UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

AGAS
Arbeitsgruppe Aktives Sehen

Entwicklung eines Algorithmus zur Detektion von Räumen in Gebäudegrundrissen

Studienarbeit
im Studiengang Computervisualistik

vorgelegt von

Oliver Lammersdorf

Betreuer: Dipl.-Inf. Johannes Pellenz, Institut für Computervisualistik, Fachbereich Informatik

Koblenz, im März 2009

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Richtlinien der Arbeitsgruppe für Studien- und Diplomarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden.	ja ☐	nein ☐
Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.	ja ☐	nein ☐

Koblenz, den Unterschrift

Inhaltsverzeichnis

Verzeichnis der Bilder

1 Einleitung

In dieser Arbeit wird ein neuer Algorithmus zur Detektion von Räumen in Gebäudegrundrissen beschrieben. Die Einleitung liefert dazu Informationen hinsichtlich der Motivation (Kapitel 1.1), gibt einen Überblick über den entwickelten Algorithmus (Kapitel 1.2) sowie die Struktur der vorliegenden Arbeit (Kapitel 1.3).

1.1 Motivation

Die zunehmende Integration von autonomen Heimrobotersystemen in das Leben des Menschen bedingt die Notwendigkeit der Interaktion mit ihnen.

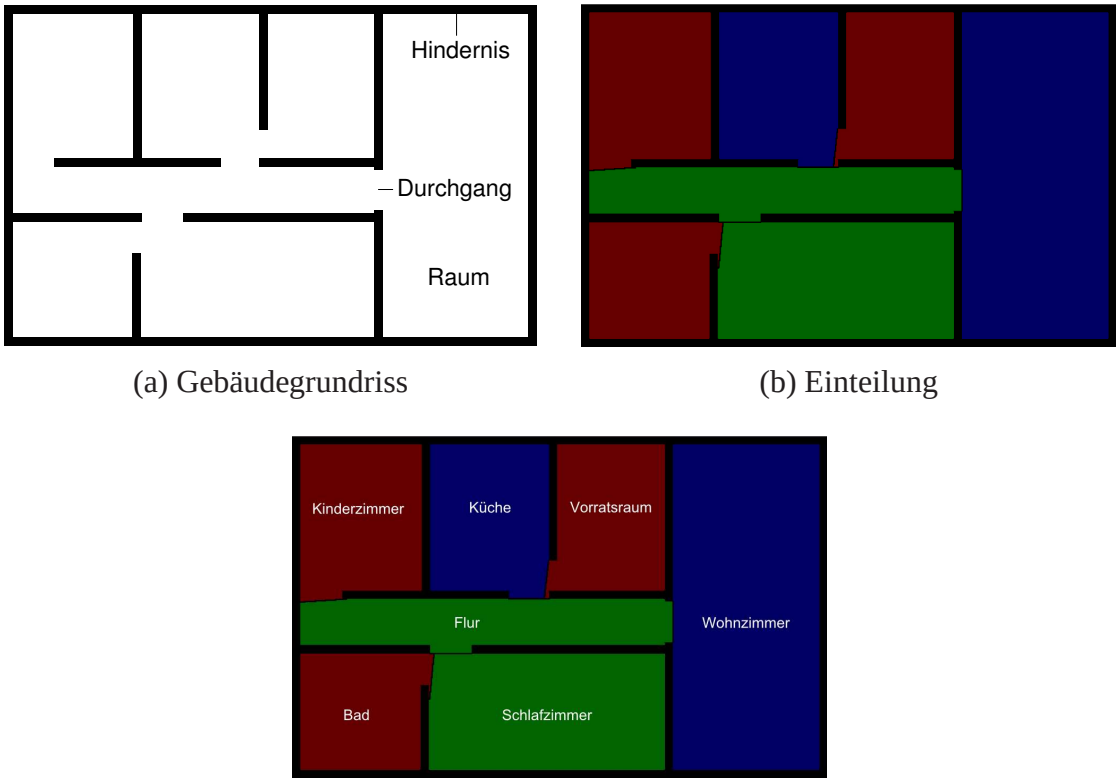
Möchte ein Mensch mit einem autonomen Robotersystem in Bezug auf die Umgebung kommunizieren, ist zur Orientierung bei beiden Wissen über die Umgebung und deren Identifikation notwendig.

Der Mensch erfasst die Umgebung visuell und speichert sein Wissen in einer virtuellen Karte. Die Einteilung einer Karte eines Gebäudegrundrisses in Räume ist für den Menschen intuitiv, da er aus Erfahrung gelernt hat, welche Gestalt Räume meist haben - Hindernisse wie Wände begrenzen Räume, Durchgänge wie Türen verbinden sie (Bild 1.1 a)).

Ebenso ist die Identifikation eines Raums, das Gleichsetzen eines Gebäudeteils mit einem Zweck oder einer Bedeutung wie Küche oder Wohnzimmer, für den Menschen anhand der Zuordnung von Einrichtungsgegenständen zu einem Zweck intuitiv.

Auch ein Robotersystem muss sich ähnlich wie der Mensch orientieren, um sich zurechtzufinden und benötigt daher ähnliches Wissen über die Umgebung und die räumliche Einteilung wie der Mensch.

In der autonomen Robotik stellt die Kartografie bereits einen wichtigen Aspekt in Hin-



(a) Gebäudegrundriss

(b) Einteilung

(c) Identifikation

Bild 1.1: Motivation

sicht auf Orientierung und Navigation dar. Ein autonomes Robotersystem erstellt eine Karte der visuellen Umgebung auf Basis der Daten von Sensoren wie Laser, Ultraschall oder Kamerasystem [vgl [PWP07]].

Mithilfe dieser Daten ist es nun für ein Robotersystem notwendig eine semantische Auswertung der Karte in Form einer umgebungsbezogenen Einteilung in Räume, die durch Wände und Türen voneinander abgegrenzt werden, vorzunehmen (Bild 1.1 b)), um später auf Grundlage dieses Wissens eine Identifikation räumlich zuordnen zu können (Bild 1.1 c)).

1.2 Überblick über den entwickelten Algorithmus

In dieser Studienarbeit wird ein neues Verfahren präsentiert, das den Schritt der räumlichen umgebungsbezogenen Einteilung einer Grundrisskarte realisiert.

Zunächst wird das Kartenmaterial in einem Vorverarbeitungsschritt (Kapitel 3.2.1) in Hindernis und Freiraum binarisiert. Je nach Beschaffenheit der Karte ist ein weiterer Vorverarbeitungsschritt (Kapitel 3.2.2) in Form von Dilatation bzw. Erosion hilfreich, um unerwünschte Störungen weniger zu gewichten.

Basierend auf der intuitiven Erfassung eines Raums durch den Menschen durch Begrenzung eines Raums auf Basis der ihn eingrenzenden Hindernisse werden im nächsten Schritt hindernisrepräsentierende Hinderniskontrollpunkte ermittelt (Kapitel 3.2.3).

Für jeden Hinderniskontrollpunkt wird jeweils ein Sichtbarkeitstest mit einem Sehstrahl zu allen anderen Hinderniskontrollpunkten vorgenommen (Kapitel 3.2.4).

Die Ergebnisse der Sichtbarkeitstests werden in einer Adjazenzmatrix eingetragen (Kapitel 3.2.5). Darin wird jeder Hinderniskontrollpunkt von jeweils allen Hinderniskontrollpunkten, einer Sichtbarkeitsmenge, repräsentiert und für jedes Element der Sichtbarkeitsmenge eingetragen, ob es den Hinderniskontrollpunkt sehen kann oder nicht.

Diese Sichtbarkeitsmengen werden paarweise auf ihre Ähnlichkeit hin untersucht (Kapitel 3.2.6). Der Grad der Ähnlichkeit wird in eine Ähnlichkeitsmatrix eingetragen.

Abhängig von ihrer Ähnlichkeit werden Elemente (Hinderniskontrollpunkte) ähnlicher Sichtbarkeitsmengen in der Raumrekonstruktion zu einem Raum zusammengefasst (Kapitel 3.2.7).

Für jeden Raum wird auf Basis seiner Raumhinderniskontrollpunktmenge ein Raumnetz aufgespannt (Kapitel 3.2.7), wobei jedem Raumnetz eine eindeutige Farbe zugeordnet wird. Anschließend werden die Raumnetze mit ihrer Farbe geflutet (Kapitel 3.2.8).

An den so entstandenen Übergangsgrenzen zwischen zwei benachbarten Räumen werden nun Türen erzeugt (Kapitel 3.2.9).

Ist die gewünschte Genauigkeit der Raumunterteilung noch nicht erreicht, kann auf Grundlage der erzeugten Türen durch einen erneuten Durchlauf eine Verfeinerung (Kapitel 3.2.11) erreicht werden. Die bereits ermittelten Türen werden dazu als Hindernis in die ursprüngliche Grundrisskarte eingetragen und das Ergebnis dieser Fusion als neue

(a) Robbie

(b) von Robbie erstellte Karte

Bild 1.2: Robbie und von Robbie erstellte Karte

Basisgrundrisskarte verwendet.

Die in die Grundrisskarte eingetragenen gefluteten Raumnetze und die Türen ergeben zusammen das Endergebnis (Kapitel 3.2.10).

Jeder Raum wird durch einen Raumdeskriptor näher beschrieben. Dieser enthält Raumname, Raumgröße, Raumschwerpunkt, Raumfarbe und eine den Raum repräsentierende Raumhinderniskontrollpunktmenge. Alle Raumdeskriptoren einer Karte werden als Element in einem Kartendeskriptor gespeichert (Kapitel 3.2.12).

Das Verfahren arbeitet sowohl auf künstlich erzeugten als auch auf von einem autonomen Robotersystem wie Robbie[1] (Bild 1.2 a)) anhand von Lasersensordaten erstellten zweidimensionalen Karten (Bild 1.2 b)).

[1] `http://robots.uni-koblenz.de`

1.3 Überblick über die Struktur der vorliegenden Arbeit

In Kapitel 2 wird der aktuelle Stand der Wissenschaft durch relevante bisherige Lösungsansätze und Resultate beschrieben bevor in Kapitel 3 die Schritte des neu entwickelte Algorithmus theorisch und visuell im Detail vorgestellt werden.

Dabei befasst sich Kapitel 3.1 mit grundlegenden Definitionen, Kapitel 3.2 mit der Beschreibung der einzelnen Schritte und Kapitel 3.3 mit der gewählten Implementationsform.

Eine Übersicht über erzielte Ergebnisse und deren Aufwände liefert Kapitel 4. Neben guten Resultaten werden an dieser Stelle auch Zwischenergebnisse, Besonderheiten und Seiteneffekte diskutiert.

Abschließend wird in Kapitel 5 eine Zusammenfassung der vorliegenden Arbeit (Kapitel 5.1) sowie ein Ausblick über mögliche Ansatzpunkte für Verbesserungen und Erweiterungen (Kapitel 5.2) präsentiert.

Der Quellcode des entwickelten Algorithmus ist auf der beiliegenden *CD* enthalten.

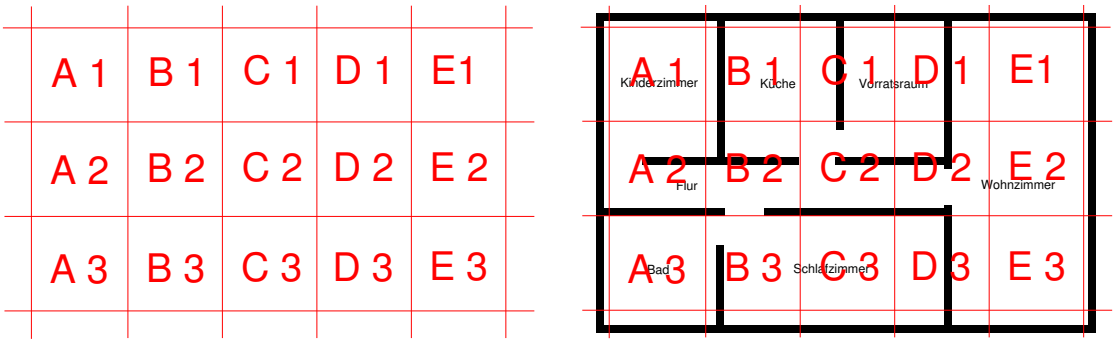
2 Stand der Wissenschaft

Bisherige Lösungsansätze zur Unterteilung von Karten in Räume sind die *Unterteilung durch ein regelmäßiges Grid* wie es bei der Unterteilung von Landkarten verwendet wird (Kapitel 2.1) und die Repräsentation von Räumen in Form von probalilistischen Quadtrees in *Finding Rooms in Probabilistic Quadtrees* [PGKKP05] wobei durch Rechtecktraversierung Räume voneinander getrennt werden (Kapitel 2.2).

Interessante andere Verfahren sind die bildverarbeitungsbasierte Unterteilung einer Karte in Räume, wie sie in *Multi-Hierarchical Semantic Maps for Mobile Robots* [GSC+05] verwendet wird (Kapitel 2.3), und durch Detektion von Räumen, Korridoren und Durchgängen während der Kartenerstellung in *Supervised Learning of Places from range Data using AdaBoost* [MMSB05] (Kapitel 2.4).

In *Bestimmung von Räumen in Gebäudegrundrissen* [Kow07] findet eine Unterteilung einer Karte in Räume auf Basis einer Türdetektion in Abhängigkeit des Verhältnisses von Umfang eines Raums zur Türbreite statt (Kapitel 2.5).

Weitere Ansätze, die an dieser Stelle nicht weiter beschrieben werden, sind in Kapitel 2.6 aufgelistet.



(a) Landkarte mit Grid (b) Raumunterteilung mit Grid

Bild 2.1: Regelmäßiges Grid

2.1 Regelmäßiges Grid

Durch die triviale Unterteilung eines Gebäudegrundrisses mithilfe eines regelmäßigen Grid kann eine Karte sehr schnell in Teilbereiche aufgeteilt werden, wobei jedes Feld des Grid einen Teilbereich repräsentiert (Bild 2.1 a)). Die Genauigkeit der Unterteilung ist dabei abhängig von der Feldgröße.

Da Räume unterschiedliche Größen und Formen aufweisen, ist die Repräsentation von Räumen durch solche Felder jedoch sehr ungenau. Das bedeutet, genau ein Feld repräsentiert nicht genau einen Raum. Ein Raum kann also unter mehreren Feldern liegen, in der Art wie der Raum **Kinderzimmer** in Bild 2.1 b) unter den Feldern **A1**, **A2**, **B1** und **B2** liegt.

Andererseits können sich unter einem Feld Teile aus mehreren Räumen befinden. So liegen die Räume **Kinderzimmer**, **Küche** und **Flur** aus Bild 2.1 b) unter dem Feld **B2**.

Eine auf diese Weise vorgenommene Einteilung kann zwar zur Orientierung genutzt werden, hat dabei jedoch den Nachteil, dass sie keinen Bezug zur abgebildeten Umgebung hat.

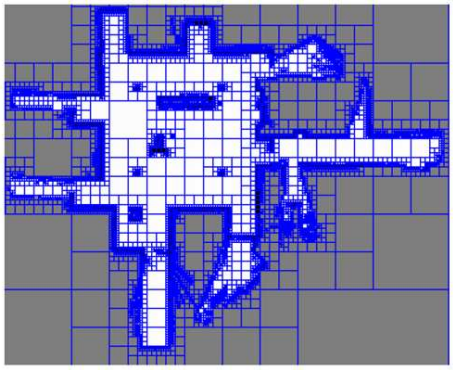


Bild 2.2: Probabilistische Quadtrees von Pages[PGKKP05]

2.2 Probabilistische Quadtrees

In *Finding Rooms on Probabilistic Quadtrees* [PGKKP05] von Gassull Pagès, Gerhard K. Kraetzschmar und Günther Palm aus dem Jahr 2005 werden auf Basis der Auswertung eines Occupancy Grid möglichst große, von Hindernissen begrenzte und sich nicht gegenseitig überdeckende Rechtecke in Freiflächen aufgespannt. Dabei repräsentiert jedes dieser Rechtecke einen Raum (Bild 2.2).

Zunächst wird für jedes Rasterfeld die Belegung (Occupancy) ermittelt. Im nächsten Schritt wird in der Karte das erste nicht belegte Rasterfeld gesucht und als Ausgangspunkt zum Aufspannen eines Rechtecks von dieser Position aus in den freien Raum gewählt. Das Rechteck wird also entlang der Koordinatenachsen möglichst weit aufgespannt. Die Ausdehnung des Rechtecks wird durch das naheste belegte Rasterfeld in der entsprechenden Richtung begrenzt.

Angewandt auf die Motivation der vorliegenden Arbeit ergeben sich dieselben Ungenauigkeiten hinsichtlich der Repräsentation von Räumen wie bei der Anwendung eines regelmäßigen Grid - ein Raum kann unter mehreren Rechtecken liegen und unter mehreren Rechtecken kann ein Raum liegen.

Ungenau ist dieses Verfahren bei nicht bildachsenparallelen rechtwinkligen Grundrissen und Räumen, die durch diese Eigenschaft nicht vollständig erfasst werden können. Dadurch entstehen auch viele kleine Rechtecke, die in der intuitiven Wahrnehmung eines Raums durch den Menschen zu einem Raum zusammengefasst würden.

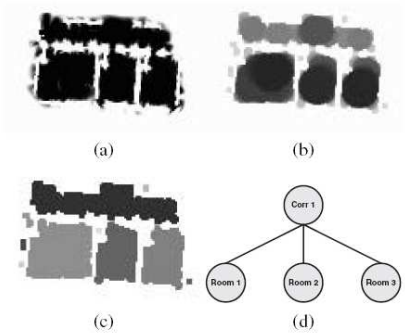


Bild 2.3: Multi-Hierarchical Semantic Maps von Galindo[GSC+05]

2.3 Multi-Hierarchical Semantic Maps

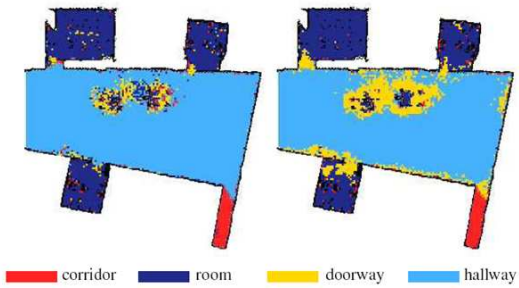
In *Multi-Hierarchical Semantic Maps for Mobile Robots* [GSC+05] von C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J.A. Fernandez-Madrigal und J. Gonzales aus dem Jahr 2005 werden räumliche Informationen mit semantischen Informationen mittels eines Ankers verknüpft, der zwischen zwei AH-Graphen (Multi-Hierarchie) erzeugt wird.

Die semantischen Informationen stammen aus der Klassifizierung von Objekten.

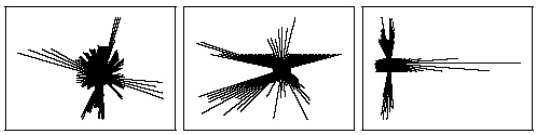
Die räumlichen Informationen stammen aus der Auswertung eines Occupancy Grid. Das dabei entstehende Grid wird mithilfe von Bildverarbeitungstechniken in möglichst große freie Bereich unterteilt. Jeder dieser Bereiche repräsentiert dabei einen Raum.

Diese Bereiche werden nach einer weiteren morphologischen Operation mithilfe von Watershedding farbig geflutet (Bild 2.3).

Nun können gefundene semantischen Informationen anhand ihrer geographischen Lage mit den gefunden Bereichen durch einen Anker verbunden werden.



(a) segmentierte Räume, Korridore und Türen



(b) Laserscans von Raum, Tür und Korridor

Bild 2.4: überwachte Durchgangdetektion von Martinez[MMSB05]

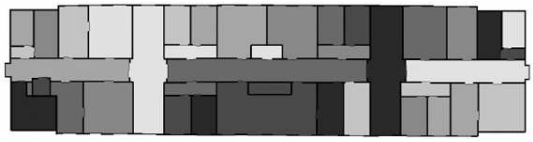
2.4 Überwachte Durchgangsdetektion

Im Verfahren in *Supervised Learning of Places from range Data using AdaBoost* [MMSB05] findet eine Abgrenzung der Räume voneinander durch Detektion von schmalen Durchgängen (Türen), großen Freiflächen (Räume) und lange Freiflächen (Korridore) auf Basis von Laserscans schon während der Erstellung der Karte statt (Bild 2.4 a)).

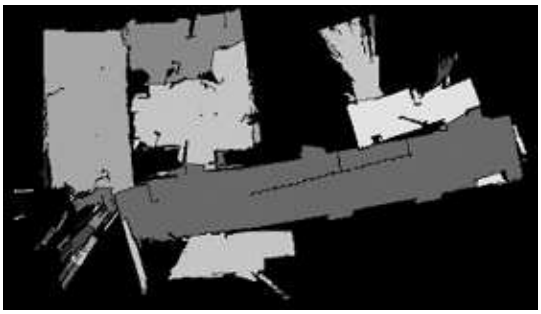
Ein Referenzscan wird sofort nach seiner Aufnahme klassifiziert in *Raum*, *Tür* oder *Korridor* (Bild 2.4 b)). Die Klassifizierung basiert auf einfachen geometrischen Charakteristika, die aus einem Referenzscan extrahiert und mithilfe des AdaBoost Algorithmus [FS95] verstärkt werden.

Zur selbstständigen Klassifizierung ist ein vorheriges Training notwendig.

Schwierigkeiten ergeben sich bei diesem Verfahren bei zu breiten Türen, die dadurch nicht als solche klassifiziert werden können und bei nicht trainierten Umgebungen, weshalb das Verfahren teilweise überwacht stattfindet.



(a) Ergebnis mit künstlicher Karte



(b) Ergebnis mit Laserscankarte

Bild 2.5: Durchgangsdetektion von Kowalski [Kow07]

2.5 Durchgangsdetektion

Auch in *Bestimmung von Räumen in Gebäudegrundrissen* [Kow07] von Kay Kowalski aus dem Jahr 2007 findet eine Detektion von Türen auf Basis eines Occupancy Grid statt.

Hier werden Freiräume von Kettencodes umschlossen. Auf diesen Ketten werden Türkandidatenpfosten anhand von Richtungswechseln des Kettencodes bestimmt und die Türkandidatenpfosten im folgenden paarweise untersucht, um zu bestimmen ob der Türkandidat, den beide bilden, als Tür detektiert wird oder nicht.

Der Kettencode wird dafür an diesen Stellen abhängig vom Verhältnis Kettencodelänge zu Türkandidatenbreite unterteilt und durch eine Tür geschlossen, wobei ein eigenständiger Kettencode für den von der Tür geschlossenen ermittelten Raum sowie für den noch zu untersuchenden Rest des Objektes entsteht.

Diese Trennung von Objekten basiert auf dem Teil *Cutting by boundary information* aus dem Verfahrens in *Real-Time Detection of Arbitrary Objects in Alternating Industrial Environments* [BEP+01] von Dirk Balthasar, Thomas Erdmann, Johannes Pellenz, Volker Rehrmann, Jörg Zeppen und Lutz Priese aus dem Jahr 2001.

Anschließend werden alle Freiflächen einzeln mithilfe von floodfill farbig geflutet (Bild 2.5). In einem Raumdeskriptor werden Informationen über die detektierten Räume abgespeichert.

Schwierigkeiten bereiten diesem Verfahren Hindernisfragmente, die jeweils einen eige-

nen Kettencode ihrer Umfanglinie haben und somit zueinandergehörende Türpfostenkandidaten nicht zugeordnet werden können, wenn sie in unterschiedlichen Kettencodes liegen.

2.6 Weitere Ansätze

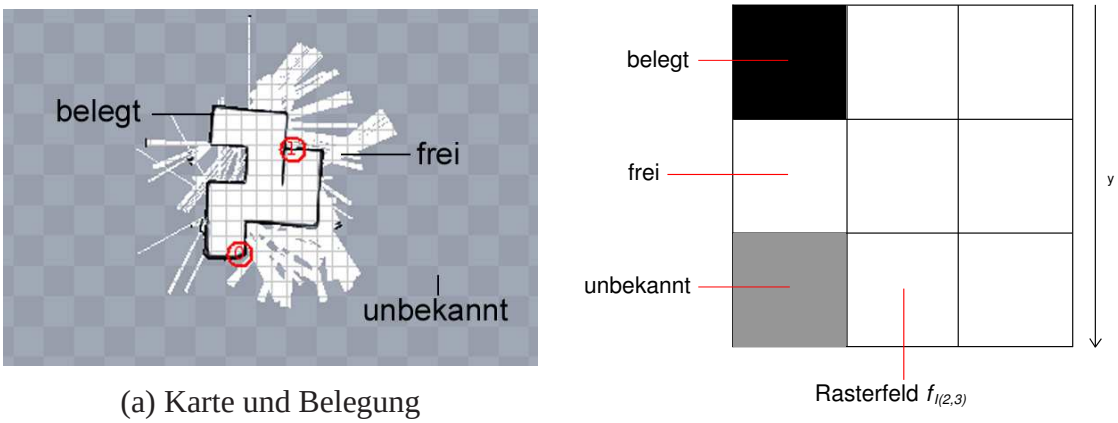
In *Supervised Learning of Places from range Data using AdaBoost* [MMSB05] werden weitere Ansätze beschrieben. Diese sind *Behaviour coordination in structured environments*[AC03] von Althaus und Christensen aus dem Jahr 2003, *A virtual sensor for room detection*[BS02] von Buschka und Saffiotti aus dem Jahr 2002, *A robot navigation architecture based on partially observable markov decision process models*[KS98] von Koenig und Simmons aus dem Jahr 1998 sowie *Context-based vision system for place and object recognition*[TMFR03] von Torralba, Murphy, Freeman und Rubin aus dem Jahr 2003 (vgl. [MMSB05]).

3 Detektion von Räumen in Gebäudegrundrissen

Um eine Basis zu schaffen, auf der der entwickelte Algorithmus aufgebaut werden kann, ist es notwendig die grundlegenden beteiligten Elemente (Kapitel 3.1.1-3.1.4) und das zu erreichende Ergebnis (Kapitel 3.1.5) zu definieren.

Im Anschluß werden die einzelnen Schritte des Algorithmus im Detail besprochen und visuell präsentiert (Kapitel 3.2.1-3.2.12) bevor die verwendete Implementationsform (Kapitel 3.3), die Klassen (Kapitel 3.3.1) und das Graphical User Interface (Kapitel 3.3.2) beschrieben werden.

Eine Übersicht über den entwickelten Algorithmus ist in Anhang A in Abbildung A.3 in Form eines grobgranularen Aktivitätsdiagramms abgebildet.



(a) Karte und Belegung

(b) Rasterbild und Belegung

Bild 3.1: Karte, Rasterbild und Belegung

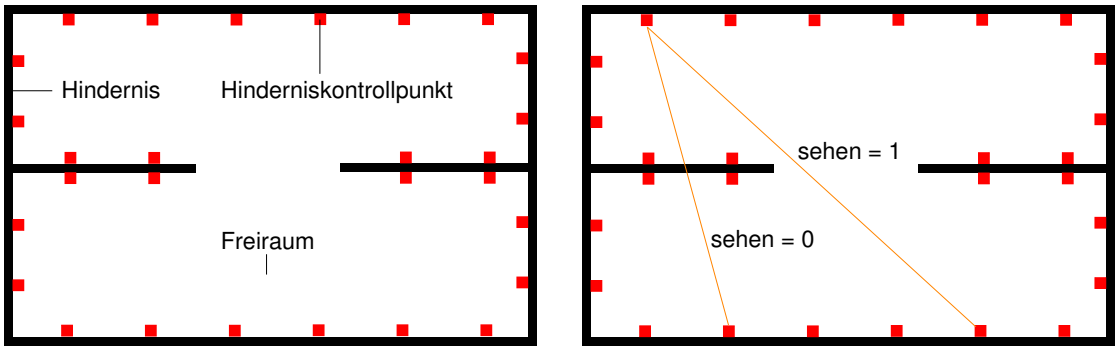
3.1 Definitionen

3.1.1 Karte, Hindernis und Freiraum

Eine die Umgebung visuell wiedergebende Karte K (Bild 3.1 a)) liegt als zweidimensionales Bild I in der Form Rasterbild vor, das ein erweitertes Occupancy Grid realisiert und einen Gebäudegrundriss anzeigt. Jedes Rasterfeld $f \in F$ aus der Menge aller Rasterfelder F des Bildes repräsentiert dabei ein Pixel im Bild mit den Bildkoordinaten $I(x, y)$ (Bild 3.1 b)).

Für jedes Rasterfeld ist eine Information bezüglich der Belegung (Occupancy) O eingetragen. Diese Information kann **belegt** für ein Hindernis, **frei** für Freiraum oder **unbekannt** für unbekanntes Gebiet sein (Bilder 3.1 b) und 3.2 a)).

Wände und Objekte im Freiraum, wie Raumeinrichtungsgegenstände, werden bei dem hier entwickelten Verfahren als Hindernis gewertet, da in den zugrundeliegenden Karten nicht zwischen Wand und Einrichtungsgegenstand unterschieden wird. Unbekanntes Gebiet wird ebenfalls als Hindernis gewertet, da seine Fläche zur Einteilung der Karte in Räume nicht verwendet werden kann. Außerdem ist es notwendig die Grenze von unbekanntem Gebiet zu Freiraum als Hindernis zu werten, um dort Hinderniskontrollpunkte erzeugen zu können, die den Freiraum beschreiben (siehe Beispiel in Kapitel



(a) Hinderniskontrollpunkte (b) Sehstrahl und Sichtbarkeit

Bild 3.2: Hinderniskontrollpunkte, Sehstrahl und Sichtbarkeit

3.2.3). Somit bleiben als mögliche Belegungen noch **belegt** für Hindernis und **frei** für Freiraum übrig.

3.1.2 Hinderniskontrollpunkt

Ein Hindernis wird durch die Menge von Rasterfeldern beschrieben, die im Freiraum liegen und an das Hindernis grenzen. Jedem alleinstehenden Hindernis in einer Karte ist eine solche Menge zugeordnet. Die Vereinigungsmenge dieser Mengen enthält alle Hinderniskontrollpunktkandidaten. Aus dieser Menge wird eine alle Hindernisse repräsentierende Hinderniskontrollpunktmenge H entnommen.

Die darin enthaltenen Hinderniskontrollpunkte $h \in H \subseteq F$ repräsentieren Rasterfelder, die äquidistant verteilt entlang der Hindernisse liegen (Bild 3.2 a)).

3.1.3 Sehstrahl und Sichtbarkeit

Ein Sehstrahl B ist ungerichtet zwischen zwei Hinderniskontrollpunkten $h_i, h_j \in H$ definiert und hat ein Attribut **sehen**. Der Wert dieses Attributs wird durch einen Test auf Sichtbarkeit $s(h_i, h_j)$ ermittelt und kann den Wert 1 oder 0 annehmen (Bild 3.2 b)). Es hat den Wert 1, wenn auf der Strecke zwischen den beiden Hinderniskontrollpunkten kein Hindernis liegt und sich die Hinderniskontrollpunkte somit gegenseitig sehen können. Ansonsten ist der Wert des Attributs 0.

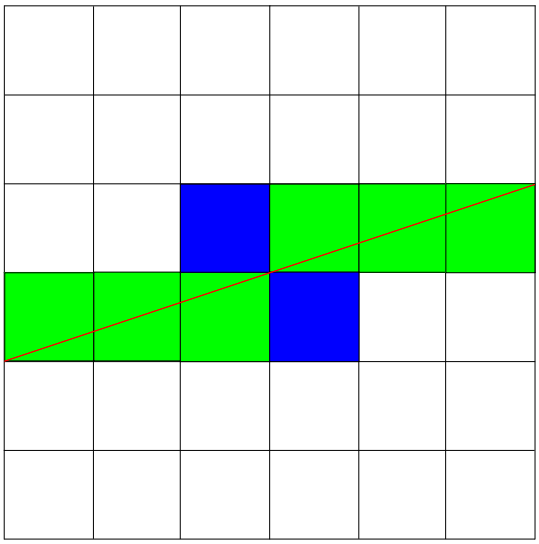


Bild 3.3: Lückenschluß einer Bresenhamlinie

Die Sehstrahlen sind ungerichtet, da der Wert des Attributs **sehen** in beide Richtungen des Strahls stets gleich ist. Das bedeutet, wenn der Ausgangspunkt h_i den Zielpunkt h_j sehen kann, so sieht der Ausgangspunkt h_j auch den Zielpunkt h_i.

Die Erzeugung eines Sehstrahls wird durch einen angepassten Bresenham-Algorithmus zur Linienerzeugung realisiert. Die Anpassung beschränkt sich auf den Bereich der Stufenstellen - also genau dort, wo sich der y-Wert der Kooridnate ändert.

An diesen Stufenstellen der Bresenhamlinie wird die Linie **verdickt** (in Bild 3.3 blaue Rasterfelder), um eine lückenlose Linie zu erhalten. Lückenlos bedeutet, dass es nicht möglich ist auf Pixelebene diese Linie zu kreuzen, ohne auf ein **belegtes** (in Bild 3.3 grüne und blaue Rasterfelder) Rasterfeld zu treffen.

Das ist wichtig, um beim Sichtbarkeitstest nicht durch ein Hindernis in Form einer dünnen Linie hindurchschlüpfen zu können. Ebenso wird durch den Lückenschluß verhindert in einem Verfeinerungsschritt eine mit dem Bresenham-Algorithmus erzeugte Tür mit einem Sehstrahl zu passieren, ohne auf ein belegtes Rasterfeld zu treffen. Auch für das farbige Füllen der Raumnetze, die mithilfe des Bresenham-Algorithmus erzeugt werden, ist dieser Lückenschluß notwendig, um nicht ein zweites Raumnetz fälschlicherweise mit der Farbe des ersten zu füllen.

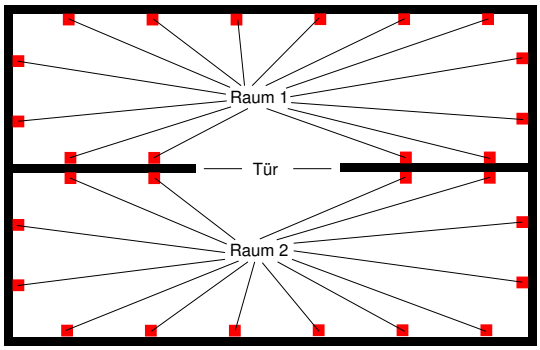


Bild 3.4: Raumzuordnung der Hinderniskontrollpunkte

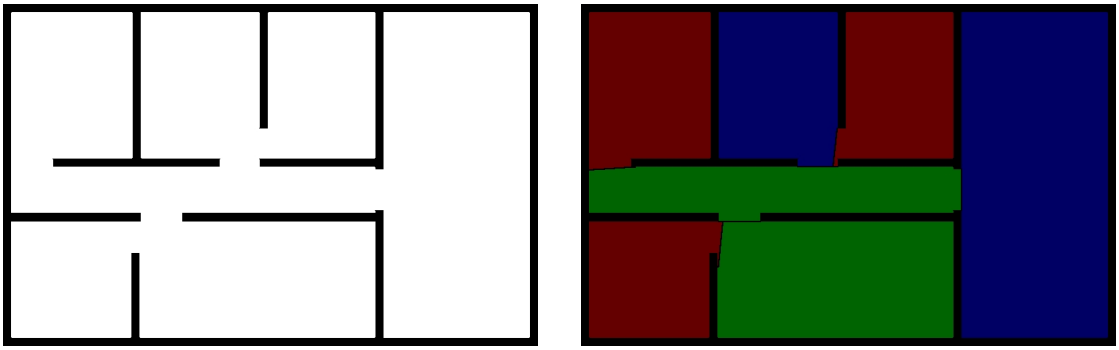
3.1.4 Raum und Tür

Die Räume $r \in R$ in einem Gebäudegrundriss sind in der Raummenge R zusammengefasst. Ein Raum wird von Freiraum repräsentiert, der durch Hindernisse begrenzt ist. Die Hinderniskontrollpunktmenge H beschreibt den gesamten Freiraum im Gebäudegrundriss.

Jede Raumhinderniskontrollpunktmenge $Q \subseteq H$ ist eine Teilmenge von H und repräsentiert somit einen Teil des gesamten Freiraums. Desweiteren ist jede Raumhinderniskontrollpunktmenge $Q_q \subseteq H$ zu jeder anderen Teilmenge $Q_r \subseteq H$ disjunkt.

Die Hinderniskontrollpunkte $h \in Q_r$ spannen den Raum $r \in R$ auf. Dabei ist jeder Hinderniskontrollpunkt $h \in H$ maximal genau einem Raum $r \in R$, also genau einer Raumhinderniskontrollpunktmenge Q, zugeordnet (Bild 3.4).

Türen teilen den gesamten Freiraum im Gebäudegrundriss in einzelne Segmente, indem sie benachbarte detektierte Räume voneinander trennen. Räume sind genau dann benachbart, wenn ihre Freiräume durch eine Öffnung in einem Hindernis aneinander grenzen. Die Grenzlinie, die im Freiraum zwischen zwei Hindernissen liegt, repräsentiert eine Tür. Türen werden als Hindernis klassifiziert und als solches in der Ergebniskarte eingetragen (Bilder 3.4 und 3.5)



(a) Ausgangskarte (b) Ergebnis

Bild 3.5: Ausgangskarte und Ergebnis

3.1.5 Ergebnis

Das angestrebte Ergebnis ist ein Bild der Grundrisskarte (Bild 3.5 a)), in das die detektierten Räume und Türen eingezeichnet sind (Bild 3.5 b)). Dabei soll die Einteilung der Freiräume der Grundrisskarte der intuitiven Einteilung durch den Menschen ähnlich sein. Jedem detektierten Raum soll eine eigene Farbe zugeordnet sein, mit der der Freiraum innerhalb des Raums ausgefüllt ist.

Zusätzlich zum Ergebnisbild soll die Grundrisskarte durch einen Kartendeskriptor D beschrieben werden, der Raumdeskriptoren $d \in D$ enthält.

Jedem detektierten Raum $r \in R$ soll ein Raumdeskriptor $d \in D$ zugeordnet sein, der Raumname d_{name}, Raumgröße d_{area}, Raumschwerpunkt d_{cog}, Raumfarbe d_{color} und eine Raumhinderniskontrollpunktmenge $Q \subseteq H$.

3.2 Algorithmus zur Detektion von Räumen

3.2.1 Belegung ermitteln

Zunächst ist es notwendig für jedes Rasterfeld $f \in F$ des Bildes mit den Bildpixelkoordinaten $I(x, y)$ anhand des Grauwertes $\eta(f) \in G$ an dieser Stelle im Bild die Belegung $O(f)$ zu entscheiden. Dies geschieht anhand eines Hindernisschwellwertes $\theta \in G$, der eine Grauwertschwelle zwischen 0 und 255 angibt.

Ist der Grauwert $\eta(f)$ eines Rasterfeldes $f \in F$ kleiner oder gleich dem Hindernisschwellwert $\theta \in G$, ist es **belegt**, ansonsten **frei** (Formel 3.1-3.5). Ist ein Rasterfeld $f_{I(x,y)}$ **belegt**, wird der Grauwert $\eta(f_{I(x,y)})$ des Rasterfeldes im Bild I auf 0 gesetzt. Ist das Rasterfeld **frei**, wird der Grauwert auf 255 gesetzt.

$$G = \{g | g \geq 0 \wedge g \leq 255\} \tag{3.1}$$

$$\theta \in G \tag{3.2}$$

$$f \in F \tag{3.3}$$

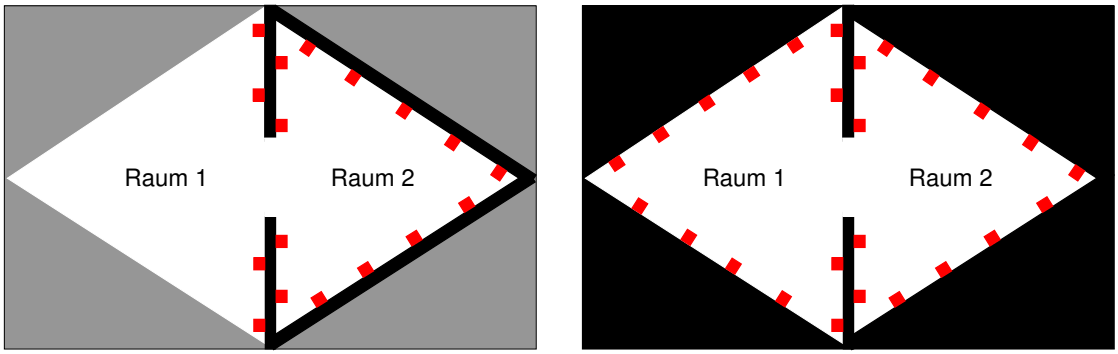
$$\eta(f_{I(x,y)}) \in G \tag{3.4}$$

$$O(f_{I(x,y)}) = \begin{cases} belegt & \text{wenn } \eta(f_{I(x,y)}) < \theta \text{ ;} \\ frei & \text{wenn } \eta(f_{I(x,y)}) \geq \theta \text{ .} \end{cases} \tag{3.5}$$

An dieser Stelle wird an die Karteneigenschaften die Anforderung gestellt, dass der Grauwert von Rasterfeldern in unbekanntem Gebiet kleiner sein muss als der Hindernisschwellwert. Unbekanntes Gebiet wird somit also als Hindernis eingestuft, um Freiräume an Übergängen zu unbekanntem Gebiet abgrenzen zu können.

In (Bild 3.6 a)) sind Hindernisse schwarz, Freiraum weiß und unbekanntes Gebiet grau eingefärbt. Wenn das unbekannte Gebiet nicht als Hindernis eingestuft werden würde, könnte zwar aus der Freifläche **Raum 2** detektiert werden, nicht jedoch **Raum 1**, da es zwar Hinderniskontrollpunkte an der Hindernisgrenze zu **Raum 2** geben würde, nicht aber in Richtung des unbekannten Gebietes. Nimmt man hingegen unbekanntes Gebiet als Hindernis an (Bild 3.6 b)), kann **Raum 1** bestimmt werden.

Zusammenhängende **belegte** Rasterfelder werden nun als Hindernissegment, zusammenhängende **freie** Rasterfelder als Freiraumsegment zusammengefasst.



(a) Unbekanntes Gebiet nicht als Hindernis (b) Unbekanntes Gebiet als Hindernis

Bild 3.6: Unbekanntes Gebiet und Hinderniskontrollpunkte

Für die weitere Verarbeitung ist es von Nutzen, wenn die Grenzen zwischen den Freiraum- und Hindernissegmenten möglichst **glatt** ist. Ist dies nicht der Fall, kann ein glättender Vorverarbeitungsschritt notwendig sein.

3.2.2 Vorverarbeitung

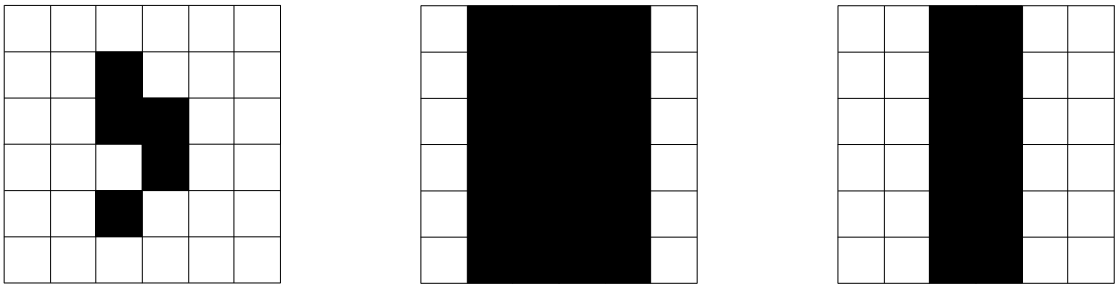
Daten von rauschanfälligen Sensoren wie Sonar oder Rauschen und Messungenauigkeiten aus SLAM [Mon03] bewirken, dass in der realen Welt **dichte** Hindernisse in eine Karte als **nicht dicht** eingetragen werden oder Hindernisse durch Punktewolken repräsentiert werden.

Dies kann beispielweise der Fall sein, wenn ein Robotersystem eine Karte auf Basis von Laserscans erzeugt und die Messungen in Form von einzelnen Punkten einträgt.

Daher ist es unter Umständen notwendig die Dichte von Hindernissen in der Karte nachträglich herzustellen, um das Ergebnis der Raumdetektion zu verbessern.

Sollte eine Karte Hindernisrepräsentation in Form von Punktewolken aufweisen oder nur Hindernisfragmente vorliegen (Bild 3.7 a)) statt geschlossene Hindernisse, kann eine glättende Vorverarbeitung in Form der morphologischen Operation Dilatation (Bild 3.7 b)) und einer anschließenden Erosion der Karte (Bild 3.7 c)) zu einer Verbesserung des Ergebnisses führen.

Dadurch wachsen Punktewolken und Hindernisfragmente zusammen und bekommen eine glattere Kante zu Freiräumen.



(a) vor Vorverarbeitung (b) nach Dilatation (c) nach Erosion

Bild 3.7: Belegungsänderung der Rasterfelder durch Vorverarbeitung mit $\delta_d = \delta_e = 1$

Dazu wird die Karte nacheinander mit zwei Filtermasken gefiltert. Zunächst wird eine Dilatation durchgeführt, die Hindernisse zusammenwachsen lässt. Anschließend folgt eine Erosion, die die verdickten Wände wieder schmälert, ohne zusammengewachsene Hindernisse wieder voneinander zu trennen.

Die Filtermaskengrößen δ_d der Dilatation und δ_e der Erosion sind unabhängig voneinander abhängig von der Hindernisrepräsentation in der Karte anzupassen. Die Länge und Breite der Filtermaske ergibt sich jeweils aus

$$\text{Länge = Breite = } 2 * \delta + 1 \text{ mit } \delta \in \mathbb{N}. \tag{3.6}$$

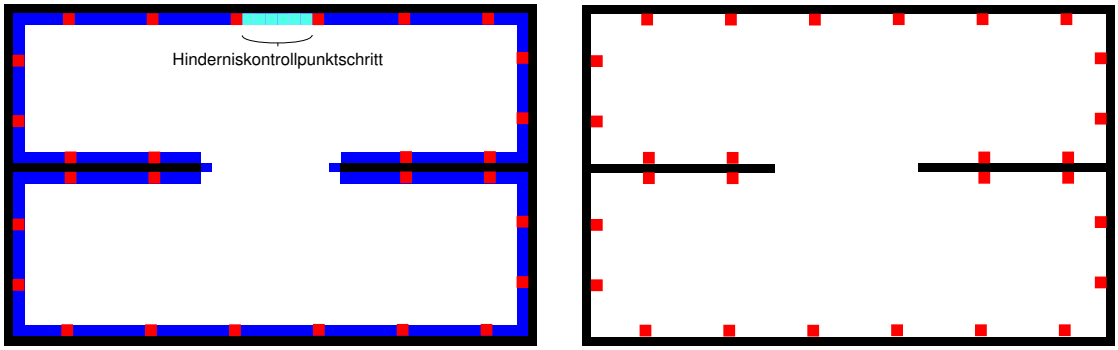
Im Dilatationsschritt wird abhängig von δ_d das aktuelle Rasterfeld $f \in F$ unter dem Zentrum der Filtermaske als **belegt** markiert, wenn es noch nicht belegt ist und mindestens ein Rasterfeld unter der Filtermaske **belegt** ist (vgl. [PH] S.127).

Im Erosionsschritt wird abhängig von δ_e das aktuelle Rasterfeld $f \in F$ unter dem Zentrum der Filtermaske als **frei** markiert, wenn es noch nicht frei ist und mindestens drei Rasterfelder unter der Filtermaske **frei** sind.

3.2.3 Hinderniskontrollpunkte erzeugen

Die Basis für die Detektion von Räumen in einem Gebäudegrundriss stellt die Menge der Hinderniskontrollpunkte $H \subseteq F$ dar.

Alle Rasterfelder $f \in F$, die auf der Grenze von einem Freiraumsegment zu einem Hindernissegment liegen, werden in die Menge der Hinderniskontrollpunktkandidaten



(a) Hinderniskontrollpunktschritt (b) Hinderniskontrollpunkte

Bild 3.8: Hinderniskontrollpunkte erzeugen

$\bar{H}$ geordnet aufgenommen (in Bild 3.8 a) blaue und rote Rasterfelder). Die Ordnung wird hergestellt, in dem alle Hinderniskontrollpunktkandidaten $\bar{h} \in \bar{H}$ ausgehend von einem Ursprung entlang der Hindernisgrenzen aufgesammelt werden.

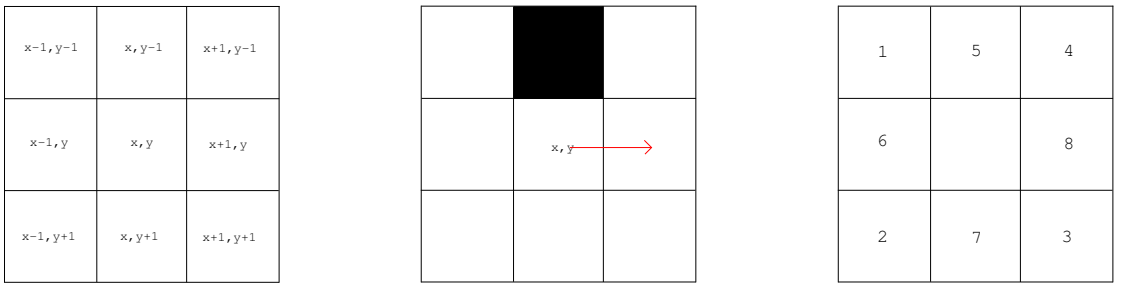
Der Test, ob ein Rasterfeld als Hinderniskontrollpunktkandidat $\bar{h}$ aufgesammelt wird oder nicht wird mit einer $3x3$ Filtermaske durchgeführt, in deren Mittelpunkt (x, y) das aufzunehmende **freie** Rasterfeld liegt (Bild 3.9 a)).

Existiert in dessen direkter Nachbarschaft ein **belegtes** Rasterfeld, wird das **freie** Rasterfeld unter dem Filtermaskenfeld mit den Filtermaskenkoordinaten (x, y) als Hinderniskontrollpunktkandidat $\bar{h}$ in $\bar{H}$ aufgenommen.

Anschließend wird der Filtermittelpunkt auf das nächste **freie** Rasterfeld unter der Filtermaske verschoben. Sind mehrere Rasterfelder in der direkten Nachbarschaft unter der Filtermaske **frei**, wird das Rasterfeld mit der höchsten Filtermaskenpriorität als neuer Maskenmittelpunkt gewählt (Bild 3.9 b)).

Die Prioritäten sind wie in Bild 3.9 c) in der Filtermaske verteilt. Dabei entscheidet das zuletzt aufgenommene Rasterfeld anhand der Filtermaskenpriorität die Richtung, in der das nächste aufzunehmende Rasterfeld gesucht wird - in Bild 3.9 b) in Richtung des roten Pfeils.

Wird keines gefunden, reißt die Suche an dieser Stelle ab und die Suche wird am anderen Ende der aktuellen Hindernisgrenze fortgesetzt, bis die Aufsammlung auch dort abreißt. Geschieht dies wird das nächste noch nicht überprüfte Rasterfeld auf einer Grenze von einem Freiraumsegment zu einem Hindernissegment als Ursprung der nächsten Suche



(a) Filtermaskekoordinaten (b) Filtermaskeverschiebung (c) Filtermaskepriorität

Bild 3.9: Filtermaske und Priorität

gewählt. Dabei wird das vorliegende Bild zeilenweise durchlaufen, bis ein passendes Rasterfeld gefunden wurde - in Bild 3.10 ist jeder neue Ursprung durch einen Pfeil gekennzeichnet.

Aus der Menge der Hinderniskontrollpunktkandidaten $\bar{H}$ werden nun abhängig von einem Parameter Hinderniskontrollpunktschritt λ jedes λ-te Rasterfeld als Hinderniskontrollpunkt h gewählt (Bild 3.8 c)) und in die Hinderniskontrollpunktmenge $H \subseteq \bar{H}$ aufgenommen (Formel 3.7).

$$H = \{h | h = \bar{h_i} \in \bar{H} \wedge i = \lambda a \wedge a, i, \lambda \in \mathbb{N}\} \tag{3.7}$$

Man könnte an dieser Stelle mit der gesamten Menge an Hinderniskontrollpunktkandidaten $\bar{H}$ weiterarbeiten. Dies ist aber nicht notwendig, da eine Menge von repräsentativen Hinderniskontrollpunktkandidaten ausreicht, um das gewünschte Ergebnis mit geringerem Aufwand zu erzielen.

Der Parameter Hinderniskontrollpunktschritt λ muss der Auflösung der Karte angepasst werden. Bei einer Karte mit hoher Auflösung muss ein höheres λ gewählt werden als bei der gleichen Karte mit niedriger Auflösung.

Außerdem ist anzumerken, dass aufgrund der gewählten Art der Prioritätsverteilung (Bild 3.9 c)) bei der Filtermaskenverschiebung (Bild 3.9 b)) leichte Variationen im Endergebnis der Raumdetektion auftreten können, wenn man das Ergebnis einer rotierten Karte mit dem Ergebnis derselben nicht rotierten Karte vergleicht.

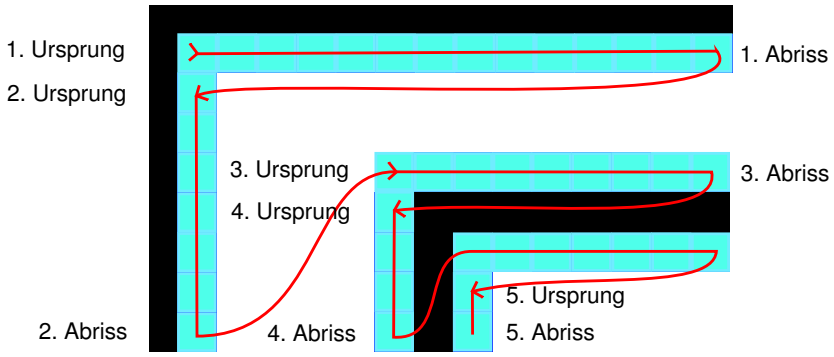


Bild 3.10: Hinderniskontrollpunktkandidatensuche

3.2.4 Sichtbarkeitstest

Auf der Menge der Hinderniskontrollpunkte H werden Sichtbarkeitstests $s(h_i, h_j)$ mit Sehstrahlen B durchgeführt.

Um die Sichtbarkeiten zu bestimmen werden eine Strahlausgangsmenge H_{Start} und eine Strahlzielmenge H_{Ziel} für Sehstrahlen definiert. Beide Mengen sind identisch mit der Hinderniskontrollpunktmenge H (Formel 3.8).

$$H = H_{Start} = H_{Ziel} \tag{3.8}$$

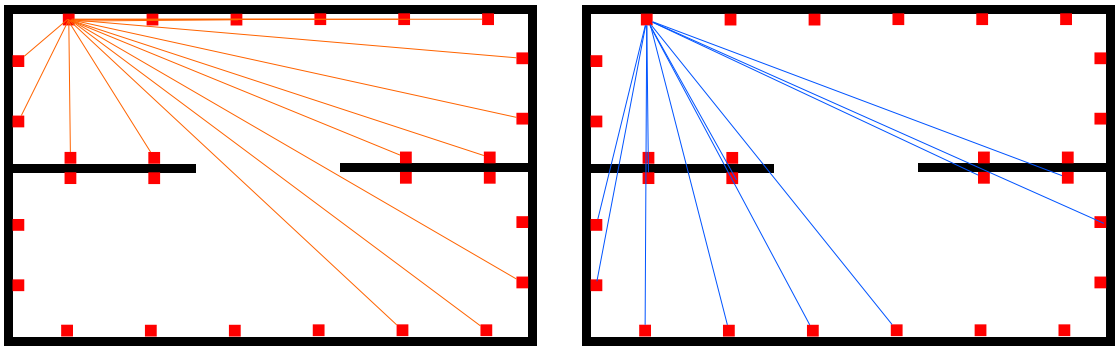
Ein Sehstrahl ist eine Bresenhamlinie von einem Element $h_{Start} \in H_{Start}$ der Strahlausgangsmenge zu einem Element $h_{Ziel} \in H_{Ziel}$ der Strahlzielmenge und hat ein Attribut **sehen**.

Die Information, dass ein Hinderniskontrollpunkt einen anderen Hinderniskontrollpunkt **sehen** kann, gibt einen Hinweis auf ihre Zusammengehörigkeit.

Die Zusammengehörigkeit wird in einem späteren Schritt auf der Basis aller Sichtbarkeitstests zwischen allen Hinderniskontrollpunkten näher untersucht.

Ein Sichtbarkeitstest wird für jedes Element der Strahlausgangsmenge zu jedem Element der Strahlzielmenge durchgeführt und bestimmt den Wert des Attributs **sehen** des Sehstrahls.

Der Wert ist 0, wenn der Sehstrahl ein **belegtes** Rasterfeld schneidet, also auf ein Hindernis trifft, oder 1, wenn der Sehstrahl kein **belegtes** Rasterfeld schneidet, also auf kein Hindernis trifft.



(a) sehen = 1 (b) sehen = 0

Bild 3.11: Sehstrahlen

In Bild 3.11 a) sind Sehtstrahlen mit dem Wert 1 des Attributs **sehen** in orange eingezeichnet. Bild 3.11 b) zeigt nur die Sehstrahlen in blau mit dem Wert 0 des Attributs **sehen**.

3.2.5 Adjazenzmatrix

Für den nächsten Schritt muss das bis hierher aufgebaute Modell mit Hinderniskontrollpunkten, Sehstrahlen und Sichtbarkeitstests in die Graphentheorie überführt werden.

In einem Graphen werden die Hinderniskontrollpunkte durch Knoten reräsentiert. Ein Sehstrahl entspricht einer ungerichteten Kante in einem ungerichteten Graphen mit dem Attribut **sehen**. Die Länge des kürzesten Weges zwischen zwei Knoten wird als **Distanz** als Anzahl der Kanten auf dem Weg angegeben (Bild 3.12) (vgl. [KN05] S.168f.).

Die Knoten an den Enden der Kante sind mit dieser adjazent, wenn das Attribut **sehen** an der Kante den Wert 1 hat und die Distanz zwischen den beteiligten Knoten genau 1 beträgt.

Der Wert des Attributs **sehen** wird in eine Adjazenzmatrix A eingetragen. Darin repräsentiert jede Zeile ein Element $h_{Start} \in H_{Start}$ der Strahlausgangsmenge und jede Spalte ein Element $h_{Ziel} \in H_{Ziel}$ der Strahlzielmenge. Dabei ist jedem Element der Strahlausgangsmenge genau eine Zeile und jedem Element der Strahlzielmenge genau eine Spalte in A zugewiesen. Dadurch sind jedem Element der Strahlausgangsmenge jeweils alle Elemente der Strahlzielmenge zugeordnet.

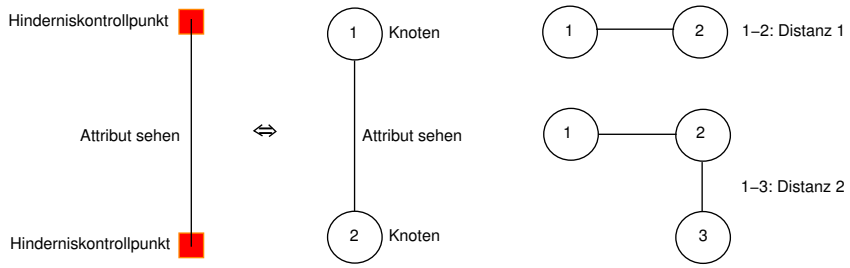


Bild 3.12: Graphrepräsentation

Die Position des Attributs **sehen** in der Matrix bestimmen die Ordnungsindizes i und j der beteiligten Elemente der Strahlausgangs- und der Strahlzielmenge. Der Wert des Attributes **sehen** an der Kante zwischen den Knoten $h_i \in H$ und $h_j \in H$ steht in der symmetrischen Adjazenzmatrix A sowohl an der Stelle (i, j) als auch an (j, i).

Da Strahlausgangsmenge und Strahlzielmenge identisch sind und der Sehstrahl ungerichtet ist, liefert der Sichtbarkeitstest zwischen dem j-ten Element der Strahlausgangsmenge und dem i-ten Element der Strahlzielmenge dasselbe Ergebnis und braucht daher nicht durchgeführt zu werden. Stattdessen kann in der Adjazenzmatrix an der Stelle (j, i) derselbe Wert eingetragen werden, der an der Position (i, j) bereits steht. Somit ist die Adjazenzmatrix symmetrisch (Formel 3.9) und braucht nur zur Hälfte berechnet zu werden.

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.9}$$

3.2.6 Distanz-1-Sichtbarkeitsmengenähnlichkeit

Nun wird für jedes Element $h_{Start} \in H_{Start}$ der Strahlausgangsmenge über die zugehörige Zeile i in der Adjazenzmatrix A eine Sichtbarkeitsmenge $\bar{S}_i$ definiert, die nur die Elemente $h_{Ziel} \in H_{Ziel}$ der Strahlzielmenge enthält, die dieses Element **sehen** kann,

also für die in der Adjazenzmatrix A der Wert 1 in der Zeile i in der zugehörigen Spalte j eingetragen ist (Formel 3.10).

$$\bar{S}_i = \{\bar{s_j} | \bar{s_j} \in H_{Ziel} \wedge A_{i,j} = 1\} \tag{3.10}$$

Die entstandenen Sichtbarkeitsmengen $\bar{S}$ werden im nächsten Schritt paarweise auf Ähnlichkeit ρ hin untersucht. Dabei errechnet sich die Ähnlichkeit $\rho(\bar{S_{i_1}}, \bar{S_{i_2}})$ zweier Sichtbarkeitsmengen $\bar{S_{i_1}}$ und $\bar{S_{i_2}}$ aus der Zeilenähnlichkeit u geteilt durch die Zeilenunähnlichkeit n (Formel 3.15).

Die Zeilenähnlichkeit u ist die Anzahl der Elemente für die gilt, dass in Zeile A_{i_1} der Wert $A_{i_1,j} = 1$ und in Zeile A_{i_2} der Wert $A_{i_2,j} = 1$ ist (Formel 3.13).

Die Zeilenunähnlichkeit n errechnet sich aus der Addition der Anzahl a_1 der Elemente von $\bar{S_{i_1}}$ (Formel 3.11) mit der Anzahl a_2 der Elemente von $\bar{S_{i_2}}$ (Formel 3.12) weniger die Zeilenähnlichkeit u (Formel 3.14). Ist $n = 0$ wird $\rho = 0$ gesetzt (Formel 3.15).

$$a_1 = \|\bar{S_{i_1}}\| \tag{3.11}$$

$$a_2 = \|\bar{S_{i_2}}\| \tag{3.12}$$

$$u = \|A_{i_1,j} = 1 \wedge A_{i_2,j} = 1\| \tag{3.13}$$

$$n = a_1 + a_2 - u \tag{3.14}$$

$$\rho(\bar{S_{i_1}}, \bar{S_{i_2}}) = \begin{cases} \frac{u}{n} & \text{wenn } n \neq 0; \\ 0 & \text{wenn } n = 0. \end{cases} \tag{3.15}$$

Vereinfacht ist die Zeilenähnlichkeit die Anzahl der Elemente $h \in H$ der Schnittmenge und die Zeilenunähnlichkeit die Anzahl der maximal möglichen Übereinstimmungen, also die Anzahl der Elemente der Vereinigungsmenge, der beiden Sichtbarkeitsmengen (Formel 3.16 und Bild 3.13).

$$\rho(\bar{S_{i_1}}, \bar{S_{i_2}}) = \frac{\bar{S_{i_1}} \cap \bar{S_{i_2}}}{\bar{S_{i_1}} \cup \bar{S_{i_2}}} \tag{3.16}$$

$$\rho \in \{p | p \in \mathbb{R} \wedge p \geq 0 \wedge p \leq 1\} \tag{3.17}$$

Im Beispiel (Bild 3.13) ist die Ähnlichkeit $\rho(\bar{S_1}, \bar{S_2})$ der Mengen $\bar{S_1}$ und $\bar{S_2}$ demnach $\frac{3}{13}$.

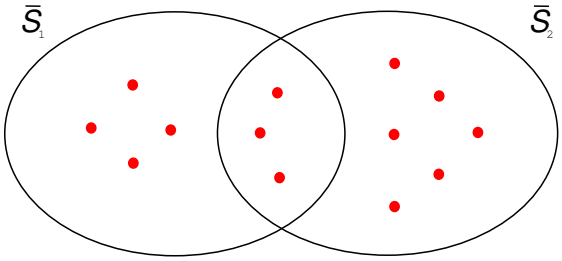


Bild 3.13: Beispiel Sichtbarkeitsmengenähnlichkeit

Der Wert der Sichtbarkeitsmengenähnlichkeit ρ ist somit ein Dezimalwert und liegt zwischen 0 und 1 (Formel 3.17).

Er wird in eine Ähnlichkeitsmatrix $\bar{A}$ an den Stellen (i, j) und (j, i) eingetragen. Darin repräsentieren Zeilen und Spalten jeweils die Sichtbarkeitsmengen $\bar{S}$ zu den Hindernis-kontrollpunkten $h \in H$.

In Formel 3.18 ist die zur Adjazenzmatrix A (Formel 3.9) korrespondierende Ähnlichkeitsmatrix $\bar{A}$ dargestellt.

$$\bar{A} = \begin{pmatrix} 1 & \frac{3}{4} & 1 & \frac{2}{4} & 0 & 0 \\ \frac{3}{4} & 1 & 1 & \frac{1}{4} & 0 & 0 \\ 1 & 1 & 1 & \frac{1}{4} & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.18}$$

Da die Sichtbarkeitsmenge $\bar{S}_i$ jeder Zeile über den Zeilenindex i eindeutig einem Hinderniskontrollpunkt h_{Start} der Strahlausgangsmenge und die Sichtbarkeitsmenge $\bar{S}_j$ jeder Spalte über den Spaltenindex j eindeutig einem Hinderniskontrollpunkt h_{Ziel} der Strahlzielmenge zugeordnet ist und $H = H_{Start} = H_{Ziel}$ gilt, können anhand der in der Ähnlichkeitsmatrix $\bar{A}$ eingetragenen Ähnlichkeitswerte $\rho(i, j)$ nun Rückschlüsse auf Zusammengehörigkeit von Hinderniskontrollpunkten $h \in H$ gezogen werden.

3.2.7 Raumrekonstruktion und Raumnetze

Für die Raumrekonstruktion auf Basis der Ähnlichkeitsmatrix $\bar{A}$ ist es nun nötig einen Sichtbarkeitsmengenähnlichkeitsschwellwert ψ festzulegen (Formel 3.19).

$$\psi = \{z | z \in \mathbb{R} \wedge z \geq 0 \wedge z \leq 1\} \tag{3.19}$$

Der Schwellwert ist ein Dezimalwert, liegt zwischen 0 und 1 und gibt an, ab welchem Ähnlichkeitswert ρ zwei Sichtbarkeitsmengen $\bar{S_{i_1}}$ und $\bar{S_{i_2}}$ zueinander ähnlich genug sind, damit sie als zusammengehörig gelten.

$$(\bar{S_{i_1}}, \bar{S_{i_2}}) = \begin{cases} \text{ähnlich} & \text{wenn } \rho(\bar{S_{i_1}}, \bar{S_{i_2}}) \geq \psi \text{ ;} \\ \text{unähnlich} & \text{wenn } \rho(\bar{S_{i_1}}, \bar{S_{i_2}}) < \psi \text{ .} \end{cases} \tag{3.20}$$

Aus der Zusammengehörigkeit zweier Sichtbarkeitsmengen kann geschlossen werden, dass auch die zugehörigen Hinderniskontrollpunkte $h \in H$, die sie repräsentieren, zusammengehören und somit in demselben Raum $r \in R$ aus der Menge aller Räume R liegen.

In der Raumrekonstruktion werden nun die den Sichtbarkeitsmengen zugeordneten Hinderniskontrollpunkte $h \in H$ als Raum $r \in R$ zusammengefasst.

Das Aufsammeln und Abarbeiten der Hinderniskontrollpunkte wird durch ein Verfahren auf Basis der Breitensuche (breadth first search = BFS) (vgl. [BVG00] S.332ff.) aus der Graphentheorie nach dem Prinzip First-In-First-Out realisiert. Dabei werden ausgehend von einem Ausgangselement e zunächst alle dessen Blätter untersucht und eingefügt, wenn sie dem Elter e **ähnlich** sind.

Anschließend wird die Suche für diese Blätter fortgesetzt, indem die Blätter des ersten Blattes untersucht und angefügt werden, wenn diese dem Element e ähnlich sind, dann die des zweiten Blattes und so weiter.

Die Suche bewegt sich also im Gegensatz zur Depth-First-Suche im Baum Ebene für Ebene (in Bild 3.14, E0 - E3) nach unten. Die Blätter in Bild 3.14 würden also in der Reihenfolge ihrer Nummerierung getestet und u.U. eingefügt.

Zunächst wird die erste noch zu keinem Raum zugeordnete Sichtbarkeitsmenge $\bar{S_{i_1}}$ als Basismenge C_Q gesucht und deren zugehöriger Hinderniskontrollpunkt $h \in H$ einer neuen Raumkontrollpunktmenge Q_r zugeordnet, die einen Raum $r \in R$ repräsentiert, und die Sichtbarkeitsmenge $\bar{S_{i_1}}$ als zugeordnet markiert.

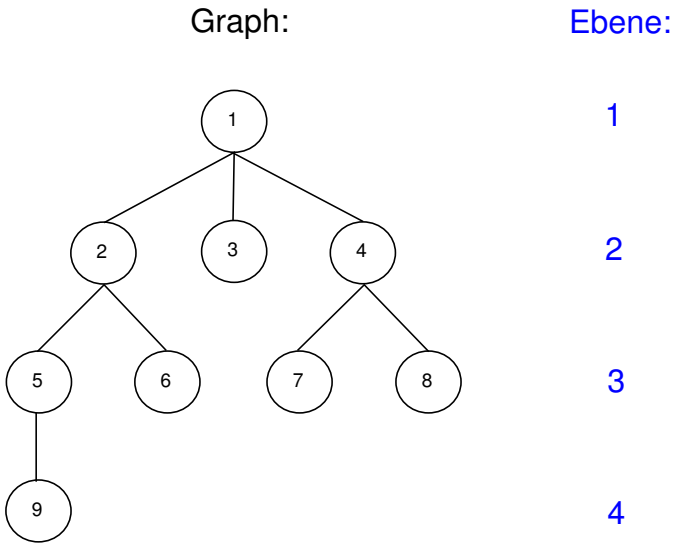


Bild 3.14: Breitensuche

Anschließend werden alle noch keinem Raum $r \in R$ zugeordneten Sichtbarkeitsmengen $\bar{S}_j$ auf ihre Ähnlichkeit $\rho(\bar{S}_{i_1}, \bar{S}_j)$ mit der Basismenge C_Q verglichen, in dem der Ähnlichkeitswert $\rho(\bar{S}_{i_1}, \bar{S}_j)$ in der Ähnlichkeitsmatrix $\bar{A}$ an der Stelle $A_{i_1,j}$ mit dem Sichtbarkeitsmengenähnlichkeitsschwellwert ψ verglichen wird (Formel 3.20).

Liegt der Ähnlichkeitswert ρ über dem Sichtbarkeitsmengenähnlichkeitsschwellwert ψ, wird der zur Sichtbarkeitsmenge $\bar{S}_j$ gehörende Hinderniskontrollpunkt $h \in H$ der Raumkontrollpunktmenge Q_r zugeordnet und die Zeilen- und Spaltensichtbarkeitsmenge $\bar{S}_j$ als zugeordnet markiert.

Ist die Basismenge mit allen anderen noch nicht zugeordneten Sichtbarkeitsmengen verglichen worden, wird die zum nächsten Element der aktuellen Raumkontrollpunktmenge gehördende Sichtbarkeitsmenge zur nächsten Basismenge, deren Ähnlichkeitswert wiederrum mit den Ähnlichkeitswerten aller noch keinem Raum zugeordneten Sichtbarkeitsmengen verglichen wird.

Dies geschieht solange bis das letzte Element der Raumkontrollpunktmenge abgearbeitet ist und dabei kein neues Element in die Raumkontrollpunktmenge aufgenommen wurde. Damit ist die Rekonstruktion eines Raumes $r \in R$ abgeschlossen.

Die Elemente der Raumkontrollpunktmenge Q_r beschreiben nun einen Raum $r \in R$ und werden in einen Raumdeskriptor $d_r \in D$ gespeichert.

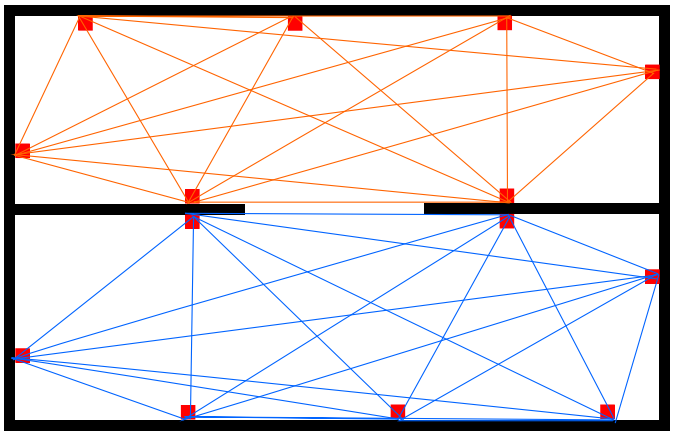


Bild 3.15: Raumnetze

Die Raumkontrollpunktmenge wird geleert, die nächste noch keinem Raum zugeordnete Sichtbarkeitsmenge als neue Basismenge gesucht und die Rekonstruktion des nächsten Raums durchgeführt. Wird keine noch keinem Raum zugeordnete Sichtbarkeitsmenge gefunden ist die Raumrekonstruktion abgeschlossen.

Rekonstruierte Räume, die aus weniger als drei Hinderniskontrollpunkten bestehen, können als Rauschen der Rekonstruktion eingestuft und verworfen werden, da sie keinen Raum aufspannen.

Die rekonstruierten Raumkontrollpunktmengen Q enthalten also Hinderniskontrollpunkte $h \in H$ (Formel 3.21) und sind zueinander disjunkt (Formel 3.22).

$$h \in Q \subseteq H \tag{3.21}$$

$$\forall Q_q, Q_r \subseteq H : Q_q \cap Q_r = \emptyset \tag{3.22}$$

Nach abgeschlossener Raumrekonstruktion wird nun jedem Raum $r \in R$ eine eindeutige Raumfarbe c_r zugeordnet und für jeden Raum ein Raumnetz in der zugeordneten Farbe aufgespannt (Bild 3.15), indem jeder Hinderniskontrollpunkt $h \in Q_r$ mit jedem anderen Hinderniskontrollpunkt $k \in Q_r$ durch eine Strecke verbunden wird. Die Raumfarbe c_r wird ebenfalls in dem dem Raum $r \in R$ zugehörigen Raumdeskriptor $d_r \in D$ als $d_{color} \in d_r$ gespeichert.

Dies geschieht nur genau dann, wenn die Anzahl der Hinderniskontrollpunkte $\|h \in Q_r\|$ größer ist als der Schwellwert Mindestraumgröße ϵ. Ist dies nicht der Fall, wird der Raum Q_r verworfen.

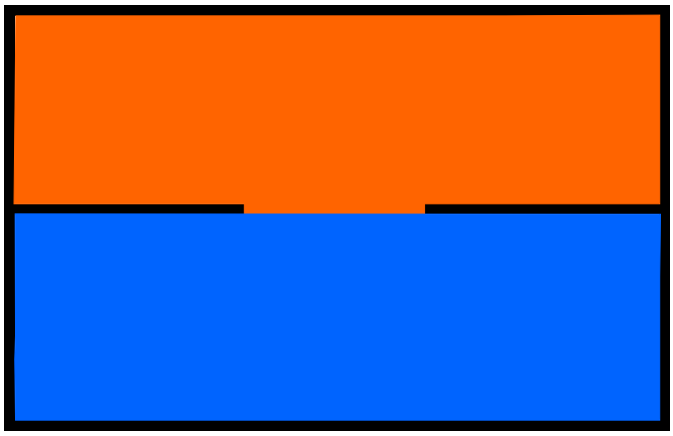


Bild 3.16: geflutete Raumnetze

3.2.8 Raumnetze fluten

Um zu jedem Rasterfeld $f \in F$, das im Freiraum einer Karte K liegt, eine Aussage über dessen Raumzugehörigkeit treffen und für jeden Raum $r \in R$ die Raumgröße bestimmen zu können, ist es notwendig jedes Rasterfeld mit einer eindeutigen Raumfarbe zu belegen.

Die Rasterfelder unter den Strecken der Raumnetze sind bereits mit der dem Raum zugehörigen Raumfarbe belegt und dienen zusammen mit den durch ein Hindernis belegten Rasterfeldern als Grenzen bei der Flutung der Raumnetze.

Zum Fluten wird nun das erste einem Freiraum zugeordnete und noch mit keiner Raumfarbe belegte Rasterfeld $f_1 \in F$ gesucht. Ist es gefunden wird die Raumfarbe dieses Rasterfeldes ermittelt, indem die Raumfarbe c_r des räumlich nahesten Rasterfeldes $f_n \in F$ ausgewählt wird, das bereits mit einer Raumfarbe belegt ist.

Jetzt wird das gesamte Freiraumsegment, in dem das neu belegte Rasterfeld f_1 liegt und das durch Hindernisse und die Strecken der Raumnetze begrenzt wird, mit dieser ermittelten Raumfarbe geflutet. Dabei wird jedem Rasterfeld dieses Freiraumsegments, dem noch keine Raumfarbe zugeordnet ist, die ermittelte Raumfarbe c_r zugewiesen.

Ist das Fluten dieses Freiraumsegments abgeschlossen wird das nächste einem Freiraum zugeordnete und noch mit keiner Raumfarbe belegte Rasterfeld gesucht und auf gleiche Weise verfahren.

Dies geschieht solange bis es kein Rasterfeld mehr gibt, das einem Freiraum zugeordnet ist und dem noch keine Raumfarbe zugewiesen wurde.

Auf diese Art ist jedes Rasterfeld $f \in F$, das in einem Freiraum liegt, genau einem Raum $r \in R$ zugeordnet.

3.2.9 Türen ermitteln

Auf Basis der gefluteten Raumnetze können Türen ermittelt werden.

Türen sind als Grenzlinie, die im Freiraum zwischen zwei benachbarten Räumen liegt, definiert, werden als Hindernis klassifiziert und als solches in die Ergebniskarte eingetragen (vgl. Kapitel 3.1.4).

Das bedeutet, Türen liegen auf der Grenze zweier benachbarter gefluteter Raumnetze. Da jedem Raumnetz eine eindeutige Raumfarbe zugeordnet ist, müssen zur Türdetektion nur genau die Rasterfelder gefunden werden, in deren direkter Nachbarschaft mindestens ein Rasterfeld mit einer anderen Raumfarbe belegt ist.

Dazu wird über jedes Rasterfeld eine $3x3$ Maske gelegt und die Raumfarben der Rasterfelder unter dieser Maske mit der Raumfarbe des Zentrumrasterfeldes verglichen. Wird ein Rasterfeld mit einer anderen Raumfarbe als die Raumfarbe des Zentrumrasterfeldes gefunden, wird das Zentrumrasterfeld als Hindernis markiert.

Somit gehört das Zentrumrasterfeld nicht mehr zu einem Raum, sondern ist Element einer Tür.

3.2.10 Resultat

Das Resultat (Bild 3.17) ergibt sich aus der binarisierten und vorverarbeiteten Karte K, in die die gefluteten Raumnetze (Kapitel 3.2.8) und die Türen (Kapitel 3.2.9) eingezeichnet sind, sowie den Raumdeskriptoren (Kapitel 3.2.12), die in einem Kartendeskriptor zusammengefasst werden, wenn das Resultat genau genug ist.

Ist das Resultat noch nicht genau genug, kann eine Verfeinerung (Kapitel 3.2.11) vorgenommen werden. Das Ergebnis der Verfeinerung ist dann das neue Resultat der Raumdetektion.

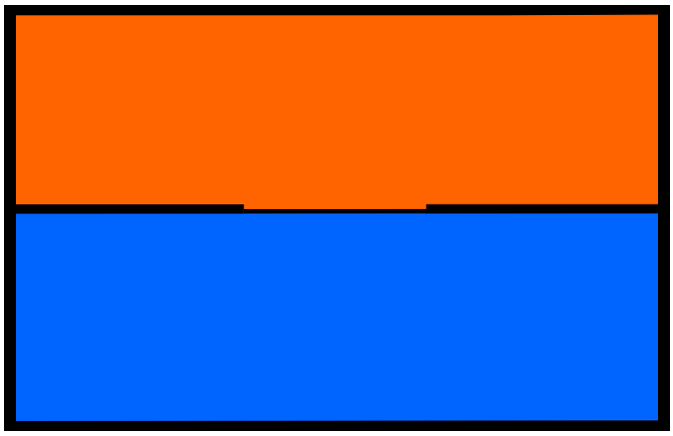


Bild 3.17: Resultat

Wenn das Resultat genau genug ist, können weitere Eigenschaften der Räume berechnet und in die zugehörigen Raumdeskriptoren aufgenommen werden (Kapitel 3.2.12).

3.2.11 Verfeinerung

Ist eine Verfeinerung gewünscht, wird der gesamte Algorithmus der Raumdetektion nochmals durchgeführt. Jedoch ist die Ausgangskarte nicht mehr die bisherige Ausgangskarte allein, denn in sie werden die bislang ermittelten Türen als Hindernisse eingetragen. Außerdem können die Parameter zur Raumdetektion abgeändert werden. Somit wird in einem Verfeinerungsschritt jeder bislang detektierte Raum näher untersucht und u.U. in weitere Räume unterteilt.

Für jeden Verfeinerungsschritt werden die bislang berechneten Räume und Raumdeskriptoren verworfen. Die Verfeinerung kann beliebig oft angewendet werden.

3.2.12 Raumdeskriptoren und Kartendeskriptor

Jeder Raumdeskriptor $d_r \in D$ repräsentiert einen Raum $r \in R$ einer Karte K und ist Element des Kartendeskriptors D_K.

Er enthält die Identifikation eines Raums als den Raumnamen d_{name}, die Raumfarbe d_{color} als RGB-Wert, die Raumgröße d_{area} als Anzahl der zum Raum gehörenden Rasterfelder, den Raumschwerpunkt d_{cog} und die Menge der Koordinaten der den Raum

beschreibenden Hinderniskontrollpunkte $h \in Q \subseteq H$.

Da die automatische Identifikation eines Raums nicht Teil dieser Arbeit ist, wird als Raumname der Nummerierungsindex des Raums $r \in R$ gewählt.

Die Raumgröße d_{area} wird bestimmt, indem die Rasterfelder gezählt werden, die in dem den Raum $r \in R$ beschreibenden gefluteten Raumnetz liegen.

Der Raumschwerpunkt d_{cog} ist eine Koordinate, die sich aus den Bildpixelkoordinaten (x, y) der den Raum $r \in R$ beschreibenden Hinderniskontrollpunkte $h \in Q_r$ ergibt. Dabei ist der x-Achsenwert der Quotient aus der Summe der x-Achsenwerte aller den Raum beschreibenden Hinderniskontrollpunkte und ihrer Anzahl. Der y-Achsenwert wird äquivalent dazu bestimmt.

3.3 Implementierung

Die praktische Umsetzung der wesentlichen Bestandteile des Algorithmus zur Raumdetektion wurde in der Programmiersprache C++ [Str00] implementiert, um eine Kompatibilität mit der **Programmierumgebung für die Muster-Analyse** (PUMA[1]) zu erreichen. Zur Programmierung der unterstützenden Funktionalitäten wurden C++ und das plattformunabhängige QT4.3 [BS06] von Trolltech eingesetzt.

Um eine Wiederverwertung des Codes an anderer Stelle zu erleichtern, sind die Teilalgorithmen des Gesamtalgorithmus in einzelnen allgemeineren Klassen (Kapitel 3.3.1) implementiert.

Das entwickelte Programm wird über ein Graphical User Interface (Kapitel 3.3.2) gesteuert, in dem auch die Zwischenergebnisse und das Resultat visualisiert werden.

3.3.1 Klassenbeschreibung

Im folgenden werden die Funktionalitäten der einzelnen Klassen kurz beschrieben. Zur detailierteren Dokumentation dienen die Kommentare im Quellcode. Modellierungen der einzelnen Klassen sind in Anhang A in den Abbildungen A.1-A.2 abgebildet.

Controller

Die Klasse Controller dient der Verknüpfung der Elemente des Graphical User Interface (Klasse Ui_MainWindow) mit den Funktionalitäten der Raumdetektion (Klasse Roomfinder) und den Funktionalitäten zum Arbeiten mit Dateien (Klasse Image und Klasse Text).

Außerdem verknüpft sie die einzelnen Schritte des Algorithmus miteinander und verwaltet die Inhalte der Zwischenergebnisse und des Resultats.

Die Klasse Controller ist in C++ und Qt entwickelt und nutzt Qt-Funktionalitäten von QMainWindow, QCloseEvent, QString, QFileDialog, QGraphicScene, QPixmap, QListWidgetItem und QMessageBox.

[1] `http://www.uni-koblenz.de/FB4/Institutes/ICV/AGPaulus/puma`

Roomfinder

Die Klasse Roomfinder implementiert den Algorithmus zur Raumdetektion. Jeder der einzelnen Schritte des Algorithmus (Kapitel 3.2.1-3.2.12) ist hier in einer eigenen Funktion gekapselt, die mit den für den jeweiligen Schritt notwendigen Parametern aufgerufen werden muss.

Die Klasse Roomfinder nutzt die Klassen Image, Text, ImageProcessing, Coordinate und Color und ist ausschließlich in C++ implementiert.

ImageProcessing

Die Klasse ImageProcessing stellt die für die Raumdetektion notwendigen Bildverarbeitungsmethoden zur Binarisierung, Dilatation, Erosion, Objektumrissermittlung, Berechnung von erweiterten Bresenhamgeraden, Flutung und Nachbarschaftsberechnung.

Weiterhin werden hier die Funktionalitäten zur Berechnung des Raumschwerpunkts und der Generierung von Markierungskreuzkoordinaten bereitgestellt.

Die Klasse ImageProcessing ist ausschließlich in C++ programmiert.

FileProcessing

Die Klasse FileProcessing gruppiert die Funktionalitäten zur Dateiverarbeitung, indem sie die Zugriffe auf Funktionalitäten der Klassen Image und Text ermöglicht.

Sie ist ausschließlich in C++ implementiert.

Coordinate und Operator

Die Klasse Coordinate implementiert karthesische zweidimensionale Koordinaten als Datenstruktur zum vereinfachten Arbeiten mit Koordinaten.

Die Klasse Operator definiert die grundlegenden Vergleichsoperationen **äquivalent** und **kleiner als** für die Datenstruktur Coordinate.

Beide Klassen sind ausschließlich in C++ programmiert.

Color

Die Klasse Color implementiert RGB-Farben als Datenstrukturen, sowie Methoden zur deren Konvertierung in Grauwerte. Sie dient dem vereinfachten Arbeiten mit Farben und Grauwerten.

Die Klasse Color ist in C++ und Qt entwickelt und nutzt Qt-Funktionalitäten von QColor und QRgb.

Image

Die Klasse Image stellt Bilder als Datenstruktur zur Verfügung. Außerdem verfügt sie über Methoden zum Laden, Speichern und Auslesen von Bildern.

Die Klasse Image ist in C++ und Qt programmiert und nutzt Qt-Funktionalitäten von QImage, QString, QPixmap und QRgb.

Text

Die Klasse Text implementiert Text als Datenstruktur und enthält Methoden zur Strukturierung von Text. Hauptsächlich dient sie zur Erstellung des Logs, das den Ablauf des Algorithmus dokumentiert, der Raumdeskriptoren und des Kartendeskriptors.

Sie ist in C++ und Qt entwickelt und nutzt Qt-Funktionalitäten von QObject, QString, QFile, QMessageBox, QTextStream, QStringList, QListWidget und QVariant.

Ui_MainMainWindow

Die Klasse Ui_MainWindow wird dynamisch aus der Datei roomfinder.ui beim Buildprozess durch Qt generiert und enthält die Bestandteile des Graphical User Interface und dessen Layout in Form von Qt-Elementen.

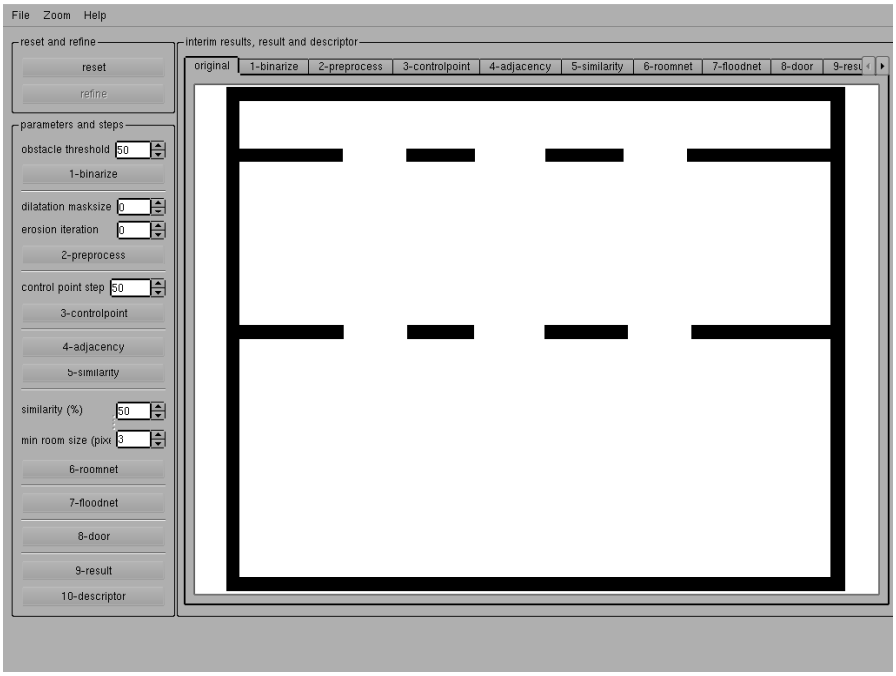


Bild 3.18: Snapshot Graphical User Interface

3.3.2 Graphical User Interface

Das entwickelte Programm kann über ein Graphical User Interface (Bild 3.18) gesteuert werden. Es besteht aus einer Menüleiste, Eingabe- und Steuerfeldern sowie einem Tabbereich, in deren Tabs die Zwischenergebnisse und das Resultat angezeigt werden.

Die Menüleiste bietet Interaktionsmöglichkeiten zum Öffnen eines Bildes, zum Speichern der Zwischenergebnisbilder und des Ergebnisbildes im Format *PNG* sowie zum Speichern des Logs und des Kartendeskriptors im Format *TXT*. Desweiteren kann über eine Zoomfunktion in die Zwischenergebnisse und das Ergebnis hinein bzw. heraus gezoomt werden.

Unter dem Menüeintrag *Help* ist eine detailierte Orientierungshilfe verfügbar, die in dieser Arbeit auch in Anhang C aufgelistet ist.

In den Eingabe- und Steuerfeldbereichen *reset and refine* und *parameters and steps* links können die Werte der Parameter Hindernisschwellwert, Filtermaskengröße Dilatation, Filtermaskengröße Erosion, Hinderniskontrollpunktschritt und Sichtbarkeitsmengenähnlichkeitsschwellwert gesetzt werden.

Für jeden der Schritte des Algorithmus gibt es hier einen Button, um den jeweiligen Schritt durchzuführen. Die Schrittzugehörigkeit der Buttons ergibt sich über die Buttonbeschriftung.

Wird dabei ein Schritt gestartet, der von einem Schritt abhängig ist, der vorher ausgeführt werden muss, wird dieser Schritt automatisch vorher ausgeführt.

Werden die Parameterwerte eines Schrittes geändert, kann der entsprechende Schritt durch Drücken des zugehörigen Buttons nochmals ausgeführt werden.

Soll ein Schritt nochmals ausgeführt werden, werden alle von nachfolgenden Schritten schon berechneten Ergebnisse verworfen, da sich die Vorraussetzungen zur deren Berechnung u.U. geändert haben.

Der Button *reset* setzt alle Berechnungen zurück.

Möchte man ein Resultat verfeinern, muss der Button *refine* gedrückt werden. Er löst ein *reset* aus und setzt das bisherige Resultat als neues Ausgangsbild.

Im Tabbereich *interim results, result and descriptor* werden die Zwischenergebnisse und das Resultat, sowie ein Arbeitslog und der Kartendeskriptor angezeigt. Jeder Tab repräsentiert das Ergebnis eines Schrittes des entwickelten Algorithmus.

Das Originalbild bzw. das Ausgangsbild für einen Verfeinerungsschritt ist im Tab *original* abgebildet.

4 Experimente und Ergebnisse

In diesem Kapitel werden Zwischenergebnisse und Ergebnisse des in Kapitel 3 vorgestellten Algorithmus zur Raumdetektion an repräsentativen Beispielen vorgestellt und bewertet (Kapitel 4.1). Anschließend wird der Algorithmus hinsichtlich seiner Performanz und Qualität betrachtet (Kapitel 4.2).

Als Kartenmaterial dient hier sowohl künstlich erzeugtes, als auch vom Robotersystem Robbie[1] der Arbeitsgruppe Aktives Sehen (AGAS)[2] autonom auf Basis von Laserdaten erstelltes Material.

4.1 Validierung und Verifikation anhand von Beispielen

In jedem der Unterkapitel 4.1.2-4.1.9 wird ein Beispiel erläutert und auf dessen Besonderheiten näher eingegangen.

Zugunsten der Übersichtlichkeit sind Log, Kartendeskriptor und die Bilder der Zwischenergebnisse und des Ergebnisses in Anhang B eingefügt. Dort listet das Log jeweils die Arbeitsschritte mit ihren Parameterwerten auf, die zu dem Kartendeskriptor, den Zwischenergebnissen und dem Ergebnis geführt haben. Die Struktur von Log, Kartendeskriptor und der Abbildung der Zwischenergebnisse und des Ergebnisses wird in Kapitel 4.1.1 beschrieben.

Um Wiederholungen zu vermeiden, wird in Beispiel 4.1.2 zunächst jeder dort wichtige Schritt des Algorithmus im Detail anhand der zugehörigen Abbildungen im Anhang B erläutert und in den folgenden Beispielen 4.1.3-4.1.9 darauf aufbauend nur noch auf

[1] `http://robots.uni-koblenz.de`
[2] `http://www.uni-koblenz.de/~agas`

wesentliche Unterschiede zwischen den Beispielen und Besonderheiten eingegangen.

4.1.1 Struktur der Beispiele

Log

Im Log geben die nicht eingerückten Bezeichnungen die Bezeichnungen für die einzelnen Schritte des Algorithmus an, wobei die eingerückten Bezeichnungen für Parameter bzw. Ergebnisse stehen.

Dabei repräsentiert *Filename* den Namen der verwendeten Karte, *Image Width* und *Image Height* die Größe der Karte in Rasterfeldern, *Count of Control Points* die Anzahl der generierten Hinderniskontrollpunkte $h \in H$, *Count of Rooms* die Anzahl der rekonstruierten Räume, *Obstacle Threshold* den Parameter Hindernisschwellwert θ, *Dilatation* die Filtermaskengröße δ_d für die Dilatation, *Erosion* die Filtermaskengröße d_e der Erosion, *Control Point Distance* den Hinderniskontrollpunktschritt λ, *Similarity Threshold* den Sichtbarkeitsmengenähnlichkeitsschwellwert ψ als Prozentwert und *Minimum Room Size* die Mindestraumgröße ϵ.

Weiterhin stehen *Open Image* für das Öffnen einer Karte, *Binarize* für den Schritt *Belegung ermitteln* (Kapitel 3.2.1), *Preprocess* für den Schritt *Vorverarbeitung* (Kapitel 3.2.2), *Control Point* für den Schritt *Hinderniskontrollpunkte erzeugen* (Kapitel 3.2.3), *Adjacency* für die Schritte *Sichtbarkeitstest* (Kapitel 3.2.4) und *Adjazenzmatrix* (Kapitel 3.2.5), *Similarity* für den Schritt *Distanz-1-Sichtbarkeitsmengenähnlichkeit* (Kapitel 3.2.6), *Roomnet* für den Schritt *Raumrekonstruktion und Raumnetze* (Kapitel 3.2.7), *Floodnet* für den Schritt *Raumnetze fluten* (Kapitel 3.2.8), *Door* für den Schritt *Türen ermitteln* (Kapitel 3.2.9), *Result* für den Schritt *Resultat* (Kapitel 3.2.10), *Refine* für den Schritt *Verfeinerung* (Kapitel 3.2.11), *Descriptor* für den Schritt *Raumdeskriptoren und Kartendeskriptor* (Kapitel 3.2.12), *Save File* für das Speichern eines Bildes und *Reset* für das Verwerfen der Zwischenergebnisse und des Ergebnisses.

Kartendeskriptor

Zu jedem Beispiel listet der Kartendeskriptor im Anhang B die einzelnen Raumdeskriptoren und deren Eigenschaften auf.

Dabei stehen *Room Name* für den Raumnamen d_{name}, *Room Size* für die Raumgröße d_{area} in Rasterfeldern, *Center of Gravity* für den Raumschwerpunkt d_{cog} als (x, y)-Bildkoordinate, *Room Color* für die Raumfarbe d_{color} als RGB-Wert ergänzt durch einen $Alpha$-Wert a und *Room Control Points* für die Raumhinderniskontrollpunkte $h \in Q \subseteq H$ in Form von (x, y)-Bildkoordinaten.

Abbildungen Zwischenergebnisse und Ergebnis

In der *Beispiel*-Abbildung zu jedem Beispiel im Anhang B illustrieren die Teilabbildungen (a)-(j) die Ergebnisse der einzelnen Schritte. Wurde ein Verfeinerungsschritt durchgeführt, sind in einer zusätzlichen *Verfeinerung*-Abbildung die Teilabbildungen (a)-(j) zu den einzelnen Schritten der Verfeinerung abgebildet.

Dabei ist jeweils *(a) Original* das geöffnete Bild als die Ausgangskarte, *(b) Binarisiert* das Ergebnis des Schrittes *Belegung ermitteln* (Kapitel 3.2.1), *(c) Dilatation und Erosion* das Ergbnis des Schrittes *Vorverarbeitung* (Kapitel 3.2.2), *(d) Hinderniskontrollpunkte* das Ergebnis des Schrittes *Hinderniskontrollpunkte erzeugen* (Kapitel 3.2.3), *(e) Adjazenzmatrix* das Ergebnis der Schritte *Sichtbarkeitstest* (Kapitel 3.2.4) und *Adjazenzmatrix* (Kapitel 3.2.5), *(f) Ähnlichkeitsmatrix* das Ergebnis des Schrittes *Distanz-1-Sichtbarkeitsmengenähnlichkeit* (Kapitel 3.2.6), *(g) Raumnetz* das Ergebnis des Schrittes *Raumrekonstruktion und Raumnetze* (Kapitel 3.2.7), *(h) Flutung* das Ergebnis des Schrittes *Raumnetze fluten* (Kapitel 3.2.8), *(i) Türen* das Ergebnis des Schrittes *Türen ermitteln* (Kapitel 3.2.9), *(j) Resultat*, *(j) Resultat vor Verfeinerung* und *(j) Resultat der Verfeinerung* das Ergebnis des Schrittes *Resultat* (Kapitel 3.2.10) und *(a) Bisheriges Resultat* das Ergebnis des Schrittes *Verfeinerung* (Kapitel 3.2.11).

4.1.2 Beispiel ohne Tür

In Beispiel B.1 soll die Funktionsweise des entwickelten Algorithmus an einem einfachen Beispiel erläutert werden.

Dem Beispiel liegt eine Ausgangskarte (Bild B.1 a)) zugrunde, die aus drei nicht verbundenen Freiraumsegmenten (*weiß*) und einem Hindernissegment (*schwarz*) besteht. In dem Hindernissegment gibt es keine Öffnungen und somit liegt offensichtlich eine Karte mit drei Räumen vor, zwischen denen es keine Türen gibt.

Nach der Binarisierung (Bild B.1 b)) sind nur noch die Grauwerte 0 und 255 im Bild vorhanden. Im Binarisierungsschritt wurden alle Grauwerte, die im Originalbild kleiner als der Hindernisschwellwert $\theta = 50$ (vgl. Kapitel B.1.1) waren, im Ergebnis der Binarisierung auf den Grauwert 0 und alle übrigen auf den Grauwert 255 abgebildet.

Dilatation und Erosion (Bild B.1 c)) sind in diesem Beispiel nicht notwendig, da das Hindernissegment nicht aus Fragmenten besteht und eine *glatte* Grenze zu den Freiraumsegmenten aufweist. Dazu wurde der Vorverarbeitungsschritt mit den Filtermaskengrößen für Dilatation und Erosion $\delta_d = \delta_e = 0$ (vgl. Kapitel B.1.1) durchgeführt.

Mit Hinderniskontrollpunktschritt $\lambda = 50$ (vgl. Kapitel B.1.1) ergeben sich für das Ergebnis (Bild B.1 d)) 123 Hinderniskontrollpunkte (vgl. Kapitel B.1.1).

Als Ergebnis der Sichtbarkeitstests ergibt sich eine Adjazenzmatrix (Bild B.1 e)), die nur die Werte 0 (*schwarz*) und 1 (*weiß*) enthält. Darin repräsentieren sowohl Spalten als auch Zeilen die Hinderniskontrollpunkte $h \in H$. Der Wert 0 (*schwarz*) gibt an, dass sich Zeilen- und Spaltenhinderniskontrollpunkt *sehen* können, während der Wert 1 (*weiß*) bedeutet, dass sich Zeilen- und Spaltenhinderniskontrollpunkt *nicht sehen* können.

In diesem Beispiel wurden die Hinderniskontrollpunkte geordnet aufgesammelt. Das bedeutet, dass die Hinderniskontrollpunkte *freiraumsegmentweise* in die Adjazenzmatrix gruppiert aufgenommen wurden. Von links nach rechts sowie von oben nach unten befinden sich die Hinderniskontrollpunkte in der Reihenfolge: Hinderniskontrollpunkte des obersten Freiraumsegments, Hinderniskontrollpunkte des mittleren Freiraumsegments, Hinderniskontrollpunkte des untersten Freiraumsegments aus Bild B.1 c).

Das Bild des Adjazenzmatrix (Bild B.1 e)) zeigt, dass sich alle Hinderniskontrollpunkte eines Freiraumsegments gegenseitig sehen können und dass es zwischen den Freiraumsegmenten keine Verbindungen gibt, da kein Hinderniskontrollpunkt eines Freiraumsegments einen Hinderniskontrollpunkt eines anderen Freiraumsegments sehen kann.

Die Ähnlichkeitsmatrix (Bild B.1 f)) beweist nun, dass es tatsächlich so ist. Sie hat den gleichen Spalten- und Zeilenaufbau wie die Adjazenzmatrix und enthält nur die Werte 0 (*schwarz*) und 255 (*weiß*). Dabei bedeutet der Wert 0, dass sich die von Ähnlichkeitsmatrixzeile und Ähnlichkeitsmatrixspalte repräsentierten Zeilen aus der Adjazenzmatrix (Bild B.1 e)) völlig unähnlich sind. Je näher ein Wert in der Ähnlichkeitsmatrix dem Wert 255 kommt, desto ähnlicher sind sich die von Ähnlichkeitsmatrixzeile und Ähnlichkeitsmatrixspalte repräsentierten Zeilen aus der Adjazenzmatrix. Das bedeutet nun,

dass die Zeilen aus der Adjazenzmatrix, für die der Wert 255 in der Ähnlichkeitsmatrix eingetragen ist, völlig gleich sind.

Mit dem Sichtbarkeitsmengenähnlichkeitsschwellwert $\psi = 50\% \Rightarrow Grauwert = 127$ und der Mindestraumgröße $\epsilon = 5$ (vgl. Kapitel B.1.1) ergibt sich für das Ergebnis der Raumrekonstruktion, dass die Hinderniskontrollpunkte, die von den Zeilen den Adjazenzmatrix, für die in der Ähnlichkeitsmatrix ein Wert eingetragen ist, der größer ist als der vom Sichtbarkeitsmengenähnlichkeitsschwellwert ψ repräsentierte Grauwert, zu einem Raum zusammengefasst werden können.

Daraus ergeben sich die drei Raumnetze in Bild B.1 g), die von den zugehörigen Hinderniskontrollpunkten aufgespannt werden und drei Räume repräsentieren. Hier ist jedem der drei entstandenen Räume eine eindeutige Farbe zugeordnet.

Die noch bestehenden Zwischenräume sind in Bild B.1 h) geflutet. Somit ist jedes Rasterfeld der Karte aus Bild B.1 c) genau einem Raum zugeordnet.

Da die Farbbereiche der entstandenen Räume in Bild B.1 h) nicht aneinandergrenzen, ergeben sich keine Türen, die in Bild B.1 i) eingetragen werden müssten.

Aus Bild B.1 h) und Bild B.1 i) entsteht das Resultat (Bild B.1 j)) und der Kartendeskriptor (Kapitel B.1.2), der die Eigenschaften der drei rekonstruierten Räume enthält.

4.1.3 Beispiel mit einer Tür

Für Beispiel B.2 wird in die Karte aus Beispiel B.1 eine Tür zwischen dem obersten und dem mittleren Freiraumsegment eingetragen (Bild B.2 a)).

Dadurch verschmelzen das oberste und mittlere Freiraumsegment aus Bild B.1 c) aus Beispiel B.1 zu einem Freiraumsegment in Bild B.2 c), das es nun wieder an der Stelle der Tür zu trennen gilt.

Mit dem gleichen Hinderniskontrollpunktschritt $\lambda = 50$ (vgl. Kapitel B.2.1) wie in B.1 ergibt sich mit 120 (vgl. Kapitel B.2.1) eine ähnliche Anzahl der Hinderniskontrollpunkte $h \in H$ wie in B.1. Die Koordinaten variieren leicht (Bild B.2 d)), da die Reihenfolge der Generierung der Hinderniskontrollpunkte sich unterscheidet.

Wurden in B.1 zunächst die Hinderniskontrollpunkte des obersten, dann des mittleren

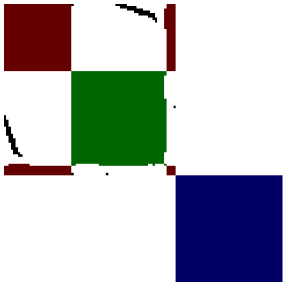


Bild 4.1: gefärbte Adjazenzmatrix

und anschließend des untersten Freiraumsegments aufgesammelt, ändert sich diese Reihenfolge in B.2, da das oberste und mittlere Freiraumsegment zu einem einzigen verschmolzen sind, welches nun aus einem oberen und einem unteren Teil besteht.

In B.2 werden also zunächst Hinderniskontrollpunkte des oberen Teils des oberen Freiraumsegments, dann alle des unteren Teils des oberen Freiraumsegments, dann die restlichen noch nicht aufgesammelten Hinderniskontrollpunkte des oberen Teils des oberen Freiraumsegments und anschließend alle des unteren Freiraumsegments aufgesammelt (vgl. Kapitel 3.2.3).

In exakt dieser Reihenfolge der Aufsammlung der Hinderniskontrollpunkte sind nun die Spalten und Zeilen der Adjazenzmatrix (Bild B.2 e)) geordnet. In Bild 4.1 sind die Bereiche der Hinderniskontrollpunkte farbig den Freiraumsegmenten zugeordnet. Hinderniskontrollpunkte, in deren Spalte bzw. Zeile die Eigenschaft *sehen rot* eingetragen ist, liegen im oberen Teil des oberen Freiraumsegments. Äquivalent dazu ist die Eigenschaft *sehen* für Hinderniskontrollpunkte aus dem unteren Teil des oberen Freiraumsegments *grün* und für Hinderniskontrollpunkte des unteren Freiraumsegments *blau* eingefärbt eingetragen.

In der Adjazenzmatrix ist auch schon erkennbar, dass zwei der in B.1 getrennten Freiraumsegmente zu einem verschmolzen und durch eine Öffnung in einem Hindernis miteinander verbunden sein müssen, da jetzt manche Hinderniskontrollpunkte des oberen Teils des oberen Freiraumsegments manche Hinderniskontrollpunkte des unteren Teils des oberen Freiraumsegments *sehen* können. Für sie wird in der Adjazenzmatrix der Wert 0 (*schwarz*) eingetragen. Auch in Bild 4.1 sind diese Stellen *schwarz* eingefärbt.

Da sich die Zeilen der Adjazenzmatrix, die das obere Freiraumsegment repräsentieren,

voneinander unterscheiden, entstehen durch deren Ähnlichkeitsvergleich in der Ähnlichkeitsmatrix (Bild B.2 f)) Werte zwischen 0 und 255. Anhand eines Sichtbarkeitsmengenähnlichkeitsschwellwert $\psi = 50\% \Rightarrow Grauwert = 127$ und der Mindestraumgröße $\epsilon = 5$ (vgl. Kapitel B.2.1) werden nun die Hinderniskontrollpunkte zu drei Räumen zusammengefasst.

Anschließend werden die drei Raumnetze aufgespannt (Bild B.2 g)). In dem Bild der gefluteten Raumnetze (Bild B.2 h)) kann nun eine Tür in dem Bereich detektiert werden, in dem das obere geflutete Raumnetz an das mittlere geflutete Raumnetz stößt.

Diese Tür wird in Bild B.2 i) als Hindernis eingetragen und es entsteht durch Kombination von Bild B.2 h) und Bild B.2 i) das Resultat (Bild B.2 j)) und der Kartendeskriptor (Kapitel B.2.2).

4.1.4 Beispiel mit mehreren Türen und Raumeinrichtungsgegenständen

Für Beispiel B.3 sind in die Ausgangskarte von Beispiel B.2 weitere Türen sowie Raumeinrichtungsgegenstände in Form von in den Räumen liegenden Hindernisblöcken eingetragen worden. Auch durch die zusätzlichen Türen entstehen mehrere Hindernissegmente. Die erwartete Anzahl von zu rekonstruierenden Räumen in der Ausgangskarte liegt weiterhin bei drei.

Die Änderungen an der Ausgangskarte wirken sich abermals auf die Reihenfolge der generierten Hinderniskontrollpunkte aus (Bild B.3 d). Zunächst werden die Hinderniskontrollpunkte entlang des äußersten, größten Hindernissegments aufgesammelt und anschließend Hindernissegment für Hindernissegment von oben links nach unten rechts in Bild B.3 d) fortgefahren.

Das bedeutet nun für die Adjazenzmatrix (Bild B.3 e)), dass die die Hinderniskontrollpunkte $h \in H$ repräsentierenden Spalten bzw. Zeilen räumlich zusammengehöriger Hinderniskontrollpunkte größtenteils nicht mehr benachbart sind und weiterhin, dass aus der Adjazenzmatrix allein keine direkten Schlüsse bzgl. der räumlichen Einteilung mehr gezogen werden können, wie das noch in B.1 und B.2 möglich war.

Mit dem Sichtbarkeitsmengenähnlichkeitsschwellwert $\psi = 50\% \Rightarrow Grauwert = 127$ und der Mindestraumgröße $\epsilon = 4$ (vgl. Kapitel B.3.1) liefert die Auswertung der aus

der Adjazenzmatrix entstandenen Ähnlichkeitsmatrix (Bild B.3 f)) die Raumnetze von drei rekonstruierten Räumen (Bild B.3 g)), die die Raumeinrichtungsgegenstände einschließen.

Interessant ist an dieser Stelle, dass in Bereichen, in denen später die Türen entstehen, keine Strecken der Raumnetze liegen. Somit erscheinen sie in Bild B.3 g) als größere weiße Flächen.

Diese Flächen, die im nächsten Schritt zur Raumzuordnung mit der Farbe eines ihrer angrenzenden Nachbarraumnetze geflutet werden, sind tatsächlich Bereiche in der Karte, die keinem Raum eindeutig zugeordnet werden können, da sie außerhalb von Raumnetzen und damit zwischen diesen liegen.

Sie werden trotzdem geflutet, um Türen an Grenzen gefluteter Raumnetze erzeugen zu können (Bilder B.3 h) und i)), mit deren Hilfe das Resultat (Bild B.3 j)) und der Kartendeskriptor (Kapitel B.3.2) entstehen.

4.1.5 Beispiel mit mehreren Räumen und notwendiger Verfeinerung

Das nächste Beispiel B.4 demonstriert die Notwendigkeit eines Verfeinerungsschritts.

Im ersten Durchlauf des Algorithmus mit Bild B.4 a) als Ausgangskarte können nicht alle erwarteten Räume rekonstruiert werden. Hier kommt die Auswirkung der globalen Anwendung des Sichtbarkeitsmengenähnlichkeitsschwellwertes ψ zum Tragen.

Im Resultat des ersten Durchlaufs (Bild B.4 j)) hängen der Korridor in der Mitte und der große Raum rechts noch zusammen (beide *grün*), obwohl man eine Trennung erwartet hätte. Gleiches gilt für die zusammenhängenden Räume in *Blau* und *Rot*.

Um alle Freiraumsegmente in die erwarteten Räume zu zerlegen, wäre es notwendig, für verschiedene Bereiche in der Ausgangskarte bei der Auswertung der Ähnlichkeitsmatrix des ersten Durchlaufs (Bild B.4 f)) mehrere Sichtbarkeitsmengenähnlichkeitsschwellwerte ψ anzuwenden. Da im Vorfeld nicht entscheidbar ist, wann welcher Wert angewandt werden müsste, ist eine zweistufige Anwendung des Algorithmus notwendig.

Im ersten Durchlauf werden die Freiraumsegmente in kleinere Bereiche unterteilt und

die Karte mit den eingetragenen bislang ermittelten Türen (Bild B.4 i)) als Ausgangskarte (Bild B.5 a)) für den Verfeinerungsschritt verwendet.

Dadurch, dass es nun mehrere kleinere Freiraumsegmente gibt, können Hinderniskontrollpunkte eines bereits abgetrennten Freiraumsegments keine Hinderniskontrollpunkte eines anderen Freiraumsegments mehr sehen. In der Adjazenzmatrix (Bild B.5 e)) gibt es dadurch weniger Stellen, für die gilt, dass ein Hinderniskontrollpunkt eines Raumes einen Hinderniskontrollpunkt eines anderen Raums sehen kann. Weiterhin enthält die Ähnlichkeitsmatrix daraufhin weniger Rauschen und ist kontrastreicher (Bild B.5 f)).

Der erhöhte Kontrast in der Ähnlichkeitsmatrix führt dazu, dass Freiraumsegmente besser unterteilt werden können. Durch die lokalere Anwendung des Sichtbarkeitsmengenähnlichkeitsschwellwertes ψ im Verfeinerungsschritt kann dieser Schwellwert zur besseren Unterteilung der Freiraumsegmente auch erhöht werden.

In diesem Beispiel ist dies mit $\psi = 50\% \Rightarrow Grauwert = 127$ im ersten Durchlauf auf $\psi = 75\% \Rightarrow Grauwert = 191$ im Verfeinerungsschritt geschehen (vgl. Kapitel B.4.1).

Im Verfeinerungsschritt wurde außerdem die Anzahl der Hinderniskontrollpunkte durch einen kleineren Hinderniskontrollpunktschritt von $\lambda = 50$ auf $\lambda = 40$ erhöht (vgl. Kapitel B.4.1), um eine genauere Abtastauflösung für die Tests auf Sichtbarkeit zu erhalten.

Die Auswertung der gefluteten Raumnetze (Bild B.5 h)) liefert die noch fehlenden Türen (Bild B.5 i)), Räume (Bild B.5 j)) und den Kartendeskriptor (Kapitel B.4.2).

4.1.6 Beispiel mit notwendiger Vorverarbeitung

Beispiel B.5 liegt eine *verrauschte* Ausgangskarte (Bild B.6 a)) zugrunde, die eine Binarisierung und eine Vorverarbeitung notwendig macht.

Die Hindernisse sind in Bild B.6 a) nicht dicht. Das bedeutet, dass ein Hindernissegment von vielen kleineren Hindernissegmenten (Fragmenten) repräsentiert wird, anstatt von einem einzigen. Es liegt sowohl Rauschen innerhalb von Hindernissegmenten vor, als auch im Übergangsbereich zwischen Hindernissegmenten zu Freiraumsegmenten. Auch innerhalb der Freiraumsegmente tritt Rauschen auf.

Das ist deshalb problematisch, da bei der Generierung der Hinderniskontrollpunkte die Hinderniskontrollpunkte entlang der Umrißlinien von Hindernissen aufgesammelt wer-

den und somit Hinderniskontrollpunkte an Fragmenten liegen, anstatt an einer glatten Übergangskante eines dichten Hindernissegments zu einem Freiraumsegment.

Würden weder Binarisierung noch Vorverarbeitung durchgeführt, würden die vielen Fragmente dazu führen, dass bei den Tests auf Sichtbarkeit viele Hinderniskontrollpunkte nur ganz wenige andere Hinderniskontrollpunkte *sehen* könnten, die sie für die Raumrekonstruktion notwendigerweise *sehen* müssten.

Besser wären für die weitere Verarbeitung glatte Kanten im Übergangsbereich von einem Hindernissegment zu einem Freiraumsegment sowie minimiertes Rauschen innerhalb von Hindernissegmenten und Freiraumsegmenten.

Das Rauschen innerhalb von Freiraumsegmenten wird durch die Binarisierung minimiert. Anhand des Hindernisschwellwertes $\theta = 50$ (vgl. KapitelB.5.1) wird Rasterfeldern, deren Grauwert kleiner als θ ist, der Grauwert 0 zugewiesen und sie dadurch als Hindernis eingestuft. Ist der Grauwert größer oder gleich θ, so wird der Grauwert 255 zugewiesen und das Rasterfeld als Freiraum eingestuft.

Die Minimierung des Rauschens innerhalb der Hindernissegmente und im Übergangsbereich von Hindernissegmenten zu Freiraumsegmenten wird durch eine zweistufige Vorverarbeitung erreicht. Im ersten Schritt wird abhängig von der Filtermaskengröße $d_d = 12$ (vgl. Kapitel B.5.1) eine Dilatation durchgeführt, bei der die Fragmente mit den Hindernissegmenten zusammenwachsen und Rauschen innerhalb von Hindernissegmenten minimiert wird. Der zweite Schritt besteht aus einer Erosion, die anhand der Filtermaskengröße $d_e = 12$ (vgl. Kapitel B.5.1) auf das Ergebnis der Dilatation angewandt wird.

Im Ergebnis der Vorverarbeitung (Bild B.6 c)) hat das Hindernissegment glattere Übergangskanten zu Freiraumsegmenten und das Rauschen ist minimiert. Dadurch werden Sehstrahlen, die bei den Tests auf Sichtbarkeit von und zu den generierten Hinderniskontrollpunkten (Bild B.6 d)) gebildet werden, nicht mehr an Fragmenten abgefangen und es entsteht eine brauchbare Adjazenzmatrix (Bild B.6 e)).

In diesem Beispiel B.5 treten weitere Besonderheiten auf.

Zunächst ist in Bild B.6 g) etwa in der Mitte des Bildes ein sehr großer Bereich, der keinem Raumnetz zugeordnet werden konnte. Intuitiv würde man diesen Bereich sogar als eigenständigen Raum einstufen. Im ersten Durchlauf des Algorithmus wird dies nicht getan. Stattdessen repräsentiert dieser Bereich eine sehr große Tür und es ist von der

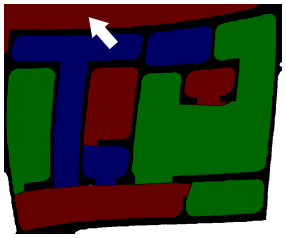


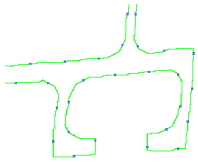
Bild 4.2: Raum außerhalb

Orientierung der Karte abhängig, zu welchem der benachbarten Raumnetze dieser Bereich hinzugefügt (Bild B.6 h)) und wo anschließend eine Tür erzeugt wird (Bild B.6 i)).

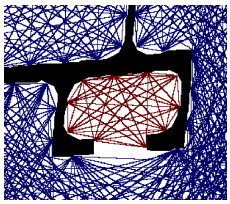
In B.5 wurde dieser Bereich dem *grünen* Raum zugeordnet, da das erste nicht belegte Rasterfeld im Türbereich näher am Raumnetz des *grünen* Raums liegt als am Raumnetz des *blauen* Raums (Bild B.6 h)). Dadurch wird ein Verfeinerungsschritt notwendig, um u.a. diesen eigenständigen Raum vom grünen Raum abzuspalten (B.7 j)) und einen genaueren Kartendeskriptor (B.5.2) zu erstellen.

Zwei weitere Besonderheiten sind im Resultat des Verfeinerungsschrittes (Bild B.7 j)) ersichtlich. Zum einen ist außerhalb des Grundrisses ein Raum entstanden. In Bild 4.2 ist das der *rote* Raum, auf den der weiße Pfeil zeigt. Das kommt daher, dass der Bereich außerhalb des Grundrisses bei der Binarisierung (Bild B.7 b)) als Freiraum eingestuft wird, da der Grauwert der Rasterfelder größer als oder gleich dem Hindernisschwellwert $\theta = 50$ (vgl. Kapitel B.5.1) des Verfeinerungsschrittes ist. Durch die Krümmung des Hindernissegments, das an diesen Bereich grenzt, können sich die in diesem Freiraumsegment generierten Hinderniskontrollpunkte *sehen* und es entsteht ein Raumnetz (Bild B.7 g)), da die Anzahl der Hinderniskontrollpunkte, die dieses Raumnetz aufspannen, größer ist als die Mindestraumgröße $\epsilon = 5$ (vgl. Kapitel B.5.1).

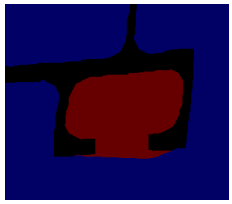
Zur Lösung dieses Problems könnte man nun die Mindestraumgröße ϵ anheben oder diesen Bereich mit einer anderen Belegung versehen. Da die Mindestraumgröße genauso wie der Sichtbarkeitsmengenähnlichkeitsschwellwert ψ global angewendet wird, würde das Anheben der Mindestraumgröße dazu führen, dass auch kleine Räume innerhalb des Grundrisses verschwänden, wenn die Anzahl der sie repräsentierenden Hinderniskontrollpunkte unter die Mindestraumgröße fallen würde.



(a) Hinderniskontrollpunkte

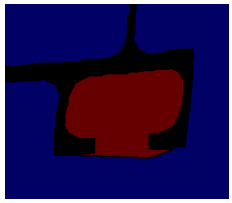


(b) Raumnetz



(c) Flutung

(d) Tür erzeugen



(e) Ergebnis

Bild 4.3: ungenaue Tür

Die Besonderheit einer *ungenauen Tür* ist in Bild 4.3 aufgezeigt.

Dort ist die Entstehung des *roten* Raums oben rechts in Bild B.6 j) im Detail skizziert. Schon beim Schritt der Erzeugung der Hinderniskontrollpunkte (Bild 4.3 a)) werden die Weichen für die spätere Ungenauigkeit der Position der Tür gestellt.

Die Hinderniskontrollpunkte des später *blau* gefärbten Raums, zwischen denen die Tür entstehen wird, liegen so am Hindernissegment, dass sie sich gegenseitig *nicht sehen* können. Dadurch kann beim Aufspannen des Raumnetzes keine Strecke zwischen diesen beiden Hinderniskontrollpunkten generiert werden, ohne ein Hindernis zu schneiden (Bild 4.3 b)). Da sich nun das erste nicht belegte Rasterfeld im Türbereich näher am *roten* Raumnetz befindet, liegt die Grenze zwischen den gefluteten Raumnetzen in einem Bereich des Freiraumsegments, der zum *blauen* Raum gehört (Bild 4.3 c)).

Daraufhin wird auch die Tür im Bereich des *blauen* Raums erstellt (Bild 4.3 d) und es entsteht das Ergebnis (Bild 4.3 e)), in dem ein Teil des *roten* Raums in einem Bereich liegt, der eigentlich zu dem *blauen* Raum gehört. Daraus resultieren eine etwas ungenaue Position der Tür und eine ungenaue Zuordnung der Rasterfelder zu einem Raum.

4.1.7 Beispiel mit von Robbie erzeugter Karte

Eine vom Robotersystem Robbie[3] der Arbeitsgruppe Aktives Sehen (AGAS)[4] autonom auf Basis von Laserdaten erstellte Karte liegt Beispiel B.6 zugrunde.

Wie schon in Beispiel B.5 ist hier eine Vorverarbeitung notwendig, da Hindernisse Rauschen enthalten und die Übergangskanten von Hindernissegmenten zu Freiraumsegmenten nicht *glatt* sind. Außerdem werden Raumfragmente entfernt, die bei der Kartenerstellung mithilfe von Lasersensordaten an solchen Stellen entstehen können, an denen der Sensor durch eine kleine Öffnung im Hindernis nur einen kleinen Bereich eines anderen Raums sehen kann. In Bild B.8 a) sind Raumfragmente im Übergangsbereich zwischen dem erkundeten Gebiet und dem unerkundeten Gebiet in Form von langen, schmalen Räumen zu erkennen.

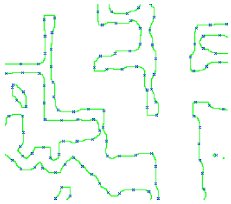
Sowohl Rauschen als auch die Anzahl der Raumfragmente ist nach der Vorverarbeitung mit Filtermaskengröße der Dilatation $d_d = 5$ und Filtermaskengröße der Erosion $d_e = 3$ (Bild B.8 c)) minimiert.

Um ein genaues Ergebnis zu erhalten, ist ein kleiner Hinderniskontrollpunktschritt $\lambda = 20$ zu wählen, wodurch 516 Hinderniskontrollpunkte generiert werden (vgl. Kapitel B.6.1). Wählt man einen größeren Hinderniskontrollpunktschritt λ, so entsteht als Ergebnis eine grobe Annährung an das gewünschte genaue Resultat.

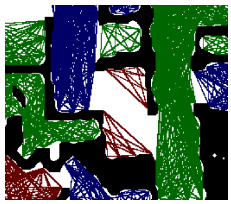
Durch einen Sichtbarkeitsmengenähnlichkeitsschwellwert $\psi = 50 \Rightarrow Grauwert = 127$ und eine Mindestraumgröße $\epsilon = 5$ werden in ganz kleinen Räumen keine Raumnetze aufgespannt. Diese Bereiche werden als Raumfragmente eingestuft und weder einem Hindernis noch einem Raum zugeordnet (Bild B.8 j)).

Mitunter enstehen große Türbereiche (vgl. Beispiel B.5), die erst beim Fluten Räumen zugeordnet werden (Bild B.8 h)). Abhängig von der Orientierung der Karte werden somit in diesen Bereichen Türen an anderen Positionen gebildet (Bild B.8 i)). So hätte der in Bild B.8 j) dem großen *grünen* Raum links in der Mitte zugeordnete schmale Gang entlang des *blauen* Raums dem *blauen* Raum zugeordnet werden können, wenn das erste nicht belegte Rasterfeld im Türbereich näher an dem *blauen* Raum liegen würde. Dadurch würde auch die generierte Tür nicht entlang des *blauen* Raums, sondern etwa rechtwinklig dazu zwischen dem *grünen* und dem *blauen* Raum entstehen.

[3] `http://robots.uni-koblenz.de`
[4] `http://www.uni-koblenz.de/~agas`



(a) Hinderniskontrollpunkte



(b) Raumnetz

(c) Flutung

(d) Türen

(e) Ergebnis

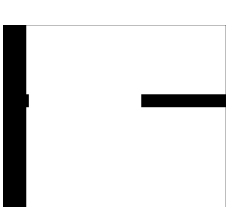
Bild 4.4: unerwartete Tür

In Bild B.8 j) ist auch eine unerwartete Tür zwischen den *roten* Räumen etwa in der Mitte des Bildes zu erkennen, deren Entstehung Bild 4.4 im Detail zeigt. Im betroffenen Bereich werden entlang der Hindernisse Hinderniskontrollpunkte erzeugt (Bild 4.4 a)). Dabei fällt auf, dass die Hinderniskontrollpunkte, die in Bild 4.4 e) innerhalb der beiden *roten* Räume in der Mitte liegen, bei den Tests auf Sichtbarkeit zum Teil erheblich andere Hinderniskontrollpunkte sehen.

Aus den daraus resultierenden sehr unterschiedlichen Sichtbarkeitsmengen werden in diesem Bereich zwei Raumnetze rekonstruiert (Bild 4.4 b)), zwischen denen ein großer Türbereich liegt, der nur einen Hinderniskontrollpunkt enthält. Weiterhin grenzen an diesen großen Türbereich nicht nur die *roten* Raumnetze, sondern auch ein *blaues* sowie ein *grünes* Raumnetz.

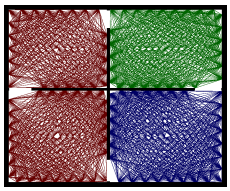
Abhängig von der Orientierung der Karte wird nun der Türbereich einem der angrenzenden vier Raumnetze zugeordnet und gefärbt. In diesem Beispiel wurde der Türbereich dem oberen *roten* Raum zugewiesen (Bild 4.4 c)).

In Folge dessen entstehen die drei Türen in Bild 4.4 d) zwischen dem oberen und dem unteren *roten* Raum, sowie zwischen dem oberen *roten* und dem *grünen* und *blauen* Raum, wodurch sich das Ergebnis mit einer unerwarteten Tür ergibt (Bild 4.4 e)).



(a) Türpfosten

(b) Adjazenzmatrix



(c) Raumnetze

Bild 4.5: Ergänzung von Türpfosten

Durch eine höhere Mindestraumgröße ϵ könnte man der Entstehung einer solchen unerwarteten Tür entgegenwirken, wobei sich diese Änderung der Mindestraumgröße jedoch global auswirken würde

4.1.8 Beispiel mit unerwartetem Resultat

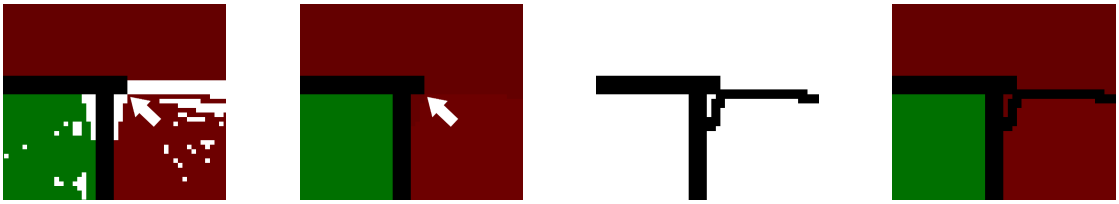
Ein weiteres Beispiel, in dem ein unerwarteter Raum entsteht, ist in B.7 aufgeführt. Hier besteht die Ausgangskarte (Bild B.9 a)) aus einem umschließenden Hindernissegment und einem kreuzförmigen Hindernissegment in der Mitte des Freiraumsegments.

Intuitiv würde man vier kurze Türen an den Enden des kreuzförmigen Hindernissegments erwarten, die die Lücken zum umschließenden Hindernissegment schließen und die Ausgangskarte dadurch in vier gleichförmige Räume einteilen würden.

Der hier entwickelte Algorithmus erzeugt jedoch vier Türen *zwischen* den Enden des kreuzförmigen Hindernissegments ohne die Lücken zum umschließenden Hindernissegment zu schließen (Bild B.9 i)). Es entstehen somit vier kleine, dreieckige Räume und ein großer Raum in Form eines Umgangs (Bild B.9 j) sowie der Kartendeskriptor in Kapitel B.7.2).

Der Grund hierfür ist, dass die Hinderniskontrollpunkte entlang des umschließenden Hindernissegments jeweils mehr Hinderniskontrollpunkte *sehen* können, die am umschließenden Hindernissegment liegen, als solche, die am kreuzförmigen Hindernissegment liegen.

Damit sind die zu den entlang des umschließenden Hindernissegments liegenden Hinderniskontrollpunkten gehörenden Sichtbarkeitsmengen ähnlicher zueinander, als zu



(a) Lücke (b) falscher Bereich (c) falsche Tür (d) falscher Raum

Bild 4.6: Lücke in Raumnetz bei Flutung

Sichtbarkeitsmengen zu Hinderniskontrollpunkten entlang des kreuzförmigen Hindernissegments.

Würde man in Bild B.9 a) zusätzlich Türpfosten eintragen, die gegenüber den Enden des kreuzförmigen Hindernissegments am umschließenden Hindernissegment liegen (Bild 4.5 a)), ergäben sich deutlich andere Raumnetze (Bild 4.5 c)), da die Hinderniskontrollpunkte entlang des umschließenden Hindernissegments nicht mehr so viele Hinderniskontrollpunkte *sehen* könnten, die entlang des umschließenden Hindernissegments liegen.

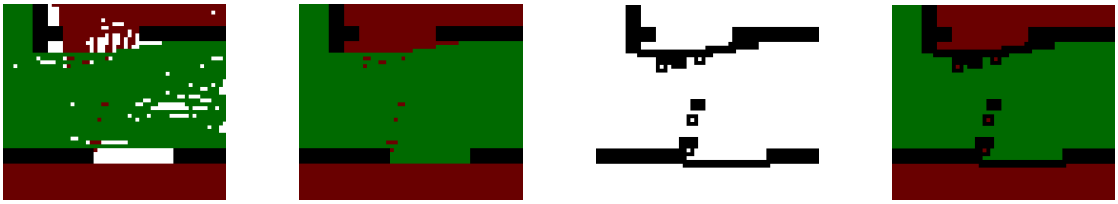
Aus der daraus resultierenden Adjazenzmatrix (Bild 4.5 b)) folgen dann die vier erwarteten Räume aus den Raumnetzen in Bild 4.5 c).

4.1.9 Beispiel mit verrauschtem Ergebnis

In Beispiel B.8 werden zwei weitere Ursachen für die Entstehung von unerwarteten Türen aufgezeigt, die zu Rauschen im Resultat führen. Rauschen meint in diesem Zusammenhang, dass Fragmente eines Raumes fälschlicherweise in einem anderen Raum liegen.

Mit einem Hinderniskontrollpunktschritt $\lambda = 15$ ergeben sich zunächst in Beispiel B.8 1113 Hinderniskontrollpunkte (vgl. Kapitel B.8.1). Mit einem Sichtbarkeitsmengenähnlichkeitsschwellwert $\psi = 45\% \Rightarrow Grauwert = 114$ und der Mindestraumgröße $\epsilon = 5$ (vgl. Kapitel B.8.1) entsteht schon ohne einen Verfeinerungsschritt ein gutes Ergebnis (Bild B.10 j) und Kapitel B.8.2) mit 38 Räumen (vgl. Kapitel B.8.1).

Die eine Ausprägung des Rauschens wird durch die Position eines Hinderniskontrollpunktes bedingt, der nicht glatt an einem Hindernissegment liegt, sondern an der Spitze



(a) Raumüberschneidung (b) Flutung (c) falsche Türen (d) falsche Räume

Bild 4.7: Raumüberschneidung in Raumnetzen

einer Ecke eines Hindernissegments (in Bild 4.6 a) zeigt der weiße Pfeil auf diese Position).

Wegen der Weite des Hinderniskontrollpunktschritts λ entstehen beim Aufspannen der Raumnetze zum einen der Türbereich zwischen den *roten* Raumnetzen und zum anderen ein kleiner Bereich, der sich außerhalb der Raumnetze links neben dem weißen Pfeil in Bild 4.6 a) befindet.

Werden nun die Raumnetze geflutet, wird dieser kleine Bereich abhängig von der Orienierung der Karte entweder dem unteren *roten* Raum oder aber dem oberen *roten* Raum zugeordnet.

In diesem Beispiel B.8 wurde der obere Türbereich in der Farbe des oberen *roten* Raums geflutet. Durch die Lücke an dem Hinderniskontrollpunkt, auf den der weiße Pfeil in Bild 4.6 b) zeigt, wird dabei auch der kleine weiße Bereich links neben dem Pfeil dem oberen *roten* Raum zugeordnet (Bild 4.6 b)) und durch eine Tür (Bild 4.6 c)) von dem unteren *roten* Raum abgetrennt (Bild 4.6 d)).

Dieser kleine Bereich ist zwar vollständig vom oberen Raum abgetrennt, wird diesem aber aufgrund der gleichen Raumfarbe zugeordnet und ist somit ein Raumfragment des oberen *roten* Raums.

Der Grund für das in Bild 4.7 visualisierte Rauschen ist die getroffene Wahl des Parameterwertes des Sichtbarkeitsmengenähnlichkeitsschwellwertes ψ und die Position eines einzigen Hinderniskontrollpunktes. Die dem Hinderniskontrollpunkt im Türbereich zwischen dem unteren *roten* und dem *grünen* Raum in Bild 4.7 a) zugeordnete Sichtbarkeitsmenge ist den von den Hinderniskontrollpunkten des oberen *roten* Raumnetzes repräsentierten Sichrbarkeitsmengen ähnlicher als denen des *grünen* Raumnetzes.

Dadurch wird dieser Hinderniskontrollpunkt dem oberen Raumnetz zugeordnet und es

entsteht eine Überschneidung des *roten* und des *grünen* Raumnetzes. Abhängig von der Reihenfolge der Generierung der Raumnetze liegt entweder das *grüne* über dem *roten* Raumnetz, wie es in Beispiel B.8 der Fall ist, oder umgekehrt.

Da die Raumnetze nicht alle Rasterfelder der darunter liegenden Karte *abdecken*, kann es vorkommen, dass Teile des überdeckten Raumnetzes sichtbar sind (Bild 4.7 a)). Diese Rasterfelder werden bei der Flutung nicht mit einer anderen Raumfarbe überlagert, da nur nicht belegten Rasterfelder eine Raumfarbe zugeordnet wird.

Dadurch tritt nach der Flutung Rauschen in Form von Raumfragmenten auf, die innerhalb eines anderen Raums liegen (Bild 4.7 b)). Da an allen Übergangsgrenzen zwischen unterschiedlichen Raumfarben Türen erzeugt werden (Bild 4.7 c)), werden die Raumfragmente umschließende Türen generiert, woraus das verrauschte Ergebnis (Bild 4.7 d)) resultiert.

4.2 Bewertung des Algorithmus

In Kapitel 4.2.1 werden Laufzeitdiagramme zu den einzelnen Schritten des entwickelten Algorithmus präsentiert.

Die abgebildeten Ergebnisse wurden unter dem UNIX-Betriebssystem *Linux* mit der Kernelversion $2.6.20$ mit der Linuxdistribution *ubuntu Feisty Fawn*[5] und einem *AMD Athlon 3200+ 64 Processor*[6] mit 2.20 GHz und 512 MB RAM erzielt.

Die Zeitangabe in den Diagrammen ist einheitlich in *Millisekunden* angegeben.

Kapitel 4.2.2 bietet eine abschließende qualitative Betrachtung des entwickelten Algorithmus in Bezug auf das erwartete Ergebnis und die Performanz.

4.2.1 Performanz

Die Laufzeit des Schrittes *Belegung ermitteln* (Kapitel 3.2.1) ist nahezu linear und nur von der Größe (Anzahl der Rasterfelder) der Ausgangskarte abhängig. In Bild 4.8 ist diese Verhältnis anhand der linear steigenden Bildbreite veranschaulicht, die ein lineares

[5] `http://www.ubuntu.com`
[6] `http://www.amd.com`

Wachstum der Zeit bedingt.

Die Laufzeiten der *Vorverarbeitung* (Kapitel 3.2.2) sind abhängig von den Filtermaskengrößen der Dilatation d_d und der Erosion d_e. Der Aufwand der Erosion steigt etwa linear, wobei das bei der Dilatation nicht der Fall ist (Bild 4.9).

Die Begründung für den Unterschied ist, dass bei dem Erosionstest für jedes Rasterfeld in der Karte nur die direkte $3x3$-Nachbarschaft überprüft wird. Die Karte wird hier d_e-mal durchlaufen.

Bei der Dilatation hingegen wird die Karte nur einmal durchlaufen und dabei der Dilatationstest mit einer $(2 * d_d + 1)x(2 * d_d + 1)$-Filtermaske durchgeführt, wodurch der Aufwand bei der Verwendung großer Filtermasken erheblich steigt.

Die Performanz der übrigen hier diskutierten Schritte des entwickelten Algorithmus sind wesentlich nur vom Hinderniskontrollpunktschritt λ und der daraus resultierenden Anzahl der Hinderniskontrollpunkte $h \in H$ abhängig.

Der geringe Aufwand des Schrittes *Hinderniskontrollpunkte erzeugen* (Kapitel 3.2.3) ist in Bild 4.10 in Abhängigkeit des Hinderniskontrollpunktschrittes aufgezeigt. Dabei fällt auf, dass auch ein kleiner Hinderniskontrollpunktschritt und die dadurch resultierende Anzahl an Hinderniskontrollpunkten den Aufwand nicht signifikant beeinflussen.

Das sieht für die Schritte *Sichtbarkeitstest* (Kapitel 3.2.4) und *Adjazenzmatrix* (Kapitel 3.2.5) ganz anders aus (Bild 4.11). Hier ist deutlich zu erkennen, dass der Hinderniskontrollpunktschritt λ und die dadurch bedingte Anzahl der Hinderniskontrollpunkte $h \in H$ einen großen Einfluß auf die Performanz haben.

Das bedeutet, je kleiner der Hinderniskontrollpunktschritt ist, desto höher wird die Anzahl der Hinderniskontrollpunkte und desto größer wird der Aufwand. Signifikant sinkt die Performanz ab einem Hinderniskontrollpunktschritt $\lambda \leq 20$. Die Struktur des in der Ausgangskarte abgebildeten Grundrisses, also die Anzahl der Räume und Türen, spielt dabei keine wesentliche Rolle.

Auch der Aufwand für den Schritt *Distanz-1-Sichtbarkeitsmengenähnlichkeit* (Kapitel 3.2.6) ist nur vom Hinderniskontrollpunktschritt λ abhängig (Bild 4.12). Je größer die Ähnlichkeitsmatrix wird, desto größer wird der Aufwand sie zu Füllen. Die Anzahl der rekonstruierten Räume und Türen in der verwendeten Karte hat nur einen minimalen Einfluß auf die Performanz, wie Bild 4.13 verdeutlicht. Hier sind die Aufwände für

vier Karten abgebildet, die sich nur durch die Anzahl der Türen in einem Hindernissegment unterscheiden. Durch die dabei leicht sinkende Anzahl der Hinderniskontrollpunkte sinkt auch der Aufwand der Berechnung der Ähnlichkeitsmatrix.

Im Schritt *Raumrekonstruktion und Raumnetze* (Kapitel 3.2.7) kommt zu der Abhängigkeit der Performanz vom Hinderniskontrollpunktschritt λ zusätzlich eine Abhängigkeit von der Größe der zu rekonstruierenden Räume und damit der Größe der Raumnetze hinzu (Bild 4.14). Grundsätzlich steigt auch hier der Aufwand mit der Anzahl der Hinderniskontrollpunkte $h \in H$, wie es schon bei der Berechnung der Ähnlichkeitsmatrix der Fall war. Den Unterschied macht jedoch die Struktur der Ausgangskarte durch die Größe der Raumnetze der zu rekonstruierenden Räume aus.

Je kleiner die zu rekonstruierenden Räume sind, desto kleiner ist die Anzahl der Strecken pro Raumnetz und kürzer die Strecken. In Bild (Bild 4.14) ist zu erkennen, dass die Bearbeitung einer Karte mit sehr vielen kleinen Räumen performanter ist, als die Bearbeitung der Karten mit wenigen großen Räumen.

Im Gegensatz dazu sinkt im Schritt *Raumnetze fluten* (Kapitel 3.2.8) der Aufwand für das Fluten der Raumnetze mit dem Hinderniskontrollpunktschritt λ (Bild 4.15). Die Ursache hierfür ist unmittelbar ersichtlich. Die *Dichte* der Raumnetze bei groß gewähltem Hinderniskontrollpunktschritt ist gering, da die Netze aus wenigen Strecken bestehen und sehr viele Lücken aufweisen, die geflutet werden müssen. Je kleiner der Hinderniskontrollpunktschritt gewählt wird, desto *dichter* werden die Raumnetze und desto kleiner werden die übrigbleibenden zu flutenden Lücken.

Der Schritt *Türen ermitteln* (Kapitel 3.2.9) schließlich ist unabhängig vom Hindernis-kontrollpunktschritt λ und nur in geringem Maße abhängig von der Anzahl der zu erzeugenden Türen und kann deshalb bei der Betrachtung der Performanz vernachlässigt werden.

Auch die Schritte *Resultat* (Kapitel 3.2.10), *Verfeinerung* (Kapitel 3.2.11) und *Raumdeskriptoren und Kartendeskriptor* (Kapitel 3.2.12) können an dieser Stelle vernachlässigt werden, da keine teuren Operationen oder Berechnungen ausgeführt werden.

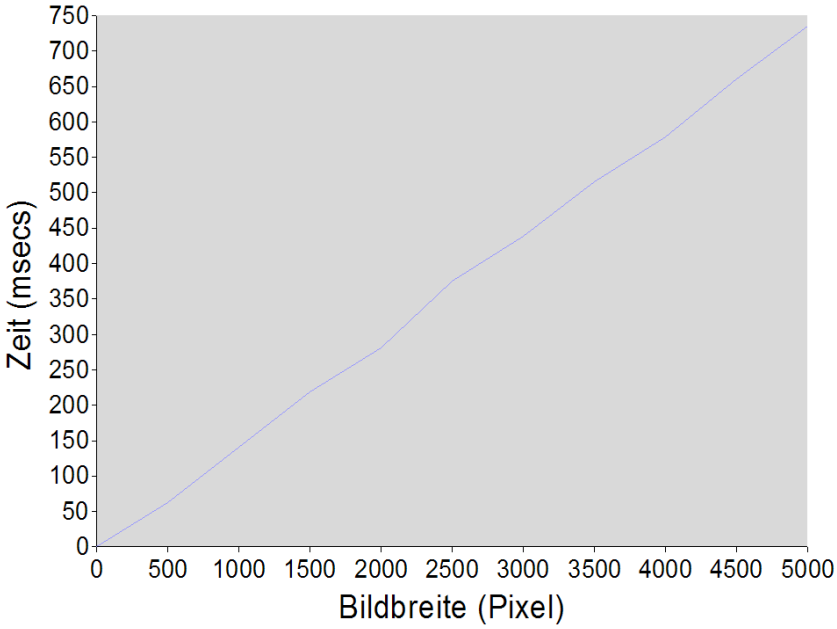


Bild 4.8: Laufzeit Binarisierung

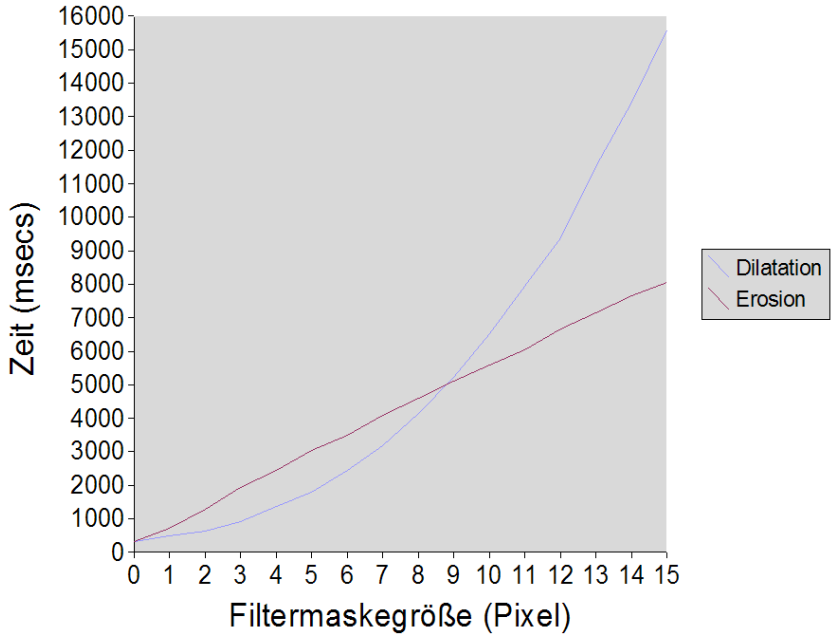


Bild 4.9: Laufzeit Vorverarbeitung

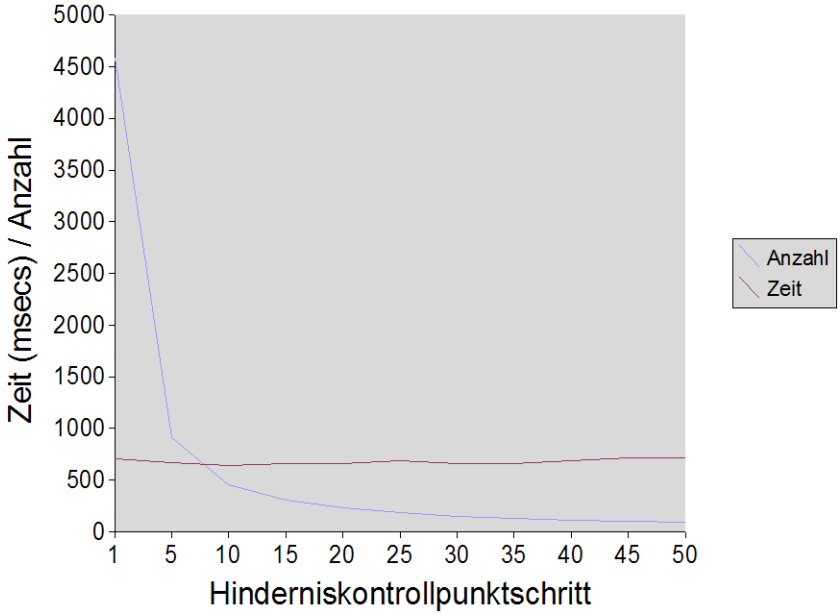


Bild 4.10: Laufzeit Hinderniskontrollpunkte erzeugen

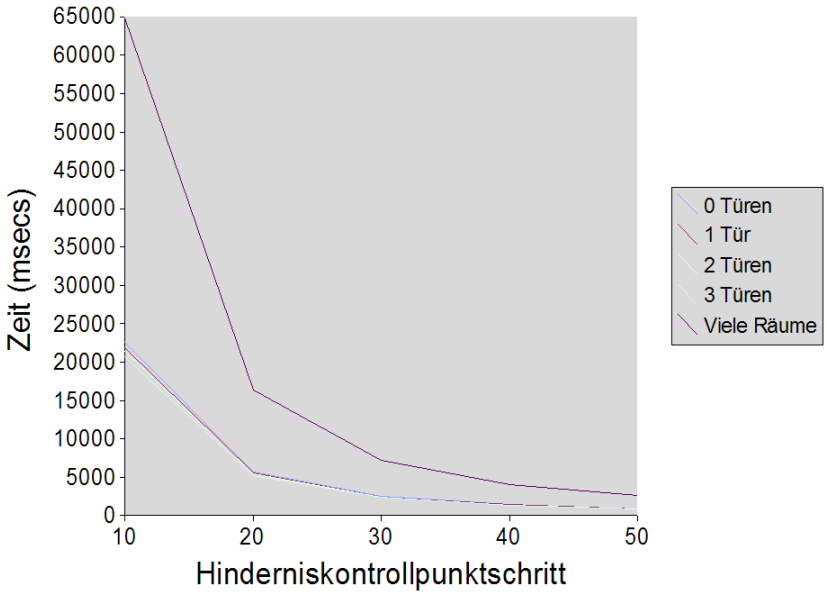


Bild 4.11: Laufzeit Tests auf Sichtbarkeit und Adjazenz

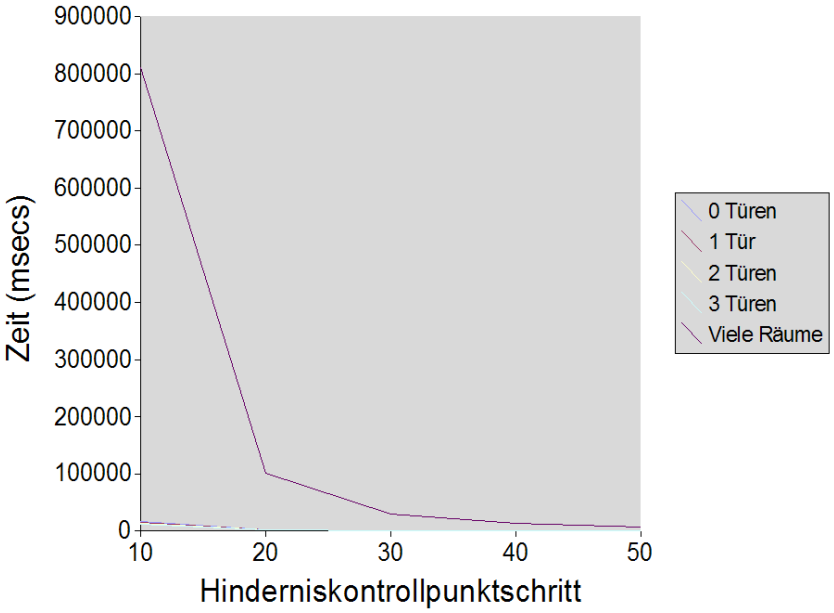


Bild 4.12: Laufzeit Ähnlichkeitsmatrix

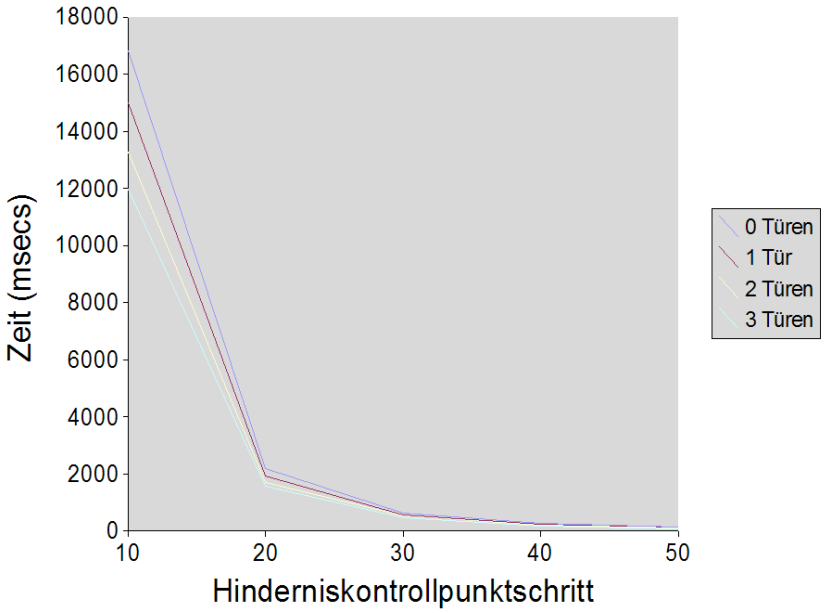


Bild 4.13: Laufzeit Ähnlichkeitsmatrix Detail

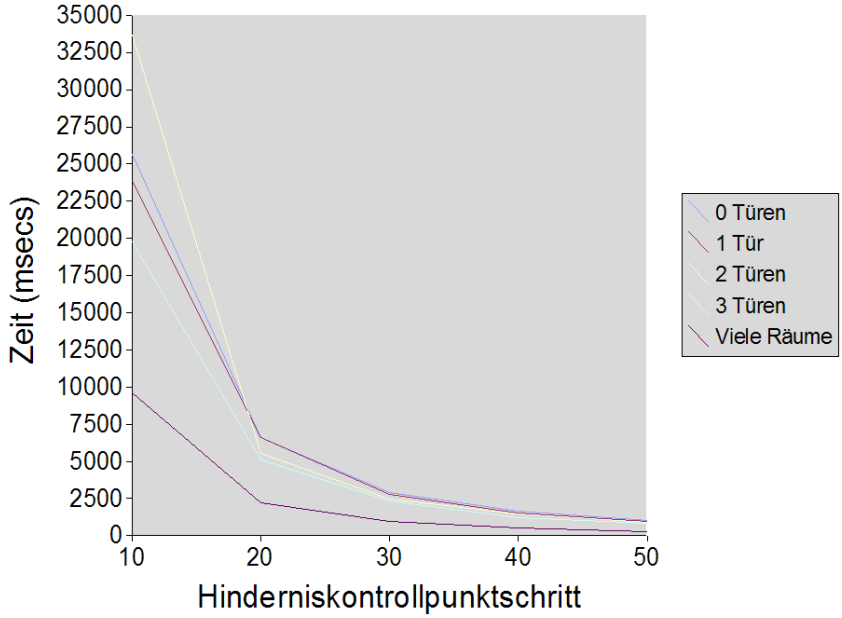


Bild 4.14: Laufzeit Raumrekonstruktion und Raumnetze

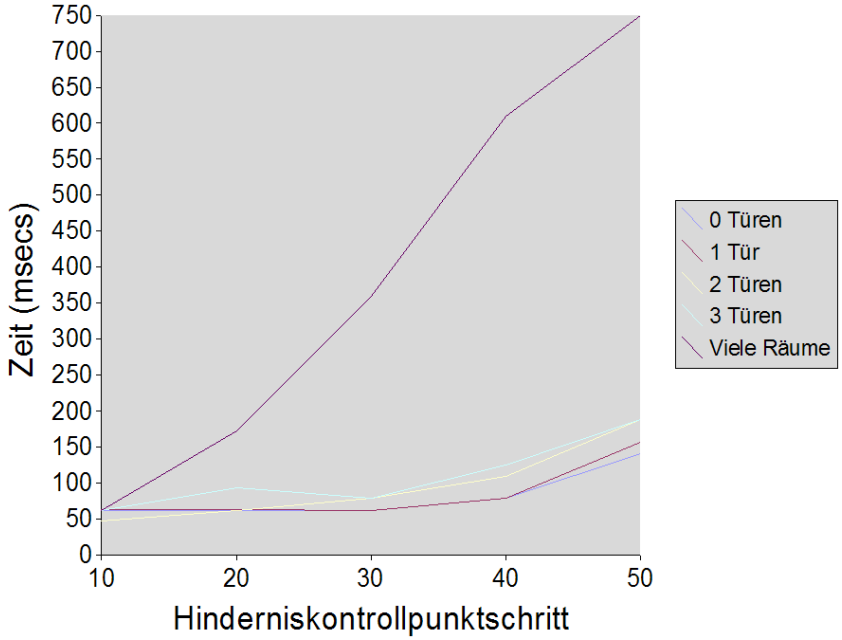


Bild 4.15: Laufzeit Raumnetze fluten

4.2.2 Qualität

Für die meisten Ausgangskarten liefert der entwickelte Algorithmus in den meisten Fällen das intuitiv erwartete, eindeutige Resultat, wobei u.U. ein Verfeinerungsschritt notwendig sein kann.

Dabei können die Ausgangskarten sowohl künstlich erzeugt, als auch auf Sensordaten basieren.

In der Ausgangskarte auftretendes Rauschen kann so minimiert werden, dass stets ein dem Ergebnis der gleichen rauschfreien Karte qualitativ sehr ähnliches Resultat erzielt wird.

Die im Kapitel 4.1 diskutierten auftretenden Seiteneffekte erzeugen in manchen Ergebnissen ein geringes Rauschen, dass das Ergebnis jedoch in keinem besonderen Maße verschlechtert.

Auch die Performanz ist in Bezug auf die Genauigkeit des Resultats gut (vgl. Kapitel 4.2.1) und hauptsächlich abhängig von den Parametern Hinderniskontrollpunktschritt λ und den Filtermaskengrößen für Dilatation d_d und Erosion d_e.

Eine gute Annährung an das erwartete Ergebnis kann in den meisten Fällen mit einem großen Hinderniskontrollpunktschritt λ bei sehr guter Performanz erreicht werden.

In erheblichem Maße ist jedoch das Ergebnis qualitativ abhängig vom Parameter Sichtbarkeitsmengenähnlichkeitsschwellwert ψ, der zwar für die meisten Ausgangskarten um $50\% \Rightarrow Grauwert = 127$ liegt, aber für ein optimales Resultat stets manuell an die Struktur der jeweiligen Ausgangskarte angepasst werden muss.

5 Zusammenfassung und Ausblick

5.1 Zusammenfassung

Der in dieser Arbeit vorgestellte Algorithmus zur Detektion von Räumen in Gebäudegrundrissen liefert bei akzeptabler Laufzeit im Allgemeinen ein stabiles intuitiv erwartetes Resultat.

Die ermittelte Einteilung eines Gebäudegrundrisses in Räume kann dazu verwendet werden, eine Identifikation räumlich zuzuordnen und erfüllt damit die in Kapitel 1 an den Algorithmus gestellten Anforderungen.

Durch die im Kartendeskriptor aufgelisteten Eigenschaften der Räume ist in Verbindung mit der Ausgangskarte eine Schnittstelle zur Verknüpfung mit einem Algorithmus zur Identifikation gegeben.

Die aufwändigsten Teilalgorithmen des entwickelten Algorithmus sind die von der Anzahl der Hinderniskontrollpunkte $h \in H$ abhängigen Schritte *Sichtbarkeitstest* (Kapitel 3.2.4), *Adjazenzmatrix* (Kapitel 3.2.5) und *Distanz-1-Sichtbarkeitsmengenähnlichkeit* (Kapitel 3.2.6), deren Aufwand durch eine Einteilung der Ausgangskarte in lokale Bereiche teilweise erheblich minimiert werden könnte.

Das Ergebnis ist nur in sehr geringem Maße abhängig von der Art und der Qualität des verwendeten zweidimensionalen Kartenmaterials.

Aufgrund der Position der generierten Hinderniskontrollpunkte $h \in H$ kann es in manchen Fällen zum Auftreten von Seiteneffekten (siehe Kapitel 4.1) kommen, die die Güte des Resultats aber nur wenig mindern.

Die Qualität des Ergebnisses ist jedoch in hohem Grade abhängig vom Wert des Sichtbarkeitsmengenähnlichkeitsschwellwertes ψ, der stets manuell an die Struktur der jeweiligen Karte angepasst werden muss.

Der Algorithmus kann im Allgemeinen neben dem präzisen Ergebnis auch durch eine Erhöhung des Hinderniskontrollpunktschrittes eine gute Annäherung des gewünschten Resultats bei erheblich geringerem Aufwand und somit eine sehr schnelle Schätzung liefern.

In der Implementation ist der Gesamtalgorithmus in kleinere Teilalgorithmen so zerlegt, dass diese ohne großen Aufwand an anderer Stelle wiederverwertet werden können. Weiterhin können die entwickelten Quellcodeartefakte unmittelbar im Robotersystem Robbie[1] der Arbeitsgruppe Aktives Sehen (AGAS)[2] Verwendung finden.

Zur Erweiterung und Verbesserung des hier vorgestellten Algorithmus (Kapitel 5.2) bieten sich eine Reihe von Möglichkeiten an: eine vollständige Automatisierung des Algorithmus, eine Variation der Erzeugung der Hinderniskontrollpunkte, die lokale Anpassung der Parameter, die Unterscheidung von Raumeinrichtungsgegenständen von Hindernissen und die Erweiterung von zweidimensionalen auf dreidimensionale Ausgangskarten.

5.2 Ausblick

Um den Algorithmus vollständig von manuellen Einstellungen zu befreien, ist es notwendig, sowohl den Hinderniskontrollpunktschritt λ als auch den Sichtbarkeitsmengenähnlichkeitsschwellwert ψ automatisiert anhand der Struktur der vorliegenden Ausgangskarte zu schätzen und u.U. nach einer ersten Einteilung in Räume anzupassen.

Interessant ist auch in Bezug auf Eindeutigkeit und Performanz eine an die Struktur der Ausgangskarte angepasste Verteilung der Hinderniskontrollpunkte $h \in H$. Eine Ballung von Hinderniskontrollpunkten in *Ecken* oder an Enden von Hindernissegmenten könnte zur Minderung des Auftretens von Seiteneffekten (siehe Kapitel 4.1) führen.

In Bezug auf die globale Wirkung der Parameter Sichtbarkeitsmengenähnlichkeitsschwellwert ψ und der Mindestraumgröße ϵ könnte es für eine präzisere Einteilung der Ausgangskarte in Räume bei einer gleichzeitigen Verringerung der Laufzeiten hilfreich sein, die Wirkung dieser Parameter auf lokale Bereiche in der Ausgangskarte zu beschränken und verschiedene Parameterwerte für diese lokalen Bereiche zu bestimmen.

[1] `http://robots.uni-koblenz.de`
[2] `http://www.uni-koblenz.de/~agas`

Momentan werden Raumeinrichtungsgegenstände in der Ausgangskarte wie Sofa oder Tisch als Hindernis gewertet und nehmen somit teilweise großen Einfluß auf das Resultat. Würde in der Ausgangskarte zwischen Hindernis und Raumeinrichtungsgegenstand unterschieden werden, so könnten die Raumeinrichtungsgegenstände bei der Detektion von Räumen ignoriert werden und dadurch keinen Einfluß mehr auf das Ergebnis nehmen.

Grundsätzlich ist der für zweidimensionale Ausgangskarten entwickelte Algorithmus auch auf dreidimensionale Ausgangskarten übertragbar und muss nur unwesentlich angepasst werden. In Hinsicht auf die Performanz ist die Verwendung mit dreidimensionalen Ausgangskarten jedoch fraglich und müsste genauer untersucht werden.

A Diagramme

Die “Unified Modelling Language“ (UML) [FS97] stellt eine gute Möglichkeit zur Visualisierung von Klassen dar.

Zur guten Darstellung von Klassendiagrammen und Aktivitätsdiagrammen eignet sich die “Unified Modelling Language“ (UML) [FS97]. Aufgelistet sind an dieser Stelle *UML Modellierungen der Klassen der Implementierung* A.1-A.2 und eine *UML Modellierung der Aktivitäten des Gesamtalgorithmus* A.3.

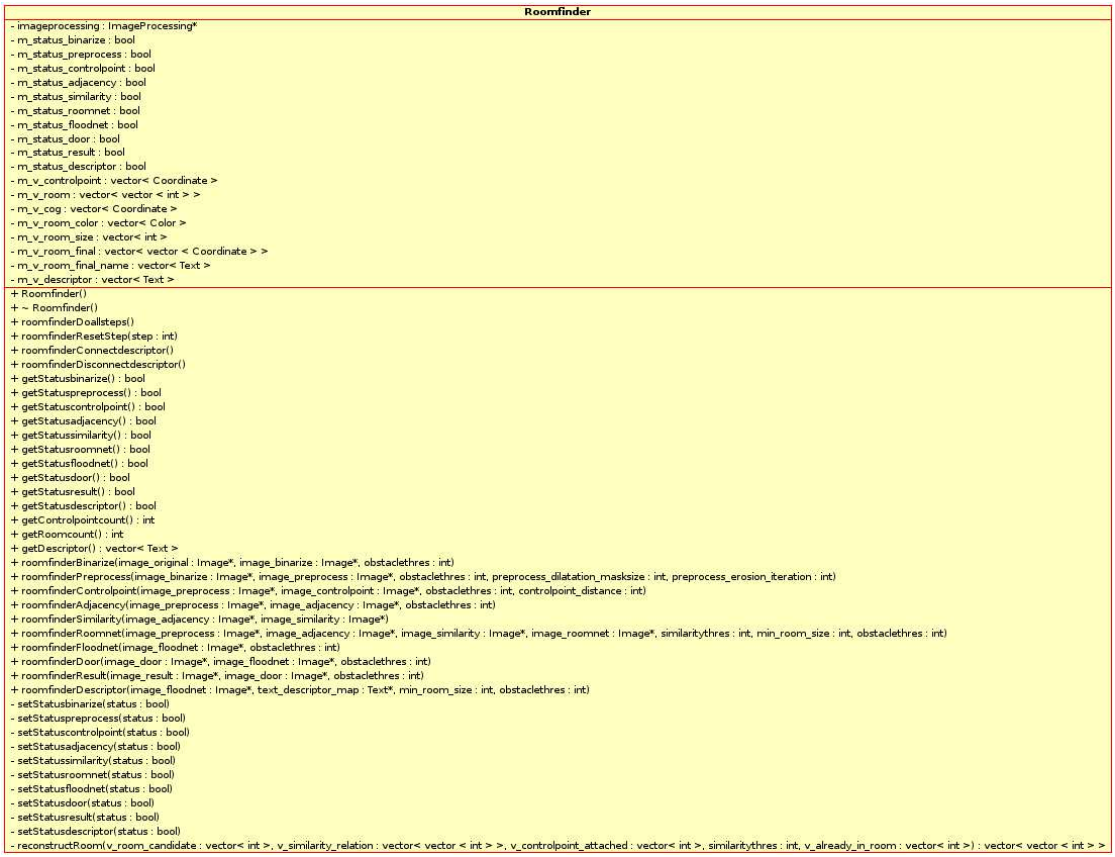


Bild A.1: Klasse Roomfinder

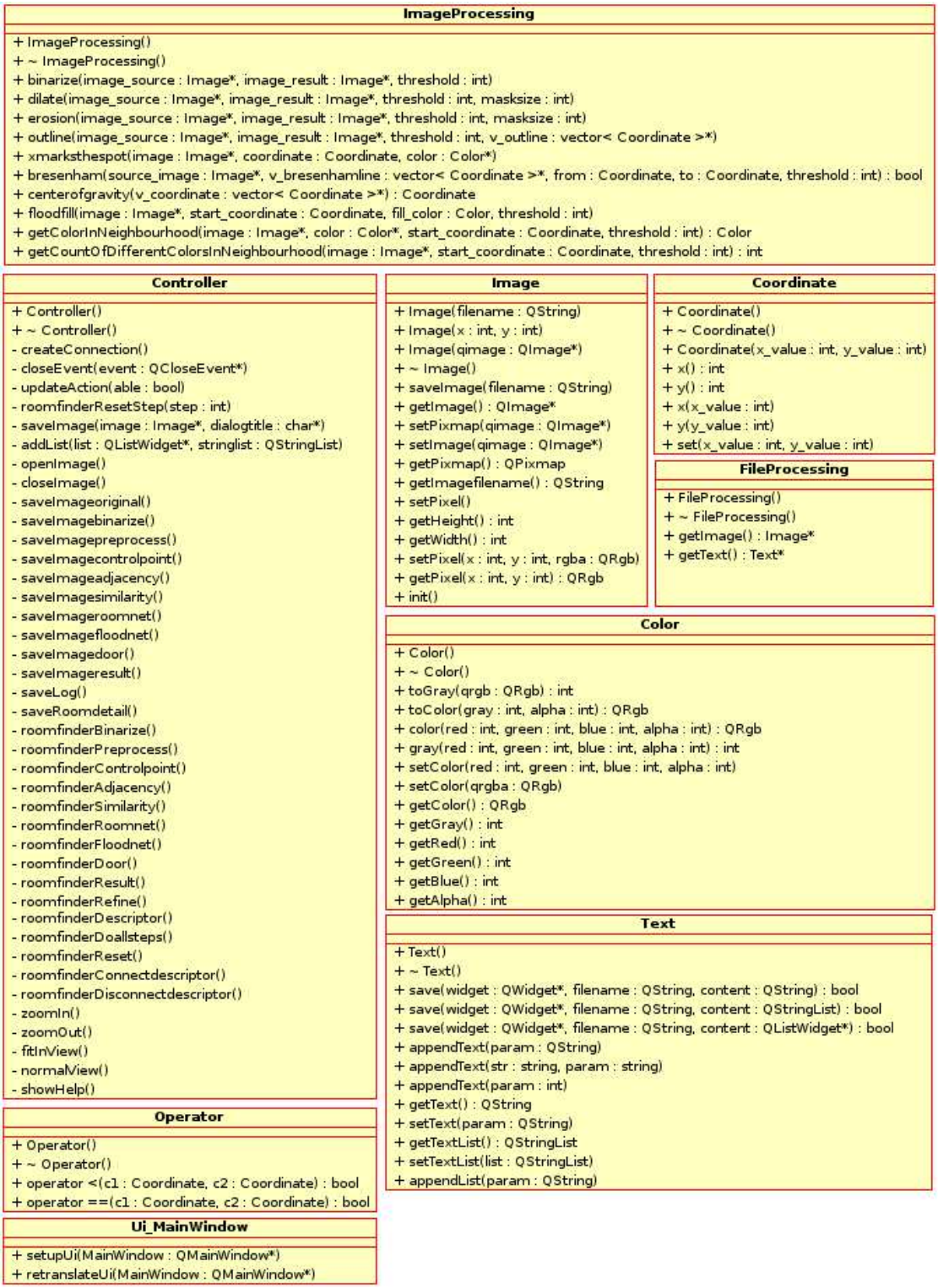


Bild A.2: Hilfsklassen

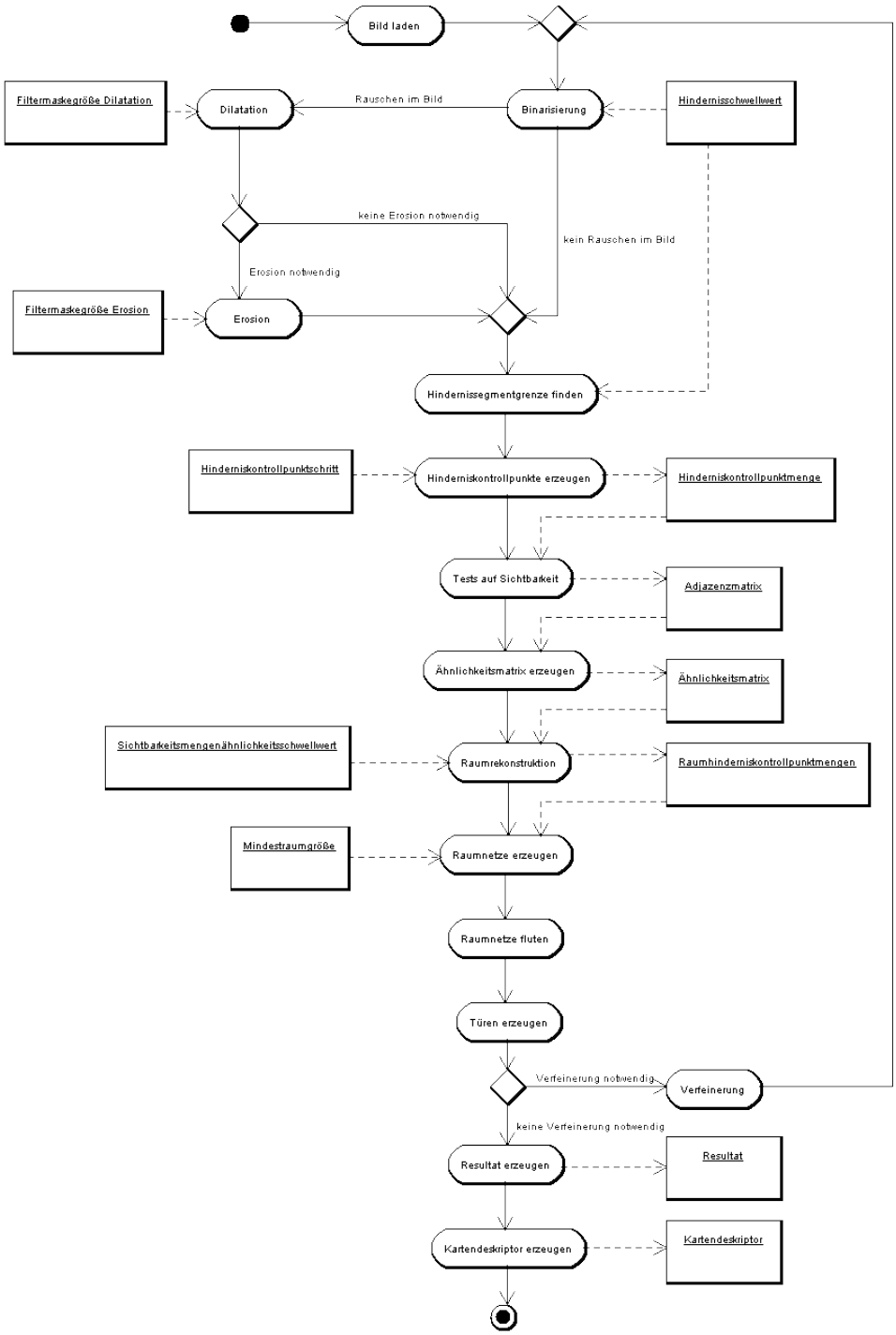


Bild A.3: UML Modellierung der Aktivitäten des Gesamtalgorithmus

B Beispiele

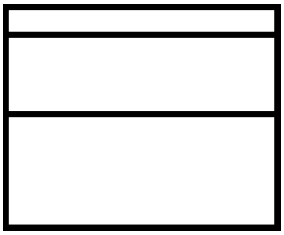
B.1 Beispiel ohne Tür

B.1.1 Log

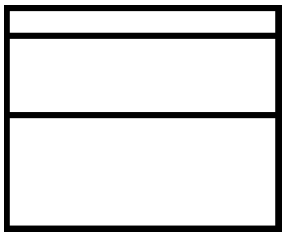
```
Open Image
 Filename: 2001_three_closed_rooms.gif
 Image Width: 863
 Image Height: 704
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 50
 Count of Control Points: 123
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 50
 Minimum Room Size (as count of control points): 5
 Count of Rooms: 3
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
```

B.1.2 Kartendeskriptor, Zwischenergebnisse und Resultat

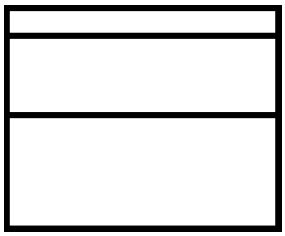
```
Room Details
 of map (image): 2001_three_closed_rooms.gif

Room Name: 1
Room Size (pixel): 55208
Center of Gravity (x,y): (435,52)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (67,19) (117,19) (167,19) (217,19) (267,19) (317,19)
(367,19) (417,19) (467,19) (517,19) (567,19) (617,19) (667,19) (717,19) (767,19)
(817,19) (841,45) (831,85) (781,85) (731,85) (681,85) (631,85) (581,85) (531,85)
(481,85) (431,85) (381,85) (331,85) (281,85) (231,85) (181,85) (131,85) (81,85)
(31,85) (18,48)

Room Name: 2
Room Size (pixel): 187048
Center of Gravity (x,y): (429,217)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (39,105) (89,105) (139,105) (189,105) (239,105)
(289,105) (339,105) (389,105) (439,105) (489,105) (539,105) (589,105) (639,105)
(689,105) (739,105) (789,105) (839,105) (841,153) (841,203) (841,253) (841,303)
(819,331) (769,331) (719,331) (669,331) (619,331) (569,331) (519,331) (469,331)
(419,331) (369,331) (319,331) (269,331) (219,331) (169,331) (119,331) (69,331)
(19,331) (18,282) (18,232) (18,182) (18,132)

Room Name: 3
Room Size (pixel): 274392
Center of Gravity (x,y): (431,517)
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (41,351) (91,351) (141,351) (191,351) (241,351)
(291,351) (341,351) (391,351) (441,351) (491,351) (541,351) (591,351) (641,351)
(691,351) (741,351) (791,351) (841,351) (841,401) (841,451) (841,501) (841,551)
(841,601) (841,651) (823,683) (773,683) (723,683) (673,683) (623,683) (573,683)
(523,683) (473,683) (423,683) (373,683) (323,683) (273,683) (223,683) (173,683)
(123,683) (73,683) (23,683) (18,638) (18,588) (18,538) (18,488) (18,438) (18,388)
```



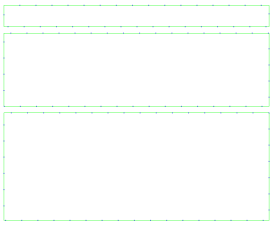
(a) Original



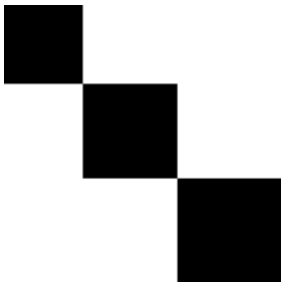
(b) Binarisiert



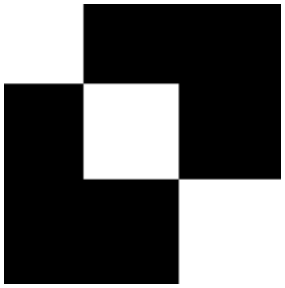
(c) Dilatation und Erosion



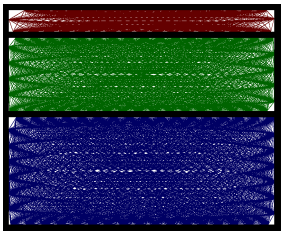
(d) Hinderniskontrollpunkte



(e) Adjazenzmatrix

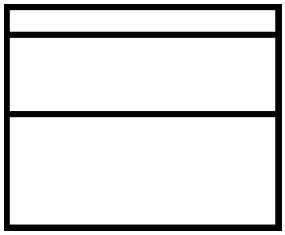


(f) Ähnlichkeitsmatrix



(g) Raumnetz

(h) Flutung



(i) Türen

(j) Resultat

Bild B.1: Beispiel ohne Tür

B.2 Beispiel mit einer Tür

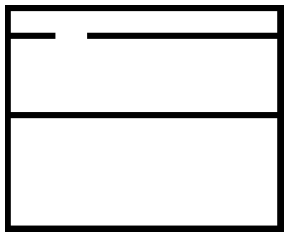
B.2.1 Log

```
Open Image
 Filename: 2002_three_one_opened.gif
 Image Width: 863
 Image Height: 704
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 50
 Count of Control Points: 120
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 50
 Minimum Room Size (as count of control points): 5
 Count of Rooms: 3
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
```

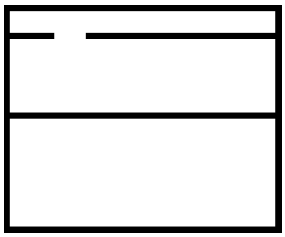
B.2.2 Kartendeskriptor, Zwischenergebnisse und Resultat

```
Room Details
 of map (image): 2002_three_one_opened.gif

Room Name: 1
Room Size (pixel): 56994
Center of Gravity (x,y): (449,50)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (67,19) (117,19) (167,19) (217,19) (267,19) (317,19)
(367,19) (417,19) (467,19) (517,19) (567,19) (617,19) (667,19) (717,19) (767,19)
(817,19) (841,45) (831,85) (781,85) (731,85) (681,85) (631,85) (581,85) (531,85)
(481,85) (431,85) (381,85) (331,85) (281,85) (125,85) (75,85) (25,85) (18,42)
```

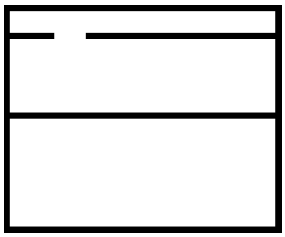
```
Room Name: 2
Room Size (pixel): 187124
Center of Gravity (x,y): (434,220)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (255,105) (305,105) (355,105) (405,105) (455,105)
(505,105) (555,105) (605,105) (655,105) (705,105) (755,105) (805,105) (841,119)
(841,169) (841,219) (841,269) (841,319) (803,331) (753,331) (703,331) (653,331)
(603,331) (553,331) (503,331) (453,331) (403,331) (353,331) (303,331) (253,331)
(203,331) (153,331) (103,331) (53,331) (18,316) (18,266) (18,216) (18,166)
(18,116) (57,105) (107,105) (156,104)

Room Name: 3
Room Size (pixel): 274392
Center of Gravity (x,y): (431,517)
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (45,351) (95,351) (145,351) (195,351) (245,351)
(295,351) (345,351) (395,351) (445,351) (495,351) (545,351) (595,351) (645,351)
(695,351) (745,351) (795,351) (841,355) (841,405) (841,455) (841,505) (841,555)
(841,605) (841,655) (819,683) (769,683) (719,683) (669,683) (619,683) (569,683)
(519,683) (469,683) (419,683) (369,683) (319,683) (269,683) (219,683) (169,683)
(119,683) (69,683) (19,683) (18,634) (18,584) (18,534) (18,484) (18,434) (18,384)
```



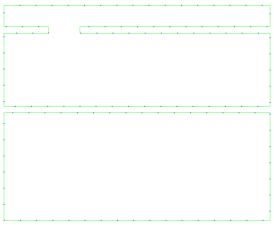
(a) Original



(b) Binarisiert

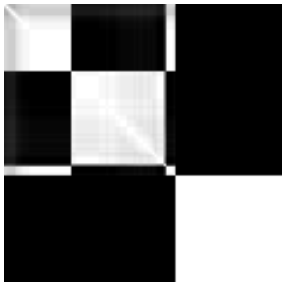


(c) Dilatation und Erosion

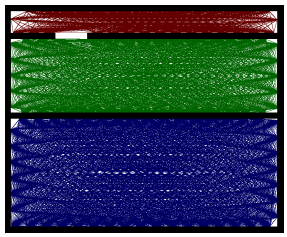


(d) Hinderniskontrollpunkte

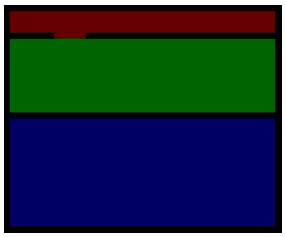
(e) Adjazenzmatrix



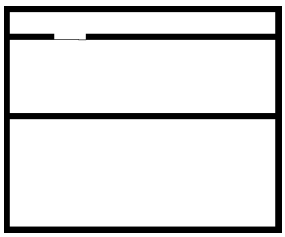
(f) Ähnlichkeitsmatrix



(g) Raumnetz



(h) Flutung



(i) Türen

(j) Resultat

Bild B.2: Beispiel mit einer Tür

B.3 Beispiel mit mehreren Türen und Raumeinrichtungsgegenständen

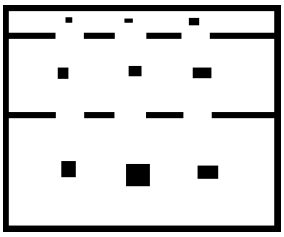
B.3.1 Log

```
Open Image
 Filename: 1009_three_two_opened_six_doors.gif
 Image Width: 863
 Image Height: 704
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 50
 Count of Control Points: 135
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 50
 Minimum Room Size (as count of control points): 4
 Count of Rooms: 3
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
```

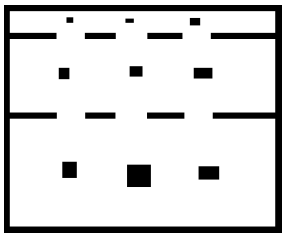
B.3.2 Kartendeskriptor, Zwischenergebnisse und Resultat

```
Room Details
 of map (image): 1009_three_two_opened_six_doors.gif

Room Name: 1
Room Size (pixel): 58983
Center of Gravity (x,y): (440,45)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (67,19) (117,19) (167,19) (217,19) (267,19) (317,19)
(367,19) (417,19) (467,19) (517,19) (567,19) (617,19) (667,19) (717,19) (767,19)
```

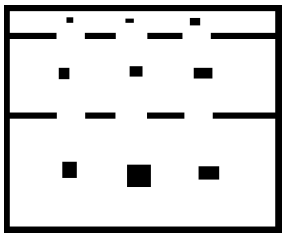
```
(817,19) (841,45) (831,85) (781,85) (731,85) (681,85) (147,85) (97,85) (47,85)
(18,64) (214,37) (291,85) (341,85) (403,41) (477,85) (527,85) (599,39)

Room Name: 2
Room Size (pixel): 187905
Center of Gravity (x,y): (434,216)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (681,105) (731,105) (781,105) (831,105) (841,145)
(841,195) (841,245) (841,295) (827,331) (777,331) (727,331) (677,331) (161,331)
(111,331) (61,331) (18,324) (18,274) (18,224) (18,174) (18,124) (49,105) (99,105)
(149,105) (174,193) (323,105) (273,105) (258,331) (308,331) (394,188) (450,331)
(500,331) (550,331) (551,105) (501,105) (451,105) (597,193) (647,193)

Room Name: 3
Room Size (pixel): 264412
Center of Gravity (x,y): (438,538)
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (647,351) (697,351) (747,351) (797,351) (841,357)
(841,407) (841,457) (841,507) (841,557) (841,607) (841,657) (817,683) (767,683)
(717,683) (667,683) (617,683) (567,683) (517,683) (467,683) (417,683) (367,683)
(317,683) (267,683) (217,683) (167,683) (117,683) (67,683) (18,682) (18,632)
(18,582) (18,532) (18,482) (18,432) (18,382) (37,351) (87,351) (137,351) (208,534)
(304,351) (254,351) (456,512) (456,562) (406,562) (540,351) (490,351) (652,539)
(603,538)
```



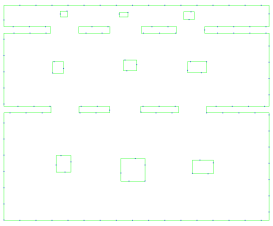
(a) Original



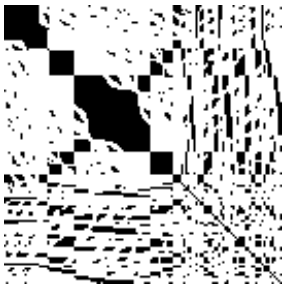
(b) Binarisiert



(c) Dilatation und Erosion

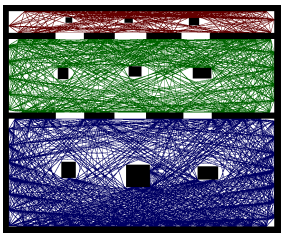


(d) Hinderniskontrollpunkte

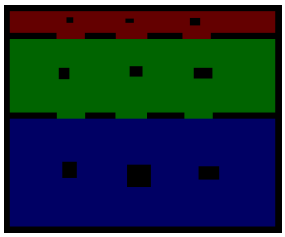


(e) Adjazenzmatrix

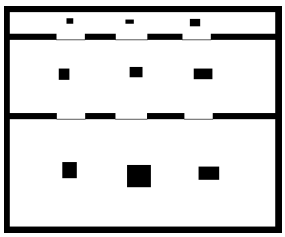
(f) Ähnlichkeitsmatrix



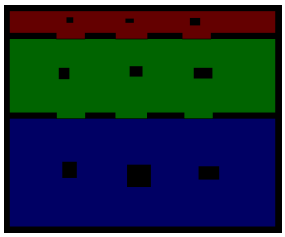
(g) Raumnetz



(h) Flutung



(i) Türen



(j) Resultat

Bild B.3: Beispiel mit mehreren Türen und Raumeinrichtungsgegenständen

B.4 Beispiel mit mehreren Räumen und notwendiger Verfeinerung

B.4.1 Log

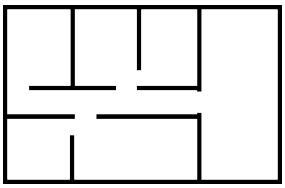
```
Open Image
 Filename: 2023_motivation.gif
 Image Width: 800
 Image Height: 511
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 50
 Count of Control Points: 124
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 50
 Minimum Room Size (as count of control points): 3
 Count of Rooms: 4
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Save File
 0.png
Refine
 based on door image of: 0.png
Reset Binarize
Reset Preprocess
Reset Control Point
Reset Adjacency
Reset Similarity
Reset Roomnet
Reset Floodnet
Reset Door
Reset Result
Binarize
 Obstacle Threshold: 50
```

```
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 40
 Count of Control Points: 165
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 75
 Minimum Room Size (as count of control points): 3
 Count of Rooms: 7
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
```

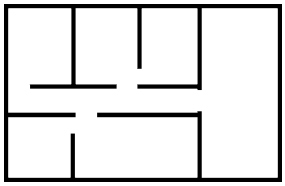
B.4.2 Kartendeskriptor, Zwischenergebnisse und Resultat

```
Room Details
 of map (image):

Room Name: 1
Room Size (pixel): 40727
Center of Gravity (x,y): (103,131)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (50,12) (90,12) (130,12) (170,12) (193,29) (193,69)
(193,109) (193,149) (193,189) (193,229) (154,230) (114,230) (74,230) (49,243)
(12,246) (12,206) (12,166) (12,126) (12,86) (12,46)

Room Name: 2
Room Size (pixel): 38752
Center of Gravity (x,y): (277,276)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (19,249) (56,246) (93,243) (133,243) (173,243) (213,243)
(253,243) (293,243) (332,244) (372,244) (411,243) (451,243) (491,243) (531,243)
(559,307) (525,311) (485,311) (445,311) (405,311) (365,311) (325,311) (285,311)
(189,311) (149,311) (109,311) (69,311) (29,311) (12,288)

Room Name: 3
Room Size (pixel): 106651
Center of Gravity (x,y): (685,253)
```

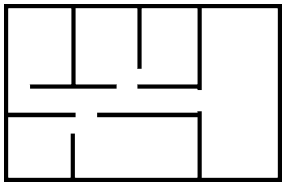
```
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (569,209) (569,169) (569,129) (569,89) (569,49)
(572,12) (612,12) (652,12) (692,12) (732,12) (772,12) (787,37) (787,77)
(787,117) (787,157) (787,197) (787,237) (787,277) (787,317) (787,357) (787,397)
(787,437) (787,477) (768,498) (728,498) (688,498) (648,498) (608,498) (569,497)
(569,457) (569,417) (569,377) (569,337)

Room Name: 4
Room Size (pixel): 32198
Center of Gravity (x,y): (90,412)
Room Color (r,g,b,a): (101,0,0,255)
Room Control Points (x,y): (13,325) (53,325) (93,325) (133,325) (173,325)
(191,401) (191,441) (191,481) (168,498) (128,498) (88,498) (48,498) (12,494)
(12,454) (12,414) (12,374) (12,334)

Room Name: 5
Room Size (pixel): 61040
Center of Gravity (x,y): (388,414)
Room Color (r,g,b,a): (0,101,0,255)
Room Control Points (x,y): (212,326) (252,326) (291,325) (331,325) (371,325)
(411,325) (451,325) (491,325) (531,325) (556,340) (556,380) (556,420) (556,460)
(554,498) (514,498) (474,498) (434,498) (394,498) (354,498) (314,498) (274,498)
(234,498) (204,488) (204,448) (204,408)

Room Name: 6
Room Size (pixel): 39416
Center of Gravity (x,y): (290,115)
Room Color (r,g,b,a): (0,0,101,255)
Room Control Points (x,y): (237,12) (277,12) (317,12) (357,12) (383,26) (383,66)
(383,106) (383,146) (382,185) (376,241) (309,230) (269,230) (229,230) (206,213)
(206,173) (206,133) (206,93) (206,53) (206,13)

Room Name: 7
Room Size (pixel): 36017
Center of Gravity (x,y): (480,117)
Room Color (r,g,b,a): (102,0,0,255)
Room Control Points (x,y): (396,161) (396,121) (396,81) (396,41) (407,12) (447,12)
(487,12) (527,12) (556,23) (556,63) (556,103) (556,143) (556,183) (556,223)
(523,230) (483,230) (443,230) (403,230)
```



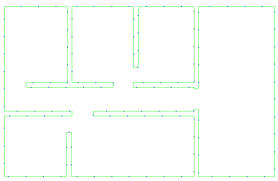
(a) Original



(b) Binarisiert

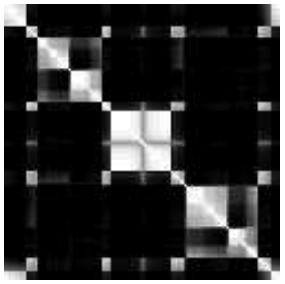


(c) Dilatation und Erosion

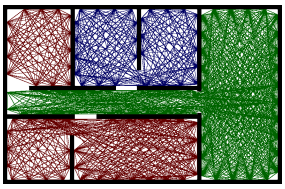


(d) Hinderniskontrollpunkte

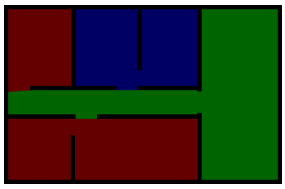
(e) Adjazenzmatrix



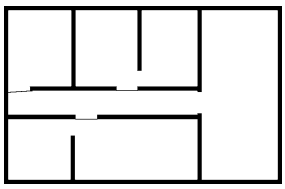
(f) Ähnlichkeitsmatrix



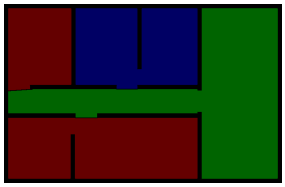
(g) Raumnetz



(h) Flutung

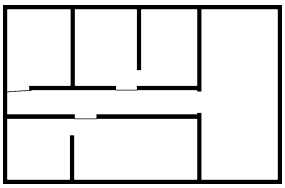


(i) Türen

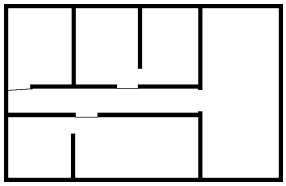


(j) Resultat vor Verfeinerung

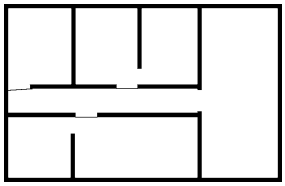
Bild B.4: Beispiel mit mehreren Räumen und notwendiger Verfeinerung vor Verfeinerung



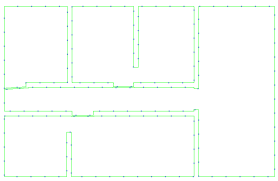
(a) Bisheriges Resultat



(b) Binarisiert

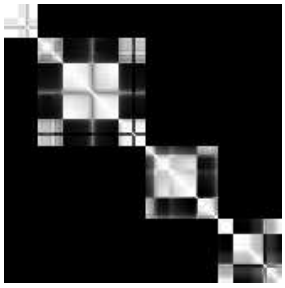


(c) Dilatation und Erosion

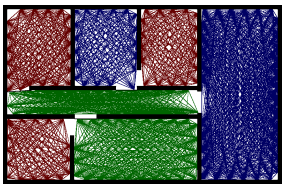


(d) Hinderniskontrollpunkte

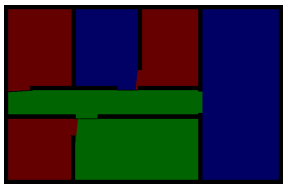
(e) Adjazenzmatrix



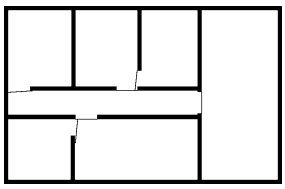
(f) Ähnlichkeitsmatrix



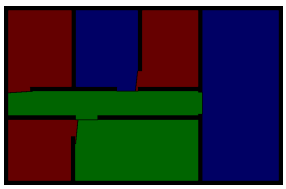
(g) Raumnetz



(h) Flutung



(i) Türen



(j) Resultat der Verfeinerung

Bild B.5: Beispiel mit mehreren Räumen und notwendiger Verfeinerung nach Verfeinerung

B.5 Beispiel mit notwendiger Vorverarbeitung

B.5.1 Log

```
Open Image
 Filename: roomsMulti0dist.png
 Image Width: 878
 Image Height: 726
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 12
 Erosion: 12
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 50
 Count of Control Points: 231
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 50
 Minimum Room Size (as count of control points): 5
 Count of Rooms: 9
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
Save File
 0.png
Refine
 based on door image of: 0.png
Reset Original Image
 Filename: 0.png
Reset Binarize
Reset Preprocess
Reset Control Point
Reset Adjacency
Reset Similarity
Reset Roomnet
Reset Floodnet
Reset Door
Reset Result
Binarize
 Obstacle Threshold: 50
```

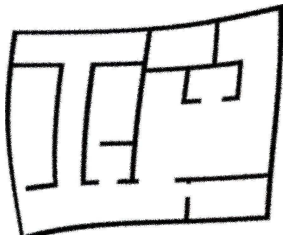
```
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 50
 Count of Control Points: 249
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 54
 Minimum Room Size (as count of control points): 5
 Count of Rooms: 10
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
```

B.5.2 Kartendeskriptor, Zwischenergebnisse und Resultat

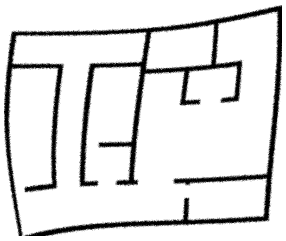
```
Room Details
 of map (image):

Room Name: 1
Room Size (pixel): 51035
Center of Gravity (x,y): (429,57)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (22,78) (118,80) (166,78) (215,77) (262,76) (357,73)
(405,71) (451,67) (541,57) (584,50) (628,44) (712,28) (754,20) (793,9)

Room Name: 2
Room Size (pixel): 50058
Center of Gravity (x,y): (80,393)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (9,304) (12,257) (21,216) (59,204) (108,203) (151,208)
(158,243) (154,289) (150,335) (147,382) (145,430) (145,478) (145,528) (125,558)
(79,562) (58,591) (31,590) (24,547) (18,503) (14,457) (9,412) (8,363)

Room Name: 3
Room Size (pixel): 65395
Center of Gravity (x,y): (222,267)
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (28,131) (49,102) (96,101) (146,101) (193,100) (241,98)
```

```
(289,98) (337,96) (385,94) (428,97) (435,132) (420,167) (381,176) (332,177)
(283,178) (262,207) (255,250) (249,294) (245,340) (241,386) (239,434) (236,481)
(234,529) (230,573) (184,577) (159,560) (156,513) (155,464) (158,417) (161,370)
(165,324) (170,279) (174,233) (178,187) (87,180) (39,179)

Room Name: 4
Room Size (pixel): 58648
Center of Gravity (x,y): (278,635)
Room Color (r,g,b,a): (101,0,0,255)
Room Control Points (x,y): (43,602) (82,591) (125,584) (171,582) (216,577)
(261,574) (310,573) (358,571) (406,571) (454,569) (549,564) (563,622) (554,663)
(513,672) (464,673) (416,675) (368,677) (321,680) (274,683) (228,687) (184,693)
(140,699) (98,707) (59,712) (45,676) (37,634)

Room Name: 5
Room Size (pixel): 25406
Center of Gravity (x,y): (702,611)
Room Color (r,g,b,a): (0,101,0,255)
Room Control Points (x,y): (593,566) (637,560) (687,560) (733,556) (781,554)
(818,565) (819,610) (809,650) (770,661) (723,664) (676,667) (627,668) (583,664)
(575,622)

Room Name: 6
Room Size (pixel): 15320
Center of Gravity (x,y): (318,503)
Room Color (r,g,b,a): (0,0,101,255)
Room Control Points (x,y): (249,493) (251,445) (299,449) (349,449) (389,459)
(391,499) (380,538) (350,558) (312,570) (283,549) (251,531)

Room Name: 7
Room Size (pixel): 33902
Center of Gravity (x,y): (341,305)
Room Color (r,g,b,a): (102,0,0,255)
Room Control Points (x,y): (254,398) (259,353) (264,308) (270,264) (277,221)
(307,201) (354,200) (403,199) (420,228) (415,273) (409,317) (404,362) (398,406)
(368,426) (319,427)

Room Name: 8
Room Size (pixel): 148603
Center of Gravity (x,y): (653,313)
Room Color (r,g,b,a): (0,102,0,255)
Room Control Points (x,y): (440,566) (488,564) (531,557) (564,540) (612,538)
(660,536) (707,533) (755,531) (801,527) (827,503) (832,458) (837,413) (841,367)
(845,321) (849,275) (853,229) (855,181) (858,134) (858,84) (854,38) (816,34)
(776,44) (735,53) (693,61) (667,85) (665,133) (671,173) (710,172) (744,186)
(740,232) (736,278) (717,309) (680,322) (632,320) (584,318) (553,299) (557,253)
(547,219) (503,217) (456,220) (434,248) (430,294) (422,336) (417,381) (412,426)
(408,472) (404,518) (414,556)

Room Name: 9
Room Size (pixel): 22019
```

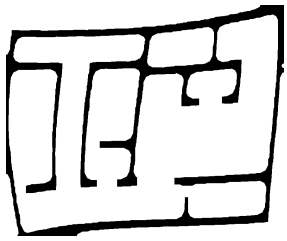
```
Center of Gravity (x,y): (556,132)
Room Color (r,g,b,a): (0,0,102,255)
Room Control Points (x,y): (466,195) (512,193) (559,190) (604,185) (643,174)
(652,133) (656,87) (626,71) (582,77) (538,83) (493,88) (459,104) (450,145)

Room Name: 10
Room Size (pixel): 14599
Center of Gravity (x,y): (650,242)
Room Color (r,g,b,a): (103,0,0,255)
Room Control Points (x,y): (570,267) (580,227) (613,210) (658,205) (704,201)
(728,225) (720,267) (689,286) (590,293)
```



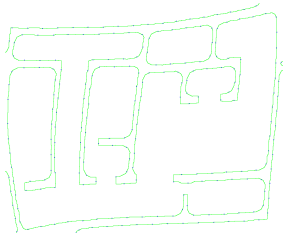
(a) Original



(b) Binarisiert

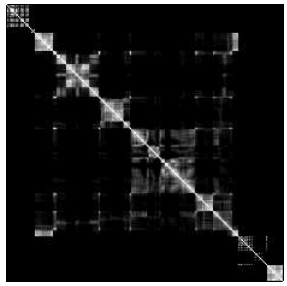


(c) Dilatation und Erosion

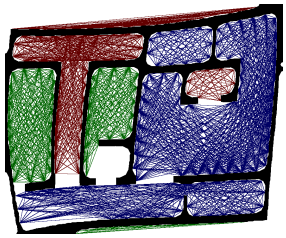


(d) Hinderniskontrollpunkte

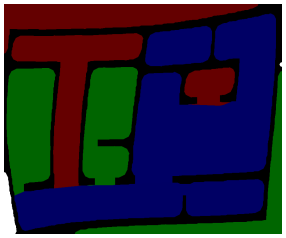
(e) Adjazenzmatrix



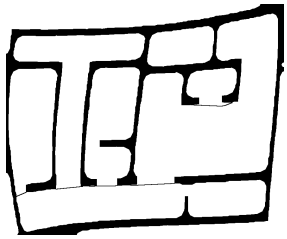
(f) Ähnlichkeitsmatrix



(g) Raumnetz



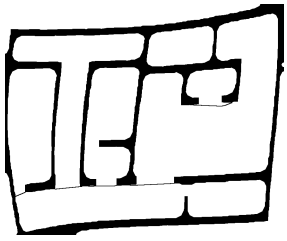
(h) Flutung



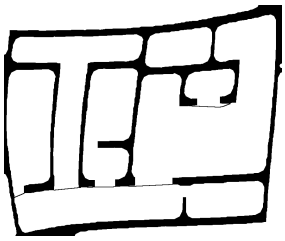
(i) Türen

(j) Resultat vor Verfeinerung

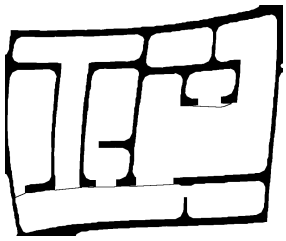
Bild B.6: Beispiel mit notwendiger Vorverarbeitung vor Verfeinerung



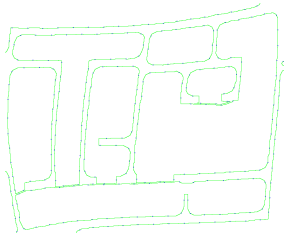
(a) Bisheriges Resultat



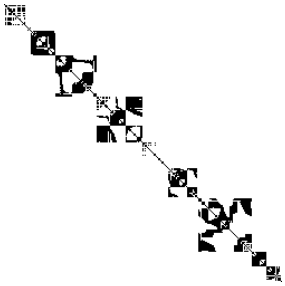
(b) Binarisiert



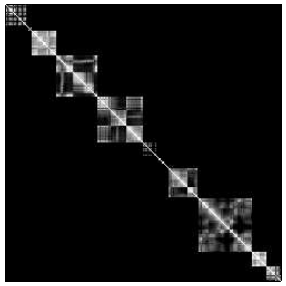
(c) Dilatation und Erosion



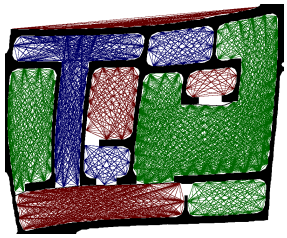
(d) Hinderniskontrollpunkte



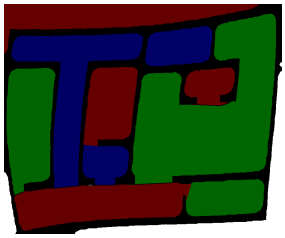
(e) Adjazenzmatrix



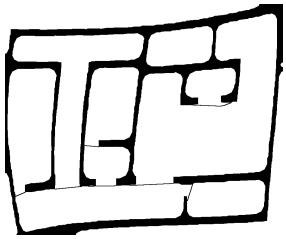
(f) Ähnlichkeitsmatrix



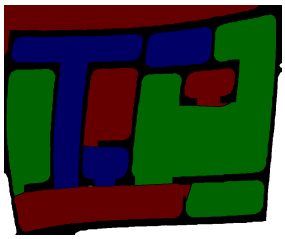
(g) Raumnetz



(h) Flutung



(i) Türen



(j) Resultat der Verfeinerung

Bild B.7: Beispiel mit notwendiger Vorverarbeitung nach Verfeinerung

B.6 Beispiel mit von Robbie erzeugter Karte

B.6.1 Log

```
Open Image
 Filename: 3003_resko_map_german_open_2008.gif
 Image Width: 801
 Image Height: 710
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 5
 Erosion: 3
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 20
 Count of Control Points: 516
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 50
 Minimum Room Size (as count of control points): 5
 Count of Rooms: 28
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
```

B.6.2 Kartendeskriptor, Zwischenergebnisse und Resultat

```
Room Details
 of map (image): 3003_resko_map_german_open_2008.gif

Room Name: 1
Room Size (pixel): 1227
Center of Gravity (x,y): (71,341)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (47,327) (63,331) (79,335) (97,337) (106,347) (89,348)
(71,350) (55,350) (39,346)

Room Name: 2
Room Size (pixel): 18731
```

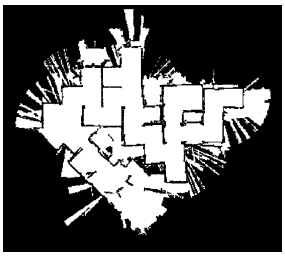
```
Center of Gravity (x,y): (178,339)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (112,366) (121,377) (130,388) (140,398) (150,408)
(160,400) (174,394) (186,402) (195,413) (208,414) (220,406) (241,393) (243,375)
(252,364) (265,357) (252,351) (221,344) (221,324) (224,307) (223,288) (226,238)
(206,238) (187,239) (171,243) (169,273) (166,290) (149,287) (130,286) (115,291)
(113,309) (114,328) (114,348) (104,358)

Room Name: 3
Room Size (pixel): 2414
Center of Gravity (x,y): (223,458)
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (227,419) (217,429) (207,439) (197,449) (203,461)
(213,471) (224,480) (234,484) (247,477) (263,473)

Room Name: 4
Room Size (pixel): 7198
Center of Gravity (x,y): (299,494)
Room Color (r,g,b,a): (101,0,0,255)
Room Control Points (x,y): (264,486) (253,495) (247,505) (258,514) (268,524)
(278,534) (305,533) (334,464) (309,465) (299,455) (288,446) (277,437) (329,521)
(322,530) (349,499) (356,486) (348,506)

Room Name: 5
Room Size (pixel): 11545
Center of Gravity (x,y): (391,587)
Room Color (r,g,b,a): (0,101,0,255)
Room Control Points (x,y): (317,575) (327,585) (338,594) (339,611) (352,610)
(364,618) (366,636) (376,638) (392,640) (401,651) (407,637) (419,629) (430,620)
(436,606) (450,600) (461,591) (453,581) (452,562) (457,547) (441,543) (425,539)
(410,536) (343,525) (335,537) (336,556) (371,521) (355,519) (399,652)

Room Name: 6
Room Size (pixel): 4821
Center of Gravity (x,y): (406,494)
Room Color (r,g,b,a): (0,0,101,255)
Room Control Points (x,y): (401,527) (413,523) (430,526) (448,524) (459,507)
(457,491) (439,489) (423,493) (400,482) (391,471) (380,462) (370,452) (360,460)
(366,496) (365,507)

Room Name: 7
Room Size (pixel): 1486
Center of Gravity (x,y): (498,526)
Room Color (r,g,b,a): (102,0,0,255)
Room Control Points (x,y): (462,528) (476,534) (488,542) (501,537) (519,535)
(532,528) (529,517) (511,515) (491,515) (472,514)

Room Name: 8
Room Size (pixel): 11219
Center of Gravity (x,y): (271,274)
Room Color (r,g,b,a): (0,102,0,255)
```

```
Room Control Points (x,y): (233,292) (229,308) (229,328) (236,341) (256,341)
(272,341) (289,344) (309,344) (329,344) (337,332) (337,312) (336,293) (281,297)
(279,281) (279,261) (279,241) (279,221) (278,202) (275,185) (257,183) (237,183)
(224,190) (225,205) (224,222)

Room Name: 9
Room Size (pixel): 20299
Center of Gravity (x,y): (354,243)
Room Color (r,g,b,a): (0,0,102,255)
Room Control Points (x,y): (348,291) (345,308) (344,327) (344,347) (346,365)
(347,382) (352,397) (370,399) (387,396) (394,348) (398,284) (394,268) (394,250)
(393,231) (393,211) (393,191) (393,171) (393,151) (392,132) (380,126) (361,125)
(338,148) (338,168) (333,183) (313,183) (293,183) (284,194) (284,214) (289,229)
(307,231) (327,231) (338,240) (387,124)

Room Name: 10
Room Size (pixel): 1553
Center of Gravity (x,y): (421,434)
Room Color (r,g,b,a): (103,0,0,255)
Room Control Points (x,y): (405,398) (404,417) (408,433) (409,450) (423,450)
(442,449) (460,447)

Room Name: 11
Room Size (pixel): 17401
Center of Gravity (x,y): (501,346)
Room Color (r,g,b,a): (0,103,0,255)
Room Control Points (x,y): (465,462) (465,482) (468,499) (483,502) (501,502)
(519,502) (519,470) (516,455) (515,436) (516,417) (509,404) (511,388) (505,336)
(510,321) (512,303) (528,299) (547,298) (565,296) (569,280) (569,260) (568,241)
(556,233) (536,233) (516,233) (498,235) (480,237) (462,239) (459,256) (459,274)
(463,290) (458,305) (461,322) (465,338) (463,356) (460,373) (465,388)

Room Name: 12
Room Size (pixel): 13047
Center of Gravity (x,y): (602,316)
Room Color (r,g,b,a): (0,0,103,255)
Room Control Points (x,y): (527,392) (543,396) (559,394) (572,387) (589,390)
(609,390) (625,386) (627,368) (627,348) (626,329) (626,309) (629,292) (645,288)
(665,288) (681,288) (685,272) (685,252) (674,243) (654,243) (635,244) (619,248)
(601,246) (584,245) (576,257) (577,276) (577,296) (576,315) (574,331) (572,349)
(557,344) (539,342) (524,347) (509,352)

Room Name: 13
Room Size (pixel): 4676
Center of Gravity (x,y): (439,362)
Room Color (r,g,b,a): (104,0,0,255)
Room Control Points (x,y): (455,394) (455,374) (452,357) (443,347) (426,350)
(408,350)

Room Name: 14
Room Size (pixel): 1633
```

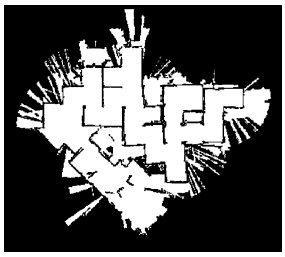
```
Center of Gravity (x,y): (424,316)
Room Color (r,g,b,a): (0,104,0,255)
Room Control Points (x,y): (401,335) (418,332) (435,329) (447,321) (447,303)
(419,297) (403,299)

Room Name: 15
Room Size (pixel): 722
Center of Gravity (x,y): (160,227)
Room Color (r,g,b,a): (0,0,104,255)
Room Control Points (x,y): (151,228) (155,244) (163,240) (166,223) (166,203)
(160,205) (159,249)

Room Name: 16
Room Size (pixel): 1524
Center of Gravity (x,y): (184,470)
Room Color (r,g,b,a): (105,0,0,255)
Room Control Points (x,y): (166,487) (182,491) (198,491) (207,485) (197,475)
(188,464) (182,450) (186,422) (169,463) (167,481)

Room Name: 17
Room Size (pixel): 384
Center of Gravity (x,y): (198,611)
Room Color (r,g,b,a): (0,105,0,255)
Room Control Points (x,y): (185,627) (195,617) (207,609) (218,600) (214,597)
(202,605) (190,613) (179,622)

Room Name: 18
Room Size (pixel): 591
Center of Gravity (x,y): (214,526)
Room Color (r,g,b,a): (0,0,105,255)
Room Control Points (x,y): (195,525) (211,521) (229,519) (238,522) (224,528)
(209,533) (193,537)

Room Name: 19
Room Size (pixel): 3128
Center of Gravity (x,y): (255,149)
Room Color (r,g,b,a): (106,0,0,255)
Room Control Points (x,y): (235,128) (254,127) (273,128) (294,141) (293,153)
(281,161) (270,170) (257,163) (243,169) (230,164) (222,152) (219,135)

Room Name: 20
Room Size (pixel): 6541
Center of Gravity (x,y): (321,399)
Room Color (r,g,b,a): (0,106,0,255)
Room Control Points (x,y): (269,368) (278,357) (292,355) (311,354) (330,355)
(336,369) (336,389) (337,408) (356,409) (375,408) (393,408) (400,421) (392,431)
(374,433) (358,433) (335,440) (322,447) (311,450) (312,423) (311,404) (295,404)
(288,417) (277,408) (268,397) (257,388) (300,368) (312,376) (315,393) (304,386)
(299,371)

Room Name: 21
```

```
Room Size (pixel): 984
Center of Gravity (x,y): (327,92)
Room Color (r,g,b,a): (0,0,106,255)
Room Control Points (x,y): (314,83) (322,95) (324,113) (333,116) (344,107)
(339,94) (339,80) (327,72) (309,72)

Room Name: 22
Room Size (pixel): 1429
Center of Gravity (x,y): (363,55)
Room Color (r,g,b,a): (107,0,0,255)
Room Control Points (x,y): (360,14) (371,23) (371,43) (371,63) (373,79) (369,95)
(359,87) (358,68) (355,51) (350,36)

Room Name: 23
Room Size (pixel): 1028
Center of Gravity (x,y): (398,89)
Room Color (r,g,b,a): (0,107,0,255)
Room Control Points (x,y): (389,101) (392,84) (393,65) (403,69) (411,78)
(405,92) (403,110) (395,116)

Room Name: 24
Room Size (pixel): 3569
Center of Gravity (x,y): (426,246)
Room Color (r,g,b,a): (0,0,107,255)
Room Control Points (x,y): (405,207) (417,215) (431,213) (443,215) (447,231)
(452,246) (447,261) (445,279) (432,286) (414,284) (408,270) (403,255) (402,240)

Room Name: 25
Room Size (pixel): 786
Center of Gravity (x,y): (501,212)
Room Color (r,g,b,a): (108,0,0,255)
Room Control Points (x,y): (486,209) (492,223) (507,224) (521,218) (512,211)
(503,200) (489,200)

Room Name: 26
Room Size (pixel): 932
Center of Gravity (x,y): (541,321)
Room Color (r,g,b,a): (0,108,0,255)
Room Control Points (x,y): (520,312) (539,311) (557,313) (566,324) (554,332)
(534,332) (520,326)

Room Name: 27
Room Size (pixel): 332
Center of Gravity (x,y): (605,419)
Room Color (r,g,b,a): (0,0,108,255)
Room Control Points (x,y): (596,418) (607,427) (618,420) (607,411) (591,411)
(613,429)

Room Name: 28
Room Size (pixel): 6862
Center of Gravity (x,y): (718,291)
```

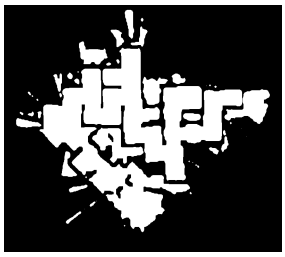
```
Room Color (r,g,b,a): (109,0,0,255)
Room Control Points (x,y): (689,299) (689,281) (689,265) (695,251) (711,247)
(729,245) (745,243) (760,248) (764,262) (752,270) (754,308) (747,321) (741,335)
(729,339) (718,330) (707,321) (691,319) (674,322) (658,324)
```



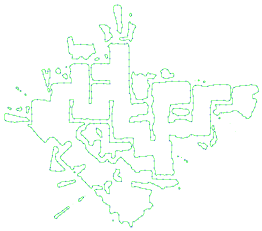
(a) Original



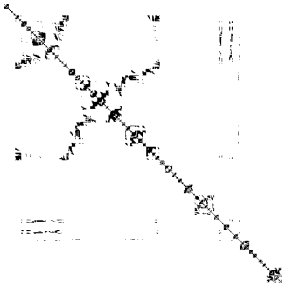
(b) Binarisiert



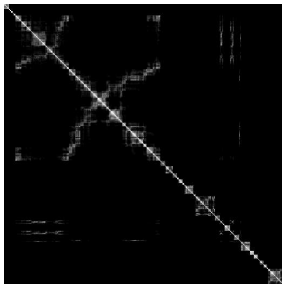
(c) Dilatation und Erosion



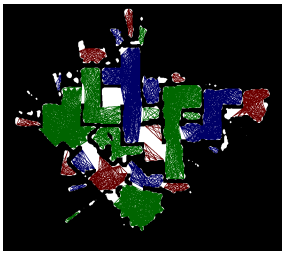
(d) Hinderniskontrollpunkte



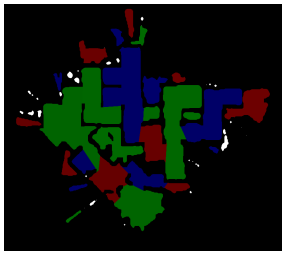
(e) Adjazenzmatrix



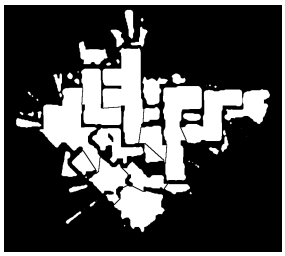
(f) Ähnlichkeitsmatrix



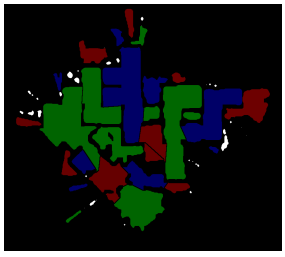
(g) Raumnetz



(h) Flutung



(i) Türen



(j) Resultat

Bild B.8: Beispiel mit von Robbie erzeugter Karte

B.7 Beispiel mit unerwartetem Resultat

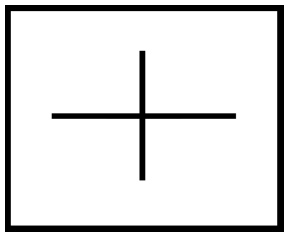
B.7.1 Log

```
Open Image
 Filename: cross.png
 Image Width: 863
 Image Height: 704
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 50
 Count of Control Points: 98
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 70
 Minimum Room Size (as count of control points): 5
 Count of Rooms: 5
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
Save File
```

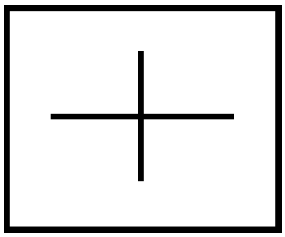
B.7.2 Kartendeskriptor, Zwischenergebnisse und Resultat

```
Room Details
 of map (image): cross.png

Room Name: 1
Room Size (pixel): 434672
Center of Gravity (x,y): (432,353)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (67,19) (117,19) (167,19) (217,19) (267,19) (317,19)
(367,19) (417,19) (467,19) (517,19) (567,19) (617,19) (667,19) (717,19) (767,19)
(817,19) (841,45) (841,95) (841,145) (841,195) (841,245) (841,295) (841,345)
(841,395) (841,445) (841,495) (841,545) (841,595) (841,645) (829,683) (779,683)
```

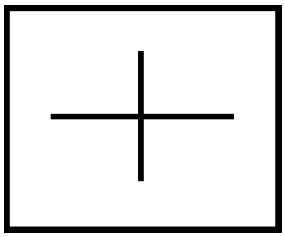
```
(729,683) (679,683) (629,683) (579,683) (529,683) (479,683) (429,683) (379,683)
(329,683) (279,683) (229,683) (179,683) (129,683) (79,683) (29,683) (18,644)
(18,594) (18,544) (18,494) (18,444) (18,394) (18,344) (18,294) (18,244) (18,194)
(18,144) (18,94) (18,44)

Room Name: 2
Room Size (pixel): 19420
Center of Gravity (x,y): (333,299)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (169,335) (219,335) (269,335) (319,335) (369,335)
(415,331) (415,281) (415,231) (415,181)

Room Name: 3
Room Size (pixel): 19373
Center of Gravity (x,y): (515,300)
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (434,182) (434,232) (434,282) (434,332) (481,335)
(531,335) (581,335) (631,335) (681,335)

Room Name: 4
Room Size (pixel): 29870
Center of Gravity (x,y): (506,394)
Room Color (r,g,b,a): (101,0,0,255)
Room Control Points (x,y): (665,353) (615,353) (565,353) (515,353) (465,353)
(434,372) (434,422) (434,472) (434,522)

Room Name: 5
Room Size (pixel): 28022
Center of Gravity (x,y): (329,395)
Room Color (r,g,b,a): (0,101,0,255)
Room Control Points (x,y): (415,535) (415,485) (415,435) (415,385) (397,353)
(347,353) (297,353) (247,353) (197,353) (147,353)
```



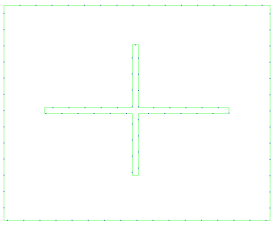
(a) Original



(b) Binarisiert

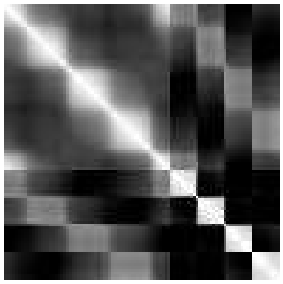


(c) Dilatation und Erosion

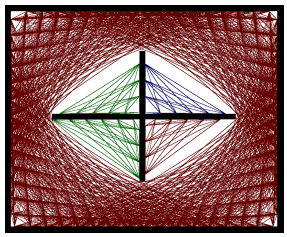


(d) Hinderniskontrollpunkte

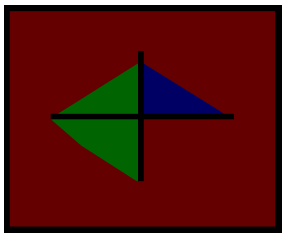
(e) Adjazenzmatrix



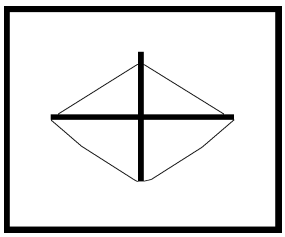
(f) Ähnlichkeitsmatrix



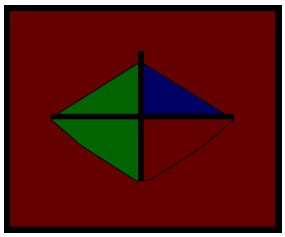
(g) Raumnetz



(h) Flutung



(i) Türen



(j) Resultat

Bild B.9: Beispiel mit unerwartetem Resultat

B.8 Beispiel mit verrauschtem Ergebnis

B.8.1 Log

```
Open Image
 Filename: 2015_agas_viele_raeume.gif
 Image Width: 1500
 Image Height: 400
Binarize
 Obstacle Threshold: 50
Preprocess
 Obstacle Threshold: 50
 Dilatation: 0
 Erosion: 0
Control Point
 Obstacle Threshold: 50
 Control Point Distance: 15
 Count of Control Points: 1113
Adjacency
 Obstacle Threshold: 50
Similarity
Roomnet
 Obstacle Threshold: 50
 Similarity Threshold(%): 45
 Minimum Room Size (as count of control points): 5
 Count of Rooms: 38
Floodnet
 Obstacle Threshold: 50
Door
 Obstacle Threshold: 50
Result
 Obstacle Threshold: 50
Descriptor
```

B.8.2 Kartendeskriptor, Zwischenergebnisse und Resultat

```
Room Details
 of map (image): 2015_agas_viele_raeume.gif

Room Name: 1
Room Size (pixel): 54874
Center of Gravity (x,y): (536,196)
Room Color (r,g,b,a): (100,0,0,255)
Room Control Points (x,y): (39,180) (41,167) (56,167) (91,167) (106,167) (121,167)
(136,167) (151,167) (166,167) (181,167) (330,168) (345,168) (360,168) (367,176)
(474,176) (482,169) (497,169) (512,169) (527,169) (562,167) (577,167) (592,167)
(607,167) (622,167) (637,167) (652,167) (667,167) (832,168) (847,168) (862,168)
(877,168) (892,168) (907,168) (922,168) (937,168) (974,170) (989,170) (1004,170)
```

```
(1019,170) (1024,180) (1024,222) (1015,228) (1000,228) (668,227) (653,227) (638,227)
(623,227) (608,227) (593,227) (578,227) (563,227) (516,229) (501,229) (486,229)
(474,226) (374,227) (369,217) (364,227) (349,227) (334,227) (168,226) (153,226)
(147,217) (104,218) (38,213) (26,210) (26,195) (203,226) (218,226) (233,226)
(248,226) (263,226) (278,226) (293,226) (289,168) (274,168) (259,168) (244,168)
(229,168) (214,168) (799,169) (784,169) (769,169) (731,168) (716,168) (701,168)
(692,228) (707,228) (722,228) (737,228) (752,228) (767,228) (782,228) (797,228)
(812,228) (849,229) (864,229) (879,229) (894,229) (909,229) (924,229)

Room Name: 2
Room Size (pixel): 1971
Center of Gravity (x,y): (60,143)
Room Color (r,g,b,a): (0,100,0,255)
Room Control Points (x,y): (53,162) (41,159) (41,144) (46,134) (61,134) (76,134) (102,136)

Room Name: 3
Room Size (pixel): 5979
Center of Gravity (x,y): (69,77)
Room Color (r,g,b,a): (0,0,100,255)
Room Control Points (x,y): (68,129) (53,129) (41,126) (41,111) (41,96) (41,81)
(41,66) (41,51) (41,36) (54,34) (69,34) (84,34) (99,34) (101,47) (101,62) (101,77)
(101,92) (101,107) (101,122)

Room Name: 4
Room Size (pixel): 8219
Center of Gravity (x,y): (139,95)
Room Color (r,g,b,a): (101,0,0,255)
Room Control Points (x,y): (106,125) (106,110) (106,95) (106,80) (106,65) (106,50)
(106,35) (120,34) (135,34) (150,34) (165,34) (169,45) (169,60) (169,75) (169,90)
(169,105) (169,120) (169,135) (169,150) (166,162) (151,162) (136,162) (121,162) (108,160)

Room Name: 5
Room Size (pixel): 11402
Center of Gravity (x,y): (211,90)
Room Color (r,g,b,a): (0,101,0,255)
Room Control Points (x,y): (183,162) (174,156) (174,141) (174,126) (174,111) (174,96)
(174,81) (174,66) (174,51) (174,36) (176,23) (191,23) (204,21) (219,21) (233,22)
(248,22) (252,33) (252,48) (252,63) (211,163) (226,163) (241,163) (252,159) (252,144)
(252,129) (252,114) (252,99)

Room Name: 6
Room Size (pixel): 17076
Center of Gravity (x,y): (318,88)
Room Color (r,g,b,a): (0,0,101,255)
Room Control Points (x,y): (257,57) (257,42) (257,27) (265,20) (280,20) (294,21)
(309,21) (324,21) (337,19) (352,19) (365,21) (374,27) (374,42) (374,57) (374,72)
(374,87) (374,102) (374,117) (374,132) (374,147) (374,162) (360,163) (345,163)
(330,163) (257,91) (257,106) (257,121) (257,136) (257,151) (260,163) (275,163) (290,163)

Room Name: 7
Room Size (pixel): 29411
```

```
Center of Gravity (x,y): (420,193)
Room Color (r,g,b,a): (102,0,0,255)
Room Control Points (x,y): (372,170) (379,162) (379,147) (379,132) (379,117)
(379,102) (379,87) (379,72) (379,57) (379,42) (379,27) (388,21) (402,20) (417,20)
(432,20) (446,21) (461,21) (461,36) (461,51) (461,66) (461,81) (461,96) (461,111)
(461,126) (461,141) (461,156) (463,169) (469,178) (461,244) (461,259) (461,274)
(461,289) (461,304) (461,319) (461,334) (461,349) (461,364) (454,372) (439,372)
(424,372) (409,372) (396,370) (381,370) (381,355) (381,340) (381,325) (381,310)
(381,295) (381,280) (381,265) (381,250) (381,235)

Room Name: 8
Room Size (pixel): 4925
Center of Gravity (x,y): (536,148)
Room Color (r,g,b,a): (0,102,0,255)
Room Control Points (x,y): (517,164) (502,164) (487,164) (472,164) (466,155) (466,140)
(471,130) (486,130) (501,130) (532,130) (575,131) (590,131) (605,131) (608,143)
(608,158) (597,162) (582,162) (567,162) (552,162)

Room Name: 9
Room Size (pixel): 7464
Center of Gravity (x,y): (499,69)
Room Color (r,g,b,a): (0,0,102,255)
Room Control Points (x,y): (505,125) (490,125) (475,125) (466,119) (466,104) (466,89)
(466,74) (466,59) (466,44) (466,29) (473,21) (487,20) (502,20) (517,20) (530,22)
(535,32) (535,47) (535,62) (535,77) (535,92) (535,107) (535,122)

Room Name: 10
Room Size (pixel): 7377
Center of Gravity (x,y): (574,72)
Room Color (r,g,b,a): (103,0,0,255)
Room Control Points (x,y): (544,127) (540,116) (540,101) (540,86) (540,71) (540,56)
(540,41) (540,26) (551,22) (564,20) (579,20) (592,22) (607,22) (608,36) (608,51)
(608,66) (608,81) (608,96) (608,111) (608,126) (593,126) (578,126) (564,127)

Room Name: 11
Room Size (pixel): 36080
Center of Gravity (x,y): (750,87)
Room Color (r,g,b,a): (0,103,0,255)
Room Control Points (x,y): (657,162) (642,162) (627,162) (613,161) (613,146) (613,131)
(613,116) (613,101) (613,86) (613,71) (613,56) (613,41) (613,26) (624,22) (636,19)
(651,19) (664,21) (679,21) (694,21) (708,20) (723,20) (737,21) (752,21) (767,21)
(781,20) (796,20) (809,22) (824,22) (837,20) (852,20) (867,20) (880,22) (886,31)
(886,46) (886,61) (886,76) (886,91) (886,106) (886,121) (886,136) (886,151) (883,163)
(868,163) (853,163) (838,163) (698,163) (702,152) (702,137) (706,126) (721,126) (736,126)
(751,126) (766,126) (781,126) (794,128) (794,143) (794,158) (803,164)

Room Name: 12
Room Size (pixel): 5030
Center of Gravity (x,y): (962,148)
Room Color (r,g,b,a): (0,0,103,255)
Room Control Points (x,y): (943,163) (928,163) (913,163) (898,163) (891,155) (891,140)
```

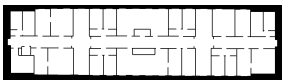
```
(896,130) (911,130) (926,130) (957,130) (993,131) (1008,131) (1023,131) (1032,137)
(1032,152) (1030,165) (1015,165) (1000,165) (985,165) (970,165)

Room Name: 13
Room Size (pixel): 7458
Center of Gravity (x,y): (924,69)
Room Color (r,g,b,a): (104,0,0,255)
Room Control Points (x,y): (928,125) (913,125) (898,125) (891,117) (891,102) (891,87)
(891,72) (891,57) (891,42) (891,27) (901,22) (914,20) (929,20) (944,20) (957,22)
(960,34) (960,49) (960,64) (960,79) (960,94) (960,109) (960,124)

Room Name: 14
Room Size (pixel): 7282
Center of Gravity (x,y): (999,70)
Room Color (r,g,b,a): (0,104,0,255)
Room Control Points (x,y): (967,125) (965,112) (965,97) (965,82) (965,67) (965,52)
(965,37) (965,22) (980,22) (993,20) (1008,20) (1021,22) (1032,26) (1032,41) (1032,56)
(1032,71) (1032,86) (1032,101) (1032,116) (1027,126) (1012,126) (997,126)

Room Name: 15
Room Size (pixel): 30900
Center of Gravity (x,y): (1080,197)
Room Color (r,g,b,a): (0,0,104,255)
Room Control Points (x,y): (1037,163) (1037,148) (1037,133) (1037,118) (1037,103)
(1037,88) (1037,73) (1037,58) (1037,43) (1037,28) (1046,22) (1061,22) (1075,21)
(1090,21) (1103,23) (1118,23) (1122,34) (1122,49) (1122,64) (1122,79) (1122,94)
(1122,109) (1122,124) (1122,139) (1122,154) (1123,168) (1129,177) (1130,224)
(1119,228) (1116,240) (1116,255) (1116,270) (1116,285) (1116,300) (1116,315)
(1116,330) (1116,345) (1116,360) (1116,375) (1101,375) (1088,377) (1073,377)
(1059,376) (1044,376) (1037,368) (1037,353) (1037,338) (1037,323) (1037,308)
(1037,293) (1037,278) (1037,263) (1037,248) (1037,233)

Room Name: 16
Room Size (pixel): 20301
Center of Gravity (x,y): (1310,201)
Room Color (r,g,b,a): (105,0,0,255)
Room Control Points (x,y): (1134,177) (1142,170) (1157,170) (1172,170) (1210,171)
(1225,171) (1240,171) (1255,171) (1270,171) (1285,171) (1327,171) (1342,171)
(1357,171) (1372,171) (1387,171) (1402,171) (1436,171) (1451,171) (1461,176)
(1465,187) (1473,194) (1473,209) (1469,220) (1461,226) (1450,230) (1435,230)
(1420,230) (1405,230) (1290,230) (1275,230) (1260,230) (1245,230) (1230,230)
(1215,230) (1200,230) (1168,228) (1153,228) (1138,228) (1135,216) (1320,230)
(1335,230) (1350,230) (1365,230) (1319,231)

Room Name: 17
Room Size (pixel): 17085
Center of Gravity (x,y): (1185,89)
Room Color (r,g,b,a): (0,105,0,255)
Room Control Points (x,y): (1166,165) (1151,165) (1136,165) (1127,159) (1127,144)
(1127,129) (1127,114) (1127,99) (1127,84) (1127,69) (1127,54) (1127,39) (1127,24)
(1141,23) (1154,21) (1169,21) (1182,23) (1197,23) (1212,23) (1227,23) (1242,23)
```

```
(1244,36) (1244,51) (1244,66) (1244,81) (1244,96) (1244,111) (1244,126) (1244,141)
(1244,156) (1239,166) (1224,166) (1209,166)

Room Name: 18
Room Size (pixel): 11522
Center of Gravity (x,y): (1287,90)
Room Color (r,g,b,a): (0,0,105,255)
Room Control Points (x,y): (1287,166) (1272,166) (1257,166) (1249,159) (1249,144)
(1249,129) (1249,114) (1249,99) (1249,84) (1249,69) (1249,54) (1249,39) (1250,25)
(1265,25) (1278,23) (1293,23) (1308,23) (1321,25) (1328,33) (1328,48) (1328,63)
(1328,78) (1328,93) (1328,108) (1328,123) (1328,138) (1328,153) (1326,166)

Room Name: 19
Room Size (pixel): 6143
Center of Gravity (x,y): (1429,89)
Room Color (r,g,b,a): (106,0,0,255)
Room Control Points (x,y): (1413,167) (1405,139) (1401,128) (1401,113) (1401,98)
(1401,83) (1401,68) (1401,53) (1401,38) (1415,37) (1430,37) (1445,37) (1460,37)
(1461,51) (1461,66) (1461,81) (1461,96) (1461,111) (1461,126) (1454,134) (1439,134)
(1424,134)

Room Name: 20
Room Size (pixel): 10127
Center of Gravity (x,y): (1380,110)
Room Color (r,g,b,a): (0,106,0,255)
Room Control Points (x,y): (1399,166) (1384,166) (1369,166) (1354,166) (1339,166)
(1333,157) (1333,142) (1333,127) (1333,112) (1333,97) (1333,82) (1333,67) (1333,52)
(1333,37) (1348,37) (1363,37) (1378,37) (1393,37) (1396,49) (1396,64) (1396,79)
(1396,94) (1396,109) (1396,124) (1396,139) (1434,139) (1449,139) (1461,142)
(1461,157) (1455,166) (1440,166)

Room Name: 21
Room Size (pixel): 16560
Center of Gravity (x,y): (1400,301)
Room Color (r,g,b,a): (0,0,106,255)
Room Control Points (x,y): (1401,235) (1416,235) (1431,235) (1446,235) (1461,235)
(1461,250) (1461,265) (1461,280) (1461,295) (1461,310) (1461,325) (1461,340)
(1461,355) (1456,365) (1441,365) (1426,365) (1411,365) (1396,365) (1381,365)
(1366,365) (1351,365) (1336,365) (1335,351) (1335,336) (1373,235) (1358,235)
(1343,235) (1336,243) (1336,258) (1336,273) (1336,288) (1336,303)

Room Name: 22
Room Size (pixel): 10486
Center of Gravity (x,y): (1292,309)
Room Color (r,g,b,a): (107,0,0,255)
Room Control Points (x,y): (1330,342) (1330,357) (1330,372) (1320,377) (1306,378)
(1291,378) (1278,376) (1263,376) (1259,365) (1259,350) (1259,335) (1259,320)
(1259,305) (1259,290) (1259,275) (1259,260) (1259,245) (1264,235) (1279,235)
(1294,235) (1331,309) (1331,294) (1331,279) (1331,264) (1331,249) (1330,235)

Room Name: 23
```

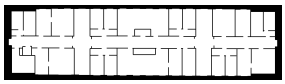
```
Room Size (pixel): 4918
Center of Gravity (x,y): (1189,250)
Room Color (r,g,b,a): (0,107,0,255)
Room Control Points (x,y): (1210,235) (1225,235) (1240,235) (1254,236) (1254,251)
(1254,266) (1197,268) (1151,268) (1136,268) (1121,268) (1121,253) (1121,238)
(1131,233) (1146,233) (1161,233) (1176,233) (1219,268) (1234,268) (1249,268)

Room Name: 24
Room Size (pixel): 6221
Center of Gravity (x,y): (1225,330)
Room Color (r,g,b,a): (0,0,107,255)
Room Control Points (x,y): (1254,281) (1254,296) (1254,311) (1254,326) (1254,341)
(1254,356) (1254,371) (1244,376) (1231,378) (1216,378) (1203,376) (1195,369)
(1195,354) (1195,339) (1195,324) (1195,309) (1195,294) (1195,279) (1225,273) (1240,273)

Room Name: 25
Room Size (pixel): 7386
Center of Gravity (x,y): (1155,325)
Room Color (r,g,b,a): (108,0,0,255)
Room Control Points (x,y): (1186,272) (1190,283) (1190,298) (1190,313) (1190,328)
(1190,343) (1190,358) (1190,373) (1178,376) (1165,378) (1150,378) (1137,376)
(1123,375) (1121,362) (1121,347) (1121,332) (1121,317) (1121,302) (1121,287)
(1122,273) (1137,273) (1152,273) (1164,270)

Room Name: 26
Room Size (pixel): 10119
Center of Gravity (x,y): (998,308)
Room Color (r,g,b,a): (0,108,0,255)
Room Control Points (x,y): (1003,233) (1018,233) (1032,234) (1032,249) (1032,264)
(1032,279) (1032,294) (1032,309) (1032,324) (1032,339) (1032,354) (1032,369)
(1024,376) (1010,377) (995,377) (980,377) (967,375) (963,364) (963,349) (963,334)
(963,319) (963,304) (963,289) (963,274) (963,259) (963,244) (966,232)

Room Name: 27
Room Size (pixel): 9856
Center of Gravity (x,y): (926,306)
Room Color (r,g,b,a): (0,0,108,255)
Room Control Points (x,y): (958,236) (958,251) (958,266) (958,281) (958,296)
(958,311) (958,326) (958,341) (958,356) (958,371) (947,375) (933,376) (918,376)
(905,374) (890,374) (889,360) (889,345) (932,234) (917,234) (902,234) (890,237)
(890,252) (890,267) (890,282) (890,297)

Room Name: 28
Room Size (pixel): 34787
Center of Gravity (x,y): (745,308)
Room Color (r,g,b,a): (109,0,0,255)
Room Control Points (x,y): (884,342) (884,357) (884,372) (871,374) (858,376)
(843,376) (830,374) (815,374) (801,375) (786,375) (771,375) (758,373) (743,373)
(730,375) (715,375) (700,375) (688,372) (673,372) (660,374) (646,375) (631,375)
(618,373) (612,364) (612,349) (612,334) (612,319) (612,304) (612,289) (612,274)
(612,259) (612,244) (615,232) (630,232) (645,232) (660,232) (822,233) (819,245)
```

```
(819,260) (811,267) (796,267) (781,267) (766,267) (751,267) (736,267) (721,267)
(706,267) (694,264) (694,249) (694,234) (886,308) (885,294) (885,279) (885,264)
(885,249) (885,234) (870,234) (855,234)

Room Name: 29
Room Size (pixel): 4858
Center of Gravity (x,y): (535,249)
Room Color (r,g,b,a): (0,109,0,255)
Room Control Points (x,y): (568,232) (583,232) (598,232) (607,238) (607,253)
(604,265) (589,265) (574,265) (539,265) (508,265) (493,265) (478,265) (466,262)
(466,247) (468,234) (483,234) (498,234) (513,234)

Room Name: 30
Room Size (pixel): 6922
Center of Gravity (x,y): (574,323)
Room Color (r,g,b,a): (0,0,109,255)
Room Control Points (x,y): (568,270) (583,270) (598,270) (607,276) (607,291)
(607,306) (607,321) (607,336) (607,351) (607,366) (599,373) (586,375) (571,375)
(556,375) (544,372) (542,359) (542,344) (542,329) (542,314) (542,299) (542,284)
(543,270)

Room Name: 31
Room Size (pixel): 7492
Center of Gravity (x,y): (500,324)
Room Color (r,g,b,a): (110,0,0,255)
Room Control Points (x,y): (535,270) (537,283) (537,298) (537,313) (537,328)
(537,343) (537,358) (536,372) (521,372) (508,374) (493,374) (480,372) (466,371)
(466,356) (466,341) (466,326) (466,311) (466,296) (466,281) (470,270) (485,270)
(500,270)

Room Name: 32
Room Size (pixel): 18777
Center of Gravity (x,y): (312,302)
Room Color (r,g,b,a): (0,110,0,255)
Room Control Points (x,y): (324,232) (339,232) (354,232) (369,232) (376,240)
(376,255) (376,270) (376,285) (376,300) (376,315) (376,330) (376,345) (376,360)
(371,370) (357,371) (342,371) (327,371) (312,371) (297,371) (283,370) (268,370)
(253,370) (243,365) (243,350) (243,335) (295,231) (280,231) (265,231) (250,231)
(243,239) (243,254) (243,269) (243,284) (243,299)

Room Name: 33
Room Size (pixel): 10536
Center of Gravity (x,y): (197,300)
Room Color (r,g,b,a): (0,0,110,255)
Room Control Points (x,y): (238,327) (238,342) (238,357) (236,370) (221,370)
(206,370) (191,370) (177,369) (169,362) (169,347) (169,332) (169,317) (169,302)
(169,287) (169,272) (160,266) (151,260) (151,245) (152,231) (167,231) (238,295)
(238,280) (238,265) (238,250) (238,235) (227,231) (212,231)

Room Name: 34
Room Size (pixel): 2004
```

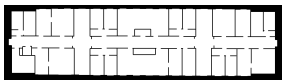
```
Center of Gravity (x,y): (124,247)
Room Color (r,g,b,a): (111,0,0,255)
Room Control Points (x,y): (143,222) (146,234) (146,249) (146,264) (133,266)
(118,266) (103,266) (102,252) (102,237) (103,223)

Room Name: 35
Room Size (pixel): 12567
Center of Gravity (x,y): (98,305)
Room Color (r,g,b,a): (0,111,0,255)
Room Control Points (x,y): (81,236) (81,251) (81,266) (91,271) (106,271) (121,271)
(136,271) (151,271) (164,273) (164,288) (164,303) (164,318) (164,333) (164,348)
(162,361) (147,361) (132,361) (117,361) (102,361) (87,361) (72,361) (57,361) (43,360)
(43,345) (43,330) (43,315) (43,300) (43,285) (43,270) (43,255) (43,240) (50,232)

Room Name: 36
Room Size (pixel): 420
Center of Gravity (x,y): (91,249)
Room Color (r,g,b,a): (0,0,111,255)
Room Control Points (x,y): (87,232) (97,237) (97,252) (96,266) (86,261) (86,246)

Room Name: 37
Room Size (pixel): 2874
Center of Gravity (x,y): (748,145)
Room Color (r,g,b,a): (112,0,0,255)
Room Control Points (x,y): (765,164) (780,164) (789,158) (789,143) (786,131) (771,131)
(756,131) (741,131) (726,131) (711,131) (707,142) (707,157) (716,163) (731,163)

Room Name: 38
Room Size (pixel): 3480
Center of Gravity (x,y): (757,247)
Room Color (r,g,b,a): (0,112,0,255)
Room Control Points (x,y): (706,233) (721,233) (736,233) (751,233) (766,233) (781,233)
(796,233) (811,233) (814,245) (814,260) (801,262) (786,262) (771,262) (756,262)
(741,262) (726,262) (711,262) (699,259) (699,244)
```



(a) Original

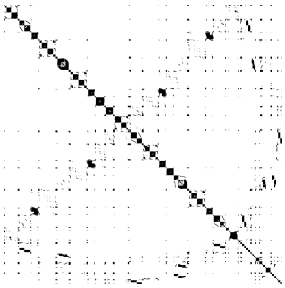


(b) Binarisiert

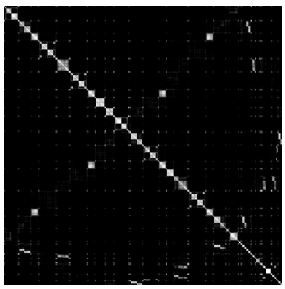


(c) Dilatation und Erosion

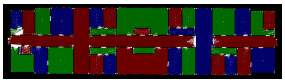
(d) Hinderniskontrollpunkte



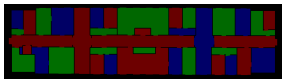
(e) Adjazenzmatrix



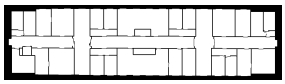
(f) Ähnlichkeitsmatrix



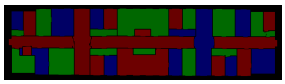
(g) Raumnetz



(h) Flutung



(i) Türen



(j) Resultat

Bild B.10: Beispiel mit verrauschtem Ergebnis

C Orientierungshilfe zum Graphical User Interface

```
Menuebar
 File
  Open Image
   to open an pixel image as map
  Save Image
   to save an pixel image in format png
   savable images are
    the original image
    the binarized image
    the preprocessed image
    the control point image
    the adjacency matrix as image
    the similarity matrices as image
    the roomnet image
    the floodnet image
    the door image
    the result image
  Close Image
   to close the original image
   the interim results, the result, the log and the room details are also abolished
  Save Descriptor
   log
    to save the descriptor log as text file
   room details
    to save the descriptor room details as text file
  Exit
    to terminate the roomfinder program
 Zoom
  Zoom In
   to zoom into the image of the actual tab
  Zoom Out
   to zoom out of the image of the actual tab
  Normal Size
   to zoom the image of the actual tab to its original size
  Fit to Page
   to zoom the image of the actual tab to the tab’s size so that the entire image is shown
 Help
```

```
  Show Help
   to display this help

reset and refine
 reset
  to reset the interim results, the result and the room details
  rest does not reset the parameters and the log
 refine
  to set the result image as new original image for refinement
  the interim results and the room details are abolished
  the log remains untouched

parameters
 obstacle threshold
  to set the obstacle threshold as grey value [0..255] to define obstacles
 dilatation masksize
  to set the dilatation masksize as nonnegative integer
  e.g. masksize 1 creates a 3x3-mask, masksize 3 creates a 7x7-mask
 erosion iteration
  to set the erosion iteration as nonnegative integer
 control point step
  to set the distance between two neighboured control points as nonnegative integer
    representing pixel
  each control-point-step-th point on an obstacle's outline is chosen as control point
 similarity
  to set the similarity as per cent value [0..100]
 min room size
  to set the minimum room size as nonnegative integer representing pixel

steps
 the step buttons generate the interim resuls, the result and the room details
 the interim results are displayed in the tab corresponding to the step button

 if a button is pressed, every interim result that is needed and has not been computed
   yet is computed
 an interim result, that has been previously computed remains untouched

 the step buttons are:
 1-binaritze
  to binarize the original image
 2-preprocess
  to preprocess the binarized image by dilatation and erosion
 3-controlpoint
  to generate control points on the outline of obstacles in the preprocessed image
 4-adjacency
  to lookup the adjacencies of each to control points to every other control point
  each row of the adjacency matrix represents a visibility set
 5-similarity
  to compute similarity, unsimilarity and similarity relation between visibility sets
    of the adjacency matrix
  the similarity image contains three similarity matrices (from left to right)
```

```
   1. similarity_relation ( which is: unsimilarity / similarity )
   2. similarity (which is: count of concurrant control points / count of maximal
      concurant control points)
   3. unsimilarity (which is: count of unconcurrant control points / count of maximal
      concurrant control points)
 6-roomnet
  to draw the reconstructed rooms as nets
  the nodes of the nets are control points belonging to the room
  each net has it’s own color
 7-floodnet
  to flood the nets in image roomnet
 8-door
  to generate doors at regions where different rooms (different colors collide)
 9-result
  to generate the result image by combining the the floodnet image and the door image
 10-descriptor
  to compute the room descriptors and the map descriptor as room details
  each room descriptor contains room name, room size, center of gravity, room color and
    room control points
  the map descriptor contains the room descriptors

interim results
 original
  displays the original image
 1-binaritze
  displays the binarized image
 2-preprocess
  displays the preprocessed image
 3-controlpoint
  displays the control point image
 4-adjacency
  displays the adjacency matrix as image
 5-similarity
  displays the similarity matrices as image
 6-roomnet
  displays the roomnet image
 7-floodnet
  displays the floodnet image
 8-door
  displays the door image
 9-result
  displays the result image
 10-descriptor
  log
   displays the desriptor log as list
  room details
   displays the descriptor room details as list
```

D Bezeichnungen

Adjazenzmatrix A
Ähnlichkeitsmatrix $\bar{A}$

Basismenge C_Q
Belegung O
Bild I

Elter e

Filtermaskengröße Dilatation δ_d
Filtermaskengröße Erosion δ_e

Grauwert g
Grauwertfunktion η
Grauwertmenge G

Hinderniskontrollpunkt $h, k \in H$
Hinderniskontrollpunktkandidat $\bar{h}$
Hinderniskontrollpunktkandidatenmenge $\bar{H}$
Hinderniskontrollpunktmenge H
Hinderniskontrollpunktschritt λ
Hindernisschwellwert θ

Karte K
Kartendeskriptor D

Mindestraumgröße ϵ

Rasterfeld $f \in F$
Rasterfeldmenge F
Raum r

Raumdeskriptor d
Raumfarbe c, d_{color}
Raumgröße d_{area}
Raumhinderniskontrollpunktmenge Q
Raummenge R
Raumname d_{name}
Raumschwerpunkt d_{cog}

Sehstrahl B Sichtbarkeitsmenge $\bar{S}$
Sichtbarkeitsmengenähnlichkeit ρ
Sichtbarkeitsmengenähnlichkeitsschwellwert ψ
Spaltenunähnlichkeit n
Strahlausgangsmenge H_{Start}
Strahlzielmenge H_{Ziel}

Test auf Sichtbarkeit s

Zeilenähnlichkeit u

Literaturverzeichnis

[AC03] ALTHAUS, P. ; CHRISTENSEN, H.I.: *Behaviour coordination in structured environments*. 2003. – 657–674 S.

[BEP+01] BALTHASAR, Dirk ; ERDMANN, Thomas ; PELLENZ, Johannes ; REHRMANN, Volker ; ZEPPEN, Jörg ; PRIESE, Lutz: Real-Time Detection of Arbitrary Objects in Alternating Industrial Environments. In: *In Proceedings Scandinavian Conference on Image Analysis* 1 (2001), 6, 321-328. `http://www.uni-koblenz.de/~lb/publications/Balthasar2001RDO.pdf`

[BS02] BUSCHKA, P. ; SAFFIOTTI, A.: *A virtual sensor for room detection*. 2002. – 637–642 S.

[BS06] BLANCHETTE, Jasmin ; SUMMERFIELD, Mark: *C++ GUI Programming with Qt 4*. Prentice Hall International, 2006

[BVG00] BAASE, Sara ; VAN GELDER, Allen: *Computer Algorithms - Introduction to Design and Analysis Third Edition*. 2000

[FS95] FREUND, Y ; SCHAPIRE, R.E.: *A decision-theoretic generalization of online learning and an application to boosting*. 1995

[FS97] FOWLER, Martin ; SCOTT, Kendall: *UML Distilled - Applying the Standard Object Modeling Language*. Addison-Wesley, www.addison-wesley.de, 1997

[GSC+05] GALINDO, C. ; SAFFIOTTI, A. ; CORADESCHI, S. ; BUSCHKA, P. ; FERNANDEZ-MADRIGAL, J.A. ; GONZALEZ, J.: Multi-Hierarchical Semantic Maps for Mobile Robotics. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-05)*, 2005

[KN05] KRUMKE, Sven O. ; NOLTEMEIER, Hartmut: *Graphentheoretische Konzepte und Algorithmen*. 2005

[Kow07] KOWALSKI, Kay: Bestimmung von Räumen in Gebäudegrundrissen Universität of Koblenz-Landau, Fachbereich Informatik, 2007

[KS98] KOENIG, S. ; SIMMONS, R.: *A robot navigation architecture based on partially observable markov decision process models*. 1998

[MMSB05] MARTINEZ MOZOS, Oscar ; STACHNISS, Cyrill ; BURGARD, Wolfram: Supervised Learning of Places from Range Data using AdaBoost. In: *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005

[Mon03] MONTEMERLO, Michael: *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association*. Pittsburgh, PA, Robotics Institute, Carnegie Mellon University, http://www.ri.cmu.edu/home.html, Diss., 7 2003. `http://www.ri.cmu.edu/pubs/pub_4434.html`

[PGKKP05] PAGÈS GASSULL, Gassull ; K. KRAETZSCHMAR, Gerhard ; PALM, Günther: Finding Rooms on Probabilistic Quadtrees University of Ulm, Department of Neuroinformatics, 2005

[PH] PAULUS, Dietrich W. ; HORNEGGER, Joachim: *Applied Pattern Recognition 4th Edition*

[PWP07] PELLENZ, Johannes ; WIRTH, Stephan ; PAULUS, Dietrich: Localization, Mapping and Exploration of the Mobile Rescue Robot Robbie 8 Universität of Koblenz-Landau, Fachbereich Informatik, 2007

[Str00] STROUSTRUP, Bjarne: *The C++ Programming Language*. Addison-Wesley, www.addison-wesley.de, 2000

[TMFR03] TORRALBA, A. ; MURPHY, K. ; FREEMAN, W. ; RUBIN, M.: *Context-based vision system for place and object recognition*. 2003