Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Harald Kirschenmann

Angereicherte Realität in einer Smart Home Umgebung durch kamerabasierte Verfahren

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Harald Kirschenmann

Angereicherte Realität in einer Smart Home Umgebung durch kamerabasierte Verfahren

Bachelorarbeit - eingereicht im Rahmen der Bachelorprüfung

im Studiengang Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. Kai von Luck
Zweitgutachter : Prof. Dr. Philipp Jenke

Abgegeben am 24. März 2016

Harald Kirschenmann

Thema der Bachelorarbeit

Angereicherte Realität in einer Smart Home Umgebung durch kamerabasierte Verfahren

Stichworte

Lokalisation, Landmarken, OpenCV, Objekterkennung, SIFT- und SURF-Algorithmus, Merkmale, Deskriptoren, Smartglass, Angereicherte Realität, Smart Home

Kurzzusammenfassung

In dieser Arbeit wurde ein System aus dem Bereich der Angereicherten Realität in einer Smart Home Umgebung konzipiert, entwickelt und evaluiert. Die Arbeit konzentriert sich auf die Objekterkennung mit kamerabasierten Verfahren. Zusätzlich wird die Lokalisation über Landmarken besprochen. Auf Basis dieser Technologien wurde das System entworfen und anschließend realisiert. Auf Grundlage eines vorgestellten Szenarios wird die Realisierung anhand von Anforderungen evaluiert.

Harald Kirschenmann

Title of the paper

Augmented Reality in a smart home environment with camera based algorithms

Keywords

Localization, landmark, OpenCV, object recognition, SIFT, SURF, feature points, descriptor, smartglass, augmented reality, smart home

Abstract

In this paper, an augmented reality system was designed, developed and evaluated in a smart home environment. The paper focuses on the object recognition with camera based algorithms. In addition, the localization on landmarks is discussed. On the base of these technologies, the system was designed and afterwards implemented. Based on a featured scenario, the implementation is evaluated according to the requirements.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Mit der fortschreitenden Entwicklung von *Smartglasses* und *Virtual-Reality*-Brillen gewinnen die Themengebiete der *Augmented Reality* und der *Virtual Reality* an Attraktivität. Galten diese als Zukunftstechnologien, die bei der Forschung und der Industrie mit zahlreichen Anwendungsgebieten auf Interesse stießen, so werden in den letzten Monaten vermehrt Geräte für den Heimanwender angekündigt und veröffentlicht. Die Begeisterung an dem digitalen Fortschritt und dem Erweitern der Realität wird nun nach Science-Fiction Filmen und Werbeanwendungen einer breiten Öffentlichkeit zugänglich und sorgt für die Erwartung, dass sich die *Augmented Reality* und *Virtual Reality* in der Zukunft stark ausbreiten werden.

In der *Virtual Reality* wird die Wahrnehmungswelt des Anwenders künstlich geschaffen, so dass dieser in die virtuelle Realität eintauchen kann. Im Gegensatz dazu wird in der *Augmented Reality* die wahrgenommene Realität angereichert, aber nicht durch eine künstliche ersetzt. Meine Arbeit beschäftigt sich mit dem Themenfeld der *Angereicherten Realität*.

In vielen Situationen des Alltags lassen sich nicht alle Dinge der Umwelt erfassen, so dass es hilfreich ist, ergänzende Informationen über die Umwelt abzurufen. Derartige Situationen ergeben sich bei der Suche nach dem nächsten Bankautomaten oder Carsharing Auto. Die Informationen lassen sich mit dem mitgeführten Smartphone über das Internet abrufen.

Mit neueren Entwicklungen wurde die Informationsbeschaffung mit dem Smartphone vereinfacht. Mit Anwendungen wurde es möglich, Hintergrundinformationen zu Gemälden und Sehenswürdigkeiten zu erhalten, von denen Bilder mit dem Smartphone aufgenommen wurden. Mit dem Bild erkennt die Anwendung die Sehenswürdigkeit und stellt die passenden Hintergrundinformationen zur Verfügung.

Die Arbeit konzentriert sich auf eine kleinere Umgebung, den *Smart Homes*. In einem *Smart Home* steht die Wohn- und Lebensqualität im Mittelpunkt und wird durch die Vernetzung der Technik im Haus geschaffen. Die Wohnqualität wird durch zahlreiche Systeme erhöht,

die in dem *Smart Home* integriert sind und durch Interaktion den Menschen unterstützen oder seine Aufgaben übernehmen.

1.1 Gliederung der Arbeit

Diese Arbeit gliedert sich in mehrere Kapitel, die aufeinander aufbauen. Im Kapitel 2 wird das Thema der Arbeit zunächst in den Kontext eingeordnet und ein Anwendungsszenario beschrieben. Aus dem Szenario werden die Anforderungen abgeleitet, bevor die Gesten, die visuellen Objekterkennungsverfahren und die Lokalisation besprochen werden. Im Anschluss werden einige Anwendungen aufgeführt, die in der Praxis zum Einsatz kommen.

In dem Kapitel 0 wird zunächst das Design beschrieben, das für ein System der *Angereicherten Realität* im Smart Home entwickelt wurde und die Anforderungen aus Kapitel 2 erfüllt. Mit der anschließenden Realisierung des Systems in Kapitel 0 wird das Design umgesetzt und anschließend anhand der Anforderungen evaluiert.

In dem Kapitel 0 wird die Arbeit in einen übergeordneten Kontext gestellt und ein Ausblick auf zukünftige Arbeiten und Entwicklungen gegeben.

2 Analyse

In dieser Arbeit soll ein System entworfen werden, das es ermöglicht, Objekte der realen Welt mit digitalen Informationen zu verknüpfen. Das System soll hierbei mit kamerabasierten Verfahren und im Rahmen einer Smart Home Umgebung umgesetzt werden. Neben dem Erkennen der Objekte sollen die digitalen Informationen für den Anwender sichtbar gemacht werden, so dass sich diese Arbeit in das Themengebiet der *Angereicherten Realität* einordnen lässt.

Die Aufgabenstellung wird mit einem speziellen Szenario und den daraus resultierenden Anforderungen in Kapitel 2.2 präziser beschrieben. In dem Szenario wird beschrieben, wie dem Nutzer zusätzliche Informationen in Form von digitalen Notizen zur Verfügung gestellt und diese verwendet werden. Die digitale Notizen werden mit Gegenständen der realen Welt verknüpft und über technische Lösungen, wie Smartglasses oder Smartphones, für den Menschen wahrnehmbar. Mit einer Kamera der Smartglass wird der Sichtbereich aufgenommen und die Kamerabilder anschließend nach bereits bekannten Gegenständen untersucht. Wenn auf den Bildern Gegenstände gefunden werden, die zuvor mit digitalen Notizen verknüpft wurden, werden diese dem Anwender angezeigt. Das Erstellen und Manipulieren neuer Notizen sowie das Registrieren neuer Gegenstände im System ist ebenso möglich.

In dem Kapitel 2.1 werden zunächst die Themenfelder der A*ngereicherten Realität* und der *Virtuellen Realität* und in Kapitel 2.1.1 die Smartglasses beschrieben.

In dem Kapitel 2.3 werden die Anforderungen, die sich aus dem Szenario und den Anwendungsfällen in Kapitel 2.2 ergeben, herausgearbeitet. Die Anforderungen konkretisieren die Aufgabenstellung und beeinflussen die Design-Entscheidungen in Kapitel 3.

2.1 Angereicherte Realität und Smartglasses

Unter dem Begriff *Erweiterte Realität (*[Toennis 2010] *und* [Doerner 2013]*)*, auch *Angereicherte Realität* oder *Augmented Reality* (AR) genannt, wird eine Erweiterung der wahrgenommenen Realität verstanden. Zur Erweiterung der Realität wird die Wahrnehmung mit zusätzlichen Informationen ergänzt. Mit diesen Zusatzinformationen können unterschiedliche Sinne angesprochen werden. Verbreitet ist die visuelle Darstellung der zusätzlichen Informationen, hierbei überlagern diese das Bild in der Darstellung. Aus der Perspektive des Anwenders erfolgt die Darstellung korrekt und in Echtzeit.

In vielen Bereichen kann mit AR ein sinnvoller Mehrwert erzielt werden. im „Marketing" z.B. lassen sich Produktpräsentationen innovativ gestalten und komplexe Informationen leicht und verständlich vermitteln. Neben Navigationssystemen bietet die AR auch für Helfer in Katastrophensituationen eine passende Einsatzmöglichkeit. Das Auffinden und Retten von Verletzten oder eingeschlossenen Menschen ist ein elementarer Bereich der Rettungskräfte, doch das Auffinden und Navigieren durch komplexe oder verrauchte Räume stellt eine große Herausforderung dar. Mit Hilfe der AR lässt sich der passende Weg für den Retter gut darstellen. Weitere bereits fertiggestellte Anwendungen werden in Kapitel 2.7 beschrieben.

Im Gegensatz zur *Augmented Reality* wird in der *Virtuellen Realität*, auch als *Virtual Reality* (VR) bezeichnet, die Wahrnehmungswelt künstlich erschaffen. In der VR taucht der Nutzer in die virtuelle Welt ein, die Wahrnehmung der realen Welt wird verringert und der Nutzer fühlt sich als Teil der virtuellen Welt. Die künstlich geschaffene Welt muss dabei plausibel, d.h., in sich stimmig sein und Interaktionen des Nutzers müssen Einfluss auf die Umgebung haben. Durch die Möglichkeit, mit der Welt in Echtzeit zu interagieren, wird für den Nutzer eine Illusion erschaffen.

In begrenztem Rahmen kann mit der VR Einfluss auf den Nutzer genommen werden. Üblicherweise werden Bewegungen des Nutzers in der virtuellen und der realen Welt parallel ablaufen. Bewegt sich der Nutzer in der VR gerade aus, so geschieht dies parallel in der realen Welt. Wie folgende Experimente zeigen, kann die Wahrnehmung paralleler Bewegungen mit Hilfe der VR manipuliert werden:

Experimente zeigen, dass Menschen es nicht schaffen, sich auf einer geradlinigen Bahn zu bewegen, wenn ihnen die visuelle und auditive Wahrnehmung fehlt. Anstelle der erwarteten geradlinigen Bahn bewegten sich die Versuchsteilnehmer auf einer Kreisbahn. Es wird angenommen, dass sich verschiedene Fehler in der Wahrnehmung summieren und

nicht über visuelle oder auditive Sinne korrigiert werden. Die Kreisbahn einiger Versuchsteilnehmer betrug einen Radius von etwa 20 Metern [Souman u.a. 2009].

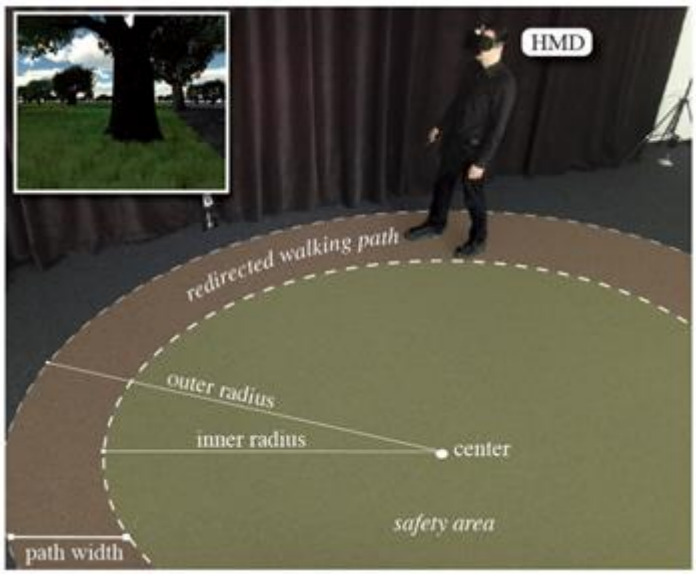

Abbildung 1: Kreisbahn und Nutzer mit Head-Mounted Display [Lubos u.a. 2014]

Weitere Versuche zeigen, dass Menschen in einer virtuellen Welt in einem Kreis mit dem Radius von 20 Metern geleitet werden können, ohne dass diese die Manipulation bemerken. Aus diesen Ergebnissen lässt sich die Erkenntnis gewinnen, dass zur Simulation einer unbegrenzten virtuellen Welt die Größe eines begrenzten Raumes genügt.

In den Bereichen AR und VR eignen sich die Verwendung von Head-Mounted-Displays[1] (HMD). Kommen HMDs in der VR zum Einsatz, werden optisch geschlossene Brillen verwendet. In der AR dagegen werden HMDs mit sogenannten *Optical See-Through Displays* verwendet. Diese Art der Displays ermöglicht es dem Nutzer, die reale Umwelt durch die Brille zu sehen und zeitgleich dem Nutzer ein Bild vor dem Auge einzublenden. Zur mobilen Anwendung eignen sich im besonderen Maße Smartglasses.

[1] Digitale Anzeigen (Displays), die vom Nutzer am Kopf getragen werden.

2.1.1 Smartglass

> "We define eye-wearable technology as any technology having a digital display that is worn close to the eye and does not fully obstruct the real world view; this is also referred as smartglasses, or digital eye glasses" [Zheng 2015]

Smartglasses oder auch *Datenbrillen* sind Brillen, die das HMD mit einer integrierten Rechnereinheit kombinieren. Markant an den Brillen ist das Display, welches sich vor beiden Augen (binokular) oder nur einem Auge (monokular) befinden kann.

In einer Smartglass werden zahlreiche Sensoren integriert, um eine große Bandbreite an Anwendungen realisieren zu können. So werden in der „Google Glass" ein Touchpad, Mikrofon, Beschleunigungssensor, Kompass, Eyetracker[2], Lagesensor, Gyroskop, Näherungssensor, Helligkeitssensor und eine Kamera verbaut, hinzu kommt ein Micro-USB Anschluss. Über WLAN und Bluetooth kann die Brille mit anderen Geräten in der Umgebung kommunizieren.

Die Firma „*Google*" erreichte mit ihrem „*Project Glass*" (später „*Google Glass*") eine große mediale Öffentlichkeit und begeisterte viele für die Einsatzmöglichkeiten der Smartglasses in der Augmented Reality. Einige Unternehmen stellten schon zuvor erste Modelle der Öffentlichkeit vor oder folgten kurz darauf.

Neben der Begeisterung stießen die Datenbrillen in der Öffentlichkeit auch oft auf Ablehnung. So konnten anwesende Personen nicht unterschieden, ob mit der Brille Videos oder Bilder aufgenommen werden. Daher fühlten sich viele Menschen in der Öffentlichkeit der Gefahr ausgesetzt heimlich, gefilmt zu werden, und zeigten ihren Unmut deutlich. Dies führte dazu, dass viele Anwender die Brillen in der Öffentlichkeit nicht mehr tragen. In Deutschland äußerten auch zahlreiche Datenschützer im Vorfeld der Veröffentlichung Kritik an dem Einsatz der Brillen (vgl: [Schulz 2014]).

[2] Systeme, die eine Analyse der Blickbewegung ermöglichen

2.2 Anwendungsfälle

In diesem Abschnitt wird ein Szenario entworfen, das die Verwendung einer Smartglass in einem Smart Home darstellt. Die Realität wird durch das Erstellen und Verwenden von digitalen Notizen angereichert. Die Notizen werden virtuell an markante Gegenstände geheftet. Die Erkennung der Gegenstände wird durch kamerabasierte Verfahren realisiert.

Das Szenario wird in die Arbeitsschritte der Objektregistrierung, Notizerstellung, Manipulation und die Anzeige der Informationen unterteilt.

2.2.1 Szenario

Smartglasses gehören zu den alltäglichen Accessoires von Herrn M und seiner Mitbewohnerin Frau B. Nach einem langen Arbeitstag lässt Herr M sich den besten Weg in das Smart Home zeigen. Zuhause angekommen entscheidet er sich, etwas zum Abendessen zu kochen. Hierzu sucht er über die Smartglass ein Rezept, das ihm über das Head-Mounted-Display dargestellt wird. In der Küche überprüft Herr M den Inhalt des Kühlschranks. Die Eindrücke inspirieren ihn zu einem neuen Gericht, das Herr M am kommenden Tag zubereiten möchte. Für das neue Gericht fehlen einige Zutaten, die für den morgigen Tag gekauft werden müssen. Herr M betrachtet daher den Kühlschrank ruhig mit der Brille und signalisiert der Brille, dass er eine neue digitale Notiz hinzufügen möchte. Die Notiz lässt sich nun per Spracheingabe oder mit Hilfe weiterer Eingabegeräte erstellen. Herr M schreibt in die digitale Notiz die benötigten Lebensmittel und hinterlegt das Rezept für sein Gericht.

Nach der Zubereitung des Abendessens begibt sich Herr M in sein Wohnzimmer. Bei dem Betreten erscheint auf dem HMD der Fernseher farblich hinterlegt. Die Farbgebung signalisiert, dass eine Notiz an dem Gerät hinterlegt wurde. Nach ruhigem Blick auf das Gerät erscheint die Notiz auf dem HMD. In der Notiz befindet sich der Hinweis auf einen ausgeliehenen Film, den sich Herr M daraufhin ansieht. Parallel zu dem Film sieht sich Herr M eine Zeitschrift mit Programmvorschau an und entdeckt eine Sendung, die seine Mitbewohnerin Frau B interessieren könnte. Die neue Zeitschrift wurde zuvor weder für digitale Notizen verwendet, noch als Objekt registriert. Damit die Zeitschrift vom System wieder erkannt wird, muss Herr M die Zeitschrift registrieren. Zur Registrierung betrachtet Herr M die Zeitschrift aus verschiedenen Perspektiven und legt nun eine neue Notiz für die Mitbewohnerin an.

Später betritt Frau B nach einer Nachtschicht die gemeinsame Wohnung. Beim Betreten des Wohnzimmers fällt ihr schnell die Programmzeitschrift ins Auge. Diese wird vom HMD farblich hinterlegt. Sie ruft die Information ihres Mitbewohners auf und löscht diese nach dem Lesen. Die Zeitschrift wird nun nicht mehr farbig hinterlegt. Beim Gang in die Küche betrachtet Frau B den Kühlschrank und ruft die dazugehörige Notiz ab. Die Einkaufsliste des Herrn M ergänzt sie um die Lebensmittel, die sie für die kommenden Tage benötigt.

2.2.2 Objektregistrierung und Informationsverknüpfung

Mit Hilfe der Smartglass soll der Anwender jedes ausreichend markante Objekt zur Registrierung nutzen können. Der Anwender wählt im Raum ein Objekt, das er mit digitalen Informationen verknüpfen möchte. Mit der Kamera der Brille betrachtet der Anwender das Objekt aus verschiedenen Perspektiven formatfüllend und nimmt aus verschiedenen Perspektiven Bilder auf. Handelt es sich um ein unbewegliches Objekt, bestimmt das System die Position des Anwenders. Die Bilder werden in der Datenbank hinterlegt und mit der Position verknüpft. Wurde keine Position bestimmt, so wird in allen Räumen nach diesem Objekt gesucht. Die Bilder werden analysiert und aufbereitet, um ein späteres Erkennen der registrierten Objekte zu ermöglichen. Die Objekte werden mit der Position verknüpft. Jedem registrierten Objekt können Informationen, wie digitale Notizen, hinzugefügt werden.

Zum Hinzufügen von digitalen Notizen wählt der Anwender ein bereits registriertes Objekt durch direkte Betrachtung aus. In dem Videostream der Kamera erkennt das System die registrierten Objekte wieder. Hierzu werden die Informationen aus den hinterlegten Bildern mit dem Videostream verglichen. Um die Suche einzuschränken, wird die Position des Anwenders bestimmt. Mit der Position lässt sich die Anzahl der gesuchten registrierten Objekte auf die, die sich in dem gleichen Raum befinden, begrenzen. Nachdem das Objekt vom System erkannt worden ist, kann der Anwender digitale Notizen an dieses binden. Die Notiz kann über verschiedene Eingabegeräte erstellt werden. Sowohl eine Text- als auch Spracheingabe bietet sich für die Bereitstellung der Informationen an. Die erstellten Informationen werden nach Abschluss mit dem Objekt verknüpft und stehen dem System für den späteren Zugriff bereit.

2.2.3 Erkennung und Darstellung von Informationen

Die gespeicherten Informationen sind im Raum nicht sichtbar. Nur mit technischen Hilfsmitteln stehen diese dem Anwender zur Verfügung. Dieser befindet sich in bekannter Umgebung und betrachtet ein zuvor registriertes Objekt. Die Position des Anwenders wird vom System ermittelt. Bei der Betrachtung befindet sich das Objekt möglichst vollständig und groß im Sichtbereich der Kamera in der Brille. Betrachtet der Anwender das registrierte Objekt, so wird dieses vom System wiedererkannt. Um die Erkennung zu unterstützen, wird die Position hinzugezogen und die Anzahl der möglichen Objekte dadurch gefiltert. Nach erfolgreichem Erkennen wird das Objekt ausgewählt und Informationen, die mit dem Objekt verknüpft sind, werden gesucht. Werden Informationen gefunden, wird das Objekt auf dem Head-Mounted-Display farbig hinterlegt dargestellt. Betrachtet der Anwender das Objekt für einen bestimmten Zeitraum, so werden die bereitgestellten Informationen auf dem HMD angezeigt.

2.2.4 Manipulation von Informationen

Die Informationen, die mit unterschiedlichen Objekten verknüpft wurden, lassen sich auch ändern. Um die verknüpften Informationen eines Objektes zu ändern, betrachtet der Anwender dieses Objekt. Befindet sich das Objekt möglichst vollständig und groß in dem Sichtbereich der Kamera, so wird dieses von dem System erkannt. Wie in 2.2.3 dargestellt, wird auch für die Manipulation die Position bestimmt und zum Filtern der registrierten Objekte verwendet. Nachdem das Objekt erkannt wurde, wird nach vorhandenen Informationen gesucht. Werden Informationen gefunden, die dem Objekt zugeordnet werden können, so wird das Objekt im HMD farbig hinterlegt und steht nun zur Manipulation bereit. Die Informationen können wie bei der Erstellung mit Hilfe verschiedener Eingabesysteme, wie Text- oder Spracheingabe geändert werden. Zum Abschließen der Manipulation wird die Eingabe bestätigt.

2.3 Anforderungen

In diesem Abschnitt werden die Anforderungen aus dem Szenario und den Anwendungsfällen, in Kapitel 2.2 beschrieben, extrahiert. Die Anforderungen konkretisieren die Aufgabenstellung und beeinflussen die Designentscheidungen in Kapitel 0. Diese werden für den Entwurf eines Systems benötigt, das das Szenario und die daraus resultierenden Anwendungsfälle umsetzt.

2.3.1 Lokalisation von Objekten

Aus dem Anwendungsfall 2.2.2 ergibt sich die Anforderung, die Position von Objekten zu bestimmen. Die Position bestimmt sich aus der Lage der Objekte in einem Referenz-Koordinatensystem. In dieser Anwendung erfüllt die Lokalisation den Zweck zu bestimmen, in welchem Raum sich das Objekt befindet. Falls es sich um ein bewegliches Objekt handelt, so kann dem System signalisiert werden, dass nach dem Objekt in jedem Raum gesucht werden soll oder die Position neu bestimmt werden muss, sobald sich der Standort ändert.

2.3.2 Bildaufnahme

Das Aufnehmen von Bildern mit der Brille ist eine der Voraussetzungen für das Registrieren von Objekten. Dafür werden Referenzbilder der Objekte aufgenommen. Auf diesen Bildern soll das Objekt fokussiert sein, welches das System anschließend erkennen soll. Hierbei müssen verschiedene Faktoren beachtet werden. Das Objekt muss ausreichend belichtet werden, darf jedoch nicht überbelichtet sein. Das Bild muss in einem ruhigen Moment aufgenommen werden, um ein Verwischen des Bildes zu vermeiden. Die Entfernung zwischen Kamera und Objekt muss so abgestimmt werden, dass das Objekt formatfüllend aufgenommen wird. Die Perspektive kann ebenfalls einen wichtigen Faktor darstellen.

2.3.3 Objekte im System registrieren

In dem Anwendungsfall 2.2.2 wird beschrieben, dass Objekte im System registriert werden sollen. Dies geschieht, um die Objekte später wieder zu erkennen. In der Anforderung 2.3.1 wurde bereits die Lokalisation der Objekte beschrieben. Der Raum, in dem sich das Objekt befindet, wird mit einem eindeutigen Bezeichner in dem System hinterlegt. In der Anforderung 2.3.2 wurde die Anforderung beschrieben, Bilder mit der Brille aufzunehmen. Von dem unverdeckten Objekt werden Bilder aus verschiedenen Perspektiven angefertigt. Diese werden als Vergleichsbilder verwendet, aus denen wiederum „Merkmale“ extrahiert und gemeinsam mit der Position im System hinterlegt werden.

2.3.4 Erstellen und Verknüpfen von Informationen

Aus dem Anwendungsfall 2.2.2 ergibt sich die Anforderung, Informationen zu erstellen. Der Nutzer kann Informationen über unterschiedliche Schnittstellen eingeben.

Nach dem Erstellen werden die Informationen mit registrierten Objekten verknüpft und im System gespeichert. Hierzu muss der Gegenstand, mit dem die Informationen verknüpft werden sollen, von dem Anwender betrachtet und ausgewählt werden.

2.3.5 Lokalisation des Nutzers

Die Lokalisation des Nutzers wird, wie in 2.2.3 beschrieben, genutzt, um zu ermitteln, in welchem Raum sich dieser befindet. Die Position bestimmt sich, wie in der Anforderung 2.3.1, aus der Lage des Nutzers in einem Referenz-Koordinatensystem. Der Nutzer bewegt sich in dem Smart Home, dementsprechend muss die Position des Nutzers jeweils aktualisiert werden.

2.3.6 Objekte in Bildern erkennen

Aus dem Anwendungsfall 2.2.3 ergibt sich die Anforderung, in einem Bild Objekte wieder zu erkennen. Die Gegenstände, die erkannt werden sollen, sind in Form von „Merkmalen“ im System hinterlegt.

Mit der Kamera werden kontinuierlich Bilder aufgenommen (Video), die anschließend vom System untersucht werden. Aus dem Kamerabild werden die Merkmale extrahiert und anschließend mit den Merkmalen der im System hinterlegten Bilder verglichen. Wird nach dem Vergleich eine ausreichende Übereinstimmung der Merkmale festgestellt, so wird das Objekt in dem Kamerabild wiedererkannt.

Das Erkennen von Objekten soll hierbei gegenüber Änderungen von verschiedenen Parametern robust sein. Dazu gehören:

- Änderungen in der Belichtung
- Änderungen in der Skalierung
- Rotation des Objektes
- Verdeckung von Teilen des Objektes
- Perspektivwechsel

Um die Suche nach Objekten zu beschleunigen, kann der Suchraum eingeschränkt werden. Dazu wird neben dem Kamerabild auch die Position des Nutzers bestimmt. So ist es möglich, nur nach Objekten zu suchen, die sich in dem gleichen Raum wie der Nutzer befinden. Zu berücksichtigen sind Objekte, deren Position sich häufig ändern kann, wie beispielsweise Zeitschriften. Die Positionsbestimmung ist in Anforderung 2.3.5 beschrieben.

Das Objekt auf dem Kamerabild darf nicht zu stark verdeckt sein. Eine Verdeckung kann das Erkennen einschränken. Notwendigerweise müssen die Objekte, wie in 2.3.3 beschrieben, im System hinterlegt sein, bevor diese erkannt werden können.

2.3.7 Auswahl von Objekten

Werden in dem Kamerabild registrierte Objekte gefunden, so besteht die Möglichkeit diese auszuwählen. Dem System kann die Auswahl über Gesten signalisiert werden oder indem das Objekt für einen bestimmten Zeitraum betrachtet wird. Werden mehrere Objekte in dem Kamerabild ermittelt, so gewinnt die Selektion an Bedeutung.

2.3.8 Abfrage von Informationen

Die Anwendungsfälle 2.2.3 und 2.2.4 enthalten die Abfrage von Informationen. Die Informationen sind im System gespeichert und mit registrierten Objekten verknüpft. Sobald Informationen zu einem registrierten Objekt abgefragt werden, wird die Information anhand der Verknüpfung abgefragt und zugeordnet.

2.3.9 Manipulation von Informationen

Wie in 2.2.4 beschrieben, ist ein Anwendungsfall das Manipulieren der Informationen, die mit einem Objekt verknüpft wurden. Änderungen für zuvor abgefragte Informationen können über verschiedene Schnittstellen eingegeben werden.

2.3.10 Darstellung von Informationen

Aus dem Anwendungsfall 2.2.3 ergibt sich die Anforderung, die abgefragten Informationen für den Nutzer sichtbar darzustellen. Hierzu kommt das Display des Head-Mounted-Displays zum Einsatz. Die Informationen werden auf dem HMD dargestellt.

Bei der Darstellung der Informationen muss darauf geachtet werden, dass die Informationen eingeblendet werden, ohne wichtige Bereiche im Sichtbereich zu verdecken.

Das Objekt, mit dem die Informationen verknüpft wurden, kann auf dem HMD farblich hervorgehoben werden. Dies geschieht, um die Selektion zu verdeutlichen. Die Überlagerung muss optisch passend geschehen.

2.3.11 Interaktion

Für einige Aktionen muss dem System verdeutlicht werden, dass eine bestimmte Aktion folgen soll. Die Einleitung oder die Auswahl zwischen mehreren Aktionen wird durch Interaktionen mit dem System gesteuert. Wenn mehrere Objekte erkannt wurden, so kann die Auswahl eines bestimmten Objektes über eine Interaktion bestimmt werden. Die Manipulation des Systems kann durch eine andere Interaktion eingeleitet werden. Dementsprechend sollen die Interaktionen dem System verdeutlichen, welche Aktionen ausgeführt werden sollen. Eine solche Interaktion kann über Sprache oder Gesten erfolgen.

2.3.12 Informationsüberladung vermeiden

Durch die Darstellung auf dem HMD darf der Sichtbereich nicht überladen werden. Wenn sich in dem Sichtbereich viele registrierte Objekte befinden, kann es leicht passieren, dass auf dem Display zu viele Gegenstände dargestellt werden und die Übersicht verloren geht. Es kann stark ablenken, wenn sich der Nutzer in einem Raum umsieht, der Sichtbereich sich schnell ändert und daher alle registrierten Objekte schnell hintereinander im Bereich dargestellt werden. Eine Lösung kann sich ergeben, wenn der Sichtbereich für einen bestimmten Zeitraum betrachtet werden muss, bevor die Gegenstände erkannt werden.

2.3.13 Zeitliche Bedingungen

Eine weitere Anforderung an das System stellen zeitliche Bedingungen an das System dar. Das System soll innerhalb eines bestimmten Zeitraumes, auch *Latenz* genannt, die Objekterkennung und die Darstellung auf dem HMD durchführen. Für die Objekterkennung und die Darstellung gelten jeweils eigene Vorgaben.

Bei Anwendungen der Virtual Reality können bei hoher Latenz Nebenwirkungen, wie Schwindel und Übelkeit, auftreten. Bei dem Anwender treten die Schwindelgefühle auf, wenn sich durch Bewegungen das tatsächliche Bild von dem erwarteten Bild unterscheidet

(vgl. [Doerner 2013], S. 195-197). In der Augmented Reality kann der sogenannte *Schwimmeffekt* auftreten. Wenn das die Realität überlagernde virtuelle Bild zeitlich versetzt zu der Realität erscheint, tritt der Effekt auf. Bei einem ruhenden Bild oder einer möglichst niedrigen Latenz ist der Effekt nicht auffällig (vgl:[Hellwig 2013], S. 18). Daher sollte versucht werden, die Latenz zur Darstellung des Bildes möglichst gering zu halten, so dass eine Verzögerung von 200 Millisekunden für die Anzeige nicht überschritten wird.

Wie in Anforderung 2.3.12 beschrieben, soll vermieden werden, dass während des Umsehens im Raum zu viele Objekte auf dem Display erscheinen und die Darstellung somit überladen wird. Aus dem Grund sollen die Objekte für einen Zeitraum von 1,5 Sekunden ruhig betrachtet werden. Danach wird dem System verdeutlicht, dass in diesem Bereich nach Objekten gesucht werden soll.

2.3.14 Zusammenfassung und Ausblick

In dem Kapitel 2.3 wurden aus dem Szenario die Anforderungen extrahiert und ausführlich beschrieben. Zu den Anforderungen gehören das Registrieren von Objekten, die Verknüpfung und Abfrage von Informationen, die zeitlichen Bedingungen und das Wiedererkennen von registrierten Objekten. In den folgenden Abschnitten des Kapitel 2 werden zu den einzelnen Anforderungen die Hintergründe und Ansätze zur Problemlösungen näher erläutert. Auf Basis der hier beschriebenen Technologien wird nachfolgend das Design entworfen.

In dem Abschnitt 2.4 werden passend zu der Anforderung „Interaktion“ einige Arten vorgestellt. Darunter befinden sich verschiedene Gesten und die Sprachsteuerung.

Zum Wiedererkennen von Gegenständen in Bildern gibt es verschiedene Algorithmen, von denen einige in dem Abschnitt 2.5 vorgestellt werden. Hierbei werden drei verschiedene Arbeitsschritte unterschieden. Zunächst wird unter 2.5.1 die Berechnung der Keypoints und anschließend in 0 die Ermittlung der Deskriptoren, gefolgt von dem *Matching* unter 2.5.3 beschrieben.

Die Anforderung zur Positionsbestimmung des Nutzers oder der Objekte wird in dem Abschnitt Lokalisation 2.6 näher betrachtet. Dabei werden verschiedene Arten der Lokalisation dargestellt und anschließend zwei Praxisbeispiele genannt.

Am Ende dieses Kapitels werden vergleichbare Arbeiten (2.7) vorgestellt. Darunter befindet sich ein anderer Ansatz und verschiedene Anwendungen der Augmented Reality, die in verschiedenen Bereichen, wie in der Produktpräsentation, zum Einsatz kommen

2.4 Interaktion und Gesten

Im Folgenden beschäftigt sich dieses Kapitel mit Gesten und verschiedenen Möglichkeiten der Interaktion. Gesten sind Bewegungen, die ausgeführt werden, um Gegenstände zu manipulieren oder um zu kommunizieren. Eine Schwierigkeit in der Erkennung von Gesten liegt in der Unterscheidung zwischen Gesten und unabsichtlich ausgeführten Bewegungen (vgl. [Pavlovic u. a. 1997], S. 680).

Bei den Gesten kann unterschieden werden, ob diese berührungslos oder mit einer Berührung an der Brille durchgeführt werden. Gesten begegnen dem Menschen im alltäglichen Leben, sei es bei der Nutzung eines Smartphones oder in der nonverbalen Kommunikation mit anderen Menschen. Mit den Gesten wird die Interaktion mit den Geräten ermöglicht und eine möglichst natürliche Bedienung angestrebt. Durch die Verwendung möglichst intuitiver Gesten wird dem Anwender die Benutzung der Geräte ohne lange Einarbeitungszeiten ermöglicht. Die Suche nach den passenden Gesten beschäftigt die Wissenschaft weiterhin. So zeigen Untersuchungen, dass Nutzer bei der intuitiven Anwendung von Gesten unterschiedliche Gesten für die gleichen Befehle verwenden. Die zuvor definierten Gesten der Autoren entsprachen mit 43,5 % nur einem kleinen Teil der verwendeten Gesten (vgl: [Wobbrock 2009], S. 8). Für einzelne Anweisungen lassen sich häufig mehrere passende Gesten finden, diese können auch über unterschiedliche Sensoren erfasst werden. Der Anwender kann sich nach seinen Präferenzen die für ihn passende wählen. Die Auswahlmöglichkeit mehrerer Gesten für eine Anweisung führt zu einer intuitiveren Bedienung und einer reduzierten Fehleranfälligkeit (vgl: [Koelsch 2006], S. 68-69).

In Smartglasses werden mehrere Sensoren verbaut, die es ermöglichen, unterschiedliche Gesten zu erkennen. In einigen Brillen wird ein Touchpad an dem Brillengehäuse verbaut, an dem *Touch-Gesten*[3] ausgeführt werden können. Für berührungslose Gesten bieten Datenbrillen verschiedene Sensoren. In den Brillen werden zur Spracheingabe Mikrofone verwendet und eine Kamera verbaut, die den Bereich vor dem Kopf abdeckt. Einige weitere Brillen bieten neben dieser Kamera eine weitere, die die Bewegung der Augäpfel registriert,

[3] Gesten, die im 2-Dimensionalen Raum durch Berührung einer Oberfläche ausgeführt werden.

anhand derer sich die Pupillenposition berechnen lässt (vgl: [Schulz 2014]). Im Folgenden wird der Fokus des Kapitels auf berührungslose Gesten und Interaktionen gelegt.

2.4.1 Augensteuerung mit Eye-Tracking

Mit der Augenpartie lassen sich verschiedene Gesten durchführen. Gesten, wie der Blick in unterschiedliche Richtungen, das Blinzeln, Augenbrauenbewegungen oder dem Blick auf einen bestimmten Bereich vor dem Kopf, können mit den Augen, Augenlidern und den Augenbrauen ausgeführt werden (vgl: [Ashwash]).

Die Augen werden mit infrarotem Licht erleuchtet und mit Kamerasensoren gefilmt. Aus den Bildern lässt sich die Pupillenposition berechnen ([Babcock 2004], Seite 2). Daraus wiederum lässt sich die Blickrichtung der Augen bestimmen. Indem die Blickrichtung als „Zeiger“ angesehen wird, kann diese als Geste interpretiert werden. So kann der Blick auf ein bestimmtes Objekt als Selektion interpretiert werden. Ruht der Blick auf einem wählbaren Gegenstand, so wird dieser nach einer definierten Zeit ausgewählt.

2.4.2 Handgesten

Gesten, die mit der Hand ausgeführt werden, können viele Bedeutungen repräsentieren. In der Gebärdensprache werden Handgesten mit Mimik und Körperhaltung kombiniert. Einzelne Handgesten repräsentieren in dem Fingeralphabet[4] einzelne Buchstaben. Diese werden in der Gebärdensprache verwendet, um Eigennamen sowie unbekannte Wörter zu buchstabieren. In der Abbildung 2 ist eine Auswahl an Handgesten für die Interaktion mit Geräten dargestellt:

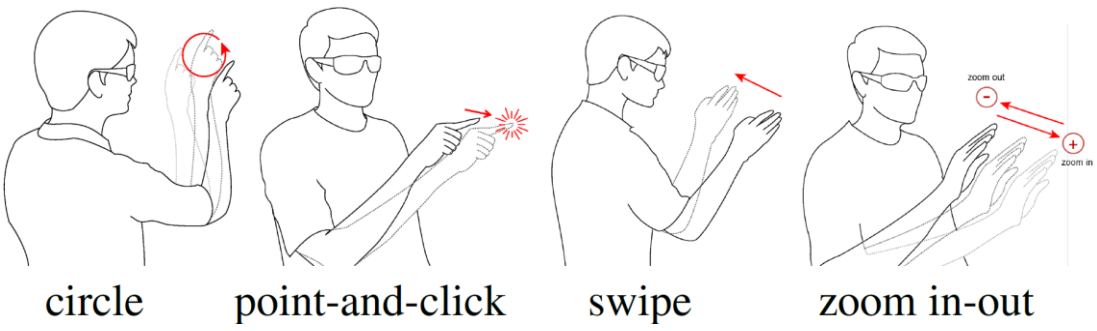

Abbildung 2: Mit der Hand ausgeführte Gesten ([Heidemann 2004], S.51)

[4] http://www.sign-lang.uni-hamburg.de/fa/

Für die Selektion von Auswahlmöglichkeiten in Menüs stellen Zeigegesten mit der Hand eine Möglichkeit zur Interaktion. Auch zur Auswahl von Objekten bieten sich Zeigegesten an. Probleme bei der Erkennung von Zeigegesten mit der Hand können anhand von zwei Faktoren auftreten. Zum einen ist die Hand frei beweglich, weshalb die Gesten aus verschiedenen Winkeln erkannt werden müssen und zum anderen ist der Mensch daran gewöhnt, die Richtung genau anzugeben. Aus dem Kontext ergibt sich üblicherweise der genaue Hinweis. Diese Problematik kann gelöst werden, indem Objekte von Interesse gefunden werden und diese anschließend zur Auswahl stehen (vgl: [Heidemann 2004]).

Ein Sensor, der Handbewegungen mit einer hohen Genauigkeit ermitteln kann, wurde mit dem *Leap Motion*[5] im Jahr 2012 der Öffentlichkeit vorgestellt. Der Sensor kann neben der gesamten Hand auch die Position einzelner Finger ermitteln. Hierdurch nimmt die Bandbreite an erkennbaren Handgesten zu. In praktischen Messungen wurde eine Genauigkeit von 0,7 mm festgestellt. Die Funktionsweise der *Leap Motion* basiert auf *Stereo Vision*[6]. Neben drei Infrarot (IR)-Emitter verfügt der Sensor über zwei IR-Kameras (vgl: [Weichert 2013]).

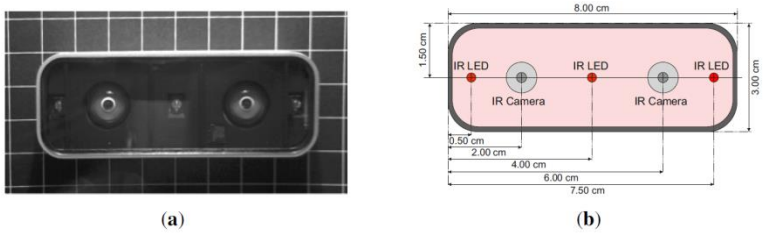

Abbildung 3: Schematischer Aufbau der Leap Motion ([Weichert 2013])

Eine Kombination von *Leap Motion* und der *Kinect*[7] stellt sich in Untersuchungen als sinnvoll heraus, denn diese erhöht die Interaktionsmöglichkeiten auf eine schnelle und einfach anzuwendende Weise. Die *Kinect* generiert ein 3-dimensionales Bild des Anwenders und die *Leap Motion* ermittelt die genauen Handbewegungen (vgl: [Penelle 2014]).

[5] https://www.leapmotion.com/
[6] Räumliches Sehen
[7] https://dev.windows.com/en-us/kinect

2.4.3 Sprachsteuerung

Die Sprache ist für den Menschen ein gewohntes Kommunikationsmittel. Mit der Sprache lassen sich unter anderem Wünsche und Befehle ausdrücken. Mit Sprachsteuerung werden die Anweisungen des Menschen interpretiert und als Befehle für die Geräte umgesetzt.

Für die Spracherkennung wird erst das Audiosignal vorverarbeitet und anschließend mit dem *Hidden-Markov-Modell* (HMM) oder mit *Neuronalen Netzen* analysiert. Das HMM ist ein statistisches Modell zur Signalverarbeitung, das über einen endlichen Automaten mit finiten Zuständen und Zustandsübergängen, die Wahrscheinlichkeiten abbildet ([Blunsom 2004]). Die gesprochene Sprache bildet sich aus sogenannten Phonemen, d.h. Lauten die aneinander gereiht (Lautfolge) Wörter ergeben. Das Erkennen von Phonemen ist ein Kernproblem der Spracherkennung und wurde in [Sarma und Sarma 2012] exemplarisch mit *Neuronalen Netzen* für die assamesische Sprache umgesetzt.

Spracherkennung kommt in verschiedenen Bereichen der *Mensch-Computer Interaktion* zum Einsatz, wie z.B. in der Robotik oder in Automobilen. In den Automobilen kann die Musik oder das Navigationssystem gesteuert werden. Hierbei kommuniziert das System mit dem Menschen über einen Dialog und leitet ihn so durch die Auswahlmöglichkeiten.

In einem Dialog erweist es sich als sinnvoll, geschlossene Fragen zu stellen und die Antwortmöglichkeiten vorzugeben. Die mentale Belastung des Anwenders wird so reduziert und die Ablenkung von der Sprachsteuerung reduziert. Somit kann sich der Anwender weiterhin auf sein eigentliches Anliegen konzentrieren (vgl: [Rabiner 1993] und [Hamberger u.a. 2010]).

2.4.4 Kombinierte Gesten

Die kombinierte Interaktion durch verschiedene Gesten und Gestenarten wird *Multimodale Interaktion* genannt und ermöglicht eine intuitivere Anwendung. Durch die kombinierte Anwendung steigen die Möglichkeiten der Interaktionen. Die Kombination von Gesten kann sequentiell erfolgen. Die sequentielle Ausführung ist ein verbreitetes Konzept und den meisten Menschen bei der Benutzung eines Computers geläufig. Auch der Wechsel zwischen den Eingabegeräten ist mit dem Wechsel zwischen Maus und Tastatur geläufig. Für eine konfliktfreie Kombination von Gesten werden mit der sequentiellen Ausführung die besten Ergebnisse erzielt (vgl: [Koelsch 2006], S. 69).

Als Beispiel für die Kombination von Gesten verschiedener Eingabearten bietet sich die Augensteuerung mit Handgesten oder Sprachsteuerung an. Mit Hilfe der Augensteuerung wird auf einen Gegenstand gezeigt. Mit einer Handgeste oder mit Hilfe der Sprachsteuerung kann der Gegenstand ausgewählt oder manipuliert werden.

2.5 Visuelle Objekterkennungs-Verfahren

In diesem Kapitel werden Algorithmen vorgestellt, die Merkmale in Bildern erkennen. Die Algorithmen erkennen und beschreiben die Bildmerkmale. In dem Kapitel 2.5.1 werden die Mechanismen erläutert, mit denen die Bilder nach Bildmerkmalen untersucht werden.

Anschließend wird im Kapitel 0 beschrieben, wie die Bildmerkmale in sogenannten Deskriptoren beschrieben werden. Die Deskriptoren ermöglichen ein späteres Vergleichen und sind gegenüber verschiedenen Parametern, wie Belichtungsänderungen und Skalierung, robust.

In dem darauf folgenden Kapitel 2.5.3 wird dargestellt, wie die Deskriptoren verglichen werden. Der Vergleich wird dazu genutzt, um gleiche Punkte in Bildern zu finden und so Gegenstände auf den Bildern zu erkennen oder Bilderpaare zu finden.

Die Objekterkennungs-Verfahren „Scale Invariant Feature Transform“ (SIFT) und „Speeded Up Robust Features“ (SURF) werden in diesem Abschnitt ausführlich erläutert. Weitere Verfahren sind *Oriented FAST and Rotated BRIEF* (ORB) ([Rublee u.a. 2011]) und *Features from accelerated segment test* (FAST) ([Rosten2005]). Das ORB-Verfahren nutzt eine Variante des *Binary Robust Independent Elementary Features* (BRIEF)-Deskriptors ([Calonder 2010]).

2.5.1 KeyPoints berechnen (Feature Detection)

Dieser Abschnitt beschreibt die Ermittlung von KeyPoints[8], auch als Featurepoint bezeichnet, bei den Algorithmen SIFT und SURF. Die Vorgehensweise der Algorithmen wird erst erläutert und anschließend weiter ausgeführt.

[8] KeyPoint - Markante Punkte eines Bildes, die über die 2D-Position, Skalierung, Orientierung beschrieben werden.

2.5.1.1. SIFT

Das von Lowe entwickelte SIFT-Verfahren ([Lowe 2004]) dient der Ermittlung und Beschreibung von KeyPoints in einem Bild. Die KeyPoints beschreiben Merkmale eines Bildes, die mit Hilfe des SIFT-Verfahrens unabhängig von Skalierung[9] und Orientierung[10] einander zugeordnet werden.

SIFT basiert auf vier grundlegenden Schritten des Algorithmus:

1. Ermittlung von KeyPoints im Skalenraum des Eingangsbildes. Eine *Difference of Gaussian* (DoG) Pyramide wird erzeugt und nach Interessenpunkten durchsucht.
2. Lokalisierung und Filterung von ungeeigneten KeyPoints. Es wird die genaue Position des KeyPoints bestimmt und anhand des Kontrastes und möglichen Rauschens gefiltert.
3. Orientierung der KeyPoints bestimmen. Die Orientierung der verbleibenden Keypoints wird relativ zu ihrer Nachbarschaft bestimmt.
4. Deskriptor an der Orientierung ausrichten. Die Eigenschaften der Keypoints werden für späteres Vergleichen als Vektor gespeichert.

Ermittlung von KeyPoints im Skalenraum

Die KeyPoints werden im differentiellen Skalenraum eines Bildes ermittelt. Um den Skalenraum zu erstellen, wird das Bild I(x, y) mit einem Gaußfilter G(x, y, σ) gefaltet. Zunächst werden verschiedene Auflösungen des Bildes erstellt und übereinander gelagert, so dass eine pyramidenähnliche Struktur entsteht. Anschließend wird jede Ebene mehrfach mit dem Gaußfilter gefaltet. Die Differenz zwischen zwei Nachbarn einer Ebene bildet den differentiellen Skalenraum, auch *Difference of Gaussian* genannt.

[9] Skalierung beschreibt die Größenänderung eines Bildes
[10] Die Orientierung beschreibt Intensitätsverteilung der Umgebung als Ausrichtung des KeyPoints

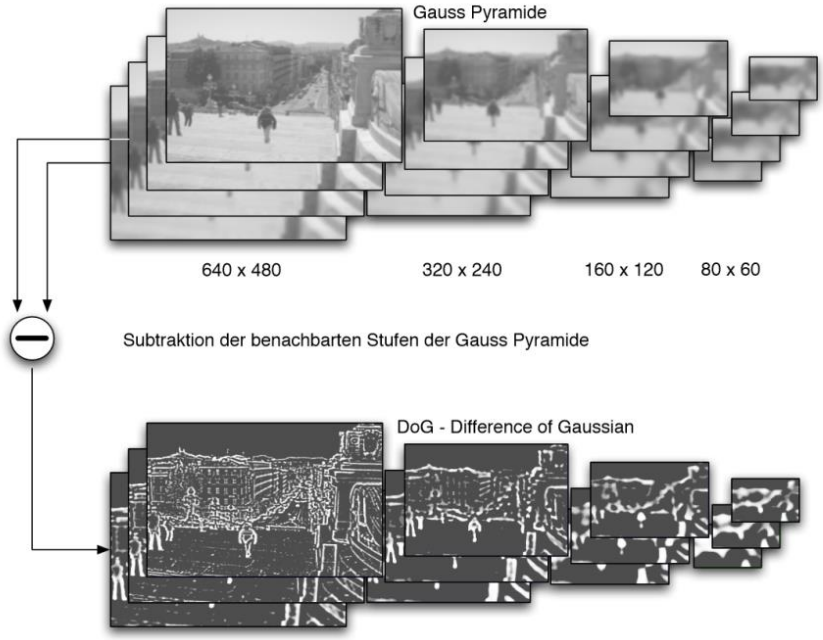

Abbildung 4: Gauss Pyramide und DoG Pyramide ([Heymann 2005])

Die lokalen Maxima und Minima des differentiellen Skalenraums werden als Kandidaten für KeyPoints betrachtet. Für die lokalen Maxima und Minima werden die umliegenden acht Punkte der gleichen und die neun umliegenden der vorherigen und der nächsten Ebene miteinander verglichen.

Filtern ungeeigneter KeyPoints

Die zuvor ermittelten Kandidaten für KeyPoints werden näher betrachtet, um ungeeignete Kandidaten herauszufiltern. Als Filter für ungeeignete Kandidaten können verschiedene Eigenschaften verwendet werden.

Der Kontrast des Extremums wird durch die Differenz zur Nachbarschaft betrachtet. Wird ein bestimmter Grenzwert unterschritten, wird der Kandidat als ungeeignet betrachtet. Hierdurch werden „Extrempunkte, die durch Bildrauschen entstanden sind und auch Punkte mit einer geringen Stabilität eliminiert“ (vgl: [Heymann 2005]). Eine weitere Verbesserung ergibt sich, wenn Kantenpixel[11] verworfen werden. Eine spätere Wiedererkennung könnte zu falschen Ergebnissen führen, da diese Punkte an einer

[11] Pixel, die sich entlang einer Kante mit großer Graustufenvariation auszeichnen..

beliebigen Stelle der Kante erkannt werden. Die Kantenpixel werden mit einer 2x2 Hesse Matrix[12] anhand der Position und Skalierung der KeyPoints bestimmt (vgl: [Lowe 2004]).

Orientierung der KeyPoints bestimmen

Die Orientierung der Keypoints – relativ zum Bild - wird durch seine Nachbarschaft bestimmt, um eine Rotations-Invarianz[13] zu ermöglichen. Lowe ermittelte nach Erfahrungen in der Praxis folgende stabile Methode:

Es wird ein Orientierungs-Histogramm[14] in der Region des KeyPoints berechnet. Das Histogramm ist in 36 mögliche Orientierungen unterteilt, um in 10er Schritten die 360° abzudecken. Die Teilwerte des Histogramms werden nach der Größe des Gradienten und der Entfernung vom Zentrum des KeyPoints gewichtet. Um die Richtung zu bestimmen, wird der stärkste Orientierungsgradient aus dem Histogramm gesucht.

Enthält ein KeyPoint weitere starke Gradienten, mit einer Länge von 80% zum stärksten Gradienten, wird ein weiterer KeyPoint erzeugt. Dies geschieht in etwa 15% aller Fälle und führt zu einer Stabilitätssteigerung (vgl: [Lowe 2004]).

2.5.1.2. SURF

Einige Jahre nach dem SIFT-Verfahren wurde das SURF–Verfahren von Herbert Bay ([Bay 2008][Bay 2008]) vorgestellt. Der auf dem SIFT-Verfahren basierende Algorithmus ermittelt lokale KeyPoints, anschließend werden die Deskriptoren zur Beschreibung geeigneter KeyPoints berechnet. Das SURF-Verfahren beschleunigt die Berechnung und Beschreibung der KeyPoints im Vergleich zum SIFT-Verfahren signifikant und erreicht diese durch die Verwendung von Mittelwert- anstelle der Gaußfilter. Mit der Verwendung von Integralbildern bleibt der Zeitaufwand konstant.

Zur Berechnung der Deskriptoren, werden erst die KeyPoints erkannt und anschließend beschrieben:

[12] Hesse Matrix - Bestimmte Art von Matrizen in der Form m × m
[13] Die KeyPoints können trotz einer Änderung in der Ausrichtung (Drehung) verglichen werden.
[14] Das Orientierungs-Histogramm entsteht aus der Orientierung der Gradienten

1. Integralbild berechnen
2. Hesse-Matrix Erkennung von KeyPoints
3. Skalenraum
4. Lokalisierung der KeyPoints
5. Beschreibung der KeyPoints

Integralbild berechnen

Ein Integralbild berechnet sich aus den Graustufenwerten des Eingangsbildes. Der Zeitaufwand zur Berechnung des Integralbildes ist linear, da jeder Bildpunkt nur einmal berechnet werden muss. Die Summe für einen Bildpunkt ergibt sich aus den Punkten der aktuellen Bildzeile und der Summe der vorangegangenen Bildzeilen, siehe Abbildung 5.

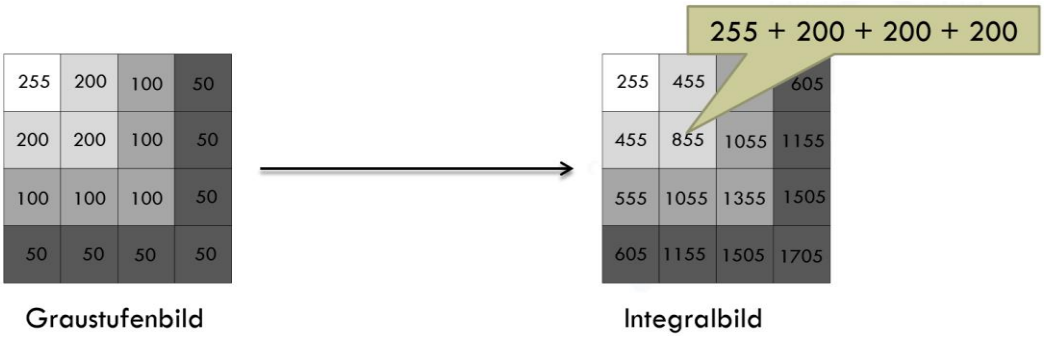

Abbildung 5: Berechnung des Integralbildes aus einem Graustufenbild ([Potsdam 2012], S. 7)

Der Zeitaufwand zur Verwendung von Integralbildern ist konstant, da sich die Pixelsumme einer beliebigen Fläche aus nur 4 Punkten des Bildes berechnen lässt.

Hesse-Matrix Erkennung

Die Erkennung der Keypoints basiert im SURF-Verfahren auf der Hesse-Matrix. Mit Hilfe der Hesse-Matrix lassen sich lokale Maxima, Minima und Sattelpunkte über die Determinante[15] bestimmen. Dementsprechend werden die approximierten Determinanten berechnet und auf lokale Minima und Maxima untersucht. Diese Punkte werden als Kandidaten für die KeyPoints betrachtet (vgl: [Fuss 2011]).

Skalenraum

[15] Rechenoperation auf m × m Matrizen, es entsteht eine reelle Zahl.

Die KeyPoints müssen invariant gegenüber der Skalierung sein, damit diese in verschiedenen Skalierungen erkannt werden können. Um diese Eigenschaft sicher zu stellen, werden die KeyPoints, wie bei SIFT, in unterschiedlichen Skalierungen gesucht. Ausgehend vom Ursprungsbild wurden im SIFT-Verfahren verschiedene geringere Auflösungen erstellt, so dass diese der Struktur einer Pyramide ähnelten und anschließend mit dem Gaußfilter gefaltet wurden. Im Gegensatz dazu werden im SURF-Verfahren nur die Filtermatrizen vergrößert. Durch dieses Vorgehen entstehen keine Aliasing-Effekte[16] und die Geschwindigkeit wird erhöht (vgl: [Fuss 2011]).

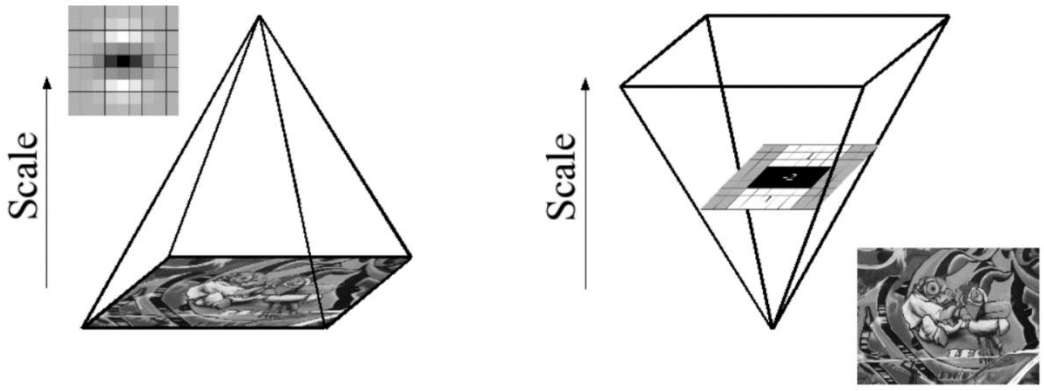

Abbildung 6: Statt der geringeren Auflösungen bei SIFT (links) werden bei SURF größere Filtermatrizen (rechts) verwendet ([Bay 2008], S. 4).

Lokalisierung der KeyPoints

Um die KeyPoints des Bildes in dem Skalenraum zu lokalisieren, wird in einer 3x3x3 Pixel-Umgebung gearbeitet. Es werden die angrenzenden Pixel des KeyPoint-Kandidaten in der gleichen, größeren und niedrigeren Skalierung untersucht. Die Gewichtung des Kandidaten muss maximal sein. Ist die Gewichtung eines Nachbarn größer, so entfällt der Kandidat als KeyPoint. Mit Hilfe von Schwellenwerten lässt sich die Anzahl der KeyPoints minimieren. Die niedrigste und die höchste Skalierung dienen nur dem Erstellen und Vergleichen der Umgebungen (vgl: [Huber u.a. 2013]).

[16] Aliasing-Effekte erzeugen ungewünschte Bildartefakte bzw. Pixeleffekte.

2.5.2 Deskriptor

In diesem Abschnitt wird die Beschreibung der in Kapitel 2.5.1 ermittelten KeyPoints durch Deskriptoren ausgeführt. Die Robustheit der Deskriptoren gegenüber verschiedenen Parametern wird aufgeführt.

Die Deskriptoren sind Vektoren, um die Region eines KeyPoints eindeutig zu beschreiben. Mit den Deskriptoren lassen sich die KeyPoints wiedererkennen. Deskriptoren sind gegenüber Rotation und Skalierung invariant.

2.5.2.1. SIFT

In den vorherigen Schritten des SIFT-Verfahrens ([Lowe 2004]) wurden die Lokalisierung, Skalierung und Orientierung der KeyPoints bestimmt und die Invarianz den Parametern gegenüber sichergestellt. Im nächsten Schritt wird der Deskriptor für die KeyPoints berechnet. Die Deskriptoren sind sehr markant und so invariant wie möglich gegenüber Blickwinkel und Belichtung.

Die bereits ermittelten Gradienten und ihre Länge werden zur Berechnung der Deskriptoren verwendet. Der Gradientenbereich wird in 4x4 Bereiche eingeteilt, aus denen jeweils ein Gradientenhistogramm mit acht Richtungen berechnet wird. Der SIFT-Deskriptor besteht somit aus 128 Elementen.

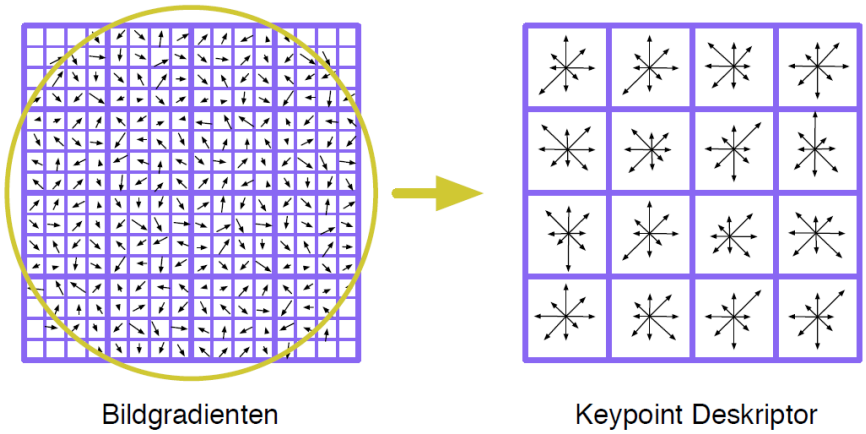

Abbildung 7: Gradientenbereich in 4x4 Bereiche mit jeweils einem Gradientenhistogramm (vgl: [Heymann 2005], S. 27)

Die Berechnungen werden anhand der Daten aus Orientierung und Größe durchgeführt und erhalten weiterhin die Rotationsinvarianz. Eine Invarianz gegenüber Belichtungsänderungen wird erreicht, indem die berechneten Vektoren normalisiert werden (vgl: [Heymann 2005]).

2.5.2.2. SURF

Bestimmung der Orientierung

Die Orientierung eines Punktes wird bestimmt, um eine Rotationsinvarianz zu erreichen. Zunächst werden die "*Haar-Wavelet Resonanzen*" der Umgebung in horizontaler und vertikaler Richtung bestimmt. Anschließend werden die Resonanzen mit Hilfe der Gauß-Funktion um den KeyPoint gewichtet. Bei der Bestimmung der Orientierung erhalten somit die Punkte, die näher an dem KeyPoint liegen, eine höhere Gewichtung. Anhand der Resonanzen wird entsprechend der Gewichtung für jeden Punkt ein Vektor erzeugt. Diese Vektoren werden in ein Koordinatensystem übertragen und vom Ursprung ausgehend in $\frac{1}{3}\pi\, rad$ -Schritten zu Gesamtvektoren zusammengefasst. Der Längste der Vektoren bestimmt die Orientierung des Keypoints (vgl: [Bay 2008]).

Ermittlung des Vektors

Basierend auf der Orientierung wird der Vektor des Deskriptors berechnet ([Bay 2008] und [Fuss 2011]). Hierzu wird über dem KeyPoint ein Quadrat, an der Orientierung ausgerichtet, der Größe 20s[17] gelegt. Das Quadrat wird in 4x4 Regionen aufgeteilt, in denen jeweils für 25 gleichmäßig verteilte Punkte die Haar-Wavelet-Resonanzen berechnet werden. Um die Robustheit gegenüber Bildrauschen zu erhöhen, werden diese zunächst mit einem Gaußfilter geglättet. Um den Vektor zu berechnen, werden in den 4x4-Regionen die Werte der jeweils 25 Punkte summiert. Die Vektoren setzen sich aus den Werten der 4x4-Regionen zusammen (vgl: und [Bay 2008]).

2.5.3 Matching

Unter dem Begriff des Matchings ([Fuss 2011]) wird die Suche nach Paaren, zwei gleichen Punkten, aus zwei verschiedenen Bildern verstanden. Nach dem Ermitteln geeigneter KeyPoints und deren Deskriptoren lassen sich die Punkte zweier Bilder vergleichen. Mit dem Ziel, Paare gleicher Featurepoints zu finden, wird versucht, zu jedem Featurepoint des Bildes

[17] Das „s“ steht für die Skalierung des Keypoints.

B_1 ein bestes Match aus dem Bild B_2 zu finden. Von der gewählten *matching-strategy* hängt ab, ob ein Match akzeptiert wird. Je geringer die Distanz zwischen zwei Deskriptoren ist, desto ähnlicher sind sich die Featurepoints. Im Folgenden wird eine Auswahl der Strategien dargestellt:

1. Euklidische Distanz
2. *Nearest Neighbor Matching*
3. *Nearest Neighbor Matching* mit nächsten Nachbarn
4. *Structure of distance*
5. Hamming-Distanz

Zur Berechnung des euklidischen Abstands gilt für zwei n-dimensionale Vektoren x und x' ([Brauer 2010], S. 38):

$$d(x_1, x_2) = \left||x - x'|\right|_2 = \sqrt{x_1 - x'_1 + \cdots + x_n - x'_n}$$

Nach der ersten Methode wird die euklidische Distanz zwischen zwei Deskriptoren D_A und D_B gebildet und als Match akzeptiert, sobald die Differenz einen Schwellenwert unterschreitet. Hierbei können mehrere Möglichkeiten zu einem Match führen.

Für die zweite Methode, dem *Nearest Neighbor Matching*, wird der Abstand zwischen D_A und einer Menge von Deskriptoren berechnet und verglichen. Ein Match wird akzeptiert, sobald die Distanz zwischen den Deskriptoren D_A und D_B am kleinsten ist und unter einem Schwellenwert liegt. Es kann nur ein einzelner Deskriptor als Match akzeptiert werden.

Ein weiterer Ansatz lehnt sich dem zweiten an. Es wird nicht nur der nächste Nachbar gesucht und betrachtet, sondern auch der zweitnächste Deskriptor D_C. Für diese Methode muss zusätzlich der Abstand zwischen den Deskriptoren D_A und D_C deutlich größer sein als zwischen D_A und D_B (vgl: [Fuss 2011]).

$$\frac{\left||D_A - D_B|\right|}{\left||D_A - D_C|\right|} < distRatio$$

Oder auch

$$distRatio * \left||D_A - D_B|\right| < \left||D_A - D_C|\right|$$

Bei falschen Matches kommt es oft vor, dass der Abstand zu vielen Nachbarn ähnlich ist. Ist der Abstand zwischen den beiden nächsten Nachbarn jedoch nicht groß genug, wird das

Ergebnis verworfen. Mit einem hohen Schwellenwert sinkt die Anzahl der Matches, viele falsche Matches werden hierbei eliminiert. Lowe empfiehlt einen distRatio-Wert von 0,8. Dieser soll 90% der falschen Matches eliminieren, wobei etwa 5% der richtigen Matches entfallen (vgl: [Lowe 2004]).

Die bisher aufgeführten Strategien sind Arten der linearen Suche, die mit ihrer hohen Komplexität von $O(n^2)$ eine schlechte Performance bieten und für größere Datensätze ungeeignet sind. Eine geeignete Alternative stellen kd-Bäume dar, die im Abschnitt 3.2.1.2 näher beschrieben werden.

Ein weiteres Verfahren verwendet eine gewichtete Abstimmungsstrategie (Weighted Voting Strategy), inspiriert durch den PageRank-Algorithmus ([Henzinger 2008]). Übliche Matching-Strategien vernachlässigen die Position eines Featurepoints in Beziehung zu anderen des gleichen Bildes. Dies kann leicht zu Fehl-Matches führen, so dass Featurepoints gegenüber verschiedenen Objekten matchen.

Die Featurepoints aus zwei Bildern werden miteinander verglichen, indem die sogenannte Strukturelle Distanz[18] ([Hu 2015]) zwischen diesen ermittelt wird. Die Strukturelle Distanz zwischen zwei Featurepoints errechnet sich aus der Summe folgender Parameter:

- Euklidische Distanz zwischen den zwei Featurepoints
- Gewichtete Summe der optimierten Strukturellen Distanz der anderen Featurepoints des ersten Bildes, wo die optimierte Strukturelle Distanz eines jeden Punktes, aus dem ersten Bild, der Distanz zu dem optimalen Matchingpunkt, aus dem zweiten Bild, entspricht.
- Gewichtete Summe der optimierten Strukturellen Distanz der anderen Featurepoints des zweiten Bildes, wo die optimierte Strukturelle Distanz eines jeden Punktes, aus dem zweiten Bild, der Distanz zu dem optimalen Matchingpunkt, aus dem erstem Bild, entspricht.

Mit jeder Iteration wird die Strukturelle Distanz zwischen allen Featurepoints aus Bild 1 und allen aus Bild 2 aktualisiert. Die Optimierung wird beendet, sobald die vordefinierte Anzahl an Iterationen erreicht wurde oder sich keine weiteren Änderungen ergeben.

[18] Strukturelle Distanz ist ein Maß, um die Ähnlichkeit zwischen zwei Strukturen zu bestimmen.

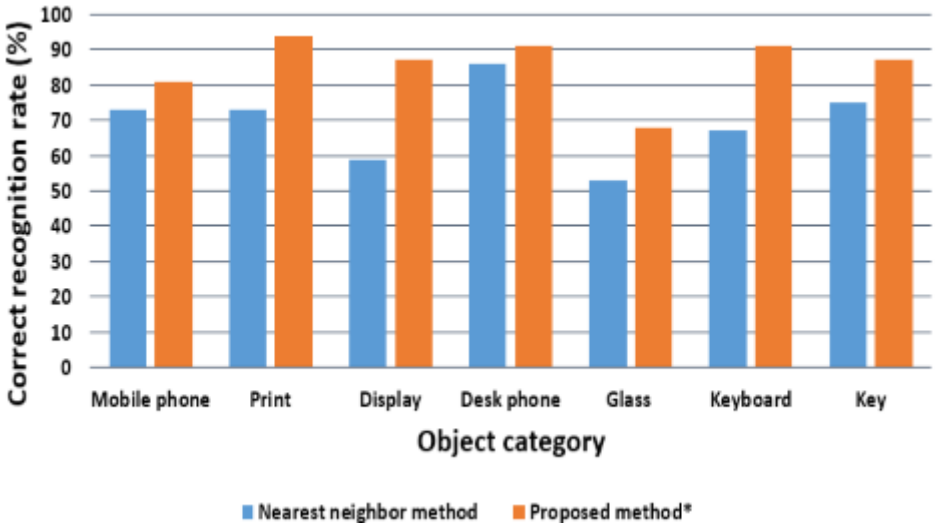

Abbildung 8: Anteil der korrekten Matches bei *Nearest Neigbor Matching* und *Weighted Voting Strategy* ([Hu 2015])

Die Abbildung 8 zeigt in den durchgeführten Experimenten eine Verbesserung des Anteils der korrekten Matches gegenüber dem oben aufgeführten *nearest neighbor matching* (vgl: [Hu 2015]).

Eine weitere Variante für das Matching von Deskriptoren ist die Hamming-Distanz, die beispielsweise bei dem Matching des ORB-Deskriptors verwendet wird. Bei der Hamming-Distanz werden die Bitstrings der Deskriptoren miteinander verglichen (vgl: [Batzill 2013]).

2.6 Lokalisation

Für den Menschen ist die Fragestellung nach dem Ort, an dem er sich befindet, meist leicht zu beantworten. Der Mensch findet sich in seiner Umgebung problemlos zurecht und kann seine Position im Raum bestimmen. Gelingt dies nicht, so kann er Hilfsmittel wie Hinweisschilder oder Karten nutzen.

Für die Robotik eröffnet sich in dieser Fragestellung das Themenfeld der Lokalisation. In der Lokalisation werden verschiedene Ansätze zur Ortsbestimmung behandelt. Die Auswahl des geeigneten Ansatzes kann anhand verschiedener Unterscheidungen getroffen werden. Handelt es sich um eine Indoor- oder Outdoor-Lokalisation? Soll die absolute oder relative Position bestimmt werden? Von Umgebung und Kontext abhängig kann die Position auf das Weltmodell oder den Raum, in dem sich der Roboter befindet, bezogen werden.

Im Folgenden werden zwei Ansätze der Lokalisation beschrieben. Dazu gehören die Lokalisation über Landmarken und das Indoor-Lokalisation System Ubisense, welches in dem *Living Place* der HAW-Hamburg eingesetzt wird (Lokalistion vgl: [Elias 2006], S. 8-12 und [Rupp 2001], S. 85-87).

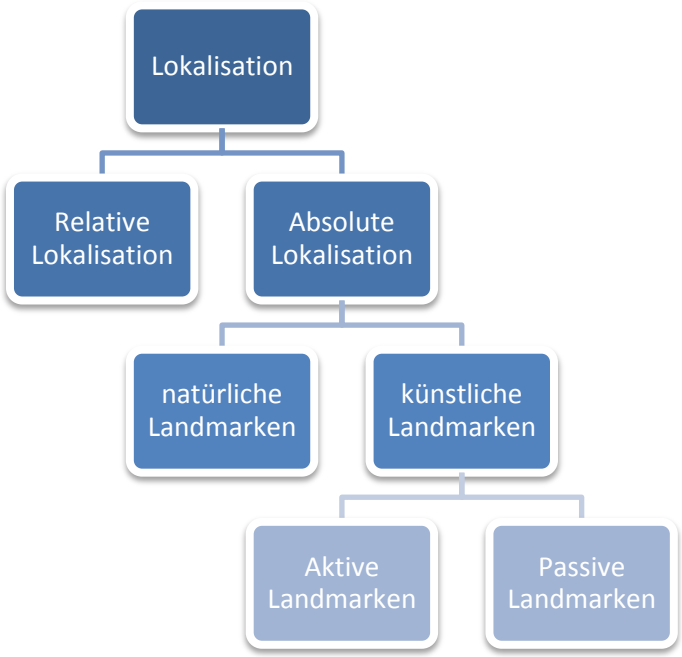

Abbildung 9: Übersicht der Unterteilung von Lokalisation und den Landmarken

2.6.1 Relative Lokalisation

Wird die Lage in Relation zu einem zuvor bekannten Punkt bestimmt, spricht man von relativer Lokalisation. Diese Änderung lässt sich gut mit einem Vektor darstellen. Zur Berechnung der Lageänderung lassen sich verschiedene Größen nutzen. Bei einem fahrenden Roboter lässt sich beispielsweise die zurückgelegte Strecke mit Odometrie[19] messen. Über die zurückgelegte Strecke und Richtung lässt sich die neue Position des Roboters berechnen. Wird die relative Lokalisation mehrfach angewendet, so wird in der Literatur von „Tracking“ gesprochen. Sobald die Position über einen längeren Zeitraum bestimmt werden soll, stößt die relative Lokalisation schnell an ihre Grenzen, denn kleinere

[19] Messung der Radrotation

Fehler der Berechnungsschritte summieren sich zu großen Ungenauigkeiten auf (vgl: [Rupp 2001], S. 85-87).

2.6.2 Absolute Lokalisation

Die Positionsbestimmung erfolgt aus der Beobachtung der Umgebung. Vorausgesetzt wird das Wissen über markante Fixpunkte, sogenannte Landmarken, deren Position im Raum bekannt ist. Nach diesen Punkten wird in der Umgebung gesucht. Die eigene Lage wird im Verhältnis zu den gefundenen Landmarken bestimmt. Aus der eigenen Lage und dem Wissen der Landmarkenpositionen im Raum lässt sich die eigene absolute Position bestimmen. Wird die Position ohne Berücksichtigung der vorangegangen Position neu bestimmt, wird von absoluter Lokalisation gesprochen ((vgl: [Rupp 2001], S. 87).

2.6.2.1. Landmarken

Als Landmarken werden markante und für Sensoren gut sichtbare bzw. messbare Objekte bezeichnet. Als Voraussetzung zur Lokalisation müssen die Positionen der Landmarken, Freiflächen und Hindernisse in einem Koordinatensystem bekannt sein. Zur Positionsbestimmung werden der Abstand und die Richtungswinkel zu den Landmarken gemessen. Mit den Messdaten und der Position der dazugehörigen Landmarken wird die eigene absolute Position in einem passenden Koordinatensystem bestimmt.

Zur Erfassung von Landmarken kommen verschiedenste Sensoren in Betracht. Unterschiedliche Eigenschaften der Landmarken können erfasst werden, darunter fallen unter anderem Abstand, Größe, Farbe, Schall und Richtung. Diese lassen sich mit Sensoren wie Radioempfänger, Kamerasystemen, Lasersystemen und Ultraschallsensoren erfassen.

Zur Positionsbestimmung mit Landmarken sind Triangulation[20] und Trilateration[21] weit verbreitete Verfahren. Bei der Triangulation wird die Position über eine Winkelmessung mit mindestens drei bekannten Landmarken bestimmt. Bei der Trilateration hingegen wird der Abstand zu mindestens drei bekannten Landmarken gemessen. Das in 1.4.3 beschriebene NAVSTAR GPS berechnet die Position mittels Trilateration (vgl. [Jotzo 2002], S. 28).

In der Gruppe der Landmarken wird eine Unterteilung in künstliche und natürliche Landmarken vollzogen. Natürliche Landmarken können zur Navigation in unbekanntem

[20] Triangolation – Methode zur Abstandsmessung durch Winkelmessung innerhalb von Dreiecken
[21] Trilateration – Methode zur Positionsbestimmung durch Abstandsbestimmung zu drei Punkten

Gelände verwendet werden. Dagegen müssen künstliche Landmarken erst bereitgestellt werden. Daher muss das Gelände zuvor erkundet bzw. betreten werden.

Natürliche Landmarken

Objekte, die sich als Landmarken zur Lokalisation anbieten und nicht zu diesem Zweck künstlich geschaffen wurden, werden natürliche Landmarken genannt. Die Landmarken befinden sich bereits in der Umgebung und erfüllen die entsprechenden Anforderungen. In der Umgebung finden sich leicht erkennbare Strukturen wie Türen, Wände, Säulen oder Bäume. Diese weisen meist klare Linien, Flächen oder weitere geometrische Elemente auf. Diese Landmarken können in der Praxis verschiedene Nachteile aufweisen, denn natürliche Landmarken können leicht verdeckt sein oder nicht in ausreichender Anzahl bzw. Nähe vorhanden sein.

Künstliche Landmarken

Künstliche Landmarken sind Objekte, die zum Zweck der Lokalisation künstlich im Raum platziert oder geschaffen wurden. Die Objekte erfüllen die Anforderungen an Landmarken. Oft werden die Landmarken durch Reflektoren, künstliche Magnetfelder oder Barcodes erzeugt (vgl: [Braun 2005], S. 8-9). Abhängig von der Umgebung und den vorhandenen Sensoren des Anwenders kommen verschiedene Arten der Landmarken zum Einsatz. Vor der Lokalisation werden die Landmarken in die Umgebung eingefügt. Damit schließt sich eine Navigation in unbekanntem Terrain aus. Die Art der verwendeten künstlichen Landmarken wird durch passive und aktive Landmarken unterschieden.

Passive Landmarken

Unter passiven Landmarken werden leicht erkennbare geometrische Formen, markante Farben, Barcodes etc. verstanden. Bei passiven Landmarken findet keine Kommunikation mit dem Anwender statt. Es werden keine Signale in die Umwelt ausgesandt, die Erkennung erfolgt meist über visuelle Verfahren des Anwenders.

Aktive Landmarken

Aktive Landmarken senden ständig Signale in die Umgebung aus. Der Empfänger empfängt die Signale und kann mit den gesendeten Informationen den eigenen Standort ermitteln. Für aktive Landmarken kommen verschiedene Signale in Frage, darunter fallen Signale wie Licht, Ultraschall oder Radiowellen.

2.6.3 Lokalisation – Praxisbeispiele

Im Folgenden werden zwei Beispiele für die Lokalisation vorgestellt. NAVSTAR GPS ist ein Beispiel für die Outdoor Lokalisation und Ubisense ein Beispiel für die Indoor Lokalisation.

2.6.3.1. Outdoor Lokalisation – NAVSTAR GPS

Das *Navigational Satellite Timing and Ranging – Global Positioning System* (NAVSTAR GPS), umgangssprachlich auch GPS genannt, stellt eine allgemein bekannte Anwendung für die Ortung mittels aktiver Landmarken dar. Die Satelliten senden ständig ihre Position und die genaue Uhrzeit. Aus den Signallaufzeiten lässt sich die Entfernung zum Satelliten berechnen. Zur Lokalisation werden in der Theorie die Signale von mindestens drei Satelliten benötigt. In der Praxis werden vier Satelliten benötigt, da die Genauigkeit der Empfängeruhren sonst nicht ausreicht. Der vierte Satellit dient der genauen Zeitbestimmung. Im Outdoor-Bereich wird GPS häufig zur Standortbestimmung verwendet. Aus militärischen Gründen wurde die Genauigkeit allerdings lange künstlich verringert. Wo im Outdoor-Bereich GPS eine praktikable Lösung darstellt, scheitert es im Indoor-Bereich (vgl. [Strobel 1995] und [Jotzo 2002], S. 28).

2.6.3.2. Indoor Lokalisation - Ubisense

Die Lokalisation im *Living Place Hamburg*[22], einem Smart Home auf dem Campus der HAW Hamburg, findet mit dem Ubisense-System statt. Ubisense ist ein Indoor-Lokalisations-System, welches aktive Landmarken verwendet. Die Landmarken senden Radiowellen im Ultrabreitband-Bereich aus. Ultrabreitband wird verwendet, da die Landmarken auch verdeckt sein dürfen und sich das Signal leicht herausfiltern lässt.

[22] http://livingplace.informatik.haw-hamburg.de/blog/

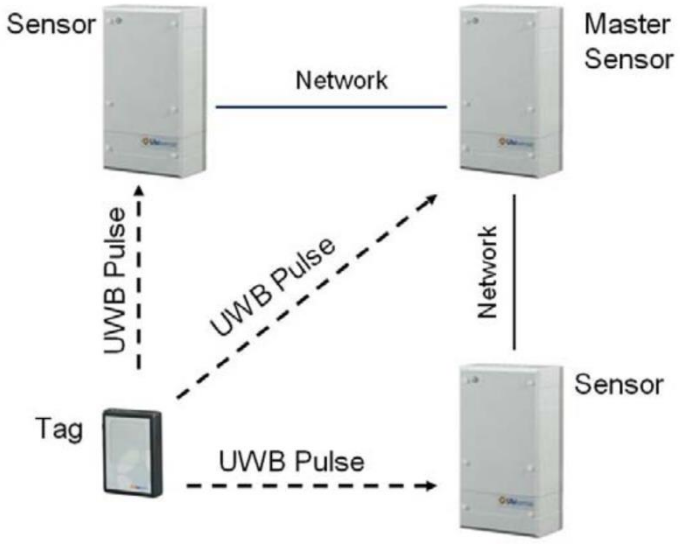

Abbildung 10: Hardware Komponenten von Ubisense ([Steggles und Gschwind 2005], S. 74)

Die Sensoren des Ubisense-Netzwerks befinden sich an festen Positionen des Raumes. Die Positionen der Sensoren sind dem System bekannt. Jeder Sensor hat einen RF-Transceiver und einen Phased-Array[23] Ultrabreitband Empfänger.

Sogenannte Ubitags befinden sich an Objekten oder werden von den Anwendern getragen. Wie die Sensoren enthalten die Ubitags einen RF-Transceiver und einen Ultrabreitband-Sender. Aktive Ubitags senden ihre Identität als RF-Nachricht und eine Ultrabreitband Sequenz, mit der die Lage des Ubitags bestimmt wird. Zur Berechnung der Lage werden an den Sensoren die Zeitdifferenz und die ankommenden Winkel der Signale bestimmt (vgl: [Steggles und Gschwind 2005]).

2.7 Vergleichbare Arbeiten

In dem Bereich der Augmented Reality kommen bereits verschiedene Anwendungen zum Einsatz, die in manchen Punkten vergleichbar mit der hier vorgestellten Arbeit sind. In dem Abschnitt 2.7.1 wird ein weiterer Ansatz für die Aufgabenstellung dieser Arbeit betrachtet. Darauf folgend werden Anwendungen, die in unterschiedlichen Bereichen zum Einsatz kommen, vorgestellt.

[23] In Gruppen angeordnete Antenne - https://de.wikipedia.org/wiki/Phased-Array-Antenne

2.7.1 Vergleichbarer Ansatz

Zu der Problemstellung mit digitalen Notizen können andere Ansätze verfolgt werden. In einem weiteren Ansatz werden die Gegenstände, die mit Notizen verknüpft werden können, ebenfalls als Landmarken betrachtet. Die Positionen der Landmarken (siehe 2.6.2.1) sind im System bekannt. Die genaue Position des Nutzers ist unbekannt und muss in diesem Ansatz genau ermittelt werden. Die Lage der Gegenstände wird relativ zu der Position des Nutzers berechnet. Die Lokalisation muss genau erfolgen, da die Gegenstände anhand der Lagebestimmung auf dem Ausgabegerät angezeigt werden. Im Gegensatz zu dem von mir verfolgten Ansatz wird in diesem auf kamerabasierte Verfahren verzichtet. Der HAW-Student Sebastian Rudolf hat diesen Ansatz im Jahr 2011 näher betrachtet und einem ersten Test unterworfen ([Rudolf 2011]). Im Rahmen eines Masterprojekts stellte er den Lösungsansatz näher da und kam zu einem zufriedenstellenden Ergebnis.

Als Ausgabegerät verwendet Rudolf ein Smartphone. Die Lokalisation erfolgt über das im *Living Place Hamburg* integrierten Ubisense. Zusätzlich wird eine 3D-Modell der Umgebung verwendet, um eine mögliche Verdeckung der Gegenstände zu erkennen .

Die digitalen Notizen werden von Rudolf als *Shared Notes* bezeichnet, also Notizen, die von mehreren Nutzern gemeinsam genutzt werden, aber auch als private Notizen verwendet werden können.

Als Schnittstelle zwischen Nutzer und System fungiert ein Smartphone, dessen Position mit Hilfe der Sensoren lokalisiert und die Ausrichtung des Smartphones mit der integrierten *Intertial Measurement Unit*[24] (IMU) ermittelt wird. Ausgehend von der Position wird die Blickrichtung (die Ausrichtung des Smartphones) in einem 3D-Modell betrachtet. Befinden sich Objekte des 3D-Modells in dem Sichtbereich, kann der Nutzer diese auswählen und Notizen hinzufügen. Diese Notizen können ebenfalls manipuliert werden. Die *Shared Notes* im Sichtbereich werden ebenfalls über das 3D-Modell ermittelt und anschließend auf dem Smartphone dargestellt.

An die Umgebung werden besondere Anforderungen gestellt, die die Einsatzmöglichkeiten der Anwendung stark einschränken. So muss die Position der Objekte immer korrekt im System hinterlegt sein oder zu jeder Zeit aktualisiert werden können. Bei der Lokalisation mit *Ubisense* müssen die Gegenstände dementsprechend mit *Ubitags* ausgestattet sein oder andere Lokalisationsverfahren angewandt werden. Aus der Verwendung

[24] Mit Kompass und 3-Achsen-Gyrosensor kann die IMU die Orientierung im Raum ermitteln.

sensorbasierter Tracking-Verfahren ergibt sich eine Einschränkung in der Nutzung, denn heutzutage werden diese nur in wenigen Umgebungen angewendet. Eine weitere Problematik ergibt sich daraus, dass ein 3D-Modell vorhanden sein muss. Sollte sich die Gestaltung des Raumes ändern, muss dieses ständig aktualisiert werden.

Der Ansatz hat den Vorteil, dass verschiedene Problematiken der optischen Verfahren keine Auswirkungen auf das System haben. So arbeitet es auch unter verschiedenen Bedingungen wie Verdeckung oder Dunkelheit zuverlässig.

2.7.2 Augmented Reality Shopping Apps

Mit der „Augmented Reality Mobile Shopping App“[25] von IBM soll das Einkaufserlebnis in einem Laden verbessert werden. Die App kann Gegenstände in den Regalen erkennen und mit Informationen aus einer Datenbank verknüpfen, die dem User auf dem Smartphone angezeigt werden. Des Weiteren ist es möglich, vor dem Einkauf eine Produktauswahl zu treffen oder ein gewünschtes Sortiment nach bestimmten Parametern herauszufiltern. Diese werden mit Hilfe des Kamerabildes markiert und auf dem Smartphone dargestellt.

2.7.3 Interaktive (Print-)Werbung

Die Firma „Lego“ hat in einigen Geschäften Bildschirme installiert, auf denen der Bereich vor dem Bildschirm angezeigt wird. Wird die Verpackung eines Spielzeugs in den Kamerabereich gehalten, wird das Spielzeug als 3D-Modell auf dem Bildschirm oberhalb der Verpackung gezeigt. Die Anwendung für die „Lego Digital Box“ wurde von Metaio[26] entwickelt, (vgl: [Gruendel]).

Ein ähnliches Prinzip macht sich IKEA mit ihren Katalogen zunutze. Auf der Smartphone/Tablett-App „IKEA Catalog App“ ([Metaio]) wird ein gewünschter Einrichtungsgegenstand ausgewählt. Anschließend wird ein Katalog im Wohnraum platziert und gefilmt. An der entsprechenden Stelle des Katalogs wird auf dem Smartphone der gewünschte Gegenstand projiziert. So lässt sich ermessen, ob der Gegenstand zur vorhandenen Einrichtung passt.

[25] http://ibmsmartershopping.tumblr.com/
[26] https://www.metaio.com/

2.7.4 Google Goggles

Die Firma „*Google Inc.*“ entwickelte mit „*Goggles*“ eine Bilderkennungssoftware, die Sehenswürdigkeiten, Gemälde, etc. in Bildern erkennt. Um die Objekte erkennen und zuordnen zu können, wurden Bilder von *Picasa*[27] und *Panoramio*[28], Webseiten auf denen Fotos geteilt werden, gesammelt. Aus diesen Bildern und Bildern der Google Bildersuche[29] erstellten die Entwickler einen großen Datensatz. Die Bilder wurden anhand der Annahme gefiltert, dass Bilder der Landmarken visuelle Ähnlichkeiten aufweisen. Die Suche nach den Ähnlichkeiten erfolgte mit dem SIFT-Algorithmus, wie in 2.5.1.1 und 2.5.2.1 beschrieben.

In diesem Datensatz mit 5.312 (Stand 2009) Objekten sucht Goggles nach übereinstimmenden Objekten. Die Objekte werden als Landmarken angesehen, deren Position bei der Zusammenstellung der Datensätze berücksichtigt wurde. Mit Hilfe der Lokalisation kann der „Suchraum“ eingeschränkt werden. Wird ein Bild beispielsweise in Paris aufgenommen, muss die Software den Datensatz nicht nach allen Landmarken durchsuchen, sondern kann sich auf die Umgebung beschränken. Für die gefundenen Objekte werden Informationen aus der Datenbank verknüpft und dem Nutzer mit weitergehenden Informationen, sowie per Weiterleitung auf Webseiten, angeboten (vgl. [Zheng 2009]).

2.7.5 Digitale Betriebsanleitung

Eine weitere Idee ist es, die üblichen Betriebsanleitungen zu digitalisieren. Hierbei macht sich z.B. Audi mit der „Audi eKurzinfo“[30] die Möglichkeiten von Smartphones zunutze. Elemente des Fahrzeugs werden übers Smartphone erkannt und die Anleitung passend eingeblendet. So kann sich der Nutzer das Armaturenbrett des Autos erklären lassen oder das Smartphone zur Suche unter der Motorhaube nutzen. Der Ölwechsel und das Auffinden des Kühlwasserbehälters werden so erleichtert.

[27] http://picasa.google.com/
[28] http://panoramio.com
[29] http://images.google.com
[30] http://www.audi.de/de/brand/de/kundenbereich/apps/pool/audi-ekurzinfo.html

2.8 Fazit

In den vorherigen Abschnitten des Kapitels wurden die Anforderungen aus dem Szenario extrahiert. Die Lokalisation, Objekterkennung, Darstellung und Abfrage von Informationen sind einige der Anforderungen, die sich aus dem Szenario für den Systementwurf ergeben. Die jeweiligen Hintergründe wurden in den Kapiteln „Interaktionen und Gesten“ 2.4, „Visuelle Objekterkennungs-Verfahren“ 2.5 und Lokalisation 2.6 ausgeführt. Neben den Interaktionsmöglichkeiten oder Gesten wurden die Lokalisation und die visuellen Objekterkennungs-Verfahren wie SIFT und SURF vorgestellt.

Im Rahmen dieser Arbeit wird der Fokus in Kapitel 0 „Design und Realisierung“ auf die Anforderung 2.3.6, das Erkennen von Objekten in Bildern, gelegt. Für die Umsetzung eines Systems für digitale Notizen über kamerabasierte Verfahren ist die Objekterkennung elementar.

Die Anforderungen 2.3.11 mit den Gesten und die Lokalisation, in 2.3.1 und 2.3.5 beschrieben, sind nicht Gegenstand dieser Arbeit. In dem beschriebenen Szenario und den Anforderungen wirkt die Lokalisation vor allem unterstützend. Die Lokalisation ist ein komplexes Thema, das hier genutzt wird, um die Suche auf Objekte einzuschränken, die sich in dem gleichen Raum wie der Anwender befinden können. Dies kann das System beschleunigen und gegebenenfalls zu einer Verbesserung der Suche führen.

Gesten und die Sprachsteuerung dienen zur Interaktion mit dem System. So kommt die Interaktion bei der Selektion zwischen mehreren gefundenen Objekten oder um unterschiedliche Aktionen, wie die Manipulation von Informationen, zu initiieren zum Einsatz.

In weiteren Arbeiten kann untersucht werden, welche Formen der Interaktionen für den Menschen intuitiv ausführbar sind und sich für die Verwendung in einer *Augmented Reality* eignen. Neben der Interaktion bietet die Lokalisation ein Themenfeld für weitergehende Untersuchungen.

3 Design und Realisierung

In diesem Kapitel werden das Design und die Realisierung des Systems beschrieben. In dem Abschnitt 3.1 wird zunächst auf die Architektur der Software eingegangen. Die Architektur wird mit dem Schichtenmodell und der Bausteinsicht detailliert beschrieben. Anschließend werden die Abläufe durch die Laufzeitsicht dargestellt. Das Design bildet die Grundlage für die Realisierung der Software.

Die Realisierung wird in dem Kapitel 3.2 beschrieben. Hier wird das Design mit dem Fokus auf die Objekterkennung umgesetzt. Dazu werden externe Software und Frameworks aufgeführt, die in der Realisierung des Designs zum Tragen kommen.

In dem Abschnitt 3.3 wird die Realisierung anhand der Anforderungen an der Objekterkennung und den zeitlichen Bedingungen evaluiert.

3.1 Software-Architektur

Die Rechenleistung von Smartglasses und Smartphones hat in den letzten Jahren erhebliche Fortschritte gemacht. Mit rechenintensiven Berechnungen, wie sie für die Objekterkennung nötig sind, stoßen diese jedoch noch immer an ihre Grenzen. Aus diesem Grund wird die Logik der Software über das Netzwerk ausgelagert. Der Client wird möglichst einfach gehalten und die Logik auf dem Server ausgeführt, es wird auch von einem *Thin-Client* gesprochen.

Das System befindet sich in einer Smart Home Umgebung und stellt somit die Möglichkeit bereit, einen Server zu betreiben, auf den die Logik der Anwendung ausgelagert werden kann. In dem verteilten System ist die Smartglass dafür zuständig, das Kamerabild aufzunehmen und dieses über das Netzwerk an den Server zu übermitteln. Der Server berechnet die Bildausgabe, die dem Anwender über das *Head-Mounted Display* der Brille dargestellt wird.

Auf dem Server wird das Kamerabild der Smartglass empfangen sowie die Verarbeitung und Berechnung der Bildausgabe auf dem Server koordiniert. Von dem Server aus werden

Anfragen an die Datenbank gestellt und die Position von der Lokalisierungs-Komponente abgefragt. Auf dem Server werden die Objekterkennung und die Bildausgabe, unter Einbeziehung der abgefragten Daten, berechnet. Die Bildausgabe wird von dem Server an die Smartglass gesendet.

Die Datenbank verwaltet die gespeicherten Daten des Systems, zu denen die digitalen Notizen und die Bildmerkmale gehören.

Wie nachfolgend in Abschnitt 0 beschrieben, kann die Lokalisierung ausgelagert und als externes System des *Smart Homes* verwendet werden.

In dem Schichtenmodell 3.1.1, der Bausteinsicht 3.1.2 und der Laufzeitsicht 3.1.3 wird die Architektur des Systems spezifiziert. In der Laufzeitsicht wird das *Request-Response*-Modell des Systems verdeutlicht. In dem Abschnitt Message Broker 3.1.4 wird die Kommunikation der einzelnen Komponenten des Systems über einen Message Broker besprochen. Außerdem werden in Lokalisierung 0 und Datenformate 3.1.6 weitere Details des Designs erläutert.

3.1.1 Schichtenmodell

Die Software, die im Rahmen dieser Bachelorarbeit entworfen wurde, ist ein verteiltes System, das sich in drei Schichten aufteilen lässt. Diese bestehen aus der Präsentations-, der Logik- und der Datenhaltungsschicht, welche in der Abbildung 11 dargestellt sind. Mit dem Aufteilen der Software in mehrere Schichten ergibt sich eine Unabhängigkeit der einzelnen Schichten voneinander. Die Schichten sind logisch voneinander getrennt und werden in diesem Kapitel weiter ausgeführt.

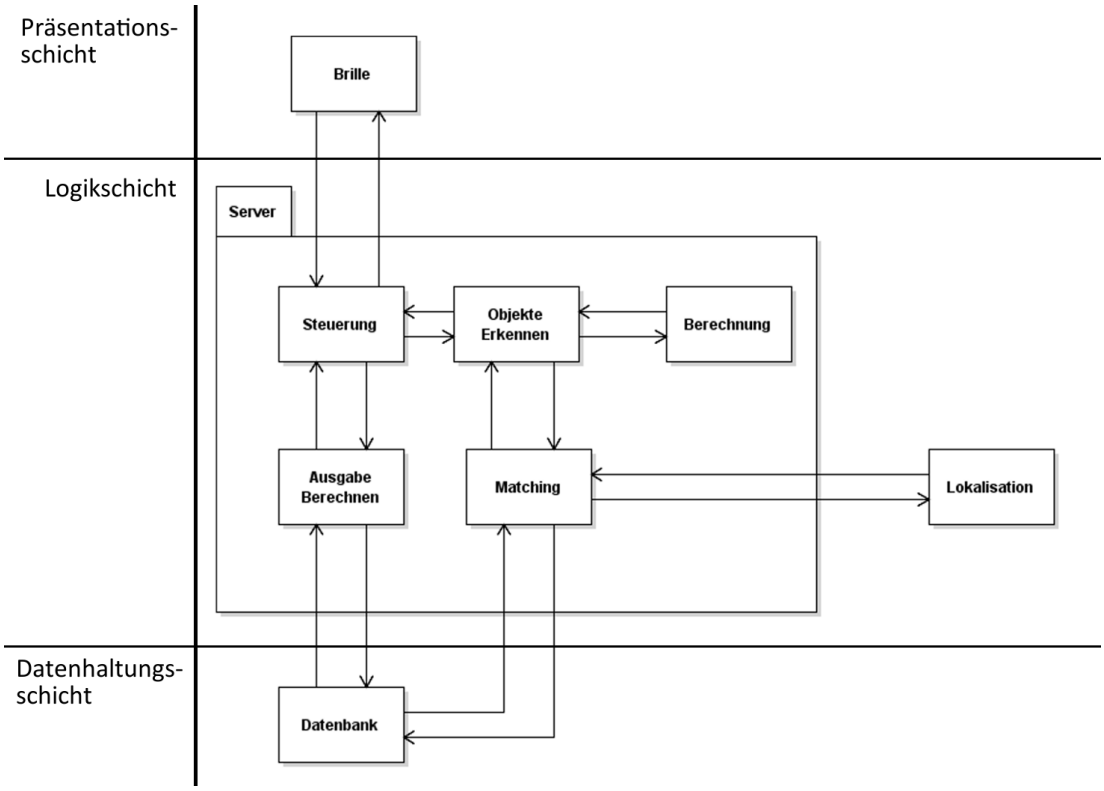

Abbildung 11: Schichtenmodell

Die Präsentationsschicht ist für die grafische Darstellung und für Interaktionen des Anwenders mit dem System zuständig. In der Logikschicht wird die Logik der Software mit den Algorithmen und Berechnungen bereitgestellt. In der Datenbankschicht werden die Daten der Software gespeichert und verwaltet.

3.1.1.1. Präsentationsschicht

In diesem System nimmt der Anwender nur die Präsentationsschicht wahr. Hier wird die grafische Ausgabe für den Nutzer dargestellt. Nachdem die Brille das Kamerabild aufgenommen hat, wird dieses an die Logikschicht gesendet.

In diesem Entwurf wird die Präsentationsschicht durch die Anwendung auf der Brille repräsentiert.

3.1.1.2. Logikschicht

In diesem Entwurf ist die Logikschicht komplex und setzt die Logik des Systems um. In der Logikschicht befinden sich der Server und ein System für die Lokalisierung 0.

In der Server-Komponente wird die Anwendungslogik gesteuert und berechnet. Für die Positionsbestimmung wird die Lokalisierungskomponente angesprochen und zum Speichern oder Abfragen von Daten wird auf die Datenbank in der Datenhaltungsschicht zugegriffen. Der Server nimmt das Kamerabild der Smartglass entgegen. Die Berechnung der Merkmale des Kamerabildes und das Matching der Merkmale mit den Gegenständen aus der Datenbank werden mit visuellen Objekterkennungs-Verfahren umgesetzt. Um den Suchraum der gesuchten Gegenstände einzuschränken, wird die Position über die Lokalisierung bestimmt und bei der Datenbankabfrage berücksichtigt.

In der Lokalisierung wird die Position des Anwenders über Lokalisationsverfahren berechnet. Diese Komponente ist von dem Server getrennt und kann als eigenständiger Dienst in der Smart Home Umgebung fungieren oder als externer Dienst verwendet werden.

3.1.1.3. Datenhaltungsschicht

In der Datenhaltungsschicht werden die Daten, die für diese Anwendung benötigt werden, gespeichert und dem System zur Verfügung gestellt. Für die Datenhaltungsschicht bietet sich ein Datenbanksystem an. Hierin müssen die Position, die digitalen Notizen und die Bildmerkmale gespeichert werden. Es muss möglich sein, die Bildmerkmale einzelnen Objekten zuzuordnen. Außerdem müssen die digitalen Notizen und die Positionen in der Datenbank mit den Objekten verknüpft werden können.

3.1.2 Bausteinsicht

In diesem Kapitel werden die Struktur und die Zusammenhänge der Softwarekomponenten statisch dargestellt. Die Bausteinsicht vermittelt eine weitere Sicht auf die Architektur des Systems und verdeutlicht, welche Elemente zusammengehören. Es werden drei Level der Bausteinsicht gezeigt, die in verschiedenen Abstufungen das System oder die Bausteine des Systems beschreiben. Zu jeder Abstufung werden die Teilsysteme kurz beschrieben.

Level-0

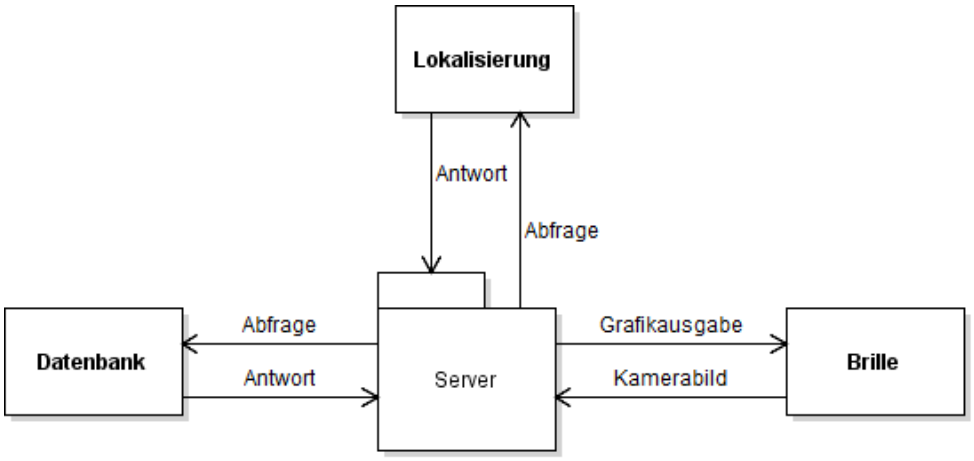

Abbildung 12: Bausteinsicht – Level 0

Die „Level-0“ Ebene zeigt vier Komponenten des Systems: die *Brille*, die der Anwender des Systems trägt, den *Server*, die *Lokalisierung* und die *Datenbank*.

Die *Brille* überträgt das Kamerabild an den *Server* und empfängt von diesem die Grafikausgabe, die auf dem HMD ausgegeben wird.

Der *Server* empfängt das Kamerabild der *Brille*, fragt bei der *Lokalisierung* die Position und bei der *Datenbank* Informationen (die digitalen Notizen) und die *Daten* zu den registrierten Gegenständen ab. Nach dem Abschluss aller Berechnungen sendet der Server die Grafikausgabe an die *Brille* zurück.

Die *Lokalisierung* erhält Anfragen von dem *Server*, daraufhin wird die Position des Anwenders bestimmt und als Antwort an den *Server* übermittelt.

In der *Datenbank* werden Daten gespeichert, die für den Betrieb des Systems benötigt werden. Der *Server* stellt Anfragen an die Datenbank, um die registrierten Gegenstände für die Objekterkennung oder die Informationen zu gefundenen Gegenständen für die Grafikausgabe zu erhalten.

Level-1

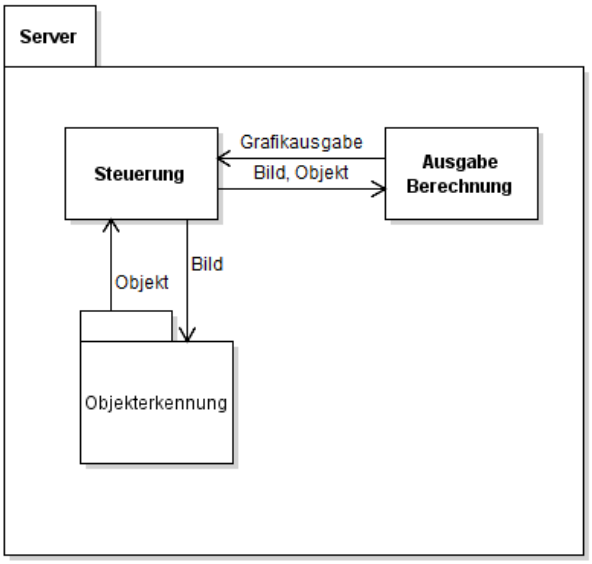

Abbildung 13: Bausteinsicht – Level 1

Die “Level-1“ Ebene stellt eine Verfeinerung des *Servers* dar. Das Steuerungselement koordiniert die Abläufe innerhalb des Servers. So übergibt die *Steuerung* das empfangene Kamerabild der *Brille* an die *Objekterkennung* und erhält als Rückgabe ein gefundenes Objekt. Das Objekt wird gemeinsam mit dem empfangenen Bild an die *Ausgabe-Berechnung* übergeben. Dort findet die Berechnung der Grafikausgabe statt, die über die *Steuerung* an die *Brille* gesendet wird.

Level-2

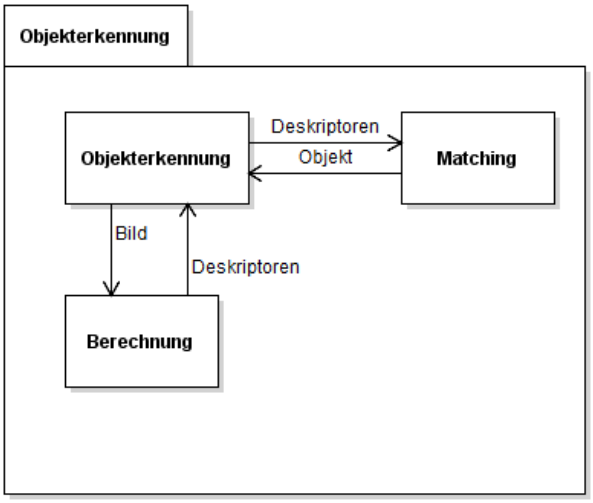

Abbildung 14: Bausteinsicht – Level 2

Die „Level-2“ Ebene beschreibt die Objekterkennung innerhalb des *Servers* näher. Der *Objekterkennung* wird ein Bild übergeben, aus dem in der *Berechnung* die Deskriptoren berechnet werden. Die Deskriptoren werden über die *Objekterkennung* an das *Matching* übermittelt. Für die Berechnung des Matchings wird dort eine Anfrage an die Lokalisierung gestellt. Mit der Position wird eine Abfrage an die *Datenbank* gestellt und die registrierten Objekte mit der Position gefiltert. Von der *Datenbank* erhält das Matching die benötigten Merkmale bzw. Deskriptoren der registrierten Gegenstände, die mit den Deskriptoren des Bildes verglichen werden. Gefundene Objekte werden von dem *Matching* an die Objekterkennung weitergeleitet, die das gefundene Objekt an die *Steuerung* des *Servers* übermittelt.

3.1.3 Laufzeitsicht

In der Laufzeitsicht wird das dynamische Zusammenwirken der einzelnen Elemente des Systems beschrieben. Der Fokus in der Laufzeitsicht besteht in der Kommunikation zur Laufzeit: Welches Element kommuniziert wann und wie mit einem anderen? Die Darstellung der Laufzeitsicht erfolgt mit einem Sequenzdiagramm.

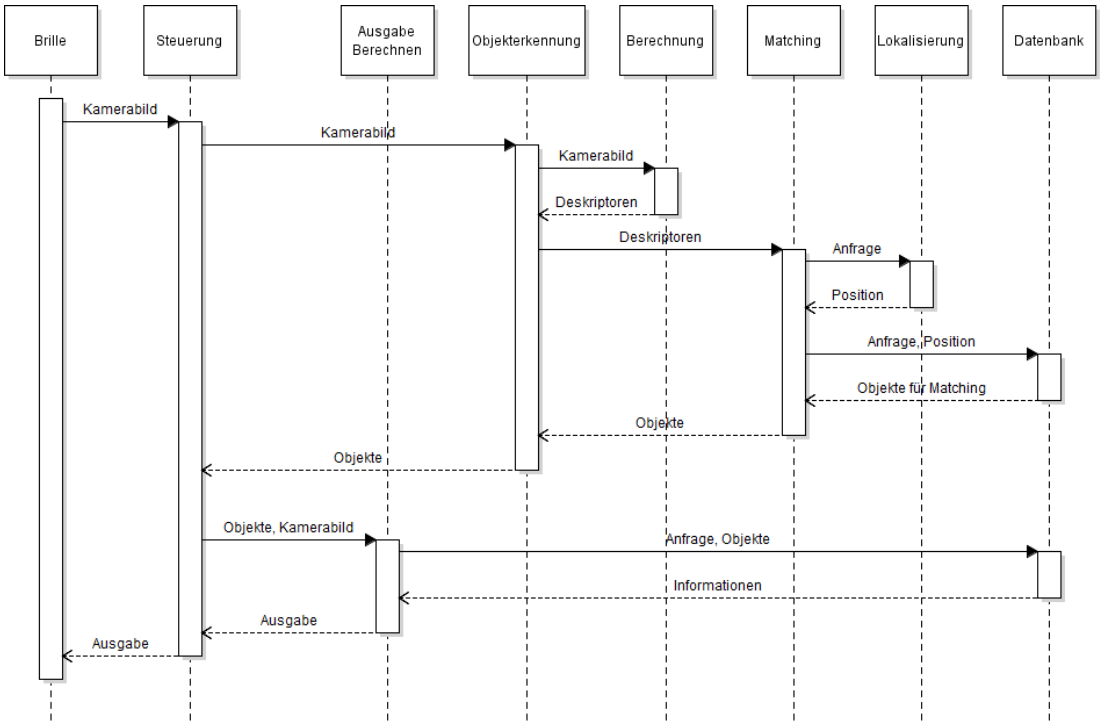

Abbildung 15: Laufzeitsicht

In der Abbildung 15 wird die Laufzeitsicht des Designs dargestellt. In dieser Sicht wird die *Request-Response* Eigenschaft zwischen den Komponenten der Anwendung deutlich. Nach der Anfrage einer Komponente erhält diese die Antwort übermittelt.

Die *Brille* stellt eine Anfrage an die *Steuerung* und übermittelt hierbei das Kamerabild. Als Antwort erhält die Brille die fertige Grafikausgabe.

Die *Steuerung* gibt das empfangene Kamerabild an die *Objekterkennung* weiter und bekommt die auf dem Bild gefundenen Objekte als Antwort. Im nächsten Schritt wird die Grafikausgabe berechnet. Hierzu werden das Kamerabild und die gefundenen Objekte an die Komponente *Ausgabe Berechnen* gesendet und von dieser die fertige Grafikausgabe empfangen. Die Grafikausgabe wird als Antwort an die *Brille* gesendet.

Die *Objekterkennung* erhält das Kamerabild und leitet dieses an die *Berechnungs*-Komponente weiter, um die Deskriptoren zu erhalten. Die Deskriptoren werden an das *Matching* weitergereicht. Dort werden die Deskriptoren mit den bereits registrierten Objekten gematcht. Als Antwort erhält die *Objekterkennung* die gefundenen Objekte, die zurück an die *Steuerung* gesendet werden.

Die *Berechnungs*-Komponente berechnet aus dem Kamerabild die KeyPoints und ermittelt aus diesen die Deskriptoren. Die Deskriptoren werden als Antwort zurück an die *Objekterkennung* gesendet.

Das *Matching* stellt bei der *Lokalisierung* eine Anfrage und erhält als Antwort die Position des Anwenders. Mit der Position wird eine Anfrage an die *Datenbank* gestellt. Von dieser werden die Objekte, die den Suchraum des Matchings bestimmen, in Form von Deskriptoren an die *Matching*-Komponente übermittelt. Aus den Deskriptoren des Kamerabildes und den Deskriptoren der Datenbank wird ein passendes Matching gesucht. Wenn ein Matching gefunden wird, werden die gefundenen Objekte an die Objekterkennung übermittelt.

Die *Lokalisierung* erhält eine Anfrage und bestimmt daraufhin die Position des Anwenders. Diese Position wird als Antwort erwidert.

Die Komponente *Ausgabe Berechnen* erhält von der *Steuerungs*-Komponente das Kamerabild der Brille und die gefundenen Objekte. Aus diesen wird die Grafikausgabe berechnet. Um das Kamerabild mit den Objekten und den Informationen anzureichern, wird in der *Datenbank* nach den entsprechenden Informationen gesucht. Hierzu wird an die *Datenbank* eine Anfrage mit den Objekten gestellt und als Antwort die zugehörigen Informationen erwartet. Die berechnete Grafikausgabe wird an die Steuerung übergeben.

Die Datenbank erhält Anfragen der *Matching*- und der *Ausgabe Berechnen*-Komponenten. Bei einer Anfrage der *Matching*-Komponente wird mit der Anfrage die Position übermittelt. Die registrierten Objekte werden nach der Position gefiltert und die dazugehörigen Deskriptoren an das *Matching* übergeben. Bei einer Anfrage der *Ausgabe Berechnen*-Komponente wird die Anfrage mit den gefundenen Objekten gestellt. In der *Datenbank* wird dort nach Informationen gesucht, die mit diesen Objekten verknüpft wurden. Gefundene Informationen werden an die *Ausgabe Berechnen*-Komponente übermittelt.

3.1.4 Message Broker

Die Kommunikation zwischen Anwendungen wird mit Hilfe sogenannter *Message Broker* (MB) ([Curry 2004] und [Tomova 2014]) umgesetzt. Mit dem MB wird eine Zwischenschnittstellte zwischen dem Sender und dem Empfänger einer Nachricht geschaffen, die es möglich macht, die Kommunikation zu steuern. Diese Schnittstelle ermöglicht die Kommunikation von Anwendungen zwischen Betriebssystemen und unterschiedlichen Programmiersprachen. Durch den Einsatz dieser zusätzlichen

Schnittstelle lassen sich Änderungen der einzelnen Komponenten leichter durchführen, da die Kopplung lose ist.

Message Broker gehören zu der Gruppe der *Message oriented middleware* (MOM), die eine Alternative zu den *Remote Procedure Calls* (RPC), die den Wunsch nach besserer Performance nicht mehr erfüllen konnten, bieten. RPCs kommunizieren meist synchron und MOMs kommunizieren üblicherweise asynchron.

Bei der synchronen Kommunikation blockieren die Prozesse während des Sendens oder Empfangens. Im Falle einer Client-Server Applikation stellt der Client eine Anfrage an den Server und wartet/blockiert, bis der Server eine Antwort übertragen hat. In diesem Konzept wird eine sofortige Antwort erwartet.

Bei der asynchronen Kommunikation blockieren die Prozesse nicht, um nach dem Senden einer Anfrage auf den Empfang der Antwort zu warten. Nach dem Senden der Anfrage kann der Prozess seine Berechnung fortführen und erhält zu einem späteren Zeitpunkt die Antwort. In diesem Konzept wird keine sofortige Antwort erwartet.

Für die Funktionsweise einer MOM wird zwischen den Produzenten (Sender), Konsumenten (Empfänger) und dem *Message Broker* unterschieden. Die Produzenten erzeugen Nachrichten, die diese an den MB senden. Die Konsumenten fordern beim MB neue Nachrichten an.

Die Kernaufgabe des *Message Brokers* ist das Empfangen von Nachrichten der Produzenten und die Weiterleitung dieser an bestimmte Konsumenten. Weitere Aufgabe eines MB kann die Umwandlung des Nachrichtenformates vom Sender in ein oder mehrere Formate für die Empfänger sein.

Die MBs verwenden sogenannte *Message Queues*, mit denen es ermöglicht wird, Nachrichten für die Konsumenten vorrätig zu halten, auch wenn Produzent und Konsument nicht zeitgleich aktiv sind. Die *Message Queues* halten die Nachrichten nach dem „First In - First Out“-Prinzip vorrätig.

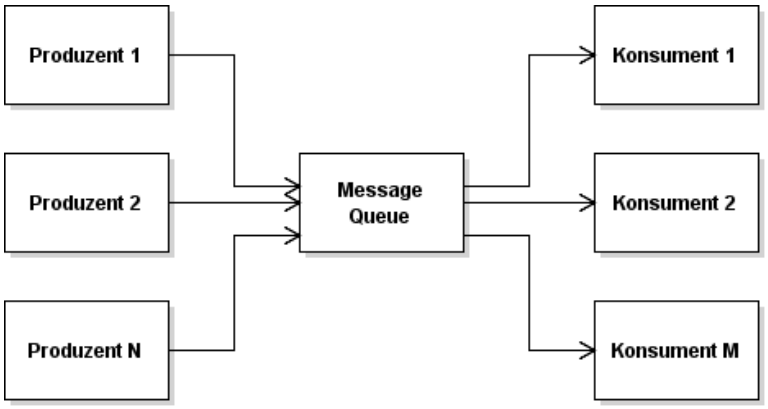

Abbildung 16: Message Queue mit Produzenten und Konsumenten

Bei den *Message Queues* wird zwischen *Queue* und *Topic* unterschieden. Bei den *Queues* findet die Kommunikation zwischen einem oder mehreren Produzenten und nur einem Konsumenten (n:1) statt. Bei den *Topics* senden die Produzenten die Nachrichten an ein themenspezifisches *Topic* und die Konsumenten abonnieren das *Topic*, es wird auch von *Publish/Subscribe* gesprochen. Bei den Topics müssen die Produzenten und Konsumenten keine Kenntnis voneinander haben.

In den MOMs wird über Protokolle, wie XMPP[31] oder JMS[32], miteinander asynchron kommuniziert. Beispiele für Implementierungen einer MOM sind u.a. *RabbitMQ*[33] oder *Apache ActiveMQ*[34]. *Apache ActiveMQ* ist eine Implementation für ein MOM, die unter der Apache-Lizenz (Open Source) veröffentlicht wird und ein verbreiteter MB ist. Die Implementation wird in Java entwickelt, unterstützt verschiedene Protokolle, wie JMS und bietet Schnittstellen für verschiedene Programmiersprachen, dazu gehören C/C++, Ruby, PHP und Python.

[31] http://xmpp.org/
[32] http://www.oracle.com/technetwork/java/jms/index.html
[33] https://www.rabbitmq.com/
[34] http://activemq.apache.org/

3.1.5 Lokalisierung

Die Lokalisation, in Kapitel 2.6 beschrieben, und speziell die Indoor-Lokalisation ist eine Problemstellung, die in Smart Homes Umgebungen häufig auftritt. So kann davon ausgegangen werden, dass in einem Smart Home Systeme zur Verfügung stehen, die eine derartige Ortung zulassen. So wurde in dem *Living Place Hamburg* das System UbiSense (2.6.3.2) integriert.

In diesem Systementwurf wird für die Lokalisierung des Anwenders, in Kapitel 2.6 beschrieben, auf ein externes System zugegriffen. Hierbei wird eine Anfrage an das System gestellt und als Antwort die Position des Anwenders erwartet. Abhängig vom System können weitere technische Anforderungen, wie zusätzliche Geräte, für die Nutzung der Lokalisierung entstehen.

3.1.6 Datenformate

Für die Kommunikation zwischen den Teilsystemen, siehe Abbildungen in Kapitel 3.1.2, müssen die Datenformate geklärt werden.

Wenn das System auf externe Anwendungen, wie die Lokalisierung, zugreift, ist das Datenformat bereits festgelegt. Falls das Format für die Anwendung passend ist, so wird dieses übernommen. Sollte das Datenformat nicht kompatibel sein, so kann dieses durch eine Middleware angepasst werden.

Für die Kommunikation zwischen der Brille und dem Server muss ein Datenformat festgelegt werden, um den Bildstrom von der Brille zum Server zu übertragen. Für die Übertragung der Ausgabe, für das HMD, zwischen dem Server und der Brille muss ein weiteres passendes Format gefunden werden.

3.2 Realisierung

In dem Kapitel 3.1 wurde die Architektur des Systems entworfen und fokussierte sich auf die Erkennung von mehreren Objekten in dem Bild der Smartglass. Auf der Basis der Systemarchitektur wird in diesem Abschnitt die Realisierung der Software gezeigt.

Wie in Kapitel 0 dargestellt, soll in der Realisierung die Objekterkennung über kamerabasierte Verfahren umgesetzt und evaluiert werden. Die Anforderungen, auf die

sich die Realisierung konzentriert, sind die Erkennung mehrerer Objekte in dem Kamerabild (2.3.6) und die zeitliche Vorgabe, die Objekte innerhalb von 1,5 Sekunden zu ermitteln (2.3.13). Aus diesem Grund wurde der Baustein der Objekterkennung, siehe Abbildung 14, umgesetzt. Das Kamerabild und die Referenzbilder für die registrierten Objekte wurden durch direkten Zugriff abgerufen und der Objekterkennung zur Verfügung gestellt, so dass auf den Einsatz einer *Message Queue* verzichtet wurde.

In dem Abschnitt 3.2.1 werden die für die Realisierung verwendete Software und die Bibliotheken beschrieben. Das System wurde in C++ entwickelt, welches sich anhand der verwendeten Software als geeignet erwies.

3.2.1 Verwendete Software und Bibliotheken

In diesem Abschnitt werden die verwendeten Bibliotheken bzw. Software für die Bausteine der Objekterkennung beschrieben. Für diese wurden jeweils andere Bibliotheken verwendet.

Mit OpenCV (Version 2.4.10) wurde eine Bibliothek für Probleme der Computer Vision verwendet, die weitere Bibliotheken, wie die Objekterkennungsverfahren und FLANN-Bibliothek, integriert, siehe 0. Neben diesen bietet OpenCV weitere Algorithmen für die Bildverarbeitung.

Für die *Berechnen*-Komponente wurden die SIFT-, SURF-, ORB-, FAST- und BRIEF-Bibliotheken verwendet, siehe 3.2.1.2, um die Keypoints und die Deskriptoren zu berechnen.

Für die *Matching*-Komponente wurde die *Fast Approximate Nearest Neighbors* (FLANN)[35]-Bibliothek verwendet, die es ermöglicht, unterschiedliche Deskriptor-Arten zu matchen, siehe 3.2.1.3. Außerdem ist es möglich, eine Deskriptor-Menge mit den Deskriptoren mehrerer Referenzbilder zu vergleichen.

[35] http://www.cs.ubc.ca/research/flann/

3.2.1.1. OpenCV

Mit OpenCV[36], auch *Open Source Computer Vision* genannt, wurde eine Bibliothek unter der BSD Open Source Lizenz veröffentlicht, die laufend erweitert wird (siehe auch: [Pulli 2012]). Die Bibliothek stellt Algorithmen für Probleme der Computer-Vision, dem maschinellen Sehen, bereit. Darunter befinden sich Algorithmen zur Gesichtserkennung, der Objekterkennung, dem Tracking und weiteren Anwendungsgebieten der Computer Vision oder des Maschinellen Lernens. Auch einfachere Algorithmen zur Bildverarbeitung werden durch OpenCV zur Verfügung gestellt.

Die Entwicklung begann 1998 im Rahmen eines Projekts von Intel[37]. OpenCV wird in C++ entwickelt und ermöglicht es seit 2010, Rechenoperationen auf die GPU auszulagern. Neben der C++ Schnittstelle stehen Wrapper[38] für andere Programmiersprachen zur Verfügung.

Einige Verfahren zur Objekterkennung, siehe 2.5 und 3.2.1.2, wurden in OpenCV implementiert. Zu den Algorithmen gehören die Verfahren SIFT und SURF, die auf dem *Difference of Gaussians* (DoG, siehe 2.5.1.1) basieren, sowie *ORB* und *FAST*, die auf *Corner Detection* (siehe: [Harris und Stephens 1988]) basieren.

Eine Implementierung der FLANN-Bibliothek wird auch in OpenCV verwendet. Eine nähere Beschreibung zu FLANN findet sich in Abschnitt 3.2.1.2.

3.2.1.2. Visuelle Objekterkennungsverfahren

Für die Objekterkennung wurden mit SIFT, SURF, ORB und FAST vier unterschiedliche Verfahren verwendet. Die Verfahren SIFT und SURF verwenden eigene Deskriptoren, die durch Fließkommazahlen repräsentiert werden. Der FAST Algorithmus ist für das Ermitteln von KeyPoints zuständig und hat keinen eigenen Deskriptor. Hier wird der BRIEF-Deskriptor verwendet, während ORB wiederum eine eigene Variante des BRIEF-Deskriptors nutzt.

In dem Kapitel 3.3.2 werden die Verfahren anhand der Rechenzeiten miteinander verglichen und ihre Eignung für das System überprüft.

[36] http://opencv.org/
[37] http://www.intel.de
[38] Schnittstelle, die es ermöglicht Software auch von anderen Programmiersprachen zu nutzen.

Einige der Verfahren können miteinander kombiniert werden, so können SIFT und SURF parallel verwendet werden. ORB und FAST können ebenfalls kombiniert werden. Weitere Kombinationen, wie zwischen SURF und ORB, sind nicht zulässig, da die Deskriptoren unterschiedlich repräsentiert werden. Die Verfahren SIFT und SURF setzen auf Fließkommazahlen und der BRIEF-Deskriptor auf Bitstrings.

3.2.1.3. FLANN

Als weitere Software wurde die OpenCV-Implementation der FLANN-Bibliothek in der Realisierung verwendet. FLANN ist eine Bibliothek, die Algorithmen für die Nächste-Nachbarn-Suche in großen Datensätzen zur Verfügung stellt und es ermöglicht passende Matches innerhalb mehrerer Deskriptor-Mengen zu finden. Erstmals vorgestellt wurde FLANN im Jahre 2009 von Marius Muja und David G. Lowe.

> *„The most computationally expensive part of many computer vision algorithms consists of searching for the closest matches to high-dimensional vectors.“* ([Muja und Lowe 2009])

Die Berechnung passender Matches ist sehr zeitaufwändig. Von der zugrundeliegenden Struktur der Deskriptoren abhängig, eignen sich unterschiedliche Algorithmen zur Berechnung passender Matches. Um in großen Datensätzen passende Matches zu finden, stellt die Open-Source Bibliothek FLANN verschiedene Methoden bereit. Hierbei wählt die Bibliothek selbstständig einen geeigneten Algorithmus mit optimalen Parametern.

> *“In our experiments, one of two algorithms obtained the best performance, depending on the dataset and desired precision. These algorithms used either the hierarchical k-means tree or multiple randomized kd-trees.”* ([Muja und Lowe 2009])

Für die Entwicklung der FLANN-Bibliothek wurden mehrere Algorithmen für die Nächste-Nachbarn-Suche in kd-Bäumen[39] untersucht. Für kleine Datensätze sind kd-Bäume hocheffizient, in großen Datensätzen bricht die Performance jedoch schnell ein. Aus diesem Grund werden Variationen des kd-Baumes verwendet. Im Vordergrund stehen bei diesen nicht die idealen Ergebnisse, sondern eine gute Performance für große Datensätze mit

[39] Unbalancierter Suchbaum in einem K-dimensionalen Raum

guten Ergebnissen. In der FLANN-Bibliothek werden *multiple randomized kd-bäume* und *hierarchical k-means-bäume*, zwei Arten der kd-Bäume, verwendet.

Kd-Bäume sind unbalancierte Suchbäume, die einen k-dimensionalen Raum beschreiben. Für den Aufbau des Baumes wird der k-dimensionale Raum rekursiv in zwei Teile aufgeteilt. Jeder Knoten beschreibt einen Teil des kd-Raumes. Die Knoten haben jeweils zwei Kindsknoten, die den Teilraum weiter unterteilen. Ein Anwendungsgebiet der kd-Bäume ist die Nächste-Nachbarn-Suche, die beim Matching genutzt wird und diese daher in der FLANN-Bibliothek verwendet werden (vgl: [Birn 2009]).

3.3 Evaluation

In diesem Kapitel wird die Realisierung des Designs im Hinblick auf die Anforderungen der Objekterkennung und der zeitlichen Bedingungen evaluiert. Hierzu werden zum einen die Objekterkennung und zum anderen die Rechenzeiten für die Ermittlung der KeyPoints, der Deskriptoren und die Bestimmung der Matchings betrachtet. Auf die Messung der Abfragezeit der Informationen wurde verzichtet, da es sich hierbei um einfache Datenbankabfragen handelt und somit von einer zu vernachlässigenden Latenz ausgegangen werden kann.

3.3.1 Objekterkennung

Die Anforderungen an die Objekterkennung wurden in 2.3.6 definiert. In einem Bild sollen zuvor registrierte Objekte wiedererkannt werden, hierzu soll das passende Objekt aus einer Menge an Objekte gefunden werden. Außerdem sollte die Objekterkennung robust gegenüber Änderungen in der Belichtung, Skalierung, Perspektive, Rotation und einer Verdeckung von Teilen des Objektes sein.

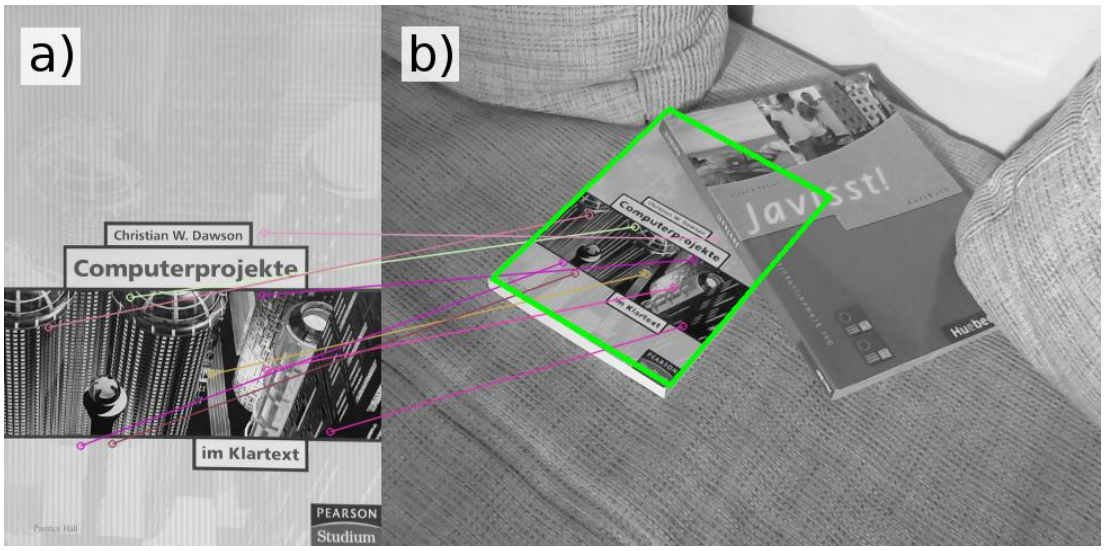

Abbildung 17: Referenzbild (a) und Kamerabild (b) mit gefundenen Matches, dargestellt durch die Linien

Auf der Abbildung 17 ist eine erfolgreiche Objekterkennung der Realisierung dargestellt, die KeyPoints und Deskriptoren wurden mit SURF und die Matchings mit FLANN ermittelt. Auf der linken Seite ist (a) das gefundene Referenzbild und rechts (b) das Kamerabild, auf dem das Buch erfolgreich erkannt wurde, zu sehen. Auf dem Kamerabild wurde das gefundene Objekt durch die Realisierung grün umrandet. Die Linien zwischen dem Referenzbild und dem gefundenen Objekt stellen die korrekten Matches der Deskriptoren dar.

Das Buch wurde erfolgreich aus einer Auswahl an Objekten wiedererkannt. In diesem Beispiel wird die Robustheit der Verfahren gegenüber den geforderten Parametern Belichtung, Skalierung, Perspektive, Rotation und Verdeckung deutlich.

3.3.2 Rechenzeit der Objekterkennung

In diesem Abschnitt werden die Objekterkennnungs-Verfahren SIFT, SURF, FAST und ORB betrachtet und die Rechenzeiten miteinander verglichen. Die Messungen wurden auf dem gleichen System und unter gleichen Rahmenbedingungen durchgeführt, so wurden auch das gleiche Video und die gleichen Referenzbilder verwendet. Das SIFT- und das SURF-Verfahren enthalten eigene Deskriptoren. Das ORB-Verfahren verwendet eine Weiterentwicklung des „Binary Robust Independent Elementary Features“ (BRIEF)-Deskriptors, der diesen um Rotationsinvarianz erweitert. Das FAST-Verfahren verfügt über keinen definierten Deskriptor, für die Messungen wurde ebenfalls der schnelle BRIEF-Deskriptor verwendet. Das Matching erfolgte bei allen vier Verfahren über die FLANN-

Bibliothek. Die angegebenen Zeiten sind Mittelwerte, die sich aus den gemessenen Zeiten der Verfahren ergeben.

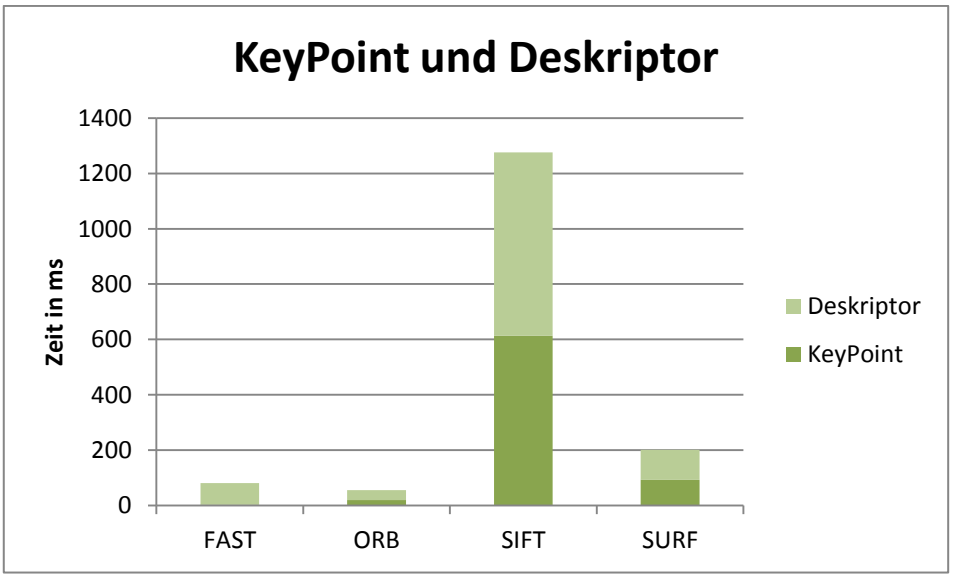

Abbildung 18: Durchschnittliche Berechnungszeit der Algorithmen für die KeyPoints und Deskriptoren

In der Abbildung 18 wird die Berechnungszeit der vier Algorithmen für die Ermittlung der KeyPoints und der Deskriptoren dargestellt. Das SIFT-Verfahren zeigt mit etwa 1,27 Sekunden die längste Berechnungszeit. Das SURF-Verfahren benötigt 0,2 Sekunden, FAST 0,08 Sekunden und ORB benötigt 0,05 Sekunden. Damit ist für diese Berechnungen ORB am schnellsten und SIFT benötigt eine bedeutend längere Zeit. Die Dauer zur Berechnung der KeyPoints und der Deskriptoren ist bei SIFT, SURF und ORB ähnlich aufgeteilt. Bei dem Fast Algorithmus ist die Berechnung der KeyPoints jedoch etwa 20 mal schneller als die Berechnung des Deskriptors.

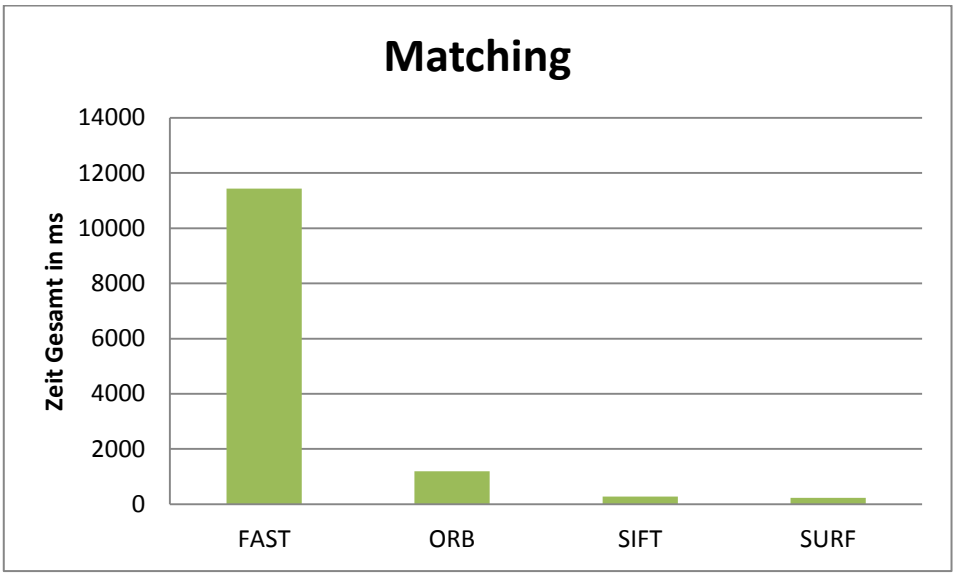

Abbildung 19: Durchschnittliche Berechnungsdauer für das Matching mit FLANN

Für das Matching über die FLANN-Bibliothek zeigt sich eine andere Aufteilung der Berechnungszeiten. Das SIFT-Verfahren benötigt durchschnittlich 0,28 Sekunden, SURF 0,23 Sekunden, ORB 1,18 Sekunden und FAST 11,4 Sekunden. Damit ist das Matching für die SIFT- und SURF-Deskriptoren deutlich schneller als die Deskriptoren für die ORB- und FAST-Verfahren.

Die deutlichen Unterschiede von ORB und FAST zu SIFT und SURF überraschen, da der BRIEF-Deskriptor gegenüber dem SURF-Deskriptor bis zu 41-fach schneller sein soll. Es ist für mich nicht ersichtlich, warum es zu den deutlichen „Ausreißern“ bei der Berechnung des Matchings für die FAST- und ORB-Deskriptoren kam. Mit den SIFT und SURF Deskriptoren arbeitet die FLANN-Bibliothek schnell und zuverlässig, mit dem BRIEF-Deskriptor scheint diese jedoch nicht effizient zu arbeiten.

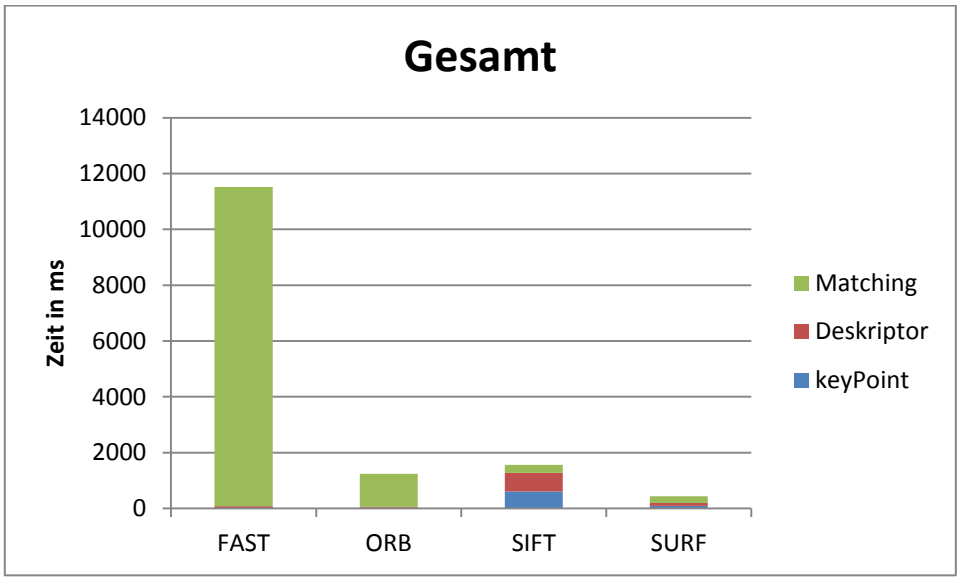

Abbildung 20: Durchschnittliche Berechnungsdauer der Algorithmen

In der Übersicht der Verfahren in Abbildung 20 wird deutlich, dass das Matching der rechenintensivste Vorgang bei der Objekterkennung ist. Nur das SIFT-Verfahren macht hier, aufgrund der langen KeyPoint- und Deskriptor-Berechnung, eine Ausnahme.

Die Verwendung des SURF-Verfahrens erfüllt die Zeitvorgaben in 2.3.13, auch wenn es für die Keypoint- und Deskriptor-Berechnungen längere Zeit als FAST und ORB benötigt. Sobald die Probleme mit dem BRIEF-Deskriptors im Zusammenspiel mit der FLANN-Bibliothek behoben werden, sollten FAST und ORB die Zeitvorgaben ebenfalls erreichen.

3.4 Fazit

In diesem Kapitel wurde ein Design des Systems entworfen und mit der Realisierung umgesetzt. Das Ziel war es, ein System zu entwerfen, das für die digitalen Notizen geeignet ist und die daraus resultierenden Anforderungen erfüllt. Die Realisierung wurde mit der Objekterkennung, einem elementaren Ausschnitt des Designs und wichtige Anforderung, umgesetzt. Anschließend wurde die Realisierung anhand der Anforderungen evaluiert.

Die Evaluation zeigt, dass die Realisierung die Anforderungen erfüllt. Die Anforderung registrierte Objekte zu erkennen, wurde, wie in Abbildung 17 zu sehen, erfüllt und auch die benötigte Robustheit ist gegeben. Die zeitlichen Bedingungen werden mit dem Einsatz des SURF-Verfahrens ebenfalls erfüllt. In den Messungen erzielte das SURF-Verfahren die

besten Ergebnisse, doch auch die ORB- und FAST-Verfahren zeigten ihr Potenzial, indem diese bei der Berechnung der KeyPoints und Deskriptoren am besten abschnitten. Für den Gesamteindruck gab das Matching über die FLANN-Bibliothek die entscheidende Wendung, da die Berechnung des Matchings der Deskriptoren für FAST(BRIEF) und ORB unerwartet viel Zeit beanspruchte.

Mit dem Einsatz der in 3.2.1 aufgeführten Bibliotheken habe ich gute Erfahrungen gemacht. OpenCV ist eine umfangreiche Bibliothek mit einer Vielzahl an Verfahren für Algorithmen der Bildverarbeitung. Neben dem Umfang kann festgestellt werden, dass die eingesetzten Verfahren zuverlässig arbeiten. Die FLANN-Bibliothek arbeitete mit den SIFT- und SURF-Deskriptoren schnell und zuverlässig, ausschließlich bei den FAST(BRIEF)- und ORB-Deskriptoren zeigten sich Nachteile.

Zusammengefasst zeigt sich, dass das Design unter Berücksichtigung der gewonnenen Erkenntnisse gut umgesetzt werden kann.

4 Ausblick

Mit dieser Arbeit wurde die Anreicherung von Realität in geschlossenen Räumen gezeigt. Auf Basis der Objekterkennung wurden Gegenstände wiedererkannt und mit Informationen angereichert. Das Zusammenwirken einer Smartglass, der Lokalisation und der Objekterkennung wurde in der Arbeit ebenfalls ausgeführt.

In weitergehenden Arbeiten muss geklärt werden, welche Formen der Interaktionen intuitiv ausführbar sind und in dem Bereich der *Angereicherten Realität* sinnvoll eingesetzt werden können.

Die Erkenntnisse, die mit dieser Arbeit gewonnen werden, können auch auf andere Anwendungen übertragen werden. Sobald ausreichend Modelle von Objekten vorhanden sind, können ähnliche Anwendungen für den Einsatz im Museum oder in anderen Ausstellungen verwendet werden. In diesen Anwendungsbereichen können die Objekte mit den Hintergrundinformationen, die vermittelt werden sollen, angereichert werden.

Bei der Verwendung *Angereicherter Realität* stellen sich zahlreiche Fragen, die zukünftig beantwortet werden. So ist es bisher nicht absehbar, ob es sinnvoll ist, die Realität permanent oder nach Bedarf zu erweitern.

Außerdem kann nicht abgesehen werden, wie sich das Leben der Menschen durch den Einsatz der *Angereicherten Realität* ändert, so kann sich die Wahrnehmung der Realität ändern. Eine weitere Trennung der Menschen zwischen denen, die nur die Realität wahrnehmen und denen, die Zugriff auf die *Angereicherte Realität* haben, könnte sich etablieren.

Bereits heute zeigen sich Vorteile, wenn der Mensch Informationen durch digitale Systeme bezieht und die Realität bereits jetzt als erweitert wahrnimmt. Dies zeigt sich in der Nutzung von Navigationssystemen oder neueren Anwendungen zum Auffinden der günstigsten Tankstelle oder Standorten von Ausleihfahrzeugen in der Umgebung.

Einen spannenden Aspekt stellt die Ausbreitung der *Angereicherten Realität* dar. Wird die Technik von allen Alters- und Sozialschichten genutzt oder findet die diese nur in bestimmten Milieus Anwendung? Auch hier könnte sich eine Trennung zwischen Bildungs- oder Altersschichten vollziehen.

Die Frage, ob die Angereicherte Realität wünschenswert ist oder beunruhigt, wird sich in der Zukunft zeigen und ist bisher nicht absehbar.

Literaturverzeichnis

[Ashwash] ASHWASH, I.; HU, W. & MARCOTTE, G.: Eye Gestures Recognition: A Mechanism for Hands-Free Computer Control. Online unter: https://www.cs.princeton.edu/courses/archive/fall08/cos436/FinalReports/Eye_Gesture_Recognition.pdf Abruf: 2016-03-22

[Babcock 2004] BABCOCK, J. S. & PELZ, J. B.: Building a Lightweight Eyetracking Headgear. In: *ACM*. : *Proceedings of the 2004 Symposium on Eye Tracking Research & Applications*., 2004, S. 109—114. Online unter: http://dl.acm.org/citation.cfm?id=968386 Abruf: 2016-03-22

[Batzill 2013] BATZILL, Adrian: *Interaktive Lokalisierung durch Objekterkennung*. Universität Freiburg, Bachelorarbeit, 2013. Online unter: http://ad-publications.informatik.uni-freiburg.de/theses/Bachelor_Adrian_Batzill_2013.pdf Abruf: 2016-03-22

[Bay 2008] BAY, H.; ESS, A.; TUYTELAARS, T. & VAN GOOL, L.: Speeded-Up Robust Features (SURF). In: *Comput. Vis. Image Underst.* 110 (2008), Nr. 3, S. 346—359. Online unter: http://campar.in.tum.de/twiki/pub/Chair/TeachingWs09MATDCV/SURF_paper.pdf Abruf: 2016-03-22

[Birn 2009] BIRN, Marcel: *Nächster-Nachbar-Suche mittels Knotenhierarchie in der Delaunay-Triangulierung*. Universität Karlsruhe, Studienarbeit, 2009. Online unter: http://algo2.iti.kit.edu/documents/NaechsterNachbarKnotenHierarchieBirn2009.pdf Abruf: 2016-03-22

[Blunsom 2004] BLUNSOM, Phil: *Hidden Markov Models*. University Melbourne, 2005. Online unter: http://digital.cs.usu.edu/~cyan/CS7960/hmm-tutorial.pdf Abruf: 2016-03-22

[BRAUER 2010] BRAUER, Henrik: *Entwicklung eines Augmented Reality Frameworks auf Basis von kamera-basierten Trackingverfahren*. HAW Hamburg, Masterarbeit, 2010

[Braun 2005] BRAUN, Tim: *Detektion von Landmarken zur Navigation von Laufmaschinen*. Universität Kaiserslautern, Seminararbeit, 2005. Online unter: https://agrosy.informatik.uni-kl.de/fileadmin/Literatur/Braun05d.pdf Abruf: 2016-03-22

[Calonder 2010] CALONDER, M.; LEPETIT, V.; STRECHA, C. & FUA, P.: Computer Vision – ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV. In: DANIILIDIS, K.; MARAGOS, P. & PARAGIOS, N. (Hrsg.), Berlin, Heidelberg: *Springer Berlin Heidelberg*., 2010, S. 778—792. Online unter: http://link.springer.com/chapter/10.1007%2F978-3-642-15561-1_56 Abruf: 2016-03-22

[Curry 2004] CURRY, E. & MAHMOUD, Q. H.: Message-Oriented Middleware : *Middleware for Communications*., 2004, S. 1—28. Online unter: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.418.173&rep=rep1&type=pdf Abruf: 2016-03-22

[Doerner 2013] Virtual und Augmented Reality (VR/AR). In: DÖRNER, R.; BROLL, WOLFGANG GRIMM, P. & JUNG, B. (Hrsg.), Berlin: *Springer Vieweg*., 2013. URL - http://link.springer.com/book/10.1007%2F978-3-642-28903-3 Abruf: 2016-03-23

[Elias 2006] ELIAS, Birgit: *Extraktion von Landmarken für die Navigation*. Universität Hannover, Dissertation, 2006. Online unter: https://www.ikg.uni-hannover.de/fileadmin/ikg/staff/publications/Begutachtete_Zeitschriftenartikel_und_Buchkapitel/diss_elias_2006.pdf Abruf: 2016-03-22

[Fuss 2011] FUSS, Christoffer: *Evaluierung von Feature Deskriptoren*. Humboldt-Universität zu Berlin, 2011. Online unter: https://www.informatik.hu-berlin.de/de/forschung/gebiete/viscom/thesis/final/Studienarbeit_Fuss_201110.pdf Abruf: 2016-03-22

[Gruendel] GRUENDEL, Verena: *„Lebendige Klötzchen – Augmented Reality von metaio erweckt Lego-Modelle zum Leben und lässt sie überall auf der Welt kleine Geschichten zum Happy End erzählen.“*. URL - http://metaio.com/fileadmin/upload/documents/press_articles/ds_3_2010_lego.pdf Abruf: 2016-03-23

[Hamberger u.a. 2010] HAMBERGER, W.; RIGLEY, P.; BETZ, M.; CHRIST, S.; EBNER, A. & DETTMER, K.: Bedienkonzept. In: *ATZextra* 15 (2010), Nr. 11, S. 222—227. Online unter: http://link.springer.com/article/10.1365/s35778-010-0491-0 Abruf: 2016-03-22

[Harris und Stephens 1988] HARRIS, C. & STEPHENS, M.: A combined corner and edge detector. In: *In Proc. of Fourth Alvey Vision Conference*., 1988, S. 147—151. Online unter: https://www.cis.rit.edu/~cnspci/references/dip/feature_extraction/harris1988.pdf Abruf: 2016-03-22

[Heidemann 2004] HEIDEMANN, G.; BAX, I. & BEKEL, H.: Multimodal Interaction in an Augmented Reality Scenario. ACM. In: *Proceedings of the 6th International Conference on Multimodal Interfaces*, 2004, S. 53—60. Online unter: http://dl.acm.org/citation.cfm?id=1027944 Abruf: 2016-03-22

[Hellwig 2013] HELLWIG, Andreas: *Interaktionen mit ubiquitären User-Interfaces*. Universität Koblenz-Landau, 2013. Online unter: https://kola.opus.hbz-nrw.de/files/758/MA_AndreasHellwig_print.pdf Abruf: 2016-03-22

[Henzinger 2008] HENZINGER, M.: Encyclopedia of Algorithms. In: KAO, M.-Y. (Hrsg.), Boston, MA: *Springer US*., 2008, S. 624—625. Online unter: http://link.springer.com/referenceworkentry/10.1007/978-0-387-30162-4_277 Abruf: 2016-03-22

[Heymann 2005] HEYMANN, Sebastian: *Implementierung und Evaluierung von Video Feauture Tracking auf moderner Grafikhardware*. Heinrich Hertz Institut Berlin, 2015. Online unter: http://web.uni-weimar.de/cms/fileadmin/medien/vr/documents/thesis/DiplomaThesisSebastianHeymann.pdf Abruf: 2016-03-22

[Hu 2015] HU, M.; LIU, Y. & FAN, Y.: Advances in Image and Graphics Technologies: 10th Chinese Conference, IGTA 2015, Beijing, China, June 19-20, 2015, Proceedings. In: TAN, T.; RUAN, Q.; WANG, S.; MA, H. & DI, K. (Hrsg.), Berlin, Heidelberg: *Springer Berlin Heidelberg*., 2015, S. 142—149. Online unter: http://link.springer.com/chapter/10.1007%2F978-3-662-47791-5_17 Abruf: 2016-03-22

[Huber u.a. 2013] HUBER, J.; KOPF, S. & SCHABER, P.: Analyse von Bildmerkmalen zur Identifikation wichtiger Bildregionen. In:*Technical reports*, Mannheim 13-004, 2013. Online

unter: https://ub-madoc.bib.uni-mannheim.de/33097/1/Kopf_2013d.pdf Abruf: 2016-03-22

[Jotzo 2002] JOTZO, Joachim: *Aktive Landmarken zur Positionsbestimmung von autonomen Fahrzeugen*. TU Chemnitz, Dissertation, 2002. Online unter: http://www.qucosa.de/fileadmin/data/qucosa/documents/4625/data/haupt.pdf Abruf: 2016-03-22

[Koelsch 2006] KOELSCH, M.; BANE, R.; HOELLERER, T. & TURK, M.: Multimodal Interaction with a Wearable Augmented Reality System. In: *IEEE Comput. Graph. Appl.* 26 (2006), Nr. 3, S. 62—71. Online unter: http://www.cs.ucsb.edu/~holl/pubs/Kolsch-2006-CGaA.pdf Abruf: 2016-03-22

[Lubos u.a. 2014] LUBOS, P.; BRUDER, G. & STEINICKE, F.: Safe-&-round: bringing redirected walking to small virtual reality laboratories. In: *Proceedings of the 2nd ACM Symposium on Spatial UWser Interaction, SUI 2014, Honolulu, HI, USA, October 4-5, 2014*., 2014, S. 154. Online unter: http://doi.acm.org/10.1145/2659766.2661219 Abruf: 2016-03-22

[Lowe 2004] LOWE, D. G.: Distinctive Image Features from Scale-Invariant Keypoints. In: *International Journal of Computer Vision* 60 (2004), S. 91—110. Online unter: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf Abruf: 2016-03-22

[Metaio] “Augmented Reality Product Visualization”. URL - http://www.metaio.com/fileadmin/upload/documents/pdf/product_flyers/metaio_ar_product_visualization.pdf Abruf: 2016-03-23

[Muja und Lowe 2009] MUJA, M. & LOWE, D. G.: Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. In: *INSTICC Press*. : *International Conference on Computer Vision Theory and Application VISSAPP'09)*., 2009, S. 331-340. Online unter: http://www.cs.ubc.ca/research/flann/uploads/FLANN/flann_visapp09.pdf Abruf: 2016-03-22

[Pavlovic u. a. 1997] PAVLOVIC, Vladimir I.; SHARMA, Rajeev; HUANG, Thomas S.: Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review. In: IEEE Trans. Pattern Anal. Mach. Intell. 19 (1997), Nr. 7, S. 677–695. – ISSN 0162-8828. Online unter: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=598226 Abruf: 2016-03-22

[Penelle 2014] PENELLE, B. & DEBEIR, O.: Multi-sensor Data Fusion for Hand Tracking Using Kinect and Leap Motion. In: *ACM*. : *Proceedings of the 2014 Virtual Reality International Conference*., 2014, S. 22:1--22:7. Online unter: http://dl.acm.org/citation.cfm?id=2620710 Abruf: 2016-03-22

[POTSDAM 2012] *Intelligente Datenanalyse in Matlab - Einführungsveranstaltung*. Universität Potsdam, 2012. Online unter: https://www.cs.uni-potsdam.de/ml/teaching/ws12/ida/Einfuehrungsveranstaltung.pdf Abruf: 2016-03-22

[Pulli 2012] PULLI, K.; BAKSHEEV, A.; KORNYAKOV, K. & ERUHIMOV, V.: Real-time Computer Vision with OpenCV. In: *Commun. ACM* 55 (2012), Nr. 6, S. 61—69. Online unter: http://dl.acm.org/citation.cfm?id=2184337 Abruf: 2016-03-22

[Rabiner 1993] RABINER, L. & JUANG, B.-H.: Fundamentals of Speech Recognition, Upper Saddle River, NJ, USA: *Prentice-Hall, Inc.*., 1993

[Rublee u.a. 2011] RUBLEE, E.; RABAUD, V.; KONOLIGE, K. & BRADSKI, G.: ORB: An efficient alternative to SIFT or SURF. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*., 2011, S. 2564-2571. Online unter: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6126544 Abruf: 2016-03-22

[Rudolf 2011] RUDOLF, SEBASTIAN: *Shared Notes – virtuelle Notizen im Smarthome – Vorarbeit Augmented Reality*. HAW Hamburg, 2011. Online unter: https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2011-proj1/rudolf.pdf Abruf: 2016-03-22

[Rupp 2001] RUPP, TORSTEN: *Absolute Lokalisation mobiler Roboter durch Codierungen mit Landmarken*. Universität Stuttgart, Dissertation, 2001. Online unter: http://www.kigen.de/files/Rupp2001.pdf Abruf: 2016-03-22

[Rosten2005] ROSTEN, E. & DRUMMOND, T.: Fusing points and lines for high performance tracking.. In: *IEEE International Conference on Computer Vision*., 2005, S. 1508—1511. Online unter: http://www.edwardrosten.com/work/rosten_2005_tracking.pdf Abruf: 2016-03-22

[Sarma und Sarma 2012] SARMA, M. & SARMA, K. K.: Recognition of Assamese Phonemes Using Three Different ANN Structures. In: *ACM*. : *Proceedings of the CUBE International Information Technology Conference*., 2012, S. 299—302. Online unter: http://dl.acm.org/citation.cfm?id=2381772 Abruf: 2016-03-22

[Schulz 2014] SCHULZ, Thomas: „Wearables: Das Silicon Valley zweifelt an Google Glass“. URL - http://www.spiegel.de/netzwelt/web/google-glass-silicon-valley-zweifelt-an-datenbrille-a-961413.html Abruf: 2016-03-22

[Souman u.a. 2009] SOUMAN, J. L.; FRISSEN, I.; SREENIVASA, M. N. & ERNST, M. O.: *Walking Straight into Circles*. In: *Current Biology* 19 (2009), Nr. 18, S. 1538 – 1542. Online unter: http://dx.doi.org/10.1016/j.cub.2009.07.053 Abruf: 2016-03-22

[Steggles und Gschwind 2005] STEGGLES, P.; GSCHWIND, S.: *The Ubisense Smart Space Platform*. In: Adjunct Proceedings of the Third International Conference on Pervasive Computing, Vol. 191*, Mai 2005,* S. 73 – 76. Online unter: http://www.pervasive.ifi.lmu.de/adjunct-proceedings/demo/p073-076.pdf Abruf: 2016-03-22

[Strobel 1995] STROBEL, J.: Global Positioning System: GPS ; Technik und Anwendung der Satellitennavigation: *Franzis*., 1995

[Tomova 2014] TOMOVA, MIHAELA TODOROVA: *Nachichtenbasierte Middleware – Ein Vergleich: Protokolle, Fähigkeiten, Implementierungen und Anwendungsgebiete*. TU Ilmenau, 2014. Online unter: http://yukon.e-technik.tu-ilmenau.de/~webkn/Abschlussarbeiten/Hauptseminararbeiten/hs_tomova.pdf Abruf: 2016-03-22

[Toennis 2010] TÖNNIS, M.: Augmented Reality – Einblicke in die Erweiterte Realität: *Springer*, 2010. Online unter: http://link.springer.com/book/10.1007%2F978-3-642-14179-9 Abruf: 2016-03-22

[Weichert 2013] WEICHERT, F.; BACHMANN, D.; RUDAK, B.; FISSELER, D.: *Analysis of the Accuracy and Robustness of the Leap Motion Controller*. TU Dortmund, 2013. Online unter: https://ls7-www.cs.uni-dortmund.de/publications/sensors-13-06380.pdf Abruf: 2016-03-22

[Wobbrock 2009] WOBBROCK, J. O.; MORRIS, M. R. & WILSON, A. D.: User-defined Gestures for Surface Computing. In: *ACM*. : *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*., 2009, S. 1083—1092. Online unter: http://dl.acm.org/citation.cfm?id=1518866 Abruf: 2016-03-22

[Zheng 2009] ZHENG, Y.-T.; ZHAO, M.; SONG, Y.; ADAM, H.; BUDDEMEIER, U.; BISSACCO, A.; BRUCHER, F.; CHUA, T.-S.; NEVEN, H. & YAGNIK, J.: *Tour the World: A Technical Demonstration of*

a Web-scale Landmark Recognition Engine. In: *ACM*. : *Proceedings of the 17th ACM International Conference on Multimedia*., 2009, S. 961—962. Online unter: http://dl.acm.org/citation.cfm?id=1631468 Abruf: 2016-03-22

[Zheng 2015] ZHENG, X. S.; FOUCAULT, C.; MATOS DA SILVA, P.; DASARI, S.; YANG, T. & GOOSE, S.: *Eye-Wearable Technology for Machine Maintenance: Effects of Display Position and Hands-free Operation*. In: *ACM*. : *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*., 2015, S. 2125—2134. Online unter: http://doi.acm.org/10.1145/2702123.2702305 Abruf: 2016-03-22

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, den _____ _____