Werte beeinflussen den Algorithmus direkt in seiner Erkennungsrate. Aus diesem Grund wird die Änderung dieser Parameter auf die jeweilige Erkennungsrate des Algorithmus projiziert. Die Änderung in der Anzahl von Aufnahmen wird dementsprechend einfach auf die Erkennungsrate modelliert. Allerdings lassen sich die Positionen der Aufnahme nicht gleichermaßen vergleichen. Aus diesem Grund wird für diesen Wert der durchschnittliche Unterschied zwischen den Aufnahmewinkeln in *Grad* zu den verschiedenen Aufnahme-Positionen mit der jeweiligen Erfolgsrate verglichen. Die Modellierung des durchschnittlichen Unterschieds zwischen den Aufnahmewinkeln in Grad ist nur möglich, weil der Abstand zu dem Objekt für jede Aufnahme-Position gleich bleibt. Um die Auswertung der Einflüsse der Parameter zu vereinfachen, werden die verschiedenen Projektionen in Graphen modelliert.

6.1 Einfluss des Abstandes auf den Algorithmus

Die Abbildung 6.1 zeigt die Auswertung der erfassten Daten von Punktewolkenaufnahmen aus unterschiedlicher Entfernung zu der Sitzreihe in Form eines Graphen und einer Visualisierung zweier Punktewolken. Um den optimalen Abstand zu bestimmen, werden alle möglichen Abstände in 0,5 Meter Schritten betrachtet. Aufgrund der Größe des Objektes und der Limitierung der Kamera wurde der Abstand auf eins bis drei Meter beschränkt. Zur Auswertung wird die Anzahl an aufgenommenen Punkte auf der Sitzbank der jeweiligen Abstände betrachtet. Das Ziel ist den Abstand zum Objekt zu minimalisieren, ohne dass eine Reduzierung in Punkten erfolgt. Demnach wird nach der höchsten Anzahl an Punkten



(a) Graph zur Visualisierung der Sitzpunkte-Anteile



(b) Punktewolken der Abstände 1,5 und 2,0 Meter

Abbildung 6.1: Auswertung des Abstands zum Objekt

gesucht. Allerdings sollte die Sitzbank komplett in der Aufnahme vorhanden sein. Deshalb werden die Ergebnisse im Anschluss nochmal auf diese Eigenschaft abgeglichen.

In dem Graphen aus der Abbildung 6.1 ist zunächst abzulesen, dass die meisten Punkte bei 1,5 und 2,0 Metern Abstand aufgenommen wurden. Um diese Ergebnisse auszuwerten, muss zunächst überprüft werden, ob die gewählte Darstellungsform ausreichend ist um einen Idealen Abstand zu bestimmen. In der Grafik ist zunächst ein Anstieg in der Anzahl an Punkten zu erkennen. Nachdem ein Höhepunkt erreicht wurde, fällt dieser allerdings wieder. Zudem ist in der Abbildung ein Ausreißer zu erkennen, bei dem die Punktewolke überprüft wird. Dabei fällt auf, dass in der Punktewolke mit 1,5 Metern Abstand die Befestigung der Sitzreihe nicht vorhanden ist. Dieser Ausreißer wurde aufgrund der fehlenden Abbildung von wichtigen Details der Sitzbank ausgeschlossen. Dieser Fehler kann mithilfe von mehreren Aufnahmen aus unterschiedlichen Position behoben werden, indem die fehlenden Teile separat aufgenommen werden. In dieser Arbeit wird jedoch nach der möglichst niedrigen Anzahl an Aufnahmen gesucht, weshalb dieses Vorgehen suboptimal ist. Deshalb wurde

dieser Ausreißer als negativ bzw. nicht idealer Wert festgehalten. Im Anschluss wird der Bereich identifiziert, in dem möglichst ein großer Anteil an Punkten aufgenommen wird. In der anderen Punktewolke, die mit dem Abstand von zwei Metern aufgenommen wurde, sind die zuvor fehlenden Details wiederzufinden. Dies kann nicht in den Werten der Abbildung 6.1 erkannt werden, weil die Befestigung eine geringe Oberfläche hat und bei 1,5 Metern Abstand mehr Punkte der Sitzoberfläche aufgenommen werden. Demnach liegt der optimale Abstand zwischen 1,5 und 2,0 Metern. Deshalb wird im Fortlauf dieser Arbeit der Abstand von 2,0 Metern benutzt.

6.2 Einfluss der Beleuchtung auf den Algorithmus

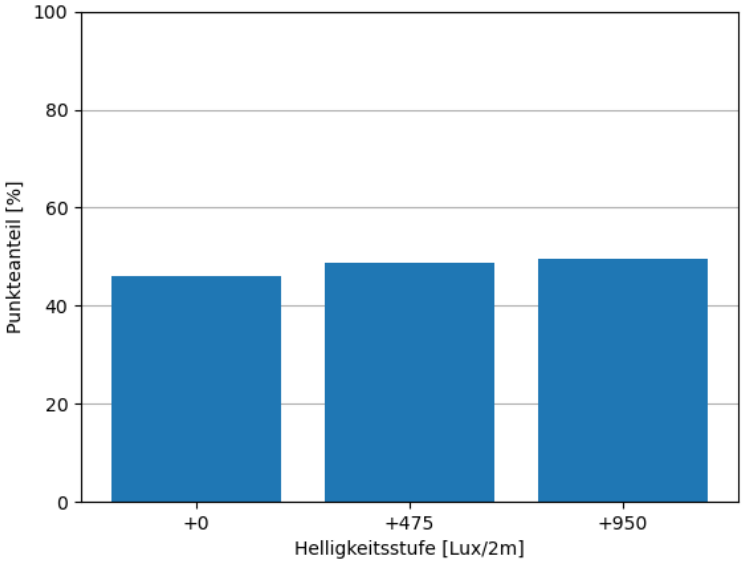


Abbildung 6.2: Visualisierung der aufgenommenen Daten zur Auswertung der Beleuchtungsstufen

In Abbildung 6.2 ist eine Visualisierung zur Auswertung der Aufnahmen mit verschiedenen Beleuchtungsstufen zu erkennen. Auf der y-Achse ist die Anzahl an Punkten, die zu den Sitzen in Relation zu der gesamten Punktewolke gehören abgebildet, welche nach der Gleichung 6.1 berechnet wird.

$$Punkteanteil = \frac{\sum Punkte_{Sitz}}{\sum Punkte_{Szene}} \tag{6.1}$$

Die x-Achse zeigt die verschiedenen Helligkeitsstufen.

In der Abbildung 6.2 ist zu erkennen, dass bei keiner Beleuchtung nur ca. 46 % der Punkte zu den Sitzen gehören. Mit Beleuchtung hingegen steigt der Anteil auf ca. 49–50 % bei 50–100 % Beleuchtung. Diese 3–4 % beziehen sich auf ca. 2,6 Millionen Punkte, weshalb

selbst diese vergleichsweise geringe prozentuale Änderung ca. 78–104 Tausend Punkte Unterschied, darstellt.

6.3 Einfluss der Anzahl an Aufnahmen auf den Algorithmus

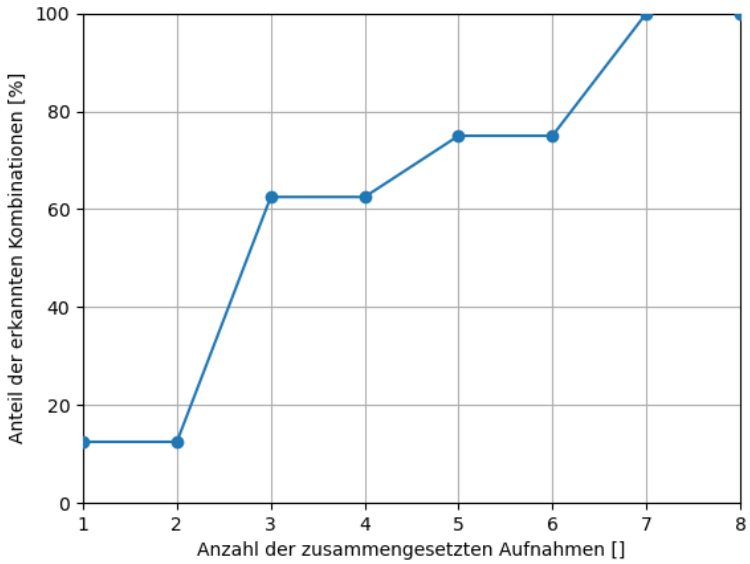


Abbildung 6.3: Fehlerhafte Auswertung des Einflusses der Anzahl an Scans auf die Erkennungsrate

Die Abbildung 6.3 ist durchschnittliche Erkennungsrate von den unterschiedlichen Kombinationen aus den vorher aufgenommen Punktewolken gegenüber der in diesen Kombinationen enthaltenen Anzahl an Punktewolken aufgetragen. Mithilfe der Abbildung ist zu erkennen, dass bei einer höheren Anzahl an Punkten eine höhere Erkennungsrate vorliegt. Um diese Daten zu verifizieren und fehlerhafte Ergebnisse auszuschließen wurden die aus der jeweiligen Szene, die durch die Kombination an Punktewolken gebildet wurde, extrahierte Punktewolke, die zu dem erkannten Objekt gehört betrachtet. Das ist in Abbildung 6.4 zu erkennen. Beim genauen Betrachten dieser Punktewolke fällt auf, dass nicht die gesamte Sitzreihe abgebildet ist. Stattdessen ist nur ein Teil der Sitzreihe (blau) zu erkennen, während der andere Teil nicht in der Punktewolke vorhanden ist. In der Abbildung ist lediglich eine der extrahierten Punktewolken zu erkennen. Die anderen Punktewolken weisen allerdings ähnliche Defizite auf, sodass kein erfolgreich erkanntes Objekt wiederzufinden ist. Demnach sind alle in Abbildung 6.3 enthaltenen Werte fehlerhaft und die optimale Anzahl an Scans kann durch diese Abbildung nicht bestimmt werden. In der Abbildung 6.4 ist zusätzlich das CAD-Modell (rot) zu erkennen, welches zeigt, an welcher Stelle ein Teil des Modells fehlt.

Das Fehlerhafte erkennen liegt beim genaueren Betrachten der möglichen Ursachen an dem Rauschen in dem Scan der Szene. Dazu ist es möglich sich die Abbildung 4.2 ins Gedächtnis zu rufen. Beim Betrachten des Spin-Images ist ein deutlicher Rausch Anteil in der detaillierteren Abbildung 4.2b wiederzufinden. Beim Messen dieses Rauschens in der Punktewolke ergeben sich Amplituden über $1cm$.

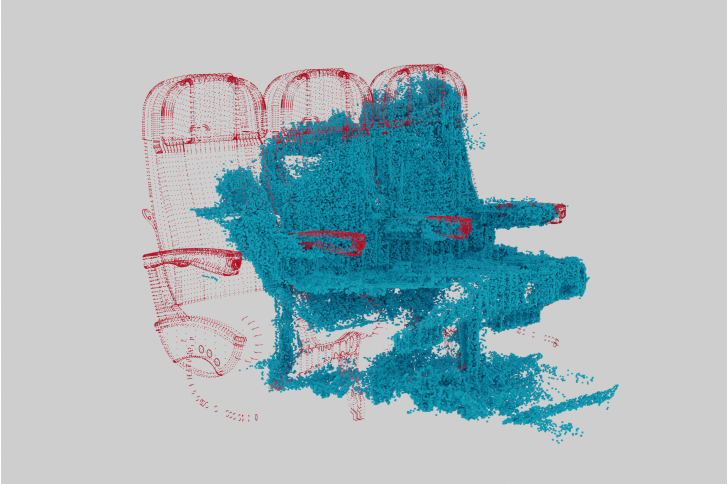


Abbildung 6.4: Darstellung der fehlerhaft ausgeschnittenen Sitzreihe

An dieser Stelle wurde die Auswertung zum Erfassen eines optimalen Aufnahmewinkels für die optimale Anzahl an Aufnahmen nicht in die Arbeit aufgenommen. Zum einen zeigen die vorherigen Versuche bereits eine fehlerhafte Erkennung der Objekte in der Szene beim Verwenden des Roboception Scanners. Andererseits lassen sich zusätzliche Auswertungen gegenüber des Spin-Image-Algorithmus durchführen. Der Spin-Image-Algorithmus bietet, wie bereits in Kapitel 4 aufgeführt, die Möglichkeit nur bestimmte Ausrichtungen der Punkte in das Spin-Image aufzunehmen. Diese Anpassung sollte in Referenz zu dem Aufnahmewinkel gemessen und die beiden Abhängigen Faktoren gemeinsam ausgewertet werden.

In dieser Versuchsreihe sind ebenfalls andere Fehler aufgetreten. Zum Beispiel ist relativ häufig eine Wand anstatt des Sitzes erkannt worden. Um das zu verstehen, ergibt es Sinn, nochmal die Abbildung 4.2 anzusehen. Dort fällt auf, dass manchmal eine gerade Linie auf dem Spin-Image in Richtung seiner x-Achse geformt wird. Bei der Überlegung, wie eine Wand durch ein Spin-Image dargestellt wird, fällt auf, dass diese ebenfalls durch eine dem Normalenvektor orthogonale Linie getan wird.

7 Auswertung der Analyse des Algorithmus und Ergebnisse des Versuchsaufbaus

In den folgenden Kapiteln werden die zuvor vorgestellten Versuchsdurchführungen und deren Ergebnisse ausgewertet und in Bezug auf die Motivation diskutiert. Dafür wird zunächst auf die Algorithmus-unspezifischen Lösungen, wie das Extrahieren der Punktewolken, eingegangen.

7.1 3D-Data Extraktion

Damit der Algorithmus zur Objekterkennung überhaupt funktioniert, braucht dieser Daten zum Auswerten. Darunter 3D-Daten von Objekten, die Mithilfe des Algorithmus erkannt werden sollen. Dafür wurden CAD-Modelle als Referenz verwendet, um ein möglichst kleines, aber akkurates Modell zur Verfügung zu haben. Wie in Kapitel 5.3 vorgestellt, wird aus dem CAD-Modell eine Punktewolke extrahiert. Dabei sind unterschiedliche Merkmale aufgefallen. Das CAD-Modell beinhaltet aufgrund seiner möglichst einfachen Form sehr wenige Punkte an Stellen, die wenige Details aufweisen. Im Gegensatz dazu sind an Stellen mit vielen Details eine Vielzahl an Punkten vorhanden. Ein solches Ungleichgewicht der Verteilung der Punkte in der Punktewolke tritt in den Scans nicht auf. Deshalb ist es wichtig, die aus dem CAD-Modell extrahierten Punkte anzupassen. Dafür wurde in dieser Arbeit das gesamte Oberflächennetz in Punkte umgewandelt und im Nachhinein gefiltert. Teile dieser Filter wurden ebenfalls bei dem Scan angewendet, um die Punktedichte zu verringern. Durch das Verwenden von CAx kompatiblen Dateiformaten ist eine direkte Integration in deren Prozesse möglich. Dadurch können neu erstellte Modelle direkt nach dem Scannen für den Algorithmus verwendet werden, ohne das manuelle Anpassungen an dem Modell vorgenommen werden.

Für die Scan-Punktewolke ist es allerdings wichtig, dass nicht zu viele Punkte entfernt werden. Diese wird im Nachhinein zur weiteren Verarbeitung bereitgestellt und beinhaltet aus diesem Grund alle erfassten Punkte, die der Komponente zugeordnet werden. Für den Algorithmus hingegen sind die Details nicht wichtig und die Punktewolke kann vereinfacht werden.

Kapitel 5.2 erläutert, dass für die Versuchsdurchführung eine große Menge an auszuwertenden Punktewolken zusammengesetzt wird. Die Anzahl dieser Punktewolken ist zu groß und es entsteht ein zeitlicher Aufwand, um diese zu analysieren, der nicht im Umfang dieser Bachelorarbeit aufgebracht werden kann. Deshalb wurde die Datenmenge mithilfe der Auswertung der Ergebnisse angepasst. Es wurde bereits bei der Aufnahme der Punktewolke auf die Qualität dieser geachtet. Zudem wurde der ideale Abstand und die ideale Beleuchtung erfasst und diese für die weiteren Messungen konstant gehalten. Eine weitere Möglichkeit, die Anzahl an möglichen Punktewolken zu reduzieren, liegt darin, die Kombination der Punktewolken zu reduzieren. Aus diesem Grund werden für jede Anzahl an zusammengesetzten Punktewolken lediglich 8 beispielhafte Zusammensetzungen betrachtet. Dadurch, dass Aufnahmen aus 8 unterschiedlichen Winkeln getätigt wurden, sind die Kombinationen mit einer Punktewolke auf 8 Kombinationen beschränkt. Für die höchste Anzahl an Kombinationen ist nur eine Möglichkeit vorhanden. Allerdings ist dieser Wert irrelevant, weil eine möglichst geringe Anzahl an Kombinationen gesucht wird. Von den anderen möglichen Kombinationen werden ebenfalls pro Anzahl zufällig 8 Stück ausgewählt, um jeweils die gleiche Anzahl an ausgewerteten Kombinationen auszuwerten. Beim Betrachten von Abbildungen der Sitzreihe wie Abbildung 5.3 fällt auf, dass der Aufbau der Sitzreihe gespiegelt ist. Dadurch sind zum Beispiel die Zusammensetzung der Position aus Kapitel 5.2, 1 und 2 gespiegelt zu der 4 und 5. Dadurch ist es nötig, die zufällig ausgewählten Kombinationen ohne Doppelungen zu wählen. Die durch die Aufnahme veränderten Faktoren werden im folgenden Kapitel weiter erläutert.

7.2 Objekterkennung

Das Ziel dieser Arbeit ist es, die Anforderungen an die Scan- und Scanner-Qualität zu untersuchen und dem Spin-Image-Algorithmus zuzuordnen. Dafür wurde der in Kapitel 5.1 beschriebene Versuchsaufbau gewählt und durchgeführt. Zudem wurde der Algorithmus auf die in Kapitel 4 genannten Parametern wie die Bildgröße und das Größenverhältnis untersucht. Die Funktionalität des Algorithmus wurden infolge mit den in Kapitel 4 untersuchten Parametern validiert. Um den Spin-Image-Algorithmus zu validieren, wurde der detaillierte Scan eines Flugzeuges mit Leica Hochleistungsscanner genutzt, um dieses mithilfe eines CAD-Modells zu identifizieren.

Für die Anforderungen an den Algorithmus wurden dabei festgestellt, dass kleinere Werte für die Bildgröße *width* und das Verhältnis *scale* für eine höhere Korrelation sorgen. Die Ergebnisse aus Kapitel 6 zeigen, dass die gewählten Werte in der Szene für Fehler sorgen. Zum Beispiel findet der Algorithmus die Sitzbank in der Wand anstatt seiner eigentlichen Stelle. Dieser Fehler kann bisher noch nicht automatisch erkannt werden. Das sorgt dafür,

dass entweder fehlerhafte Daten automatisch weitergegeben werden oder ein manueller Zwischenschritt zum Herausfiltern dieser Fehler hinzugefügt werden muss. Dieser manuelle Zwischenschritt liegt nicht im Sinn der vollständigen Automatisierung des Scannens und Erkennens von Objekten. Das Kapitel 6 zeigt zusätzlich, dass die Korrelationswerte für die aufgenommene Sitzbank im Vergleich zu denen von dem Hochleistungsscanner deutlich niedriger sind. Beim Betrachten der mit der Roboception aufgenommen Punktewolke ist bereits aufgefallen, das sehr viel Rauschen vorliegt. Die Ergebnisse aus Kapitel 6.3 zeigen zudem, dass dieses Rauschen höher als der vom Hersteller vorgegebene Wert von der Roboception ist. Zusammen mit den Ergebnissen aus Kapitel 4.1, die besagen, dass ein zu großes Rauschen zu deutlichen Erkennungsproblemen führt, ergibt sich ein deutliches Problem mit dem Rauschen der Roboception.

In Kapitel 6.1 wurde die Punktewolken Qualität in Relation zu dem Abstand zwischen dem Objekt und dem 3D-Scanner aufgeführt. Dabei wurde der ideale Abstand zu der Komponente bei 2 Metern festgestellt. In der Flugzeugkabine ist, wie bereits erwähnt, nicht sehr viel Platz, um Scans aus verschiedenen Winkeln aufzunehmen. Das lässt sich ebenfalls auf den Abstand zu den Komponenten in der Kabine beziehen. Dort ist ebenfalls nicht ausreichend Platz, um einen Abstand von 2 Metern einzuhalten. Deshalb wird das Objekt mit mehr Aufnahmen erfasst. Das führt allerdings sowohl zu einem größeren Zeitaufwand für das gesamte Erfassen einer Komponente als auch zu einer größeren Fehlerfortpflanzung von ungenauen Beschreibungen der Messstelle. Das kann dafür sorgen, dass die automatische Registrierung der Punktewolke Ungenauigkeiten aufweist. Diese Ungenauigkeiten können ebenfalls als eine Art von Rauschen betrachtet werden, welches bereits in geringen Ausmaßen zu starken Erkennungsproblemen führt. Diese Erkennungsprobleme sorgen erneut für eine benötigte Erkennungsmethode, die Fehler herausfiltern kann.

Das folgende Kapitel 6.2 erläutert die Beobachtungen beim Verstellen der Helligkeitsstufen für die Aufnahme. Diese Ergebnisse zeigen bereits, dass Beleuchtung zum Aufnehmen der Punktewolke benötigt wird. Allerdings können noch weitere zusätzliche Informationen aus dieser Beobachtung gezogen werden. Die Ergebnisse aus Kapitel 6.2 zeigen, dass bei unterschieden in der Helligkeit ein deutlicher Unterschied in der Anzahl an Punkten vorliegt. Die Helligkeit sollte demnach nicht nur bei 400 Lumen und mehr liegen, sondern auch gleichmäßig über das Objekt verteil sein, um eine gleichmäßige Verteilung der Punkte in der Punktewolke zu ermöglichen. Das gleichmäßige Beleuchten der Szene ist durch zum Beispiel eine an der Kamera befestigte Beleuchtung möglich. Diese kann in derselben Ausrichtung wie die Kamera befestigt werden. Allerdings sollte hierbei darauf geachtet werden, dass die Beleuchtung nicht die Bewegung der Kamera und somit die möglichen Aufnahmepositionen einschränkt. Andernfalls besteht die Möglichkeit, Lichtquellen in der Scan-Umgebung aufzustellen. Das führt allerdings ebenfalls zu Einschränkungen von Aufnahmewinkeln, sofern Schatten geworfen werden. Die Beleuchtung ist lediglich bei der

hier verwendeten Art von Scannern wichtig. Laser- und andere Scanner benötigen nicht unbedingt diese Art von Beleuchtung aufgrund ihrer unterschiedlichen Funktionsweisen.

Ein weiterer besprochener Faktor ist der Einfluss der Anzahl an Aufnahmen. Diese steht im direkten Zusammenhang mit dem benötigten Zeitaufwand für die Aufnahmen. Da dieser so gering wie möglich gehalten werden soll, ist es von Vorteil, eine geringe Anzahl an Aufnahmen zu benötigten. Die in Kapitel 6.3 dargestellten Ergebnisse, die zu einer Analyse der Mindestanzahl an Aufnahmen genutzt werden sollten, sind bei genauerer Betrachtung allerdings fehlerhaft. Aus diesem Grund kann keine optimale Anzahl festgestellt werden. Allerdings ist davon auszugehen, dass eine höhere Anzahl immer zu besseren Werten führt, weil durch die hohe Anzahl an Aufnahmen die Punkte gleichmäßig über die gesamte Oberfläche des Objektes verteilt sind. In Kapitel 4.1 wird zudem beschrieben, wie eine Anpassung des Winkels für die in das Spin-Image aufgenommenen Punkte diese Anzahl von benötigten Scans verringern können bzw. die Erfolgsrate für geringere Anzahlen erhöht. Die fehlerhaften Werte sind in diesem Fall auf das hohe Rauschen in der Aufnahme mit der Roboception Kamera zurückzuführen. Dafür wurden teilweise regelmäßige Abweichungen von über $1cm$ festgestellt, wodurch ein eindeutiger Fehler in der Erkennungsrate auftreten.

An den hier dargestellten Auswertungen ist zu erkennen, dass das Hauptproblem momentan an dem Rauschen in der Szene und dem Erstellen von akkuraten Parametern liegt, die nicht nur für eine gute Erkennungsrate, sondern zusätzlich für das Verringern von falschen Erkennungen sorgen. Die Parameter, die den Scanner betreffen, wie die Entfernung zum Objekt und die benötigte Belichtungen geben klare Vorgaben für deren optimale Werte. Dabei sollte die Entfernung groß genug sein, um das Objekt vollständig aufzunehmen und somit die Anzahl an insgesamt benötigten Punktewolken zu verringern. Zudem sollte eine gleichmäßige Beleuchtung vorliegen. Dadurch wird eine gleichmäßige Punktewolke ermöglicht. Die Auswertungen zeigen zudem, dass um nützliche Versuche durchzuführen, die die fehlenden Parameter eingrenzen können, zunächst das Rauschen aus der Szene entfernt werden muss. Das kann durch einen besseren Scanner oder durch das Vorbereiten der Punktewolke auf den Algorithmus mit Filtern erreicht werden. Beide Möglichkeiten bieten sich an, allerdings ist in Bezug auf eine kostengünstige Option das Verwenden von Filtern deutlich günstiger. Der Spin-Image-Algorithmus zeigt in dieser Ausführung der Automatisierung mithilfe der Roboception keine Vorteile gegenüber des manuellen erstellen der 3D-Objekte. Allerdings zeigt die Validierung, dass beim Eliminieren der hier vorgestellten Probleme die automatische Objekterkennung mit dem Spin-Image-Algorithmus möglich ist.

8 Zusammenfassung und Ausblick

In dieser Arbeit wurde der Spin-Image-Algorithmus auf mögliche Anwendungen zur automatischen Objekterkennung in der Flugzeugkabine untersucht. Dazu wurden benötigte Parameter an den Algorithmus und das Aufnehmen von Punktewolken identifiziert und ausgewertet. Nachdem der Algorithmus mit einem Hochleistungsscanner validiert wurde, wurde die Verwendung eines kostengünstigen, auf einem Roboter montierbarem Scanners untersucht. Dabei sind ideale Werte für den Abstand und die Beleuchtung der Kabine für die jeweiligen Scans erfasst worden. Allerdings sind bei der Messung unerwartet hohe Rauschanteile in der Punktewolke aufgetreten. Bei der Analyse des Algorithmus hat sich ergeben, dass bereits kleine Rauschanteile zum Scheitern der mit dem Spin-Image-Algorithmus erarbeiteten Objekterkennung führen. Aufgrund dessen konnten Parameter wie die Anzahl an benötigten Punktewolken und der benötige durchschnittliche Abstand in Grad zwischen den unterschiedlichen Aufnahmen nicht ausreichend erarbeitet werden. Es wurde festgestellt, dass der Spin-Image-Algorithmus alleine nicht ausreichend ist, um mit der Roboception, dem hier gewählten kostengünstigen Scanner, eine automatische Objekterkennung zu ermöglichen. Dafür treten zu viele Fehler aufgrund des Rauschens auf und es besteht keine Möglichkeiten, diese automatisch herauszufiltern. Zudem wurde festgestellt, dass dieser Algorithmus nur Objekte mit einem vorhandenen 3D-Modell erkennen kann. Aus diesem Grund wird festgehalten, dass der Roboception-Scanner keine ausreichende Qualität für die Ergebnisse liefert, die der Spin-Image-Algorithmus Auswerten kann, um die in der Motivation definierte Lücke der automatischen Objekterkennung und 3D-Modell Erstellung zu schließen.

Es bestehen allerdings Möglichkeiten, an diese Arbeit anzuknüpfen. Zum einen mithilfe von Filtern den Rauschanteil der Punktewolke zu entfernen und somit bessere Eingaben für den Algorithmus zu generieren. Andernfalls kann ein anderer Scanner getestet werden, der weniger anfällig gegenüber Rauschens ist. Zudem besteht die Möglichkeit, mithilfe von anderen Algorithmen, wie zum Beispiel der KI die hier definierten Probleme zu lösen oder zu umgehen. Zusätzlich müssen Möglichkeiten gefunden werden, neue Objekte zu erkennen und herauszufiltern. Des Weiteren hat diese Ausarbeitung nicht die Lücke des Erstellens von neuen CAD-Modellen geschlossen. Sobald ein Objekt identifiziert wurde, werden automatische Weiterverarbeitungsmöglichkeiten der extrahierten Punktewolke benötigt.

Literatur

[1] X. W. Xu * u. a. „STEP-compliant NC research: the search for intelligent CAD/-CAPP/CAM/CNC integration“. In: *International Journal of Production Research* 43.17 (2005), S. 3703–3743. DOI: 10.1080/00207540500137530. eprint: https://doi.org/10.1080/00207540500137530. URL: https://doi.org/10.1080/00207540500137530.

[2] AIRBUS. *Production - Building aircraft on time and top quality*. URL: https://www.airbus.com/en/products-services/commercial-aircraft/the-life-cycle-of-an-aircraft/production (besucht am 21. 08. 2023).

[3] Bogdan Apostol, Constantina Raluca Mihalache und Vasile Manta. „Using spin images for hand gesture recognition in 3D point clouds“. In: *2014 18th International Conference on System Theory, Control and Computing (ICSTCC)*. 2014, S. 544–549. DOI: 10.1109/ICSTCC.2014.6982473.

[4] Monika Bansal, Munish Kumar und Manish Kumar. „2D Object Recognition Techniques: State-of-the-Art Work“. In: *Archives of Computational Methods in Engineering* 28.3 (2021), S. 1147–1161. ISSN: 1886-1784. DOI: 10.1007/s11831-020-09409-1. URL: https://doi.org/10.1007/s11831-020-09409-1.

[5] Francois Chollet. *Deep learning mit python und keras: das praxis-handbuch vom entwickler der keras-bibliothek*. MITP-Verlags GmbH & Co. KG, 2018.

[6] McKinsey & Company. „Testing and validation: From hardware focus to full virtualization?“ In: (2017). URL: https://www.mckinsey.com/capabilities/operations/our-insights/testing-and-validation-from-hardware-focus-to-full-virtualization#/ (besucht am 21. 08. 2023).

[7] Konstantinos G Derpanis. „Overview of the RANSAC Algorithm“. In: *Image Rochester NY* 4.1 (2010), S. 2–3.

[8] Multi-Object Detection u. a. „Part and appearance sharing: Recursive Compositional Models for multi-view“. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, S. 1919–1926. DOI: 10.1109/CVPR.2010.5539865.

[9] H.Q. Dinh und S. Kropac. „Multi-Resolution Spin-Images“. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Bd. 1. 2006, S. 863–870. DOI: 10.1109/CVPR.2006.197.

[10] DLR. *Programm und Strategie*. URL: https://www.dlr.de/de/forschung-und-transfer/luftfahrt/programm-und-strategie (besucht am 14. 08. 2023).

[11] Martin A. Fischler und Robert C. Bolles. „Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". In: *Commun. ACM* 24.6 (Juni 1981), S. 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692. URL: https://doi.org/10.1145/358669.358692.

[12] Andrea Frome u. a. „Recognizing Objects in Range Data Using Regional Point Descriptors". In: *Computer Vision - ECCV 2004*. Hrsg. von Tomáš Pajdla und Jiří Matas. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, S. 224–237. ISBN: 978-3-540-24672-5.

[13] Kexue Fu u. a. „Robust Point Cloud Registration Framework Based on Deep Graph Matching". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Juni 2021, S. 8893–8902.

[14] Mara Fuchs u. a. „Virtual reconfiguration and assessment of aircraft cabins using model-based systems engineering". In: *33rd Congress of the International Council of the Aeronautical Sciences, ICAS 2022*. 2022. URL: https://elib.dlr.de/188336/.

[15] Ross Girshick u. a. „Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, S. 580–587. DOI: 10.1109/CVPR.2014.81.

[16] Roboception GmbH. *Roboception rc_visard Produktseite*. URL: https://roboception.com/de/rc_visard/ (besucht am 13. 07. 2023).

[17] Yujie Guo u. a. „Variationally consistent isogeometric analysis of trimmed thin shells at finite deformations, based on the STEP exchange format". In: *Computer Methods in Applied Mechanics and Engineering* 336 (2018), S. 39–79. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2018.02.027. URL: https://www.sciencedirect.com/science/article/pii/S0045782518301075.

[18] Michael Haag und Hans-Hellmut Nagel. „Combination of Edge Element and Optical Flow Estimates for 3D-Model-Based Vehicle Tracking in Traffic Image Sequences". In: *International Journal of Computer Vision* 35.3 (Dez. 1999), S. 295–319. ISSN: 1573-1405. DOI: 10.1023/A:1008112528134. URL: https://doi.org/10.1023/A:1008112528134.

[19] S. Haufe u. a. „Digital Twins Storage and Application Service Hub (Twinstash)". In: *Deutscher Luft- und Raumfahrtkongress (DLRK)*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V, 2023. DOI: 10.25967/570104.

[20] Dirk Holz u. a. „Registration with the Point Cloud Library: A Modular Framework for Aligning in 3-D". In: *IEEE Robotics & Automation Magazine* 22.4 (2015), S. 110–124. DOI: 10.1109/MRA.2015.2432331.

[21] D. Huber u. a. „Parts-based 3D object classification“. In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* Bd. 2. 2004, S. II–II. DOI: 10.1109/CVPR.2004.1315148.

[22] Anton Jezernik und Gorazd Hren. „A solution to integrate computer-aided design (CAD) and virtual reality (VR) databases in design and manufacturing processes“. In: *The International Journal of Advanced Manufacturing Technology* 22.11 (Dez. 2003), S. 768–774. ISSN: 1433-3015. DOI: 10.1007/s00170-003-1604-3. URL: https://doi.org/10.1007/s00170-003-1604-3.

[23] A.E. Johnson und M. Hebert. „Using spin images for efficient object recognition in cluttered 3D scenes“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.5 (1999), S. 433–449. DOI: 10.1109/34.765655.

[24] Andrew E Johnson. „Spin-images: a representation for 3-D surface matching“. Diss. The Robotics Institute, Carnegie Mellon Univ., 1997.

[25] Md Tarique Hasan Khan und Saki Rezwana. „A review of CAD to CAE integration with a hierarchical data format (HDF)-based solution“. In: *Journal of King Saud University - Engineering Sciences* 33.4 (2021), S. 248–258. ISSN: 1018-3639. DOI: https://doi.org/10.1016/j.jksues.2020.04.009. URL: https://www.sciencedirect.com/science/article/pii/S1018363920302282.

[26] K. Koster und M. Spann. „MIR: an approach to robust clustering-application to range image segmentation“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.5 (2000), S. 430–444. DOI: 10.1109/34.857001.

[27] Jason Ligon u. a. „3D point cloud processing using spin images for object detection“. In: *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC).* 2018, S. 731–736. DOI: 10.1109/CCWC.2018.8301688.

[28] Zhe Liu u. a. „EPNet++: Cascade Bi-Directional Fusion for Multi-Modal 3D Object Detection“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.7 (2023), S. 8324–8341. DOI: 10.1109/TPAMI.2022.3228806.

[29] Hao Men, Biruk Gebre und Kishore Pochiraju. „Color point cloud registration with 4D ICP algorithm“. In: *2011 IEEE International Conference on Robotics and Automation.* 2011, S. 1511–1516. DOI: 10.1109/ICRA.2011.5980407.

[30] E. Mollick. „Establishing Moore's Law“. In: *IEEE Annals of the History of Computing* 28.3 (2006), S. 62–75. DOI: 10.1109/MAHC.2006.45.

[31] Shi Pu, George Vosselman u. a. „Automatic extraction of building features from terrestrial laser scanning“. In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 36.5 (2006), S. 25–27.

[32] Tahir Rabbani u. a. „An integrated approach for modelling and global registration of point clouds“. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 61.6 (2007), S. 355–370. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2006.09.006. URL: https://www.sciencedirect.com/science/article/pii/S0924271606001213.

[33] F. Rauscher u. a. „Permanent aktualisierte 3D-Realgeometrie des ISTAR im Digitalen Zwilling“. In: *Deutscher Luft- und Raumfahrtkongress (DLRK)*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V, 2023. DOI: 10.25967/570374.

[34] Daniel Raymer. *Aircraft design: a conceptual approach*. American Institute of Aeronautics und Astronautics, Inc., 2012.

[35] Radu Bogdan Rusu u. a. „Towards 3D Point cloud based object maps for household environments“. In: *Robotics and Autonomous Systems* 56.11 (2008). Semantic Knowledge in Robotics, S. 927–941. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2008.08.005. URL: https://www.sciencedirect.com/science/article/pii/S0921889008001140.

[36] OPEN CASCADE SAS. *Open Cascade Technology main website*. URL: https://dev.opencascade.org/ (besucht am 11. 07. 2023).

[37] Michael Stark, Michael Goesele und Bernt Schiele. „Back to the Future: Learning Shape Models from 3D CAD Data“. In: Jan. 2010, S. 1–11. DOI: 10.5244/C.24.106.

[38] I. Stroud und P.C. Xirouchakis. „STL and extensions“. In: *Advances in Engineering Software* 31.2 (2000), S. 83–95. ISSN: 0965-9978. DOI: https://doi.org/10.1016/S0965-9978(99)00046-0. URL: https://www.sciencedirect.com/science/article/pii/S0965997899000460.

[39] Toru Tamaki u. a. „Scale Matching of 3D Point Clouds by Finding Keyscales with Spin Images“. In: *2010 20th International Conference on Pattern Recognition*. 2010, S. 3480–3483. DOI: 10.1109/ICPR.2010.850.

[40] Luca Tiedemann. *Erstellen eines 3D Modells aus unterschiedlich orientierten Punktewolken*. Unveröffentlichte Projektarbeit. DHBW Mannheim. 2023.

[41] Hu Wang u. a. „“Seen Is Solution” a CAD/CAE integrated parallel reanalysis design system“. In: *Computer Methods in Applied Mechanics and Engineering* 299 (2016), S. 187–214. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2015.10.022. URL: https://www.sciencedirect.com/science/article/pii/S0045782515003515.

[42] www.open3d.org. *OPEN3D Manual*. URL: http://www.open3d.org/docs/latest/index.html (besucht am 09. 08. 2023).

[43] Yu Xiang und Silvio Savarese. „Estimating the aspect layout of object categories“. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, S. 3410–3417. DOI: 10.1109/CVPR.2012.6248081.

[44] Pingkun Yan, Saad M. Khan und Mubarak Shah. „3D Model based Object Class Detection in An Arbitrary View“. In: *2007 IEEE 11th International Conference on Computer Vision*. 2007, S. 1–6. DOI: 10.1109/ICCV.2007.4409042.

[45] Chi Yuan, Xiaoqing Yu und Ziyue Luo. „3D point cloud matching based on principal component analysis and iterative closest point algorithm“. In: *2016 International Conference on Audio, Language and Image Processing (ICALIP)*. 2016, S. 404–408. DOI: 10.1109/ICALIP.2016.7846655.

[46] M. Zeeshan Zia u. a. „Detailed 3D Representations for Object Recognition and Modeling“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.11 (2013), S. 2608–2623. DOI: 10.1109/TPAMI.2013.87.