



Universität Karlsruhe (TH)  
Forschungsuniversität • gegründet 1825

Institut für Technische Informatik (ITEC)  
Lehrstuhl Prof. Dr.-Ing. Rüdiger Dillmann

# Probabilistische Szenenanalyse mittels interaktiver Annotierung

Studienarbeit  
von

Stefan Kästle

10. Mai 2009 – 10. August 2009

verantw. Betreuer: Prof. Dr.-Ing. Rüdiger Dillmann  
betreuender Mitarbeiter: Dipl.-Inform. Alexander Kasper

Stefan Kästle  
Schüttestraße 11  
72417 Jungingen

Hiermit erkläre ich an Eides statt, dass ich die von mir vorgelegte Arbeit selbstständig verfasst habe, dass ich die verwendeten Quellen, Internet-Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Karlsruhe, den 10. August 2009

(Unterschrift)

---

Stefan Kästle

# Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b>	<b>5</b>
1.1	Zielsetzung . . . . .	5
<b>2</b>	<b>Szenen-Annotierung</b>	<b>7</b>
2.1	Einleitung . . . . .	7
2.2	Analyse vorhandener Arbeiten . . . . .	8
2.3	Objekte markieren . . . . .	9
2.4	Hilfsmittel zur Ableitung von Relationen . . . . .	10
<b>3</b>	<b>Relationen</b>	<b>13</b>
3.1	Einleitung . . . . .	13
3.2	Analyse vorhandener Arbeiten . . . . .	13
3.3	Definition des Bezugssystems in der Szene . . . . .	15
3.4	Referenz-Ebenen-Winkel . . . . .	15
3.5	Horizontaler Ebenen-Winkel . . . . .	16
3.6	Normierte und nicht-normierte Distanz . . . . .	17
3.7	Abgeleitete Relationen . . . . .	19
<b>4</b>	<b>Implementierung</b>	<b>21</b>
4.1	Einleitung . . . . .	21
4.2	Verwendete Programme und Bibliotheken . . . . .	21
4.3	Testräume für den Annotierungsprozess . . . . .	22
4.4	Annotierung der Objekte . . . . .	23
4.5	Ontologien als Speicher für Annotierungen . . . . .	24
4.6	Annotierungsdatenbank in SQLite . . . . .	25
4.7	Durchführung der Evaluierung . . . . .	28
4.8	Erweiterbarkeit der Relationen . . . . .	29
<b>5</b>	<b>Schlussfolgerung</b>	<b>31</b>
5.1	Bewertung der Arbeit . . . . .	31
5.1.1	Verwendung von SQLite als Datenspeicher . . . . .	31
5.1.2	Bewertung Referenz-Ebenen-Winkel . . . . .	31
5.1.3	Bewertung Horizontaler Ebenen-Winkel . . . . .	33
5.1.4	Bewertung der Gültigkeitsanalyse anhand der Varianz . . . . .	34
5.2	Ausblick . . . . .	37
5.3	Fazit . . . . .	38
	<b>Abbildungsverzeichnis</b>	<b>39</b>



# Kapitel 1

## Einführung

Das Forschungsgebiet der humanoiden Roboter gewinnt in jüngster Zeit immer mehr an Bedeutung. Dabei geht es darum einen Roboter zu konstruieren, der in gewöhnlichen Haushalts-Umgebungen autonom und zusammen mit Menschen agieren kann. Er soll seine Umgebung selbstständig wahrnehmen und interpretieren können. Außerdem soll er in der Lage sein mit Menschen in Kommunikation über diese Umgebung zu treten. Um Verständnis über seine Umgebung erlangen zu können ist eine große Wissensdatenbank erforderlich, die in weiten Teilen bereits a-priori aufgebaut werden kann. Diese Datenbank muss systematisch erzeugtes Wissen über Objekte in der Umgebung des Roboters und deren Verwendung enthalten. Das Wissen muss so allgemein in der Datenbank abgelegt werden, dass der Roboter in verschiedenen Räumlichkeiten gleicher Art arbeiten kann. Möglichst wenig soll von Hand kodiert werden. Da nur der Mensch über dieses Wissen verfügt müssen eine Vielzahl von Szenen einer bestimmten Art manuell analysiert werden, um durch spätere Aggregation der Einzelergebnisse ein allgemeines Wissen über die Zusammenhänge einzelner Objekte in verschiedene Szenen gewinnen zu können.

### 1.1 Zielsetzung

Thema dieser Studienarbeit ist der Entwurf eines Programms zur probabilistischen Analyse bestimmter Szenen. Schwerpunkt dieser Analyse ist die räumliche Verteilung bestimmter vordefinierter Objekte in diesen Szenen beziehungsweise die Wahrscheinlichkeit dafür, dass ein bestimmtes Objekt in einer Szene überhaupt vorkommt. Zur Gewinnung dieser Informationen muss eine große Anzahl von Szenen gleicher Art annotiert werden, das heißt die Objekte müssen innerhalb der einzelnen Szenen gefunden und zugeordnet werden.

Die untersuchten Objekte können verschiedenster Art sein. Denkbar in Haushalts-Umgebungen wären beispielsweise Möbel und verschiedene Gebrauchsgegenstände. Je nach Qualität der vorhandenen 3D-Modelle der Szenen schwankt die Anzahl annotierbarer Objekte. Bei automatisch durch Laserscanner erfassten Räumlichkeiten ist die Qualität deutlich schlechter als bei selbst modellierten 3D-Szenen. Es ist dann kaum möglich Objekte wie Tassen, Besteck oder Bücher richtig zu erkennen. Ist die Aufnahme-Qualität der Szenen aber hinreichend gut spricht Nichts gegen die Annotierung von derart kleinen Objekten. Um eine möglichst große Zuverlässigkeit der gewonnenen Informationen zu erhalten findet die Annotierung von Hand



Abbildung 1.1: Vollständig annotierte Küchen-Szene

statt. Von einer automatischen Analyse der Szenen wird hier also abgesehen.

Nach der Annotierung aller definierten Objekte werden die räumlichen Relationen zwischen diesen Objekten untersucht. Kapitel 3 befasst sich deshalb mit der Frage wie diese Relationen definiert werden können, um möglichst viel Spielraum für die spätere Auswertung der Relationen aller annotierten Räume zu haben.

Der spätere Nutzen der kumulierten Ergebnisse liegt in der Verwendung durch einen humanoiden Roboter. Dieser kann die Informationen zur Klassifizierung von Szenen und Objekten in Haushalts-Umgebungen einsetzen. Erkennt der Roboter eine bestimmte Anzahl von Objekten kann er daraus eventuell auf die Art der Szene schließen. Ist die Szene bekannt und eine hinreichend große Anzahl von Objekten bereits klassifiziert, können weitere Objekte auf Grund der gewonnenen relativen Positionierungen unter Umständen ebenfalls mit einer gewissen Wahrscheinlichkeit korrekt klassifiziert werden. Programme zur Verwendung der Ergebnisse zu implementieren ist allerdings nicht Teil dieser Studienarbeit.

# Kapitel 2

## Szenen-Annotierung

### 2.1 Einleitung

Die Szenen-Annotierung dient zur Analyse der zu untersuchenden 3D-Szenen. Die in den Szenen vorhandenen Objekte müssen markiert und den bereits bekannten Objekten in einer Objekt-Datenbank zugeordnet werden. Die Szenen-Analyse kann unter Umständen sehr komplex sein. Das liegt zum einen daran, dass sich selbst ähnliche Szenen teilweise stark unterscheiden und signifikante Unterschiede zu bereits untersuchten Szenen des gleichen Typs aufweisen können. Man denke hier beispielsweise an die Tatsache, dass in den Laboren und Pool-Räumen an der Universität Karlsruhe die Rechner oft auf den Tischen anstatt, wie sonst in Büro-Umgebungen üblich, unter den Tischen stehen. Andererseits sind die 3D-Szenen unter Umständen recht ungenau, sodass selbst der Mensch mit seinen stark ausgeprägten Fähigkeiten für die Objekt-Erkennung oft Schwierigkeiten bei der Zuordnung hat. Aus diesem Grund scheint eine automatische Klassifikation der Objekte in der Szene ungeeignet. Ziel dieses Kapitels ist es deshalb einen Weg zur effizienten manuellen Szenen-Annotierung durch den Menschen unter Einsatz seines Wissens und seiner stark ausgeprägten perzeptionellen Fähigkeiten zu entwickeln.

Um später eine statistische Aussage über die Objektanordnung treffen zu können, müssen viele gleichartige Räume annotiert werden. So können Regelmäßigkeiten in der relativen Positionierung zwischen Objekten gleichen Typs in diesen Räumen gefunden werden.

Der Benutzer betrachtet die zu untersuchende Szene also und findet die Objekte, die für die jeweilige Anwendung von Interesse sind. Es werden dann die zugehörigen Bereiche der 3D-Szene markiert und einer Liste bekannter Objekte zugeordnet. Eine quantitative Analyse der Szene ist für die spätere statistische Analyse gut geeignet und reduziert den Aufwand der Annotierung drastisch. Nachdem die Annotierung für viele gleichartige Räume durchgeführt wurde, können die Relationen zwischen je zwei Objekten verglichen werden. Dann findet eine wahrscheinlichkeitstheoretische Analyse der Menge von Ergebnissen für eine bestimmte Relation zwischen Objekten statt.

Die Beschreibung der räumlichen Relationen zwischen Objekten soll in möglichst menschenähnlicher Weise durchgeführt werden. Deshalb erscheint eine Abbildung der 3D-Szene in zweidimensionale Sichtbereiche sinnvoll. Viele linguistische Ausdrücke zur Objektpositionierung beziehen sich direkt auf das aktuelle menschliche Sichtfeld. Dieses entspricht ebenfalls

einer Abbildung der Realität in ein zweidimensionales Bezugssystem. Dafür muss eine analoge Art der Annotierung gefunden werden.

## 2.2 Analyse vorhandener Arbeiten

Das in Bezug auf die Annotierung vorherrschende Thema in der Literatur ist das Labeling von Bildern. Dabei werden oft zweidimensionale Bounding-Boxen zur Markierung von Objekten verwendet. Eine weitere Vereinfachung kann vorgenommen werden, wenn anstatt frei platzierbarer Boxen ein Raster verwendet wird. Die Position eines Objektes wird dann als Menge der geschnittenen Raster-Zellen angegeben. Eine weitere Form der Annotierung könnten einfache eindimensionale Striche zur Angabe einer Richtung und durch die Länge des Striches auch der Skalierung in dieser Richtung darstellen. Ein wichtiges Werkzeug zur Unterscheidung verschiedener Annotierungen kann die Verwendung unterschiedlicher Farben sein. [Divvala 09]

Jüngste Trends im Internet erfordern häufig eine Zusammenarbeit vieler räumlich verteilter Menschen. Oft gibt es keine Möglichkeit sich an einem gemeinsamen Ort zu treffen. Es muss deshalb eine Möglichkeit gefunden werden, gemeinsam an Projekten zu arbeiten. In diesem Zusammenhang gibt es neue Ansätze, wie eine Annotierung in 3D-Szenen durchgeführt werden kann. Dies ist zum Beispiel in der Architektur oder für andere Modellierungsaufgaben nötig. „Space Pen“ ist ein solches Annotierungswerkzeug. Unterstützt werden Text-Label im 3D-Raum und das direkte Zeichnen innerhalb des 3D-Models durch eine Art Gesten-Erkennung. Dazu wird eine transparente Ebene im 3D-Raum erstellt. Auf dieser Ebene kann dann, wie in Bildbearbeitungsprogrammen üblich, gezeichnet werden. Die Text-Labels werden an der gewünschten Stelle in der 3D-Szene „angeheftet“ oder ebenfalls auf einer temporären transparenten Ebene eingefügt. Zum Setzen der Annotierung bewegt sich die Kamera ähnlich zur Ich-Perspektive eines realen Betrachters in der Szene. Diese Art der Bewegung bewährt sich seit vielen Jahren in der Welt der Ego-Shooter Spiele [Jung 02] [Jung 99].

In [Jung 99] werden ebenfalls Annotierungen in 3D-Szenen vorgenommen. Dort werden allerdings nur einzelne Punkte in der Szene gesetzt und mit einer Text-Information versehen. Es werden also keine Informationen über die Abmessung und Orientierung der annotierten Objekte geliefert. Zur Unterscheidung verschiedener Annotierungen werden ebenfalls Farben verwendet.

Weitere Konzepte zur Annotierung finden sich in [Kubat 07]. Dort soll das Erlernen der menschlichen Sprache eines Kleinkindes analysiert werden. Mehrere Tausend Stunden an Audio- und Video-Material werden aufgenommen und müssen im Anschluss annotiert werden. Die Videos werden von Fischaugen-Kameras aufgenommen. In diesen Videos müssen im wesentlichen die Köpfe der in der aufgenommenen Szene befindlichen Personen markiert werden. Wichtig ist die Zuordnung der Person zu einer bekannten Identität, die Position der Person im Raum und die ungefähre Ausrichtung des Kopfes. Die Annotierung muss sehr schnell durchgeführt werden können, da eine Vielzahl von Szenen von Hand annotiert werden müssen. Dazu werden zweidimensionale Bounding-Boxen um die Köpfe der Menschen in einer Szene gesetzt. Danach findet eine Zuordnungen der Personen statt. Alle anderen Informationen werden automatisch gewonnen um die benötigte Zeit für den manuellen Teil der Annotierung zu minimieren.



## 2.3 Objekte markieren

Um das Markieren von Objekten zu erleichtern muss eine einfache Navigation in der 3D-Szene möglich sein. Vorstellbar wäre eine menschenähnliche Bewegung innerhalb des Raumes. Die Bewegung würde sich dann auf die x- und z-Richtung beschränken, das heißt, der Abstand zum Boden würde sich nicht ändern. Der Blick auf die Szene erfolgt dann immer aus einer Höhe, die der Augenhöhe eines stehenden Menschen entspricht. Das hat sich auch in zahlreichen First-Person Ego-Shootern bewährt. Da zur Annotierung eines Objektes aber Punkte auf mehreren Seiten markiert werden müssen, ist es unbedingt erforderlich schnell um ein Objekt drehen zu können. Das ist aus der First-Person Perspektive nicht möglich. Außerdem müssen auch Punkte markiert werden, die eventuell gar nicht durch eine solche Bewegung anvisiert werden können. Dazu zählt beispielsweise die Rückseite eines Kühlschranks. Eine völlig freie Bewegung innerhalb der Szene scheint deshalb besser geeignet. Die Position der Kamera kann dabei in alle drei Richtungen verschoben werden. Dazu werden vier Tasten auf der Tastatur verwendet. Außerdem muss die Kamera drehbar sein. Dazu wird die Maus verwendet.

Um die Anforderungen an den Annotierungsprozess möglichst gering zu halten sollten keine zusätzlichen Hardware-Komponenten zur Markierung von Objekten benötigt werden. Die Maus ist dafür also ein geeignetes und dem Benutzer wohl bekanntes Werkzeug. Mit ihr lassen sich Punkte innerhalb des aktuellen Sichtbereiches der 3D-Ansicht bequem anklicken. Ein einzelner Punkt auf der Oberfläche des Objektes als Annotierung reicht nicht aus, da dieses Vorgehen keine Informationen über die Größe eines Objektes liefert. Es werden deshalb nach und nach immer mehr Punkte auf der Oberfläche des zu markierenden Objektes hinzugefügt. Um die benötigten Punkte auf der Objektoberfläche möglichst gering zu halten wird rekursiv eine *Axis-Aligned Bounding-Box* aufgebaut. Das ist eine Box, deren Seiten jeweils parallel zu einer der Achsen des Weltkoordinatensystems der Szene liegen. Die Box wird in jedem Schritt so vergrößert, dass alle bisher markierten Punkte innerhalb dieser Bounding-Box liegen. Es werden so lange weitere Punkte markiert bis das gesamte Objekt innerhalb der Box liegt. Diese quantitative Art der Annotierung ist für den Benutzer schnell und mit wenig Aufwand durchzuführen. Das ist insbesondere deshalb wichtig, weil eine sehr große Anzahl von Annotierungen durchgeführt werden muss. Eine effiziente Durchführung dieser Arbeit ist deshalb unverzichtbar. Außerdem sind die Algorithmen zur Analyse bei Axis-Aligned Bounding-Boxen sehr effizient. Eine qualitative Annotierung würde sowohl die Annotierung als auch die spätere Analyse der Annotierungen deutlich erschweren und das Ergebnis der statistischen Analyse höchstens unwesentlich verbessern.

Die Verwendung von Axis-Aligned Bounding-Boxen funktioniert allerdings nur dann gut, wenn die zu annotierenden Objekte ebenfalls Axis-Aligned sind. Abbildung 2.1 zeigt die in rosa dargestellte Bounding-Box um einen Tisch im Falle eines optimal gedrehten Raumes. Die Bounding-Box des Tisches im um  $45^\circ$  gedrehten Fall fällt deutlich ungenauer aus. Dies ist in Abbildung 2.2 dargestellt. Das Problem lässt sich in vielen Fällen aber leicht dadurch lösen, dass der Raum bei der Vorverarbeitung in einem 3D Rendering Programm so gedreht wird, dass die gewünschte Referenz-Ebene etwa orthogonal zur x- oder z-Achse ist. Da alle interessanten Objekte in der Nähe dieser Ebene liegen und in Haushalts-Umgebungen eine Drehung von Gegenständen gegenüber den Gegenständen in der Nähe eher selten auftritt, sollte eine solche optimale Rotation des Raumes in den meisten Fällen problemlos gefunden werden. Ein Beispiel dazu bietet Abbildung 2.3. Dort sind die zu untersuchenden Objekte alle in der Referenz-Ebene und nicht gegeneinander gedreht. Aber selbst, wenn die Bounding-

Box im Verhältnis zur tatsächlichen Ausrichtung des Objektes gedreht ist, reicht das zur Einschätzung der Lage und Größe eines Objektes aus. Außerdem wird bei der Weiterverarbeitung der annotierten Daten hauptsächlich der Mittelpunkt der Bounding-Box verwendet, was die Folgen einer gegenüber dem Koordinatensystem der Szene stark gedrehten Bounding-Box nochmals drastisch abschwächt. Der Mittelpunkt lässt sich dabei kaum anders markieren als durch das sukzessive Hinzufügen von Punkten auf der Oberfläche eines Objektes, da er selbst ja verdeckt ist.

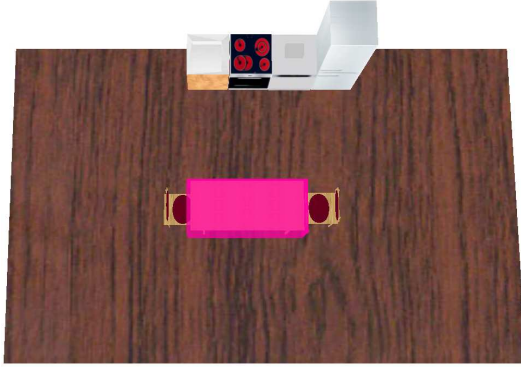


Abbildung 2.1: Axis-Aligned Bounding-Box eines Tisches bei optimal gedrehtem Raum

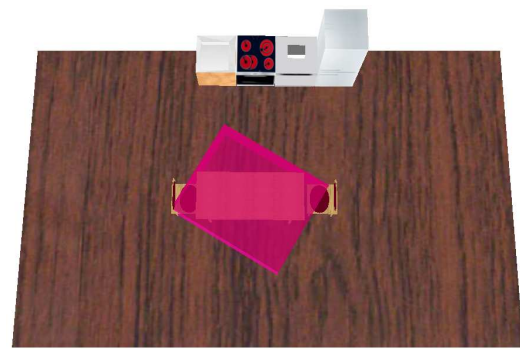


Abbildung 2.2: Axis-Aligned Bounding-Box eines Tisches im um 45° gedrehten Raum

Zur Unterscheidung verschiedener Klassifizierungen von Objekten werden mehrere Farben zur Darstellung der Bounding-Boxen verwendet. So hat das Referenzobjekt zum Beispiel eine andere Farbe als die anderen annotierten Objekte. In Abbildung 2.3 ist das Referenz-Objekt zum Beispiel in orange und alle anderen annotierten Objekte in rosa dargestellt.

## 2.4 Hilfsmittel zur Ableitung von Relationen

Zur Abbildung der dreidimensionalen Szene in ein zweidimensionales Bezugssystem wird ein *Referenzobjekt* und einen Bezugspunkt auf dem Boden definiert. Dieses Bezugssystem heißt im Folgenden *Viewport* oder kurz *VP*, der Bezugspunkt auf dem Boden wird mit *VPP* bezeichnet. Es wird dann ein Vektor  $\vec{v}$  als Verbindungsvektor des VPP mit der Mitte des Referenzobjektes definiert. Später wird auch noch eine Ebene, die *reference\_plane*, deren Normale mit  $\vec{v}$  bezeichnet wird, benötigt. Die Ebene wird so platziert, dass sich der Mittelpunkt des Referenzobjektes in ihr befindet.  $\vec{v}$  entspricht dem Vektor  $\vec{v}$  mit auf 0 gesetzter y-Komponente.  $\vec{v}$  liegt also parallel zur x/z-Ebene und damit steht die *reference\_plane* orthogonal auf die x/z-Ebene. Die eingeführten Hilfsmittel sind in Abbildung 2.3 dargestellt.



Abbildung 2.3: Überblick über eine vollständig annotierte Szene: Die grüne Ebene ist die `reference_plane`; bei dem rosa Punkt im Vordergrund handelt es sich um den VPP; das Referenzobjekt wird durch eine orange Bounding-Box markiert, die rosa Bounding-Boxen stellen die anderen annotierten Objekte dar.



# Kapitel 3

## Relationen

### 3.1 Einleitung

Um die räumliche Positionierung zwischen zwei Objekten zu untersuchen muss zunächst eine Menge von Basis-Relationen definiert werden. Die Basis-Relationen sollten so gewählt werden, dass sie *alle* Informationen der relativen Objektpositionierung enthalten, so dass später beliebige andere Relationen aus diesen Basis-Relationen aufgebaut werden können. Trotzdem sollte ihre Darstellung möglichst einfach und intuitiv sein. Das hat den Vorteil, dass die Relationen möglichst gut an die jeweilige Situation und die damit verbundenen Anforderungen angepasst werden können. Es sollte also möglich sein die neu aufgebauten Relationen als Kombination vorhandener Relationen zu beschreiben.

Es fällt zunächst schwer im menschlichen Sprachgebrauch Ausdrücke zur Beschreibung von dreidimensionalen Relationen zu finden. Viel häufiger stößt man auf Ausdrücke, die eine Positionierung innerhalb des menschlichen Sichtfeldes beschreiben. Relationen wie zum Beispiel „rechts von“, „über“ oder „neben“ basieren auf dem aktuellen Sichtfeld des Betrachters. Dieses Sichtfeld ist eine Abbildung der dreidimensionalen Umgebung in eine zweidimensionale Momentaufnahme. Die Abbildung lässt sich also mit einem Foto der Szene aus Sicht des Betrachters vergleichen. Um zusätzlich Begriffe wie „hinter“ oder „davor“ zu unterstützen wird ein weiteres Bezugssystem benötigt. Dieses sollte möglichst orthogonal zu dem bereits eingeführten sein. So werden möglichst gering korrelierte Ergebnisse erzeugt und die Redundanz der beiden Werte wird auf ein Minimum reduziert.

### 3.2 Analyse vorhandener Arbeiten

Die meisten untersuchten Arbeiten beziehen sich auf die Definition zweidimensionaler diskreter räumlicher Relationen. [Clementini 97] schlägt zum Beispiel eine Klassifizierung der Entfernungen nach Zugehörigkeit zu einem bestimmten Intervall vor. Ist der Abstand zweier Objekte beispielsweise 3,95 Größeneinheiten, also zwischen 3 und 4 Größeneinheiten, wird die Relation beispielsweise als „nah“ klassifiziert, bei einem Abstand von 4,05 Größeneinheiten könnte aber bereits eine Klassifizierung als „mittelweit“ erfolgen. Egal wie feingranular eine solche Klassifizierung definiert wird, es bilden sich immer scharfe Kanten zwischen den verschiedenen Klassen. Solche scharfen Kanten sind natürlich problematisch, da zwei ähnliche

Relationen auf zwei unterschiedliche Klassen abgebildet werden könnten. Dagegen ist es möglich, dass bei eher unterschiedlichen Werten eine Abbildung auf dieselbe Klasse erfolgen kann. Eine probabilistische Zuordnung wäre deshalb wünschenswert. Die Relation zwischen zwei Objekten wird mit einer bestimmten Wahrscheinlichkeit als „nah“ klassifiziert und ebenfalls mit einer anderen Wahrscheinlichkeit als „mittelweit“. Das selbe Objekt kann also gleichzeitig mehrere Relationen unterschiedlich gut erfüllen.<sup>1</sup>

Ein weiterer Grund von einer zu exakten Zuordnung abzusehen ist die Qualität der 3D-Modelle. Die Ableitung exakter Informationen aus einem ungenauen Modell verfälscht die Qualität der gewonnenen Ergebnisse. Des Weiteren bringen exakte Informationen oft keinen Vorteil. Ein Beispiel ist die Größe von Alaska: obwohl  $1518800 \text{ km}^2$  die exakte Größe von Alaska ist, bringt diese Information in vielen Fällen keinen nennenswerten Informationsgewinn. Viel interessanter könnte zum Beispiel die *ungefähre, relative* Größe zu etwas Bekanntem sein: Alaska ist größer als alle Staaten der amerikanischen Ostküste von Maine bis Florida zusammen [Clementini 97].

Aus diesem Grund wird die Abbildung der kumulierten Werte auf eher wenige Klassen erfolgen. Diese sind sehr stabil, da eine große Anzahl von Annotierungen durchgeführt wurden um den kumulierten Wert zu erhalten. Wenn die untersuchten Objekte tatsächlich einer Relation unterliegen<sup>2</sup>, kann und muss eine Abbildung auf Klassen stattfinden. Der Roboter kann mit dem exakten Mittelwert einer Relation nämlich wenig anfangen. Viel besser eignet sich hier eine Unterscheidung anhand weniger möglicher Zustände. Diese sind für den Roboter leichter in der Realität zu unterscheiden und geben einen guten Eindruck der Verteilung der Objekte in *allen* Räumen eines bestimmten Typs.

Allerdings sind räumliche Relationen zwischen Objekten nicht ausreichend um eine Szene vollständig zu beschreiben [Clementini 97]. Es muss zusätzlich die Größe der Objekte und gegebenenfalls auch die Szene beziehungsweise der Kontext um die jeweiligen Objekte berücksichtigt werden. Man würde den Abstand zwischen Fahrrad und Mülleimer von zwei Metern in einem vier Quadratmeter großen Keller sicher anders beschreiben als auf dem Karlsruher Marktplatz. Die Unterschiede in der Beschreibung zwischen Esszimmer und Bad sind dagegen eher minimal. Eine Berücksichtigung der Szene beziehungsweise des Kontextes bringt für diese Aufgabenstellung also keine nennenswerten zusätzlichen Informationen. Die betrachteten Räume in einem durchschnittlichen Haus sind nämlich alle ähnlich groß. Trotzdem ist für den Abstand zwischen zwei Objekten die Objektgröße möglicherweise interessant. Die Relationen sind dann aber nicht mehr zwangsläufig symmetrisch.

[Clementini 97] macht außerdem Vorschläge zur Definition eines Bezugssystems ähnlich zu dem in der menschlichen Vorstellung. Die vertikale Achse wird durch die Erdanziehung bestimmt. Daraus folgen bereits die Begriffe „darüber“ und „darunter“. Um die Begriffe „vorne“ und „hinten“ bilden zu können wird der menschliche Körperbau herangezogen. Die Blickrichtung ist „vorne“. Die Bezeichnungen „rechts“ und „links“ leiten sich aus der Blickrichtung ab.

---

<sup>1</sup>Man spricht in diesem Zusammenhang von Fuzzy-Logik [wikipedia.de 09]

<sup>2</sup>Das ist dann der Fall, wenn die Einzelergebnisse nicht zu stark streuen

### 3.3 Definition des Bezugssystems in der Szene

Da ein Bezugssystem in einem dreidimensionalen Modell nicht implizit gegeben ist, muss es explizit fest gelegt werden. Die vertikale Achse lässt sich analog zu den Vorschlägen in [Clementini 97] als Erdanziehungskraft definieren. Das stellt natürlich Ansprüche an die Orientierung im 3D-Modell. Vorstellbar ist zum Beispiel, dass die Erdanziehungskraft immer in die negative  $y$ -Richtung zeigt. Daraus folgt dann, dass die erste Achse in  $y$ -Richtung des 3D Modells zeigt. Wäre das nicht gegeben müsste eine manuelle Festlegung dieser Achse im eigentlichen Programm erfolgen.

In den Szenen befindet sich zunächst kein Betrachter und damit kein Sichtfeld. Allerdings lässt sich über die aktuelle Ansicht in der 3D-Szene ein künstlicher Betrachter simulieren. Wie in Kapitel 2.4 beschrieben ist durch den Verbindungsvektor zwischen VPP und Mittelpunkt des Referenzobjektes eine Blickrichtung vorgegeben. Diese Blickrichtung wird aus Gründen der Einfachheit so normiert, dass die  $y$ -Komponente des resultierenden Vektors 0 ist. Daraus folgt dann die zweite Achse parallel zur  $x/z$ -Ebene und damit orthogonal zur bereits vorhandenen Achse durch die Erdanziehungskraft. Diese zweite Achse gibt also eine Blickrichtung vor.

Wie in Kapitel 3.2 beschrieben wird die dritte Achse orthogonal zu den bereits vorhanden gewählt. Damit ist das Koordinatensystem orthogonal. Wie bereits besprochen wird die Skalierung der Szene nicht explizit berücksichtigt, da die untersuchten Räume vergleichsweise ähnliche Größen haben.

### 3.4 Referenz-Ebenen-Winkel

Zur Definition des Referenz-Ebenen-Winkels wird die `reference_plane` aus Kapitel 2.4 benötigt. Es soll die Relation zwischen den Objekten  $a$  und  $b$  untersucht werden. Zunächst wird  $\vec{o}$  definiert als  $b_{\text{Mitte}} - a_{\text{Mitte}}$ . Dieser Vektor wird in die in den Ursprung verschobenen `reference_plane` projiziert und anschließend der Winkel im Uhrzeigersinn zum Einheitsvektor in  $y$ -Richtung bestimmt.

$$\vec{g} = \vec{o} + \lambda \vec{n}$$

Dabei ist  $\vec{g}$  die Gerade durch  $\vec{o}$  parallel zur Normalen der `reference_plane`  $\vec{n}$  (dabei:  $\vec{n} \neq \vec{0}$ ). Die Ebenen-Gleichung der in den Ursprung verschobenen `reference_plane`<sup>3</sup> ist:

$$\vec{n}\vec{x} = 0$$

Einsetzen von  $\vec{g}$  in die Ebenen-Gleichung und auflösen nach  $\lambda$  ergibt:

$$\vec{n}\vec{g} = \vec{n}(\vec{o} + \lambda \vec{n}) = 0$$

$$\lambda = \frac{-\vec{n}\vec{o}}{\vec{n}\vec{n}}$$

Der in die Ebene projizierte Vektor  $\vec{o}_{\text{II}}$  ist also:

---

<sup>3</sup>Im Gegensatz zur tatsächlichen `reference_plane`, die den Mittelpunkt des Referenz-Objektes enthält

$$\vec{o}_{\Pi} = \vec{o} + \frac{-\vec{n}\vec{o}}{\vec{n}\vec{n}} \vec{n}$$

Man beachte, dass  $\vec{n}\vec{o}$  und  $\vec{n}\vec{n}$  Skalarprodukte sind und reellen Zahlen entsprechen. Ein Vektorraum definiert mit dem Skalarprodukt keinen Körper. Eine weitere Vereinfachung des Terms ist daher nicht möglich.

In Abbildung 3.1 ist die Berechnung des Referenz-Ebenen-Winkels dargestellt. Die dunkel-grüne Ebene ist die Referenz-Ebene, deren Normale  $\vec{n}$  als hell-grüne Gerade dargestellt wird. Der rote Vektor ist der Verbindungsvektor  $\vec{o}$  zwischen den beiden zu untersuchenden Objekten. Dieser wird nun in die Referenz-Ebene projiziert. Es entsteht  $\vec{o}_{\Pi}$ . Dieser ist im Bild rosa dargestellt.

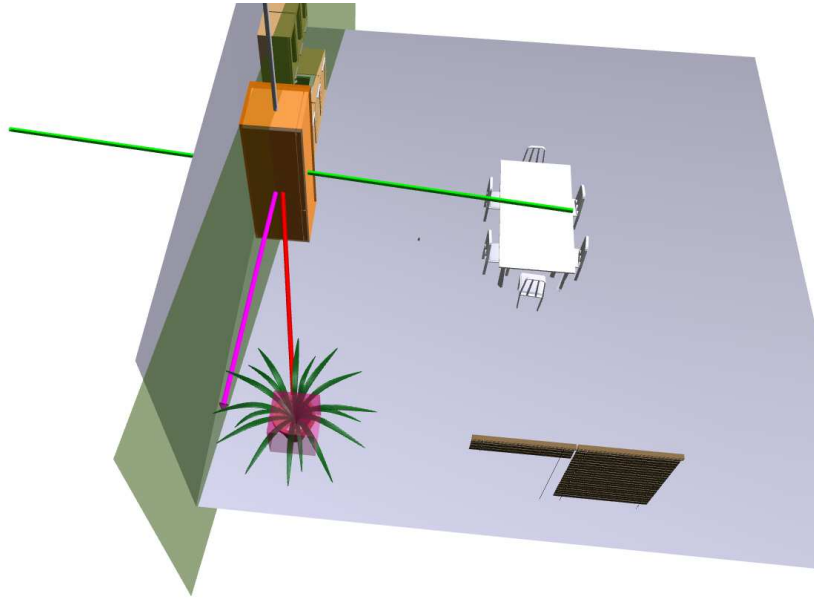


Abbildung 3.1: Diese Abbildung stellt die Berechnung des Referenz-Ebenen-Winkels dar. Aus Gründen der besseren Darstellung wurde die Szene verschoben, so dass der Ursprung des Koordinatensystems der Mittelpunkt des orange dargestellten Objektes  $a$  ist.

### 3.5 Horizontaler Ebenen-Winkel

Die „horizontale Ebene“ wird so definiert, dass sie parallel zur x/z-Ebene liegt. Eine mögliche Normale ist also  $(0, 1, 0)$ . Die Ebene soll den Ursprung enthalten. Der Vektor  $\vec{o}$  wird wieder definiert als  $\vec{b}_{\text{Mitte}} - \vec{a}_{\text{Mitte}}$ , wobei  $\vec{b}_{\text{Mitte}}$  dem Mittelpunkt des Secondary-Object und  $\vec{a}_{\text{Mitte}}$  dem Mittelpunkt des Primary-Objects entspricht. Analog zu Kapitel 3.4 wird eine Projektion  $\vec{o}_{\pi}$  dieses Vektors in die horizontale Ebene bestimmt. In diesem Fall genügt es natürlich die y-Koordinate des Vektors  $\vec{o}$  auf 0 zu setzen um  $\vec{o}_{\pi}$  zu erhalten.



In Abbildung 3.2 ist die Berechnung des Winkels dargestellt. Der Kühlschrank ist das Primary-Object. Das zu betrachtende Secondary-Object ist die Blumenvase. Der Verbindungsvektor  $\vec{o}$  ist rot dargestellt. Er soll nun in die grün dargestellte horizontale Ebene projiziert werden. Das Ergebnis ist der rosa Vektor  $\vec{o}_\pi$ . Er liegt in der Ebene.

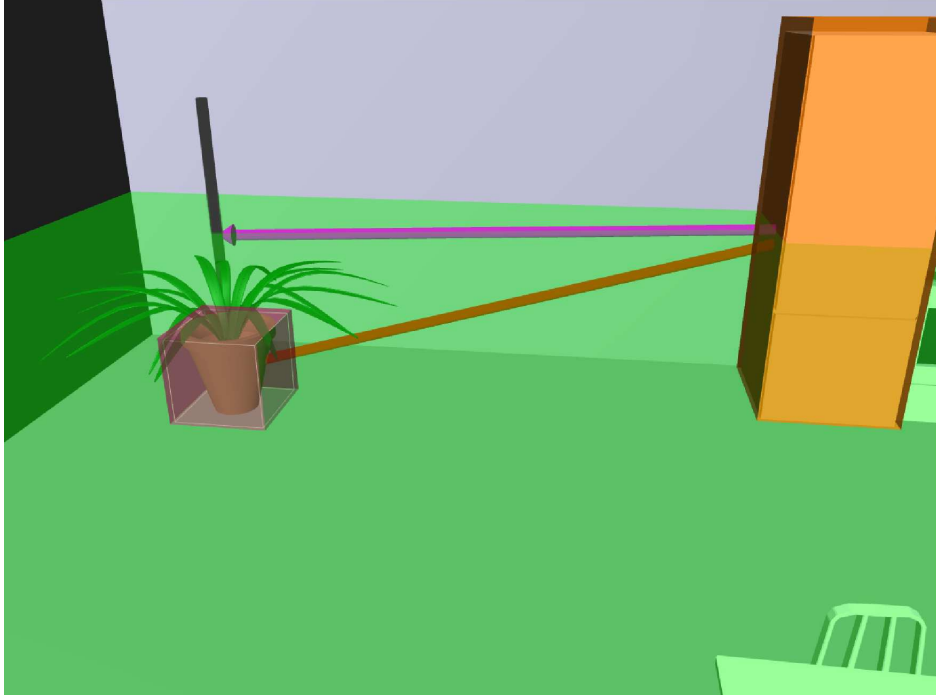


Abbildung 3.2: Darstellung der Berechnung des horizontalen Ebenen-Winkels. Die Szene wurde hier so verschoben, dass der Ursprung des Koordinatensystems im Mittelpunkt des Primary Objects liegt.

Um dem Vektor  $\vec{o}_\pi$  nun einen Winkel zuzuweisen wird eine Referenzrichtung benötigt. Diese entspricht der Normalen der `reference_plane`. Betrachtet man den Raum nun von oben wird der Winkel zwischen der Referenzrichtung und dem soeben bestimmten Vektor  $\vec{o}_\pi$  im Uhrzeigersinn bestimmt.

### 3.6 Normierte und nicht-normierte Distanz

Ein trivialer Ansatz zur Bestimmung des Abstands zwischen zwei Objekten ist die Berechnung der Länge des Verbindungsvektors der beiden Objektmittelpunkte. Dieses extrem einfache Vorgehen berücksichtigt aber nicht die Ausmaße der einzelnen Bounding-Boxen um die Objekte. Berühren sich die Bounding-Boxen zweier Objekte kann deren Abstand mit dieser Berechnung trotzdem beliebige Werte annehmen, indem eine der Bounding-Boxen beliebig vergrößert wird.

Zur Berechnung des *exakten* Abstandes zwischen zwei Axis-Aligned Bounding-Boxen kann folgender Algorithmus verwendet werden:

Die Koordinaten der x-, y- und z-Richtung zweier Boxen werden jeweils getrennt voneinander untersucht. Ohne Beschränkung der Allgemeinheit wird im Folgenden die x-Richtung

untersucht. Es gibt zwei Möglichkeiten:

1. *Überlagerung*: In diesem Fall ist der Abstand in dieser Richtung  $d_x$  0. Die Richtung wird nicht weiter untersucht. Die jeweils anderen Komponenten - also y und z - werden als ein Problem der um eins niedrigeren Dimension weiter betrachtet
2. *keine Überlagerung*: Der Abstand in dieser Richtung  $d_x$  wird berechnet und für spätere Schritte aufgehoben.

Zuletzt muss die Länge des Vektors  $\vec{d} = (d_x, d_y, d_z)$  bestimmt werden. Das ist dann der Abstand der beiden Objekte.

Anschaulich bedeutet das:

Falls es in allen Richtungen Überlagerungen gibt berühren sich die Objekte und der Abstand zwischen ihnen ist 0.

Wird eine Überlagerung in *zwei* der drei übrig bleibenden Richtungen gefunden handelt es sich um ein eindimensionales Problem. Der Abstand der beiden Objekte ist dann gleich dem Abstand in der einzigen Richtung ohne Überlagerung. Es ist:  $d = d_x$

Gibt es eine Überlagerung in *nur einer der drei* Richtungen, findet eine Projektion der Objekte in den zweidimensionalen Unterraum der übrigen Richtungen statt. Das resultierende Problem des Abstands zwischen zwei Rechtecken im 2D-Raum ist einfach zu lösen. Die Lösung ist anschaulich die Länge der Diagonalen durch das Rechteck zwischen den beiden gegebenen Rechtecken, und das ist genau  $\sqrt{d_x^2 + d_y^2}$ .

Gibt es in keiner der Richtungen eine Überlagerung, findet sich eine Box die die Lücken zwischen den beiden vorhandenen Boxen auffüllt. Es muss dann wieder die Länge der Diagonalen durch diese Box gefunden werden. Die Diagonale ist dabei:  $\vec{d} = (d_x, d_y, d_z)$

Wird als Grundlage der Annotierung eine Bounding-Box verwendet, die nicht parallel zu den Achsen liegt, kann das Verfahren in [Meyer 86] zur Berechnung verwendet werden.

Der tatsächliche Abstand zweier Objekte ist für den linguistischen Gebrauch aber nicht ausreichend. [Clementini 97] fordert folgende weitere Informationen zur Abbildung auf linguistische Begriffe:

**Szene** Wie bereits besprochen ist das Ausmaß der Szene weniger relevant wie beispielsweise beim Vergleich einer Szene auf dem Karlsruher Marktplatz mit einer Szene in einem kleinen Kellerraum. Eine Berücksichtigung des Ausmaßes der Szene scheint also vernachlässigbar.

**Objektgröße** In den beiden Aussagen "die Bleistifte liegen *nah* beieinander" und "der Stuhl steht *nah* am Tisch" hat der Ausdruck *nah* eine jeweils andere Dimensionierung. Liegen zwei Stifte 50 Zentimeter auseinander würde man vermutlich nicht zwangsläufig von nah reden, bei Stuhl und Tisch schon eher. Die linguistische Begriffsbildung scheint also auch von der Objektgröße abzuhängen.

Aus diesem Grund wird nach der Annotierung ein zweiter, normierter Abstand zwischen den Objekten gespeichert. Es ist bemerkenswert, dass dieser normierte Abstand *keine* symmetrische Relation zwischen zwei Objekten darstellt.

**distance** Der tatsächliche Abstand zwischen zwei Objekten. Dieser kann je nach Skalierung des Raumes und jeweiligen Objektgrößen stark variieren.

**distance\_normalized** Der normierte Abstand zwischen zwei Objekten. Zur Berechnung wird die *distance* durch das Ausmaß des Primary-Objects geteilt. Dieses entspricht der Länge der Diagonalen durch dessen Bounding-Box.

Durch den normierten Abstand werden zwei Fliegen mit einer Klappe geschlagen. Da zum Beispiel zwei Äpfel in zwei verschiedenen Szenen immer ähnlich groß sind, wird darüber auch die Skalierung der ganzen Szenen berücksichtigt.

### 3.7 Abgeleitete Relationen

Um einen möglichst großen Spielraum bei der Auswertung der annotierten Daten zu lassen wird die Ableitung weiterer Relationen aus den Basisrelationen zugelassen. Der Art der neuen Relationen sind dabei praktisch keine Grenzen gesetzt. Die Werte der Basisrelationen und auch anderer abgeleiteter Relationen können beliebig kombiniert werden. Ermöglicht wird das durch ein Konzept aus abstrakten Klassen in C++ und der Vererbung dieser Klassen. Eine detaillierte Übersicht zur tatsächlichen Implementierung gibt Abschnitt 4.8.

Ein Beispiel für eine abgeleitete Relation ist „In Front Of“. „In Front Of“ könnte als Tuple-Relation definiert werden, das heißt, sie kombiniert zwei Basis-Relationen. In diesem Fall wären „horizontal\_angle“ und „distance\_normalized“ vorstellbar. Eine Kombination mehrerer Basis-Relationen ist sinnvoll, weil der Wert einer Basis-Relation in vielen Fällen nicht genügend Informationen bietet. Dass ein Objekt vor einem anderen steht mag vielleicht nur dann eine Hilfe für den Roboter sein, wenn der Abstand der beiden Objekte einen bestimmten Wert nicht überschreitet. Denkbar wäre hier zum Beispiel folgendes: Ist so viel Platz zwischen zwei Objekten, dass der Roboter dazwischen hindurch fahren kann, soll nicht als „In Front Of“ klassifiziert werden.



# Kapitel 4

## Implementierung

### 4.1 Einleitung

Um die Annotierung benutzerfreundlich durchzuführen ist die dreidimensionale Darstellung des Raumes gut geeignet. Dort können Objekte dann beispielsweise durch einfaches Anklicken einiger Eckpunkte markiert werden. Das am Institut für Anthropomatik an der Universität Karlsruhe (TH) entwickelte OViSE Programm bietet bereits eine gute Grundlage zur Darstellung der Szene. Es verwendet dazu die mächtige Ogre3D Engine [Ogre3D 09b]. Außerdem ist dort bereits alles, was zur Bewegung im Raum benötigt wird, vorhanden. Eine Annotierung ist dort allerdings nicht vorgesehen und muss deshalb noch hinzugefügt werden. Trotzdem ist das OViSE Programm eine gute Grundlage für eine Annotierungsumgebung. Abbildung 4.1 zeigt das OViSE Programm in seiner unveränderten Form.

### 4.2 Verwendete Programme und Bibliotheken

Das OViSE-Programm wurde zur Darstellung wissenschaftlicher Inhalte mit einem Schwerpunkt auf der Szenen-Visualisierung für Roboter entwickelt. Eine grafische Oberfläche zum Aufruf der angebotenen Funktionalitäten ist bereits vorhanden. Einige Features von OViSE sind das dynamische Laden und Löschen von Mesh-Dateien, die Manipulation der verwendeten Materials, Visualisierung von sensorischen Daten und Unterstützung zum Einlesen vollständiger Szenen.

Die Darstellung der geladenen Mesh-Dateien wird mit einer 3D-Ansicht realisiert. Die Navigation innerhalb dieser Szenen ist bereits gut realisiert und bedarf keiner weiteren Verbesserung. Lediglich die Szenen-Annotierung muss noch nachgerüstet werden, da das OViSE Programm nur die Markierung der einzelnen geladenen 3D-Modelle als ein Ganzes, nicht aber bestimmter Teile innerhalb dieser Modelle erlaubt. Es werden außerdem weitere Funktionen benötigt. So muss das Setzen des VPP, des Referenz-Objekts und das Zuordnen eines markierten Objektes zu den Objekten der Objektdatenbank ermöglicht werden. Diese Funktionalität muss in die grafische Oberfläche des Programms eingebaut werden.

Als Datenspeicher für die annotierten Objekte und berechneten Relationen wird eine SQLite Datenbank verwendet. Die einzelnen SQLite Datenbanken werden in Abschnitt 4.6 im Detail

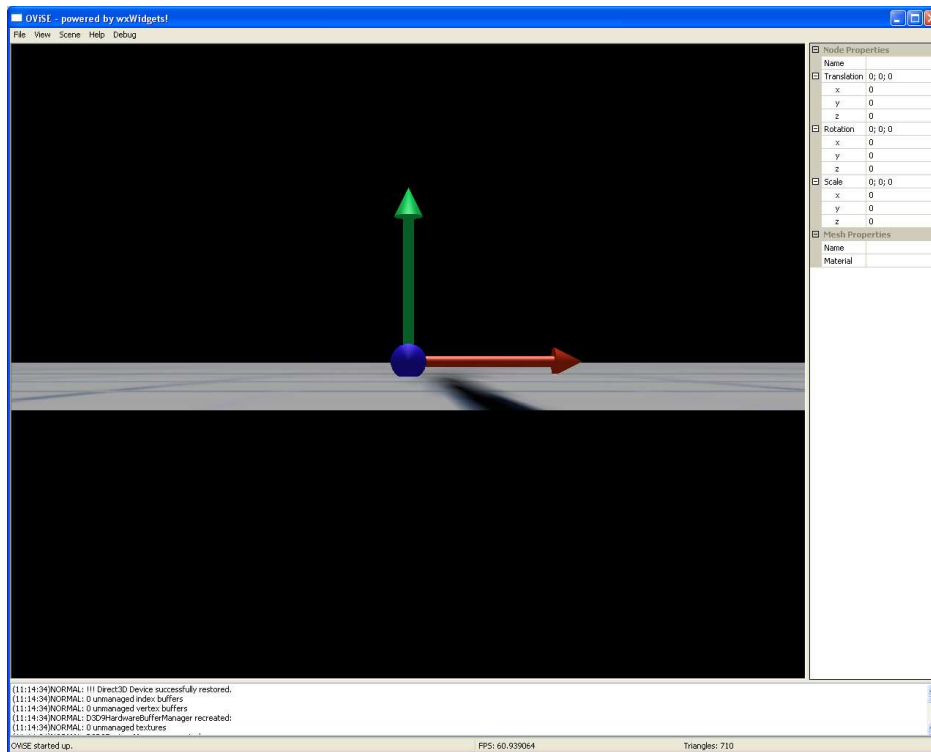


Abbildung 4.1: Screenshot des ursprünglichen OViSE Programms

erläutert. SQLite ist ein kleines und einfaches relationales Datenbanksystem. Da die Datenbanken in einer einzigen Datei abgespeichert werden und plattformunabhängig sind, eignet sich SQLite hervorragend als Grundlage für den Datenspeicher. Leider unterstützt SQLite keine referentielle Integrität. Da aber nur das modifizierte OViSE-Programm eine Manipulation der Daten in der Datenbank durchführt, ist die Korrektheit des Programms in Bezug auf die Integrität der Daten leicht von Hand zu prüfen. Es müssen dazu lediglich einige wenige Funktionen im OViSE-Programm auf Korrektheit überprüft werden.

### 4.3 Testräume für den Annotierungsprozess

Leider lagen zum Zeitpunkt der Studienarbeit noch keine automatisch durch einen Laser-Scanner aufgenommenen 3D-Szenen vor. Deshalb mussten im Rahmen der Studienarbeit noch einige Test-Räume konstruiert werden. Einer dieser Räume ist die Küche, in der der humanoide Roboter Armar entwickelt wird.

Um die Annotierung unter extremen Bedingungen zu testen wurden einige der Küchen-Szenen um  $30^\circ$  gedreht und nochmal annotiert. Es tritt dann das Problem auf, dass die Axis-Aligned Bounding-Boxen die annotierten Objekte nicht mehr perfekt umschließen. Die Basisversionen der Küchen sind aber so gedreht, dass die gewünschten Referenz-Ebenen etwa parallel zu der x- oder z-Achse sind. Außerdem wurde der Boden jeweils parallel zur x/z-Ebene festgelegt.

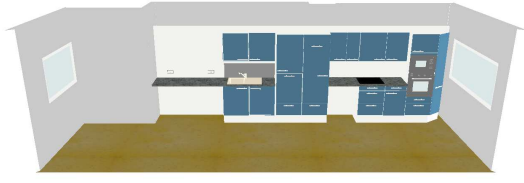


Abbildung 4.2: Diese Küche dient auch als Test-Umgebung für den Armar Roboter

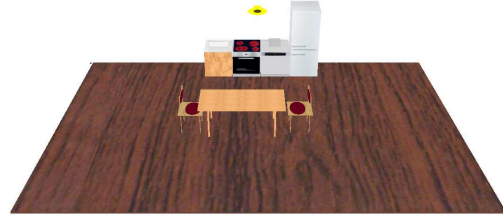


Abbildung 4.3: Eine einfache Küche mit Sitzgruppe

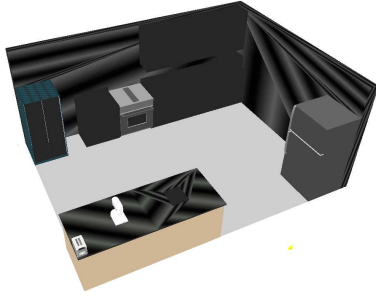


Abbildung 4.4: Diese Küche enthält die meisten Objekte, die von Anfang an in der Objektdatenbank vorhanden sind



Abbildung 4.5: Eine weitere Küche mit Sitzgruppe.

## 4.4 Annotierung der Objekte

Zur Annotierung von Objekten bieten sich Bounding-Boxen an. Um die späteren Berechnungen drastisch zu vereinfachen werden Axis-Aligned Bounding-Boxen verwendet.

Wie in Abschnitt 2.2 beschrieben muss das OViSE Programm erweitert werden um die Funktion zur Annotierung von Objekten anzubieten. Zur Markierung der Eckpunkte eines Objektes bietet sich ein Doppelklick mit der Maus an. Dieser wird in OViSE allerdings schon zum Markieren einzelner Objekte benutzt. Um zwischen dem klassischen Modus und der neuen Funktionalität zur Annotierung umzuschalten kann in der modifizierten Version deshalb die Taste **p** verwendet werden. Ist der Modus zur Annotierung von Objekten aktiv werden sukzessive Punkte des gewünschten Objektes markiert. Nach jedem neu markierten Punkt wird eine minimale Axis-Aligned Bounding-Box um das Objekt so berechnet, dass alle zur Zeit markierten Punkte innerhalb dieser Box liegen.

Die Formeln zur sukzessiven Berechnung der neuen Ausmaße der Bounding-Box sind im Folgenden aufgelistet. Die Bounding-Box wird dabei durch zwei Vektoren  $u$  und  $l$  angegeben [Ogre3D 09a]. Der Vektor  $a$  ist der jeweils zuletzt zur Annotierung hinzugefügte Punkt.

$$\begin{pmatrix} l_x \\ l_y \\ l_z \end{pmatrix} = \begin{pmatrix} \min(l_x, a_x) \\ \min(l_y, a_y) \\ \min(l_z, a_z) \end{pmatrix}$$

$$\begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} = \begin{pmatrix} \max(u_x, a_x) \\ \max(u_y, a_y) \\ \max(u_z, a_z) \end{pmatrix}$$

## 4.5 Ontologien als Speicher für Annotierungen

Die Verwendung von Ontologien ist ein übliches Mittel zur Speicherung von Wissen in kognitiven Umgebungen. Auch in humanoiden Robotern verwendet man zur Repräsentation des Wissens über die Umgebung gerne Ontologien. Es stellt sich deshalb die Frage ob bereits die Annotierungen in eine Ontologie eingetragen werden sollen oder die Zusammenführung des durch die Annotierung gewonnen Wissens mit der Ontologie im Roboter erst nach der Evaluierung der Annotierungen vieler Räume erfolgen soll. Zu diesem Zweck soll im Folgenden ein Überblick über einige bekannte Ontologien gegeben werden.

**DOLCE** wird von [Hois 06] als Ontologie vorgeschlagen. Auf Grund des Schwerpunkts auf kognitive Aspekte und menschenähnliche linguistische Beschreibungen scheint DOLCE<sup>1</sup> tatsächlich eine ziemlich gute Wahl. Die kognitiven Aspekte sind im Hinblick auf die Weiterverwendung des gewonnenen Wissens durch einen Roboter und eventuell notwendiges logisches Ableiten von weiterem Wissen sicher von großem Interesse. Eine menschenähnliche linguistische Beschreibung ist wichtig, weil der Roboter später mit den Menschen in seiner Umgebung interagieren können muss. Die räumlichen Positionierungen der Objekte zueinander muss deshalb in der natürlichen Sprache beschreibbar sein und der menschlichen Beschreibung entsprechen. Eine einfache Erweiterbarkeit der Objektdatenbank um weitere Objekte ist gegeben [Hois 06]. Dazu wird ein Fallback-Objekt benötigt. Passt die Klassifizierung auf kein vorhandenes Objekt, wird es als Instanz des Fallback-Objektes angesehen. Im Nachhinein können Objekte dieser Klasse dann in neue Subklassen von „Perdurant“ eingeteilt werden [Masolo 03].

Einige Konzepte aus DOLCE würden allerdings überhaupt nicht benötigt werden. Ein Beispiel dafür ist die Klasse *Perdurant*. Diese beschreibt zeitliche Ereignisse. Da sich die Annotierung auf einen festen Zeitpunkt bezieht, findet sich für dieses Feature keine Anwendung. Deshalb würden alle Objekte, die annotiert werden können der Klasse *Endurants* angehören. Über die Relationen zwischen Objekten ist a-priori nichts bekannt. Jedes Objekt kann zu jedem anderen Objekt eine sinnvolle räumliche Relation haben. Deshalb müsste die Ontologie bei  $n$  Objekten  $\Theta(n^2)$  Relationen definieren. Bei einer relativen Positionierung wie in [Hois 06], bei der die Klassifizierung der räumlichen Relation außerdem von einem Bezugspunkt abhängt, kommt noch ein Faktor  $k$  dazu. Dieser gibt die Anzahl der verwendeten Kombination aus Bezugspunkt und Bezugssystem an. Wenn man berücksichtigt, dass ein Großteil dieser möglichen Relationen gar nicht erfüllt ist, ist das ein unnötig großer Overhead.

**SUMO - Ontologie** Auch SUMO<sup>2</sup> scheint für die gegebene Aufgabe überladen. Die Ontologie ist sehr umfangreich. Es würde ebenfalls nur ein Bruchteil der Ontologie wirklich

---

<sup>1</sup>Descriptive Ontology for Linguistic and Cognitive Engineering

<sup>2</sup>The Suggested Upper Merged Ontology



benutzt. Dafür geht die Ontologie in Bezug auf Haushalts-Umgebungen nicht hinreichend ins Detail. So werden z.B. Tiere relativ ausführlich klassifiziert, Möbel und Haushaltsobjekte aber gar nicht [ontologyportal.org 70] [Scott Farrar 70].

**DomoML-env** Die DomoML-env Ontologie wurde als Grundlage für Vernetzung von Haushaltsobjekten entworfen. In Folge dessen ist dort eine große Anzahl von Objekten vordefiniert. Außerdem sind die Objekte hier direkt mit den jeweiligen Funktionen abgespeichert. So verfügt ein Radio beispielsweise über einen Lautstärke-Regler. Diese Informationen sind für die spätere Verwendung als Wissensdatenbank für den Roboter zwar toll, bringen für die Annotierung aber kaum Vorteile. Lediglich Informationen darüber, ob ein Objekt weitere Objekte enthalten kann oder nicht, wäre daraus möglicherweise zu entnehmen. Allerdings lässt sich eine Annotierung von Objekten innerhalb anderer Objekte zur Zeit nur dann realisieren, wenn die 3D-Szene mit offenen Objekten modelliert wurde. Ein Objekt in einem anderen, völlig geschlossenen Objekt lässt sich also gar nicht annotieren. Die in dieser Ontologie vordefinierten Haushaltsobjekte scheinen aber hinreichend gut gewählt zu sein. Allerdings gibt es hier keine räumlichen Relationen, dadurch ist diese Ontologie auch nicht ohne Weiteres verwendbar, könnte aber als Grundlage für eine Objektdatenbank dienen. [Sommaruga 70]

**Sind Ontologien als Datenspeicher der Annotierung geeignet?** Ontologien an sich scheinen etwas über das Ziel hinaus zuschießen. Über die annotierten Objekte werden außer den Bounding-Boxen um diese Objekte keine weiteren Informationen benötigt. Für die Annotierung spielt es keine Rolle, ob zum Beispiel eine Tasse ein oben geöffnetes Gefäß ist oder nicht. Eine aufwändige Datenstruktur zur Spezifikation solcher Eigenschaften wird deshalb zum Zeitpunkt der Annotierung noch nicht benötigt. Es muss lediglich eine Zuordnung zu Objekten aus einer Objektdatenbank stattfinden. Diese Zuordnung ist nötig, weil später die relativen Positionen von Objekten in vielen verschiedenen Räumen gleicher Art verglichen werden müssen. Dafür ist ein eindeutiger Name für alle Objekte desselben Typs zwingend erforderlich.

Bei der späteren Verwendung der Daten sieht der Sachverhalt anders aus. Der Roboter braucht sehr wohl detailliertes Wissen über den Zweck und die Funktionalität von Objekten. Dazu scheint eine Ontologie eine gute Wahl.

Insgesamt könnte es sich also als sinnvoll erweisen die Integration der durch die Annotierung gewonnen Daten in die Ontologie erst nach deren Auswertung durchzuführen. Es wird dann zuerst ausgewertet, wie wahrscheinlich beispielsweise die Anordnung des Kochfelds über dem Herd ist. Die Auswertung findet bei der Daten-Aggregation aller Räumlichkeiten statt. Erst nach dieser Aggregation findet eine Integration in die Wissensdatenbank in Form der Ontologie statt.

## 4.6 Annotierungsdatenbank in SQLite

Eine schnelle, plattformunabhängige und unkompliziert zu handhabende Art der Datenspeicherung könnte die Verwendung einer einfachen SQL-Datenbank sein. Es würden dafür mindestens zwei SQL-Relationen benötigt:

**Objektdatenbank** Eine vollständige Liste der unterstützten Objekte. Diese Liste lässt sich, wenn in Form einer Datenbank realisiert, während der Annotierung unkompliziert erweitern.

**Räumliche Relationen** Hier werden die ermittelten Werte für die räumlichen Relationen zwischen je zwei Objekten aus der Objekt-Datenbank gespeichert.

In der SQL-Relation zur Speicherung der Werte der räumlichen Relationen werden auf jeden Fall zwei Referenzen auf Datensätze in der Objektdatenbank benötigt. Des Weiteren ist ein Feld für den Wert der räumlichen Relation unbedingt erforderlich. Es wäre natürlich möglich je ein Feld in der Datenbank für jede vorhandene räumlichen Relationen zu verwenden. Es würde dann für jedes paar aus Objekten *ein* Datensatz angelegt werden, der den Wert *aller* vorhandener Relationen enthält. Dieses Vorgehen hätte allerdings gewaltige Nachteile in Bezug auf die Erweiterbarkeit der Menge an untersuchten Relationen. Obwohl neue Relationen eher selten dazu kommen und im Zuge der Erweiterung des OViSE-Programms die Umgestaltung der Datenbank kein wesentlicher zusätzlicher Zeitaufwand darstellen würde, ist eine andere Lösung wünschenswert. Es könnten in diesem Zusammenhang beispielsweise Probleme in Bezug auf die Integrität vorhandener Datensätze beim Hinzufügen neuer Relationen geben. Mit welchen Werten sollten die neuen Felder in der Datenbank der räumlichen Relationen von älteren Annotierungen initialisiert werden, wenn keine echte Messung vorliegt? Deshalb wird eine zusätzliche SQL-Relation zum Speichern der angebotenen räumlichen Relationen verwendet.

Zur späteren Evaluierung der Relationen über mehrere Räume hinweg wird ein weiteres Programm entwickelt, das die aggregierten Daten dann in die vom Roboter verwendete Wissensdatenbank einträgt. Es bietet sich an die Evaluierung vom OViSE-Programm zu lösen, da eine 3D-Ansicht für die Werte der Relation nach der Evaluierung keine Vorteile bringt. Diese Werte beziehen sich schließlich nicht auf eine einzige Szene, eine dreidimensionale Darstellung gestaltet sich also schwierig. In dieses zusätzliche Programm lässt sich dann später auch gut ein "Ontologie-Exporter" einbauen, der das gewonnene Wissen so aufbereitet, dass es in die Wissensdatenbank des Roboters exportiert werden kann. In diesem Schritt muss auch die Zuordnung der Objekte aus der SQL-Objektdatenbank zu den Objekten der Wissensdatenbank im Roboter erfolgen.

Die folgende SQL-Relationen werden in der Implementierung verwendet:

```
CREATE TABLE rooms (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    name TEXT  
);
```

Diese Relation enthält alle verfügbaren Klassen von Räumen. Die Relation kann um weitere Informationen ergänzt werden. Denkbar wäre zum Beispiel, welches Objekt im jeweiligen Raum als Referenzobjekt dienen soll (Herd in der Küche, Fernseher im Wohnzimmer, ...)

```
CREATE TABLE objects (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    name TEXT UNIQUE  
);
```

Eine Liste verfügbarer Objekte. Diese Liste könnte zum Beispiel automatisch aus einer bestehenden Ontologie erstellt werden. Während der Annotierung muss eine Zuordnung zu den Objekten dieser Klasse erfolgen. Die Erweiterung der Liste von Objekten sollte zur Laufzeit der Annotierung möglich sein.

```
CREATE TABLE relation_type (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    name TEXT,  
    description TEXT  
);
```

Hier werden die verfügbaren Relationen spezifiziert. Die Auslagerung der Relations-Typen in eine extra Datenbank stellt die spätere einfache Erweiterbarkeit sicher.

```
CREATE TABLE relation (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    room INTEGER REFERENCES rooms (id),  
    object_po INTEGER REFERENCES objects (id),  
    object_so INTEGER REFERENCES objects (id),  
    relation_type INTEGER REFERENCES relation_type (id),  
    relation_value REAL  
);
```

Hier werden die eigentlichen relativen Positionierungen gespeichert. Das *object\_po* ist das Primary-Object, aus dessen Sicht die Relation berechnet wurde. *object\_so* ist das Secondary-Object.

Für jedes Tupel aus Objekten werden auch zwei Datensätze angelegt, da die Relationen nicht zwangsläufig symmetrisch sein müssen. Jedes dieser Objekte ist dann einmal Primary-Object und einmal Secondary-Object. Manche räumliche Relationen sind aber leicht umkehrbar. Wenn Objekt 1 rechts von Objekt 2 steht, steht Objekt 2 links von Objekt 1. Speichern dieser Daten führt zu unnötigen Redundanzen. Diese Eigenschaft wurde in den bisherigen Definitionen der SQL-Datenbanken noch nicht berücksichtigt.

In allen Definitionen ist zu beachten, dass SQLite *keine* Fremdschlüssel-Relationen unterstützt. Die obigen Anweisungen sind zwar syntaktisch korrekt, die darin enthaltenen Fremdschlüssel-Relationen werden von SQLite aber nicht sichergestellt.

Eine weitere Datenbank wird aus Komfort-Gründen angelegt. Dort werden die Annotierungen gespeichert und beim nächsten Laden des Raumes wieder eingelesen. Ohne dieses Feature wäre das Erweitern vorhandener Annotierungen schwierig. Würde ein Objekt dazu kommen müssten auch die bereits in früheren Annotierungen zugeordneten Objekte neu annotiert wer-

den. Auch, wenn neue räumliche Relationen hinzugefügt werden, ist es sinnvoll die früheren Annotierungen erneut laden zu können. Es muss dann lediglich die Berechnung der neuen räumlichen Relation, nicht aber die Annotierung erneut durchgeführt werden.

```
CREATE TABLE annotation (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    room INTEGER REFERENCES room (id) ,
    object INTEGER REFERENCES objects (id) ,
    min_x INTEGER,
    min_y INTEGER,
    min_z INTEGER,
    max_x INTEGER,
    max_y INTEGER,
    max_z INTEGER,
    is_reference_object INTEGER
);
```

## 4.7 Durchführung der Evaluierung

Um aus den Annotierungen allgemein gültiges und stabiles Wissen über alle Räume einer bestimmten Art ableiten zu können, muss eine Evaluierung der Einzelergebnisse stattfinden. Es wird immer die Evaluierung zweier Objekte `object_po` und `object_so` in Bezug auf einen Raumtyp durchgeführt. Alle in der Annotierung für die jeweiligen Objekte in den Räumen des Typs gewonnen Daten, werden aus der Datenbank ausgelesen und im Folgenden als *Stichprobe* einer statistischen Analyse behandelt. An dieser Stelle sei angemerkt, dass durchaus mehrere gleiche Objekte in derselben Instanz eines Raumes erlaubt sind. Außerdem muss auch nicht jedes Paar von Objekten in jedem Raum vorkommen. Das bedeutet bei der Auswertung der annotierten Daten, dass die Größe der Stichprobe aller Annotierungen eines Paares von Objekten in einem Raumtyp sowohl kleiner, gleich oder größer der Menge annotierter Räume dieses Typs sein kann.

Es ist zu beachten, dass zur Bestimmung einer räumlichen Relation zwischen zwei Objekten zwei Datensätze gespeichert werden, so dass jedes Objekt einmal Secondary- und einmal Primary-Objekt ist.

Zur Analyse der Stichprobe werden *Varianz* und *Erwartungswert* berechnet. Ist die Varianz klein genug, findet eine Klassifizierung anhand des Erwartungswertes statt. Bei großen Varianzen ist davon auszugehen, dass eine Klassifizierung nicht möglich ist, die Objekte also gar nicht der zu untersuchenden Relation unterliegen.

Untersucht man zum Beispiel die Relation `horizontal_angle` für die beiden Objekte Tisch und Stuhl ist die Varianz sehr groß. Es können sich Stühle auf quasi jeder Seite des Tisches befinden. Eine Klassifizierung in *links von*, *rechts von* oder *hinter* ist hier nicht möglich. Anders sieht das bei der `reference_plane_angle` Stichprobe für Backofen und Kochfeld aus. Die meisten `reference_plane_angle` "zeigen" nach oben. Die Varianz wäre klein und eine Klassifizierung anhand des Erwartungswertes deshalb möglich. Der dem Erwartungswert entsprechenden Winkel wäre nahe Null. Das Kochfeld befindet sich also über dem Backofen!

Alle Werte in der Stichprobe sind nach Definition des Wertebereichs der räumlichen Relationen aus  $\mathbb{R}$ . Wie in [Henze 00] beschrieben wird daraus nun die Varianz dieser Stichprobe wie folgt berechnet:

$$s_x^2 := \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2 \text{ mit } (n \geq 2)$$

Dabei ist das Stichproben-Mittel  $\bar{x}$  gegeben als:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

**Verbesserung: getrimmte Stichprobe** Die einfache Art der Auswertung ist allerdings sehr empfindlich gegenüber Ausreißern [Henze 00]. Um dieses Problem zu beheben wird zusätzlich das  $\alpha$ -getrimmte Stichproben-Mittel berechnet. Für  $\alpha = 0,05$  bedeutet das, dass die 5% größten *und* kleinsten Werte nicht berücksichtigt werden. Insgesamt werden also die 10% am weitesten abweichenden Messungen verworfen.

**Verbesserungen: ungleichmäßiges Trimmen** Wenn insgesamt  $n$  Werte entfernt werden sollen, könnte man, anstatt  $n/2$  Werte vom Anfang und  $n/2$  Werte vom Ende, auch die  $n$  am weitesten vom Erwartungswert entfernt liegenden Werte verwerfen. Dazu müsste man immer den ersten und den letzten Wert berücksichtigen und mit dem aktuellen Mittelwert vergleichen. Den weiter entfernt liegenden Wert würde man dann aus der Stichprobe entfernen und anschließend den neuen Erwartungswert berechnen, solange bis  $n$  Werte aus der Stichprobe entfernt wurden.

## 4.8 Erweiterbarkeit der Relationen

Wie in Abschnitt 3.7 bereits angedeutet wird die Erweiterung der vorhandenen Basis-Relationen direkt in Form weiterer C++ Klassen vorgenommen. Folgende C++ Klassen sind im AnnotationEvaluation Teil der Implementierung bereits vorhanden:

**Relation** Die Klasse Relation definiert die abstrakte Schnittstelle zur Abfrage der Werte einer Relation. Die Idee ist für jeweils zwei Objekte eine Instanz zu erzeugen, die dann eine Ergebnis-Menge liefert. Außerdem werden hier die mathematischen Hilfsmittel zur statistischen Analyse des Ergebnisses implementiert. Diese werden in Kapitel 4.7 ausführlich erklärt.

**GaussRelation** Hier findet die Abbildung der Ergebnis-Werte auf einen einzigen double-Wert statt. Es wird ein Soll-Wert festgelegt. Je weiter der Erwartungswert vom Soll-Wert entfernt liegt, desto weniger unterliegt die Stichprobe der gewünschten Relation. Um ein weiches Abfallen der Zugehörigkeit zu erreichen, wird die Gauß-Funktion zum zuweisen der endgültigen Wahrscheinlichkeit der Klassenzugehörigkeit verwendet. Die Klasse GaussRelation erweitert die Klasse „Relation“.

**DBRelation** DBRelation erbt von der Klasse „GaussRelation“. Instanzen dieser Klasse können nur für die Basis-Relationen erstellt werden. Für jede Instanz wird eine Datenbank-Anfrage ausgeführt und alle dort gespeicherten Relationen für jeweils zwei Objekte abgefragt und als Ergebnis-Menge zurückgegeben.

**TupleRelation** Diese abstrakte Klasse kombiniert zwei DBRelation Objekte. Es werden alle Mechanismen zur Abfrage der Werte der beiden Relationen zur Verfügung gestellt. Die eigentliche Verknüpfung der Relationen findet hier allerdings nicht statt. Dazu muss von TupleRelation geerbt werden.

Wie oben beschrieben muss zum Erstellen einer zweistelligen Relation eine neue Klasse erzeugt werden, die von TupleRelation erbt. Es wird dann die Art und Weise in der die Werte aus beiden Ausgangsklassen kombiniert werden sollen definiert.

# Kapitel 5

## Schlussfolgerung

### 5.1 Bewertung der Arbeit

Im Folgenden soll nun die Qualität der in der Studienarbeit eingeführten Mechanismen analysiert werden. Dazu wird zunächst die Verwendung einer SQLite Datenbank und anschließend die beiden Basis-Relationen Referenz-Ebenen-Winkel und Horizontaler Ebenen-Winkel an Hand typischer Szenen bewertet.

#### 5.1.1 Verwendung von SQLite als Datenspeicher

Nach erfolgter Annotierung können mit einfachen, aus SQL bekannten Mitteln die Annotierungen aus der Datenbank gelesen werden. Die Annotierungsdatenbank gestaltet sich, wie in Kapitel 4.6 besprochen, als plattformunabhängig. Auch das OViSE-Programm muss zum Auslesen der Datenbank nicht auf dem jeweiligen System installiert sein. SQL ist eine gut bekannte und einfache Sprache zur Abfrage von Werten aus einer großen Datenbasis. Viele Entwickler kennen sich bereits gut damit aus. Deshalb entfällt eine Einarbeitung oft vollständig. SQL lässt in einem gewissen Rahmen sogar eine Vorverarbeitung der gelesenen Werte zu. In Listing 5.2 werden die als Radian gespeicherten Winkel zum Beispiel direkt in die dazu gehörigen Werte in Grad umgerechnet. Weitere interessante Mechanismen zur Vorverarbeitung könnten zum Beispiel die Bildung des Mittelwertes, Summen über Wertereihen oder die Abfrage des Maximums, sowie Minimums sein. Das sollte viele Programme, die die gewonnenen Daten weiter verwenden wollen, vereinfachen.

#### 5.1.2 Bewertung Referenz-Ebenen-Winkel

Die Bestimmung des Referenz-Ebenen-Winkels in der Referenz-Ebene wurde in Kapitel 3.4 beschrieben. Nun soll die Qualität dieser abstrakten Beschreibung in Bezug auf die relative Positionierung zweier Objekte bewertet werden. Dazu wird eine einfache Annotierung in einer Küchenszene durchgeführt und das unkumulierte Ergebnis mit der intuitiven Beschreibung dieser Szene verglichen. Da hier die Bewertung der Ergebnisse innerhalb der Referenz-Ebene, also einer Ebene senkrecht zum Fußboden, durchgeführt werden soll, eignet sich eine Küchenszene sehr gut zur Analyse. Dort befinden sich bei einer geschickten Wahl der Referenz-

Ebene alle Objekte nahe dieser Ebene. Die Abbildung der Verbindungsvektoren zwischen den jeweils untersuchten Objekten ist deshalb sehr stabil und wenig fehleranfällig. Die Analyse lässt deshalb realistische Aussagen über die Qualität der eigentlichen Relation zu, ohne von anderen Mechanismen gestört zu werden.



Abbildung 5.1: Screenshot der für die Evaluierung verwendeten Annotierung

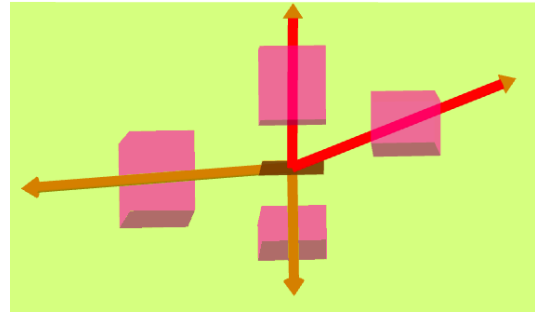


Abbildung 5.2: Darstellung der Annotierung nach Ausblendung der ursprünglichen Szene

Tabelle 5.1: Objekte in der untersuchten Szene

Raum	Objekt	Referenzobjekt
ArmarKitchen	fridge	0
ArmarKitchen	drawer	0
ArmarKitchen	microwave oven	0
ArmarKitchen	cupboard	0
ArmarKitchen	cooker	1

In Abbildung 5.1 ist die zur Evaluierung verwendete Küchenszene zu sehen. In der in Tabelle 5.1 dargestellten Annotierungsdatenbank erkennt man, dass das Referenz-Objekt in diesem Fall der Herd ist. Darüber hinaus wurden vier weitere Objekte um das Referenz-Objekt herum annotiert. In der Abbildung wird jeweils ein Verbindungsvektor vom Mittelpunkt des Referenz-Objektes zum Mittelpunkt eines der anderen Objekte dargestellt. Abbildung 5.2 zeigt dieselbe Szene noch einmal. Allerdings wurde die eigentliche Küche hier ausgeblendet. Es werden nur noch Boxen um die annotierten Objekte und die jeweiligen Verbindungsvektoren dargestellt.

Nach der Annotierung der Szenen können die berechneten Werte für die einzelnen Relationen abgefragt werden. Hier ist die Relation „reference\_plane\_angle“ von Interesse. Die in Listing 5.1 aufgelistete SQLite-Anfrage liefert die gewünschten Werte. In diesem Fall werden die Werte sogar schon vorverarbeitet ausgelesen um den berechneten Winkel zusätzlich in Grad anzuzeigen.



Tabelle 5.2: Ergebnis der Annotierung

Primary Object	Secondary Object	Winkel (Rad)	Winkel (Grad)
cooker	fridge	1.6292	266.6
cooker	drawer	3.150	179.5
cooker	microwave oven	5.0793	69.0
cooker	cupboard	6.278	0.3

Listing 5.1: Gespeicherte Werte für eine bestimmte Relation auslesen

```

SELECT
    po.name AS "Primary Object",
    so.name AS "Secondary Object",
    relation.relation_value AS Wert,
    relation.relation_value / 3.14159 * 180 AS "Wert (Grad)"
FROM
    relation,
    relation_type,
    rooms,
    objects AS po,
    objects AS so
WHERE
    relation_type.id = relation.relation_type AND
    rooms.id = relation.room AND
    po.id = relation.object_po AND
    so.id = relation.object_so AND
    relation_type.name = "reference_plane_angle" AND
    rooms.name = "ArmarKitchen" AND
    po.name = "cooker";

```

Tabelle 5.2 zeigt das Ergebnis der Annotierung. Die Werte werden als Radian sowie in Grad angezeigt. Vergleicht man diese Werte nun mit den Abbildungen 5.1 und 5.2 sind diese Daten sofort nachvollziehbar. Direkt über der Kochplatte befindet sich ein Schrank. Bestimmt man also den Winkel zwischen dem dazu gehörigen Verbindungsvektor und dem Einheitsvektor in y-Richtung, ergibt sich ein Wert nahe  $0^\circ$ . Die anderen Werte sind ähnlich plausibel. So befindet sich die Mikrowelle zum Beispiel rechts oberhalb der Kochplatten, was durchaus einem Winkel von rund  $70^\circ$  entsprechen kann. Recht eindeutig ist die Beschreibung der relativen Positionierung der Schublade unterhalb der Kochplatte, was sich in einem Winkel nahe  $180^\circ$  äußert. Auch ein Ergebnis von  $266,6^\circ$  für den Kühlschrank ist nicht überraschend. Ein Mensch würde diesen als „links“ von den Kochplatten beschreiben. Die Qualität der Referenz-Ebenen-Winkel scheint also hinreichend gut gewählt um der menschlich intuitiven Beschreibung zu genügen.

### 5.1.3 Bewertung Horizontaler Ebenen-Winkel

Analog zur Bewertung des Referenz-Ebenen-Winkel in Kapitel 5.1.2 soll hier nun untersucht werden, ob der horizontale Ebenen-Winkel den gestellten Forderungen entspricht. Eine Szene, deren Objekte aus der Vogelperspektive betrachtet um ein anderen Objekt angeordnet sind, eignet sich hierfür hervorragend. Die Abbildungen 5.3 und 5.4 zeigen eine typische Sitzgruppe.

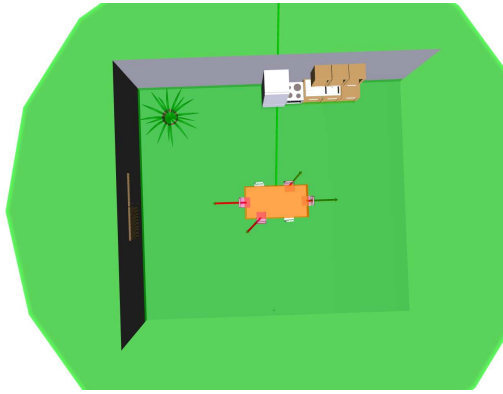


Abbildung 5.3: Screenshot der für die Evaluierung des horizontalen Ebenen-Winkels verwendeten Annotierung

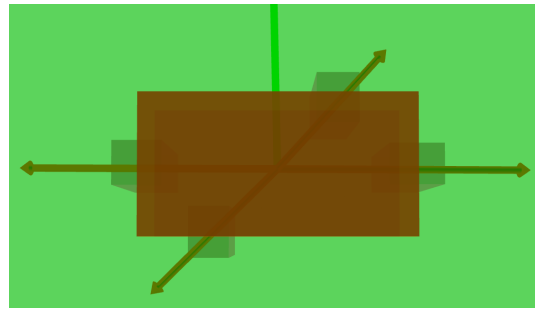


Abbildung 5.4: Darstellung derselben Annotierung nach Ausblendung der ursprünglichen Szene

Der in der Mitte befindliche Tisch wurde dabei als Referenz-Objekt annotiert. Des weiteren wurden einige der umliegenden Stühle markiert. Die grün dargestellte Ebene ist in diesem Fall die horizontale Ebene. Sie ist, wie in Kapitel 3.5 beschrieben, parallel zur x/z-Ebene und durchschneidet das Referenz-Objekt im Mittelpunkt. Des Weiteren ist die Orientierung der Szene als grüner Vektor eingezeichnet. Die Richtung dieses Vektors definiert sich aus VPP und Mittelpunkt des Referenz-Objekts. Um die Qualität der Relation bewerten zu können müssen nun die gespeicherten Werte aus der Datenbank abgefragt werden. Analog zu der SQL-Anfrage in Listing 5.2 werden die gewünschten Werte bereits vorverarbeitet als Grad und Radian ausgelesen. Das Ergebnis dieser Anfrage findet sich in Tabelle 5.3 wieder.

Tabelle 5.3: Ergebnis der Annotierung für den horizontalen Ebenen-Winkel

Primary Object	Secondary Object	Winkel (Rad)	Winkel (Grad)
table	chair	0.686	39.3
table	chair	1.521	87.1
table	chair	3.889	222.8
table	chair	4.671	267.7

Auch hier lässt sich die Plausibilität der gewonnen Winkel leicht überprüfen. Eine Zuordnung von gebräuchlichen Begriffen zu den jeweiligen Winkeln erscheint hier allerdings ungleich schwieriger. Ziel ist es eine Beschreibung der Situation aus Sicht des VPPs, also entlang des grün dargestellten Vektors, zu finden. Denkbar wären also zum Beispiel „links“ und „rechts“ für  $90^\circ$  beziehungsweise  $270^\circ$ . Für Winkel von ungefähr  $180^\circ$  wäre „dahinter“ eine mögliche Beschreibung. Die gefunden Werte scheinen also ganz brauchbar zu sein.

#### 5.1.4 Bewertung der Gültigkeitsanalyse anhand der Varianz

Nach der Annotierung einer Vielzahl von Räumen muss eine Bewertung der Stichprobe durchgeführt werden. Dabei muss für jeden Relationstyp entschieden werden, ob tatsächlich eine solche Relation zwischen jeweils zwei Objekten besteht. Im Falle der Existenz einer Relation muss die Stichprobe auf einen Durchschnittswert abgebildet werden. Das genaue Vorgehen wird in Kapitel 4.7 beschrieben. Die Qualität der Zuordnung des Durchschnittswertes soll nun mit Hilfe der Objekte Backofen und Kochfeld untersucht werden. Tabelle 5.4 listet die aus

der Annotierungsdatenbank gelesenen Ergebnisse auf. Alle verwendeten Räume wurden zwei Mal ausgewertet um eine größere Stichprobe zu erhalten. Dabei wurde jedes Mal ein leicht verschobener VPP verwendet. Das ändert das Bezugssystem geringfügig und führt deshalb zu leicht geänderten Werten.

Tabelle 5.4: Die horizontalen Ebenen-Winkel für die Objekte Kochfeld und Backofen

Raum	Winkel (Rad)	Winkel (Grad)
k_no_texture	3.143	180.1
ArmarKitchen	1.572	90.1
ownKitchen	3.134	179.6
k001	3.133	179.5
k002	3.174	181.8
k_no_texture	3.151	180.5
ArmarKitchen	1.572	90.1
ownKitchen	3.135	179.6
k001	3.132	179.5
k002	3.175	181.9

Beim Vergleich der gelesenen Werte mit der tatsächlichen Einrichtung der Räume ist die Plausibilität dieser Werte leicht nachvollziehbar. Bei näherer Betrachtung fällt auf, dass in der Test-Küche für den Armar-Roboter eine starke Abweichung zu den anderen Werten vorliegt. Ursache dafür ist, dass dort der Herd nicht, wie bei den meisten Küchen üblich, unter dem Kochfeld angebracht ist. Stattdessen ist dieser in einen der Schränke rechts eingearbeitet. Trotzdem gilt die Aussage „der Herd befindet sich unter dem Kochfeld“ für einen Großteil der untersuchten Szenen. Das Programm zur Auswertung der Stichprobe sollte diesen Zusammenhang also trotzdem feststellen und ausgeben.

Listing 5.2: Ausgabe des AnnotationEvaluation Programms zur Evaluierung der Stichprobe für den reference\_plane\_anlge zwischen Kochfeld und Backofen

```

1  Untrimmed sample mean is: 2.832
2  Untrimmed sample variance is: 0.441474
3
4  Trimming by 1
5  =====
6  1.57168
7  3.13194
8  3.13275
9  3.13404
10 3.13534
11 3.14335
12 3.15056
13 3.17355
14 =====
15 Trimmed sample mean is: 2.94665
16 Trimmed sample variance is: 0.308856
17
18 Trim(2) 2 at front and 0 at the end
19 =====
20 3.13194
21 3.13275
22 3.13404
23 3.13534
24 3.14335
25 3.15056
26 3.17355
27 3.17507
28 =====
29 Trim(2) sample mean is: 3.14708
30 Trim(2) sample variance is: 0.000321743

```

Listing 5.2 stellt die Ausgabe des AnnotationEvaluation Programms zur Auswertung der Stichprobe dar. Die beiden ersten Zeilen zeigen Stichproben-Mittel und Stichproben-Varianz der ursprünglichen Stichprobe. Es fällt auf, dass der Mittelwert auf Grund der beiden Aus-

reißer in ArmarKitchen fernab dem erwarteten Ergebnis ohne die Ausreißer ist. Die Varianz ist sehr hoch.

Berechnet man nun die getrimmte Stichprobe wird am Anfang und Ende der sortierten Probe jeweils ein Wert entfernt. Die Zeilen 6 bis 13 listen diese neue, um einen Ausreißer gekürzte, Stichprobe auf. Es wurde dabei allerdings auch ein Wert nahe dem Stichproben-Mittel verworfen, weil immer der jeweils größte und kleinste Wert entfernt wird. Der Mittelwert und die Varianz der verbesserten Stichprobe liegen etwas näher an den erwarteten Werten.

Anstatt jeweils gleich viele der größten und kleinsten Werte zu verwerfen, werden jetzt die  $n$  am weitesten entfernt liegenden Werte aussortiert. Dabei werden beide Ausreißer aus den ArmarKitchen Instanzen als solche erkannt und verworfen. In den Zeilen 20 bis 27 ist das sehr schön zu erkennen. Das neue Stichproben-Mittel von 3,147 liegt jetzt sehr nahe bei  $\Pi$ , also  $180^\circ$  und entspricht jetzt sehr genau den Erwartungen. Der Backofen wird also als *unter* dem Kochfeld erkannt. Die Varianz dieser letzten Stichprobe ist sehr klein, die Objekte unterliegen also offensichtlich der untersuchten Relation.

Listing 5.3: Ausgabe des AnnotationEvaluation Programms zur Evaluierung der Stichprobe für den horizontal\_angle zwischen Tisch und Stuhl

```

1  Untrimmed sampe is :
2  =====
3  1.06493
4  1.65422
5  1.75378
6  1.92734
7  2.75844
8  4.23228
9  4.74807
10 4.86586
11 5.02732
12 5.81154
13 =====
14 Untrimmed sample mean is: 3.38438
15 Untrimmed sample variance is: 2.99007
16
17 Trimming by 1
18 =====
19 1.65422
20 1.75378
21 1.92734
22 2.75844
23 4.23228
24 4.74807
25 4.86586
26 5.02732
27 =====
28 Trimmed sample mean is: 3.37091
29 Trimmed sample variance is: 2.23403
30
31 Trim(2) 1 at front and 1 at the end
32 =====
33 1.65422
34 1.75378
35 1.92734
36 2.75844
37 4.23228
38 4.74807
39 4.86586
40 5.02732
41 =====
42 Trim(2) sample mean is: 3.37091
43 Trim(2) sample variance is: 2.23403

```

Im Gegensatz dazu zeigt Listing 5.3 eine Stichprobe, die der untersuchten Relation nicht unterliegt. Es wird ein Zusammenhang zwischen Stühlen und Tischen in Bezug auf die horizontal\_angle Relation gesucht. Anschaulich bedeutet das, dass für die aus der Vogelperspektive betrachtete Sitzgruppe eine Regelmäßigkeit der Anordnung der Stühle um den Tisch gesucht wird. Diese Regelmäßigkeit existiert in der Realität aber nicht, da Stühle auf jeder Seite eines Tisches stehen können. Deshalb bleibt die Varianz in der Ausgabe des Programms selbst nach

dem Trimmen noch groß, da die Werte eine zu große Streuung aufweisen. Dieses Ergebnis deckt sich mit der intuitiven Beobachtung. Das AnnotationEvaluation Programm würde also erkennen, dass die gegebenen Objekte nicht der untersuchten Relation unterliegen.

Die Varianz erscheint also ein brauchbares Mittel um zu bestimmen, ob eine Stichprobe einer Relation unterliegt.

## 5.2 Ausblick

Um die entwickelte Software tatsächlich produktiv einsetzen zu können sind noch einige weitere Schritte erforderlich. Die annotierten Werte müssen schließlich abhängig vom letztendlichen Einsatzgebiet ausgewertet werden. Dazu ist es erforderlich genau zu definieren, welche Relationen für die jeweilige Anwendung wünschenswert sind. Danach müssen diese Relationen aus den Basis-Relationen abgeleitet werden. Zuletzt müssen die gefundenen Relationen zwischen den Objekten der Objektdatenbank dann in die gewünschte Anwendung integriert werden. Alle genannten Schritte finden in der Applikation zur Evaluierung der Stichproben statt. Das Programm zur Annotierung muss dafür nicht modifiziert werden, da die dort bereitgestellten Hilfsmittel zur Bestimmung der Basis-Relationen völlig ausreichend sind.

Während der Test-Phase haben sich einige mögliche Verbesserungen heraus kristallisiert. Das zur Zeit wohl größte Problem im Programm zur Evaluierung der Stichproben ist der nicht zusammenhängende Wertebereich für die beiden Winkel-Relationen. Ein Wert von 0 entspricht  $2\Pi$ . Dies wird bei der Evaluierung zur Zeit nicht berücksichtigt. Untersucht man Relationen, deren Stichproben-Mittel nahe bei 0 oder  $2\Pi$  liegt, ergeben sich große Varianzen, weil manche Messungen in Werten nahe 0 und andere nahe  $2\Pi$  resultieren. Diese Problem muss während der Evaluierung gelöst werden, da aus Sicht des Programms zur Annotierung alles korrekt abgelaufen ist.

Ein weiteres Problem ergibt sich bei der Definition bestimmter abgeleiteter Relationen. Als Beispiel eignet sich die Relation „daneben“, die als „rechts“ oder „links“ definiert werden könnte. Eine Relation soll erkannt werden, wenn jede Instanz eines Objekt mit hoher Wahrscheinlichkeit entweder rechts oder links vom jeweiligen Primary-Object steht. Wenn die Secondary-Objects nun tatsächlich gleichmäßig „rechts“ und „links“ verteilt sind, ist die Varianz aber sehr hoch. Es bilden sich sowohl viele Treffer für Werte in der Nähe von  $\frac{\Pi}{2}$  als auch  $\frac{3}{2}\Pi$ . Auf Grund der hohen Varianz unterliegt die Stichprobe keiner der beiden einzelnen Relationen. Da die Objekte weder der Relation „links“ noch der Relation „rechts“ unterliegen, kann auch keine Zuordnung zu „daneben“, also links oder rechts, erfolgen.

Die beiden genannten Probleme lassen sich in sofern beheben, als dass bei der Auswertung der annotierten Daten Gauß-Funktionen verwendet werden. Alle gewonnenen Winkel werden dann anhand einer Funktion auf das Intervall  $[0..1]$  abgebildet. Je Näher der Mittelwert dieser abgebildeten Werte an 1 liegt, desto eher unterliegt die Stichprobe der gewünschten Relation. Um nun mehrere Relationen zu kombinieren verwendet man einfach zwei Gauß-Funktion als Abbildungsfunktion. Im obigen Beispiel würden die Funktionen so gewählt werden, dass die eine Funktion ihr Maximum für  $\frac{\Pi}{2}$  und die andere bei  $\frac{3}{2}\Pi$  hat. Es wird dann immer der größte Funktionswert aller verwendeten Gauß-Funktionen als Wert für die Stichprobe gesetzt. Um den harten Übergang von 0 auf  $2\Pi$  zu beheben können auch mehrere Gauß-Funktionen verwendet werden. Dazu wird für jede Gauß-Funktion, die ihr Maximum im Intervall  $[0..\Pi]$  hat eine um  $2\Pi$  verschobenen weitere Gauß-Funktion hinzugefügt. Falls die gewünschte Funktion

ihr Maximum im Intervall  $(\Pi..2\Pi]$  hat ist die Verschiebung  $-2\Pi$ . Es wird dann erneut das Maximum dieser Funktionen als Stichproben-Wert interpretiert.

### 5.3 Fazit

Mit dieser Arbeit wurde ein Werkzeug geschaffen, das alle benötigten Tools für die komplexe Aufgabe der Szenen-Annotierung zur Verfügung stellt. Die Wahl der Relationen hat sich dabei bewährt. Durch den `reference_plane.angle`, den `horizontal_angle` und den Abstand zwischen zwei Objekten ist die relative Positionierung zwischen je zwei Objekten eindeutig und vollständig definiert. Zugleich ist diese Menge an Eigenschaften minimal. Es kann keiner der drei genannten Werte entfernt werden ohne die Vollständigkeit zu verlieren. Trotz der Vollständigkeit der Beschreibung sind die Werte unabhängig, intuitiv verständlich und eignen sich hervorragend zur Ableitung weiterer Relationen. Außerdem sind die ermittelten Werte unabhängig von der Orientierung und Lage der Szene. Das Konzept der einfachen Erweiterbarkeit, das von Anfang an in der Entwicklung berücksichtigt worden ist, garantiert die Zukunftsfähigkeit des Geschaffenen.

Die zur Navigation in der Szene und Annotierung verwendeten Eingabemethoden sind allgemein bekannt. Eine Einarbeitung in das Programm entfällt deshalb fast vollständig. Der Aufwand zur Annotierung einzelner Objekte ist denkbar gering. Für viele Objekte ist die Markierung mit zwei bis drei Eckpunkten möglich. Das Programm eignet sich deshalb auch zur Verarbeitung einer großen Anzahl an Szenen. Dies ist im Hinblick auf die Ableitung stabiler Relationen ein notwendiges Kriterium. Da die Annotierungen für die Objekte in der Datenbank sofort gespeichert werden, ist das Programm sehr resistent gegenüber Ausfällen im Annotierungsprozess.

Durch die Einarbeitung der neuen Funktionen in ein bereits bestehendes Programm konnte bei der Entwicklung mehr Zeit in Details gesteckt werden. Viele Funktionen waren bereits im OViSE-Programm eingebaut. Durch die gute Trennung zwischen dem ursprünglichen OViSE-Programm und den neuen Funktionen können zukünftige Verbesserungen im OViSE-Programm leicht in das Annotierungsprogramm zurückportiert werden. Die Verwendung von SQL als Datenspeicher und insbesondere von SQLite, das die vollständigen Daten in einer einzigen Datei speichert, macht den Datenspeicher plattformunabhängig. Das ist insbesondere im Hinblick auf die Trennung von Annotierung und Evaluierung in zwei verschiedene Programme nützlich. Die eigentliche Verarbeitung der Daten kann also auf einem anderen Rechner oder sogar von mehreren verschiedenen Arbeitsgruppen für unterschiedliche Projekte parallel verwendet werden.

Auch die Trennung von Annotierung und Evaluierung ist ein Vorteil für zukünftige Entwicklungen. Das Hinzufügen neuer Relationen gestaltet sich einfach und im Anschluss daran muss keine neue Annotierung erfolgen. Außerdem ist es möglich die selbe Annotierung zur Generierung von Wissensdatenbanken verschiedener Endanwendungen zu nutzen. So kann ein Roboter die Daten zum Beispiel zur Navigation in den Szenen und gleichzeitig zur menschlichen Interaktion nutzen. Die dabei verwendeten Relationen, die aus den Basis-Relationen abgeleitet wurden, können dabei völlig unterschiedlich sein.

# Abbildungsverzeichnis

1.1	Vollständig annotierte Küchen-Szene . . . . .	6
2.1	Axis-Aligned Bounding-Box eines Tisches bei optimal gedrehtem Raum . . .	10
2.2	Axis-Aligned Bounding-Box eines Tisches im um $45^\circ$ gedrehten Raum . . .	10
2.3	Überblick über eine vollständig annotierte Szene: Die grüne Ebene ist die reference-plane; bei dem rosa Punkt im Vordergrund handelt es sich um den VPP; das Referenzobjekt wird durch eine orange Bounding-Box markiert, die rosa Bounding-Boxen stellen die anderen annotierten Objekte dar. . . . .	11
3.1	Diese Abbildung stellt die Berechnung des Referenz-Ebenen-Winkels dar. Aus Gründen der besseren Darstellung wurde die Szene verschoben, so dass der Ursprung des Koordinatensystems der Mittelpunkt des orange dargestellten Objektes $a$ ist. . . . .	16
3.2	Darstellung der Berechnung des horizontalen Ebenen-Winkels. Die Szene wurde hier so verschoben, dass der Ursprung des Koordinatensystems im Mittelpunkt des Primary Objects liegt. . . . .	17
4.1	Screenshot des ursprünglichen OViSE Programms . . . . .	22
4.2	Diese Küche dient auch als Test-Umgebung für den Armar Roboter . . . . .	23
4.3	Eine einfache Küche mit Sitzgruppe . . . . .	23
4.4	Diese Küche enthält die meisten Objekte, die von Anfang an in der Objektdatenbank vorhanden sind . . . . .	23
4.5	Eine weitere Küche mit Sitzgruppe. . . . .	23
5.1	Screenshot der für die Evaluierung verwendeten Annotierung . . . . .	32
5.2	Darstellung der Annotierung nach Ausblendung der ursprünglichen Szene . .	32
5.3	Screenshot der für die Evaluierung des horizontalen Ebenen-Winkels verwendeten Annotierung . . . . .	34
5.4	Darstellung derselben Annotierung nach Ausblendung der ursprünglichen Szene	34





# Literaturverzeichnis

- [Clementini 97] Eliseo Clementini, Paolino Di Felice, Daniel Hernández. Qualitative representation of positional information. *Artif. Intell.*, 95(2):317–356, 1997.
- [Divvala 09] Santosh Kumar Divvala, Derek Hoiem, James H. Hays, Alexei A. Efros, Martial Hebert. An empirical study of context in object detection. Tagungsband: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), June 2009.
- [Henze 00] Prof. Dr. N. Henze, Priv.-Doz. Dr. D. Kadelka. Wahrscheinlichkeitstheorie und Statistik. Skriptenverkauf Universität Karlsruhe, 2000.
- [Hois 06] Joana Hois, Michael Wüstel, John A. Bateman, Thomas Röfer. Dialog-based 3d-image recognition using a domain ontology. Tagungsband: Int. Conf. Spatial Cognition 2006, Seiten 24–28, 2006.
- [Jung 02] Thomas Jung, Mark D. Gross, Ellen Yi luen Do. Sketching annotations in a 3d web environment. Tagungsband: CHI’02 Extended Abstracts, Seiten 618–619, 2002.
- [Jung 99] Thomas Jung, Mark D. Gross, Ellen Yi luen Do. Immersive redlining and annotation of 3d design models on the web. Tagungsband: 8 th International Conference on Computer Aided Architectural Design Futures, Seiten 81–98, 1999.
- [Kubat 07] Rony Kubat, Philip DeCamp, Brandon Roy. Totalrecall: visualization and semi-automatic annotation of very large audio-visual corpora. Tagungsband: ICMI ’07: 9th international conference on Multimodal interfaces, Seiten 208–215, New York, NY, USA, 2007. ACM.
- [Masolo 03] Claudio Masolo. Wonderweb deliverable d17. <http://www.loa-cnr.it/Papers/DOLCE2.1-FOL.pdf>.
- [Meyer 86] Walter Meyer. Distance between boxes: applications to collision detection and clipping. Tagungsband: IEEE International Conference on Robotics and Automation, vol. 3, Seiten 597–602, 1986.
- [Ogre3D 09a] Ogre3D. Axis-aligned-box. [http://www.ogre3d.org/docs/api/html/classOgre\\_1\\_1AxisAlignedBox.html](http://www.ogre3d.org/docs/api/html/classOgre_1_1AxisAlignedBox.html).
- [Ogre3D 09b] Ogre3D. Engine. <http://www.ogre3d.org/>.
- [ontologyportal.org 70] ontologyportal.org. Klassen in sumo. <http://www.ontologyportal.org/images/SUM0classes.gif>.
- [Scott Farrar 70] Universität Bremen Scott Farrar. Available ontologies for ai. <http://www.informatik.uni-bremen.de/~roefer/rgo/1.1.pdf>.

- [Sommaruga 70] Lorenzo Sommaruga, Antonio Perri, Francesco Furfari. Domoml-env: an ontology for human home interaction.
- [w3.org 09] w3.org. Semanticwebtools. <http://esw.w3.org/topic/SemanticWebTools>.
- [wikipedia.de 09] wikipedia.de. Fuzzy-logik. <http://de.wikipedia.org/wiki/Fuzzy-Logik>.

