

3D-Objektposenbestimmung von Medikamentenverpackungen mit einer RGB-Kamera und synthetischen Trainingsdaten

Bachelorarbeit
im Fachbereich Informatik
am Fraunhofer IFAM

von

Sedef Pek
sedef2@uni-bremen.de

Matrikelnummer:
6116238

Betreuer:
Jannes Adam

Gutachter:
Prof. Dr.-Ing. Udo Frese
Dr. Hauke Brüning

Abgabedatum:
12.01.2026

Hinweise zu den offiziellen Erklärungen

1. Die folgende Seite mit den offiziellen Erklärungen

- A)** Eigenständigkeitserklärung
- B)** Erklärung zur Veröffentlichung von Bachelor- oder Masterarbeiten
- C)** Einverständniserklärung über die Bereitstellung und Nutzung der Bachelorarbeit / Masterarbeit in elektronischer Form zur Überprüfung durch eine Plagiatssoftware

ist entweder direkt in jedes Exemplar der Bachelor- oder Masterarbeit fest mit einzubinden oder unverändert im Wortlaut in jedes Exemplar der Bachelor- oder Masterarbeit zu übernehmen.

Bitte achten Sie darauf, jede Erklärung in allen drei Exemplaren der Arbeit zu unterschreiben.

2. In der digitalen Fassung kann auf die Unterschrift verzichtet werden. Die Angaben und Entscheidungen müssen jedoch enthalten sein.

Zu B)

Die Einwilligung kann jederzeit durch Erklärung gegenüber der Universität Bremen, mit Wirkung für die Zukunft, widerrufen werden.

Zu C)

Das Einverständnis der dauerhaften Speicherung des Textes ist freiwillig.

Die Einwilligung kann jederzeit durch Erklärung gegenüber der Universität Bremen, mit Wirkung für die Zukunft, widerrufen werden.

Weitere Informationen zur Überprüfung von schriftlichen Arbeiten durch die Plagiatsoftware sind im Nutzungs- und Datenschutzkonzept enthalten. Diese finden Sie auf der Internetseite der Universität Bremen.

Notes on the official declarations

1. The following pages with the official declarations

- A)** Declaration of Authorship
- B)** Declaration on the Publication of Bachelor's or Master's Thesis
- C)** Declaration of Consent for the Provision and Use of the Bachelor's Thesis / Master's Thesis in Electronic Form for Review by Plagiarism Software

is to be either integrated directly into each copy of the bachelor's or master's thesis or adopted unchanged in the wording of each copy of the bachelor's or master's thesis.

Please be sure to sign each declaration in all three copies of the thesis.

2. The signature can be omitted from the digital version. However, the information and decisions must be included.

Regarding B)

The consent can be revoked at any time with future effect by notifying the University of Bremen.

Regarding C)

Consent for the permanent storage of the text is voluntary. The consent can be revoked at any time with future effect by notifying the University of Bremen.

Further information on the checking of written work using plagiarism software can be found in the data protection and usage concept. This can be found on the University of Bremen website.

**Eigenständigkeits- und Einverständniserklärung zur Überprüfung mit Plagiatsoftware
sowie die Erklärung zur Veröffentlichung bei Bachelor- und Masterarbeiten****Declarations of Authorship and Consent for Checking with Plagiarism Software and the
Declaration of Publication for Bachelor's and Master's Thesis**

Studierenden-Angaben / Student Information:

Matrikelnr. / Student ID	6116238
Nachname / Surname	Pek
Vorname / First Name	Sedef

Titel der Arbeit / Title of Thesis

3D-Objektposenbestimmung von Medikamentenverpackungen mit einer RGB-Kamera und synthetischen Trainingsdaten.**A) Eigenständigkeitserklärung / Declaration of Authorship**

Ich versichere, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Teile meiner Arbeit, die wortwörtlich oder dem Sinn nach anderen Werken entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht. Gleiches gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet, dazu zählen auch KI-basierte Anwendungen oder Werkzeuge. Die Arbeit wurde in gleicher oder ähnlicher Form noch nicht als Prüfungsleistung eingereicht.

I hereby affirm that I have written the present work independently and have used no sources or aids other than those indicated. All parts of my work that have been taken from other works, either verbatim or in terms of meaning, have been marked as such, indicating the source. The same applies to drawings, sketches, pictorial representations and sources from the Internet, including AI-based applications or tools. The work has not yet been submitted in the same or a similar form as a final examination paper.

- ☒ Ich habe KI-basierte Anwendungen und/oder Werkzeuge genutzt und diese im Anhang "Nutzung KI basierte Anwendungen" dokumentiert.

I have used AI-based applications and/or tools and documented them in the appendix "Use of AI-based applications".

B) Erklärung zur Veröffentlichung von Bachelor- oder Masterarbeiten
Declaration regarding the publication of bachelor's or master's thesis

Die Abschlussarbeit wird zwei Jahre nach Studienabschluss dem Archiv der Universität Bremen zur dauerhaften Archivierung angeboten. Archiviert werden:

Two years after graduation, the thesis is offered to the archive of the University of Bremen for permanent archiving. The following are archived:

- 1) Masterarbeiten mit lokalem oder regionalem Bezug sowie pro Studienfach und Studienjahr 10 % aller Masterarbeiten
Master's theses with a local or regional focus, as well as per subject and academic year 10% of all Master's thesis
- 2) Bachelorarbeiten des jeweils ersten und letzten Bachelorabschlusses pro Studienfach und Jahr.
Bachelor's thesis for the first and last bachelor's degrees per subject and year.

☐ Ich bin damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
I agree that my thesis may be viewed by third parties in the university archive for academic purposes.

☐ Ich bin damit einverstanden, dass meine Abschlussarbeit nach 30 Jahren (gem. §7 Abs. 2 BremArchivG) im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
I agree that my thesis may be viewed by third parties for academic purposes in the university archive after 30 years (in accordance with §7 para. 2 BremArchivG).

☒ Ich bin **nicht** damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
I do not consent to my thesis being made available in the university archive for third parties to view for academic purposes.

C) Einverständniserklärung zur elektronischen Überprüfung der Arbeit auf Plagiate
Declaration of consent for electronic checking of the work for plagiarism

Eingereichte Arbeiten können nach § 18 des Allgemeinen Teil der Bachelor- bzw. der Masterprüfungsordnungen der Universität Bremen mit qualifizierter Software auf Plagiatvorwürfe untersucht werden.

Zum Zweck der Überprüfung auf Plagiate erfolgt das Hochladen auf den Server der von der Universität Bremen aktuell genutzten Plagiatssoftware.

Submitted papers can be checked for plagiarism using qualified software in accordance with § 18 of the General Section of the Bachelor's or Master's Degree Examination Regulations of the University of Bremen. For the purpose of checking for plagiarism, the upload to the server is done using the plagiarism software currently used by the University of Bremen.

- ☐ Ich bin damit einverstanden, dass die von mir vorgelegte und verfasste Arbeit zum oben genannten Zweck dauerhaft auf dem externen Server der aktuell von der Universität Bremen genutzten Plagiatssoftware, in einer institutionseigenen Bibliothek (Zugriff nur durch die Universität Bremen), gespeichert wird.

I agree that the work I have submitted and written will be stored permanently on the external server of the plagiarism software currently used by the University of Bremen, in a library belonging to the institution (accessed only by the University of Bremen), for the above-mentioned purpose.

- ☒ Ich bin **nicht** damit einverstanden, dass die von mir vorgelegte und verfasste Arbeit zum o.g. Zweck dauerhaft auf dem externen Server der aktuell von der Universität Bremen genutzten Plagiatssoftware, in einer institutionseigenen Bibliothek (Zugriff nur durch die Universität Bremen), gespeichert wird.

I do not consent to the work I submitted and wrote being permanently stored on the external server of the plagiarism software currently used by the University of Bremen, in a library belonging to the institution (accessed only by the University of Bremen), for the above-mentioned purpose.

Das Einverständnis der dauerhaften Speicherung des Textes ist freiwillig. Die Einwilligung kann jederzeit durch Erklärung gegenüber der Universität Bremen, mit Wirkung für die Zukunft, widerrufen werden. Weitere Informationen zur Überprüfung von schriftlichen Arbeiten durch die Plagiatsoftware sind im Nutzungs- und Datenschutzkonzept enthalten. Diese finden Sie auf der Internetseite der Universität Bremen.

Consent to the permanent storage of the text is voluntary. Consent can be withdrawn at any time by making a declaration to this effect to the University of Bremen, with effect for the future. Further information on the checking of written work using plagiarism software can be found in the data protection and usage concept. This can be found on the University of Bremen website.

Mit meiner Unterschrift versichere ich, dass ich die obenstehenden Erklärungen gelesen und verstanden habe und bestätige die Richtigkeit der gemachten Angaben.

With my signature, I confirm that I have read and understood the above explanations and confirm the accuracy of the information provided.

Datum / Date

Unterschrift/ Signature

Danksagung

Ich danke herzlich meinem Betreuer Jannes Adam, der die Arbeit gelesen, beim Code geholfen und mir eine Starthilfe mit der Blender-Datei gegeben hat.

Außerdem bedanke ich mich bei Udo Frese für die hilfreichen Tipps bei theoretischen und praktischen Fragen.

Ebenso danke ich Hauke Brüning, der mir ermöglicht hat, die Bachelorarbeit am Fraunhofer zu schreiben, und mir in den Terminen nützliche Hinweise gegeben hat.

Ich bedanke mich auch bei Ole Rogas, der die Arbeit korrektur gelesen und mich auf Fehler im Code hingewiesen hat.

Zusammenfassung

Diese Bachelorarbeit behandelt ein echtes Szenario aus der Logistik, bei dem Verpackungen auf einem Förderband transportiert und von einem Greifer gegriffen werden. Um das Szenario zu vereinfachen, wurde beispielhaft mit Medikamentenverpackungen gearbeitet und im synthetischen Raum modelliert. Die Arbeit beschränkt sich auf die Posenbestimmung der Verpackung im synthetischen Raum und bildet damit eine praktische Grundlage, die in Zukunft auf echte Szenarien angewendet werden kann. Bestimmt werden die X-, Y-, Z-Position der Verpackung sowie die Yaw-Rotation.

Zur Detektion der Verpackung wurde ein U-Net-Modell trainiert. Aus der segmentierten Kontur werden vier Eckpunkte extrahiert, die als 2D-Koordinaten dienen. Die dazugehörigen 3D-Koordinaten ergeben sich aus den bekannten Verpackungsgrößen. Diese Informationen werden genutzt, um mittels PnP-Verfahren die Verpackungspose relativ zur Kamera zu berechnen. Über Orientierungsmarker im Szenario erfolgt anschließend die Transformation in Weltkoordinaten.

Die Methode wurde an 500 neu generierten Bildern evaluiert. Die Positionsbestimmung erreicht durchschnittliche Fehler von 1 mm (X-Achse), 3 mm (Z-Achse) und 2 cm (Y-Achse). Die Rotationsbestimmung weist größere Abweichungen auf (durchschnittlicher Fehler: 3° , Standardabweichung: 6°).

Bereits in der Segmentierungsphase können kleine Fehler entstehen, die sich durch nachfolgende Verarbeitungsschritte aufaddieren, wobei die Eckpunktfindung besonders kritisch ist. Die identifizierten Fehlerquellen bieten jedoch konkrete Ansatzpunkte für zukünftige Optimierungen.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Motivation	1
1.2	Szenario	1
1.2.1	Reales Anwendungsszenario	1
1.2.2	Vereinfachtes Szenario der Bachelorarbeit	2
1.3	Ziele	3
1.4	Aufbau	3
2	Stand der Technik	4
2.1	RGB-Ansätze	4
2.1.1	Moderne RGB-basierte Ansätze	4
2.1.2	Herausforderungen bei RGB-Ansätzen	6
2.2	U-Net	6
2.2.1	Aufbau eines U-Nets	6
2.3	Synthetische Trainingsbilder	7
2.3.1	Moderne Ansätze zur Generierung synthetischer Bilder	7
2.3.2	Labeling bei synthetischen Bildern	8
2.3.3	Herausforderungen	8
3	Methodik	10
3.1	Synthetische Trainingsdaten	10
3.1.1	Erstellung der synthetischen Bilder	10
	Modifikationen der Ausgangsszene	11
3.1.2	Erstellung der Bildmaske	13
3.1.3	Erstellung der Metadaten	13
3.2	Segmentierung mit U-Net	14
3.2.1	Anpassungen der Basis-Implementierung	14
3.2.2	Training	15
3.2.3	Evaluation	16
3.3	Objektposenbestimmung	17
3.3.1	Orientierungsmarkererkennung	17
3.3.2	2D-Eckpunkterkennung der Verpackung	18
3.3.3	Berechnung der Kamerapose	19
3.3.4	Berechnung der Rotation, Position und Distanz	20
4	Ergebnisse	22
4.1	Synthetische Trainingsdaten	22
4.1.1	Synthetische Bilder	22
4.1.2	Bildmaske	25
4.1.3	JSON-Datei	26
4.2	U-Net	26
4.2.1	Erkennung von neuen synthetisch generierten Bildern	27
4.3	Objektposenbestimmung	27

4.3.1	Orientierungsmarkererkennung	27
4.3.2	2D-Eckpunkterkennung der Verpackung	28
4.3.3	Berechnung der Verpackungsposition und -distanz	29
4.3.4	Berechnung der Verpackungsrotation	30
5	Diskussion	32
5.1	Synthetische Trainingsdaten	32
5.1.1	Erstellung der synthetischen Bilder	32
5.1.2	Erstellung der Segmentierungsmasken	33
5.1.3	Erstellung der JSON	34
5.2	U-Net	35
5.3	Objektposenbestimmung	36
5.3.1	Orientierungsmarkererkennung	36
5.3.2	2D-Eckpunkterkennung der Verpackung	37
5.3.3	Berechnung der Verpackungsposition und -distanz	38
5.3.4	Berechnung der Verpackungsrotation	39
6	Fazit	41
	Abkürzungsverzeichnis	42
	Abbildungsverzeichnis	43
	Tabellenverzeichnis	43
	Literaturverzeichnis	44
	Anhang	I

1

Einleitung

1.1 Motivation

Die Objekterkennung und Positionsschätzung spielen heutzutage eine große Rolle, beispielsweise beim autonomen Fahren [17] oder in der Lagerlogistik [1]. Das Erkennen von Objektposition und -rotation ist essenziell, damit Robotersysteme Objekte präzise greifen können [11].

Aktuelle Ansätze nutzen meist Tiefeninformationen aus beispielsweise RGB-D-Kameras, um Objekte zuverlässig zu lokalisieren. Jedoch können solche Kameras kostspielig sein und Herausforderungen bei glänzenden oder transparenten Objekten haben [1]. RGB-Kameras hingegen sind kostengünstig und einfach zu bedienen. Sie sind eine praktikable Alternative zur teuren Hardware.

Für die Objekterkennung werden häufig neuronale Netze eingesetzt, die mit tausenden gelabelten Bildern trainiert werden müssen [11]. Hier entsteht jedoch eine neue Hürde: Das manuelle Labeling dieser Daten ist zeitaufwendig und kann potenziell fehlerhaft sein, etwa durch menschliche Fehler oder Bias [22]. Synthetische Bilder bieten hier eine Lösung. Sie können automatisch in großen Mengen generiert werden und liefern dabei präzise Labels ohne großen Aufwand [22].

Diese Arbeit untersucht, ob RGB-basierte 3D-Objektposenbestimmung mit ausschließlich synthetisch generierten Trainingsdaten eine praktikable und kostengünstige Alternative darstellen kann. Die Evaluation erfolgt auf synthetischen Testdaten am Beispiel medizinischer Verpackungen und bildet die Grundlage für eine zukünftige Anwendung in einem realen Szenario.

1.2 Szenario

1.2.1 Reales Anwendungsszenario

Die Bachelorarbeit bezieht sich auf ein reales Szenario aus der Logistik, bei dem verschiedene Verpackungen auf einem Förderband transportiert und von einem Greifer in das entsprechende Regal einsortiert werden. Dabei erfasst eine am Robotersystem befestigte RGB-Kamera das Förderband.

Das Problem bei dem Szenario ist, dass die Verpackungen auf dem Förderband gelegentlich rotiert liegen oder nicht mittig vor dem Greifer positioniert sind. In diesen Fällen kann der Greifer die Verpackungen nicht zuverlässig greifen und fährt wiederholt vor und zurück.

Das Ziel für das reale Szenario ist eine verbesserte Objekterkennung, die präzise die Position und Rotation einer Verpackung bestimmt, ihre Entfernung ermittelt und bewertet, ob der Greifer unter diesen Bedingungen greifen kann. Die Orientierung der Verpackungsflächen ist dabei irrelevant, da von jeder Seite gegriffen werden kann.

Das Förderband hat eine konstante, bekannte Höhe. Für die Posenschätzung sind daher nur die seitliche Position und die Tiefe relevant. Da die Verpackungen flach aufliegen, beschränkt sich die Rotationsschätzung auf die Yaw-Rotation.

Das System muss zunächst die Verpackung finden und anschließend den Greifarm entsprechend der Tiefe ausfahren. Dabei sollte die Erkennung möglichst präzise sein. Das Szenario setzt eine bestehende RGB-Kamera voraus, wobei auf zusätzliche Hardware wie RGB-D-Kameras verzichtet werden soll. Die Verpackungsgrößen sind bekannt.

Daraus ergeben sich verschiedene Schwierigkeiten: Über dem Förderband befinden sich Regale mit mehreren verschiedenen Verpackungen, was die Objekterkennung erschwert. Auch sind die Lichtverhältnisse nicht optimal, da die Szene nicht gut beleuchtet ist. Außerdem fehlen jegliche Tiefeninformationen, da nur eine RGB-Kamera verfügbar ist. Ein weiteres Problem stellen die verschiedenen Verpackungstypen mit unterschiedlichen Größen dar.

1.2.2 Vereinfachtes Szenario der Bachelorarbeit

Das oben genannte Szenario wurde für diese Arbeit vereinfacht und im virtuellen Raum implementiert. Um den Trainingsaufwand zu begrenzen, wurden beispielhaft Medikamentenverpackungen als Fokus dieser Arbeit gewählt.

Die Trainingsdaten wurden vollständig synthetisch mit Blender generiert. Dabei wurden folgende Vereinfachungen vorgenommen:

- Auf dem Förderband befindet sich nur eine einzelne Verpackung.
- Alle Verpackungen sind quaderförmig.
- Die Texturen und Details der Verpackungen sowie der Umgebung sind weniger komplex als in der Realität.
- Es wurden Marker auf der Greiferplatte platziert, um die Kamerakalibrierung zu vereinfachen.
- Es wurde keine Linsenverzeichnung modelliert.
- Die Regale im Hintergrund sind leer.

1.3 Ziele

Im Rahmen der Bachelorarbeit wird untersucht, inwieweit eine RGB-basierte Posen-schätzung für Medikamentenverpackungen realisierbar ist und welche Genauigkeit dabei erreicht werden kann. Die Frage wird ausschließlich anhand von synthetischen Trainingsbildern evaluiert. Dabei soll demonstriert werden, dass auch ohne aufwendiges manuelles Labeling und zusätzliche Hardware wie Tiefensensoren praktikable Ergebnisse erzielt werden können.

Wissenschaftliche Ziele:

1. Bestimmung der Genauigkeit der 6D-Posenschätzung der Verpackung mit den vorgestellten Methoden.
2. Analyse potenzieller Fehlerquellen der entwickelten Methode und möglicher Lösungsansätze.

Praktische Ziele:

1. Entwicklung einer Pipeline zur Objektposenbestimmung, die ausschließlich auf RGB-Daten basiert.
2. Erzeugung eines synthetischen Datensatzes in Blender mit verschiedenen Verpackungspositionen und -größen.
3. Training eines U-Net-Modells zur Segmentierung der Verpackungen.
4. Bestimmung der 6D-Pose mittels PnP-Verfahren.
5. Evaluation an neu generierten synthetischen Bildern.

1.4 Aufbau

Kapitel 2 gibt einen Überblick über den aktuellen Stand der Forschung zu den für diese Arbeit relevanten Themen. Anschließend wird die Methodik beschrieben, gegliedert in die Generierung synthetischer Trainingsdaten, das Training des U-Net-Modells und die 3D-Objektposenbestimmung. Kapitel 4 präsentiert die Ergebnisse, gefolgt von einer Diskussion in Kapitel 5 und einem Fazit in Kapitel 6.

2

Stand der Technik

In diesem Kapitel werden die Themen RGB-Ansätze, U-Net und synthetische Bilder mit aktuellen Arbeiten beleuchtet.

2.1 RGB-Ansätze

RGB-basierte Verfahren zur Objekterkennung nutzen ausschließlich Bildinformationen einer RGB-Kamera, welche keine Tiefendaten liefert. Das RGB-Bild liefert lediglich Farb-, Textur- und Lichtinformationen der Szene [25].

2.1.1 Moderne RGB-basierte Ansätze

Die 6D-Posenbestimmung mit ausschließlich RGB-Kameras ist ein aktives Forschungsgebiet mit vielfältigen Lösungsansätzen. Im Folgenden werden drei Methoden vorgestellt, die unterschiedliche Herangehensweisen zur RGB-basierten Posenschätzung verdeutlichen.

HccePose(BF)

HccePose(BF) belegte 2024 den ersten Platz bei der BOP Challenge (Benchmark for 6D Object Pose Estimation), einem jährlichen Wettbewerb zur Vorhersage der 6D- oder 2D-Pose von Objekten aus RGB- und/oder RGB-D-Daten [2]. Das zugehörige Paper wurde im Oktober 2025 von Wang et al. veröffentlicht [23]. Für diese Bachelorarbeit ist die Kategorie “Model-based tasks on seen objects: 6D detection” relevant: Das Modell wird auf bekannte Objekte trainiert und muss diese zunächst im Bild lokalisieren, bevor die 6D-Pose bestimmt wird [2].

Bei der vorgestellten Methode von Wang et al. [23] wird für die 6D-Pose nicht nur die Vorderseite des Objekts betrachtet, sondern auch die Rückseite. Das Eingabebild wird einem neuronalen Netz übergeben, welches mit Hierarchical Continuous Coordinate Encoding (HCCE) die Objektmaske sowie die 3D-Koordinaten der Vorder- und Rückseite schätzt. Die Pixel innerhalb der Objektmaske bilden dabei die 2D-Koordinaten für das RANSAC-PnP-Verfahren. Für jeden 2D-Punkt werden die zugehörigen 3D-Punkte (Vorder- und Rückseite) geschätzt, wobei zusätzlich zwischen diesen interpoliert wird. Alle diese 3D-Punkte sind zwar unterschiedlich, teilen

aber denselben 2D-Punkt. Pro RANSAC-PnP-Iteration wird pro 2D-Punkt nur ein 3D-Punkt aus dieser Menge gewählt. Die Pose mit dem geringsten Fehler wird als Ergebnis gewählt [23].

BOP bietet sieben Datensätze an. Die Evaluation erfolgt anhand von zwei Metriken (MSSD, MSPD), die von BOP vorgegeben werden [2]. Davon wird dann der Durchschnitt berechnet und bildet so die Genauigkeit für das jeweilige Datenset. Der Average Precision Core (AP_{Core}) ist der Durchschnitt aller Ergebnisse der Datensätze. Dabei erreicht HcpePose(BF) im Leaderboard einen AP_{Core} von 78,4% [2].

RGBTrack

Ein weiterer moderner Ansatz ist RGBTrack von Guo et al. (2025), der auf FoundationPose von Wen et al. (NVIDIA) aufbaut, jedoch ohne Tiefendaten auskommt [9][24]. Der Ansatz setzt voraus, dass das Objekt bereits lokalisiert wurde und eine Objektmaske sowie das CAD-Modell vorliegen.

Guo und Yu [9] schätzen zuerst mit einer Binärsuche die Tiefe des Objekts. In jeder Iteration wird das CAD-Modell auf einer angenommenen Tiefe gerendert und die Fläche der gerenderten Maske mit der beobachteten Objektmaske verglichen. Bei Abweichungen wird die Tiefe angepasst, bis die gerenderte Maske mit der beobachteten übereinstimmt. Sobald die initiale 6D-Pose gefunden wurde, erfolgt das Tracking nur noch auf Basis der RGB-Features und der vorherigen Pose [9].

RGBTrack läuft in Echtzeit mit 22 FPS bei einer Auflösung von 640×480 . Auf dem YCBInEOAT-Dataset erreicht RGBTrack bei 7 von 9 Testobjekten Recall-Werte über 70%. Im Vergleich dazu erreicht FoundationPose, das zusätzlich Tiefendaten nutzt, bei 8 von 9 Objekten Recall-Werte über 89% [9].

Der Vergleich zeigt, dass der Verzicht auf Tiefendaten hier zu einer geringeren Genauigkeit führt.

GDR-Net

GDR-Net ist ein weiterer RGB-basierter Ansatz zur 6D-Posenschätzung, der 2021 von Wang et al. veröffentlicht wurde [13].

Im Paper wird beschrieben, dass eine bereits implementierte 2D-Objektdetektion genutzt wird, wie beispielsweise R-CNN oder FCOS, um das Objekt im Bild zu lokalisieren [13]. Anschließend wird bei jeder Detektion auf die Region of Interest gezoomt und dem neuronalen Netz übergeben, welches mehrere Feature Maps ausgibt. Diese werden dann dem Patch-PnP-Modul übergeben, welches die 6D-Pose ausgibt [13].

Im Gegensatz zu klassischen PnP-Verfahren, die die 6D-Pose mathematisch aus 2D-3D-Korrespondenzen berechnen, lernt Patch-PnP als neuronales Netz die Pose aus den geometrischen Feature Maps zu schätzen [13].

Auf dem LM-O-Dataset erreicht GDR-Net mit einem Netz für alle Objekte einen Recall-Wert von 47,4% und übertrifft damit PoseCNN (24,9%) unter gleichen Trainingsbedingungen [13].

2.1.2 Herausforderungen bei RGB-Ansätzen

RGB-Kameras liefern keine Tiefeninformationen, weshalb die Tiefe für die Objekterkennung selbst berechnet werden muss. Dies kann fehleranfällig sein, da Informationen bei der 3D-zu-2D-Projektion verloren gehen [17].

Um diese Tiefe berechnen zu können, sind intrinsische und extrinsische Kameraparameter nötig. Diese sind jedoch nicht immer verfügbar, was die Berechnung erschwert [17].

Eine weitere Herausforderung ist die Okklusion, die Verdeckung von Objekten. Dabei gibt es zwei Fälle: Zum einen verdeckt sich das Objekt selbst, zum anderen wird das Objekt von einem anderen Objekt verdeckt. In beiden Fällen ist es mit RGB-Kameras schwierig, die Objekte voneinander zu unterscheiden, da ohne Tiefeninformation der Abstand zwischen den Objekten nicht erkannt werden kann. Da hier nur mit Farbinformationen gearbeitet wird, können Objekte mit ähnlichen Farben vermischt werden [4].

Laut Chandel und Vatta lässt sich das Okklusionsproblem mit verschiedenen Ansätzen lösen. Ein Ansatz ist die Verwendung mehrerer Kameras. Weitere Ansätze umfassen Cross Correlation, Kalman-Filter oder Support-Vektor-Maschinen [4].

2.2 U-Net

U-Net ist eine Convolutional Neural Network (CNN) Architektur, die 2015 von Ronneberger, Fischer und Brox für die biomedizinische Bildsegmentierung entwickelt wurde [18].

Die Grundidee von U-Net ist, mit wenigen annotierten Trainingsdaten präzise Segmentierungen zu erzielen. So benötigt ein U-Net im Vergleich zu anderen tiefen neuronalen Netzen weniger Trainingsdaten, die dann effizienter verwendet werden [18]. Ronneberger et al. zeigen, dass bei einem 512×512 Bild die Segmentierung unter einer Sekunde benötigt [18]. Außerdem ist das Training End-to-End möglich [18].

2.2.1 Aufbau eines U-Nets

Abbildung 2.1 zeigt den Aufbau der U-Net-Architektur. Im absteigenden Pfad (Kontraktionspfad) werden pro Block zwei 3×3 -Faltungen, jeweils gefolgt von ReLU-Aktivierung und einer 2×2 -Max-Pooling-Operation zur Reduktion der räumlichen Auflösung, durchgeführt. Dabei verdoppelt sich bei jedem Schritt die Anzahl der Merkmalskanäle. Im aufsteigenden Pfad (Expansionspfad) wird die Auflösung durch Upsampling mit 2×2 -Faltungen wiederhergestellt, wobei die Anzahl der Kanäle jeweils halbiert wird. Um keine Informationen zu verlieren, werden die Feature Maps aus dem Kontraktionspfad zugeschnitten und mit dem Expansionspfad konkateniert (Copy and Crop). Abschließend reduziert eine 1×1 -Faltung die Merkmalskanäle auf die Anzahl der Zielklassen [18].

Vereinfacht ausgedrückt analysiert das U-Net das Bild, indem es dieses zunächst verkleinert, um Merkmale zu extrahieren, und dann wieder vergrößert, wobei es jedem Pixel eine Klasse zuweist. Beim Training lernt es anhand von Bildmasken die korrekte Zuordnung und kann anschließend neue Bilder selbstständig segmentieren.

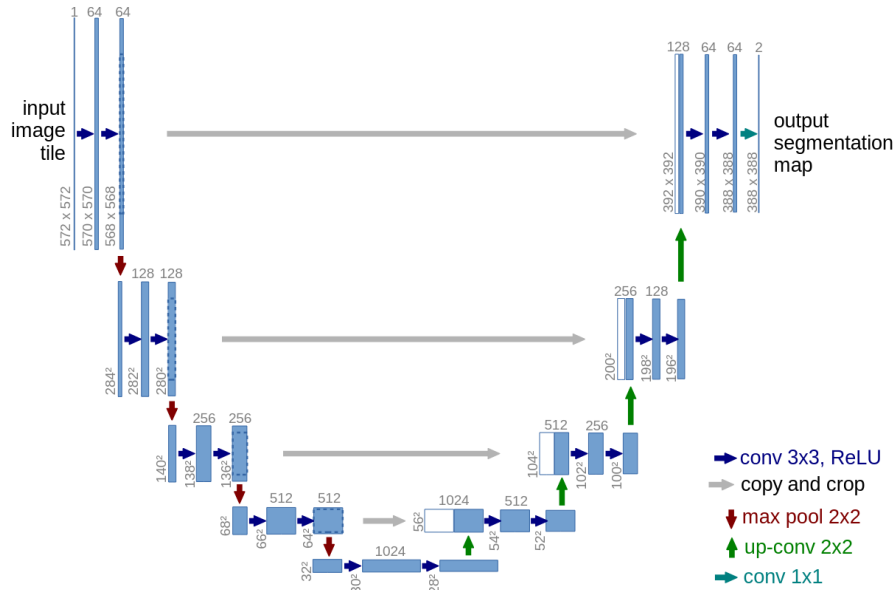


Abbildung 2.1: Architektur des U-Net mit Kontraktionspfad (links) und Expansionspfad (rechts). [18]

2.3 Synthetische Trainingsbilder

Für das Training von Deep-Learning-Modellen werden je nach Komplexität des Modells große Mengen annotierter Bilder benötigt. Die manuelle Annotation ist jedoch zeit- und kostenintensiv [22]. Eine Methode, um das Erstellen von Trainingsbildern zu vereinfachen, ist die Verwendung synthetischer Trainingsbilder [22].

2.3.1 Moderne Ansätze zur Generierung synthetischer Bilder

In diesem Abschnitt werden drei verschiedene Ansätze beleuchtet, um synthetische Bilder zu generieren.

Synthetic-to-Real Domain Adaptation using CUT

Imbusch et al. veröffentlichen 2022 ein Paper, in dem zwei Ansätze zur Erzeugung realistischer synthetischer Bilddaten kombiniert werden [10].

Zunächst werden mithilfe des von Schwarz und Behnke 2020 vorgestellten Frameworks Stillleben [20] synthetische Szenen erzeugt, indem 3D-Objekte zufällig auf einer Tischoberfläche platziert und gerendert werden [10]. Anschließend werden die synthetischen Bilder durch Contrastive Unpaired Translation (CUT), einen von Park et al. 2020 vorgestellten Ansatz [15], an die visuelle Erscheinung realer Fotografien angepasst [10]. Laut Imbusch et al. ermöglicht dieser kombinierte Ansatz die schnelle Generierung realitätsnaher Trainingsdaten, die innerhalb eines Tages einsatzbereit sind [10].

Am YCB-Video-Datensatz zeigen Imbusch et al., dass die Segmentierung mit synthetischen Bildern, die mit CUT verfeinert wurden, fast so gut ist wie mit echten Bildern [10].

InstaGen

Einen weiteren Ansatz untersucht das Paper “InstaGen: Enhancing Object Detection by Training on Synthetic Dataset” [8].

Dabei wird Stable Diffusion genutzt, das Texte in fotorealistische Bilder umwandelt und zusätzlich automatisch Bounding Boxes generiert. Durch diese Methode entfällt der Aufwand, manuelle Annotationen für die Trainingsdaten zu erstellen [8].

Dieser Ansatz hat jedoch auch Nachteile: So besteht immer noch eine Domain Gap, da die synthetischen Bilder nicht dieselbe Komplexität wie die echten Bilder aufweisen. Außerdem hat der Ansatz Schwierigkeiten seltene Objekte zu erzeugen [8].

Trotz dieser Einschränkungen zeigt sich in anderen Bereichen bereits großes Potenzial für synthetisch generierte Daten: Die FRCSyn Challenge bei CVPR 2024 zeigt, dass synthetische Bilder stetig besser werden. So wird gezeigt, dass mit unbegrenzten synthetischen Daten sehr gute Ergebnisse erzielt werden können, die in einigen Fällen sogar das Training mit echten, aber limitierten Daten übertreffen [5].

BlenderProc

Ein hilfreiches Werkzeug bei der Erstellung synthetischer Trainingsbilder mit Blender ist die Software BlenderProc. BlenderProc wurde 2019 vom Deutschen Zentrum für Luft- und Raumfahrt (DLR) entwickelt und automatisiert den Prozess der Generierung realistischer synthetischer Bilder [6].

Die Pipeline hat Module für Objekte, Kameras und Licht, um große Mengen fotorealistischer Trainingsbilder erstellen zu können. Dadurch wird die Erstellung synthetischer Datensätze deutlich effizienter [6].

Außerdem hat BlenderProc verschiedene Ausgabemöglichkeiten, beispielsweise Farbbilder, Tiefenbilder und Segmentierungsmasken, die später als Ground Truth für CNNs genutzt werden können [6], wie es in dieser Arbeit auch gemacht wird.

2.3.2 Labeling bei synthetischen Bildern

Präzise Labels sind entscheidend für gut trainierte Modelle, doch das Labeln von Bildern kann sehr zeitaufwendig sein. Ein Vorteil synthetischer Daten liegt darin, dass die passenden Labels präzise und in kurzer Zeit mitgeneriert werden können [22].

Es gibt viele verschiedene Methoden zu Labeln, beispielsweise Bounding Boxen oder Keypoints, doch für diese Arbeit wurde semantische Segmentierung genutzt, welche für jedes Pixel im Bild eine Klasse zuweist [21].

Da die Szene vereinfacht in Blender nachmodelliert wurde, können die Segmentierungsmasken mit BlenderProc zusammen mit den Bildern ausgegeben werden [6].

2.3.3 Herausforderungen

Das Hauptproblem synthetischer Bilder liegt im sogenannten Domain Gap: Synthetische Daten unterscheiden sich in ihrer visuellen Erscheinung deutlich von realen

Aufnahmen, insbesondere bei den Details und Texturen, aber auch an den Kameraeigenschaften, beispielsweise Bildrauschen. Diese Diskrepanz kann dazu führen, dass Modelle, die ausschließlich auf synthetischen Daten trainiert wurden, bei realen Anwendungen schlechtere Leistungen zeigen [7].

Nach Eversberg und Lambrecht kann dieses Problem wie folgt gelöst werden: Das Modell wird zunächst mit synthetischen Bildern vortrainiert und anschließend mit echten Bildern verfeinert. Damit wird die Leistung deutlich verbessert [7]. Zu diesem Schluss kamen auch Movshovitz-Attias et al. im Paper “How useful is photo-realistic rendering for visual learning?” [14].

Ein weiteres Problem, das nicht nur bei synthetischen Bildern vorkommt, ist die Klassen-Imbalance. Dieses Ungleichgewicht entsteht, wenn eine Klasse (z.B. Verpackung) deutlich seltener vorkommt als andere Klassen (z.B. Hintergrund), wodurch das Modell zur häufigeren Klasse tendiert [19].

Ein Ansatz, um dieses Ungleichgewicht zu korrigieren, ist der sogenannte Focal Loss. Dabei wird die Verlustfunktion so verändert, dass das Modell nicht mehr so stark auf die ganzen einfachen Hintergrundbereiche achtet, sondern sich mehr auf die schwierigen Stellen fokussiert. Dies ermöglicht ein gutes Training, selbst bei starken Ungleichheiten [12].

3

Methodik

In diesem Kapitel wird die gesamte Pipeline für die 3D-Objektposenbestimmung ausgeführt. Die Pipeline ist in drei Hauptphasen gegliedert: Die Erstellung der synthetischen Bilder, das Training des U-Net-Modells und die 3D-Objektposenbestimmung. Jede Phase baut auf der vorherigen auf.

3.1 Synthetische Trainingsdaten

Für das Training des U-Net-Modells werden RGB-Bilder mit präzisen Segmentierungsmasken benötigt. Zusätzlich sind Metadaten in Form einer JSON-Datei erforderlich, welche die Szene mathematisch beschreibt und sowohl als Vorwissen für die Objektposenbestimmung als auch als Ground Truth für die Evaluation dient.

Die Generierung umfasst drei Hauptkomponenten: Die Bilderzeugung, die automatische Erstellung der Segmentierungsmaske und die Extraktion der Metadaten.

Ziel ist es, synthetische Bilder medizinischer Verpackungen zu generieren, welche die wesentlichen visuellen Merkmale realer Aufnahmen abbilden.

3.1.1 Erstellung der synthetischen Bilder

Für die Generierung der synthetischen Bilder wurde Blender 4.3 in Kombination mit BlenderProc verwendet.

Als Ausgangsbasis diente ein Python-Skript und eine Blender-Datei von Jannes Adam, welche einfache quaderförmige Objekte mit unterschiedlichen Rotationen und Farben auf einem Förderband darstellte. Vor dem Paket befindet sich eine Greiferplatte mit zwei Greiferarmen. Das Basisskript automatisiert die komplette Bildgenerierung: Es lädt die vordefinierte Blender-Szene mit Förderband, Greifer, Wand und Verpackung, randomisiert relevante Eigenschaften wie beispielsweise Verpackungsgröße, rendert das Bild und speichert das RGB-Bild ab.

Diese Basis-Implementation, wie beispielhaft in Abbildung 3.1 gezeigt, wurde als Ausgangspunkt genutzt und für die spezifischen Anforderungen dieser Arbeit erweitert.

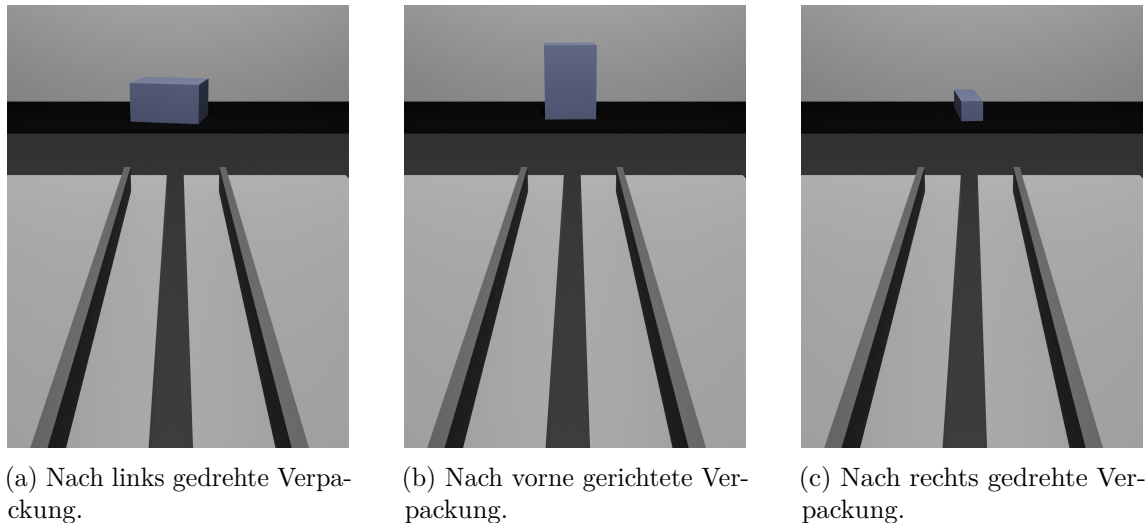


Abbildung 3.1: Initiale Blender-Szene vor Bearbeitung.

Modifikationen der Ausgangsszene

Variierende Position, Rotation und Größe Im echten Szenario befindet sich die Verpackung an unterschiedlichen Positionen, was ein großes Problem für den Greifer darstellt, da das System annimmt, dass sich die Verpackung mittig befindet.

Mit den variierenden Verpackungspositionen wird sichergestellt, dass das trainierte Modell überall auf dem Förderband die Verpackung erkennt. Außerdem wird der Fall, dass die Verpackung sich nicht im Bild befindet, ebenfalls behandelt. Damit soll das Modell lernen, zwischen Szenen mit und ohne Verpackung zu unterscheiden.

Zusätzlich können die Verpackungen um $\pm 45^\circ$ um die Z-Achse (Yaw) rotiert sein. Da die Verpackung quaderförmig ist, wiederholt sich die Ansicht alle 90° . Wenn nämlich die Verpackung um 90° gedreht wird, so sieht sie aus wie eine Frontansicht. Der Bereich von $\pm 45^\circ$ deckt somit alle möglichen Orientierungen ab, da eine Drehung von beispielsweise 50° optisch einer Drehung von -40° mit einer anderen Frontseite entspricht.

Die Verpackungen haben, wie es bereits in der Basis-Implementation vorhanden ist, unterschiedliche Größen, um verschiedene Verpackungstypen abzudecken. Dabei wurde darauf geachtet, dass die Abmessungen physikalisch plausibel sind. So können die Verpackungen maximal 16 cm in jeder Dimension haben und minimal 4 cm in Breite und Tiefe sowie 2 cm in Höhe haben. Damit ist es ausgeschlossen, dass sehr dünne Verpackungen extrem hoch werden, da im echten Szenario diese flach auf dem Förderband liegen würden.

Kameraposition Die Kameraposition bleibt, wie im Basisskript, fest montiert, um die reale Bedingung einer statischen Kamera auf einem Greifarm nachzubilden. Durch die feste Kameraposition kann die Kamera leichter kalibriert werden und präzise Messdaten, beispielsweise für die Distanz zwischen Verpackung und Kamera, liefern.

Realistische Beleuchtung Die Lichtquelle im echten Szenario kommt von vorne, wo sich auch die Kamera befindet, da die Lichtquelle mit dem Greifer verbunden

ist. Jedoch wurden leichte Schwankungen eingebaut, was die Position angeht, um verschiedene Lichtpositionen abzudecken, sowie für den Fall, dass das Licht wackelt bei einer Bewegung. Damit soll das Modell robuster für solche Fälle sein.

Oberfläche der Verpackung Die Verpackungen wurden mit Text erweitert. So wird ein zufällig generierter Text auf die Verpackung eingefügt, da Medizinverpackungen meist beschriftet sind. Dabei variieren die Schriftgrößen, um Überschriften und kleinere Texte widerzuspiegeln. Um verschiedene Logos und Markierungen auf den Verpackungen abzubilden, wurden zufällige farbliche Quadrate auf der Verpackung platziert.

Damit soll das Modell erkennen, dass eine Verpackung trotz verschiedenfarbiger Bereiche als ein einzelnes Objekt segmentiert wird und nicht fälschlicherweise als mehrere separate Verpackungen klassifiziert werden.

Grundfarbe der Verpackung Arzneiprodukte haben meist helle Verpackungen, damit die Aufschrift besser gelesen werden kann. Um dies abzubilden, wurde die Wahrscheinlichkeit für weiße Verpackungen auf 50 % gesetzt. Die restlichen 50 % verteilen sich auf unterschiedliche Farben, zwischen sanften Pastelltönen bis hin zu knalligen Farben. So werden sowohl typische Verpackungen als auch seltene Extremfälle abgedeckt, wodurch das Modell robuster gegenüber verschiedenen Verpackungsdesigns wird.

Orientierungsmarker Für die Objektposenbestimmung sind neben den Kameraparametern auch Referenzpunkte in der Umgebung notwendig. Dafür wurden vier schwarz-gelbe Symbole (Crash-Test-Markierungen) auf die Greiferplatte angebracht: Jeweils zwei Marker links und zwei Marker rechts, die zusammen ein Rechteck aufspannen (siehe Abbildung 3.2).

Da die Kamera fest am Greifer montiert ist, ist die Greiferplatte das einzige Objekt, was immer im Sichtfeld der Kamera ist. Eine Platzierung weiter hinten würde zwar Extrapolation vermeiden, ist jedoch aus diesem Grund nicht umsetzbar.

Die Marker dienen als feste Orientierungspunkte und können auch im realen System verwendet werden.

Skalierung Die Skalierung wurde angepasst, damit die Paketgrößen und der Abstand zur Kamera mehr der Realität entsprechen. So kann die Verpackung maximal 16 cm hoch, breit und tief sein. Das Minimum liegt bei der Breite und Tiefe bei 4 cm, während das Minimum bei der Höhe bei 2 cm liegt. Damit sollen so viele realistische Verpackungsgrößen dargestellt werden wie möglich. Außerdem beträgt der Abstand zur Kamera ungefähr 1 m. Zusätzlich wurde die Bildgröße auf 640 px hoch und 480 px breit eingestellt, da genauso das echte Szenario aufgenommen wurde.

Die Abbildung 3.2 zeigt die oben genannten Veränderungen an drei verschiedenen Outputs. Dabei ist jede Verpackung einzigartig aufgrund der verschiedenen Farben und Texte.

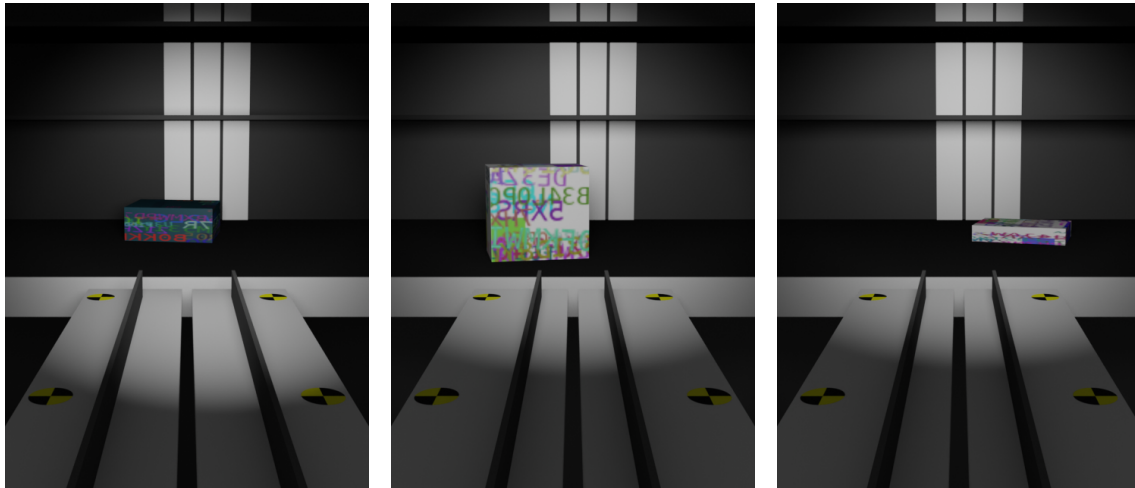


Abbildung 3.2: Beispielbilder nach Anpassung der Basisszene. Auf der Boxoberfläche gibt es zufällige Farb- und Textmuster, die Lichtquelle variiert leicht und es gibt Orientierungsmarker.

3.1.2 Erstellung der Bildmaske

Die ursprüngliche Implementation enthielt keine automatische Label-Generierung für die Trainingsbilder. Da für das U-Net-Training präzise Labels als Ground Truth erforderlich sind, wurde ein Algorithmus in der Pythondatei eingefügt.

Für jedes generierte RGB-Bild wird automatisch eine binäre Bildmaske erstellt. Da Blender genutzt wird, ist es eindeutig, welche Pixel zur Verpackung gehören. Die Maske kennzeichnet alle Pixel, die zur Verpackung gehören, mit dem Wert 1 (weiß), während der Hintergrund mit 0 (schwarz) markiert wird. Anschließend werden die Bildmasken als 1-Kanal-Bilder im PNG-Format mit derselben Auflösung wie die RGB-Bilder (480×640 Pixel) gespeichert.

Wenn die Verpackung nicht vor der Kamera ist, sind auch auf der Bildmaske keine Verpackungspixel vorhanden. Es wird nur das gerendert, was vor der virtuellen Kamera sichtbar ist. Dasselbe gilt für abgeschnittene Verpackungen: Die Verpackungspixel sind dann ebenfalls abgeschnitten. Die abgeschnittenen Verpackungen sind jedoch nur Sonderfälle, die manuell generiert werden müssen.

Verdeckungen durch andere Objekte würden ebenfalls dazu führen, dass die entsprechenden Verpackungspixel auf der Bildmaske nicht angezeigt werden. In diesem vereinfachten Szenario treten solche Verdeckungen jedoch nicht auf.

3.1.3 Erstellung der Metadaten

Für jedes Bild werden zusätzlich Metadaten in Form einer JSON-Datei generiert, die folgende Informationen enthält:

- Bildeigenschaften (ID, Auflösung).
- Informationen zur Verpackung (Position, Rotation, Größe, Eckpunkte).
- Informationen zur Kamera (Position, Rotation, Brennweite, Sensorgröße, Kamera intrinsische Matrix).

- Position der Orientierungsmarker in 2D und 3D, sowie ihre Abstände zueinander.
- Der Abstand zwischen Kamera und Verpackung.

Die meisten Eigenschaften wie Objektposition, Rotation, Kameramatrix usw. wurden mit bereits vordefinierten Funktionen aus Blender und BlenderProc ausgegeben. Für die Distanzberechnung zwischen Kamera und Verpackung wurde die Euklidische Distanz verwendet.

Für die Bestimmung der 3D-Eckpunkte wurden mit der Blender-Funktion `bound_box` die acht Koordinaten ausgegeben und mit `world_to_camera_view` in 2D-Pixelkoordinaten übertragen. Bei den Crash-Symbolen wurden die 3D-Positionen manuell festgelegt, die Umwandlung ins 2D erfolgte jedoch analog zu den Eckpunkten.

Die Kameraeigenschaften und 3D-Orientierungsmarker-Positionen werden für die Objektposenbestimmung benötigt (siehe Abschnitt 3.3.3). Die Informationen zu der Verpackung und der Abstand zur Kamera dienen als Ground Truth und werden zur Evaluation der Ergebnisse verwendet.

3.2 Segmentierung mit U-Net

Für die semantische Segmentierung wurde ein U-Net-Modell basierend auf der Implementierung von Aladdin Persson verwendet [16]. Diese Basis besteht aus vier Pythondateien:

- `train.py`: Die Haupttrainingsschleife.
- `utils.py`: Hilfsfunktionen für die Loader, Checkpoints und Evaluation.
- `model.py`: Die U-Net Architektur selbst.
- `dataset.py`: Lädt die Trainingsbilder.

Das Modell verarbeitet RGB-Bilder (3 Kanäle) mit 480×640 Pixeln Auflösung und erzeugt eine binäre Segmentierungsmaske gleicher Größe. Die Ausgabe wird durch eine Sigmoid-Funktion in Wahrscheinlichkeiten zwischen 0 und 1 transformiert.

Es wurde die U-Net Architektur gewählt, da sie sich besonders gut für pixelgenaue Klassifikation eignet. Die Architektur ist zudem effizient und leicht anzuwenden.

Ziel ist es, die medizinische Verpackung auf dem Förderband so präzise wie möglich zu erkennen. Das Modell soll robust sein und in zukünftigen Arbeiten potenziell auf echten Bildern angewendet werden können.

3.2.1 Anpassungen der Basis-Implementierung

Die ursprüngliche Implementierung wurde für die spezifischen Anforderungen dieser Arbeit erweitert:

Test-Set Es wurde eine Funktion hinzugefügt, die am Ende des Trainings das Modell an einem separaten Testdatensatz evaluiert. Damit soll geprüft werden, ob das Modell overfittet und ob die synthetischen Verpackungen korrekt segmentiert werden.

Zusätzliche Augmentationen Die Augmentationen wurden angepasst (siehe Abschnitt 3.2.2). Diese sollen das Modell auf beispielsweise verschiedene Beleuchtungsbedingungen vorbereiten und die Domain Gap zwischen synthetischen und realen Bildern reduzieren.

Early Stopping Zuvor wurde nach jeder Epoche das neueste Modell abgespeichert, was jedoch problematisch sein kann, wenn das Modell overfittet.

Um Overfitting zu vermeiden, wurde Early Stopping implementiert. Nach jeder Epoche wird der aktuelle Checkpoint gespeichert. Zusätzlich wird bei jeder Verbesserung des Dice Scores das Modell separat als bestes Modell gespeichert. Verbessert sich der Dice Score drei aufeinanderfolgende Epochen nicht, wird das Training beendet.

Angepasste Verlust-Funktion Die ursprüngliche Verlustfunktion `nn.BCEWithLogitsLoss()` ist für den Datensatz dieser Arbeit nicht geeignet, da der Datensatz deutlich mehr Hintergrundpixel als Verpackungspixel enthält. Diese Klassenungleichheit kann dazu führen, dass das Modell ausschließlich Hintergrund vorhersagt und dennoch hohe Scores erreicht. Daher wurde der Focal Loss als Verlustfunktion implementiert, da dieser, wie in Abschnitt 2.3.3 erklärt, für Klassenungleichheiten am besten geeignet ist.

3.2.2 Training

Der Datensatz umfasst 12.000 synthetische Bilder, aufgeteilt in 8.000 Trainingsbilder (66,67 %), 2.000 Validierungsbilder (16,67 %) und 2.000 Testbilder (16,67 %).

Das Training des U-Net-Modells wurde mit folgenden Hyperparametern durchgeführt:

- Lernrate: $1e-4$.
- Batch-Size: 8.
- Epochen: 100.
- Optimizer: Adam.
- Verlustfunktion: Focal Loss.
- GPU: NVIDIA RTX 4060.

Das Training wurde für 100 Epochen angesetzt, aber das Early Stopping hat bereits nach der 15. Epoche (1 Stunde, 30 Minuten) das Training beendet.

Das beste Modell wurde nach 12 Epochen gespeichert und erzielte auf dem Validierungsset eine Accuracy von 99,98 % und einen Dice Score von 99,70 % bei 1.000 Batches. Es wurden 614.292.430 von 614.400.000 Pixeln korrekt klassifiziert.

Bei den Testbildern wurden ebenfalls eine Accuracy von 99,98 % und ein Dice Score von 99,71 % erzielt. Mehr zu den Metriken siehe Abschnitt 3.2.3.

Um die Robustheit des Modells zu erhöhen und die Domain Gap zwischen synthetischen und realen Bildern zu reduzieren, wurden verschiedene Augmentationen auf den Trainingsdaten angewendet:

- HorizontalFlip zur Simulation verschiedener Aufnahmepositionen.
- RandomBrightnessContrast, CLAHE und HueSaturationValue für variierende Farb- und Lichtverhältnisse.
- ISONoise, Blur und RandomShadow zur Simulation von Kamerarauschen und Schattenwurf.

Validierungs- und Testdaten wurden ohne Augmentation evaluiert, um die Leistung unter realistischen Bedingungen zu messen.

3.2.3 Evaluation

Die Modellleistung wird anhand zweier Metriken für binäre Segmentierung bewertet. Das U-Net-Modell klassifiziert jeden Pixel entweder als Verpackung (Klasse 1) oder Hintergrund (Klasse 0).

$$\text{Accuracy} = \frac{\text{tp} + \text{tn}}{\text{tp} + \text{tn} + \text{fp} + \text{fn}}$$

Dabei bezeichnet:

- tp (true positive): Verpackungspixel, die korrekt als Verpackung erkannt wurden.
- tn (true negative): Hintergrundpixel, die korrekt als Hintergrund erkannt wurden.
- fp (false positive): Hintergrundpixel, die fälschlicherweise als Verpackung erkannt wurden.
- fn (false negative): Verpackungspixel, die fälschlicherweise als Hintergrund erkannt wurden.

Accuracy misst den Anteil aller korrekt klassifizierten Pixel. Jedoch ist der Anteil an Hintergrundpixel deutlich höher als Verpackungspixel, weshalb Accuracy bei starken Klassenungleichgewichten hohe Werte erzielen kann, obwohl das Resultat eher schlecht ist.

$$\text{Dice Score} = \frac{2 \cdot \text{tp}}{\text{tp} + \text{fp} + \text{tp} + \text{fn}}$$

Der Dice Score berechnet die Überlappung zwischen vorhergesagter Maske und Ground-Truth-Maske. Die Formel zeigt, dass tn nicht einberechnet wird in den Score.

Das heißt, es wird sich hauptsächlich auf die Segmentierung der Verpackung fokussiert, was den Dice Score besser macht bei Klassenungleichgewichten. Aus diesem Grund ist der Dice Score für diese Arbeit wichtiger als Accuracy und wird auch für das Early Stopping eingesetzt.

Bei der Evaluation der Metriken wird für die binäre Klassifikation ein Schwellenwert von 0,5 auf die Sigmoid-Ausgabe angewendet. Pixel mit Wahrscheinlichkeit gleich oder über 0,5 werden als Verpackung klassifiziert, alle anderen als Hintergrund.

Nach jeder Epoche werden die Metriken auf dem Validierungsset berechnet, um den Trainingsverlauf zu überwachen. Dabei wird der Dice Score genutzt, um das beste Modell immer abzuspeichern. Nach Abschluss des Trainings erfolgt eine finale Evaluation auf dem Testset mit dem besten Modell, welches ausschließlich ungesehene, synthetische Bilder enthält. Damit soll zusätzlich geprüft werden, ob das Modell overfittet ist.

3.3 Objektposenbestimmung

Die 3D-Objektposenbestimmung ist die letzte Hauptphase der Pipeline und bestimmt die 6D-Pose der Verpackung, bestehend aus Position (X, Y, Z) und Rotation (Roll, Pitch, Yaw). Es ist bekannt, dass Roll aufgrund der flachen Lage der Verpackung 0° sein muss. Pitch ist ebenfalls 0° , da die Orientierung der Verpackungsflächen nicht relevant ist. Außerdem ist hier bekannt, dass Yaw $\pm 45^\circ$ ist.

Die 3D-Posenbestimmung erfolgt in vier Schritten:

1. Zunächst wird die Kamerapose (Rotationsmatrix R und Translationsvektor t) mithilfe der vier Orientierungsmarker auf der Greiferplatte berechnet.
2. Danach werden die 2D-Eckpunkte der Verpackung auf Basis der U-Net-Segmentierung extrahiert und ihre zugehörigen 3D-Positionen mithilfe der Verpackungsgröße bestimmt.
3. Die 2D- und 3D-Koordinaten der Eckpunkte, die Kameramatrix K und die Verzerrungskoeffizienten werden dann der OpenCV-Funktion `solvePnP` übergeben und so erhält man die 6D-Pose relativ zur Kamera.
4. Um sie in den Weltkoordinatenraum zu transformieren, wird dann R und t , welche zuvor mit den Orientierungsmarkern berechnet wurden, verwendet.

Die Genauigkeit der Posenbestimmung hängt maßgeblich von der Qualität der U-Net-Segmentierung sowie der Bestimmung der Eckpunkte ab. Die synthetischen Bilder und ihre JSON-Dateien liefern sowohl die Eingabedaten für die Kamera als auch die Ground-Truth-Werte für die Evaluation der berechneten Posen.

3.3.1 Orientierungsmarkererkennung

Das Eingabebild wird in den HSV-Farbraum konvertiert, wobei nur die unteren 40 % des Bildes analysiert werden, da sich die Orientierungsmarker ausschließlich in diesem Bereich befinden. Damit wird auch verhindert, dass gelbe Verpackungen fälschlicherweise als Orientierungsmarker erkannt werden.

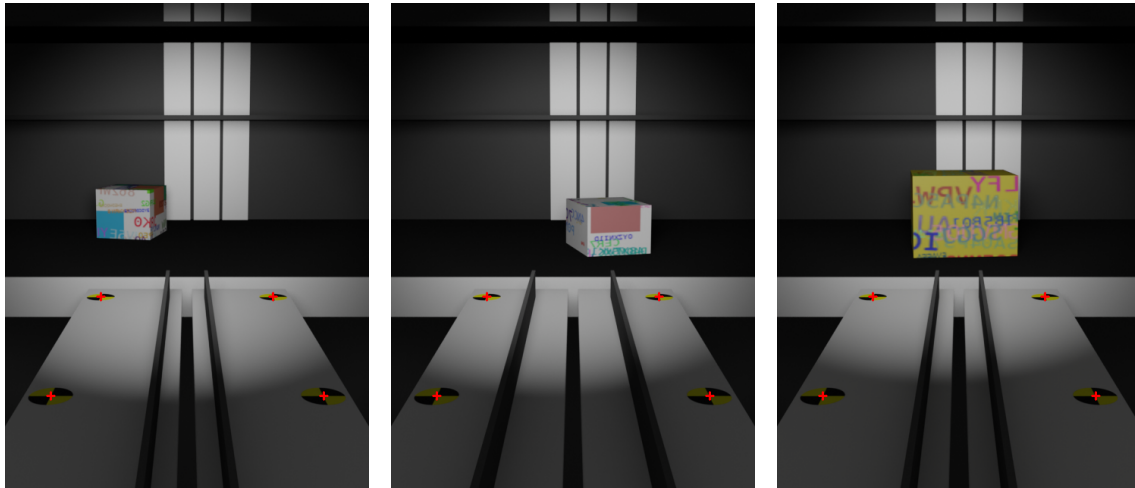


Abbildung 3.3: Gefundene Orientierungsmarker mithilfe des in Abschnitt 3.3.1 beschriebenen Skripts.

Anschließend wird der Schwellwert für die gelbe Farbe definiert, welche gefunden werden soll. Dabei wurden unterschiedliche Lichtverhältnisse in Betracht gezogen. Mit der Bibliothek OpenCV lassen sich die Orientierungsmarker detektieren: Die Funktion `findContours` findet alle gelben Bereiche und `moments` ermittelt deren Schwerpunkte.

Da im Bild mehr als vier Konturen detektiert werden können, erfolgt eine Filterung: Es werden die ersten vier Kandidaten gewählt, die jeweils mindestens 50 Pixel Abstand zu allen bereits ausgewählten Markern aufweisen. Der erste detektierte Marker dient als Startpunkt, anschließend werden die ersten Positionen genommen, die mindestens 50 Pixel Abstand zu allen bereits ausgewählten Markern aufweisen.

Die Abbildung 3.3 zeigt die detektierten Orientierungsmarker im Bild, die mit roten Kreuzen markiert sind. Die Koordinaten werden zurückgegeben.

3.3.2 2D-Eckpunkterkennung der Verpackung

Für das PnP-Verfahren werden vier 2D-Eckpunkte ermittelt, da diese stabil detektierbar sind und eine eindeutige Zuordnung zu den entsprechenden 3D-Eckpunkten ermöglichen.

Das trainierte U-Net-Modell segmentiert die gesamte Verpackungsfläche, ohne einzelne Kanten, Flächen oder Ecken zu unterscheiden. Die vier Eckpunkte lassen sich daher nur aus den Rändern der Segmentierung ableiten, wofür ein geometrischer Ansatz verwendet wird.

Zuerst werden die beiden unteren Eckpunkte (links und rechts) über einen score-basierten Ansatz ermittelt. Es wird berechnet, welcher Punkt am Rand am weitesten links/rechts ist und gleichzeitig dabei am tiefsten (höchstmöglicher Y-Wert) ist. Damit sind jetzt zwei Eckpunkte für das PnP-Verfahren bekannt.

Die beiden unteren Punkte spannen eine Gerade auf, anhand derer ein dritter unterer Punkt bestimmt wird. Dieser dritte Punkt hat die höchste Summe aus Y-Koordinate und Abstand zur Geraden. Dieser dritte Punkt wird nur für die

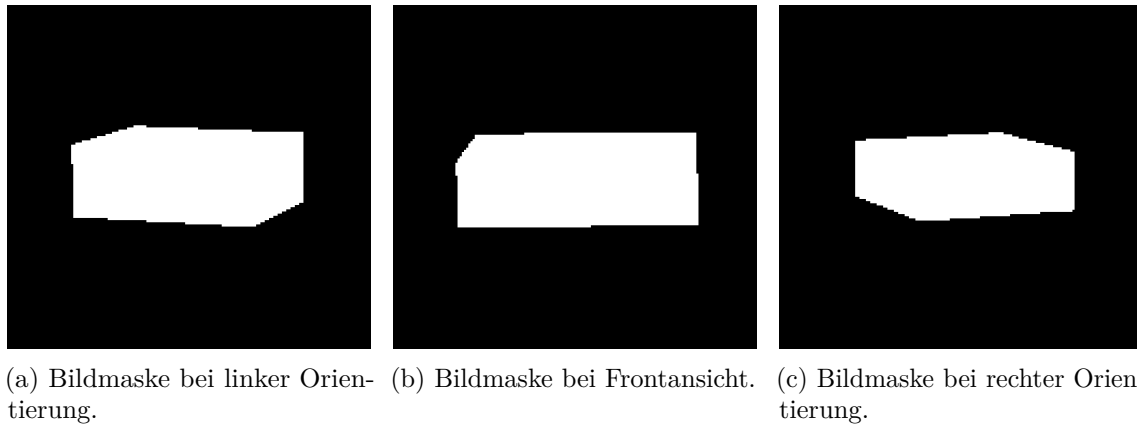


Abbildung 3.4: Beispielhafte Bildmasken der möglichen Orientierungen (500 % Zoom).

Orientierung verwendet: Wenn unter der Geraden nämlich eine Fläche ist, heißt es, dass die Verpackung gedreht ist. Geprüft wird dies, indem der Y-Unterschied gemessen wird. Wenn der maximale Y-Unterschied zwischen den drei unteren Punkten mindestens 5 Pixel beträgt, dann handelt es sich um eine Seitenansicht, sonst um eine Frontansicht. Die Unterscheidung zwischen linker und rechter Orientierung liegt dann ebenfalls am dritten Punkt. Wenn der dritte Punkt näher am linken Eckpunkt ist, dann handelt es sich um eine rechte Orientierung, da davon ausgegangen wird, dass die Tiefe der Verpackung kleiner oder gleich der Breite ist. Das Gleiche gilt für die andere Orientierung.

Die Bestimmung der oberen Eckpunkte für das PnP-Verfahren erfolgt orientierungsabhängig: Bei seitlicher Orientierung stechen die jeweils beiden äußersten linken und rechten Punkte hervor (siehe Abbildung 3.4a und 3.4c). Bei einer Frontansicht verschwindet der obere rechte Punkt der Vorderseite mit der oberen Fläche (siehe Abbildung 3.4b), weshalb dieser Ansatz nicht funktioniert. Stattdessen werden die beiden höchsten Punkte gewählt, da diese besser zu erkennen sind.

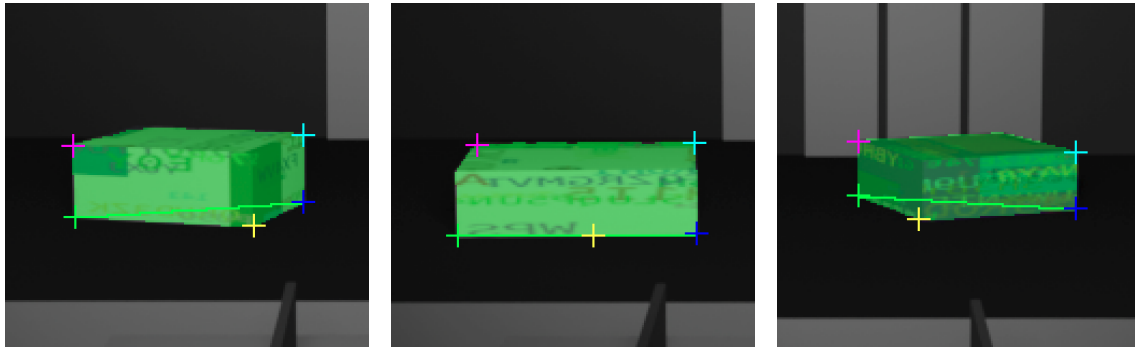
Die Abbildung 3.4 zeigt Bildmasken von Verpackungen verschiedener Orientierungen. Sie soll darstellen, was das U-Net-Modell im Idealfall erkennen soll. Die Bilder sollen zeigen, dass die Extrempunkte bei Frontansicht und Seitenansicht unterschiedlich sind.

Abbildung 3.5 zeigt die Eckpunktfindung bei den drei verschiedenen Orientierungen. Die gefundenen vier Eckpunkte liefern wichtige Informationen über die Verpackung: Wenn die Verpackung gedreht ist, liegen die beiden linken Punkte auf einer Kante und entsprechen der Höhe, dasselbe gilt für die beiden rechten Punkte. Bei einer Frontansicht liegen die beiden oberen und die beiden unteren Punkte auf einer Kante und entsprechen der Breite.

Die ermittelten Eckpunkte und die Orientierung werden später für die Positions- und Rotationsberechnung benötigt.

3.3.3 Berechnung der Kamerapose

Mithilfe der Orientierungsmarker lässt sich die Kamerapose bestimmen. Diese wird benötigt, um später die Verpackungsposition ins Weltkoordinatensystem zu transformieren.



(a) Verpackung mit einer linken Orientierung.

(b) Verpackung mit einer mittleren Orientierung.

(c) Verpackung mit einer rechten Orientierung.

Abbildung 3.5: Gefundene 2D-Eckpunkte mithilfe des in Abschnitt 3.3.2 beschriebenen Skripts (500 % Zoom).

Die Berechnung der Kamerapose erfolgt in vier Schritten:

1. Die 2D-Positionen der Orientierungsmarker im Eingabebild werden mittels des Skripts aus Abschnitt 3.3.1 detektiert.
2. Die entsprechenden 3D-Koordinaten der Orientierungsmarker werden aus der JSON-Datei gelesen.
3. Die 2D-Positionen werden im Uhrzeigersinn sortiert, sodass sie mit den 3D-Positionen übereinstimmen.
4. Die OpenCV-Funktion `cv2.solvePnP` erhält als Eingabe die 3D-Koordinaten, die zugeordneten 2D-Bildkoordinaten, die intrinsische Kameramatrix K (aus der JSON) sowie die Verzerrungskoeffizienten (in diesem Fall 0, da keine Verzerrung vorliegt).

Das Ergebnis ist die Rotationsmatrix R_{marker} und der Translationsvektor t_{marker} , welche die Pose der Orientierungsmarker relativ zur Kamera beschreiben. Für die Evaluation befindet sich die Kamera auf dem Nullpunkt, die Methode funktioniert jedoch unabhängig von der Kameraposition.

3.3.4 Berechnung der Rotation, Position und Distanz

Mithilfe des Skripts aus Abschnitt 3.3.2 wurden die 2D-Positionen der Eckpunkte der Verpackung bestimmt.

Die 3D-Positionen der Eckpunkte werden aus den bekannten Verpackungsabmessungen (Breite w , Tiefe d , Höhe h) berechnet. Je nach Orientierung (Frontansicht, linke/rechte Seitenansicht) werden unterschiedliche Eckpunkte berechnet, da sich die 2D-Positionen ebenfalls für jede Orientierung unterscheiden. Für die Frontansicht werden beispielsweise die hinteren oberen und vorderen unteren Eckpunkte verwendet, deren Koordinaten sich aus $\pm w, \pm d, \pm h$ ergeben. Bei einer Verpackungsgröße von $10 \text{ cm} \times 10 \text{ cm} \times 10 \text{ cm}$ entspricht der hintere obere Punkt beispielsweise den Koordinaten $-5 \text{ cm}, +5 \text{ cm}, +5 \text{ cm}$. Das Koordinatensystem ist an Blender angepasst, das

heißt der Ursprung befindet sich in der Verpackungsmitte, weshalb die Abmessungen halbiert werden.

Die Position und Rotation der Verpackung wird mit der OpenCV-Funktion `solvePnP` ermittelt. Als Eingabe dienen die 3D-Koordinaten der Eckpunkte für die jeweilige Orientierung, die dazugehörigen 2D-Koordinaten der Eckpunkte, die Kameramatrix K (aus der JSON) sowie die Verzerrungskoeffizienten, welche wie zuvor auf 0 gesetzt werden.

Das Ergebnis ist eine Rotationsmatrix und ein Translationsvektor, die die Pose der Verpackung relativ zur Kamera beschreiben. Um die Position und Rotation im Weltkoordinatensystem zu erhalten, werden diese mithilfe der zuvor bestimmten Marker-Pose (`R_marker`, `t_marker`) transformiert.

Für die Rotation wird die transformierte Rotationsmatrix in Eulerwinkel (YZX-Format) umgewandelt, da dies die besten Ergebnisse liefert, wobei nur Yaw extrahiert wird. Dabei wird die Yaw-Rotation auf $\pm 45^\circ$ angepasst. Wenn das Ergebnis beispielsweise -50° beträgt, wird es zu $+40^\circ$ korrigiert, da angenommen wird, dass 90° als Frontansicht genauso aussieht wie 0° . Außerdem sind die synthetischen Verpackungen in diesem Bereich rotiert.

Die Position ergibt sich direkt aus dem transformierten Translationsvektor. Die Distanz wird als euklidische Norm des Translationsvektors berechnet.

4

Ergebnisse

In diesem Kapitel werden die Ergebnisse der in Kapitel 3 beschriebenen Methoden präsentiert. Die Ergebnisse erfolgen entsprechend der drei Hauptphasen: Die Erstellung der synthetischen Bilder, das Training mit U-Net und die 3D-Objektposenbestimmung.

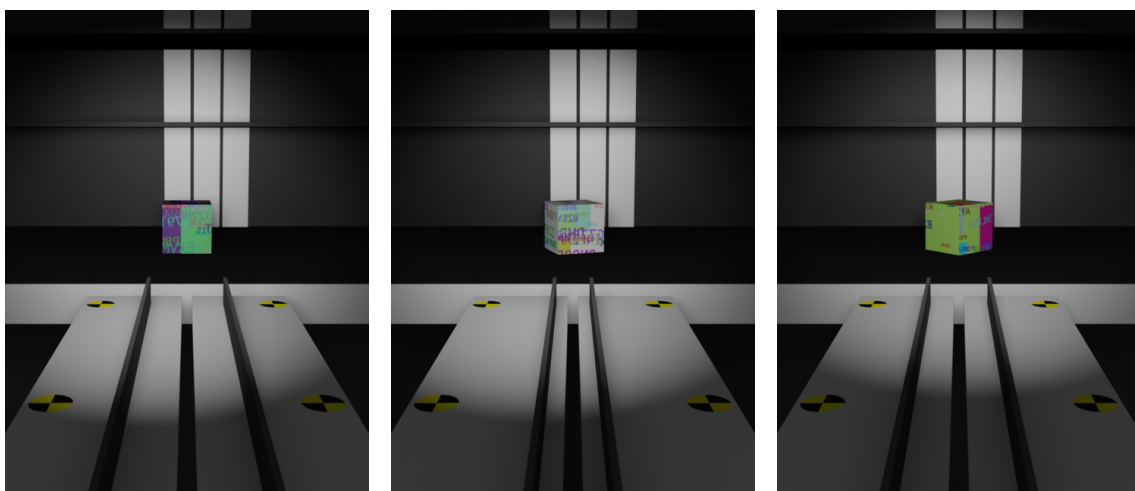
4.1 Synthetische Trainingsdaten

Wie in Abschnitt 3.1 erwähnt, wurden 12.000 synthetische Bilder generiert, die jeweils eine Bildmaske und eine JSON-Datei enthalten. Dabei sind die 12.000 Bilder aufgeteilt in 8.000 Bilder im Trainingsset, 2.000 Bilder im Validierungsset und 2.000 Bilder im Testset. Jedes Set hat jeweils 100 Bilder mit zwei verschiedenen Sonderfällen, die unten beschrieben werden.

4.1.1 Synthetische Bilder

Die folgenden Bilder wurden für eine bessere Visualisierung auf $8\text{ cm} \times 8\text{ cm} \times 8\text{ cm}$ skaliert, sofern nicht explizit anders angegeben.

Rotationsunterschiede



(a) Rotation um 0° .

(b) Rotation um 20° .

(c) Rotation um 45° .

Abbildung 4.1: Verpackungen bei unterschiedlichen Rotationswinkeln.

Die Abbildung 4.1 zeigt eine Rotation bei 0° , 20° sowie 45° . Generell liegt die Rotation der Verpackung zwischen $\pm 45^\circ$. So wurden alle möglichen Rotationsfälle abgedeckt, die die Verpackung aufweisen kann.

Positionsunterschiede

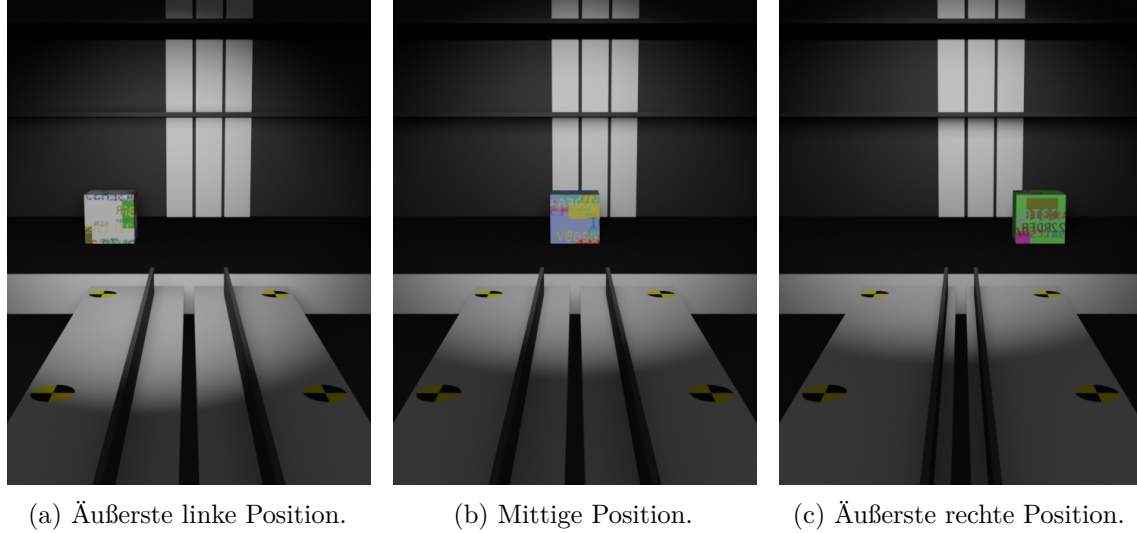


Abbildung 4.2: Verpackungspositionen entlang der X-Achse.

In Abbildung 4.2 ist die Verpackung an den äußersten eingestellten Positionen links und rechts sowie mittig dargestellt. Die Verpackung bleibt in allen Fällen vollständig im Bild sichtbar, auch bei der größten Verpackungsgröße. Der maximale Abstand zwischen der am weitesten links und rechts positionierten Verpackung beträgt vom Ursprung aus ca. 26 cm.

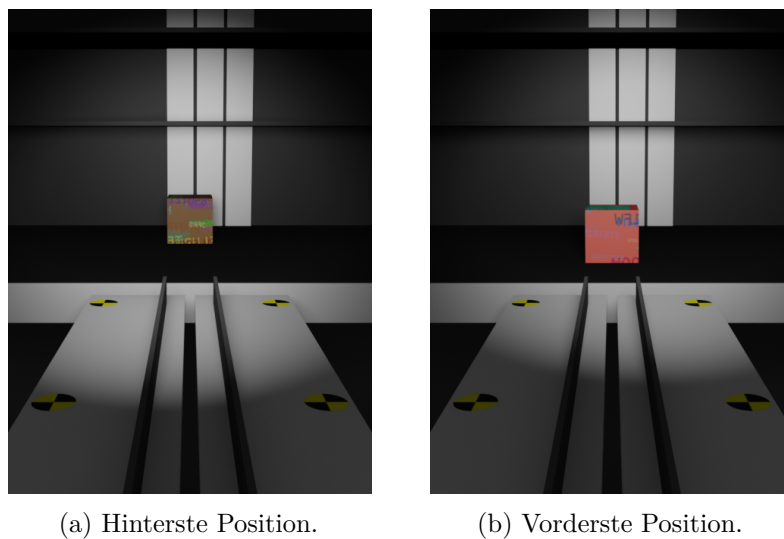
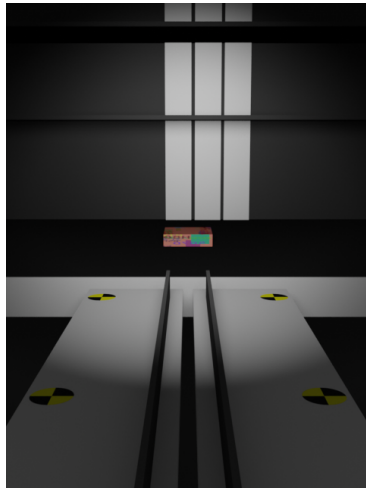


Abbildung 4.3: Verpackungspositionen entlang der Y-Achse.

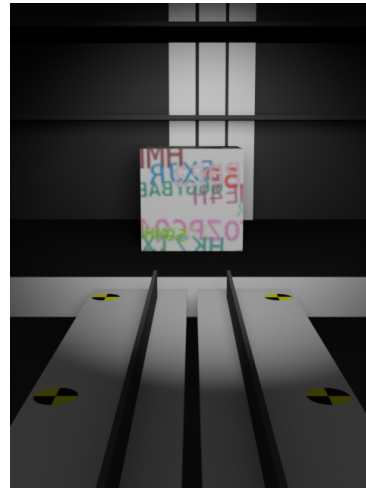
Abbildung 4.3 zeigt, dass sich die Verpackung bei größerem Y-Wert (Tiefe) weiter von der Kamera entfernt (siehe Abbildung 4.3a). Dabei wurde sichergestellt, dass die Box auch bei der größten Verpackungsgröße nicht die hintere Wand schneidet.

In Abbildung 4.3b ist die Verpackung so nah wie möglich positioniert. Auch bei der größten Verpackungsgröße bleibt die Verpackung auf dem Förderband. Der Abstand zwischen der hintersten und vordersten Verpackung beträgt 18 cm.

Visuelle Unterschiede



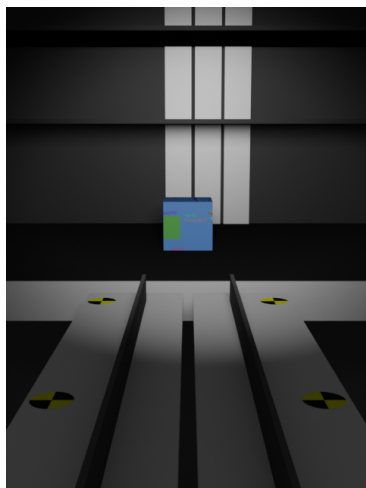
(a) Minimale Verpackungsgröße.



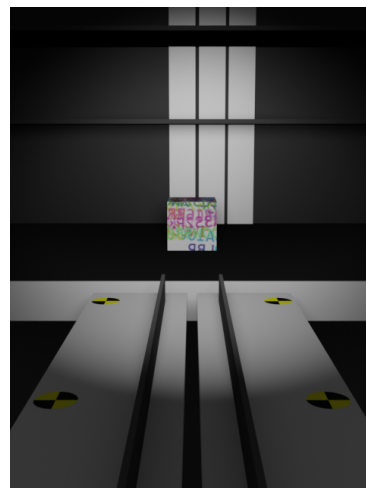
(b) Maximale Verpackungsgröße.

Abbildung 4.4: Minimale und maximale Verpackungsdimensionen.

Abbildung 4.4 zeigt die kleinstmögliche Verpackung mit 8 cm Breite und Tiefe sowie 2 cm Höhe. Die größtmögliche Verpackung im zweiten Bild entspricht 16 cm in Höhe, Breite und Tiefe. Damit werden besonders große Verpackungen abgedeckt.



(a) Minimale Schriftgröße.

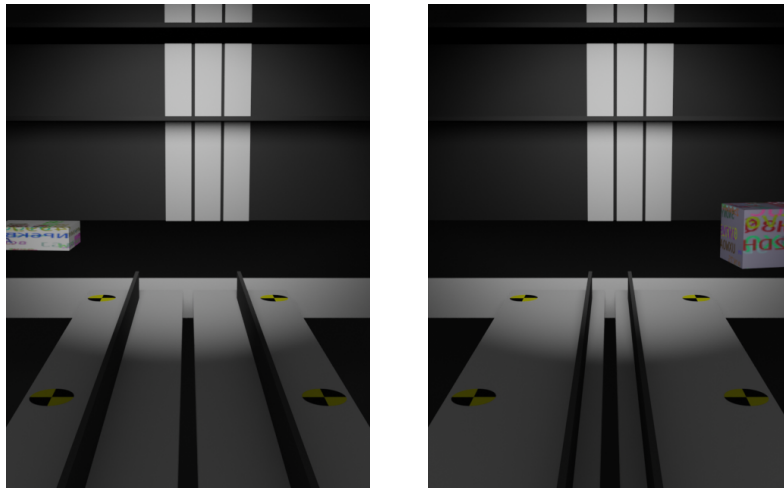


(b) Maximale Schriftgröße.

Abbildung 4.5: Minimale und maximale Textgröße.

Die Abbildung 4.5 zeigt die kleinste und größte Schriftgröße auf der Verpackung. Damit werden sowohl kleine Texte als auch Überschriften abgedeckt.

Sonderfälle



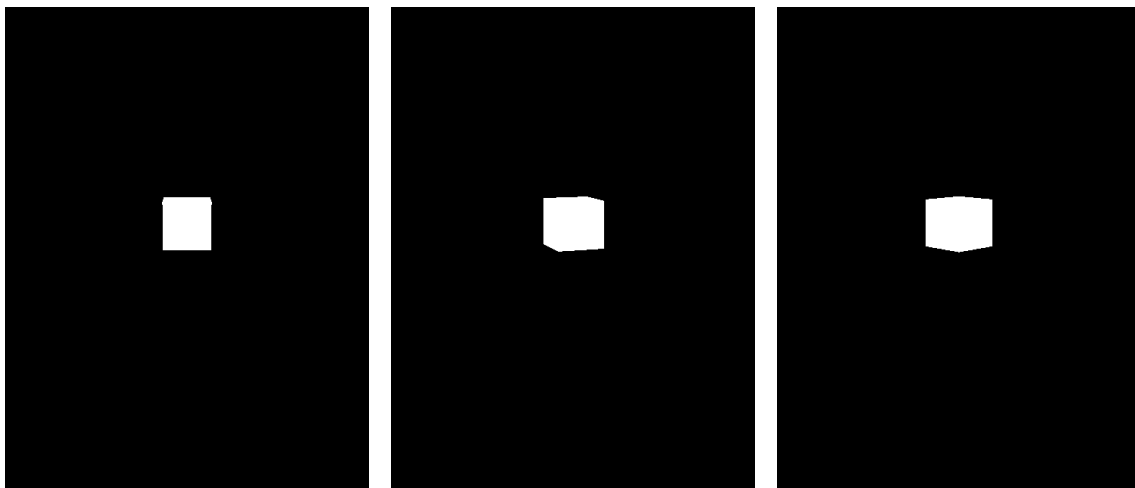
(a) Verpackung, die linken Bildrand schneidet.

(b) Verpackung, die rechten Bildrand schneidet.

Abbildung 4.6: Behandelte Sonderfälle im Training.

Zusätzlich zu den regulären Aufnahmen wurden zwei Sonderfälle generiert (Abbildung 4.6): Einmal, dass die Verpackung den linken Bildrand schneidet und einmal, dass die Verpackung den rechten Bildrand schneidet. Diese wurden manuell eingefügt und werden nicht automatisch mit der `box.py` generiert. Diese Szenarien treten in der Praxis selten auf, wurden jedoch für die Robustheit des Modells dennoch eingefügt.

4.1.2 Bildmaske



(a) Maske bei 0° Rotation.

(b) Maske bei 20° Rotation.

(c) Maske bei 45° Rotation.

Abbildung 4.7: Segmentierungsmasken der Verpackungen aus Abb. 4.1.

Die Abbildung 4.7 zeigt die binären Segmentierungsmasken der Verpackungen aus Abbildung 4.1. Weiße Pixel (Wert 1) markieren die Verpackung, schwarze Pixel (Wert 0) den Hintergrund.

4.1.3 JSON-Datei

Die JSON-Datei ist wie folgt aufgebaut:

```
{
  "image_id": "...",
  "image_resolution": {...},
  "cube": {
    "position": {...},
    "rotation": {...},
    "dimension_m": {...},
    "corners": [...]
  },
  "camera": {
    "position": {...},
    "rotation": {...},
    "focal_length_mm": {...},
    "sensor_size_mm": {...},
    "camera_matrix": {...}
  },
  "crash_symbols_position": {...},
  "crash_symbols_distances_m": {...},
  "distance_m": {...}
}
```

Es werden alle relevanten Daten gespeichert. Einige Informationen wie die Distanz zwischen den Orientierungsmarkern wurden in der finalen Methodik nicht verwendet. Einige Parameter wurden einmalig ausgegeben, wie beispielsweise die Rotationsmatrix der Kamera zur Evaluation der Orientierungsmarkererkennung.

4.2 U-Net

Das U-Net wurde mit den in Abschnitt 3.2.2 beschriebenen Parametern trainiert und erreichte in der 12. Epoche einen Dice Score von 99,70 % auf den 2.000 Testbildern.

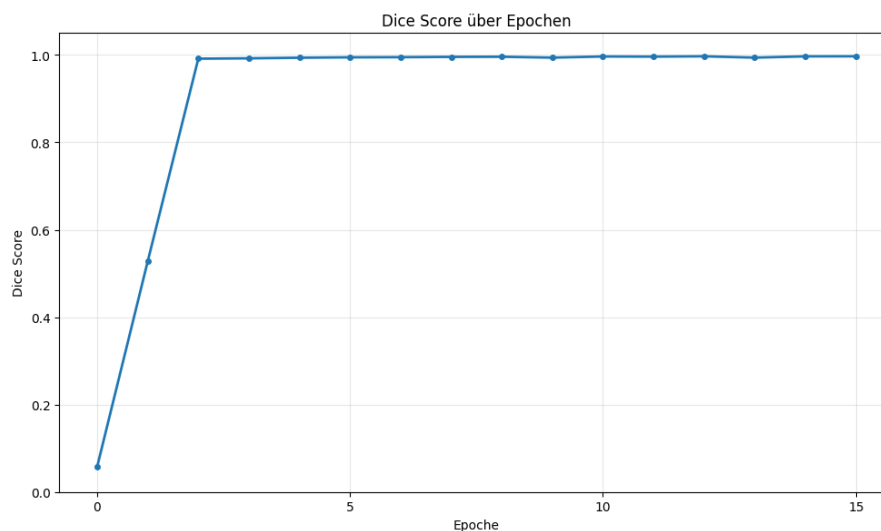


Abbildung 4.8: Verlauf des Dice Scores während des Trainings.

Abbildung 4.8 zeigt den Verlauf des Dice Scores. Der initiale Wert lag bei 5,76 %. In der ersten Epoche stieg der Dice Score auf 52,79 % und in der zweiten Epoche auf 99,16 %. In den folgenden Epochen verbesserte sich der Wert nur noch leicht bis auf 99,70 % in der 12. Epoche. Nach Epoche 15 wurde das Training durch Early Stopping beendet.

4.2.1 Erkennung von neuen synthetisch generierten Bildern

Das trainierte Modell wurde auf 650 neuen synthetischen Testbildern evaluiert, die nicht im Trainingsdatensatz enthalten waren. Auf 500 synthetischen Testbildern, bei denen die Verpackung vollständig zu sehen ist, erreichte das Modell einen Dice Score von 99,69 %. Dabei betrug die Anzahl falsch positiver Pixel auf allen Bildern 10.550 und falsch negativer Pixel 13.296.

Zusätzlich wurden 100 Bilder generiert, die angeschnittene Verpackungen enthalten. Das Modell erreichte dabei einen Dice Score von 99,40 %. Außerdem wurden 50 Bilder ohne Verpackungen generiert, um zu testen, ob das Modell falsch positive Pixel findet. Dabei wurde nichts detektiert, was zeigt, dass das Modell zwischen Förderband mit und ohne Verpackung unterscheiden kann.

4.3 Objektposenbestimmung

Für die Objektposenbestimmung wurde für jeden Zwischenschritt ein Skript geschrieben, welches die Genauigkeit der ausgegebenen Werte testet. Dabei werden jeweils der durchschnittliche Fehler und die Standardabweichung berechnet, sofern nicht explizit anders angegeben.

Die mittlere Abweichung zeigt, wie stark sich die Berechnungen von der Ground Truth unterscheiden und die Standardabweichung zeigt, wie diese Abweichung gestreut ist. Eine geringe Streuung bedeutet, dass die Ergebnisse nah am Mittelwert liegen. Das deutet darauf hin, dass die Methode robuste Ergebnisse liefert.

Die Evaluation wurde auf 500 neu generierten synthetischen Bildern durchgeführt, bei denen die Verpackung komplett sichtbar ist. Dieselben Bilder wurden auch für die Evaluation des U-Net verwendet.

4.3.1 Orientierungsmarkererkennung

Um die Genauigkeit der Orientierungsmarkererkennung zu bestimmen, wurde ein Skript geschrieben, welches die Ground Truth 2D-Positionen der Orientierungsmarker hat und diese mit den detektierten Markern aus Abschnitt 3.3.1 vergleicht.

Die Genauigkeit dieser Erkennung ist entscheidend für die Bestimmung der Rotationsmatrix und des Translationsvektors, welche die Grundlage für alle nachfolgenden Berechnungen bilden.

Die Evaluation erfolgte sowohl für jeden Marker einzeln als auch für alle vier Marker gemeinsam.

	Mittlere Abweichung (Pixel)	Standardabweichung (Pixel)
Oberer, linker Marker	0,28	0,01
Oberer, rechter Marker	0,47	0,03
Unterer, linker Marker	0,62	0,02
Unterer, rechter Marker	0,49	0,02

Tabelle 4.1: Evaluationsergebnisse der Orientierungsmarkererkennung.

Tabelle 4.1 zeigt, dass die Abweichungen der Erkennung pro Orientierungsmarker im Durchschnitt unter einem Pixel liegen. Der untere linke Marker schneidet mit einem Fehler von 0,62 px am schlechtesten ab.

Wenn die schlechteste Erkennung mit der besten Erkennung verglichen wird, so hat die schlechteste Erkennung im Durchschnitt 0,5 px Fehler, während das beste erkannte Bild im Durchschnitt 0,4 px Fehler aufweist. Damit ist der Unterschied zwischen Best und Worst Case extrem gering.

Die korrekte Rotationsmatrix R sowie Translationsvektor t der Kamera ist:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0,1736 & -0,9848 \\ 0 & 0,9848 & 0,1736 \end{bmatrix}$$

$$t = \begin{bmatrix} 0,0 \\ 0,0 \\ 0,0 \end{bmatrix}$$

Das Ergebnis von R und t bei der schlechtesten Erkennung ist:

$$R = \begin{bmatrix} 0,9999 & -0,0008863 & 0,0002583 \\ 0,0001007 & -0,1734 & -0,9848 \\ 0,0009177 & 0,9848 & -0,1734 \end{bmatrix}$$

$$t = \begin{bmatrix} 0,0009798 \\ -0,0000484 \\ -0,0005894 \end{bmatrix}$$

Die durchschnittliche Abweichung der Werte in der Rotationsmatrix beträgt 0,077 und die des Translationsvektors 0,0005392. Anzumerken ist, dass die Ground-Truth-Rotationsmatrix an Position (2,2) und (3,3) den Wert -0,1734 aufweist, während die berechnete Matrix 0,1734 liefert.

4.3.2 2D-Eckpunkterkennung der Verpackung

Da die Verpackung quaderförmig ist, besitzt sie acht Eckpunkte. Je nach erkannter Orientierung sind unterschiedliche Eckpunkte sichtbar. Für die Evaluation wurde zunächst die vom Skript (Abschnitt 3.3.2) erkannte Orientierung verwendet, um die entsprechenden Ground-Truth-Eckpunkte aus der JSON-Datei auszuwählen. Diese

wurden dann mit den detektierten Eckpunkten verglichen. Entscheidend ist, ob das Skript für die gewählte Orientierung die korrekten Eckpunkte ermittelt hat.

Die Evaluation erfolgte sowohl für jeden Eckpunkt einzeln als auch für alle vier Punkte gemeinsam.

	Mittlere Abweichung (Pixel)	Standardabweichung (Pixel)
Oberer, linker Punkt	1,81	6,8
Oberer, rechter Punkt	2,34	6,47
Unterer, linker Punkt	2,36	6,96
Unterer, rechter Punkt	2,46	6,41

Tabelle 4.2: Evaluationsergebnisse der Eckpunkterkennung.

Tabelle 4.2 zeigt, dass im Durchschnitt jeder Eckpunkt ca. 2 Pixel daneben liegt. Der untere rechte Eckpunkt schneidet mit einem durchschnittlichen Fehler von 2,46 px am schlechtesten ab, der obere linke mit 1,81 px am besten. Die Standardabweichung liegt bei allen Eckpunkten über 6 px.

4.3.3 Berechnung der Verpackungsposition und -distanz

Für die Berechnung der Position und Distanz wurde ein Skript geschrieben, welches die Ground-Truth-Position aus der JSON entnimmt und mit den Berechnungen aus Abschnitt 3.3.4 vergleicht. Da die Positionsfehler in einem dreidimensionalen Koordinatensystem auftreten, müssen die Absolutbeträge der Abweichungen verwendet werden. Andernfalls würden sich positive und negative Fehler gegenseitig ausgleichen.

Die Evaluation der Position erfolgte sowohl für jede Achse einzeln als auch für alle drei Achsen gemeinsam.

	Mittlere Abweichung (Meter)	Standardabweichung (Meter)
X	0,001	0,002
Y	0,021	0,025
Z	0,003	0,004

Tabelle 4.3: Evaluationsergebnisse der Position (X,Y,Z).

Tabelle 4.3 zeigt, dass die X-Position (seitliche Position) mit einem durchschnittlichen Fehler von 1 mm am besten abschneidet und die Y-Position (Tiefe) mit 2 cm am schlechtesten.

Die durchschnittliche Abweichung bei der Distanz beträgt 2,1 cm bei einer Standardabweichung von 2,6 cm.

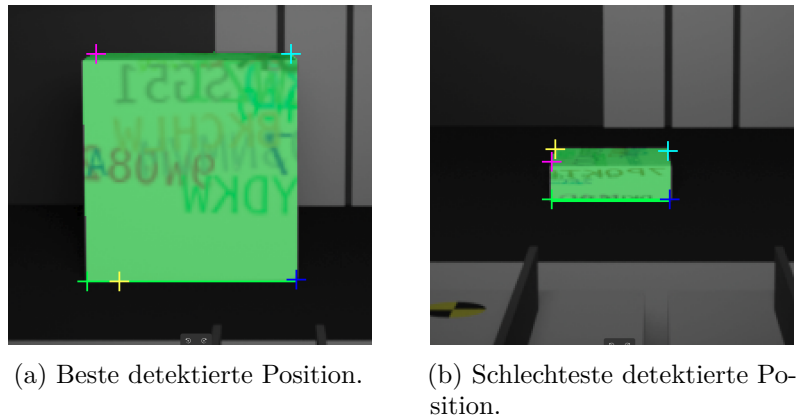


Abbildung 4.9: Best und Worst Case bei der Positionsbestimmung (500 % Zoom).

Abbildung 4.9 zeigt die beste und schlechteste Erkennung. Bei der besten Erkennung wurde korrekt eine Frontansicht geschätzt und die entsprechenden vorderen und hinteren Punkte korrekt gewählt. Der Distanzfehler beträgt hier 1 mm (0,1 %). Bei der schlechtesten Erkennung wurde fälschlicherweise eine rechte Orientierung gewählt, dazu wurden die Punkte falsch zugeordnet. Der Distanzfehler liegt hier bei 24,7 cm (28,1 %).

4.3.4 Berechnung der Verpackungsrotation

Für die Evaluation der Rotation wurde der minimale Winkelfehler statt des absoluten Fehlers berechnet. Das heißt, es wird berücksichtigt, dass eine 90° gedrehte Verpackung als Frontansicht klassifiziert wird und damit auch wie eine 0° Rotation wirkt. Die Gründe dafür werden in Abschnitt 5.3.4 diskutiert.

	Mittlere Abweichung (Grad)	Standardabweichung (Grad)
Yaw	3,02	5,52

Tabelle 4.4: Evaluationsergebnis der Rotation (Yaw).

Tabelle 4.4 zeigt einen durchschnittlichen Rotationsfehler von 3° und eine Standardabweichung von fast 6°.

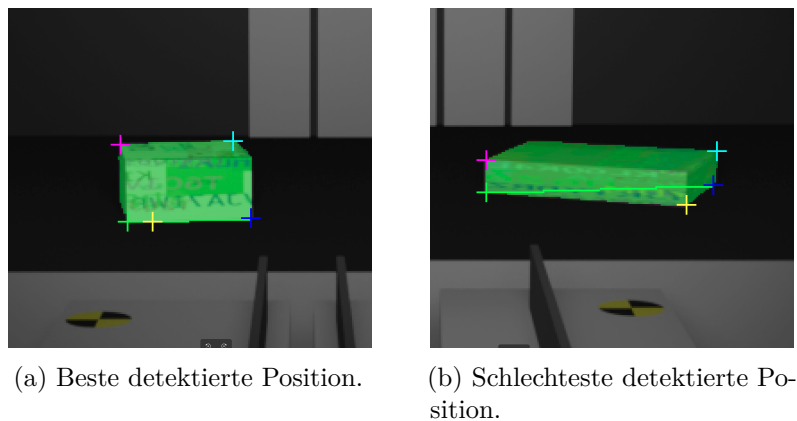


Abbildung 4.10: Best und Worst Case bei der Rotationsbestimmung (500 % Zoom).

Die beste Erkennung in Abbildung 4.10 weist einen Rotationsfehler von unter $0,001^\circ$ auf, die schlechteste einen Fehler von ca. 43° . Beim schlechtesten Ergebnis wurden $24,6^\circ$ berechnet, die Lösung wäre $-21,8^\circ$.

5

Diskussion

In diesem Kapitel werden die Ergebnisse aus Kapitel 4 analysiert und diskutiert.

5.1 Synthetische Trainingsdaten

5.1.1 Erstellung der synthetischen Bilder

Die echte Szene wurde vereinfacht im virtuellen Raum (Blender) nachgebaut. Sie soll die wesentlichen Merkmale der realen Szene enthalten, damit das trainierte Modell auch auf der echten Szene angewendet werden kann.

Szenenaufbau Der Abstand von Kamera zur Verpackung beträgt ungefähr einen Meter und die Auflösung liegt bei 480×640 , wie es ebenfalls in der realen Szene ist. Informationen zur Kamera, außer dass sie keine Tiefeninformationen enthält, gab es keine. Daher ist nicht bekannt, ob die Brennweite der Kamera mit der echten Kamera übereinstimmt. Dies kann jedoch leicht über Blender korrigiert werden. Generell können viele Parameter mit Blender und BlenderProc angepasst werden. Damit können unterschiedliche Kamera-Setups im virtuellen Raum getestet werden, ohne echte Kameras zu kaufen. Das Erstellen der Szene war trotz wenig Erfahrung in Blender gut umsetzbar.

Domain Gap Die synthetischen Verpackungen sehen im Vergleich zu echten Medikamentenverpackungen weniger realistisch aus.

Die Texte sind zufällig verteilt und können schräg auf der Fläche liegen. Die farblichen Rechtecke sollen farbliche Übergänge und Logos darstellen, sind jedoch zumindest für Logos zu simpel gehalten. Echte Verpackungen wirken strukturierter.

Auch die Greiferarme und die Umgebung sind weniger detailliert. Das Förderband ist im synthetischen Bild einfach schwarz, ohne Texturen oder Verschmutzungen, die bei echten Förderbändern auftreten können. Generell fehlen realistische Texturen und Materialeigenschaften. So können die synthetischen Verpackungen kein Licht

reflektieren, was bei einigen Medikamentenverpackungen der Fall ist. Auch reflektieren die Metallstäbe im Hintergrund kein Licht.

Verzeichnung und Kamerarauschen fehlen in den synthetischen Bildern. Zu Beginn wurden Störgeräusche eingebaut, wodurch die Verpackungen jedoch matt wirkten, was nicht Medikamentenverpackungen entspricht. Daher wurde dies wieder entfernt. Einige Parameter wie Rauschen wurden bewusst nicht in die synthetischen Bilder eingefügt, sondern als Augmentation ins Training integriert. So werden zufällige Schatten in unterschiedlicher Stärke platziert und ISO Noise angewendet. Diese Augmentationen werden nur mit geringer Wahrscheinlichkeit aktiviert und bieten verschiedene Variationen, die das Modell robuster machen.

Dadurch mussten nicht alle Parameter bei der Bildgenerierung variiert werden, was den Prozess erleichtert hat.

Behandelte Sonderfälle Es wurden Bilder eingefügt, bei denen die Verpackung den linken oder rechten Bildrand schneidet. Dabei wurde die X-Position auf einen festen Wert gesetzt, weshalb alle diese Verpackungen an derselben Stelle liegen. Hier hätten mehr Variationen eingefügt werden können, etwa unterschiedliche Abstufungen der Schneidung von minimal bis stark. Diese und weitere Sonderfälle können mit BlenderProc einfach hinzugefügt werden.

Unbehandelte Sonderfälle Die betrachteten Verpackungen sollten einen Großteil typischer Medikamentenverpackungen abdecken. Folgende Fälle wurden nicht berücksichtigt:

- Zylinderförmige Verpackungen.
- Ganz kleine Verpackungen wie Nasensprays.
- Mehrere Verpackungen gleichzeitig auf dem Förderband (potenzielle Verdeckungen).
- Geöffnete, verschmutzte oder beschädigte Verpackungen.

Der vorgestellte Ansatz würde mit diesen Sonderfällen voraussichtlich Schwierigkeiten haben.

Die Erstellung synthetischer Bilder mit Blender und BlenderProc ist eine gute Methode, um die Szene vereinfacht nachzubilden. Mit mehr Erfahrung in Blender können weitere Details eingefügt werden, beispielsweise verschiedene Materialeigenschaften, um die Domain Gap weiter zu reduzieren.

5.1.2 Erstellung der Segmentierungsmasken

Die Erstellung der Segmentierungsmasken war mit BlenderProc simpel. Durch die Automatisierung wurde viel Zeit gespart, da nicht jeder Pixel manuell annotiert werden musste. Außerdem entstehen so keine menschlichen Fehler, die das Training negativ beeinflussen könnten.

Binäre Segmentierung Durch die binäre Maske gehen ein paar Informationen verloren: Flächen, Kanten und Eckpunkte der Verpackung werden nicht unterschieden. Dies erschwert die Eckpunktfindung sehr, da nur mit den äußeren Kanten der segmentierten Maske gearbeitet werden kann. Dennoch wurde entschieden, die gesamte Verpackung zu segmentieren, da Kantenpixel allein nur einen minimalen Anteil ausmachen würden und so das Training erschwert wird. Zusätzlich würde bei einer reinen Kantensegmentierung der innere Bereich der Verpackung als Hintergrund klassifiziert werden, da bei binärer Segmentierung nur zwischen Objekt und Hintergrund unterschieden wird.

Limitationen Wie in Abschnitt 3.3.2 erwähnt, variieren die Extrempunkte je nach Rotation der Verpackung. Bei 0° sind andere Eckpunkte sichtbar als bei 45° . Bei der binären Maske wird nicht klar, welcher erkannte Punkt welchem 3D-Eckpunkt der Verpackung entspricht, weshalb diese Information zusätzlich durch Orientierungserkennung abgeleitet wird.

Es wurden alternative Ansätze getestet, beispielsweise Multi-Class-Segmentierung, bei der jede Fläche eine andere Klasse erhält und so die Erkennung der Eckpunkte erleichtert werden könnte. Jedoch wurde sich für den jetzigen Ansatz entschieden, da dieser weniger komplex ist und im echten Szenario jede Fläche gleich behandelt wird.

Der Ansatz mit der binären Segmentierungsmaske reicht für diese Bachelorarbeit, um zu zeigen, wie eine 6D-Posenbestimmung mit nur RGB-Daten möglich ist.

5.1.3 Erstellung der JSON

Die JSON-Dateien speichern die Ground-Truth-Informationen jeder Szene, darunter Informationen über die Verpackung, Kamera und Orientierungsmarker.

JSON-Auswahl Es wurde sich für das JSON-Format entschieden, da diese einfach mit Python zu erstellen und gut lesbar sind. Auch die BOP Challenge nutzt JSON-Dateien, um Informationen über die Ground Truth auszugeben, jedoch nutzt BOP mehrere verschiedene JSON-Dateien für unterschiedliche Zwecke. Für diese Arbeit war diese Unterteilung nicht notwendig, da insgesamt nur wenig Daten aufgezeichnet wurden.

Schwierigkeiten bei den Metadaten Die JSON-Dateien dienen sowohl als Ground Truth als auch als Input für die 6D-Posenschätzung. Fehlerhafte Inhalte beeinflussen entsprechend die Ergebnisse. Beispielsweise war die Verpackungsgröße anfangs falsch berechnet worden. Dies fiel erst spät auf, da zunächst ein Fehler in der Methodik vermutet wurde. Bei synthetischen Szenen lässt sich die Korrektheit solcher Werte nur schwer visuell überprüfen. Die Dokumentation von BlenderProc und Blender war hierbei hilfreich. Aus diesem Grund sollte sichergestellt werden, dass alle Werte in der JSON korrekt sind.

Ein weiterer Punkt ist, dass der Inhalt der JSON-Datei an das Blender-Koordinatensystem angepasst ist. Werden Ergebnisse mit anderen Koordinatensystemen verglichen, wie in dieser Arbeit mit OpenCV, kann dies zu falschen Interpretationen führen. Deswegen ist eine korrekte Umwandlung in das jeweilige

Koordinatensystem wichtig, damit korrekte Ergebnisse nicht falsch interpretiert werden.

Bei korrekten Werten eignen sich die JSON-Dateien gut zur Informationsübertragung sowie Evaluation und können auch in Zukunft verwendet werden.

5.2 U-Net

Architekturwahl Die Wahl fiel auf U-Net, da es eine einfache und gut dokumentierte Architektur ist. Zudem ist U-Net für semantische Segmentierung konzipiert, was für diese Arbeit wichtig ist, da die Verpackung pixelgenau erkannt werden muss. Das Modell wird schnell geladen und ermöglicht eine schnelle Inferenz. Andere CNN-Architekturen wurden nicht evaluiert, da U-Net sich früh als ausreichend herausstellte.

Trainingsdurchlauf Es wurden mehrere Trainingsläufe durchgeführt, wobei der Dice Score auf dem Testset zwischen 99,7 % und 99,83 % lag. Der beste Wert von 99,83 % wurde mit einer Patience von 5 erreicht, das heißt das Training wurde abgebrochen, wenn sich der Dice Score über fünf aufeinanderfolgende Epochen nicht verbessert hatte. Dies hatte jedoch zur Folge, dass das Training über 7 Stunden dauerte, da auch minimale Verbesserungen die Patience zurücksetzten.

Da wiederholt neue Bilder generiert wurden und das Training entsprechend oft wiederholt werden musste, wurde eine Patience von 3 gewählt. Dies reduzierte die Trainingsdauer auf 2 Stunden und lieferte dennoch fast genauso gute Werte.

Die voreingestellten Hyperparameter des genutzten Skripts von Persson [16] wurden an die Hardware angepasst. So erlaubte die für diese Bachelorarbeit genutzte GPU mit 8 GB VRAM eine maximale Batch Size von 8. Die 12.000 Bilder waren ausreichend. Wie bereits in Kapitel 2 erwähnt, reicht für U-Net eine geringe Datenmenge, um gute Ergebnisse zu erzielen.

Ergebnisse Nach dem Training lag der Dice Score auf Validierungs- und Testset bei 99,8 %, was zeigt, dass kein Overfitting vorliegt. Die in Abschnitt 3.2.2 beschriebenen Augmentationen trugen zusätzlich zur Robustheit des Modells bei.

Der Dice Score wurde zusätzlich auf 500 neu generierten Evaluationsbildern berechnet und erreichte 99,69 %. Das bedeutet, dass 0,31 % der Pixel falsch klassifiziert wurden.

Das Ergebnis zeigt, dass häufiger Hintergrundpixel als Verpackung klassifiziert wurden (13.891 Pixel) als umgekehrt (9.514 Pixel). Bei der Untersuchung aller 500 Bilder wurde deutlich, dass die Fehler an den Kanten auftreten. Gerade treppenförmige Kanten sind schwierig, da die Unterscheidung zwischen Verpackung und Hintergrund nicht eindeutig bestimmbar ist.

Eine weitere Beobachtung ist, dass sich falsch negative Pixel häufig an den Eckpunkten befinden. Ein Grund dafür könnte sein, dass ein Eckpunkt viele Hintergrundpixel um sich herum hat, wodurch das Modell die Wahrscheinlichkeit für Hintergrund höher einschätzt.

Die falsch positiven Pixel an den Kanten sind nicht besonders kritisch für die 6D-Posenschätzung, doch die fehlenden Eckpunkte können ein Problem darstellen.

Auswirkung auf die Eckpunktfindung Die leichten Segmentierungsfehler an den Kanten haben keine nennenswerten Auswirkungen auf die Eckpunktfindung, da Eckpunkte als Extrempunkte der Kontur robust erkannt werden. Dass einzelne Eckpunktpixel nicht erkannt werden, ist nicht kritisch, da meist benachbarte Pixel korrekt segmentiert wurden. Bei einem Kameraabstand von einem Meter ist ein einzelner Pixelfehler vernachlässigbar. Problematisch wäre es, wenn ein Eckpunkt großflächig nicht erkannt würde. Dies trat in den Evaluationsbildern jedoch sehr selten auf.

Das U-Net-Modell erfüllt seinen Zweck. Das Modell hat kurze Ladezeiten und arbeitet effizient. Die segmentierten Masken sind ausreichend präzise, um später die Eckpunkte mit dem geometrischen Ansatz zu bestimmen. Getestet wurde ausschließlich auf synthetischen Bildern. Eine Evaluation auf echten Aufnahmen oder detaillierten 3D-Modellen von Medikamentenverpackungen wäre wünschenswert gewesen, wurde jedoch aus Zeitmangel nicht umgesetzt und sollte bei einer Fortführung des Projekts nachgeholt werden.

5.3 Objektposenbestimmung

5.3.1 Orientierungsmarkererkennung

Die gelb-schwarzen Orientierungsmarker eignen sich aufgrund ihres hohen Farbkontrasts gut für die automatische Erkennung. Zudem zeigt sich die Methode robust gegenüber unterschiedlichen Beleuchtungsbedingungen. Alternative Marker wie ArUco-Codes könnten ebenfalls verwendet werden, jedoch erfüllen die vorhandenen Crash-Symbole den Zweck ausreichend.

Ergebnisse Die beiden oberen Marker werden im Durchschnitt am genauesten erkannt. Ein Grund dafür könnte ihre konstant gute Beleuchtung sein. Der untere linke Marker schneidet mit einem durchschnittlichen Fehler von 0,62 px am schlechtesten ab. Im Vergleich zum rechten unteren Marker wird der linke untere Marker seltener und weniger stark beleuchtet, was die höhere Abweichung erklären kann.

Limitationen Da die Methode genau vier Marker erwartet, führen bereits geringe Verdeckungen einzelner Marker zu Fehlern. Außerdem beschränkt sich die Suche auf die unteren 40 % des Bildes, da sich dort die Greiferplatte mit den Markern befindet. Eine vollständige Sichtbarkeit aller Marker muss daher gewährleistet sein.

Berechnung der Kamerapose Die hohe Erkennungsgenauigkeit der Orientierungsmarker spiegelt sich auch in der berechneten Rotationsmatrix und dem Translationsvektor wider, da diese nahe an der Ground-Truth-Lösung liegen. Das einzige Problem bei der berechneten Rotationsmatrix sind die vertauschten Vorzeichen in den Zellen (2,2) und (3,3). Dies könnte auf einen Fehler bei der Zuordnung der Orientierungsmarker hindeuten. Die genaue Ursache konnte nicht ermittelt werden, aber sollte in zukünftigen Arbeiten untersucht werden, da dies ein potenzieller Grund für die unpräzisen Ergebnisse bei der Rotation sein könnte.

Insgesamt zeigt Abschnitt 4.3.1, dass die Methode auf synthetischen Bildern sehr gut funktioniert. Der durchschnittliche Fehler pro Marker liegt unter einem Pixel und die Kamerapose wird entsprechend präzise berechnet, mit Ausnahme der vertauschten Vorzeichen in den Zellen (2,2) und (3,3).

5.3.2 2D-Eckpunkterkennung der Verpackung

Die Genauigkeit der 2D-Eckpunkte ist entscheidend für die 6D-Posenschätzung mittels PnP-Verfahren.

Ergebnisse der Eckpunktfindung Der durchschnittliche Fehler von 2,24 px (Abschnitt 4.3.2) erscheint zunächst akzeptabel. Die hohe Standardabweichung von 7,25 px zeigt jedoch, dass die Methode nicht robust ist. Einige Eckpunkte unterscheiden sich stark von dem Ground Truth. Im Vergleich zur Orientierungsmarkererkennung, welche im Durchschnitt eine Standardabweichung von 0,02 px hat, ist dies deutlich unzuverlässiger. Ein Grund dafür ist, dass der Score-basierte Ansatz mit seinen festen Gewichtungen nicht bei allen Perspektiven den optimalen Eckpunkt findet. Auch können Fehler in der Segmentierung ein Grund dafür sein.

Limitationen Durch die Eckpunktfindung wird nicht klar, welche Seite Tiefe und welche Breite ist. Dies kann bei der 3D-Bounding-Box-Modellierung dazu führen, dass die Verpackung falsch gedreht ist. Bei einem Würfel wäre dies kein Problem, aber andere Quader sollten berücksichtigt werden. Es könnte in den Algorithmus eingebaut werden, dass die Längen verglichen werden. Dabei können jedoch Probleme aufgrund der Perspektive entstehen.

Qualitative Orientierungsbestimmung Die Orientierung (front, left, right) wird anhand eines dritten Punktes ermittelt. Bei frontaler Ansicht funktioniert dies zuverlässig. Bei seitlichen Perspektiven kann die Verpackung jedoch rotiert erscheinen, obwohl sie es nicht ist, was zu einer falschen Orientierungszuweisung führt. Auch ist die Unterscheidung zwischen front und rotiert mit einem festen Schwellwert von 5 px nicht für alle Verpackungsgrößen und Perspektiven angebracht. Die Orientierung dient der korrekten Zuordnung von 2D- zu 3D-Eckpunkten. Dies klappt nicht immer, da bei den Punkten nicht bekannt ist, welche Seite Tiefe und welche Breite ist. Für die Positionsfindung reicht es jedoch, da hier nach dem Zentrum der Verpackung gesucht wird. Für die Rotation könnte dies ein Problem darstellen.

Alternativer Ansatz In einem Paper von Chamzas et al. wird cMinMax vorgestellt, welcher ebenfalls die Eckpunkte eines konvexen Polygons findet [3]. Dabei wird nach Extrempunkten gesucht, wie in der Methode aus dieser Arbeit, doch zusätzlich wird das Bild schrittweise rotiert und nach jeder Rotation werden erneut die Extrempunkte bestimmt. Durch diese wiederholte Rotation werden alle Eckpunkte gefunden, auch jene, die ohne Rotation keine Extremwerte aufweisen [3]. Diese Methode könnte in Zukunft in Erwägung gezogen werden.

Die Eckpunkterkennung liefert mittelmäßige Ergebnisse und erweist sich aufgrund der hohen Standardabweichung als nicht robust. Für eine zuverlässigere 6D-Posenschätzung

sollte die Methode optimiert werden, beispielsweise durch den cMinMax-Ansatz. Die Orientierungsbestimmung reicht für die Zuweisung der 2D- zu 3D-Eckpunkten aus, hat aber dennoch Verbesserungspotenzial.

5.3.3 Berechnung der Verpackungsposition und -distanz

Ergebnisse Die Position und Distanz werden trotz der Pixelfehler gut gefunden. Die Y-Achse (Tiefe) und die dazugehörige Distanzberechnung weisen mit 2 cm den höchsten durchschnittlichen Fehler auf. Dies war zu erwarten, da bei einem RGB-Input keine Tiefeninformationen vorhanden sind und diese berechnet werden müssen. Im Abschnitt 4.3.2 ist zu sehen, dass die Eckpunkte im Durchschnitt um 2 Pixel daneben liegen. Diese 2 Pixel sorgen dafür, dass die Berechnung der Tiefe ungenauer wird. Die Standardabweichung von 2,6 cm zeigt, dass das PnP-Verfahren nicht robust genug die Tiefe bestimmen kann. Doch die Schwankungen sind nur wenige Zentimeter. Mit einer verbesserten Eckpunktfindung kann dies auch verbessert werden.

Die X-Achse liefert mit einem durchschnittlichen Fehler von 1 mm das beste Ergebnis, die Z-Achse ist mit 3 mm ebenfalls präzise. Die Standardabweichungen sind bei beiden sehr gering, was zeigt, dass das PnP-Verfahren hier robustere Ergebnisse liefert. Das liegt daran, dass X- und Z-Position direkt aus einem RGB-Bild abgeleitet werden können.

Das Best-Case-Bild aus Abbildung 4.9a zeigt, dass die oberen beiden Punkte perfekt getroffen wurden. Der untere linke Punkt hat den größten Fehler, dennoch wurden Position und Distanz sehr gut ermittelt. Beim Worst-Case-Bild aus Abbildung 4.9b wurden für die geschätzte Orientierung die falschen Punkte gewählt. Komisch ist hierbei, dass der gelbe Punkt, welcher den dritten Punkt darstellt, der die tiefste Stelle sucht, sich ganz oben befindet. Es wurde eine rechte Orientierung geschätzt und dementsprechend die falschen Eckpunkte gewählt.

Industrielle Anwendung Wie bereits in der Einleitung erwähnt, ist die Z-Achse weniger relevant, da in der realen Anwendung die Höhe bekannt ist. Der Y-Fehler von ca. 2 cm könnte jedoch problematisch sein: Wird die Distanz unterschätzt, erreicht der Greifarm die Verpackung nicht. Eine Überschätzung wäre weniger kritisch. Die X-Position ist für die industrielle Anwendung ebenfalls sehr relevant und liefert mit 1 mm Fehler präzise Werte. Damit lässt sich zuverlässig die Verpackung auf dem Förderband finden.

Das PnP-Verfahren liefert insgesamt gute Ergebnisse, könnte aber durch mehr Eckpunkte robuster werden, um die Tiefe präziser zu bestimmen. Die Eckpunktfindung beeinflusst das Ergebnis stark, da falsche Eckpunkte zu schlechten Ergebnissen mit dem PnP-Verfahren führen. Die Eckpunktfindung sollte in Zukunft verbessert werden, da sie derzeit nicht robust genug für den industriellen Einsatz ist. Auch könnte die bekannte Information zur Höhe (Z-Achse) genutzt werden als Randbedingung.

Ein weiteres Problem ist, dass die Eckpunkte nicht verraten, welche Seite Tiefe und welche Breite ist. Dadurch kann das PnP-Verfahren die Maße vertauschen, was die Verpackung im 3D-Raum falsch orientiert, obwohl das Zentrum und damit die Position stimmt. Dies sollte beim Greifen berücksichtigt werden. Der Ansatz liefert im synthetischen Raum gute Werte und bildet eine Basis für spätere Tests an echten Daten.

5.3.4 Berechnung der Verpackungsrotation

Da die Verpackung flach auf dem Förderband liegt, sind Roll und Pitch gleich 0° . Es wird daher nur die Rotation um die Z-Achse (Yaw) betrachtet.

Interpretation der Rotation Zu Beginn wurde die Evaluation mit absoluten Differenzen durchgeführt. Dabei entstand beispielsweise bei einer geschätzten Rotation von $+45^\circ$ und einer Ground Truth von -45° ein Fehler von 90° , obwohl beide Rotationen bei einem Würfel optisch gleich aussehen.

Bei einer quadratischen Verpackung (Ausnahme Würfel), bei der sich die Länge von Breite und Tiefe unterscheidet, kann argumentiert werden, dass diese Rotationen ($+45^\circ$ und -45°) nicht gleich sind. Wenn betrachtet wird, wie der Roboter die Szene wahrnimmt, dann wäre bei -45° und $+45^\circ$ ein Unterschied vorhanden, nämlich welche Seite länger ist. Wenn also beispielsweise ein zusätzlicher Arm die Verpackung richtig positionieren soll, dann würde er bei -45° statt $+45^\circ$ daneben treffen, da die Kantenlängen nicht übereinstimmen würden. Auch wie weit die Arme ausgestreckt werden müssen, wäre hier unsicher. Wird nämlich die kleinere Breite angenommen, ist der Arm nicht weit genug gestreckt und könnte die Verpackung wegschieben. In diesem Fall wäre der absolute Fehler die richtige Metrik.

Mit dem absoluten Fehler liegt der durchschnittliche Fehler bei ca. $5,9^\circ$ bei einer Standardabweichung von $16,2^\circ$. Die hohe Standardabweichung zeigt, dass die Bestimmung der Orientierung bei der Eckpunktfindung nicht zuverlässig funktioniert. Wenn bei der Verpackung eine falsche Orientierung geschätzt wird, entstehen Fehler wie im obigen Beispiel mit $\pm 45^\circ$.

Wenn der oben beschriebene Fall jedoch kein Problem darstellt, weil die Greiferarme so weit ausfahren wie die Diagonale der Verpackung (also beispielsweise der Abstand von unten links nach oben rechts), dann könnte der absolute Fehler ein übertriebenes Ergebnis liefern. In diesem Fall wäre der Fehler zwischen $+45^\circ$ und -45° nicht 90° , sondern 0° . Denn auch wenn die quadratische Verpackung zu einer anderen Seite zeigt, werden die Greiferarme dennoch weit genug geöffnet sein, um die Verpackung greifen zu können. Entscheidend wäre hier dann das Zentrum der Verpackung.

Wenn also der minimale Fehler betrachtet wird, liegt der durchschnittliche Fehler bei nur 3° bei einer Standardabweichung von 6° , was deutlich bessere Ergebnisse sind. Es kommt also stark auf die Interpretation an, welche Evaluationsmethode korrekt ist. Da davon ausgegangen wird, dass der Greifer in beide Richtungen die Arme ausfahren kann, wurde die minimale Fehlermethode gewählt.

Ergebnisse Das Ergebnis mit 3° durchschnittlichem Fehler und 6° Standardabweichung ist trotz des anderen Blickwinkels nicht präzise. Dies liegt daran, dass die vier gefundenen Eckpunkte einen durchschnittlichen Fehler von 2 Pixeln aufweisen. Der Vergleich mit der Positionsschätzung zeigt, dass die Rotation stärker auf Fehler in der Eckpunktfindung reagiert als die Positionsbestimmung. Dies könnte daran liegen, dass die Positionsbestimmung mithilfe der bekannten Verpackungsgrößen genauere Werte liefern kann, während die Rotation nur anhand der Eckpunkte berechnet wird.

Das Best-Case-Bild aus Abbildung 4.10a hat mit unter einem Grad Fehler sehr gut abgeschnitten. Die Eckpunkte werden nicht ganz genau getroffen und dennoch ist

die Rotationsberechnung sehr gut. Wahrscheinlich haben sich die kleinen Fehler gegenseitig ausgeglichen. Anders sieht es beim Worst Case aus Abbildung 4.10b aus, der 45° Fehler aufweist. Das Problem ist, dass die Orientierung falsch bestimmt wurde, kombiniert mit den Fehlern aus der Eckpunktfindung.

Industrielle Anwendung Das Ergebnis der Rotationsberechnung ist nicht präzise genug für die industrielle Anwendung. Je nachdem, ob beide Greiferarme gestreckt werden oder nicht, kann es zu schlechten Ergebnissen führen. Ein durchschnittlicher Fehler von 3° wäre noch im Toleranzbereich, doch die hohe Standardabweichung von 6° macht den Ansatz zu unpräzise.

Insgesamt fiel die Berechnung der Rotation eher schlecht aus. Eine verbesserte Eckpunkterkennung ist notwendig für eine präzisere Berechnung. Auch die Orientierungsbestimmung bei der Eckpunktfindung muss verbessert werden, damit die 3D-Eckpunkte richtig zugeordnet werden. Zusätzlich wären mehr Eckpunkte hilfreich, um das PnP-Verfahren robuster zu machen. Wie im vorherigen Abschnitt können bekannte Informationen genutzt werden um die Methode robuster zu gestalten. Da Roll und Pitch gleich 0 sind, kann dies als Randbedingung in das PnP-Verfahren eingebaut werden. Außerdem sollte untersucht werden, inwieweit die vertauschten Vorzeichen aus Abschnitt 4.3.1 Auswirkungen auf die berechnete Rotation haben.

6

Fazit

Diese Arbeit untersucht, ob die 6D-Pose synthetisch generierter Medikamentenverpackungen ausschließlich mit RGB-Daten zuverlässig bestimmt werden kann. Die Auswertung der Ergebnisse zeigt, dass der vorgestellte Ansatz bei synthetischen Bildern vielversprechende Ergebnisse liefert, jedoch für den industriellen Einsatz noch Verbesserungen benötigt.

Es wurden synthetische Bilder mit Blender und BlenderProc generiert, welche die Szene vereinfacht wiedergeben. Die Szene ähnelt der realen Umgebung, jedoch fehlen Details, beispielsweise die Verpackungen in den Regalen, sodass eine Domain Gap besteht. Die dazugehörigen Segmentierungsmasken und Metadaten lieferten gute Grundlagen für Training und Evaluation.

Das U-Net-Modell erreichte einen Dice Score von 99,70 %, was eine zuverlässige Segmentierung ermöglicht.

Die Positionsbestimmung liefert durchschnittliche Fehler von 1 mm bei der X-Position (seitliche Position), 3 mm bei der Z-Position (Höhe) und 2 cm bei der Y-Position (Tiefe). Der Distanzfehler betrug dementsprechend auch durchschnittlich 2 cm. Die Yaw-Rotation wies einen durchschnittlichen Fehler von 3° bei einer Standardabweichung von 6° auf.

Die größte Schwäche des Ansatzes liegt in der Eckpunkterkennung. Die Eckpunkte werden im Durchschnitt um etwa 2 px verfehlt, und die Zuordnung der Kanten zu Tiefe, Breite oder Höhe ist nicht eindeutig. Außerdem wurden nur Idealfälle betrachtet. Demnach würden Extremfälle potenziell schlechtere Ergebnisse liefern.

Für zukünftige Arbeiten sollte daher die Eckpunkterkennung mit den vorgestellten Verbesserungen ausgebaut werden. Außerdem wären mehr Eckpunkte als Eingabe für das PnP-Verfahren sinnvoll. Zusätzlich können bekannte Randbedingungen eingebaut werden, um robustere Ergebnisse zu erzielen.

Der Ansatz zeigt theoretisches Potenzial für die präzise Erkennung der 6D-Pose für industrielle Zwecke ohne Tiefensensoren. Der nächste Schritt wäre eine verbesserte Eckpunkterkennung und ein Finetuning des U-Net-Modells mit echten Aufnahmen.

Abkürzungsverzeichnis

BOP	Benchmark for 6D Object Pose Estimation
CAD	Computer-Aided Design
CLAHE	Contrast Limited Adaptive Histogram Equalization
CNN	Convolutional Neural Network
CUT	Contrastive Unpaired Translation
DLR	Deutsches Zentrum für Luft- und Raumfahrt
FCOS	Fully Convolutional One-Stage Object Detection
fn	false negative
FoV	Field of View
fp	false positive
FRCSyn	Face Recognition Challenge in the Era of Synthetic Data
HCCE	Hierarchical Continuous Coordinate Encoding
HSV	Hue Saturation Value
ICP	Iterative Closest Point
MSPD	Maximum Symmetry-Aware Projection Distance
MSSD	Maximum Symmetry-Aware Surface Distance
PnP	Perspective-n-Point
R-CNN	Region-based Convolutional Neural Networks
RANSAC	Random Sample Consensus
ReLU	Rectified Linear Unit
RGB	Red Green Blue
RGB-D	Red Green Blue Depth
tn	true negative
tp	true positive

Abbildungsverzeichnis

2.1	Architektur des U-Net mit Kontraktionspfad (links) und Expansionspfad (rechts). [18]	7
3.1	Initiale Blender-Szene vor Bearbeitung.	11
3.2	Beispielbilder nach Anpassung der Basisszene. Auf der Boxoberfläche gibt es zufällige Farb- und Textmuster, die Lichtquelle variiert leicht und es gibt Orientierungsmarker.	13
3.3	Gefundene Orientierungsmarker mithilfe des in Abschnitt 3.3.1 beschriebenen Skripts.	18
3.4	Beispielhafte Bildmasken der möglichen Orientierungen (500 % Zoom).	19
3.5	Gefundene 2D-Eckpunkte mithilfe des in Abschnitt 3.3.2 beschriebenen Skripts (500 % Zoom).	20
4.1	Verpackungen bei unterschiedlichen Rotationswinkeln.	22
4.2	Verpackungspositionen entlang der X-Achse.	23
4.3	Verpackungspositionen entlang der Y-Achse.	23
4.4	Minimale und maximale Verpackungsdimensionen.	24
4.5	Minimale und maximale Textgröße.	24
4.6	Behandelte Sonderfälle im Training.	25
4.7	Segmentierungsmasken der Verpackungen aus Abb. 4.1.	25
4.8	Verlauf des Dice Scores während des Trainings.	26
4.9	Best und Worst Case bei der Positionsbestimmung (500 % Zoom).	30
4.10	Best und Worst Case bei der Rotationsbestimmung (500 % Zoom).	30

Tabellenverzeichnis

4.1	Evaluationsergebnisse der Orientierungsmarkererkennung.	28
4.2	Evaluationsergebnisse der Eckpunkterkennung.	29
4.3	Evaluationsergebnisse der Position (X,Y,Z).	29
4.4	Evaluationsergebnis der Rotation (Yaw).	30

Literatur

- [1] Soofiyan Atar u. a. *OptiGrasp: Optimized Grasp Pose Detection Using RGB Images for Warehouse Picking Robots*. 2024. arXiv: 2409.19494 [cs.R0]. URL: <https://arxiv.org/abs/2409.19494>.
- [2] BOP Challenge. *BOP: Benchmark for 6D Object Pose Estimation*. 2025. URL: <https://bop.felk.cvut.cz/home/> (besucht am 31.12.2025).
- [3] Dimitrios Chamzas, Constantinos Chamzas und Konstantinos Moustakas. „cMinMax: A Fast Algorithm to Find the Corners of an N-dimensional Convex Polytope“. In: *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science und Technology Publications, 2021, S. 229–236. DOI: 10.5220/0010259002290236. URL: <http://dx.doi.org/10.5220/0010259002290236>.
- [4] Himanshu Chandel und Sonia Vatta. „Occlusion Detection and Handling: A Review“. In: *International Journal of Computer Applications* 120.10 (2015), S. 33–38. DOI: 10.5120/21264-3857. URL: https://www.researchgate.net/publication/278729405_Occlusion_Detection_and_Handling_A_Review.
- [5] Ivan DeAndres-Tame u. a. „Second Edition FRCSyn Challenge at CVPR 2024: Face Recognition Challenge in the Era of Synthetic Data“. In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, Juni 2024, S. 3173–3183. DOI: 10.1109/cvprw63382.2024.00323. URL: <http://dx.doi.org/10.1109/CVPRW63382.2024.00323>.
- [6] Maximilian Denninger u. a. *BlenderProc*. 2019. arXiv: 1911.01911 [cs.CV]. URL: <https://arxiv.org/abs/1911.01911>.
- [7] Leon Eversberg und Jens Lambrecht. „Combining Synthetic Images and Deep Active Learning: Data-Efficient Training of an Industrial Object Detection Model“. In: *Journal of Imaging* 10.1 (2024), S. 16. DOI: 10.3390/jimaging10010016. URL: <https://www.mdpi.com/2313-433X/10/1/16>.
- [8] Chengjian Feng u. a. *InstaGen: Enhancing Object Detection by Training on Synthetic Dataset*. 2024. arXiv: 2402.05937 [cs.CV]. URL: <https://arxiv.org/abs/2402.05937>.
- [9] Teng Guo und Jingjin Yu. *RGBTrack: Fast, Robust Depth-Free 6D Pose Estimation and Tracking*. 2025. arXiv: 2506.17119 [cs.CV]. URL: <https://arxiv.org/abs/2506.17119>.
- [10] Benedikt T. Imbusch, Max Schwarz und Sven Behnke. „Synthetic-to-Real Domain Adaptation using Contrastive Unpaired Translation“. In: *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, Aug. 2022, S. 595–602. DOI: 10.1109/case49997.2022.9926640. URL: <http://dx.doi.org/10.1109/CASE49997.2022.9926640>.

- [11] Kilian Kleeberger u. a. „A Survey on Learning-Based Robotic Grasping“. In: *Current Robotics Reports* 1 (2020), S. 239–249. DOI: 10.1007/s43154-020-00021-6. URL: <https://link.springer.com/article/10.1007/s43154-020-00021-6>.
- [12] Tsung-Yi Lin u. a. *Focal Loss for Dense Object Detection*. 2018. arXiv: 1708.02002 [cs.CV]. URL: <https://arxiv.org/abs/1708.02002>.
- [13] Xingyu Liu u. a. *GDRNPP: A Geometry-guided and Fully Learning-based Object Pose Estimator*. 2025. DOI: <https://doi.org/10.1109/TPAMI.2025.3553485>. arXiv: 2102.12145 [cs.CV]. URL: <https://arxiv.org/abs/2102.12145>.
- [14] Yair Movshovitz-Attias, Takeo Kanade und Yaser Sheikh. *How useful is photo-realistic rendering for visual learning?* 2016. arXiv: 1603.08152 [cs.CV]. URL: <https://arxiv.org/abs/1603.08152>.
- [15] Taesung Park u. a. *Contrastive Learning for Unpaired Image-to-Image Translation*. 2020. arXiv: 2007.15651 [cs.CV]. URL: <https://arxiv.org/abs/2007.15651>.
- [16] Aladdin Persson. *Machine Learning Collection*. 2021. URL: https://github.com/aladdinpersson/Machine-Learning-Collection/tree/master/ML/Pytorch/image_segmentation/semantic_segmentation_unet (besucht am 31. 12. 2025).
- [17] Uchitha Rajapaksha u. a. „Deep Learning-based Depth Estimation Methods from Monocular Image and Videos: A Comprehensive Survey“. In: *ACM Computing Surveys* 56 (Okt. 2024), S. 1–51. DOI: 10.1145/3677327. URL: <https://dl.acm.org/doi/10.1145/3677327>.
- [18] Olaf Ronneberger, Philipp Fischer und Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV]. URL: <https://arxiv.org/abs/1505.04597>.
- [19] Manisha Saini und Seba Susan. „Tackling class imbalance in computer vision: a contemporary review“. In: *Artificial Intelligence Review* 56 (2023), S. 1279–1335. DOI: 10.1007/s10462-023-10557-6. URL: <https://link.springer.com/article/10.1007/s10462-023-10557-6>.
- [20] Max Schwarz und Sven Behnke. *Stillleben: Realistic Scene Synthesis for Deep Learning in Robotics*. 2020. arXiv: 2005.05659 [cs.CV]. URL: <https://arxiv.org/abs/2005.05659>.
- [21] Irem Ulku und Erdem Akagündüz. „A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D Images“. In: *Applied Artificial Intelligence* 36.1 (Feb. 2022). ISSN: 1087-6545. DOI: 10.1080/08839514.2022.2032924. URL: <http://dx.doi.org/10.1080/08839514.2022.2032924>.
- [22] Bram Vanherle u. a. *Analysis of Training Object Detection Models with Synthetic Data*. 2022. arXiv: 2211.16066 [cs.CV]. URL: <https://arxiv.org/abs/2211.16066>.
- [23] Yulin Wang u. a. *HccePose(BF): Predicting Front & Back Surfaces to Construct Ultra-Dense 2D-3D Correspondences for Pose Estimation*. 2025. arXiv: 2510.10177 [cs.CV]. URL: <https://arxiv.org/abs/2510.10177>.
- [24] Bowen Wen u. a. *FoundationPose: Unified 6D Pose Estimation and Tracking of Novel Objects*. 2024. arXiv: 2312.08344 [cs.CV]. URL: <https://arxiv.org/abs/2312.08344>.

-
- [25] Hui Zeng u. a. „RGB-D Object Recognition Using Multi-Modal Deep Neural Network and DS Evidence Theory“. In: *Sensors* 19.3 (2019), S. 529. doi: 10.3390/s19030529. URL: <https://www.mdpi.com/1424-8220/19/3/529>.

Anhang

Nutzung KI basierter Anwendungen

Für die vorliegende Bachelorarbeit wurden KI-basierte Tools (Claude, Academic Cloud) als Hilfsmittel eingesetzt. Der Umfang beschränkt sich auf folgende Bereiche:

Sprachliche Korrekturen: KI-Tools wurden zur Überprüfung und Korrektur von Rechtschreibung und Grammatik verwendet. Die inhaltliche Formulierung erfolgte eigenständig.

Literaturrecherche: KI-Tools unterstützten bei der Suche nach relevanten wissenschaftlichen Publikationen zum Themengebiet. Die Analyse der Quellen erfolgte eigenständig.

Programmiertechnische Hilfestellungen: KI-Tools wurden bei technischen Fragestellungen zur Implementierung herangezogen. Die Konzeption, Entwicklung und Implementierung des Codes sowie alle zugrundeliegenden Überlegungen erfolgten eigenständig und mit Hilfe des Betreuers.

Sämtliche fachlichen Inhalte, Analysen, Diskussionen und Schlussfolgerungen dieser Arbeit sind eigenständige Leistungen.