Authentische Verdeckung virtueller 3D-Objekte in Bilderwelten

Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften

vorgelegt beim Fachbereich Informatik und Mathematik
der Johann Wolfgang Goethe-Universität
in Frankfurt am Main

von
Frank Nagl
geboren in Heilbad Heiligenstadt, Deutschland

Frankfurt 2013
D 30

vom Fachbereich Informatik und Mathematik der
Johann Wolfgang Goethe-Universität als Dissertation
angenommen.

Dekan: Prof. Dr. Thorsten Theobald
Gutachter: Prof. Dr.-Ing. Detlef Krömker, Prof. Dr. Paul Grimm
Datum der Disputation: 16. Mai 2013

Abstract

The goal of this work is the authentic occlusion of embedded virtual objects in augmented *photo collections* by using only a small number of photos. The realization of occlusions of virtual and real objects in an Augmented Reality application require depth information. Usually this information is computed by performing a complete 3D reconstruction of the image-based scene, which requires a big number of input photos. In contrast, in this work a system was developed, which based on a direct image-based rendering approach without performing a 3D reconstruction. This image-based rendering approach works with incomplete depth information and realizes a high image quality regarding authentic occlusion. This work opens up new scopes of application like automatic visualization of 3D planning data and 3D product presentations in photos or photo collections, because in these scopes a large number of input photos is often not available. Especially for these application areas authentic occlusions are important for the user acceptance of an augmentation. An authentic occlusion is the visually, correct superimposition of virtual objects and image parts of one or more photos according to human perception. The result is presented as an augmented photo collection. A photo collection is an image-based 3D world, in which the photos are spatialized by their visual content. Thus, this work is part of the field of research Augmented Reality. In context of this work, an approach for image-based rendering with authentic occlusions based on incomplete depth information and several approaches for computing the necessary depth information were developed. The *Sliced-Image-Rendering* technique uses the incomplete depth information to render 2D images as a three-dimensional relief in order to realize an authentic occlusion of embedded virtual objects. The computation of necessary depth information of a 2D image represents a separate challenge, because a high-quality 3D reconstruction cannot be performed. Thus, the question is how individual depth information can be computed and mapped to larger image areas. In this work, three depth mapping approaches were developed that differ in terms of used data and their processing. The *Segment-Depth-Matching* technique assigns depth to image segments using the 3D scene information of the photo collection. This assumes segment images. As a result, for each photo a depth map exists. To allow a depth mapping even without a prior segmentation, the

Key-Point-Depth-Matching technique has been developed. In this technique, the 3D scene information of the photo collection is projected onto the image plane as circular sprites. All projected sprites of an image are combined to produce the corresponding depth map. Both techniques produce surfaces with depth information, but no pixel-accurate depth maps. To create pixel-accurate depth maps, the *Geometry-Depth-Matching* technique has been developed. In this method, geometry is generated for the region that is visible in a photo and thereby a pixel-accurate depth-map is created. The generation of geometry requires a semi-automatic segmentation step. The generated scene geometry is not a full 3D reconstruction of the global scene shown in complete photo collection, as only geometry for a single image is generated. The approaches were implemented into a system and validated. The subsequent results are analyzed and evaluated in terms of authentic occlusion by using several scenes with different properties (exterior and interior scenes, rich and poor in detail, different number of input photos) as test data. The evaluation of the Sliced-Image-Rendering technique shows that authentic occlusion can be realized by using incomplete depth information obtained by the developed depth-matching techniques. With the aid of the developed system in this work, image-based applications can realize augmentations with high image quality in terms of an authentic occlusion even though only a small number of input photos is given.

Kurzfassung

Das Ziel dieser Arbeit ist es, eine authentische Verdeckung eingebetteter virtueller 3D-Objekte in augmentierten Bilderwelten bei einer geringen Anzahl an Fotos innerhalb der Bilderwelt zu erreichen. Für die Verdeckung von realen und virtuellen Anteilen einer Augmented Reality-Szene sind Tiefeninformationen notwendig. Diese stammen üblicherweise aus einer 3D-Rekonstruktion, für deren Erstellung sehr viele Eingangsbilder notwendig sind. Im Gegensatz dazu wurde in dieser Arbeit ein System entwickelt, das eine vollständige 3D-Rekonstruktion umgeht. Dieses beruht auf einem direkten bildbasierten Rendering-Ansatz, welcher auch mit unvollständigen Tiefeninformationen eine hohe Bildqualität in Bezug auf eine authentische Verdeckung erreicht. Daraus erschließen sich neue Anwendungsgebiete, wie z.B. die automatisierte Visualisierung von 3D-Planungsdaten und 3D-Produktpräsentationen in Bildern bzw. Bilderwelten, da in diesen Bereichen oftmals nicht genügend große Bildmengen vorhanden sind. Gerade für diese Anwendungsgebiete sind authentische Verdeckungen für die Nutzerakzeptanz der Augmentierung wichtig. Unter authentischer Verdeckung wird die entsprechend der menschlichen Wahrnehmung visuell korrekte Überlagerung zwischen virtuellen Objekten und einzelnen Bildanteilen eines oder mehrerer Fotos verstanden. Das Ergebnis wird in Form einer Bilderwelt (eine bildbasierte 3D-Welt, die die Fotos entsprechend der Bildinhalte räumlich anordnet) präsentiert, die mit virtuellen Objekten erweitert wurde. Folglich ordnet sich diese Arbeit in das Fachgebiet der Augmented Reality ein. Im Rahmen dieser Arbeit wurde ein Verfahren für die bildbasierte Darstellung mit authentischen Verdeckungen auf der Basis von unvollständigen Tiefeninformationen sowie unterschiedliche Verfahren für die notwendige Berechnung der Tiefeninformationen entwickelt und gegenübergestellt. Das *Sliced-Image-Rendering*-Verfahren rendert mithilfe unvollständiger Tiefeninformationen ein Bild ohne 3D-Geometrie als dreidimensionale Darstellung und realisiert auf diese Weise eine authentische Verdeckung. Das Berechnen der dafür notwendigen Tiefeninformationen eines 2D-Bildes stellt eine gesonderte Herausforderung dar, da die Bilderwelt nur wenige und unvollständige 3D-Informationen der abgebildeten Szene bereitstellt. Folglich kann eine qualitativ hochwertige 3D-Rekonstruktion nicht durchgeführt werden. Die Fragestellung ist daher, wie

einzelne Tiefeninformationen berechnet und diese anschließend größeren Bildbereichen zugeordnet werden können. Für diese Tiefenzuordnung wurden im Rahmen der vorliegenden Arbeit drei verschiedene Verfahren konzipiert, die sich in Bezug auf genutzte Daten und deren Verarbeitung unterscheiden. Das *Segment-Depth-Matching*-Verfahren ordnet Segmenten eines Bildes mithilfe der 3D-Szeneninformationen der Bilderwelt eine Tiefe zu. Hierfür werden Segmentbilder vorausgesetzt. Als Ergebnis liegt für jedes Foto eine Depth-Map vor. Um eine Tiefenzuordnung auch ohne eine vorangehende Segmentierung zu ermöglichen, wurde das *Key-Point-Depth-Matching*-Verfahren entwickelt. Bei diesem Verfahren werden die 3D-Szeneninformationen der Bilderwelt auf die Bildebene als kreisförmige *Sprites* projiziert. Die Distanz zur Kamera wird dabei als Tiefenwert für das Sprite verwendet. Alle projizierten Sprites einer Kamera ergeben die Depth-Map. Beide Verfahren liefern Flächen mit Tiefeninformationen, aber keine pixelgenauen Depth-Maps. Um pixelgenaue Depth-Maps zu erzeugen, wurde das *Geometry-Depth-Matching*-Verfahren entwickelt. Bei diesem Verfahren wird eine Szenengeometrie des abgebildeten Szenenausschnittes erzeugt und dadurch eine pixelgenaue Depth-Map erstellt. Hierfür wird ein semiautomatischer Skizzierungsschritt vorausgesetzt. Die erzeugte Szenengeometrie stellt keine vollständige 3D-Rekonstruktion der Bilderweltenszene dar, da nur ein Szenenausschnitt aus Sicht einer Kamera rekonstruiert wird. Anhand einer technischen Umsetzung erfolgte eine Validierung der konzeptionellen Verfahren. Die daraus resultierenden Ergebnisse wurden anhand verschiedener Bilderweltenszenen mit unterschiedlichen Eigenschaften (Außen- und Innenraumszenen, detailreich und -arm, unterschiedliche Bildmengen) evaluiert. Die Evaluierung des Sliced-Image-Renderings zeigt, dass mithilfe unvollständiger Tiefeninformationen der entwickelten Depth-Matching-Verfahren und unter Einhaltung der gestellten Anforderungen (wenig Eingabefotos, kleine Szenen, keine 3D-Rekonstruktion) eine authentische Verdeckung eingebetteter virtueller 3D-Objekte in Bilderwelten realisiert werden kann. Mithilfe des entwickelten Systems können bildbasierte Anwendungen auch mit kleinen Fotomengen Augmentierungen mit hoher Bildqualität in Bezug auf eine authentische Verdeckung realisieren.

Textliche Hervorhebungen

In dieser Arbeit werden bestehende Fachbegriffe der *Grafischen Datenverarbeitung* (*GDV*) verwendet und eigene Begriffe vom Autor definiert. Bestehende Fachbegriffe sowie eigene Begriffe werden bei erstmaliger Verwendung in einem (Unter-)Kapitel oder nach längeren Textabsätzen wiederholt hervorgehoben. Voranstehende Adjektive hervorgehobener Begriffe werden (nur in der Hervorhebung) groß geschrieben. Die folgende Tabelle erläutert die Arten der Hervorhebungen.

Hervorhebung	Bedeutung
Bestehende Fachbegriffe	Kursiv geschriebene Begriffe stellen Fachbegriffe dar, die nicht vom Autor definiert wurden. Diese Begriffe werden im Sachregister dieser Arbeit (Seite 250 ff.) aufgelistet.
Absätze und Kapitel	Ebenfalls kursiv werden Verweise auf (Unter-)Kapitel und Absätze geschrieben.
Vom Autor definierte Begriffe	Dunkelblaue und kursiv geschriebene Begriffe sind im Rahmen dieser Arbeit vom Autor definierte Begriffe. Diese Begriffe werden im Glossar dieser Arbeit (Seite 255 ff.) zusammengetragen und werden nicht im Sachregister dieser Arbeit aufgelistet.

Tabelle 1: Erläuterungen der Bedeutung textlicher Hervorhebungen in dieser Arbeit.

Danksagung

Die vorliegende Arbeit entstand im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter an der Fachhochschule Erfurt und der Hochschule Fulda für das Projekt PoP-EYE (*Entwicklung von Methoden und Werkzeugen zur virtuellen Produktpräsentation in 3D Community Photo Collections*). PoP-EYE wurde durch das Bundesministerium für Bildung und Forschung gefördert (Projekt-Nummer: 17N0909).

Mein besonderer Dank gilt Herrn Prof. Dr. Paul Grimm für sein Vertrauen in meine Person und die damit verbundene Möglichkeit an diesem Projekt zu arbeiten sowie für seine intensive fachliche Betreuung während dieser Arbeit.

Ebenfalls danken möchte ich Herrn Prof. Dr. Detelf Krömker für seine Betreuung und fortwährende Unterstützung.

Weiterhin möchte ich mich bei meinen Büro- und Projektkollegen sowie langjährigen Freund Konrad Kölzer für seine Unterstützung in Rahmen von fachlichen Gesprächen, Ideenfindungen und einigen Kneipentouren bedanken.

Des Weiteren danke ich allen anderen wissenschaftlichen Mitarbeitern der Angewandten Informatik der FH Erfurt für die technische und fachliche Unterstützung, die ausgezeichnete Zusammenarbeit und viele lustige Momente.

Nicht zu unterschätzen und deshalb mit großem Dank versehen ist der Rückhalt, den ich aus meinem privaten Lebensumfeld erfahren habe. Allen voran danke ich meinen Eltern Reinhold und Sylvia Nagl, meinen Großeltern Horst und Helga Hunstock sowie meinen Geschwistern, Neffen und Nichten. Ebenfalls danke ich meinen Großeltern Reinhold und Gertrud Nagl, obgleich sie zu meinem tiefsten Bedauern während der Entstehung dieser Arbeit verstarben.

Zum Schluss geht ein spezieller Dank an meine Freunde, die mich während der nicht immer ganz leichten Zeit stets unterstützt und motiviert haben.

Ich widme diese Arbeit all denjenigen, die trotz kleiner und großer Hürden stets an mich geglaubt haben. Danke für euer Vertrauen.

Inhaltsverzeichnis

Abkürzungsverzeichnis

Abb.	Abbildung
AR	Augmented Reality
AV	Augmented Virtuality
Bsp.	Beispiel
bzw.	beziehungsweise
d. h.	das heißt
i. d. R.	in der Regel
MR	Mixed Reality
u. a.	unter ander[e]m, unter ander[e]n, und and[e]re, und and[e]res
VR	Virtual Reality
z. T.	zum Teil
z. B.	zum Beispiel

Abbildungsverzeichnis

Formelverzeichnis

Programmcodeverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Mixed Reality (*MR*) bezeichnet allgemein das Vermischen von Realität und Virtualität. Angelehnt an Milgram's *Realitäts-Virtualitäts-Kontinuum* (siehe Abb. 1.1) definieren die Begriffe *Augmented Reality* (*AR*), *Augmented Virtuality* (*AV*) und *Virtual Reality* (*VR*) den Grad der vermischten realen und virtuellen Anteile [MTUK94]. Der Begriff Augmented Reality liegt dementsprechend näher bei der Realität, nur einzelne virtuelle Objekte werden in eine reale Umgebung eingebettet. Augmented Virtuality kann als Gegenstück interpretiert werden, bei dem reale Objekte in eine virtuelle Welt eingeblendet werden [Tön10]. Die hier zugrunde liegende Arbeit ist der AR zuzuordnen.

In [Tön10] wird die Realisierung eines AR-Systems durch die Verwendung verschiedener Komponenten aus drei Bereichen definiert. Diese Bereiche sind Darstellung, Tracking und Interaktion. Entsprechend dieser Unterteilung ist die Dissertation den Bereich Darstellung zuzuordnen, genauer gesagt in den Bereich 3D-*Rendering*. Das dreidimensionale Darstellen/Rendern einer AR-Umgebung mit virtuellen 3D-Objekten ist wiederum der 3D-Computergrafik zuzuordnen.

Eine spezielle Form von AR-Szenen stellen *Bilderwelten* dar, die eine 3D-Welt aus mehreren Fotos unterschiedlicher Perspektiven aufbauen. In Absatz

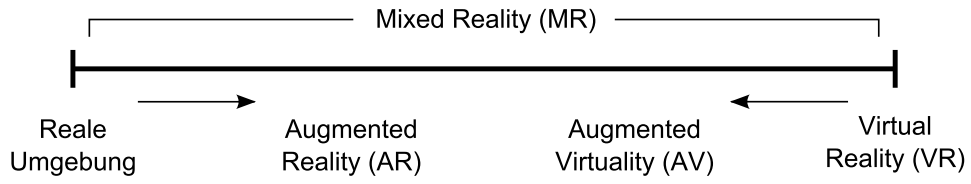


Abbildung 1.1: Milgram's Realitäts-Virtualitäts-Kontinuum [MTUK94].

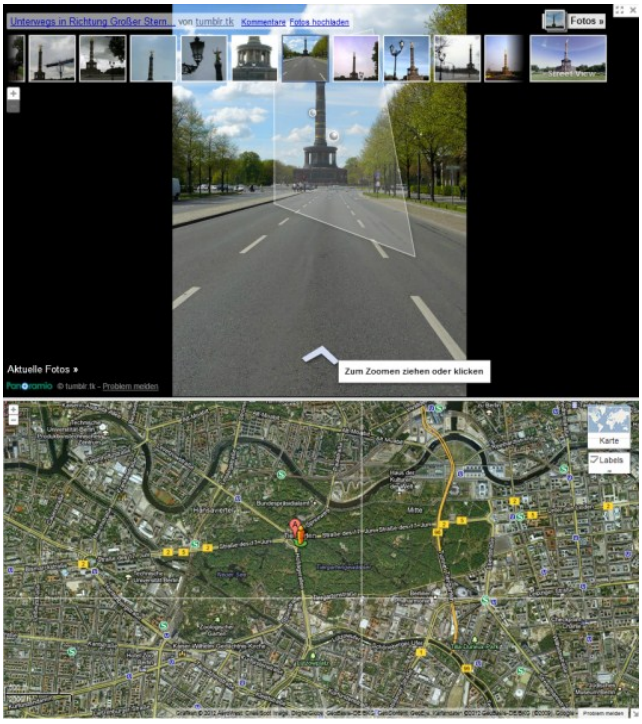


Abbildung 1.2: Google Street View: Erweiterung des zweidimensionalen Kartendienstes Google Maps mithilfe von Bilderwelten [Goo07].

2.6 findet sich eine detaillierte Erläuterung zu Bilderwelten. Abb. 1.2 zeigt den Google-Dienst Street View, der mithilfe von Bilderwelten den zweidimensionalen Kartendienst Google Maps als 3D-Welt erweitert [Goo07].

Ein weiteres Beispiel für die Verwendung von Bilderwelten stellt Microsoft's Online-Bildverwaltungssoftware Photosynth dar [Mic08]. Im Sinne des *User-Generated-Content* werden 3D-Welten aus den von Nutzern bereitgestellten Bildern erzeugt (siehe Abb. 1.3).

1.1 Motivation

In vielen Alltags- und Arbeitssituationen sowie bei der Erfassung von Prozessabläufen können Augmented Reality-Anwendungen die erforderliche Vorstellungskraft unterstützen. Eine AR-basierte Vorschau von Architektur- und Einrichtungsplanungen können Architekten, Bauingenieuren und Bauherren einen vorzeitigen Blick auf das potenzielle Ergebnis ihrer Vorhaben geben. Medizinische Operationen oder Wartungs- und Reparaturarbeiten können vir-

Abbildung 1.3: Photosynth: Bilderwelten werden aus den von Nutzern bereitgestellten Bildern erzeugt [Mic08].

tuell simuliert und trainiert werden. AR-basierte Spezialeffekte in der Film- und Computerspieleindustrie liefern fotorealistische Ergebnisse und reduzieren Kosten und Aufwand. AR erweitert also die Perzeption des Nutzers gegenüber der realen Welt sowie die Interaktion mit ihr [Azu97]. Hierbei spielt der visuelle Eindruck die größte Rolle für die Glaubwürdigkeit des Bildes und damit für die Akzeptanz des Nutzers. Je schwieriger die visuelle Trennbarkeit von virtuellen und realen Anteilen für den Nutzer erscheint, desto besser der authentische Eindruck bei ihm. Dies ist zurückzuführen auf die empfindliche Wahrnehmung des menschlichen Auges auf *visuelle Störungen* sowie auf physikalische Unstimmigkeiten in Abbildungen [Reg99]. Unter visuellen Störungen in Bilderwelten werden visuelle Auffälligkeiten bezeichnet, die einen homogenen Gesamteindruck der Bilderwelt verhindern. Dementsprechend ist das Hauptziel der Darstellung einer AR-Szene, dass der Nutzer die virtuellen und realen Anteile (fast) nicht unterscheiden kann. In dieser Arbeit wird dies als *Visuelle Verschmelzung* der virtuellen und realen Anteile bezeichnet. Das visuelle Verschmelzen ohne visuelle Störungen von verschiedenen realen Fotos und synthetischen Bildern bzw. 3D-Objekten wird im weiteren Text als *Authentische Augmentierung* bezeichnet. Für den realen Anteil innerhalb einer AR-Szene werden i.d.R. digitale Bilder (oder Videoströme) genutzt. Dies bietet zusätzlich Potenzial für die Glaubwürdigkeit und Akzeptanz, da der Nutzer einen persönlichen Bezug zu der AR-Szene entwickeln kann, beispielsweise wenn die Fotos persönlichen

Charakter haben.

Ein potentielles Einsatzgebiet von Augmented Reality stellt die so genannte *Planungsvisualisierung* dar. Der Begriff Planungsvisualisierung beinhaltet die Darstellung und Präsentation einer Planung sowie dazugehörige Daten in Bereichen wie Architektur, Innen- und Außeneinrichtung und Geoinformationssysteme. Für das Erstellen von dreidimensionalen Planungsvisualisierungen existieren viele VR-basierte Planungs- und 3D-Modellierungsanwendungen, wie z. B. der IKEA Home Planner [IKE12] oder der pCon.planner [Eas11]. Jedoch muss der Nutzer dafür erst den Umgang mit dem Programm erlernen und das maßstabsgetreue Modellieren als Abbild der realen Welt bewerkstelligen. Hierzu sind Kenntnisse über 3D-Geometrie sowie über Transformationen im 3D-Raum notwendig. Auch das Integrieren realer Anteile beschränkt sich i. d. R. auf das Einfügen von Fotos als Texturen in eine rein virtuelle Umgebung. Der Szeneninhalt wird also nicht durch reale Anteile repräsentiert. Abb. 1.4 zeigt eine Planungsvisualisierung in einer rein virtuellen Umgebung mit Fotos als Texturen [Eas11]. Eine automatische Erzeugung einer AR-basierten dreidimen-

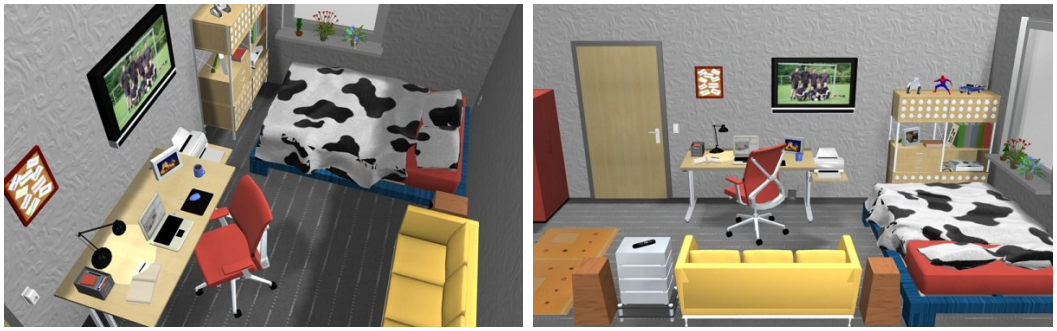


Abbildung 1.4: Planungsvisualisierung in einer rein virtuellen Umgebung [Eas11].

sionalen Planungsvisualisierung ohne spezielle Hardware vorauszusetzen, wäre wünschenswert.

Eine einfachere Art der Erstellung von dreidimensionalen Visualisierungen stellt die Verwendung von Bilderwelten dar. Dafür werden Fotos beliebiger Kameras aus beliebigen Standorten ohne vorangehende geometrische Kalibrierungen genutzt. Die Erstellung der dreidimensionalen Visualisierung (oder zumindest die Bereitstellung erforderlicher Transformationsdaten dafür) erfolgt automatisch und setzt keinerlei Spezialhardware voraus. Aktuelle Bilderwelten-

Anwendungen (wie Photosynth [Mic08] und Google Street View [Goo07], weitere Anwendungen und Projekte finden sich in Absatz *3.2*) beschränken sich auf das Darstellen der Bilder in einer 3D-Umgebung. Eine Augmentierung mit virtuellen 3D-Objekten und somit der Einsatz als Planungsvisualisierung existiert nicht. Das Einbetten virtueller Objekte in eine Bilderwelt kann jedoch als AR-basierte dreidimensionale Planungsvisualisierung gesehen werden. Dies würde eine neue Qualität der Visualisierung repräsentieren, da im Gegensatz zu VR-basierten Planungs- und 3D-Modellierunganwendungen kein händisches Modellieren durch den Nutzer erfolgen muss und die tatsächliche, reale Welt in den Fotos der Bilderwelt abgebildet und augmentiert ist. Die abgebildete reale Umgebung der Bildinhalte einer Bilderwelt wird im weiteren Text *Bilderweltenszene* oder als Kurzform nur *Szene* genannt. In den Fotos dargestellte einzelne reale Objekte werden als *Abgebildete Objekte* bezeichnet.

Für die authentische Augmentierung der Bilderwelten müssen die eingebetteten virtuellen Objekte physikalische Gegebenheiten der Szene (bzw. der realen Welt, die abgebildet wird) berücksichtigen (z. B. räumlicher Aufbau, Maßstab und Beleuchtungssituation der Szene). Zusätzlich müssen visuelle Störungen (z. B. unterschiedliche Aufnahme- und Bildqualität der Fotos, fehlerhafte Verdeckung der virtuellen Objekte) innerhalb der augmentierten Bilderwelt vermieden bzw. minimiert werden. Daraus ergeben sich verschiedene Problemfelder, die in dieser Arbeit in fünf Schwerpunkte (im weiteren Text als *Augmentierungsschwerpunkte* bezeichnet) unterteilt und eingeordnet werden (siehe Abb. 1.5).

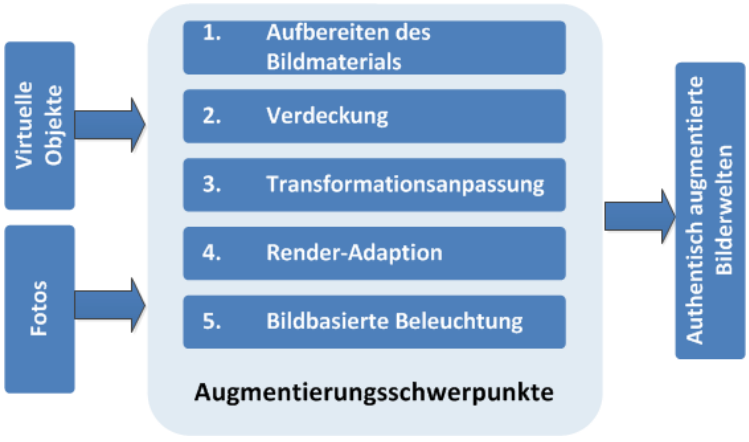


Abbildung 1.5: Schwerpunkte für das authentische Verschmelzen realer und virtueller Anteile in Bilderwelten (Augmentierungsschwerpunkte).

Diese Augmentierungsschwerpunkte werden an dieser Stelle kurz erläutert. Die Reihenfolge ergibt sich aus den Abhängigkeiten der Schwerpunkte untereinander, wie in den folgenden Erläuterungen erklärt wird.

1.1.1 Aufbereiten des Bildmaterials

Da Bilderwelten aus gewöhnlichen RGB-Fotos herkömmlicher Consumer-Kameras erstellt werden, können unterschiedliche Aufnahme- und Bildqualitäten der Fotos den visuellen Eindruck einer Bilderwelt stören, wie in Abb. 1.6(a) dargestellt ist. Ursachen hierfür sind Bildrauschen, unterschiedliche Auflösungen, Über-/Unterbelichtungen, qualitative Unterschiede von CCD-Sensoren verschiedener Aufnahmegeräte, unterschiedliche Beleuchtungssituationen der Szene, durch den Fotografen oder bewegte Objekte verursachte Verwacklungen/Unschärfe sowie unterschiedliche Ausdehnungen des scharfen Bereichs eines Bildes (Tiefenschärfe). Somit ist eine Anpassung der Bildinhalte der Fotos notwendig. Das vorhandene Bildmaterial muss so aufbereitet werden, dass die Fotos als eine homogene Szene in der Bilderwelt dargestellt werden [BGKN09].

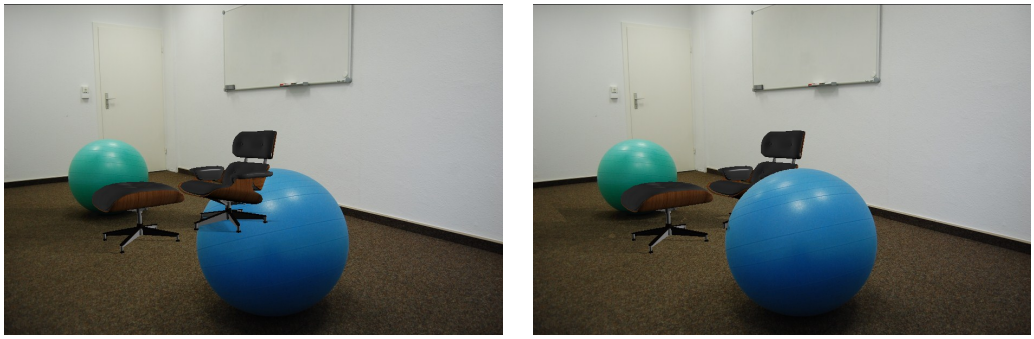
(a) Originalbilder (b) Angepasste Bilder

Abbildung 1.6: Darstellung einer Bilderwelt (a) ohne bzw. (b) mit der Darstellungsanpassung der Bilder

1.1.2 Verdeckung

Nach dem *Aufbereiten des Bildmaterials* ist die Bilderwelt für eine Augmentierung vorbereitet, da nur mit einer homogenen Szene eine glaubwürdige Augmentierung erfolgen kann. Ein Problem bei der Integration virtueller Objekte in die Bilderweltenszene stellen notwendige Verdeckungen dar. In die Bilderwelten eingebettete, virtuelle Objekte müssen unter Umständen visuell

durch einzelne Bildsegmente eines oder mehrerer Fotos überlagert werden. Dieser Umstand wird gefordert, wenn sich virtuelle Objekte hinter anderen abgebildeten Objekten der Szene befinden sollen. Erfolgt bei einer AR-Darstellung eine notwendige Überlagerung nicht, wirkt die Augmentierung nicht authentisch [Aba06]. Die fehlende Überlagerung führt zu einem störenden Eindruck beim Betrachter. Die (allgemeine) Überlagerung zwischen virtuellen Objekten und einzelnen Bildanteilen eines oder mehrerer Fotos wird *Verdeckung* genannt. Eine der menschlichen Wahrnehmung entsprechenden, korrekten Überlagerung wird im weiteren Text als *Authentische Verdeckung* bezeichnet. Abb. 1.7 zeigt eine Augmentierung in Bilderwelten ohne und mit korrekter Verdeckung.



(a) Ohne Verdeckung (b) Mit Verdeckung

Abbildung 1.7: Augmentierte Bilderwelt (a) ohne bzw. (b) mit Berücksichtigung korrekter Verdeckungen. Der eingebettete virtuelle Stuhl muss für eine authentische Verdeckung von dem vorderen abgebildeten realen Ball überlagert werden.

1.1.3 Transformationsanpassung

Ein weiteres Problemfeld bei der authentischen Integration virtueller Objekte stellen geometrische Gegebenheiten der Objekte dar. Für einen authentischen Eindruck muss die Größe der virtuellen Objekte an den Maßstab der Bilderweltenszene angepasst sein. Diese Anpassung wird im weiteren Text als *Transformationsanpassung* bezeichnet.

1.1.4 Render-Adaption

Durch die *Transformationsanpassung* und einer korrekten *Verdeckung* erscheinen die virtuellen Objekte geometrisch (räumlich) korrekt innerhalb der Bilderweltenszene. Für eine authentische Integration müssen neben geometrischen Gegebenheiten auch Darstellungsanpassungen zwischen Bildern und virtuellen Objekten vorgenommen werden. Zwar sind durch das *Aufbereiten des Bildmaterials* die Fotos untereinander abgestimmt, jedoch kann weiterhin die Bildqualität der angepassten Fotos bei der Integration hochaufgelöster virtueller 3D-Objekte zu visuellen Störungen führen. Folglich müssen Anpassungen zwischen der Bildqualität der Fotos und dem Rendering der virtuellen Objekte erfolgen [NGBA10]. Diese Darstellungsanpassung zwischen virtuellen Objekten und der Bilderwelt wird weiterführend als *Render-Adaption* bezeichnet.

(a) Hochaufgelöstes virtuelles Objekt in einem niedrig aufgelösten Bild

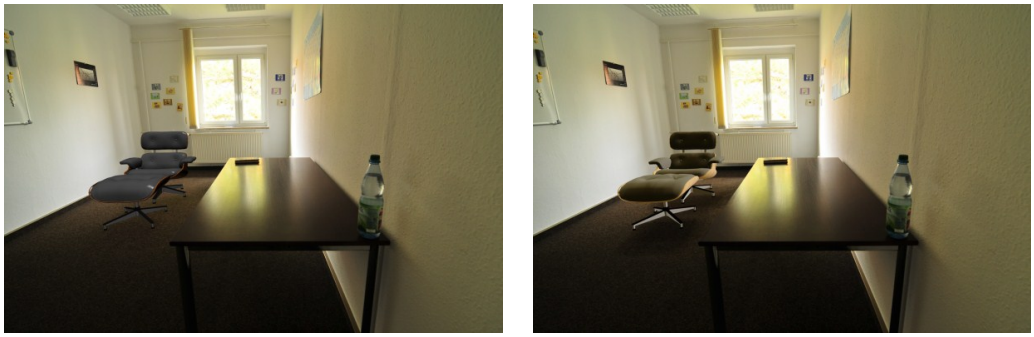
(b) An die Bildauflösung angepasstes virtuelles Objekt

Abbildung 1.8: Augmentierung (a) ohne bzw. (b) mit der Render-Adaption zwischen virtuellen und realen Anteilen [NGBA10].

1.1.5 Bildbasierte Beleuchtung

Zusätzlich zu der *Render-Adaption* müssen die virtuellen Objekte mit einer zu der Bilderweltenszene konformen Beleuchtung dargestellt werden. Die abgebildete reale Umgebung der Bildinhalte beinhaltet eigene Lichtquellen. Für eine authentische Augmentierung müssen daher die virtuellen Objekte eine zu den Bildinhalten passende Beleuchtungssituation aufweisen [KNG11]. Hierzu gehört auch ein zu der Szene passender Schattenwurf. Dies wird im weiteren Text

als *Bildbasierte Beleuchtung* bezeichnet. Abb. 1.9 zeigt eine augmentierte Bilderwelt mit Standard- bzw. bildbasierter Beleuchtung des virtuellen Objektes.



(a) Standardbeleuchtung (b) Bildbasierte Beleuchtung mit Schatten

Abbildung 1.9: Augmentierte Bilderwelt mit (a) Standard- bzw. (b) bildbasierter Beleuchtung des virtuellen Objektes.

Nur das Zusammenspiel von allen fünf Augmentierungsschwerpunkten ermöglicht eine authentische Augmentierung von Bilderwelten. Folglich müssen für eine authentische Verdeckung alle weiteren Augmentierungsschwerpunkte berücksichtigt werden, da sonst (trotz korrekter Verdeckungsdarstellung) *visuelle Störungen* den homogenen Gesamteindruck der Bilderwelt verhindern und somit das Ziel der authentischen Verdeckung nicht erreicht werden kann. Der Kern dieser Arbeit konzentriert sich auf den Augmentierungsschwerpunkt *Verdeckung*. Daraus ergibt sich die Zielstellung im folgenden Absatz.

1.2 Zielsetzung der Arbeit

Ziel dieser Arbeit ist es, *in augmentierten Bilderwelten eine authentische Verdeckung zu ermöglichen*. Dies wird im weiteren Text als *Hauptziel* bezeichnet. Dabei soll das Hauptziel auch bei einer geringen Anzahl an Fotos innerhalb der Bilderwelt realisierbar sein. Des Weiteren sollen beliebige Fotos beliebiger Kameras genutzt werden können. Es sollen keine Kalibrierungen der Kameras, keine Stereokameras oder weitere Spezialhardware vorausgesetzt werden. Aufgrund der geringen Anzahl an Eingabefotos soll auf eine vollständige 3D-Rekonstruktion der Bilderweltenszene verzichtet werden (eine 3D-Rekonstruktion erfordert eine

verhältnismäßig große Anzahl an Eingabefotos sowie eine große Abdeckung der abgebildeten Szene, siehe Absatz *3.3*). Es soll ein Konzept aufgebaut werden, dass (ohne eine vollständige 3D-Rekonstruktion) Mechanismen und Verfahren für die Erreichung des Hauptziels bereitstellt. Hierzu sollen vorliegende Daten einer Bilderwelt genutzt werden. Mithilfe der (unvollständigen) 3D-Punktwolke und der rekonstruierten Kamerapositionen und -ausrichtungen soll einzelnen Bildanteilen der Fotos Tiefeninformationen zugeordnet werden. Auf Basis von *Image-Based-Rendering*-Mechanismen und den zugeordneten Tiefeninformationen sollen eingebettete, virtuelle Objekte (je nach Notwendigkeit) visuell durch die entsprechenden Bildanteile bei der Darstellung der Bilderwelt überlagert werden. Aufbauend auf dem Konzept soll eine technische Umsetzung erfolgen, die das authentische Verdecken virtueller 3D-Objekte in augmentierten Bilderwelten realisiert. Abschließend sollen die damit erzielten Ergebnisse, stets im Hinblick auf das Hauptziel, bewertet und verglichen werden.

Der Augmentierungsschwerpunkt *Verdeckung* erklärte die Notwendigkeit visueller Überlagerungen zwischen virtuellen und in Fotos abgebildeten Objekten (Bildsegmente). Diese abgebildeten Objekte werden im weiteren Text als *Occluder* bezeichnet [Aba06].

In Absatz *1.1* (*Motivation*) wurde erklärt, dass eine authentische Augmentierung neben der Verdeckung vier weitere Augmentierungsschwerpunkte beinhaltet. Nur im Zusammenspiel kann eine authentische Augmentierung realisiert werden. Folglich müssen für eine authentische Verdeckung alle weiteren Augmentierungsschwerpunkte berücksichtigt werden, da sonst (trotz korrekter Verdeckungsdarstellung) *visuelle Störungen* den homogenen Gesamteindruck der Bilderwelt verhindern und somit das Ziel der authentischen Verdeckung nicht erreicht werden kann.

Aktuelle Bilderwelten-Anwendungen (wie Photosynth [Mic08] und Google Street View [Goo07]) ermöglichen bei der Darstellung in ihrer 3D-Umgebung das Navigieren von einer Kamera zu einer anderen bzw. von einem Foto zu einem anderen Foto. Die Navigation erfolgt auf einem vorgegebenen Pfad. Eine freie, kameraunabhängige Navigation durch die Bilderweltenszene ist nicht möglich. Diese Navigationsart wird im weiteren Text als *Kameranavigation* bezeichnet. Das bedeutet, die authentische Verdeckung kann für jedes Foto unabhängig von anderen Fotos realisiert und dargestellt werden. Des Weite-

ren ist keine vollständige Analyse/Rekonstruktion der Tiefe des Bildinhaltes notwendig. Es ist ausreichend, bei der Darstellung des Bildes einer Kamera zu entscheiden, welche Pixel (Bildsegmente) ein eingebettetes, virtuelles Objekt visuell überlagern (verdecken) und welche nicht.

Eine Erweiterung der bestehenden Navigationsart in Bilderwelten ist die so genannte *Freie Navigation*. Bei der freien Navigation bewegt sich der Betrachter aus der Egoperspektive beliebig durch die Bilderweltenszene, vergleichbar mit der Navigation eines Ego-Shooter-Computerspiels. Das bedeutet, die Betrachterpositionen entsprechen, im Gegensatz zur Kameranavigation, nicht zwangsläufig einer Kameraposition für ein benutztes Bild. Dafür muss die Tiefe des Bildinhaltes gewonnen werden und das Bild innerhalb der Bilderwelt als eine dreidimensionale Darstellung der abgebildeten Szene gerendert werden, damit auch bei freier Navigation die Darstellung einer augmentierten Bilderwelt ohne visuelle Störungen und mit authentischen Verdeckungen ermöglicht werden kann. Die dafür benötigte Projektion der rekonstruierten Szene aus einer beliebigen Position und Ausrichtung innerhalb der 3D-Welt auf eine Bildebene wird im weiteren Text als *3D-Bilddarstellung* bezeichnet.

Da die jeweiligen Vorgehensweisen bei der Realisierung einer authentischen Verdeckung aufeinander aufbauen, werden zwei Teilziele definiert:

1. Authentische Verdeckung augmentierter Bilderwelten bei Kameranavigation,

2. Authentische Verdeckung augmentierter Bilderwelten bei freier Navigation.

1.3 Kernanforderungen an die Arbeit

Aus der definierten Zielstellung heraus ergeben sich einige Anforderungen an diese Arbeit. Dieser Absatz trägt die Anforderungen noch einmal zusammen und definiert die Kernanforderungen an die vorliegende Arbeit für das Erreichen des Hauptziels.

Für eine authentische Verdeckung in einer 3D-Szene stellt das Konstruieren und Verwenden einer 3D-Geometrie der Szene einen etablierten Weg dar [FBS04] [LB00]. Hierzu existieren verschiedene Ansätze (siehe Absatz *3.3* (*Image-Based-*

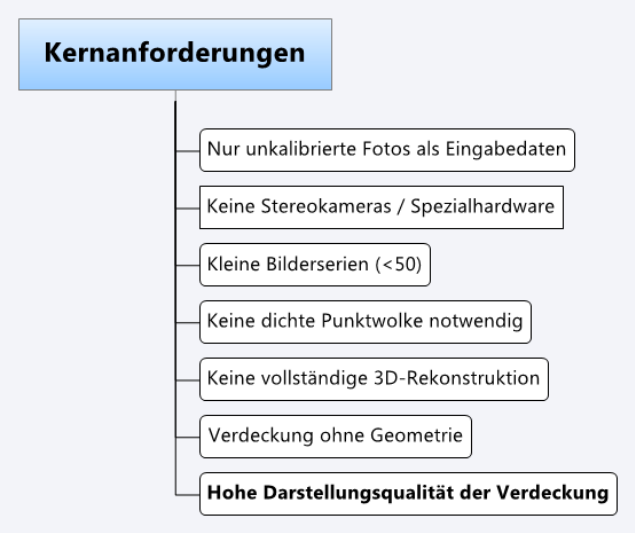


Abbildung 1.10: Kernanforderungen an die vorliegende Arbeit für das Erreichen des Hauptziels.

Modeling und 3D-Rekonstruktion aus Bildern)), die aus einer Serie von Bildern eine abgebildete Szene vollständig in 3D rekonstruieren. Dies setzt jedoch eine verhältnismäßig große Anzahl an Eingabefotos (>100), eine große Abdeckung der abgebildeten Szene und in einigen Verfahren kalibrierte Kameras/Fotos voraus. Abgrenzend dazu soll die hier vorliegende Arbeit mit kleinen Bilderserien (<50) als Eingabedaten arbeiten. Deshalb soll auf eine 3D-Rekonstruktion verzichtet werden. Die authentische Verdeckung soll ohne Geometriedaten und ausschließlich auf Basis von *Image-Based-Rendering*-Verfahren realisiert werden. Dennoch soll eine hohe Darstellungsqualität gesichert sein. Weiterführend bieten Bilderwelten die Möglichkeit unkalibrierte Bilderserien als Eingabedaten zu nutzen. Folglich sollen in dieser Arbeit keine kalibrierten Kameras (und Fotos), keine Stereokameras und keine sonstige Spezialhardware vorausgesetzt werden. Einige bestehende Arbeiten können eine authentische Verdeckung auch ohne 3D-Geometrie auf Basis der 3D-Punktwolke einer Bilderwelt realisieren. Dies setzt jedoch eine ausgeprägte, dichte Punktwolke voraus, die die abgebildete Szene (fast) vollständig widerspiegelt. In dieser Arbeit soll eine flächendeckende, dichte Punktwolke nicht vorausgesetzt werden. *Abb. 1.10* zeigt eine Übersicht der zusammengetragenen Kernanforderungen.

1.4 Teilschritte dieser Arbeit

In Absatz *1.1* (*Motivation*) wurde bei der Definition der *Augmentierungsschwerpunkte* erläutert, dass nur das Zusammenspiel aller fünf Augmentierungsschwerpunkte eine authentische Augmentierung von Bilderwelten ermöglicht. Schlussfolgernd wurde erklärt, dass für eine authentische Verdeckung alle weiteren Augmentierungsschwerpunkte berücksichtigt werden müssen, da sonst (trotz korrekter Verdeckungsdarstellung) *visuelle Störungen* den homogenen Gesamteindruck der Bilderwelt verhindern und somit das Ziel der authentischen Verdeckung nicht erreicht werden kann. Folgerichtig müssen alle Augmentierungsschwerpunkte in der vorliegenden Arbeit Berücksichtigung finden. An dieser Stelle wird der Konzeption etwas vorgegriffen. Dies dient dazu, notwendige Begriffe einzuführen, das Verständnis dieser Arbeit zu erleichtern und für die Kapitel *2* und *3* zu motivieren. Für das Erreichen des Hauptziels dieser Arbeit werden vier Teilschritte definiert, deren Ziele und Funktionen nachfolgend beschrieben werden:

1. Datenaufbereitung,
2. Teilrekonstruktion der Bilderweltenszene,
3. Authentische Darstellung der Bilderwelten,
4. Authentische Darstellung der virtuellen Objekte.

1.4.1 Datenaufbereitung

Notwendige Daten für den Aufbau einer Bilderwelt beinhalten Kameraparameter sowie Daten einer 3D-Punktwolke und werden durch *Structure-from-Motion*-Algorithmen gewonnen [Sna10b]. Diese werden im weiteren Text als *Rohdaten* bezeichnet. Detaillierte Erläuterungen der Rohdaten sowie Generierung und Aufbau von Bilderwelten finden sich in Absatz *2.6* (*Bilderwelten*). Die Rohdaten sowie die zu den Kameras dazugehörigen Bilddaten müssen bereitgestellt werden, da diese Daten die Grundlage des gesamten Konzeptes bilden. Neben der Datenbereitstellung müssen Mechanismen zur Analyse und Bewertung der Datenzuverlässigkeit der Kameraparameter und der Punktwolke erfolgen, da fehlerhafte oder ungenaue Daten zu verzerrten Ergebnissen der weiteren

Teilschritte führen. Mithilfe der aufbereiteten Rohdaten lässt sich ein Maßstab für die räumliche Abmessung der Bilderweltenszene ermitteln. Dieser ist im Rahmen des Augmentierungsschwerpunktes *Transformationsanpassung* für eine geometrische Anpassung der integrierten virtuellen Objekte notwendig.

Zusätzlich müssen die Bildinhalte der Fotos so aufbereitet werden, dass die Fotos als eine homogene Szene in der Bilderwelt dargestellt werden können. Dies entspricht dem Augmentierungsschwerpunkt *Aufbereiten des Bildmaterials*.

1.4.2 Teilrekonstruktion der Bilderweltenszene

Eine Bilderwelt beinhaltet zwar in einer 3D-Welt räumlich angeordnete Bilder sowie eine dreidimensionale Punktwolke der Bilderweltenszene, jedoch bleiben die Bildinhalte in ihrer Darstellung zweidimensional. Somit müssen für eine authentische Verdeckung die Tiefeninformationen der Bilderweltenszene gewonnen werden. Dies erfolgt über eine Teilrekonstruktion der Szene. Hierbei ist die Form der Rekonstruktion abhängig von dem Teilziel, das erreicht werden soll. Bei Teilziel *1* reicht eine Klassifizierung in Bezug auf eingebettete virtuelle Objekte aus. Die real abgebildeten Objekte werden in Vordergrund- und Hintergrundobjekte unterteilt. Vordergrundobjekte stellen dementsprechend *Occluder* dar. Hierfür reichen Tiefeninformationen (des Kamerakoordinatensystems der entsprechenden Kamera) der real abgebildeten Objekte aus. Tiefeninformationen des Kamerakoordinatensystems einer Kamera der Bilderwelt (also Tiefeninformationen aus der Sicht der Kamera im eigenen Koordinatensystem) werden im weiteren Text als *Kameratiefe* bezeichnet. Gewonnene Kameratiefeninformationen werden in einer *Depth-Map* gespeichert. Genauere Erläuterungen zu Depth-Maps folgen in Absatz *2.5* (*Depth-Maps*).

Bei der freien Navigation (Teilziel *2*) reicht eine Klassifizierung der real abgebildeten Objekte nicht aus. Visualisierungen müssen uneingeschränkt aus kameraunabhängigen Positionen innerhalb der 3D-Welt betrachtet werden können. Hierfür muss die annähernd vollständige Kameratiefe der Fotos (also die annähernd vollständige Rekonstruktion der Tiefe des Bildinhaltes) *pixelgenau* (d. h. für jedes Pixel einen individuell zu ihm zugehörigen Wert) gewonnen werden, damit im folgenden Teilschritt das Bild als *3D-Bilddarstellung* gerendert werden kann und somit auch bei freier Navigation die Darstellung einer augmentierten Bilderwelt ohne visuelle Störungen und mit authentischen Verde-

ckungen ermöglicht werden kann. Das bedeutet, die erstellte Depth-Map eines Fotos soll (annähernd) vollständig und pixelgenau die Tiefeninformationen des abgebildeten Szenenausschnittes beinhalten.

Die gewonnenen Tiefeninformationen stellen die Grundlage für den kommenden Teilschritt *Authentische Darstellung der Bilderwelten* dar. Zusätzlich ist er Grundlage für den Augmentierungsschwerpunkt *Verdeckung*, kann aber auch bei den anderen Augmentierungsschwerpunkten eine Rolle spielen.

1.4.3 Authentische Darstellung der Bilderwelten

Dieser Teilschritt soll das Darstellen der Bilderwelten so realisieren, dass eingebettete virtuelle Objekte authentisch verdeckt werden. Mithilfe der gewonnenen Tiefeninformationen aus Teilschritt *Teilrekonstruktion der Bilderweltenszene* kann entschieden werden, welche Bildsegmente der Fotos einer Bilderwelt ein eingebettetes virtuelles Objekt verdecken. Zusätzlich soll bei freier Navigation virtuelle Objekte der Bilderwelt weiterhin authentisch verdeckt werden. Hierfür müssen die zweidimensionalen Bilder als *3D-Bilddarstellungen* gerendert werden, damit uneingeschränkt aus beliebigen Positionen die Darstellung einer augmentierten Bilderwelt ohne visuelle Störungen und mit authentischen Verdeckungen ermöglicht werden kann. Für die Umsetzung dieser Anforderungen sollen geeignete *Image-Based-Rendering*-Mechanismen konzipiert werden.

1.4.4 Authentische Darstellung der virtuellen Objekte

Für eine authentische Augmentierung sind neben einer korrekten Verdeckung Mechanismen für die Augmentierungsschwerpunkte *Render-Adaption* und *Bildbasierte Beleuchtung* erforderlich. Sie bilden nicht den Kern dieser Arbeit, müssen aber für das Erreichen des Hauptziels berücksichtigt werden.

Aufbauend auf den grundlegenden Kenntnissen (Kapitel *2*) und den analysierten Stand der Technik (Kapitel *3*) wird in Kapitel *4* eine verfeinerte und technisch versierte Zielstellung sowie eine genaue Definition weiterer Anforderungen und Voraussetzungen für die einzelnen Teilschritte zusammengetragen.

1.5 Anwendungsgebiete der Ergebnisse der Arbeit

Es existieren für das Erstellen von dreidimensionalen Planungsvisualisierungen viele VR-basierte Planungs- und 3D-Modellierungsanwendungen. Hierbei muss der Nutzer die Planung händisch erzeugen/modellieren, was Kenntnisse über 3D-Geometrie und über Transformationen im 3D-Raum voraussetzt. Des Weiteren wird der Szeneninhalt nicht durch reale Anteile repräsentiert. Eine automatische Erzeugung einer AR-basierten dreidimensionalen Visualisierung stellen Bilderwelten dar. Durch das Einbetten virtueller 3D-Objekte können augmentierte Bilderwelten als alternative, moderne Planungsvisualisierungen eingesetzt werden. Dies erfordert eine authentische Augmentierung für die Glaubwürdigkeit und Akzeptanz des Nutzers. Das in dieser Arbeit entwickelte System soll eine authentische Verdeckung (und folglich eine authentische Augmentierung) integrierter 3D-Objekte in Bilderwelten für den Zweck der glaubwürdigen Planungsvisualisierung bewerkstelligen.

Der Einsatz der Planungsvisualisierung durch Bilderwelten umfasst Bereiche wie Architekturplanungen, Visualisierung von Lager- und Industriehallen, Inneneinrichtungen- und Innenarchitekturplanungen, Stadt- und Landschaftsplanungen bis hin zu dem Einsatz als Test-, Modellierungs- und Planungstool ergonomischer Arbeitsräume, wie Werkstätten oder OP-Räume.

Schlussfolgernd können Einsatzbereiche von augmentierten Bilderwelten überall dort erschlossen werden, bei denen eine visuelle Vorschau einer realen Abbildung bzw. Situation auf eine zukünftige Sicht generiert werden soll.

1.6 Aufbau der Arbeit

Alle Kapitel dieser Arbeit (sowie die Unterkapitel des Konzeptes) beinhalten zu Beginn einen Absatz, der einen *Überblick* über das (Unter-)Kapitel gibt und enden mit einem Absatz *Zusammenfassung und Diskussion*. Des Weiteren soll an dieser Stelle noch einmal auf die in *Tab. 1* erklärten Texthervorhebungen dieser Arbeit verwiesen werden. Die vorliegende Arbeit ist wie folgt aufgebaut:

Kapitel 2 – Grundlagen und Begriffserläuterungen: In diesem Kapitel werden die Grundlagen erläutert, die für die Konzeption zur Erreichung des in Kapitel *1* definierten Hauptziels unter Berücksichtigung der zusammengetragenen Kernanforderungen und unter Einsatz der spezifizierten Teilschritte vorausgesetzt werden. Dabei ist das Kapitel nach dem Bottom-Up-Prinzip aufgebaut. Kenntnisse der einzelnen Unterkapitel werden für nachfolgende Unterkapitel vorausgesetzt. Dementsprechend werden zu Beginn elementare Grundlagen erläutert, die die Basis für weitere tiefergehende Grundlagen folgender Unterkapitel bilden.

Kapitel 3 – Heutiger Stand der Technik und Wissenschaft: Mithilfe der definierten Ziele, Kernanforderungen und Teilschritte aus Kapitel *1* sowie der in Kapitel *2* zusammengetragenen Grundlagen und Begriffe werden in diesem Kapitel aktuelle Forschungsarbeiten und Projekte mit verwandten (Teil-)Zielstellungen erläutert. Hierbei ist das Ziel, bestehende Verfahren und Mechanismen zu analysieren und auf Einsetzbarkeit in dieser Arbeit zu prüfen. Folglich werden Abgrenzungen gezogen, inwieweit die vorgestellten Arbeiten zum Einsatz in dieser Arbeit geeignet sind.

Kapitel 4 – Einordnung, erweiterte Anforderungen und verfeinerte Zielsetzung: In Kapitel *1* wurde die Motivation und Herausforderung sowie daraus abgeleitete Augmentierungsschwerpunkte für das authentische Verdecken virtueller Objekte in Bilderwelten erläutert. Daraus ergab sich die Zielstellung und die Kernanforderungen dieser Arbeit. Für eine technisch versierte Spezifizierung der Zielstellung in Hinblick auf die Machbar- und Erreichbarkeit der Teilziele wurden in Kapitel *2* grundlegende Thematiken und damit verbundene Begriffe erläutert. Aufbauend auf den Grundlagen wurden in Kapitel *3* aktuelle Arbeiten mit verwandter Zielstellung analysiert und individuelle Abgrenzungen zu dieser Arbeit (und ihrer Zielstellung) gezogen. Mithilfe der Erkenntnisse der ersten drei Kapitel wird in diesem Abschnitt eine Abgrenzung und Einordnung dieser Arbeit sowie eine detaillierte und technisch versierte Zielstellung mit definierten (erweiterten) Anfordungen und Grenzen zusammengetragen.

Kapitel 5 – Konzeption: In diesem Kapitel wird das Konzept für das Erreichen einer authentischen Verdeckung in augmentierten Bilderwelten unter Berücksichtigung der in Kapitel *1* zusammengetragenen Kernanforderungen sowie der in Kapitel *4* zusammengetragenen (erweiterten) Anfordungen und Grenzen beschrieben. Somit stellt das Kapitel den Kern dieser Arbeit dar. Das Kapitel unterteilt sich in die in Kapitel *1* erläuterten Teilschritte als Unterkapitel *5.2* bis *5.5*. Folglich wird das Konzept jeden einzelnen Teilschrittes erläutert. Alle Unterkapitel (bis zur dritten Ebene) beinhalten zu Beginn einen Absatz, der einen *Überblick* über das Unterkapitel gibt und enden mit einem Absatz *Zusammenfassung und Diskussion*.

Kapitel 6 – Technische Umsetzung der Konzeption: In diesem Kapitel wird die technische Umsetzung der in der Konzeption erörterten Verfahren der Teilschritte für die authentische Verdeckung virtueller 3D-Objekte in Bilderwelten beschrieben. Die Teilschritte *Teilrekonstruktion der Bilderweltenszene* und *Authentische Darstellung der Bilderwelten* stellen den Kern dieser Arbeit dar. Deshalb wurden vom Autor während der Implementierungsphase das Hauptaugenmerk auf die Verfahren dieser beiden Teilschritte gelegt und weiterführende Verfahren der anderen Teilschritte zum Teil vernachlässigt oder als gegeben vorausgesetzt. Dementsprechend werden in diesem Kapitel jene weiterführenden Verfahren nicht tiefer erläutert und auf die entsprechenden Veröffentlichungen verwiesen.

Kapitel 7 – Evaluierung der Ergebnisse: Dieses Kapitel zeigt, analysiert und bewertet die Ergebnisse des zusammengetragenen Konzeptes und der dazugehörigen technischen Umsetzung. Aufbauend auf den Augmentierungsschwerpunkten wurde das Konzept sowie die dazugehörige Umsetzung in vier Teilschritte zerlegt. Die hier beschriebene Evaluierung konzentriert sich auf den Augmentierungsschwerpunkt *Verdeckung* und die verbundenen Teilschritte und ihre Verfahren. Für das Gewinnen der Tiefeninformationen des abgebildeten Szenenausschnittes der Kamerabilder wurden drei Verfahren entwickelt. Die Ergebnisse sowie deren Gegenüberstellung wird in diesem Kapitel beschrieben. Weiterführend werden auf der Basis der gewonnenen Tiefeninformationen die Ergebnisse des *Sliced-Image-Renderings* evaluiert.

Kapitel 8 – Zusammenfassung und Ausblick: Im abschließenden Kapitel werden die Arbeit und die aus ihr hervorgegangenen Erkenntnisse zusammengefasst und ein Ausblick auf weiterführende Forschungsaktivitäten in der Thematik gegeben.

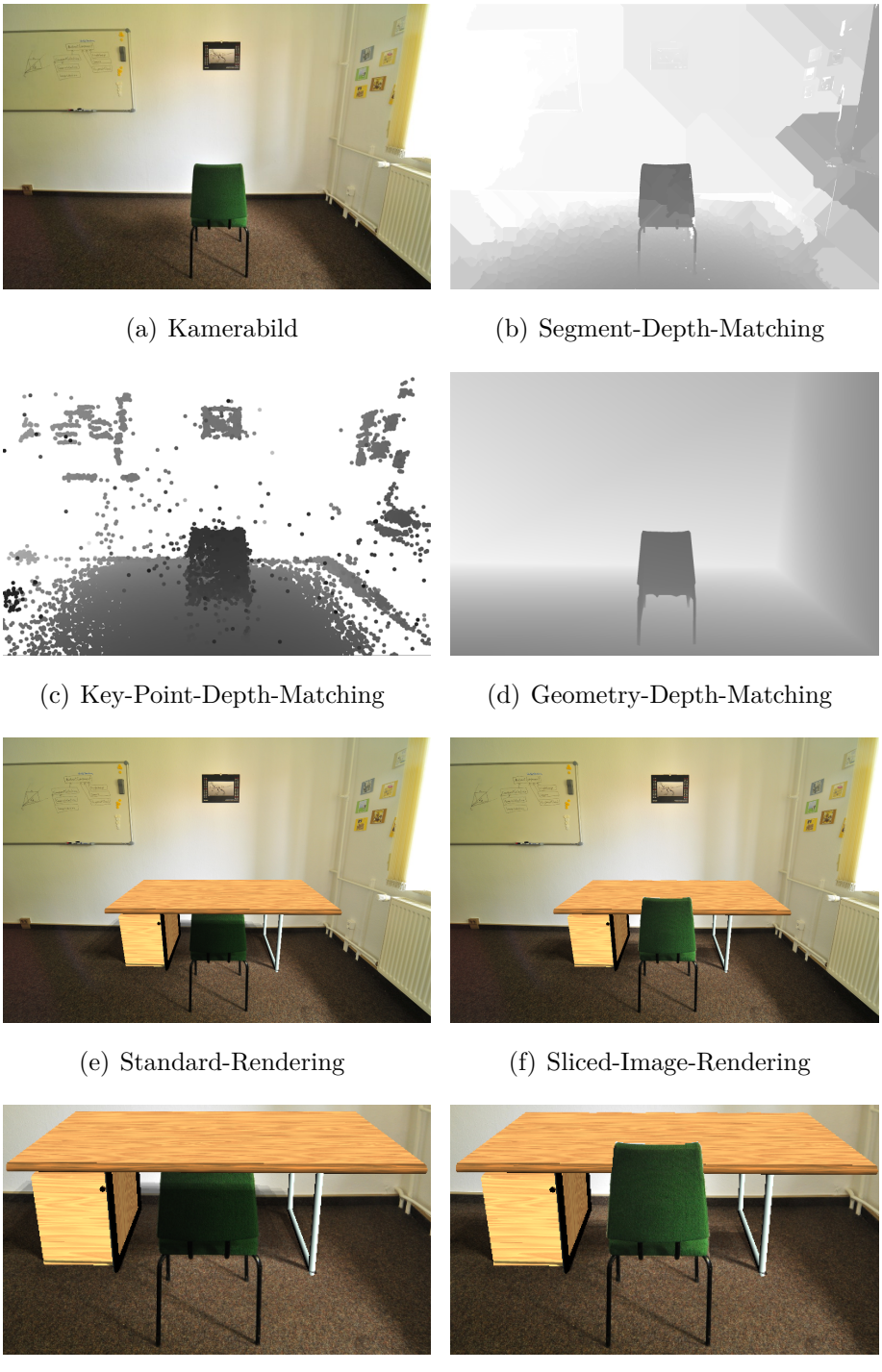
1.7 Hauptergebnisse der Arbeit

Dieser Absatz zeigt vorab die Hauptergebnisse der vorliegenden Arbeit. Hierbei wird (noch) nicht auf technische bzw. konzeptionelle Details eingegangen sowie eine Evaluierung der Ergebnisse durchgeführt. Weitere Ergebnisabbildungen und eine ausführliche Evaluierung der Ergebnisse finden sich in Absatz *7* (*Evaluierung der Ergebnisse*). Für das Erreichen der authentischen Verdeckung unter Berücksichtigung der gegebenen Anforderungen werden den abgebildeten Inhalten der Fotos der Bilderwelt Tiefeninformationen zugeordnet. Das Speichern der Tiefeninformationen erfolgt in Depth-Maps. Die Tiefenzuordnung erfolgt auf der Basis der Key-Points (Punkte der Bilderweltenpunktwolke). Für die Tiefenzuordnung werden drei verschiedene Verfahren entwickelt:

- *Segment-Depth-Matching*-Verfahren,
- *Key-Point-Depth-Matching*-Verfahren,
- *Geometry-Depth-Matching*-Verfahren.

Aufbauend auf den gewonnenen Tiefeninformationen und unter Berücksichtigung der Anforderung keine 3D-Rekonstruktion zu betreiben, realisiert das *Sliced-Image-Rendering*-Verfahren die authentische Verdeckung virtueller 3D-Objekte in Bilderwelten. Auf Basis der Tiefeninformationen wird ein Bild innerhalb der Bilderweltenszene in Slices zerlegt. Ein Slice repräsentiert eine Bildregion des Originalbildes. Jedes Slice wird an die für die Region entsprechende Tiefenposition, die in der Depth-Map gespeichert wurde, verschoben.

Abb. 1.11 zeigt anhand einer beispielhaften Innenraumszene eine Gegenüberstellung der erzeugten Depth-Maps der drei entwickelten Depth-Matching-Verfahren. Des Weiteren wird die die Innenraumszene (augmentiert mit einem



(a) Kamerabild (b) Segment-Depth-Matching

(c) Key-Point-Depth-Matching (d) Geometry-Depth-Matching

(e) Standard-Rendering (f) Sliced-Image-Rendering

(g) Standard-R. (Nahaufnahme) (h) Sliced-Image-R. (Nahaufnahme)

Abbildung 1.11: (a) Eingabebild. (b-d) Gegenüberstellung der erzeugten Depth-Maps der drei entwickelten Depth-Matching-Verfahren. (e-h) Innenraumszene augmentiert mit einem virtuellen Tisch: (e+g) Standard-Rendering mit fehlerhafter Verdeckung und (f+h) Sliced-Image-Rendering mit authentischer Verdeckung.

virtuellen Tisch) zum einen mit Standard-Rendering und fehlerhafter Verdeckung und zum anderen mit dem Sliced-Image-Rendering und einer authentischen Verdeckung gegenübergestellt.

1.8 Zusammenfassung und Begriffsübersicht

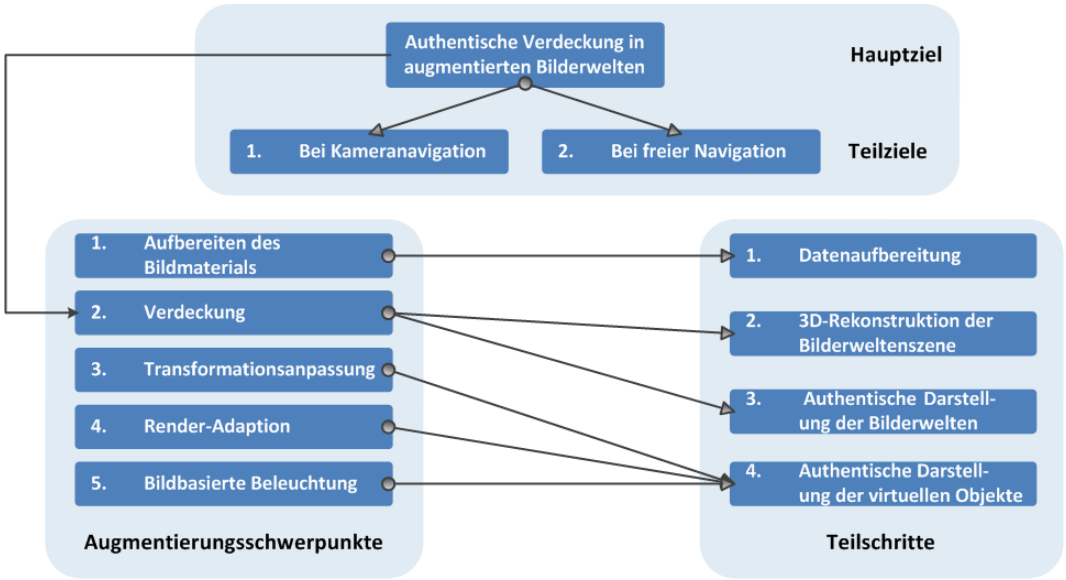


Abbildung 1.12: Begriffsübersicht: Augmentierungsschwerpunkte, Ziele und Teilschritte dieser Arbeit.

In Absatz *1.1* (*Motivation*) wurden fünf *Augmentierungsschwerpunkte* für das visuelle Verschmelzen virtueller und realer Anteile eingeführt. Nur im Zusammenspiel ist eine authentische Integration virtueller Objekte in Bilderwelten realisierbar. In Absatz *1.2* (*Zielsetzung der Arbeit*) wurde das *Hauptziel* definiert und daraus zwei Teilziele abgeleitet. Da eine authentische Integration alle Augmentierungsschwerpunkte erfordert, müssen neben der *Verdeckung* die anderen Schwerpunkte mit berücksichtigt werden. Unter diesem Gesichtspunkt wurden für das Erreichen der Zielstellung dieser Arbeit in Absatz *1.4* (*Teilschritte dieser Arbeit*) vier Teilschritte spezifiziert. In *Abb. 1.12* werden die eingeführten Begriffe und ihre Zusammenhänge untereinander schematisch dargestellt. Des Weiteren wurden einige Begriffe eingeführt.

Als *Bilderweltenszene* (oder kurz *Szene*) wird die abgebildete reale Umgebung der Bildinhalte einer Bilderwelt genannt. In den Fotos dargestellte einzelne reale

Objekte werden als *Abgebildete Objekte* bezeichnet. Die (allgemeine) visuelle Überlagerung von virtuellen Objekten durch reale Bildanteile wird *Verdeckung* genannt. Eine visuell korrekte Überlagerung wird als *Authentische Verdeckung* bezeichnet. Reale, abgebildete Objekte (Bildsegmente) der Fotos, die für die visuelle Überlagerung genutzt und so virtuelle Objekte verdecken, werden als *Occluder* bezeichnet.

Kapitel 2

Grundlagen und Begriffserläuterungen

2.1 Überblick

In Absatz *1.4* (*Teilschritte dieser Arbeit*) wurden die Teilschritte (*1. Datenaufbereitung*, *2. Teilrekonstruktion der Bilderweltenszene*, *3. Authentische Darstellung der Bilderwelten*, *4. Authentische Darstellung der virtuellen Objekte*) für das Erreichen des Hauptziels (*Authentische Verdeckung in augmentierten Bilderwelten*) dieser Arbeit definiert. In diesem Kapitel werden die Grundlagen erläutert, die für die Konzeption zur Erreichung des definierten Hauptziels unter Einsatz der spezifizierten Teilschritte vorausgesetzt werden. Dabei ist das Kapitel nach dem Bottom-Up-Prinzip aufgebaut. Das bedeutet, die Kenntnisse der einzelnen Unterkapitel werden für nachfolgende Unterkapitel vorausgesetzt. Dementsprechend werden zu Beginn elementare Grundlagen erläutert, die die Basis für weitere tiefergehende Grundlagen folgender Unterkapitel bilden.

2.2 Repräsentation von Bilddaten

Die spezifizierten Teilschritte dieser Arbeit arbeiten mit den Bilddaten der Fotos. Dafür sind grundlegende Kenntnisse in der Art der Datenrepräsentation sowie deren Repräsentation durch Farbräume notwendig. Im diesem Unterkapitel wird erläutert, wie Bilddaten verwaltet, organisiert und repräsentiert werden. Neben der Art der Datenrepräsentation werden die für diese Arbeit relevanten

Farbräume sowie das Vergleichen und Transformieren von Farben erläutert.

2.2.1 Vektor- und Rasterdaten

In der digitalen Welt existieren verschiedene Arten der Datenrepräsentation von Bildern. In *Vektorgrafik*en wird der Bildinhalt in Form von geometrischen Objekten mit kontinuierlichen Koordinaten repräsentiert und die Rasterung erfolgt erst bei der Darstellung auf einem konkreten Gerät, wie Drucker oder Bildschirm. Im Gegensatz dazu werden *Rasterdaten* durch eine regelmäßige Matrix (mit diskreten Koordinaten) von Pixelwerten beschrieben. Konkret wird das Bild in Zeilen und Spalten zerlegt [BB06a]. Der Farbwert jedes *Bildpunkt*es (*Pixel*), das durch diese Zerlegung vorliegt, wird an seiner Koordinate innerhalb der Matrix abgelegt. Das bedeutet, dass für die digitale Bildverarbeitung die Bilddaten in einem zweidimensionalen Array vorliegen und modifiziert werden können. Es existieren verschiedene Arten von Bildern (Schwarzweiß-, Graustufen-, Farb-, Spezialbilder mit Gleitkommadaten), die unterschiedliche *Pixelformat*e (mit unterschiedlicher Anzahl von Farbkanälen) voraussetzen [BB06a]. Gebräuchliche Byte-basierte Pixelformate besitzen i. d. R. eine Farbtiefe von 8-Bit pro Kanal (8-Bit Graustufen, 24-Bit RGB, 32-Bit RGB+Alpha) [Wag06]. Es existieren weitere Formate mit anderen Farbtiefen pro Kanal, die in dieser Arbeit jedoch keine Rolle spielen. Spezialbilder (z. B. *HDR*) mit *Gleitkomma*daten arbeiten häufig mit einer Farbtiefe von 32-Bit (Float-basiert, einfache Genauigkeit) oder 64-Bit (Double-basiert, doppelte Genauigkeit) pro Kanal [War12].

In dieser Arbeit spielen im Rahmen der Repräsentation der Bilddaten Vektordaten keine Rolle. Daher wird im weiteren Text immer von Rasterdaten ausgegangen.

2.2.2 Farbräume

Für die Analyse und Verarbeitung von farbigen Bilddaten spielt das verwendete *Farbmodell* sowie alle darin enthaltenen Farben (der so genannte *Farbraum*) eine wichtige Rolle [BB06b]. Da auch in dieser Arbeit Farbbilder analysiert und be-/verarbeitet werden, wird in diesem Absatz ein kurzer (nicht vollständiger) Abriss zur Thematik Farbmodelle und Farbräume gegeben.

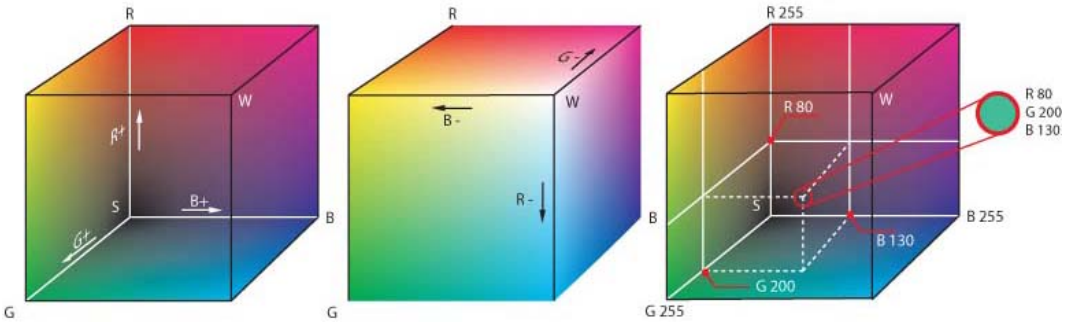
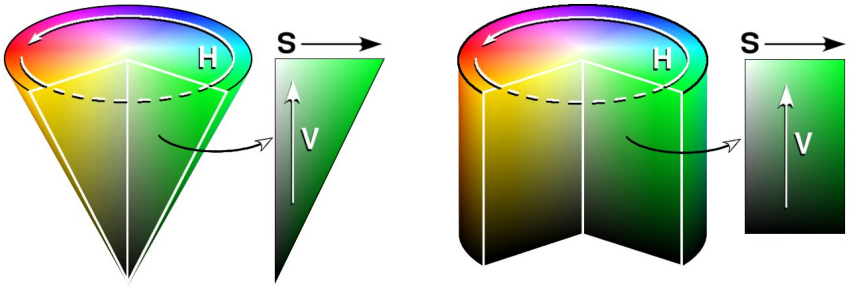


Abbildung 2.1: Der RGB-Farbraum kann als Einheitswürfel dargestellt werden [Wik12c]. Jede Achse stellt eine der drei primären Farben dar.

Der *RGB*-Farbraum stellt das in Farbbildschirmen und Kameras verwendete Farbsystem dar. Er nutzt ein kartesisches Koordinatensystem, indem jede Achse eine der drei primären Farben (rot, grün, blau) abbildet. Jede Farbe wird durch das Addieren gewichteter RGB-Komponenten definiert. Der RGB-Farbraum kann durch einen Einheitswürfel veranschaulicht werden. Die Hauptdiagonale des Würfels bildet die Graustufen-Funktion ab [CM04b]. Es existieren verschiedene Vertreter des RGB-Farbmodells (z.B. *Standard-RGB/sRGB* oder Adobe-RGB), die sich in der Definition der drei Primärfarben unterscheiden. Der sRGB-Farbraum ist in [IEC99] spezifiziert und definiert exakt die drei Primärfarben, den Weißpunkt, die Umgebungsbeleuchtung und die Gammawerte für eine Gammakorrektur. Jedoch eignet sich die Metrik des RGB-Farbraumes nur bedingt für das manuelle Be- und Verarbeiten von Farben. Für das Auswählen oder Verändern des Farbtons, der Farbsättigung oder der Helligkeit müssen die drei Komponenten kombiniert werden. Dies ist darin begründet, dass sich Farbton, Farbsättigung und Helligkeit gleichzeitig bei jeglicher Verschiebung eines Farbpunktes im RGB-Raum ändern [BB06b].

Eine Alternative ohne diesen Nachteil stellt der *HSV*-Farbraum dar, bei dem subjektiv wichtige Farbeigenschaften explizit dargestellt werden. Die Farbeigenschaften werden durch die Komponenten *Hue*, *Saturation* und *Value* repräsentiert. Das HSV-Farbmodell kann in verschiedenen geometrischen Körpern visualisiert werden, z. B. als sechseckige Pyramide, Kegel, Zylinder oder Halbkugel [CM04b] [BB06b] [Wik12b]. *Abb. 2.2* zeigt den Aufbau des HSV-Modells anhand eines Kegels und Zylinders. Die vertikale Achse entspricht dem V-Wert, die horizontale Achse dem S-Wert und der Winkel zwischen beiden

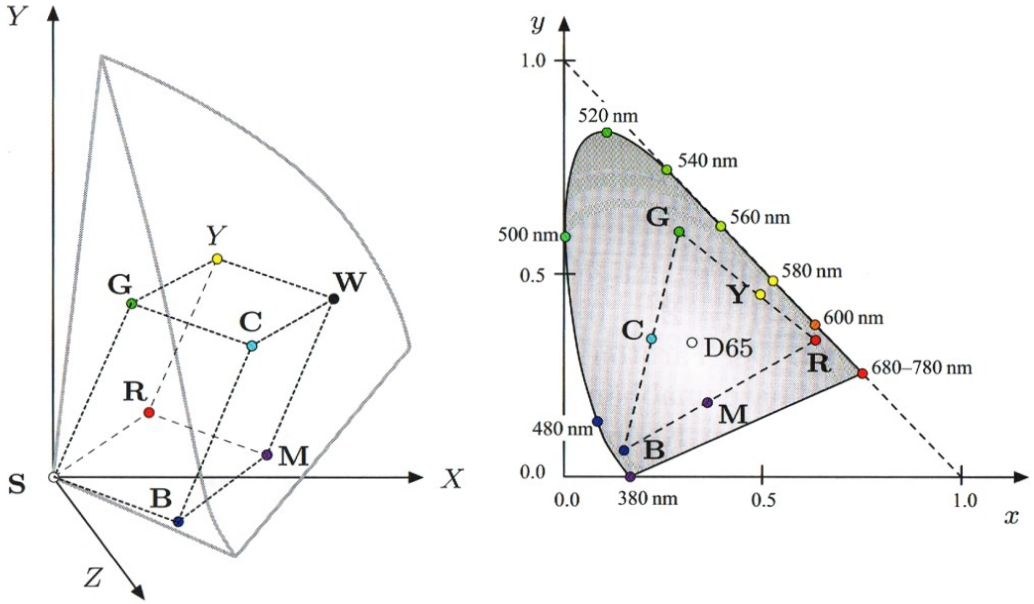


(a) HSV-Farbraum als Kegel (b) HSV-Farbraum als Zylinder

Abbildung 2.2: Visualisierungen des HSV-Modells [Wik12b].

Achsen entspricht dem H-Wert. Im Gegensatz zum RGB-Farbmodell kann eine Farbdefinition in Hinblick auf die menschliche Farbwahrnehmung intuitiver und einfacher erstellt werden, da *Farbton*, *Farbsättigung* und *Helligkeit*swert getrennt voneinander definiert werden. Neben HSV existieren noch abgewandelte Farbmodelle, wie *HSB*, *HSI* und *HSL*.

Der RGB- und HSV-Farbraum beziehen sich auf physische Eigenschaften von Ausgabegeräten. Für eine geräteunabhängige Beschreibung von Farben werden so genannte *colorimetrische* Farbräume benötigt. Der *XYZ*-Farbraum (auch *CIEXYZ*-Farbraum) wurde von der Internationalen Beleuchtungskommission (*CIE*) 1931 standardisiert und kann als Grundlage aller colorimetrischen Farbräume gesehen werden. Er stellt eine Relation zwischen dem menschlichen Farbempfinden und einer mathematischen Konstruktion her, sodass alle wahrnehmbaren Farben abgedeckt werden können. Der XYZ-Farbraum wird durch drei imaginäre Primärfarben X, Y, Z aufgespannt, die wiederum selbst nicht wahrgenommen werden können. Hierbei repräsentieren die Y-Koordinate die Helligkeit und die XZ-Koordinaten die Farbigkeit. Alle sichtbaren Farben liegen in einem Zuckerhut-ähnlichen 3D-Teilraum des Koordinatenraumes. Abb. 2.3(a) verdeutlicht diese mathematische Beschreibung. Zusätzlich wird der RGB-Würfel aus Abb. 2.1 in den Zuckerhut-förmigen 3D-Teilraum projiziert. Somit wird deutlich, dass der RGB-Farbraum nicht alle wahrnehmbaren Farben abdecken kann. Für die Darstellung von Farbtönen in einem zweidimensionalen Koordinatensystem definiert die CIE die Variablen x und y als „Farbgewichte“ [BB06b]. Abb. 2.3(b) zeigt das zweidimensionale CIE xy-Diagramm (auch *Normfarbtafel* genannt) und entspricht einem horizontalen Schnitt durch den XYZ-Raum



(a) XYZ-Farbraum und projizierter RGB-Würfel [BB06b]

(b) xy-Diagramm (Normfarbtafel) [BB06b]

Abbildung 2.3: (a) Der XYZ-Farbraum: Die Y-Koordinate repräsentiert die Helligkeit und die XZ-Koordinaten die Farbigkeit. Alle sichtbaren Farben liegen in einem Zuckerhut-förmigen 3D-Teilraum des Koordinatenraumes. (b) Das CIE xy-Diagramm (Normfarbtafel) als Ergebnis der Projektion der sichtbaren Farben des Zuckerhut-förmigen 3D-Teilraums. Die Normfarbtafel entspricht einem horizontalen Schnitt durch den XYZ-Raum an der Höhe $Y = 1$. Somit sind alle sichtbaren Farbtöne enthalten, jedoch keine Helligkeitsinformationen. Der RGB-Farbraum kann darin mit beliebiger Definition seiner RGB-Primärfarben alle Farben innerhalb des durch die Primärfarben aufgespannten Dreiecks darstellen. D65 markiert die xy-Koordinate der Normlichtart [BB06b].

an der Höhe $Y = 1$. Diese Projektion lässt sich durch *Formel 2.1* berechnen.

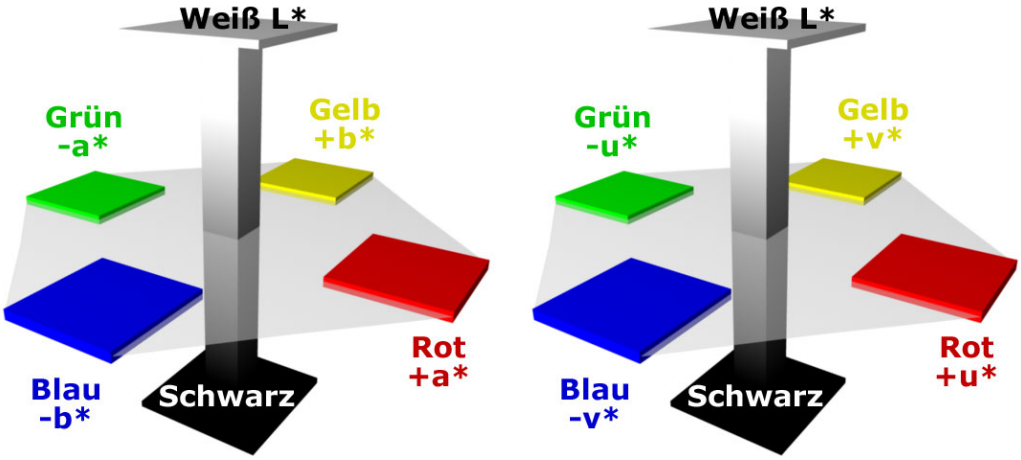
$$x = \frac{X}{X+Y+Z} \quad , \quad y = \frac{Y}{X+Y+Z} \quad , \quad Y = 1 \tag{2.1}$$

Formel 2.1: Projektion der sichtbaren Farben des Zuckerhut-förmigen 3D-Teilraums auf das CIE xy-Diagramm (Normfarbtafel) [BB06b].

Somit sind alle sichtbaren Farbtöne (mit Wellenlängen von ca. 380-780 Nanometer) enthalten, jedoch keine Helligkeitsinformationen. Der RGB-Farbraum kann darin mit beliebiger Definition seiner RGB-Primärfarben alle Farben innerhalb des durch die Primärfarben aufgespannten Dreiecks darstellen. Die Normlichtart *D65* definiert die Farbtemparatur ca. 6500°K und simuliert somit typische Tageslichtbeleuchtung. Diese Normbeleuchtung wurde von der CIE spezifiziert mit dem Ziel eine objektive Messung von Farben in der physischen Realität zu ermöglichen [BB06b].

Der XYZ-Farbraum sowie das abgeleitete xy-Diagramm bringen den Nachteil mit sich, dass geometrische Abstände innerhalb des Farbraumes unterschiedlich wahrgenommen werden. Zum Beispiel sorgen kurze Strecken im Magenta-Raum für eine große visuelle Änderung, während lange Strecken im Grün-Bereich relativ ähnliche Farbtöne erzeugen. Aufgrund dieser *Ungleichabständigkeit* der Farben entwickelte die CIE 1976 zwei weitere Farbraumsysteme (LAB- und LUV-Farbraum) mit dem Ziel, entsprechend der menschlichen Wahrnehmung eine (annähernd) lineare *Gleichabständigkeit* innerhalb der Farbräume zu realisieren. Gleiche Abstände in diesen Farbräumen sollen also gleiche empfindungsgemäße Abstände zwischen menschlichen Farbreizen entsprechen [dlC11].

Der *LAB*-Farbraum (auch *CIELAB* oder *L*a*b**-Farbraum) ist eine Adaption des XYZ-Farbraumes mit der Farbunterschiede aufgrund der linearen Gleichabständigkeit und Geräteunabhängigkeit objektiv und numerisch bestimmt werden können. Analog zu dem XYZ-Farbraum repräsentieren zwei Achsen den Farbton (a* und b*) und eine Achse die Helligkeit (L*). Achse L* liegt in einem Wertebereich von 0 für reines Schwarz bis 100 für reines Weiß. Achse a* repräsentiert die Rot-Grün-Achse bei der negative Werte grün und positive Werte rot dargestellt werden. Achse b* repräsentiert die Gelb-Blau-Achse (negative Werte gelb, positive Werte rot). Der Aufbau des LAB-Farbraumes ist



(a) LAB-Farbraum [Sch12a] (b) LUV-Farbraum [Sch12b]

Abbildung 2.4: Aufbau der gleichabständigen Farbräume (a) L*a*b* und (b) L*u*v*. An den Rändern liegen die bunten Farben und in der Mitte befindet sich unbuntes Grau. Neben dieser prismaförmigen Darstellung existieren auch kreiskegelförmige Varianten. Somit ergibt sich an einer definierten Position auf der L*-Achse aus den farbgebenden Achsen ein Farbkreis.

in Abb. 2.4(a) dargestellt. Somit kann jede Farbe exakt beschrieben werden. Die Berechnung erfolgt unter Berücksichtigung eines Standardlichts (z.B. D65). Dementsprechend werden die Farbwerte unter einer definierten physikalischen Bedingung berechnet [Sch12a].

Ebenfalls eine transformierte Variante des XYZ-Farbraumes, jedoch unter Verwendung eines anderen Formelsatzes, ist der *LUV*-Farbraum (auch *CIE-LUV* oder *L*u*v**-Farbraum). Der Unterschied zu L*a*b* besteht darin, dass L*u*v* linear vom XYZ-Farbsystem transformiert wurde und deshalb einen verkleinerten Grün- und vergrößerten Blaubereich aufweist. Farbabstände werden analog wie bei L*a*b* berechnet. Die Bezeichnung der Achsen a* und b* lauten hier u* und v* [Sch12b]. Eine genaue mathematische Beschreibung der Umrechnung vom XYZ-System in den LAB- und LUV-Farbraum findet sich in [BNS04], [BB06b] sowie in der ISO 13655 [ISO96].

2.2.3 Vergleichen und Transformieren von Farben

Für das objektive und messbare Beurteilen von Unterschieden oder Ähnlichkeiten zwischen beliebigen Farben, muss ein Farbraum eine (annähernde) Gleichabständigkeit aller auftretenden Farben aufweisen. Dies entspricht der menschlichen Farbwahrnehmung. Die Farbmetrik des Lab- und der Luv-Farbraumes besitzt die Eigenschaft der (annähernden) Gleichabständigkeit aller auftretenden Farben. Somit eignen sich diese Farbräume zum Vergleichen von Farben. Liegen die Daten in anderen Farbräumen vor (wie digitale Fotos, die i.d.R. den sRGB-Farbraum verwenden), müssen gegebene Farbräume in andere Farbräume transformiert werden. Die meisten gängigen Farbräume können durch eine lineare Koordinatentransformation in den XYZ-Raum transformiert werden und umgekehrt. Somit kann über das XYZ-System Farbräume in andere Farbsysteme hin und her transformiert werden. Sollen Farbwerte der Pixel von Digitalbildern verglichen werden, so empfiehlt es sich, jeden Farbwert der Pixel von seinem sRGB-Farbraum in den LUV-(oder LAB-)Farbraum zu transformieren. Dies erfolgt in drei Schritten.

1. Konversion des vorliegenden sRGB-Farbwertes in lineare RGB-Werte durch Umkehrung der Gammakorrektur,

2. Transformation der linearen RGB-Werte in XYZ-Werte durch Multiplikation mit einer $3x3$-Abbildungsmatrix,

3. Transformation der XYZ-Werte in LUV-Werte nach ISO 13655 [ISO96].

Detaillierte mathematische Formelsätze sowie dazugehörige CIE-Farbwert-Tabellen für diese drei Schritte finden sich in [BB06b].

Der *Farbabstand*, also die Größe des empfindungsgemäßen Unterschiedes zwischen zwei Farben [DIN09], ist ein messbarer Wert, der für das objektive Vergleichen von Farben verwendet wird. Er entspricht der Distanz zwischen zwei Farbpunkten im dreidimensionalen Farbraum und wird als euklidischer Abstand ΔE berechnet:

$$\Delta E = \sqrt{(L_1^* - L_2^*)^2 + (u_1^* - u_2^*)^2 + (v_1^* - v_2^*)^2} \tag{2.2}$$

Formel 2.2: LUV-Farbabstandsberechnung

Nach [HN12] entscheidet der Anwendungsfall über die Größe des Grenzwertes von ΔE, der zwei untersuchte Farben als gleiche (mit einer noch tolerierbaren Farbabweichung) oder unterschiedliche Farben einordnet. Zwei typische Grenzwerte haben sich in der Forschung etabliert. Einige Forschungsarbeiten sagen $\Delta E = 1.0$ ist der kleinste noch wahrnehmbare Farbunterschied für das menschliche Sehsystem. Andere Arbeiten belegen, dass bei digitalen Fotos kleinere Werte als $\Delta E = 2.5$ vom menschlichen Auge nicht wahrgenommen werden [HN12].

Damit eine korrekte Transformation der sRGB-Werte in XYZ-Werte (und somit in L*u*v*-Werte) erfolgen kann, muss die Spezifizierung des Weißpunktes berücksichtigt werden. Der Weißpunkt im sRGB-Farbraum ist typischerweise als D65-Weißpunkt spezifiziert. Es gibt aber auch Systeme, bei denen andere Weißpunkte (z.B. Normbeleuchtung D50) verwendet werden. Dies ändert auch die Abbildungsmatrix bei der Transformation [BB06b].

2.3 Bildverarbeitung und Bildanalyse

Mithilfe der Bilddaten und ihrer Farbcharakteristiken (siehe Absatz *2.2* (*Repräsentation von Bilddaten*)) können weitere Informationen aus Bildern extrahiert werden. Aus dem breiten Gebiet der Bildverarbeitung und Bildanalyse soll sich in diesem Unterkapitel auf die für diese Arbeit relevanten Bereiche

- Lineare Filter,
- Kanten- und Feature-Point-Detektion, sowie
- Segmentierung

beschränkt werden.

2.3.1 Lineare Filter

In der digitalen Bildverarbeitung gibt es u. a. *Punktoperation*en und Operationen im Ortsbereich. Während Punktoperationen rein bildpunktbezogen arbeiten, nutzen Operationen im Ortsbereich Informationen von benachbarten Pixeln. Eine sehr wichtige Klasse der ortsbezogenen Operationen stellen die digitalen *Filter* dar. Dabei werden lineare und nichtlineare Filter unterschieden [NFH07].

Filter werden als linear bezeichnet, wenn sie die Pixelwerte der Filterregion in linearer Form, also durch lineare mathematische Operatoren verknüpfen. Dabei ergibt sich der Bildpunkt $p_A(x, y)$ des Ausgabebildes P_A durch eine gewichtete, additive Verknüpfung des Pixels $p_E(x, y)$ des Eingabebildes P_E mit benachbarten Bildpunkten. Eine Matrix, auch als Faltungsmaske oder *Filterkern* beschrieben, bestimmt dabei die Nachbarschaft und die Gewichtung der Nachbarpixel. Diese Operation wird lineare *Faltung* genannt [NFH07].

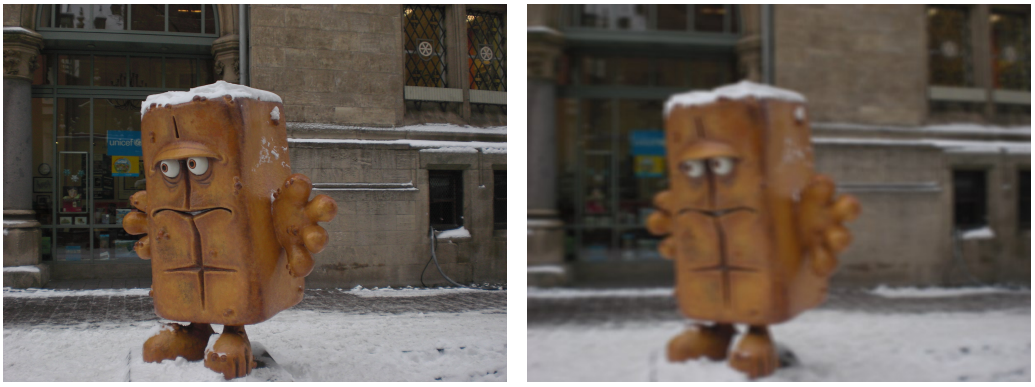
$$H = h(u, v) \tag{2.3}$$

$$k = \frac{m-1}{2} \tag{2.4}$$

$$p_A(x, y) = \frac{1}{m^2} \sum_{u=0}^{m-1} \sum_{v=0}^{m-1} p_E(x + k - u, y + k - v) * h(u, v) \tag{2.5}$$

Formel 2.5: Lineare Faltung des Bildes P_E zu P_A.

Glättungsfilter kennzeichnen sich dadurch aus, dass alle Koeffizienten des Faltungskerns positiv sind. Daher besitzt die lineare Faltungsformel auch einen Normierungsfaktor $\frac{1}{m^2}$, um das Ergebnis in den nutzbaren Wertebereich, beispielsweise der Grauwertmenge $G = 0, \ldots, 255$, zu skalieren. Zur Veranschaulichung wird an dieser Stelle das Beispiel des bewegten Mittelwerts von Nischwitz vorgestellt. Jene Faltungsmaske besitzt eine Größe von $m = 3$, kann aber auch mit größeren Filterkernen angewendet werden [NFH07].



(a) (b)

Abbildung 2.5: Beispiel eines Glättungsfilters: (a) Eingabebild [NKG12] und (b) Ergebnis eines Filterkerns mit 29x29 Koeffizienten [NB12].

$$H = \frac{1}{9} * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.6}$$

Formel 2.6: Faltungsmaske des bewegten Mittelwerts.

Abb. 2.5 demonstriert die Anwendung eines bewegten Mittelwerts mit großem Filterkern. In der Literatur wird für den bewegten Mittelwert auch der Begriff *Boxcar-Filter* oder *Box-Filter* verwendet [Tön05]. Es existieren weitere, über eine lineare Faltung herausgehende Glättungsalgorithmen, wie z. B. der Gauß-Filter oder der *Meanshift*-Algorithmus [Col06], der neben der Glättung bereits in den Bereich Bildsegmentierung eingeordnet werden kann. Diese Verfahren werden an dieser Stelle nicht weiter erörtert.

Im Gegensatz zu Glättungsfiltern besitzen Filterkerne von *Differenzfilter*n auch negative Koeffizienten. In der Regel ergibt die Summe der Koeffizienten eines Differenzfilters 0. Daher kann auf den Normierungsfaktor $\frac{1}{m^2}$ in *Formel 2.5* verzichtet werden. Schlussfolgernd berechnet die lineare Faltung in homogenen Bildbereichen ebenfalls den Wert 0. Bei Grauwertübergängen im Bild, also dort wo eine Kante zu vermuten ist, liefert der Filter eine Maßzahl für die Intensität des Übergangs. Deshalb werden Differenzfilter häufig eingesetzt um Kanten, Konturen und Linien eines Bildes zu schärfen und zu extrahieren

[NFH07]. Beispielhaft wird in *Formel 2.7* ein einfacher Filter vorgestellt, der die Differenzen der direkten Nachbarpixel in Zeilen- und Spaltenrichtung berechnet und den Wert des eigentlichen Pixel ignoriert, daher der Wert 0 für den zentralen Koeffizienten.

$$H_x = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad H_y = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.7}$$

Formel 2.7: Beispielmaske für einen einfachen Differenzenfilter.

Differenzfilter bilden die Grundlage für die Gradienten-basierte Kantendetektion [BB06c].

2.3.2 Kanten- und Feature-Point-Detektion

Aufgrund von physiologischen Untersuchungen lässt sich sagen, dass das menschliche Auge wesentlich besser auf Diskontinuität als auf Kontinuität reagiert [dW11]. Das bedeutet, dass diejenigen Bereiche in einem Bild, die Helligkeitsübergänge aufweisen, dem Betrachter mehr Informationen mitteilen als flächendeckende einfarbige Bereiche. Genau diese Tatsache wird in der digitalen Bildverarbeitung zur Extraktion und Weiterverarbeitung von Kanten und Konturen ausgenutzt. Kanten beschreiben jene Orte im Bild, an denen sich auf kleinem Raum und entlang einer ausgeprägten Richtung die Intensität stark ändert. *Abb. 2.6* zeigt, wie beim Betrachten eines Objekts der Blick des menschlichen Auges von einem interessanten Punkt zum nächsten springt [dW11].

Gradienten-basierte Kantendetektion

Eine Kante beschreibt also eine Intensitätsänderung. Je stärker diese Änderung, desto höher die Wahrscheinlichkeit einer Kante. Die Stärke der Änderung bezogen auf die Distanz entspricht der ersten Ableitung. Dies stellt den Ansatz für die Gradienten-basierte Kantendetektion dar [BB06d] [CM04a]. Die nachfolgende Beschreibung von [BB06d] soll die Vorgehensweise verdeutlichen. *Abb. 2.7(a)* zeigt eine helle Region im Zentrum, umgeben von einem dunklen Hintergrund.

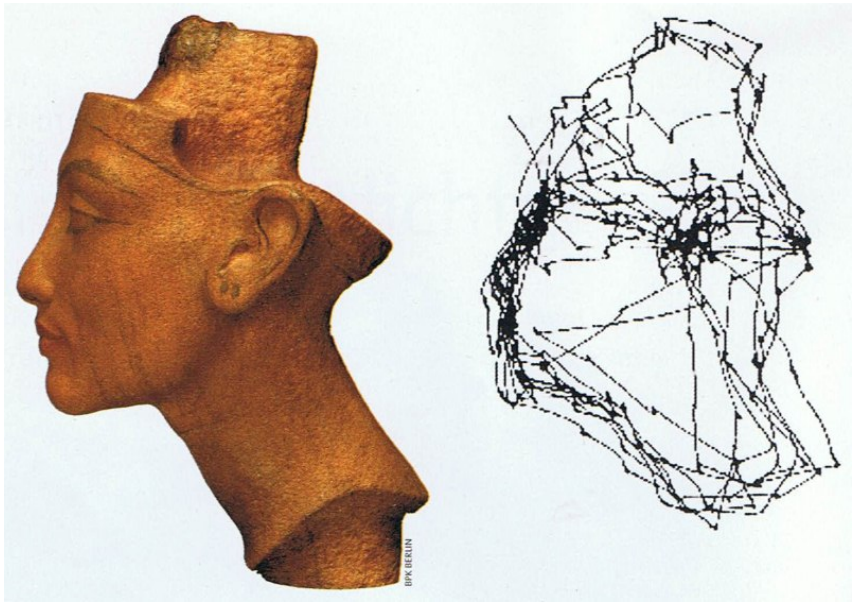
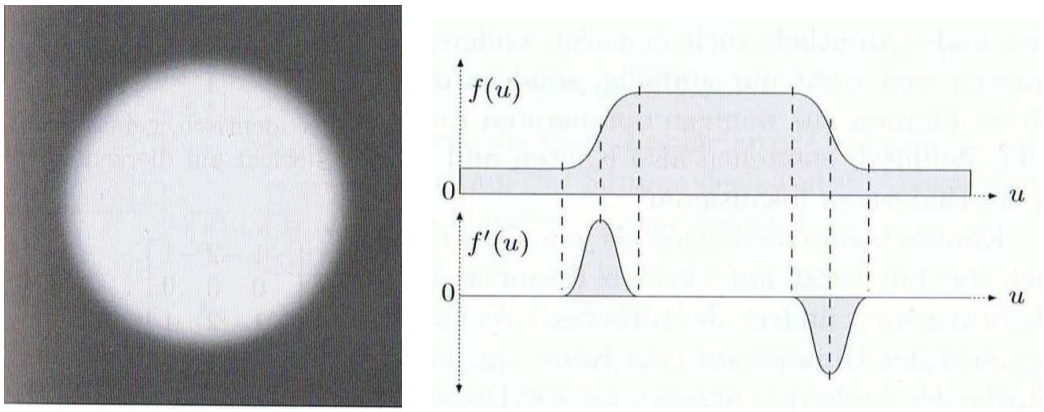


Abbildung 2.6: Funktionsweise der menschlichen Sehwahrnehmung: Beim Betrachten eines Objekts springt der Blick des menschlichen Auges von einem interessanten Punkt zum nächsten. Das rechte Bild zeigt die Augenbewegungsspuren eines Menschen, der die Büste der Nofretete (linkes Bild) betrachtet [dW11].



(a) (b)

Abbildung 2.7: Prinzip der Gradienten-basierten Kantendetektion: (a) Synthetisches Eingabebild; (b) Grauwertfunktion der mittleren Bildzeile und dessen erste Ableitung [BB06d].

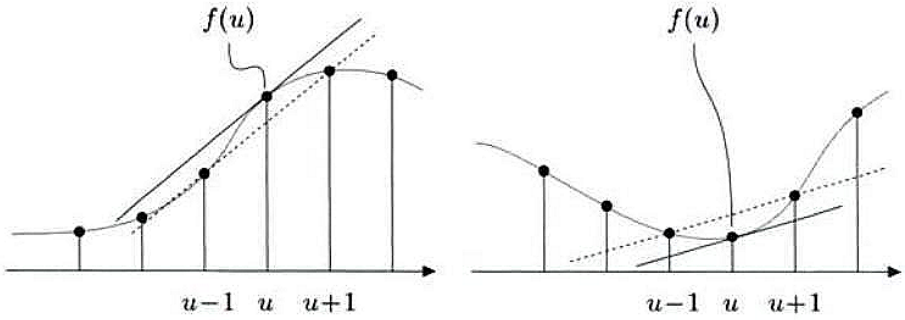


Abbildung 2.8: Schätzung der ersten Ableitung einer diskreten Funktion [BB06d].

In *Abb. 2.7(b)* repräsentiert der obere Graph die eindimensionale Funktion des Intensitäts- oder Grauwertprofils der mittleren Bildzeile des Eingabebildes. Darunter ist die erste Ableitung (siehe *Formel 2.8*) jener Funktion abgebildet.

$$f'(u) = \frac{df}{du}(u) \tag{2.8}$$

Formel 2.8: Erste Ableitung der Funktion $f(u)$ [BB06d].

Bei einer kontinuierlichen Funktion kann die erste Ableitung der Stelle x als Anstieg der Tangente an diesem Punkt interpretiert werden. Dies funktioniert aber für diskrete Funktionen wie $f(u)$ nicht, da die Ableitung nicht definiert ist. Der Tangentenanstieg an der Stelle u kann jedoch geschätzt werden, indem eine Gerade durch die Abtastwerte der benachbarten Pixel $u-1$ und $u+1$ angelegt und ihr Anstieg berechnet wird [BB06d]. *Formel 2.9* zeigt die Berechnung des approximierten Anstiegs und *Abb. 2.8* verdeutlicht die Schätzung der ersten Ableitung.

$$\frac{df}{du}(u) \approx \frac{f(u+1) - f(u-1)}{2} \tag{2.9}$$

Formel 2.9: Berechnung des approximierten Anstiegs zwischen benachbarten Pixeln [BB06d].

Die eindimensionale Funktion $f(u)$ repräsentiert die Bildzeilen. Der gleiche

Vorgang kann auch mit den Bildspalten durchgeführt werden. Bekanntermaßen werden die Ableitungen einer mehrdimensionalen Funktion als partielle Ableitungen bezeichnet. So ergibt sich in *Formel 2.10* für die Bildfunktion $I(u, v)$ entlang der u- bzw. v-Koordinate folgender Vektor, der auch als Gradientenvektor oder kurz als *Gradient* bezeichnet wird.

$$\bigtriangledown I(u,v) = \begin{pmatrix} \frac{\delta I}{\delta u}(u,v) \\ \frac{\delta I}{\delta v}(u,v) \end{pmatrix} \tag{2.10}$$

Formel 2.10: Gradient der Funktion $I(u, v)$ [BB06d].

Die Kantenstärke (siehe *Formel 2.11*) wird bei den Gradienten-basierten Kantendetektoren über den Betrag des Gradienten bestimmt [CM04a].

$$\mid \bigtriangledown I \mid = \sqrt{\left(\frac{\delta I}{\delta u}\right)^2 + \left(\frac{\delta I}{\delta v}\right)^2} \tag{2.11}$$

Formel 2.11: Betrag des Gradienten [BB06d].

Da der Gradient einen Vektor repräsentiert, ist sein Betrag invariant gegenüber Bilddrehungen. Dies ist eine notwendige Bedingung für eine richtungsunabhängige Lokalisierung von Kanten und daher die Grundlage vieler Kantendetektoren [Jäh05]. Auch die Richtung des Vektors (siehe *Formel 2.12*) kann aus den partiellen Ableitungen ermittelt werden. Für die direkte Kantenbestimmung spielt die Richtung jedoch keine Rolle.

$$\theta(u,v) = \arctan\left(\begin{pmatrix} \left(\frac{\delta I}{\delta v}\right) \\ \left(\frac{\delta I}{\delta u}\right) \end{pmatrix}\right) \tag{2.12}$$

Formel 2.12: Richtung des Gradienten [BB06d].

Wie in *Formel 2.10* verdeutlicht, entsprechen die Komponenten des Gradienten den ersten Ableitungen der Bildzeilen und Bildspalten. Die in *Abb. 2.8* skizzierte Schätzung der ersten Ableitung kann als lineare Filter realisiert werden. Dabei ergeben sich die in *Formel 2.14* aufgeführten Filterkerne für die

horizontale und vertikale Richtung.

$$H(u) = \begin{bmatrix} -0.5 & 0 & 0.5 \end{bmatrix} \qquad = 0.5 * \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.13}$$

$$H(v) = \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix} \qquad = 0.5 * \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.14}$$

Formel 2.14: Filterkerne für die horizontale und vertikale Richtung [BB06d].

Zur Veranschaulichung des Einsatzes der Gradientenfilter folgt *Abb. 2.9*. Diese zeigt ein synthetisches Testbild, auf dass die Gradientenfilter $\frac{\delta I}{\delta u}(u)$ und $\frac{\delta I}{\delta v}(v)$ angewendet wurden sowie der Betrag des Gradienten $\mid \bigtriangledown I \mid$, der als Bild ein Kantenbild ergibt. Wie in der Abb. zu sehen ist, stellt der Betrag des Gradienten ein Maß für die Kantenstärke dar.

Typische Vertreter der Gradienten-basierten Kantendetektion sind *Prewitt-*, *Sobel-*, *Kirsch-* und der *Roberts*-Filter, die sich i .d. R. lediglich durch die Größe bzw. durch die Koeffizientenwerte ihrer Filterkerne unterscheiden [Jäh05] [Tön05]. Die Filterkerne des Sobel-Filter bauen sich wie folgt auf:

$$H(u) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad H(v) = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.15}$$

Formel 2.15: Filterkerne des Sobel-Filters [BB06d].

Kantendetektion zweiter Ableitungen

Die Ergebnisse der gradientenbasierten Kantenfilter liefern Kanten ohne konkrete Positionsangaben. Die detektierten Linien werden genauso breit dargestellt wie die Länge des dazugehörigen Anstiegs in der Bildfunktion. Einige Kantendetektoren verwenden daher zusätzlich noch die zweite Ableitung. Diese beschreibt bekanntermaßen die lokale Krümmung einer Funktion. Dort, wo der

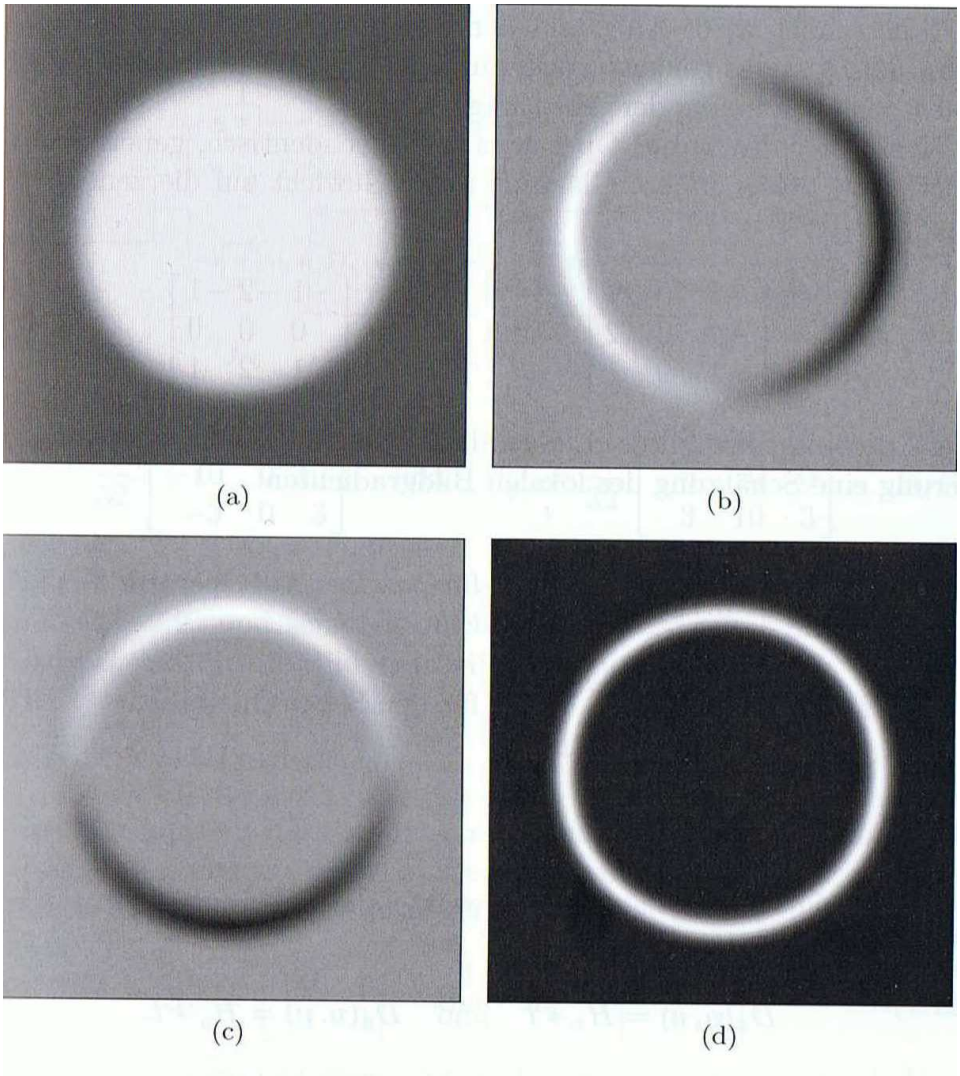
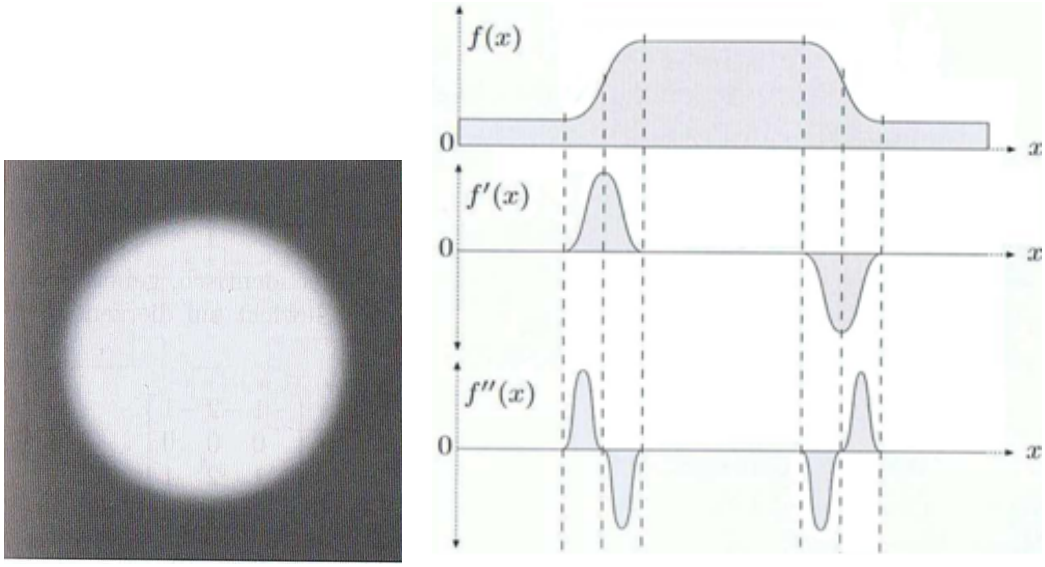


Abbildung 2.9: Einsatz der Gradientenfilter [BB06d]: Auf ein (a) synthetisches Testbild wurden (b+c) die Gradientenfilter $\frac{\delta I}{\delta u}(u)$ und $\frac{\delta I}{\delta v}(v)$ angewandt. Abb. (d) zeigt den Betrag des Gradienten $\mid \bigtriangledown I \mid$. In (b+c) werden maximal positive Werte weiß, maximal negative Werte schwarz und Nullwerte grau dargestellt. Wie in (d) zu sehen, stellt der Betrag des Gradienten ein Maß für die Kantenstärke dar.



(a) (b)

Abbildung 2.10: Erweiterung der *Abb. 2.7* durch die zweite Ableitung [BB06d]. In der zweiten Ableitung $f''(x)$ markiert jeweils die mittlere gestrichelte Linie die benannten Nulldurchgänge, die als genaue Kantenpositionen dienen.

Betrag der ersten Ableitung den maximalen Wert erreicht, gibt es einen Vorzeichenwechsel der zweiten Ableitung, einen Nulldurchgang. Die Idee dahinter ist, dass die Kantenfilter diese Nullstellen bzw. die Positionen der Nulldurchgänge als Kantenpositionen nutzen [Tön05]. *Abb. 2.10* erweitert *Abb. 2.7* und veranschaulicht das Prinzip der zweiten Ableitung. Analog zu den ersten Ableitungen können auch die zweiten Ableitungen als lineare Filter approximiert werden. Typische Vertreter sind der *Laplace-*, *Laplacian-of-Gaussian-* sowie der *Difference-of-Gaussian*-Filter [Jäh05] [Tön05] [BB06d]. Ein ebenfalls bekannter Vertreter ist der *Canny-Algorithmus* [Can86]. Er repräsentiert einen mehrstufigen Algorithmus, der verschiedene Faltungsoperationen hintereinander durchläuft:

- Glättung mithilfe eines Gauß-Filters.
- Gradienten-basierte Kantendetektion: In der Regel wird an dieser Stelle der Sobel-Filter verwendet. Ebenfalls werden in diesem Schritt der Betrag und die Richtung des Gradienten ermittelt.
- Non-Maxima-Suppression: Im Gradientenbetragsbild werden alle Pixel

unterdrückt, die entlang der Gradientenrichtung kein Maximum darstellen. Die Maxima lassen sich mithilfe der zweiten Ableitungen finden.

- Linienverfolgung mit Hysterese: Ausgehend von Pixeln, die über einem gegebenen oberen Schwellwert h liegen, werden alle Pixel entlang der Linie, die über einem gegebenen unteren Schwellwert l liegen, als zur Kante gehörend markiert.

Es werden also alle Kanten detektiert, die wenigstens ein über der Schwelle h befindlichen Pixelwert besitzen. Die beiden Schwellwerte und die Standardabweichung des Gauß-Filters σ (auch Glättungsparameter oder Filterradius genannt) bilden die drei Parameter zur Beeinflussung und Steuerung des Canny-Algorithmus. Dieser wird aufgrund seiner Robustheit und wesentlich verbesserter Kantendetektion gegenüber einfachen Gradienten-basierten Filtern bevorzugt [NFH07].

Einsatz eines Kantendetektors

Die Auswahl des Kantendetektors hängt immer von dem Vorhaben des Nutzers ab. Generell bringen Algorithmen unter Zuhilfenahme der zweiten Ableitung und zusätzlichen Glättungsoperationen deutlich bessere Ergebnisse. Dafür benötigen sie aber wesentlich mehr Rechenzeit als ein einfacher Differenzenfilter.

Bei der Gradienten-basierten Kantendetektion findet in der Regel der Sobel-Operator Verwendung, da er die besten Ergebnisse liefert, obgleich er dazu neigt, die Grauwertkanten zu dick darzustellen. Der Roberts-Operator reagiert zu schlecht auf verrauschte Bilder aufgrund seiner extrem schlanken 2x2-Faltungsmatrix. Kompass-Operatoren zeigen kaum Vorteile gegenüber einfachen Operatoren, was den Rechenmehraufwand nicht rechtfertigt.

Bei der Kantendetektion mithilfe der zweiten Ableitung bietet der Canny-Algorithmus bessere Resultate als der Laplace-Filter. Auch die Beeinflussung der beiden Algorithmen ist bei dem Canny-Operator mit drei Parametern vorteilhafter als beim Laplace-Filter mit nur einem Parameter (Standardabweichung des Gauß-Filters σ) [BB06d]. Auch in dieser Arbeit kommt im Rahmen des Konzeptes (siehe Absatz *5.3.2* (*Konturbasierte Segmentierung*)) der Canny-Algorithmus zum Einsatz.

Feature-Point-Detektion

Bekanntermaßen lässt sich ein markanter Bildpunkt (z. B. Eckpunkt), auch *Feauture-Point* oder *Points-of-Interest* genannt, als der Ort beschreiben, an denen begrenzende Linien oder Flächen aufeinander treffen. Die verbindende Stecke zwischen zwei Feature-Points wird als Kante bezeichnet. Dementsprechend stellen die beiden Feature-Points die Start- und Endposition der Kante dar [CM04a].

[BB06d] bezeichnet Feature-Points als markante strukturelle Ereignisse in einem Bild. Sie spielen daher bei der Zuordnung von Bildstrukturen und als Referenzpunkte bei der geometrischen Vermessung eine wichtige Rolle. Trotz ihrer Auffälligkeit bringt das automatisierte Lokalisieren von Feature-Points einige Schwierigkeiten mit sich. Es gibt verschiedene Algorithmen zur Feature-Point-Detektion, dennoch basieren die meisten Detektoren, auch *Interest-Operator*en genannt, auf folgender gemeinsamer Grundlage:

- Eine Kante wird definiert als eine Position im Bild, an der der Gradient der Bildfunktion in einer Richtung auffällig hoch, ansonsten niedrig ist.
- Ein Eckpunkt/Feature-Point entspricht einer Bildposition mit auffällig starkem Gradientenwert in mehr als einer Richtung.

Ähnlich den Kantendetektoren verwenden deshalb die meisten Interest-Operatoren die ersten oder zweiten Ableitungen der zweidimensionalen Bildfunktion und nutzen daher in der Regel Grauwertbilder als Ausgangslage [BB06d]. Die wichtigsten Vertreter der Feature-Point-Detektion sind der *Moravec*-, der *Harris*-, der *Susan*- sowie der *Sift*-Detektor. Der Sift-Detektor (*Scale-invariant feature transform*) [Low04] unterteilt ein Bild in lokale Merkmalspunkte, die unempfindlich gegen perspektivische Verzerrung sind. Markant sind Objekte, deren Eigenschaften von ihrem Hintergrund abweichen. Sie lassen sich durch ihre Histogramme kennzeichnen, deren Merkmale in Vektoren gespeichert werden, die zum Vergleich dienen. Die extrahierten Merkmalspunkte sind unempfindlich gegenüber Translation, Rotation und Skalierung. Des Weiteren sind sie robust gegen Beleuchtungsvariation, Bildrauschen und geringere geometrische Deformation höherer Ordnung, wie sie zum Beispiel durch projektive Abbildung eines Objekts von verschiedenen Standpunkten im Raum entstehen [Wik12d]. Der Sift-Detektor findet zudem Einsatz bei der Erzeugung von Bilderwelten.

Eine dafür spezifizierte und ausführlichere Beschreibung des SIFT-Algorithmus findet sich in Absatz *2.6.3* (*Generieren einer Bilderwelt*).

2.3.3 Segmentierung

Neben den vorgestellten Detektoren für spezielle charakteristische Bildpunkte, wie Kanten- oder Eckpunktpixel, können lineare Filter für das Gruppieren mehrerer, benachbarter Pixel zu einer Fläche verwendet werden. Hierbei ist das Ziel, ein Eingabebild in „sinnvolle“ Bereiche und Regionen zu zerlegen [CM04c]. Für das Zusammenlegen benachbarter Pixel müssen jedoch für die Region charakteristische Merkmale bestimmt werden, wie z. B. eine Farbe. Dementsprechend werden benachbarte Pixel gleicher oder ähnlicher Farbe zu einer Region, zu einem *Segment* zusammengesetzt. Dieses Zusammensetzen benachbarter Pixel anhand eines Kriteriums oder mehrerer Kriterien wird *Segmentierung* genannt. Für die Segmentierung existieren verschiedene Ansätze, wie

- Pixel-basierte Segmentierung,
- Regionen-basierte Segmentierung,
- Histogramm-basierte Segmentierung,
- Textur-basierte Segmentierung.

Ein bekanntes Beispiel für die Pixel-basierte Segmentierung ist der *Flood-Fill*-Algorithmus, bei dem (ausgehend von einem Startpixel) alle über Nachbarschaften direkt verbundene Pixel mit gleicher (oder ähnlicher) Farbe wie das Startpixel neu eingefärbt werden (siehe *Abb. 2.11*). Eine Regionen-basierte Farbsegmentierung kann anhand der in Absatz *2.2.3* (*Vergleichen und Transformieren von Farben*) erläuterten Vergleichsmechanismen ähnlich dem Flood-Fill-Algorithmus umgesetzt werden. Ein weiteres Beispiel für eine Regionen-basierte Segmentierung stellt der *Meanshift*-Algorithmus [Col06] dar, der eine Bildglättung segmentbasiert durchführt. Eine genaue Beschreibung des mathematischen Vorgangs des Meanshift-Algorithmus sowie eine beispielhafte Umsetzung findet sich in [Col06] und [Bar07].

(a) (b)

Abbildung 2.11: Flood-Fill-Algorithmus zum neuen Einfärben eines Segments (grüner Stuhl). Alle über Nachbarschaften direkt verbundene Pixel mit gleicher (oder ähnlicher) Farbe wie das Startpixel werden neu eingefärbt [NKG12].

2.4 Perspektivische Projektion

Für das Augmentieren einer abgebildeten Szene in Bildern sowie das Extrahieren von Tiefeninformationen aus Fotos werden grundlegende Kenntnisse über Projektionsverfahren (also das Überführen einer dreidimensionalen realen Szene in eine zweidimensionale Bildebene) notwendig.

Um räumliche Objekte in einer Ebene darzustellen oder räumliche, geometrische Berechnungen in jener Ebene betreiben zu können, muss der dreidimensionale Raum in eine zweidimensionale *Bildebene* überführt werden. Als Bildebene wird im Allgemeinen eine orthogonal zur Kamera stehende 2D-Ebene bezeichnet, die in der Perspektive zwischen Betrachter und Objekt, innerhalb des Objektes oder dahinter angenommen wird. Diese Abbildung des dreidimensionalen Raumes auf die zweidimensionale Bildebene (bei dem sich die Projektionsstrahlen in einem Punkt, dem Projektionszentrum, schneiden) wird *Perspektivische Projektion* genannt [Kli01]. Nachfolgend wird die perspektivische Projektion vorgestellt. Zuvor werden jedoch die extrinsischen und intrinsischen Kameraparameter kurz erläutert.

2.4.1 Extrinsische und Intrinsische Kameraparameter

Ein von einer Kamera abzubildendes Objekt besitzt Weltkoordinaten, die in das Kamerakoordinatensystem überführt werden müssen. Dazu dienen die *Ex-*

trinsischen Kameraparameter [Hof09]. Sie definieren Position und Orientierung der Kamera im Raum. Folglich dienen sie für die Wiederherstellung des Zusammenhangs zwischen Welt- und Kamerakoordinatensystem. Die extrinsischen Kameraparameter beinhalten die Translations- sowie die Rotationsvorschrift. Das bedeutet, sie beschreiben die Verschiebung der Kamera zum Ursprung des Weltkoordinatensystems sowie die Drehung um die drei Euler-Winkel.

Die *Intrinsischen Kameraparameter* transformieren die 3D-Punkte (die eine Kamera aufnimmt) mithilfe der Kamerakoordinaten in 2D-Punkte (in Pixeleinheiten) [Hof09]. Intrinsische Kameraparameter beschreiben die innere Geometrie der Kamera. Sie dienen zur Wiederherstellung des Zusammenhangs zwischen 3D-Kamera- und 2D-Bildkoordinatensystem. Intrinsische Kameraparameter beinhalten die Brennweite einer Kamera, die Position des Bildmittelpunktes in Pixeleinheiten sowie die Pixelskalierung in x- und y-Richtung. Des Weiteren sind sie unabhängig von den extrinsischen Kameraparametern.

2.4.2 Prinzip des Lochkameramodells

Die *Perspektivische Projektion* lässt sich grundlegend am einfachsten Kameramodell erläutern, dem *Lochkameramodell*. Die Lochkamera besteht aus einer Box mit einem kleinen Loch an der Vorderseite (O, entspricht dem Ursprung des dazugehörigen Kamerakoordinatensystems) und der Bildebene an der Rückseite. Die optische Achse ist der orthogonal zur Kamera verlaufende *Hauptsehstrahl* und kann als Z-Achse des Kamerakoordinatensystems gesehen werden [BB06e]. In Abbildung 2.12 erläutert [BB06e] das Prinzip der perspektivischen Projektion am Beispiel des Lochkameramodells und einem Objekt (Kaktus) zum Abbilden.

Ein sichtbarer Objektpunkt (im Beispiel die Kaktusspitze) befindet sich in einer Distanz Z von der Lochebene und im vertikalen Abstand Y über der optischen Achse. Die *Brennweite* f beschreibt den Abstand vom Ursprung des Kamerakoordinatensystems zu dem Zentrum der Bildebene. Somit kann mithilfe der Brennweite f die Höhe der dazugehörigen Projektion y berechnet werden (siehe Formel 2.16). Dies gilt analog für die Breite x.

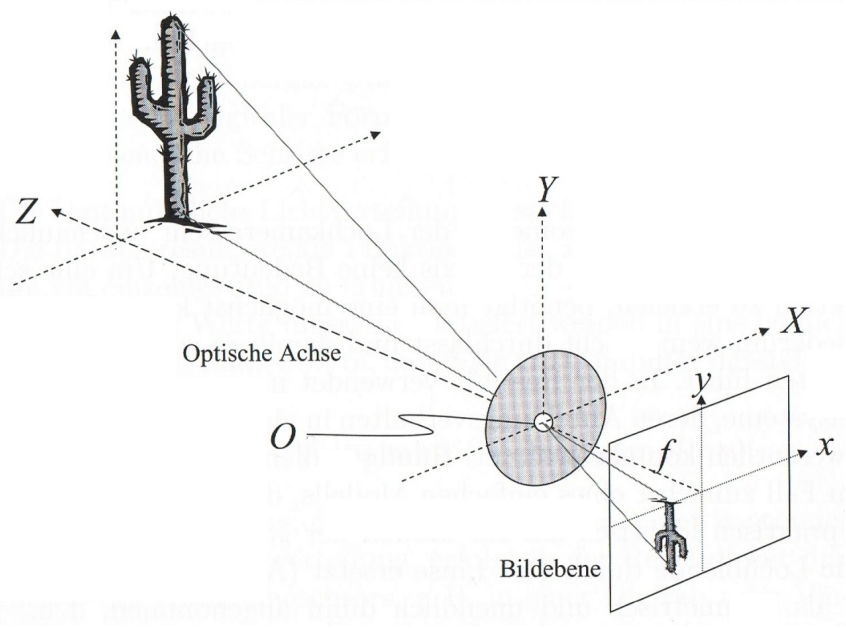


Abbildung 2.12: Prinzip der perspektivischen Projektion anhand des Lochkameramodells [BB06e].

$$y = -f\frac{Y}{Z} \qquad \text{und} \qquad x = -f\frac{X}{Z} \tag{2.16}$$

Formel 2.16: Positionsberechnung auf der Bildebene [BB06e].

Das Bild entspricht dem Eindruck des menschlichen Auges mit einem Unterschied: Die Bildfläche des menschlichen Auges stellt eine Hohlkugel dar, während die Bildfläche der Perspektive eine Ebene darstellt [Tho01b]. In einer perspektivischen Projektion stellt ein *Fluchtpunkt* den Schnittpunkt aller Geraden dar, die im Original parallel verlaufen. Genauer gesagt, stellt ein Fluchtpunkt den Durchstoßpunkt von Richtungsparallelen ausgehend vom Augpunkt durch die Bildebene dar. Der Fluchtpunkt ist also ein Schnittpunkt von Geraden in der Bildebene, die im originalen dreidimensionalen Raum parallel verlaufen. Dabei gilt, dass jede Schar von parallelen Geraden einen gemeinsamen Fluchtpunkt besitzt, es sei denn, die Geraden sind parallel zur Bildebene. Jene Parallelen bleiben in ihren Richtungen erhalten. Dementsprechend behalten zur Bildebene parallele Flächen ihre Form, nur die Größe ändert sich mit dem Abstand zur Bil-

debene. Im Gegensatz dazu schneiden sich Geraden, die senkrecht zur Bildebene verlaufen, im *Hauptfluchtpunkt*, also im Durchstoßpunkt des Hauptsehstrahls (optische Achse) [Tho01a].

Des Weiteren kann die perspektivische Projektion mathematisch durch eine homogene *Projektionsmatrix* beschrieben werden. In diese Matrix fließen für die 3D-Computergrafik wichtige Informationen ein, wie die Entfernung der nahen und fernen *Clipping-Plane*, das Blickfeld des Betrachters (*Field-of-View*) sowie das Seitenverhältnis des Bildes. Die Projektionsmatrix bestimmt somit den Sichtbereich des Beobachters [Sch06].

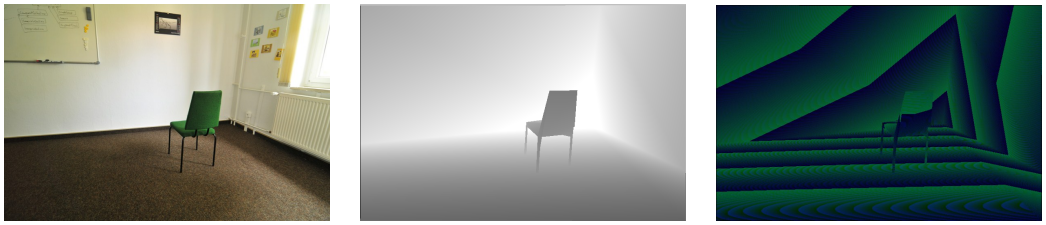
2.5 Depth-Maps

Aufbauend auf den Grundlagen der *Perspektivischen Projektion* soll die Rekonstruktion der *Kameratiefe* (Tiefeninformationen aus der Sicht der Kamera im eigenen Koordinatensystem) so genannte *Tiefeninformationen*, also Informationen über die (orthogonale) Distanz abgebildeter realer Objekte zur Kamera (Bildebene) liefern. Somit kann jedem Pixel eines Fotos eine Tiefe zugeordnet werden. Diese Tiefe eines Pixels wird im weiteren Text als *Tiefenwert* (in der Regel ein Gleitkommawert) bezeichnet. Werden diese Tiefenwerte als Füllwerte für die Pixel verwendet, entsteht ein Bild, die *Tiefenkarte* oder *Depth-Map* genannt wird [Wik12a]. Auf diese Weise ist es möglich, die rekonstruierte Tiefe eines Fotos pixelgenau und effizient zu speichern. Der Name Depth-Map wird häufig im Zusammenhang mit dem *Z-Puffer-Algorithmus* [Wat02b] genannt. Dieser bestimmt beim 3D-*Rendering*, welche Polygone gezeichnet und welche aufgrund von Verdeckungen nicht gezeichnet werden. Hierzu wird neben dem *Frame-Buffer* ein weiterer Speicher benötigt, der *Z-Buffer* [Wat02b]. Dieser wird verwendet, um für jedes Pixel des Bildschirms die Entfernung des entsprechenden Polygons zur Kamera zu speichern. Der Inhalt des Z-Buffers kann über Depth-Maps gespeichert und visualisiert werden [Agr12].

Im weiteren Text dieser Arbeit wird der Begriff Depth-Map verwendet und damit auf eine Textur mit den Tiefenwerten eines Fotos (als Füllwerte für ihre Pixel) verwiesen. Das Kodieren der Tiefenwerte als Füllwerte der Depth-Map kann in verschiedenen Pixelformaten (mit unterschiedlicher Präzision) erfolgen. Dafür werden für diese Arbeit folgende Begriffe eingeführt:

- Grayscale-Depth-Maps,
- Float-Depth-Maps,
- Rgb-Divided-Depth-Maps.

Da in vielen Problemstellungen eine hohe Präzision der Tiefenwerte gefordert wird, empfiehlt sich das Speichern der Depth-Map als einkanälige Gleitkommazahl-Textur, im weiteren Text als *Float-Depth-Map* bezeichnet, mit 32-Bit Kodierungsgenauigkeit pro Tiefenwert. Andere Problemstellungen fordern vorrangig eine möglichst geringe Speicherverwendung und akzeptieren Ungenauigkeiten bei der Speicherung der Tiefenwerte. In diesem Fall kann eine *Grayscale-Depth-Map* mit 8-Bit Kodierungsgenauigkeit pro Tiefenwert verwendet werden. Eine dritte, eigens für diese Arbeit entwickelte Variante bietet eine höhere Präzision als Grayscale-Depth-Maps, ohne Gleitkommazahlen zu verwenden. Diese Variante wird *Rgb-Divided-Depth-Map* genannt und arbeitet mit gewöhnlichen RGB-Texturen und 8 Bit pro Farbkanal. Der Tiefenwert wird auf die drei RGB-Farbkanäle (jeweils als Ganzzahl und im Dezimalsystem) zerlegt. Die Vorkommastellen des Tiefenwertes werden im Rot-Kanal abgelegt. Die ersten zwei Nachkommastellen werden im Grün-Kanal und die dritte und vierte Nachkommastelle im Blau-Kanal abgelegt. Diese Speicherform beschränkt den Tiefenbereich. Die Tiefe darf einen Wert von 255 Einheiten nicht überschreiten, da im Rot-Kanal für die Vorkommastellen nur 8 Bit zur Verfügung stehen. Des Weiteren werden nur vier Nachkommastellen berücksichtigt. Dennoch reicht diese Beschränkung des Tiefenbereiches für die hier vorliegende Arbeit aus. Während bei der Float-Depth-Map und der Rgb-Divided-Depth-Map der Tiefenwert gespeichert wird, erfolgt bei der Graustufen-Depth-Map eine Skalierung der Tiefenwerte auf den Byte-Wertebereich. Abb. 2.13 zeigt ein Foto und die dazugehörige Graustufen-Depth-Map und Rgb-Divided-Depth-Map. Alle in dieser Arbeit verwendeten Depth-Maps (unabhängig vom verwendeten Pixelformat) kodieren die Kameraposition als den kleinstmöglichen Tiefenwert (Tiefe = Null). Das bedeutet im Falle der 8-Bit-Graustufen-Depth-Map, dass Objekte mit einer geringeren Distanz zur Kamera (also kleinerer Tiefenwert) dunkler und Objekte mit größerer Distanz zur Kamera (also größerer Tiefenwert) heller kodiert werden. Es existiert in anderen Arbeiten aber auch die invertierte Variante.



(a) (b) (c)

Abbildung 2.13: Beispiele einer Depth-Map: (a) Ausgangsbild. (b) Dazugehörige Graustufen-Depth-Map. Objekte mit geringerer Distanz zur Kamera (also kleinerer Tiefenwert) werden dunkler und Objekte mit größerer Distanz zur Kamera (also größerer Tiefenwert) werden heller dargestellt. (c) Dazugehörige Rgb-Divided-Depth-Map. Der Tiefenwert wird auf die drei RGB-Farbkanäle (jeweils als Ganzzahl) zerlegt. Die Vorkommastellen des Tiefenwertes werden im Rot-Kanal abgelegt. Die ersten vier Nachkommastellen werden im Grün- und Blau-Kanal abgelegt.

Der Vollständigkeit halber wird der Begriff *Height-Map* erklärt. Height-Maps werden zur Speicherung von Höhenprofilen dreidimensionaler Oberflächen (z. B. Landschaften) verwendet. Dementsprechend haben sie eine ähnliche Funktion wie Depth-Maps, da sie Höhenwerte einer dreidimensionalen Umgebung in einer zweidimensionalen Abbildung speichern [Men11]. Height-Maps werden in der Computergrafik für die effiziente Speicherung von Geländestrukturen und für die Erstellung von 3D-Landschaften (z. B. in Computerspielen) verwendet, wie in Abb. 2.14 dargestellt wird.

2.6 Bilderwelten

Aufbauend auf den Grundkenntnissen von Bilddaten, Bildverarbeitungsmechanismen und der perspektivischen Projektion wird in diesem Absatz der Begriff Bilderwelten erklärt.

2.6.1 Definition Bilderwelt

Eine *Bilderwelt* stellt eine bildbasierte 3D-Welt, bestehend aus gewöhnlichen Fotos, dar. Hierbei sind für jedes Foto die intrinsischen (Brennweite, Pixelskalie-

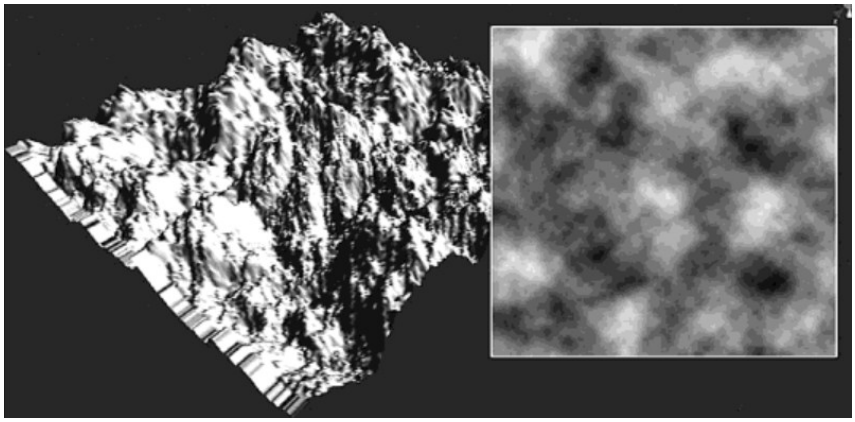


Abbildung 2.14: Eine Height-Map und das damit erstellte 3D-Terrain [Men11].

rung) und extrinsischen Kameraparameter (Kameraposition und Orientierung im 3D-Raum) rekonstruiert [SSS06a]. Das zu einer Kamera zugehörige Foto wird im weiteren Text als *Kamerabild* bezeichnet. Zu jeder rekonstruierten Kamera wird eine *Image-Plane* erzeugt. Eine Image-Plane in einer Bilderwelt stellt eine planare, rechteckige Fläche im 3D-Raum dar, die vollständig und passgenau mit dem Kamerabild texturiert ist und orthogonal in einem berechneten Abstand vor diese rekonstruierte Kamera positioniert wird (Weitere Informationen zur Abstandsbestimmung finden sich unter Absatz *4.4.3* (*Verfeinerte Zielsetzung der Authentischen Darstellung der Bilderwelten*)). Ein Kamerabild bezeichnet also das eigentliche Foto und die Image-Plane repräsentiert das Foto als 3D-Objekt innerhalb der Bilderwelt. Es werden keine kalibrierten Kameradaten und Kamerabilder vorausgesetzt. Es können beliebige Fotos beliebiger Kameras, erstellt aus beliebigen Positionen und Blickwinkeln, verwendet werden [SSS06a]. Im Gegensatz zu Panorama-Systemen müssen die Fotos dementsprechend nicht von einer konstanten Position erstellt werden. Die Fotos müssen jedoch Bildüberlappungen aufweisen [Kra04]. Abb. 2.15 zeigt eine Bilderwelt des Gebrüder Grimm-Nationaldenkmals in Hanau.

Zusätzlich enthält eine Bilderwelt neben den Image-Planes eine 3D-Punktwolke. Die Punktwolke repräsentiert eine dreidimensionale, unvollständige Darstellung der abgebildeten Szene bestehend aus 3D-Punkten, im weiteren Text als *Key-Points* bezeichnet. Eine nähere Erläuterung zu den Key-Points folgt in Absatz *2.6.3* (*Generieren einer Bilderwelt*). Abb. 2.16 zeigt die Punktwolke und die rekonstruierten Kamera-Positionen der Brüder Grimm-Nationaldenkmals-

Abbildung 2.15: Eine Bilderwelt (des Gebrüder Grimm-Nationaldenkmals in Hanau) stellt eine 3D-Welt bestehend aus Fotos dar. Rekonstruierte Kameraparameter werden genutzt, um mit den Kamerabildern texturierte Image-Planes zu positionieren [NKG12] (Fotos von [Kru08]).

Bilderwelt.

2.6.2 Anforderungen für das Generieren einer Bilderwelt

Ziel der Erstellung einer Bilderwelt ist die Darstellung einer bildbasierten 3D-Welt mit einer möglichst hohen Abdeckung des 3D-Raumes durch Fotos. Die Abdeckung des 3D-Raumes durch Fotos in einer Bilderwelt wird im weiteren Text als *Flächenabdeckung* bezeichnet. Hierbei werden die Fotos als zweidimensionale Texturen auf den Image-Planes innerhalb der 3D-Welt verwendet. Wenn Fotos bzw. benachbarte Image-Planes innerhalb der 3D-Welt keine Überdeckungen haben, entstehen dreidimensionale Bereiche ohne Bildmaterial. So genannte *Schwarze Bereiche* sollen in der Bilderwelt minimiert oder ganz vermieden werden. Schwarze Bereiche sind Bereiche innerhalb der 3D-Welt, die nicht mit Bildmaterial der Bilderweltenszene texturiert sind. Sie wirken störend beim Betrachten der Bilderwelt, da sie den Zusammenhang der Szene unterbrechen.

Infolgedessen ist das Eingabematerial (Fotos einer Szene mit überlappenden

Abbildung 2.16: Punktwolke und rekonstruierte Kameras der Brüder Grimm-Nationaldenkmal-Bilderwelt aus Abb. 2.15 [NKG12].

Bildanteilen) von essentieller Bedeutung. Nach [Kra04] müssen die überlappenden Bildanteile zwischen den Kamerabildern mindestens 60% Längs- und 20% Querüberdeckung bei möglichst heterogenen Strukturen im Bild betragen. Des Weiteren spielen Anzahl der Fotos, Abdeckung der Szene durch die Bildinhalte der Fotos sowie die Qualität, Auflösung und Bildstruktur der Fotoaufnahmen eine Rolle. Die *Bildstruktur* eines Fotos, genauer ausgedrückt die Struktur des Bildinhaltes, ist definiert durch die Anzahl der Objekte in der abgebildeten Szene, deren geometrische Formen und Größen sowie deren Struktur und Oberfläche. Einfarbige, glatte und spiegelnde Flächen werden auch als homogene Bildstruktur und mehrfarbige und rauhe Flächen als heterogene Bildstruktur bezeichnet. Spiegelnde und sich wiederholende (so genannte homogene) Strukturen in einem Foto führen zu Problemen, während sich mehrfarbige und rauhe Strukturen (texturierte heterogene Flächen) vorteilhaft auf die Generierung einer Bilderwelt auswirken. Je heterogener die Bildinhalte der Fotos, desto besser kann eine Bilderwelt generiert werden. Je größer die Anzahl der Fotos und je besser die Abdeckung der Realszene durch den Bildinhalt der Fotos, desto detaillierter wird eine Realszene in einer Bilderweltenszene abgebildet. Im Ergebnis bedeutet dies eine höhere Flächenabdeckung in der Bilderwelt. Dementsprechend werden Fotos als Eingabematerial vorausgesetzt, die dies

(möglichst gut) erfüllen. Es wird an dieser Stelle bewusst nicht definiert, wie groß die Mindestanzahl von Fotos für das Erstellen einer Bilderwelt ist, da dies nicht allgemeingültig festgelegt werden kann. Dies hängt von realer Szenengröße, Bildqualität bzw. Aufnahmequalität und Anwendungsfall bzw. Ziel der Bilderweltendarstellung ab. Beispielsweise können für eine panoramaartige Darstellung weniger als 20 Fotos ausreichen, während eine vollständige Flächenabdeckung für die Abbildung einer großen Realszene (z. B. eine komplette Stadt) mehrere 1000 Fotos voraussetzt.

Des Weiteren gilt, je mehr Fotos bzw. deren Kameras rekonstruiert werden, desto mehr Key-Points entstehen. Das bedeutet, je mehr Kameras mit ähnlichen Bildinhalten in einer Bilderwelt enthalten sind, desto dichter ist die Punktwolke an jener Stelle der Bilderweltenszene. Zusätzlich spielt wieder die *Bildstruktur* eine Rolle. Je heterogener die Bildstruktur der Fotos, desto mehr Key-Points werden rekonstruiert.

2.6.3 Generieren einer Bilderwelt

Für das Generieren einer Bilderwelt werden so genannte *Structure-from-Motion*-Algorithmen (im weiteren Text als *SfM*-Algorithmus bezeichnet) verwendet. Dieser Absatz erläutert das Generieren einer Bilderwelt am Beispiel der Software *Bundler* [Sna10a], da Bundler auch als Werkzeug in dieser Arbeit eingesetzt wird.

Ein SfM-Algorithmus ermittelt neben den extrinsischen (Position und Orientierung) auch die intrinsischen (Brennweite, optisches Zentrum in der Bildebene) Kameraparameter. Bundler arbeitet hierbei in zwei Arbeitsschritten:

- Extraktion lokaler Bildmerkmale,
- Bündelblockausgleichung.

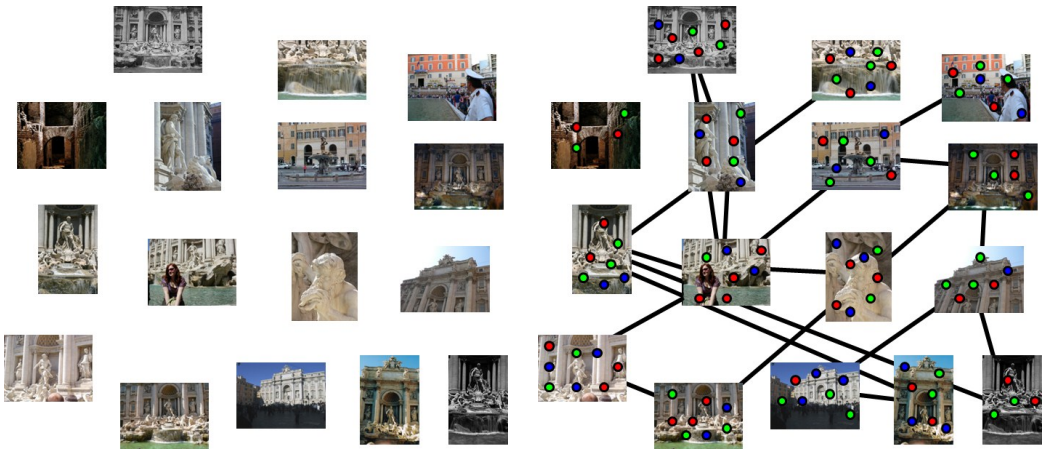
Extraktion lokaler Bildmerkmale

Im ersten Schritt wendet Bundler den *Scale-invariant feature transform*-Algorithmus (kurz *SIFT*) [Low04] an, um lokale, markante Punkte bzw. Flächen (z. B. von abgebildeten Objekten), so genannte *Bildmerkmale*, in einem Bild zu detektieren. Je heterogener die Bildstruktur der Fotos, desto mehr Bildmerkmale werden extrahiert. Die Position und Orientierung dieser Bildmerkmale

(a) (b)

Abbildung 2.17: Der SIFT-Algorithmus [SSS06d]: (a) Eingabebild. (b) Die Ergebnisvektoren der Feature-Points sind in das Bild als Quadrate eingezeichnet. Die Skalierung eines Quadrats kodiert die Größe des Bildmerkmals im Eingabebild. Die Rotation eines Quadrats kodiert die Orientierung des Bildmerkmals im Eingabebild. Der Mittelpunkt eines Quadrats kodiert die zentrale Position eines Bildmerkmals im Eingabebild.

wird mithilfe einer lokalen Nicht-Maxima-Unterdrückung sowie durch das Vergleichen der Gradienten der Nachbarpixel bestimmt. Zusätzlich ermittelt SIFT für jeden dieser Bildmerkmale einen Skalierungswert, indem ihre Skaleninvarianz durch mehrere Skalierungsdurchläufe mittels *Laplacian-of-Gaussian*-Filter (*LoG* [MH80], Anwenden eines Laplace-Operators auf eine Gauß-Funktion) bestimmt wird. Diese extrahierten Bildmerkmale werden im weiteren Text als *Feature-Points* bezeichnet. Die drei Kenngrößen (Position, Orientierung, Skalierung) der Feature-Points werden als Vektoren gespeichert und dienen Bundler zum bildübergreifenden Vergleichen von Bildmerkmalen. Abb. 2.17 zeigt ein Eingabebild und das dazugehörige Ergebnis des SIFT-Algorithmus. Die Ergebnisvektoren der Feature-Points sind zur Veranschaulichung in das Bild als Quadrate eingezeichnet. Die Skalierung eines Quadrats kodiert die Größe des Bildmerkmals im Eingabebild. Die Rotation eines Quadrats kodiert die Orientierung des Bildmerkmals im Eingabebild. Der Mittelpunkt eines Quadrats kodiert die zentrale Position eines Bildmerkmals im Eingabebild.



(a) (b)

Abbildung 2.18: Verknüpfen von Bildmerkmalen in verschiedenen Fotos [SSS06d]: (a) Eingabebilder mit Feature-Points. (b) Verknüpfungen der Feature-Points.

Bündelblockausgleichung

Mithilfe einer Sammlung von Fotos mit überlappenden Bildanteilen und den Feature-Points stellt Bundler Verknüpfungen zwischen den Fotos her, rekonstruiert die intrinsischen und extrinsischen Kameraparameter pro Foto und generiert eine Punktwolke der abgebildeten Szene als vereinfachte Szenengeometrie. Dies erfolgt durch eine modifizierte Version einer *Bündelblockausgleichung* (auch *Bundle-Adjustment* genannt). Die folgende Beschreibung der Bündelblockausgleichung in Bezug auf Bundler ist angelehnt an [Kra04]. Die Bündelblockausgleichung stellt auf rechnerische Weise einen direkten Zusammenhang zwischen Bildkoordinaten (Feature-Points) und 3D-Objektkoordinaten her ohne Verwendung bekannter Modellkoordinaten. Voraussetzung hierfür sind Bilder mit überlappenden Bildanteilen, ein so genannter *blockförmiger Bildverband*.

Die Bildkoordinaten und das dazugehörige Projektionszentrum definieren ein räumliches Strahlenbündel (Jeder Bildpunkt eines Bildes definiert einen Strahl durch das Projektionszentrum). Die Verknüpfung der Bilder erreicht Bundler mit Hilfe von korrespondierenden Feature-Points, die in mehreren, sich überlappenden Bildern erkennbar sind und jeweils exakt lokalisiert werden können. Anhand eines Vergleichsalgorithmus der Vektoren (Kenngrößen) der Feature-Points, findet Bundler diese korrespondierenden Feature-Points,

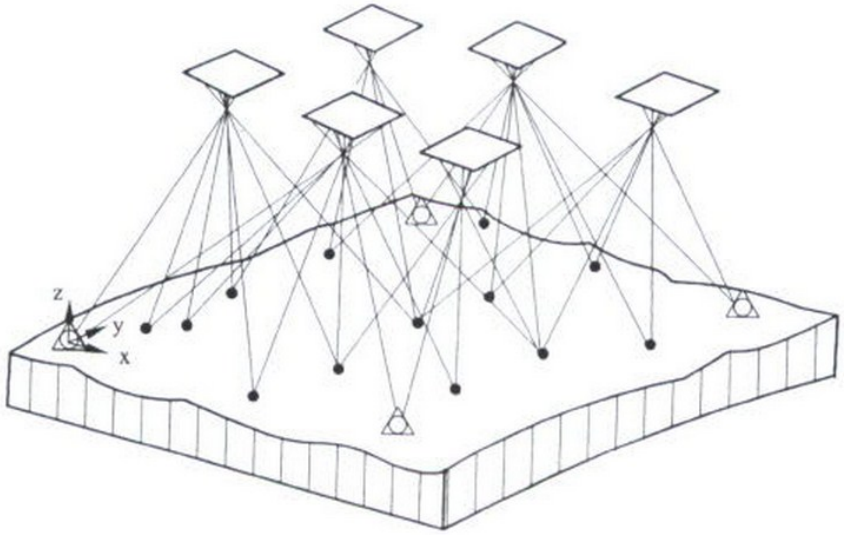


Abbildung 2.19: Verschieben und Rotieren bis die korrespondierenden Strahlen an den Verknüpfungspunkten möglichst gut zum Schnitt kommen [Kra04].

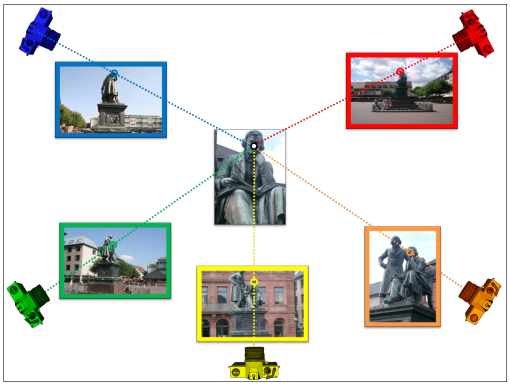
wie in Abb. 2.18 verdeutlicht wird. Mithilfe der Korrespondenzen werden die Stahlenbündel (und somit die Kameras der Bilder) dreidimensional verschoben und gedreht bis die entsprechenden Strahlen an den Verknüpfungspunkten möglichst gut zum Schnitt kommen. Dieser Fitting-Mechanismus wird Ausgleichungsprinzip oder Ausgleichungsrechnung genannt [Kra04], wodurch dieses Verfahren in Kombination mit dem blockförmigen Bildverband seinen Namen hat. Abb. 2.19 zeigt das Grundprinzip des Fitting-Mechanismus. Durch den Fitting-Mechanismus der Bündelblockausgleichung entstehen neben den Transformationsmatrizen der Kameras die dreidimensionalen Verknüpfungspunkte, die bereits erwähnten *Key-Points* (siehe Absatz *2.6.1*). Diese besitzen neben einer 3D-Position auch einen Farbwert, der sich aus dem Durchschnitt aller Bildpunkte, die den Key-Point in den Fotos repräsentieren, bildet. Da es sich um eine Ausgleichung, also eine Annäherung handelt, sind Fehler einkalkuliert. Daher wird das Fehlerrisiko um so geringer, je mehr korrespondierende Bildpunkte ein Key-Point besitzt. In diesem Zusammenhang ergeben sich zwei neue Begriffe: Key-Points, die zu einer bestimmten Kamera zugehörig sind, werden im weiteren Text als *Observed-Points* dieser Kamera bezeichnet. Kameras, die einen bestimmten Key-Point als Bildpunkt in ihrem Foto enthalten, werden als *Observer* des Key-Points bezeichnet. Der korrespondierende Feature-Point als Bildpunkt in Pixeleinheiten wird *Observed-Pixel* genannt.

2.6.4 Kurzfassung Bilderwelten

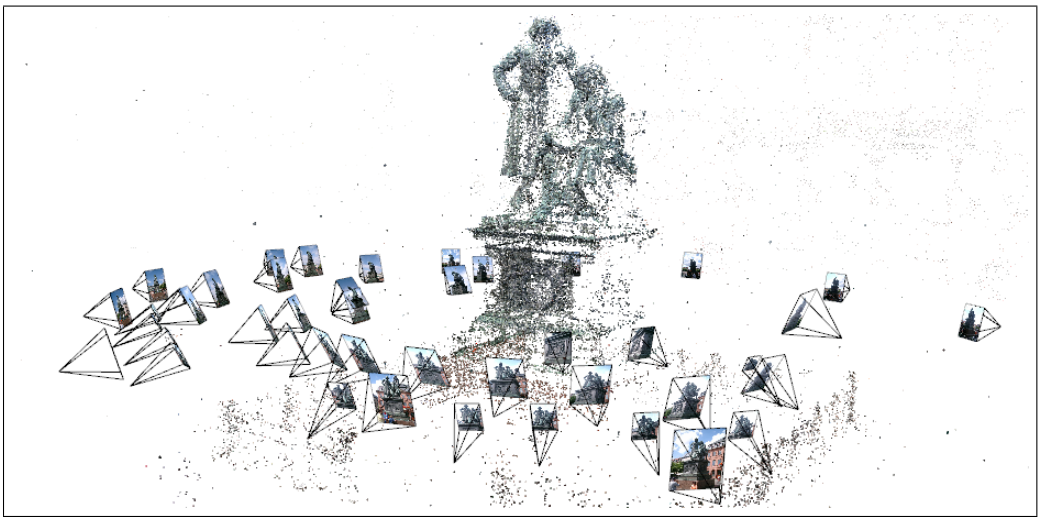
Aus gewöhnlichen, unkalibierten Fotos (mit überlappenden Bildanteilen) werden lokale Bildmerkmale (Feature-Points) extrahiert. Diese werden anhand überlappender Bildanteile in verschiedenen Fotos verknüpft. Anhand eines Referenz- bzw. Startbildes und dazugehörige Referenz-Feature-Points werden über einen Fitting-Mechanismus Kameras so verschoben und rotiert bis verknüpfte Feature-Points über einen Strahl möglichst gut zum Schnitt kommen. Somit werden die Kameraparameter rekonstruiert und aus korrespondierenden Feature-Points (Observed-Pixel) mehrerer Bilder ein dreidimensionaler Verknüpfungspunkt (ein Key-Point) berechnet. Die so gewonnenen Daten werden im weiteren Text als *Rohdaten* bezeichnet. *Abb. 2.20* zeigt diesen Ablauf. Aus gewöhnlichen, unkalibrierten Fotos eines Denkmals (mit überlappenden Bildanteilen) werden lokale Bildmerkmale (Feature-Points) extrahiert. Diese werden anhand überlappender Bildanteile in verschiedenen Fotos verknüpft. Anhand eines Referenz- bzw. Startbildes und dazugehörige Referenz-Feature-Points werden über einen Fitting-Mechanismus Kameras so verschoben und rotiert, bis verknüpfte Feature-Points (als entsprechend der Kamerafarben eingefärbte Ringe dargestellt) über einen Sehstrahl (ebenfalls entsprechend der Kamerafarben) in einen dreidimensionalen Verknüpfungspunkt verbunden sind. Als Ergebnis der Bilderwelten-Generierung werden folgende Daten (Rohdaten) bereitgestellt:

- Kamerapositionen im 3D-Raum,
- Kameraorientierungen im 3D-Raum,
- Brennweite der Kameras (transformiert in Pixeleinheiten),
- Key-Point-Positionen im 3D-Raum,
- Key-Point-Farben,
- Observed-Points,
- Observed-Pixel (in Pixeleinheiten),
- Observer.

(a) Lose Fotosammlung (mit überlappenden Bildanteilen)



(b) Fitting-Mechanismus anhand des Referenzbildes (in der Mitte) und korresondierender Feature-Points (entsprechend der Kamerafarbe eingefärbte Ringe)



(c) Ausschnitt des Denkmals aus der Bilderwelt von *Abb. 2.15*: Rekonstruierte Kameras und ihre und ein rekonstruierter Key-Point

Abbildung 2.20: Aus gewöhnlichen, unkalibrierten Fotos (mit überlappenden Bildanteilen) werden Kameraparameter und Key-Points rekonstruiert [NKG12] (Fotos von [Kru08]).

2.7 Zusammenfassung und Diskussion

In diesem Kapitel wurden grundlegende Mechanismen, Strukturen und Algorithmen erklärt, die für das Grundverständnis notwendig sind und/oder als notwendiges Werkzeug in der vorliegenden Arbeit verwendet werden.

Hierbei wurde auf die Repräsentation von Bilddaten (inklusive ihrer Transformation in Farbräume) und grundlegende Mechanismen der Bildverarbeitung und -analyse eingegangen. Des Weiteren wurden Projektionsverfahren und Depth-Maps erklärt. Der grundlegende Aufbau, der mathematische Hintergund und die Generierung von Bilderwelten waren ebenfalls Bestandteil dieses Kapitels.

Mithilfe der definierten Zielsetzung dieser Arbeit und den grundlegenden Kenntnissen dieses Kapitels werden im folgenden Kapitel aktuelle Forschungsarbeiten und Projekte mit verwandten (Teil-)Zielstellungen erläutert. Hierbei ist das Ziel, bestehende Verfahren und Mechanismen zu analysieren und auf Einsetzbarkeit in dieser Arbeit zu prüfen.

Kapitel 3

Heutiger Stand der Technik und Wissenschaft

3.1 Überblick

Aufbauend auf den Grundlagen werden in diesem Kapitel aktuelle Forschungsarbeiten und Projekte mit verwandten (Teil-)Zielstellungen erläutert. Hierbei ist das Ziel, bestehende Verfahren und Mechanismen auf Einsetzbarkeit in dieser Arbeit zu prüfen. Folglich werden Abgrenzungen gezogen.

3.2 Image-Based-Rendering und Bilderwelten

Dieses Unterkapitel erläutert Arbeiten aus dem Bereich *Image-Based-Rendering* sowie das Erstellen von Bilderwelten.

Die 3D-Computergrafik beschäftigt sich traditionell damit, dreidimensionale Modelle zu erzeugen und diese in einer zweidimensionalen Umgebung zu präsentieren. Im Gegensatz dazu wird mit Hilfe der *Computer Vision*, also dem maschinellen Sehen, der umgekehrte Weg bestritten. Das heißt, es werden aus zweidimensionalen Abbildungen einer Szene Informationen gewonnen, welche dreidimensionale Rückschlüsse zulassen. Zu diesen Informationen zählen zum Beispiel Kanten, Ecken oder andere auffällige Merkmale in einem Bild, welche sich vergleichen und einordnen lassen, um so Aussagen über deren Lage und Ausrichtung im dreidimensionalen Raum treffen zu können. Ziel dessen ist es, eine 3D-Szene auf Basis von gewöhnlichen Fotos (mit Bildüberlappungen)

darzustellen, ohne oder in nur einem sehr geringen Maß zuvor geometrische Informationen über die Szene zu kennen [ZC03]. Diese Technik wird als *Image-Based-Rendering* (*IBR*) bezeichnet [KLTS06] [KLT07]. Weitere Arbeiten zu *IBR* sind unter anderem in [BG01], [ZKU+04a], [ZKU+04b] und [RGL04] zu finden.

Die Idee, Fotografien für den Aufbau einer virtuellen *Bilderwelt* zu nutzen, besteht bereits seit über 25 Jahren. Als Vorreiter auf diesem Gebiet kann das Projekt Aspen Movie-Map [Lip80a] [Lip80b] gesehen werden, welches 1980 vorgestellt wurde. Hier wurde die Stadt Aspen in Colorado mit einem Auto abgefahren, an dem mehrere Kameras montiert waren, welche die Vorder-, Rück- und Seitenansichten in bestimmten Abständen fotografierten. Aus dieser Vielzahl von Bildern wurde ein virtueller Rundgang durch die Stadt ermöglicht. Mittels einfacher Interaktionsmöglichkeiten konnten eigene Routen durch die Stadt „durchfahren“ werden. Jedoch war es nicht möglich, fließende Übergänge zwischen den einzelnen Bildern zu erhalten oder neue Blickwinkel aus dem vorhandenen Bildmaterial zu erzeugen. Google präsentierte mit Street View [Goo07] einen zur Aspen Movie-Map sehr ähnlichen Service. Auch hier können Städte virtuell „durchfahren“ werden. Die dafür verwendeten Bilder wurden ebenfalls von einem Fahrzeug aufgenommen, auf dem eine 360°-Kamera mit insgesamt 11 Linsen montiert war. *Abb. 3.1* zeigt, wie Street View in den zweidimensionalen Kartendienst Google Maps integriert wird. Zwar bieten die Aufnahmen der 360°-Kamera die Möglichkeit der freien Drehung an einer Position, jedoch wird zwischen den einzelnen Bildern keine freie Navigation ermöglicht.

In [AYFC03] wird der Ansatz „Sea-of-Images“ vorgestellt, der ebenfalls aus einer Vielzahl von Bildern eine virtuelle Welt aufbaut. Hierfür wurden Fotos von einem Roboter, der sich nach einem vorgegeben Muster durch ein Gebäude bewegt, angefertigt. Es handelt sich dabei nicht um Aufnahmen aus fest ausgerichteten Perspektiven, sondern um ungerichtete Aufnahmen während der Fahrt durch das Gebäude. Die dabei entstehende Masse an Bildern wird in mehrfacher Auflösung in einer bestimmten Hierarchie gespeichert, welche es erlaubt, beim Durchlaufen des virtuellen Gebäudes die passenden Bilder in Echtzeit zu laden. Weiterhin wird gezeigt, wie es möglich ist, mit diesen Aufnahmen interpolierte Ansichten innerhalb des Gebäudes zu erzeugen. Hierzu

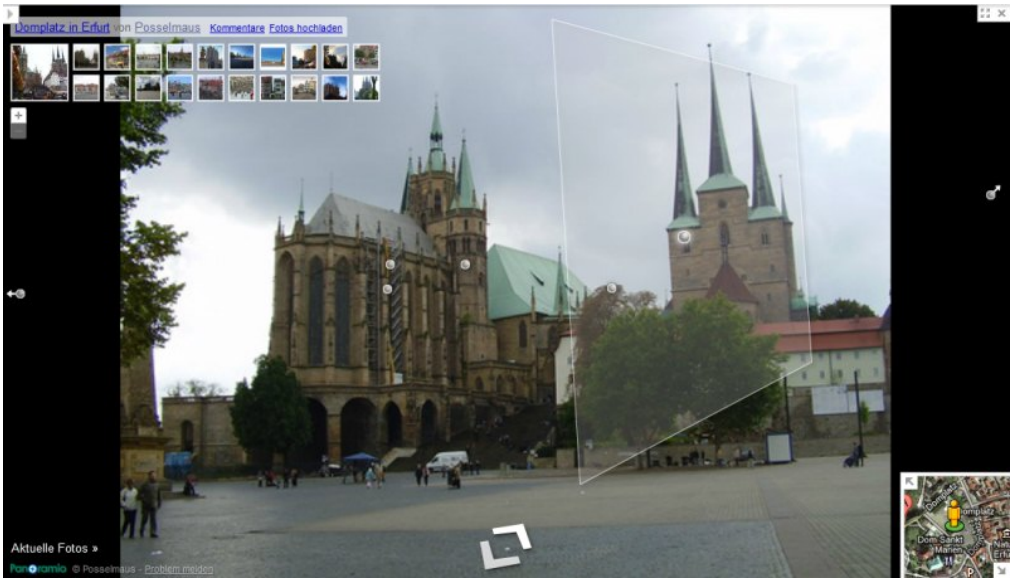


Abbildung 3.1: Google Street View: Erweiterung des zweidimensionalen Kartendienstes Google Maps mithilfe von Bilderwelten [Goo07].

werden echte Übergänge zwischen den einzelnen Bildern errechnet und generiert. Somit wird eine freie Navigation innerhalb des interaktiven Systems ermöglicht.

Bei den bisherigen Verfahren werden Bilder nach einem bestimmten Muster angefertigt. Zum einen durch fest montierte Kameras auf einem fahrenden Fahrzeug (Aspen MovieMap oder Google Street View), zum anderen durch eine vorgegebene Fahrstrecke (Sea-of-Images). Bilder, die unabhängig vom vorgegebenen Muster erstellt wurden, lassen sich nicht in die Anwendungen integrieren. Solche unabhängigen Bilder sind zum Beispiel in Onlinebilddatenbanken wie Flickr [Yah04] zu finden. Hier können Nutzer im Sinne des *User-Generated-Content* eigene Bilder hochladen und zur Verfügung stellen. Daraus ergeben sich große Datenmengen an unkalibrierten Fotos, die mit den bisher vorgestellten Verfahren nicht verarbeitet werden können.

Das Projekt PhotoTourism [SSS06b] [SSS06c] verarbeitet unkalibrierte Fotos der Plattform Flickr zu einer dreidimensionalen Bilderwelt. Es handelt sich dabei um einen *Photo Explorer*, mit dem eine Szene (auf Basis von dreidimensional angeordneten Fotos) im 3D-Raum erkundet werden kann. Aus einer Anzahl an Eingabebildern werden automatisch die dreidimensionalen Zusammenhänge der Bilder errechnet und auf dieser Basis präsentiert. Hierzu dient als Grundlage ein Structure-from-Motion-Algorithmus, dessen Arbeitsweise in

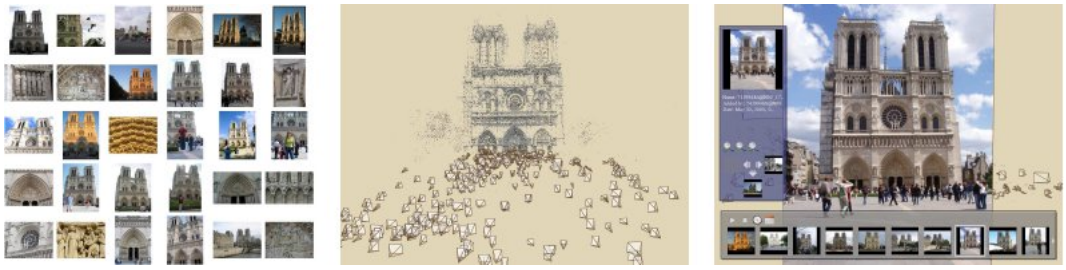


Abbildung 3.2: PhotoTourism: Darstellung einer Bilderweltenszene im Photo Explorer [SSS06b]. Die Szene zeigt die Kirche Notre Dame. Die Eingabebilder stammen von der Onlinebilddatenbank Flickr [Yah04].

Absatz *2.6.3* (*Generieren einer Bilderwelt*) erläutert wurde.

Abb. 3.2 zeigt, wie in PhotoTourism aus einer Vielzahl unterschiedlicher Eingabebilder eine Abbildung der Bilderweltenszene dargestellt wird. Eine kommerzielle Variante des PhotoTourism-Projektes veröffentlichte die Firma Microsoft unter dem Namen Photosynth [Mic08]. Eine Beispielszene findet sich in *Abb. 1.3*.

Hierbei wird ein Structure-from-Motion-Algorithmus genutzt, der aus einer Vielzahl von unterschiedlichen Bildern räumliche Zusammenhänge erkennt. In [BL05] wurde ein SfM-Algorithmus vorgestellt, der ohne Benutzereingriffe aus verhältnismäßig wenigen Eingabebildern eine 3D-Punktwolke erzeugt. Der Algorithmus setzt dabei keinerlei Kalibrierungen oder Konfigurationen der Bilder voraus und erkennt verschiedene Szenen innerhalb der Eingabebilder. *Abb. 3.3* zeigt, wie zwei unterschiedliche Punktwolken aus einer unsortierten Auswahl an Eingabebildern erstellt wurden. Zusätzlich werden die Kamerapositionen, aus denen die verwendeten Bilder aufgenommen wurden, dargestellt.

Ein weiterer SfM-Algorithmus ist *Bundler* [Sna10a], der bereits in Absatz *2.6.3* (*Generieren einer Bilderwelt*) erläutert wurde. Aufgrund seiner freien Open-Source-Lizensierung und der stabilen Ergebnisse wird für das Erzeugen der Bilderwelten auch in dieser Arbeit die Software Bundler verwendet.

Aufbauend auf den Projekten PhotoTourism und Bundler wird in [SSS08] die Vision einer weltumfassenden Bilderwelt aus Fotos des Internets als Eingabebilder beschrieben. Eine Weiterentwicklung und Optimierung der bestehenden Algorithmen wird in [AFS+10] und [AFS+11] erläutert und beschreibt, wie aus Internetfotos eine Stadt als Bilderwelten (und dazugehörige 3D-Punktwolken)

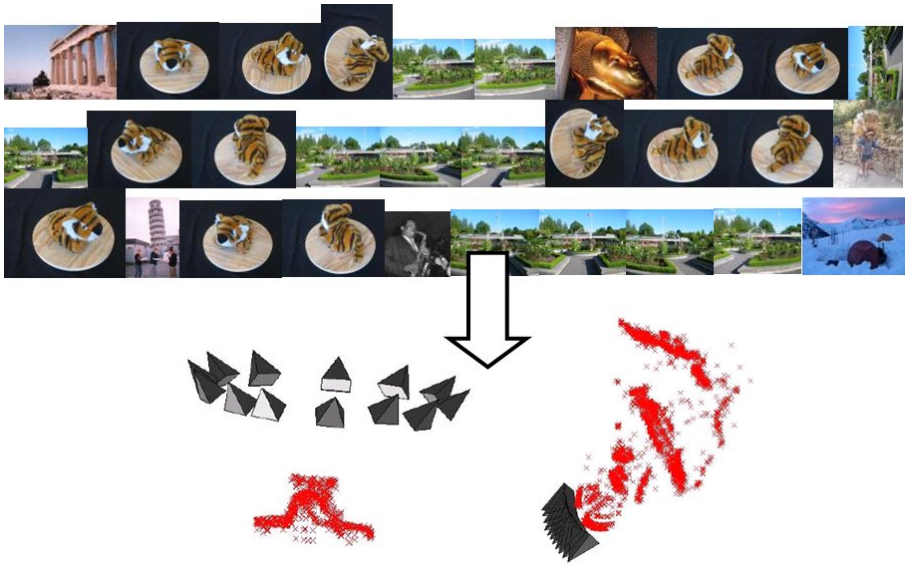


Abbildung 3.3: Erstellung zweier Punktwolken (automatisierte Detektion von zwei Szenen) aus einer unsortierten Auswahl an Eingabebildern. Zusätzlich werden die Kamerapositionen, aus denen die verwendeten Bilder aufgenommen wurden, dargestellt [BL05].

in einem kurzen Zeitraum rekonstruiert werden kann. *Abb. 3.4* zeigt ein beispielhaftes Ergebnis der Stadt Dubrovnik (Kroatien). Als Eingabedaten dienten 57845 Fotos der Onlinebilddatenbank Flickr. Onlinebilddatenbanken wie Flickr stellen zwar große Datenmengen von Sehenswürdigkeiten und Orten mit großen öffentlichen Interesse bereit, jedoch wird eine Stadt (oder ein Region) nie vollständig flächendeckend durch die angebotenen Fotos abgedeckt. Das Projekt PhotoCity stellt den Ansatz vor, als eine Art Online-Game Spieler/Fotografen zu „trainieren“ und fehlende Flächen durch Fotos der Spieler zu füllen [TSH+11].

Die vorgestellten Arbeiten beschäftigen sich mit der Generierung großer, flächendeckender Bilderwelten, jedoch nicht mit der Darstellung während der Navigation zwischen einzelnen Fotos innerhalb der Szene. In [GAF+10a] und [GAF+10b] wird das „ambient point cloud“-Verfahren erläutert. Es dient dazu, während der Navigation innerhalb der Bilderwelt zwischen den Fotos zu interpolieren. Anhand der Punktwolke werden auf den Fotos abgebildete Objekte einer Szene segmentiert und während der Navigation die entsprechenden Bildanteile der Objekte zwischen den Fotos interpoliert. Ein weiteres IBR-Verfahren für die

Abbildung 3.4: Ergebnisse der Bilderwelten-Generierung der Stadt Dubrovnik (Kroatien) [AFS+11]. Als Eingabedaten dienten 57845 Fotos der Onlinebilddatenbank Flickr.

dreidimensionale Darstellung der abgebildeten Szene einer Bilderwelt bei freier Navigation wurde vom Verfasser dieser Arbeit als Co-Autor in [KNGM12] veröffentlicht. Dieses Verfahren benötigt als Eingabedaten eine hohe Anzahl an Fotos bzw. wird eine detailreiche und ausgeprägte 3D-Punktwolke vorausgesetzt.

Arbeiten aus dem Bereich Bilderwelten und IBR, die als Zielstellung (neben einer ansätzlichen 3D-Punktwolke) 3D-Modelle oder bildbasierte 3D-Szenendarstellungen haben, werden unter der Bezeichnung *Image-Based-Modeling* im nachfolgenden Unterkapitel erläutert.

3.3 Image-Based-Modeling und 3D-Rekonstruktion aus Bildern

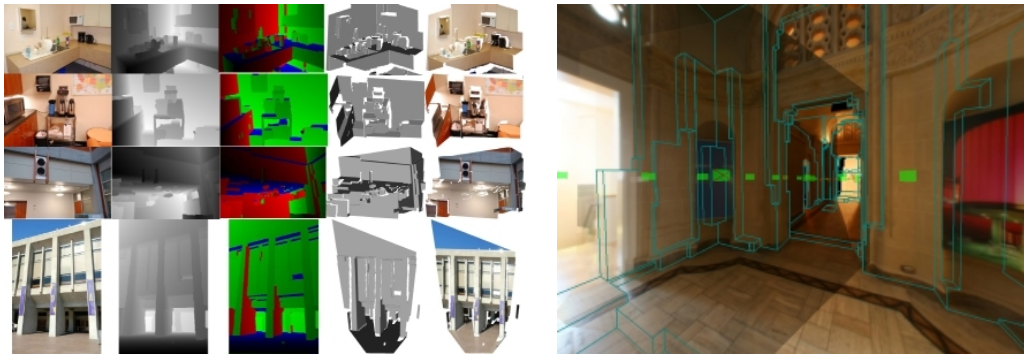
Eine spezielle Form des Image-Based-Renderings stellt das *Image-Based-Modeling* (kurz *IBM*) dar. Hier werden auf Grundlage der Daten von Bilderwelten (bzw. den Ergebnissen von Structure-from-Motion-Algorithmen) die Szene als bildbasierte 3D-Modelle dargestellt. In [GSC+07] wird ein *Multi-View-Stereo*-Algorithmus präsentiert, der sich mit Bilderwelten nutzen lässt. Hierbei werden neben Depth-Maps, auch 3D-Modelle aus dem Szeneninhalt einer Bilderwelt generiert. Ebenfalls für die Erzeugung eines 3D-Modells aus in Fotos abgebildeten Objekten dient das in [FP05] und [FP09] präsentierte Verfahren. Hier wird eine konvexe Hülle aus dem abgebildeten Objekt erzeugt. Als Eingabe-

Abbildung 3.5: Übersicht der einzelnen Schritte des PMVS-Verfahrens vom Eingabebild zur resultierenden konvexen Hülle [FP10].

daten werden exakt kalibrierte Fotos vorausgesetzt. Eine Erweiterung dieses Verfahrens für die Kombination mit einem *Bundle-Adjustment*-Algorithmus und somit der Nutzbarkeit unkalibrierter Fotoserien (bzw. Bilderwelten) stellt das *Patch-based-Multi-view-Stereo* (*PMVS*) dar [FP10]. In *Abb. 3.5* wird eine Übersicht der einzelnen Schritte vom Eingabebild zur resultierenden konvexen Hülle gezeigt. *Multi-View-Stereo*-Algorithmen benötigen hohe Rechenzeiten und große Speicherkapazitäten bei großen Mengen von Bilddaten. In [FCSS10] wird für dieses Problem ein Clustering-Verfahren vorgestellt, das den Umgang solcher Algorithmen mit großen Mengen an Bilddaten optimiert.

Die in diesem Absatz vorgestellten IBM-Verfahren benötigen als Eingabedaten eine hohe Anzahl an Fotos. Zudem setzen [FP05] und [FP09] exakt kalibrierte Fotos voraus. Da die Anforderung an diese Arbeit das Verwenden kleiner Szenen (also wenig Eingabefotos) gestellt wird, eignen sich die vorgestellten IBM-Verfahren für diese Arbeit nicht.

Multi-View-Stereo- sowie *Structure-from-Motion*-Algorithmen benötigen texturierte heterogene Flächen in den Fotos. Dies gestaltet sich bei Innenraumszenen mit einfarbigen Flächen (z. B. Wände) schwierig. In [FCSS09a] wird ein neuer Multi-View-Stereo-Ansatz erläutert, bei denen planare, einfarbige Flächen einer *Manhattan-World-Szene* trotzdem verarbeitet werden können. *Abb. 3.6(a)* zeigt Ergebnisse dieses Verfahrens. Das Verfahren setzt allerdings kalibrierte Eingabefotos voraus, weshalb ein Einsatz in dieser Arbeit nicht möglich ist. Eine ähnliche Arbeit findet sich in [FCSS09b]. Diese Arbeit kombiniert einen Structure-from-Motion-, einen Multi-View-Stereo- und einen Stereo-Algorithmus, der speziell für Innenraumszenen im Manhattan-World-Format



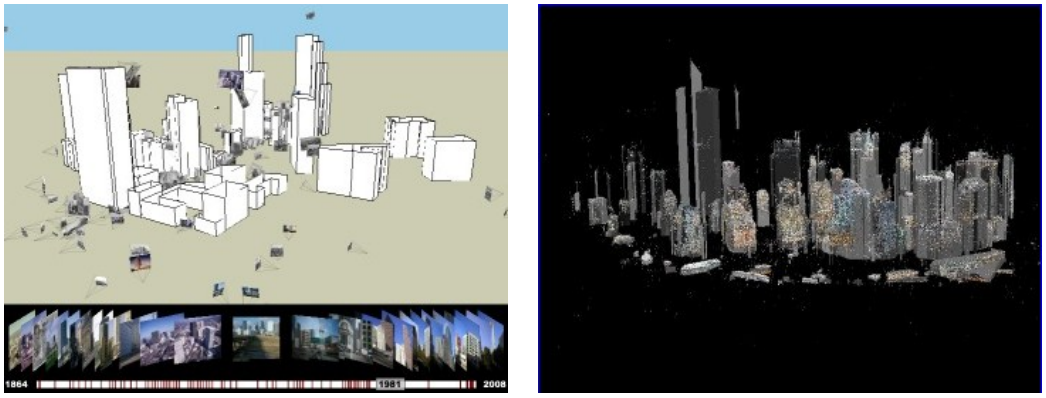
(a) (b)

Abbildung 3.6: Verfahren für das IBM von Innenraumszenen im Manhattan-World-Format: (a) [FCSS09a] und (b) [FCSS09b].

zugeschnitten ist. Als Ergebnis werden die Innenraumszenen als 3D-Modell bzw. als 3D-Abbildung dargestellt. Ein Bild dieses rekonstruierten 3D-Modells einer Beispielszene findet sich in *Abb. 3.6(b)*. Auch dieses Verfahren setzt kalibrierte Eingabefotos voraus und schließt daher den Einsatz in der vorliegenden Arbeit aus.

Weitere Arbeiten auf diesem Gebiet sind das Projekt Façade [DTM96], welches genutzt wurde, um halbautomatisch den Campus der Berkeley Universität in Kalifornien nachzubilden oder automatische Rekonstruktionsverfahren von Gebäuden wie in [DTC04] oder dem MIT City Scanning Project [TAB+03], bei dem tausende Bilder einer kalibrierten Kamera genutzt wurden, um ein 3D-Modell des MIT Campus zu generieren. In [AFM+06] werden aus mehreren Videoströmen von Städte-Szenarien eine bildbasierte 3D-Rekonstruktion erzeugt. Das 4-D-Cities Projekt [SDK07] erstellt anhand von historischen und aktuellen Aufnahmen der Stadt Atlanta ein so genanntes „spatio-temporal model“ (also ein zeitliches 3D-Modell) bei dem sich Veränderungen an den Bauwerken im Laufe der Jahre verfolgen lassen. Eine Weiterentwicklung dieser Arbeit für das Bestimmen der Aufnahmezeitpunkte in Fotoserien (ohne Aufnahmedaten) findet sich in [SD10]. *Abb. 3.7* zeigt Beispielergebnisse der beiden Projekte.

Das Unternehmen Autodesk ermöglicht mit ihrem Programm *123D Catch* [Aut12] aus (unkalibrierten) Fotoserien 3D-Meshes (eine 3D-Geometrie) der abgebildeten Szene zu generieren. Auch an dieser Stelle muss eine bestimmte



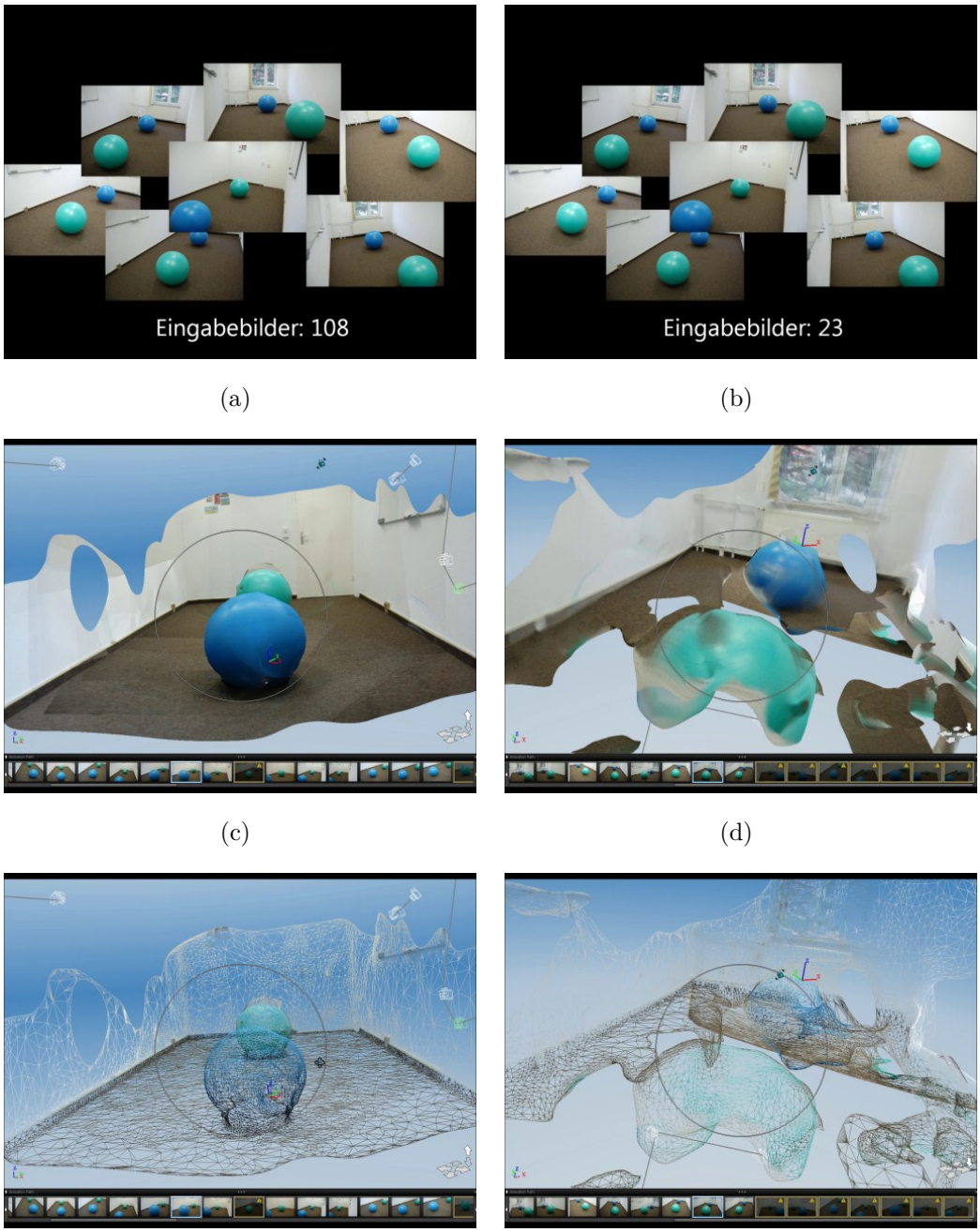
(a) (b)

Abbildung 3.7: (a) Das 4-D-Cities Projekt [SDK07] erstellt eine Art zeitliches 3D-Modell, mit dem sich Veränderungen an Bauwerken im Laufe der Jahre zurückverfolgen lassen sowie (b) die Weiterentwicklung für das Bestimmen der Aufnahmezeitpunkte in Fotoserien (ohne Aufnahmedaten) [SD10].

Anzahl an Eingabebilder vorausgesetzt werden, damit eine 3D-Rekonstruktion mit guten Ergebnissen erfolgen kann. *Abb. 3.8* zeigt eine Gegenüberstellung einer Szene in *123D Catch*, die als Eingabebilder in der linken Spalte 108 Fotos und in der rechten Spalte 23 Fotos verwendet. Die gleiche Szene mit weniger Fotos erzeugt eine unbrauchbare 3D-Rekonstruktion. Dies zeigt sich in fehlerhaften, löchrigen 3D-Meshes (siehe *Abb. 3.8(f)*). Folglich eignet sich 123D Catch nicht für kleine Szenen (mit wenigen Eingabefotos) und kann daher nicht in dieser Arbeit verwendet werden.

Im medizinischen und biologischen Bereich existieren viele Verfahren, die aus kalibrierten (Spezial-)Bildern eine *3D-Segmentierung* oder 3D-Rekonstruktion berechnen. In [NS10] hat der Autor der vorliegenden Arbeit in Zusammenarbeit mit einem weiteren Autor *3D-SurReAL* veröffentlicht. Dieses Verfahren erzeugt aus kalibrierten Bildserien eines Mikroskops ein 3D-Modell des abgebildeten realen Objekts. *Abb. 3.9* zeigt die einzelnen Schritte von 3D-SurReAL (von der Bildserie zu dem 3D-Modell). Arbeiten der 3D-Segmentierung von CT-Bildmaterial finden sich in [SNN+11], [TO09] und [KSS+07]. Da in diesen Arbeiten kalibrierte Fotos sowie spezielle medizinische Bildformate zum Einsatz kommen, eignen sich solche Verfahren nicht für die vorliegende Arbeit.

Die erläuterten Verfahren und Projekte aus Absatz *3.2* und diesem Ab-



(a) (b)

(c) (d)

(e) (f)

Abbildung 3.8: Gegenüberstellung einer Szene in 123D Catch [Aut12], die als Eingabebilder in der linken Spalte 108 Fotos und in der rechten Spalte 23 Fotos verwendet. Abb. (c+d) zeigt das erzeugte texturierte 3D-Modell und Abb. (e+f) das dazugehörige 3D-Mesh. Die gleiche Szene mit weniger Fotos erzeugt eine unbrauchbare 3D-Rekonstruktion. Dies zeigt sich in einem fehlerhaften, löchrigen 3D-Mesh.

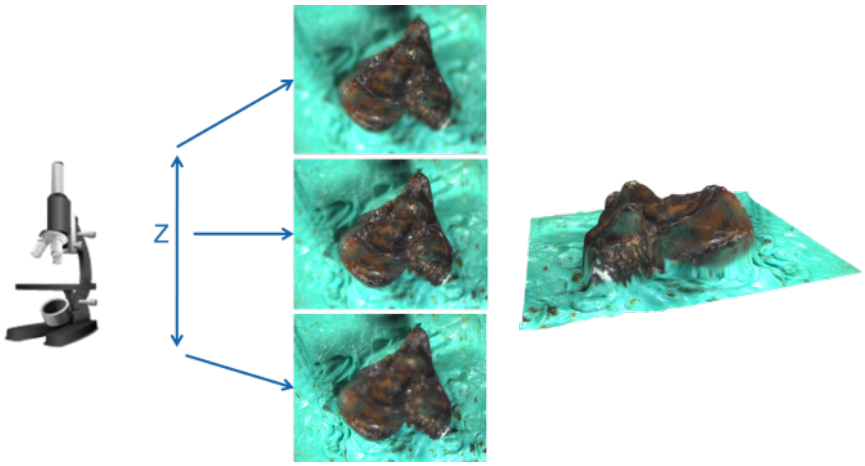


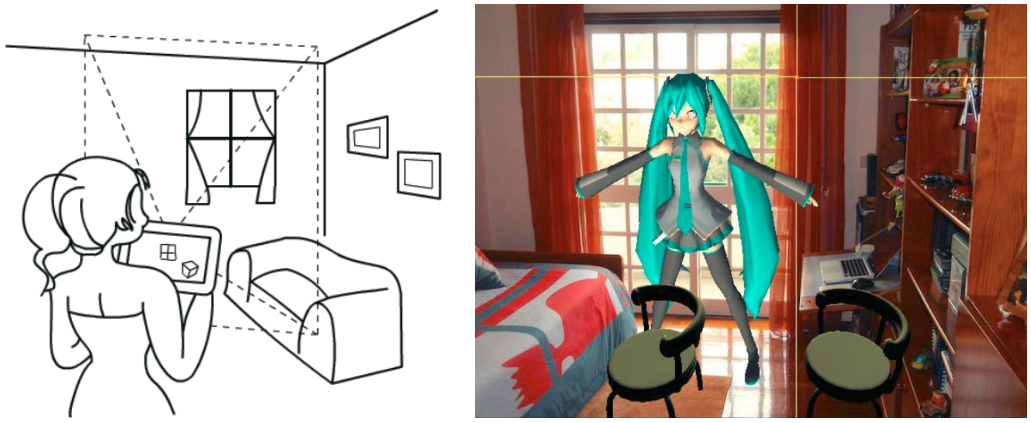
Abbildung 3.9: Das 3D-SurReAL-Verfahren erzeugt aus kalibrierten Bildserien eines Mikroskops ein 3D-Modell des abgebildeten realen Objekts [NS10].

satz beschäftigen sich ausschließlich mit den Bereichen IBR, IBM und 3D-Rekonstruktion aus Fotos (und Bilderwelten) ohne Erweiterungen durch virtuelle 3D-Objekte im Sinne der Augmented Reality.

3.4 Authentische Augmentierung von Bildern

In diesem Absatz werden Arbeiten aus den Bereich der authentischen Augmentierung von Bildern vorgestellt.

Bilderwelten ermöglichen die automatisierte Erstellung von dreidimensionalen Visualisierungen aus Fotos. Ein weiterführender Schritt zur Nutzung von Bilderwelten als Planungsvisualisierung stellt die Anreicherung (Integration) mit virtuellen Objekten dar, welche nahtlos und automatisiert in die rekonstruierte Umgebung eingefügt werden sollen. Auf Basis von geometrischen Berechnungen einer rekonstruierten Welt können die Kameras der virtuellen Objekte entsprechend der Kamera der Szene kalibriert werden. Somit soll die korrekte Perspektive für das Einblenden der virtuellen Objekte ermittelt werden. In [NC12] wird ein Verfahren für das perspektivisch korrekte Einbetten virtueller Objekte in Fotos erläutert. Hierbei müssen die Fotos mit einem mobilen Endgerät aufgenommen werden. Während des Aufnahmeprozesses muss der Nutzer solange einen Bildausschnitt auswählen, bis dieser dem Verfahren genügend Details im Bildinhalt liefert. Somit beinhaltet dieses Verfahren einen



(a) (b)

Abbildung 3.10: Ein interaktives Verfahren für das „magnet-ähnliche“ Platzieren virtueller Objekte an reale Objekte in einem Foto [NC12].

interaktiven Schritt. Der detailreiche Bildinhalt eines Fotos wird nach bekannten Elementen, wie Oberflächen, Wände und Raumorientierung analysiert. Einzubettende virtuelle Objekte platzieren sich „magnet-ähnlich“ an diese Elemente. *Abb. 3.10* zeigt Aufnahmeprozess und ein Ergebnisbild. Da ein interaktiver Schritt vorausgesetzt und das primäre Ziel keine authentische Darstellung der virtuellen Objekte ist, kann dieses Verfahren in der vorliegenden Arbeit nicht verwendet werden.

Das Verfahren von [PS02] beschäftigt sich ebenfalls mit dem Integrieren virtueller Objekte in Einzelbildern. Diese Arbeit unterliegt der Beschränkung, dass das Einblenden des Objektes nur für die einzelne Sicht korrekt dargestellt wird und der Vorgang zur Kalibrierung der Kamera des einzubettenden Objektes nicht vollständig automatisiert abläuft.

In [NBKG09] wurde vom Autor unter Mitarbeit von Co-Autoren ein Verfahren veröffentlicht, womit virtuelle 3D-Objekte in Fotos perspektivisch korrekt eingebettet werden. Hier wird die Kamerakalibrierung gegenüber [PS02] vereinfacht und dem Nutzer lediglich die Auswahl von drei automatisch ermittelten Kanten im abgebildeten Innenraum des Einzelfotos abverlangt. Mithilfe der Kanten wird der Fluchtpunkt ermittelt und damit die Bodenfläche des Raumes im 3D-Raum rekonstruiert, sowie die Kamera der einzubettenden virtuellen Objekte kalibriert. Weiterführend wird eine Begrenzung der Positionierung der einzubettenden Objekte bestimmt, sodass diese nur auf der Bodenfläche

(a) (b)

Abbildung 3.11: Das Ergebnis einer perspektivisch korrekten Integration virtueller 3D-Objekte in ein Foto durch das vom Autor dieser Arbeit in *Abb. 3.11* vorgestellte Verfahren.

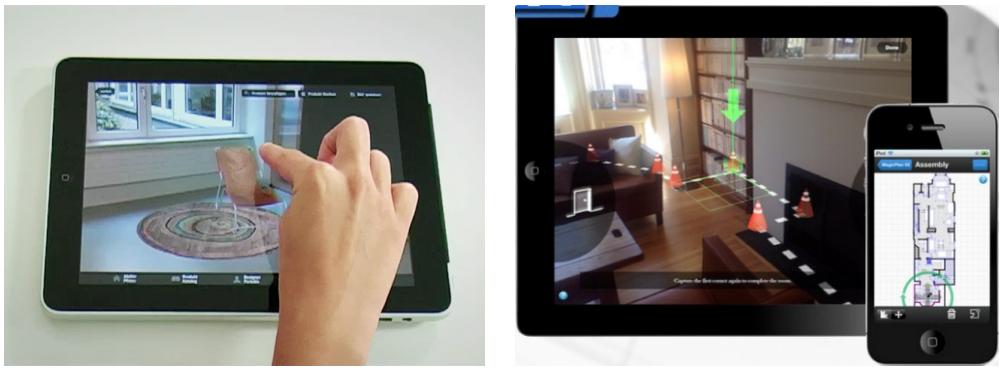
positioniert werden können. *Abb. 3.11* zeigt das Ergebnis einer perspektivisch korrekten Integration virtueller 3D-Objekte in ein Foto. Auch hier werden die virtuellen Objekte nur für die Sicht des Einzelfotos korrekt dargestellt, weshalb dieses Verfahren sich für Bilderwelten und folgerichtig für die vorliegende Arbeit nicht eignet.

Eine automatisierte Einbettung virtueller Objekte in Einzelfotos realisiert die Software *Click and Design* [KG12]. Hierzu muss das Eingabefoto in der Abbildung einen Papier-Marker enthalten. Anhand des Markers können die Bodenfläche und die richtige Perspektive für die einzubettenden virtuellen Objekte bestimmt werden. Eine Behandlung von optischer Verdeckung sowie Positionsbegrenzungen am Rand der Bodenfläche gibt es nicht. *Abb. 3.12* zeigt ein augmentiertes Beispielfoto. Wie die zuvor erläuterten Arbeiten bezieht sich auch diese auf Einzelbilder.

Eine Lösung für mobile Endgeräte bietet die Softwarereihe der Firma Metaio [Met12]. Hier können über Tablets oder Smartphones Kamerabilder augmentiert werden, wie in *Abb. 3.13(a)* dargestellt wird. Weitere Projekte für die AR-basierte Planungsvisualisierung von Kamerabildern mobiler Endgeräte stellen die Produkte *Happy Measure* [Lab12] und *Magic Plan* [Sen12] (siehe *Abb. 3.13(b)*) dar.

Die Software *Click and Design* sowie die vorgestellten mobilen Produkte arbeiten mit Spezialkonfigurationen (Marker oder Spezialhardware, wie Sensoren

Abbildung 3.12: *Click and Design*: Automatisierte Einbettung virtueller Objekte in Einzelfotos mithilfe eines Papier-Markers [KG12].



(a) (b)

Abbildung 3.13: AR-basierte Produkte für die Planungsvisualisierung von Kamerabildern mobiler Endgeräte: (a) *Metaio* [Met12]; (b) *Magic Plan* [Sen12].

im mobilen Endgerät) und sind auf die Augmentierung eines einzelnen Bildes ausgerichtet. Deshalb eignen sich diese Programme nicht für den Einsatz in Bilderwelten.

3.5 Depth-Maps und Verdeckung in augmentierten Bildern

Neben der Erzeugung von 3D-Geometrien existieren auch Ansätze, die anstelle von Geometrie-Daten die *Kameratiefe* bzw. zu Fotos korrespondierende *Depth-Maps* rekonstruieren. Rekonstruierte Kameratiefen (oder Depth-Maps) werden benötigt, um bei der Augmentierung von Bildern eine authentische Verdeckung zu ermöglichen. Dieser Absatz zeigt bestehende Arbeiten und Projekte auf, die sich mit der Thematik Depth-Map-Gewinnung und authentische Verdeckung augmentierter Bilder befassen.

Das *Immersion*-Projekt [NDWV12] [AMM+95] ermöglicht eine Pseudo-Bewegung um einen Meter in der abgebildeten Szene auf Fotos. Dafür werden Stereo-Fotografien mit exakt ausgemessenem Abstand zwischen zwei Kameras aufgenommen und damit eine Depth-Map errechnet. Mithilfe der Depth-Map können einzelne Bildfragmente so verschoben werden, dass der Eindruck einer Vorwärts- oder Rückwärts-Bewegung entsteht. Die Depth-Map wurde mittels speziell entwickelten Stereo-Korrespondenz-Algorithmus bestimmt. *Abb. 3.14* zeigt an einem Beispiel den Arbeitsablauf des Immersion-Projektes. Abgrenzend zu unkalibrierten Kameras von Bilderwelten sind hier exakt parallel ausgerichtete und um einen genauen Abstand entfernte Kameras oder Stereokameras notwendig.

In [BCN08] wird ein Verfahren erläutert, wie mithilfe einer speziellen Objektivblende eine Depth-Map von einem Einzelbild erstellt werden kann. Die Idee dieser Arbeit ist, ein Vordergrundobjekt aus einem Foto zu extrahieren und mit einem anderen Hintergrund zu versehen. Da hierfür ein spezielles, von der Arbeitsgruppe selbst entwickeltes Objektiv mit einem RGB-Filter in der Blende vorausgesetzt wird, kann dieses Verfahren nicht in der hier vorliegenden Arbeit verwendet werden. Ein weiteres Verfahren [MNBN07] zur Bestimmung einer Depth-Map aus einem Einzelbild nutzt Fotos mit einem Punkt-Raster im Hintergrund. Mithilfe der Defokussierung der Punkte in Kombination mit

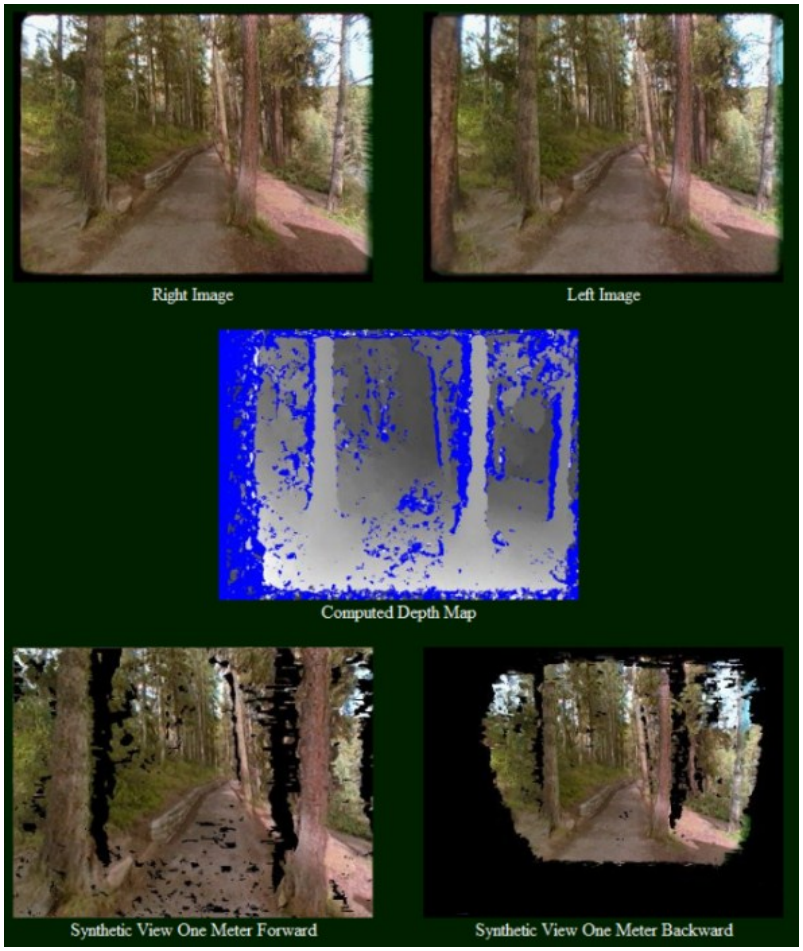
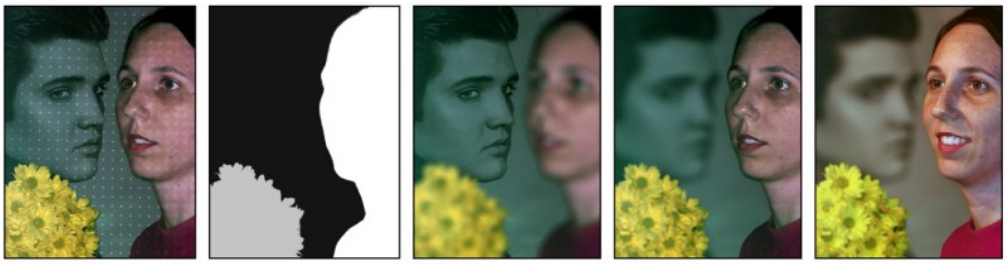


Abbildung 3.14: Der Arbeitsablauf des *Immersion*-Projekts [NDWV12] [AMM+95]: Stereobilder (oben), die errechnete Depth-Map (mittig), 2 Ergebnisbilder (unten).

Figure 1: *(a) Top: camera lens with color filters placed in the aperture. Bottom: filter arrangement. (b) Captured image. The background color is misaligned (see Fig. 16(c) for a closeup). (c) Estimated depth (the darker, the nearer). (d) Extracted matte. (e) Composite image.*

(a) Arbeitsablauf von [BCN08]



(b) Arbeitsablauf von [MNBN07]

Abbildung 3.15: Verfahren für die Bestimmung einer Depth-Map aus Einzelbildern mit spezieller Hardware/Konfiguration.

einer Farbsegmentierung können Vordergrundobjekte identifiziert und eine Depth-Map bestimmt werden. Da alle Fotos mit einem orthogonal zur Kamera stehenden Punkt-Raster im Hintergrund versehen werden müssen, kann dieses Verfahren ebenfalls nicht in dieser Arbeit angewandt werden. *Abb. 3.15* zeigt die Arbeitsabläufe der Verfahren [BCN08] und [MNBN07].

Eine andere Vorgehensweise für die Erstellung einer Depth-Map stellt das in [KS04] erläuterte Verfahren dar. Hier werden mehrere Bilder bzw. Bildausschnitte genutzt und daraus eine Depth-Map generiert (siehe *Abb. 3.16*). Zwar müssen keine Stereobilder verwendet werden, jedoch muss dennoch eine geometrische Kalibrierung der Eingabebilder vorgenommen werden. Folglich kann dieses Verfahren nicht in der vorliegenden Arbeit verwendet werden.

Auch thematisch von dieser Arbeit abweichende Systeme nutzen das Erstellen von Depth-Maps als Zwischenschritt für weitere Ziele. Als Beispiel soll die Arbeit von [EN05] genannt werden. Hier werden weiche Schatten mithilfe von Lichtstrahlen, vorberechneten Kanten-Masken-Tabellen und daraus generierten Depth-Maps gerendert. Weitere Arbeiten erstellen keine direkte Depth-Map oder nutzen vollständige 3D-Rekonstruktionen, sondern versuchen Verdeckungs-

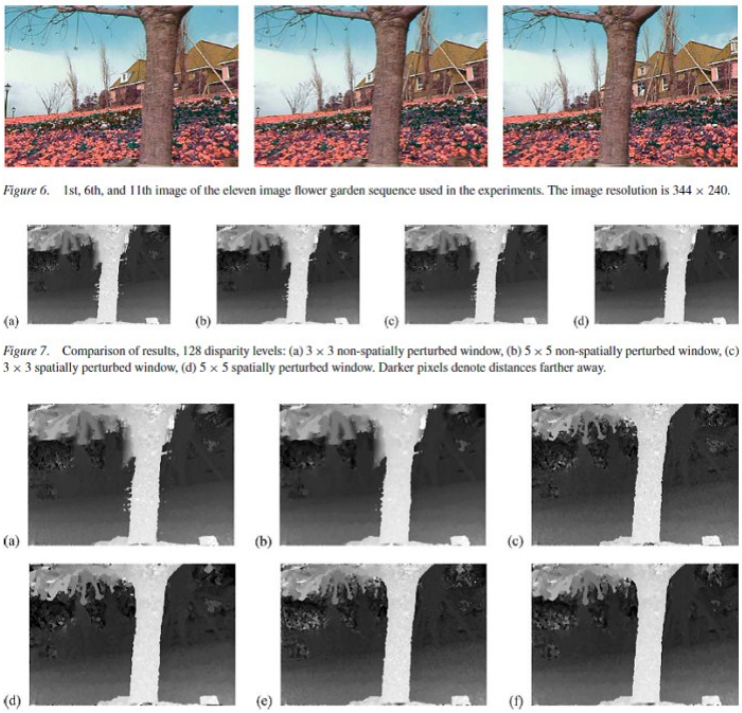


Abbildung 3.16: Eingabebilder (aufgenommen aus sehr ähnlicher Position und Ausrichtung) und daraus resultierende Depth-Maps nach [KS04].

probleme interaktiv zu lösen. In [FBS04] wird ein System für eine authentische Verdeckung von AR-gestützten medizinischen Aperturen präsentiert. Hierbei werden die 3D-Modelldaten der medizinischen Instrumente eingelesen und Hüllkörper-Daten generiert. Diese werden anschließend in einer bereits bestehenden AR-Anwendung integriert, wie in *Abb. 3.17* verdeutlicht wird. Da 3D-Modelldaten vorausgesetzt und mit einem Marker gearbeitet wird, eignet sich dieses Verfahren nicht für die vorliegende Arbeit.

Ein weiteres, interaktives System wird in [PLF07] präsentiert. Hier können Fotos, die als formbare, texturierte 3D-Modelle vorliegen, um zusätzliche Bildanteile und Objekte erweitert und „retexturiert"werden. Dafür muss sich die Beleuchtungs- bzw. die Schattensituation der Szene ändern. An Stellen, bei denen eine veränderte Beleuchtungssituation vorliegt, können neue Bildanteile eingeblendet werden. Eine veränderte Beleuchtungssituation wird durch Verdeckungen des Bildes realisiert, wie in *Abb. 3.18* gezeigt wird. Da interaktiv die Beleuchtungssituation (durch eine Nutzerinteraktion) durchgeführt wird und Fotos als formbare, texturierte 3D-Modelle vorliegen müssen, eignet sich auch

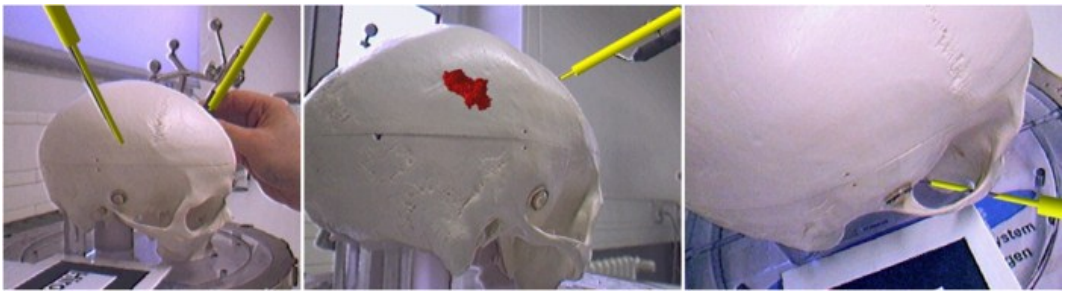


Abbildung 3.17: Authentische Verdeckung von AR-gestützten medizinischen Apparaturen auf der Basis von gegebenen 3D-Modelldaten der medizinischen Instrumente [FBS04].

Abbildung 3.18: Erweiterung und „Retexturierung“ von Fotos durch veränderte Beleuchtungssituation. Eine veränderte Beleuchtungssituation wird durch Verdeckungen des Bildes realisiert [PLF07].

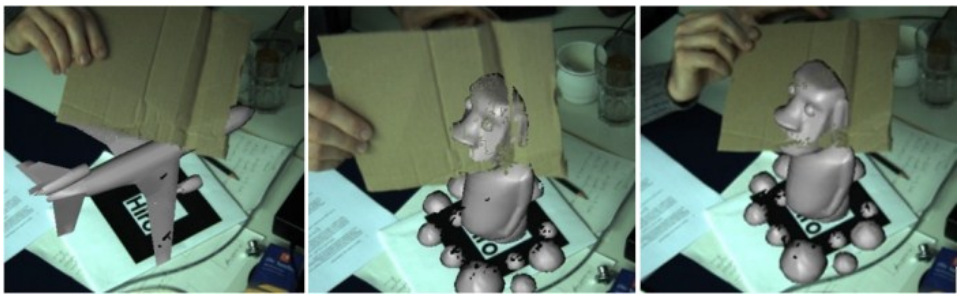


Abbildung 3.19: Virtuelle Objekte werden im laufenden Video durch reale Objekte verdeckt [FHS07]. Als Voraussetzung wird eine Stereo-Kamera benötigt.

dieses Verfahren nicht für die vorliegende Arbeit.

Ebenfalls ein interaktives System für authentische Verdeckung stellt das in [FHS07] vorgestellte Verfahren dar. Eingebettete, virtuelle Objekte werden im laufenden Video durch reale Objekte verdeckt, wie in *Abb. 3.19* demonstriert wird. Als Voraussetzung wird eine Stereo-Kamera benötigt, die eine Tiefenerkennung durchführt bzw. eine Depth-Map der Szene erstellt. Daher kann dieses System ebenfalls nicht für die vorliegende Arbeit verwendet werden.

Ein ähnliches Verfahren, jedoch ohne Nutzung einer direkten Stereo-Kamera, wird in [ZP08] erklärt. Hier werden über zwei Videoströme virtuelle Objekte in die abgebildete reale Szene eingebettet, Tiefeninformationen mit einem Stereo-View-Algorithmus abgeschätzt und somit eine korrekte Verdeckung ermöglicht, wie in *Abb. 3.20* dargestellt wird. Voraussetzung für dieses Verfahren sind zwei geometrisch kalibrierte Kameras für den Stereo-Effekt. Da die vorliegende Arbeit auf kalibrierte Kameras verzichtet, eignet sich dieser Stereo-View-Algorithmus ebenfalls nicht.

In [BF02] wird ein Verfahren zur Behandlung von Verdeckungen von realen Objekten durch virtuelle Objekte (also eine umgekehrte Situation als in dieser Arbeit) vorgestellt. Hier werden Schatten eines virtuellen Objektes erzeugt und auf das reale Objekt projiziert. Die Schattenfläche wird anschließend genutzt, um das virtuelle Objekt ohne falsche Verdeckungen einzubetten.

Ein semiautomatisches Verfahren für Videos, bei dem der Nutzer die Konturen eines potentiellen *Occluder* in mehreren Frames einzeichnet, wird in [LB00] erläutert. Mithilfe der Umrisse des Occluders kann das abgebildete Objekt ansatzweise in 3D rekonstruiert werden und beim Einblenden von virtuellen Objekten hinsichtlich potentieller Verdeckungen berücksichtigt werden. Auf-

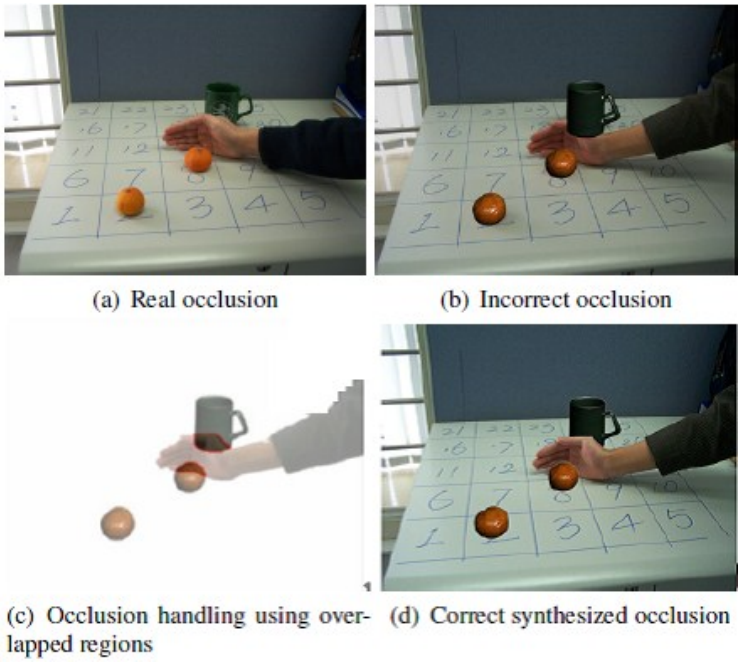


Abbildung 3.20: Korrekte Verdeckung virtueller Objekte durch einen Stereo-View-Algorithmus, der zwei geometrisch kalibrierte Kameras verarbeitet [ZP08].

grund der Verwendung eines Videostroms sowie der Voraussetzung, dass der Nutzer die Konturen der zu verdeckenden Objekte manuell einzeichnen muss, eignet sich dieses System ebenfalls nicht für die vorliegende Arbeit.

3.6 Segmentierung und Konturdetektion in Bildern

Als Werkzeuge für die Gewinnung weiterer Szeneninformationen aus den Bildinhalten von Fotos wurde in Absatz *2.3* (*Bildverarbeitung und Bildanalyse*) grundlegende Verfahren der Bildverarbeitungsbereiche Segmentierung, Kanten- und Konturdetektion und Feature-Point-Detektion eingeführt. In diesem Absatz werden aktuelle Arbeiten und Ansätze aus diesen Bereichen vorgestellt.

In [GPW03] wird ein biologisch motivierter Algorithmus erläutert, der die Resultate des *Canny-Algorithmus* für Bilder mit Landschafts- und Naturmotiven optimiert. Hierfür werden isolierte Konturen von Kanten/Konturen als Teil einer Textur unterschieden. Da sich der Einsatz auf Bilder mit Landschafts- und Naturmotiven beschränkt, ist der Einsatz in der vorliegenden Arbeit nicht

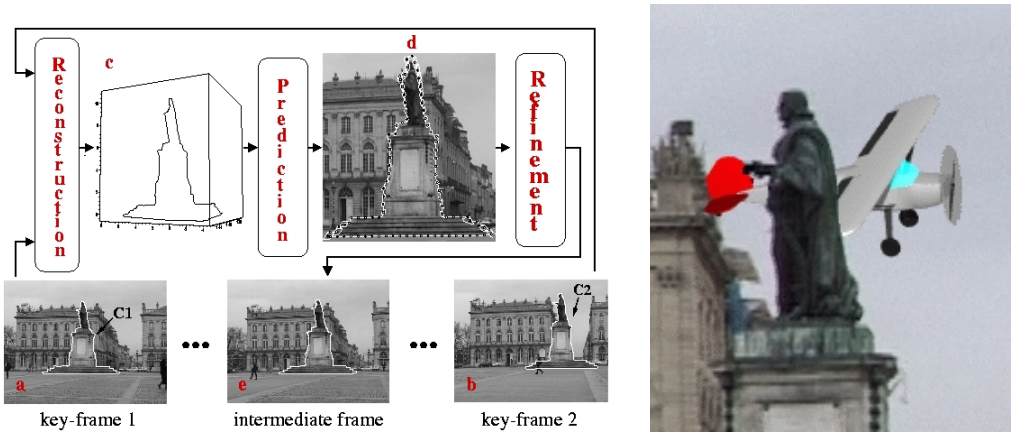


Abbildung 3.21: Semiautomatisches Verfahren für eine korrekte Verdeckung eingeblendeter virtueller Objekte in einem Videostrom [LB00].

möglich. Ein weiterer Ansatz für Bilder mit Landschafts- und Naturmotiven wird in [PCPN07] präsentiert. Im ersten Schritt wird der Gradient eines Fotos für unterschiedliche Auflösungen ermittelt. Darauf aufbauend werden Kanten und Konturen umliegender Texturen unterdrückt. Ein Kontur-orientierter Binär-Algorithmus detektiert vorrangig „lange“ Konturlinien, da dies ein Indiz für die Zugehörigkeit zu einem abgebildeten Objekt darstellt. Auch dieser Ansatz eignet sich nicht in dieser Arbeit, da er sich ausschließlich auf Bilder mit Landschafts- und Naturmotiven beschränkt.

Weitere Arbeiten aus den Bereichen der Segmentierung und Konturdetektion in Bildern setzen i. d. R. Fuzzylogik und/oder künstliche neuronale Netze ein. Beispielhaft sollen an dieser Stelle die Verfahren [GS06], [LBLM08], [MFM04], [MAFM08] und [CSS+10] genannt werden. Der Nachteil solcher Ansätze stellt unter anderem die Notwendigkeit von Trainingsdaten (z.B. Bilder, die von Personen händisch „gelabelt“ und skizziert wurden) dar. Da das Erstellen von Trainingsdaten im Rahmen dieser Arbeit sich sehr schwierig gestaltet, soll in der vorliegenden Arbeit kein Fuzzylogik-System und kein künstliches neuronales Netzwerk aufgebaut werden. Folglich können Segmentierungs- und Konturdetektionsverfahren aus diesen Bereichen nicht eingesetzt werden.

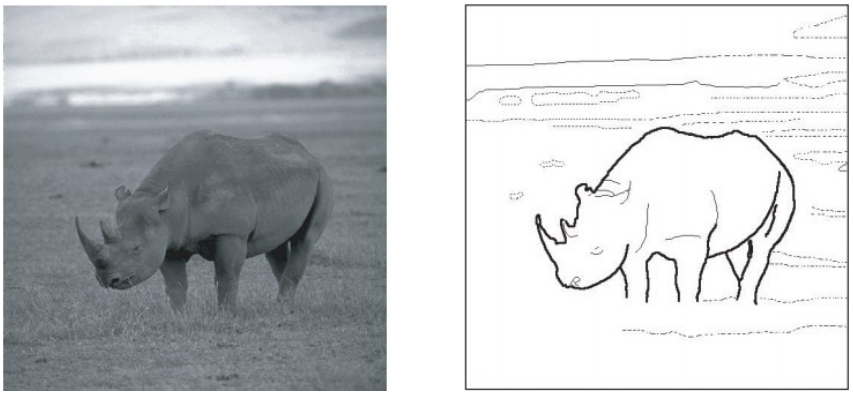


Abbildung 3.22: Konturdetektion von Bildern mit Landschafts- und Naturmotiven nach [PCPN07].

3.7 Zusammenfassung und Diskussion

In diesem Kapitel wurden, basierend auf der definierten Zielsetzung dieser Arbeit und den grundlegenden Kenntnissen aus Kapitel 2 aktuelle Forschungsarbeiten und Projekte mit verwandten (Teil-)Zielstellungen erläutert. Hierbei war das Ziel, bestehende Verfahren und Mechanismen auf Einsetzbarkeit in dieser Arbeit zu prüfen und abzugrenzen.

Hierzu wurden Forschungsarbeiten und Projekte aus den Bereichen

- Image-Based-Rendering,
- Bilderwelten,
- Image-Based-Modeling,
- 3D-Rekonstruktion aus Bildern,
- Augmentierung von Bildern,
- Depth-Map-Gewinnung und Verdeckung in augmentierten Bildern,
- Segmentierung und Konturdetektion

zusammengetragen. Mithilfe der Erkenntnisse der ersten beiden Kapitel und dem Zusammentragen bestehender Forschungsarbeiten in diesem Kapitel, wird im folgenden Kapitel eine Abgrenzung und Einordnung dieser Arbeit sowie eine detaillierte und technisch versierte Zielstellung mit definierten Anfordungen und Grenzen zusammengetragen.

Kapitel 4

Einordnung, erweiterte Anforderungen und verfeinerte Zielsetzung

4.1 Überblick

In Kapitel *1* wurde als Motivation die Verwendung von augmentierten Bilderwelten als 3D-Planungsvisualisierungen und die damit verbundene Herausforderung, die Augmentierung möglichst authentisch darzustellen, erläutert. Daraus ließen sich *Augmentierungsschwerpunkte* für die authentische Integration virtueller Objekte in Bilderwelten ableiten. Das Hauptaugenmerk dieser Arbeit liegt dabei auf dem Schwerpunkt *Verdeckung*. Mithilfe der Erkenntnisse der ersten drei Kapitel wird in diesem Abschnitt eine Abgrenzung und Einordnung dieser Arbeit sowie eine detaillierte und technisch versierte Zielstellung mit definierten Anfordungen und Grenzen zusammengetragen.

4.2 Abgrenzung und Einordnung dieser Arbeit

Dieser Absatz stellt in Bezug zu dem heutigen Stand der Technik und Wissenschaft (siehe Kapitel *3*) die Abgrenzungen zu der vorliegenden Arbeit her und ordnet somit diese Arbeit in die Wissenschaft ein. Hierfür werden stets die Kernanforderungen dieser Arbeit (siehe Absatz *1.3*) berücksichtigt. Für die in Kapitel *3* analysierten Arbeiten wurde bereits die Nutzbarkeit bestehender

Verfahren für die vorliegende Arbeit bewertet. Folglich wird an dieser Stelle eine allgemeine, für die Einordnung in die Wissenschaft notwendige Abgrenzung gezogen.

Für das Erreichen einer authentischen Verdeckung in einer 3D-Szene stellt die 3D-Rekonstruktion einen etablierten Weg in der 3D-Computergrafik dar. Hierfür wurden verschiedene Ansätze in Absatz *3.3* (*Image-Based-Modeling und 3D-Rekonstruktion aus Bildern*) analysiert. Programme, wie *123D Catch* [Aut12] erzeugen aus einer Serie von Bildern eine 3D-Geometrie der abgebildeten Szene. *Multi-View-Stereo-* [GSC+07] und *Patch-based-Multi-view-Stereo-*Verfahren [FP05] [FP09] [FP10] ermöglichen die Erzeugung von Depth-Maps und/oder konvexen Hüllen abgebildeter Objekte. Diese Verfahren und Programme setzen eine hohe Anzahl an Eingabefotos mit einer entsprechend hohen Flächenabdeckung der abgebildeten Szene voraus. Da die Kernanforderungen der vorliegenden Arbeit die Verarbeitung kleiner Szenen (Szenen mit einer geringen Anzahl an Fotos) verlangen, können diese Verfahren und Programme nicht verwendet werden. Einige Arbeiten ermöglichen eine 3D-Rekonstruktion aus weniger oder Einzelfotos [FCSS09a] [FCSS09b], jedoch werden kalibrierte Eingabefotos vorausgesetzt. In dieser Arbeit soll jedoch auf kalibrierte Kameras/Fotos verzichtet werden. Weiterführend soll an dieser Stelle erwähnt werden, dass die meisten in Absatz *3.2* und Absatz *3.3* analysierten Verfahren nicht das Vorhaben beabsichtigen, die Bilderwelt oder bildbasierte Szene zu augmentieren. Folglich eignen sich diese Verfahren nicht für die Zielstellung dieser Arbeit.

Die hier vorliegende Arbeit ordnet sich in die Bereiche *Image-Based-Rendering* und *Image-Based-Modeling* ein, mit dem Hintergrund Bilderwelten authentisch zu augmentieren und integrierte virtuelle Objekte authentisch zu verdecken.

Eine authentische Verdeckung erfordert stets das Rekonstruieren von Tiefeninformationen. In Absatz *3.5* (*Depth-Maps und Verdeckung in augmentierten Bildern*) wurden Verfahren für die Gewinnung von Depth-Maps sowie für eine korrekte Verdeckung in augmentierten Bildern analysiert. Diese Verfahren erfordern spezielle Hardware, Stereo-Kameras, Kalibrierungen über Marker oder gegebenen 3D-Modelldaten der Szene [ZP08] [BCN08] [FHS07] [MNBN07] [FBS04]. Diese Anforderungen können nicht in der vorliegenden Arbeit vorausgesetzt werden. Somit ordnen sich Teilschritte dieser Arbeit (siehe Absatz

1.4) in den Bereich der 3D-Rekonstruktion, genauer definiert, in die Bereiche Rekonstruktion von Tiefeninformationen und Gewinnung von Depth-Maps aus Bildern bzw. Bilderwelten ein, ohne hierbei spezielle Hardware, Stereo-Kameras, Kalibrierungen über Marker oder gegebene 3D-Modelldaten der Szene vorauszusetzen.

Als Teilmechanismus der Rekonstruktion von Tiefeninformationen aus Bildern sowie der Verdeckung durch *Image-Based-Rendering*-Verfahren müssen Fotos (bzw. deren Bildinhalte) segmentiert und/oder Konturen der abgebildeten Objekte detektiert werden. In Absatz *3.6* (*Segmentierung und Konturdetektion in Bildern*) wurden verschiedene Segmentierungs- und Konturdetektionsansätze analysiert. Einige Verfahren beschränken sich auf spezielle Bildinhalte (z. B. Bilder mit Landschafts- und Naturmotiven). Diese Beschränkungen sind mit der vorliegenden Arbeit nicht vereinbar. Weitere Arbeiten ermöglichen die Segmentierung/Konturdetektion beliebiger Bilder und setzen Fuzzylogik-Systeme oder künstliche neuronale Netze ein [GS06], [LBLM08] [MFM04] [MAFM08] [CSS+10]. Solche Ansätze setzen unter anderem die Notwendigkeit von Trainingsdaten (z.B. Bilder, die von Personen händisch „gelabelt“ und skizziert wurden) voraus. Da das Erstellen von Trainingsdaten im Rahmen dieser Arbeit sich sehr schwierig gestaltet, soll in der vorliegenden Arbeit kein Fuzzylogik-System und kein künstliches neuronales Netzwerk aufgebaut werden. Folglich soll in dieser Arbeit ein eigenes Segmentierungsverfahren zum Einsatz kommen, welches sich nicht auf spezielle Bildinhalte beschränkt und keine Trainingsdaten voraussetzt.

Somit lässt sich die vorliegende Arbeit zum einen in den Bereich der Augmented Reality und zum anderen in die drei großen Forschungsbereiche Image-Based-Rendering, Bildverarbeitung und 3D-Rekonstruktion einordnen.

4.3 Erweiterte Anforderungen

In diesem Unterkapitel werden neben den Kernanforderungen (siehe Absatz *1.3* (*Kernanforderungen an die Arbeit*)) weitere Anforderungen zusammengetragen, die für das Erreichen des Hauptziels dieser Arbeit notwendig sind (oder als Unterstützung angesehen werden).

Für eine authentische Darstellung einer Bilderwelt ist eine möglichst hohe

Flächenabdeckung durch Fotos (bzw. Image-Planes) innerhalb der 3D-Welt vorteilhaft. So genannte *Schwarze Bereiche* sollen in der Bilderwelt möglichst wenig vorhanden sein (siehe Absatz *2.6.2* (*Anforderungen für das Generieren einer Bilderwelt*)). Des Weiteren wurde in Absatz *2.6.2* erklärt, dass das Eingabematerial (Fotos einer Szene mit überlappenden Bildanteilen) für das Erstellen einer Bilderwelt von essentieller Bedeutung ist. Hierbei spielen Anzahl der Fotos, Abdeckung der Szene durch die Bildinhalte der Fotos sowie die Qualität der Fotoaufnahmen eine Rolle. Je größer die Anzahl der Fotos und je besser die Abdeckung der Szene in den Fotos ist, desto besser kann ein Structure-from-Motion-Algorithmus arbeiten. Im Ergebnis bedeutet dies eine höhere Flächenabdeckung in der Bilderwelt und je mehr Fotos bzw. deren Kameras rekonstruiert werden (und je heterogener die *Bildstruktur* der Fotos ist), desto mehr *Key-Points* entstehen. Das bedeutet, je mehr Kameras mit ähnlichen Bildinhalten in einer Bilderwelt enthalten sind, desto dichter ist die Punktwolke an jener Stelle der Bilderweltenszene. Analog zu der Mindestzahl von Fotos kann auch eine Mindestzahl von Key-Points (und somit die Dichte der Punktwolke) für diese Arbeit nicht genau definiert werden. Dies ist abhängig von der Größe der Bilderweltenszene, der Bildstruktur der Fotos sowie der Anforderung an die Genauigkeit der Ergebnisse. Je heterogener die Bildstruktur eines Fotos ist, desto mehr 3D-Informationen in Form von Key-Points werden für den Teilschritt der *Teilrekonstruktion* benötigt. Mit zunehmender Anzahl von Key-Points nimmt auch die Genauigkeit der Teilrekonstruktion zu. Für eine genaue Gewinnung von Tiefeninformationen der abgebildeten Szene in einem Foto müssen Key-Points für alle räumlich relevanten Objekte der Szene (insbesondere für Occluder) vorhanden sein. Beispielsweise müssen in einer Innenraumszene Key-Points an allen Wänden, am Fußboden und an der Decke zu finden sein. Dies ist eine grundlegende Voraussetzung für die Teilrekonstruktion. Im Zusammenhang mit der Bildstruktur der Fotos und der Teilrekonstruktion werden zusätzlich Anforderungen an *Abgebildete Objekte* gestellt. Die *Objektstruktur* beinhaltet Farbe und Beschaffenheit des abzubildenden Objektes (oder Teilobjektes) sowie dessen Beleuchtungs- und Reflexionssituation innerhalb der realen (abzubildenden) Szene. Diese Struktur sollte über das gesamte Objekt (oder Teilobjekt) gleich (oder ähnlich) bleiben und dementsprechend in der Abbildung gleichbleibend dargestellt werden. Dies ist unterstützend für

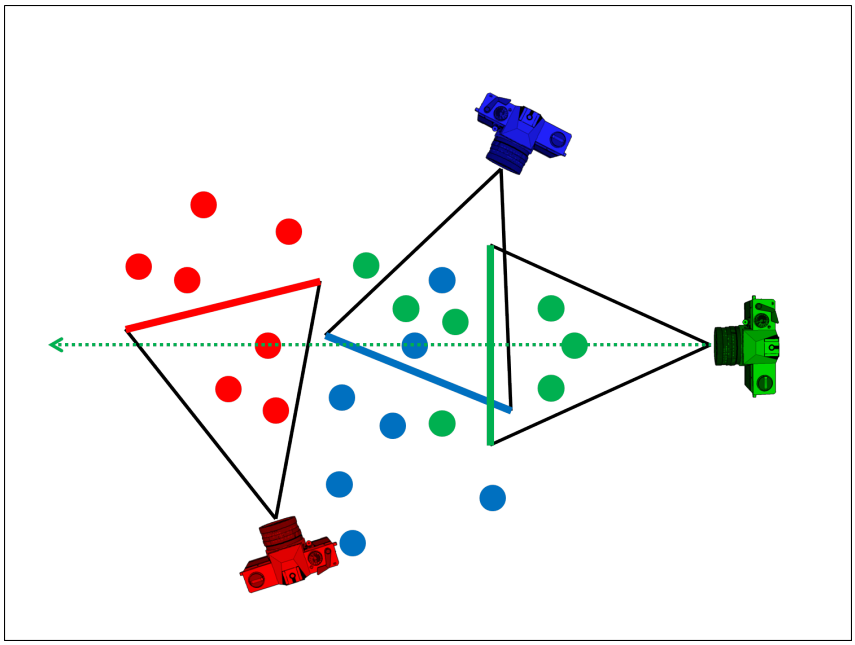


Abbildung 4.1: Schematische Darstellung von drei rekonstruierten Kameras und ihren dazugehörigen Observed-Points. Kamera, dazugehörige Image-Plane und Observed-Points sind in gleicher Farbe dargestellt.

Segmentierungsmechanismen innerhalb der Teilrekonstruktion.

Da Tiefeninformationen aus Sicht einer Kamera mithilfe von Key-Points gewonnen werden (*Kameratiefe*), soll als Eingabe nicht die globale, kameraübergreifende Punktwolke, sondern die *Observed-Points* der Kamera genutzt werden. Die Verwendung der Observed-Points anstelle der vollständigen Punktwolke als Eingabematerial für die Teilrekonstruktion verringert zwar die Anzahl an 3D-Informationen, erhöht aber die Korrektheit und Genauigkeit. Die schematische *Abb. 4.1* zeigt am Beispiel von drei rekonstruierten Kameras diesen Sachverhalt. Der *Hauptsehstrahl* der grünen Kamera „sieht“ einen Key-Point (grün eingefärbt). Dieser Key-Point stellt dementsprechend ein Observed-Point der grünen Kamera dar. Die Punktwolke ist die Gesamtheit aller roten, grünen und blauen Key-Points. Würde die Information fehlen, welcher Key-Point von welcher Kamera zu sehen ist, würden drei Key-Points (jene Key-Points, die vom Hauptsehstrahl durchstoßen werden) für den Pixel des Fotos der grünen Kamera an der Stelle des Durchstoßpunktes des Hauptsehstrahls in Frage kommen. Dies verfälscht das Ergebnis der Gewinnung von Tiefeninformationen. Die einzig korrekte Tiefeninformation liefert nur der grüne Key-Point, der

Observed-Point. Die Skizze von *Abb. 4.1* stellt die Observed-Points als eine Eins-zu-Eins-Relation zwischen Kameras und Key-Points dar. Tatsächlich kann aber ein Key-Point von mehreren Kameras gesehen werden. Für das Augmentieren der Bilderwelten werden 3D-Objekte (und ihre Geometrie-Daten) benötigt. Diese werden in dieser Arbeit als gegeben gesehen und somit vorausgesetzt.

Neben Anforderungen, die die Bilderweltenszene direkt betreffen, werden auch Anforderungen an das Eingangsmaterial, die Fotos gestellt. Es sollen beliebige digitale Farbbilder im RGB-Format (24- oder 32-bpp) verwendet werden, damit jede beliebige Consumer-Kamera verwendet werden kann. Es sollen keine Kalibrierungen der Kameras oder Spezialhardware vorausgesetzt werden. Die Bilder müssen in den Metadaten die Brennweite (in mm) der Kamera bereitstellen, da diese für die Erstellung von Bilderwelten benötigt wird. Da die Brennweite nur aussagefähig ist, wenn die auch die Sensorgröße der Kamera bekannt ist, muss diese ebenfalls vorliegen. Weiterführend sollen potentielle *Occluder* (möglichst) genau, scharf (z. B. bei Fotos mit geringer Tiefenschärfe) und flächendeckend fotografiert und in der Bilderweltenszene abgebildet sein. Dies ist Voraussetzung für die Generierung von Key-Points im Bereich der Occluder innerhalb der Bilderweltenszene. Key-Points, die Occluder repräsentieren, sind wiederum Voraussetzung für die *Teilrekonstruktion der Bilderweltenszene* als Teilschritt dieser Arbeit. Die Dichte der Punktwolke kann nicht genau festgelegt werden. Folglich kann auch eine genaue Mindestanzahl von Fotos potentieller Occluder nicht festgelegt werden.

Neben der Quantität spielt auch die Qualität der Fotos eine Rolle. Je größer die Auflösung eines Fotos, desto detaillierter (hochaufgelöster) der Bildinhalt. Dies ist vorteilhaft für die Generierung einer Bilderwelt sowie für die Mechanismen und Verfahren dieser Arbeit. Dennoch muss berücksichtigt werden, dass mit zunehmender Bildauflösung die Verarbeitung langsamer und speicherintensiver absolviert wird. Zusätzlich ist die Schärfe eines Bildes von großer Bedeutung. Je schärfer ein Foto, desto deutlicher und detailreicher sind Konturen der abgebildeten Objekte zu erkennen. Die Erkennbarkeit von Konturen ist vorteilhaft für einige Mechanismen und Verfahren dieser Arbeit.

Der Kern dieser Arbeit bezieht sich auf den Augmentierungsschwerpunkt *Verdeckung*. Eine authentische Augmentierung erfordert neben der *Verdeckung* vier weitere Augmentierungsschwerpunkte. Jeder Schwerpunkt besitzt für die

Realisierbarkeit eigene Anforderungen. Diese Anforderungen werden in dieser Arbeit nicht näher erörtert und berücksichtigt.

4.4 Verfeinerte und technisch detaillierte Zielsetzung

Mithilfe der Erkenntnisse der ersten drei Kapitel sowie den erweiterten Anforderungen wird in diesem Abschnitt die Zielstellung aus Absatz *1.2* (*Zielsetzung der Arbeit*) verfeinert. Dabei werden die bereits erläuterten Teilschritte dieser Arbeit (siehe Absatz *1.4*) mit technischen Details erweitert.

4.4.1 Verfeinerte Zielsetzung der Datenaufbereitung

Das Ziel dieses Schrittes ist die Bereitstellung der *Rohdaten* sowie die Erhöhung der Qualität der Roh- und Bilddaten (bzw. der Bildinhalte). Aus einer Serie von Fotos sollen über einen SfM-Algorithmus (siehe Absatz *2.6* (*Bilderwelten*)) die Rohdaten für eine Bilderwelt bereitgestellt werden. Diese sollen extrinsische und intrinsische Kameraparameter sowie Daten einer 3D-Punktwolke beinhalten. Die extrinsischen Kameraparameter sollen die dreidimensionalen Transformationsparameter einer rekonstruierten Kamera *Rotation*, *Translation* und *Skalierung* als Vektoren oder Matrizen beinhalten. Die intrinsischen Kameraparameter sollen die Brennweite bereitstellen. Zusätzlich soll die Information zur Verfügung stehen, welche Kamera „sieht“ welche Key-Points. Dies ergibt für jede Kamera eine Liste von *Observed-Points* sowie die dazugehörigen *Observed-Pixel* (korrespondierender Bildpunkt an der Stelle des Kamerabildes, an der sich der Key-Point in dem Foto befindet).

Die Daten der 3D-Punktwolke sollen für jeden Key-Point die 3D-Position und eine Liste der *Observer* (Kameras, die diesen Key-Point „sehen“) beinhalten. Zusätzlich soll ein gemittelter Farbwert aller Farbwerte der korrespondierenden Observed-Pixel bereitgestellt werden.

Neben der Bereitstellung der Rohdaten müssen Mechanismen zur Erhöhung der Datenqualität der Kameraparameter und der Punktwolke erfolgen, da fehlerhafte oder ungenaue Daten zu verzerrten Ergebnissen der weiteren Teilschritte führen.

Da SfM-Algorithmen mit dem in Absatz *2.6.3* (*Bündelblockausgleichung*) beschriebenen *Bundle-Adjustment*-Verfahren arbeiten und dies ein Fitting-Mechanismus (Annäherungsverfahren) darstellt, können in den Daten ungenaue oder verzerrte Kameras und Key-Points vorkommen. Die Kameras sollen mithilfe ihrer Observed-Points, dem Bildinhalt ihrer Fotos und der räumlichen Struktur der Bilderweltenszene durch Vergleichsmechanismen mit anderen Kameradaten bezüglich ihrer Korrektheit und Zuverlässigkeit bewertet werden.

Analog dazu sollen die Key-Points mithilfe ihrer Observer, ihrer Position und Farbe sowie der räumlichen Struktur der Bilderweltenszene durch Vergleichsmechanismen mit anderen Key-Points bezüglich ihrer Korrektheit und Zuverlässigkeit bewertet werden.

Mithilfe der aufbereiteten Rohdaten kann ein Maßstab für eine korrekte geometrische Skalierung und Positionierung einzubettender virtueller Objekte im Rahmen der *Transformationsanpassung* erfolgen. Die Daten sollen in diesem Teilschritt dafür vorbereitet werden. Die tatsächliche Transformationsanpassung erfolgt in Teilschritt *Authentische Darstellung der virtuellen Objekte*.

Des Weiteren können in qualitativ unterschiedlichen Bildern Bildfehler, Auflösungsdifferenzen, Über-/Unterbelichtungen und Verwacklungen/Unschärfe zu visuellen Störungen innerhalb der Bilderwelt führen. Daher müssen die Bilddaten (bzw. die Bildinhalte der Fotos) so aufbereitet werden, dass die Fotos als eine homogene Szene in der Bilderwelt dargestellt werden können. Hierfür sollen geeignete Bildanalyse- und Bildverarbeitungsverfahren bereitgestellt werden.

4.4.2 Verfeinerte Zielsetzung der Teilrekonstruktion der Bilderweltenszene

Das Ziel dieses Teilschrittes ist das Gewinnen der Tiefeninformationen der Bilderweltenszene mithilfe der aufbereiteten Bild- und Rohdaten. Dies erfolgt über eine Teilrekonstruktion anhand der 3D-Informationen der gegebenen Key-Points. Da die aus Key-Points bestehende Punktwolke lediglich ein unvollständiges 3D-Punktmodell der Bilderweltenszene darstellt, müssen Mechanismen für die Erweiterung und Zerlegung der Punktwolke in Teilmengen entwickelt werden, um eine sinnvolle Teilrekonstruktion zu ermöglichen.

Bereits in Absatz *1.4.2* (*Teilrekonstruktion der Bilderweltenszene*) wurde die Zielstellung der Teilrekonstruktion je nach Teilziel unterschieden in

1. Klassifizierung abgebildeter (realer) Objekte in Vordergrund- und Hintergrundobjekte (in Bezug zu eingebetteten virtuellen Objekten) und

2. (annähernd) vollständiges und pixelgenaues Gewinnen der *Kameratiefe* der Fotos.

Der Begriff *pixelgenau* bedeutet, dass für jedes Pixel individuell ein zu ihm zugehöriger Wert ermittelt wird. Die Rekonstruktion der Kameratiefen stellt das Gewinnen von Tiefeninformationen aus dem abgebildeten Szenenauschnitt der Kamera bzw. des Fotos dar. Hierfür soll für jede Kamera (jedes Foto) eine *Depth-Map* erstellt werden. Für die Speicherung der Tiefeninformationen sollen Gleitkommazahlen pro Tiefenwert verwendet werden. Die entstehende einkanälige Gleitkommazahl-Textur wird als *Float-Depth-Map* bezeichnet. Das Verwenden von Gleitkommazahlen ist im Vergleich zu Ganzzahlen speicherintensiver und folglich auch rechenintensiver. Des Weiteren gibt es Systeme, die nicht mit Gleitkommazahl-Texturen arbeiten können. Für diese Umstände sollen die bereits vorgestellten *Rgb-Divided-Depth-Maps* verwendet werden, bei denen der Tiefenwert auf die RGB-Farbkanäle als Ganzzahlen zerlegt wird. Die *Observed-Points* sollen als Eingabematerial für die Rekonstruktion der Kameratiefe verwendet werden. Ausgehend von den Observed-Points bzw. den korrespondierenden *Observed-Pixeln* sollen mit geeigneten Bildanalyse- und Bildsegmentierungsverfahren die Fotos in *Depth-Patches* zerlegt werden. Ein Depth-Patch stellt ein Bildsegment dar, dem eine Tiefe zugeordnet wurde und dessen Pixel als Füllwerte die Tiefeninformation beinhalten. Die Gesamtheit aller Depth-Patches ergibt die Depth-Map. Da aus einen Observed-Pixel eine Fläche (Segment) entsteht, ist die Depth-Map, bestehend aus Depth-Patches nicht pixel-, sondern nur flächengenau. Dies ist für eine Klassifizierung in Vorder- und Hintergrundobjekte und somit zur Erreichung des Teilziels *1* (*Authentische Verdeckung bei Kameranavigation*) ausreichend. Dies ist darin begründet, dass bei der Kameranavigation der Bildinhalt immer orthogonal betrachtet wird. Folglich reicht eine flächengenaue Unterscheidung von Vorder- und Hintergundobjekten aus. Eine pixelgenaue, vollständige Tiefenrekonstruktion des gesamten Bildausschnitts ist nicht notwendig.

Teilziel *2* (*Authentische Verdeckung bei freier Navigation*) erfordert die Darstellung einer augmentierten Bilderwelt ohne visuelle Störungen und mit authentischen Verdeckungen aus beliebigen Positionen. Hierfür sollen annähernd pixelgenaue Depth-Maps der Bilder erstellt werden, damit durch tiefenbasierte Image-Based-Rendering-Mechanismen die authentische Verdeckung und Darstellung aus beliebigen Blickwinkeln und Positionen erreicht werden. Hierfür muss die vollständige Kameratiefe der Fotos (also die vollständige Rekonstruktion der Tiefe des Bildinhaltes) annähernd pixelgenau gewonnen werden, damit im folgenden Teilschritt *Authentische Darstellung der Bilderwelten* das Bild als *3D-Bilddarstellung* gerendert werden kann und somit auch bei freier Navigation die Darstellung einer augmentierten Bilderwelt ohne visuelle Störungen und mit authentischen Verdeckungen ermöglicht werden kann.

Mithilfe einer Geometrie-Generierung aus den Observed-Points einer Kamera soll eine 3D-Geometrie erzeugt werden, die die Szene aus der Sicht der Kamera ansatzweise als 3D-Objekt repräsentiert. Dieses 3D-Objekt soll ausschließlich für die pixelgenaue Gewinnung der Tiefeninformationen genutzt werden. Anzumerken ist, dass das 3D-Objekt keine vollständige 3D-Rekonstruktion der Bilderweltenszene darstellt, sondern eine Teilrekonstruktion des im Kamerabild abgebildeten Szenenausschnittes repräsentiert. Da es sich nicht um eine vollständige 3D-Rekonstruktion handelt und die Geometrie ausschließlich für das Gewinnen der (annähernd) pixelgenauen und vollständigen Kameratiefe und nicht für das Rendering verwendet wird, werden die Kernanforderungen dieser Arbeit (siehe Absatz *1.3*) nicht verletzt.

4.4.3 Verfeinerte Zielsetzung der Authentischen Darstellung der Bilderwelten

Das Ziel dieses Teilschrittes ist es, unter Zuhilfenahme der Depth-Maps als Ergebnis von Teilschritt *Teilrekonstruktion der Bilderweltenszene*, die Darstellung der Bilderwelten in Hinblick auf das Erreichen einer authentischen Verdeckung eingebetteter virtueller Objekte anzupassen.

Bevor die Darstellung hinsichtlich des Hauptziels adaptiert werden kann, muss eine Standard-Darstellung von Bilderwelten entwickelt werden. Hierfür muss eine geeignete Positionierung der *Image-Planes* orthogonal zur rekon-

struierten Kamera im 3D-Raum bestimmt werden. Die Distanz zwischen einer Kamera und seiner korrespondierenden Image-Plane in der Bilderweltenszene wird im weiteren Text als *Image-Distance* bezeichnet. Zusätzlich müssen die Image-Planes in Abhängigkeit der Image-Distance korrekt skaliert werden, damit im Zusammenspiel aller Image-Planes eine bildbasierte 3D-Welt mit annähernd nahtlosen Übergängen zwischen den Fotos im 3D-Raum entsteht. *Abb. 2.15* verdeutlicht den Aufbau einer Bilderwelt mit annähernd nahtlosen Übergängen zwischen den Fotos. Diese grundlegende bildbasierte Darstellung einer Bilderwelt wird im weiteren Text als *Default-Rendering* bezeichnet. Eine weiterführende Adaption der Darstellung für eine Augmentierung und für die Erreichung des Hauptziels dieser Arbeit erfolgt beim Default-Rendering nicht.

Durch die Anforderung virtuelle Objekte zu integrieren, reicht das Default-Rendering für eine authentische Verdeckung/Darstellung der augmentierten Bilderwelten nicht aus. Zur Erreichung des ersten Teilziels müssen die *Kamerabilder* mithilfe der Depth-Maps in einer Form dargestellt werden, dass entsprechende Bildanteile des Kamerabildes als *Occluder* für die virtuellen Objekte dienen. Hierfür soll ein Kamerabild in so viele Bildanteile (Segmente) zerlegt werden, wie *Depth-Patches* in der Depth-Map enthalten sind. Dementsprechend soll jedem Depth-Patch ein passendes Bildsegment zugeordnet werden. Diese Bildsegmente sollen anschließend an die kodierte Tiefe der Depth-Patches verschoben werden. Demnach ist die Darstellung nicht bild-, sondern segment- bzw. flächenbasiert. Für die Realisierung des zweiten Teilziels reicht eine Zerlegung der Kamerabilder in Vordergrund- (Occluder) und Hintergrundobjekte nicht aus, da die Betrachtung des Bildinhaltes nicht stets orthogonal erfolgt, sondern aus beliebigen Positionen betrachtet werden kann. Eine authentische Verdeckung kann nur durch eine dreidimensionale Darstellung (*3D-Bilddarstellung*) des abgebildeten Szenenausschnittes realisiert werden.

4.4.4 Verfeinerte Zielsetzung der Authentischen Darstellung der virtuellen Objekte

Die vorangegangenen drei Teilschritte dienen ausschließlich dem Ziel einer authentischen Verdeckung integrierter virtueller Objekte in einer Bilderweltenszene. Das Ziel dieses Teilschrittes ist es, integrierte virtuelle Objekte fotorealistisch

darzustellen. Dies beinhaltet Mechanismen der Augmentierungsschwerpunkte *Transformationsanpassung*, *Render-Adaption* und *Bildbasierte Beleuchtung*. Bei der Transformationsanpassung müssen integrierte virtuelle Objekte an den Maßstab der Bilderwelt adaptiert werden. Dementsprechend muss Skalierung, Position und Ausrichtung der virtuellen Objekte bestimmt und an die räumliche Struktur der Bilderweltenszene angepasst werden. Die Render-Adaption soll visuelle Störungen (z. B. hervorgerufen durch Bildfehler, geringe Auflösung, Über-/Unterbelichtungen und Verwacklungen/Unschärfe der Fotos) zwischen den Kamerabildern und den hochaufgelösten virtuellen Objekten korrigieren oder weitestgehend verringern. Zusätzlich soll eine bildbasierte Beleuchtung und ein authentischer Schatten das Integrieren der virtuellen Objekte an die Bilderweltenszene anpassen.

Diese Schwerpunkte betreffen nicht das Hauptziel und somit nicht den Kern dieser Arbeit. Folglich sollen konzeptionelle Ansätze zwar vorgestellt, aber nicht in der Tiefe konzipiert werden.

4.5 Zusammenfassung und Diskussion

An dieser Stelle soll noch einmal auf Absatz *1.8* (*Zusammenfassung und Begriffsübersicht*) und die dort dargestellte *Abb. 1.12* verwiesen werden. Dort findet sich eine Übersicht eingeführter Begriffe (*Augmentierungsschwerpunkte*, *Hauptziel* und *Teilziele*, *Teilschritte*) und ihre Zusammenhänge untereinander, die für dieses Kapitel voraussetzend sind.

In diesem Kapitel wurde die vorliegende Arbeit in die Wissenschaft eingeordnet (Absatz *4.2*). Hierfür wurden Abgrenzungen zu den aktuellen Stand der Technik und Wissenschaft gezogen. Weiterführend wurden erweiterte Anforderungen an diese Arbeit zusammengetragen. Die in Absatz *1.2* beschriebene Zielsetzung wurde in Bezug auf die definierten Teilschritte (Absatz *1.4*) verfeinert und erweitert. Hierzu flossen die aus Kapitel *2* und *3* gewonnenen Erkenntnisse sowie technische Details ein.

Im folgenden Kapitel wird die Konzeption der vorliegenden Arbeit auf Basis der verfeinerten Zielsetzung zusammengetragen.

Kapitel 5

Konzeption

5.1 Überblick

In Kapitel 1 bis 4 wurde das notwendige Fundament für die Konzeption dieser Arbeit gebildet. In diesem Kapitel wird das Konzept für das Erreichen einer authentischen Verdeckung in augmentierten Bilderwelten beschrieben. Somit stellt das Kapitel den Kern dieser Arbeit dar. Für das Erreichen der gesetzten Ziele unter Berücksichtigung der fünf *Augmentierungsschwerpunkte* wird die Arbeit in vier Teilschritte (*1. Datenaufbereitung*, *2. Teilrekonstruktion der Bilderweltenszene*, *3. Authentische Darstellung der Bilderwelten*, *4. Authentische Darstellung der virtuellen Objekte*) zerlegt. Teilschritt 2 und 3 bilden den Schwerpunkt des Konzeptes, da sie Verfahren und Mechanismen für eine authentische Verdeckung augmentierter Bilderwelten bereitstellen (siehe *Abb. 5.1*). Hierbei werden unterschiedliche Verfahren und Mechanismen entwickelt und zusammengetragen, die entweder für eine *Kameranavigation* (Teilziel 1) oder für eine *Freie Navigation* (Teilziel 2) konzipiert sind. Die Trennung in Teilziele bzw. Navigationsarten erfolgt ausschließlich in diesen beiden Teilschritten.

Methoden und Techniken für eine authentische Integration virtueller Anteile in reale Szenen ist ein breites Forschungsgebiet der AR. Je nach AR-Umgebung existieren unterschiedliche Anforderungen und Ansprüche. Die hier vorliegende Arbeit basiert auf den Einsatz von Bilderwelten als AR-Umgebung. Folglich sind die Methoden und Verfahren dieser Arbeit sowie deren Ergebnisse auf den Einsatz in Bilderwelten konzipiert. Dies soll jedoch nicht die Verwendbarkeit in anderen AR-Systemen ausschließen.

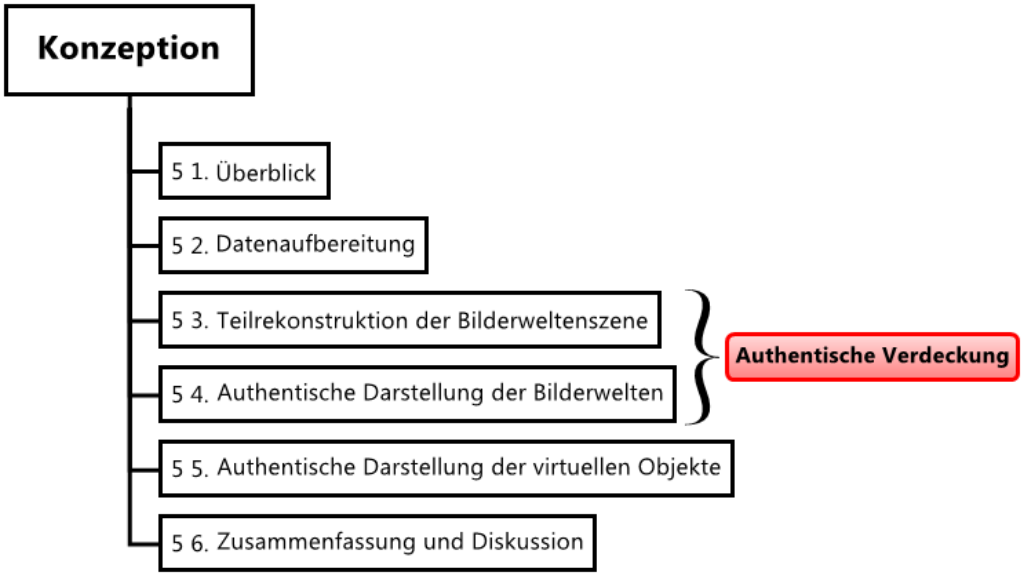


Abbildung 5.1: Aufbau des Konzeptes dieser Arbeit. Unterkapitel *5.3* und *5.4* bilden die Schwerpunkte dieser Arbeit.

Aufbau des Konzeptes

Dieses Kapitel unterteilt sich in die erläuterten Teilschritte als Unterkapitel *5.2* bis *5.5*. Folglich wird das Konzept jedes einzelnen Teilschrittes erläutert. Dieses Kapitel sowie seine Unterkapitel beinhalten zu Beginn einen Absatz, der einen *Überblick* über das (Unter-)Kapitel gibt und enden mit einem Absatz *Zusammenfassung und Diskussion*. In *Abb. 5.1* sind die Absätze der Unterkapitel (der Teilschritte) dargestellt und zeigen somit die Gliederung dieses Kapitels.

Wie mehrfach erklärt, kann eine authentische Augmentierung nur unter Berücksichtigung aller Augmentierungsschwerpunkte erfolgen. Dementsprechend decken Teilschritt 1 und 4 (*Datenaufbereitung* und *Authentische Darstellung der virtuellen Objekte*) die Realisierung der anderen Augmentierungsschwerpunkte (neben der *Verdeckung*) ab. Da sie nicht das Hauptziel und somit nicht den Kern dieser Arbeit betreffen, werden konzeptionelle Ansätze zwar vorgestellt, aber nicht in der Tiefe konzipiert. Teilschritt 2 und 3 (*Teilrekonstruktion der Bilderweltenszene* und *Authentische Darstellung der Bilderwelten*) werden als Schwerpunkte für das Erreichen der Ziele dieser Arbeit hervorgehoben. Sie beinhalten die Realisierung einer authentischen Verdeckung in augmentierten Bilderwelten.

5.2 Datenaufbereitung

5.2.1 Überblick

Das Ziel des ersten Teilschrittes ist es, die Rohdaten für den Aufbau und die Darstellung einer Bilderwelt zur Verfügung zu stellen. Weiterführend wird die Qualität der Rohdaten verbessert, indem ihre Genauigkeit und Zuverlässigkeit bestimmt bzw. bewertet wird. Ebenfalls werden die Fotos, die als Eingabematerial für die Erstellung der Rohdaten dienen, vorverarbeitet, sodass trotz qualitativ unterschiedlicher Fotos die Bilderwelt als eine homogene Szene dargestellt werden kann.

5.2.2 Bereitstellen der Rohdaten

Die Berechnung der Rohdaten wird für diese Arbeit vorausgesetzt. Dafür existieren so genannte *Structure-from-Motion*-Algorithmen und Programme. Die *SfM*-Algorithmen berechnen aus Fotos die extrinsischen und intrinsischen Kameraparameter sowie die Daten einer 3D-Punktwolke (bestehend aus Key-Points). Dabei werden Beziehungen zwischen Key-Points und Kameras hergestellt. Key-Points, die zu einer bestimmten Kamera zugehörig sind, werden als *Observed-Points* dieser Kamera bezeichnet. Kameras, die einen bestimmten Key-Point als Bildpunkt in ihrem Foto enthalten, werden als *Observer* des Key-Points bezeichnet. Der korrespondierende Bildpunkt wird *Observed-Pixel* genannt.

Ein *SfM*-Algorithmus liefert eine 3D-Punktwolke und die Zuordnung zwischen Key-Points und Observern sowie zwischen Observed-Points und Kameras (bzw. Kamerabild). Jedoch kann nicht davon ausgegangen werden, dass ein SfM-Algorithmus die Zuordnungen direkt als Rohdaten mitliefert. Dies ist abhängig von seiner Umsetzung. In diesem Fall müssen die Observed-Points und Observed-Pixel für jede Kamera berechnet werden. Hierfür werden sie auf die Bildebene (Image-Plane) der entsprechenden Kamera projiziert, wie in *Abb. 5.2* skizziert ist. An der Bildposition, bei dem sich der Durchstoßpunkt des Projektionsstrahls befindet, befindet sich der Observed-Pixel des Key-Points. Der Key-Point kann somit als Observed-Point der Kamera gesehen werden. Die nachfolgende Berechnung für die Projektion der Key-Points erfolgt auf der

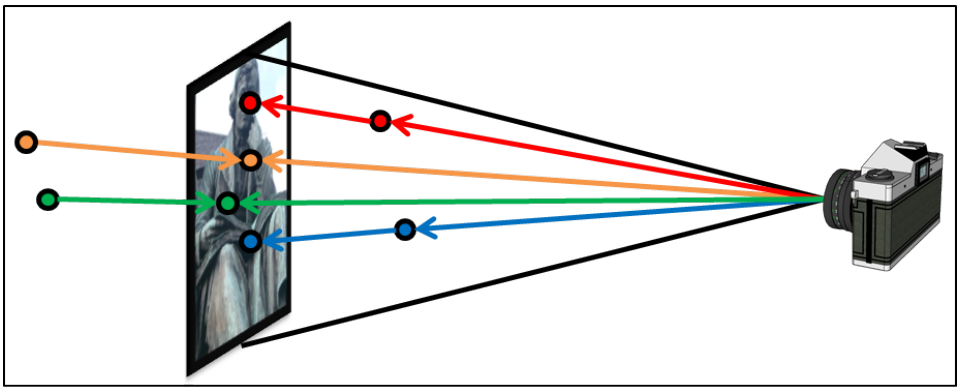


Abbildung 5.2: Rückprojektion von Key-Points auf die Bildebene (Image-Plane) für die Generierung von Observed-Points und Observed-Pixel einer Kamera.

Basis von [Sna10b]. Für jedes Bild ist die Brennweite (f), die Rotationsmatrix (R) und der Translationsvektor (t) gegeben. Die Brennweite wird zuvor in Pixeleinheiten umgerechnet (*Formel 5.1*), insofern dies die Rohdaten nicht mitliefern.

$$f = w * \frac{fmm}{ccd} \tag{5.1}$$

Formel 5.1: Umrechnung der Brennweite in Pixeleinheiten [Sna10b].

Die Bildbreite (w) in Pixeln, die Brennweite in Millimetern (fmm), sowie die Breite des CCD-Sensors (ccd) in Millimetern der verwendeten Kamera, werden aus den Meta-Daten des Bildes ausgelesen. *Formel 5.2* bis *Formel 5.4* dienen zur Projektion eines Key-Points auf ein Bild.

$$P = R * X + t \tag{5.2}$$

$$p = \frac{-P}{P.z} \tag{5.3}$$

$$p' = f * p \tag{5.4}$$

Formel 5.4: Projektion der Key-Points auf die Bildebene [Sna10b].

Formel 5.2 konvertiert die Weltkoordinaten des Key-Points in Kamerakoor-

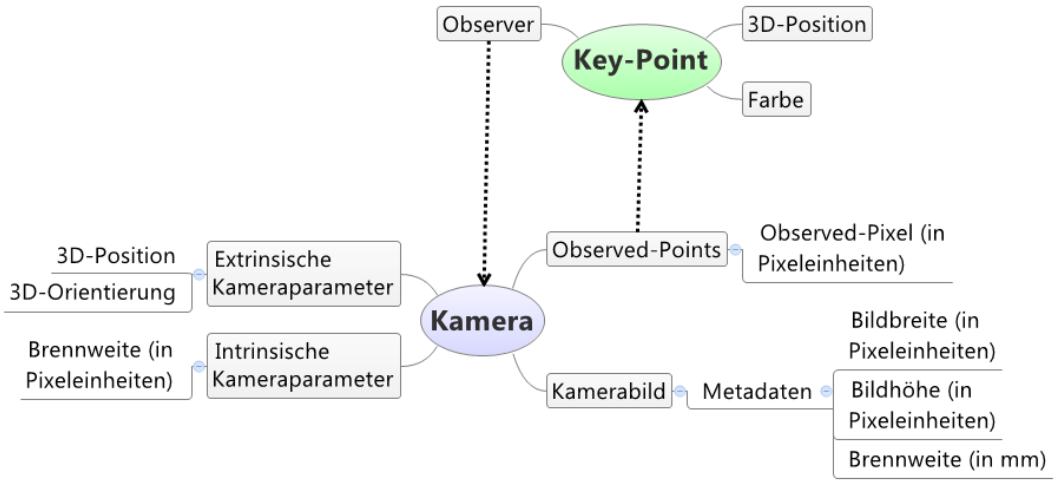


Abbildung 5.3: Die gegebenen Rohdaten als Ergebnis eines *SfM*-Algorithmus. Die Pfeilverbindungen verdeutlichen die Beziehungen der Rohdaten untereinander. Ein Key-Point besitzt *Observer*, also die Information von Kameras, dass sie den Key-Point als Bildpunkt in ihrem Foto enthalten. Kameras besitzen *Observed-Points*, also Key-Points, die zu der Kamera zugehörig sind.

dinaten. *Formel 5.3* repräsentiert die perspektivische Division und *Formel 5.4* überführt p zu einen 2D-Punkt in Pixeleinheiten.

Abb. 5.3 zeigt schematisch die gegebenen Rohdaten und ihre Beziehungen untereinander.

5.2.3 Aufbereiten der Kameradaten

In Absatz *2.6.3* (*Generieren einer Bilderwelt*) wurde erklärt, dass Sructure-from-Motion-Algorithmen auf dem Prinzip der *Bündelblockausgleichung* beruhen. Dies stellt ein Näherungsverfahren dar, weshalb es zu Ungenauigkeiten in den Ergebnissen kommen kann. Deshalb müssen die Kameraparameter auf ihre Genauigkeit geprüft und bewertet werden. Das prozentuale Ergebnis der Bewertung wird im weiteren Text (relative) *Kameragenauigkeit* genannt.

Jede Kamera besitzt ein oder mehrere Observed-Points. Je mehr Observed-Points berechnet wurden, desto korrekter kann die Kamera interpretiert werden. Die Bündelblockausgleichung stellt bildübergreifend Beziehungen zwischen den *Feature-Points* (extrahierte, markante Bildmerkmale) her und berechnet so die Kameraparameter. Neben den Kameraparametern entstehen dreidimensionale

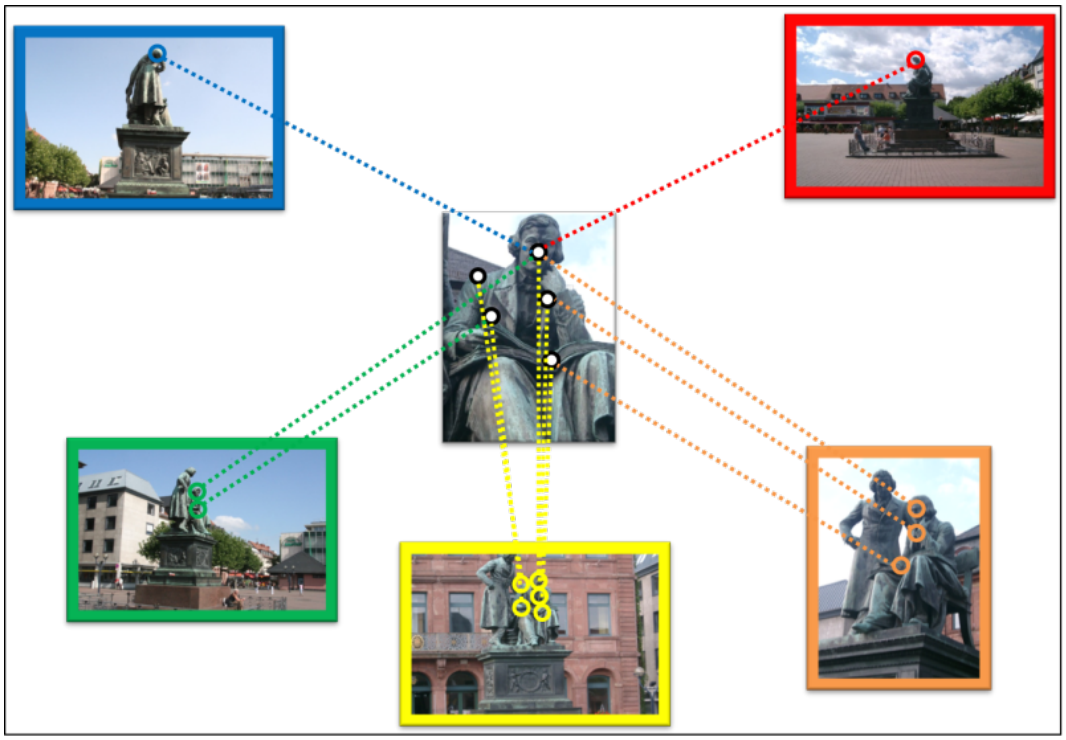


Abbildung 5.4: Erweiterung der *Abb. 2.20* um fünf Key-Points (weiße Kreise) und die korrespondierenden Feature-Points (Observed-Pixel) der Kamerabilder (entsprechend der Kamerafarben in *Abb. 2.20* eingefärbte Ringe). Die Korrespondenz zwischen Key-Point und Observed-Pixel ist als (entsprechend der Kamerafarben) eingefärbter Sehstrahl eingezeichnet.

Verknüpfungspunkte verschiedener Feature-Points der Kamerabilder, die Key-Points. Da es sich um eine Ausgleichung, also eine Annäherung handelt, sind Fehler einkalkuliert. Daher wird das Fehlerrisiko um so geringer, je mehr korrespondierende Feature-Points der Kamerabilder ein Key-Point besitzt. Im Umkehrschluss reduziert sich das Fehlerrisiko der Kameraparameter, je mehr korrespondierende Key-Points (also Observed-Points) eine Kamera besitzt.

In *Abb. 5.4* wird die schematische Darstellung der Bilderwelt von *Abb. 2.20* um fünf Key-Points (weiße Kreise) und die korrespondierenden Feature-Points (Observed-Pixel) der Kamerabilder (entsprechend der Kamerafarben in *Abb. 2.20* eingefärbte Ringe) erweitert. Die Korrespondenz zwischen Key-Point und Observed-Pixel ist als (entsprechend der Kamerafarben) eingefärbter Sehstrahl eingezeichnet. Durch den SfM-Algorithmus wurden den Kameras eine unterschiedliche Anzahl von Observed-Points zugeordnet. Anzumerken ist, dass in

Kamera	Observed-Pixel	Kameragenauigkeit
rot	1	20 %
orange	3	60 %
gelb	5	100 %
grün	2	40 %
blau	1	20 %

Tabelle 5.1: (Relative) Kameragenauigkeiten der schematischen Bilderwelt in *Abb. 5.4*.

einer Bilderwelt die höchste Zahl an Observed-Points einer Kamera immer noch wesentlich kleiner ist als die Gesamtzahl der Key-Points. Dies ist wieder auf die Bündelblockausgleichung zurückzuführen. Ein Verknüpfungspunkt zwischen verschiedenen Bildern (aus dem die Key-Points entstehen), kann offensichtlich nicht von allen Bildern/Kameras einer Szene „gesehen“ werden. Folglich wird nicht die Gesamtzahl der Key-Points als Maß für die höchstmögliche (relative) Kameragenauigkeit verwendet, sondern die höchste Zahl an Observed-Points (aus allen Kameras). Das bedeutet, die höchste Zahl an Observed-Points einer Kamera innerhalb der Bilderwelt definiert die höchste (relative) Kameragenauigkeit. Dementsprechend erreicht die gelbe Kamera eine (relative) Kameragenauigkeit von 100%, da sie fünf Observed-Points besitzen und dies die höchste Zahl an Observed-Points in der Bilderwelt entspricht. In *Tab. 5.1* sind entsprechend der Anzahl der Observed-Points die (relativen) Kameragenauigkeiten aufgelistet.

Mithilfe der (relativen) Kameragenauigkeiten können Strategien entwickelt werden, um „schlechte“ Kameras innerhalb einer Bilderwelt anders oder gar nicht darzustellen (siehe Absatz *5.2.6* (*Aufbau der Bilderweltenszene*)).

5.2.4 Aufbereiten der Daten der Punktwolke

Analog zu Absatz *5.2.3* müssen aufgrund des einkalkulierten Fehlerrisikos bei der Generierung einer Bilderwelt die Key-Points auf Genauigkeit und Zuverlässigkeit untersucht werden. Wie dort bereits erwähnt, wird das Fehlerrisiko um so geringer, je mehr korrespondierende Feature-Points der Kamerabilder ein Key-Point besitzt. Das bedeutet, je öfter der Key-Point als Observed-Point einer Kamera

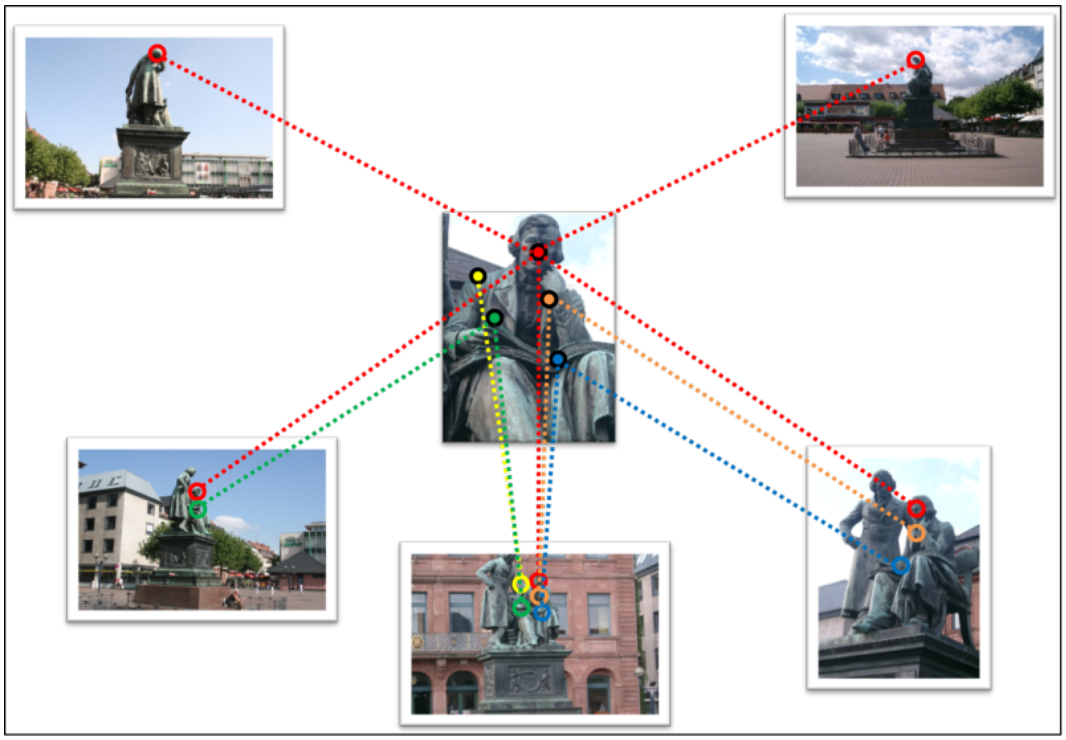


Abbildung 5.5: Modifikation von *Abb. 5.4*. Die Key-Points und nicht die Kamerabilder sind eingefärbt. Die korrespondierenden Feature-Points (Observed-Pixel) der Observer werden als (entsprechend der Key-Point-Farben eingefärbte) Ringe dargestellt. Die Korrespondenz zwischen Key-Point und Observed-Pixel des Observers ist als eingefärbter Sehstrahl eingezeichnet.

fungiert, desto genauer und korrekter kann er interpretiert werden. Folglich bestimmt die Anzahl der Observer (Kameras, die den Key-Point als Observed-Point zugeordnet haben) eines Key-Points die Genauigkeit (im weiteren Text als relative *Punktgenauigkeit* bezeichnet). In *Abb. 5.5* sind die Key-Points und nicht die Kamerabilder eingefärbt. Die korrespondierenden Feature-Points (Observed-Pixel) der Observer werden als (entsprechend der Key-Point-Farben eingefärbte) Ringe dargestellt. Die Korrespondenz zwischen Key-Point und Observed-Pixel des Observers ist als eingefärbter Sehstrahl eingezeichnet. Der rot eingefärbte Key-Point besitzt die meisten Observer. Dementsprechend wird durch ihm die Obergrenze der (relativen) Punktgenauigkeit definiert. Seine fünf Observer entsprechen 100 % (relativer) Punktgenauigkeit.

Daraus lässt sich *Tab. 5.2* aufstellen, die die (relative) Punktgenauigkeit der Key-Points aufzeigt. Die (relative) Punktgenauigkeit wird innerhalb des

Key-Point	Observer	Punktgenauigkeit
rot	5	100 %
orange	2	40 %
gelb	1	20 %
grün	2	40 %
blau	2	40 %

Tabelle 5.2: (Relative) Punktgenauigkeiten der schematischen Bilderwelt in *Abb. 5.5.*

Teilschrittes *Teilrekonstruktion der Bilderweltenszene* für die Tiefenzuordnung benötigt (siehe Absatz *5.3.3* (*Tiefenzuordnung*)).

5.2.5 Aufbereiten des Bildmaterials

Qualitativ unterschiedliche Bilder führen zu visuellen Störungen innerhalb der Bilderwelt (siehe auch *Abb. 1.6(a)*). Deshalb muss das Bildmaterial so aufbereitet werden, dass die Fotos als eine homogene Szene in der Bilderwelt dargestellt werden.

In dem vorliegenden Konzept wird sich diese Problematik auf unnatürliche Farbunterschiede (Farbstich) und Helligkeitsunterschiede (Über-/Unterbelichtungen) in den Fotos beschränken, da diese die wichtigsten Bereiche bei visuellen Störungen einer Bilderwelt darstellen. Dies ist auf die menschliche Sehwahrnehmung in Bezug auf Helligkeit und Farbe zurückzuführen [Scz11]. Anhand der Observed-Points und ihren korrespondierenden Observed-Pixel werden Farb- und Helligkeitsabweichungen der Bilder untereinander ermittelt und angepasst.

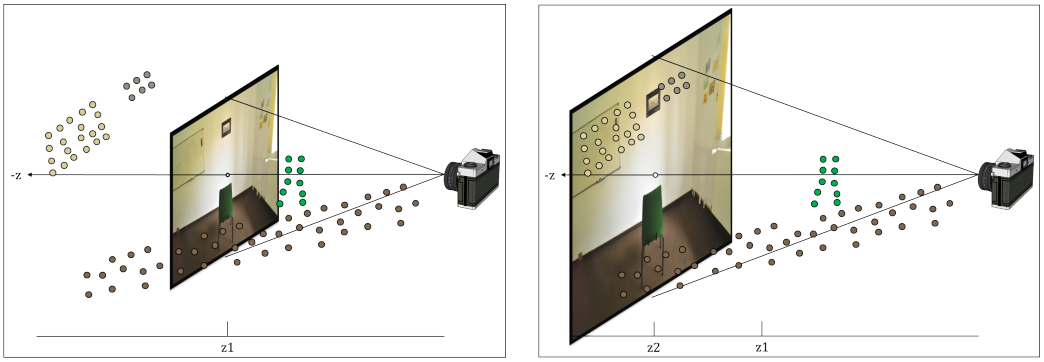
Key-Points und ihre korrespondierende Observed-Pixel besitzen Farbwerte. Der Key-Point-Farbwert stellt den gemittelten Wert seiner Observed-Pixel-Farbwerte dar. Der Observed-Pixel-Farbwert repräsentiert den tatsächlichen Farbwert des Pixels in dem Foto. In [BGKN09] wurde unter Mitarbeit des Autors ein Verfahren vorgestellt, welches an dieser Stelle zum Einsatz kommt. Mithilfe der Key-Points und korrespondierenden Observed-Pixel wird für jedes Foto ein Vergleichshistogramm erstellt. Dieses gibt eine Aussage darüber, wie sich Helligkeit und Farbe (ein Histogramm für jeden Farbkanal) gegenüber den Key-Point-Farbwerten ändern. Durch die Mittelung der Histogramme aller

Fotos entsteht ein globales Vergleichshistogramm für die gesamte Bilderwelt. Anhand des globalen Vergleichshistogramms werden die einzelnen Bilder (und ihren Vergleichshistogrammen) angepasst. Als Ergebnis wird eine Bilderwelt mit dem angepassten Bildmaterial als homogene Szene dargestellt (siehe *Abb. 1.6(b)*).

5.2.6 Aufbau der Bilderweltenszene

Die berechnete (relative) *Kameragenauigkeit* wird verwendet, um den Aufbau der Bilderwelten zu optimieren. Mithilfe der (relativen) Kameragenauigkeiten können Strategien entwickelt werden, um „schlechte"Kameras innerhalb einer Bilderwelt nicht darzustellen. Über einen Schwellwert wird definiert, welche Kameras aus der Bilderwelt entfernt werden. Je nachdem wie hoch der Schwellwert T_{Cam} der prozentualen (relativen) Kameragenauigkeit gesetzt wird, werden die verwendeten Kameras akkurater, aber in der Anzahl weniger. Mit dem mittleren Wert von $T_{Cam} = 50\%$ können Ausreißer detektiert und verworfen werden und dennoch ein großer Teil an rekonstruierten Kameras innerhalb der Bilderwelt verwendet werden. Ein genauer Wert lässt sich an dieser Stelle nicht definieren, da dies vom Anwendungsfall und Größe der Bilderwelt (bzw. Anzahl an Eingabefotos) abhängt.

Neben der Bereitstellung und Aufbereitung der Rohdaten müssen auch Daten für den zielgerechten (für die Integration virtueller 3D-Objekte optimierten) Aufbau der Bilderweltenszene berechnet werden. Die Rohdaten liefern die rekonstruierten Kameraparameter, mit denen eine Bilderwelt aufgebaut werden kann. Projekte wie PhotoTourism [SSS06a] konstruieren den Aufbau der Bilderwelten in der Form, dass die Bilder unmittelbar vor die Kameras positioniert werden. Dies ist jedoch für eine Integration virtueller 3D-Objekte in die Szene sehr ungünstig, da kein oder nur wenig Platz zwischen den *Image-Planes* und ihren Kameras zur Verfügung steht. Folglich müssen die Image-Planes weiter entfernt von den Kameras angeordnet werden. Hierfür wird mithilfe der Observed-Points einer Kamera die *Image-Distance* (Distanz zur Kamera) berechnet. Die durchschnittliche Distanz der Observed-Points zur Kamera wird als Image-Distance genutzt, wie in *Abb. 5.6(a)* dargestellt. Das bewirkt zwar, dass Platz zwischen Image-Plane und Kamera zur Verfügung steht, jedoch befinden sich weiterhin etwa die Hälfte der Observed-Points hinter der Image-Plane. Das



(a) (b)

Abbildung 5.6: Positionierung der Image-Plane in Abhängigkeit von den Entfernungen der Observed-Points zur Kamera: (a) $z1$ ist der Mittelwert der Entfernungen aller Observed-Points. (b) $z2$ ist der Mittelwert der Entfernungen der 10% zur Kamera weit entferntesten Observed-Points.

bedeutet, die 3D-Szene wird weiterhin zum Teil verdeckt. Somit kann nicht die gesamte Szene für eine Augmentierung genutzt werden. Deshalb werden für die Berechnung nur die weit entferntesten Observed-Points genutzt. Um Ausreißer zu vermeiden, werden 10% der weit entferntesten Observed-Points ermittelt und mithilfe eines Ausreißer-robusten Mittelwertes (z. B. Median) die Image-Distance berechnet. Dies ist in *Abb. 5.6(b)* dargestellt. Hierdurch werden die Image-Planes so weit von den Kameras entfernt positioniert, dass sich (fast) alle Observed-Points zwischen ihrer Kamera und der dazugehörigen Image-Plane befinden. Folgerichtig ergibt sich somit ein großer Bereich, der für die Integration von virtuellen Objekten genutzt werden kann.

5.2.7 Zusammenfassung und Diskussion

In dem Teilschritt *Datenaufbereitung* wurden die gegebenen Rohdaten als Ergebnis eines SfM-Algorithmus bereitgestellt (siehe *Abb. 5.3*). Neben der Bereitstellung wurden die Kameras und die Key-Points auf ihre Genauigkeit untersucht und bewertet. Im Rahmen der Bewertung entstanden zwei neue Daten: Die (relative) *Kameragenauigkeit* und die (relative) *Punktgenauigkeit*. Des Weiteren wurde für einen zielgerechten Aufbau der Bilderweltenszene (in Hinblick auf die Integration virtueller 3D-Objekte) die *Image-Distance*

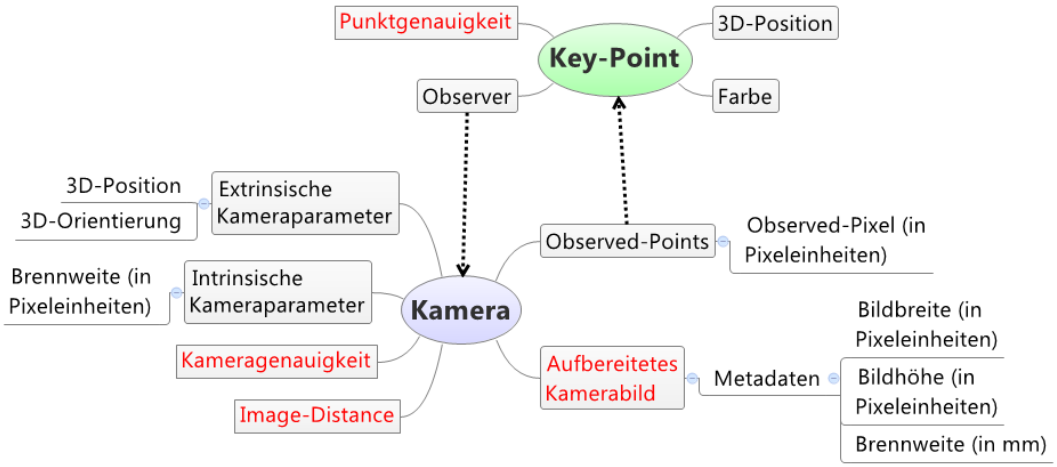


Abbildung 5.7: Ausgabedaten nach Teilschritt *Datenaufbereitung*. Neue Daten sind rot beschriftet.

berechnet. *Abb. 5.7* zeigt die Ausgabedaten nach der Datenaufbereitung.

5.3 Teilrekonstruktion der Bilderweltenszene

5.3.1 Überblick

Dieser Teilschritt realisiert zusammen mit dem nachfolgenden Teilschritt *Authentische Darstellung der Bilderwelten* den Augmentierungsschwerpunkt *Verdeckung* innerhalb augmentierter Bilderwelten und bildet somit den konzeptionellen Kern dieser Arbeit. Hierfür werden die Tiefeninformationen der Bilderweltenszene mithilfe der aufbereiteten Rohdaten gewonnen. Dies erfolgt über eine Teilrekonstruktion anhand der 3D-Informationen der gegebenen Key-Points. Als Ergebnisse der Teilrekonstruktion liegen *Depth-Maps* für jedes Foto der Bilderwelt vor.

Das vorliegende Konzept stellt verschiedene Verfahren für die Teilrekonstruktion vor, die je nach Teilziel eine Kombination aus den Mechanismen

- *Bildsegmentierung* und
- *Tiefenzuordnung*

darstellen. Entsprechend dieser Mechanismen ergibt sich der Aufbau dieses Unterkapitels (dieses Teilschrittes). Als Ergebnis liegen in Segmente und *Depth-*

Patches unterteilte Bilddaten vor. Somit ergeben sich zwei neue Daten: Das Segmentbild und die Depth-Map.

5.3.2 Bildsegmentierung

Überblick

Das Ziel der Bildsegmentierung ist es, jedes Foto der Bilderwelt in die dort abgebildeten realen (Teil-)Objekte zu unterteilen. Als Ergebnis entstehen Segmente, die in einem (neuen) Bild gespeichert und verwaltet werden. Dieses Bild wird im weiteren Text *Segmentbild* genannt. Die Segmente werden für den nachfolgenden Mechanismus *Tiefenzuordnung* benötigt. In Absatz *2.3* (*Bildverarbeitung und Bildanalyse*) und Absatz *3.6* (*Segmentierung und Konturdetektion in Bildern*) wurden bereits grundlegende und moderne Verfahren im Bereich der Segmentierung vorgestellt. Aufbauend auf den gewonnenen Kenntnissen und den Eigenschaften eines gewöhnlichen RGB-Farbbildes werden in diesem Absatz vier Ansätze der Bildsegmentierung beschrieben:

- *Farbbasierte Segmentierung*,
- *Konturbasierte Segmentierung*,
- *Farb- und Konturbasierte Segmentierung*,
- *Objektbasierte Segmentierung*.

Hierbei werden Vor- und Nachteile sowie Aufwand und Nutzen herausgearbeitet und gegenübergestellt.

Vor der eigentlichen Bildsegmentierung wird jedes Bild vorverarbeitet. Ein Blurring-Filter (siehe Absatz *2.3* (*Bildverarbeitung und Bildanalyse*)) wirkt potentiellem Bildrauschen vor der Segmentierung entgegen. *Abb. 5.8* zeigt ein Kamerabild sowie das per *Meanshift*-Blurring-Filter dazugehörige aufbereitete Bild. Die Vorverarbeitung wird für die nachfolgenden Segmentierungsverfahren vorausgesetzt und daher nicht mehr im Text erwähnt.

In den folgenden Abbildungen werden im Rahmen der Segmentierungsverfahren die Segmente mit Falschfarben neu eingefärbt. Dies dient ausschließlich der Ergebnisvisualisierung.

(a) Kamerabild

(b) Meanshift

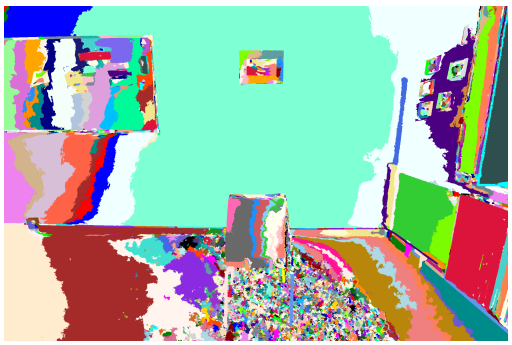
Abbildung 5.8: Kamerabild und das per Mean-Shift-Blurring-Filter aufbereitete Bild.

Farbbasierte Segmentierung

Bei der *Farbbasierten Segmentierung* werden zusammenhängende (benachbarte) ähnlich- oder gleichfarbige Pixel eines Fotos als ein Segment und somit als ein Objekt interpretiert. Für eine objektive und messbare Beurteilung von Unterschieden oder Ähnlichkeiten zwischen beliebigen Farben muss ein Farbraum eine (annähernde) Gleichabständigkeit aller auftretenden Farben aufweisen. Diese Eigenschaft besitzt der RGB-Farbraum, indem die Eingabedaten (die Fotos) vorliegen, nicht. Folglich müssen die RGB-Werte zum Vergleichen der Pixel in einen anderen Farbraum (mit der Eigenschaft der Farbgleichabständigkeit) transformiert werden. Zusätzlich kann für den Vergleichsmechanismus der Schwellwert für den *Farbabstand* ΔE verschoben werden. Nach [HN12] liegt der kleinste noch wahrnehmbare Farbunterschied in der Digitalfotografie bei $\Delta E = 2.5$. Je nach Bildinhalt kann es sinnvoll sein, benachbarte, ähnlich farbige Segmente zu einem Segment zusammenzufassen oder ein Segment in mehrere kleinere Segmente (mit kleinabständigeren unterschiedlichen Farben) zu zerlegen. Dies wird durch die Erhöhung bzw. Erniedrigung des Farbabstandes ΔE parametrisiert.

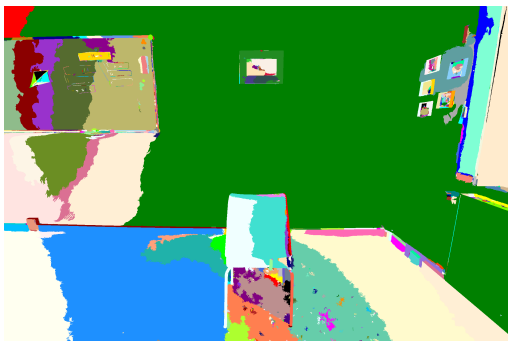
Abb. 5.9 zeigt das (farbbasiertes) Segmentbild von *Abb. 5.8* unter Verwendung des LUV- sowie des Lab-Farbraumes mit unterschiedlichen Schwellwerten.

Der Vorteil der farbbasierten Segmentierung liegt in der Einfachheit des Mechanismus. Benachbarte Pixel mit kleinen Farbabständen (also ähnliche Farben) werden als ein Segment gruppiert. Nachteilig gestalten sich Fotos mit unsatten Farben und Abbildungen mit mehreren benachbarten Objekten gleichen Farbtons. Objekte mit unsatten, ähnlichen Farben können „übersehen“ werden und fälschlicherweise mit anderen Segmenten verschmolzen werden. Benachbarte Objekte mit ähnlichen oder gleichen Farbton können fälschlicherweise als ein Segment detektiert werden. *Abb. 5.10* zeigt ein farbunsattes Bild. Der abgebildete vordergründige Raumbalken besitzt eine ähnliche Farbe wie die Wände im Hintergrund. Das dazugehörige Segmentbild detektiert den Balken nicht als ein eigenes Objekt, sondern sie verschmelzen mit der hinteren Wand und dem Fußboden. Dies kann auch nicht durch die Erniedrigung des Schwellwertes für den Farbabstand korrigiert werden, da die Farbe des Raumbalkens sehr ähnlich der Farbe der Wände im Hintergrund ist.

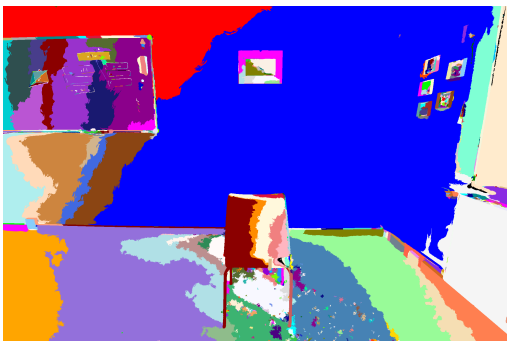


(a) LAB $\Delta E = 1.0$

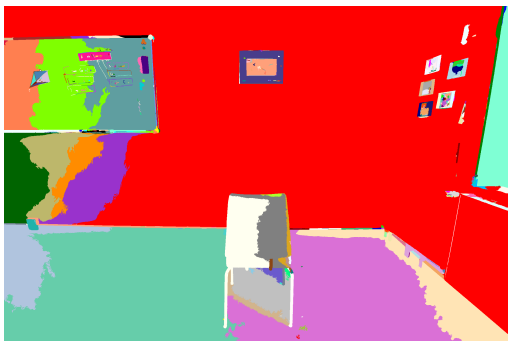
(b) LUV $\Delta E = 1.0$



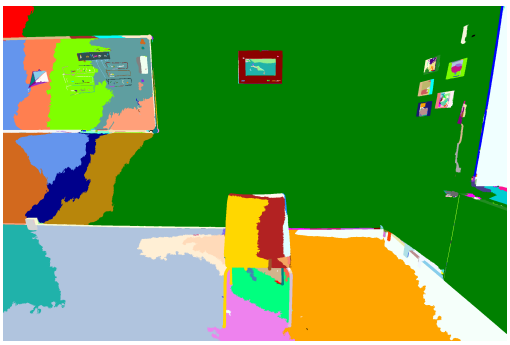
(c) LAB $\Delta E = 2.5$



(d) LUV $\Delta E = 2.5$

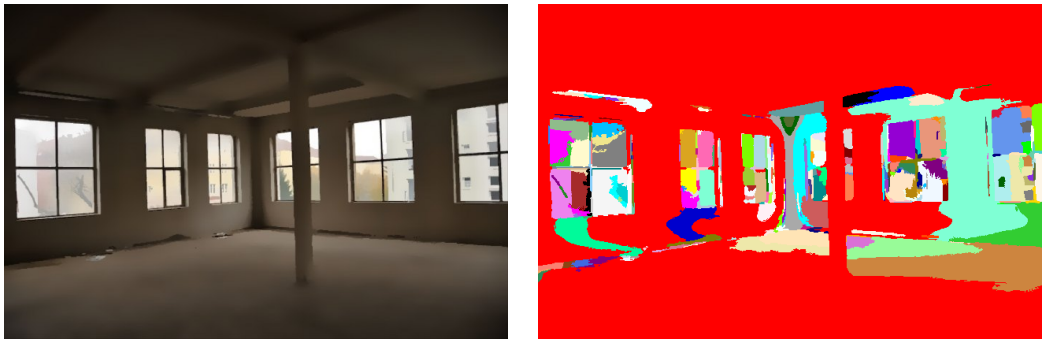


(e) LAB $\Delta E = 5.0$



(f) LUV $\Delta E = 5.0$

Abbildung 5.9: Farbsegmentierung (von *Abb. 5.8*) mithilfe des LAB- und LUV-Farbraumes unter Verwendung unterschiedlicher Schwellwerte für den Farbabstand.



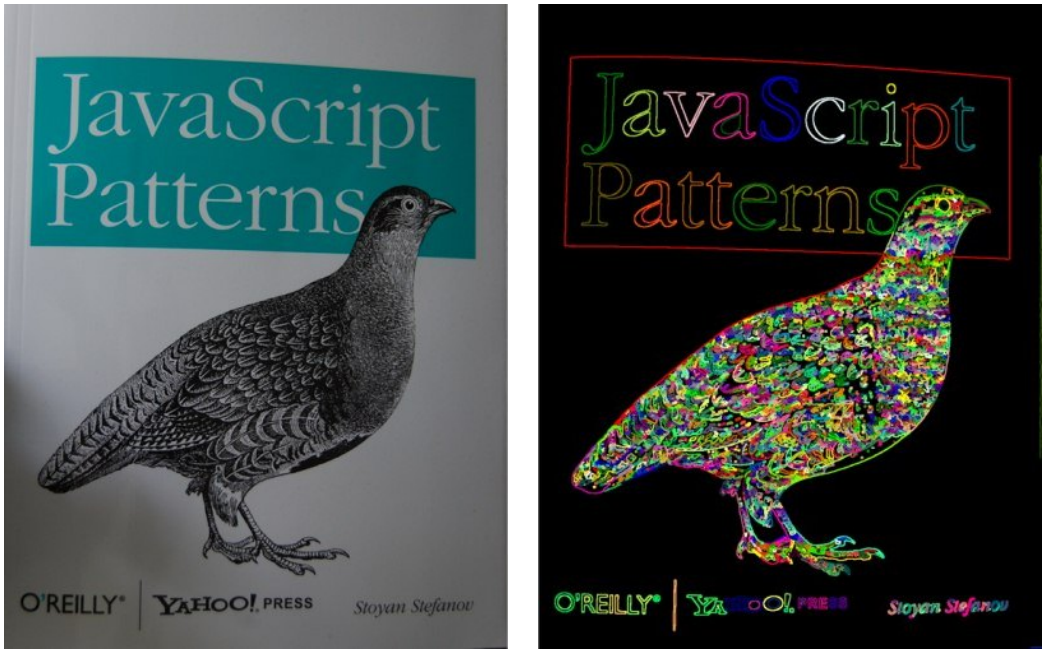
(a) Farbunsattes Kamerabild (b) LUV $\Delta E = 2.5$

Abbildung 5.10: (a) Ein farbunsattes Bild und (b) das dazugehörige fehlerhafte Segmentbild: Der abgebildete vordergründige Raumbalken besitzt eine ähnliche Farbe wie die Wände im Hintergrund. Das dazugehörige Segmentbild detektiert den Balken nicht als ein eigenes Objekt, sondern er verschmelzt mit der hinteren Wand und dem Fußboden.

Konturbasierte Segmentierung

Aufgrund der Anfälligkeit der farbbasierten Segmentierung gegenüber farbunsatten Fotos sowie dem Verschmelzen abgebildeter, benachbarter Objekte mit ähnlichem Farbton müssen weitere Maßnahmen zur besseren Abgrenzung von Segmenten getroffen werden. Eine Möglichkeit ist die Verwendung von Kanten und Konturen der abgebildeten Objekte. Bei der *Konturbasierten Segmentierung* wird die Farbeigenschaft der Objekte nicht beachtet, sondern mithilfe detektierter Kanten ein Segmentbild erzeugt. Für die Detektion von Kanten wird das Foto in ein Graustufenbild überführt. Ein Segment (ein Objekt) wird dementsprechend nicht durch Farbe, sondern durch zusammengehörige Kanten repräsentiert. Da konventionelle Kantendetektoren als Ergebnis zwar ein Kantenbild jedoch keine Rückschlüsse über die durch die Kantenpixel repräsentierten abgebildeten Objekte liefern, muss für eine Segmentierung aus den Kantenpixel Segmente (Objekte) detektiert werden.

In [NKG+11b] wurde vom Autor der vorliegenden Arbeit in Zusammenarbeit mit Co-Autoren *ConGrap* (*Contour Detection based on Gradient Map of Images*), ein konturbasiertes Segmentierungsverfahren auf der Basis einer Gradient-basierten Kantendetektion, veröffentlicht. Das Ziel von ConGrap ist es, anhand eines Kantenbildes die abgebildeten Objekte als eine geschlossene



(a) (b)

Abbildung 5.11: ConGrap – Konturbasiertes Segmentierungsverfahren auf der Basis einer Gradient-basierten Kantendetektion.

Kontur (Polygon) zu interpretieren. Im Gegensatz zu anderen Kantendetektionsverfahren bestimmt ConGrap neben einem Kantenbild die Zuordnungen der Pixel zu den Konturen. Da die Pixel der (geschlossenen) Konturen bekannt sind, kann das Konturbild auch als ein Segmentbild interpretiert werden. *Abb. 5.11* zeigt ein abgebildetes Buchcover sowie das dazugehörige Ergebnis von ConGrap.

Trotz unsatter Farben und Objekte (Buchstaben) mit gleichem Farbton werden die Objekte als geschlossene Konturen (Segmente) detektiert. ConGrap lässt sich in zwei Schritte gliedern:

a) Gradient-basierte Kantendetektion und

b) Konturverfolgung.

Die Schritte werden nachfolgend beschrieben.

a) Gradient-basierte Kantendetektion

Ein Kantendetektor ermittelt die Diskontinuitäten der zweidimensionalen Grauwertfunktion (in horizontaler und vertikaler Richtung) eines Graustufenbildes.

Die Ableitungen ($f'(x)$ und $f'(y)$) in beiden Richtungen der Grauwertfunktion bilden einen Gradienten. Dessen Betrag wird als Maß für das Identifizieren eines Kantenpixels genutzt [CM04a] [BB06d]. Es entsteht ein Bild, indem Pixel den Gradienten als Füllwert zugewiesen bekommen. Dementsprechend besitzen nur Pixel, die eine Kante repräsentieren, einen signifikant größeren Wert als null. Sie werden im weiteren Text als *Kantenpixel* und das entstandene Bild als *Kantenbild* bezeichnet.

Zusätzlich zu dem Kantenbild geben die Ableitungen Informationen über die Richtungsveräufe (Orientierungen) der Kanten. Mithilfe des Winkels θ in *Formel 5.5* wird die Orientierung berechnet [Gre11].

$$\theta = \begin{cases} 0° & \text{if f'(y)} = 0 \text{ and f'(x)} \neq 0 \\ 90° & \text{if f'(y)} = 0 \text{ and f'(x)} = 0 \\ \arctan\left(\frac{f'(x)}{f'(y)}\right) & \text{if f'(y)} \neq 0 \text{ and f'(x)} \neq 0 \end{cases} \tag{5.5}$$

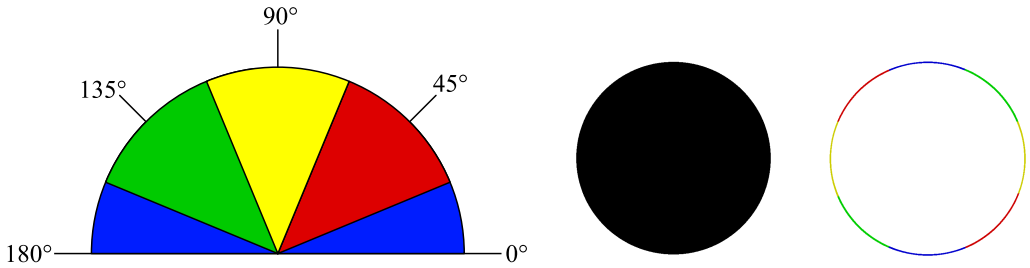
Formel 5.5: Berechnung der Orientierung von Kantenpixel.

Im diskreten bzw. grob quantisierten Raum können direkte Nachbarpixel vier Hauptrichtungen annehmen. Dementsprechend muss der Wertebereich für θ in vier Bereiche zerlegt werden, wie in *Abb. 5.12(a)* dargestellt wird. Die Orientierung θ kann folglich die vier Werte $0°, 45°, 90°, 135°$ annehmen und wird farb-kodiert für jedes Pixel des Kamerabildes gespeichert. Als Ergebnis entsteht ein neues Bild, dass im weiteren Text als *Gradient-Map* bezeichnet wird. *Abb. 5.12* zeigt die Gradient-Map eines abgebildeten Kreises.

b) Konturverfolgung

Das Ziel der Konturverfolgung ist das Detektieren von geschlossenen Konturen aus den gegebenen Kanteninformationen. Hierfür werden alle Pixel des erstellten Kantenbildes durchlaufen. Sollte ein Kantenpixel noch nicht zu einer Kontur zugeordnet sein, so entsteht mit diesem Pixel eine neue Kontur (im weiteren Text *Konturstartpixel* genannt). Auf diese Weise wird jedes Kantenpixel eindeutig einer Kontur zugeordnet. Basierend auf einem Konturstartpixel beginnt ein dreistufiger hierarischer Prozess, der die Nachbarpixel in einem definierten Radius r analysiert. Es wird untersucht, ob ein Nachbarpixel

1. ein Kantenpixel darstellt, falls nicht ...



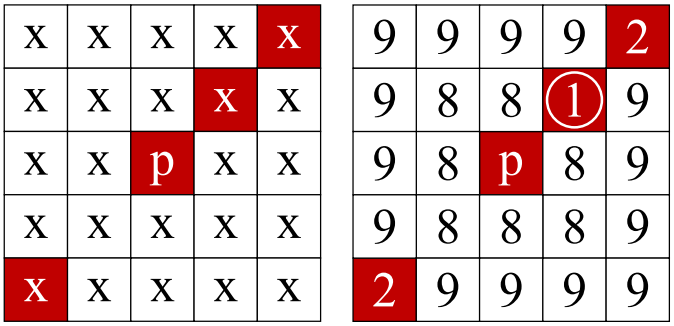
(a) (b) (c)

Abbildung 5.12: Erstellung einer Gradient-Map am Beispiel eines synthetischen Bildes: (a) Einteilen des Wertebereiches des Winkels θ (Kantenorientierung) in vier Bereiche [Gre11]. (b) Kamerabild mit abgebildeten Kreis. (c) Dazugehörige Gradient-Map.

2. ein Konturpixel der gleichen Kontur darstellt, falls nicht ...

3. ein Konturpixel einer anderen Kontur darstellt.

Sollte Schritt eins innerhalb des definierten Radius kein Kantenpixel finden, geht der Analyse-Prozess zu Schritt zwei über. Sollte ebenfalls innerhalb des definierten Radius kein Konturpixel der gleichen Kontur detektiert werden, geht der Analyse-Prozess zu Schritt drei über. Sollte dort ebenfalls kein Konturpixel einer anderen Kontur detektiert werden, wird das erste Pixel der Kontur mit dem letzten Pixel verbunden. Für jeden der drei Schritte wird der benachbarte Kanten- oder Konturpixel mit der kleinsten Distanz zu dem Konturstartpixel ermittelt. Je kleiner die Distanz, desto „passender“ soll ein Nachbarpixel für die Kontur interpretiert werden. Zusätzlich zur Distanz sollen Nachbarn mit gleicher Orientierung wie der Konturstartpixel bevorzugt werden. Dies soll mögliche Kantenpixel anderer abgebildeter Objekte bei der Konturverfolgung ausschließen. Dafür wird die Distanz von Pixeln mit ungleicher Orientierung mit einer vordefinierten Gewichtung *weight* erhöht. Um die Berechnung zu vereinfachen und im Ganzzahlbereich durchzuführen, wird (anstelle der euklidischen Distanz) der größere Wert der horizontalen und vertikalen Distanz verwendet. Folgender Pseudocode zeigt die Berechnung:



(a) Bildausschnitt (b) Distanz-Matrix

Abbildung 5.13: (a) Raster eines 5x5-Pixel-großen Bildausschnittes, in dem Kantenpixel dargestellt werden. Die rot eingefärbten Pixel besitzen die gleiche Orientierung. Der Pixel p fungiert als Konturstartpixel.(b) Dazugehörige Distanz-Matrix, erstellt mit $weight = 5$ (als Beispielwert). Der eingekreiste Pixel besitzt den kleinsten Distanzwert.

```
1 if (θ(Konturstartpixel) == θ(Nachbarpixel))
2   distance = Max(distanceX, distanceY)
3 else
4   distance = Max(distanceX, distanceY) + weight
```

Programmcode 5.1: Berechnung der Pixeldistanz (mit Gewichtung)

Somit lässt sich für den definierten Radius eine Distanz-Matrix für den Konturstartpixel aufstellen. *Abb. 5.13* zeigt das Raster eines 5x5-Pixel-großen Bildausschnittes, in dem Kantenpixel dargestellt werden. Die rot eingefärbten Pixel besitzen die gleiche Orientierung. Das Pixel p soll als Konturstartpixel fungieren.

Aus der resultierenden Distanz-Matrix lässt sich der direkte, rechte obere Nachbarpixel (eingekreist) als der Nachbar mit der kleinsten Distanz ermitteln. Dementsprechend wird dieses Pixel zu der Kontur des Konturstartpixels hinzugefügt. Des Weiteren werden die beiden Pixel miteinander „verbunden“. Die dabei entstehenden „Verbindungspixel“ werden ebenfalls zu der Kontur hinzugefügt.

Arbeitet der Analyse-Prozess in Schritt zwei oder drei und detektiert ein Konturpixel, so wird die Kontur an jenem Konturpixel geschlossen. Im Falle

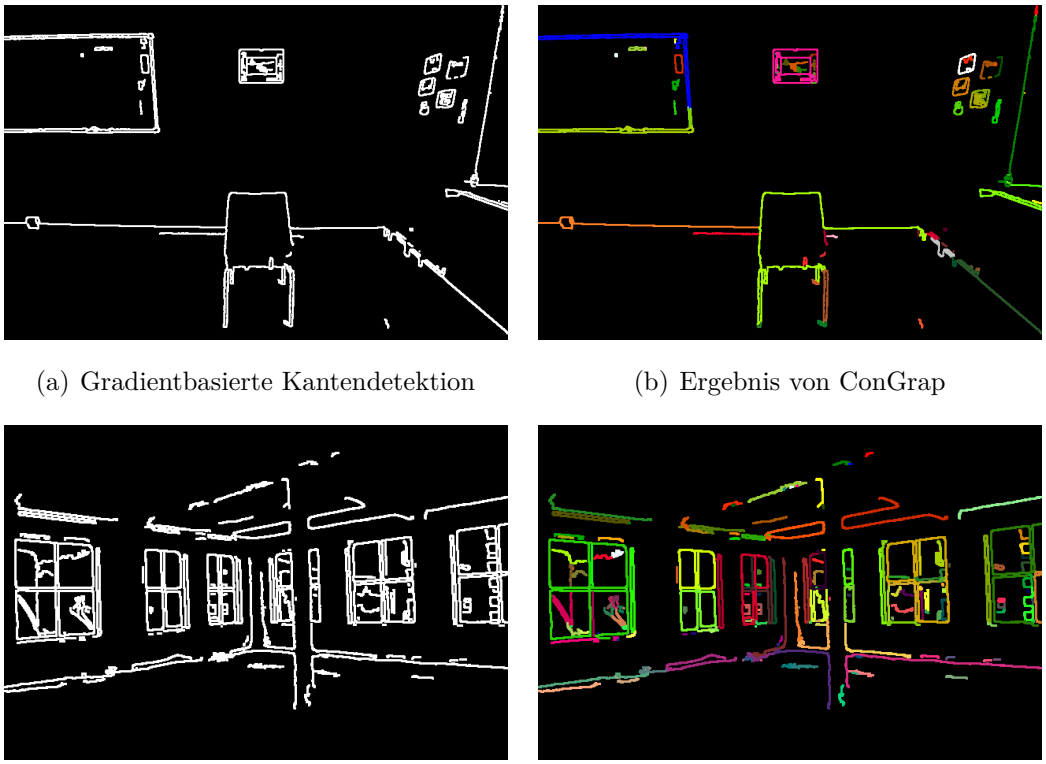
von Schritt eins und der Detektion eines benachbarten Kantenpixels, wird der Nachbar selbst zu einem Konturstartpixel und der Analyse-Prozess beginnt von vorne. Dieser rekursive Prozess erfolgt so lange, bis der Analyse-Prozess zu Schritt zwei oder drei übergeht und somit die Kontur schließt. Im Beispiel von *Abb. 5.13* wird der direkte, rechte obere Nachbarpixel (eingekreist) des Pixels p zum Konturstartpixel und der Analyse-Prozess startet erneut.

Als Ergebnis der Konturverfolgung liegen Konturen und ihre zugehörigen Pixel vor. Aus den Konturen wird ein Segmentbild erstellt. Jede Kontur repräsentiert ein Segment (siehe *Abb. 5.11(b)*). Der Vorteil der konturbasierten Segmentierung ist die Robustheit gegenüber unsatten Farben. Dies ist zurückzuführen auf die Gradient-basierte Kantendetektion, die lediglich die Intensitäten (Helligkeitswerte) des Fotos bzw. des dazugehörigen Graustufenbildes, nicht aber die Farbinformationen verarbeitet. Dementsprechend wirkt sich die Beschränkung auf die Helligkeitsverläufe eines Bildes nachteilig aus, wenn die Helligkeitsverläufe nicht sehr ausgeprägt sind. Diese Segmentierungsform setzt ein detailiertes Kantenbild voraus. Sie ist fehleranfällig im Falle von fehlenden Kanteninformationen (z.B. häufig unterbrochene Kanten) eines Objektes.

Abb. 5.14(a) zeigt ein Kantenbild (von *Abb. 5.8*) mit Unterbrechungen (siehe rechte Seite der abgebildeten Stuhllehne). Dies führt in *Abb. 5.14(b)* zu falsch gezogenen Konturen. Die Kontur der oberen Stuhllehne verbindet sich nicht mit dem unteren Teil, sondern verschmilzt mit der Bodenkante (grün gefärbte Kontur). Das zweite Beispiel zeigt in *Abb. 5.14(c)* das Kantenbild von *Abb. 5.10(a)*. Der im Vordergrund befindliche Balken wird nicht mit durchgängigen Kantenpixeln versehen. Folglich wird im Konturbild (*Abb. 5.14(d)*) der Balken nicht als ein geschlossenes Segment detektiert.

Farb- und Konturbasierte Segmentierung

Um dennoch Vorteile beider Verfahren auszunutzen und die Fehleranfälligkeiten der Einzelverfahren zu reduzieren, werden in diesem Schritt die *Farbbasierte Segmentierung* und die *Konturbasierte Segmentierung* kombiniert. Hierfür werden im ersten Schritt die Konturen in das Kamerabild eingezeichnet (anstelle eines eigenständigen Kantenbildes). Anschließend wird die farbbasierte Segmentierung verarbeitet, bei denen die Konturen als zusätzliche Begrenzung während des Segmentierungsvorganges dienen. Die eingezeichneten Kanten sorgen auch



(a) Gradientbasierte Kantendetektion

(b) Ergebnis von ConGrap

(c) Gradientbasierte Kantendetektion

(d) Ergebnis von ConGrap

Abbildung 5.14: Kantenbilder mit unterbrochenen Kanten abgebildeter Objekte führen im Konturbild zu fehlerhaft gezogenen Konturen.

(a) Farbunsattes Kamerabild (*Abb. 5.10(a)*)

(b) Farbsegmentierung (LUV $\Delta E = 2.5$)

(c) Kamerabild mit eingezeichneten Kanten

(d) Farb- und konturbasiertes Segmentbild

Abbildung 5.15: Vergleich der Farbsegmentierung mit der Kombination der farb- und konturbasierten Segmentierung anhand eines farbunsatten Kamerabildes.

bei farbunsatten Bildern sowie bei benachbarten abgebildeten Objekten mit gleichem Farbton für die Unterstützung einer korrekten Abgrenzung bei der Segmentierung.

Abb. 5.15 stellt die Farbsegmentierung als Einzelverfahren der Kombination aus farb- und konturbasierten Segmentierung gegenüber. Aufgrund der Kombination beider Segmentierungsverfahren werden die bereits erwähnten Fehleranfälligkeiten der einzelnen Verfahren reduziert. Die beiden Segmentbilder der Abbildung zeigen, dass das zusätzliche Verwenden (Einzeichnen) der Konturen bei der Farbsegmentierung eine Unterstützung bei der Abgrenzung einzelner abgebildeter Objekte (z. B. der im Vordergrund befindliche Balken) bewirkt. Somit ist diese kombinierte Segmentierungsform gegenüber den Einzelverfahren robuster gegenüber unsatten Farben und benachbarten abgebildeten Objekten mit gleichem Farbton, als auch gegenüber fehlenden Kanteninformationen (z.B. häufig unterbrochene Kanten) eines abgebildeten Objektes im

Kantenbild. Sollte ein Kamerabild nicht sehr ausgeprägte Helligkeitsverläufe und unsatte Farben abbilden, kann es auch hier zu fehlerhaften Segmenten kommen. Des Weiteren werden zwei Segmentierungsverfahren kombiniert bzw. nacheinander abgearbeitet. Dies führt zu einer erhöhten Rechenzeit gegenüber den Einzelverfahren.

Objektbasierte Segmentierung

Trotz robuster Eigenschaften der farb- und konturbasierten Segmentierung können bei Kombination nicht sehr ausgeprägter Helligkeitsverläufe und unsatten Farben im Kamerabild fehlerhafte Segmente entstehen. Des Weiteren weisen die bisher vorgestellten Segmentierungsverfahren keinerlei Zusammenhänge der abgebildeten räumlichen Struktur auf. Dies lässt sich anhand des Beispielbildes (siehe *Abb. 5.8*) erläutern. Die vorgestellten Segmentierungsverfahren zerlegen zwar das Bild anhand von Farben und/oder Konturen in Segmente, berücksichtigen aber keinerlei räumlichen Zusammenhang der abgebildeten Szene. An dieser Stelle wäre eine Segmentierungsform wünschenswert, die im Beispielbild jede Wand, den Fußboden sowie den Stuhl als ein Segment (eventuell mit mehreren Teilsegmenten) zerlegt. Dies ist die Zielstellung der *Objektbasierten Segmentierung*.

Mithilfe eines semiautomatischen Schrittes wird das Kamerabild in räumlich zusammenhängende Segmente zerlegt. Hierfür werden unterstützend Eckpunkte der Segmente durch den Nutzer eingezeichnet. Dieser semiautomatische Schritt wird im weiteren Text als *Segmentskizzierung* bezeichnet.

Abb. 5.16 zeigt das Beispielbild (aus *Abb. 5.8*) mit vier eingezeichneten Segmenten sowie das eigentliche Segmentbild mit eingezeichneten Eckpunkten der Segmente (rot eingefärbt, nur für Visualisierungszwecke eingezeichnet). Auf diese Weise wird das Bild in Segmente zerlegt, die der abgebildeten räumlichen Struktur entsprechen. Die Wände, der Fußboden und der Stuhl bilden jeweils ein Segment.

Der Vorteil dieses Segmentierungsverfahrens liegt in der (annähernden) Fehlerfreiheit. Fälschlich detektierte Segmente können (fast) nicht entstehen. Als Nachteil muss die semiautomatische Segmentskizzierung genannt werden. Sollte das Einzeichnen der Segmente nicht genau durchgeführt werden, ist auch das Segmentierungsergebnis ungenau.

(a) Kamerabild mit eingezeichneten Segmenten

(b) Objektbasiertes Segmentbild mit eingezeichneten Eckpunkten der Segmente

Abbildung 5.16: Objektbasierte Segmentierung mithilfe eines semiautomatischen Schrittes (Segmentskizzierung).

Zusammenfassung und Diskussion

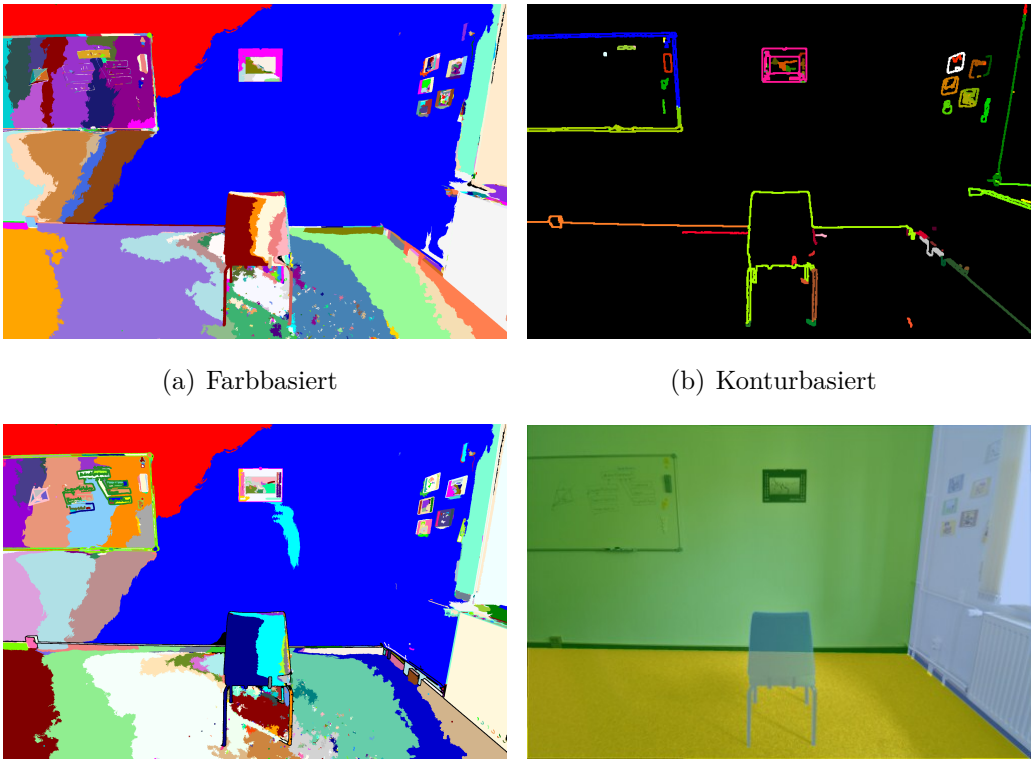
In diesem Absatz wurden verschiedene Verfahren für die Bildsegmentierung entwickelt. Hierbei wurden jeweils die Vor- und Nachteile der einzelnen Segmentierungsverfahren aufgezeigt. Diese werden in *Tab. 5.3* noch einmal dargestellt. Die Nachteile und Schwachstellen eines Verfahrens bildeten den Anstoß und die Motivation für das jeweils nachfolgende Verfahren. Somit verfolgte dieser Absatz das Bottom-Up-Prinzip. Als Ergebnis der Bildsegmentierung liegt ein Kamerabild zerlegt in Segmente vor, die für nachfolgende Mechanismen der *Teilrekonstruktion der Bilderweltenszene* als Repräsentanten von abgebildeten (Teil-)Objekten der Bilderweltenszene interpretiert werden. Die Segmentbilder aller vorgestellten Verfahren werden in *Abb. 5.17* noch einmal gegenübergestellt.

Eine Evaluierung der vorgestellten Segmentierungsverfahren findet sich in Absatz *7.2.1* (*Evaluierung der Bildsegmentierung*). *Abb. 5.18* zeigt die Ausgabedaten nach der Bildsegmentierung.

5.3.3 Tiefenzuordnung

Überblick

Das Ziel der Tiefenzuordnung ist es, den Bildern der Bilderwelt (annähernd) pixel- oder segmentgenau die *Kameratiefe* (Tiefeninformationen einer Bilderweltenszene, adaptiert auf das Sichtfrustum einer Kamera) zuzuordnen. Die



(a) Farbbasiert (b) Konturbasiert

(c) Farb- und konturbasiert (d) Objektbasiert

Abbildung 5.17: Segmentbilder aller vorgestellten Segmentierungsverfahren.

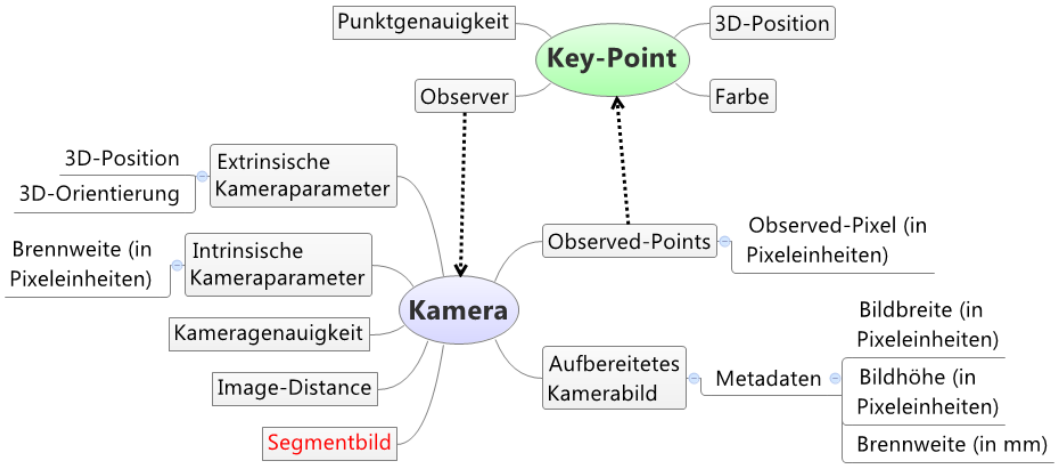


Abbildung 5.18: Ausgabedaten nach der Bildsegmentierung. Neue Daten sind rot beschriftet.

Segmentierung	Vorteile	Nachteile
Farbbasiert	- Einfache Umsetzung - Sehr gute Ergebnisse bei farbsatten Bildern - Keine Kantendetektion als Vorverarbeitung notwendig	- Fehleranfällig bei farbunsatten Bildern - Fehleranfällig bei benachbarten abgebildeten Objekten mit gleichem Farbton
Konturbasiert	- Robust gegenüber unsatten Farben - Robust gegenüber benachbarten abgebildeten Objekten mit gleichem Farbton	- Fehleranfällig bei nicht sehr ausgeprägten Helligkeitsverläufen, da detailiertes Kantenbild vorausgesetzt wird - Fehleranfällig bei fehlenden Kanteninformationen (z.B. häufig unterbrochene Kanten) eines Objektes
Farb- und Konturbasiert	- Robust gegenüber unsatten Farben - Robust gegenüber benachbarten abgebildeten Objekten mit gleichem Farbton - Robust gegenüber fehlenden Kanteninformationen (z.B. häufig unterbrochene Kanten) eines Objektes	- Fehleranfällig bei nicht sehr ausgeprägten Helligkeitsverläufen und unsatten Farben - Kantendetektion als Vorverarbeitung notwendig - Höhere Rechenzeit, da doppelte Segmentierung
Objektbasiert	- (Annähernd) Fehlerfreie Segmente - Zusammenhang zwischen Segmente und der abgebildeten räumlichen Struktur	- Semiautomatische Segmentskizzierung - Ungenaues Einzeichnen führt zu ungenauem Segmentierungsergebnis

Tabelle 5.3: Vor- und Nachteile der Segmentierungsverfahren.

Tiefeninformationen sind notwendig für das Realisieren einer authentischen Verdeckung und werden vom nachfolgenden Teilschritt *Authentische Darstellung der Bilderwelten* vorausgesetzt. Das Speichern der Tiefeninformationen erfolgt in *Depth-Maps*. Die Tiefenzuordnung erfolgt auf der Basis der *Observed-Points*. Diese wurden auf Genauigkeit und Zuverlässigkeit untersucht und bewertet. Die (relative) *Punktgenauigkeit* bestimmt an dieser Stelle, ob ein Observed-Point für die Tiefenzuordnung verwendet wird. Analog zu dem *Aufbau der Bilderweltenszene* (siehe Absatz *5.2.6*; Verwendung des Schwellwertes T_{Cam} für die Kameragenauigkeiten, um „schlechte“ Kameras herauszufiltern) wird ein Schwellwert für die (relative) Punktgenauigkeit (T_{Point}) definiert.

Mit dem mittleren Wert von $T_{Point} = 50\%$ können Ausreißer detektiert und verworfen werden und dennoch ein großer Teil an Observed-Points für die Tiefenzuordnung verwendet werden. Ein genauer Wert für T_{Point} lässt sich an dieser Stelle nicht definieren, da dies vom Anwendungsfall bzw. von der Anforderung an die Genauigkeit der Tiefenzuordnung, sowie von der Größe der Punktwolke abhängt.

In diesem Absatz werden drei Verfahren der Tiefenzuordnung entwickelt, die verschiedene Eingabedaten voraussetzen und unterschiedliche Ergebnisse produzieren:

- *Segment-Depth-Matching*,
- *Key-Point-Depth-Matching*,
- *Geometry-Depth-Matching*.

Das Segment-Depth-Matching-Verfahren generiert eine segmentgenaue Depth-Map, d. h. nicht jedes Pixel, sondern Flächen (Segmente) des Bildes bekommen eine Tiefe zugewiesen. Die Depth-Map kodiert also die Tiefe von senkrecht zur Blickrichtung stehenden, planaren Flächen. Als Eingabedaten werden die Key-Points und die Segmentbilder vorausgesetzt. Das bedeutet, eine fehlerhafte Segmentierung kann zu fehlerhaften Depth-Maps führen. Um eine Tiefenzuordnung auch ohne eine vorangehende Segmentierung zu ermöglichen, wurde das Key-Point-Depth-Matching-Verfahren entwickelt. Bei diesem Verfahren werden die Key-Points auf die Bildebene als kreisförmige *Sprites* zurückprojiziert. Die Distanz des Key-Points zur Kamera wird dabei als Füllwert für

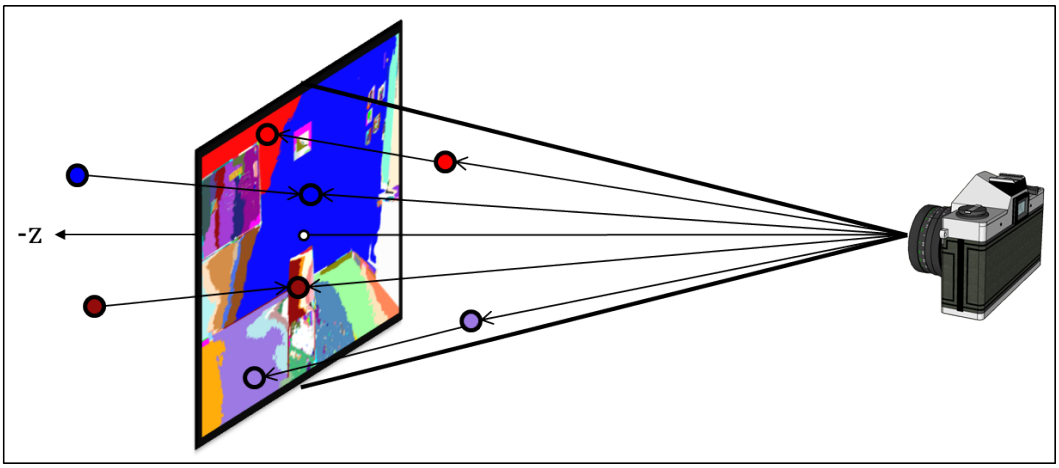


Abbildung 5.19: Zuordnung von Observed-Points zu Segmenten mithilfe der Observed-Pixel.

die Pixel eines Sprites verwendet. Dies ermöglicht eine Tiefenzuordnung ohne Segmentierung, jedoch sind die Konturgrenzen der *Depth-Patches* ungenauer als bei dem Segment-Depth-Matching-Verfahren. Dies ist auf das Verwenden kreisförmiger Sprites zurückzuführen. Beide Verfahren liefern flächengenaue, aber keine pixelgenauen Depth-Maps. Um pixelgenaue Depth-Maps zu erzeugen, wurde das Geometry-Depth-Matching-Verfahren entwickelt. Bei diesem Verfahren wird eine Szenengeometrie des abgebildeten Szenenausschnittes erzeugt und dadurch eine pixelgenaue Depth-Map erstellt. Die erzeugte Szenengeometrie stellt keine vollständige 3D-Rekonstruktion der Bilderweltenszene dar, da nur ein Szenenausschnitt aus der Sicht einer Kamera rekonstruiert wird.

Segment-Depth-Matching

Das *Segment-Depth-Matching*-Verfahren verwendet die Observed-Points einer Kamera, um den Segmenten (des Segmentbildes der Kamera) eine Tiefe zuzuordnen. Jeder Observed-Point besitzt für das Kamerabild einen *Observed-Pixel*. Über die Observed-Pixel werden alle Observed-Points einem Segment zugeordnet. Als Ergebnis dieser Zuordnung besitzt jedes Segment eine Anzahl von Observed-Points, wie in *Abb. 5.19* visualisiert wird. Die Anzahl ist abhängig von der Dichte der Punktwolke im entsprechenden Bereich der Bilderweltenszene, die im Kamerabild an der Stelle des Segmentes abgebildet wird.

Die zweite Information, die für die Tiefenzuordnung benötigt wird, ist die

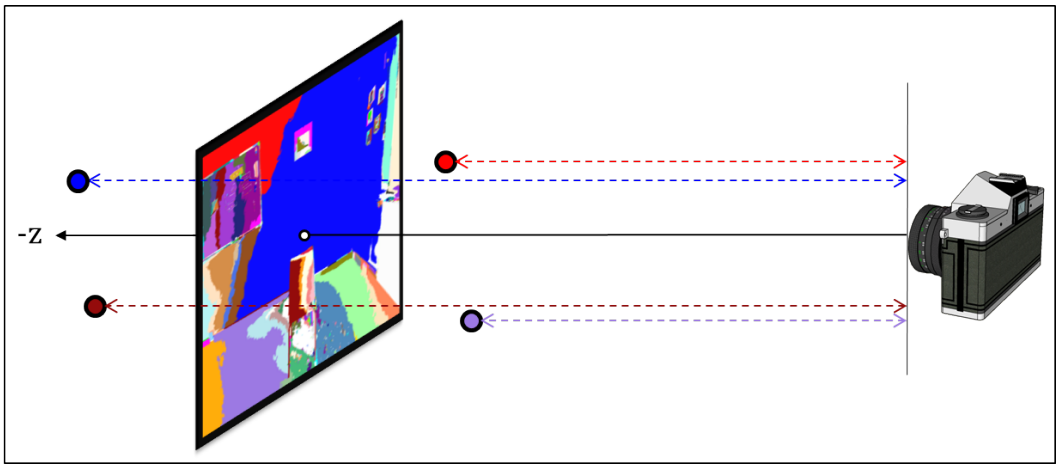


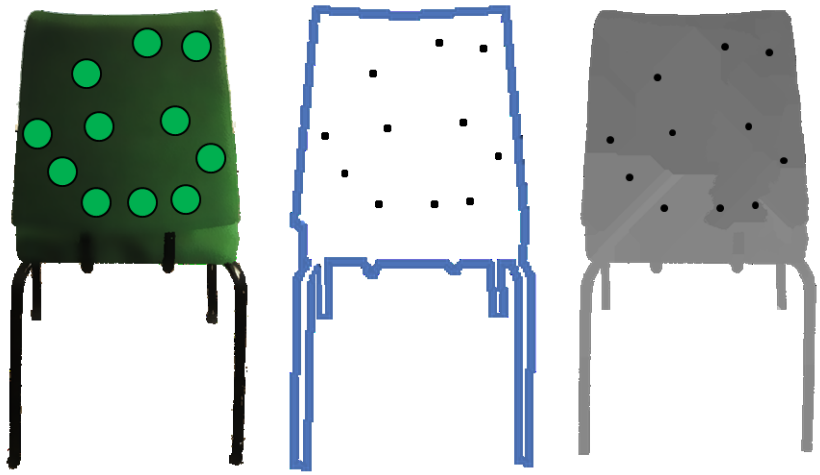
Abbildung 5.20: Visualisierung der Z-Distanzen, der in *Abb. 5.19* zugeordneten Observed-Points.

Z-Distanz zwischen Observed-Point und Kamera. Als *Z-Distanz* eines Observed-Points wird die Entfernung des Observed-Points zu seiner Kamera auf der bzw. parallel zur (negativen) Z-Achse des *Kamerakoordinatensystem*s (*KKS*) bezeichnet. *Abb. 5.20* zeigt schematisch die Z-Distanzen, der in *Abb. 5.19* zugeordneten Observed-Points.

Für jedes Segment wird ein Voronoi-Diagramm erzeugt. Die zugeordneten Observed-Points (bzw. ihre Observed-Pixel) dienen als Zellkern einer Voronoizelle und somit als Startpunkt für den Voronoi-Prozess. Die Z-Distanzen der Observed-Points dienen als Füllwerte für die Voronoizellen. Jede Voronoizelle repräsentiert einen Depth-Patch. Folglich besitzen Segmente so viele Depth-Patches wie zugeordnete Observed-Points. Dieser Vorgang ist in *Abb. 5.21* dargestellt.

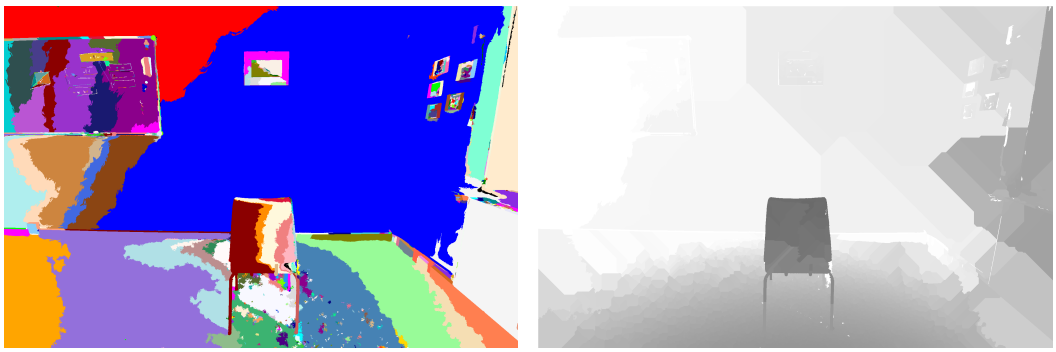
Als Ergebnis entsteht eine segmentbasierte Depth-Map, wie in *Abb. 5.22* dargestellt ist.

Der Vorteil dieses Verfahrens ist die segmentgenaue Zuordnung von Tiefenwerten mit nur geringen Tiefeninformationen bzw. mit einer geringen Anzahl an Observed-Points. Prinzipiell reicht ein Observed-Point pro Segment aus. Da eine Segmentierung vorausgegangen ist, sind die Konturgrenzen der abgebildeten Objekte innerhalb der Depth-Map sehr genau (vorausgesetzt die Segmentierung liefert konturgenaue Segmente). Der Nachteil dieses Verfahrens liegt in der Abhängigkeit bzw. Voraussetzung einer Segmentierung. Ein ungenaues oder fehlerhaftes Segmentbild führt zu einer fehlerhaften Depth-Patches. Ein wei-



(a) (b) (c)

Abbildung 5.21: Bildung einer Depth-Map durch das Segment-Depth-Matching-Verfahren: (a) Abgebildetes Objekt und Observed-Points; (b) Dazugehöriges Segment und Observed-Pixel (schwarz eingezeichnet); (c) Voronoi-Diagramm mit den eingezeichneten Observed-Pixel als Zellkerne und der Z-Distanz der Observed-Points als Füllwerte für die Voronoizellen (Depth-Patches).



(a) (b)

Abbildung 5.22: (a) Segmentbild und (b) dazugehörige Depth-Map des Segment-Depth-Matching-Verfahrens.

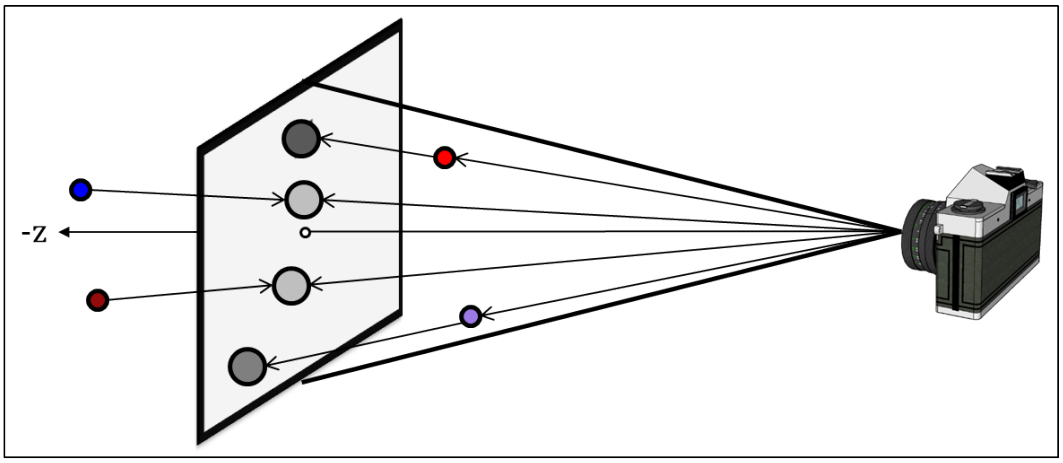


Abbildung 5.23: Rückprojektion von Observed-Points auf die Bildebene als kreisförmige Sprites mit der Z-Distanz des Observed-Points als Füllwert.

terer Nachteil stellt die Zuordnung von Segmenten und Observed-Points dar. Segmenten, denen keine Observed-Points zugeordnet werden können, können auch keine Tiefe zugeordnet bekommen. Eine Strategie für das Behandeln dieses Falles ist es, als Tiefenwert die Image-Distance der Kamera zu verwenden.

Um auch ohne vorangehende Segmentierung eine Tiefenzuordnung zu ermöglichen, wird im nachfolgenden Absatz das *Key-Point-Depth-Matching*-Verfahren erläutert.

Key-Point-Depth-Matching

Das *Key-Point-Depth-Matching*-Verfahren setzt keine vorangegangene Segmentierung voraus. Die Observed-Points werden auf die Bildebene als kreisförmige *Sprites* zurück projiziert, wie in *Abb. 5.23* dargestellt wird. Die Z-Distanzen der Key-Points werden dabei als Füllwerte für die Pixel der Sprites verwendet. Dabei ist (neben der Dichte der Punktwolke) der Radius der Sprites für die Flächenabdeckung und Genauigkeit der entstehenden Depth-Map verantwortlich. Je größer der Radius ist, desto größer ist die Flächenabdeckung, aber desto ungenauer werden die Depth-Patches.

Da nicht Punkte, sondern Flächen projiziert werden, kann es innerhalb der entstehenden Depth-Map zu Überschreibungen bereits eingezeichneter Sprites (bzw. Teile von ihnen) kommen. Deshalb werden vor der Rückprojektion die Observed-Points nach ihrer Z-Distanz absteigend sortiert. Entsprechend

(a) (b)

Abbildung 5.24: (a) Kamerabild und (b) dazugehörige Depth-Map des Key-Point-Depth-Matching-Verfahrens.

der absteigenden Sortierung werden weit entfernte Observed-Points vor nah liegenden Observed-Points projiziert. Dies dient dazu, nähere Tiefen genauer aufzulösen. Das Kreiszentrum eines Sprites ist durch den korrespondierenden Observed-Pixel definiert. *Abb. 5.24* zeigt eine Depth-Map, erstellt durch das Key-Point-Depth-Matching-Verfahrens.

Der Vorteil dieses Verfahrens liegt in der Einfachheit. Es müssen keinerlei Vorverarbeitungen durchgeführt werden. Dementsprechend muss keine vorangegangene Segmentierung vorausgesetzt werden. Folglich ist dieses Verfahren im Vergleich zu dem Segment-Depth-Matching-Verfahren weniger rechenintensiv. Der Nachteil dieses Verfahrens ist die Bedingung einer Punktwolke, die den abgebildeten Szenenausschnitt des Fotos verhältnismäßig vollständig abdeckt. Ist dies nicht der Fall, entstehen Bereiche in der Depth-Map ohne Tiefeninformationen. Eine Strategie für das Behandeln dieser Bereiche ist es, als Tiefenwert die Image-Distance der Kamera zu verwenden. Ein weiterer Nachteil stellen die Konturgrenzen der Depth-Patches dar. Diese sind ungenauer als bei dem Segment-Depth-Matching-Verfahren. Dies ist auf das Verwenden kreisförmiger Sprites zurückzuführen. Die Ungenauigkeit steigt mit zunehmenden Radius der Sprites.

Das Key-Point-Depth-Matching- sowie das Segment-Depth-Matching-Verfahren liefern keine pixel-, sondern flächengenaue Depth-Maps. Jedes Depth-Patch entspricht einer planaren Fläche mit einheitlichem Tiefenwert innerhalb der Depth-Map. Für Teilziel 2 (*Authentische Verdeckung bei freier Navigation*)

müssen jedoch pixelgenaue Tiefeninformationen vorliegen, um eine authentische Darstellung und Verdeckung der augmentierten Bilderwelt auch bei freier Navigation zu ermöglichen. Hierfür wurde vom Autor das Geometry-Depth-Matching-Verfahren entwickelt.

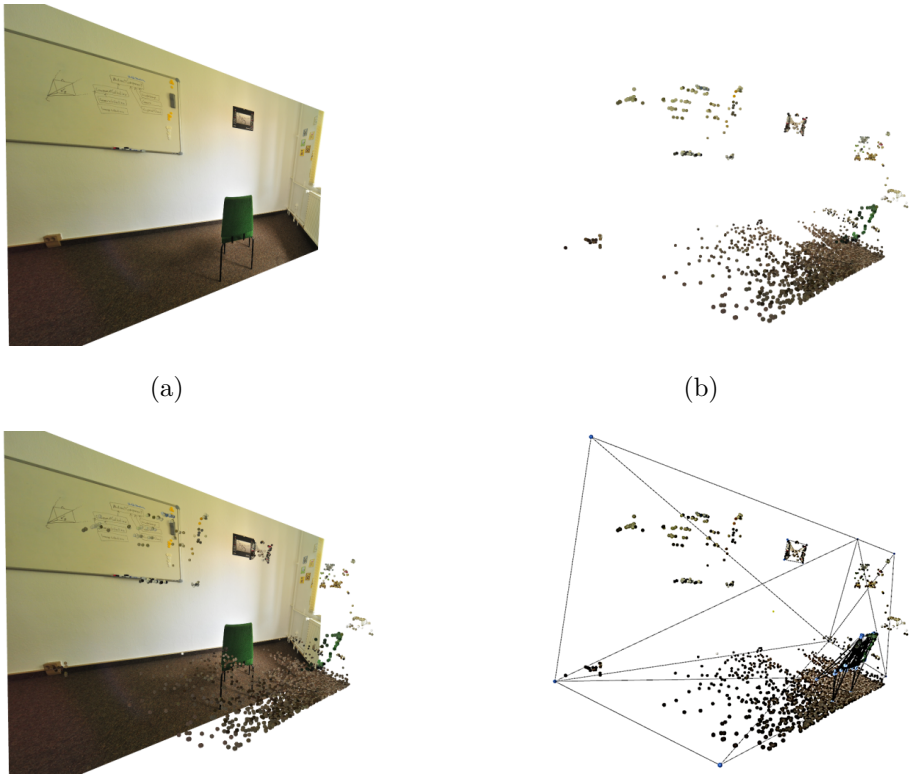
Geometry-Depth-Matching

Bei dem *Geometry-Depth-Matching*-Verfahren wird eine Szenengeometrie des abgebildeten Szenenausschnittes eines Fotos erzeugt und dadurch eine pixelgenaue Depth-Map erstellt.

Die Idee des Verfahrens ist es, aus gegebenen Informationen (ein Kamerabild, dessen Kameraparameter im 3D-Raum und eine zu der Abbildung dazugehörige 3D-Punktwolke) eine 3D-Geometrie zu erzeugen, die den abgebildeten Szenenausschnitt rekonstruiert. *Abb. 5.25* zeigt die grundlegende Idee dieses Verfahrens.

Eine für die Computergrafik offensichtliche Methode wäre das Triangulieren der Punktwolke. Die Key-Points besitzen neben ihrer 3D-Position eine Farbinformation. Dementsprechend könnte eine Gruppierung der Key-Points anhand ähnlicher 3D-Positionen und ähnlicher Farbe erfolgen, wie in *Abb. 5.26* dargestellt. Anhand eines Radius, der als prozentualer Wert zur Größe des abgebildeten Szenenausschnittes definiert ist, können Key-Points anhand ihrer 3D-Position gruppiert werden. Der Vergleich von Farben erfolgt analog zur *Farbbasierten Segmentierung* über den *Farbabstand* ΔE in einem Farbraum mit (annähernder) Gleichabständigkeit aller auftretenden Farben.

Die gruppierten Key-Points werden anschließend trianguliert. Auf diese Weise könnte die 3D-Geometrie des abgebildeten Szenenausschnittes eines Fotos erfolgen. Falls nicht nur die Observed-Points, sondern die gesamte Punktwolke verwendet wird, könnte somit die gesamte Bilderweltenszene als 3D-Geometrie rekonstruiert werden. In Unterkapitel *3.3* wurde aufgezeigt, dass 3D-Rekonstruktionen von Bilderwelten mit geringen Eingabematerial (wenig Eingabefotos) zu ungenauen und unbrauchbaren Ergebnissen führen. Auch bei der Triangulierung gruppierter Key-Points reichen die vorhandenen 3D-Informationen in Form der Punktwolke nicht aus, um eine brauchbare 3D-Geometrie eines Szenenausschnittes, noch der gesamten Bilderweltenszene zu rekonstruieren. *Abb. 5.27* zeigt die erzeugte 3D-Geometrie und die dazugehörige



(a) (b)

(c) (d)

Abbildung 5.25: Grundidee des Geometry-Depth-Matching-Verfahrens: Aus gegebenen Informationen wie das (a, c) Kamerabild und dessen Kameraparameter im 3D-Raum, sowie (b, c) eine zu der Abbildung dazugehörige 3D-Punktwolke soll eine (d) 3D-Geometrie des abgebildeten Szenenausschnittes rekonstruiert werden.

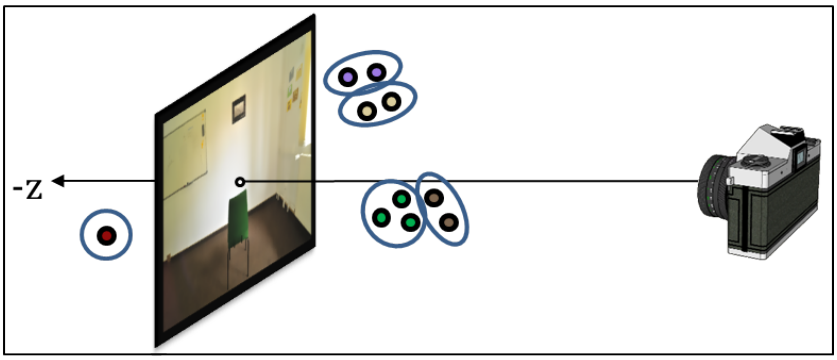
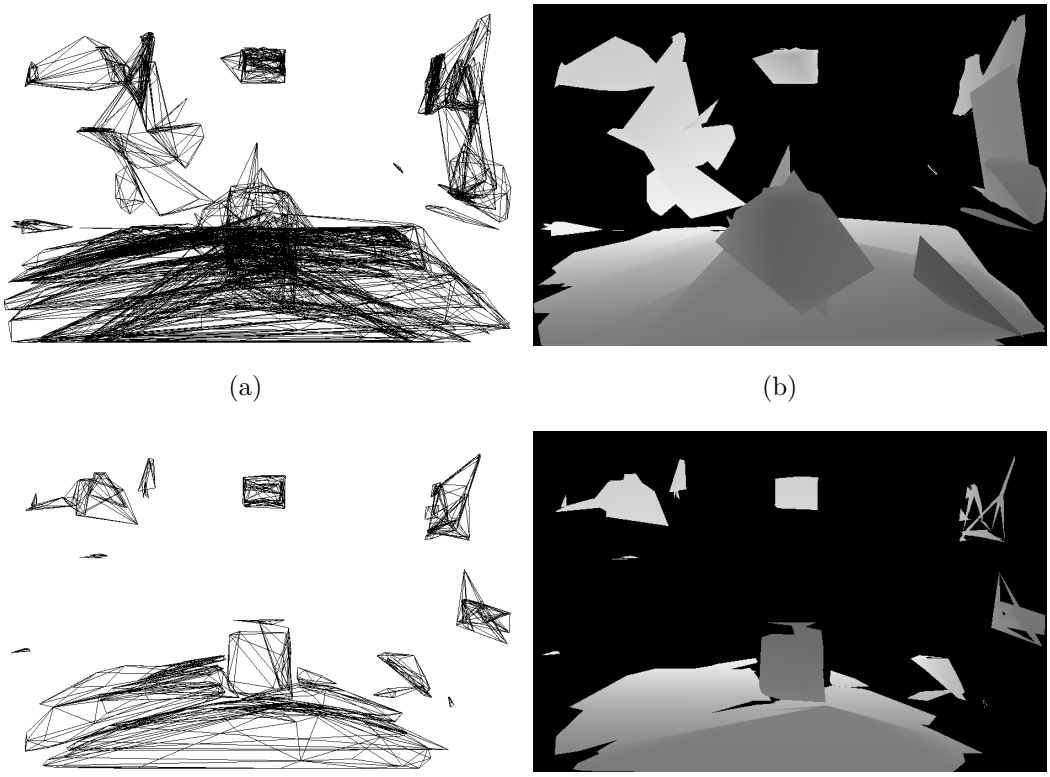


Abbildung 5.26: Gruppieren von Key-Points anhand ähnlicher Farbe und 3D-Positionen.



(a) (b)

(c) (d)

Abbildung 5.27: 3D-Mesh und Depth-Map der Triangulierung gruppierter Key-Points anhand von ähnlicher Farbe (LUV $\Delta E = 2.5$) und 3D-Positionen ($Radius = 5\%$): (a, c) Verwendung der gesamten Punktwolke (alle Key-Points); (b, d) Verwendung der Observed-Points.

Depth-Map von *Abb. 5.8*. Hierfür wurden zum einen die gesamte Punktwolke (alle Key-Points) und zum anderen die Observed-Points verwendet. Wie deutlich zu sehen ist, zeigt die Geometrie und somit auch die Depth-Map Ungenauigkeiten (speziell an den Objektkonturen) in den rekonstruierten Objekten. Des Weiteren existieren viele Bereiche ohne Geometrie-Information und somit ohne Tiefeninformationen in der Depth-Map. Folglich erweist sich die Triangulierung der Punktwolke als unzureichendes Verfahren zur Gewinnung pixelgenauer Tiefeninformationen.

Aus diesem Grund erstellt das Geometry-Depth-Matching-Verfahren die Geometrie des abgebildeten Szenenausschnittes eines Fotos auf eine andere Weise. Als Voraussetzung dient das Ergebnis der *Objektbasierten Segmentierung*. Das bedeutet, es liegt ein (fast) fehlerfreies Segmentbild auf Basis einer

semiautomatischen Segmentskizzierung vor, sowie die Kenntnis der Eckpunkte der Segmente (siehe *Abb. 5.16(b)*). Das Geometry-Depth-Matching-Verfahren arbeitet in drei Schritten:

a) Bildung von Eckpunkt-Tripeln,

b) Erzeugung von 3D-Triangles,

c) Erzeugung eines 3D-Meshes.

a) Bildung von Eckpunkt-Tripeln

Aus den Eckpunkten werden Dreiecksbeziehungen hergestellt, die im weiteren Text als *Eckpunkt-Tripel* bezeichnet werden. Ein Eckpunkt-Tripel wird erzeugt, indem die Strecken, gebildet aus einem Eckpunkt A und jedem anderen Eckpunkt des gleichen Segmentes, abgetastet werden. Befindet sich eine Strecke zwischen zwei Eckpunkten innerhalb des Segmentes (oder auf der Konturgrenze), wird (ausgehend vom zweiten Eckpunkt B) ebenfalls die Strecken zu allen weiteren Eckpunkten des Segmentes abgetastet. Sobald auf diese Weise ein dritter Eckpunkt C detektiert wurde, wird die Strecke von C zu A abgetastet. Befindet sich diese Strecke wieder innerhalb des Segmentes (oder auf der Konturgrenze) ist ein Tripel, bestehend aus den Eckpunkten $\{A, B, C\}$, gefunden wurden. Folglich entstehen eine Anzahl Tripel, dessen 2D-Eckpunkte im folgenden Schritt in 3D überführt werden.

b) Erzeugung von 3D-Triangles

Den vorliegenden Segmenten werden (analog zu dem *Segment-Depth-Matching*-Verfahren) Observed-Points zugeordnet, wie in *Abb. 5.28* dargestellt wird. Folglich besitzt jedes Segment eine Anzahl von Eckpunkt-Tripeln sowie eine Anzahl zugeordneter Observed-Points. Im diesem Schritt wird für jeden 2D-Eckpunkt eines Tripels ein neuer 3D-Key-Point erzeugt. Auf diese Weise entstehen aus den 2D-Eckpunkt-Tripeln *3D-Triangle*s. Hierfür wird für den Eckpunkt jener Key-Point aus den zugeordneten Observed-Points ausgewählt, der im Foto (also im 2D-Bildkoordinatensystem) dem Eckpunkt am nächstliegenden ist. Das Berechnen der Entfernungen erfolgt über die *Euklidische Distanz* der Observed-Pixel und den 2D-Eckpunktpositionen und wird im weiteren Text als

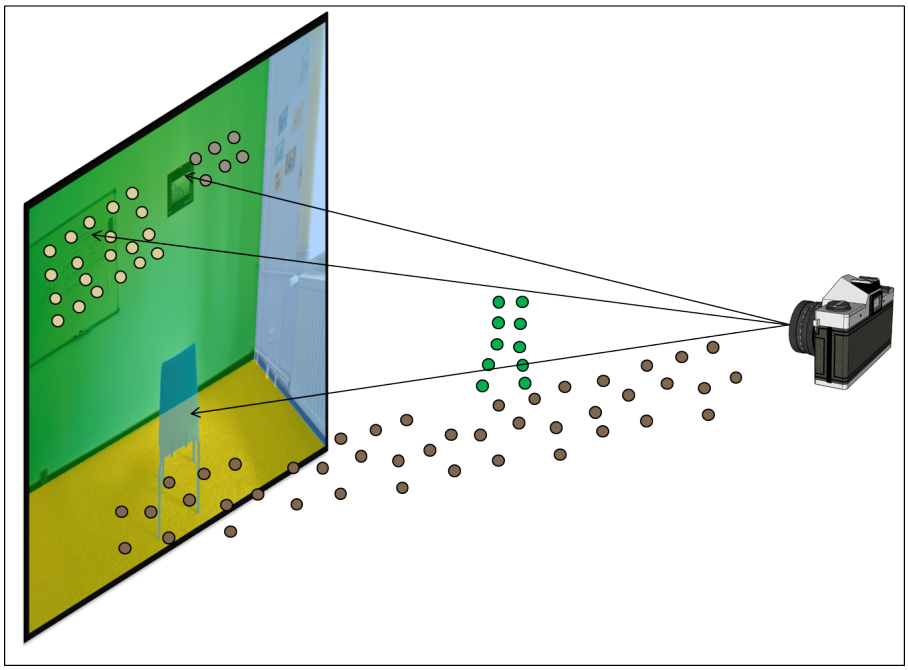


Abbildung 5.28: Zuordnen der Observed-Points zu den Segmenten der *Objekt-basierten Segmentierung*.

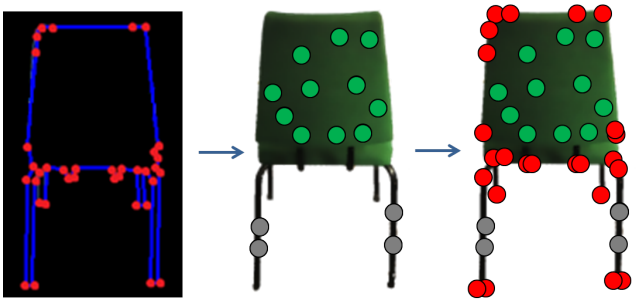


Abbildung 5.29: Erzeugen neuer 3D-Key-Points (rechts, rot dargestellt) als 3D-Rekonstruktion der 2D-Eckpunkte (links, ebenfalls rot) basierend auf dem Segment zugeordneten Observed-Points (grün und grau dargestellt).

d_{2D} bezeichnet. Als Ergebnis besitzt jeder 2D-Eckpunkt einen zugeordneten 3D-Key-Point. Auf Basis des zugeordneten Key-Points wird für jeden Eckpunkt ein neuer Key-Point erzeugt, der die gleiche Z-Distanz besitzt, wie der zugeordnete Key-Point, jedoch die x- und y-Koordinate neu berechnet wird. In *Abb. 5.29* ist dies schematisch dargestellt. Hierfür sollen als Vorüberlegung eine Auflistung bereits gegebener Informationen zusammen getragen werden:

- Bildauflösung ($resolution$),
- Image-Distance ($zDistance_{image}$),
- Größe der Image-Plane ($3dSize_{original}$),
- 3D-Position und Z-Distanz ($zDistance_{keypoint}$) des zugeordneten Key-Points,
- 2D-Position des zugeordneten Key-Points (Observed-Pixel),
- 2D-Position des Eckpunktes,
- Distanz zwischen beiden 2D-Positionen (d_{2D}).

Für das Erzeugen des neuen Key-Points werden mehrere Schritte durchlaufen. *Abb. 5.30* zeigt diese Schritte. Die Ausgangssituation ist in *Abb. 5.30(a)* dargestellt. Die Image-Plane wird auf die gleiche Z-Distanz ($zDistance_{keypoint}$) wie der zugeordnete Key-Point verschoben. (siehe *Abb. 5.30(b)*). Da die Image-Plane in einer Bilderwelt stets orthogonal zur Kamera platziert wird, lässt sich die Größe der verschobenen Image-Plane durch eine lineare Verhältnisgleichung berechnen (siehe *Formel 5.6*).

$$3dSize_{translated} = \frac{3dSize_{original} * zDistance_{keypoint}}{zDistance_{image}} \tag{5.6}$$

Formel 5.6: Berechnung der Größe der verschobenen Image-Plane.

In *Formel 5.7* wird mithilfe der berechneten Größe der verschobenen Image-Plane die dreidimensionale Distanz zwischen zugeordnetem und neu zu erzeugendem Key-Point berechnet (ebenfalls in *Abb. 5.30(b)* dargestellt).

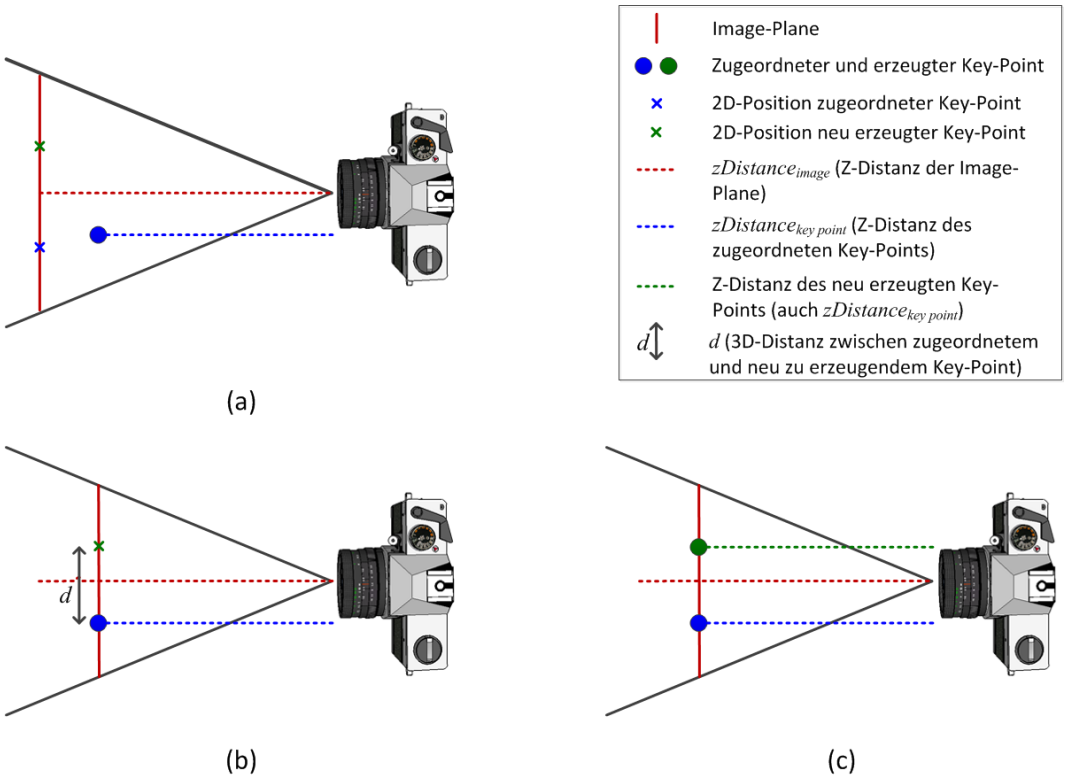


Abbildung 5.30: Erzeugen eines neuen Key-Points: (a) Ausgangssituation; (b) Verschieben und Skalieren der Image-Plane an die Z-Distanz des zugeordneten Key-Points sowie das Berechnen der Distanz (im 3D-Raum) zwischen zugeordneten und neu zu erzeugenden Key-Point; (c) Erzeugen eines neuen Key-Points anhand der Position des zugeordneten Key-Points, verschoben um die berechnete Distanz.

$$d = \frac{3dSize_{translated} * d_{2D}}{resolution} \tag{5.7}$$

Formel 5.7: Distanz zwischen zugeordnetem und neu zu erzeugendem Key-Point.

Der neu zu erzeugende Key-Point erhält die 3D-Position des zum Eckpunkt zugeordneten Key-Points verschoben um d. Das Ergebnis ist in *Abb. 5.30(c)* dargestellt.

Als Sonderfall muss bei der Erzeugung neuer Key-Points berücksichtigt werden, dass im Rahmen der *Objektbasierten Segmentierung* Eckpunkte für mehrere Segmente verwendet werden können. Dementsprechend muss für einen Eckpunkt, der in mehreren Segmenten verwendet wird, nur ein Key-Point erzeugt werden. Als Ergebnis dieses Schrittes entstehen aus den 2D-Eckpunkt-Tripeln so genannte 3D-Triangles (bestehend aus neu erzeugten Key-Points). Die 3D-Triangles werden im nachfolgenden Schritt für die Erzeugung von 3D-Meshes verwendet.

c) Erzeugung von 3D-Meshes

Aus den erzeugten 3D-Triangles, werden in diesem Schritt *Surface*s für ein 3D-Mesh erzeugt. Neben den gegebenen 3D-Koordinaten $(p1, p2, p3)$ muss die korrespondierende Normale berechnet werden (siehe *Formel 5.8*). Um sicherzustellen, dass die Normale des Surfaces ($\overrightarrow{n}$) in die Richtung der Kamera zeigt, muss der Winkel θ, der sich aufspannt zwischen dem Vektor von der Kamera zu dem Surface ($\overrightarrow{ray}$) und der Normale $\overrightarrow{n}$, berechnet werden. *Formel 5.11* zeigt, wie der Winkel θ eine Negation der Normalen bestimmt [MM12]. Der Vektor $\overrightarrow{ray}$ wird in *Formel 5.9* berechnet und der Winkel θ wird über das Skalarprodukt in *Formel 5.10* berechnet. Anzumerken ist, dass für die Berechnung von θ die Vektoren $\overrightarrow{ray}$ und $\overrightarrow{n}$ normalisiert werden müssen.

$$\overrightarrow{n} = \overrightarrow{(p1 - p3)} \times \overrightarrow{(p2 - p3)} \tag{5.8}$$

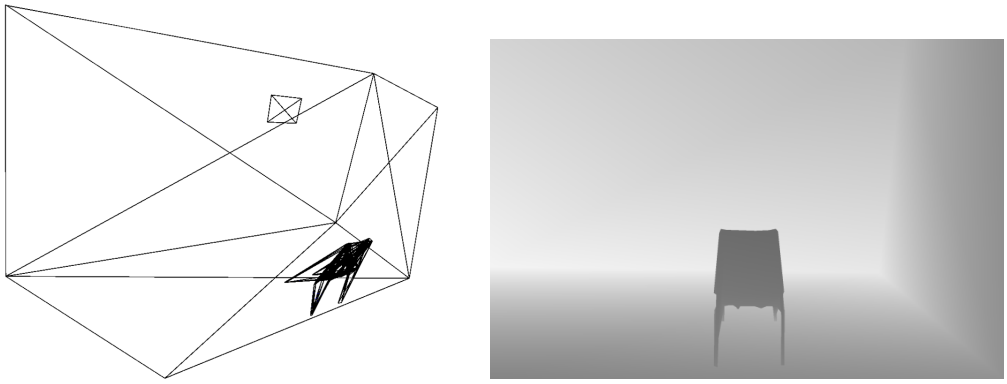
$$\overrightarrow{ray} = \frac{\overrightarrow{(p1 + p2 + p3)}}{3} - zDistance_{keypoint} \tag{5.9}$$

$$\theta = \overrightarrow{ray}_{normalized} \cdot \overrightarrow{n}_{normalized} \tag{5.10}$$

$$\overrightarrow{n} = \begin{cases} -\overrightarrow{n} & \text{if } \theta > 90° \\ \overrightarrow{n} & else \end{cases} \tag{5.11}$$

Formel 5.11: Erzeugung und korrekte Ausrichtung der Normale eines Surfaces.

Als Ergebnis dieses Schrittes liegen Surfaces für jedes Segment vor. Folglich steht die 3D-Geometrie des in dem Foto abgebildeten Szenenausschnittes zur



(a) 3D-Geometrie (b) Depth-Map

Abbildung 5.31: (a) 3D-Geometrie des in dem Foto abgebildeten Szenenausschnittes und (b) die dazugehörige (annähernd) pixelgenaue Depth-Map.

Verfügung. Somit kann über die Z-Distanz eines jeden Punktes der Geometrie eine (annähernd) akkurate, pixelgenaue Depth-Map erzeugt werden, wie in *Abb. 5.31* dargestellt wird.

Der Vorteil dieses Verfahrens gegenüber den anderen beiden Depth-Matching-Verfahren stellt die Pixelgenauigkeit der gewonnenen Tiefeninformationen dar. Während das Segment-Depth-Matching- und das Key-Point-Depth-Matching-Verfahren planare Flächen in den Depth-Maps erzeugen, wird bei dem Geometry-Depth-Matching-Verfahren eine (annähernd) akkurate, pixelgenaue Tiefenkarte generiert. Nachteilig an diesem Verfahren ist die Voraussetzung der *Objektbasierten Segmentierung*, die wiederum einen semiautomatischen Schritt (*Segmentskizzierung*) bedingt.

Zusammenfassung und Diskussion

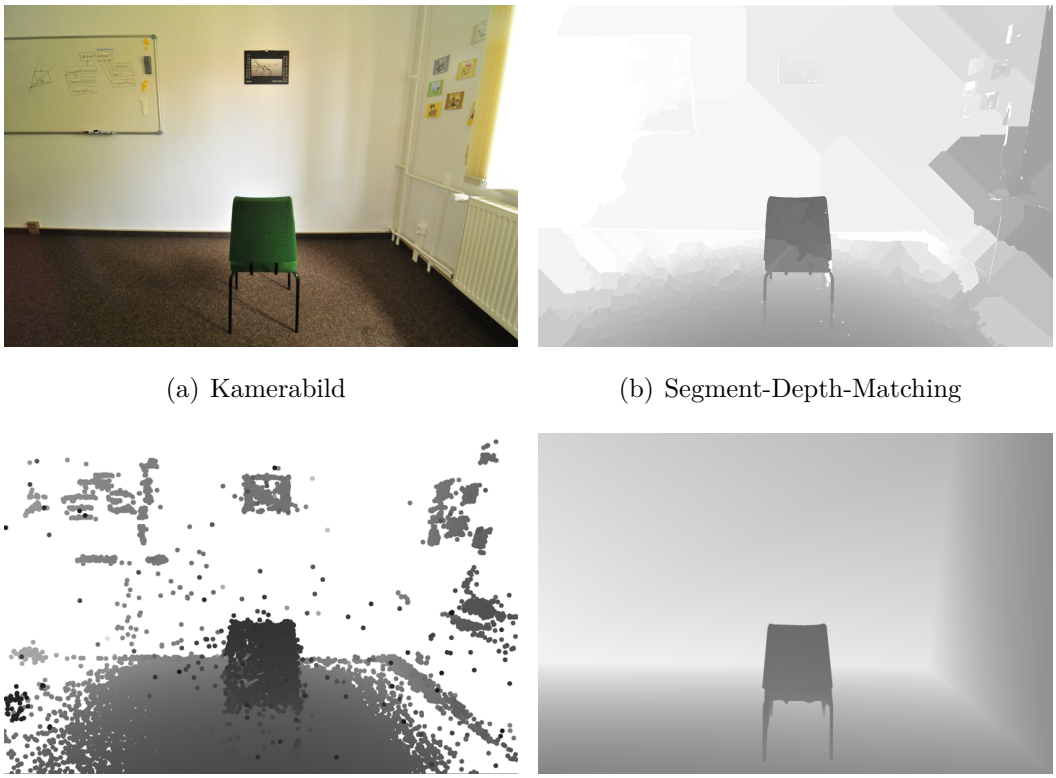
In diesem Absatz wurden drei verschiedene Verfahren zur Tiefenzuordnung entwickelt. Das *Segment-Depth-Matching*-Verfahren zerlegt anhand einer farbbasierten Segmentierung das Bild in die dort abgebildeten Objekte. Den einzelnen Segmenten wird mithilfe der Observed-Points eine Tiefe zugeordnet. Auf diese Weise entsteht eine segmentbasierte Depth-Map. Dieses Verfahren setzt nur wenig Key-Points voraus und eignet sich daher auch für kleine Szenen mit wenig Bildern. Das *Key-Point-Depth-Matching*-Verfahren verzichtet auf eine Segmentierung des Bildes. Es werden die Observed-Points der Punktwolke als

Verfahren	Vorteile	Nachteile
Segment-Depth-Matching	- Sehr gute Ergebnisse bei akkuraten Segmentbildern als Eingabedaten - Wenig 3D-Informationen (Observed-Points) notwendig - Vollautomatische Berechnung	- Fehleranfällig bei ungenauen oder fehlerhaften Segmentbildern als Eingabedaten - Für Segmente ohne zugeordneten Observed-Points keine Tiefenzuordnung möglich - Keine pixelgenaue (nur flächengenaue) Tiefenzuordnung
Key-Point-Depth-Matching	- Keine vorangehende Segmentierung als Voraussetzung - Weniger rechenintensiv als die anderen Verfahren - Einfache Umsetzung - Vollautomatische Berechnung	- Bedingung einer Punktwolke, die den abgebildeten Szenenausschnitt des Fotos verhältnismäßig vollständig abdeckt - Ungenaue Konturgrenzen der abgebildeten Objekte innerhalb der Depth-Map - Bereiche in der Depth-Map ohne Tiefeninformationen unvermeidbar - Keine pixelgenaue (nur flächengenaue) Tiefenzuordnung
Geometry-Depth-Matching	- Akkurate, pixelgenaue Depth-Maps	- Voraussetzung der *Objektbasierten Segmentierung* (semiautomatische *Segmentskizzierung*)

Tabelle 5.4: Vor- und Nachteile der Tiefenzuordnungsverfahren.

zweidimensionale kreisförmige Sprites direkt auf die Bildebene projiziert. Der Pixelfüllwert der Sprites kodiert die Tiefe des dazugehörigen Observed-Points. Ein definierter Radius bestimmt die Größe der Sprites. Mit zunehmenden Radius werden größere Flächen mit Tiefeninformationen versehen, jedoch steigt damit die Ungenauigkeit der Depth-Map. Dieses Verfahren eignet sich speziell bei großen Szenen mit einer ausgeprägten Punktwolke. Das *Geometry-Depth-Matching*-Verfahren generiert aus einer Kamera und den Observed-Points eine 3D-Geometrie des im Kamerabildes abgebildeten Szenenausschnittes. Aufbauend auf den Segmenten der *Objektbasierten Segmentierung* werden neue Key-Points als 3D-Rekonstruktion der 2D-Segment-Eckpunkte erzeugt. Diese neuen Key-Points werden trianguliert und erzeugen somit eine 3D-Geometrie. Abgrenzend zu einer vollständigen 3D-Rekonstruktion wird bei diesem Verfahren nur eine Teilrekonstruktion des im Foto abgebildeten Szenenausschnittes rekonstruiert. Mithilfe dieses Verfahrens kann eine (annähernd) pixelgenaue Depth-Map generiert werden.

Bei der Erläuterung der einzelnen Verfahren wurden jeweils die Vor- und Nachteile aufgezeigt. Diese werden in *Tab. 5.4* noch einmal zusammengetragen. Die Nachteile und Schwachstellen eines Verfahrens bildeten den Anstoß und die Motivation für das jeweils nachfolgende Verfahren. Somit verfolgte dieser Absatz (analog zu Absatz *5.3.2* (*Bildsegmentierung*)) das Bottom-Up-Prinzip. Als Ergebnis liegen zu einem Kamerabild die korrespondierenden Tiefeninformationen (in Form einer Depth-Map) vor. Diese sind Voraussetzung für die Verfahren, die im nachfolgenden Absatz *5.4* (*Authentische Darstellung der Bilderwelten*) beschrieben werden. Die Depth-Maps der drei entwickelten Verfahren werden noch einmal in *Abb. 5.32* gegenübergestellt. Eine Evaluierung der Tiefenzuordnungsverfahren findet sich in Absatz *7.2.2* (*Evaluierung der Tiefenzuordnung*). *Abb. 5.33* zeigt die Ausgabedaten nach der Tiefenzuordnung.



(a) Kamerabild

(b) Segment-Depth-Matching

(c) Key-Point-Depth-Matching

(d) Geometry-Depth-Matching

Abbildung 5.32: Gegenüberstellung der erzeugten Depth-Maps der drei entwickelten Tiefenzuordnungsverfahren.

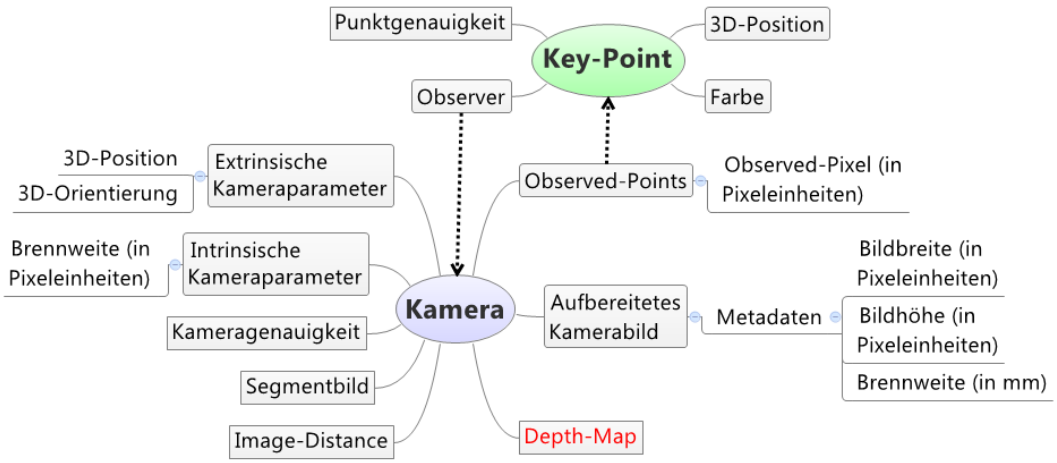


Abbildung 5.33: Ausgabedaten nach der Tiefenzuordnung. Neue Daten sind rot beschriftet.

5.4 Authentische Darstellung der Bilderwelten

5.4.1 Überblick

Aufbauend auf den gewonnenen Tiefeninformationen realisiert dieser Teilschritt den Augmentierungsschwerpunkt *Verdeckung* innerhalb augmentierter Bilderwelten.

Auf Basis gewonnener Tiefeninformationen wird mit einem vom Autor entwickelten *Image-Based-Rendering*-Verfahren eine authentische Verdeckung realisiert, ohne hierzu eine 3D-Geometrie der Bilderweltenszene vorauszusetzen. Zusätzlich wird ein alternatives Renderverfahren vorgestellt, das auf Basis rekonstruierter Szenengeometrie des in einem Kamerabild abgebildeten Szenenausschnittes eine authentische Darstellung und Verdeckung ermöglicht.

5.4.2 Flächenbasierte Darstellung

Die gewonnene Depth-Map beinhaltet die Tiefeninformationen für ein Foto der Bilderwelt. Somit kann im Falle einer Augmentierung anhand der 3D-Position der integrierten virtuellen Objekte bewertet werden, ob einzelne Bildanteile des Bildes virtuelle Anteile überlagern und somit verdecken müssen. Ein offensichtlicher Ansatz in der Computergrafik ist es, den Tiefenpuffer-Algorithmus [Cat74] zu verwenden. Hierzu muss der Tiefenpuffer mit den Tiefeninformationen der Depth-Map überschrieben werden. Aufgrund der perspektivischen Projektion (siehe *2.4.2*) und der nicht linearen Auflösung des Tiefenpuffers [Tre99] ergibt sich ein Wert für den Tiefenpuffer entsprechend der *Formel 5.12*. Z_{Near} beschreibt den Wert der Near-Clipping-Plane, Z_{Far} den Wert der Far-Clipping-Plane und Z den Wert aus der Depth-Map.

$$Z_{Buffer} = \frac{\frac{1}{Z} - \frac{1}{Z_{Near}}}{\frac{1}{Z_{Far}} - \frac{1}{Z_{Near}}} \tag{5.12}$$

Formel 5.12: Berechnung des Tiefenpufferwertes ([Tre99]).

Auf diese Weise kann eine korrekte Verdeckung eingebetteter virtueller Objekte aus der Sicht der Kamera (Teilziel 1 *Authentische Verdeckung bei Kameranavigation*) erreicht werden. Jedoch wird das Kamerabild weiterhin als

(a) (b)

Abbildung 5.34: Überschreiben des Tiefenpuffers (anhand der Tiefeninformationen des Geometry-Depth-Matching-Verfahrens): (a) Korrekte Verdeckung aus der Kamerasicht (Teilziel 1). (b) Fehlerhafte Verdeckung bei freier Navigation (Teilziel 2).

zweidimensionale, texturierte Image-Plane im 3D-Raum dargestellt. Dementsprechend kann bei freier Navigation keine korrekte Verdeckung umgesetzt werden. *Abb. 5.34* zeigt das Ergebnis aus der Sicht der Kamera und aus einer freien Position. Somit eignet sich dieser Ansatz nicht für das Erreichen des zweiten Teilziels (*Authentische Verdeckung bei freier Navigation*).

In [NGBA10] sowie in [NKG11c] wurde vom Autor in Zusammenarbeit mit Co-Autoren das Konzept des *Sliced-Image-Renderings* veröffentlicht. Das Sliced-Image-Rendering stellt ein *Image-Based-Rendering*-Verfahren dar, dass ohne 3D-Geometrie ein 2D-Bild anhand gegebener Tiefeninformationen als dreidimensionale Darstellung rendern und auf diese Weise eine authentische Verdeckung auch bei freier Navigation ermöglichen kann. Auf Basis gewonnener Tiefeninformationen wird ein Bild innerhalb der Bilderweltenszene in Slices zerlegt. Ein Slice repräsentiert eine Bildregion des Originalbildes. Jedes Slice wird an die für die Region entsprechende Tiefenposition, die in der Depth-Map gespeichert wurde, verschoben. *Abb. 5.35* zeigt den schematischen Aufbau eines Sliced-Image. Die Slices befinden sich in der Tiefe, in denen sich die korrespondierenden Key-Points (die Key-Points, die für die Tiefenzuordnung der entsprechenden Bildregion genutzt wurden) befinden. Die Gesamtheit aller Slices ergibt aus der Sicht der Kamera das Kamerabild.

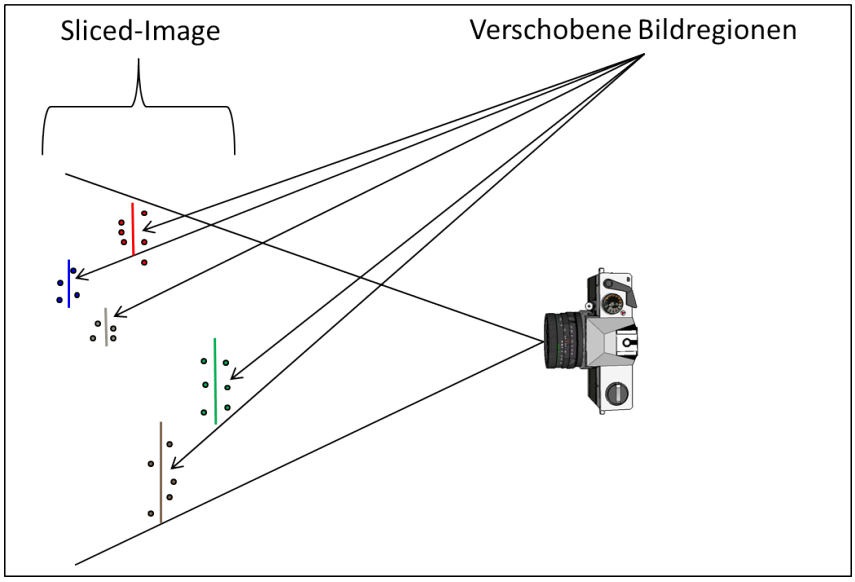


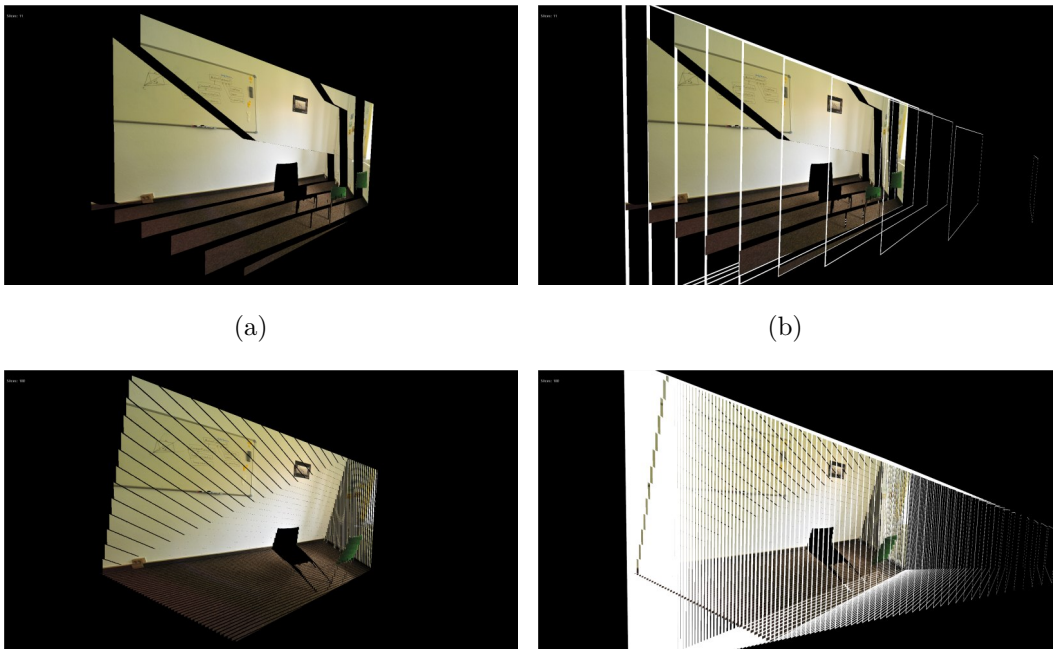
Abbildung 5.35: Schematischer Aufbau eines Sliced-Image. Die Gesamtheit aller Slices ergibt aus Sicht der Kamera das Kamerabild.

Damit alle Regionen zusammen den Eindruck eines Bildes vermitteln, müssen die verschobenen Slices perspektivisch korrekt projiziert werden. Zur Kamera verschobene Slices müssen verkleinert und von der Kamera weg verschobene Slices müssen vergrößert werden. Da die Fotos immer orthogonal zur Kamera stehen und die Maße sowie die Distanz der Image-Plane bekannt sind, kann diese Skalierung durch eine lineare Verhältnisgleichung beschrieben werden:

$$\frac{zDistance_{image}}{height_{image}} = \frac{zDistance_{slice}}{height_{slice}} \tag{5.13}$$

Formel 5.13: Perspektivische Projektion der Slices.

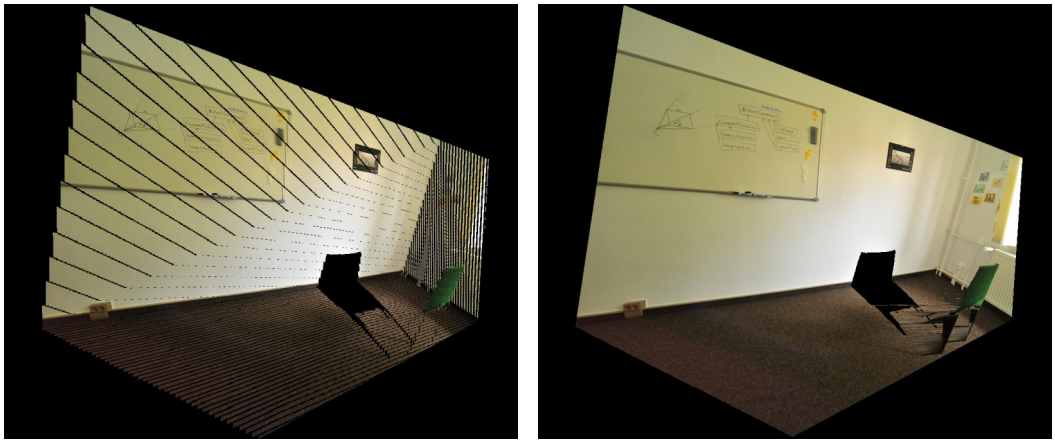
Die Gesamtheit aller verschobenen, skalierten Regionen wird Sliced-Image genannt. Jedes Slice muss als eigenes Bild dargestellt werden, bei dem lediglich die zugesprochene Bildregion opak und der restliche Bildinhalt transparent gezeichnet wird. Um den Darstellungsaufwand möglichst gering zu halten, können Slices ähnlicher Tiefe zusammengefasst werden. Eine geringere Anzahl an Slices bedeutet weniger Renderlast. Dies bedeutet aber auch größere Abstufungen zwischen den Slices, was zum einen zu einer ungenaueren Darstellung des abgebildeten Szenenausschnittes und zum anderen zu größeren schwarzen Bereichen



(a) (b)

(c) (d)

Abbildung 5.36: Darstellung eines Sliced-Images bei freier Navigation (anhand der Tiefeninformationen des Geometry-Depth-Matching-Verfahrens): (a) Mit 10 Slices (b) und weißem Rand (ausschließlich aus Visualisierungsgründen). (c) Mit 100 Slices (d) und weißem Rand. Die Abbildungen zeigen, dass mit zunehmender Anzahl an Slices auch die Darstellungsgenauigkeit des Szenenausschnittes zunimmt.



(a) (b)

Abbildung 5.37: Darstellung des Sliced-Images (a) ohne und (b) mit Interpolation benachbarter Slices mittels *Displacement-Mapping* [WWT+03].

zwischen den Slices bei der freien Navigation führt. Dementsprechend richtet sich die Anzahl und Abstufung der Slices an den Anwendungsfall. Für das Erreichen von Teilziel 1 und der damit verbundenen Kameranavigation reichen nur wenige Slices aus, da lediglich unterschieden werden muss, welche Bildanteile ein virtuelles Objekt überlagert und welche nicht. Für eine *Authentische Verdeckung bei freier Navigation* (Teilziel 2) wird eine höhere Anzahl an Slices benötigt, da eine möglichst detailreiche Darstellung des abgebildeten Szenenausschnittes notwendig ist. *Abb. 5.36* zeigt die Darstellung eines Sliced-Images bei freier Navigation (anhand der Tiefeninformationen des Geometry-Depth-Matching-Verfahrens) mit geringer Anzahl an Slices (a+b) und mit größerer Anzahl an Slices (c+d). Die Abbildungen zeigen, dass mit zunehmender Anzahl an Slices auch die Darstellungsgenauigkeit des Szenenausschnittes zunimmt.

Dennoch werden auch bei sehr großer Anzahl an Slices schwarze Bereiche zwischen den Slices vorhanden sein. Diese Bereiche werden mittels *Displacement-Mapping* [WWT+03] als Interpolation zwischen benachbarten Slices geschlossen. *Abb. 5.37* zeigt einen Vergleich des Sliced-Images (mit 100 Slices) aus *Abb. 5.36* ohne und mit Interpolation durch Displacement-Mapping.

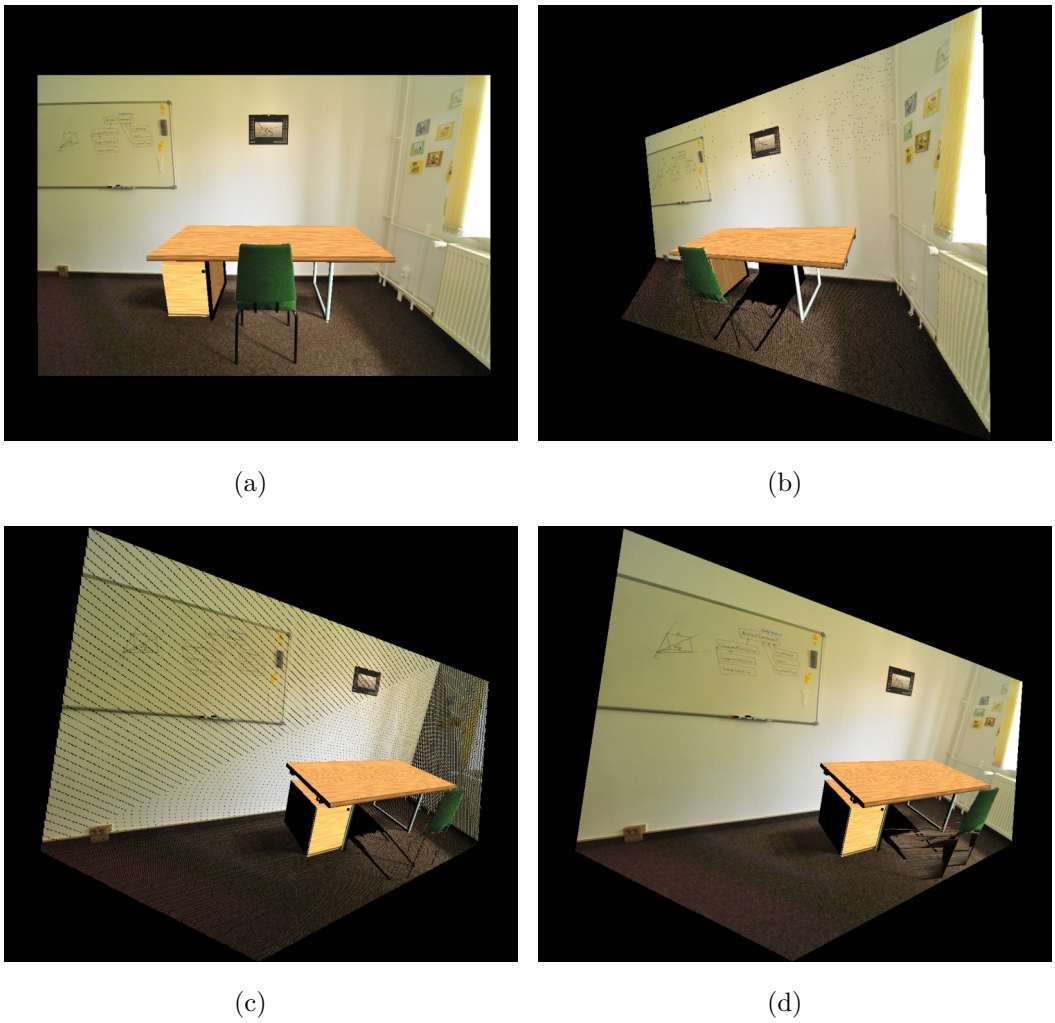
Der Vorteil des Sliced-Image-Renderings ist die authentische Verdeckung in augmentierten Bilderwelten bei freier Navigation, ohne dabei eine 3D-Szenengeometrie vorauszusetzen, wie in *Abb. 5.38* gezeigt wird. Es werden ledig-

lich Tiefeninformationen des Kamerabildes vorausgesetzt. Folglich kann dieses Image-Based-Rendering-Verfahren auch in Anwendungen verwendet werden, die keinerlei Handling von 3D-Geometrien ermöglichen. Beispielhaft soll an dieser Stelle Google Street View [Goo07] genannt werden. Zwar existieren in dieser Bilderweltenanwendung virtuelle Anteile als 2D-Sprites (z. B. virtuelle Pfeile und Verbindungslinien entlang einer Wegstrecke), jedoch kann Street View keine 3D-Geometrien einbetten. Eine dreidimensionale Darstellung einer Szene wäre dennoch mit dem Sliced-Image-Rendering möglich. Der Nachteil des Sliced-Image-Renderings ist, dass mit abnehmender Korrektheit der Depth-Map auch die Darstellung der Slices an Darstellungsqualität verliert. Des Weiteren können nicht alle schwarzen Bereiche durch Interpolation restlos beseitigt werden.

5.4.3 Darstellung gegebener 3D-Geometrie

In Absatz *1.3* (*Kernanforderungen an die Arbeit*) wurde als Anforderung an diese Arbeit definiert, dass zur Erreichung einer authentischen Verdeckung keine 3D-Geometrie verwendet sowie keine vollständige 3D-Rekonstruktion der Bilderweltenszene durchgeführt werden soll. Dies ist durch das *Sliced-Image-Rendering*-Verfahren in Verbindung mit einem der drei *Depth-Matching*-Verfahren realisierbar. Im Rahmen des *Geometry-Depth-Matching*-Verfahrens wird eine Szenengeometrie des in einem Kamerabild abgebildeten Szenenausschnittes rekonstruiert, um daraus eine (annähernd) pixelgenaue Depth-Map zu generieren. Je nach Anwendungsfall bzw. nach Anforderungen und Unterstützung von 3D-Geometriedaten der Bilderweltenanwendung kann diese intern erzeugte Geometrie innerhalb des Bilderwelten-Renderprozesses verwendet werden. Dies wird im weiteren Text als *Image-Geometry-Rendering* bezeichnet. Hierfür wird die Geometrie mittels perspektivischer Projektion des Kamerabildes texturiert. Es entsteht eine *3D-Bilddarstellung*. Somit kann ebenfalls eine authentische Verdeckung eingebetteter virtueller Objekte innerhalb der Bilderwelten erfolgen, wie in *Abb. 5.39* gezeigt wird.

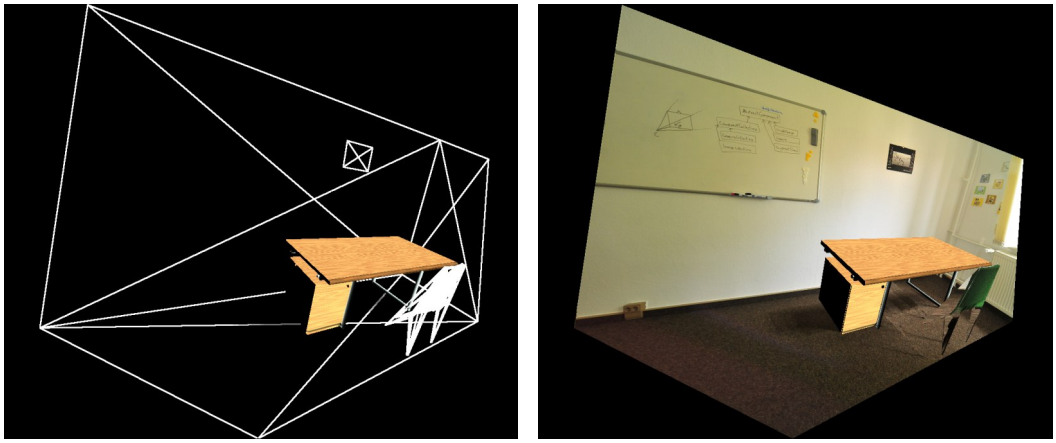
Der Vorteil des Image-Geometry-Renderings ist der geringere Renderaufwand gegenüber dem Sliced-Image-Rendering, da lediglich ein texturiertes 3D-Objekt dargestellt werden muss. Jedoch fordert dieses Verfahren zwingend die Szenengeometrie des in einem Kamerabild abgebildeten Szenenausschnittes. Folglich



(a) (b)

(c) (d)

Abbildung 5.38: Authentische Verdeckung eines virtuellen Objektes durch die dreidimensionale Darstellung eines 2D-Bildes anhand gegebener Tiefeninformationen mittels Sliced-Image-Rendering: (a+b+c) Slices ohne und (d) mit Darstellungsoptimierung.



(a) (b)

Abbildung 5.39: Dreidimensionale Darstellung des Kamerabildes mithilfe des rekonstruierten abgebildeten Szenenausschnittes (Nebenprodukt des Geometry-Depth-Matching-Verfahrens).

muss die Bilderweltenanwendung zwingend das Handling von 3D-Geometrien unterstützen.

5.4.4 Zusammenfassung und Diskussion

In diesem Absatz wurden zwei verschiedene Renderverfahren zur Erreichung einer authentischen Darstellung und Verdeckung augmentierter Bilderwelten entwickelt. Das *Sliced-Image-Rendering*-Verfahren rendert (ohne 3D-Geometrie) ein 2D-Bild anhand gegebener Tiefeninformationen als dreidimensionale Darstellung und ermöglicht auf diese Weise eine authentische Verdeckung auch bei freier Navigation. Auf Basis gewonnener Tiefeninformationen wird ein Bild innerhalb der Bilderweltenszene in Slices zerlegt. Ein Slice repräsentiert eine Bildregion des Originalbildes und wird an die entsprechende Tiefenposition verschoben. Die Gesamtheit aller verschobenen, skalierten Regionen wird Sliced-Image genannt.

Das *Image-Geometry-Rendering*-Verfahren verwendet die im Rahmen des Geometry-Depth-Matching-Verfahrens rekonstruierte Szenengeometrie des in einem Kamerabild abgebildeten Szenenausschnittes und texturiert diese mittels perspektivischer Projektion des Kamerabildes. Somit wird eine 3D-Geometrie vorausgesetzt und verwendet.

Verfahren	Vorteile	Nachteile
Sliced-Image-Rendering	- Authentische Verdeckung auch bei freier Navigation - Keine Szenengeometrie notwendig - Echtzeitfähig	- Fehlerhafte oder ungenaue Depth-Maps führen zur Abnahme der Darstellungsqualität - Nicht alle schwarzen Bereiche zwischen Slices können beseitigt werden
Image-Geometry-Rendering	- Geringerer Renderaufwand als beim *Sliced-Image-Rendering*	- Voraussetzung der Szenengeometrie des im Kamerabild abgebildeten Szenenausschnittes - Voraussetzung der Unterstützung des Handlings von 3D-Geometrien durch die Bilderweltenanwendung - Anforderung einer authentischen Verdeckung ohne Geometrie nicht erfüllt

Tabelle 5.5: Vor- und Nachteile der Renderverfahren für eine authentische Darstellung und Verdeckung augmentierter Bilderwelten.

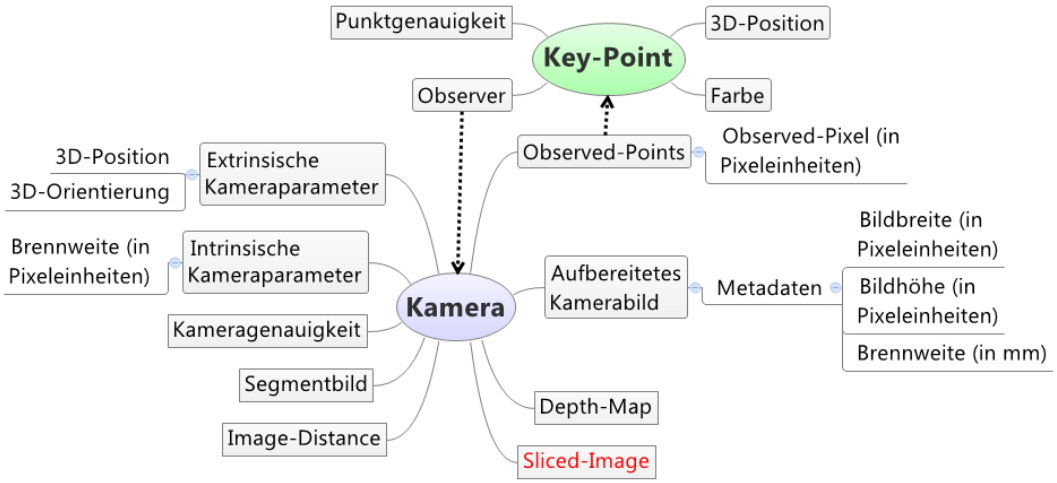


Abbildung 5.40: Ausgabedaten nach den Renderverfahren für eine *Authentische Darstellung der Bilderwelten*. Neue Daten sind rot beschriftet.

Bei der Erläuterung der einzelnen Verfahren wurden jeweils die Vor- und Nachteile aufgezeigt. Diese werden in *Tab. 5.5* gegenübergestellt. Eine Evaluierung der Renderverfahren findet sich in Absatz *7.3* (*Evaluierung der authentischen Darstellung der Bilderwelten*). *Abb. 5.40* zeigt die Ausgabedaten nach der Tiefenzuordnung.

5.5 Authentische Darstellung der virtuellen Objekte

5.5.1 Überblick

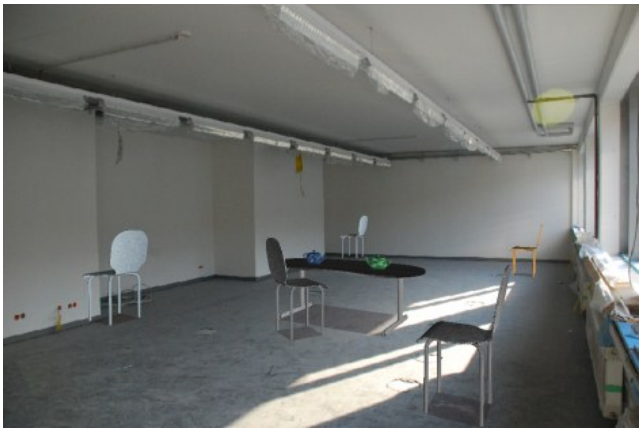
In diesem Absatz werden Ansätze und Arbeiten des Autors (zum Teil als Co-Autor) vorgestellt, die die Augmentierungsschwerpunkte *Transformationsanpassung*, *Render-Adaption* und *Bildbasierte Beleuchtung* abdecken. Da dies nicht der Kern dieser Arbeit darstellt, wird lediglich ein Überblick gegeben.

5.5.2 Transformationsanpassung virtueller Objekte

Ein Problem bei der Integration virtueller Objekte in die Bilderweltenszene stellen geometrische Gegebenheiten der Objekte dar. Für einen authentischen Eindruck müssen Größe (Skalierung), Position und Ausrichtung der virtuellen

(a)

(b)



(c)

Abbildung 5.41: Perspektivisch korrektes Einbetten virtueller 3D-Objekte in einem Foto: Aus dem (a) Originalbild wurden (b) Raumkanten und der Hauptfluchtpunkt ermittelt und (c) damit die virtuelle Kamera für die einzubettenden 3D-Objekte kalibriert, sowie die 3D-Objekte auf der Bodenfläche positioniert [NBKG09].

Objekte an den Maßstab der Bilderweltenszene angepasst sein. In [NBKG09] wurde vom Autor unter Mitarbeit von Co-Autoren ein Verfahren veröffentlicht, womit virtuelle 3D-Objekte in Fotos perspektivisch korrekt eingebettet werden. Hierbei wird die virtuelle Kamera der 3D-Objekte entsprechend der Perspektive des Eingabefotos kalibriert. Dies erfolgt durch die Bestimmung des *Hauptfluchtpunktes* (Durchstoßpunkt des Hauptsehstrahls im Bild) von senkrecht zur Bildebene verlaufenden Kanten des Fotos. Zur Positionierung der virtuellen Objekte wird die Bodenfläche der im Foto abgebildeten Raumstruktur rekonstruiert.

5.5.3 Render-Adaption virtueller Objekte

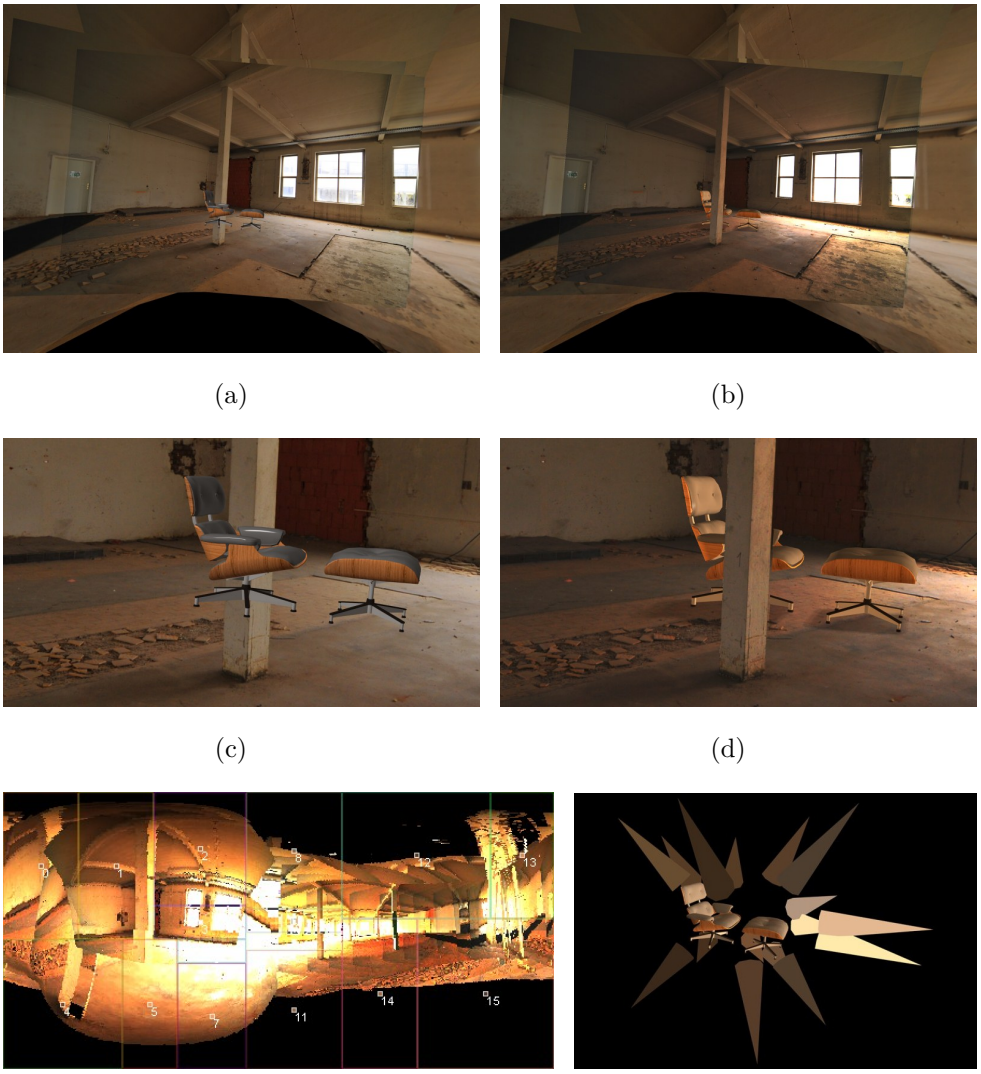
Für eine authentische Integration müssen neben geometrischen Gegebenheiten auch Darstellungsanpassungen zwischen Bildern und virtuellen Objekten vorgenommen werden. Geringe Auflösung, Über-/Unterbelichtungen und Verwacklungen/Unschärfe der angepassten Fotos können bei der Integration hochaufgelöster virtueller 3D-Objekte zu visuellen Störungen führen. Daher ist für eine authentische Verschmelzung das Anpassen eventueller Qualitätsunterschiede zwischen virtuellen und realen Anteilen notwendig.

In [BGKN09] wurde unter Mitarbeit des Autors ein Verfahren vorgestellt, welches an dieser Stelle zum Einsatz kommt. Hierbei werden die Fotos für einen homogenen Gesamteindruck aufbereitet (siehe Absatz *5.2.5*) und anschließend Auflösung und Darstellungsqualität der Bildinhalte analysiert und das Rendering der eingebetteten virtuellen Objekte daran angepasst.

5.5.4 Bildbasierte Beleuchtung virtueller Objekte

Zusätzlich zu der *Render-Adaption* müssen die virtuellen Objekte mit einer zu der Bilderweltenszene konformen Beleuchtung dargestellt werden. Die abgebildete reale Umgebung der Bilder beinhalten eigene Lichtquellen. Dementsprechend müssen für eine authentische Augmentierung die virtuellen Objekte eine zu den Bildinhalten passende Beleuchtungssituation aufweisen. Dabei ist auch ein zu der Szene passender Schattenwurf eingeschlossen.

In [NKG11d] wurde vom Autor unter Zusammenarbeit mit Co-Autoren ein neuartiges *Image-Based-Lighting*-Verfahren aufbauend auf dem *Sliced-Image-*



(a) (b)

(c) (d)

(e) (f)

Abbildung 5.42: Image-Based-Lighting mittels Sliced-Images als Light-Probes [NKG11d]: (a+c) Ausgangssituation, keine korrekte Verdeckung, keine bildbasierte Beleuchtung. (b+d) Sliced-Image-Rendering und das Image-Based-Lighting-Verfahren führen zu authentischer Verdeckung und Beleuchtung. (e) SEM als Zylinder-Projektion mit 16 extrahierten Lichtquellen. (f) Generierte direktionale Lichtquellen (mit Lichtfarbe und -stärke) des virtuellen Objektes.

Rendering veröffentlicht. Anstelle vollständig ausgemessener Umgebungsbeleuchtung (etwa in Form von Light-Probes [Deb98], [Deb05]) werden die Slices des Sliced-Image-Renderings als Eingabedaten verwendet. Daraus wird eine *Synthetisierte Environment-Map* (*SEM*) konstruiert. Durch die bekannten Kameraparameter werden die Fotos als räumlicher Ausschnitt einer Light-Probe interpretiert. Alle gefundenen Ausschnitte werden unter Verwendung einer Gewichtungsfunktion in die synthetisierte Environment-Map akkumuliert. Zusätzlich wird für Ausschnitte mit redundanter Abdeckung der dynamische Bereich der Lichtintensitäten erweitert. Aus der SEM werden 16 Lichtquellen sowie ihre Lichtstärke und Lichtfarbe ermittelt. Mithilfe der extrahierten Lichtquellen werden die virtuellen Objekte (entsprechend der in den Fotos abgebildeten Beleuchtungssituation) beleuchtet und ein fotorealistischer Schatten erzeugt. *Abb. 5.42* zeigt das Anwenden des Verfahrens sowie einzelne Teilschritte.

5.5.5 Zusammenfassung und Diskussion

Dieser Teilschritt hat einen Überblick über Konzepte und Verfahren des Autors (zum Teil als Co-Autor) gegeben, die sich (etwas abseits vom Kern dieser Arbeit) mit den Augmentierungsschwerpunkten *Transformationsanpassung*, *Render-Adaption* und *Bildbasierte Beleuchtung* befassen.

5.6 Zusammenfassung und Diskussion

In dem vorliegenden Konzept wurden Lösungen und Verfahren für das Erreichen des Hauptziels und der Teilziele entwickelt, unter Beachtung der in Absatz *4.3* zusammengetragenen Anforderungen. Hierzu wurde, orientiert an den Augmentierungsschwerpunkten (siehe *Abb. 1.5*), das Konzept in vier Teilschritte unterteilt (siehe *Abb. 1.12*).

Der erste Teilschritt bereitet die Rohdaten auf sowie den Aufbau der Bilderwelt für eine authentische Augmentierung vor. Hierbei liegt das Hauptaugenmerk auf die qualitative Darstellung der Bilderwelten (in Bezug auf Qualität der Verlässlichkeit rekonstruierter Kameras und Key-Points sowie die Darstellungsqualität der Fotos). Als Ergebnis liegt eine visuell homogene Bilderweltenszene vor, deren Image-Planes zielgerichtet für eine Integration virtueller Objekte in die Szene positioniert wurden. Der Kern dieser Arbeit bezieht sich auf

den Augmentierungsschwerpunkt *Verdeckung*. Deshalb bilden die Teilschritte *Teilrekonstruktion der Bilderweltenszene* und *Authentische Darstellung der Bilderwelten* den Hauptteil und den Kern dieses Konzeptes. Für eine authentische Verdeckung werden bei der *Teilrekonstruktion der Bilderweltenszene* Tiefeninformationen der abgebildeten Szenenausschnitte der Fotos generiert. Hierfür wurde vom Autor das *Segment-Depth-Matching*-Verfahren entwickelt. Dieses Verfahren zerlegt eine abgebildete Szene in Segmente. Die Segmente repräsentieren reale abgebildete Objekte der Fotos. Die Observed-Points eines Fotos werden verwendet, um den Segmenten eine Tiefe (*Z-Distanz*) zuzuordnen. Auf diese Weise entsteht eine segmentbasierte, also flächengenaue Depth-Map. Da die Segmentierung unter Umständen zu Fehlern führen kann (siehe *Tab. 5.3*), wurde vom Autor ein weiteres Depth-Matching-Verfahren entwickelt. *Key-Point-Depth-Matching*-Verfahren benötigt vor der Tiefenzuordnung keinen vorausgehenden Segmentierungsschritt. Die Observed-Points werden direkt als kreisförmige Sprites auf die Bildebene projiziert und enthalten als Pixelfüllwerte die notwendige Tiefe (*Z-Distanz*). Zwar ist dieses Verfahren sehr effizient, jedoch setzt es eine flächendeckende Punktwolke voraus. Bereiche im Bild ohne Observed-Points erhalten keine Tiefeninformationen. Somit enthält die Depth-Map unter Umständen große Bereiche ohne Tiefeninformationen. Des Weiteren stellen die Sprites wieder Flächen dar. Dementsprechend ist die Depth-Map, wie bei dem Segment-Depth-Matching-Verfahren flächen-, aber nicht pixelgenau. Aufgrund dieser Nachteile wurde vom Autor das *Geometry-Depth-Matching*-Verfahren entwickelt. Dieses liefert eine akkurate, pixelgenaue Depth-Map. Hierfür wird ein semiautomatischer Segmentskizzierungsschritt vorausgesetzt. Die 2D-Eckpunkte eines Segmentes werden mithilfe der (dem Segment) zugeordneten Observed-Points als neue Key-Points rekonstruiert. Diese neu erzeugten Key-Points werden trianguliert und es entsteht eine Szenengeometrie des abgebildeten Szenenausschnittes. Mithilfe der gegebenen Szenengeometrie werden akkurate, pixelgenaue Depth-Maps generiert. Eine Gegenüberstellung der drei entwickelten Depth-Matching-Verfahren findet sich in *Tab. 5.4*.

Aufbauend auf den gewonnenen Tiefeninformationen wurde im Rahmen des Teilschrittes *Authentische Darstellung der Bilderwelten* das Verfahren des *Sliced-Image-Renderings* veröffentlicht. Das Sliced-Image-Rendering stellt ein *Image-Based-Rendering*-Verfahren dar, das ohne 3D-Geometrie ein Bild an-

hand gegebener Tiefeninformationen als dreidimensionale Darstellung rendern und auf diese Weise eine authentische Verdeckung auch bei freier Navigation ermöglichen kann. Auf Basis der Tiefeninformationen wird ein Bild innerhalb der Bilderweltenszene in Slices zerlegt. Ein Slice repräsentiert eine Bildregion des Originalbildes. Jedes Slice wird an die für die Region entsprechende Tiefenposition, die in der Depth-Map gespeichert wurde, verschoben. Die Gesamtheit aller Slices ergibt aus Sicht der Kamera das Kamerabild. Da jedes Slice als ein eigenes Bild gerendert werden muss, ist der Darstellungsaufwand eines Sliced-Images gegenüber einem einzelnen Foto so viel mal höher wie die Anzahl der Slices.

Für eine weniger renderaufwendige Darstellung wurde neben dem Sliced-Image-Rendering vom Autor das *Image-Geometry-Rendering*-Verfahren vorgestellt. Dieses verwendet die erstellte 3D-Geometrie des *Geometry-Depth-Matching*-Verfahren für das Darstellen des Kamerabildes als eine *3D-Bildprojektion*. Hierzu wird die Geometrie mittels perspektivischer Projektion des Kamerabildes texturiert. Dieses Verfahren benötigt zwar keine vollständige 3D-Rekonstruktion der gesamten Bilderweltenszene, jedoch setzt es eine 3D-Geometrie des im Foto abgebildeten Szenenausschnittes voraus. Zwar ist der Renderaufwand geringer als bei Sliced-Images, jedoch erfüllt die Voraussetzung von Szenengeometrie nicht alle an das Konzept gestellten Anforderungen.

Der Teilschritt *Authentische Darstellung der virtuellen Objekte* stellt nicht Bestandteil des Kerns dieser Arbeit dar. Dennoch muss er für eine authentische Augmentierung und folglich auch für eine authentische Verdeckung berücksichtigt werden. Im Rahmen dieses Teilschrittes wurde vom Autor ein Verfahren für die *Transformationsanpassung virtueller Objekte* in Fotos entwickelt. Weiterführend wurde unter Mitarbeit als Co-Autor Verfahren für die *Render-Adaption virtueller Objekte* und *Bildbasierte Beleuchtung virtueller Objekte* entwickelt.

Durch die Summe der Teilschritte ermöglicht das Konzept eine authentische Verdeckung integrierter virtueller Objekte in Bilderwelten bei Kameranavigation als auch bei freier Navigation.

Kapitel 6

Technische Umsetzung der Konzeption

6.1 Überblick

In diesem Kapitel wird die technische Umsetzung der in der Konzeption erörterten Verfahren der Teilschritte für die authentische Verdeckung virtueller 3D-Objekte in Bilderwelten beschrieben. Die Teilschritte *Teilrekonstruktion der Bilderweltenszene* und *Authentische Darstellung der Bilderwelten* stellen den Kern dieser Arbeit dar. Deshalb wurden vom Autor während der Implementierungsphase das Hauptaugenmerk auf die Verfahren dieser beiden Teilschritte gelegt und weiterführende Verfahren der anderen Teilschritte zum Teil vernachlässigt oder als gegeben vorausgesetzt. Dementsprechend werden in diesem Kapitel jene weiterführenden Verfahren nicht tiefer erläutert und auf die entsprechenden Veröffentlichungen verwiesen.

Die technische Umsetzung dieser Arbeit erfolgte im Rahmen des Drittmittel-Projektes *„PoP-EYE – Entwicklung von Methoden und Werkzeugen zur virtuellen Produktpräsentation in 3D Community Photo Collections“* der Förderlinie „IngenieurNachwuchs"2009 des Bundesministeriums für Bildung und Forschung (BMBF), Projekt-Nummer 17N0909. Hierzu entstand die Komponenten-basierte Software *PoP-EYE* [NKG12]. Die PoP-EYE-Software ist eine auf Bilderwelten-basierende AR-Umgebung, die eine einfache und schnelle Integration virtueller 3D-Objekte erlaubt mit dem Hintergrund 3D-Planungsvisualisierungen und Produktpräsentationen in einer fotobasierten 3D-Welt zu ermöglichen. Hierbei steht

eine authentische Darstellung der Planungsvisualisierung im Vordergrund. Alle konzeptionellen Umsetzungen der vorliegenden Arbeit flossen als Komponenten in diese Software ein.

Für den Aufbau der PoP-EYE-Software wurde das *IP3D-Framework* verwendet. Das IP3D-Framework wurde unter Mitarbeit des Autors entwickelt und veröffentlicht [GNA10]. Es ermöglicht das einfache und effiziente Erstellen von Bilderweltenanwendungen mit dem Ziel diese augmentieren zu können. Dementsprechend stellt das IP3D-Framework den zugrunde liegenden Kern von PoP-EYE dar. IP3D sowie PoP-EYE wurden entwickelt mit dem XNA-Framwork 4.0 [Mic10b] als 3D-Grafikframework und der .NET Programmiersprache C# [Mic10a].

6.2 Umsetzung der Datenaufbereitung

6.2.1 Überblick

Dieser Absatz beschreibt die technische Umsetzung des konzeptionellen Absatzes *5.2* (*Datenaufbereitung*).

6.2.2 Umsetzung der Bereitstellung der Rohdaten

Als *Structure-from-Motion*-Algorithmus wurde bei der Umsetzung dieser Arbeit die Open-Source-Software *Bundler* [Sna10a] verwendet. Der genaue Arbeitsablauf von Bundler findet sich in Absatz *2.6.3* (*Generieren einer Bilderwelt*). Als Ergebnis liegen die berechneten Rohdaten einer Bilderwelt in Form von Textdateien vor. Diese beinhalten Transformationsmatrizen der Kameras sowie die 3D-Positionen und Farbwerte der *Key-Points*. Des Weiteren wird eine Zuordnung von *Observed-Points* und *Observern* angegeben.

6.2.3 Umsetzung der Aufbereitung der Kameradaten

Es kann zu Ungenauigkeiten bei den Ergebnissen der Kamerarekonstruktion eines Structure-from-Motion-Algorithmus kommen. Dies ist zurückzuführen auf das Prinzip der *Bündelblockausgleichung*. Deshalb muss die *Kameragenauigkeit* (siehe Absatz *5.2.3*) der berechneten Kameraparameter ermittelt

werden. Diese wird in Absatz *6.2.6* (*Umsetzung des Aufbaus der Bilderweltenszene*) benötigt. Durch den SfM-Algorithmus wurden den Kameras eine unterschiedliche Anzahl von Observed-Points zugeordnet. Die höchste Zahl an Observed-Points einer Kamera innerhalb der Bilderwelt definiert die höchste Kameragenauigkeit. Folglich wird nicht die Gesamtzahl der Key-Points als Maß für die höchstmögliche Kameragenauigkeit verwendet, sondern die höchste Zahl an Observed-Points (aus allen Kameras). Folgender Pseudocode zeigt die Berechnung der Kameragenauigkeiten:

```
// biggest number of Observed-Points
int maxNumber = 0;

foreach (Camera c in Cameras) {
  maxNumber = Max(maxNumber, c.ObservedPoints.Count)
     ;
  }

foreach (Camera c in Cameras) {
  // Calculate the camera accuracy
  c.ca = c.ObersverdPoints.Count / maxNumber;
  }
```

Programmcode 6.1: Berechnung der Kameragenauigkeit

6.2.4 Umsetzung der Aufbereitung der Daten der Punktwolke

Analog zu der Kameragenauigkeiten müssen ebenfalls die Key-Points bewertet werden. Das Fehlerrisiko wird um so geringer, je mehr korrespondierende Feature-Points der Kamerabilder ein Key-Point besitzt. Das bedeutet, je öfter der Key-Point als Observed-Point einer Kamera fungiert, desto genauer und korrekter kann er interpretiert werden. Folglich bestimmt die Anzahl der Observer (Kameras, die den Key-Point als Observed-Point zugeordnet haben) eines Key-Points die (relative) *Punktgenauigkeit*. Folgender Pseudocode zeigt die Berechnung der (relativen) Punktgenauigkeiten:

```
// biggest number of Observer
int maxNumber = 0;
foreach (KeyPoint kp in KeyPoints) {
  maxNumber = Max(maxNumber, kp.Observer.Count);
  }
foreach (KeyPoint kp in KeyPoints) {
  // Calculate the point accuracy
  kp.pa = kp.Observer.Count / maxNumber;
  }
```

Programmcode 6.2: Berechnung der (relativen) Punktgenauigkeit

6.2.5 Umsetzung der Aufbereitung des Bildmaterials

Das Aufbereiten des Bildmaterials ist ein zwingender Schritt, um aus Fotos innerhalb einer Bilderwelt eine visuell homogene Szene darzustellen. Bildfehler, Auflösungsdifferenzen, Über-/Unterbelichtungen und Verwacklungen/Unschärfe und qualitativ unterschiedliche Bilder führen zu visuellen Störungen innerhalb der Bilderwelt.

Da diese Thematik nicht den Kern dieser Arbeit betrifft, wird innerhalb der technischen Umsetzung eine Lösung dieses Problems als gegeben betrachtet. In [BGKN09] wurde unter Mitarbeit des Autors ein Verfahren vorgestellt und als Komponente für PoP-EYE umgesetzt, welches an dieser Stelle zum Einsatz kommt.

6.2.6 Umsetzung des Aufbaus der Bilderweltenszene

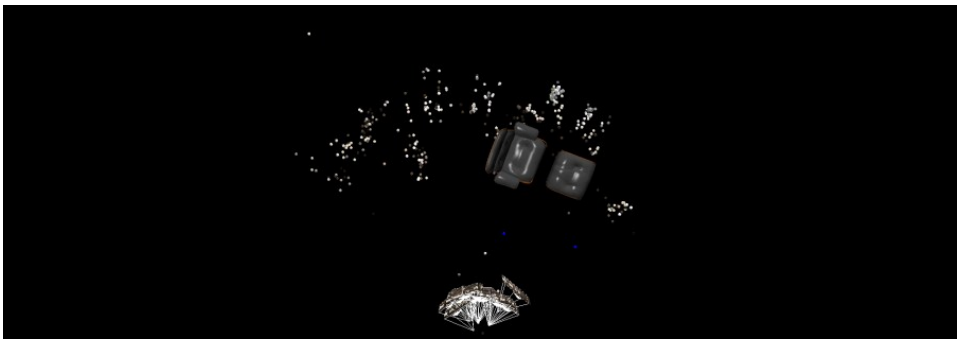
Mithilfe der berechneten Kameragenauigkeiten werden sehr ungenaue Kameras aus der Bilderwelt aussortiert. Als praktikablen Schwellwert für die Software PoP-EYE hat sich $T_{Cam} = 50\%$ bewährt. Bei sehr kleinen Szenen (wenig Fotos bzw. wenig rekonstruierte Kameras) ist es sinnvoll, diesen Wert nach unten zu korrigieren, um eine größere Flächenabdeckung mit Bildern innerhalb der Szene zu erreichen.

Für die *Image-Distance*-Berechnung und die damit verbundene *Image-Plane*-Positionierung wurden verschiedene Varianten umgesetzt. Die aus Projekten wie

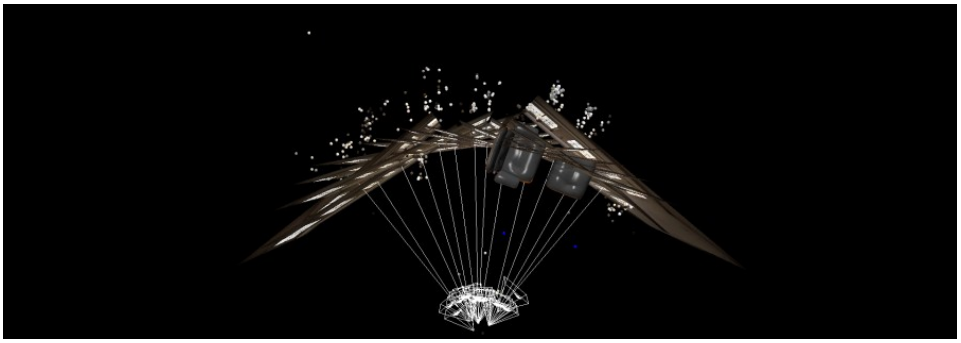
PhotoTourism [SSS06a] Positionierung platziert die Image-Planes unmittelbar vor die Kameras. Diese Umsetzung eignet sich nicht für die Integration virtueller 3D-Objekte, da (fast) kein Platz zwischen Kamera und Bildebene zur Verfügung steht. Eine weitere Variante nutzt die durchschnittliche Distanz der Observed-Points zur Kamera als Image-Distance. Das bewirkt zwar, dass Platz zwischen Image-Plane und Kamera zur Verfügung steht, jedoch befinden sich weiterhin etwa die Hälfte der Observed-Points hinter der Image-Plane. Das bedeutet, die 3D-Szene wird weiterhin zum Teil verdeckt. Somit kann nicht die gesamte Szene für eine Augmentierung genutzt werden. Eine dritte Positionierungsmöglichkeit stellt das ausschließliche Verwenden der weit entferntesten Observed-Points für die Distanzberechnung. Um Ausreißer zu vermeiden, werden 10% der weit entferntesten Observed-Points ermittelt und mithilfe des Median die Image-Distance berechnet. Die Image-Planes werden so weit von den Kameras entfernt positioniert, dass sich (fast) alle Observed-Points zwischen ihrer Kamera und der dazugehörigen Image-Plane befinden. Folgerichtig ergibt sich somit ein großer Bereich, der für die Integration von virtuellen Objekten genutzt werden kann. *Abb. 6.1* stellt alle drei Varianten gegenüber und zeigt, dass die letzte Variante für eine Augmentierung am geeignetsten ist.

6.2.7 Zusammenfassung und Diskussion

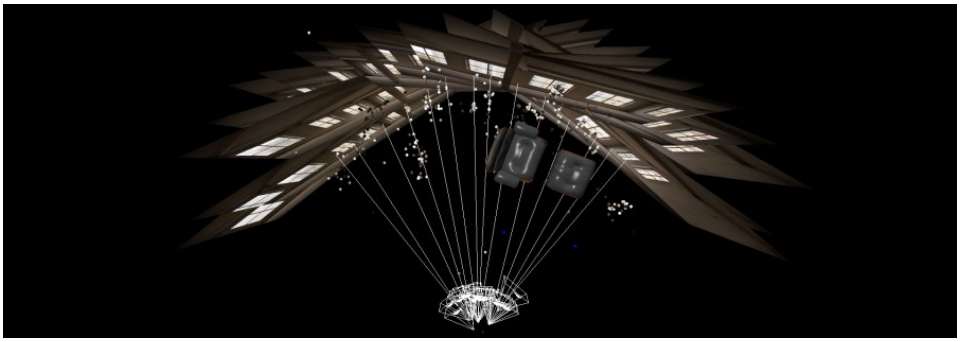
In diesem Absatz wurde beschrieben, wie die Rohdaten der *Structure-from-Motion*-Software *Bundler* und die Fotos innerhalb der PoP-EYE-Software verarbeitet und aufbereitet werden. Im Rahmen der Datenaufbereitung werden (relative) *Kameragenauigkeiten* und (relative) *Punktgenauigkeiten* berechnet. Für den Aufbau der Bilderweltenszene dient die (relative) Kameragenauigkeit als Ausschlusskriterium für ungenaue und fehlerhaft rekonstruierte Kameraparameter. Die Positionierung der Image-Planes wurde so umgesetzt, dass möglichst viel Raum zwischen Kamera und Image-Plane für die Integration virtueller Objekte zur Verfügung steht.



(a) Positionierung der Image-Plane unmittelbar vor die Kamera



(b) Durchschnittliche Z-Distanz aller Observed-Points als Image-Distance



(c) 10% der weit entferntesten Observed-Points

(d) Abb. (c) aus der Frontalansicht

Abbildung 6.1: Varianten zur Berechnung der Image-Distance.

6.3 Umsetzung der Teilrekonstruktion der Bilderweltenszene

6.3.1 Überblick

Dieser Absatz erläutert die Implementierungsdetails der technischen Umsetzung des konzeptionellen Absatzes *5.3* (*Teilrekonstruktion der Bilderweltenszene*). Im Rahmen dessen wird die *Umsetzung der Bildsegmentierung* sowie die *Umsetzung der Tiefenzuordnung* in diesem Absatz erläutert.

6.3.2 Umsetzung der Bildsegmentierung

Die Implementierung der in Absatz *5.3.2* (*Bildsegmentierung*) beschriebenen Segmentierungsverfahren erfolgte in dem vom Autor (unter Mitarbeit weiterer Co-Autoren) entwickelten Bildverarbeitungsframework *SBIP* [NFST10] [NKG11a] [NB12]. Analog zu PoP-EYE wurde für die Implementierung von SBIP das XNA-Framework 4.0 [Mic10b] als 3D-Grafikframework und der .NET Programmiersprache C# [Mic10a] verwendet. Für die Farbsegmentierung wurde ein Bildverarbeitungsfilter auf Basis des *LUV*-Farbraumes umgesetzt. Auf eine detaillierte Beschreibung der Umsetzung des Filters wird an dieser Stelle verzichtet. Eine umfangreiche mathematische Beschreibung für die Implementierung von Konvertierungen von RGB-Farbwerten zu LUV-Farbwerten und zurück findet sich in [Lin07].

Folgender Programmcode zeigt die Verwendung des Filters:

```
1 using SBIP.Filter.NonSBIP;
2 ...
3 NSLuvColorSegmentation segmentation = new
     NSLuvColorSegmentation();
4 segmentation.ColorDistance = 2.5f;
5 Image segmentImage = segmentation.Apply(image);
```

Programmcode 6.3: Verwenden der LUV-basierten Farbsegmentierung des SBIP-Frameworks [NB12]

Für die *Konturbasierte Segmentierung* wurde im Konzept das Verfahren *ConGrap* [NKG+11b] erläutert. Im Rahmen der Umsetzung von ConGrap wurde

für die Gradient-basierte Kantendetektion der *Canny-Algorithmus* (siehe Absatz *2.3.2* (*Kanten- und Feature-Point-Detektion*)) der Bibliothek *AForge.NET* [Kir10] erweitert. ConGrap selbst ist Teil des SBIP-Frameworks. Folgender Programmcode zeigt die Verwendung:

```
1 using SBIP.Filter.NonSBIP;
2 ...
3 ConGrap conGrap = new ConGrap();
4 conGrap.HighThreshold = 10;
5 conGrap.LowThreshold = 5;
6 conGrap.TraceRadius = 3;
7 conGrap.Weight = 3;
8 Image segmentImage = conGrap.Apply(image);
```

Programmcode 6.4: Verwenden von ConGrap als konturbasierte Segmentierung des SBIP-Frameworks [NB12]

Für die *Objektbasierte Segmentierung* wurde vom Autor ein .NET *Windows-Forms-Control* implementiert, das dem Nutzer per Maussteuerung erlaubt, in ein abgebildetes Foto Segmente einzuzeichnen. Hierfür wird ein Polygon-Linienzug erstellt, indem per Mausklick ein neuer Eckpunkt des Polygons definiert wird. *Abb. 6.2* zeigt das Windows-Forms-Control für die semiautomatische Segmentskizzierung der objektbasierten Segmentierung.

6.3.3 Umsetzung der Tiefenzuordnung

In Absatz *5.3.3* (*Tiefenzuordnung*) wurden drei Verfahren konzeptionell entwickelt. Die Umsetzung dieser Verfahren erfolgte als Komponenten für die *PoP-EYE*-Software.

Das *Segment-Depth-Matching*-Verfahren setzt ein Segmentbild voraus. In dieses Segmentbild werden die Observed-Points des dazugehörigen Kamerabildes projiziert mithilfe der *Formel 5.2*. Ausgehend von den 2D-Bildpositionen der projizierten Observed-Points (*Observed-Pixel*) erfolgt ein *Flood-Fill*-Algorithmus. Als Füllwert wird, wie im Konzept beschrieben, die Z-Distanz des Observed-Points verwendet. Das Ergebnis des Flood-Fill-Mechanismus kann als vereinfachtes Voronoidiagramm gesehen werden. Als Abbruchbedingungen für den Flood-Fill-Vorgang dienen das Erreichen einer anderen Flood-Fill-Zelle

Abbildung 6.2: Ein .NET *Windows-Forms-Control* für die semiautomatische Segmentskizzierung der objektbasierten Segmentierung.

oder die Konturgrenze eines Segmentes. Die Z-Distanzen entsprechen Gleitkommazahlen. Deshalb werden *Float-Depth-Maps* verwendet. Dafür dient die *Texture2D*-Klasse des XNA-Frameworks. Der Flood-Fill-Algorithmus ist (analog zu den Segmentierungsverfahren) als Bildfilter im SBIP-Framework umgesetzt:

```
1  using IP3D.Framework.Utils;
2  using Microsoft.Xna.Framework.Graphics;
3  using SBIP.Filter.NonSBIP;
4  ...
5  // flood filling with key points
6  NSKeyPointBasedFloodfill floodfill =
7      new NSKeyPointBasedFloodfill();
8  floodfill.KeyPoints = ObservedPoints;
9  Image depthMap = floodfill.Apply(image);
10 Texture2D floatDepthMap = TextureConverter.
     RGBToFloat(depthMap, imageDistance);
```

Programmcode 6.5: Verwenden des Key-Point-basierten Flood-Fill-Algorithmus des SBIP-Frameworks [NB12]

Im Rahmen der Umsetzung des *Key-Point-Depth-Matching*-Verfahrens wer-

den die Observed-Points nach ihrer Z-Distanz absteigend sortiert und anschließend als 2D-Sprites in das Bild projiziert. Die Z-Distanz stellt wieder den Füllwert der Sprites dar. Durch die absteigende Sortierung werden weiter entfernte Punkte zuerst als Sprites gezeichnet und können näher liegende Punkte bzw. deren Sprites nicht überschreiben. Somit wird erreicht, dass der nähere Bereich besser aufgelöst wird und Vordergrundobjekte nicht durch Hintergrundobjekte der abgebildeten Szene fälschlich überlagert werden. Für das Zeichnen kreisförmiger Sprites in eine Textur werden die Klassen *Graphics*, *SolidBrush* und *Image* der Bibliothek *System.Drawing* des .NET-Frameworks verwendet, wie im folgenden Code-Ausschnitt gezeigt wird:

```
1  using System.Drawing;
2  ...
3  int r = 5; // radius of 5 pixels
4  Graphics g = Graphics.FromImage(image);
5  // drawing sprite for every observed pixel
6  foreach (ObservedPixel op in ObservedPixels)
7  {
8      SolidBrush brush = new SolidBrush(
9          Color.FromArgb(255, op.ZDistance, op.
               ZDistance, op.ZDistance));
10     g.FillEllipse(brush, op.X - r / 2, op.Y - r / 2,
           r, r);
11 }
```

Programmcode 6.6: Zeichnen eines kreisförmigen 2D-Sprites mit einem definierten Radius.

Im Rahmen der Umsetzung des *Geometry-Depth-Matching*-Verfahrens werden aus den gegebenen Eckpunkten der *Objektbasierten Segmentierung* so genannte *Eckpunkt-Tripel* erstellt. Hierzu wird mithilfe der Bibliothek *System.Drawing* des .NET-Frameworks entlang der Pixellinie (bzw. Strecke) zwischen zwei Eckpunkten die Strecke eines Eckpunktes A mit jedem anderen Eckpunkt des gleichen Segmentes abgetastet. Befindet sich eine Strecke zwischen zwei Eckpunkten innerhalb des Segmentes (oder auf der Konturgrenze), wird (ausgehend vom zweiten Eckpunkt B) ebenfalls die Strecke zu allen weite-

ren Eckpunkten des Segmentes abgetastet. Sobald auf diese Weise ein dritter Eckpunkt C detektiert wurde, wird die Strecke von C zu A abgetastet. Befindet sich diese Strecke wieder innerhalb des Segmentes (oder auf der Konturgrenze), ist ein Tripel, bestehend aus den Eckpunkten $\{A, B, C\}$, gefunden wurden. Aus den Eckpunkten der Tripel werden, entsprechend der mathematischen Vorschrift aus *Formel 5.6* und *Formel 5.7*, neue Key-Points erzeugt. Es entstehen 3D-Triangles. Die 3D-Triangles erhalten durch die mathematische Vorschrift *Formel 5.11*, eine Orientierung (*Normale*). Somit kann aus den erstellten 3D-Triangles eines 2D-Segmentes ein 3D-Mesh und folglich die Szenengeometrie des abgebildeten Szenenausschnittes erzeugt werden. Für das Generieren einer Depth-Map aus gegebener Szenengeometrie wird *Shader*-basiert die Z-Distanzen der einzelnen *Vertices* über den *Pixelshader* pixelgenau in eine Textur geschrieben. Hierfür wird die *Projektionsmatrix* auf die exakte Größe der Image-Plane des Kamerabildes gesetzt:

```
1  using Microsoft.Xna.Framework;
2  ...
3  // calculate aspect ratio
4  float ratio = imageSize.X / imageSize.Y;
5
6  // calculate full field of view
7  float fieldOfView = (float)(Math.Atan(imageSize.X *
       0.5 / (imageDistance * ratio)) * 2.0);
8
9  // Calculate projection matrix
10 Matrix projection = Matrix.
       CreatePerspectiveFieldOfView(
11     fieldOfView,
12     ratio,
13     nearClippinPlane,
14     farClippinPlane);
```

Programmcode 6.7: Setzen der Projektionsmatrix auf die exakte Größe der Image-Plane des Kamerabildes.

6.3.4 Zusammenfassung und Diskussion

In diesem Absatz wurde die Umsetzung des konzeptionellen Absatzes *5.3* (*Teilrekonstruktion der Bilderweltenszene*) erläutert. Entsprechend der Konzeption wurden folgende Verfahren der *Bildsegmentierung* umgesetzt:

- *Farbbasierte Segmentierung*,
- *Konturbasierte Segmentierung*,
- *Farb- und Konturbasierte Segmentierung*,
- *Objektbasierte Segmentierung*.

Aufbauend auf den Segmentierungsergebnissen erfolgte die Umsetzung der *Depth-Matching*-Verfahren im Rahmen der *Tiefenzuordnung*. Hierfür wurden folgende Verfahren implementiert:

- *Segment-Depth-Matching*,
- *Key-Point-Depth-Matching*,
- *Geometry-Depth-Matching*.

6.4 Umsetzung der authentischen Darstellung der Bilderwelten

6.4.1 Überblick

Dieser Absatz erläutert die Implementierungsdetails der technischen Umsetzung des konzeptionellen Absatzes *5.4* (*Authentische Darstellung der Bilderwelten*). Hierzu wurde im Konzept das *Sliced-Image-Rendering* entwickelt. Weiterführend wurde im Konzept das *Image-Geometry-Rendering* als Alternative zu Sliced-Images erklärt. Die Umsetzung beider Renderverfahren wird im folgenden Text erläutert.

6.4.2 Umsetzung der flächenbasierten Darstellung

Für die Umsetzung des Sliced-Image-Rendering wird ein Kamerabild in eine Anzahl an Slices zerlegt. Dabei stellt jedes Slice ein eigenes Bild bzw. eine Image-Plane dar, die nur einen entsprechenden Bildanteil darstellt. Die Slices sind an die in der Depth-Map kodierte Z-Distanz verschoben und perspektivisch korrekt projiziert. Die gegebene Image-Distance des Kamerabildes dient als End-Slice, d. h. alle weiteren Slices befinden sich aus Sicht der Kamera vor der originalen Image-Plane. Die Image-Distance wird durch die Anzahl der Slices geteilt. Dies ergibt den Bereich *Range*, indem Tiefenwerte zu einer Slice zusammengefasst werden. Das Rendering erfolgt Shader-basiert:

```
1  uniform texture FloatDepthMapTex;
2  uniform float DepthValue;
3  uniform float Range;
4  sampler2D FloatDepthMap = sampler_state
5  {
6    Texture = (FloatDepthMapTex);
7  };
8  ...
9
10 bool IsPixelSameFloatDepth(float2 texCoord)
11 {
12   float4 depth = tex2D(FloatDepthMap, texCoord);
```

```
13   float tolerance = Range / 2.0f;
14   return (IsEqual(depth.r, DepthValue, tolerance)) ?
         true : false;
15 }
16
17 float4 PS_SlicedImage(Output input) : COLOR0
18 {
19   float2 texCoord = input.TexCoord;
20   float4 color = tex2D(image, texCoord);
21   if (IsPixelSameFloatDepth(texCoord))
22     color.a = 1;
23   else
24     // when not correct depth, discard pixel
25     discard;
26   return color;
27 }
```

Programmcode 6.8: *HLSL*-Pixelshader des Sliced-Image-Renderings.

Für das Ansprechen des Shaders wird in einer Schleife der zu rendernde Tiefenwert *DepthValue* berechnet und an den Shader für das Zeichnen der Slice übergeben. Die Schleife bewirkt, dass so viele Slices wie Tiefenbereiche gerendert werden.

```
1 float range = image.DistanceToCamera /
     numberOfSlices;
2 float depthValue = nearClippingPlane;
3
4 while (depthValue < image.ImageDistance)
5 {
6   Matrix world = image.CreateWorldFromCameraDistance
       (depthValue);
7   Shader.Parameters["World"].SetValue(world);
8   Shader.Parameters["DepthValue"].SetValue(
       depthValue);
9   Shader.Parameters["Range"].SetValue(range);
```

```
10    image.Draw(Shader);
11    depthValue += range;
12  }
```

Programmcode 6.9: Ansprechen des Sliced-Image-Shaders für das Zeichnen der Slices innerhalb der Render-Loop.

Für das Rendern der Slices werden keine 3D-Geometrien der Szene, sondern nur planare Ebenen im 3D-Raum (Image-Planes) benötigt.

6.4.3 Umsetzung der Darstellung gegebener 3D-Geometrie

Im Rahmen des *Geometry-Depth-Matching*-Verfahrens wird eine Szenengeometrie des in einem Kamerabild abgebildeten Szenenausschnittes rekonstruiert, um daraus eine pixelgenaue Depth-Map zu generieren. Folglich steht bei dieser Depth-Matching-Variante eine Szenengeometrie als Teilrekonstruktion des Kamerabildes zur Verfügung. Diese kann als Alternative zu dem *Sliced-Image-Rendering* verwendet werden. Das dafür entwickelte *Image-Geometry-Rendering* texturiert die Geometrie mittels perspektivischer Projektion des Kamerabildes. Für die Umsetzung der Texturierung wird das *Texture-Mapping*-Verfahren [Wat02a] verwendet.

6.4.4 Zusammenfassung und Diskussion

Dieser Absatz erläuterte die Umsetzung des vom Autor entwickelten Sliced-Image-Rendering-Verfahrens. Dieses stellt ein *Image-Based-Rendering*-Verfahren dar, das anhand gegebener Tiefeninformationen ein 2D-Bild als 3D-Abbildung darstellen kann, ohne eine Szenengeometrie vorauszusetzen. Auf diese Weise ist eine authentische Darstellung und Verdeckung augmentierter Bilderwelten bei freier Navigation möglich.

Im Rahmen des *Geometry-Depth-Matching*-Verfahrens wurde eine 3D-Geometrie des abgebildeten Szenenausschnittes erstellt. Hierfür wurde Image-Geometry-Rendering implementiert, das die Geometrie mit dem Kamerabild texturiert. Für die perspektivische Projektion der Textur kam das *Texture-Mapping*-Verfahren [Wat02a] zum Einsatz.

6.5 Umsetzung der authentischen Darstellung der virtuellen Objekte

Eine authentische Darstellung und folglich eine authentische Verdeckung einer augmentierten Bilderwelt kann nur im Zusammenspiel mit allen fünf *Augmentierungsschwerpunkten* realisiert werden. Die technische Umsetzung der Augmentierungsschwerpunkte *Aufbereiten des Bildmaterials* und *Verdeckung* (als Kern dieser Arbeit) wurde bereits in den vorangegangenen Unterkapiteln beschrieben.

Die Augmentierungsschwerpunkte *Transformationsanpassung*, *Render-Adaption* und *Bildbasierte Beleuchtung* wurden konzeptionell in Absatz *5.5* (*Authentische Darstellung der virtuellen Objekte*) behandelt. Da dies nicht der Kern der vorliegenden Arbeit betrifft, wird an dieser Stelle die Umsetzung dieser Schwerpunkte innerhalb der PoP-EYE-Anwendung als gegeben angesehen und auf die entsprechenden Veröffentlichungen und Arbeiten des Autors (zum Teil als Co-Autor) in diesen Themenbereichen verwiesen.

In [NBKG09] wurde vom Autor unter Mitarbeit von Co-Autoren ein Verfahren umgesetzt, womit virtuelle 3D-Objekte in Fotos perspektivisch korrekt eingebettet werden (*Transformationsanpassung*). Hierbei wird die virtuelle Kamera der 3D-Objekte entsprechend der Perspektive des Eingabefotos kalibriert. Dies erfolgt durch die Bestimmung des Hauptfluchtpunktes von ermittelten Kanten im Bild. Zur Positionierung der virtuellen Objekte wird die Bodenfläche der im Foto abgebildeten Raumstruktur rekonstruiert.

In [BGKN09] wurde unter Mitarbeit des Autors ein Verfahren veröffentlicht, das für die *Render-Adaption* zum Einsatz kommt. Hierbei werden die Fotos für einen homogenen Gesamteindruck aufbereitet und anschließend Auflösung und Darstellungsqualität der Bildinhalte analysiert und das Rendering der eingebetteten virtuellen Objekte daran angepasst.

In [NKG11d] wurde vom Autor unter Zusammenarbeit mit Co-Autoren ein neuartiges *Image-Based-Lighting*-Verfahren aufbauend auf dem *Sliced-Image-Rendering* veröffentlicht. Anstelle vollständig ausgemessener Umgebungsbeleuchtung werden die Slices des Sliced-Image-Renderings als Eingabedaten für eine *Bildbasierte Beleuchtung* verwendet.

6.6 Zusammenfassung und Diskussion

In diesem Absatz wurde die technische Umsetzung des Konzeptes dieser Arbeit beschrieben. Die im Konzept entwickelten Teilschritte finden sich auch in diesem Kapitel wieder.

Für die Umsetzung des Konzeptes wurde im Rahmen des Drittmittel-Projektes *„PoP-EYE“* der Förderlinie „IngenieurNachwuchs“ 2009 des Bundesministeriums für Bildung und Forschung (BMBF) die Komponenten-basierte Software *PoP-EYE* [NKG12] umgesetzt. Die PoP-EYE-Software ist eine auf Bilderwelten-basierende AR-Umgebung, die eine einfache und schnelle Integration virtueller 3D-Objekte erlaubt, mit dem Hintergrund 3D-Planungsvisualisierungen und Produktpräsentationen in einer fotobasierten 3D-Welt zu ermöglichen. Hierbei steht eine authentische Darstellung der Planungsvisualisierung im Vordergrund. Alle konzeptionellen Umsetzungen der vorliegenden Arbeit flossen als Komponenten in diese Software ein.

Die *Umsetzung der Datenaufbereitung* liest die von der Structure-from-Motion-Software Bundler zur Verfügung gestellten Rohdaten (in Form von Textdateien) ein. Innerhalb der Datenaufbereitung werden die eingelesenen Kameradaten und Punktdaten analysiert und bewertet. Mit den bewerteten Daten wird die Bilderweltenszene in PoP-EYE aufgebaut.

Die *Umsetzung der Teilrekonstruktion der Bilderweltenszene* gewinnt aus den gegebenen Punktdaten der Bilderwelt für die Kamerabilder Tiefeninformationen. Dafür wurden (entsprechend dem Konzept) drei Verfahren implementiert:

- *Segment-Depth-Matching*,
- *Key-Point-Depth-Matching*,
- *Geometry-Depth-Matching*.

Als Ergebnis liegt eine *Float-Depth-Map* vor. Mithilfe dieser wurde in *Umsetzung der authentischen Darstellung der Bilderwelten* das *Sliced-Image-Rendering* implementiert, das anhand gegebener Tiefeninformationen ein Kamerabild als 3D-Abbildung darstellen kann ohne eine Szenengeometrie vorauszusetzen. Auf diese Weise ist eine authentische Darstellung und Verdeckung augmentierter Bilderwelten bei freier Navigation möglich. Als Alternative für das Sliced-Image-Rendering und im Falle von gegebener Szenengeometrie eines

abgebildeten Szenenausschnittes des Kamerabildes wurde das *Image-Geometry-Rendering* entwickelt, dass über *Texture-Mapping* [Wat02a] die Geometrie mit dem Kamerabild perspektivisch korrekt texturiert.

Kapitel 7

Evaluierung der Ergebnisse

7.1 Überblick

Dieses Kapitel zeigt, analysiert und bewertet die Ergebnisse des zusammengetragenen Konzeptes und der dazugehörigen technischen Umsetzung. Der Augmentierungsschwerpunkt *Verdeckung* stellt den Kern dieser Arbeit dar. Deshalb konzentriert sich die hier beschriebene Evaluierung auf die dafür verantwortlichen Teilschritte und ihre Verfahren. Für das Gewinnen von Tiefeninformationen des abgebildeten Szenenausschnittes der Kamerabilder wurden die drei Verfahren *Segment-Depth-Matching*, *Key-Point-Depth-Matching* und *Geometry-Depth-Matching* entwickelt. Die Ergebnisse sowie deren Gegenüberstellung wird in Absatz *7.2* (*Evaluierung der Teilrekonstruktion der Bilderweltenszene*) beschrieben. Auf der Basis der gewonnenen Tiefeninformationen wird in Absatz *7.3* (*Evaluierung der authentischen Darstellung der Bilderwelten*) die Ergebnisse des *Sliced-Image-Renderings* und des *Image-Geometry-Renderings* evaluiert.

Für das Bildmaterial der hier vorliegenden Arbeit wurde (aufgrund des Wiedererkennnungswertes) bis zu diesem Kapitel stets eine Innenraumszene mit einem grünen Stuhl verwendet. Im Rahmen dieses Kapitels werden einige andere Szenen und Bildserien eingeführt (siehe *Tab. 7.1*), um die allgemeine Einsatzfähigkeit des entwickelten Konzeptes sowie dessen Umsetzung zu unterstreichen. Hierbei wurden Szene 2 und Szene 3 jeweils mit großen und kleinen Bildserien als Eingabematerial für die Erstellung der Bilderwelt verwendet.

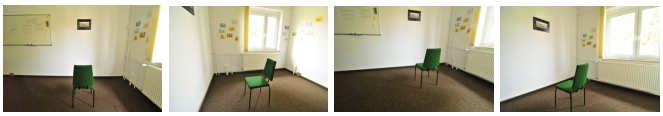
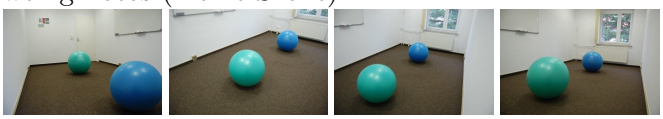
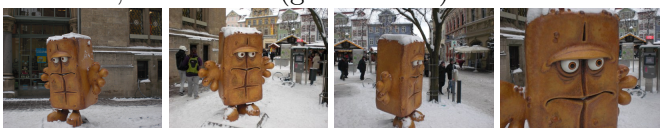
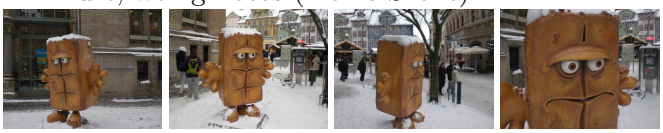
Szenenname der Bilderwelt	Kameras/ Bilder	Key-Points
Szene 1: Innenraumszene mit einem Occludder 	40	4194
Szene 2(a): Innenraumszene mit zwei Occludder, viele Fotos (große Szene) 	470	262059
Szene 2(b): Innenraumszene mit zwei Occludder, wenig Fotos (kleine Szene) 	23	3877
Szene 3(a): Außenszene der Bernd-das-Brot-Statue in Erfurt, viele Fotos (große Szene) 	254	34509
Szene 3(b): Außenszene der Bernd-das-Brot-Statue in Erfurt, wenig Fotos (kleine Szene) 	15	665
Szene 4: Außenszene des Gebrüder-Grimm-Denkmals in Hanau 	299	36082

Tabelle 7.1: Im Rahmen der Evaluierung verwendete Bilderweltenszenen.

7.2 Evaluierung der Teilrekonstruktion der Bilderweltenszene

Für die Teilrekonstruktion wurden verschiedene Verfahren entwickelt, die je nach Teilziel aus einer Kombination aus den Mechanismen

- *Bildsegmentierung* und
- *Tiefenzuordnung*

bestehen.

7.2.1 Evaluierung der Bildsegmentierung

Im Rahmen der Bildsegmentierung wurden vier Varianten umgesetzt:

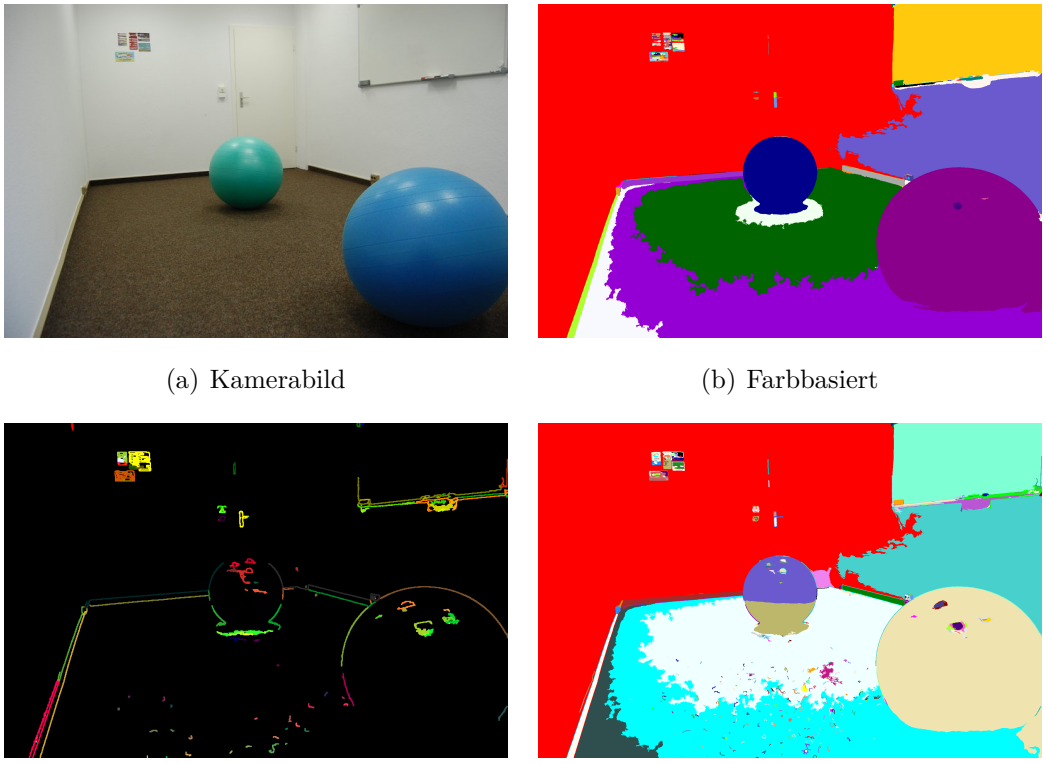
- *Farbbasierte Segmentierung*,
- *Konturbasierte Segmentierung*,
- *Farb- und Konturbasierte Segmentierung*,
- *Objektbasierte Segmentierung*.

Im folgenden Textabschnitt werden noch einmal die zusammengetragenen Vor- und Nachteile der im Konzept beschriebenen Segmentierungsverfahren aufgezeigt. Der Vorteil der *farbbasierten Segmentierung* liegt in der Einfachheit des Mechanismus. Benachbarte Pixel ähnlicher Farbe werden als ein Segment gruppiert. Nachteilig gestalten sich Fotos mit unsatten Farben und Abbildungen mit mehreren benachbarten Objekten gleichen Farbtons. Objekte mit unsatten, ähnlichen Farben können „übersehen“ werden und fälschlicherweise mit anderen Segmenten verschmolzen werden. Benachbarte Objekte mit ähnlichen oder gleichen Farbton können fälschlicherweise als ein Segment detektiert werden. *Abb. 5.10* zeigt ein farbunsattes Bild. Der abgebildete vordergründige Raumbalken besitzt eine ähnliche Farbe wie die Wände im Hintergrund. Das dazugehörige Segmentbild detektiert den Balken nicht als ein eigenes Objekt, sondern sie verschmelzen mit der hinteren Wand und dem Fußboden. Dies kann auch nicht durch die Erniedrigung des Schwellwertes für den Farbabstand korrigiert werden, da die Farbe des Raumbalkens sehr ähnlich der Farbe der

Wände im Hintergrund ist. Der Vorteil der *konturbasierten Segmentierung* ist die Robustheit gegenüber unsatten Farben. Dies ist zurückzuführen auf die Gradient-basierten Kantendetektion, die lediglich die Intensitäten (Helligkeit) des Bildes, nicht aber die Farbinformationen verarbeitet. Dementsprechend wirkt sich die Beschränkung auf die Intentitätsverläufe eines Bildes nachteilig aus, wenn die Helligkeitsverläufe nicht sehr ausgeprägt sind. Diese Segmentierungsform setzt ein detailiertes Kantenbild voraus. Sie ist fehleranfällig im Falle von fehlenden Kanteninformationen (z.B. häufig unterbrochene Kanten) eines Objektes. *Abb. 5.15* stellt die Farbsegmentierung als Einzelverfahren der Kombination aus *farb- und konturbasierten Segmentierung* gegenüber. Aufgrund der Kombination beider Segmentierungsverfahren werden die bereits erwähnten Fehleranfälligkeiten der einzelnen Verfahren reduziert. Die beiden Segmentbilder der Abbildung zeigen, dass das zusätzliche Verwenden (Einzeichnen) der Konturen bei der Farbsegmentierung eine Unterstützung bei der Abgrenzung einzelner abgebildeter Objekte (z. B. der im Vordergrund befindliche Balken) bewirkt. Somit ist diese kombinierte Segmentierungsform gegenüber den Einzelverfahren robuster gegenüber unsatten Farben und benachbarten abgebildeten Objekten mit gleichem Farbton, als auch gegenüber fehlenden Kanteninformationen (z.B. häufig unterbrochene Kanten) eines abgebildeten Objektes im Kantenbild. Sollte ein Kamerabild nicht sehr ausgeprägte Helligkeitsverläufe und unsatte Farben abbilden, kann es auch hier zu fehlerhaften Segmenten kommen. Des Weiteren werden zwei Segmentierungsverfahren kombiniert bzw. nacheinander abgearbeitet. Dies führt zu einer erhöhten Rechenzeit gegenüber den Einzelverfahren.

Eine Gegenüberstellung der Vor- und Nachteile findet sich in *Tab. 5.3*.

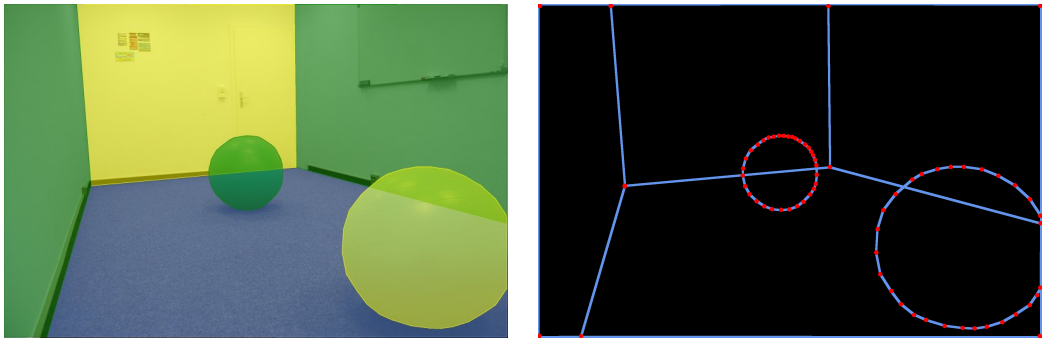
Neben der im Konzept bereits gegenübergestellten Vor- und Nachteile soll an dieser Stelle ein Fazit gezogen werden. Dazu soll eine weitere Bildserie (Szene 2, *Tab. 7.1*) in *Abb. 7.1* betrachtet werden, die (im Sinne der Evaluierung) repräsentativ für die gewonnenen Erkenntnisse aus einer Vielzahl von Bildserien und Testszenen steht. Die farbbasierte Segmentierung ermöglicht eine verhältnismäßig stabile Segmentierung der realen abgebildeten Objekte in Fotos. Die konturbasierte Segmentierung ohne Kombination mit der Farbsegmentierung führt häufig zu fehlerhaften Segmenten. Die Kombination aus farb- und konturbasierter Segmentierung erzielt (trotz des Mehraufwandes gegenüber der



(a) Kamerabild (b) Farbbasiert

(c) Konturbasiert (d) Farb- und konturbasiert

Abbildung 7.1: Gegenüberstellung der farb- und konturbasierten Segmentierung.



(a) Semiautomatische Segmentskizzierung (b) Segmentbild mit Eckpunkten

Abbildung 7.2: Gegenüberstellung der farb- und konturbasierten Segmentierung.

Farbsegmentierung als Einzelverfahren) nur unwesentlich bessere und teilweise sogar schlechtere Resultate als die alleinige Farbsegmentierung. Folglich wird das Fazit gezogen, dass die farbbasierte Segmentierung i. d. R. die besten Ergebnisse von den drei Verfahren erzielt.

Neben den automatisch ablaufenden Segmentierungsverfahren wurde die semiautomatische objektbasierte Segmentierung entwickelt. Der Vollständigkeit halber wird in *Abb. 7.2* das Segmentbild dieses Segmentierungsverfahrens dargestellt, obwohl die Qualität des Segmentbildes nicht bewertbar ist (aufgrund, dass die Segmente in einem semiautomatischen Skizzierungsschritt vom Nutzer eingezeichnet werden).

7.2.2 Evaluierung der Tiefenzuordnung

Im Rahmen der Tiefenzuordnung wurden drei Verfahren entwickelt, die verschiedene Eingabedaten voraussetzen und unterschiedliche Ergebnisse produzieren:

- *Segment-Depth-Matching*,
- *Key-Point-Depth-Matching*,
- *Geometry-Depth-Matching*.

Evaluierung des Segment-Depth-Matching-Verfahrens

Das *Segment-Depth-Matching*-Verfahren setzt ein Segmentbild voraus. Dementprechend ist das Ergebnis der dort erstellten *Depth-Map* direkt abhängig von

dem Segmentierungsergebnis. In *Abb. 7.3* wurden für die Kamerabilder von Szene 2 (siehe *Tab. 7.1*) unter Verwendung der vorgestellten Segmentierungsverfahren (siehe *Abb. 7.1*) entsprechende Depth-Maps erstellt. Da die Segmentierungsergebnisse der konturbasierten Segmentierung (siehe *Abb. 7.1(c)*) zu stark fehlerhaften Depth-Maps führte, wurden diese Depth-Maps in den folgenden Abbildungen vernachlässigt. Es gibt kaum nennenswerte Unterschiede zwischen der farbbasierten Segmentierung als Einzelverfahren und der kombinierten farb- und konturbasierten Segmentierung (vgl. *Abb. 7.3(c)* und *Abb. 7.3(d)*). Folgerichtig sind die erstellten Depth-Maps ebenfalls qualitativ kaum unterschiedlich (vgl. *Abb. 7.3(e)* und *Abb. 7.3(f)*). In diesem Beispiel muss sogar erwähnt werden, dass die Depth-Map, basierend auf der kombinierten farb- und konturbasierten Segmentierung, qualitativ schlechter ist, da einige Farbsegmente (z. B. das untere Segment des hinteren Balls) nicht mit Tiefeninformationen gefüllt wurden. Da unter Verwendung der gleichen Szene mit mehr Fotos (Szene 2(a), *Tab. 7.1*) keine qualitative Verbesserungen der Depth-Maps festzustellen war, werden für Szene 2 und 3 jeweils die Ergebnisse der kleineren Szenen (weniger Fotos) gezeigt, da dies auch den Anforderungen an diese Arbeit entspricht.

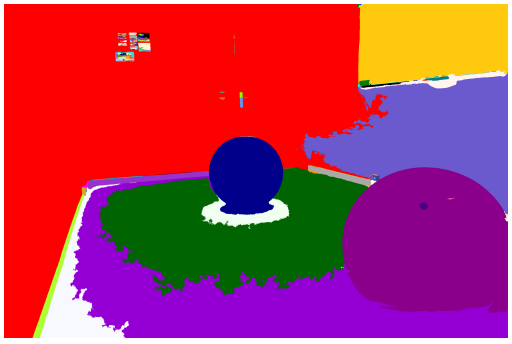
Die Ergebnisse von Szene 3(b) (siehe *Tab. 7.1*) finden sich in *Abb. 7.4*. Hier muss erwähnt werden, dass die Depth-Map, basierend auf der kombinierten farb- und konturbasierten Segmentierung, qualitativ etwas besser erscheint, da das fehlerhafte „Herausragen“ einiger *Depth-Patches*, wie in *Abb. 7.3(e)*, aufgrund der Konturgrenzen der Segmentierung unterbunden wird. Dennoch sorgt das Einzeichnen von Konturen dafür, dass einige Segmente keine Tiefeninformationen erhalten.

Ähnlich verhält es sich in Szene 4 (siehe *Tab. 7.1*), wie in *Abb. 7.5* dargestellt wird. Auch hier wird das „Herausragen“ einiger Depth-Patches durch die kombinierte farb- und konturbasierten Segmentierung verringert.

Der Vorteil des Segment-Depth-Matching-Verfahrens ist die segmentgenaue Zuordnung von Tiefenwerten mit nur geringen Tiefeninformationen bzw. mit einer geringen Anzahl an Observed-Points. Prinzipiell reicht ein Observed-Point pro Segment aus. Da eine Segmentierung vorausgegangen ist, sind die Konturgrenzen der abgebildeten Objekte innerhalb der Depth-Map sehr genau (vorausgesetzt die Segmentierung liefert genaue Segmente). Der Nachteil dieses Verfahrens liegt in der Abhängigkeit bzw. Voraussetzung einer Segmentie-

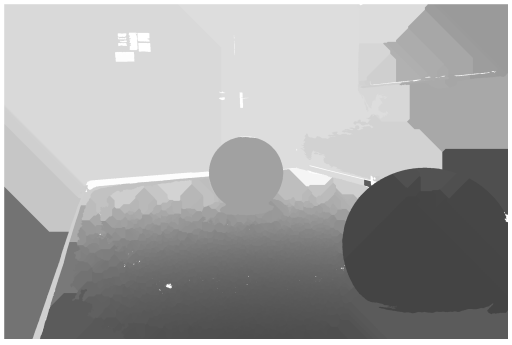
(a) Kamerabild

(b) Mit eingezeichneten Konturen

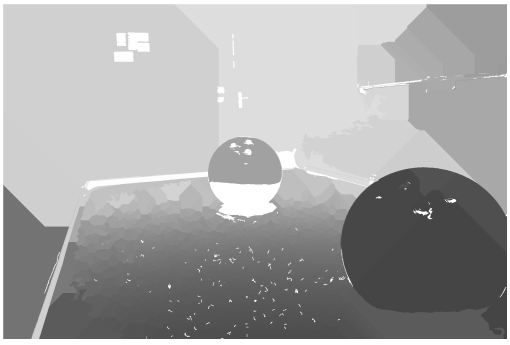


(c) Farbsegmentierung

(d) Farb- und und Kontursegmentierung



(e) Depth-Map (mittels Farbsegmentierung)



(f) Depth-Map (Farb- und Kontursegmentierung)

Abbildung 7.3: Gegenüberstellung der Segment-Depth-Matching-Ergebnisse von Szene 2(b) (siehe *Tab. 7.1*) durch verschiedene Segmentierungsverfahren.

(a) Kamerabild

(b) Mit eingezeichneten Konturen

(c) Farbsegmentierung

(d) Farb- und und Kontursegmentierung

(e) Depth-Map (mittels Farbsegmentierung)

(f) Depth-Map (Farb- und Kontursegmentierung)

Abbildung 7.4: Gegenüberstellung der Segment-Depth-Matching-Ergebnisse von Szene 3(b) (siehe *Tab. 7.1*) durch verschiedene Segmentierungsverfahren.

(a) Kamerabild

(b) Mit eingezeichneten Konturen

(c) Farbsegmentierung

(d) Farb- und und Kontursegmentierung

(e) Depth-Map (mittels Farbsegmentierung)

(f) Depth-Map (Farb- und Kontursegmentierung)

Abbildung 7.5: Gegenüberstellung der Segment-Depth-Matching-Ergebnisse von Szene 4 (siehe *Tab. 7.1*) durch verschiedene Segmentierungsverfahren.

rung. Ein ungenaues oder fehlerhaftes Segmentbild führt zu einer fehlerhaften Depth-Patches. Ein weiterer Nachteil stellt die Zuordnung von Segmenten und Observed-Points dar. Segmente, ohne zugeordnete Observed-Points, können auch keine Tiefe zugeordnet werden. Eine Strategie für das Behandeln dieses Falles ist es, als Tiefenwert die Image-Distance der Kamera zu verwenden.

In den Ergebnisbildern der erstellten Depth-Maps ist deutlich zu sehen, dass die Depth-Patches flächengenau und nicht pixelgenau sind. Das heißt ein Depth-Patch besteht aus mehreren Pixeln mit dem gleichen Füllwert. Daraus folgt, je größer das Depth-Patch, desto ungenauer ist die Tiefenzuordnung, abgesehen von Segmenten abgebildeter realer Objekte, die sich orthogonal zur Kamera befinden.

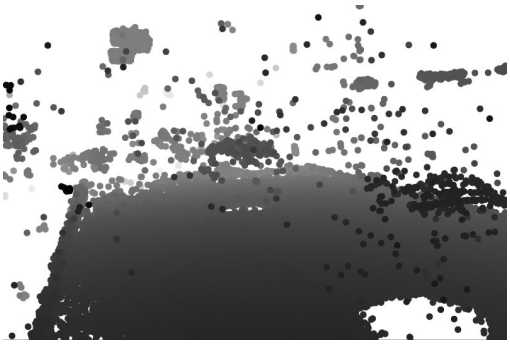
Evaluierung des Key-Point-Depth-Matching-Verfahrens

Im Gegensatz zu dem Segment-Depth-Matching-Verfahren, setzt das *Key-Point-Depth-Matching*-Verfahren keine Segmentierung voraus, sondern projiziert die Observed-Points auf die Bildebene als kreisförmige Sprites mit einem definierten Radius r. Folglich ist dieses Verfahren im Vergleich zu dem Segment-Depth-Matching-Verfahren weniger rechenintensiv. Der Nachteil dieses Verfahrens ist die Bedingung einer Punktwolke, die den abgebildeten Szenenausschnitt des Fotos verhältnismäßig vollständig abdeckt. Ist dies nicht der Fall, entstehen Bereiche in der Depth-Map ohne Tiefeninformationen. Dies widerspricht zum Teil der Anforderung, keine dichten Punktwolken voraussetzen zu müssen. Eine Strategie für das Behandeln dieser Bereiche ist es, als Tiefenwert die Image-Distance der Kamera zu verwenden. Einen weiteren Nachteil stellen die Konturgrenzen der Depth-Patches dar. Diese sind ungenauer als bei dem Segment-Depth-Matching-Verfahren. Dies ist auf das Verwenden kreisförmiger Sprites zurückzuführen. Die Ungenauigkeit steigt mit zunehmenden Radius der Sprites. In *Abb. 7.6* sowie in *Abb. 7.7* wurde das Key-Point-Depth-Matching-Verfahren mit verschiedenen Radien der Sprites angewandt. Aus diesen Abbildungen ist deutlich zu erkennen, dass eine nicht flächendeckende Punktwolke zu lückenhaften und ungenauen Depth-Maps führt. Auch das Erhöhen des Radius der Sprites kann die Bereiche mit fehlenden Tiefeninformationen nicht vollständig und nicht korrekt schließen. Denn mit zunehmenden Radius der Sprites steigt die Ungenauigkeit.

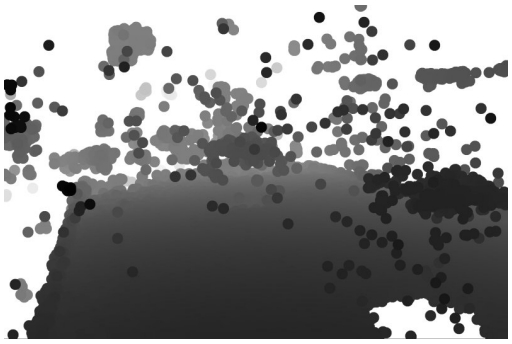
(a) Kamerabild

(b) Kamerabild mit Key-Points

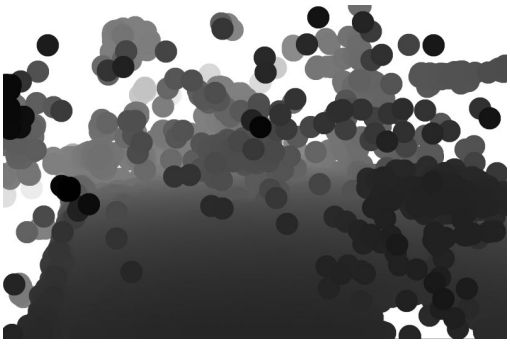
(c) Radius $r = 20px$



(d) Radius $r = 30px$



(e) Radius $r = 50px$

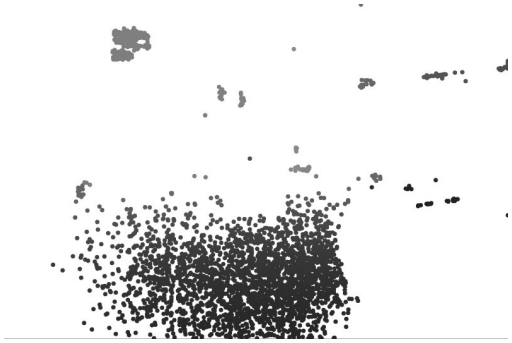


(f) Radius $r = 100px$

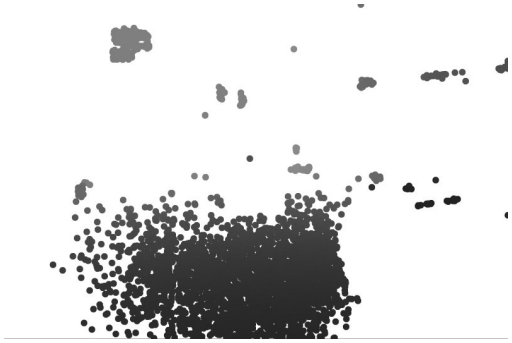
Abbildung 7.6: Gegenüberstellung der Key-Point-Depth-Matching-Ergebnisse der Szene 2(a) (siehe *Tab. 7.1*) mit unterschiedlichen Radien der Sprites. Trotz großer Szene (viele Fotos) können nicht alle Bereiche mit Tiefeninformationen versorgt werden. Dies ist darauf zurückzuführen, dass keine flächendeckende Punktwolke vorliegt.

(a) Kamerabild

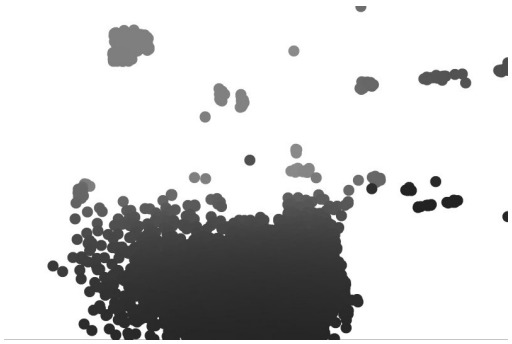
(b) Kamerabild mit Key-Points



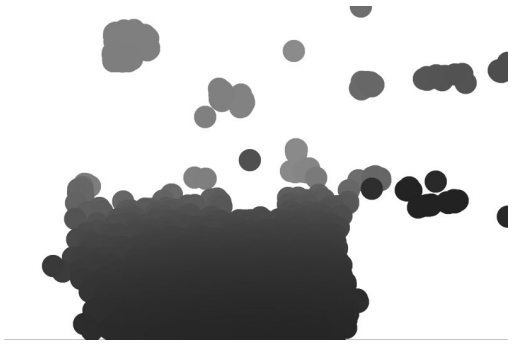
(c) Radius $r = 20px$



(d) Radius $r = 30px$

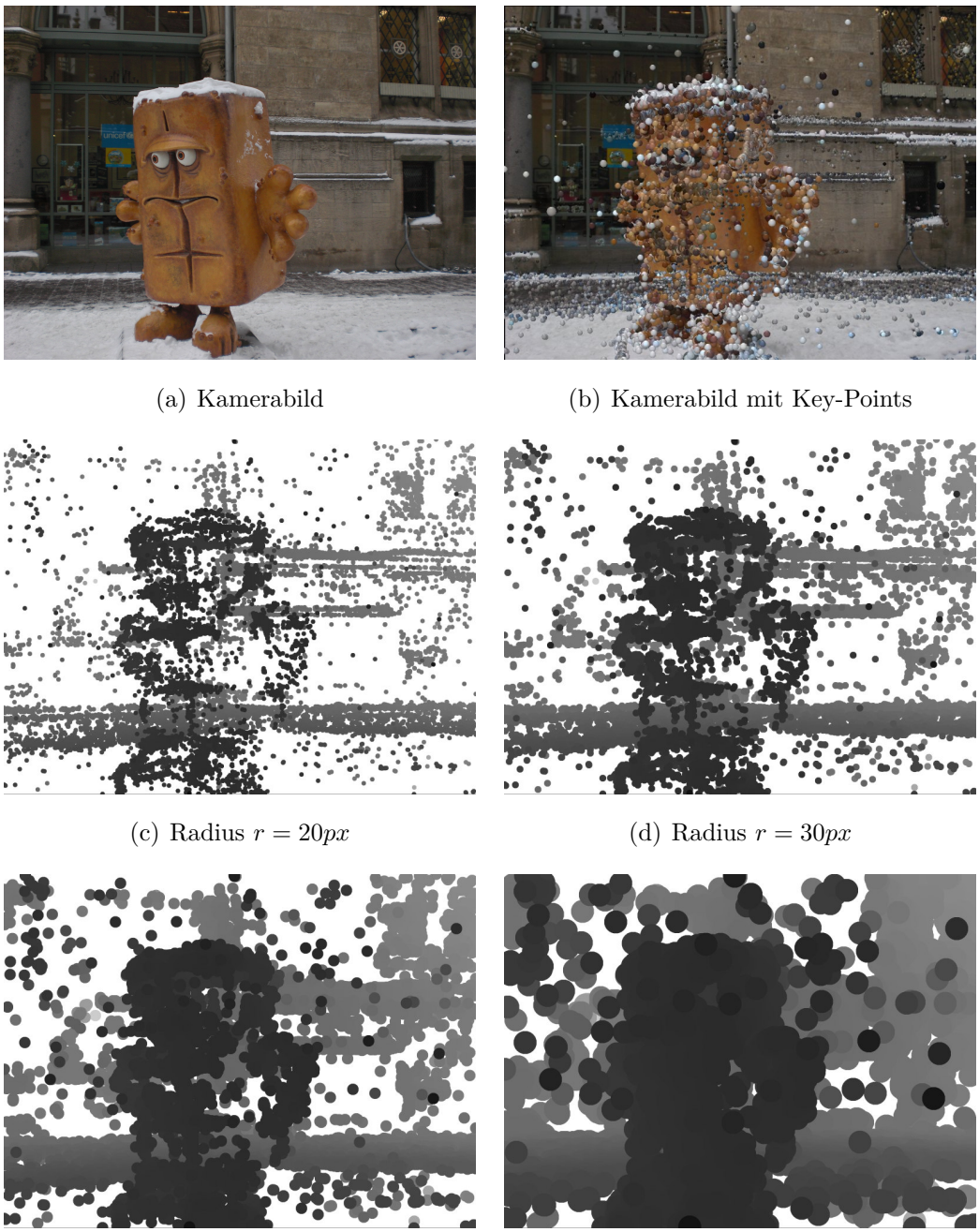


(e) Radius $r = 50px$



(f) Radius $r = 100px$

Abbildung 7.7: Gegenüberstellung der Key-Point-Depth-Matching-Ergebnisse der Szene 2(b) (siehe *Tab. 7.1*) mit unterschiedlichen Radien der Sprites. Da nur eine kleine, nicht flächendeckende Punktwolke vorliegt, ist das Ergebnis der Depth-Maps für das Erreichen des Hauptziels dieser Arbeit unbrauchbar.



(a) Kamerabild

(b) Kamerabild mit Key-Points

(c) Radius $r = 20px$

(d) Radius $r = 30px$

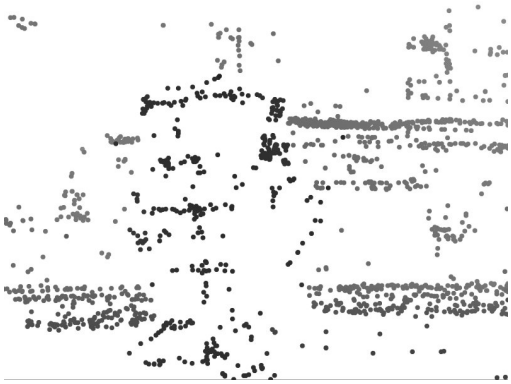
(e) Radius $r = 50px$

(f) Radius $r = 100px$

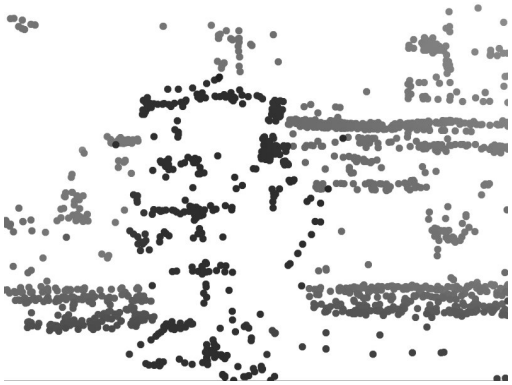
Abbildung 7.8: Gegenüberstellung der Key-Point-Depth-Matching-Ergebnisse der Szene 3(a) (siehe *Tab. 7.1*) mit unterschiedlichen Radien der Sprites. In dieser Szene ist die Punktwolke besser bzw. flächendeckender verteilt als in Szene 2(a) in *Abb. 7.6* . So kann unter Verwendung eines geeigneten Radius eine nutzbare Depth-Map erstellt werden.

(a) Kamerabild

(b) Kamerabild mit Key-Points

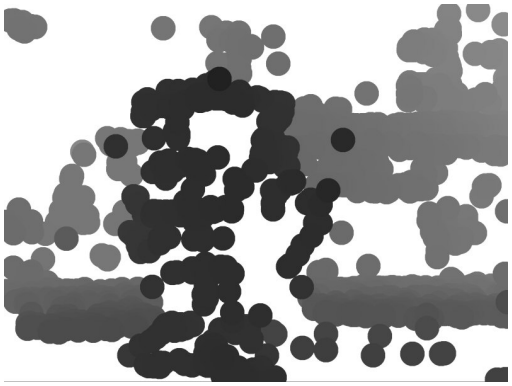


(c) Radius $r = 20px$



(d) Radius $r = 30px$

(e) Radius $r = 50px$



(f) Radius $r = 100px$

Abbildung 7.9: Gegenüberstellung der Key-Point-Depth-Matching-Ergebnisse der Szene 3(b) (siehe *Tab. 7.1*) mit unterschiedlichen Radien der Sprites. Im Gegensatz zu *Abb. 7.8* bleibt bei Verwendung weniger Fotos in dieser Szene die Depth-Map unbrauchbar, da zu viele Bereiche ohne Tiefeninformationen vorhanden sind.

Trotz großer Szene (viele Fotos) in *Abb. 7.6* können nicht alle Bereiche mit Tiefeninformationen versorgt werden. Dies ist darauf zurückzuführen, dass keine flächendeckende Punktwolke vorliegt. In *Abb. 7.7* wurde die (gleiche) Szene 2(b) mit einer kleineren Punktwolke (im Gegensatz zu Szene 2(a) in *Abb. 7.6*) verwendet. Da nur eine kleine, nicht flächendeckende Punktwolke vorliegt, ist das Ergebnis der Depth-Maps für das Erreichen des Hauptziels dieser Arbeit unbrauchbar.

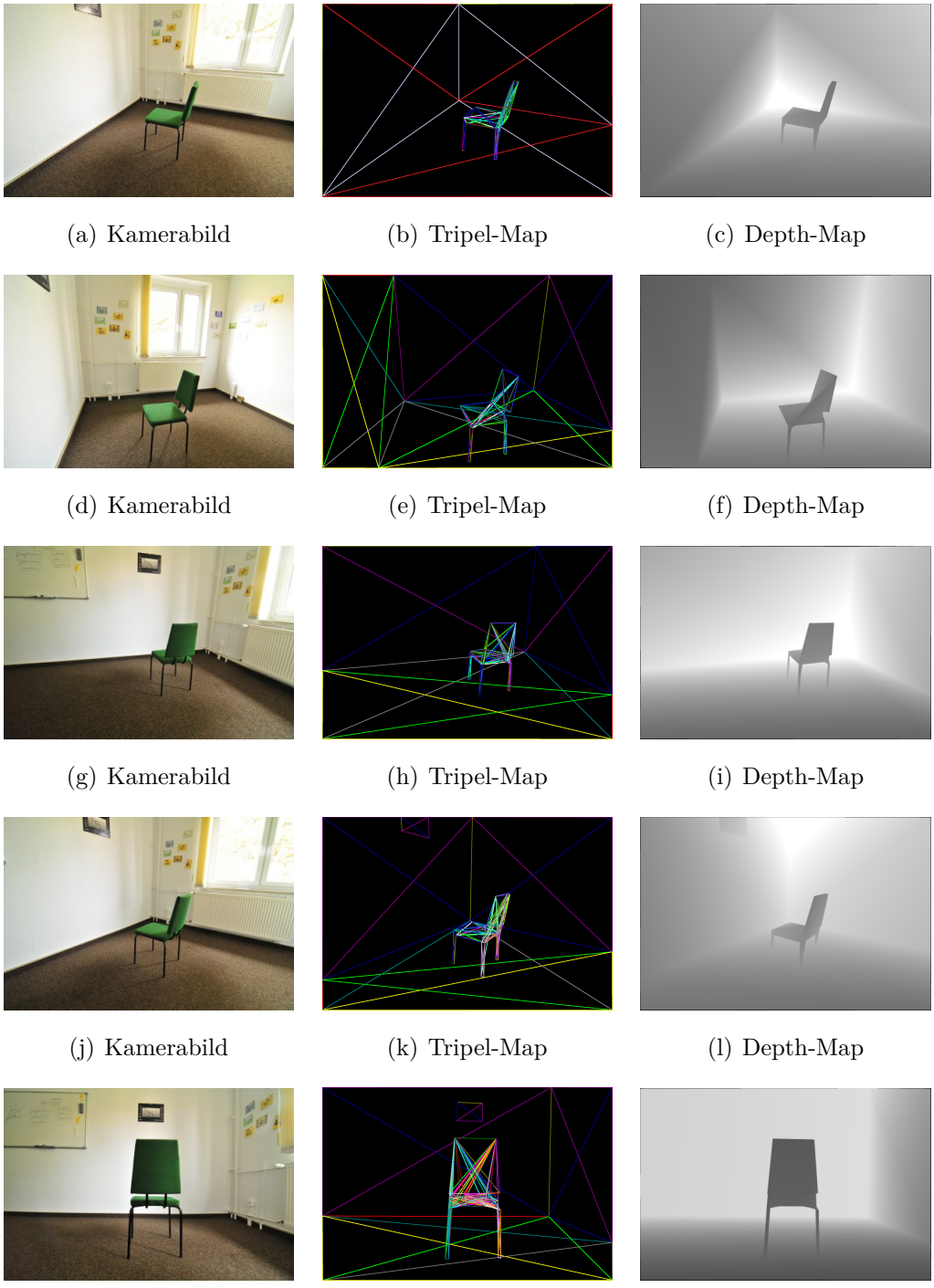
In *Abb. 7.8* und *Abb. 7.9* wurde für eine weitere Szene (Szene 3, siehe *Tab. 7.1*) das Key-Point-Depth-Matching-Verfahren mit verschiedenen Radien der Sprites angewandt. In dieser Szene ist die Punktwolke besser bzw. flächendeckender verteilt als in Szene 2(a) in *Abb. 7.6*. So kann, zumindest in Szene 3(a) unter Verwendung eines geeigneten Radius eine nutzbare Depth-Map erstellt werden. Bei Verwendung weniger Fotos in Szene 3(b) bleibt jedoch die Depth-Map unbrauchbar, da zu viele Bereiche ohne Tiefeninformationen vorhanden sind.

Analog zu dem Segment-Depth-Matching-Verfahren, enthalten die erstellten Depth-Maps des Key-Point-Depth-Matchings flächengenaue und nicht pixelgenaue Depth-Patches. Das heißt die Sprites (Depth-Patches) bestehen aus mehreren Pixeln mit dem gleichen Füllwert. Daraus folgt, je größer der Radius eines Sprites, desto ungenauer ist die Tiefenzuordnung.

Evaluierung des Geometry-Depth-Matching-Verfahrens

Das Segment-Depth- sowie das Key-Point-Depth-Matching liefern keine pixelgenauen Depth-Maps. Dafür wurde das *Geometry-Depth-Matching*-Verfahren vom Autor dieser Arbeit entwickelt. Dieses Verfahren beinhaltet einen semiautomatischen Skizzierungsschritt (Einzeichnen der Eckpunkte der abgebildeten Objekte). Dementsprechend ist das Verfahren auf eine (annähernd) genaue Skizzierung durch den Nutzer angewiesen. Andernfalls verschlechtern ungenaue Einzeichnungen die Ergebnisse. Für die Evaluierung wurden in *Abb. 7.10* und *Abb. 7.11* sehr genau eingezeichnete Segmentskizzierungen verwendet. Dennoch muss darauf hingewiesen werden, dass dies nicht immer vorausgesetzt werden kann und ungenaue Einzeichnungen die Ergebnisse verschlechtern. Daher wurde in *Abb. 7.12* absichtlich eine ungenaue Segmentskizzierung durchgeführt, um die Nutzbarkeit des Verfahrens auch unter diesen Bedingungen zu verdeutlichen.

In *Abb. 7.10* werden die Ergebnisse des Geometry-Depth-Matching-Verfah-



(a) Kamerabild (b) Tripel-Map (c) Depth-Map

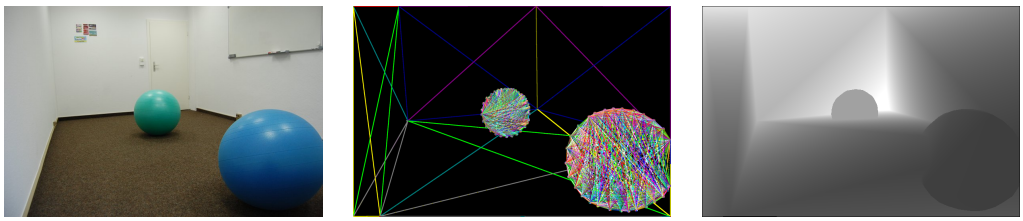
(d) Kamerabild (e) Tripel-Map (f) Depth-Map

(g) Kamerabild (h) Tripel-Map (i) Depth-Map

(j) Kamerabild (k) Tripel-Map (l) Depth-Map

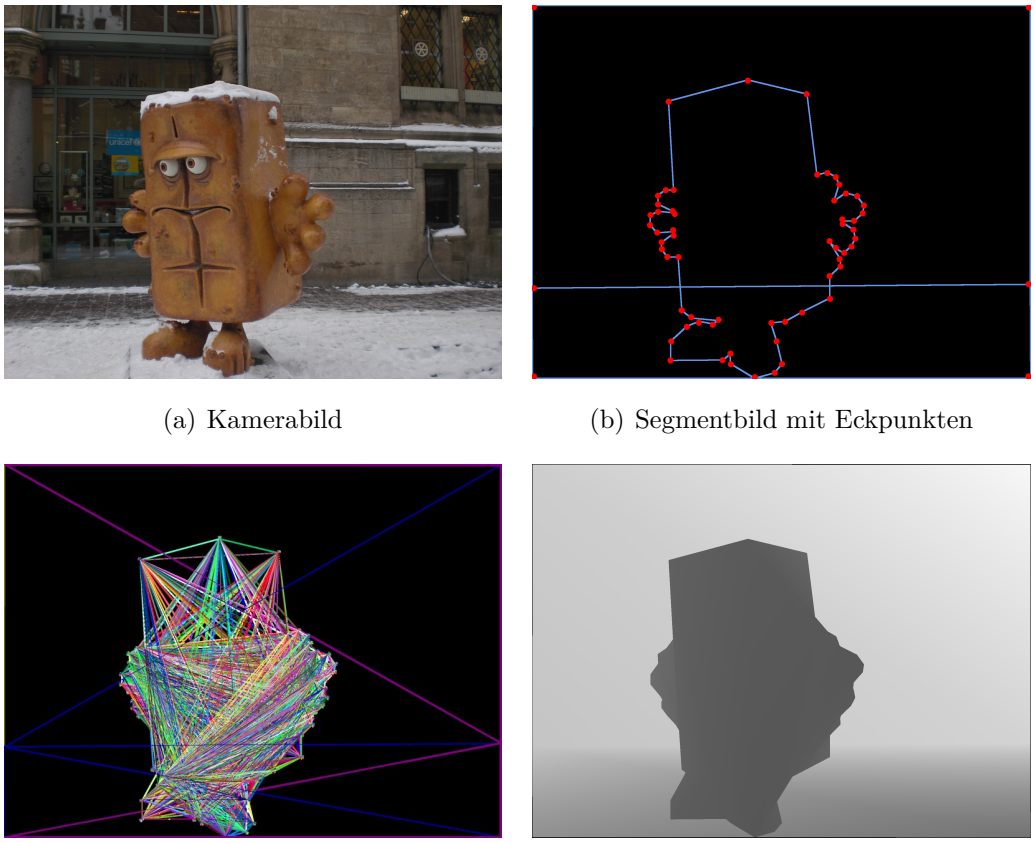
(m) Kamerabild (n) Tripel-Map (o) Depth-Map

Abbildung 7.10: Ergebnisse des Geometry-Depth-Matching-Verfahrens der Szene 1 (siehe *Tab. 7.1*) für unterschiedliche Fotos der Bilderwelt.



(a) Kamerabild (b) Tripel-Map (c) Depth-Map

Abbildung 7.11: Beispiel der Ergebnisse des Geometry-Depth-Matching-Verfahrens der Szene 2(b) (siehe *Tab. 7.1*).



(a) Kamerabild (b) Segmentbild mit Eckpunkten

(c) Tripel-Map (d) Depth-Map

Abbildung 7.12: Beispiel der Ergebnisse des Geometry-Depth-Matching-Verfahrens der Szene 3(b) (siehe *Tab. 7.1*). Trotz verhältnismäßig grober Segmentskizzierung entsteht eine brauchbare, pixelgenaue Depth-Map.

rens der Szene 1 (siehe *Tab. 7.1*) für unterschiedliche Fotos der Bilderwelt aufgezeigt. Neben der pixelgenauen Depth-Map wird die dazugehörige *Tripel-Map* dargestellt. Diese beinhaltet die in Schritt *Bildung von Eckpunkt-Tripeln* detektierten Tripel der zweidimensionalen Eckpunkte, aus denen die 3D-Triangles der Szenengeometrie erstellt wird. *Abb. 7.11* zeigt beispielhaft das Ergebnis eines Bildes aus Szene 2(b). In *Abb. 7.12* wurde Szene 3(b) verwendet. Hier wurden die Segmente verhältnismäßig grob und ungenau eingezeichnet. Dies wurde durchgeführt, um die Nutzbarkeit des Verfahrens auch unter diesen Bedingungen zu demonstrieren. Trotz grober Segmentskizzierung entsteht ein brauchbare, pixelgenaue Depth-Map. Wie in den Abbildungen gezeigt, kann in allen drei Beispielszenarien mittels des Geometry-Depth-Matching-Verfahrens eine pixelgenaue Depth-Map des Kamerabildes erzeugt werden. Probleme ergeben sich jedoch, sobald Segmente sehr ungenau skizziert werden oder nicht alle Flächen eines Bildes einem Segment zugeordnet werden. Nicht zugeordnete Bereiche bleiben ohne 3D-Geometrie und folglich ohne Tiefeninformationen innerhalb der Depth-Map (ähnlich den Ergebnissen aus *Abb. 5.27*).

7.3 Evaluierung der authentischen Darstellung der Bilderwelten

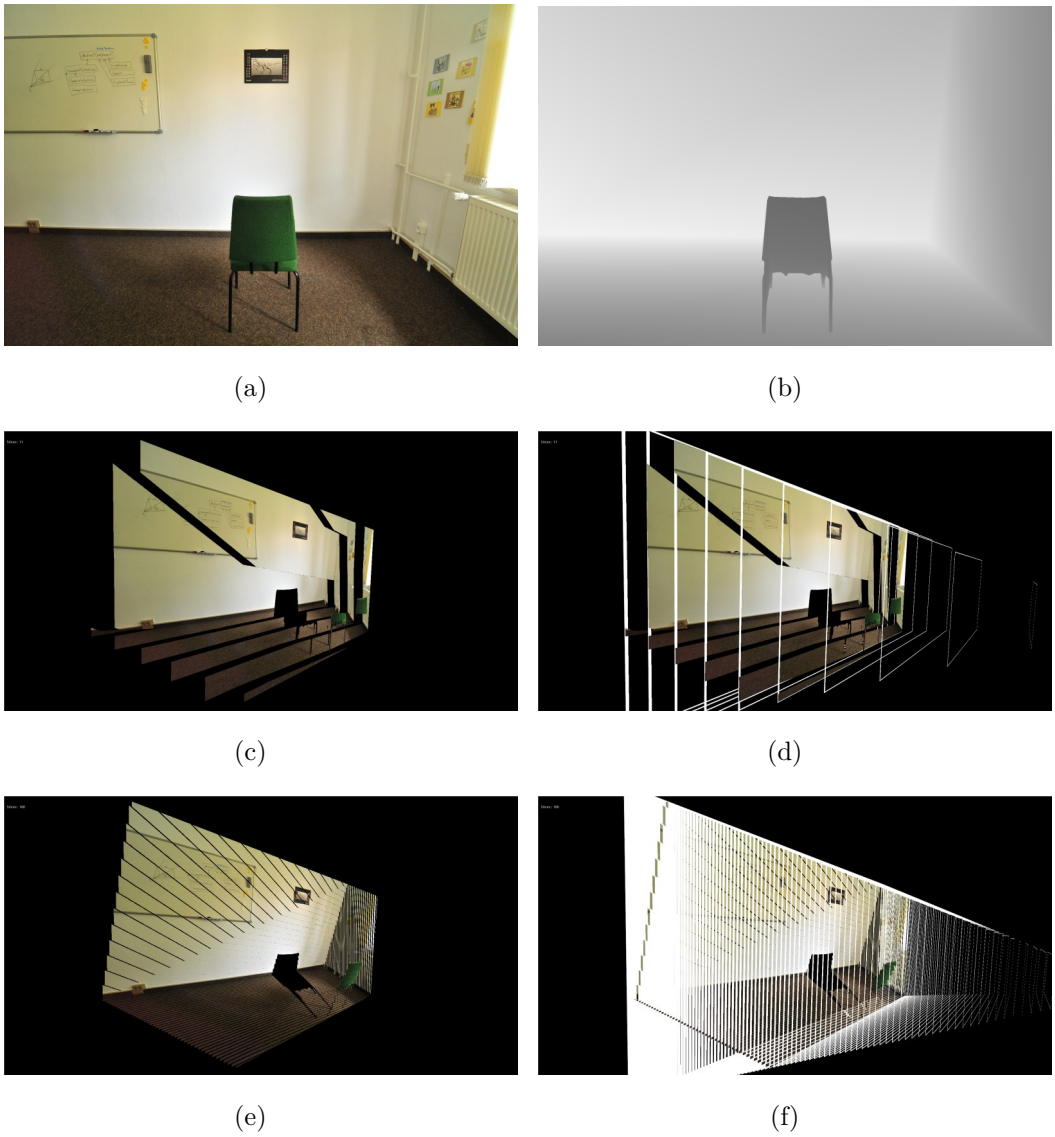
Im Rahmen der *Authentischen Darstellung der Bilderwelten* wurde der Fokus auf das Hauptziel der Arbeit gelegt. Dementsprechend wird sich die Evaluierung der Mechanismen dieses Teilschrittes auf die *Authentische Verdeckung* bei der Darstellung augmentierter Bilderwelten beschränken. Dennoch soll an dieser Stelle noch einmal auf das notwendige Zusammenspiel aller Augmentierungsschwerpunkte (siehe *Abb. 1.5*) für eine authentische Darstellung und Augmentierung von Bilderwelten hingewiesen werden. Die Umsetzung der *Authentischen Darstellung der Bilderwelten* umfasste das

- *Sliced-Image-Rendering* sowie das
- *Image-Geometry-Rendering*.

Hierbei muss erwähnt werden, dass das Image-Geometry-Rendering als „Abfallprodukt“ des *Geometry-Depth-Matching*-Verfahrens zu sehen ist und nicht alle Kernanforderung (siehe Absatz *1.3* (*Kernanforderungen an die Arbeit*)) an diese Arbeit erfüllt. Dementsprechend liegt im Rahmen der Evaluierung das Hauptaugenmerk auf dem Sliced-Image-Rendering.

7.3.1 Evaluierung des Sliced-Image-Renderings

In *Abb. 7.13* wurde noch einmal anhand von Szene 1 (siehe *Tab. 7.1*) als Einzelbild das Rendering eines Sliced-Images bei freier Navigation (anhand der Tiefeninformationen des *Geometry-Depth-Matching*-Verfahrens) dargestellt. Hierbei wurde das Sliced-Image zum einen mit 10 Slices und zum anderen mit 100 Slices gerendert. Die Abbildungen zeigen, dass mit zunehmender Anzahl an Slices auch die Darstellungsgenauigkeit des Szenenausschnittes zunimmt. Da jedes Slices als ein eigenständiges Bild (bzw. Image-Plane) gerendert wird, steigt jedoch auch der Renderaufwand. In *Abb. 7.14* wurde die Szene mit einem virtuellen Tisch augmentiert. In der Abbildung wird gezeigt, wie mithilfe des Sliced-Image-Renderings eine authentische Verdeckung bei Kamera- sowie bei freier Navigation realisiert werden kann. Anzumerken ist, dass hierfür die Tiefeninformationen des Kamerabildes pixelgenau vorliegen. In *Abb. 7.15* wurde die Szene aus verschiedenen Kameraperspektiven innerhalb der Bilderwelt

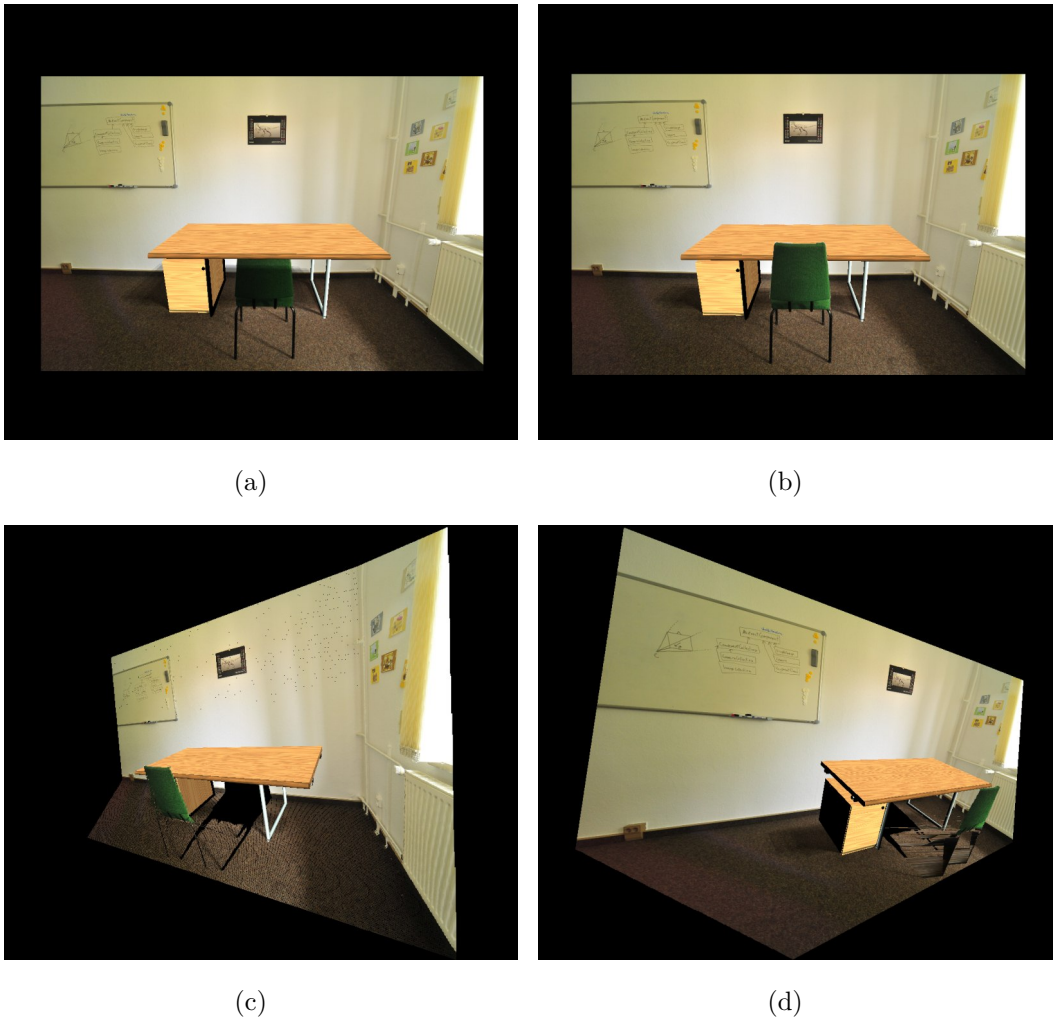


(a) (b)

(c) (d)

(e) (f)

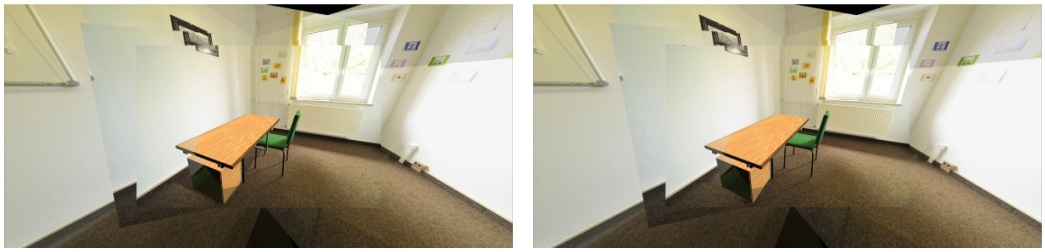
Abbildung 7.13: Szene 1 (siehe *Tab. 7.1*) als (a) Einzelbild: Darstellung eines Sliced-Images bei freier Navigation (anhand der (b) Tiefeninformationen des Geometry-Depth-Matching-Verfahrens): (c) Mit 10 Slices (d) und weißem Rand (ausschließlich aus Visualisierungsgründen). (e) Mit 100 Slices (f) und weißem Rand. Die Abbildungen zeigen, dass mit zunehmender Anzahl an Slices auch die Darstellungsgenauigkeit des Szenenausschnittes zunimmt.



(a) (b)

(c) (d)

Abbildung 7.14: Augmentierte Szene 1 (siehe *Tab. 7.1*) als Einzelbild: (a) Standard-Rendering mit fehlerhafter Verdeckung und (b+c+d) authentische Darstellung und Verdeckung des augmentierten Bildes durch das Sliced-Image-Rendering bei freier Navigation. Als Tiefeninformationen dienten die Depth-Maps aus *Abb. 7.10*.

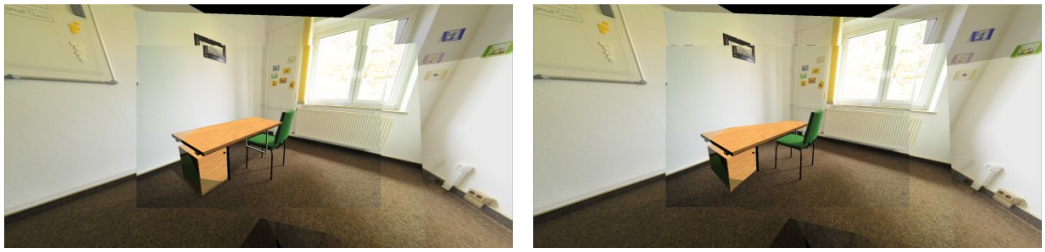


(a) Standard (b) Sliced-Image

(c) Standard (d) Sliced-Image

(e) Standard (f) Sliced-Image

(g) Standard (h) Sliced-Image



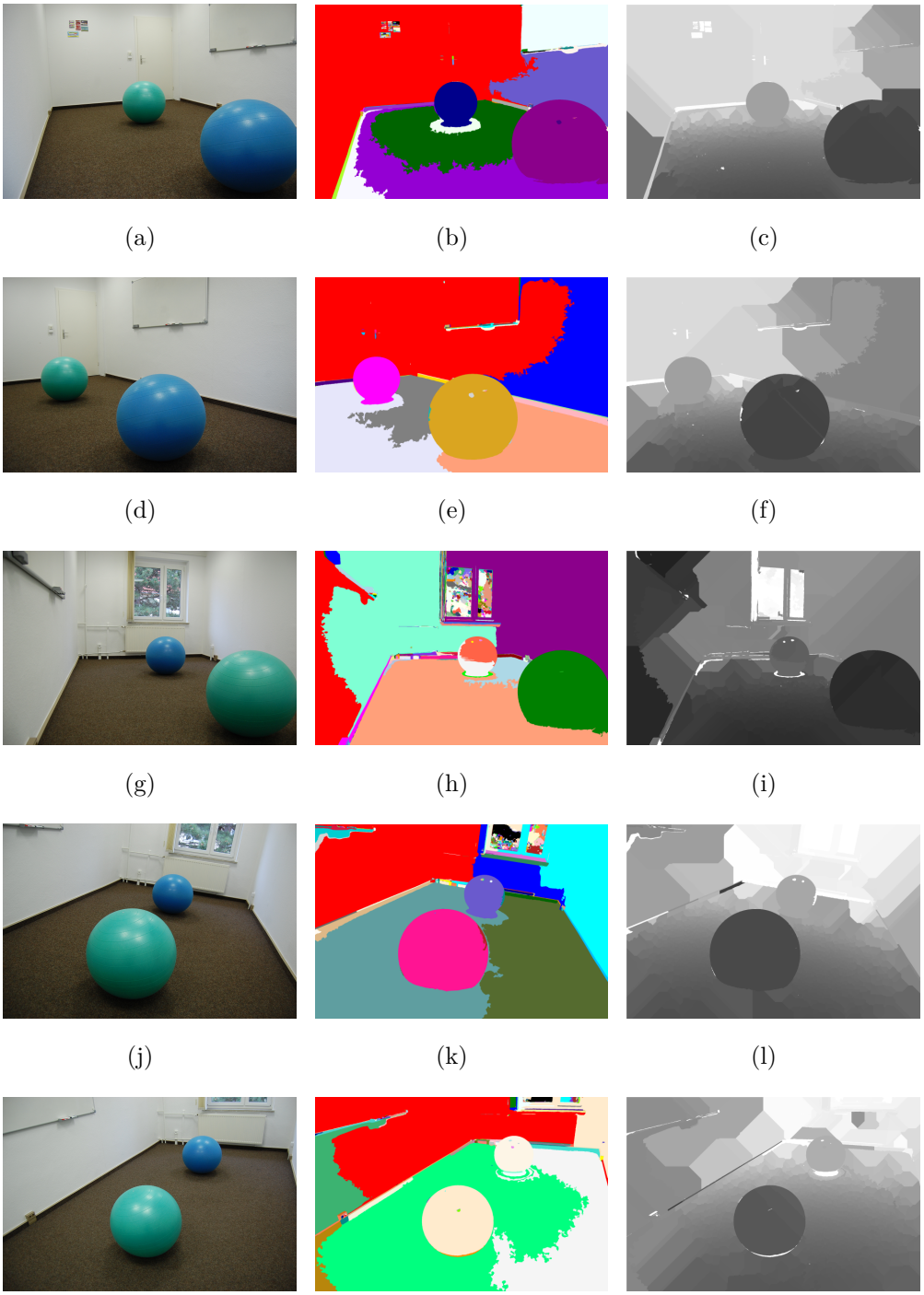
(i) Standard (j) Sliced-Image

Abbildung 7.15: Augmentierte Szene aus *Abb. 7.14* als Bilderwelt: Standard-Rendering (links) und Sliced-Image-Rendering mit authentischer Verdeckung (rechts).

betrachtet und das Standard-Rendering dem Sliced-Image-Rendering gegenübergestellt. Als Tiefeninformationen dienten die Depth-Maps aus *Abb. 7.10*. Während das Standard-Rendering eine fehlerhafte Verdeckung und somit keine authentische Darstellung der augmentierten Bilderwelt bewirkt, realisiert das Sliced-Image-Rendering eine authentische Verdeckung aus unterschiedlichen Perspektiven.

Die nachfolgenden Abbildungen beziehen sich auf Szene 2(b) (siehe *Tab. 7.1*). Diese Innenraumszene weist zwei *Occluder* auf. Zwischen beide Occluder soll ein virtuelles Objekt platziert werden, um die Nutzbarkeit des Sliced-Image-Renderings zu demonstrieren und zu bewerten. In *Abb. 7.16* wurden Depth-Maps mittels des *Segment-Depth-Matching*-Verfahren und der farbbasierten Segmentierung generiert. Diese sind notwendig für das Sliced-Image-Rendering in den nachfolgenden Abbildungen. Es wurden unter Verwendung der gleichen Szene mit mehr Fotos (Szene 2(a)) keine signifikant besseren Ergebnisse bei der Tiefenzuordnung erzielt. Deshalb werden ausschließlich die Depth-Maps der kleineren Szenen (weniger Fotos) verwendet, da dies auch den Anforderungen an diese Arbeit entspricht. In *Abb. 7.17* wird das Kamerabild sowie die augmentierte Variante mit Standard- und Sliced-Image-Rendering gezeigt. Wieder kann das Fazit gezogen werden, dass Standard-Rendering eine fehlerhafte Verdeckung und somit keine authentische Darstellung der augmentierten Bilderwelt bewirkt, während das Sliced-Image-Rendering eine authentische Verdeckung aus unterschiedlichen Perspektiven realisiert. Dies wird ebenfalls in *Abb. 7.18* für die gleiche Szene als Bilderwelt verdeutlicht.

Die nachfolgenden Abbildungen beziehen sich auf Szene 3(b) (siehe *Tab. 7.1*). Ebenfalls wird an dieser Stelle auf das Verwenden der größeren Szene 3(a) verzichtet, da dies keine signifikant besseren Ergebnisse bei der Tiefenzuordnung erzielt. Diese Szene stellt (im Gegensatz zu den bisher verwendeten Szenen 1 und 2) eine Außenszene dar. Die folgenden Abbildungen sollen die Nutzbarkeit des Sliced-Image-Renderings sowie die damit verbundenen Zielstellungen (authentische Darstellung und Verdeckung) auch für solche Szenarien demonstrieren und bewerten. In *Abb. 7.19* wurden Depth-Maps mittels *Segment-Depth-Matching*-Verfahren und farbbasierter Segmentierung von Szene 3 erzeugt. Diese wurden in *Abb. 7.20* für das Sliced-Image-Rendering verwendet. Die Szene wurde mit einem virtuellen Objekt erweitert und die Darstellung des Standard-Renderings



(a) (b) (c)

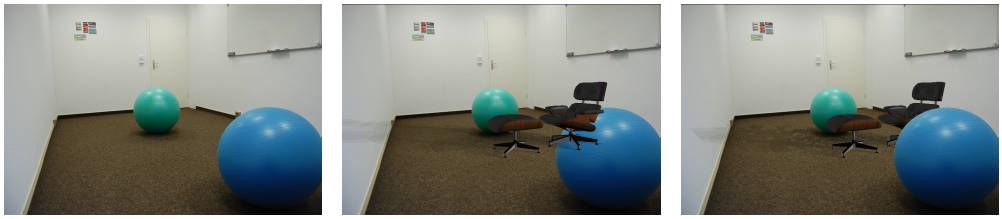
(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m) (n) (o)

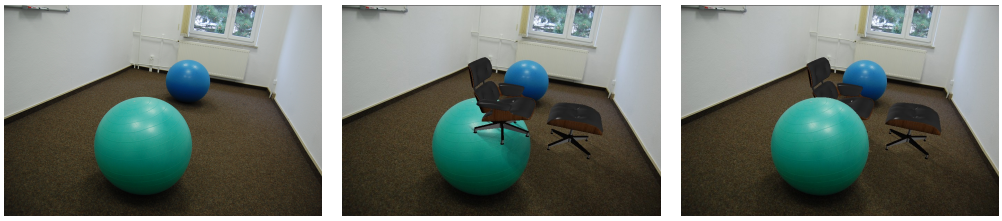
Abbildung 7.16: Erstellung der Depth-Maps (rechte Spalte) mittels Segment-Depth-Matching-Verfahren und farbbasierter Segmentierung (Mitte) von Szene 2(b) (siehe *Tab. 7.1*), notwendig für das Sliced-Image-Rendering in *Abb. 7.17*.



(a) (b) (c)

(d) (e) (f)

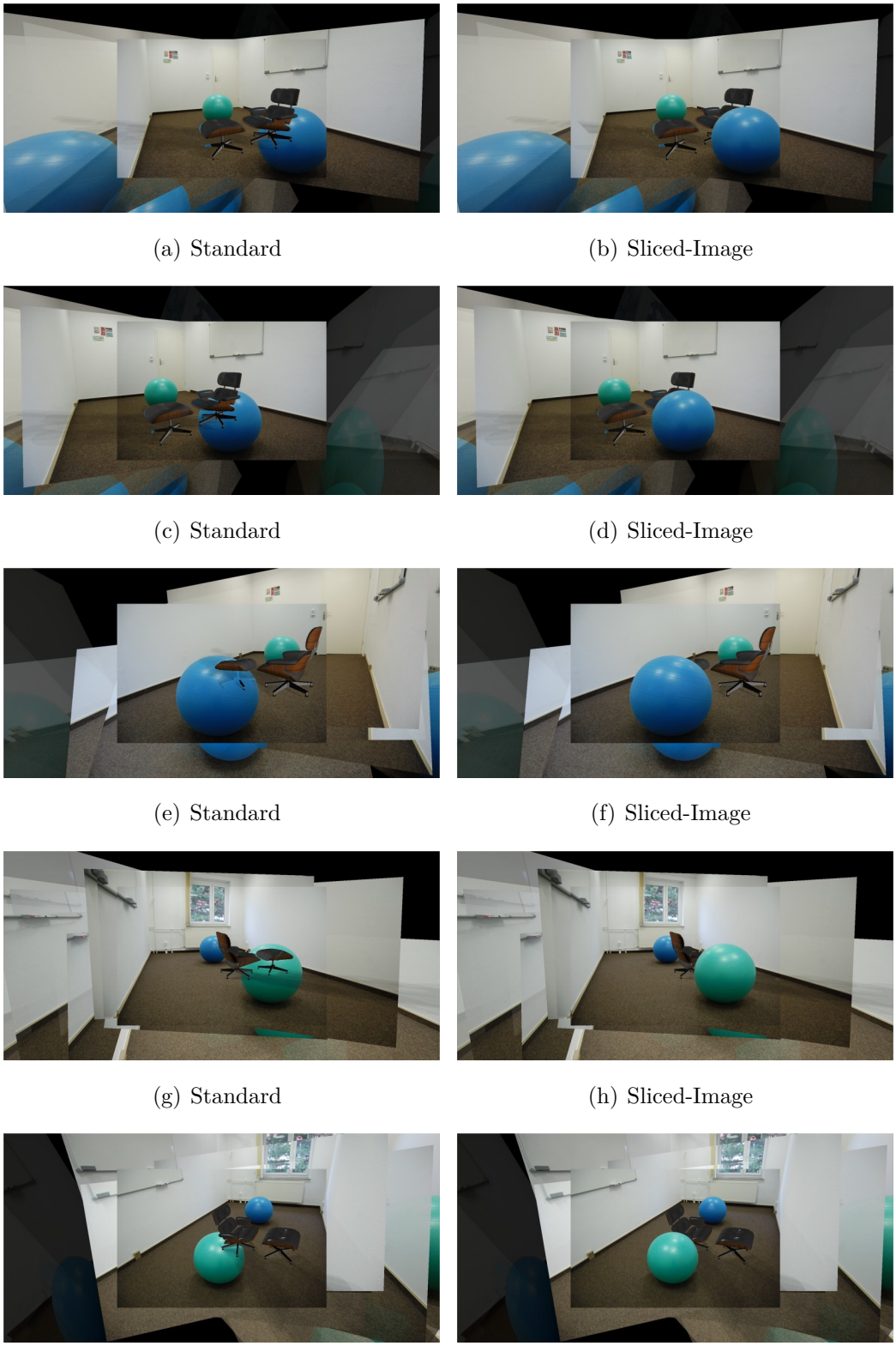
(g) (h) (i)



(j) (k) (l)

(m) (n) (o)

Abbildung 7.17: Augmentierung der Szene 2 (siehe *Tab. 7.1*): (links) Kamerabilder ohne Augmentierung; (mittig) Standard-Rendering mit fehlerhafter Verdeckung und (rechts) Sliced-Image-Rendering mit authentischer Verdeckung des virtuellen Objektes. Als Tiefeninformationen dienten die segmentbasierten Depth-Maps aus *Abb. 7.16*.



(a) Standard (b) Sliced-Image

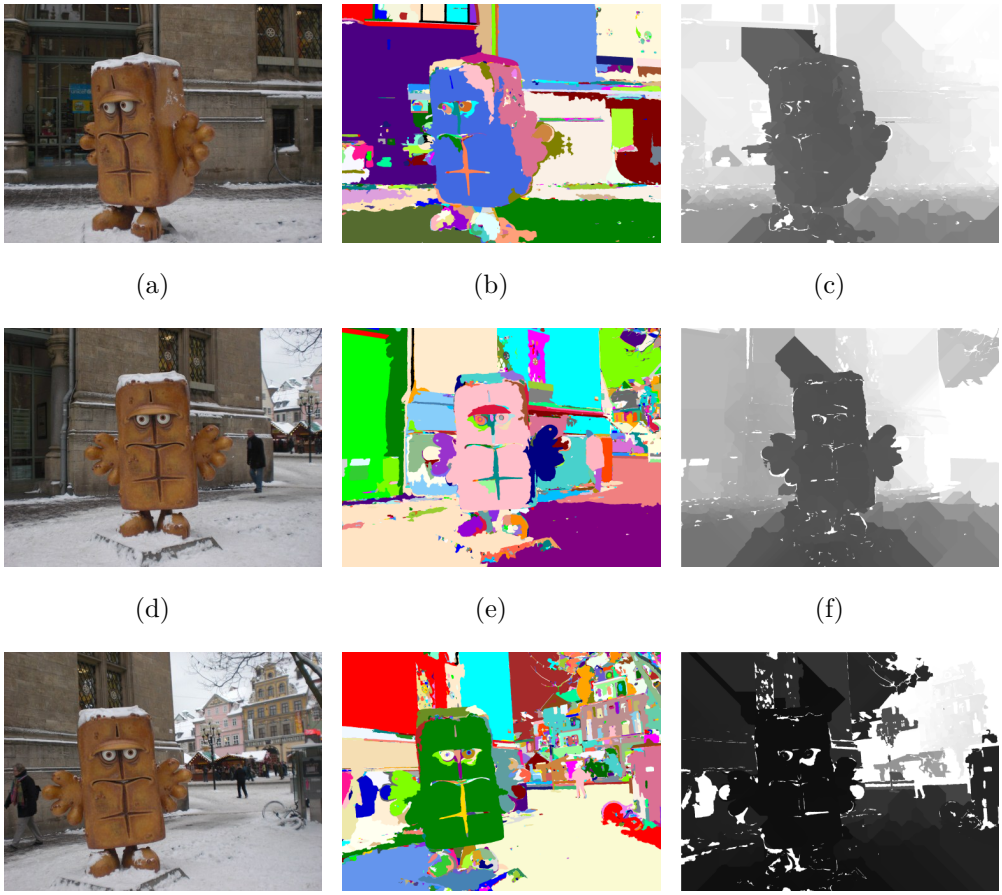
(c) Standard (d) Sliced-Image

(e) Standard (f) Sliced-Image

(g) Standard (h) Sliced-Image

(i) Standard (j) Sliced-Image

Abbildung 7.18: Augmentierte Szene aus *Abb. 7.17* als Bilderwelt: (links) Standard-Rendering und (rechts) Sliced-Image-Rendering mit authentischer Verdeckung.

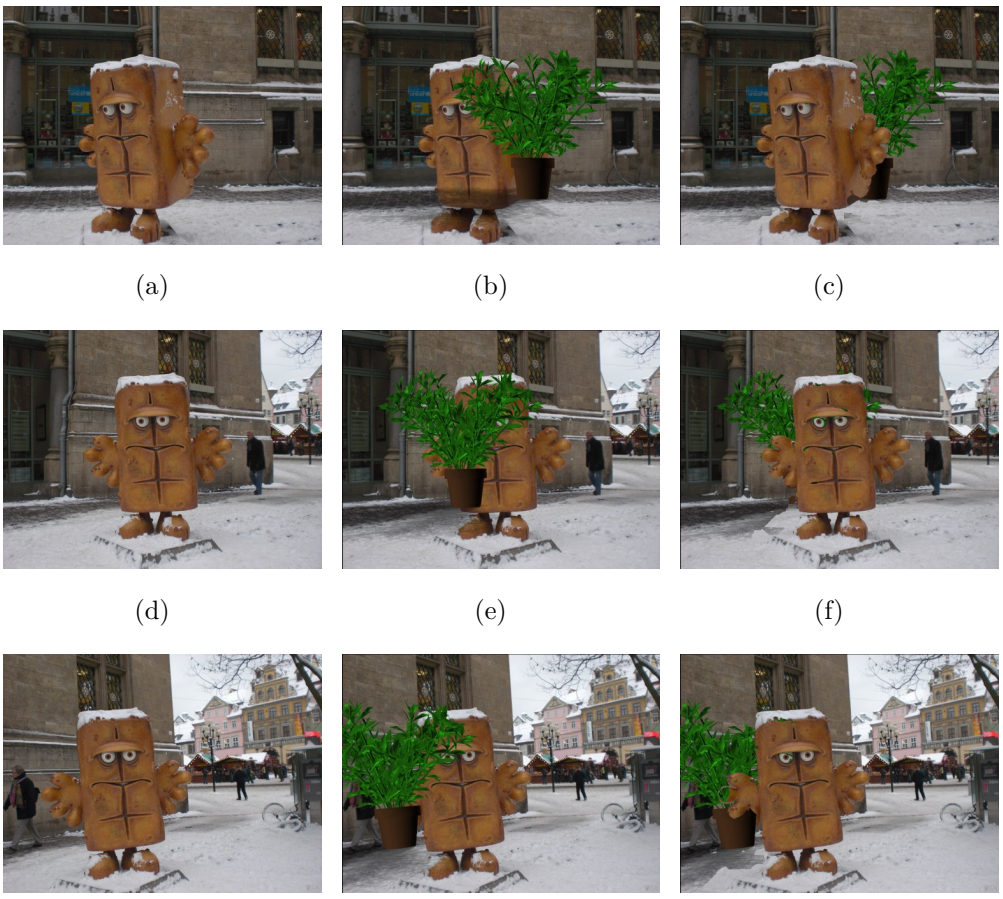


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Abbildung 7.19: Erstellung der Depth-Maps (rechte Spalte) mittels Segment-Depth-Matching-Verfahren und farbbasierter Segmentierung (Mitte) von Szene 3(b) (siehe *Tab. 7.1*), notwendig für das Sliced-Image-Rendering in *Abb. 7.20*.

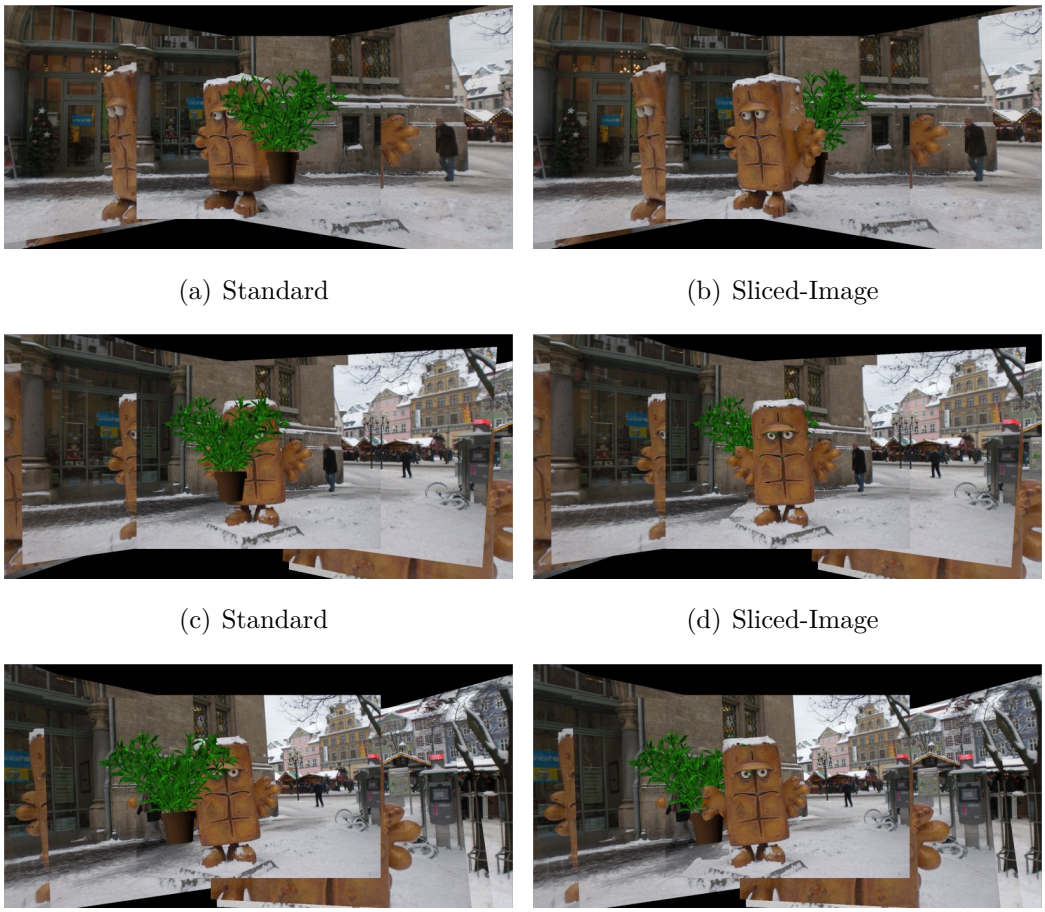


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Abbildung 7.20: Augmentierung der Szene 3 (siehe *Tab. 7.1*): (links) Kamerabilder ohne Augmentierung; (mittig) Standard-Rendering mit fehlerhafter Verdeckung und (rechts) Sliced-Image-Rendering mit authentischer Verdeckung des virtuellen Objektes. Als Tiefeninformationen dienten die segmentbasierten Depth-Maps aus *Abb. 7.19*.

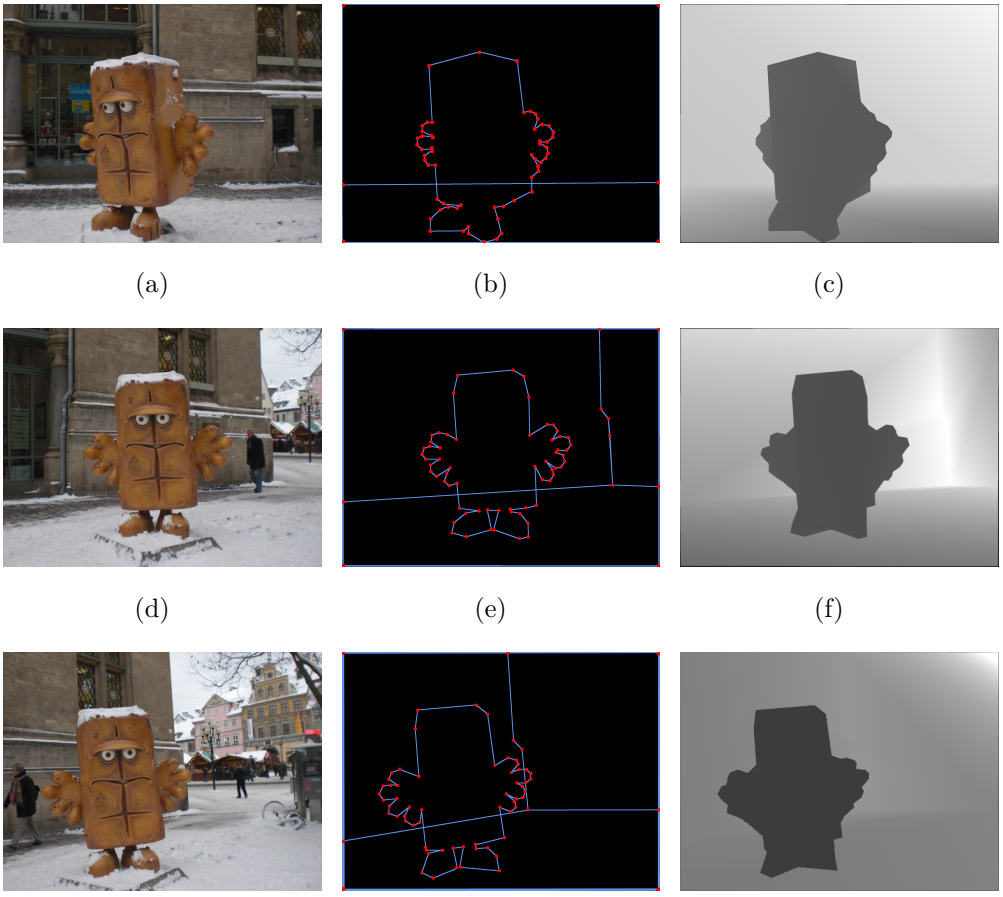


(a) Standard (b) Sliced-Image

(c) Standard (d) Sliced-Image

(e) Standard (f) Sliced-Image

Abbildung 7.21: Augmentierte Szene aus *Abb. 7.20* als Bilderwelt: (links) Standard-Rendering und (rechts) Sliced-Image-Rendering mit authentischer Verdeckung.

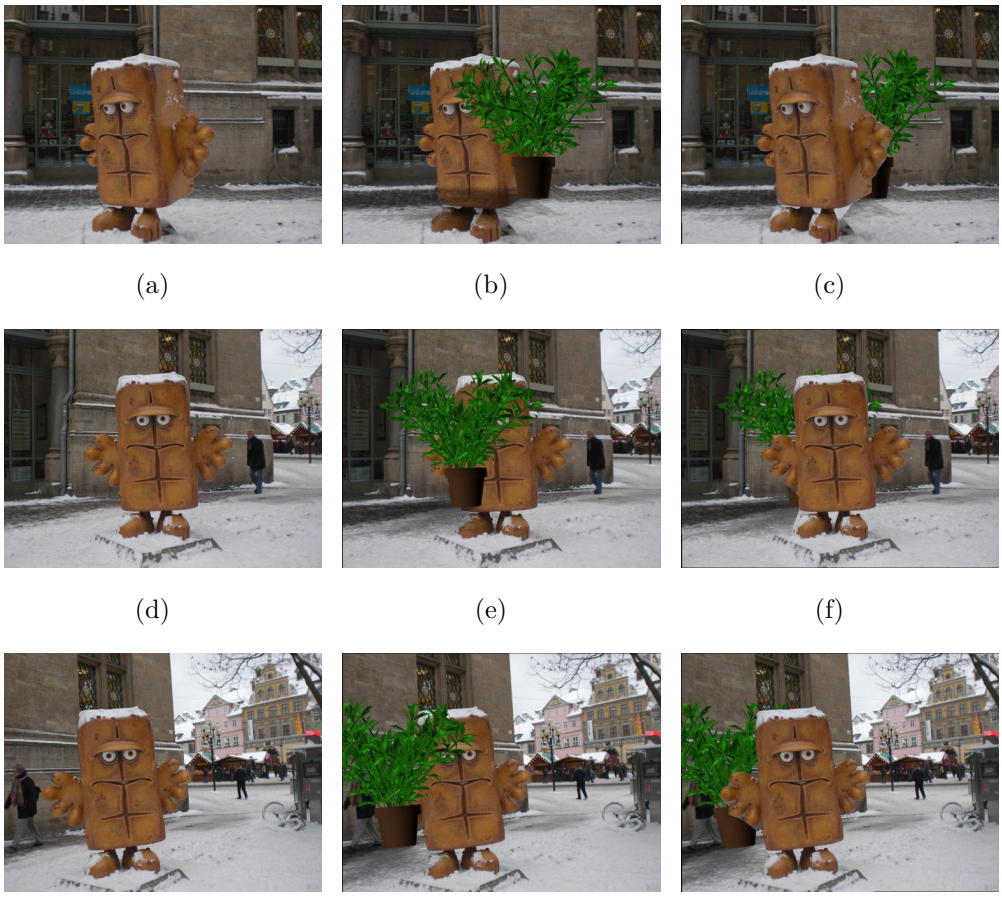


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Abbildung 7.22: Erstellung der Depth-Maps (rechte Spalte) mittels Geometry-Depth-Matching-Verfahren. Die Segmentskizzierungen finden sich in der mittleren Spalte. Eckpunkte der Segmente sind rot eingezeichnet. Die Depth-Maps sind notwendig für das Sliced-Image-Rendering in *Abb. 7.23*.



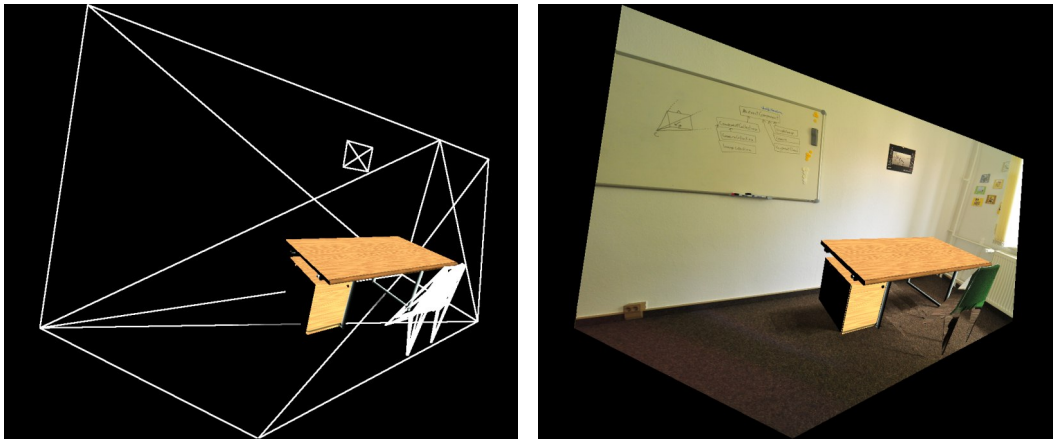
(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Abbildung 7.23: Ähnliche Abbildung wie in *Abb. 7.20* mit dem Unterschied, das als Tiefeninformationen die Depth-Maps des Geometry-Depth-Matching-Verfahrens aus *Abb. 7.22* dienten. Somit kann auch bei freier Navigation eine authentische Augmentierung der Szene 3 (siehe *Tab. 7.1*) erfolgen: (links) Kamerabilder ohne Augmentierung; (mittig) Standard-Rendering mit fehlerhafter Verdeckung und (rechts) Sliced-Image-Rendering mit authentischer Verdeckung des virtuellen Objektes.

dem Sliced-Image-Rendering gegenübergestellt. Auch bei dieser Außenszene zeigt das Standard-Rendering eine fehlerhafte Verdeckung und bewirkt keine authentische Darstellung der augmentierten Bilderwelt. Im Gegenzug dazu realisiert das Sliced-Image-Rendering eine authentische Verdeckung aus unterschiedlichen Kameraperspektiven. *Abb. 7.21* zeigt die gleiche Szene mit den gleichen Ergebnissen als Bilderwelt statt Einzelbilder. Die gezeigten Ergebnisse des Sliced-Image-Renderings der Szenen 2 und 3 verwendeten stets Depth-Maps des Segment-Depth-Matching-Verfahrens. Dies führt dazu, dass (aufgrund der fehlenden Pixelgenauigkeit der Depth-Maps) zwar eine authentische Darstellung und Verdeckung bei der Kameranavigation erfolgreich realisiert wurde, jedoch kommt es zum Teil zu Darstellungsproblemen bei freier Navigation. Dies liegt an den flächengenauen jedoch nicht pixelgenauen *Depth-Patches* der Depth-Map, die aus mehreren Pixeln mit gleichem Tiefenwert bestehen und somit zu Slices als planare Flächen innerhalb der 3D-Bilderwelt führen. Um dies zu umgehen, sind pixelgenaue Tiefeninformationen notwendig. Dafür wurde in *Abb. 7.22* mittels des *Geometry-Depth-Matching*-Verfahrens pixelgenaue Depth-Maps generiert und für das Sliced-Image-Rendering in *Abb. 7.23* verwendet. Aufbauend auf den pixelgenauen Tiefeninformationen und mit einer großen Anzahl an Slices pro Sliced-Image kann eine authentische Darstellung und Verdeckung auch bei freier Navigation erfolgen. Der Vorteil des Sliced-Image-Renderings ist die Realisierung der authentischen Verdeckung bei freier Navigation ohne dabei eine 3D-Szenengeometrie vorauszusetzen, wie in den Abbildungen verdeutlicht wurde. Es werden lediglich Tiefeninformationen des Kamerabildes vorausgesetzt. Folglich kann dieses Image-Based-Rendering-Verfahren auch in Anwendungen verwendet werden, die keinerlei Handling von 3D-Geometrien ermöglichen. Beispielhaft soll an dieser Stelle Google Street View [Goo07] genannt werden. Zwar existieren in dieser Bilderweltenanwendung virtuelle Anteile als 2D-Sprites (z. B. virtuelle Pfeile und Verbindungslinien entlang einer Wegstrecke), jedoch kann Street View keine 3D-Geometrien einbetten. Eine dreidimensionale Darstellung einer Szene wäre dennoch mit dem Sliced-Image-Rendering möglich. Der Nachteil des Sliced-Image-Renderings ist, dass mit abnehmender Korrektheit der Depth-Map auch die Darstellung der Slices an Darstellungsqualität verliert. Des Weiteren können nicht alle schwarzen Bereiche durch den im Konzept erläuterten Optimierungsschritt restlos beseitigt werden.



(a) (b)

Abbildung 7.24: Dreidimensionale Darstellung des Kamerabildes mithilfe (a) des Image-Geometry-Renderings und (b) der 3D-Geometrie des im Kamerabild dargestellten Szenenausschnittes.

7.3.2 Evaluierung des Image-Geometry-Renderings

Das Image-Geometry-Rendering stellt eine Art „Abfallprodukt“ des *Geometry-Depth-Matching*-Verfahrens dar und erfüllt nicht alle Kernanforderungen (siehe Absatz *1.3*) dieser Arbeit. Es wird eine 3D-Geometrie des im Kamerabildes dargestellten Szenenausschnittes vorausgesetzt und die authentische Verdeckung beruht auf dem Darstellen der texturierten 3D-Geometrie. Da es wichtige Kernanforderungen an diese Arbeit nicht erfüllt und nur eine Depth-Matching-Variante verwendet werden kann, kann eine vollständige Evaluierung nicht durchgeführt werden. Dennoch soll kurz auf die Ergebnisse sowie Vor- und Nachteile des Image-Geometry-Renderings eingegangen werden.

In *Abb. 7.24* wird ein Kamerabild augmentiert und mithilfe des Image-Geometry-Renderings eine authentische Darstellung und Verdeckung realisiert. Der Vorteil des Image-Geometry-Renderings ist der geringere Renderaufwand gegenüber dem Sliced-Image-Rendering, da lediglich ein texturiertes 3D-Objekt dargestellt werden muss. Jedoch fordert dieses Verfahren zwingend die Szenengeometrie des in einem Kamerabild abgebildeten Szenenausschnittes. Folglich muss die Bilderweltenanwendung das Handling von 3D-Geometrien unterstützen.

7.4 Grenzen der entwickelten Verfahren

Die Qualität des *Sliced-Image-Renderings* wie auch des *Image-Geometry-Renderings* hängt von den Ergebnissen der vorangehenden Tiefenzuordnung, also von den Depth-Matching-Verfahren ab.

Die Depth-Matching-Verfahren nutzen die Punktwolke bzw. die *Observed-Points* der Bilder innerhalb der Bilderwelt. Da die Erstellung einer Bilderwelt auch aus Fotos mit geringer Auflösung, schlechter Aufnahmequalität und/oder geringer *Bildstruktur* hervorgegangen sein kann, kann eine ausgeprägte Punktwolke bzw. für die Verfahren ausreichende und genaue Observed-Points pro Foto nicht gewährleistet werden. In *Abb. 7.25* wurde eine Bilderwelt aus 13 Fotos mit einer Auflösung von 640x425 Pixel pro Bild verwendet. Die Fotos der Bilderwelt zeigen einen wenig beleuchteten, leeren Raum als Rohbau. Die Wände sowie der Fußboden besitzen einen Braun-Grau-Farbton. Folglich weisen die Bildinhalte wenig Struktur und viele homogene Flächen auf. Bei der Generierung der Bilderwelt entstehen wenig und fehlerhafte/ungenaue Observed-Points. In *Abb. 7.25* (c, e, g) sind die Depth-Maps der einzelnen Verfahren dargestellt und in *Abb. 7.25* (d, f, h) die dazugehörigen Sliced-Images. Zur besseren Visualisierung sind bei den Sliced-Images jene Bildinformationen ausgeblendet (also schwarz), bei denen die Depth-Map keine Tiefeninformationen lieferte. Es wird deutlich, dass die Verfahren aufgrund der ungenauen und wenig ausgeprägten Punktwolke fehlerhafte und ungenaue Ergebnisse erzeugen. Für das verwendete Beispielbild in der Abbildung kann mit allen Verfahren keine authentische Verdeckung innerhalb dieser Bilderwelt realisiert werden.

Schlussfolgernd kann daraus abgeleitet werden, dass die Anforderungen und Kriterien für eine ausgeprägte und genaue Generierung der Punktwolke bzw. der Bilderwelt (siehe Absatz *2.6.2*), analog als Kriterien für die in dieser Arbeit entwickelten Verfahren gelten.

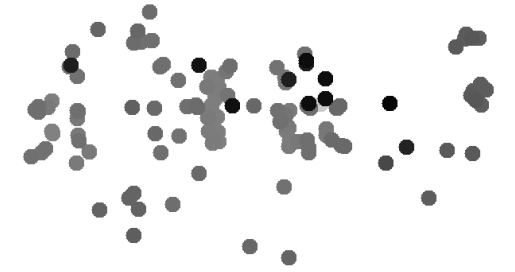
Darüber hinaus soll an dieser Stelle auf die dargestellten Vor- und Nachteile der einzelnen Verfahren in *Tab. 5.4* und *Tab. 5.5* hingewiesen werden, aus denen sich verfahrensspezifische Beschränkungen und Grenzen ergeben. Diese wurden bereits in diesem Kapitel bei der Evaluierung der jeweiligen Verfahren aufgezeigt.

(a) Bild mit seinen Observed-Points

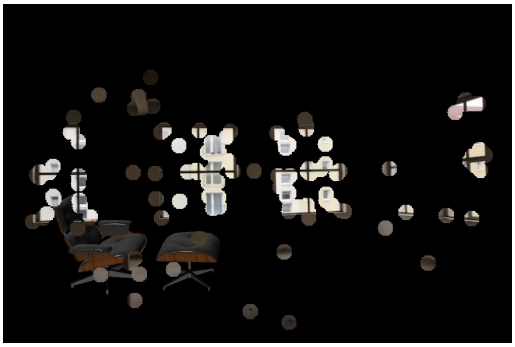
(b) Bild mit eingebetteten virtuellem Objekt

(c) Segment-Depth-Matching-Ergebnis

(d) Sliced-Image, basierend auf (c)



(e) Key-Point-Depth-Matching-Ergebnis



(f) Sliced-Image, basierend auf (e)

(g) Geometry-Depth-Matching-Ergebnis

(h) Sliced-Image, basierend auf (g)

Abbildung 7.25: Fehlerhafte oder unzureichende Ergebnisse der Verfahren.

Symbol	Bedeutung
✓	Kernanforderung oder Teilziel **erfüllt**
x	Kernanforderung oder Teilziel **nicht erfüllt**

Tabelle 7.2: Legende der Symbolbedeutungen nachfolgender Tabellen.

7.5 Evaluierung in Bezug auf die Zielstellung

In Absatz *1.2* wurde das Hauptziel (Authentische Verdeckung in augmentierten Bilderwelten) unter Berücksichtigung der Kernanforderungen (Absatz *1.3*) definiert. Es wurde die *Kameranavigation* und die *Freie Navigation* eingeführt und erläutert. Diese bedingen unterschiedliche Vorgehensweisen bei der Realisierung einer authentischen Verdeckung. Folglich wurden die Teilziele *Authentische Verdeckung bei Kameranavigation* und *Authentische Verdeckung bei freier Navigation* definiert. Dieser Absatz evaluiert, inwiefern das Hauptziel und die Teilziele insgesamt erreicht wurden. In *Tab. 7.2* wird die Legende der verwendeten Symbole für nachfolgende Tabellen dieses Kapitels aufgezeigt.

In Absatz *7.3.1* wurde gezeigt, dass das *Sliced-Image-Rendering* im Rahmen der freien Navigation aus beliebigen Positionen eine augmentierte Bilderwelt ohne visuelle Störungen und mit authentischen Verdeckungen darstellen kann. Hierbei werden keine 3D-Geometriedaten der abgebildeten Szene verwendet und dennoch eine hohe Darstellungsqualität der Verdeckung erreicht. Die Anzahl der Slices pro Sliced-Image muss entsprechend hoch sein, um (fast) keine schwarzen Bereiche zwischen Nachbar-Slices zu verursachen. Der Vollständigkeit halber soll erwähnt werden, dass das *Image-Geometry-Rendering* ebenfalls bei freier Navigation eine hohe Darstellungsqualität der Verdeckung erreicht (siehe Absatz *7.3.2*). Jedoch verwendet das Verfahren Geometriedaten und erfüllt somit nicht alle Kernanforderungen. *Tab. 7.3* zeigt zusammenfassend, inwiefern die Rendering-Verfahren die (für das Rendering relevante) Kernanforderungen erfüllen.

Voraussetzung für eine hohe Darstellungsqualität bei freier Navigation durch das Sliced-Image-Rendering ist eine genaue und vollständige Depth-Map. Nur so kann die abgebildete räumliche Struktur der Bilderweltenszene als *3D-Bilddarstellung* gerendert und somit uneingeschränkt aus beliebigen Positionen eine augmentierte Bilderwelt ohne visuelle Störungen und mit authentischen

Kernanforderungen	Sliced-Image-Rendering	Image-Geometry-Rendering
Verdeckung ohne Geometrie	✓	x
Hohe Darstellungsqualität der Verdeckung	✓	✓

Tabelle 7.3: Übersicht über die Einhaltung der Kernanforderungen durch die Renderverfahren für eine authentische Verdeckung.

Verdeckungen dargestellt werden. Die Evaluierung der einzelnen Verfahren der Tiefenzuordnung finden sich in Absatz *7.2.2*. Das *Segment-Depth-Matching*- sowie das *Key-Point-Depth-Matching*-Verfahren füllen die Depth-Map mit Tiefeninformationen, die sich auf Pixelflächen (Bildsegmente) beziehen. Die Depth-Map ist flächen-, jedoch nicht pixelgenau. Diese Ungenauigkeiten können zu Darstellungsproblemen der Sliced-Images bei freier Navigation führen.

Die Evaluierung des Segment-Depth-Matching-Verfahrens zeigt, dass das Verfahren (fast) vollständige, jedoch nicht pixelgenaue Flächen erstellt, ohne eine dichte, flächendeckende Punktwolke vorauszusetzen und ohne eine vollständige 3D-Rekonstruktion der Szene zu betreiben. Die Ergebnisse zeigen, dass dieses Verfahren, unter Berücksichtigung der Kernanforderungen, mithilfe des Sliced-Image-Renderings eine authentische Verdeckung bei der Kameranavigation (Teilziel eins) realisiert. Bei freier Navigation kann es zu Darstellungsproblemen der Sliced-Images kommen, wenn die Flächen der Depth-Map zu groß, d. h. die Tiefeninformationen zu grob aufgelöst sind. Ist dies nicht der Fall, erfüllt dieses Verfahren auch Teilziel zwei.

Die Evaluierung des Key-Point-Depth-Matching-Verfahrens zeigt, dass es nur eine nutzbare Depth-Map für eine authentische Verdeckung bei der Kameranavigation (Teilziel eins) realisiert, wenn eine dichte, flächendeckende Punktwolke vorhanden ist. Folglich kann es nicht alle Kernanforderungen erfüllen. Je nach Radius-Größe der Sprites und Dichte der Punktwolke, kann dieses Verfahren Teilziel eins und zwei realisieren.

Die Evaluierung des *Geometry-Depth-Matching*-Verfahrens zeigt, dass dieses Verfahren pixelgenaue und vollständige Depth-Maps liefert. Schlussfolgernd erfüllt dieses Verfahren beide Teilziele. Für die Generierung der Depth-Map

Kernanforderungen	Segment-Depth-Matching	Key-Point-Depth-Matching	Geometry-Depth-Matching
Nur unkalibrierte Fotos als Eingabedaten	✓	✓	✓
Keine Stereokameras / Spezialhardware	✓	✓	✓
Kleine Bilderserien (<50)	✓	✓	✓
Keine dichte Punktwolke notwendig	✓	x	✓
Keine vollständige 3D-Rekonstruktion	✓	✓	✓

Tabelle 7.4: Übersicht über die Einhaltung der Kernanforderungen durch die *Depth-Matching*-Verfahren.

erzeugt dieses Verfahren eine 3D-Geometrie des in dem Foto abgebildeten Ausschnittes der Szene. Die Geometrie dient jedoch ausschließlich der Gewinnung pixelgenauer Tiefeninformationen und repräsentiert nur einen Szenenteilausschnitt. Folglich entspricht dies nicht einer vollständigen 3D-Rekonstruktion der gesamten Bilderweltenszene. Dieses Verfahren erfüllt somit die Kernanforderungen.

Tab. 7.4 zeigt zusammenfassend, inwiefern die Depth-Matching-Verfahren die (für die Verfahren relevanten) Kernanforderungen erfüllen.

Die Evaluierung hat gezeigt, dass unter Einhaltung der Kernanforderungen das Segment-Depth-Matching-Verfahren Teilziel eins und z. T. Teilziel zwei erfüllen kann. Das Key-Point-Depth-Matching-Verfahren kann ohne Verletzung der Kernanforderungen (eine dichte Punktwolke muss vorausgesetzt werden) keine authentische Verdeckung ermöglichen. Das Geometry-Depth-Matching kann beide Teilziele erfüllen. Bei den Rendering-Verfahren kann das Sliced-Image-Rendering beide Ziele erfüllen. Das Image-Geometry-Rendering erfordert eine Geometrie und kann somit nicht die Kernanforderungen erfüllen. *Tab. 7.5* zeigt zusammenfassend eine Übersicht der Depth-Matching- und Renderverfahren über das Erreichen der Teilziele unter Berücksichtigung aller Kernanforderungen.

Verfahren	Teilziel 1	Teilziel 2
Segment-Depth-Matching	✓	(✓)
Key-Point-Depth-Matching	x	x
Geometry-Depth-Matching	✓	✓
Sliced-Image-Rendering	✓	✓
Image-Geometry-Rendering	x	x

Tabelle 7.5: Übersicht der Depth-Matching- und Renderverfahren über das Erreichen der Teilziele unter Berücksichtigung aller Kernanforderungen.

7.6 Zusammenfassung und Diskussion

Die *Evaluierung der Bildsegmentierung* in Absatz *7.2.1* zeigte, dass (bewertet an mehreren Szenen, siehe *Tab. 7.1*) die *Farbbasierte Segmentierung* die besten Ergebnisse ermöglichte. Zwar kann die Kombination aus *Farb- und Konturbasierter Segmentierung* eine geringfügige Verbesserung des Segmentierungsergebnisses bewirken, was sich aber nicht in allen Bildserien eindeutig belegen ließ. In einigen Testbildern kam es sogar zu schlechteren Resultaten. Aufgrund dieser Tatsache und dem erhöhten Rechenaufwand (durch die Kombination zweier Segmentierungsverfahren) muss die Farbsegmentierung als Einzelverfahren der Kombination aus farb- und konturbasierter Segmentierung vorgezogen werden. Die *Konturbasierte Segmentierung* als Einzelverfahren eignet sich nicht für Innenraumszenen, sondern eher für Fotos und Abbildungen mit klar abgegrenzten Objekten (z. B. Buchstaben oder Objekte im Fokus des Bildes mit klarer Abgrenzung zu der restlichen Abbildung), wie in *Abb. 5.11* gezeigt wurde.

Die *Evaluierung der Tiefenzuordnung* in Absatz *7.2.2* zeigte, dass das *Segment-Depth-Matching*-Verfahren, aufbauend auf einer vorausgegangenen Bildsegmentierung, unter Berücksichtigung aller gestellten Anforderungen an diese Arbeit (siehe Absatz *4.3* (*Erweiterte Anforderungen*)) erfolgreich eine Tiefenzuordnung bewerkstelligen kann. Hierbei enthält die gewonnene Depth-Map flächen- und nicht *pixelgenaue Depth-Patches*. Das *Key-Point-Depth-Matching*-Verfahren umgeht die Notwendigkeit einer vorausgehenden Bildsegmentierung,

erfordert jedoch eine flächendeckende Punktwolke, die den abgebildeten Szenenausschnitt auf dem *Kamerabild* repräsentiert. Dies steht im Widerspruch zu der Kernanforderung (siehe *Abb. 1.10*), keine dichte Punktwolke vorauszusetzen. Somit kann dieses Verfahren zwar unter speziellen Bedingungen erfolgreich eine Tiefenzuordnung bewerkstelligen, jedoch nicht unter allen gestellten Kernanforderungen an diese Arbeit. Analog zu dem Segment-Depth-Matching-Verfahren generiert das Key-Point-Depth-Matching keine pixel-, sondern flächengenaue Depth-Patches. Das *Geometry-Depth-Matching*-Verfahren ermöglicht, im Gegensatz zu den anderen beiden Depth-Matching-Verfahren, das Generieren einer pixelgenauen Depth-Map. Hierfür ist jedoch ein semiautomatischer Skizzierungsschritt notwendig. Dementsprechend erfolgt die Tiefenzuordnung nicht vollständig automatisch. Die Evaluierung hat gezeigt, dass bei verhältnismäßig genauer Skizzierung (auch mit leichten Ungenauigkeiten) eine pixelgenaue Depth-Map generiert werden kann. Im Rahmen dieses Verfahrens wird eine 3D-Geometrie des im Kamerabildes abgebildeten Szenenausschnittes erzeugt. Es wird jedoch keine vollständige 3D-Rekonstruktion der gesamten Bilderweltenszene durchgeführt. Des Weiteren dient die erstellte Geometrie ausschließlich für das Gewinnen pixelgenauer *Tiefeninformationen* des Kamerabildes. Folgerichtig ist auch die Kernanforderung, auf eine vollständige 3D-Rekonstruktion zu verzichten und die Verdeckung ohne 3D-Geometrie zu realisieren, erfüllt.

Die *Evaluierung des Sliced-Image-Renderings* in Absatz *7.3.1* zeigte, dass trotz fehlender 3D-Geometrie und auch unter Verwendung kleinerer Szenen (wenig Fotos) eine authentische Darstellung und Verdeckung augmentierter Bilderwelten bei *Kameranavigation* sowie bei *freier Navigation* mithilfe des *Sliced-Image-Renderings* ermöglicht werden kann. Hierfür werden lediglich die gewonnenen Tiefeninformationen der Depth-Matching-Verfahren vorausgesetzt. Für eine möglichst detailreiche und genaue Darstellung bei freier Navigation wird eine große Anzahl an Slices benötigt. Dies bedeutet zugleich einen großen Renderaufwand pro Sliced-Image.

Die *Evaluierung des Image-Geometry-Renderings* in Absatz *7.3.2* zeigte, dass auch eine Realisierung der authentischen Darstellung und Verdeckung augmentierter Bilderwelten bei Kameranavigation sowie bei freier Navigation mit geringen Renderaufwand ermöglicht werden kann. Das *Image-Geometry-Rendering* setzte jedoch eine gegebene Szenengeometrie des in einem Kamerabild

abgebildeten Szenenausschnittes voraus. Diese wird im Rahmen der Depth-Map-Generierung des *Geometry-Depth-Matching*-Verfahrens erzeugt. Somit kann die Kernanforderung an diese Arbeit, dass eine authentische Verdeckung ohne Geometrie erfolgen kann mit diesem Verfahren nicht erfüllt werden.

Kapitel 8

Zusammenfassung und Ausblick

8.1 Zusammenfassung

In dieser Arbeit wurde ein Konzept und die dazugehörige technische Realisierung für das authentische Verdecken virtueller 3D-Objekte in *Bilderwelten* entwickelt. Im Rahmen der Augmentierung wird für die Akzeptanz und Glaubwürdigkeit der Darstellung ein *Visuelles Verschmelzen* der realen und virtuellen Anteile erfordert. In die Bilderwelten eingebettete virtuelle Objekte müssen unter Umständen visuell durch einzelne Bildsegmente eines oder mehrerer Fotos überlagert werden. Dieser Umstand wird gefordert, wenn sich virtuelle Objekte hinter anderen abgebildeten Objekten der Szene befinden sollen. Erfolgt bei einer AR-Darstellung eine notwendige Überlagerung nicht, wirkt die Augmentierung nicht authentisch. Die fehlende Überlagerung führt zu einem störenden Eindruck beim Betrachter. Diese Überlagerung wird *Verdeckung* genannt. Eine entsprechend der menschlichen Wahrnehmung korrekte Überlagerung wird als *Authentische Verdeckung* bezeichnet. Für die Realisierung einer authentischen Verdeckung müssen *Tiefeninformationen* aus den Fotos gewonnen werden. Bisherige Arbeiten realisieren hierzu eine 3D-Rekonstruktion der abgebildeten Szene oder erstellen mithilfe von *Bundle-Adjustment*-Algorithmen detailreiche 3D-Punktwolken (Bilderwelten). Beide Varianten setzen eine große Menge an Fotos voraus. Der Ansatz dieser Arbeit setzte als Kernanforderung fest, mit wenigen Fotos und dementsprechend kleinen *Bilderweltenszenen* zu arbeiten und dennoch eine authentische Darstellung und Verdeckung augmentierter Bilderwelten zu ermöglichen. Hierzu generierte ein Depth-Matching-Verfahren zu den

Kamerabildern passende *Depth-Maps*. Folglich lagen die Tiefeninformationen des abgebildeten Szenenausschnittes eines Fotos vor. Darauf aufbauend wurde vom Autor das *Sliced-Image-Rendering* entwickelt. Das Sliced-Image-Rendering stellt ein *Image-Based-Rendering*-Verfahren dar, das ohne 3D-Geometrie ein 2D-Bild anhand gegebener Tiefeninformationen als dreidimensionale Darstellung rendern und auf diese Weise eine authentische Verdeckung auch bei *freier Navigation* ermöglichen kann. Auf Basis gewonnener Tiefeninformationen wird ein Bild innerhalb der Bilderweltenszene in Slices zerlegt. Ein Slice repräsentiert eine Bildregion des Originalbildes. Jedes Slice wird an die für die Region entsprechende Tiefenposition, die in der Depth-Map gespeichert wurde, verschoben. Damit alle Regionen zusammen dennoch den Eindruck eines Bildes vermitteln, müssen die verschobenen Slices perspektivisch korrekt projiziert werden. Die Gesamtheit aller verschobenen, skalierten Regionen wird Sliced-Image genannt. Mithilfe des Sliced-Image-Rendering kann eine authentische Verdeckung virtueller 3D-Objekte in Bilderwelten erfolgen, ohne dafür eine 3D-Rekonstruktion zu betreiben oder eine 3D-Geometrie vorauszusetzen. Hierfür wurde eine Evaluierung mit verschiedenen Testszenen und Fotoserien durchgeführt. Für das Generieren der Tiefeninformationen wurden drei Depth-Matching-Verfahren vom Autor entwickelt:

- *Segment-Depth-Matching*,
- *Key-Point-Depth-Matching*,
- *Geometry-Depth-Matching*.

Das Segment-Depth-Matching-Verfahren setzt eine vorangegangene Bildsegmentierung voraus und verwendet die *Observed-Points* einer Kamera, um den Segmenten eine Tiefe zuzuordnen. Jeder Observed-Point besitzt für das Kamerabild einen *Observed-Pixel*. Über die 2D-Bildpositionen des Observed-Pixel werden alle Observed-Points einem Segment zugeordnet. Als Ergebnis dieser Zuordnung besitzt jedes Segment eine Anzahl von Observed-Points. Die zweite Information, die für die Tiefenzuordnung benötigt wird, ist die *Z-Distanz* (Entfernung des Observed-Points zu seiner Kamera auf der bzw. parallel zur Z-Achse des Kamerakoordinatensystems) zwischen Observed-Point und Kamera. Für jedes Segment wird ein Voronoi-Diagramm erzeugt. Die zugeordneten

Observed-Points (bzw. ihre Observed-Pixel) dienen als Zellkern einer Voronoizelle und somit als Startpunkt für den Voronoi-Prozess. Die Z-Distanzen der Observed-Points dienen als Füllwerte für die Voronoizellen. Als Ergebnis entsteht eine segmentbasierte Depth-Map. Dieses Verfahren bedingt eine vorangegangene Bildsegmentierung. Hierzu wurden vom Autor verschiedene Segmentierungsverfahren entwickelt:

- *Farbbasierte Segmentierung*,
- *Konturbasierte Segmentierung*,
- *Farb- und Konturbasierte Segmentierung*,
- *Objektbasierte Segmentierung*.

Bei der farbbasierten Segmentierung werden zusammenhängende (benachbarte) gleichfarbige Pixel eines Fotos als ein Segment und somit als ein Objekt interpretiert. Aufgrund der Anfälligkeit der farbbasierten Segmentierung gegenüber farbunsatten Fotos sowie dem Verschmelzen abgebildeter, benachbarter Objekte mit ähnlichem Farbton mussten weitere Maßnahmen zur besseren Abgrenzung von Segmenten getroffen werden. Dazu wurde die konturbasierte Segmentierung entwickelt. Hierbei wird die Farbeigenschaft der Objekte nicht beachtet, sondern mithilfe detektierter Kanten ein Segmentbild erzeugt. Hierbei wird ein Segment (ein Objekt) durch zusammengehörige Kanten repräsentiert. Diese Segmentierungsform setzt die Erstellung eines detaillierten Kantenbildes voraus. Sie ist fehleranfällig im Falle von fehlenden Kanteninformationen (z.B. häufig unterbrochene Kanten) eines Objektes. Um dennoch Vorteile beider Segmentierungsverfahren auszunutzen und die Fehleranfälligkeiten der Einzelverfahren zu reduzieren, wurden bei der Farb- und Konturbasierten Segmentierung die Einzelverfahren kombiniert. Hierfür werden im ersten Schritt die Konturen in das Kamerabild eingezeichnet. Anschließend wird die farbbasierte Segmentierung verarbeitet, bei denen die Konturen als zusätzliche Begrenzung während des Segmentierungsvorganges dienen. Die eingezeichneten Kanten sorgen auch bei farbunsatten Bildern sowie bei benachbarten abgebildeten Objekten mit gleichem Farbton für die Unterstützung einer korrekten Abgrenzung bei der Segmentierung. Trotz robuster Eigenschaften der farb-

und konturbasierten Segmentierung können bei Kombination nicht sehr ausgeprägter Helligkeitsverläufe und unsatten Farben im Kamerabild fehlerhafte Segmente entstehen. Des Weiteren weisen die bisher vorgestellten Segmentierungsverfahren keinerlei Zusammenhänge der abgebildeten räumlichen Struktur auf. Die objektbasierte Segmentierung zerlegt mithilfe eines semiautomatischen Schrittes ein Bild in räumlich zusammenhängende Segmente. Hierfür werden unterstützend die Eckpunkte der Segmente durch den Nutzer eingezeichnet. Dieser semiautomatische Schritt wird als *Segmentskizzierung* bezeichnet. Der Vorteil dieses Segmentierungsverfahrens liegt zum einen in der (annähernden) Fehlerfreiheit und zum anderen wird ein Zusammenhang zwischen Segmente und der abgebildeten räumlichen Struktur hergestellt.

Das Segment-Depth-Matching-Verfahren setzt eine Bildsegmentierung voraus. Wenn das Segmentierungsergebnis ungenau und fehlerhaft ist, kann es zu fehlerhaften Depth-Maps kommen. Um auch ohne vorangehende Segmentierung eine Tiefenzuordnung zu ermöglichen, wurde vom Autor das Key-Point-Depth-Matching-Verfahren entwickelt. Hierbei werden die Observed-Points auf die Bildebene als kreisförmige *Sprites* projiziert. Die Z-Distanzen der *Key-Points* werden dabei als Füllwerte für die Pixel der Sprites verwendet. Dabei ist (neben der Dichte der Punktwolke) der Radius der Sprites für die Flächenabdeckung und Genauigkeit der entstehenden Depth-Map verantwortlich. Je größer der Radius ist, desto größer ist die Flächenabdeckung, aber desto ungenauer erscheint die Depth-Map.

Das Segment-Depth-Matching- wie auch das Key-Point-Depth-Matching-Verfahren generieren keine pixelgenauen, sondern flächengenaue Depth-Maps. Für pixelgenaue Depth-Maps wurde vom Autor das Geometry-Depth-Matching-Verfahren entwickelt. Hierzu wird eine Szenengeometrie des abgebildeten Szenenausschnittes eines Fotos erzeugt und dadurch eine pixelgenaue Depth-Map erstellt. Aufbauend auf einer semiautomatischen Segmentskizzierung (Objektbasierte Segmentierung) liegen Segmente und ihre 2D-Eckpunktdaten vor. Aus den Eckpunkten werden Eckpunkt-Tripel gebildet und aus diesen Tripeln werden (anhand der gegebenen Observed-Points) neue Key-Points erzeugt. Die neu gewonnenen 3D-Key-Points werden trianguliert und es entsteht dadurch eine 3D-Geometrie des im Kamerabild abgebildeten Szenenausschnittes.

Als Ergebnis der entwickelten Depth-Matching-Verfahren liegen zu einem Kamerabild die korrespondierenden Tiefeninformationen (in Form einer Depth-Map) vor. Die Tiefeninformationen sind Voraussetzung für das Sliced-Image-Rendering.

Die Qualität des Sliced-Image-Renderings hängt von den Ergebnissen der vorangehenden Tiefenzuordnung, also von den Depth-Matching-Verfahren ab. Die Depth-Matching-Verfahren nutzen die Punktwolke bzw. die Observed-Points der Bilder innerhalb der Bilderwelt. Da die Erstellung einer Bilderwelt auch aus Fotos mit geringer Auflösung, schlechter Aufnahmequalität und/oder geringer *Bildstruktur* hervorgegangen sein kann, kann eine ausgeprägte Punktwolke bzw. für die Verfahren ausreichende und genaue Observed-Points pro Foto nicht gewährleistet werden. Schlussfolgernd kann daraus abgeleitet werden, dass die Anforderungen und Kriterien für eine ausgeprägte und genaue Generierung der Punktwolke bzw. der Bilderwelt, analog als Kriterien für die in dieser Arbeit entwickelten Verfahren gelten. Sind Kriterien, wie Fotos der Szene mit überlappenden Bildanteilen, heterogene Strukturen in den Bildinhalten oder verhältnismäßig gute Aufnahmequalität und Auflösung der Fotos nicht erfüllt, wirkt sich das neben der Qualität der Bilderwelt auf die Qualität der entwickelten Verfahren aus. Ungenaue und wenig ausgeprägte Punktwolken führen zu fehlerhaften und ungenauen Ergebnissen der Depth-Matching-Verfahren.

Anhand einer technischen Umsetzung erfolgte eine Validierung der konzeptionellen Verfahren. Die daraus resultierenden Ergebnisse wurden anhand verschiedener Bilderweltenszenen mit unterschiedlichen Eigenschaften (Außen- und Innenraumszenen, detailreich und -arm, unterschiedliche Bildmengen) evaluiert. Die Evaluierung des Sliced-Image-Renderings zeigt, dass mithilfe unvollständiger Tiefeninformationen der entwickelten Depth-Matching-Verfahren und unter Einhaltung der gestellten Anforderungen (wenig Eingabefotos, kleine Szenen, keine 3D-Rekonstruktion) eine authentische Verdeckung eingebetteter virtueller 3D-Objekte in Bilderwelten realisiert werden kann.

Mithilfe des entwickelten Systems können bildbasierte Anwendungen auch mit kleinen Fotomengen Augmentierungen mit hoher Bildqualität in Bezug auf eine authentische Verdeckung realisieren.

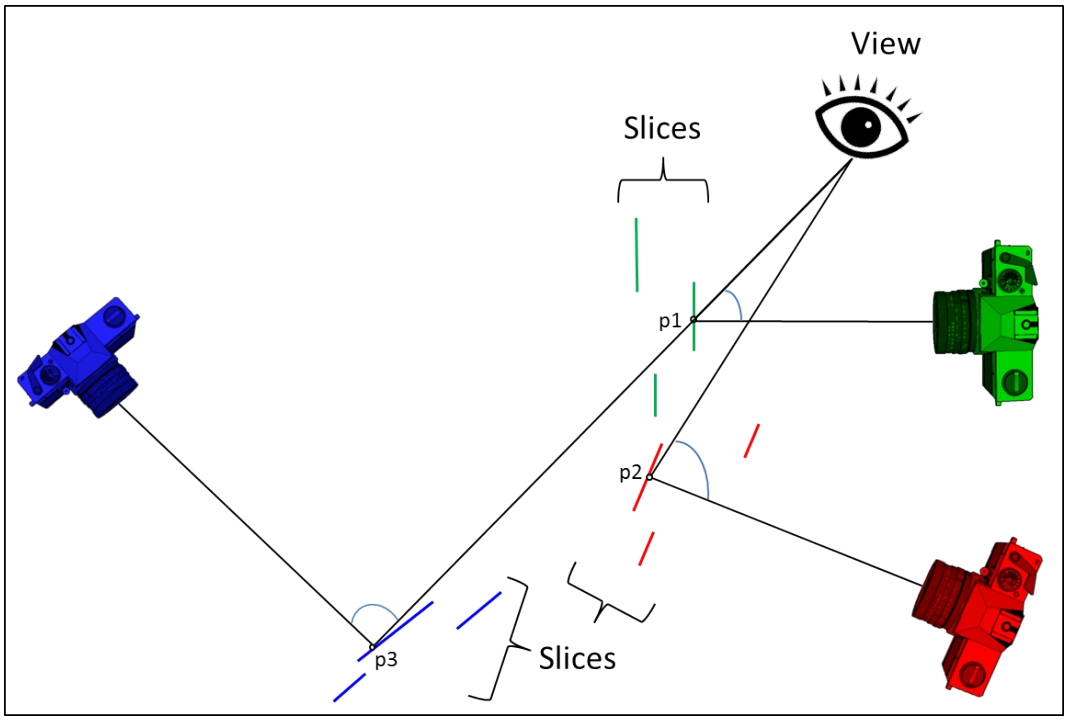


Abbildung 8.1: Ansatz des Pixelized-Image-Renderings als Erweiterung des Sliced-Image-Renderings: Der Winkel zwischen einem Pixel und dem Betrachter priorisiert die Darstellung des Pixels. Je kleiner der Winkel ist, desto höher priorisiert wird das Pixel bewertet.

8.2 Ausblick

Die im Rahmen dieser Arbeit entwickelten Verfahren für das Realisieren einer authentischen Verdeckung integrierter virtueller 3D-Objekte in Bilderwelten behandeln die Fotos (Kamerabilder) einzeln für sich. Dementsprechend werden die Tiefeninformationen der Depth-Matching-Verfahren für jedes Foto unabhängig von anderen Fotos (und abgebildeten Szenenausschnitten) generiert. Hier kann das globale, bildübergreifende Erzeugen von Tiefeninformationen als Ausblick für weiterführende Verfahren gesehen werden. Dafür könnte die globale Punktwolke (anstelle der Observed-Points) verwendet werden.

Des Weiteren arbeiten die Depth-Matching-Verfahren unabhängig voneinander. Sie benötigen unterschiedliche Eingabedaten und produzieren unterschiedliche Ergebnisse. Für eine qualitativ bessere Tiefenzuordnung der abgebildeten Szeneninhalte der Fotos kann der Ansatz der Kombination oder Iteration der einzelnen Depth-Matching-Verfahren gesehen werden. So könnten stets alle Ver-

fahren angewandt und die resultierenden Depth-Maps miteinander verglichen und zu einer neuen Depth-Map interpoliert werden.

Das *Segment-Depth-Matching*-Verfahren setzt eine Bildsegmentierung voraus. Hierfür wurden verschiedene auf Bildanalyse- und Bildverarbeitungsmechanismen aufsetzende Segmentierungsvarianten eingesetzt. Interessant hierbei wäre eine qualitative Verbesserung der Segmentierungsergebnisse durch den Einsatz von Cluster-/Klassifikationsverfahren oder künstlichen neuronalen Netzen. Hierzu müssten unter Umständen Trainingsdaten gewonnen bzw. angelegt werden. Eine Idee wäre, Bilder von Community-Websites (z. B. Flickr [Yah04]) oder sozialen Netzwerken zu extrahieren und diese mittels semiautomatischer *Segmentskizzierung* als Trainingsdaten aufzubereiten.

Analog zu den Depth-Matching-Verfahren verarbeitet auch das *Sliced-Image-Rendering* die einzelnen Kamerabilder für sich. Hier wäre eine Erweiterung des Konzeptes hinsichtlich eines globalen Image-Based-Rendering-Verfahrens, das die Bilderweltenszene auch ohne 3D-Geometrie als eine einzige globale 3D-Abbildung darstellen kann, wünschenswert. Hierzu könnte der Ansatz des Sliced-Image-Renderings in der Form erweitert werden, dass das Rendern der Pixel aller Slices in Abhängigkeit von der Ausrichtung zum Betrachter erfolgt. Diese Erweiterung wird als *Pixelized-Image-Rendering* bezeichnet. Für die Entscheidung, welches Pixel einer Slice dargestellt wird, erfolgt eine Bewertung des Pixels. Der Winkel zwischen dem Sehstrahl einer Kamera zu einem Pixel und dem Sehstrahl des Betrachters zu dem gleichen Pixel priorisiert die Darstellung des Pixels. Je kleiner der Winkel ist, desto höher priorisiert wird das Pixel bewertet. Auf diese Weise könnte eine bildübergreifende globale 3D-Darstellung aller Fotos einer Bilderweltenszene erfolgen. *Abb. 8.1* zeigt das Prinzip von Pixelized-Image-Rendering. In der Abbildung sind zwei Sehstrahlen des Betrachters eingezeichnet. Der erste Sehstrahl trifft auf den Pixel $p1$ einer Slice der grünen Kamera. Zusätzlich trifft er auf Pixel $p3$ einer Slice der blauen Kamera. Da der Winkel von $p1$ kleiner als der Winkel von $p3$ ist, wird $p1$ gezeichnet und $p3$ verworfen. Der zweite Sehstrahl des Betrachters trifft auf $p2$. Da kein konkurrierender Pixel in diesem Beispiel vorhanden ist, wird $p2$ gerendert.

Die an dieser Stelle skizzierten Ideen und Ansätze bieten Raum und Einstiegspunkte für neue Arbeiten. Diese Arbeit kann hierfür als Startpunkt und als Grundgerüst angesehen werden.

Literaturverzeichnis

[Aba06] Abawi, Daniel Farid: *Authentische Integration von virtuellen Objekten in Augmented Reality-Anwendungen.* Doktorarbeit, Johann Wolfgang Goethe-Universität in Frankfurt am Main, 2006.

[AFM+06] Akbarzadeh, A., J.-M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merrell, M. Phelps, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister und M. Pollefeys: *Towards Urban 3D Reconstruction from Video.* In: *Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)*, 3DPVT '06, Seiten 1–8, Washington, DC, USA, 2006. IEEE Computer Society.

[AFS+10] Agarwal, Sameer, Yasutaka Furukawa, Noah Snavely, Brian Curless, Steven M. Seitz und Richard Szeliski: *Reconstructing Rome.* Computer, 43:40–47, 2010.

[AFS+11] Agarwal, Sameer, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz und Richard Szeliski: *Building Rome in a day.* Commun. ACM, 54(10):105–112, Oktober 2011.

[Agr12] Agrawal, Amit: *Results using Hierarchical Framework.* `http://www.cfar.umd.edu/~aagrawal/multi_scale/results.html`, 2012. Letzter Zugriff 08. August 2012.

[AMM+95] Addison, Alonzo C., Douglas MacLeod, Gerald Margolis, Beit Hashoah, Michael Naimark und Hans-Peter Schwarz: *Museums without walls (panel session): new media for*

new museums. In: *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, Seiten 480–481, New York, NY, USA, 1995. ACM.

[Aut12] Autodesk: *123D Catch*. `http://www.123dapp.com/catch`, 2012. Letzter Zugriff 21. Juli 2012.

[AYFC03] Aliaga, Daniel G., Dimah Yanovsky, Thomas Funkhouser und Ingrid Carlbom: *Interactive image-based rendering using feature globalization*. In: *Proceedings of the 2003 symposium on Interactive 3D graphics*, I3D '03, Seiten 163–170, New York, NY, USA, 2003. ACM.

[Azu97] Azuma, Ronald T.: *A Survey of Augmented Reality*. Presence: Teleoperators and Virtual Environments, 6(4):Seiten 355–385, August 1997.

[Bar07] Barthel, Kai Uwe: *Mean Shift Filter*. `http://rsbweb.nih.gov/ij/plugins/mean-shift.html`, 2007. Letzter Zugriff 08. Juli 2012.

[BB06a] Burger, Wilhelm und Mark James Burge: *Digitale Bildverarbeitung. Dateiformate für Bilder*. X. media. press Series. Springer, 2006. Seiten 14–24.

[BB06b] Burger, Wilhelm und Mark James Burge: *Digitale Bildverarbeitung. Farbräume und Farbkonversion*. X. media. press Series. Springer, 2006. Seiten 248–269.

[BB06c] Burger, Wilhelm und Mark James Burge: *Digitale Bildverarbeitung. Filter*. X. media. press Series. Springer, 2006. Seiten 89–116.

[BB06d] Burger, Wilhelm und Mark James Burge: *Digitale Bildverarbeitung. Kanten und Konturen*. X. media. press Series. Springer, 2006. Seiten 117–137.

[BB06e] Burger, Wilhelm und Mark James Burge: *Digitale Bildverarbeitung. Perspektivische Abbildung.* X. media. press Series. Springer, 2006. Seiten 7–8.

[BCN08] Bando, Yosuke, Bing-Yu Chen und Tomoyuki Nishita: *Extracting depth and matte using a color-filtered aperture.* In: *ACM SIGGRAPH Asia 2008 papers*, SIGGRAPH Asia '08, Seiten 134:1–134:9, New York, NY, USA, 2008. ACM.

[BF02] Bimber, Oliver und Bernd Fröhlich: *Occlusion Shadows: Using Projected Light to Generate Realistic Occlusion Effects for View-Dependent Optical See-Through Displays.* In: *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, ISMAR '02, Seiten 186–, Washington, DC, USA, 2002. IEEE Computer Society.

[BG01] Boivin, Samuel und Andre Gagalowicz: *Image-based rendering of diffuse, specular and glossy surfaces from a single image.* In: *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, Seiten 107–116, New York, NY, USA, 2001. ACM.

[BGKN09] Birnbach, Bastian, Paul Grimm, Konrad Kölzer und Frank Nagl: *Automatische Bild- und Darstellungsoptimierung für 3D-Photo-Collections.* In: *6. GI-Workshop "Virtuelle und Erweiterte Realität"der GI-Fachgruppe VR/AR. ISBN 978-3-8322-8662-0*, Seiten 1–11, 2009.

[BL05] Brown, M. und D. G. Lowe: *Unsupervised 3D Object Recognition and Reconstruction in Unordered Datasets.* In: *Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling*, 3DIM '05, Seiten 56–63, Washington, DC, USA, 2005. IEEE Computer Society.

[BNS04] Bergmann, Ludwig, Heinz Niedrig und Clemens Schaefer: *Lehrbuch der Experimentalphysik Band 3: Optik. Kapitel 6 Farbmetrik.* Walter de Gruyter, 2004.

[Can86] Canny, John: *A Computational Approach to Edge Detection.* IEEE Trans. Pattern Anal. Mach. Intell., 8(6):679–698, 1986.

[Cat74] Catmull, Edwin Earl: *A subdivision algorithm for computer display of curved surfaces.* Doktorarbeit, The University of Utah, 1974. AAI7504786.

[CM04a] Chanda, Bhabatosh und Dwijesh Dutta Majumder: *Digital Image Processing and Analysis. Edge and Line Detection.* Prentice-Hall of India Pvt.Ltd, 2 2004. Seiten 239–277.

[CM04b] Chanda, Bhabatosh und Dwijesh Dutta Majumder: *Digital Image Processing and Analysis. Perception of Colour.* Prentice-Hall of India Pvt.Ltd, 2 2004. Seiten 54–57.

[CM04c] Chanda, Bhabatosh und Dwijesh Dutta Majumder: *Digital Image Processing and Analysis. Segmentation.* Prentice-Hall of India Pvt.Ltd, 2 2004. Seiten 217–238.

[Col06] Collins, Robert: *Introduction to Mean-Shift.* `http://www.cse.psu.edu/~rcollins/CSE598G/introMeanShift.pdf`, 2006.

[CSS+10] Catanzaro, Bryan, B.Y. Su, Narayanan Sundaram, Yunsup Lee, Mark Murphy und Kurt Keutzer: *Efficient, high-quality image contour detection.* In: *Computer Vision, 2009 IEEE 12th International Conference on*, Seiten 2381–2388. IEEE, 2010.

[Deb98] Debevec, Paul: *Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography.* In: *SIGGRAPH98*, 1998.

[Deb05] Debevec, Paul: *A Median Cut Algorithm for Light Probe Sampling.* In: *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*, Seiten 430–433. Morgan-Kaufmann, 2005. SIGGRAPH 2005 Poster.

[DIN09] DIN5033: *Farbmessung - Teil 1: Grundbegriffe der Farbmetrik*, 2009.

[dlC11] CIE, Commission Internationale de l'Éclairage: *Farbmetrik - Teil 4: CIE 1976 L*a*b* Farbenraum (ISO 11664-4:2008); German version EN ISO 11664-4:2011*. http://www.beuth.de/de/norm/din-en-iso-11664-4/140465806?SearchID=379298366, 2011. Letzter Zugriff 15. März 2012.

[DTC04] Dick, A. R., P. H. S. Torr und R. Cipolla: *Modelling and Interpretation of Architecture from Several Images*. Int. J. Comput. Vision, 60(2):111–134, November 2004.

[DTM96] Debevec, Paul E., Camillo J. Taylor und Jitendra Malik: *Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach*. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, Seiten 11–20, New York, NY, USA, 1996. ACM.

[dW11] Wissenschaft, Spektrum der: *Abgescannt*. Gehirn und Geist, 4:46, 2011.

[Eas11] EasternGraphics: *pCon.planner 6*. http://www.easterngraphics.com, 2011. Letzter Zugriff 23. Dezember 2011.

[EN05] Egan, Kevin und Ivan Neulander: *Ray tracing depth maps using precomputed edge tables*. In: *ACM SIGGRAPH 2005 Sketches*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[FBS04] Fischer, Jan, Dirk Bartz und Wolfgang Strasser: *Occlusion handling for medical augmented reality using a volumetric phantom model*. In: *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '04, Seiten 174–177, New York, NY, USA, 2004. ACM.

[FCSS09a] Furukawa, Yasutaka, Brian Curless, Steven M. Seitz und Richard Szeliski: *Manhattan-world stereo*. In: *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2009.

[FCSS09b] Furukawa, Yasutaka, Brian Curless, Steven M. Seitz und Richard Szeliski: *Reconstructing Building Interiors from Images*. In: *Proceedings of International Conference on Computer Vision (ICCV)*, 2009.

[FCSS10] Furukawa, Yasutaka, Brian Curless, Steven M. Seitz und Richard Szeliski: *Towards Internet-scale Multi-view Stereo*. In: *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2010.

[FHS07] Fischer, Jan, Benjamin Huhle und Andreas Schilling: *Using Time-of-Flight Range Data for Occlusion Handling in Augmented Reality*. In: *IPT/EGVE'07*, Seiten 109–116, 2007.

[FP05] Furukawa, Yasutaka und Jean Ponce: *Carved visual hulls for high-accuracy image-based modeling*. In: *ACM SIGGRAPH 2005 Sketches*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[FP09] Furukawa, Yasutaka und Jean Ponce: *Carved Visual Hulls for Image-Based Modeling*. Int. J. Comput. Vision, 81(1):53–67, Januar 2009.

[FP10] Furukawa, Yasutaka und Jean Ponce: *Accurate, Dense, and Robust Multiview Stereopsis*. IEEE Trans. Pattern Anal. Mach. Intell., 32(8):1362–1376, August 2010.

[GAF+10a] Goesele, Michael, Jens Ackermann, Simon Fuhrmann, Carsten Haubold, Ronny Klowsky und TU Darmstadt: *Ambient point clouds for view interpolation*. ACM Trans. Graph., 29:95:1–95:6, July 2010.

[GAF+10b] Goesele, Michael, Jens Ackermann, Simon Fuhrmann, Carsten Haubold, Ronny Klowsky, Drew Steedly und Richard Szeliski: *Ambient point clouds for view interpolation*. In: *ACM SIGGRAPH 2010 papers*, SIGGRAPH '10, Seiten 95:1–95:6, New York, NY, USA, 2010. ACM.

[GNA10] Grimm, Paul, Frank Nagl und Daniel Abawi: *IP3D - A Component-based Architecture for Image-based 3D Applications*. In:

SEARIS@IEEEVR2010 Proceedings, IEEE VR 2010 Workshop. ISBN 978-3-8322-8989.8, Seiten 47–52, 2010.

[Goo07] Google: *Street View.* `http://www.google.de/streetview`, 2007. Letzter Zugriff 02. Februar 2012.

[GPW03] Grigorescu, C., N. Petkov und M.A. Westenberg: *Contour detection operators based on surround inhibition.* In: *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, Band 3, Seiten 3–6. IEEE, 2003.

[Gre11] Green, Bill: *Canny Edge Detection Tutorial.* `http://www.pages.drexel.edu/~weg22/can_tut.html`, 2011. Letzter Zugriff 01. April 2011.

[GS06] Grady, Leo und Eric L Schwartz: *Isoperimetric graph partitioning for image segmentation.* IEEE transactions on pattern analysis and machine intelligence, 28(3):469–75, März 2006.

[GSC+07] Goesele, Michael, Noah Snavely, Brian Curless, Hugues Hoppe und Steven M. Seitz: *Multi-View Stereo for Community Photo Collections.* In: *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2007.

[HN12] Has, Michael und Todd Newman: *Color Management: Current Practice and The Adoption of a New Standard.* `http://www.color.org/wpaper1.xalter`, 2012. Letzter Zugriff 16. März 2012.

[Hof09] Hofmann, Tim: *High Dynamic Range Image Encodings.* `http://www.mi.hs-rm.de/~schwan/Projects/CG/CarreraCV/doku/intrinsisch/intrinsisch.htm`, 2009. Letzter Zugriff 08. August 2012.

[IEC99] IEC: *61966-2-1: Multimedia systems and equipment - Colour measurement and management - Part 2-1: Colour management - Default RGB colour space - sRGB*, 1999. http://www.iec.ch.

[IKE12] IKEA: *IKEA Planner*. http://www.ikea.com/ms/en_US/rooms_ideas/splashplanners_new.html, 2012. Letzter Zugriff 23. Dezember 2011.

[ISO96] ISO13655: *Graphic technology – Spectral measurement and colorimetric computation for graphic arts images*, 1996.

[Jäh05] Jähne, Bernd: *Digitale Bildverarbeitung*, Band 6., überarbeitete und erweiterte Auflage. Springer Verlag, 2005. Seiten 350ff.

[KG12] KG, SHD Kreative Planungs-Systeme GmbH & Co.: *KPS Click & design – Fotoplaner und Design-Software in einer Lösung*. http://www.kps-projects.de/planung-konfiguration/online-planung/kps-click-design/, 2012. Letzter Zugriff 21. Juli 2012.

[Kir10] Kirillov, Andrew: *AForge.NET*. http://aforgenet.com, 2010. Letzter Zugriff 23. Juni 2012.

[Kli01] Klix, Wolf-Dieter: *Konstruktive Geometrie: darstellend und analytisch*. Fachbuchverlag Leipzig, 2001. Seiten 38ff.

[KLT07] Kang, Sing Bing, Yin Li und Xin Tong: *Image-Based Rendering*. Now Publishers Inc., Hanover, MA, USA, 2007.

[KLTS06] Kang, Sing Bing, Yin Li, Xin Tong und Heung-Yeung Shum: *Image-based rendering*. Found. Trends. Comput. Graph. Vis., 2(3):173–258, Januar 2006.

[KNG11] Kölzer, Konrad, Frank Nagl und Paul Grimm: *Synthesizing Relative Radiance for Realistic Rendering of Virtual Objects in 3D Photo Collections*. In: *Eurographics 2011 - Areas Papers. ISSN 1017-4656*, Seiten 71–72. Eurographics Association, 2011.

[KNGM12] Kölzer, Konrad, Frank Nagl, Paul Grimm und Stefan Müller: *Rendering Image-Based Scenes in Real-Time Utilizing Sparse Scene Geometry*. In: *Proceedings of Computer Graphics International (CGI'2012)*, 2012.

[Kra04] Kraus, Karl: *Photogrammetrie. 1. Geometrische Informationen aus Photographien und Laserscanneraufnahmen. Bündelblockausgleichung.* De Gruyter Lehrbuch. Walter de Gruyter, 2004. Seiten 299–306.

[Kru08] Kruse, Rolf: *Eigene Fotografien*, 2008. Invirt GmbH.

[KS04] Kang, Sing Bing und Richard Szeliski: *Extracting View-Dependent Depth Maps from a Collection of Images.* Int. J. Comput. Vision, 58(2):139–163, Juli 2004.

[KSS+07] Korfiatis, Panayiotis, Spyros Skiadopoulos, Philippos Sakellaropoulos, Christina Kalogeropoulou und Lena Costaridou: *Automated 3D segmentation of lung fields in thin slice CT exploiting wavelet preprocessing.* In: *Proceedings of the 12th international conference on Computer analysis of images and patterns*, CAIP'07, Seiten 237–244, Berlin, Heidelberg, 2007. Springer-Verlag.

[Lab12] Laboratories, Telekom Innovations: *Happy Measure.* `http://www.developergarden.com/apis/HappyMeasure`, 2012. Letzter Zugriff 21. Juli 2012.

[LB00] Lepetit, Vincent und Marie-Odile Berger: *A Semi-Automatic Method for Resolving Occlusion in Augmented Reality.* In: *CVPR'00*, Seiten 2225–2230, 2000.

[LBLM08] Losson, Olivier, Claudine Botte-Lecocq und Ludovic Macaire: *Fuzzy Mode Enhancement and Detection for Color Image Segmentation.* EURASIP Journal on Image and Video Processing, 2008:1–19, 2008.

[Lin07] Lindbloom, Bruce Justin: *Useful Color Equations.* `http://www.brucelindbloom.com/Math.html`, 2007. Letzter Zugriff 08. Juli 2012.

[Lip80a] Lippman, Andrew: *Movie-maps: An application of the optical videodisc to computer graphics.* In: *Proceedings of the 7th annu-*

al conference on Computer graphics and interactive techniques, SIGGRAPH '80, Seiten 32–42, New York, NY, USA, 1980. ACM.

[Lip80b] Lippman, Andrew: *Movie-maps: An application of the optical videodisc to computer graphics.* SIGGRAPH Comput. Graph., 14(3):32–42, Juli 1980.

[Low04] Lowe, David G.: *Distinctive Image Features from Scale-Invariant Keypoints.* In: *International Journal of Computer Vision*, Band 60, Seiten 91–110, Hingham, MA, USA, 2004. Kluwer Academic Publishers.

[MAFM08] Maire, Michael, P. Arbeláez, Charless Fowlkes und Jitendra Malik: *Using contours to detect and localize junctions in natural images.* In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, Seiten 1–8. IEEE, 2008.

[Men11] Menard, Michelle: *Game Development with Unity. Building Height Using a Heightmap.* Course Technology Ptr, 2011. Seiten 66–69.

[Met12] Metaio: *KPS Click & design – Fotoplaner und Design-Software in einer LösungMetaio Creator.* `http://www.metaio.de`, 2012. Letzter Zugriff 21. Juli 2012.

[MFM04] Martin, D.R., C.C. Fowlkes und Jitendra Malik: *Learning to detect natural image boundaries using local brightness, color, and texture cues.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 26(5):530–549, 2004.

[MH80] Marr, David und Ellen Catherine Hildreth: *Theory of Edge Detection.* In: *Proceedings of the Royal Society of London*, Band 207 der Reihe *B*, Seiten 187–217, 1980. Number 1167.

[Mic08] Microsoft: *Photosynth.* `http://photosynth.net/`, 2008. Letzter Zugriff 02. Februar 2012.

[Mic10a] Microsoft: *Visual C#.* `http://msdn.microsoft.com/de-de/vcsharp/`, 2010. Letzter Zugriff 23. Juni 2012.

[Mic10b] Microsoft: *XNA Framework 4.0*. http://www.microsoft.com/en-us/download/details.aspx?id=23714, 2010. Letzter Zugriff 23. Juni 2012.

[MM12] MSDN und Microsoft: *Vector3.Dot Method*. http://msdn.microsoft.com/en-us/library/bb196388.aspx, 2012. Letzter Zugriff 11. Juni 2012.

[MNBN07] Moreno-Noguer, Francesc, Peter N. Belhumeur und Shree K. Nayar: *Active refocusing of images and videos*. In: *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[MTUK94] Milgram, Paul, Haruo Takemura, Akira Utsumi und Fumio Kishino: *Augmented Reality: A class of displays on the reality-virtuality continuum*. In: *Proceedings of Telemanipulator and Telepresence Technologies*, Seiten 282–292, 1994.

[NB12] Nagl, Frank und Tobias Bindel: *Shader-Based-Image-Processor (SBIP) - A Framework for real-time capable Image and Video Processing*. http://code.google.com/p/sbip/, 2012. Letzter Zugriff 23. Juni 2012.

[NBKG09] Nagl, Frank, Bastian Birnbach, Konrad Kölzer und Paul Grimm: *Perspektivisch korrektes Einblenden virtueller 3D-Objekte in Einzelfotos*. In: *6. GI-Workshop "Virtuelle und Erweiterte Realität"der GI-Fachgruppe VR/AR. ISBN 978-3-8322-8662-0*, Seiten 151–164, 2009.

[NC12] Nóbrega, Rui und Nuno Correia: *Magnetic augmented reality: virtual objects in your space*. In: *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, Seiten 332–335, New York, NY, USA, 2012. ACM.

[NDWV12] Naimark, Michael, Paul Debevec, John Woodfill und Leo Villareal: *The Immersion '94 project*. http://www.pauldebevec.com/Immersion/, 2012. Letzter Zugriff 21. Juli 2012.

[NFH07] Nischwitz, Alfred, Max Fischer und Peter Haberäcker: *Computergrafik und Bildverarbeitung. Alles für Studium und Praxis - Bildverarbeitungswerkzeuge, Beispiel-Software und interaktive Vorlesungen online verfügbar.*, Band 2., verbesserte und erweiterte Auflage. Vieweg+Teubner Verlag, 2007. Seiten 438ff.

[NFST10] Nagl, Frank, Martin Friedl, Andreas Schäfer und Andre Tschentscher: *Shader-Based-Image-Processor - Ein Framework für echtzeitfähige, grafikkartenbasierte Bilderverarbeitung.* In: *GI Seminars 9 Informatiktage. ISBN 978-3-88579-443-1*, Seiten 223–226, 2010.

[NGBA10] Nagl, Frank, Paul Grimm, Bastian Birnbach und Daniel Abawi: *PoP-EYE environment: Mixed Reality using 3D Photo Collections.* In: *ISMAR 2010 Poster Proceedings. ISBN 978-1-4244-9344-9*, Seiten 255–256, 2010.

[NKG11a] Nagl, Frank, Konrad Kölzer und Paul Grimm: *SBIP - A Framework for Real-time Capable Image Processing in Videos.* In: *MIET - Modern Informations and Electronic Technologies 2011. 12th International scientific-functional conference*, Proceedings, Section 2: Information Technologies and Artificial Intelligence, Seite 90, Odessa, 2011. Politechperiodika.

[NKG+11b] Nagl, Frank, Konrad Kölzer, Paul Grimm, Tobias Bindel und Stephan Rothe: *ConGrap – Contour Detection based on Gradient Map of Images.* In: Hamid R. Arabnia, Leonidas Deligiannidis, Gerald Schaefer (Herausgeber): *Proceedings of IPCV2011*, Band II, Seiten 870–875. CSREA Press, USA, 2011.

[NKG11c] Nagl, Frank, Konrad Kölzer und Paul Grimm: *Occlusion Handling and Image-based Lighting using Sliced Images in 3D Photo Collections.* In: *Proceedings of 21st International Conference on Artificial Reality and Telexistence (ICAT2011). ISSN: 1345-1278*, Seiten 118–125, 2011.

[NKG11d] Nagl, Frank, Konrad Kölzer und Paul Grimm: *Occlusion Handling and Image-based Lighting using Sliced Images in 3D*

Photo Collections. In: *Proceedings of 21st International Conference on Artificial Reality and Telexistence (ICAT2011). ISSN: 1345-1278*, Seiten 118–125, 2011.

[NKG12] Nagl, Frank, Konrad Kölzer und Paul Grimm: *popeye - Authentic 3D visualizations for planning data in image-based 3D worlds*. `http://code.google.com/p/popeye/`, 2012. Letzter Zugriff 22. Juni 2012.

[NS10] Nagl, Frank und Sebastian Schäfer: *3D-SurReAL - 3D Surface Reconstruction of Arbitrary (Image) Layers*. In: *2nd Eurographics Workshop on Visual Computing for Biology and Medicine (VCBM) - Poster*, 2010.

[PCPN07] Papari, Giuseppe, Patrizio Campisi, Nicolai Petkov und Alessandro Neri: *A Biologically Motivated Multiresolution Approach to Contour Detection*. EURASIP Journal on Advances in Signal Processing, 2007:1–29, 2007.

[PLF07] Pilet, Julien, Vincent Lepetit und Pascal Fua: *Retexturing in the Presence of Complex Illumination and Occlusions*. In: *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR '07, Seiten 1–8, Washington, DC, USA, 2007. IEEE Computer Society.

[PS02] Pagani, Alain und Didier Stricker: *Integration of virtual objects into an image from a single view. Geometrical properties of images and panoramas*. Diplomarbeit, TU Darmstadt. Fraunhofer IGD, 2002.

[Reg99] Regenbrecht, Holger: *Faktoren für Präsenz in virtueller Architektur*. Doktorarbeit, Bauhaus-Universität Weimar, Juni 1999.

[RGL04] Roman, Augusto, Gaurav Garg und Marc Levoy: *Interactive Design of Multi-Perspective Images for Visualizing Urban Landscapes*. In: *Proceedings of the conference on Visualization '04*, VIS '04, Seiten 537–544, Washington, DC, USA, 2004. IEEE Computer Society.

[Sch06] Scherfgen, David: *3D-Spiele-Programmierung*. Hanser, 2006.

[Sch12a] Scheidle, Wolfgang: *CIELab*. http://www.cielab.de/, 2012. Letzter Zugriff 15. März 2012.

[Sch12b] Scheidle, Wolfgang: *CIELuv*. http://www.cielab.de/cieluv.shtml, 2012. Letzter Zugriff 15. März 2012.

[Scz11] Sczepek, Jörg: *Visuelle Wahrnehmung: Eine Einführung in die Konzepte Bildentstehung, Helligkeit und Farbe, Raumtiefe, Größe, Kontrast und Schärfe*. Books on Demand, 2011. Seiten 60–90.

[SD10] Schindler, G. und F. Dellaert: *Probabilistic Temporal Inference on Reconstructed 3D Scenes*. In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[SDK07] Schindler, G., F. Dellaert und S.B. Kang: *Inferring temporal order of images from 3D structure*. In: *Proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[Sen12] Sensopia: *Magic Plan*. http://www.sensopia.com, 2012. Letzter Zugriff 21. Juli 2012.

[Sna10a] Snavely, Noah: *Bundler: Structure from Motion (SfM) for Unordered Image Collections*. urlhttp://phototour.cs.washington.edu/bundler/, 2010. Letzter Zugriff 28. Februar 2012.

[Sna10b] Snavely, Noah: *Bundler v0.4 User's Manual*. http://phototour.cs.washington.edu/bundler/bundler-v0.4-manual.html, 2010. Letzter Zugriff 28. Februar 2012.

[SNN+11] Shimizu, Akinobu, Keita Nakagomi, Takuya Narihira, Hidefumi Kobatake, Shigeru Nawano, Kenji Shinozaki, Koich Ishizu und Kaori Togashi: *Automated segmentation of 3D CT images based on statistical atlas and graph cuts*. In: *Proceedings of the 2010 international MICCAI conference on Medical*

computer vision: recognition techniques and applications in medical imaging, MCV'10, Seiten 214–223, Berlin, Heidelberg, 2011. Springer-Verlag.

[SSS06a] Snavely, Noah, Steven M. Seitz und Richard Szeliski: *Photo tourism: exploring photo collections in 3D.* In: *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, Seiten 835–846, New York, NY, USA, 2006. ACM.

[SSS06b] Snavely, Noah, Steven M. Seitz und Richard Szeliski: *Photo tourism: exploring photo collections in 3D.* In: *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, Seiten 835–846, New York, NY, USA, 2006. ACM.

[SSS06c] Snavely, Noah, Steven M. Seitz und Richard Szeliski: *Photo tourism: exploring photo collections in 3D.* ACM Trans. Graph., 25(3):835–846, Juli 2006.

[SSS06d] Snavely, Noah, Steven M. Seitz und Richard Szeliski: *Photo tourism: Exploring photo collections in 3D. Presentation.* `http://phototour.cs.washington.edu/Photo_Tourism.ppt`, 2006. Letzter Zugriff 09. April 2012.

[SSS08] Snavely, Noah, Steven M. Seitz und Richard Szeliski: *Modeling the World from Internet Photo Collections.* Int. J. Comput. Vision, 80(2):189–210, November 2008.

[TAB+03] Teller, Seth, Matthew Antone, Zachary Bodnar, Michael Bosse, Satyan Coorg, Manish Jethwa und Neel Master: *Calibrated, Registered Images of an Extended Urban Area.* Int. J. Comput. Vision, 53(1):93–107, Juni 2003.

[Tho01a] Thomae, Reiner: *Perspektive und Axonometrie. Der allgemeine Fluchtpunktsatz.* Kohlhammer, 2001. Seiten 20–21.

[Tho01b] Thomae, Reiner: *Perspektive und Axonometrie. Geometrische Projektionsverfahren.* Kohlhammer, 2001. Seite 9.

[Tön05] Tönnies, Klaus D.: *Grundlagen der Bildverarbeitung.* Addison-Wesley Verlag, 2005. Seiten 166ff.

[TO09] Tawara, Takehiro und Kenji Ono: *Direct 3D manipulation for volume segmentation using mixed reality.* In: *ACM SIGGRAPH ASIA 2009 Posters*, SIGGRAPH ASIA '09, Seiten 33:1–33:1, New York, NY, USA, 2009. ACM.

[Tön10] Tönnis, Marcus: *Augmented Reality: Einblicke in die Erweiterte Realität.* Informatik Im Fokus. Springer, 2010. Seiten 1–5.

[Tre99] Tremblay, Thierry: *3D Basics. The Z buffer.* `http://www.gamedev.net/reference/articles/article673.asp`, 1999. Letzter Zugriff 02. März 2012.

[TSH+11] Tuite, Kathleen, Noah Snavely, Dun-yu Hsiao, Nadine Tabing und Zoran Popovic: *PhotoCity: training experts at large-scale image acquisition through a competitive game.* In: *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, Seiten 1383–1392, New York, NY, USA, 2011. ACM.

[Wag06] Wagner, Patrick: *Farbtiefe bei Film-Scannern und Bilddateien.* `http://www.filmscanner.info/Farbtiefe.html`, 2006. Letzter Zugriff 02. März 2012.

[War12] Ward, Greg: *High Dynamic Range Image Encodings.* `http://www.anyhere.com/gward/hdrenc/Encodings.pdf`, 2012. Letzter Zugriff 02. Februar 2012.

[Wat02a] Watt, Alan: *3D-Computergrafik. Arbeiten mit Maps*, Band 3 der Reihe *i - Informatik.* Pearson Studium, 2002.

[Wat02b] Watt, Alan: *3D-Computergrafik. Der Z-Puffer-Algorithmus*, Band 3 der Reihe *i - Informatik.* Pearson Studium, 2002.

[Wik12a] Wikipedia: *Depth map.* `http://en.wikipedia.org/wiki/Depth_map`, 2012. Letzter Zugriff 02. Februar 2012.

[Wik12b] Wikipedia: *HSV-Farbraum*. `http://de.wikipedia.org/wiki/HSV-Farbraum`, 2012. Letzter Zugriff 12. Oktober 2012.

[Wik12c] Wikipedia: *RGB-Farbraum*. `http://de.wikipedia.org/wiki/RGB-Farbraum`, 2012. Letzter Zugriff 02. März 2012.

[Wik12d] Wikipedia: *Scale-invariant feature transform*. `http://de.wikipedia.org/wiki/Scale-invariant_feature_transform`, 2012. Letzter Zugriff 13. Oktober 2012.

[WWT+03] Wang, Lifeng, Xi Wang, Xin Tong, Stephen Lin, Shimin Hu, Baining Guo und Heung-Yeung Shum: *View-dependent displacement mapping*. In: *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, Seiten 334–339, New York, NY, USA, 2003. ACM.

[Yah04] Yahoo: *Flickr*. `http://www.flickr.com/`, 2004. Letzter Zugriff 02. Juli 2012.

[ZC03] Zhang, Cha und Tsuhan Chen: *A Survey on Image-Based Rendering*. Technical Report AMP 03-03, 2003.

[ZKU+04a] Zitnick, C. Lawrence, Sing Bing Kang, Matthew Uyttendaele, Simon Winder und Richard Szeliski: *High-quality video view interpolation using a layered representation*. In: *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, Seiten 600–608, New York, NY, USA, 2004. ACM.

[ZKU+04b] Zitnick, C. Lawrence, Sing Bing Kang, Matthew Uyttendaele, Simon Winder und Richard Szeliski: *High-quality video view interpolation using a layered representation*. ACM Trans. Graph., 23(3):600–608, August 2004.

[ZP08] Zhu, Jiejie und Zhigeng Pan: *Occlusion registration in video-based augmented reality*. In: *Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI '08, Seiten 10:1–10:6, New York, NY, USA, 2008. ACM.

Eigene Publikationen

[NKG11] Nagl, Frank, Konrad Kölzer und Paul Grimm: *Occlusion Handling and Image-based Lighting using Sliced Images in 3D Photo Collections.* In: *Proceedings of 21st International Conference on Artificial Reality and Telexistence (ICAT2011). ISSN: 1345-1278*, Seiten 118–125, 2011.

[NKG+11a] Nagl, Frank, Konrad Kölzer, Paul Grimm, Tobias Bindel und Stephan Rothe: *ConGrap - Contour Detection based on Gradient Map of Images.* In: Arabnia, Hamid R., Leonidas Deligiannidis und Gerald Schaefer (Herausgeber): *Proceedings of the 2011 International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV 2011). ISBN 1-60132-189-9, 1-60132-190-2 (1-60132-191-0)*, Band 2, Seiten 870–875. CSREA Press, 2011.

[NKG+11b] Nagl, Frank, Konrad Kölzer, Paul Grimm, Tobias Bindel und Stephan Rothe: *ConGrap - Contour Detection based on Gradient Map of Images.* Computer Technology and Application (CTA) Journal. ISSN 1934-7332 (Print). ISSN 1934-7340 (Online), 2(8):628–637, 2011.

[NKG11c] Nagl, Frank, Konrad Kölzer und Paul Grimm: *SBIP - A Framework for Real-time Capable Image Processing in Videos.* In: *MIET - Modern Informations and Electronic Technologies 2011. 12th International scientific-functional conference*, Proceedings, Section 2: Information Technologies and Artificial Intelligence, Seite 90, Odessa, 2011. Politechperiodika.

[NGBA10] Nagl, Frank, Paul Grimm, Bastian Birnbach und Daniel Abawi: *PoP-EYE environment: Mixed Reality using 3D Photo Collections.* In: *ISMAR 2010 Poster Proceedings. ISBN 978-1-4244-9344-9*, Seiten 255–256, 2010.

[NG10] Nagl, Frank und Paul Grimm: *3D-Segmentierung in Bilderwelten.* In: *Proceedings of 16th Workshop Color Image Processing. ISBN 978-3-00-032504-5*, Seiten 181–191, 2010.

[NS10] Nagl, Frank und Sebastian Schäfer: *3D-SurReAL - 3D Surface Reconstruction of Arbitrary (Image) Layers.* In: *2nd Eurographics Workshop on Visual Computing for Biology and Medicine (VCBM) - Poster*, 2010.

[NFST10] Nagl, Frank, Martin Friedl, Andreas Schäfer und Andre Tschentscher: *Shader-Based-Image-Processor - Ein Framework für echtzeitfähige, grafikkartenbasierte Bilderverarbeitung.* In: *GI Seminars 9 Informatiktage. ISBN 978-3-88579-443-1*, Seiten 223–226, 2010.

[NKG10] Nagl, Frank, Konrad Kölzer und Paul Grimm: *MOPS - A lightweight software layer for the definition and rendering of materials.* In: *MIET - Modern Informations and Electronic Technologies 2010, 11th International scientific-functional conference*, Proceedings No.1, Section 1: Information Technologies and Artificial Intelligence, Seite 125. Politechperiodika, 2010.

[NKGB09] Nagl, Frank, Konrad Kölzer, Paul Grimm und Bastian Birnbach: *Verdeckung virtueller Objekte durch reale Bildanteile in Fotos.* In: *6. GI-Workshop "Virtuelle und Erweiterte Realität"der GI-Fachgruppe VR/AR. ISBN 978-3-8322-8662-0*, Seiten 37–48, 2009.

[NKGB09a] Nagl, Frank, Konrad Kölzer, Paul Grimm und Bastian Birnbach: *Generating Depth Maps From Photo Collections.* In: *Proceedings No.1, Section 1: Information Technologies and Artificial Intelligence*, Seite 30. MIET - Modern Informations and

Electronic Technologies, 10th International scientific-functional conference., Politechperiodika, 2009.

[NBKG09] Nagl, Frank, Bastian Birnbach, Konrad Kölzer und Paul Grimm: *Perspektivisch korrektes Einblenden virtueller 3D-Objekte in Einzelfotos.* In: *6. GI-Workshop "Virtuelle und Erweiterte Realität"der GI-Fachgruppe VR/AR. ISBN 978-3-8322-8662-0*, Seiten 151–164, 2009.

[KNGM12] Kölzer, Konrad, Frank Nagl, Paul Grimm und Stefan Müller: *Rendering Image-Based Scenes in Real-Time Utilizing Sparse Scene Geometry.* In: *Proceedings of Computer Graphics International (CGI'2012)*, 2012.

[KNG11] Kölzer, Konrad, Frank Nagl und Paul Grimm: *Synthesizing Relative Radiance for Realistic Rendering of Virtual Objects in 3D Photo Collections.* In: *Eurographics 2011 - Areas Papers. ISSN 1017-4656*, Seiten 71–72. Eurographics Association, 2011.

[BNRG11] Bindel, Tobias, Frank Nagl, Stephan Rothe und Paul Grimm: *A Novel Approach for Straight-Line Segment Detection.* In: *MIET - Modern Informations and Electronic Technologies 2011. 12th International scientific-functional conference*, Proceedings, Section 2: Information Technologies and Artificial Intelligence, Seite 91, Odessa, 2011. Politechperiodika.

[GNA10] Grimm, Paul, Frank Nagl und Daniel Abawi: *IP3D - A Component-based Architecture for Image-based 3D Applications.* In: *SEARIS@IEEEVR2010 Proceedings, IEEE VR 2010 Workshop. ISBN 978-3-8322-8989.8*, Seiten 47–52, 2010.

[RN10] Röse, Jonas und Frank Nagl: *Konzeption und Umsetzung eines 3D-Bildbetrachters.* In: *GI Seminars 9 Informatiktage. ISBN 978-3-88579-443-1*, Seiten 255–258, 2010.

[KNBG09] Kölzer, Konrad, Frank Nagl, Bastian Birnbach und Paul Grimm: *Rendering Virtual Objects with High Dynamic*

Range Lighting Extracted Automatically from Unordered Photo Collections. In: *Lecture Notes in Computer Science - Advances in Visual Computing (ISVC2009). ISBN 978-3-642-10519-7*, Band 5876/2009, Seiten 992–1001. Springer Berlin / Heidelberg, 2009.

[KNBG09a] Kölzer, Konrad, Frank Nagl, Bastian Birnbach und Paul Grimm: *Automatische Extraktion von HDR-Beleuchtungsdaten aus Fotos.* In: *6. GI-Workshop "Virtuelle und Erweiterte Realität"der GI-Fachgruppe VR/AR. ISBN 978-3-8322-8662-0*, Seiten 191–200, 2009.

[BGKN09] Birnbach, Bastian, Paul Grimm, Konrad Kölzer und Frank Nagl: *Automatische Bild- und Darstellungsoptimierung für 3D-Photo-Collections.* In: *6. GI-Workshop "Virtuelle und Erweiterte Realität"der GI-Fachgruppe VR/AR. ISBN 978-3-8322-8662-0*, Seiten 1–11, 2009.

[KJN+08] Kölzer, Konrad, Yvonne Jung, Frank Nagl, Bastian Birnbach und Paul Grimm: *Grafikkartenbasierte Simulation von tröpfchenförmigen Flüssigkeiten auf Oberflächen.* In: Schumann, M. (Herausgeber): *Virtuelle und Erweiterte Realität : 5. Workshop der GI-Fachgruppe VR/AR. ISBN 978-3-8322-7572-3*, Aachen, 2008. Shaker.

[BGKN08] Birnbach, Bastian, Paul Grimm, Konrad Kölzer und Frank Nagl: *Prototyping von AR - Präsentationen mit ARBlender.* In: *5. GI-Workshop "Virtuelle und Erweiterte Realität"der GI-Fachgruppe VR/AR. ISBN 978-3-8322-7572-3*, 2008.

Betreute studentische Arbeiten

[Fle10] Fleckner, Martin: *Erweiterte Methoden der Erstellung von Depth-Maps auf der Grundlage von 3D-Bilderwelten.* Bachelorarbeit, Fachhochschule Erfurt, 2010.

[Hes12] Hesselink, Jan: *Konzeption und prototypische Implementation eines Bilderwelten-Viewers für das Web.* Bachelorarbeit, Hochschule Fulda, 2012.

[Hof10] Hoffmann, David: *Konzeption und Implementation einer Software zur Simulation eines digitalen Turntables.* Bachelorarbeit, Fachhochschule Erfurt, 2010.

[Kam09] Kampa, Benjamin: *Darstellung von Preview-Thumbnails in 3D-Bilderwelten.* Bachelorarbeit, Fachhochschule Erfurt, 2009.

[Sch11] Schäfer, Andreas: *Objekterkennung und Modellgenerierung aus Bilderwelten.* Masterarbeit, Fachhochschule Erfurt, 2011.

Sachregister

W

X

Z

Glossar

3D-Bilddarstellung

Als 3D-Bilddarstellung wird die Projektion der rekonstruierten Szene aus einer beliebigen Position und Ausrichtung innerhalb der 3D-Welt auf eine Bildebene bezeichnet. Seite 11

Abgebildete Objekte

Siehe *Bilderweltenszene*. Seite 5

Authentische Augmentierung

Eine authentische Augmentierung bezeichnet die *Visuelle Verschmelzung* ohne *Visuelle Störungen* von verschiedenen realen Fotos und synthetischen Bildern (bzw. 3D-Objekten). Seite 3

Authentische Verdeckung

Siehe *Verdeckung*. Seite 7

Bildbasierte Beleuchtung

Die reale Umgebung der Bildinhalte beinhaltet eigene Lichtquellen. Für eine *Authentische Augmentierung* müssen daher die virtuellen Objekte eine zu den Bildinhalten passende Beleuchtung aufweisen [KNG11]. Hierzu gehört auch ein zu der Szene passender Schattenwurf. Dies wird als bildbasierte Beleuchtung bezeichnet. Seite 9

Bilderwelt

Eine Bilderwelt stellt eine bildbasierte 3D-Welt, bestehend aus gewöhnlichen Fotos, dar. Hierbei sind für jedes Foto die dazugehörige Kameraposition und -orientierung (sozusagen Aufnahmestandort und -richtung) im

3D-Raum rekonstruiert. Zu jeder rekonstruierten Kamera wird eine *Image-Plane* erzeugt, die mit dem *Kamerabild* texturiert ist. Ein Kamerabild bezeichnet also das eigentliche Foto und die Image-Plane repräsentiert das Foto als 3D-Objekt innerhalb der Bilderwelt. Zusätzlich enthält eine Bilderwelt neben den Image-Planes eine 3D-Punktwolke. Diese repräsentiert eine ansatzweise 3D-Rekonstruktion der abgebildeten Szene bestehend aus *Key-Points*. Seite 50

Bilderweltenszene

Die abgebildete reale Umgebung der Bildinhalte einer *Bilderwelt* wird Bilderweltenszene (oder als Kurzform nur Szene) genannt. In den Fotos dargestellte einzelne reale Objekte werden als abgebildete Objekte bezeichnet. Seite 5

Bildstruktur

Die Bildstruktur eines Fotos, genauer ausgedrückt die Struktur des Bildinhaltes, ist definiert durch die Anzahl der Objekte in der abgebildeten Szene, deren geometrischen Formen und Größen sowie deren Struktur und Oberfläche. Einfarbige, glatte und spiegelnde Flächen werden auch als homogene Bildstruktur und mehrfarbige und rauhe Flächen als heterogene Bildstruktur bezeichnet. Seite 52

Default-Rendering

Als Default-Rendering wird die grundlegende bildbasierte Darstellung einer *Bilderwelt* bezeichnet. Hierbei werden die *Image-Planes* so gerendert, dass der Aufbau einer Bilderwelt mit annähernd nahtlosen Übergangen zwischen den Fotos (bzw. Image-Planes) im 3D-Raum erscheint. Eine weiterführende Adaption des Bilderwelten-Renderings für die Erreichung des Hauptziels dieser Arbeit erfolgt beim Default-Rendering nicht. Seite 95

Depth-Map

Siehe *Tiefeninformation*. Seite 47

Depth-Patch

Ein Depth-Patch stellt ein Bildsegment dar, dem eine Tiefe zugeordnet wurde und dessen Pixel als Füllwerte die *Tiefeninformation* beinhalten. Die Gesamtheit aller Depth-Patches ergibt die *Depth-Map*. Seite 93

Eckpunkt-Tripel

Ein Eckpunkt-Tripel stellt eine Dreiecksbeziehung von Eckpunkten der *Objektbasierten Segmentierung* dar. Ein Eckpunkt-Tripel wird erzeugt, indem die Strecken, gebildet aus einem Eckpunkt A mit jedem anderen Eckpunkt des gleichen Segmentes, abgetastet werden. Befindet sich eine Strecke zwischen zwei Eckpunkten innerhalb des Segmentes (oder auf der Konturgrenze), wird (ausgehend vom zweiten Eckpunkt B) ebenfalls die Strecken zu allen weiteren Eckpunkten des Segmentes abgetastet. Sobald auf diese Weise ein dritter Eckpunkt C detektiert wurde, wird die Strecke von C zu A abgetastet. Befindet sich diese Strecke wieder innerhalb des Segmentes (oder auf der Konturgrenze) ist ein Tripel, bestehend aus den Eckpunkten $\{A, B, C\}$ gefunden wurden. Seite 134

Farb- und Konturbasierte Segmentierung

Die *Farbbasierte Segmentierung* und die *Konturbasierte Segmentierung* werden kombiniert, um Vorteile beider Verfahren auszunutzen und die Fehleranfälligkeiten der Einzelverfahren zu reduzieren. Seite 118

Farbbasierte Segmentierung

Bei der farbbasierten Segmentierung werden zusammenhängende (benachbarte) ähnlich- oder gleichfarbige Pixel eines Fotos als ein Segment und somit als ein Objekt interpretiert. Seite 111

Feature-Point

Ein Feature-Point ist ein lokales, markantes Bildmerkmal in einem Bild, das mithilfe des SIFT-Algorithmus detektiert wurde. Er besitzt drei Kenngrößen (Position, Orientierung, Skalierung), die als Vektoren gespeichert und zum bildübergreifendem Vergleichen von Bildmerkmalen genutzt werden können. Seite 54

Float-Depth-Map

Da in vielen Problemstellungen eine hohe Präzision der Tiefenwerte gefordert wird, empfiehlt sich das Speichern der *Depth-Map* als einkanälige Gleitkommazahl-Textur. Diese wird Float-Depth-Map genannt. Seite 48

Flächenabdeckung

Die Abdeckung des 3D-Raumes durch Fotos in einer *Bilderwelt* wird Flächenabdeckung bezeichnet. Seite 51

Freie Navigation

Bei der freien Navigation bewegt sich der Betrachter aus der Egoperspektive beliebig durch die *Bilderweltenszene*, vergleichbar mit der Navigation eines Ego-Shooter-Computerspiels. Das bedeutet die Betrachterpositionen entsprechen, im Gegensatz zur *Kameranavigation*, nicht zwangsläufig einer Kameraposition für ein benutztes Bild. Seite 11

Geometry-Depth-Matching

Bei diesem Verfahren wird eine Szenengeometrie des abgebildeten Szenenausschnittes erzeugt und dadurch eine *pixelgenaue Depth-Map* erstellt. Die erzeugte Szenengeometrie stellt keine vollständige 3D-Rekonstruktion der *Bilderweltenszene* dar, da nur ein Szenenausschnitt aus der Sicht einer Kamera rekonstruiert wird. Seite 131

Grayscale-Depth-Map

Als Grayscale-Depth-Map werden *Depth-Maps* mit 8-Bit Kodierungsgenauigkeit pro Tiefenwert bezeichnet. Diese Art von Depth-Maps benötigt weniger Speicher und besitzt eine geringere Genauigkeit der Tiefenwerte als z. B. eine *Float-Depth-Map*. Seite 48

Image-Distance

Als Image-Distance wird die Distanz zwischen einer Kamera und seiner korrespondierenden *Image-Plane* in der Bilderweltenszene bezeichnet. Seite 95

Image-Geometry-Rendering

Das Image-Geometry-Rendering verwendet die im Rahmen des *Geometry-Depth-Matching*-Verfahrens rekonstruierte Szenengeometrie des in einem *Kamerabild* abgebildeten Szenenausschnittes und texturiert diese mittels perspektivischer Projektion des Kamerabildes. Seite 148

Image-Plane

Eine Image-Plane in einer *Bilderwelt* stellt eine planare, rechteckige Fläche im 3D-Raum dar, die vollständig und passgenau mit dem *Kamerabild* texturiert ist und orthogonal in einem berechneten Abstand vor die rekonstruierte Kamera positioniert wird. Ein Kamerabild bezeichnet also das eigentliche Foto und die Image-Plane repräsentiert das Foto als 3D-Objekt innerhalb der Bilderwelt. Seite 50

Kamerabild

Als Kamerabild wird das zu einer Kamera einer *Bilderwelt* zugehörige Foto bezeichnet. Es wird als Textur für die *Image-Plane* der Kamera verwendet. Ein Kamerabild bezeichnet also das eigentliche Foto und die Image-Plane repräsentiert das Foto als 3D-Objekt innerhalb der Bilderwelt. Seite 50

Kameragenauigkeit

Das Generieren einer *Bilderwelt* stellt ein Näherungsverfahren dar, weshalb es zu Ungenauigkeiten in den Ergebnissen kommen kann. Deshalb müssen die rekonstruierten Kameraparameter auf ihre Genauigkeit geprüft und bewertet werden. Das prozentuale Ergebnis der Bewertung wird im weiteren Text (relative) Kameragenauigkeit genannt. Seite 101

Kameranavigation

Die Kameranavigation ermöglicht bei der Darstellung einer *Bilderwelt* das Navigieren von einer Kamera zu einer anderen bzw. von einem Foto zu einem anderen Foto. Eine freie, kameraunabhängige Navigation durch die *Bilderweltenszene* (siehe *Freie Navigation*) ist bei dieser Navigation nicht möglich. Seite 10

Kameratiefe

Als Kameratiefe werden *Tiefeninformationen* des Kamerakoordinatensystems einer Kamera der *Bilderwelt* (also Tiefeninformationen aus der Sicht der Kamera im eigenen Koordinatensystem) bezeichnet. Seite 14

Key-Point

Key-Points sind Bestandteil der 3D-Punktwolke einer *Bilderwelt*. Sie repräsentieren eine ansatzweise 3D-Rekonstruktion der abgebildeten Szene. Neben einer 3D-Position besitzen Key-Points einen Farbwert, der sich aus dem Durchschnitt aller Bildpunkte, die den Key-Point in den Fotos repräsentieren, bildet. Key-Points, die zu einer bestimmten Kamera zugehörig sind, werden als *Observed-Points* dieser Kamera bezeichnet. Kameras, die einen bestimmten Key-Point als Bildpunkt in ihrem Foto enthalten, werden als *Observer* des Key-Points bezeichnet. Der korrespondierende *Feature-Point* als Bildpunkt in Pixeleinheiten wird *Observed-Pixel* genannt. Seite 56

Key-Point-Depth-Matching

Bei dem Key-Point-Depth-Matching-Verfahren werden die *Key-Points* auf die Bildebene als kreisförmige Sprites zurückprojiziert. Die Distanz des Key-Points zur Kamera wird dabei als Füllwert für die Pixel eines Sprites verwendet. Dies ermöglicht eine Tiefenzuordnung ohne Segmentierung. Seite 129

Konturbasierte Segmentierung

Bei der konturbasierten Segmentierung eines Bildes wird die Farbeigenschaft der abgebildeten Objekte nicht beachtet, sondern mithilfe detektierter Kanten ein Segmentbild erzeugt. Hierbei wird ein Segment (ein Objekt) durch zusammengehörige Kanten repräsentiert. Seite 113

Objektbasierte Segmentierung

Mithilfe eines semiautomatischen Schrittes wird ein Bild in räumlich zusammenhängende Segmente zerlegt. Hierfür werden unterstützend die Eckpunkte der Segmente durch den Nutzer eingezeichnet. Dieser semiautomatische Schritt wird als *Segmentskizzierung* bezeichnet. Der Vorteil

dieses Segmentierungsverfahrens liegt zum einen in der (annähernden) Fehlerfreiheit und zum anderen wird ein Zusammenhang zwischen Segmente und der abgebildeten räumlichen Struktur hergestellt. Seite 121

Objektstruktur

Die Objektstruktur beinhaltet Farbe und Beschaffenheit des abzubildenden Objektes (oder Teilobjektes) sowie dessen Beleuchtungs- und Reflexionssituation innerhalb der realen (abzubildenden) Szene. Seite 88

Observed-Pixel

Siehe *Key-Point*. Seite 56

Observed-Point

Siehe *Key-Point*. Seite 56

Observer

Siehe *Key-Point*. Seite 56

Occluder

Occluder sind reale, abgebildete Objekte (Bildsegmente) der Fotos, die für die visuelle Überlagerung genutzt und so virtuelle Objekte verdecken. Seite 10

pixelgenau

Pixelgenau bedeutet, dass für jedes Pixel individuell ein zu ihm zugehöriger Wert ermittelt wird (z. B. Tiefenwerte bei *Depth-Maps*). Seite 15

Pixelized-Image-Rendering

Das Pixelized-Image-Rendering stellt eine Erweiterung des *Sliced-Image-Rendering* hinsichtlich eines globalen Image-Based-Rendering-Verfahrens dar, das die Bilderweltenszene auch ohne 3D-Geometrie als eine einzige globale 3D-Darstellung aller Fotos rendern kann. Der Ansatz des Sliced-Image-Renderings wird in der Form erweitert, dass das Rendern der Pixel aller Slices in Abhängigkeit von der Ausrichtung zum Betrachter erfolgt. Seite 225

PoP-EYE

PoP-EYE stellt eine Komponenten-basierte Software dar, die vom Autor und weiteren Personen für die Umsetzung des BMBF-Projektes „PoP-EYE“ (Fördernummer 17N0909) entwickelt wurde. Die PoP-EYE-Software ist eine auf *Bilderwelten*-basierende AR-Umgebung, die eine einfache und schnelle Integration virtueller 3D-Objekte erlaubt mit dem Hintergrund 3D-Planungsvisualisierungen und Produktpräsentationen in einer fotobasierten 3D-Welt zu ermöglichen. Hierbei steht eine authentische Darstellung der Planungsvisualisierung im Vordergrund. Alle konzeptionellen Umsetzungen dieser Arbeit flossen als Komponenten in diese Software ein. Seite 160

Punktgenauigkeit

Das Generieren einer *Bilderwelt* stellt ein Näherungsverfahren dar, weshalb es zu Ungenauigkeiten in den Ergebnissen kommen kann. Deshalb müssen die rekonstruierten 3D-*Key-Points* auf ihre Genauigkeit geprüft und bewertet werden. Das prozentuale Ergebnis der Bewertung wird im weiteren Text (relative) Punktgenauigkeit genannt. Seite 104

Render-Adaption

Die unterschiedliche Bildqualität der Fotos einer *Bilderwelt* (hervorgerufen durch z. B. Bildrauschen, unterschiedliche Auflösungen, Über-/Unterbelichtungen, qualitative Unterschiede von CCD-Sensoren verschiedener Aufnahmegeräte, unterschiedliche Beleuchtungssituationen der Szene) können bei der Integration hochaufgelöster virtueller 3D-Objekte zu *Visuellen Störungen* führen. Folglich müssen Anpassungen zwischen der Bildqualität der Fotos und dem Rendering der virtuellen Objekte erfolgen. Diese Darstellungsanpassung zwischen wird als Render-Adaption bezeichnet. Seite 8

Rgb-Divided-Depth-Map

Die Rgb-Divided-Depth-Map arbeitet mit gewöhnlichen RGB-Texturen und 8 Bit pro Farbkanal. Der Tiefenwert wird auf die drei RGB-Farbkanäle (jeweils als Ganzzahl und im Dezimalsystem) zerlegt. Die Vorkommastellen des Tiefenwertes werden im Rot-Kanal abgelegt. Die ersten zwei

Nachkommastellen werden im Grün-Kanal und die dritte und vierte Nachkommastelle im Blau-Kanal abgelegt. Diese Speicherform beschränkt den Tiefenbereich. Die Tiefe darf einen Wert von 255 Einheiten nicht überschreiten, da im Rot-Kanal für die Vorkommastellen nur 8 Bit zur Verfügung stehen. Seite 48

Rohdaten

Rohdaten beinhalten extrinsische und intrinsische Kameraparameter sowie Daten einer 3D-Punktwolke, die durch einen *Structure-from-Motion*-Algorithmus gewonnen werden. Sie sind Voraussetzung für den Aufbau einer *Bilderwelt*. Seite 13

Schwarze Bereiche

Schwarze Bereiche sind Bereiche innerhalb der 3D-Welt, die nicht mit Bildmaterial der *Bilderweltenszene* texturiert sind. Sie entstehen, wenn Fotos bzw. benachbarte *Image-Planes* innerhalb der 3D-Welt keine Überdeckungen haben. Sie wirken störend beim Betrachten der *Bilderwelt*, da sie den Zusammenhang der Szene unterbrechen. Seite 51

Segment-Depth-Matching

Das Segment-Depth-Matching-Verfahren generiert eine segmentgenaue *Depth-Map*, d. h. nicht jedes Pixel, sondern Flächen (Segmente) des Bildes bekommen eine Tiefe zugewiesen. Die Depth-Map kodiert also die Tiefe von senkrecht zur Blickrichtung stehenden, planaren Flächen. Als Eingabedaten werden die *Key-Points* und die *Segmentbilder* vorausgesetzt. Seite 126

Segmentbild

Die Bildsegmentierung unterteilt jedes Foto der *Bilderwelt* in die dort abgebildeten realen (Teil-)Objekte. Als Ergebnis entstehen Segmente, die in einem (neuen) Bild gespeichert und verwaltet werden. Dieses Bild wird Segmentbild genannt. Seite 109

Segmentskizzierung

Die Segmentskizzierung bezeichnet den semiautomatischen Schritt für die *Objektbasierte Segmentierung*. Hierfür werden unterstützend Eckpunkte

der Segmente durch den Nutzer eingezeichnet. Mithilfe der Eckpunkte wird das *Kamerabild* in räumlich zusammenhängende Segmente zerlegt. Seite 121

Sliced-Image-Rendering

Das Sliced-Image-Rendering stellt ein Image-Based-Rendering-Verfahren dar, dass ohne 3D-Geometrie ein Bild anhand gegebener *Tiefeninformationen* als dreidimensionale Darstellung rendern und auf diese Weise eine *Authentische Verdeckung* auch bei *Freier Navigation* ermöglichen kann. Auf Basis gewonnener Tiefeninformationen wird ein Bild innerhalb der *Bilderweltenszene* in Slices zerlegt. Ein Slice repräsentiert eine Bildregion des Originalbildes. Jedes Slice wird an die für die Region entsprechende Tiefenposition, die in der *Depth-Map* gespeichert wurde, verschoben. Die Gesamtheit aller verschobenen, skalierten Regionen wird Sliced-Image genannt. Seite 144

Szene

Siehe *Bilderweltenszene*. Seite 5

Tiefeninformation

Tiefeninformationen sind Informationen über die (orthogonale) Distanz abgebildeter realer Objekte zur Kamera (Bildebene). Somit kann jedem Pixel eines Fotos eine Tiefe zugeordnet werden. Diese Tiefe eines Pixels wird als *Tiefenwert* (in der Regel ein Gleitkommawert) bezeichnet. Werden diese Tiefenwerte als Füllwerte für die Pixel verwendet, entsteht ein Bild bzw. eine Textur, die *Tiefenkarte* oder *Depth-Map* genannt wird. Auf diese Weise ist es möglich, die rekonstruierte Tiefe eines Fotos *pixelgenau* und effizient zu speichern. Seite 47

Tiefenkarte

Siehe *Tiefeninformation*. Seite 47

Tiefenwert

Siehe *Tiefeninformation*. Seite 47

Transformationsanpassung

Ein Problem bei der Integration virtueller Objekte in die *Bilderweltenszene* sind geometrische Gegebenheiten der Objekte. Für einen authentischen Eindruck muss die Größe der virtuellen Objekte an den Maßstab der Bilderweltenszene angepasst sein. Diese Anpassung wird als Transformationsanpassung bezeichnet. Seite 7

Verdeckung

Die Überlagerung zwischen virtuellen Objekten und einzelnen Bildanteilen eines oder mehrerer Fotos wird Verdeckung genannt. Eine der menschlichen Wahrnehmung entsprechenden, korrekten Überlagerung wird als *Authentische Verdeckung* bezeichnet. Seite 7

Visuelle Störungen

Unter visuellen Störungen in einer *Bilderwelt* werden visuelle Auffälligkeiten bezeichnet, die einen homogenen Gesamteindruck der Bilderwelt verhindern. Hierzu zählen z. B. unterschiedliche Aufnahme- und Bildqualität der Fotos, fehlerhafte Verdeckung der virtuellen Objekte und physikalische Unstimmigkeiten (z. B. übergroße virtuelle Objekte innerhalb der Bilderwelt). Seite 3

Visuelle Verschmelzung

Der visuelle Eindruck spielt die größte Rolle für die Glaubwürdigkeit und Akzeptanz des Nutzers. Je schwieriger die visuelle Trennbarkeit von virtuellen und realen Anteilen für den Nutzer erscheint, desto besser der authentische Eindruck bei ihm. Dementsprechend ist das Hauptziel der Darstellung einer AR-Szene, dass der Nutzer die virtuellen und realen Anteile (fast) nicht unterscheiden kann. Dies wird als visuelles Verschmelzen der virtuellen und realen Anteile bezeichnet. Seite 3

Z-Distanz

Als Z-Distanz eines *Observed-Points* wird die Entfernung des Observed-Points zu seiner Kamera auf der bzw. parallel zur (negativen) Z-Achse des Kamerakoordinatensystems bezeichnet. Seite 127

Anhang A

Lebenslauf

Name:	Frank Nagl
Geboren:	29. Oktober 1982 in Heilbad Heiligenstadt, Deutschland
Schul- und Ausbildung	
07/2000	Realschulabschluss an der Regelschule „Theodor Storm“ Heiligenstadt
08/2000 – 07/2003	Ausbildung Biologielaborant an der Georg-August-Universität Göttingen, Bereich Humanmedizin
02/2001 – 07/2003	Teilnahme am Zusatzunterricht (Abendschule) zum Erlangen der Fachhochschulreife im Bereich Technik an den Berufsbildenden Schulen II Göttingen
Hochschulausbildung	
09/2003 – 08/2006	Bachelor-Studium Angewandte Informatik an der Fachhochschule Erfurt
09/2005 – 02/2006	Auslandsaufenthalt in Bandung, Indonesien. Fachpraktikum am Institut Teknologi Bandung (ITB)
08/2006	Erreichen des akademischen Grads „Bachelor of Science“ (B.Sc.)

09/2006 – 07/2008	Master-Studium Angewandte Informatik an der Fachhochschule Erfurt
03/2007	Studienexkursion nach Bandung, Indonesien. Workshop am Institut Teknologi Bandung (ITB)
06/2008	Erreichen des akademischen Grads „Master of Science“ (M.Sc.)
Berufliche Tätigkeit	
03/2006 – 07/2007	Studentische Hilfskraft. Administration und Betreuung von Computerpools
08/2008 – 10/2008	Software-Entwickler bei EasternGraphics GmbH Ilmenau
10/2008 – 05/2009	Wissenschaftlicher Mitarbeiter an der Fachhochschule Erfurt. Forschungsprojekt „Olivia“, Leitung: Prof. Dr. Paul Grimm
06/2009 – 06/2011	Wissenschaftlicher Mitarbeiter an der Fachhochschule Erfurt. Forschungsprojekt „PoP-EYE“, Leitung: Prof. Dr. Paul Grimm
09/2009 – 02/2010	Lehrauftrag an der Fachhochschule Erfurt im Fach „Spezielle Kapitel Grafischer Datenverarbeitung“ (Master-Studiengang der Angewandten Informatik)
07/2011 – 06/2012	Wissenschaftlicher Mitarbeiter an der Hochschule Fulda. Forschungsprojekt „PoP-EYE“, Leitung: Prof. Dr. Paul Grimm
07/2012 – 09/2012	Wissenschaftlicher Mitarbeiter an der Hochschule Fulda. Forschungsprojekt „kARbon“, Leitung: Prof. Dr. Paul Grimm

Anhang B

Eidesstattliche Versicherung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Dissertation selbständig angefertigt und mich anderer Hilfsmittel als der in ihr angegebenen nicht bedient habe, insbesondere, dass alle Entlehnungen aus anderen Schriften mit Angabe der betreffenden Schrift gekennzeichnet sind.

Ich versichere, nicht die Hilfe einer kommerziellen Promotionsvermittlung in Anspruch genommen zu haben.

Des Weiteren erkläre ich hiermit, dass ich mich bisher keiner Doktorprüfung unterzogen habe.

..... Frankfurt am Main, den .....

Frank Nagl