HAW HAMBURG

BACHELOR THESIS
Björn Dittmann

Realisierung eines Interactive Tabletop als Plattform in einer Smart Home Umgebung

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Björn Dittmann

Realisierung eines Interactive Tabletop als Plattform in einer Smart Home Umgebung

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Informatik Technischer Systeme*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Thomas Lehmann
Zweitgutachter: Prof. Dr. Kai von Luck

Eingereicht am: 16. Juni 2023

Björn Dittmann

Thema der Arbeit

Realisierung eines Interactive Tabletop als Plattform in einer Smart Home Umgebung

Stichworte

Interactive Tabletop, HCI, Computer Vision

Kurzzusammenfassung

Diese Bachelorarbeit beschäftigt sich mit der Konzeption und Realisierung eines Interactive Tabletop, welcher auf der Basis bildbasierter Methoden Interaktionen erkennen kann. Das realisierte System stellt eine Plattform für die Erforschung zur Identifikation von Interaktionen im Bereich der Computer Vision sowie zur Analyse von Gestaltung und Verwendung interaktiver Anwendungen im Bereich der Interactive Tabletop dar.

Björn Dittmann

Title of Thesis

Realization of an interactive tabletop as a platform in a smart home environment

Keywords

interactive tabletop, HCI, computer vision

Abstract

This bachelor thesis deals with the conception and realization of an interactive tabletop that can identify interactions on the basis of image-based methods. The realized system represents a platform for research on the identification of interactions in the field of computer vision as well as for the analysis of design and use of interactive applications in the field of interactive tabletop.

Inhaltsverzeichnis

1 Einleitung

Im Einleitungskapitel wird zunächst die Motivation für die vorliegende Arbeit erläutert und die damit verbundenen Ziele definiert. Anschließend erfolgt ein Überblick über den Aufbau und Inhalt der restlichen Ausarbeitung.

1.1 Motivation

Die Verwendung der von Interactive Tabletop gebotenen Benutzerschnittstelle findet in verschiedensten Bereichen der heutigen Zeit bereits ihre Anwendung. So werden sie beispielsweise in der Lehre zur Gestaltung interaktiver Lerninhalte [12], in Museen zur interaktiven Präsentation von Ausstellungsinhalten [15] und in vielen weiteren professionellen Bereichen bereits genutzt [27, vgl. Tabelle 1.1]. Doch wie würde diese spezielle Art einer Benutzerschnittstelle wohl im vertrauten Umfeld des eigenen Zuhauses verwendet werden? Was für Anwendungen könnten mit ihr für die intelligente Wohnung der Zukunft, dem sogenannten Smart Home, realisiert werden?
Um genau solche Fragen zu erforschen, widmet sich das Living Place Labor der Hochschule für Angewandte Wissenschaften (HAW) in Hamburg unter anderem der Untersuchung der Interaktion zwischen Mensch und Maschine in einem natürlichen Wohnumfeld [4]. In diesem befindet sich mitunter auch ein feststehender Tresen, welcher ehemals als ein eben solcher Interactive Tabletop fungierte. Da die darin vor Jahren verbaute Technik mittlerweile veraltet ist und die zugehörige Software nicht für moderne Betriebssysteme aktualisiert wurde, ist dieser jedoch nicht mehr funktionsfähig. Um die Forschung zu diesem Thema im Labor wieder aufnehmen zu können, soll das bestehende Mobiliar des Tresens von Grund auf neu als Interactive Tabletop realisiert werden. Dadurch können neue Erkenntnisse, sowohl zur allgemeinen Handhabung der von Interactive Tabletop angebotenen Benutzerschnittstelle als auch speziell Einblicke über die Nutzung von interaktiven Anwendungen im modernen häuslichen Umfeld, gewonnen werden.

1.2 Ziele

Um die Forschung zur Verwendung von Interactive Tabletop als Benutzerschnittstelle in einem häuslichen Umfeld, sowie zur Nutzung und Entwicklung interaktiver Anwendungen darin fortzusetzen, ist es erforderlich, den Tresen als Interactive Tabletop neu zu konstruieren.
Ziel dieser Arbeit ist daher die vollständige Realisierung eines Interactive Tabletop auf Basis des bestehenden Mobiliars im Living Place Labor der HAW Hamburg. Das dabei zu entwickelnde System soll ermöglichen Interaktionen mit der Projektionsfläche des Tresens, unter der Verwendung bildgestützter Verfahren, identifizieren zu können und visuelle Inhalte passend auf dieser zu projizieren. Die Erkennung von Interaktionen soll dabei zuverlässig gegenüber möglicher Störeinflüsse aus der Umgebung erfolgen. Zudem sollte sie ohne auffällige Verzögerungen ablaufen, um dem Nutzer ein angenehmes Nutzungserlebnis zu ermöglichen.
Das realisierte System soll letztendlich eine Plattform bieten, auf welcher Fragestellungen zur Nutzung der von Interactive Tabletop dargebotenen Benutzerschnittstelle im privaten Umfeld erforscht werden können.

1.3 Aufbau

Zu Beginn erfolgt im Kapitel 2 eine Einordnung der Arbeit in den fachlichen Kontext der Interactive Tabletop. Anschließend wird die Umgebung genauer betrachtet, in der die Arbeit durchgeführt wird. Dabei werden Informationen zum Aufbau, zur technischen Infrastruktur und bestehenden Rahmenbedingungen des entsprechenden Labors angegeben. Dies ermöglicht einen umfassenden Überblick über die Gegebenheiten, in denen das System entwickelt und eingesetzt wird. Zuletzt werden konkrete Anforderungen an das zu entwickelnde System definiert. Dabei werden sowohl funktionale als auch nichtfunktionale Anforderungen, entsprechend der Zielsetzung, berücksichtigt.

Im darauffolgenden Hauptteil, dem Kapitel 3, wird anschließend die Konstruktion des Systems gemäß den gestellten Anforderungen behandelt. Zu Beginn wird ein Konzept aufgestellt, das grundlegende Hardwarekomponenten und die Elemente der Softwarearchitektur definiert. Um potenzielle Herausforderungen im Zusammenhang mit den bestehenden Rahmenbedingungen zu identifizieren, werden Experimente durchgeführt. Die Ergebnisse dieser Voruntersuchungen liefern wichtige Erkenntnisse, die zur Definition

zusätzlicher Anforderungen an das System führen. Anschließend wird ausführlich die konkrete Hardware- und Software- Implementation des Systems beschrieben.

Nach Abschluss der Konstruktion wird in Kapitel 4 eine Evaluation des realisierten Systems vorgenommen. Hierfür werden zunächst Untersuchungen durchgeführt, um eine Grundlage für die Bewertung zu schaffen. Die Ergebnisse dieser Nachuntersuchungen werden zusammen mit den Aspekten der Konstruktion diskutiert, um festzustellen, ob das System den gestellten Anforderungen entspricht.

In Kapitel 5 wird inhaltlich abschließend eine demonstrative Implementation eines Anwendungsbeispiels präsentiert. Dabei wird exemplarisch gezeigt, wie das entwickelte System in der Praxis eingesetzt werden kann und wie konkrete Anwendungsfälle damit realisiert werden können.

Zum Schluss wird im gleichnamigen Kapitel 6 eine Zusammenfassung der wichtigsten Ergebnisse und Erkenntnisse der Arbeit aufgeführt. Zudem wird ein Ausblick gegeben, welche Aspekte in Zukunft aufbauend auf dieser Arbeit weiter betrachtet werden können.

2 Analyse

Im Analysekapitel wird die Arbeit zunächst in den fachlichen Kontext der Interactive Tabletop eingeordnet. Anschließend wird die Umgebung, in der die Arbeit durchgeführt wird, vorgestellt und zuletzt konkrete Anforderungen an das zu entwickelnde System definiert.

2.1 Einordnung Interactive Tabletop

Das Gebiet der Human Computer Interaction (HCI) befasst sich mit der Untersuchung, Gestaltung und Entwicklung von Schnittstellen zwischen Menschen und Computern. Ein Teil dieses Gebiets beschäftigt sich mit dem Thema der Interactive Surfaces. Dieses behandelt interaktive Oberflächen, auf denen Benutzer digitale Inhalte und Anwendungen durch Berührung, Gesten oder andere Eingabemethoden manipulieren und mit ihnen interagieren können. Diese Oberflächen stellen eine Erweiterung gegenüber der traditionellen Benutzererfahrung herkömmlicher Bildschirme und Eingabegeräte dar. Das Themengebiet der Interactive Tabletop ist wiederum ein Teilbereich der Interactive Surfaces. Interactive Tabletop im Speziellen sind meist horizontal ausgerichtete, tischähnliche Oberflächen, auf denen Benutzer digitale Inhalte interaktiv manipulieren können. Entsprechend ihres analogen Vorbilds bieten sie durch die horizontale Ausrichtung eine natürliche und ergonomische Arbeitsfläche für digitale wie zwischenmenschliche Interaktionen.
Seit 2006 werden Workshops zum Thema der Interactive Tabletop veranstaltet. Zwischen 2009 und 2015 wurde dann die Konferenz ITS (Conference on Interactive Tabletop and Surfaces, danach ISS: Conference on Interactive Surfaces and Spaces) von der Association for Computing Machinery (ACM) jährlich abgehalten [6][7]. Die auf den Konferenzen präsentierten Beiträge behandeln in der Regel Themen bezüglich der Verwendung von Interactive Tabletop in spezifischen Bereichen, verschiedene Interaktionsarten oder die Gestaltung und Entwicklung von Anwendungen konkreter Anwendungsfälle. Im Fokus

dieser Arbeit steht jedoch hauptsächlich die praktische Realisierung eines Interactive Tabletop auf Hardware- und Softwareebene.

Das primäre Unterscheidungsmerkmal verschiedener Interactive Tabletop Implementationen stellt die jeweils verwendete Technologie zur Identifikation von Interaktionen dar. Dabei gibt es verschiedene Ansätze [8, vgl. S. 9 ff.][9]. Einige Methoden basieren auf der Erkennung von Veränderungen einer physikalischen Größe, wie es bei kapazitiven [11], resistiven oder auf Surface Acoustic Wave (SAW) [18] basierenden Verfahren der Fall ist. Andere arbeiten auf optischer Basis unter Verwendung von Kamerasystemen und passender Beleuchtung, wie bei der Rear- / Front- Diffuse Illumination (R-/F-DI) [20], Diffused Surface Illumination (DSI), Frustrated Total Internal Reflection (FTIR) [17] und Laser Light Planes (LLP). Auch möglich ist die Verwendung von Radio Frequency Identification (RFID) [28].

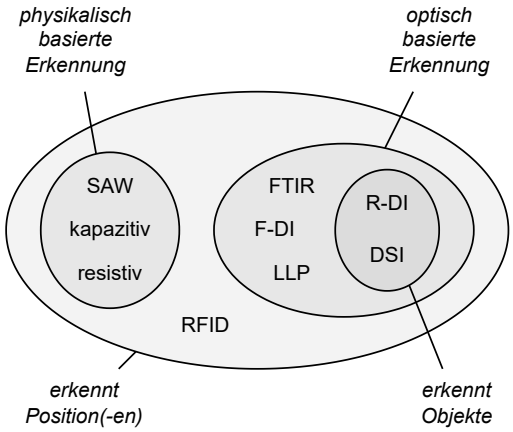


Abbildung 2.1: Eigenschaften verschiedener Technologien - Venn Diagramm

Technologien, die es lediglich ermöglichen Positionen zu erkennen, werden in der Regel verwendet, wenn eine Identifikation von Touch- oder Stifteingaben ausreichend ist. Die Verwendung optischer Varianten, zur Erkennung von positionellen Eingaben, bietet zusätzlich die Option, eine unbegrenzte Menge an verschiedenen Positionen auf der Interaktionsfläche gleichzeitig identifizieren zu können. Ein expliziter Vorteil mancher optischer Verfahren liegt außerdem darin, dass die Erkennung von Interaktionen auf der Grundlage der verfügbaren visuellen Informationen beliebig erweitert werden kann. So ermöglichen die Varianten R-DI und DSI ebenfalls die Erkennung von sich auf oder über der Interaktionsfläche befindlichen Objekten. Ein Nachteil der optischen Verfahren ist jedoch, dass vorherrschendes Umgebungslicht die Erkennung stören kann.

2.2 Umgebung

Die Durchführung der Arbeit findet im Living Place Labor statt. Es wurde 2009 gegründet und ist Teil des Forschungs- und Transferzentrums Smart Systems an der Fakultät Technik und Informatik der HAW Hamburg [4]. Das Labor entspricht der räumlichen Umgebung, in der die Ziele dieser Arbeit umgesetzt werden.

2.2.1 Living Place Labor

Das Living Place Labor besteht aus einer 140 m^2 großen Wohnungsfläche im Loft-Stil mit angrenzenden Büros, einem Kontrollraum und einer Werkstatt. Mit vollständig eingerichteten Bereichen zum Kochen (Kitchen), Speisen (Dining Room), Schlafen (Bedroom), Arbeiten (Longe) und zur Körperpflege (Restroom), bietet der Wohnbereich alle notwendigen Aspekte einer natürlichen Wohnstätte. Es stellt damit eine realitätsnahe Umgebung zur Untersuchung von Ubiquitous-Computing-Systemen sowie der Interaktion zwischen Menschen und der Technologien, die sie umgeben dar [4].

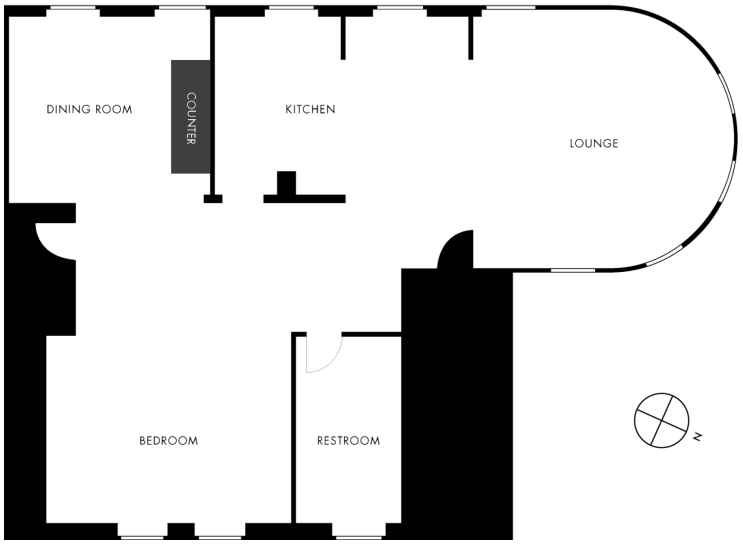


Abbildung 2.2: Living Place Labor - Grundriss der Wohnfläche [4]

Das Living Place Labor verfügt über eine digitale Infrastruktur, in der mittels der Verwendung moderner Smart Home Produkte und eigens gestalteter intelligenter Artefakte mit der Wohnung über eine zentrale MQTT-Schnittstelle interagiert werden kann. So können beispielsweise die Lichtverhältnisse der gesamten Wohnung angepasst werden,

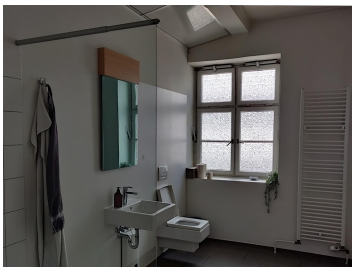
indem Verschattung an den Fenstern individuell heruntergelassen bzw. hochgefahren, die Deckenlampen ein- bzw. ausgeschaltet oder mithilfe von RGB-Leuchtmitteln bestimmte Lichtstimmungen in den jeweiligen dedizierten Bereichen realisiert werden. Darüber hinaus ermöglichen intelligente Thermostate und motorisierte Fensteröffner die Erfassung und Kontrolle der Umgebungstemperatur. Zusätzlich besteht die Möglichkeit den gesamten Bereich durch Kameras und Mikrofone zu überwachen.

(a) Essbereich

(b) Küche

(c) Lounge

(d) Schlafbereich



(e) Badezimmer

Abbildung 2.3: Living Place Labor - Fotos

Zwischen der Küche und dem Esszimmer befindet sich direkt neben einem Fenster ein Tresen (Counter, siehe Abbildung 2.2), welcher im Rahmen dieser Arbeit, entsprechend der im folgenden Kapitel 2.3 definierten Anforderungen, als Ausgangspunkt für Hardware-Implementationen dient.

2.2.2 Tresen

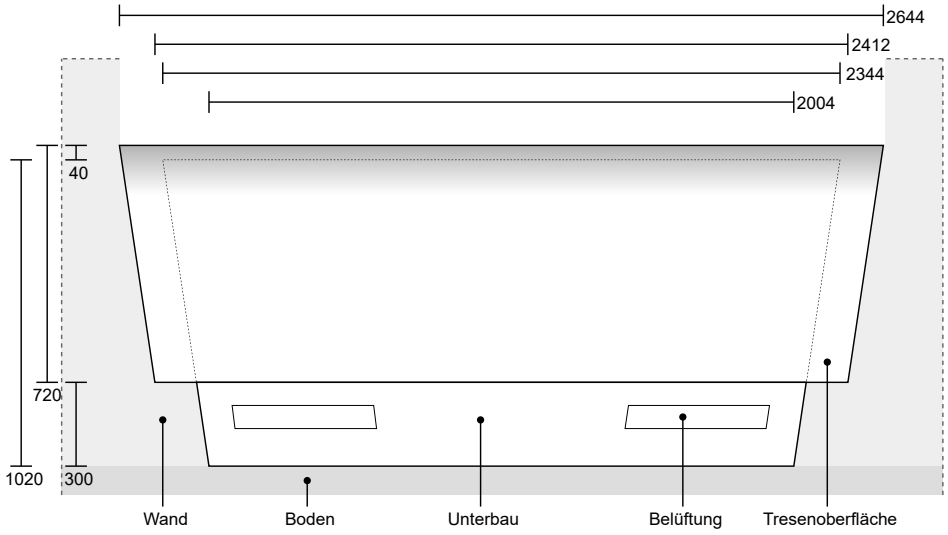
Der Tresen besteht aus einer Kombination einer weißen Platte als Oberfläche und einem grauen Unterbau. Die Oberfläche hat über ihre Breite eine Glasplatte eingelassen, welche von der Unterseite angeraut ist und somit als semitransparente Fläche zur Darstellung von Projektionen verwendet werden kann. Um sie vor Abnutzung zu schützen, ist eine gleich große Abdeckung aus Acrylglas vorhanden. Der Unterbau ist hohl und kann durch

Türen an der linken wie rechten Seite erreicht werden. Belüftungsschlitze mit Ventilatoren gewährleisten einen kontinuierlichen Luftaustausch.

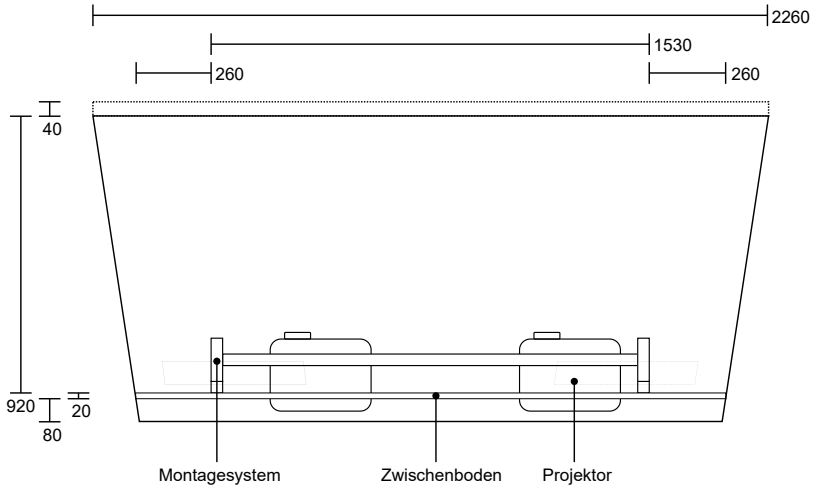
Abbildung 2.4: Tresen - Foto

Im Inneren des Unterbaus ist, knapp über dem Boden des Raumes, ein weiterer Zwischenboden mit einer Aussparung eingezogen. Auf diesem Zwischenboden ist ein Montagesystem, bestehend aus standardisierten 40 mm breiten Aluminiumprofilen, befestigt (siehe Abbildung 2.8). Das Montagesystem umfasst zwei parallele Y-Führungsachsen, an denen jeweils zwei Z-Achsen montiert sind. Diese Z-Achsen können gemeinsam entlang der Y-Richtung frei bewegt werden und sind über eine dritte X-Achse miteinander verbunden. Die X-Achse ermöglicht nicht nur lineare Bewegungen, sondern auch eine Drehung um sich selbst. An ihr sind Aluminiumplatten befestigt, an denen wiederum Projektoren montiert sind. Es handelt sich dabei um zwei Projektoren der Firma *Mitsubishi* vom Typ *EW230U-ST*. Durch den Aufbau des Montagesystems lassen sich diese Projektoren präzise entlang der Achsen ausrichten.

Die folgenden Seiten stellen den Tresen schematisch in seinem Aufbau dar. Es sei angemerkt, dass alle in dieser Ausarbeitung angegebenen Maße bzw. Distanzen, sofern nicht anders definiert, sich auf die Einheit Millimeter beziehen.

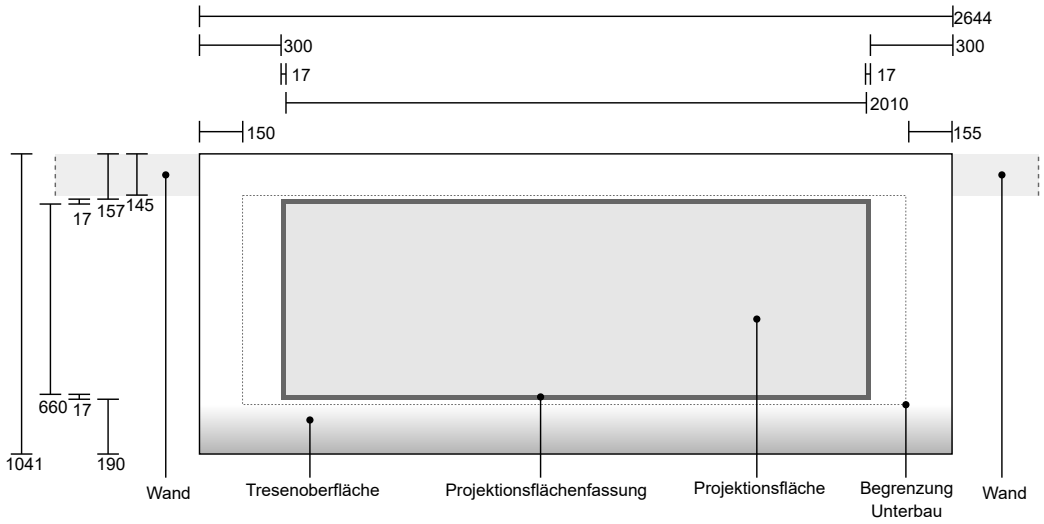


(a) Außen

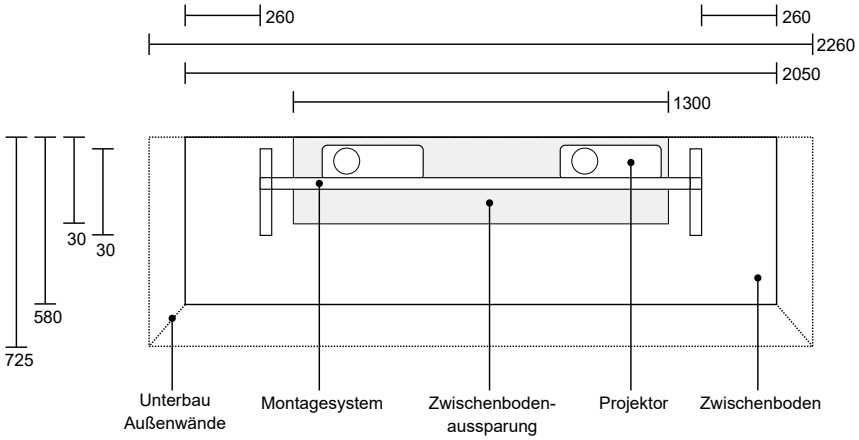


(b) Innen

Abbildung 2.5: Tresen Frontalansicht (schematisch)

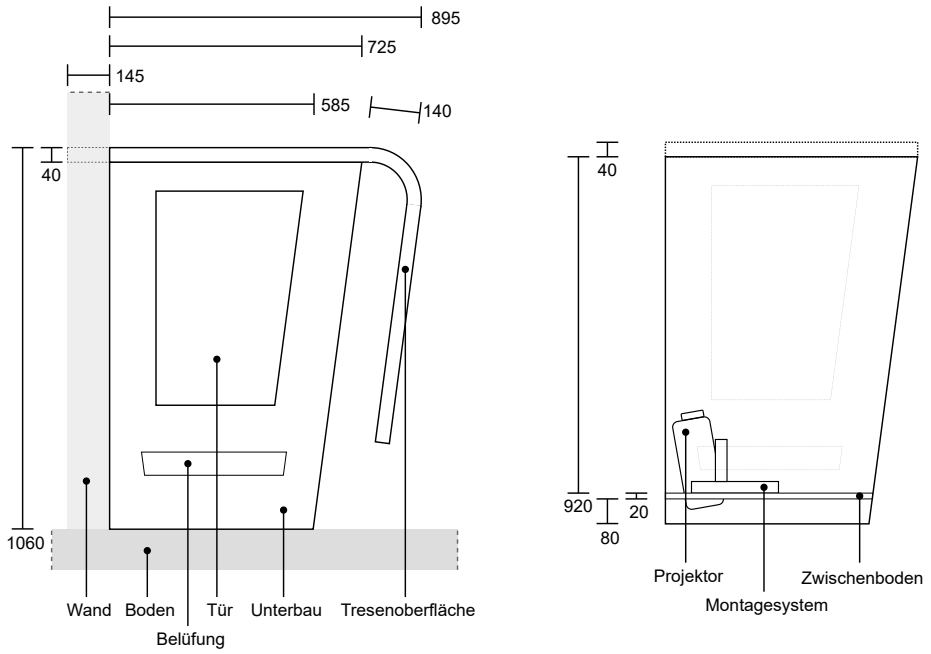


(a) Außen



(b) Innen

Abbildung 2.6: Tresen Draufsicht (schematisch)



(a) Außen (b) Innen

Abbildung 2.7: Tresen linke Seitenansicht (schematisch)

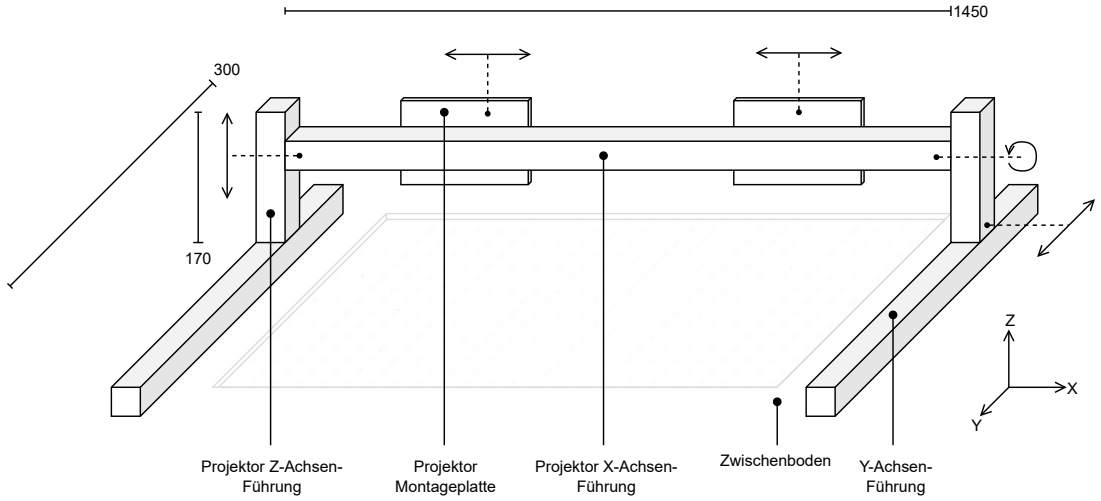


Abbildung 2.8: Montagesystem Basis (schematisch)

2.3 Anforderungen

Beginnend muss definiert werden, welche Anforderungen das zu realisierende System erfüllen muss. Hierbei werden Anforderungen aufgeführt, welche als Rahmenbedingung zur Aufgabenstellung dieser Bachelorarbeit aufgestellt wurden, als auch jene, welche sich aus der Zielsetzung der Arbeit ableiten. Es wird dabei zwischen funktionalen und nichtfunktionalen Anforderungen unterschieden [25].

2.3.1 Funktionale Anforderungen

Die funktionalen Anforderungen beschreiben die umzusetzenden Funktionalitäten sowie Eigenschaften des Systems.

Umgebung

Das zu realisierende System soll im Living Place Labor der HAW Hamburg entwickelt werden. Die erforderliche Hardware soll in das bestehende Mobiliar des Tresens zwischen dem Essbereich und der Küche integriert werden. Dabei soll das System entsprechend der Form und des Aufbaus des Tresens entworfen werden.

Funktion

Das zu realisierende System soll unter der Verwendung bildgestützter Verfahren dazu in der Lage sein, visuelle Informationen zu Interaktionen eines Nutzers mit der Projektionsfläche des Tresens zu erfassen und identifizieren zu können.

Zu identifizierende Interaktionen

Die Interaktion mit der Projektionsfläche soll in dieser Arbeit konkret nur unter der Zuhilfenahme von markierten Objekten erfolgen. Die zu identifizierenden Interaktionen sind dabei das Platzieren, Verschieben, Rotieren und Entfernen unterscheidbarer Objekte. Dabei ist die Art der Markierung frei wählbar, die Größe der Objekte soll jedoch die Maße eines Eishockeypucks nicht überschreiten.

Visualisierungen

Das zu realisierende System soll in der Lage sein, einem Nutzer visuelle Informationen auf der Projektionsfläche darzustellen. Dazu soll konkret eine Projektion auf die Unterseite der Projektionsfläche verwendet werden. Die Visualisierungen sollen den gesamten Bereich der Projektionsfläche abdecken und ausschließlich darauf begrenzt sein. Zudem soll die Visualisierung die Möglichkeit der Realisierung eines Feedbacks für Interaktionen bieten können.

Exemplarität

Das zu realisierende System soll, als Beispiel einer konkreten Verwendung, die Möglichkeit bieten als Schnittstelle zur Kontrolle einer ausgewählten Menge von Smart Home Funktionalitäten des Livin Place Labors verwendet zu werden. Dabei sollen zur Eingabe des Nutzers die zuvor definierten Interaktionen mittels markierter Objekte verwendet werden. Die Wahl der zu steuernden Funktionalität steht frei.

2.3.2 Nicht-funktionale Anforderungen

Nicht-funktionale Anforderungen beschreiben die Anforderungen an die Qualität eines Systems, die über die bloße Erfüllung seiner Funktionen hinausgehen.

Wartbarkeit

Das zu realisierende System soll über eine modular aufgebaute Architektur verfügen, um zukünftige Änderungen bzw. Erweiterungen des Systems effizient umsetzen zu können.

Zuverlässigkeit

Das zu realisierende System soll robust gegenüber vorherrschenden Umgebungseinflüssen sein, welche die Methodik der Erfassung mittels bildgestützter Verfahren negativ beeinflussen könnten, um die Identifikation von Interaktionen mit der Projektionsfläche zu gewährleisten. Sollten solche Einflüsse nicht direkt ersichtlich sein, müssen sie identifiziert werden.

Zusätzlich soll das zu realisierende System fehlertolerant sein. Dabei sollen auftretende Fehler erkannt und entsprechend darauf reagiert werden, um die Funktionalität des Systems aufrecht zu erhalten.

Effizienz

Das zu realisierende System soll ein Zeitverhalten mit hinreichend schneller Verarbeitung gewährleisten, um Latenzen zu vermeiden, die vom Nutzer als nicht reaktiv wahrgenommen werden könnten.

Zusätzlich soll das zu realisierende System einen effizienten Ressourcenverbrauch aufweisen und den Speicherbedarf sowie die benötigte Rechenleistung möglichst gering halten.

3 Konstruktion

Das Konstruktionskapitel befasst sich mit der Realisierung des Systems gemäß den Anforderungen. Dabei werden in den jeweiligen Kapiteln die verschiedenen Aspekte vom Konzept bis zur vollständigen Realisierung des Systems betrachtet.

3.1 Konzept

Begonnen wird damit ein Konzept für die Realisierung aufzustellen, welches die Hardwarebausteine und deren Funktion untereinander festlegt sowie ein Prinzip der Funktionsweise der Software-Implementation und deren Architektur vermittelt. Es dient im weiteren Verlauf als Grundlage für konkrete Designentscheidungen und Implementationen.

3.1.1 Hardware und Funktion

Die Projektionsfläche des Tresens dient als Bereich für etwaig damit möglicher Interaktionen. In dieser Arbeit wird sich konkret auf das Prinzip des Platzieren, Verschieben, Rotieren und Entfernen von markierten unterscheidbaren Objekten zur Interaktion beschränkt. Die Projektionsfläche ist semitransparent und ermöglicht somit gleichzeitig Projektionen auf ihr darzustellen, als auch visuelle Informationen durch sie hindurch zu erfassen. Sie stellt damit ein Verbindungsglied zwischen der realen Welt und der virtuell repräsentierten Interaktionen dar.
Zur Erkennung von Interaktionen lichtet ein Kamerasystem die Unterseite der Projektionsfläche vollständig ab und erfasst damit visuell identifizierbare Informationen bezüglich Interaktionen. Diese basieren auf der Reflexion von mit der Projektionsfläche interagierenden Objekten. Zu diesem Zweck wird der Bereich über der Projektionsfläche durch sie hindurch gewinnbringend beleuchtet. Die Kameradaten, in Form der aufgenommenen Bilder, werden von einem Computersystem (Verarbeitungshardware) verarbeitet und auf

Interaktionen analysiert. Die Ergebnisse dieser Analyse werden in der Form repräsentativer Interaktionsdaten über eine Schnittstelle bereitgestellt. Diese werden von Anwendungen verwendet, um anwendungsfallspezifische Reaktionen auf Interaktionen zu realisieren. Die Darstellung visuellen Feedbacks wird dabei von dem Anwendung ausführenden PC gerendert und mittels eines Projektorsystems auf die Projektionsfläche projiziert. Auf diese Weise wird dem Nutzer ermöglicht eine anwendungsspezifische Rückmeldung über die getätigten Interaktionen zu erhalten.

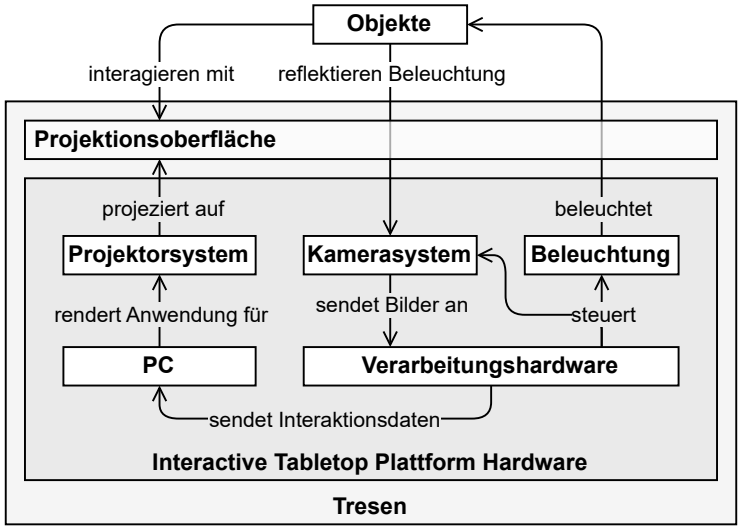


Abbildung 3.1: Hardwarekonzept - Blockdiagramm

Betrachtet man die verschiedenen Optionen an Technologien zur Identifikation von Interaktionen aus Kapitel 2.1, so lässt sich das hier dargestellte Konzept als Vertreter der Rear Diffuse Illumination (R-DI) klassifizieren. Der Ansatz eines derartigen Hardwarekonzeptes hat bereits zu erfolgreichen Realisierungen eines Interactive Tabletop, den *reacTable* von Sergi Jordà et al. [20], geführt. Die dabei angeführten Vorteile bestehen darin, dass der Nutzer das System als Kompaktlösung wahrnimmt, ohne das Projektoren oder Kameras direkt sichtbar sind. Damit gliedert sich das System in den Kontext der Untersuchung von Ubiquitous Computing des Living Place Labors ein [4]. Ebenfalls angemerkt werden die möglichen Freiheitsgrade bei der Wahl und Anzahl der interagierenden Objekte. Für letzteres besteht der limitierende Faktor lediglich in der Größe der Projektionsfläche selbst. Ein potentieller Nachteil der Verwendung bildgestützter Verfahren besteht allerdings darin, dass diese bei hohen Auflösungen einen ebenso hohen Berechnungsaufwand aufweisen können. Dies könnte zu erhöhten Latenzen führen.

3.1.2 Architektur

Die Systemarchitektur des Gesamtsystems wird zugunsten der Modularität und Erweiterbarkeit als eine *two-tiered* Architektur [35, vgl. Kapitel 2.3] auf zwei Ebenen entworfen. Dies ermöglicht eine Trennung der Zuständigkeiten. So kann der Prozess der Verarbeitung und Identifikation von Interaktionen selbst sowie Anwendungen, welche diese zur Realisierung konkreter interaktiver Anwendungsfälle verwenden, auf Basis einer gemeinsamen Schnittstelle, getrennt voneinander betrachtet werden. Ebenfalls ermöglicht die Wahl dieser Architektur eine physikalische Trennung beider Ebenen, wie im Hardwarekonzept durch das Aufteilen zwischen der Verarbeitungshardware und des Anwendung ausführenden PCs dargestellt. Um diese Trennung auch logisch zu erreichen werden die Ebenen in mehrere Schichten entsprechend des Open Systems Interconnection (OSI) Referenzmodelles [19, vgl. Kapitel 6] unterteilt. Es handelt sich somit ebenfalls um eine *multi-layered* Architektur [35, vgl. Kapitel 2.1]. Dabei stellt die Erkennung physikalischer Interaktionen und darauf basierender Reaktionen die Anwendungsschicht dar (layer 7). Diese basieren auf der Repräsentation der Interaktionen und der entsprechenden Modellierung des im Anwendungskontext damit beschriebenen Verhaltens (layer 6). Die Form der Interaktionsdaten wird durch ein Protokoll definiert (layer 5) und mittels Transportmechanismen (layer 4) über das Netzwerk ausgetauscht (layer 3).

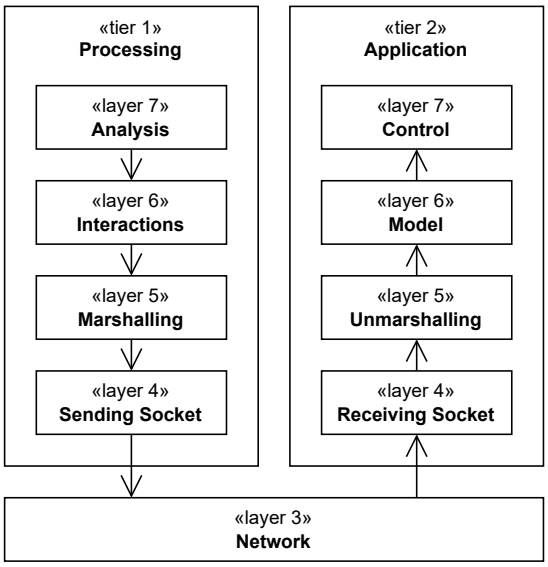


Abbildung 3.2: Konzept der Systemarchitektur

Eine vergleichbare Trennung von Ebenen zur Interaktionserkennung und konkreter Anwendungen wird auch im *reacTIVision* Framework von Martin Kaltenbrunner und Ross Bencina verwendet [22].

Folgend wird die Systemarchitektur der ersten Ebene des Gesamtsystems, im weiteren Verlauf dieser Ausarbeitung als Verarbeitungssystem bezeichnet, weiter differenziert. Zur funktionalen Abgrenzung und Modularisierung wird diese in drei Hauptkomponenten als Stufen einer Pipelinearchitektur [33, vgl. Kapitel 11] entworfen. Die erste Hauptkomponente, die Bildakquisition (Image Aquisition), abstrahiert die Steuerung der Beleuchtung und den Empfang von Bildern des Kamerasystems. Diese Bilddaten werden mittels eines unidirektionalen Datenstroms (frame stream) an die nächste Stufe weitergeleitet. Die Funktion dieser Stufe entspricht damit der eines Produzenten. Die zweite Hauptkomponente ist die Bildverarbeitung (Visual Processing), welche eine Aufbereitung der Kamerabilder vornimmt und anschließend ebenfalls weiterleitet. Sie beschreibt damit die Funktion eines Transformators. Die dritte und letzte Hauptkomponente ist die Merkmalsextraktion (Feature Extraction). Diese analysiert die aufbereiteten Kamerabilder und identifiziert relevante Merkmale bzw. Muster, welche als Interaktionen interpretiert werden können und stellt diese wiederum als Interaktionsdaten (interaction data) der Anwendung auf zweiter Ebene bereit. Als letzte Stufe der Pipeline stellt sie damit einen Konsumenten dar, welcher die finalen Resultate nach außen gesammelt anbietet.

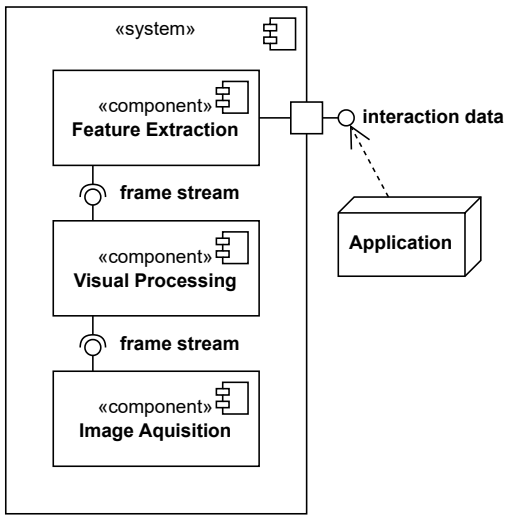


Abbildung 3.3: Konzept des Verarbeitungssystems - Komponentendiagramm

Der Vorteil der Verwendung einer Pipelinearchitektur besteht darin, dass bei der Verarbeitung der Kamerabilder deren Reihenfolge entsprechend des Aufnahmezeitpunktes beibehalten bleibt. Dies ist wichtig, da mit kontinuierlicher Veränderung der aufgenommenen realen Umgebung eine zeitliche Abhängigkeit zwischen den Bilddaten besteht. Darüber hinaus ist die Verarbeitung durch eine Pipeline gut parallelisierbar, was bei einer hohen Bildrate zur Vermeidung von stetig steigenden Latenzen elementar notwendig ist.

Damit die Erkennung von Interaktionen, außerhalb des limitierten Umfanges dieser Arbeit, zukünftig erweitert werden kann, wird die Erkennung unterschiedlicher konkreter Interaktionstypen als Sammlung von Modulen entworfen. Jedes Modul erhält dafür ein aktuelles aufbereitetes Kamerabild und führt individuell Analysen auf Interaktionen entsprechend dessen Spezialisierung durch. Die dabei identifizierten Merkmale (features) werden dann pro Bild gesammelt und in der Form repräsentativer Interaktionsdaten nach außen angeboten. So kann ein konkretes Modul die in dieser Arbeit geforderte Erkennung von markierten Objekten implementieren und ein weiteres in Zukunft hinzugefügt werden, welches beispielsweise die Erkennung von Toucheingaben o.Ä. implementiert. Die Erkennung spezifischer Interaktionen ist damit ausschließlich auf die Analysierbarkeit der Bilddaten mithilfe von Computer-Vision-Verfahren beschränkt.

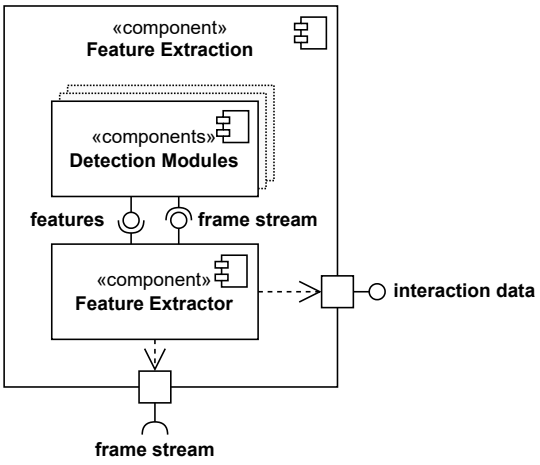


Abbildung 3.4: Konzept Hauptkomponente Feature Extraction - Komponentendiagramm

3.2 Voruntersuchung

Bevor die Realisierung des Systems konkretisiert werden kann, müssen Sachverhalte ermittelt werden, welche für das Treffen von konkreten Designentscheidungen zur Erfüllung der Anforderungen notwendig sind. Diese Randbedingungen werden mittels der Durchführung der folgenden Experimente identifiziert.

3.2.1 Identifikation von Umgebungseinflüssen

Die Anforderungen beschreiben die Notwendigkeit bei der Erkennung, unter Verwendung bildgestützer Verfahren, robust gegenüber dabei störender Umgebungseinflüsse zu sein. Um dieser Anforderung entsprechen zu können, muss zuerst ermittelt werden, welche potenziellen Umgebungseinflüsse es gibt und wie sich diese auswirken. Damit stellt sich die zu bestimmende Frage:

Welche Auswirkungen haben äußere Quellen von infrarotem Licht[1] *auf die Qualität der Bildinformationen?*

Versuchsaufbau

Verwendet wird ein Raspberry Pi 4 Einplatinencomputer in Kombination mit einem NoIR-Camera Modul. Die Kamera wird in Blickrichtung der linken Projektionsflächenunterseite auf dem Zwischenboden des Tresen-Unterbau positioniert. Zusätzlich wird ein Infrarot Langpassfilter mit einem Durchlassbereich ab 800 nm auf die Optik des NoIR-Moduls gelegt.
Der Raspberry Pi ermöglicht eine einfache und schnelle Möglichkeit live Kamerabilder einzusehen und aufzunehmen. Das NoIR-Modul ist ohne expliziten Infrarot blockierenden Filter ausgestattet, wie es bei üblichen Farbkameras meist der Fall ist. Dadurch lassen sich auch Wellenlängen, die außerhalb der menschlichen Wahrnehmung liegen erfassen. Die Verwendung des Langpassfilters limitiert wiederum das betrachtete Spektrum auf den Infrarotbereich, sodass sich nur darauf fokussiert werden kann.

[1] Vorgriff auf explizite Verwendung von infrarotem Licht. Für Begründung siehe Kapitel 3.3.1.

Durchführung

Zur Identifikation einzelner Kandidaten von Fremdeinstrahlungsquellen wird, unter Beobachtung eines Livebildes der Kamera, die Beleuchtung des Living Place Labors variiert. Dazu wird die Innenbeleuchtung an- und abgeschaltet sowie mithilfe der Verschattungsmöglichkeiten der Raum abgedunkelt und wieder erhellt. Ebenfalls wird der Einfluss der Projektoren und der Beleuchtung im inneren des Tresens im ein- wie abgeschalteten Zustand betrachtet. Anschließend wird das Vorgehen wiederholt und Aufnahmen für alle 16 möglichen Kombinationen an Zuständen angefertigt. Diese Bilder werden miteinander verglichen um Intensität, Art und Bereich einer möglichen Störung identifizieren zu können. Als Exemplare für Objekte, welche in einem Anwendungsfall mit der Projektionsfläche interagieren würden, wurde ein bedrucktes Blatt Papier, Gegenstände verschiedener Form und ein Zeigefinger auf dieser platziert.

Beobachtung

Das Einschalten der Projektoren resultiert in keiner wahrnehmbaren Änderung des Kamerabildes. Beim Variieren der Beleuchtung jedoch, fallen Differenzen auf. So resultieren sowohl das Einschalten der Innenbeleuchtung als auch die Ausleuchtung des Raumes durch die Sonne, in hellen Bereichen des Kamerabildes, ohne an dieser Stelle ein Objekt platziert zu haben, welches Licht dem Konzept nach reflektieren würde. Im Gegenteil dazu werden Bereiche, in denen ein Objekt platziert ist durch deren Verschattung sogar abgedunkelt. Dabei fällt auf, dass der Schattenwurf bei Sonnenlicht in deutlich härteren Kanten resultiert, verglichen mit dem Schattenwurf bei eingeschalteter Innenbeleuchtung. Betrachtet man die Aufnahmen, in welchen die Objekte zusätzlich von unten durch die Beleuchtung angestrahlt werden, stellt man fest, dass sich deren Reflexionen nur schwach, bis gar nicht gegen die Fremdeinstrahlungsquellen absetzten. Darüber hinaus lässt sich feststellen, dass die Projektionsfläche selbst einen kleinen Anteil der Beleuchtung reflektiert. Ebenfalls fällt auf, dass trotz der ausgeschalteten Innenbeleuchtung und Verschattung ein kleiner Restanteil an Umgebungslicht auf dem Kamerabild identifizierbar bleibt.

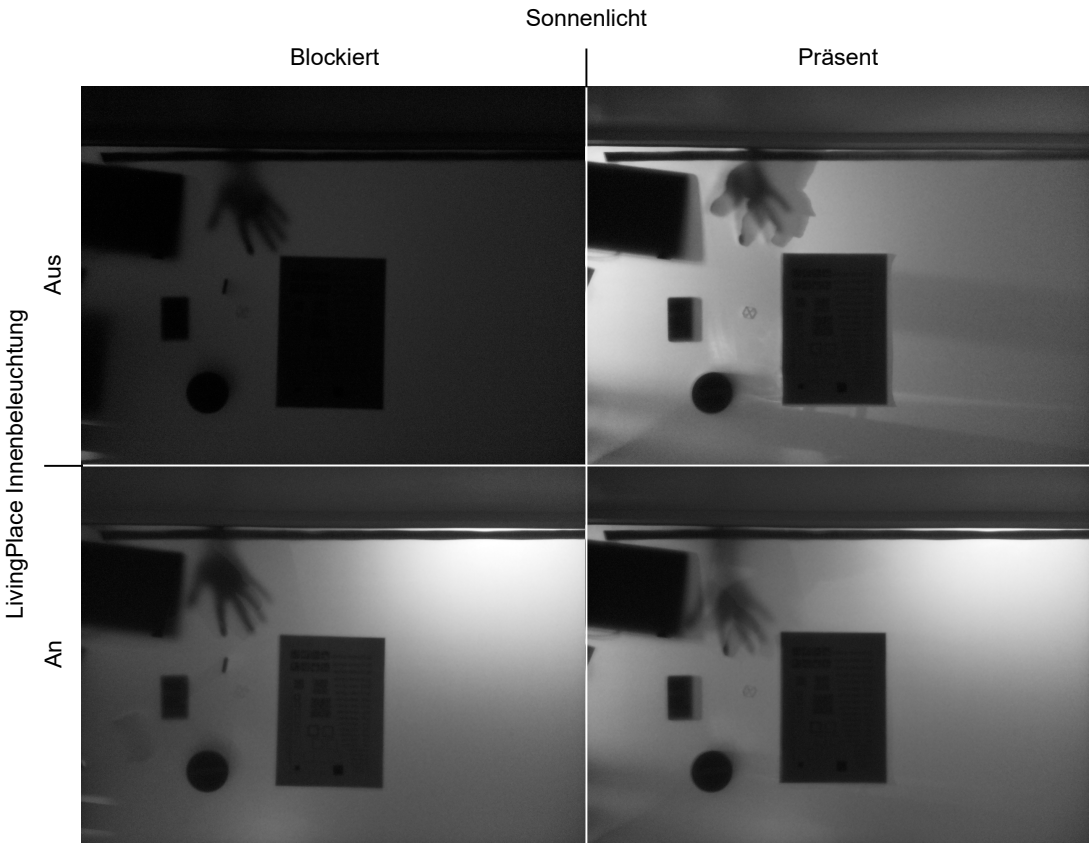
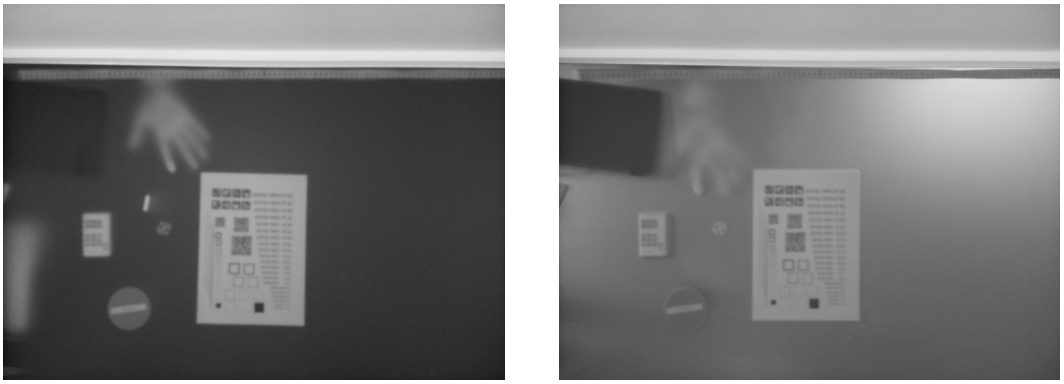


Abbildung 3.5: Gegenüberstellung signifikanter Fremdeinstrahlungsquellen



(a) Nur Beleuchtung (b) Beleuchtung und Fremdeinstrahlungsquellen

Abbildung 3.6: Absetzen von Objektreflexionen gegenüber Fremdeinstrahlung

Auswertung

Da der Betriebszustand der Projektoren keinen Einfluss auf das Kamerabild hatte, kann davon ausgegangen werden, dass die Projektion keinen Infrarotanteil enthält. Bei der Innenbeleuchtung des Living Place hingegen lässt sich diese als signifikante Fremdeinstrahlungsquelle identifizieren. Die helle Stelle im oberen rechten Bereich der erfassten Bilder resultiert dabei konkret durch einen Halogenstrahler, welcher direkt auf die Projektionsfläche gerichtet ist. Eine ebenso signifikante Fremdeinstrahlungsquelle stellt die Sonneneinstrahlung dar. Im Vergleich mit der Innenbeleuchtung ist diese dazu stark variabel, da sie von Bewölkung, Jahres- wie Tageszeit und deren Einfallswinkel einen unterschiedlich starken Einfluss auf das Kamerabild hat. Die Fremdeinstrahlungsquellen führen dazu, dass helle Stellen im Kamerabild nicht direkt als Reflexionen von Objekten angenommen werden können und sie die Differenzierung zwischen ihnen und Reflexionen von beleuchteten Objekten generell verkomplizieren. Die minimale Reflexion der Projektionsfläche, unabhängig eines darauf platzierten Objektes, lässt sich vernachlässigen, da sie auf dem gesamten Bereich identisch ausfällt. Das verbleibende Umgebungslicht bei abgeschalteter Innenbeleuchtung und Abschattung resultiert aus der Imperfektion letzterer.

Ergebnis

Der Einfall von Sonnenlicht und die Innenbeleuchtung des Living Place Labors stellen Quellen an infraroten Licht dar, welche sich störend auf die Funktionsweise des Konzeptes zur Erkennung von Objekten durch deren Reflexion auswirken. So lassen sich anhand der Bildinformationen keine direkten Rückschlüsse ziehen, ob eine helle Stelle auf Grundlage einer Reflexion oder einer Fremdeinstrahlungsquelle resultiert. Dies legt nahe, dass eine Isolation von Reflexionsinformationen vorgenommen werden muss, um mit dem im Konzept vorgestellten Ansatz weiter verfahren zu können.

3.2.2 Isolation von Reflexionsinformationen

Auf Grundlage der zuvor erlangten Erkenntnisse, über den Einfluss von Umgebungseinflüssen, muss nun untersucht werden, mit welchen Methoden eine Isolation der Reflexionsinformationen vorgenommen werden kann. Dabei muss das Verfahren adaptiv

zu der Dynamik sich ständig verändernder Präsenz bzw. Intensitäten der verschiedenen Fremdeinstrahlungsquellen sein.

Als Lösungsansatz wird folgendes Modell angenommen: Man betrachte den Anteil des Lichtes, welcher durch die Fremdeinstrahlungsquellen resultiert L_F und den Anteil des Lichtes, welcher durch Reflexionen auf Grundlage der Beleuchtung resultiert L_R. Da die Beleuchtung direkt angesteuert werden kann, ist der Anteil von L_R am Kamerabild kontrollierbar. Betrachtet man nun die Option zwei Bilder, $Bild_{Belichtet}$ und $Bild_{Unbelichtet}$, mit unterschiedlicher Komposition derer abgebildeten Lichteinflüsse anzufertigen, ermöglicht dies die Isolation der Reflexionsinformationen durch Bildung derer Differenz:

$$Bild_{Belichtet} = (L_R + L_F), Bild_{Unbelichtet} = (L_F)$$

$$Bild_{Differenz} = Bild_{Belichtet} - Bild_{Unbelichtet} = (L_R + L_F) - (L_F) = L_R$$

Mithilfe dieser Methode wäre eine Isolation der Reflexionsinformationen theoretisch realisierbar. Es stellt sich nun die zu ergründende Frage:

In welchem Umfang ist das Vorgehen einer Differenzbildbildung zur Isolation von Reflexionsinformationen praktisch nutzbringend?

Durchführung

Auf Grundlage der erhobenen Daten aus der Identifikation zu Umgebungseinflüssen im Kapitel 3.2.1 lässt sich die Methode auf ihre Praktikabilität prüfen. Zu diesem Zweck werden Bildpaare, deren einziger Unterschied die Präsenz bzw. Abwesenheit von Lichteinflüssen der Beleuchtung sind, ausgewählt und entsprechend der Methode verarbeitet. Die Ergebnisse werden anschließend darauf untersucht, inwieweit sie die Reflexionsinformationen isoliert repräsentieren.

Beobachtung

Die einzelnen Objekte sind deutlich zu erkennen und es werden keine Bereiche mehr festgestellt, welche ohne die Platzierung eines Objektes fälschlicherweise Reflexionsinformationen verdeutlichen würden. Es lässt sich ein leichtes Bildrauschen identifizieren.

In Bereichen, in denen das unbelichtete Bild durch die Intensität der Fremdeinstrahlungsquelle sehr hell ist, sind die Ränder zu Objekten sehr scharf. In Bereichen, wo der Bildsensor annähernd oder voll ausgesteuert wird, dass es zur Überbelichtung kommt, können Teile des Randes entfernt werden. Dieser Umstand scheint eher aufzutreten, wenn das Objekt nicht vollständig in Kontakt mit der Projektionsfläche ist.

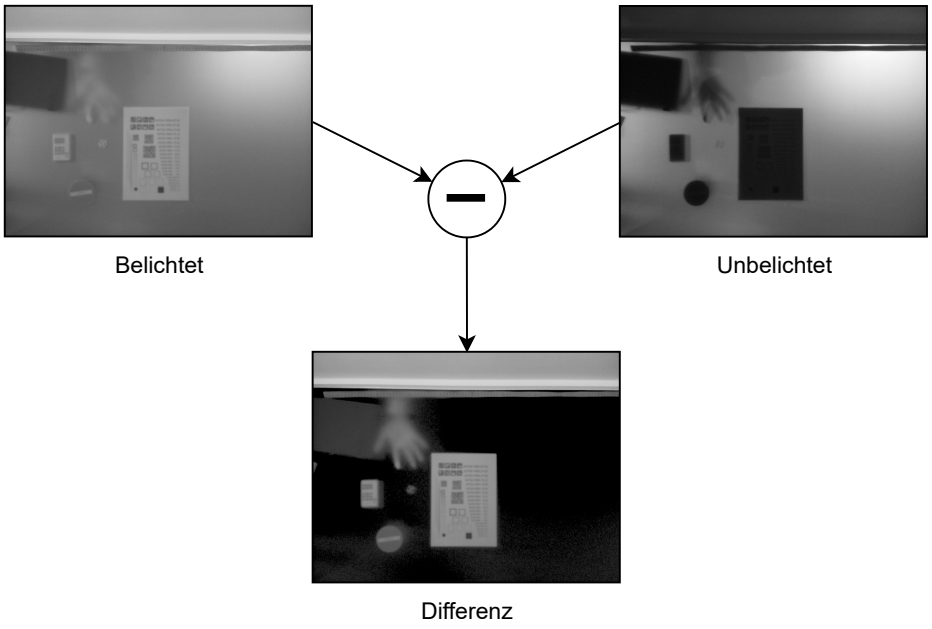


Abbildung 3.7: Differenzbildbildung zur Isolation von Reflexionsinformationen

Auswertung

Der zuvor beobachtete Umstand, dass sich die Reflexionen nur schwach bis gar nicht gegen die Fremdeinstrahlungsquellen absetzen, wird durch das Verfahren vollständig mitigiert. Das Bildrauschen resultiert vermutlich aus Differenzen im natürlichen Bildrauschen der beiden Einzelbilder. Die Elimination von Reflexionsinformationen an den Rändern von Objekten in Bereichen mit einer hohen Bildsensoraussteuerung, ist entsprechend der Operation einer Subtraktion logisch. Es ist ratsam sich in solchen Ausnahmefällen auf Interaktionen zu beschränken, welche direkten Kontakt mit der Projektionsfläche haben. Der Verwendung von Objekten mit semitransparenten Eigenschaften zur Interaktionserkennung ist bei Verwendung der Methode abzuraten.
Die logische Konsequenz aus diesem Vorgehen und ein sich daraus entwickelnder Nachteil für die Anwendung der Methodik liegt darin, dass die effektive Bilderfassungsrate

verdoppelt werden muss, um eine angestrebte Zielbildverarbeitungsrate weiterhin zu erreichen. Sie setzt außerdem die Möglichkeit der Koordination von Aufnahmezeitpunkt und dabei herrschender Belichtung durch die Beleuchtung voraus.

Ergebnis

Das Verfahren der Differenzbildbildung ermöglicht die Isolation von Reflexionsinformationen. In Situationen einer starken Fremdeinstrahlung können Details der Reflexionsinformationen an Rändern von Objekten ebenfalls entfernt werden. Die Anwendung der Methode führt darüber hinaus zu einer signifikanten Verbesserung der Qualität von Reflexionsinformationen. Es ergeben sich durch sie jedoch auch neue Anforderungen.

3.2.3 Einfluss der Projektionsfläche auf Objektdetailgrad

Bei der Untersuchung zur Identifikation von Umgebungseinflüssen in Kapitel 3.2.1 ist unabhängig des zu untersuchenden Gegenstandes aufgefallen, dass die Projektionsfläche selbst direkten Einfluss auf die Qualität der Reflexion von Objekten hat. So lässt sich in den aufgenommenen Bildern erkennen, dass Details wie der Text auf dem Blatt Papier, welches als exemplarisches Objekt auf der Projektionsfläche platziert wurde, verschwommen erscheinen. Da dieser Umstand einen Einfluss darauf hat welche Markierungen auf Objekten zuverlässig erkannt werden können, muss konkretisiert werden, wie sich dieser Einfluss gestaltet. Es stellt sich damit die konkrete Frage:

Ab welcher Informationsdichte lassen sich Details durch den Einfluss der Projektionsfläche nicht mehr zuverlässig rekonstruieren?

Versuchsaufbau

Auf die Projektionsfläche wird ein mit einem Testmuster bedrucktes DIN A4 Papier positioniert und beschwert, sodass der Kontakt zwischen Papier und der Projektionsfläche sichergestellt ist. Das Testmuster teilt sich in zehn Bereiche, welche alle eine individuelle Informationsdichte mit einem Detailgrad von einem bis zehn Millimeter abbilden (siehe Abbildung 3.8a). Jeder Bereich stellt dabei sowohl ein binäres eindimensionales Muster,

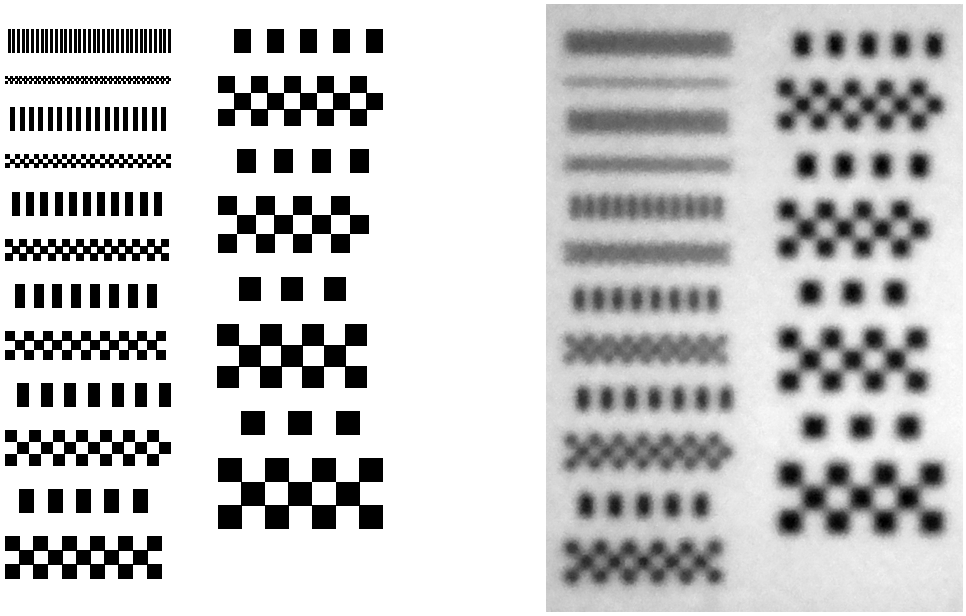
ähnlich einem regelmäßigen Barcode, als auch ein binäres zweidimensionale Muster, ähnlich einem Schachbrett, dar. Zur Aufnahme des Testmusters durch die Projektionsfläche wird der Versuchsaufbau aus Kapitel 3.2.1 wiederverwendet.

Durchführung

Das DIN A4 Papier mit dem Testmuster wird durch die Projektionsfläche abgelichtet. Anschließend wird die Ablichtung des Testmusters durch die Projektionsfläche mit dem Testmuster selbst verglichen. Hierzu werden Querschnitte aus der vertikalen Mitte der zehn Detailgrade entnommen und normalisiert. Um zu überprüfen in welchen Bereichen Fehler durch die Ablichtung durch die Projektionsfläche hindurch resultieren, wird der absolute Fehler für jeden Pixel bestimmt. Für eine Einschätzung der Größenordnung über den generellen Fehler, wird die mittlere quadratische Abweichung MSE errechnet. Um zu überprüfen mit welchem Erfolg das Testmuster rekonstruiert werden kann, wird eine Binarisierung der entnommenen Querschnitte durchgeführt. Der dafür verwendete Schwellenwert wird je Detailgrad durch das Verfahren nach Otsu [31] bestimmt. Anschließend wird der Anteil der korrekt binarisierten Pixel A berechnet.

Beobachtung

Die Ablichtung des Testmusters entspricht diesem in verzerrter Form. Der Einfluss der Projektionsfläche ähnelt optisch dem eines Weichzeichners. Die Rekonstruktionsrate A springt bei einem Detailgrad von 3 mm auf 4 mm im 1D und von 4 mm auf 5 mm im 2D von kleiner 60% auf über 90%. Danach steigt sie sehr langsam. Die Größenordnungen der Fehler MSE im 1D wie 2D sinken bis zu einem Detailgrad von 7 mm nicht kontinuierlich und unterschiedlich von ca. $0,35$ auf ca. $0,06$. Anschließend fallen sie annähernd gleich in geringem Maße weiter. Der größte absolute Fehler besteht in den Bereichen der Übergänge von Null auf Eins und umgekehrt. Bereiche mit einem Wert von Eins haben einen größeren absoluten Fehler als Bereiche mit einem Wert von Null. Die Binarisierung für Detailgrade kleiner 5 mm im 2D und kleiner 4 mm im 1D ist mit Ausnahme der Ränder überwiegend Null. Für jeweilig höhere Detailgrade beider betrachteten Dimensionen ist die Binarisierung annähernd dem Testmuster ähnlich. Differenzen sind besonders in den Übergangsbereichen zu identifizieren, weisen jedoch immer nur einen Sprung auf.



(a) Testmuster (b) Testmuster durch Projektionsfläche

Abbildung 3.8: Gegenüberstellung Testmuster und dessen Ablichtung

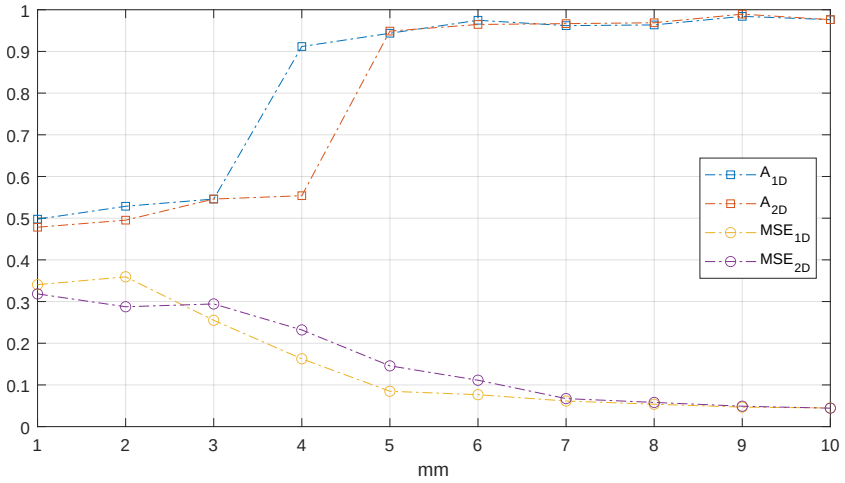
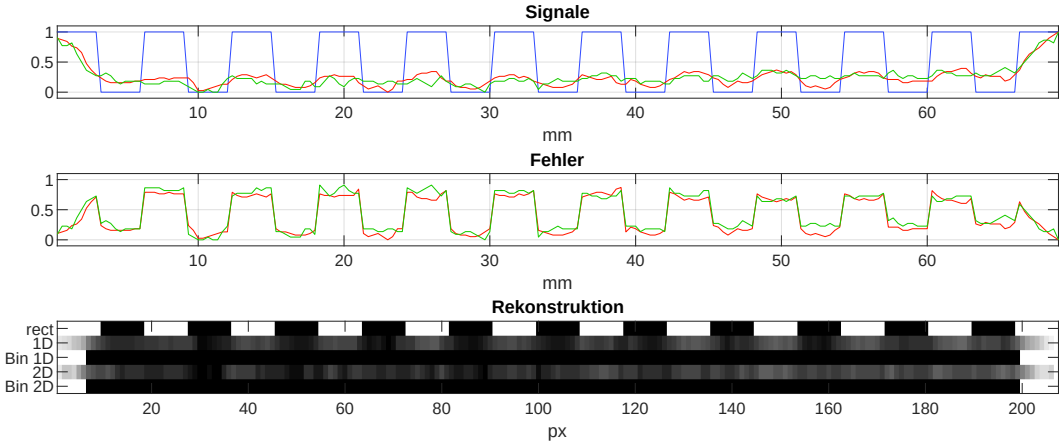
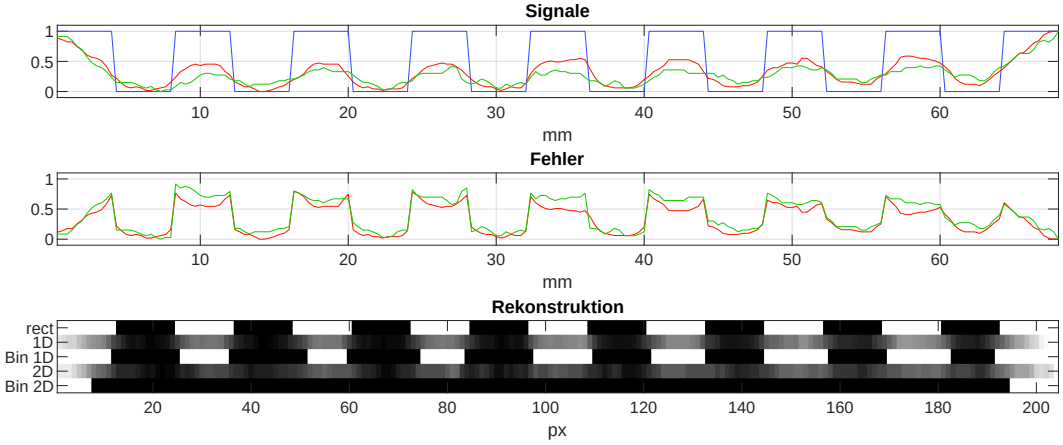


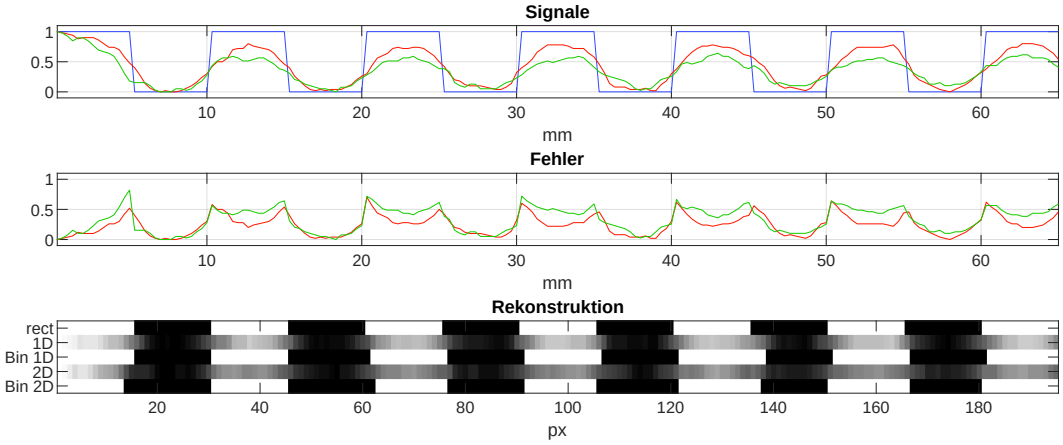
Abbildung 3.9: Rekonstruktionsrate A und Größenordnung des Fehlers MSE bei Ablichtung von Detailgraden von 1 bis 10 mm durch die Projektionsfläche



(a) Einfluss der Projektionsfläche bei 3 mm



(b) Einfluss der Projektionsfläche bei 4 mm



(c) Einfluss der Projektionsfläche bei 5 mm

Abbildung 3.10: Projektionsflächeneinflüsse auf Detailgrade von 3-5 mm (Blau: rect, Rot: 1D, Grün: 2D, "Bin": Rekonstruiert)

Auswertung

Da die Auswahl der möglichen Zustände binär ist, bedeutet eine Rekonstruktionsrate A von 50% einer Genauigkeit vergleichbar mit bloßem Raten. Daher ist eine Rekonstruktion von Informationsdichten mit Detailgraden kleiner 4 mm im 1D und kleiner 5 mm im 2D entsprechend der Beobachtungen nicht zuverlässig möglich. Der Einfluss der Projektionsfläche entwickelt sich ab einem Detailgrad von 7 mm, unabhängig der Dimensionalität, nicht mehr signifikant. Es kann angenommen werden, dass alle höheren Detailgrade einen Fehler ähnlicher Größenordnung aufweisen werden. Die bevorzugte Binarisierung zum Wert Null resultiert durch den Einfluss der weißen Ränder zu jeder Seite der verschiedenen Bereiche. Durch sie wird der Schwellenwert verschoben. Aus selbigem Grund resultiert auch der vergleichsweise größere absolute Fehler in Bereichen des Wertes Eins hingegen der Bereiche mit Wert Null.
Die Betrachtung der Querschnitte erinnert an die Konstruktion eines Rechtecksignals durch Kombination immer höher frequenter Sinus- und Kosinus- Funktionen mit fehlenden hochfrequenten Anteilen. Es wird angenommen, dass eben solche Anteile durch die Projektionsfläche gefiltert werden. Sie agiert somit als Tiefpassfilter.

Ergebnis

Bei Anwendungsfällen zum Erkennen von bestimmten Markierungen, sollte eine Informationsdichte mit einem Detailgrad von minimal 4 mm im Eindimensionalen und 5 mm im Zweidimensionalen nicht unterschritten werden. Darunter lassen sich Details nicht zuverlässig rekonstruieren. Noch bessere Ergebnisse werden mit einem Detailgrad von 7 mm und größer, unabhängig der Dimensionalität, erzielt.

3.3 Sensorik und Aktorik

Durch die zuvor unternommenen Untersuchungen kann folgend die Hardwareauswahl konkretisiert werden. Anschließend wird betrachtet, wie die verschiedenen Hardwarekomponenten untereinander konfiguriert und im Tresen installiert werden.

3.3.1 Hardwareauswahl

In den folgenden Abschnitten wird betrachtet, welche Hardware konkret für die verschiedenen Komponenten des Konzeptes verwendet wird und wie sich die Auswahl dieser untereinander bedingt.

Beleuchtung

Da sowohl die Erkennung von Interaktionen als auch die Visualisierungen für den Nutzer mittels der visuellen Verfahren von Reflexion bzw. Projektion realisiert werden sollen, stehen diese im direkten Konflikt miteinander. So würden die Projektion von Anwendungen auf die Unterseite der Projektionsfläche sich auf die Beleuchtungsmenge der interagierenden Objekte auswirken. Dieser Umstand würde die Reflexionsinformationen negativ beeinflussen, wenn nicht sogar vollständig korrumpieren. Um diesem Umstand zu entgegnen, wird als Lösungsansatz verfolgt, dass für die Beleuchtung ein differenter Teil des elektromagnetischen Spektrums verwendet werden muss, welcher außerhalb des für den Menschen sichtbaren Bereiches liegt. Konkret wird sich hierbei für den Bereich der Infrarotstrahlung entschieden, da diese von Kamerasensoren erfasst werden kann, für den Menschen aber nicht direkt wahrnehmbar ist [29, vgl. S. 230 f., Kapitel 7.2][8].

Auf Basis dieser Anforderung und der Verfügbarkeit von Produkten werden zur Beleuchtung LED-Streifen der Firma *SOLAROX*©[2] verwendet. Diese ermöglichen eine Ausleuchtung im Infrarotbereich mit einer Wellenlänge von 850 nm. Sie haben eine Betriebsspannung von 12 V und weisen einen Nennstrom von 0,4 A pro Meter auf. Es werden 8 m verwendet, wodurch eine Leistung von mindestens 38,4 W bereitgestellt werden muss. Entsprechend wird ein Netzteil der Firma *Mean Well*[3] mit einer Leistung von 60 W verwendet.

Kamerasystem

Aus vorausgegangenen Kapiteln haben sich bereits Anforderungen an das zu konkretisierende Kamerasystem ergeben. So hat die Untersuchung aus Kapitel 3.2.2 gezeigt, dass die

[2] Datenblatt: https://www.led1.de/shop/files/Datenblaetter/Bars-Strips/Solarox_Strip_IR1-60-850.pdf (Zugriffsdatum: 04.05.2023).
[3] Datenblatt: https://www.meanwell-web.com/content/files/pdfs/productPdfs/MW/LPV-60/LPV-60-spec.pdf (Zugriffsdatum: 04.05.2023).

Verwendung der Differenzbildbildung einer effektiv doppelt so hohen Bilderfassungsrate bedarf wie die Zielbildverarbeitungsrate der Interaktionserkennung. Das Kamerasystem muss daher eine hohe Bilderfassungsrate ermöglichen. Zusätzlich bedingt die konkrete Verwendung des infraroten Spektrums eine Empfindlichkeit des Kamerasystems in diesem Bereich.
Über diese Anforderungen hinaus, ergeben sich weitere aus der Dimensionierung der abzulichtenden Projektionsfläche. Entsprechend der Ausführungen aus Kapitel 2.2.2 zur Umgebung, entspricht die Projektionsfläche mit den Maßen von 2010 mm in der Breite und 660 mm in der Höhe einem Seitenverhältnis von 67:22 (siehe Abbildung 2.6). Eine derartig breite Fläche ist problematisch mit nur einer Kamera vollständig abzulichten, da der mögliche Abstand, gemäß der Dimensionen es Tresen-Unterbaus, auf maximal 920 mm begrenzt ist (siehe Abbildung 2.7). Nimmt man dennoch an, dass eine vollständige Ablichtung der Projektionsfläche mit nur einer Kamera erzielt werden könnte, würde das bei typischen Seitenverhältnissen von Bildsensoren wie z.B. 16:10 bedeuten, dass gerade einmal etwas mehr als die Hälfte der Bildinformationen die Projektionsfläche darstellen würden.[4] Es wird daher die Entscheidung getroffen zwei Kameras zu verwenden, die jeweils einen Teilbereich der Projektionsfläche ablichten. Diese Teilbereiche sollen sich dabei zusätzlich überschneiden.
Um zeitliche Inkonsistenzen der Bilddaten im Bereich der Überschneidung zu vermeiden, müssen beide Kameras dazu in der Lage sein, die Bilderfassung zur selben Zeit in derselben Dauer zu tätigen. Diese Anforderung korrespondiert ebenfalls mit der Anforderung der Differenzbildbildung, den Aufnahmezeitpunkt mit dem Zustand der Beleuchtung koordinieren zu können.

Auf Basis dieser Anforderung und der Verfügbarkeit von Produkten werden zwei Exemplare des Kameramodells *Ace acA1920-150um*[5] der Firma *Basler* ausgewählt. Diese verwenden einen Bildsensor im Format 2/3" mit einer monochromen Auflösung von 1920 Pixeln in der Breite und 1200 Pixeln in der Höhe, welcher bei globaler Belichtungsmethode eine Bilderfassungsrate von bis zu 150 Bildern pro Sekunde ermöglicht. Mit einer relativen Empfindlichkeit von ca. 50% bei 850 nm Wellenlänge, ist der Bildsensor dazu in der Lage die Reflexionen der Beleuchtung zu erfassen. Die Bilderfassung kann dabei unter anderem mittels eines sogenannten Hardware-Triggers initiiert werden, wodurch es ermöglicht wird beide Kameras miteinander zu synchronisieren. Aufgrund des Seitenverhältnis des Bildsensors von 16:10 überschneiden sich abzulichtende Teilbereiche. Darüber

[4] Für Herleitung siehe Anhang A.2.
[5] Spezifikationen: https://docs.baslerweb.com/aca1920-150um (Zugriffsdatum: 04.05.2023).

hinaus bieten die Kameras durch Konformität des USB3 Vision Standards und des bereitgestellten Software Development Kit (SDK) *Pylon* eine standardisierte und leicht zu verwendende Schnittstelle für Implementationen in den Programmiersprachen C, C++, .NET und inoffiziell Python.[6]

Objektive

Da die verwendeten Kameras keine eigene Optik besitzen, ist es notwendig entsprechend passende Objektive auszuwählen. Diese müssen mittels des standardisierten C-Mount an der Kamera montiert werden können und mit der Größe des Bildsensor kompatibel sein. Da explizit das Infrarotspektrum betrachtet wird, darf in der Optik kein Infrarot-Sperrfilter integriert sein. Darüber hinaus müssen die Charakteristika so gewählt werden, dass die Ablichtung der einzelnen Teilbereiche der Projektionsfläche möglich ist.

Auf Basis dieser Anforderung und der Verfügbarkeit von Produkten werden die Objektive *HF6XA-5M*[7] der *Fujifilm* Tochterfirma *Fujinon* verwendet. Diese haben eine Brennweite von 6 mm, einen Blendenbereich von F1.9-F16, sind mit einer Sensorgrößen von 2/3" kompatibel und lassen sich mittels eines C-Mount konformen Gewindes an der Kamera befestigen. Das Objektiv hat keinen Infrarot-Sperrfilter vorinstalliert, unterstützt jedoch die Montage von externen Filtern durch ein M37.5 x 0.5 Gewinde. Um die Teilbereiche der Projektionsfläche ablichten zu können, ist entsprechend der Brennweite ein Arbeitsabstand von ca. 700 mm minimal notwendig.[8] Damit ist eine Installation der Kameras, inklusive derer Objektive, in den Tresen-Unterbau mit einer Maximalhöhe von 920 mm ohne Probleme möglich. Aufgrund einer Verzeichnung von -2,88% besteht jedoch der Nachteil, dass diese unter Zuhilfenahme von Bildverarbeitungsmethoden korrigiert werden muss.

Infrarot Filter

Um den Anteil des Lichtes, welcher schlussendlich von den Kameras erfasst wird, auf die Wellenlänge der Beleuchtung von 850 nm zu beschränken, bedarf es passender Filter.

[6] Für Details siehe Pylon Produktwebsite: https://www.baslerweb.com/de/produkte/basler-pylon-camera-software-suite/ (Zugriffsdatum: 04.05.2023).
[7] Produktserie: https://www.fujifilm.com/de/de/business/optical-devices/mvlens/hfxa5m (Zugriffsdatum: 04.05.2023).
[8] Basierend auf Berechnungen mittels des Basler Objektiv-Selektor: https://www.baslerweb.com/de/vertrieb-support/tools/objektiv-selektor/ (Zugriffsdatum: 04.05.2023).

Diese sollen entsprechend der Möglichkeit mittels des von der Optik bereitgestellten Filtergewindes an diesen montierbar sein.

Auf Basis dieser Anforderung und der Verfügbarkeit von Produkten, werden Bandpassfilter vom Typ *BP850*[9] der Firma *Midopt* verwendet. Diese haben einen Durchlassbereich der Wellenlängen von 820 nm bis 910 nm und lassen sich mittels eines passenden Filtergewindes an den Optiken montieren.

Verarbeitungshardware

Folgend wird die bisher lediglich vage definierte Verarbeitungshardware des Konzeptes konkretisiert. Diese muss der Anforderung genügen die Koordination der Bilderfassung und Beleuchtung zu ermöglichen, als auch entsprechend des Konzeptes zur Software-Implementation dazu in der Lage sein, die aufgenommenen Bilder der Kameras zu empfangen, zu verarbeiten, auf Interaktionen zu analysieren und diese in der Form repräsentativer Interaktionsdaten bereitzustellen. Die Anforderung zur Koordination setzt dabei voraus, dass die Verarbeitungshardware dazu in der Lage ist hardwarenahe Operationen, wie das Manipulieren von GPIO-Pins, zu unterstützten, um den Hardware-Trigger der Kameras bzw. Schaltelektronik für die Beleuchtung anzusteuern. Die Anforderungen bezüglich der Verarbeitung von Kameradaten sind hingegen die Verfügbarkeit einer ausreichend schnellen USB 3 Schnittstelle sowie der Möglichkeit Bildverarbeitungsmethoden mit ausreichender Geschwindigkeit berechnen zu können.

Da die Anwendungsfälle der Bildverarbeitung und der Koordination zur Ausführung stark differente Hardwareprofile benötigen, wird die Entscheidung getroffen, die Verarbeitungshardware selbst als verteiltes System zu konzipieren. Entsprechend wird ein Desktop PC des Labors dazu verwendet die Software-Implementation bezüglich der Verarbeitung von Bildern und der Interaktionsanalyse auszuführen. Die Koordination der Bilderfassung und Beleuchtung wird mittels eines Raspberry Pi 3 realisiert.

Tabelle 3.1: Verarbeitungshardware Desktop PC - Systemspezifikation

Motherboard	Asus TUF Z270 MARK 2
CPU	Intel Core i7-7700K (4x 4,2 GHz)
RAM	64 GB DDR4
GPU	NVIDIA GeForce GTX 1080Ti
OS	Windows 10 Professional 64-bit

[9] Produkt: https://midopt.com/filters/bp850/ (Zugriffsdatum: 04.05.2023).

LED Schaltelektronik

Da der Raspberry Pi das Schalten der Spannungen von 12V nicht nativ unterstützt, muss dessen Funktionsumfang mithilfe passender Schaltelektronik erweitert werden. Dabei muss die Schaltgeschwindigkeit mindestens der maximal möglichen Bilderfassungsrate von 150 Hz entsprechen und die elektrischen Charakteristika den benötigten Anforderungen der Beleuchtung entsprechen.

Auf Basis dieser Anforderungen und der Verfügbarkeit, wird die *Dual TB9051FTG Motor Driver HAT*[10] Erweiterung der Firma *Pololu* verwendet. Dieses ursprünglich zur Ansteuerung von Motoren gedachte Erweiterungsboard ermöglicht es pro Kanal eine Spannung von bis zu 28 V bei einem kontinuierlichen Nennstrom bis zu 2,6 A mit einer Frequenz von maximal 20 kHz zu schalten. Über die zwei Kanäle lassen sich damit jeweils bis zu 6,5 m der LED-Streifen schalten. Die Wahl eines Erweiterungsboards mit einer GPIO-Schnittstelle ermöglicht darüber hinaus die Skalierung der Beleuchtung und garantiert eine schnelle Reaktionszeit. Bei Bedarf kann damit die Beleuchtungsmenge ohne Mehraufwand durch ein weiteres Exemplar erweitert werden.

Projektorsystem

Zur Darstellung von Visualisierungen für den Nutzer werden die bereits vorhandenen Projektoren *EW230U-ST*[11] der Firma *Mitsubishi* verwendet. Sie unterstützen eine maximale Auflösung von 1280 Pixeln in der Breite sowie 800 Pixeln in der Höhe und sind für die Projektion auf kurze Distanzen ausgelegt. Aus vergleichbaren Gründen der Verwendung von zwei Kameras, werden auch zwei Projektoren verwendet. Aufgrund dieses Umstandes stellt sich die Anforderungen die Projektionen der jeweiligen Projektoren derartig zu gestalten, dass für den Nutzer ein einheitliches Gesamtbild entsteht.

Auf Basis dieser Anforderungen und der Verfügbarkeit, wird der Desktop PC zusätzlich mit der Grafikkarte *Quadro K2200*[12] der Firma *NVIDIA* ausgerüstet. Diese ermöglicht die Anpassung der jeweils projizierten Bilder und Bereiche. Mehr dazu im Kapitel 3.8.

[10] Produktspezifikationen: https://www.pololu.com/product/2761/specs (Zugriffsdatum: 04.05.2023).
[11] Produktprospekt: https://www.hcinema.de/pdf/mitsubishiew230ust-de.pdf (Zugriffsdatum: 04.05.2023).
[12] Datenblatt: https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/documents/75509_DS_NV_Quadro_K2200_US_NV_HR.pdf (Zugriffsdatum: 04.05.2023).

3.3.2 Installation

Mit einer konkreten Wahl an Hardwarekomponenten abgeschlossen, kann folgend betrachtet werden, wie diese im Tresen montiert und miteinander konfiguriert werden.

Montage

Betrachtet man die Installation der Projektoren aus Kapitel 2.2.2, so lässt sich dieses Konzept auch für die Kameras anwenden. Basierend auf dem bereits vorhanden Montagesystem wird daher eine Erweiterung dessen vorgenommen:

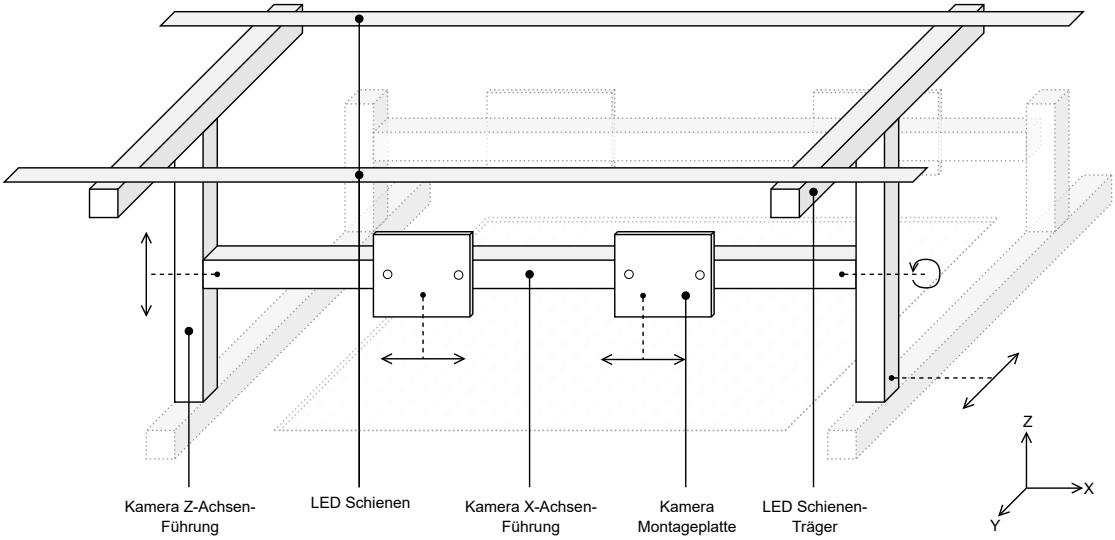


Abbildung 3.11: Montagesystem Erweiterung (schematisch)

Das Montagesystem wird um eine Achse für die Positionierung in Z-Richtung und eine Achse für die Positionierung in Y-Richtung erweitert. Außerdem ist eine Drehung auf der X-Achse möglich. Die Manipulation dieser Freiheitsgrade beeinflusst beide Kameras gleichermaßen. So kann sowohl der Abstand zur Projektionsfläche als auch die Positionierung mittig zur kurzen Seite der Projektionsfläche einheitlich gestaltet werden. Dafür werden die Kameras an Montageplatten befestigt, welche wiederum an der Kamera X-Achsen-Führung befestigt werden. Die Montageplatten lassen sich somit entlang der langen Seite der Projektionsfläche individuell verschieben. Mit dieser Methode lassen sich die abzulichtenden Teilbereiche der Kameras beliebig, aber eindeutig definieren. Darüber hinaus ist das Konzept dazu noch vollständig modular sowie rekonfigurierbar.

Für die Installation der Beleuchtung werden je 4 m der LED-Streifen in zwei Bahnen auf Schienen aufgebracht. Diese Schienen dienen dabei nicht nur als Trägermaterial für strukturelle Integrität, sondern agieren gleichzeitig als Kühlkörper. Sie werden anschließend mithilfe der LED-Schienen Träger entlang der Projektionsfläche montiert. Durch die regelmäßige Verteilung der einzelnen LEDs entlang der X-Achse wird die Projektionsfläche gleichmäßig ausgeleuchtet. Die erhöhte Position verhindert dabei einen direkten Lichteinfall in die Kameras.

Konfiguration

Die Konfiguration der einzelnen Komponenten untereinander wird entsprechend der Abbildung 3.12 vorgenommen. Dabei werden Klemmen verwendet, um der Anforderung der Wartbarkeit nachzukommen.

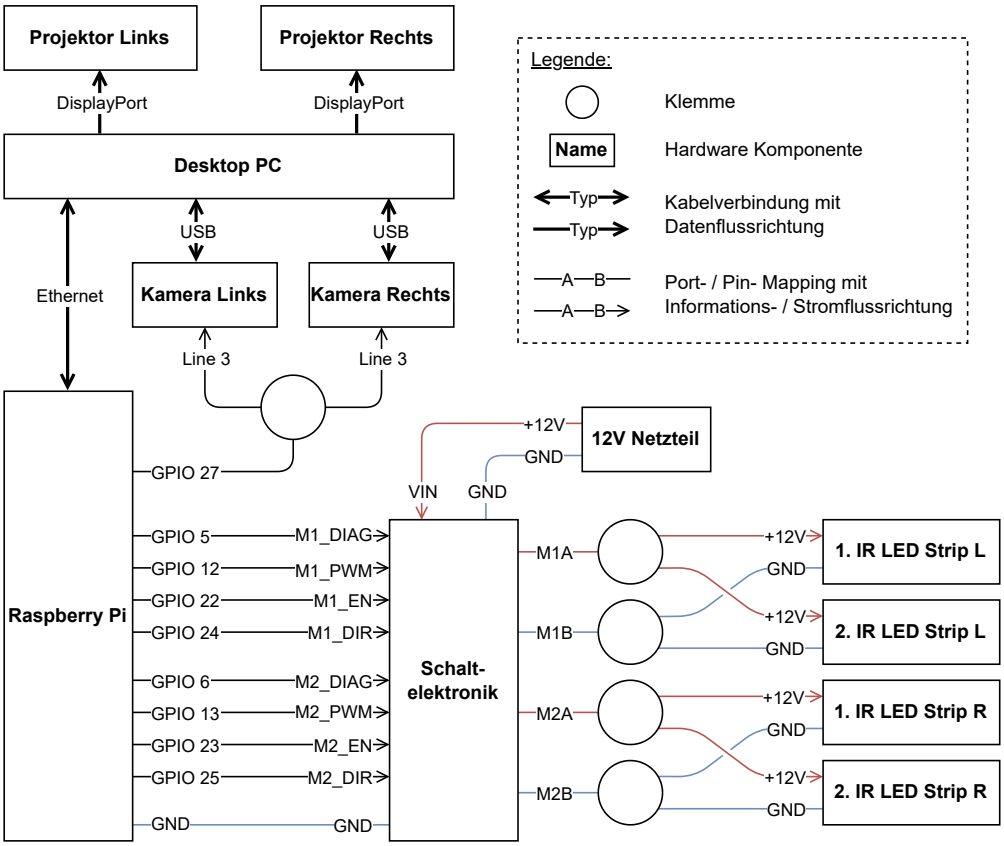
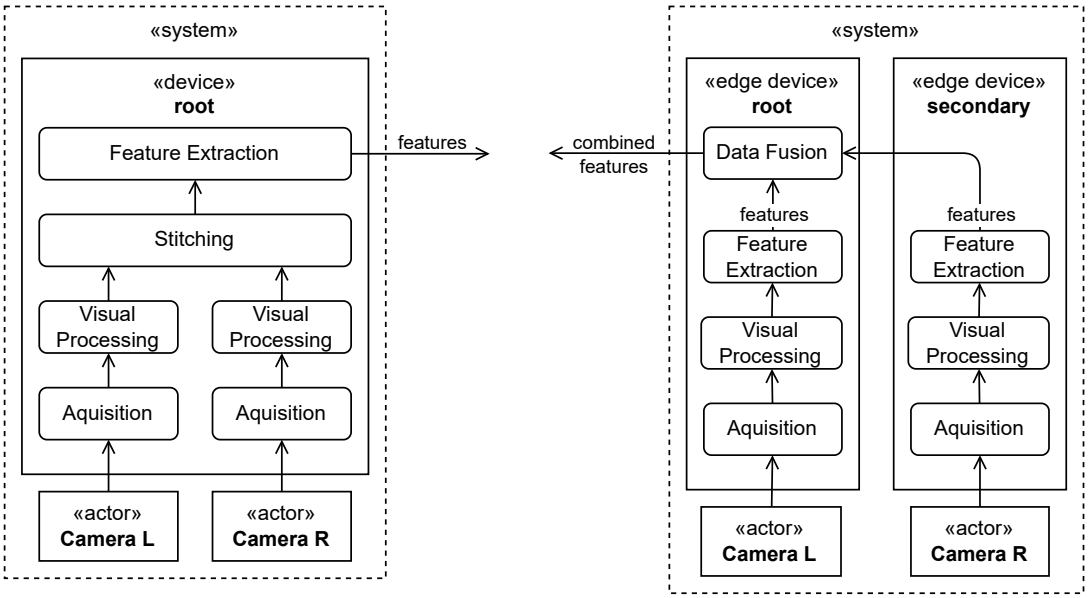


Abbildung 3.12: Hardware Konfiguration - Blockschaltbild

3.4 Systemtopologie

Mit dem Konzept der Hardware vollständig konkretisiert und der Installation abgeschlossen, kann nun das Konzept der Software-Implementation ausdifferenziert werden. Begonnen wird dabei mit der Wahl einer Systemtopologie des Verarbeitungssystems.

Es werden hier zwei Optionen betrachtet: Eine zentralisierte Topologie mit der Verarbeitung der Kamerabilder multipler Quellen in einem monolithischen System und eine verteilte Topologie mit der Verarbeitung der Kamerabilder je Quelle on Edge mit einer anschließenden, für die Anwendung transparenten, Datenfusion der resultierenden Teilergebnisse.



(a) Zentralisiert (b) Verteilt

Abbildung 3.13: Optionen zur Systemtopologie

Ein Vorteil der Verwendung einer verteilten Topologie ist deren Skalierbarkeit und Modularität. So könnten zukünftig weitere Instanzen hinzugefügt oder ersetzt werden, welche kleinere Teilbereiche der Projektionsoberfläche ablichten und verarbeiten würden. Hierbei ließe sich die konkrete Wahl der Kamerahardware individuell abstrahieren. Zusätzlich würde die Ausführungszeit einzelner Erkennungsmechanismen, durch die direkte Parallelisierung und Betrachtung kleinerer Bereiche, verringert werden. Im direkten Kontrast dazu steht jedoch die erhöhte Komplexität der Implementation von verteilten Erkennungsmechanismen, sowie die Notwendigkeit der Koordination derer verschiedener ver-

teilter Instanzen. Dieser Mehraufwand entfällt bei der Verwendung der zentralisierten Topologie, da die Erkennungsmechanismen hier lediglich auf einem System zur Ausführung kommen. Es besteht in diesem Fall keine Notwendigkeit der Koordination. Ein Nachteil entsteht jedoch daraus, dass die Operation verschiedene Kamerabilder zu einem kombinierten Kamerabild zusammenzufassen, aufgrund der Verarbeitung von Bilddaten, deutlich rechenintensiver ist als die Datenfusion der extrahierten Metadaten im verteilten Ansatz.
Es wird schlussendlich die Wahl der Verwendung einer zentralisierten Topologie getroffen, da die resultierenden Vorteile, wie Modularität und Skalierbarkeit des verteilten Ansatzes, mit den Nachteilen, wie der steigenden Komplexität der Entwicklung und Erweiterung des Systems, in keinem gewinnbringenden Verhältnis miteinander steht.

Im Rahmen dieser Arbeit, wird das Deployment beider Ebenen des Gesamtsystems auf einem Device umgesetzt. Entgegen des initialen Hardwarekonzept werden somit die Verarbeitungshardware und der Anwendung ausführende PC durch denselben Desktop PC repräsentiert. Dies begründet sich durch einen vereinfachten Hardwarebeschaffungsprozess. Die Möglichkeit einer zukünftigen Trennung wird dadurch nicht beeinträchtigt.

3.5 Bildakquisition

Die Bildakquisition entspricht der ersten Hauptkomponente des Verarbeitungssystems. Sie ist gemäß dem Konzept für die Koordination der Bilderfassung und Beleuchtung sowie das Empfangen von Bilddaten seitens der Kamerahardware verantwortlich. Im Folgenden wird die konkrete Implementation dieser beschrieben und welche Abstraktionen dabei für eine modulare und erweiterbare Architektur getroffen werden.

3.5.1 Frame coordinator

Um den Zeitpunkt der Kamerabildaufnahme und die dabei vorherrschende Beleuchtung zu koordinieren, wird ein spezielles Koordinationselement, der *Frame Coordinator*, entworfen. Dieser muss in konfigurierbaren regelmäßigen Abständen die Kameras einmal während ein- und folgend bei ausgeschalteter Beleuchtung zum Auslösen bringen. Gemäß den dafür nötigen hardwarenahen Operationen wird es als verteilte Softwarekomponente entwickelt. Zu diesem Zweck wird eine Schnittstelle definiert, welche das Starten und Stoppen der Koordination sowie das Konfigurieren der Bilderfassungsrate ermöglicht.

Zur Vollständigkeit wird auch das einmalige Auslösen der Kameras definiert. Die Teilverantwortlichkeiten der konkreten Implementation bezüglich der regelmäßigen Taktung, der Steuerung der Beleuchtung und das Auslösen der Kameras, wird dabei selbst durch Schnittstellen abstrahiert. Dies ermöglicht die Erweiterbarkeit des *Frame Coordinator* über den in dieser Arbeit verwendeten Zweck der Differenzbildbildung hinaus.

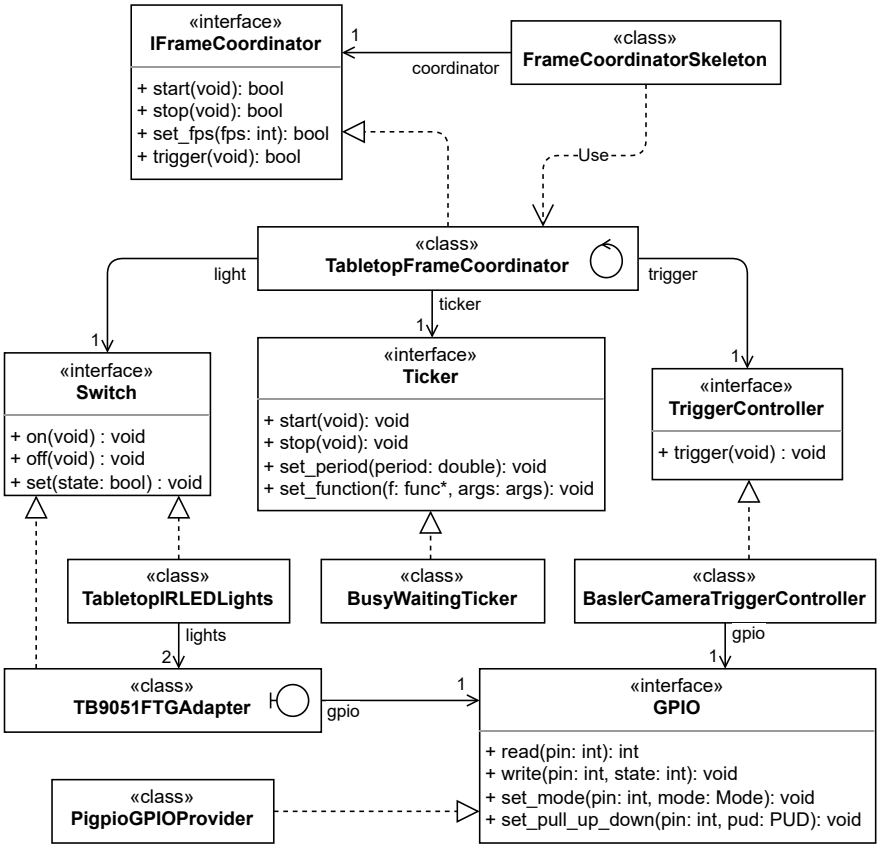


Abbildung 3.14: Frame Coordinator - Klassendiagramm

Die Funktionalitäten des *Frame Coordinator* werden nach Außen als Dienst angeboten. Dies ermöglicht die transparente Ausführung hardwarenaher Operationen aus dem Softwarekontext des restlichen Verarbeitungssystems unter Verwendung eines Remote Procedure Call (RPC) [35, vgl. Kapitel 4.2].
Die eingeführte Komponente des *Frame Coordinator* und dessen Abstraktion mittels einer Schnittstelle, ermöglicht damit eine erweiterbare Steuerung des Verhaltens über die Bildgenerierung in der Bildakquisition und somit für das gesamte Verarbeitungssystem.

3.5.2 Abstraktionen

Um das Verarbeitungssystem unabhängig von spezifischer Kamerahardware und Verarbeitungsframeworks zu gestalten, werden entsprechende Abstraktionen eingeführt.

Bilddaten

Sämtliche Bilddaten werden durch das Datenobjekt *Frame* repräsentiert. Ein *Frame* speichert dabei die Dimensionen der Bilddaten und einen Zeiger auf einen Datenpuffer der konkreten Bildpixelinformationen. Zur Erleichterung der Speicherverwaltung handelt es sich dabei um einen *shared pointer*. Dies sorgt dafür, dass der Datenpuffer der Bilddaten solange vorgehalten wird, wie mindestens ein *Frame* Objekt darauf verweist. Sobald ein *Frame* also von keiner Komponente mehr verwendet wird, wird der verwendete Speicherbereich freigegeben.[13] Zusätzlich können einem *Frame* Metainformationen mittels der Speicherung von Schlüsselwerten hinzugefügt werden. Hier werden einer frei wählbaren Zeichenkette ein Wert eines primitiven Datentyps oder ebenfalls einer Zeichenkette zugeordnet. Dies ermöglicht eine dynamische Erweiterbarkeit des Datenobjekts zur Laufzeit. Durch die Abstraktion der Bilddaten in ein verallgemeinertes Objekt, wird sich damit auf kein spezielles Format eines Kameraherstellers oder auf die Verwendung eines speziellen Frameworks zur Verarbeitung von Bilddaten festgelegt. In Ausnahmefällen kann dies zur Notwendigkeit der Konvertierung führen. Für gewöhnlich kann jedoch der Datenpuffer von monochromen Bilddaten der Größe 1 Byte pro Pixel in jedes gängige Bildverarbeitungsframework überführt werden.[14]

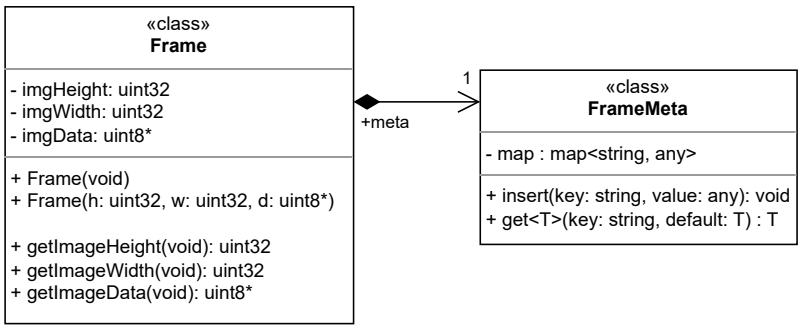


Abbildung 3.15: Abstraktion Bilddaten - Klassendiagramm

[13] Für Details siehe Dokumentation: https://en.cppreference.com/w/cpp/memory/shared_ptr (Zugriffsdatum: 09.05.2023).
[14] Für ein Beispiel anhand des Computer Vision Frameworks OpenCV siehe Anhang A.3.

Unidirektionale Bilddatenströme

Die unidirektionalen Datenströme an Bilddaten werden mittels eines Sender Empfänger Mechanismus realisiert. Die verschiedenen Spezialisierungen der abstrakten *FrameReceiver* Klasse, erlauben es, auch asynchrone Bildverarbeitungsprozesse zu implementieren.

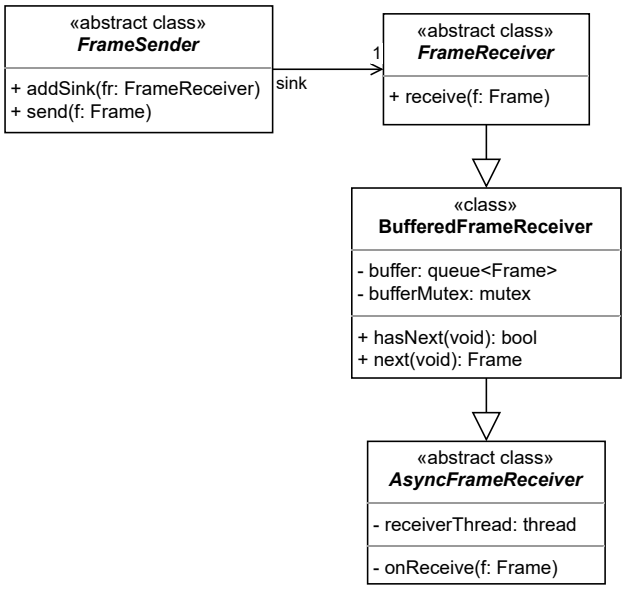


Abbildung 3.16: Abstraktion Bilddatenströme - Klassendiagramm

Kameras

Die Einführung einer allgemeinen *FrameSender* Klasse, als Quelle für beliebige Bilddaten, ermöglicht nun die Abstraktion jeglicher Kamerahardware und der zugehörigen Hardwareabstraktion bzw. API. Zu diesem Zweck wird eine allgemeine Schnittstelle *ICamera* definiert, welche das Starten und das Stoppen des Empfang von Kameradaten definiert. Die Verwendung einer konkreten Kamerahardware als Bildquelle wird nun durch die Spezialisierung der Klasse *FrameSender* und der Implementation der Schnittstelle *ICamera* ermöglicht. Am Beispiel dieser Arbeit wird also die Klasse *BaslerCamera* entworfen, welche als Adapter für die hardwarespezifische Implementation *CBaslerInstantCamera* der dazugehörigen *Pylon* API kapselt und sie damit für das Verarbeitungssystem als Spezialisierung einer *FrameSender* Klasse verfügbar macht (siehe Abbildung 3.17).

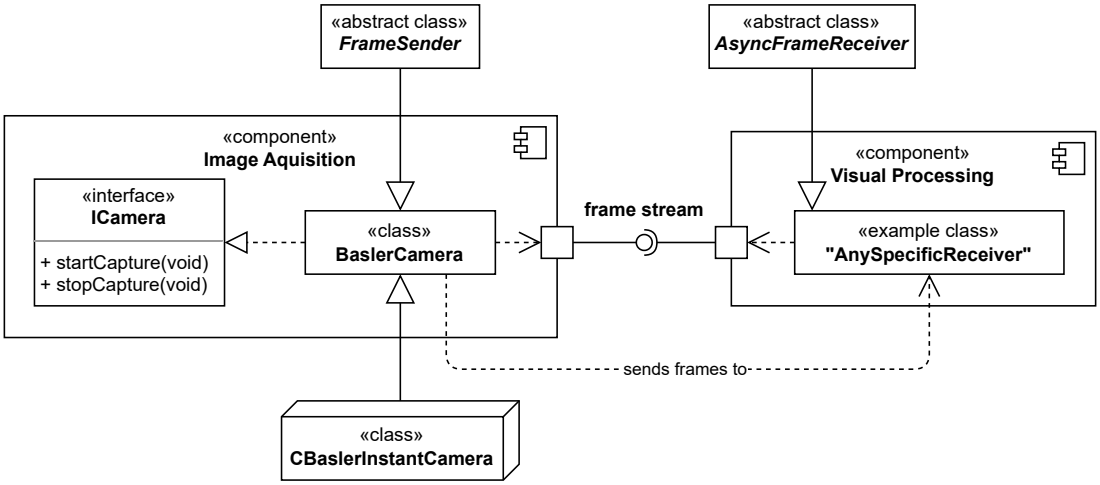


Abbildung 3.17: Abstraktion Kamera - Klassendiagramm

3.5.3 Emulation

Um die Modularität der getroffenen Abstraktionen zu demonstrieren sowie zur Erfüllung der nicht funktionalen Anforderung der Wartbarkeit, wird eine weitere Bildquelle implementiert: Eine emulierte Kamera. Diese kann zu Debugging Zwecken verwendet werden und ermöglicht die Ausführung der Software ohne die direkte Verfügbarkeit des Hardwareaufbaus. Dafür werden zuvor aufgenommene Bilder des Realsystems von der Festplatte in den Arbeitsspeicher geladen und die Funktion einer Kamera emuliert. Die Koordination der Bildgenerierung wird dabei durch eine passende Implementation der *IFrameCoordinator* Schnittstelle *EmulatedFrameCoordinator* realisiert.

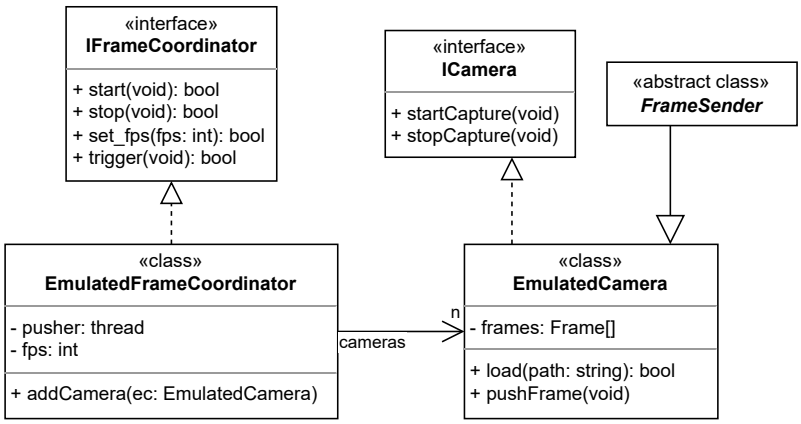


Abbildung 3.18: Emulierte Kameras - Klassendiagramm

3.6 Bildverarbeitung

Die Bildverarbeitung entspricht der zweiten Hauptkomponente des Verarbeitungssystems. Sie ist gemäß dem Konzept für die Verarbeitung der direkten Kamerabilddaten verantwortlich. Im Folgenden werden die konkreten Bildverarbeitungsschritte und deren Umsetzung beschrieben.

3.6.1 Filterung und Korrekturen

Begonnen wird mit der Filterung und Korrektur der rohen Kamerabilddaten, gegeben der zuvor identifizierten Anforderungen aus Voruntersuchungen im Kapitel 3.2 und Hardwareauswahl im Kapitel 3.3.1.

Differenzbild

Zur Isolation von Reflexionsinformationen wird das zuvor erprobte Prinzip der Differenzbildbildung verwendet. Zu diesem Zweck wird ein Bildverarbeitungsprozess implementiert, welcher ein Bildpaar akzeptiert und die Differenz der Helligkeitswerte eines jeden Bildpixels als Helligkeitswert des Zielbildpixels errechnet. Dafür wird die Implementation *absdiff* des Computer Vision Frameworks OpenCV verwendet.[15] Die Berechnung des Absolutwertes der Differenz stellt sicher, dass keine Clipping Fehler entstehen, sollte es bei der Subtraktion zu einem Unterlauffehler kommen.

Vignettierung

Aufgrund der Verwendung von Objektiven mit einer geringen Brennweite, tritt eine kreisförmige Abnahme der Beleuchtungsstärke zu den Rändern hin auf. Ein solcher Effekt wird als Vignettierung bezeichnet [32, vgl. S. 131].

Zur Korrektur der Vignettierung wird folgendes Modell angenommen: Für jeden Pixel B eines theoretisch unverfälschten Bildes wird dessen Helligkeitswert mit dem an selber Stelle vorherrschenden Faktor der Vignettierung V multipliziert. Daraus ergibt sich der Helligkeitswert für den Pixel B' im schlussendlich erfassten Kamerabild.

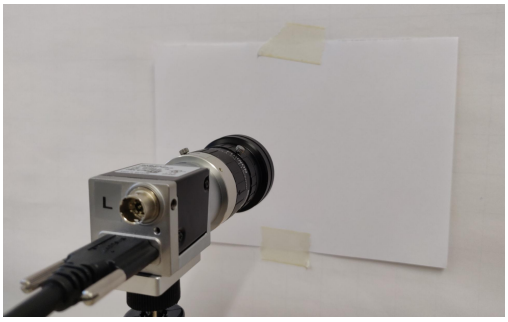
[15] Für Details zu verwendeten Implementationen siehe OpenCV Dokumentation: https://docs.opencv.org/4.7.0/ (Zugriffsdatum: 10.05.2023).

$$\lfloor B \cdot V \rfloor = B' \text{ mit } V \in \mathbb{R} \text{ wo } 0 < V \leq 1 \text{ und } B, B' \in \mathbb{N}_0 \text{ wo } 0 \leq B, B' \leq 255$$

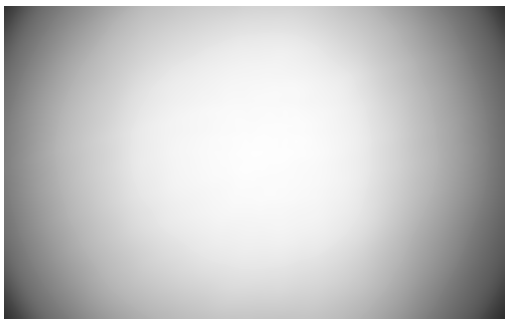
Vorausgesetzt die Faktoren der Vignettierung V sind bekannt, so kann eine Approximation der Helligkeitswerte des unverfälschten Bildes errechnet werden.

$$B \approx \frac{B'}{V}$$

Um die Vignettierung zu ermitteln, wird die Kamera vor einer gleichmäßig ausgeleuchteten ebenen weißen Oberfläche positioniert und mehrere Bilder aufgenommen. Anschließend werden diese Bilddaten gemittelt, tiefpassgefiltert und normalisiert. Durch die Mittelung mehrerer Bilder wird das Rauschen reduziert und ein realitätsnäheres Ergebnis erzielt. Die Tiefpassfilterung wird angewendet, um möglich verbleibende Details der aufgenommenen Oberfläche zu entfernen. Dadurch werden feine Strukturen oder Unregelmäßigkeiten, die nicht mit der Vignettierung zusammenhängen, geglättet und herausgefiltert. Die Normalisierung der Bilddaten erhöht die Dynamik der resultierenden Vignettierungsmaske. Dabei werden die Helligkeitswerte so angepasst, dass der volle Bereich des verfügbaren Helligkeitsspektrums ausgenutzt wird. Dieser Prozess wird für beide Kameras individuell durchgeführt.



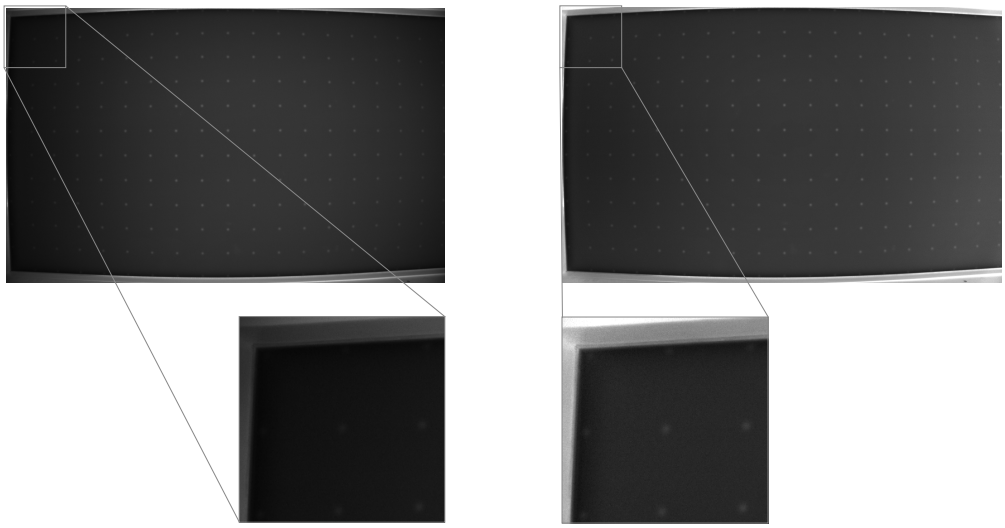
(a) Messaufbau



(b) Vignettierungsmaske Messergebnis

Abbildung 3.19: Ermittlung der Vignettierung

Um eine möglichst unverfälschte Darstellung der aufgenommenen Umwelt zu erreichen, wird die Vignettierung korrigiert. Dazu wird ein Bildverarbeitungsprozess implementiert, bei dem jeder Helligkeitswert eines Pixels des Eingabebildes durch den entsprechenden Helligkeitswert des Pixels an derselben Position der Vignettierungsmaske dividiert wird. Dafür wird die Implementation *divide* des Computer Vision Framework OpenCV verwendet. Sollte ein Vignettierungsfaktor einem Nullwert entsprechen, wird der Helligkeitswert des Zielbildpixels als voll ausgesteuert definiert.

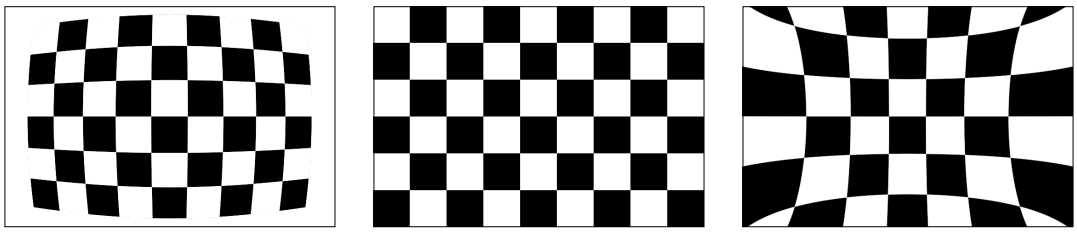


(a) mit Vignettierung (b) Vignettierung entfernt

Abbildung 3.20: Korrektur der Vignettierung

Linsenverzeichnung

Die verwendeten Objektive weisen, entsprechend deren Datenblatt, eine negative Verzeichnung auf. Diese Verzeichnung führt dazu, dass eigentlich parallele Strukturen der Realität in deren Abbildung zu den Bildrändern hin gekrümmt dargestellt werden. Die Verzeichnung nimmt dabei von der Bildmitte aus zum Rand hin zu (siehe Abbildung 3.21a). Ein solcher Effekt ist eine Folge der spezifischen Bauweise des Objektivs [32, vgl. S. 93 f., Kapitel 11.7.1].



(a) Negativ ”tonnenförmig” (b) Keine (c) Positiv ”kissenförmig”

Abbildung 3.21: Typen von Linsenverzeichnungen [1]

Da die Verzeichnung der Objektive dazu führt, dass die Objekte auf der Projektionsfläche verfälscht dargestellt werden, muss dieser Effekt korrigiert werden. Zu diesem Zweck wird die Implementation für Kamerakalibrierungen des Computer Vision Frameworks

OpenCV verwendet [16]. Dabei werden initial die intrinsischen Kameraparameter und die Verzeichnungskoeffizienten unter Verwendung der Implementation *caibrateCamera* bestimmt [1]. Diese nutzt dafür das Verfahren nach Zhang [36]. Dieser Prozess wird für beide Kameras individuell durchgeführt und die bestimmten Parameter abgespeichert.
Zur Korrektur der verzeichneten Bilddaten wird ein Bildverarbeitungsprozess implementiert, welcher ein Eingabebild akzeptiert und dessen Verzeichnung entsprechend der zuvor identifizierten Parameter entfernt. Dafür wird die Implementation *remap* des Computer Vision Framework OpenCV verwendet.

3.6.2 Stitching

Aufgrund der Wahl einer zentralisierten Topologie des Verarbeitungssystems in Kapitel 3.4, ist es erforderlich einen expliziten Bildverarbeitungsschritt durchzuführen, welcher die gefilterten Einzelbilder der verschiedenen Kameras zu einem Gesamtbild zusammenzufügt.

Herangehensweise

Die Genauigkeit der einheitlichen Positionierung der Kameras, wie sie durch die Erweiterung des Montagesystems im Kapitel 3.3.2 beschrieben wurde, reicht nicht aus, um die Bilddaten direkt und ohne zusätzliche Zwischenschritte zu kombinieren. Daher wird versucht die dafür dedizierte Implementation des Computer Vision Frameworks OpenCV zu verwenden [3]. Diese ist dazu in der Lage Panorama aus Einzelbildern zu generieren [34] und sollte daher auch das Zusammenfügen beider Teilbilder realisieren können. Der Versuch der Verwendung schlägt dabei jedoch mit der Fehlermeldung, dass nicht genug Merkmale identifiziert werden können, fehl. Dieser Umstand konnte auch nicht durch das Hinzufügen von Merkmalen mittels manueller Markierungen der Form aufgemalter Punkte auf der Projektionsfläche behoben werden. Entsprechend muss ein alternativer Ansatz verfolgt werden.
Betrachtet man das Verfahren von Brown et al. [10], nach dessen Vorbild der Prozess von OpenCV implementiert ist, werden grob betrachtet drei Schritte durchgeführt: Zuerst werden identische Merkmale in allen Teilbildern identifiziert. Anschließend werden die Teilbilder entsprechend der Merkmale zueinander passend transformiert. Schlussendlich werden die transformierten Teilbilder kombiniert und an den Übergängen miteinander verblendet. Nach diesem Vorbild wird ein vereinfachtes Verfahren realisiert.

Bild- Welt- Koordinaten

Anstatt Merkmale auf Basis der Bilddaten zu identifizieren und diese miteinander in Verbindung zu bringen, wird stattdessen ein auf der realen Welt basierender Standard eingeführt. Dafür wird die Projektionsflächenabdeckung mit menschenmöglicher Präzision in einheitlich regelmäßigen Abständen mit Punkten beklebt (siehe Abbildung 3.22). Die Projektionsflächenabdeckung lässt sich, entsprechend ihren Maßen, immer gleich auf der Projektionsfläche platzieren. Die auf der Projektionsflächenabdeckung aufgeklebten Punkte ermöglichen es nun, die Reflexionen dieser Punkte im Kamerabild als Bildkoordinaten mit ihrer Position auf der Projektionsfläche als Weltkoordinaten in Beziehung zu setzen.

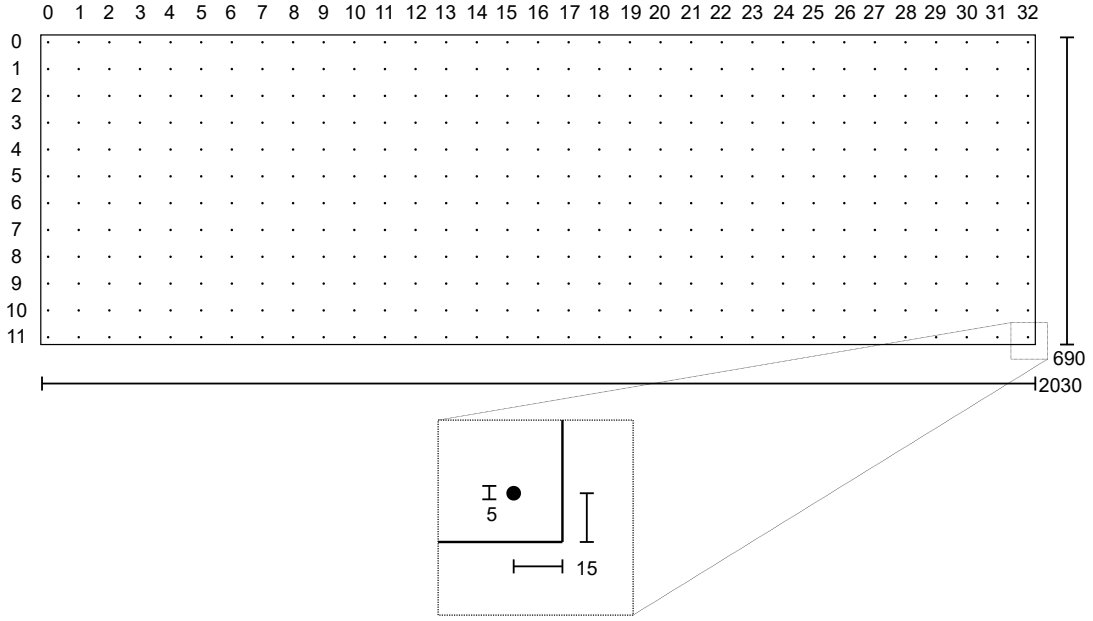


Abbildung 3.22: Projektionsflächenabdeckung mit Punktmuster (schematisch)

Dabei korrespondieren die Eckpunkte $(0, 0)$, $(32, 0)$, $(0, 11)$ und $(32, 11)$ der Projektionsflächenabdeckung mit den Eckpunkten der Projektionsfläche selbst. Durch diese Beziehung können Pixelwerte Aussagen über die Positionierung von Objekten auf der Projektionsfläche im Maßstab von Millimetern liefern. Dieses Verfahren gilt allgemein und lässt sich zukünftig auf eine veränderte Hardwarekonfiguration übertragen.

Perspektivtransformation

Da die Beziehung zwischen den Bilddaten und deren Position auf der Projektionsfläche ermittelt wurde, kann im nächsten Schritt die verbleibende perspektivische Transformation der Bilddaten, resultierend der imperfekten Positionierung der Kamera, korrigiert werden. Dadurch wird gewährleistet, dass die Bilddaten die Projektionsfläche realitätsgetreu darstellen, als würde sie perfekt parallel abgelichtet werden. Außerdem beschränkt es die Bilddaten ausschließlich auf deren Bereich. Zu diesem Zweck werden vier Bild- Welt- Koordinatenpaare verwendet, um die entsprechende Transformationsmatrix mithilfe des Computer Vision Framework OpenCV zu errechnen. Dieser Prozess wird für beide Kameras individuell durchgeführt.
Zur Korrektur der perspektivischen Transformation wird ein Bildverarbeitungsprozess implementiert, welcher ein Eingabebild entsprechend der identifizierten Transformationsmatrix verarbeitet. Dafür wird die Implementation *warpPerspective* des Computer Vision Framework OpenCV verwendet.

Zusammenfügen von Teilbildern

Im letzten Schritt des Stitching-Prozesses werden die Teilbilder zusammengefügt, um ein nahtloses Gesamtbild zu erstellen. Dazu werden die jeweiligen Bilddaten in einen gemeinsamen Datenpuffer kopiert. Im Übergangsbereich zwischen den Teilbildern werden die Pixel übereinandergelegt und linear miteinander verblendet. Durch diese Verblendung entsteht ein sanfter Übergang zwischen den Bildern, wodurch das Gesamtbild eine harmonische und zusammenhängende Darstellung der Projektionsfläche repräsentiert.
Zu diesem Zweck wird ein Bildverarbeitungsprozess implementiert, welcher ein Bildpaar akzeptiert und die Kombination und Verblendung entsprechend der Beschreibung vornimmt. Dabei werden Implementation des Computer Vision Framework OpenCV verwendet.

Spiegelung

Damit die Bilddaten der Sicht des Nutzers entsprechen, werden diese auf der vertikalen Achse gespiegelt. Dies erleichtert die Konvertierung der Positionen von Objekten bei der folgenden Merkmalsextraktion.

Zu diesem Zweck wird ein Bildverarbeitungsprozess implementiert, welcher ein Eingabebild akzeptiert, und dieses spiegelt. Dafür wird die Implementation *flip* des Computer Vision Framework OpenCV verwendet.

3.6.3 Implementationsdetails

Im Folgenden werden einige spezifische Details, bezüglich der Implementation der Bildverarbeitungskomponente, erläutert.

Parallelisierung der Bildverarbeitung

Aus vergleichbaren Gründen der Gestaltung des Verarbeitungssystems als Pipelinearchitektur, werden auch die Kernkomponenten der Bildverarbeitung selbst als Pipelines entworfen. Dabei stellen die Bildverarbeitungsprozesse zur Filterung und Korrektur aus Kapitel 3.6.1 sowie zum Stitching aus Kapitel 3.6.2 die verschiedenen Pipelinestufen dar. Es werden alle Operationen, welche spezifisch die Bilddaten einer Kamera verarbeiten, als *CorrectionPipeline* zusammengefasst. Dies umfasst die Schritte der Differenzbildbildung, der Devignettierung, der Korrektur bezüglich Verzeichnungen und der perspektivischen Transformation. Die Bildverarbeitungsprozesse, welche sich auf das Zusammenfügen der einzelnen gefilterten Kamerabilddaten beziehen, werden als *StitchingPipeline* zusammengefasst. Dies entspricht dem Zusammenfügen der Teilbilder und dem Spiegeln der Bilddaten.
Die Gestaltung der Bildverarbeitung als Menge an Pipelines macht diese modular und erweiterbar. So können zukünftig individuelle Bildverarbeitungsprozesse verändert bzw. optimiert oder neue hinzugefügt werden. Darüber hinaus ermöglicht es die Parallelisierung der Bildverarbeitung. Zu diesem Zweck implementieren die *Correction-* und *Stitching-Pipeline* Komponenten eine Methode *enqueue* zur Beauftragung die Bildverarbeitung asynchron und reihenfolgeerhaltend durchzuführen.

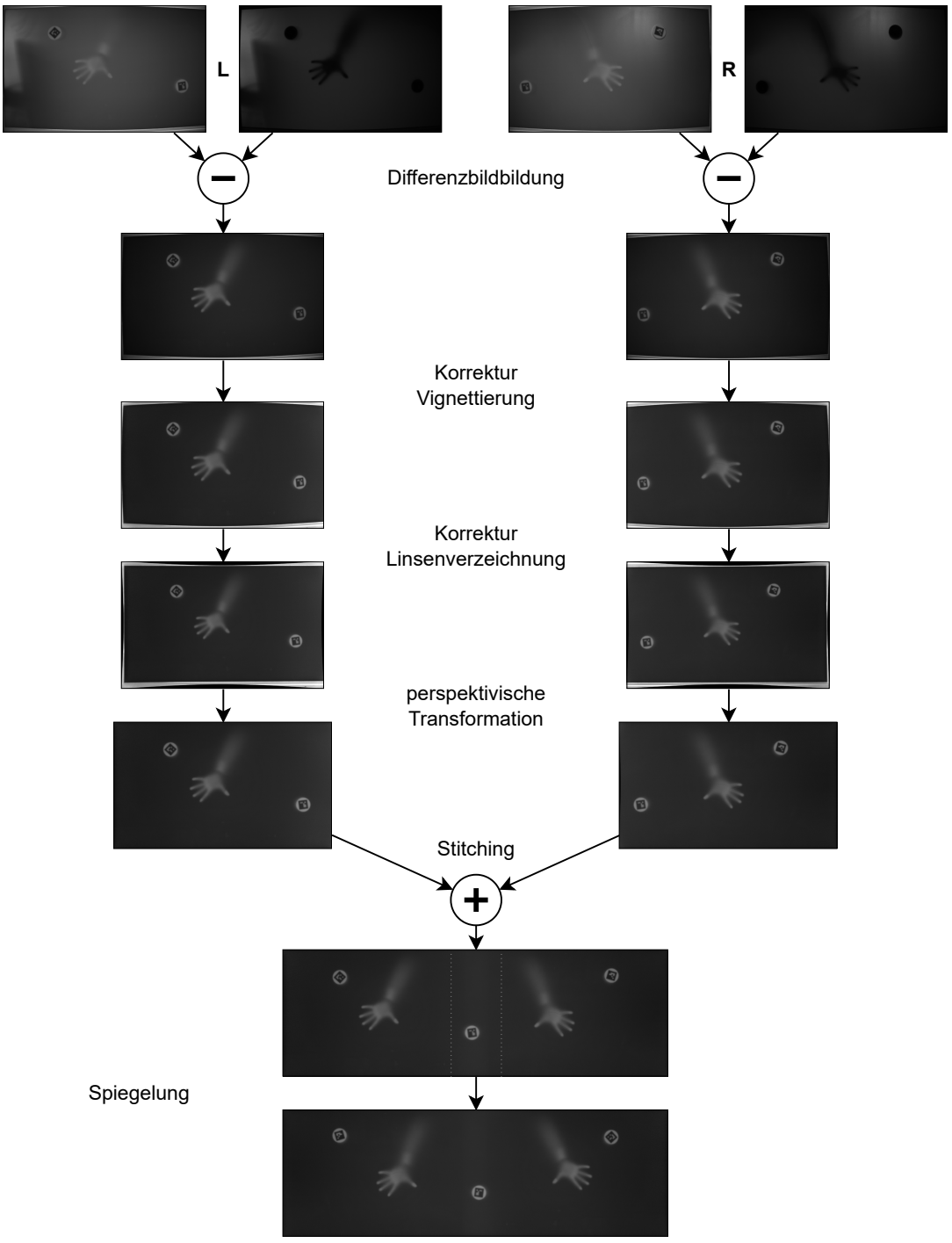


Abbildung 3.23: Bildverarbeitungspipeline

Demultiplexen von Bilddatenströmen

Aufgrund der Differenzbildbildung und der Verwendung mehrerer Kameras, müssen die Datenströme an Bilddaten erst zeitlich und anschließend räumlich demultiplext werden. Zu diesem Zweck werden sogenannte *Frame Collector* entworfen. Die Datenströme der Kameras werden von der Klasse *CorrectionPipelineFrameCollector* als Spezialisierung eines *AsyncFrameReceiver* empfangen und paarweise gesammelt. So wird aus zwei aufeinanderfolgenden Kamerabildern ein Paar aus belichteten und unbelichteten Kamerabildern gebildet, welches anschließend zu einem Differenzbild und dann weiterverarbeitet werden kann.

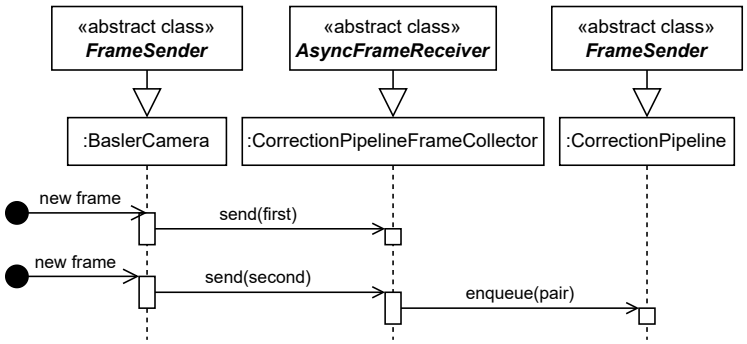


Abbildung 3.24: Zeitliches Demultiplexing für Filterung - Sequenzdiagramm

Ähnlich verhält es sich anschließend mit den vollständig gefilterten Bilddaten. Diese werden von der Klasse *StitchingPipelineFrameCollector*, ebenfalls eine Spezialisierung eines *AsyncFrameReceiver*, empfangen. Zur Differenzierung der zugehörigen Position der Bilddaten werden diesen bei der Filterung entsprechende Metadaten hinzugefügt. Es werden Paare an links- und rechtszugehörigen Bilddaten gebildet und anschließend zu einem Gesamtbild verarbeitet (siehe Abbildung 3.25).

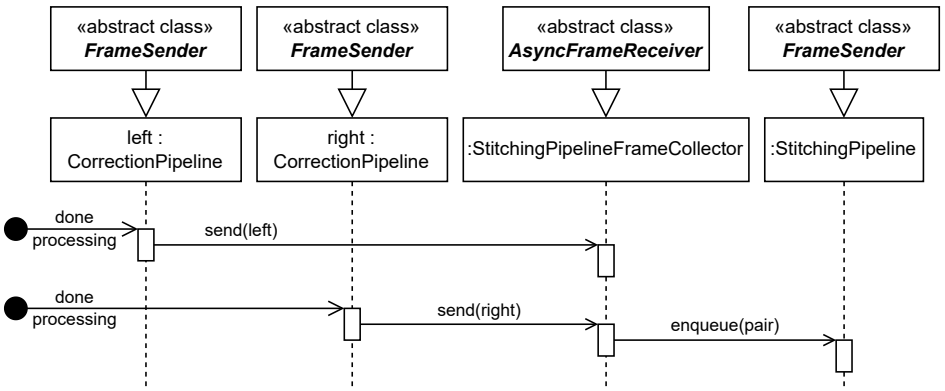


Abbildung 3.25: Räumliches Demultiplexing für Stitching - Sequenzdiagramm

Fehlerhandhabung

Falls bei der Erfassung der Kamerabilddaten oder einem Bildverarbeitungsschritt ein Fehler auftritt, wird der entsprechende *Frame* mit einer Metainformation markiert, die auf einen allgemeinen Fehler hinweist. Danach wird der *Frame* gemäß des regulären Ablaufs weitergeleitet. Dies ist wichtig, um die Reihenfolge beizubehalten. Wenn ein nachfolgendes Element einen als fehlerhaft markierten *Frame* erhält, wird dieser ebenfalls ohne weitere Bearbeitung an das nächste Element weitergegeben.
Sollte mindestens ein *Frame* des Paares zum Stitching oder das Ergebnis des Stitching selbst fehlerhaft sein, werden diese Informationen verworfen und die Bilddaten gelöscht. Dadurch wird sichergestellt, dass nur fehlerfreie und zusammengehörige Bilddaten an die nächste Hauptkomponente weitergegeben werden.

3.7 Merkmalsextraktion

Die Merkmalsextraktion entspricht der dritten und damit letzten Hauptkomponente des Verarbeitungssystems. Sie ist gemäß dem Konzept für die Analyse der Bilddaten auf spezifische Merkmale bzw. Muster verantwortlich, um konkrete Interaktionen zu erkennen.

3.7.1 TUIO

Bevor eine ausführliche Ausdifferenzierung der Merkmalsextraktion vorgenommen werden kann, muss die bisher lediglich vage definierte Schnittstelle, welche des Konzeptes entsprechend repräsentative Interaktionsdaten bereitstellt, konkretisiert werden.
Das Tangible User Interface Objects (TUIO) Protokoll, ist eine maßgeschneiderte Lösung für die Kommunikation im Bereich der Interactive Tabletop [23]. Es basiert auf der Open Sound Control (OSC) Spezifikation und ermöglicht die Übertragung von Interaktionen im Format von Zeigerinformationen, identifizierbarer und genereller Objekte [21].[16] Es bietet damit nicht nur eine ideale Grundlage für die Anforderung bezüglich der Erkennung von Interaktionen mittels markierter Objekte, sondern ermöglicht auch andere Interaktionsmöglichkeiten, wie beispielsweise Touch- oder Stifteingaben unter Verwendung der Zeigerinformationen sowie denkbar komplexeren Interaktionen unter Verwendung der generellen Objekte. Darüber hinaus profitiert man von einer bereits breiten Unterstützung dieser Schnittstelle in Anwendungen, die auf dem TUIO-Protokoll basieren.[17] Diese Vielfalt resultiert aus dem Umstand, dass über die letzten Jahre das TUIO-Protokoll als De-Facto-Standard im Bereich der Interactive Tabletop verwendet wurde [24]. Aus diesen Gründen ist die Verwendung des TUIO-Protokolls, als konkrete Wahl einer Schnittstelle für repräsentative Interaktionsdaten, eine überzeugende Wahl für das System.

Im Rahmen dieser Arbeit wird das TUIO-Protokoll in der Version 1.1 verwendet. Dies begründet sich darin, dass Softwarebibliotheken für die verwendete Programmiersprache C++ nur in dieser spezifischen Version zur Verfügung stehen.

3.7.2 Interaktionen identifizierende Module

Module zur Identifikation von Interaktionen werden durch die Schnittstelle *IFeature-Module* beschrieben. Die Implementationen dieser Schnittstelle, welche konkrete Analysen realisieren, können bei der Klasse *TabletopFeatureExtractor* registriert werden. Diese Klasse empfängt die Bilddaten der Bildverarbeitungskomponente und delegiert diese an alle registrierten Module. Dabei wird für jedes empfangendes *Frame* Objekt, welches auf Interaktionen analysiert werden soll, ein *TuioFrameBuffer* erzeugt. Dieser wird dazu verwendet, die Operationen des Hinzufügens, Aktualisierens oder Entfernens von TUIO

[16] Für Details siehe TUIO 1.1 Spezifikation: https://www.tuio.org/?specification (Zugriffsdatum: 11.05.2023).
[17] Für eine vollständige Liste siehe: https://www.tuio.org/?software (Zugriffsdatum: 11.05.2023).

Entitäten, über die parallele Ausführung der Analysen individueller Module, zu synchronisieren. Die Parallelisierung ist dabei auf die Anwendung der Menge verschiedener Module auf jeweils ein *Frame* Objekt zur Zeit beschränkt. Dies begründet sich darin, dass die Bilddaten die Entwicklung der realen Umgebung repräsentieren und daher zeitlich voneinander abhängen. Entsprechend kann ein folgender *Frame* erst analysiert werden, wenn die Analysen eines vorherigen *Frame* abgeschlossen sind. Ist letzteres der Fall, werden die im *TuioFrameBuffer* gespeicherten Operationen bezüglich der TUIO Entitäten mittels des Adapters *TuioBridge* vom *TuioServer* über das Netzwerk angeboten.
Die Abstraktion der Module mittels einer Schnittstelle ermöglicht dabei die zukünftige Erweiterung von Interaktionserkennungen und die Parallelisierung ermöglicht eine potenziell beschleunigte Ausführungszeit.

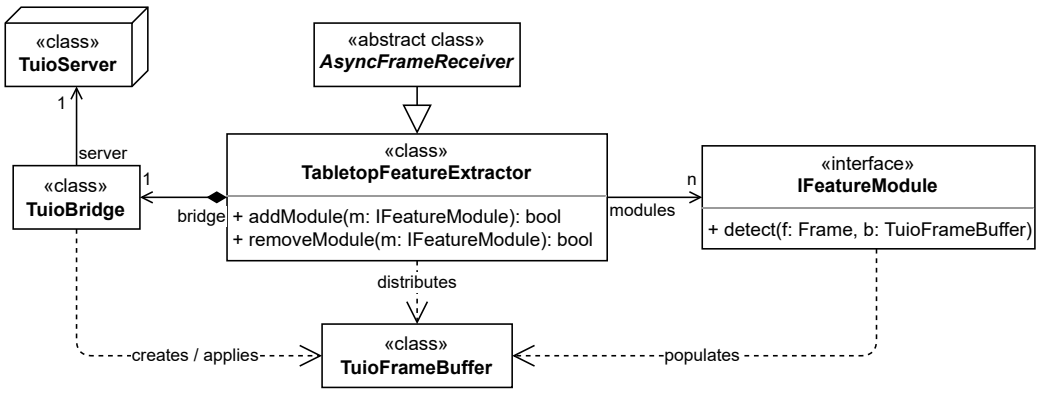


Abbildung 3.26: Feature Extraction - Klassendiagramm

3.7.3 ArUco Marker

Folgend kann konkretisiert werden, wie die zu erkennenden markierten Objekte gestaltet werden und wie deren Identifikation realisiert wird.

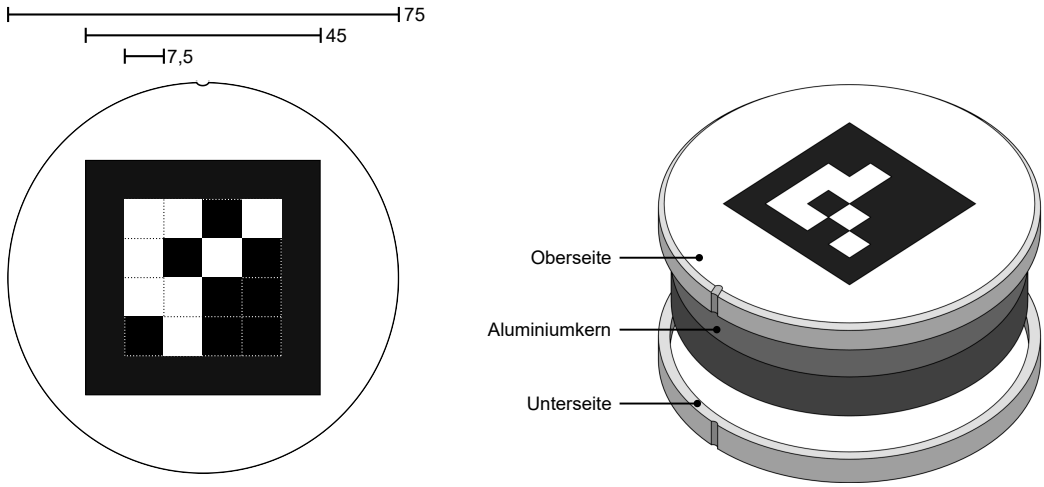
Gestaltung

Bei der Gestaltung der markierten Objekte wird zwischen den Objekten selbst und der sich darauf befindlichen Markierung unterschieden.

Die markierten Objekte werden als flache Zylinder mit einem Durchmesser von 75 mm und einer Höhe von 15 mm entworfen. Damit entsprechen sie der Anforderungen die Maße eines Eishockeypucks von 76,2 mm im Durchmesser und 25,4 mm in der Höhe nicht

zu überschreiten [30, vgl. S. 25, Regel 13.1]. Sie verfügen über eine Kerbe an der oberen Kante, die zur besseren Verdeutlichung derer aktuellen Orientierung dient. Um eine angenehmere Haptik zu gewährleisten, wird ein Aluminiumkern zur Gewichtung verwendet und die obere Kante abgerundet.

Als konkrete Ausprägung der Markierung werden ArUco Marker verwendet [14]. Dies sind quadratische visuelle Marker, die durch die Kombination von schwarzen und weißen ebenfalls quadratischen Feldern eine ID kodieren und somit zur Differenzierung verschiedener Objekte verwendet werden können. Die Wahl dieser konkreten Markierung begründet sich in der Verfügbarkeit einer Implementation zur Erkennung dieser im bereits vielseitig verwendeten Computer Vision Framework OpenCV [2] und ihrer Kompatibilität bezüglich zuvor identifizierter Anforderungen seitens der Einflüsse der Projektionsfläche im Kapitel 3.2.3. Diesen entsprechend wird die Anzahl der Felder und die Skalierung des ArUco Markers auf der zu identifizierenden Unterseite so gewählt, dass der Detailgrad des Markers 7 mm nicht unterschreitet. Damit wird sichergestellt, dass die kodierten Informationen bei der Erkennung auch zuverlässig differenziert werden können. Da die Merkmalsextraktion Bilddatentechnisch aus der Sicht des Nutzers durchgeführt wird, muss der untere Marker zusätzlich vertikal gespiegelt werden.



(a) Unterseite (schematisch)

(b) Aufbau

Abbildung 3.27: Gestaltung markierter Objekte (ID 0)

Erkennung

Die Erkennung markierter Objekte in Identität, Position und Rotation wird als Implementation der Schnittstelle *IFeatureModule* entworfen. Dabei wird die Identifikation der ArUco Marker mittels der dafür ausgelegten Implementation des Computer Vision Frameworks OpenCV realisiert [2]. Das Ergebnis dieses Prozesses ist eine Liste von identifizierten Markern, die jeweils aus einer ID und den zugehörigen vier Eckpunkten als Bildkoordinaten besteht. Die Eckpunkte werden dabei immer in der gleichen Reihenfolge im Uhrzeigersinn angegeben, wobei der erste Punkt die obere linke Ecke (UL) repräsentiert. Dieser Umstand ermöglicht die Berechnung der Rotation eines Markers, indem der rechtsläufige Winkel zwischen der negativen Y-Achse und dem Richtungsvektor zwischen dem unteren linken (LL) und oberen linken (UL) Punkt ermittelt wird. Die Position des Markers wird als Mittelpunkt aller vier Punkte bestimmt. Um Positionsangaben unabhängig der schlussendlich verwendeten Auflösung für Visualisierungen und der konkreten Skalierung des analysierten Bildes angeben zu können, werden alle Koordinaten auf den Bereich zwischen Null und Eins normiert.

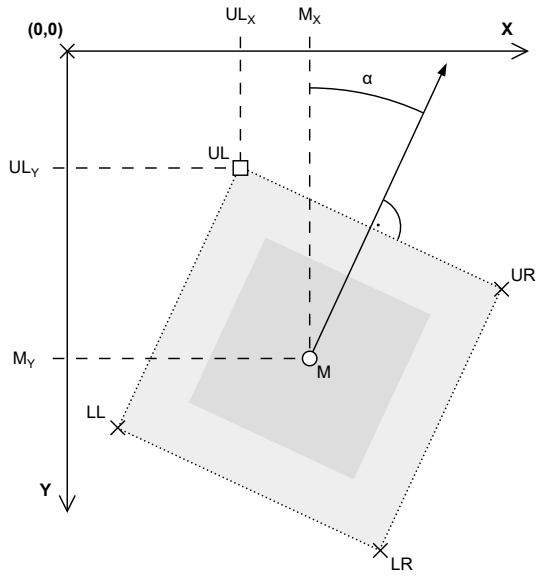


Abbildung 3.28: Identifikation von Marker Position und Rotation

Neben der Erkennung von Markern in jedem analysierten *Frame* wird auch der Zustand aller bereits erkannten markierten Objekte verwaltet. Dadurch kann festgestellt werden,

ob ein Objekt hinzugefügt, in Position oder Rotation verändert oder wieder vollständig entfernt wurde. Die entsprechenden Änderungen werden mittels des *TuioFrameBuffer* an die *TuioBridge* übermittelt.

3.7.4 Handhabung von Engpässen

Da die Analysen auf Interaktionen zeitlich voneinander abhängig sind, muss deren vollständige Ausführung abgewartet werden. Dies kann dazu führen, dass der Empfang neuer zu analysierender Bilddaten schneller geschieht als die Geschwindigkeit der Analyse voranschreitet. Entsprechend wird ein Mechanismus implementiert, welcher bei zunehmender Größe des Puffers dazu führt, dass nur noch jeder n-te *Frame* berücksichtigt wird. Dabei hängt die Zahl n von der Größe des Puffers ab. So wird beispielsweise bei einer Puffergröße von Fünf nur noch jeder zweite und bei einer Puffergröße von Zehn nur noch jeder dritte *Frame* von den Modulen analysiert. Mittels dieser Methode lassen sich Vorgehen der realen Welt weiterhin, jedoch mit reduzierter zeitlicher Auflösung, erfassen und gleichzeitig die Latenz geringhalten. Sollte dieser Mechanismus dauerhaft aktiv sein, ist dies ein Indiz, dass ein Modul einen inakzeptabel hohen Berechnungsaufwand mit korrespondierender langer Dauer aufweist.

3.8 Visualisierungen

Als finaler Schritt der Konstruktion ist es erforderlich das Projektorsystem zu kalibrieren, um dem Nutzer ein kohärentes Gesamtbild darstellen zu können.

Aufgrund der physischen Limitationen des Tresen-Unterbaus ist eine perfekte Positionierung der Projektoren, trotz des entsprechenden Montagesystems, nicht realisierbar. Dies führt zu einer typischen verzerrten Darstellung des projizierten Bildes auf der Projektionsfläche, auch bekannt als Keystone-Verzerrung. Sie entsteht aufgrund der nicht parallelen Ausrichtung der Bildebene der Projektoren zur Projektionsfläche [32, vgl. S. 502 f., Kapitel 58.4]. Dies resultiert in sich überschneidenden trapezförmigen Projektionen, welche sich zusätzlich über den Rand der Projektionsfläche selbst hinaus erstrecken und daher von der Tresenoberfläche verdeckt werden (siehe Abbildung 3.29).

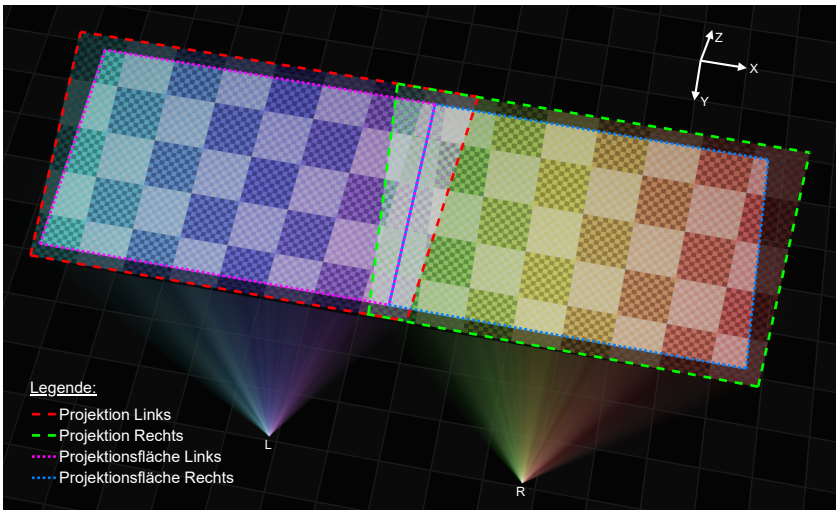


Abbildung 3.29: Projektionen in Relation zur Ebene der Projektionsfläche

Um die für den Nutzer unvollständige und verzerrte Darstellung zu korrigieren, wird die im Kapitel 3.3.1 für diesen Zweck ausgewählte Grafikkarte verwendet, welche die Funktion *Warp and Blend* unterstützt. Diese speziell für solche Fälle entwickelte Technologie ermöglicht eine Transformation des dargestellten Anzeigeinhalts auf Treiberebene. Dabei werden die Anzeigeinformationen gemäß einer definierten geometrischen Struktur angepasst [5]. Dafür werden zuerst jene Pixelkoordinaten identifiziert, die sich in den Ecken und horizontal mittig an den Rändern der Projektionsfläche befinden. Die daraus resultierenden Polygone sichtbarer Bereiche (siehe Abbildung 3.30) werden zur Gestaltung der geometrischen Struktur trianguliert. Jedes Polygon wird dabei entlang der vertikalen Achse, entsprechend der Auflösung der Projektion, in eine gleiche Anzahl an Dreiecken aufgeteilt. Durch dieses Vorgehen wird die Transformation jedes Pixels definiert. Der Ansatz die Polygone in mehr als zwei Dreiecke zu unterteilen resultiert aus dem Unvermögen stattdessen perspektivische Korrekturkoordinaten bestimmen zu können.

(a) Links (b) Rechts

Abbildung 3.30: Polygone der sichtbaren Bereiche je Projektion

Durch die Implementation der Korrektur auf Treiberebene, entfällt die Notwendigkeit einer separaten Kompatibilitätsschicht, welche nötige Transformationen darzustellender Inhalte als Softwareprozess durchführen würde. Entsprechend kann das System wie ein herkömmlicher PC verwendet werden und ist nicht auf die begrenzten Anzeigemöglichkeiten beschränkt, die eine solche Kompatibilitätsschicht bereitstellen würde.

3.9 Zusammenfassung

Das Konstruktionskapitel umfasste eine detaillierte Beschreibung der Realisierung des Systems gemäß den gestellten Anforderungen. Dabei wurde damit begonnen ein Konzept aufzustellen, welches grundlegende Hardwarebausteine definiert und die Software zum Zwecke der Modularität und Erweiterbarkeit beliebiger interaktiver Anwendungen auf zwei Ebenen aufteilt. Im Anschluss wurde dieses Konzept durch nähere Betrachtungen dessen einzelner Aspekte immer weiter konkretisiert. Um Randbedingungen bezüglich der Umgebung zu identifizieren, wurden Experimente durchgeführt und dadurch weitere Anforderungen bestimmt. Auf Basis dieser wurde anschließend eine konkrete Wahl an Hardware getroffen, die folgend in den Tresen des Living Place Labors installiert wurde. Für die Software-Implementation der ersten Ebene, dem Verarbeitungssystem, wurde eine zentralisierte Topologie gewählt, um die Weiterentwicklung so einfach wie möglich zu gestalten. Das Verarbeitungssystem besteht dabei aus den drei Hauptkomponenten zur Akquisition von Bilddaten, derer Verarbeitung und anschließender Analyse. Die Architektur wurde dabei mit passenden Abstraktionen und Schnittstellen umfassend modular realisiert, dass sie zukünftig modifiziert bzw. erweitert werden kann. Ein besonderer Fokus wurde hier auf die Erweiterbarkeit von Analysen zur Erfassung von Interaktionen basierend der Bilddaten gelegt. Identifizierte Interaktionen lassen sich mittels der bereits weitreichend verwendeten TUIO-Schnittstelle an Anwendungen auf der zweiten Ebene weitergeben, welche darauf basierend interaktive Anwendungsfälle realisieren können.

Das hiermit entwickelte System bietet somit die Möglichkeit beliebige Erkennungsmethoden zur Identifikation von Interaktionen auf Basis der visuell verfügbaren Informationen zu realisieren und darauf aufbauend interaktive Anwendungen zu entwickeln. Es stellt damit eine Plattform für die Erforschung zur Identifikation von Interaktionen im Bereich der Computer Vision sowie zur Analyse von Gestaltung und Verwendung interaktiver Anwendungen im Bereich der Interactive Tabletop dar.

4 Evaluation

Mit der Konstruktion vollständig abgeschlossen, wird folgend evaluiert, ob das dabei realisierte System den gestellten Anforderungen entspricht.

4.1 Nachuntersuchung

Um die Bewertung einiger Anforderungen durchführen zu können, ist es notwendig zunächst eine Bewertungsgrundlage zu schaffen. Hierfür werden Untersuchungen durchgeführt, deren Ergebnisse für die anschließende Diskussion dienen, inwieweit den Anforderungen entsprochen wird.

4.1.1 Zeitverhalten

Um das System in Bezug auf dessen Zeitverhalten zu bewerten, ist es erforderlich die Ausführungszeiten der verschiedenen Prozesse zu ermitteln. Zu diesem Zweck werden verschiedene Messmethoden angewandt, welche folgend weiter differenziert werden.

Ermittlung von Bildübertragungszeiten Zur Identifikation der Dauer zwischen der Initiierung der Aufnahme eines Kamerabildes durch den *Frame Coordinator* und das Empfangen der entsprechenden Kameradaten vom Verarbeitungssystem, wird die Implementation des Verarbeitungssystems zu Messzwecken soweit angepasst, dass bei Empfang eines Kamerabildes dem *Frame Coordinator* der Befehl zum Stoppen der Koordination erteilt wird. So lässt sich die Zeit zwischen der Initiierung einer Aufnahme und dem Empfangen des Befehls zum Stoppen ermitteln. Anschließend wird die benötigte Zeit der alleinigen Übermittlung eines Befehls gemessen und von der zuvor ermittelten Zeit abgezogen um die Zeit zwischen Initiierung und Empfang der Daten isoliert betrachten

zu können. Es sei dabei angemerkt, dass die Belichtungszeit der Kameras auf exakt 8 ms konfiguriert ist. Sie kann als Teil der Dauer separat betrachtet werden.

Tracing von Ausführungszeiten Die Ausführungsdauer der verschiedenen Prozesse im Verarbeitungssystem wird mithilfe eines von Yan Chernikov vorgeschlagenen Konzepts zum Benchmarking von C++ Code ermittelt.[1] Dabei werden sogenannte Scope Based Timer verwendet. Dieses Konzept nutzt die Speicherverwaltung eines in C++ definierbaren Blocks, sowie die Mechanismen des Konstruktors und Destruktors aus der objektorientierten Programmierung. Durch die Deklaration eines *ScopeBasedTimer* Objekts am Anfang eines Blocks, z.B. einer Methode, wird dessen Konstruktor aufgerufen und die aktuelle Zeit, als Startzeit der Blockausführung, erfasst. Diese Instanz existiert unabhängig von der Implementation im Block selbst. Sobald die letzte Anweisung im Block ausgeführt wurde und der Block-Stack aufgeräumt wird, wird der Destruktor des initial erstellten *ScopeBasedTimer* Objekts aufgerufen, um den Endzeitpunkt der Blockausführung zu erfassen. Eine global definierte Instanz eines *Tracer* Objekts, die dem Singleton-Pattern folgt [13, vgl. S. 127 ff.], speichert anschließend die Ausführungsdauer der Instruktionen im Block als Differenz beider erfasster Zeitpunkte sowie Metainformationen wie z.B. den Methodennamen oder die ID des ausführenden Threads. Die dabei entstehenden Trace Dateien können anschließend analysiert werden.

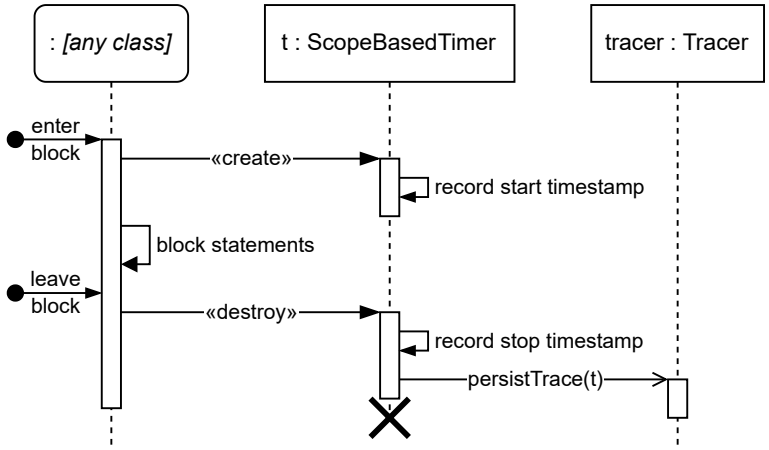


Abbildung 4.1: Tracing mit Scope Based Timer - Sequenzdiagramm

Die Definition eines *ScopeBasedTimer* Objekts zu Beginn eines Blocks, dessen Ausführungsdauer ermittelt werden soll, wird mithilfe eines Makros abstrahiert. Dies ermöglicht

[1] Siehe https://gist.github.com/TheCherno/31f135eea6ee729ab5f26a6908eb3a5e (Zugriffsdatum: 25.05.2023).

das Tracing, abhängig von Compilerflags, aus der Kompilation auszuschließen. Somit kann in einem Release Build jeglicher durch das Tracing entstehender Rechenaufwand vermieden werden.

Ermittlung von Übermittlungszeiten Zuletzt muss die Latenz, die durch den Versand von TUIO-Nachrichten zwischen den beiden Ebenen des Gesamtsystems entsteht, ermittelt werden. Da in diesem speziellen Fall beide Ebenen auf demselben Desktop-PC bereitgestellt werden, müssen die Zeiten gemessen werden, die beim Senden und Empfangen von UDP-Paketen zwischen zwei Prozessen auf demselben System auftreten. Hierfür wird eine Anwendung erstellt, die ein empfangenes UDP-Paket unverändert zurückschickt. Durch die Messung der Zeit zwischen dem Senden des Pakets und dem Empfangen dessen Echos kann die Round-Trip-Zeit (RTT) einer UDP-Verbindung auf dem localhost ermittelt werden. Die resultierende Latenz einer Übermittlung, von einer Ebene zur anderen, entspricht dann der Hälfte der RTT.

Ergebnis

Die Tracing-Daten werden unter Verwendung einer emulierten Kamera bei einer Bilderfassungsrate von 90 Bildern pro Sekunde ermittelt. Sie werden unter Verwendung des Catapult Tracing Werkzeugs ausgewertet.[2]

Tabelle 4.1: Benötigte Ausführungszeit einzelner Prozesse x in ms bei N Messungen

Prozess		**Min**	**Max**	$\overline{x}$	σ	N
Akquisition	Belichtung	8,000	8,000	8,000	0,000	-
	Übertragung	6,401	14,608	10,452	1,290	1541
Korrekturen	Differenzbild	0,672	6,721	1,459	0,693	3516
	Vignettierung	0,479	16,088	1,646	1,563	3516
	Verzeichnung	2,524	62,754	14,993	11,518	3516
Stitching	persp. Trans.	2,806	45,040	7,819	5,748	3514
	Kombination	3,048	15,713	4,586	1,982	1757
	Spiegelung	0,146	3,366	0,276	0,191	1757
Extraktion	ArUco	1,515	13,190	3,478	2,042	1757
	Übermittlung	0,119	3,500	0,520	0,141	1011
Summe		25,710	189,348	53,322	25,189	

[2] Für Details siehe Dokumentation: https://chromium.googlesource.com/catapult/+/refs/heads/main/tracing/README.md (Zugriffsdatum: 20.04.2023).

Bei der Betrachtung der Zeit zwischen dem Empfang der Kameradaten und der abgeschlossenen Übermittlung der TUIO-Nachrichten, ergibt sich die folgende Verteilung der auftretenden absoluten Latenzzeiten:

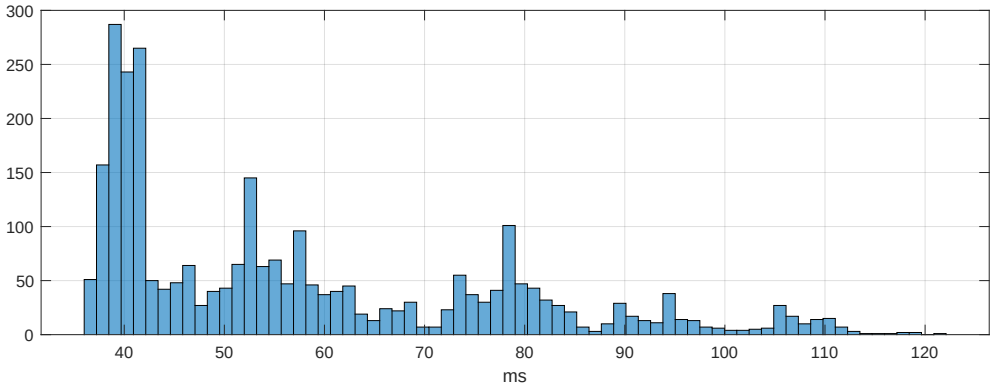


Abbildung 4.2: Häufigkeitsverteilung absoluter Latenzen - Histogramm

Tabelle 4.2: Absolute Latenzen x in ms bei N Messungen

Min	**Max**	$\overline{x}$	**Median**	σ	N
36,0	126,0	57,5	52,0	19.72	2837

4.1.2 Repräsentation der Projektionsfläche

Um das System in Bezug auf die bereitgestellten Bilddaten nach Filterung und Stitching als Repräsentation der Projektionsfläche zu bewerten, ist es erforderlich deren Realitätstreue zu ermitteln.

Vorgehen

Zu diesem Zweck wird der eingeführte Standard verwendet, um die Beziehung zwischen bekannten und gemessenen Weltpositionen zu bestimmen. Um die Position der Punkte auf der Projektionsflächenabdeckung zu identifizieren, werden die lokalen Maxima der Bilddaten im Bereich der Punkte 1-31 horizontal und 1-10 vertikal bestimmt (vgl. Abbildung 3.22). Anschließend wird die Distanz zu den erwarteten Koordinaten, sowohl absolut in X- und Y-Richtung als auch direkt, bestimmt.

Ergebnis

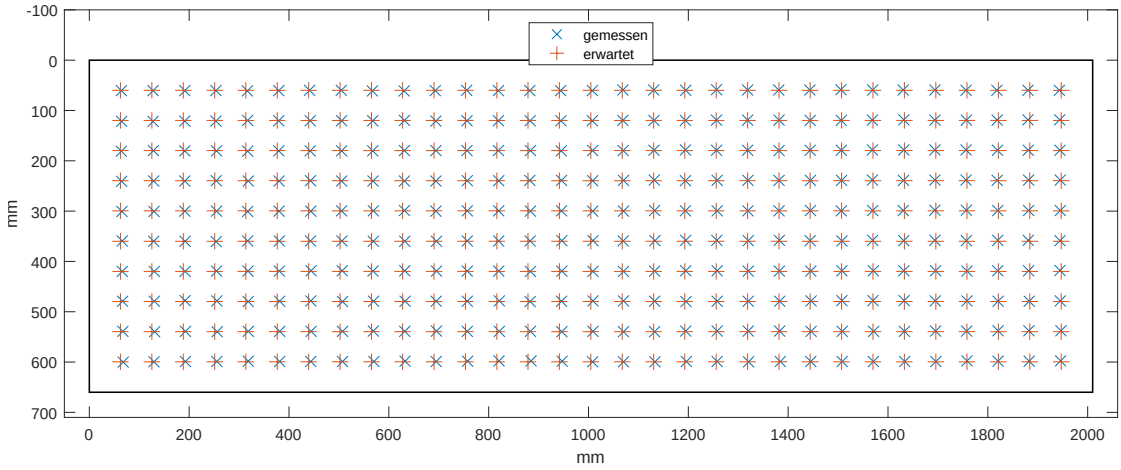
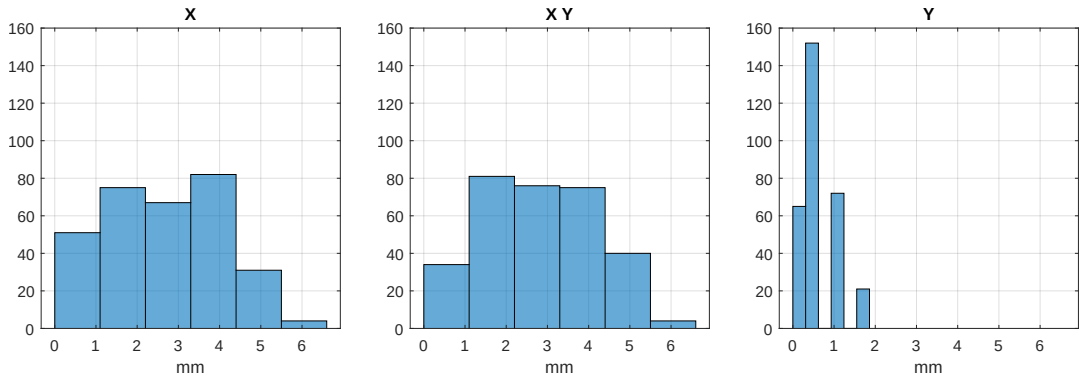


Abbildung 4.3: Gegenüberstellung erwarteter und gemessener Positionen



(a) Horizontal (b) Direkt (c) Vertikal

Abbildung 4.4: Häufigkeitsverteilung der Distanzen a - Histogramme

Tabelle 4.3: Distanzen a in mm zwischen N Punktepaaren

	Min	**Max**	$\overline{a}$	**Median**	σ	N
X	0,0	6,1	2,6	2,5	1,51	310
Y	0,0	1,8	0,7	0,6	0,51	310
XY	0,0	6,4	2,9	2,9	1,36	310

4.1.3 Identifikation markierter Objekte

Um die Fähigkeit des Systems zur Identifikation von markierten Objekten und den damit möglichen Interaktionen zu bewerten, ist es erforderlich, die Identifikation direkt am System zu erproben.

Vorgehen

Zu diesem Zweck werden zehn Exemplare der in Kapitel 3.7.3 beschriebenen markierten Objekte mit den IDs von 0-9 angefertigt und Interaktionen mit der Projektionsfläche, entsprechend der Anforderung durch das Platzieren, Verschieben, Rotieren und Entfernen, erprobt. Dieses Vorgehen wird einmal in Dunkelheit, bei eingeschalteter Innenbeleuchtung und als Extremfall bei direkter Sonneneinstrahlung durchgeführt.

Ergebnis

In allen drei Szenarien wurden alle markierten Objekte korrekt in Position und Rotation über die Projektionsfläche verteilt erkannt. Das markierte Objekt mit der ID 8 zeigt gelegentlich kurze Aussetzer, bei denen es für einen Augenblick von ca. 2-4 Bildern nicht erkannt wird.

4.1.4 Visualisierungen

Um die Darstellung visueller Informationen des Systems in Bezug auf der zu realisierenden Projektion zu bewerten, ist es erforderlich zu überprüfen, wie akkurat die getroffenen Kalibrierungen aus Kapitel 3.8 sind.

Vorgehen

Zu diesen Zweck wird eine Grafik erstellt, die bestimmte Pixelpositionen markiert und auf die Projektionsfläche projiziert. Anschließend wird die Abbildung dieser markierten Pixel auf der Projektionsfläche ermittelt, und die Abweichung von der erwarteten Position, sowohl absolut in X- und Y-Richtung als auch direkt, bestimmt.

Ergebnis

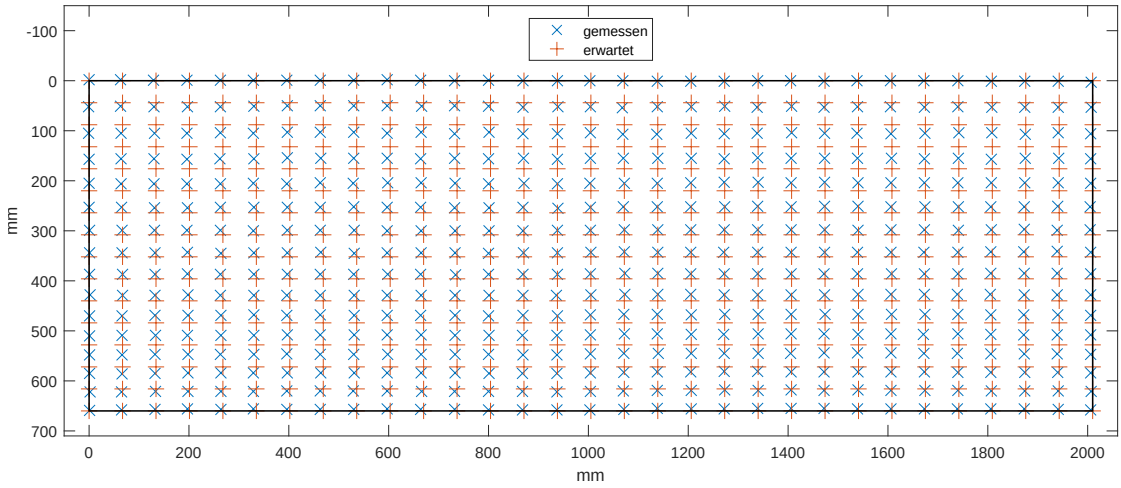
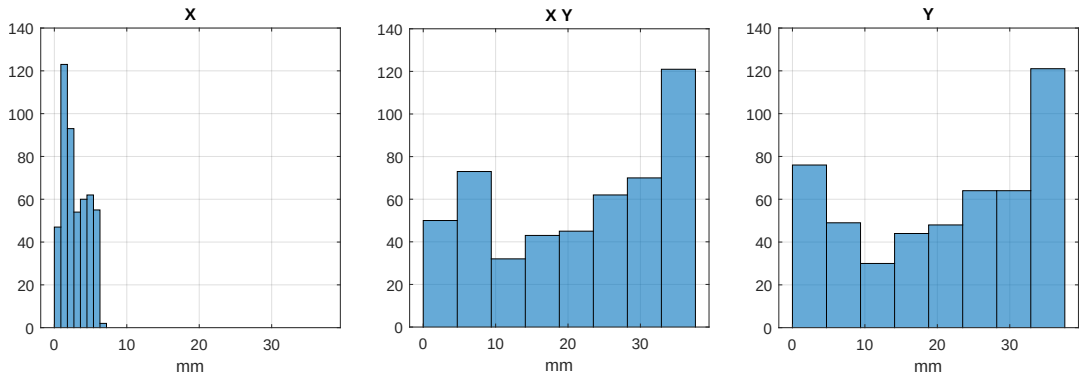


Abbildung 4.5: Gegenüberstellung erwarteter und gemessener Abbildungen



(a) Horizontal (b) Direkt (c) Vertikal

Abbildung 4.6: Häufigkeitsverteilung der Distanzen e - Histogramme

Tabelle 4.4: Distanzen e in mm bei N Punktepaaren

	Min	**Max**	$\overline{e}$	**Median**	σ	N
X	0,0	7,0	2,8	2,0	1,91	496
Y	0,0	37,0	21,0	24,0	11,92	496
XY	0,0	37,3	21,5	24,0	11,44	496

4.2 Diskussion

Unter Zuhilfenahme der Ergebnisse aus der Nachuntersuchung, kann folgend diskutiert werden, inwieweit den Anforderungen an das System entsprochen wird.

4.2.1 Funktionale Anforderungen

Umgebung

Das System wurde vollständig im Living Place Labor der HAW Hamburg entwickelt. Dabei wurden alle Designentscheidungen bezüglich der Hardware entsprechend des Tresens zwischen dem Essbereich und der Küche getroffen. Sämtliche sensorischen Komponenten und Aktoren wurden gemäß den Spezifikationen des Tresens ausgewählt und darin integriert. Um dem Nutzer eine bequeme Erreichbarkeit der Schnittstellen zum Ein- und Ausschalten des Desktop-PCs sowie zur Verbindung von Datenträgern zu ermöglichen, wurde dieser außerhalb des Tresens positioniert.

Der funktionalen Anforderung zur Umgebung des Systems ist damit entsprochen.

Funktion

Das System ermöglicht mittels einer konkret definierten Schnittstelle die Implementation einer Identifikation von Interaktionen auf Basis visueller Daten, welche mithilfe bildgestützter Verfahren erfasst werden. Die durchschnittliche Abweichung der Bilddaten, welche die reale Projektionsfläche repräsentieren, beträgt ca. 3 mm. Bei einer Auflösung von 1080 Pixeln in der Höhe und 3289 Pixeln in der Breite, gemäß der Dimensionen der Projektionsfläche von 660 mm in der Höhe und 2010 mm in der Breite, entspricht diese Abweichung etwa 2 Pixeln. Darüber hinaus hat die Untersuchung zu Einflüssen der Projektionsfläche in Kapitel 3.2.3 ergeben, dass Detailgrade kleiner als 5 mm nicht zuverlässig rekonstruiert werden können. Die Abweichung wird damit als vertretbar angesehen.

Der funktionalen Anforderung zur Funktion des Systems ist damit entsprochen.

Zu identifizierende Interaktionen

Das System ermöglicht die Erkennung der Interaktion mittels des Platzierens, Verschieben, Rotieren und Entfernen unterscheidbarer markierter Objekte. Dabei unterschreiten deren Maße die Dimensionen eines Eishockeypucks [30]. Bei einer Auswahl von zehn Objekten mit verschiedenen Markierungen, konnten 90% dieser in allen Bereichen der Projektionsfläche korrekt in Position und Rotation identifiziert werden. Das Objekt mit der ID 8, dessen Erkennung Aussetzer zeigte, hat im Vergleich zu den anderen ein besonders vollflächiges Muster als Markierung. Es wird angenommen, dass die inkonsistente Erkennung aus diesem Umstand resultiert. Mit der Verfügbarkeit von bis zu 250 verschiedenen Markierungen [2], kann alternativ eine der verbleibenden 240 Markierungen an dessen Stelle verwendet werden.

Der funktionalen Anforderung der zu identifizierenden Interaktionen des Systems ist damit entsprochen.

Visualisierungen

Das System ermöglicht die Darstellung visueller Informationen auf der Projektionsfläche unter der Verwendung von Projektionen auf dessen Unterseite. Dabei decken die Visualisierungen den gesamten Bereich der Projektionsfläche ab und sind ausschließlich darauf begrenzt. Durch die vorgenommene Kalibrierung der Projektionen im Kapitel 3.8, kann das System anzeigetechnisch wie ein herkömmlicher PC verwendet werden und Anwendungen darstellen. Aufgrund einer Verzerrung des dargestellten Bildes entlang der Y-Achse, stimmen die Positionen der Visualisierung jedoch nicht mit der Position einer Interaktion, wie zum Beispiel der Position eines markierten Objektes, überein. Die Realisierung eines Feedbacks für Interaktionen ist damit nur indirekt und verzerrt möglich.

Der Anforderung zu den Visualisierungen des Systems ist damit nicht vollständig entsprochen.

Exemplarität

Das System ermöglicht die Bereitstellung repräsentativer Interaktionsdaten entsprechend der Spezifikationen des TUIO-Protokolls. Damit bietet es die Grundlage die Anforderung

der Exemplarität, bezüglich der Gestaltung einer konkreten Anwendung auf zweiter Ebene, zu entsprechen.

Betrachtet man das folgende Kapitel 5, ist der Anforderung bezüglich der Exemplarität des Systems entsprochen.

4.2.2 Nicht-funktionale Anforderungen

Wartbarkeit

Das System verfügt über eine grundlegend modulare Architektur. Durch die Trennung des Systems in zwei Ebenen, die Wahl einer Pipelinearchitektur für die Hauptkomponenten und die Bildverarbeitung, ergänzbarer Identifikationsmodule sowie geschaffener Abstraktionen und dazugehöriger Schnittstellen, können zukünftig Änderungen bzw. Erweiterungen des Systems effizient umgesetzt werden.

Zusätzlich wurde, durch die Integration des Tracings und die Einführung eines globalen Standards für Bild-Welt-Beziehungen, die Analysierbarkeit des Systems hinsichtlich seines Zeitverhaltens und der Überprüfbarkeit erfasster und verarbeiteter Bilddaten, als realitätsnahe Repräsentation der Projektionsfläche, vereinfacht.

Der nicht-funktionalen Anforderung der Wartbarkeit wird damit mehr als entsprochen.

Zuverlässigkeit

Das System ist mittels des Verfahrens der Differenzbildbildung robust gegenüber Umgebungseinflüssen entsprechend der Fremdeinstrahlung der Innenbeleuchtung und der Sonne. Es bestehen jedoch aufgrund der Strahlungsstärke der Sonne weiterhin Limitationen, welche nur durch physische Maßnahmen außerhalb des Kontextes des Systems kompensiert werden können.

Außerdem wurden Mechanismen entwickelt und implementiert, um das System im Fehlerfall bezüglich der Erfassung und Verarbeitung von Kamerabilddaten sowie bei Engpässen der Merkmalsextraktion funktionsfähig zu halten.

Der nicht-funktionalen Anforderung der Zuverlässigkeit ist damit entsprochen.

Effizienz

Das System benötigt durchschnittlich ca. 58 ms, um eine Interaktion von der Akquisition bis zur Übermittlung zu verarbeiten. Basierend auf Untersuchungen von Kohrs et. al, nimmt der Mensch die Verzögerung einer Reaktion ab 200 ms als störend wahr [26]. Selbst bei einer Verdopplung der benötigten Zeit, verbleiben damit ca. 80 ms zur Realisierung einer Reaktion. Bei einer minimal typischen Bildwiederholungsrate von 60 Bildern in der Sekunde, bedarf eine Aktualisierung ca. 17 ms. Entsprechend wird die Verarbeitung als ausreichend schnell bewertet. Darüber hinaus fällt jedoch zusätzlich auf, dass die entstehenden Latenzen durch die Verarbeitung des Systems weit gestreut sind. Es kommt somit zu keiner kontinuierlich gleichmäßigen Verzögerung. Es wird angenommen, dass dieses Verhalten auf das Scheduling des Prozessors zurückzuführen ist.

Das System weist, bei einer Bilderfassungsrate von 90 Bildern in der Sekunde, einen Ressourcenverbrauch von ca. 100 MB Arbeitsspeicher und 44% der CPU Leistung auf. Die Mechanismen der automatischen Speicherverwaltung durch die Abstraktion der Bilddaten und die Parallelisierung der Bildverarbeitungspipeline stellen dabei einen effizienten Ressourcenverbrauch sicher. Gemessen der weiterhin verfügbaren Ressourcen des Desktop-PC ist die zeitgleiche Ausführung einer Anwendung auf zweiter Ebene realistisch realisierbar.

Es bleibt, unabhängig der Anforderung, zu beurteilen, ob die Unregelmäßigkeit der Latenzzeiten einen negativen Einfluss auf die Nutzererfahrung hat. Durch die Verwendung einer GPU, welche für Bildverarbeitungsprozesse ausgelegt ist, könnte der CPU-Verbrauch erheblich reduziert werden. Dies könnte möglicherweise auch in einer gleichmäßigen Verzögerung resultieren.

Der nicht-funktionalen Anforderung der Effizienz ist gemessen der zu beachtenden Kriterien entsprochen. Es ergeben sich jedoch interessante Aspekte, welche weiter untersucht werden können.

4.3 Fazit

Das konstruierte System erfüllt die Anforderungen weitestgehend mit nur wenigen Ausnahmen. Es wurde ein modular gestaltetes System geschaffen, das robust und fehlertolerant ist, eine zuverlässige Erfassung der Umgebung ermöglicht und gleichzeitig schnell

arbeitet und ressourcenschonend ist. Das System nutzt bildgestützte Verfahren, um Interaktionen mit der Projektionsfläche zu erfassen und spezifische Interaktionen wie hier konkret das Platzieren, Verschieben und Entfernen von unterscheidbaren markierten Objekten zu identifizieren. Es sind jedoch auch Ausnahmen identifiziert worden. So bedarf es eines alternativen Ansatzes zur Kalibrierung der Projektionen, um aktuell auftretende Verzerrungen entlang der Y-Achse zu vermeiden. Zusätzlich ist eine Untersuchung zum Ursprung weit gestreuter Latenzzeiten und einer Ergründung der unzuverlässigen Erkennung des markierten Objektes mit der ID 8 zu empfehlen.

5 Exemplarisches Anwendungsbeispiel

In den vorausgegangenen Kapiteln wurde beschrieben, wie das Gesamtsystem auf erster Ebene Interaktionen erfasst, identifiziert und übermittelt. Folgend wird exemplarisch eine Realisierung der zweiten Ebene, ein konkretes Anwendungsbeispiel, beschrieben.

5.1 Konzept

Das Anwendungsbeispiel soll, gemäß den Anforderungen zur Exemplarität, als Schnittstelle für die Kontrolle einer ausgewählten Menge von Smart Home Funktionalitäten des Living Place Labors dienen. In diesem Beispiel wird dabei die Veränderung der Beleuchtungssituation des Living Place Labors realisiert. So soll sowohl die Innenbeleuchtung ein- wie ausgeschaltet werden können, als auch die RGB-Leuchtmittel der verschiedenen Bereiche ansteuerbar sein. Da zur Eingabe des Nutzers die markierten Objekte verwendet werden sollen, soll die Platzierung eines Objektes an einer spezifischen Position das zu steuernde Element auswählen und dessen Rotation eine Zustandsänderung des zu steuernden Elements bewirken. So sind alle Interaktionsmöglichkeiten mittels markierter Objekte, für die Realisierung einer exemplarischen Anwendung, abgedeckt.

5.2 Umsetzung

Folgend wird auf die Umsetzung des Anwendungsbeispiels eingegangen. Aufgrund deren exemplarischen Charakters, werden implementationsspezifische Details in dieser Ausarbeitung lediglich oberflächlich dargestellt. Der Fokus liegt dabei auf dem Architekturansatz, eine klassische grafische Anwendung, um die Möglichkeit der Interaktion mittels der markierten Objekte zu erweitern.

5.2.1 TUIO kompatible Anwendungsfenster

Um eine Anwendung visuell auf der Anzeige darzustellen, ist ein Fenster erforderlich. Dieses Fenster soll nicht nur die übliche Interaktion mittels Maus bzw. Tastatur ermöglichen, sondern auch Reaktionen auf Interaktionen mittels der markierten Objekte realisieren können. Zu diesem Zweck wird ein Element entworfen, welches als Adapter zwischen einem *TuioWindow* und einem *TuioClient* vermittelt. Dieses *TuioWindowAdapter* Element hat die Aufgabe, die vom *TuioClient* empfangenen Tuio-Nachrichten von relativen Bildschirmkoordinaten in absolute Pixelkoordinaten innerhalb des Anwendungsfensters zu übersetzen. Damit Komponenten auf Basis der Veränderung von markierten Objekten reagieren zu vermögen, können entsprechend des Observer-Pattern [13, vgl. S. 293 ff.] Implementationen der Schnittstelle *TuioObjectListener* beim Adapter registriert werden.

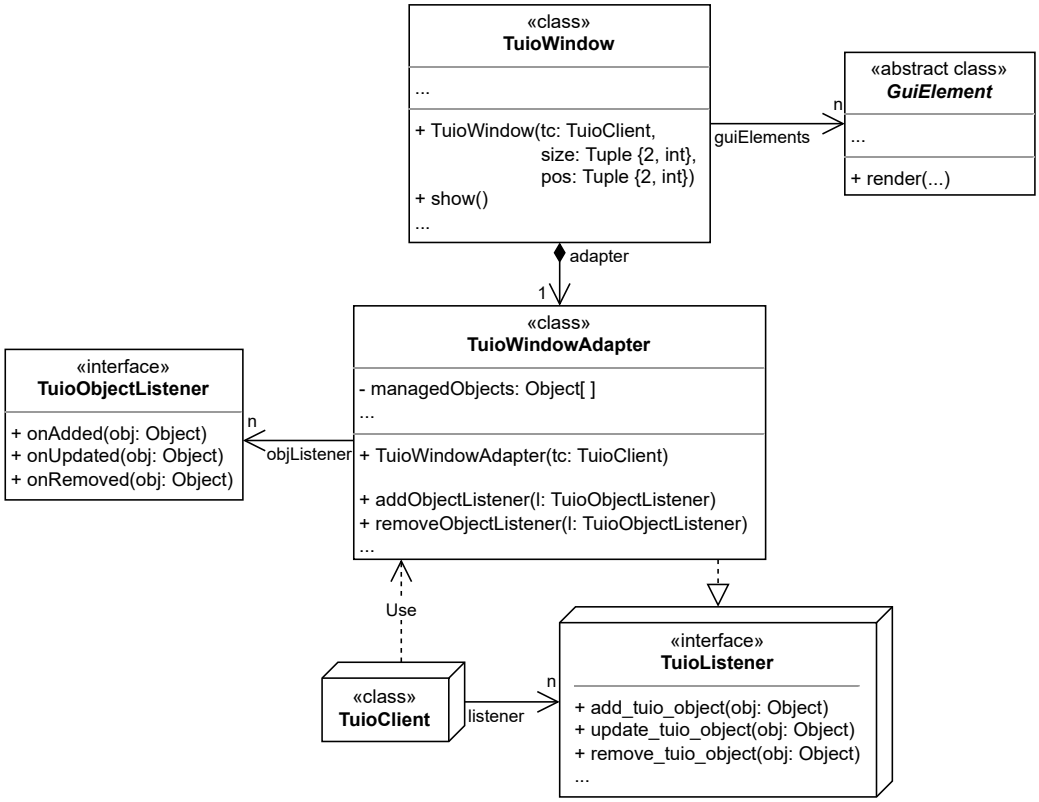


Abbildung 5.1: TUIO kompatible Anwendungsfenster - Klassendiagramm

Um die Erstellung und Darstellung spezifischer Anzeigeelemente zu ermöglichen, können Spezialisierungen der abstrakten Klasse *GuiElement* dem Anwendungsfenster *TuioWindow* hinzugefügt werden.

5.2.2 Interaktionsfähige Anzeigeelemente

Folgend bedarf es spezieller Anzeigeelemente, mit welchem mittels der markierten Objekte interagiert werden kann. Gemäß dem Konzept sollen Reaktionen durch das Platzieren markierter Objekte an spezifischen Positionen ermöglicht werden. Zu diesem Zweck wird ein *Hub* Anzeigeelement, als Spezialisierung einer spezifischen *TuioObjectListener* Implementation, realisiert, welches nur auf markierte Objekte reagiert, die sich auf bzw. im Inneren des definierten Bereichs des *Hub* befinden.
Da entsprechend des Konzeptes ebenfalls eine Reaktion durch die Rotation eines markierten Objektes ermöglicht werden soll, wird das *Hub* Anzeigeelement weiter als *SectionHub* spezialisiert. Dieses Anzeigeelement unterteilt seinen Umfang in $n + 1$ Bereiche. Dabei entspricht der erste Bereich einer Neutralposition und ist immer entlang der initialen Orientierung des markierten Objekts ausgerichtet, wenn es den *SectionHub* erreicht. Durch das Drehen des markierten Objekts in der Realität, lassen sich die n anderen Bereiche auswählen. Der Wechsel eines Bereiches stellt eine Nutzerinteraktion dar, welche mittels einer konkreten Implementation des *SectionChangedListener* eine Reaktion zugewiesen werden kann.

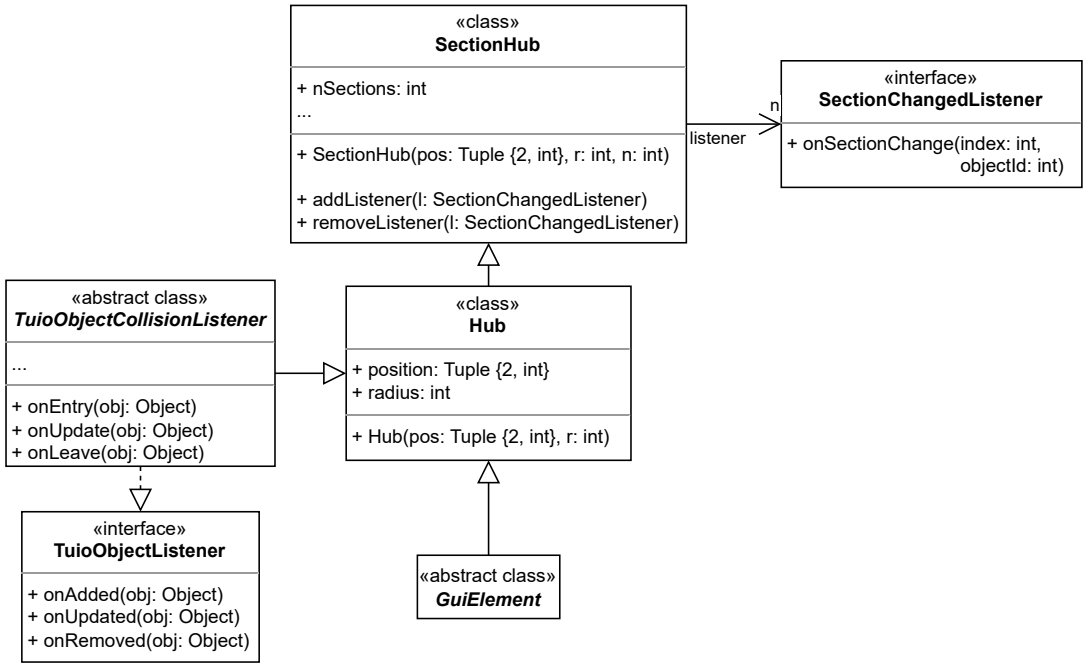


Abbildung 5.2: Interaktionsfähige Anzeigeelemente - Klassendiagramm

5.2.3 Anwendungsimplementation

Schlussendlich kann auf der Basis der TUIO kompatiblen Anzeigefenster und der interaktionsfähigen Anzeigeelemente eine Anwendung gemäß dem Konzept realisiert werden. Zu diesem Zweck wird eine Repräsentation des Living Place Labors als *LivingPlace* Element geschafften, welches die Smart Home Infrastruktur bezüglich der Beleuchtung abstrahiert und Steuerbefehle mittels des MQTT Protokolls versendet. Das Anwendungsbeispiel *DemoApplication* wird als Spezialisierung eines *TuioWindow* realisiert. Ihr werden für jeden Abschnitt und für die Innenbeleuchtung *SectionHub* Anzeigeelemente hinzugefügt. Mittels der Implementationen *LightController* und *RGBController* der Schnittstelle *SectionChangedListener* lassen sich, als Reaktion auf Rotation eines markierten Objektes, das Ein- bzw. Abschalten der Innenbeleuchtung sowie das Ändern der Farbe oder Abschalten der RGB-Leuchtmittel realisieren.

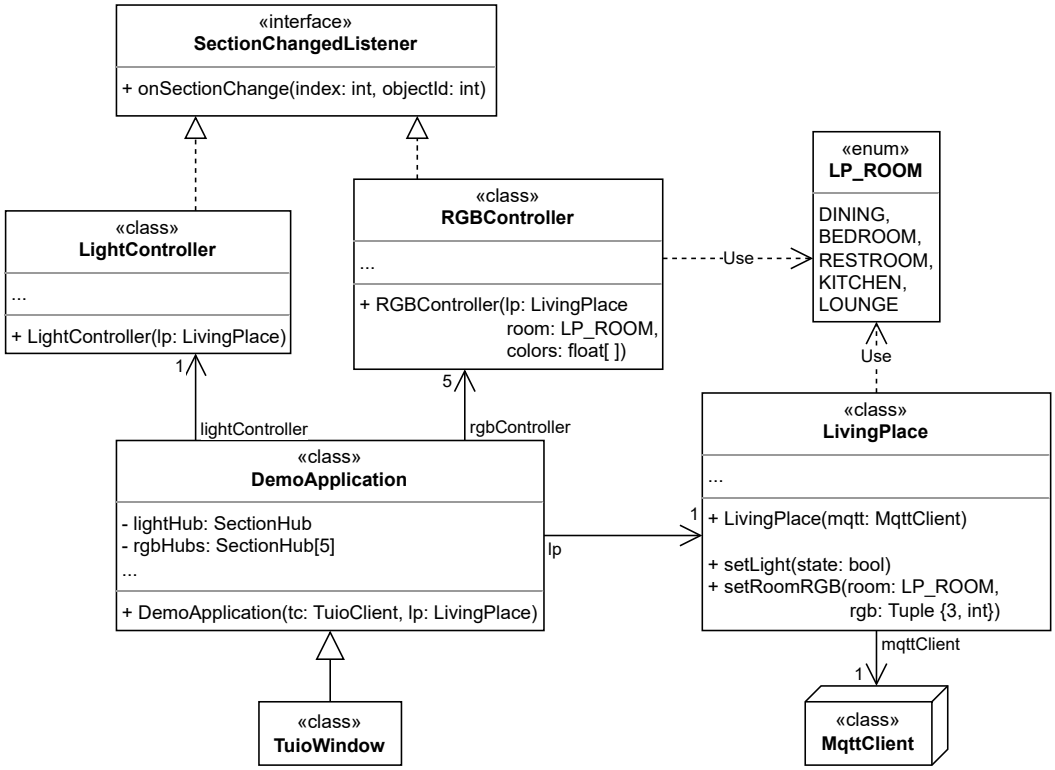


Abbildung 5.3: Anwendungsimplementation - Klassendiagramm

5.3 UI Design

Die Benutzeroberfläche des Anwendungsbeispiels wird als Grundriss der Livin Place Labor Wohnfläche gestaltet. Für jeden Abschnitt wird ein *SectionHub* Anzeigeelement zum Ändern der Lichtstimmung in die Mitte des jeweiligen Bereiches platziert. Für die Ansteuerung der Innenbeleuchtung wird ein zusätzliches *SectionHub* Anzeigeelement in die Mitte des Wohnungsgrundrisses platziert. Der Nutzer kann damit markierte Objekte auf das jeweilige Anzeigeelement legen oder ziehen und durch dessen anschließender Drehung eine Änderung der Beleuchtungssituation des Living Place Labors hervorrufen. Dabei wird die aktuelle Position eines markierten Objektes durch eine virtuelle Repräsentation dargestellt.

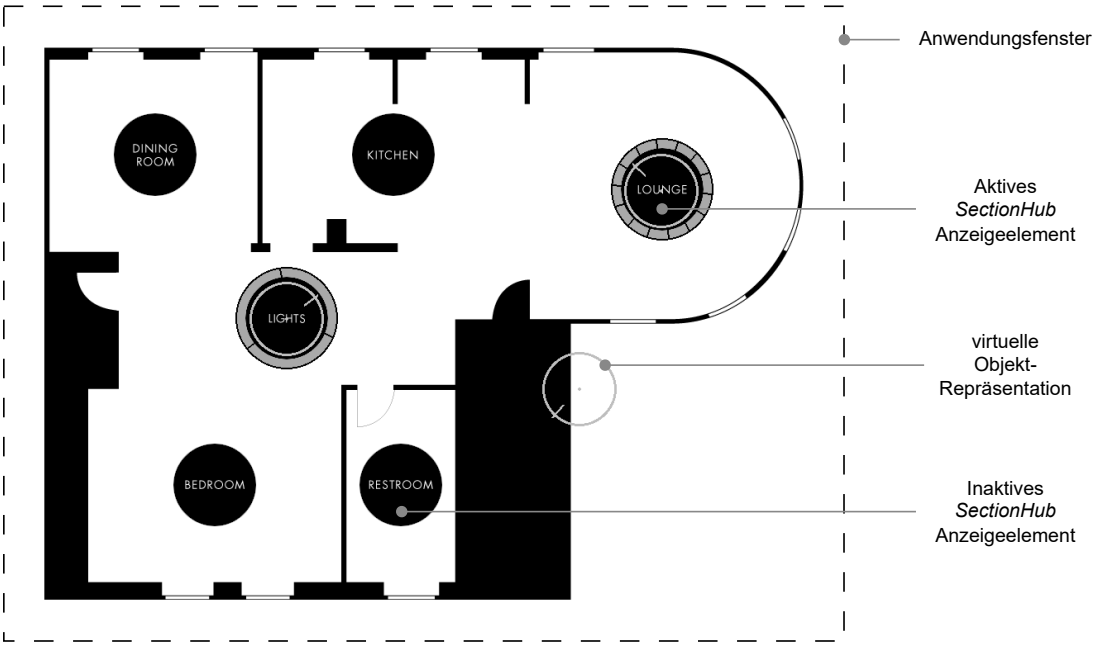


Abbildung 5.4: Benutzeroberfläche unter Verwendung drei markierter Objekte

6 Schluss

Zum Schluss dieser Ausarbeitung werden die vorherigen Kapitel abschließend zusammengefasst und ein Ausblick auf Aspekte gegeben, welche zukünftig weiter betrachtet werden können.

6.1 Zusammenfassung

In dieser Ausarbeitung wurde die Realisierung eines Interaktive Tabletop vom Konzept bis zur vollständigen Hardware- wie Software- Implementation beschrieben. Die Zielsetzung umfasste dabei eine vollständige Realisierung auf Basis des vorhandenen Mobiliars im Living Place Labor der HAW Hamburg. Das zu entwickelnde System solle es erlauben, mithilfe bildgestützter Verfahren Interaktionen mit der Projektionsfläche eines Tresens zuverlässig und ohne große Verzögerung zu erkennen sowie visuelle Inhalte darauf projizieren zu können.

Zu Beginn wurde im Kapitel 2, der Analyse, die Arbeit im fachlichen Kontext der Interactive Tabletop eingeordnet. Anschließend wurde das Labor betrachtet, welches als Umgebung die Gegebenheiten für die Entwicklung und den Einsatz des Systems definiert. Zuletzt wurden konkrete Anforderungen an das zu entwickelnde System, auf Basis der Zielsetzung, definiert.

Die Konstruktion in Kapitel 3 beschrieb die Realisierung des Systems anhand der gestellten Anforderungen. Es begann mit der Erstellung eines Konzepts, das grundlegende Hardwarebausteine definierte und die Software modular auf zwei Ebenen aufteilte. Durch Experimente wurden zusätzliche Anforderungen ermittelt, die zur Auswahl und Installation konkreter Hardware führten. Das Verarbeitungssystem auf erster Ebene umfasst drei Hauptkomponenten zur Erfassung, Verarbeitung und Analyse von Bilddaten. Dessen Architektur ist modular aufgebaut und ermöglichte zukünftige Modifikationen und

Erweiterungen. Insbesondere ist das System erweiterbar um Module zur Interaktionsanalyse basierend der aufbereiteten Bilddaten. Identifizierte Interaktionen können mittels des TUIO-Protokolls an Anwendungen auf der zweiten Ebene weitergegeben werden, welche darauf basierend interaktive Anwendungsfälle realisieren.

Nach Abschluss der Konstruktion wurde in Kapitel 4 eine Evaluation des realisierten Systems vorgenommen. Hierfür wurden zunächst Untersuchungen durchgeführt, um eine Grundlage für die Bewertung zu schaffen und anschließend dessen Ergebnisse zusammen mit den Aspekten der Konstruktion diskutiert. Die Evaluation ergab, dass das konstruierte System den Anforderungen mit wenigen Ausnahmen entspricht. So ist es wartbar, zuverlässig und effizient realisiert, erfasst Interaktionen mit der Projektionsfläche auf der Basis bildgestützter Verfahren und identifiziert die spezifizierten Interaktionen mittels markierter Objekte. Bei der Visualisierung von Inhalten mittels der Projektion treten jedoch Verzerrungen auf, welche eine Darstellung von Reaktionen auf Interaktionen nur beeinträchtigt ermöglicht.

Im Kapitel 5 wurde anhand eines Anwendungsbeispiels demonstriert, wie das entwickelte System praktisch eingesetzt und konkrete Anwendungsfälle realisiert werden können. Die dabei grundlegend vorgeschlagene Architektur ermöglicht es den präsentierten Anwendungsfall zu erweitern, oder anhand dessen Vorbilds andere zu realisieren.

Das durch diese Arbeit geschaffene System stellt eine Plattform für die Untersuchung zur Identifikation von Interaktionen auf Basis visueller Informationen im Bereich der Computer Vision, sowie zur Untersuchung von Gestaltung und Nutzung interaktiver Anwendungen im Bereich der Interactive Tabletop dar. Das Living Place Labor der HAW Hamburg bietet somit erneut die Möglichkeit, Fragestellungen zur Nutzung der von Interactive Tabletop gebotenen Benutzerschnittstelle im privaten Umfeld zu erforschen.

6.2 Ausblick

Basierend auf der in dieser Arbeit umgesetzten Realisierung, eröffnen sich verschiedene Möglichkeiten daran anzuknüpfen. So kann das System erweitert, einzelne Komponenten verbessert bzw. diese mit alternativen Ansätzen verglichen und den Umgang mit dem System selbst studiert werden.

Um das System noch vielseitiger zu gestalten, können weitere Module zur Identifikation von Interaktionen im Bereich der Computer Vision entwickelt werden, die auf den bereitgestellten Bilddaten arbeiten. Darauf aufbauend, oder unter Verwendung des bereits vorhandenen Moduls, können außerdem weitere Anwendungen entwickelt werden, die interaktive Anwendungsfälle realisieren. Eine spezifische Erweiterung könnte beispielsweise darin bestehen, Toucheingaben zu identifizieren und diese als native Benutzereingabe in das Betriebssystem zu integrieren.

Durch die Implementation eines geeigneten Verfahrens zur Kalibrierung der Projektionen kann das System soweit verbessert werden, dass es anschließend den Anforderungen der Visualisierung entspricht. Um das System weiter zu optimieren, bietet sich außerdem an, verschiedene alternative Ansätze zu entwickeln und mit den bestehenden zu vergleichen. Eine konkrete Möglichkeit zur Steigerung der Effizienz besteht beispielsweise darin, die Bildverarbeitungsprozesse, mithilfe von Hardwarebeschleunigungen durch eine GPU, neu zu gestalten.

Ebenfalls bietet sich die Möglichkeit, auf Basis des realisierten Systems, das Nutzerverhalten in Bezug auf die Interaktion mit dem System an sich sowie die Benutzung spezifischer interaktiver Anwendungen zu untersuchen.

Literaturverzeichnis

[1] : *Camera Calibration and 3D Reconstruction.* – URL https://docs.opencv.org/4.7.0/d9/d0c/group__calib3d.html. – Zugriffsdatum: 2023-05-10

[2] : *Detection of ArUco Markers.* – URL https://docs.opencv.org/4.7.0/d5/dae/tutorial_aruco_detection.html. – Zugriffsdatum: 2023-05-10

[3] : *Images stitching.* – URL https://docs.opencv.org/4.7.0/d1/d46/group__stitching.html. – Zugriffsdatum: 2023-05-10

[4] : *Living place.* – URL https://livingplace.haw-hamburg.de/. – Zugriffsdatum: 2023-04-28

[5] : *NVIDIA® Warp and Blend.* – URL https://developer.nvidia.com/warp-and-blend. – Zugriffsdatum: 2023-05-16

[6] : *ITS '09: Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces.* New York, NY, USA : Association for Computing Machinery, 2009. – ISBN 9781605587332

[7] : *ISS '16: Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces.* New York, NY, USA : Association for Computing Machinery, 2016. – ISBN 9781450342483

[8] Bedi, Rishi ; Bintahir, Mohammad T. ; Çetin, Görkem C. ; D'Intino, Paul ; Hansen, Thomas ; Khoshabeh, Ramsin ; Moore, Christian ; Muller, Laurence ; Rai, Ashish K. ; Ramseyer, Nolan ; Riggio, Justin ; Roth, Tim ; Sandler, Seth ; Solms, Donovan ; Teiche, Alex ; Yane, Chris ; Bedi, Rishi (Hrsg.) ; Çetin, Görkem C. (Hrsg.) ; Sandler, Seth (Hrsg.): *Multi-Touch Technologies.* NUI Group, 05 2009. – URL https://web.archive.org/web/20090904061358/http://nuicode.com/attachments/download/115/Multi-Touch_Technologies_v1.01.pdf. – Zugriffsdatum: 2023-05-22

[9] Bhalla, Mudit R. ; Bhalla, Anand V.: Comparative Study of Various Touchscreen Technologies. In: *International Journal of Computer Applications* 6 (2010), S. 12–18. – URL https://doi.org/10.5120/1097-1433

[10] Brown, Matthew ; Lowe, David G.: Automatic Panoramic Image Stitching using Invariant Features. In: *International Journal of Computer Vision* 74 (2007), Aug, Nr. 1, S. 59–73. – URL https://doi.org/10.1007/s11263-006-0002-3. – ISSN 1573-1405

[11] Dietz, Paul ; Leigh, Darren: DiamondTouch: A Multi-User Touch Technology. In: *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*. New York, NY, USA : Association for Computing Machinery, 2001 (UIST '01), S. 219–226. – URL https://doi.org/10.1145/502348.502389. – ISBN 158113438X

[12] Dillenbourg, Pierre ; Evans, Michael: Interactive tabletops in education. In: *I. J. Computer-Supported Collaborative Learning* 6 (2011), 12, S. 491–514

[13] Gamma, Erich ; Helm, Richard ; Johnson, Ralph ; Vlissides, John M.: *Design Patterns: Elements of Reusable Object-Oriented Software*. 1. Addison-Wesley Professional, 1994. – ISBN 0201633612

[14] Garrido-Jurado, S. ; Muñoz-Salinas, R. ; Madrid-Cuevas, F.J. ; Marín-Jiménez, M.J.: Automatic generation and detection of highly reliable fiducial markers under occlusion. In: *Pattern Recognition* 47 (2014), Nr. 6, S. 2280–2292. – URL https://www.sciencedirect.com/science/article/pii/S0031320314000235. – ISSN 0031-3203

[15] Geller, Tom: Interactive Tabletop Exhibits in Museums and Galleries. In: *IEEE Computer Graphics and Applications* 26 (2006), Nr. 5, S. 6–11

[16] Gábor, Bernát: *Camera calibration With OpenCV*. – URL https://docs.opencv.org/4.7.0/d4/d94/tutorial_camera_calibration.html. – Zugriffsdatum: 2023-05-10

[17] Han, Jefferson Y.: Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection. In: *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*. New York, NY, USA : Association for Computing Machinery, 2005 (UIST '05), S. 115–118. – URL https://doi.org/10.1145/1095034.1095054. – ISBN 1595932712

[18] Iravantchi, Yasha ; Zhao, Yi ; Kin, Kenrick ; Sample, Alanson P.: SAWSense: Using Surface Acoustic Waves for Surface-Bound Event Recognition. In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA : Association for Computing Machinery, 2023 (CHI '23). – URL https://doi.org/10.1145/3544548.3580991. – ISBN 9781450394215

[19] Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model / International Organization for Standardization. Geneva, CH, 11 1994. – Standard. – URL https://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip. – Zugriffsdatum: 2023-04-27

[20] Jordà, Sergi ; Geiger, Günter ; Alonso, Marcos ; Kaltenbrunner, Martin: The ReacTable: Exploring the Synergy between Live Music Performance and Tabletop Tangible Interfaces. In: *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*. New York, NY, USA : Association for Computing Machinery, 2007 (TEI '07), S. 139–146. – URL https://doi.org/10.1145/1226969.1226998. – ISBN 9781595936196

[21] Kaltenbrunner, Martin: ReacTIVision and TUIO: A Tangible Tabletop Toolkit. In: *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. New York, NY, USA : Association for Computing Machinery, 2009 (ITS '09), S. 9–16. – URL https://doi.org/10.1145/1731903.1731906. – ISBN 9781605587332

[22] Kaltenbrunner, Martin ; Bencina, Ross: ReacTIVision: A Computer-Vision Framework for Table-Based Tangible Interaction. In: *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*. New York, NY, USA : Association for Computing Machinery, 2007 (TEI '07), S. 69–74. – URL https://doi.org/10.1145/1226969.1226983. – ISBN 9781595936196

[23] Kaltenbrunner, Martin ; Bovermann, Till ; Bencina, Ross ; Costanza, Enrico: TUIO: A Protocol for Table-Top Tangible User Interfaces, 05 2005

[24] Kaltenbrunner, Martin ; Echtler, Florian: The TUIO 2.0 Protocol: An Abstraction Framework for Tangible Interactive Surfaces. In: *Proc. ACM Hum.-Comput. Interact.* 2 (2018), jun, Nr. EICS. – URL https://doi.org/10.1145/3229090

[25] Klessascheck, Mario: *Funktionale Anforderungen versus nicht-funktionale Anforderungen*. 04 2023. – URL https://www.johner-institut.de/blog/iec-62304-medizinische-software/funktionale-und-nicht-funktionale-anforderungen/. – Zugriffsdatum: 2023-05-17

[26] Kohrs, Christin ; Angenstein, Nicole ; Brechmann, André: Delays in Human-Computer Interaction and Their Effects on Brain Activity. In: *PLOS ONE* 11 (2016), 01, Nr. 1, S. 1–14. – URL https://doi.org/10.1371/journal.pone.0146250

[27] Kubicki, Sébastien: Contribution à la prise en considération du contexte dans la conception de tables interactives sous l'angle de l'IHM, application à des contextes impliquant table interactive RFID et objets tangibles. (2011), 12

[28] Kubicki, Sébastien ; Lepreux, Sophie ; Kolski, Christophe: RFID-driven situation awareness on TangiSense, a table interacting with tangible objects. In: *Personal and Ubiquitous Computing - PUC* 16 (2012), 12, S. 1079–1094

[29] Lynch, D.K. ; Livingston, W.C.: *Color and Light in Nature*. Cambridge University Press, 2001. – URL https://books.google.de/books?id=4Abp5FdhskAC. – ISBN 9780521775045

[30] The National Hockey League (Veranst.): *National Hockey League Official Rules 2018-2019*. 2018. – URL https://www2.nhl.com/nhl/en/v3/ext/rules/2018-2019-NHL-rulebook.pdf. – Zugriffsdatum: 2023-05-01

[31] Otsu, Nobuyuki: A Threshold Selection Method from Gray-Level Histograms. In: *IEEE Transactions on Systems, Man, and Cybernetics* 9 (1979), Nr. 1, S. 62–66

[32] Ray, Sidney: *Applied Photographic Optics*. 03. Focal Press, 10 2002. – ISBN 0240515404

[33] Richards, Mark ; Ford, Neal: *Fundamentals of Software Architecture: An Engineering Approach*. O'Reilly, 2020. – ISBN 9781492043454

[34] Rosebrock, Adrian: *Image Stitching with OpenCV and Python*. 12 2018. – URL https://pyimagesearch.com/2018/12/17/image-stitching-with-opencv-and-python/. – Zugriffsdatum: 2023-05-10

[35] Steen, Maarten v. ; Tanenbaum, Andrew S.: *Distributed Systems, version 3.01 (2017)*. Maarten van Steen, 2017. – URL https://www.distributed-

systems.net/index.php/books/ds3/. – Zugriffsdatum: 2021-03-29. – ISBN 978-90-815406-2-9

[36] Zhang, Z.: A flexible new technique for camera calibration. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22 (2000), Nr. 11, S. 1330–1334

A Anhang

A.1 Werte zum Einfluss der Projektionsfläche

Die folgende Tabelle stellt die konkreten Werte der Abbildung 3.9 zur Untersuchung des Einflusses der Projektionsfläche auf den Objektdetailgrad aus Kapitel 3.2.3 dar.

Tabelle A.1: Projektionsflächeneinflüsse Werte

mm	**A** in %		**MSE**	
	1D	2D	1D	2D
1	49.76	47.83	0.341	0.318
2	52.86	49.52	0.359	0.287
3	54.59	54.59	0.255	0.294
4	91.18	55.39	0.163	0.232
5	94.36	94.87	0.085	0.146
6	97.47	96.46	0.077	0.111
7	96.19	96.67	0.061	0.067
8	96.35	96.88	0.054	0.058
9	98.41	98.94	0.047	0.049
10	97.62	97.62	0.045	0.044

A.2 Herleitung zum Anteil des genutzten Bildsensors

Sei ein Seitenverhältnis mit Breite B und Höhe H definiert als $V = B : H$ und die Fläche des Verhältnis als $A(V) = A(B : H) = B \cdot H$

Seitenverhältnis Projektionsfläche: $V_P = B_P : H_P = 67 : 22$

Seitenverhältnis Bildsensor: $V_S = B_S : H_S$

Seitenverhältnis Bildsensor mit gleich skalierter Breite: $V'_S = B_P : (H_S \cdot \frac{B_P}{B_S})$

Anteil der Nutzung: $\frac{A(V_P)}{A(V'_S)} = \frac{B_P \cdot H_P}{B_P \cdot (H_S \cdot \frac{B_P}{B_S})} = \frac{H_P}{H_S \cdot \frac{B_P}{B_S}}$

Bei $V_S = 16 : 10$ also $\frac{H_P}{H_S \cdot \frac{B_P}{B_S}} = \frac{22}{10 \cdot \frac{67}{16}} = \frac{176}{335} \approx 0,5254$

A.3 Verwendung von Frame Objekten in OpenCV

Das folgende Codebeispiel stellt die Überführung eines *Frame* Objekts in ein *Mat* Objekt des Computervision Framework OpenCV dar.

```
cv::Mat convert(Frame& frame){

    cv::Mat frame2Mat(
        frame.getImageData(),    // image data buffer
        cv::8_U1,                // image data type
        frame.getImageHeight(), // image pixel height
        frame.getImageWidth()    // image pixel width
    );

    return frame2Mat;
}
```

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original