DGK Deutsche Geodätische Kommission
bei der Bayerischen Akademie der Wissenschaften

Reihe C | Dissertationen | Heft Nr. 649

Nora Ripperda

Rekonstruktion von Fassadenstrukturen mittels formaler Grammatiken und Reversible Jump Markov Chain Monte Carlo Sampling

München 2010

Verlag der Bayerischen Akademie der Wissenschaften
in Kommission beim Verlag C. H. Beck

ISSN 0065-5325 ISBN 978-3-7696-5061-7

Diese Arbeit ist gleichzeitig veröffentlicht in:
Wissenschaftliche Arbeiten der Fachrichtung Geodäsie und Geoinformatik der Leibniz Universität Hannover
ISSN 0174-1454, Nr. 284, Hannover 2010

DGK Deutsche Geodätische Kommission
bei der Bayerischen Akademie der Wissenschaften

Reihe C | Dissertationen | Heft Nr. 649

Rekonstruktion von Fassadenstrukturen mittels formaler Grammatiken und Reversible Jump Markov Chain Monte Carlo Sampling

Von der Fakultät für Bauingenieurwesen und Geodäsie
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades
Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation

von

Dipl.-Math. Nora Ripperda
aus Oldenburg

München 2010

Verlag der Bayerischen Akademie der Wissenschaften
in Kommission bei der C. H. Beck'schen Verlagsbuchhandlung München

ISSN 0065-5325 ISBN 978-3-7696-5061-7

Diese Arbeit ist gleichzeitig veröffentlicht in:
Wissenschaftliche Arbeiten der Fachrichtung Geodäsie und Geoinformatik der Leibniz Universität Hannover
ISSN 0174-1454, Nr. 284, Hannover 2010

Adresse der Deutschen Geodätischen Kommission:

Deutsche Geodätische Kommission

Alfons-Goppel-Straße 11 ● D – 80 539 München

Telefon +49 – 89 – 23 031 1113 ● Telefax +49 – 89 – 23 031 - 1283 / - 1100

e-mail hornik@dgfi.badw.de ● http://www.dgk.badw.de

Prüfungskommission

Vorsitzender:	Prof. Dr. Winrich Voß
Referent:	Dr. Claus Brenner
Korreferenten:	Prof. Dr. Helmut Mayer
	Prof. Dr. Christian Heipke

Tag der Promotion: 18.12.2009

ISSN 0065-5325 ISBN 978-3-7696-5061-7

Abstract

Three-dimensional building models are used in a variety of applications. These can be found in fields such as tourism, city planning, or 3D navigation. Due to the increasing number of applications, the demand on 3D models is growing. In order to cover the requirements and to keep the data up to date, automatic reconstruction methods are needed. Because the requirements in the level of detail are increasing simultaneously, in this thesis a method for automatic facade reconstruction is developed. For the reconstruction image and depth data are used, which were acquired using a terrestrial laser scanner.

In this thesis a new method is developed, which combines a Reversible jump Markov Chain Monte Carlo method with formal grammars. The use of grammar rules and priors, which are derived from relative frequencies, ensures that structural and stochastic attributes of facades are considered likewise.

The reconstruction method places special emphasis on the analysis of facade structures. Facade images are analysed to obtain information about the structure of facades. The received information is employed as prior knowledge in the reconstruction process. The knowledge about facades is formulated in a formal grammar. It contains patterns which occur frequently on facades like windows in a grid structure, symmetries or repetitions. If these structures are detected in the data, they can be used for compact storage of the data or for generalisation.

The second important issue of this thesis is the development of an automatic reconstruction procedure. This method generates the derivation tree automatically, which fits the data best, using the rules of the facade grammar. In this thesis a Reversible jump Markov Chain Monte Carlo approach is used. This stochastic process proposes a change of the Markov Chain according to the rules of the facade grammar. Depending on an acceptance probability the change will be accepted or rejected. Furthermore, this thesis addresses the determination of the acceptance probability. It is important to ensure a balance between the quality of the fit of model and data and the model complexity. To take this into account a scoring function based on minimum description length is developed.

Finally the results of the reconstruction are analysed. The reconstruction method was tested with six data sets consisting of depth and image data and one data set consisting of depth data only. Six of seven facades were reconstructed correctly in structure and position of the facade elements. The correctness and completeness of all reconstructions lies between 95.3% and 88.3%.

Additionally the expressiveness of the facade grammar was tested with the aid of a facade image database which contains 56 images. 33.9% of the facades were reconstructed correctly, 21.4% were reconstructed wrong. The remaining 45% show small variations compared to the facade. The correctness of the first group lies between 94.1% and 84.6% and the completeness between 93.9% and 81.8%.

Keywords:
Facade Reconstruction, Formal Grammars, Reversible Jump Markov Chain Monte Carlo

Zusammenfassung

Dreidimensionale Gebäudemodelle werden für eine Vielzahl von Anwendungen benötigt. Diese sind unter anderem in den Bereichen Tourismus, Stadtplanung und 3D-Navigation zu finden. Da sich diese Art von Anwendungen immer weiter verbreitet, steigt die Nachfrage nach 3D-Modellen. Um dem Bedarf gerecht zu werden und die Daten aktuell zu halten, werden automatische Rekonstruktionsverfahren benötigt. Da gleichzeitig die Anforderungen an den Detailgrad der Modelle steigen, wird hier ein Verfahren zur automatischen Rekonstruktion der Fassadenstruktur entwickelt. Als Grundlage der Rekonstruktion dienen Bild- und Entfernungsdaten, die mit einem terrestrischen Laserscanner aufgenommen wurden.

In dieser Arbeit wird ein neuer Ansatz entwickelt, der Reversible jump Markov Chain Monte Carlo Verfahren mit einer formalen Grammatik kombiniert. Die Verwendung von Grammatikregeln und Prior-Wahrscheinlichkeiten, die aus relativen Häufigkeiten abgeleitet wurden, stellt sicher, dass zugleich die strukturellen und stochastischen Eigenschaften typischer Fassaden berücksichtigt werden.

Das Rekonstruktionsverfahren stellt die Analyse der Fassadenstruktur in den Vordergrund. Um Informationen über die Struktur von Fassaden zu bekommen, werden Fassadenbilder analysiert. Die gewonnenen Informationen werden als Vorwissen für die Rekonstruktion verwendet. Das Wissen über die Fassadenstruktur wird in einer formalen Grammatik abgebildet. Es beinhaltet häufig auftretende Muster, wie z.B. gitterförmige Anordnungen von Fenstern, Symmetrien und Wiederholungen. Sind die Strukturen in den Daten erkannt worden, so können sie auch für eine kompakte Speicherung der Modelle und eine generalisierte Visualisierung verwendet werden.

Der zweite wichtige Aspekt dieser Arbeit ist die Entwicklung eines automatischen Verfahrens für die Rekonstruktion. Dies erzeugt automatisch den Ableitungsbaum aus den Regeln der Fassadengrammatik, der am besten zu den gemessenen Daten passt. In dieser Arbeit wird dazu das Reversible jump Markov Chain Monte Carlo Verfahren verwendet. Dieser stochastische Prozess schlägt anhand der Grammatikregeln eine Änderung des Zustands einer Markov-Kette vor. Anhand einer Akzeptanzwahrscheinlichkeit wird diese Änderung angenommen oder verworfen. Ein weiteres Thema der Arbeit ist die Bestimmung dieser Akzeptanzwahrscheinlichkeit. Wichtig hierbei ist die Ausgewogenheit zwischen der Qualität der Zuordnung von Modell und Daten und der Modellkomplexität. Dazu wird eine Bewertungsfunktion entwickelt, die auf der Minimum Description Length basiert.

Abschließend werden die Ergebnisse des Rekonstruktionsverfahrens untersucht. Das Verfahren wurde anhand von sechs Datensätzen bestehend aus Entfernungs- und Bilddaten und einem Datensatz ausschließlich aus Entfernungsdaten getestet. Sechs der sieben Datensätze wurden in der Struktur und auch der Position der Fassadenelemente korrekt rekonstruiert. Die Korrektheit und Vollständigkeit der Rekonstruktionen aller Testdatensätze liegt zwischen 95,3% und 88,3%.

Zusätzlich wird die Aussagekraft der Grammatik anhand einer Datenbank von 56 Fassadenbildern getestet. 33.9% der Fassaden konnten korrekt rekonstruiert werden, 21.4% wurden fehlerhaft rekonstruiert und die verbleibenden 45% weisen kleine Fehler in der Rekonstruktion auf. Die Korrektheit in der ersten Gruppe liegt zwischen 94,1% und 84,6% und die Vollständigkeit zwischen 93,9% und 81,8%.

Schlagworte:
Fassadenrekonstruktion, formale Grammatiken, Reversible Jump Markov Chain Monte Carlo

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Wissenschaftler beschäftigen sich schon seit langer Zeit damit, Verfahren zu entwickeln, die automatisch aus Messdaten anthropogener Objekte die Objekte wieder rekonstruieren. Diese Aufgabe sieht für den Menschen zunächst recht einfach aus, da er diese Objekte problemlos erkennen kann. Dies aber Computern beizubringen, stellt sich als schwierig heraus, da diese nicht über das nötige Wissen verfügen, das der Mensch implizit für diese Aufgabe einsetzt.

Ein Bereich, in dem diese Verfahren gebraucht werden, ist das Erzeugen virtueller 3D-Stadtmodelle. Ein Stadtmodell ist eine dreidimensionale Abbildung bebauter Gebiete und ihrer Umgebung. Hauptbestandteile eines 3D-Stadtmodells sind Gebäude, aber es werden auch Vegetation, Straßen und andere Objekte wie Schilder oder Laternen abgebildet. Die Verwendung von 3D-Stadtmodellen lässt sich in die Kategorien Simulation und Visualisierung gliedern.

Im Bereich der Simulation helfen Stadtmodelle z.B. dabei, die Aufstellung von Funkmasten für den Mobilfunk zu planen. Das 3D-Modell gibt Informationen über Abschattungen und ermöglicht somit die Berechnung optimaler Plätze für Sendemasten. Ebenso kann ein 3D-Stadtmodell bei der Erstellung einer Lärmausbreitungskarte helfen. Für Anwendungen dieser Art werden die Gebäude meist als Klötzchenmodelle dargestellt. Ein höherer Detailgrad ist für die Simulation nicht nötig.

Im Bereich der Visualisierung bietet ein Stadtmodell die Möglichkeit, einen virtuellen Spaziergang durch oder einen virtuellen Flug über ein Gebiet zu generieren. Dies wird im Tourismus genutzt, um für Urlaubsziele zu werben oder um nicht mehr existierende historische Stätten erkunden zu können. Weitere Anwendungsfelder für Stadtmodelle sind der Immobilienmarkt und die Stadtplanung. Hier können zu verkaufende Gebäude und ihre Umgebung dargestellt werden. Ebenso kann schon im Planungsprozess die Wirkung eines neuen Gebäudes in seiner zukünftigen Umgebung gezeigt werden. Auch Hersteller von Computerspielen verwenden mittlerweile Stadtmodelle, um reale Umgebungen zu nutzen. So gibt es Flugsimulatoren, die Flüge über realen Städten wie z.B. New York erlauben. Auch 3D-Spiele verwenden für Außenszenen teilweise Modelle realer Städte. Und für die Tourismusbranche werden mittlerweile Spiele entwickelt, die in bestimmten Städten spielen und somit die Spieler dazu bringen, sich die Stadt auch einmal in der Realität anzusehen. Die Stadt Salzburg hat so ein Spiel entwickeln lassen. Ebenso kann aus dem Stadtmodell eine 3D-Karte erstellt werden, die dann für die Navigation sowohl im Fußgänger- als auch im Fahrzeugbereich verwendet werden kann. Diese vereinfacht dem Benutzer die Wiedererkennung seiner Umgebung in der virtuellen Karte.

Gerade im Bereich der Navigation ist eine flächendeckende Erfassung sowie die ständige Aktualisierung der Daten nötig. Dieser Aufwand kann mit manueller Modellierung nicht geleistet werden, da es zu zeit- und kostenintensiv ist. Deshalb ist für Anwendungen in diesem Bereich eine automatische Erstellung der Modelle notwendig. Heutzutage wird die automatische Generierung von Stadtmodellen außerdem immer wichtiger, da die Nachfrage ständig steigt. Die Zahl an Anwendungen wächst und die Daten werden von immer größeren Teilen der Bevölkerung genutzt.

Außerdem unterscheiden sich die Anforderungen an den Detailgrad eines Modells für Visualisierungsanwendungen stark von denen für die Simulation. Für die Visualisierung kann nicht die grobe Geometrie aus der Simulation verwendet werden. Es werden auch Details, wie das Aussehen der Fassade mit Fenstern und Türen benötigt.

Die Fassadendetails können gut über eine strukturelle Beschreibung modelliert werden. Die Anordnung der Fassadenelemente verhält sich nach bestimmten Regeln. So sind Fenster oft in Zeilen und Spalten angeordnet und es treten Symmetrien und Wiederholungen bestimmter Bereiche auf. Diese Regelmäßigkeiten in der Struktur können als Vorwissen für den Rekonstruktionsprozess verwendet werden. Außerdem kann eine reichhaltige strukturelle Beschreibung sinnvoll für die Generalisierung der Gebäudegeometrie genutzt werden, wenn sie ermöglicht, die Objektrepräsentation automatisch zu verändern.

1.2 Zielsetzung

Ziel der Arbeit ist es, ein Verfahren zu entwerfen, mit dem automatisch detaillierte Fassaden für 3D-Gebäudemodelle erstellt werden können. Die schnelle und kostengünstige Generierung von Gebäudemodellen ermöglicht es, großräumige Stadtmodelle für die oben genannten Anwendungen zu erzeugen und diese aktuell zu halten. Gleichzeitig soll das Verfahren aber so allgemein gehalten sein, dass die Rekonstruktion auf andere künstliche Objekte übertragen werden kann.

Der Fokus der Arbeit wird auf die strukturelle Modellierung von Fassadenelementen, wie Türen und Fenstern gelegt, die im Gegensatz zu vielen bisherigen Modellen nicht durch Texturieren dargestellt werden sollen. Dazu wird die Struktur von Fassaden anhand von Fassadenbildern untersucht. Hierbei zeigt sich z.B., dass Fenster meist in Reihen und Spalten angeordnet sind. Aus solchen Informationen wird Modellwissen erarbeitet, das in einer formalen Grammatik formuliert wird.

Die hier entworfene Grammatik beinhaltet viele Strukturinformationen über Fassaden und geht über die in anderen Arbeiten verwendete Aufteilung durch horizontale und vertikale Schnitte über die gesamte Fassade hinaus (Alegre und Dallaert, 2004). Das Wissen über die Struktur von Fassaden hilft auch dabei, nicht sichtbare Bereiche der Fassade zu rekonstruieren. Werden Teile der Fassade von einem Baum verdeckt, so kann mithilfe der sichtbaren Strukturen eine Vorhersage über den verdeckten Bereich getroffen werden.

Die hierarchische Modellierung der Fassadenstruktur liefert ebenfalls wichtige Informationen für die Level of Detail (LoD) Darstellung. So können aus einem Modell mehrere Darstellungen in verschiedenen LoD abgeleitet werden. Diese können dann für die Visualisierung in unterschiedlichen Maßstäben verwendet werden.

Der Ableitungsprozess wird mithilfe eines stochastischen Prozesses, dem reversible jump Markov Chain Monte Carlo (rjMCMC), automatisiert. Dieses Verfahren wurde schon für die Modellierung von Gebäuden verwendet (Dick u. a., 2004). Hierbei findet jedoch keine Rekonstruktion anhand von gemessenen Daten statt, sondern es werden Gebäude nach bestimmten Regeln zusammengesetzt, die mithilfe von Expertenwissen über Fassaden entwickelt wurden. In dieser Arbeit werden mit rjMCMC mögliche Ableitungsbäume exploriert und die besten – die mit der größten Wahrscheinlichkeit – ausgewählt. Dazu werden Veränderungen in der Fassade vorgeschlagen und anhand einer Akzeptanzwahrscheinlichkeit, basierend auf den Messdaten von der Fassade, akzeptiert oder verworfen.

Gemessene Eingangsdaten bestehen hier aus einer Punktwolke und einem rektifizierten Bild, die beide mit einem terrestrischen Laserscanner mit Kamera erfasst wurden. Die Verwendung der zwei Datenquellen erlaubt es, verschiedene Strukturen zu erkennen, die jeweils nur in einer der Datenquellen zu erkennen ist.

Um die Akzeptanzwahrscheinlichkeit für den rjMCMC Prozess zu bestimmen, wird untersucht, wie gut Modell und Daten zusammenpassen. Hierbei ist es wichtig, die Balance zwischen der Kompatibilität von Daten und Modell und der Komplexität des Modells zu finden. Um dies zu erreichen, wird eine auf Minimum Description Length (MDL) basierende Bewertung entwickelt, die beide Aspekte berücksichtigt und somit ein geeignetes Modell auswählt.

1.3 Gliederung

In dieser Arbeit werden zunächst verwandte Arbeiten vorgestellt und Grundlagen vermittelt, die für die Entwicklungen des Rekonstruktionsverfahrens benötigt werden. Kapitel 2 stellt Arbeiten zur Objekterkennung vor. Es geht zunächst auf die Rekonstruktion von Objekten aus luftgestützten Daten ein. Hierbei liegt der Schwerpunkt auf wissensbasierten Methoden. Dann werden Arbeiten zur Fassadenrekonstruktion vorgestellt. Abschließend werden zwei Ansätze zur Strukturerkennung beschrieben.

Kapitel 3 gibt eine Einführung in die Theorie der formalen Grammatiken. Es werden die Grammatiken der Chomsky-Hierarchie und Lindenmayer-Systeme definiert. Anschließend werden beispielhaft Möglichkeiten besprochen, wie diese Grammatiken zur Modellierung verwendet werden können. Außerdem werden Ansätze vorgestellt, die das Konzept der formalen Grammatik auf höhere Dimensionen erweitern.

In Kapitel 4 werden die Grundlagen der Monte Carlo Simulation erläutert. Zunächst werden einfache Simulationsverfahren vorgestellt. Um die Markov Chain Monte Carlo Verfahren zu erklären, werden zuerst Grundlagen der Markov-Ketten definiert. Hiermit können dann die Algorithmen von Metropolis und Metropolis-Hastings beschrieben werden. Darauf aufbauend werden das reversible jump Markov Chain Monte Carlo Verfahren und das Simulated Annealing erklärt.

Der zweite Teil dieser Arbeit beschreibt das entwickelte Rekonstruktionsverfahren. In Kapitel 5 wird die Grammatik vorgestellt, mit der die Fassaden modelliert werden. Neben der Definition der Grammatik wird gezeigt, welche Fassaden modellierbar sind. Außerdem wird ein Programm vorgestellt, mit dem sich Fassaden manuell nach den Regeln der Grammatik modellieren lassen. Kapitel 6 beschreibt, wie die Ableitung der Fassaden mithilfe von reversible jump Markov Chain Monte Carlo automatisiert werden kann. Es werden die einzelnen Schritte, wie die Integration von Grammatiken und rjMCMC, die Bestimmung von Vorschlags- und Akzeptanzwahrscheinlichkeiten und die Anwendung von Simulated Annealing behandelt.

In Kapitel 7 werden die Ergebnisse der Rekonstruktion evaluiert. Das Verfahren wird an manuell segmentierten Daten und an Daten, die mit einem Laserscanner aufgenommen wurden, getestet. Zusätzlich wird untersucht, welchen Einfluss Verdeckung durch Vegetation auf die Rekonstruktion hat. Kapitel 8 beschreibt die technische Realisierung des Programms. Es wird die Klassenstruktur und die Interaktion der Objekte beschrieben. Abschließend gibt Kapitel 9 eine Zusammenfassung und einen Ausblick.

2 Methoden zur Objekterkennung

Die automatische Erkennung von Objekten interessiert die Forschung schon seit vielen Jahren. Die ersten Arbeiten zu dem Thema sind in den 1970er Jahren erschienen. Die Problemstellung wird sowohl im Bereich der Fernerkundung als auch der Computer Vision behandelt.

Es gibt zwei verschiedene Wege, an die Rekonstruktion von Objekten heranzugehen. Bei Bottom-up-Verfahren, oder auch datengetriebenen Verfahren, erfolgt die Rekonstruktion ausgehend von den Daten. Es werden die Eingangsdaten bearbeitet (z.B. segmentiert oder Kanten extrahiert). Dann findet eine Zuordnung zwischen den extrahierten Merkmalen und den zur Verfügung stehenden Modellen statt. Die Top-down-Verfahren, oder modellgetriebenen Verfahren, arbeiten anders herum. Zuerst wird das Aussehen des Objekts in Form eines Modells definiert und dann wird mit dieser Information die Extraktion gestartet.

Die Anforderungen an die rekonstruierten Modelle können je nach Anwendung sehr unterschiedlich sein. Das Open Geospatial Consortium hat in der City Geographic Markup Language (CityGML) fünf verschiedene LoD definiert, um diese eindeutig zu unterscheiden (Gröger u. a., 2007). Die gröbste Stufe, das LoD0, wird für die Visualisierung von Landschaften verwendet. Es besteht aus einem digitalen Geländemodell, das zusätzlich von einem Luftbild oder einer Karte überlagert sein kann. Die erste Stufe um Städte darzustellen bietet das LoD1. Hier werden Gebäude als Klötzchenmodelle mit flacher Dachstruktur dargestellt. Die nächste Stufe, das LoD2, beinhaltet verschiedene Dachstrukturen und evtl. Vegetation. Detaillierte Dach- und Fassadenstrukturen kommen im LoD3 hinzu. Hier können die Objekte mit hoch aufgelösten Texturen versehen werden und zusätzlich können detailgenaue Vegetations- und Verkehrsobjekte dargestellt werden. Die genaueste Stufe, LoD4, fügt zu den Informationen aus LoD3 noch die Innenräume von Gebäuden hinzu.

Für die unterschiedlichen LoDs werden unterschiedlich detaillierte Daten benötigt. Für ein Stadtmodell im LoD1 genügt eine luftgestützte Punktwolke mit geringer Punktdichte, wohingegen für ein Stadtmodell im LoD3 Bilder oder terrestrische Scans der Fassaden benötigt werden. Die im Folgenden vorgestellten Verfahren sind meist im Bereich LoD1 bis LoD2 anzusiedeln.

Obwohl das Erkennen von Objekten für den Menschen einfach erscheint, stellt sich diese Aufgabe für den Computer als schwierig dar. Für den Menschen ist es einfach in einem Luftbild Häuser zu erkennen oder die Fenster in einer Fassade auszumachen. Dieses Wissen, das wir für solche Aufgabenstellungen unbewusst benutzen, muss dem Verfahren für den Computer mitgegeben werden. Dazu werden verschiedene wissensbasierte Methoden aus dem Bereich der Künstlichen Intelligenz verwendet. Diese Methoden und Arbeiten, die sie verwenden, werden in diesem Abschnitt vorgestellt. Eine Einführung in die Wissensrepräsentation und das Wissensmanagement ist in Bodendorf (2006) und Reimer (1991) zu finden. Crevier und Lepage (1997) und Sowmya und Trinder (2000) geben einen Überblick über wissensbasierte Systeme zur Erkennung künstlicher Objekte.

Anschließend werden weitere Verfahren zur Objekterkennung vorgestellt, die auf Heuristiken basieren. Viele der vorgestellten Arbeiten verwenden als Daten Luftbilder oder luftgestützte Scandaten und die zu rekonstruierenden Objekte sind meist Gebäude. Dies liegt daran, dass diese Daten schon länger zur Verfügung stehen als terrestrische Scans, die in dieser Arbeit als Datengrundlage dienen. Außerdem liegen luftgestützte Daten für größere Bereiche vor, da die terrestrische Erfassung noch sehr zeitaufwendig ist. Die Zeit, die zur Erfassung eines großen Gebietes benötigt wird, ist noch zu groß, als dass man das gesamte Gebiet einer Großstadt terrestrisch erfassen würde. Dieses Problem wird aber nicht bestehen bleiben, da es schon jetzt Arbeiten gibt, die sich mit mobilem Laserscanning befassen (Hunter u. a., 2006). Dabei wird ein Laserscanner auf einem Fahrzeug montiert. Die Messung erfolgt entweder im Stop-and-Go-Verfahren oder während der Fahrt. Einige Arbeiten zur Fassadenrekonstruktion aus terrestrischen Scandaten oder Bilddaten werden am Ende dieses Kapitels vorgestellt. Abschließend werden einige allgemeinere Arbeiten zur Strukturerkennung beschrieben.

2.1 Objekterkennung in luftgestützten Daten

2.1.1 Wissensbasierte Ansätze zur Objekterkennung

Mathematische Logik

Eine Möglichkeit Wissen zu repräsentieren ist die Mathematische Logik. Liegen die Daten vollständig vor, so bietet die Logik den Vorteil, dass die Bedingungen für Objekte und deren Relationen genau formuliert werden können. Probleme treten allerdings dann auf, wenn Informationen über Objekte unsicher sind.

Die Logik stellt ein System dar, in dem Aussagen immer wahr oder falsch sind. Aus Aussagen können dann mithilfe von Regeln weitere Schlussfolgerungen gezogen werden. Eine Regel der Form „Wenn A, dann B“ liefert hauptsächlich auf zwei Arten neue Aussagen. Die erste ist der *Modus Ponens*, hierbei wird die Prämisse A untersucht. Ist A wahr, so ist B auch wahr. Im zweiten Fall, dem *Modus Tollens*, wird die Konklusion untersucht. Ist diese falsch, so ist auch die Prämisse falsch. Also B ist falsch impliziert, A ist falsch. Mit Modus Ponens wird also durch Vorwärtsverkettung auf neue Fakten geschlossen, im Modus Tollens durch Rückwärtsverkettung.

Reiter und Mackworth (1989) stellen ein logisches System zur Interpretation von Karten auf. Hierzu stellen sie Axiome getrennt für den Bildbereich und den Szenenbereich auf. Darin werden Objekte definiert und Beziehungen zwischen ihnen beschrieben. So gibt es im Bildbereich Bildobjekte, die entweder eine Kette oder eine Region sein können. Dies wird durch folgende Axiome ausgedrückt:

$$\forall x \ \textit{image-object}(x) \Leftrightarrow chain(x) \vee region(x)$$

$$\forall x \ \neg(chain(x) \wedge region(x))$$

Zu den Objekten sind Relationen wie T-Kreuzung, Kreuzung, Kette begrenzt Region, Region ist das Innere von Kette definiert. Instanzen von Ketten oder Regionen werden durch Konstanten gegeben. Zu den Relationen sind zusätzlich Kohärenzanforderungen gegeben, die den Typ festlegen. Sie sagen aus, dass z.B. die Relation T-Kreuzung zwischen Ketten definiert ist.

$$\forall x, y \ tee(x, y) \Rightarrow chain(x) \wedge chain(y)$$

Für die anderen Relationen sind ähnliche Bedingungen formuliert.

Im Szene-Bereich werden genauso die möglichen Objekte definiert. Sie unterteilen sich in die linienhaften Objekte Straße, Fluss und Küste und die flächenhaften Objekte Land und Wasser. Auch für diese Objekte werden Relationen und Typfestlegungen definiert. Hier kann jetzt auch Wissen über die Objektart mit einfließen. Z.B. kreuzen sich keine zwei Flüsse, eine Küstenlinie bildet immer eine geschlossene Linie und ein Fluss darf keine geschlossene Linie bilden. Zusätzlich gibt es die Bedingung, dass ein Objekt entweder Bild- oder Szenenobjekt ist und dass jedes Bildobjekt eindeutig ein Szenenobjekt abbildet und jedes Szenenobjekt von einem eindeutigen Bildobjekt abgebildet wird.

Zur Interpretation einer Skizze werden alle bekannten Aussagen aufgeschrieben und vereinfacht. Das Problem lässt sich dann mit Constraint-Solvern lösen. Zusatzwissen, wie „Es gibt in der Szene einen Fluss, der im Meer mündet“, kann über weitere Axiome in das System gebracht werden. Dies muss so formuliert werden, dass es zu den vereinfachten Axiomen hinzugefügt werden kann.

An das System können auch Anfragen der Art „Gibt es eine T-Kreuzung in der Szene?“ gestellt werden. Diese kann nicht immer mit wahr oder falsch beantwortet werden, da sie auch nur für einige Interpretationen wahr sein kann und für andere nicht. Ist sie für alle Interpretationen wahr oder falsch, so kann die Aussage als wahr oder falsch bezeichnet werden, sonst wird sie als möglich bezeichnet.

Nicht eindeutige oder unvollständige Daten führen bei diesem Ansatz zu Problemen. Eine Linie, die nicht exakt auf einer anderen endet, kann sowohl eine T-Kreuzung als auch eine echte Kreuzung sein. Diese Fehler treten durch falsche Segmentierung oder unsauberes Zeichnen auf. Ein solcher Fall verstößt gegen das Axiom, dass zwei Ketten sich entweder in einer Kreuzung oder in einer T-Kreuzung treffen. Das gleiche Problem der Uneindeutigkeit tritt auf, wenn nicht klar ist, ob zwei Linien verbunden sind oder ob es zwei verschiedene Linien sind. Das wirkt sich auch auf die Regionen aus, die sie evtl. trennen oder nicht.

Wird ein Fluss von einer Brücke überdeckt, so wird dieser dadurch in zwei Teile geteilt. Soll er trotzdem dem gleichen Szenenobjekt zugeordnet werden, geht dadurch die eindeutige Zuordnung verloren. Weitere Probleme treten auf, wenn Objekte gar nicht in der Karte auftauchen, weil sie außerhalb des Gebietes liegen, zu klein

sind oder nicht zum Thema der Karte passen. So wandelt sich der Vorteil der genauen Formulierung bei nicht perfekten realen Daten schnell in einen Nachteil, da diese nicht widerspruchslos modelliert werden können.

Regelbasierte Systeme und Produktionssysteme

Die Anordnung von künstlichen Objekten kann auch in regelbasierten Systemen bzw. Produktionssystemen modelliert werden. Ein Produktionssystem besteht aus einer Datenbank, in der vorhandenes Wissen gespeichert ist, einer Menge von Produktionsregeln und einer Kontrolleinheit. Die Produktionsregeln sind von der Form „Wenn eine Bedingung A erfüllt ist, dann kann eine Aktion B ausgeführt werden“ und bilden somit ein Ersetzungssystem. Dies kann auch durch eine formale Grammatik ausgedrückt werden (siehe Kapitel 3). Das allgemeine Zusammenspiel der Produktionen kann in einem Produktionsnetz dargestellt werden. Eine Kontrolleinheit steuert das System, indem sie entscheidet, welche der möglichen Produktionen als Nächstes angewandt wird.

Strat und Fischler (1991) stellt ein Verfahren zur Interpretation komplexer Szenen vor. Informationen werden aus einem oder mehreren Intensitäts-, Farb- oder Entfernungsbildern abgeleitet und zusätzlich können noch andere Modalitäten verwendet werden. Das Ergebnis ist ein 3D-Modell der Szene, in dem die einzelnen Objekte beschriftet sind.

Früher wurde in der Computer Vision häufig angenommen, dass alle Objekte von Interesse durch eine relativ kleine Zahl an Modellen dargestellt werden können und dass alle Objekte charakteristische, lokal messbare Eigenschaften haben. Dies trifft auf die meisten komplexen Szenen, wie z.B. ein in natürlicher Umgebung aufgenommenes Foto, nicht zu. Deshalb wird bei Strat und Fischler (1991) der Kontext der Objekte berücksichtigt. Das Prinzip beruht darauf, viele einfache Methoden zu entwickeln, die Bilder analysieren und die Teilergebnisse wieder zusammenzuführen. Um den Ablauf der vielen einfachen Methoden zu steuern, verwenden Strat und Fischler (1991) ein Regelsystem, in dem Wissen über die Szene und den Ablauf der einzelnen Prozesse modelliert werden. Diese Regeln sind als so genannte Kontext-Sets modelliert. Sie bilden eine Menge von Kontext-Elementen, die, wenn alle erfüllt sind, eine Operation auf dem Bild hervorrufen.

Der Prozess teilt sich in vier Teile auf. Dies sind die Erzeugung von Kandidaten, der Kandidaten-Vergleich, die Cliquenbildung und die Wahl der besten Clique. Bei der Kandidaten-Erzeugung werden die vielen einfachen Methoden zur Erkennung von Objekten angewandt, die jeweils einen kleinen Teil des Bildes interpretieren und dabei Kandidaten für die Kontext-Elemente erzeugen. Ein Kandidat ist dabei eine Bildeigenschaft, die möglicherweise Instanz einer Klasse ist. Im nächsten Schritt werden alle Kandidaten bewertet, wobei es verschiedene Bewertungsmaßstäbe gibt, die parallel verwendet werden. In der Reihenfolge der Bewertung werden die Kandidaten zu Cliquen zusammengefasst, wobei die Kandidaten einer Clique sich nicht widersprechen dürfen. Das resultierende Modell ergibt sich als beste aller Cliquen.

Die Zahl möglicher Cliquen kann sehr groß werden. Deshalb ist die Sortierung der Kandidaten wichtig. Durch sie werden die besseren Cliquen zuerst gebildet und der Prozess kann abgebrochen werden, bevor alle Cliquen behandelt wurden.

Stilla und Michaelsen (1997) stellen eine Methode zur Erkennung künstlicher Objekte vor, die auf Produktionsnetzen basiert. In der Arbeit werden zwei Anwendungen gezeigt. Die erste dient der 3D-Rekonstruktion aus Luftbildern und in der zweiten werden komplexe Gebäude aus Vektordaten extrahiert.

In einem Produktions-Netz gibt es zwei Arten von Knoten. Die eine Art sind die Konzepte und die zweite die Produktionen. Kanten verlaufen immer zwischen Knoten verschiedener Art. Eine Kante von einer Produktion zu einem Konzept stellt eine Generierungsfunktion dar. Eine Kante von einem Konzept zu einer Produktion tritt immer dann auf, wenn das Konzept in einer Eingabekonfiguration der Produktion vorkommt. Das Produktions-Netz liegt zwischen den später beschriebenen semantischen Netzen und den prozeduralen Petri-Netzen.

Das für die Rekonstruktion entwickelte Produktions-Netz besteht aus zwei Schichten. Die 2D-Schicht analysiert die im Bild erkannten Liniensegmente und bildet die Strukturen Streifen, Winkel, U-Strukturen und Parallelogramme. Diese werden in der 3D-Schicht analysiert und zu Dächern, Häusern und Hausreihen zusammengesetzt. Hier wird das Wissen verwendet, dass Häuser oft in einer Reihe stehen und ähnliche Größe und ähnlichen Abstand haben. In den Produktionen werden vor allem Eigenschaften wie Kollinearität, Parallelität und Periodizität verwendet.

Die zweite Anwendung ist die Analyse von Karten. Diese ist wichtig für die Erweiterung von Karten. In diesem Fall geht man von einer korrekten Karte aus, die mit weiteren Informationen angereichert werden soll. Die

Information, die in der Kartenanalyse gewonnen wird, kann dann als Vorwissen für die Bildanalyse genutzt werden. Ebenso wird die Kartenanalyse für die Erkennung von Veränderungen benötigt.

In dieser Arbeit wurde ein Produktions-Netz zur Erkennung komplexer Gebäude entworfen. Hierin werden zunächst Gebäudeteile gefunden. Hierzu werden aus Linien zuerst Polygone und dann geschlossene Polygone gebildet, die die Kontur eines Gebäudeteils darstellen. Eingeschlossene Konturen werden bestimmt. Ein Gebäude entsteht aus inneren und äußeren Konturen. Die einzelnen Gebäude werden dann zu komplexen Gebäuden zusammen gesetzt.

Produktionssysteme werden auch in anderen wissensbasierten Systemen verwendet. So spielen sie auch in verschiedenen Expertensystemen (siehe unten) eine Rolle (Hayes-Roth u. a., 1983).

Blackboard-Systeme

Blackboard-Systeme werden von Velthuijsen (1992) mit der folgenden Metapher beschrieben: Viele Spezialisten aus unterschiedlichen Fachgebieten stehen um eine Tafel herum und wollen ein gemeinsames Problem lösen. Jeder Spezialist schreibt seine Zwischenergebnisse und Hypothesen auf die Tafel. Mit diesen Informationen können andere Spezialisten weiterarbeiten und dann ebenfalls ihre Ergebnisse an die Tafel schreiben. Der Prozess läuft so lange, bis das Problem gelöst ist oder kein Spezialist weitere Schlüsse aus den Informationen auf der Tafel ziehen kann.

Ein Blackboard-System besteht aus verschiedenen Wissensquellen, die den Spezialisten entsprechen, einer Datenbank, die als Blackboard dient, und einer Kontrolleinheit. Die Wissensquellen arbeiten unabhängig voneinander und kommunizieren ausschließlich über das Blackboard.

Stilla (1995) entwickelte ein auf Karten basierendes System zur Interpretation von Luftbildern. Dazu wird ein Produktionssystem zur Kartenanalyse im Vorfeld und ein Blackboard-System für die Luftbildinterpretation verwendet. Die Kartenanalyse dient dazu, das Wissen für die Bildanalyse angemessen zur Verfügung zu stellen. Die Produktionen in diesem System bestehen aus zwei Eingabe-Objekten X, Y, einem Prädikat $\circledcirc$ und einem Ausgabe-Objekt Z. Erfüllen X und Y das Prädikat $\circledcirc$, so wird ein Objekt Z erzeugt. Mit einem Netz aus acht Produktionen können Objekte der Klasse Straße analysiert werden. Einige Beispiel-Produktionen lauten folgendermaßen: „Sind zwei Linien kollinear und verlängerbar, so werden sie zu einer Linie zusammengesetzt“. „Mündet eine Straße in eine Kreuzung, so werden sie in einem Kreuzungselement zusammengefasst“. „Haben zwei Straßen einen bestimmten Winkel zueinander, so bilden sie eine Kreuzung“.

Für die Bildanalyse werden in einem Vorverarbeitungsschritt Liniensegmente extrahiert. Diese werden zusammen mit ihrer Bewertung in ein Blackboard-System eingefügt. Für verschiedene Objektarten werden Objektmodelle entworfen, die auch Modellparameter, Toleranzen und Bewertungsparameter enthalten können. Zur Untersuchung einer Hypothese werden Objekte betrachtet, die dem Modell mit einer gewissen Toleranz entsprechen. Dadurch kann der Suchraum erheblich eingeschränkt werden. So wird für eine Hypothese *part_of*_LongLine nach Liniensegmenten gesucht, die eine ähnliche Steigung wie die Ausgangslinie haben und die in der Nähe von deren Endpunkten liegen. Für *part_of*_Stripe werden Linien mit ähnlicher Orientierung in einem gewissen Abstand gesucht.

Zur Verifikation der Hypothese wird im Blackboard nach einem Objekt gesucht, das diese bestätigt. Ist das der Fall, wird ein neues Objekt generiert und eine Bewertung berechnet, die aussagt, wie gut es zum Modell passt. Dies hängt z.B. von der Länge einer Linie oder der Abweichung der Orientierung zweier Linien ab. Die Bewertung geht von sehr schlecht in fünf Stufen bis zu sehr gut. Zusammengesetzte Objekte werden nur nach dem Modell bewertet und die Bewertung ihrer Bestandteile spielt keine Rolle mehr. Das neu erzeugte Objekt wird abschließend in das Blackboard geschrieben. Anschließend wird eine neue Hypothese aufgestellt.

Das Wissen aus der Kartenanalyse wird jetzt verwendet, um die einzelnen Objekte zu priorisieren. Strukturen im Bild, die korrespondierende Strukturen in der Karte haben, erhalten eine höhere Priorität als Strukturen ohne Korrespondenzen. Dadurch verändert sich das Ergebnis der Bildanalyse nicht, aber die Rechenzeit wird deutlich verringert.

Expertensysteme

Expertensysteme können in zwei Arten unterteilt werden: die deterministischen und die stochastischen Expertensysteme. Deterministische Expertensysteme sind oft regelbasiert und der Lösungsweg wird aus den Regeln

abgeleitet. Für stochastische Expertensysteme kann die im folgenden Abschnitt beschriebene Fuzzy-Logik verwendet werden. Eine weitere Möglichkeit ist, die Schlussfolgerungen nach Wahrscheinlichkeiten zu treffen. Diese Systeme nennt man dann probabilistische Expertensysteme.

Im Wesentlichen besteht ein Expertensystem aus einer Wissensbasis und einem Steuerungssystem. Die Wissensbasis basiert auf der Aussagenlogik. Mithilfe dieser kann man Produktionssysteme bauen, die dann wiederum die Basis regelbasierter Systeme bilden. Die Wissensbasis besteht aus einer Menge von Regeln. Diese setzen sich aus Prämisse und Konklusion zusammen. Eine Regel sieht also folgendermaßen aus: Wenn die Prämisse erfüllt ist, dann folgt die Konklusion.

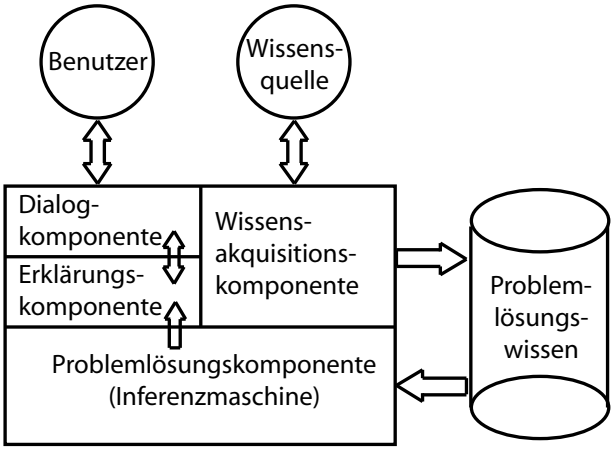


Abbildung 2.1: Komponenten eines Expertensystems (nach Reimer, 1991).

Abbildung 2.1 zeigt die wesentlichen Komponenten eines Expertensystems. Das Steuerungssystem besteht aus einer Wissensakquisitionskomponente, einer Dialog-Komponente, einer Erklärungs-Komponente und einer Problemlösungs-Komponente. Die Wissensakquisitionskomponente dient zur Eingabe von Wissen durch Experten, die Dialog-Komponente zur Interaktion mit dem Benutzer. Über sie gelangt fallspezifisches Wissen in das System. Die Erklärungs-Komponente kann Entscheidungen begründen und hilft den Experten bei der Fehlerbehebung.

Die Problemlösungs-Komponente leitet aus der Wissensbasis und dem fallspezifischen Wissen eine Lösung ab. Das Wissen in der Wissensbasis ist in Form von Fakten und Regeln abgelegt. Hieraus können mit *Modus Ponens* und *Modus Tollens* weitere Fakten geschlussfolgert werden.

Weitere Unterschiede im Vorgehen gibt es bei der Reihenfolge, in der die Regeln ausgewählt werden. Bei einer Breitensuche werden alle Regeln, deren Prämisse (bei Vorwärtsverkettung) oder Konklusion (bei Rückwärtsverkettung) mit den Fakten übereinstimmen, angewandt und erzeugen neue Fakten. Im nächsten Schritt werden die Regeln angewandt, die durch die neu erzeugten Fakten verwendet werden können. Auf diese Weise werden viele Teilbäume bearbeitet, die später gar nicht zum Ergebnis beitragen. Deshalb sollte dieses Verfahren nur eingesetzt werden, wenn der Baum nicht zu stark verzweigt ist.

Im Gegensatz hierzu wird bei der Tiefensuche die erste Regel, deren Prämisse bzw. Konklusion den Fakten entspricht, angewandt. Die nächste Regel wird hier auf Basis der neuen Fakten gewählt. Dieses Vorgehen ist besonders gut geeignet, wenn ein Dialog mit dem Nutzer stattfindet. So müssen Informationen erst dann eingegeben werden, wenn sie tatsächlich gebraucht werden.

Goodenough u. a. (1987) stellen eine Anwendung der Fernerkundungs-Expertensystem-Shell RESHELL zur Erstellung und Aktualisierung von Karten mithilfe von Luftbildem vor. In einem Vorverarbeitungsschritt werden die Karte und die Bilddaten in die gleiche Auflösung und eine gleiche symbolische Darstellung gebracht. Anschließend werden gleiche Strukturen in Form von Segmenten gesucht und Unterschiede ausgegeben. Das System beschränkt sich auf die Erkennung von Differenzen, die erkannten Fehler müssen anschließend von einem Operateur behoben werden. Hierbei werden Fehler ganz unterschiedlicher Art gefunden. Es treten sowohl zufällige als auch systematische Fehler auf, die global oder lokal begrenzt sein können.

Die symbolische Darstellung beider Datentypen beinhaltet die Lage und Größe eines Objekts sowie die Form und ein umgebendes Rechteck. Die Spektral-Attribute im Bild sind Mittelwert, Minimum und Maximum des Grauwerts eines Segments jeweils für einen Kanal X (MEAN_CH_X, MAX_CH_X, MIN_CH_X). Klassifizierungsregeln

können dann z.B. folgendermaßen aussehen:

$$MEAN\_CH\_4 > MEAN\_CH\_2 \,\&\, MEAN\_CH\_4 > MEAN\_CH\_1 \Rightarrow CLASS = LAND - COVER.$$

Das in diesem Ansatz verwendete Expertensystem beinhaltet Meta-Regeln, Objekt-Regeln und Objekt-Werte. Meta-Regeln und Objekt-Regeln bilden die Wissensbasis und stellen somit das Langzeitgedächtnis dar, wobei Meta-Regeln sich um den allgemeinen Ablauf kümmern, während Objekt-Regeln Details des Verfahrens festlegen. Das Kurzzeitgedächtnis stellt die Informationen aus Bild und Karte dar. Das Expertensystem ist hierarchisch gegliedert und teilt das Problem auf drei verschiedene Experten auf. Diese haben jeweils nur Zugriff auf ihren Teil der Wissensbasis. Der High-Level-Experte ist für die Eingabe, die Bearbeitung und die Ausgabe zuständig. Der Klassen-Experte liefert die beste Klasse für die weitere Kongruenzauswertung. Der Kategorie-Experte liefert zu dieser Klasse die beste Kategorie.

Die Liste der Kategorien wurde vom Canada Council on Surveys and Mapping übernommen und die Regeln in Gesprächen mit Fotoauswertern entwickelt. Im Klassen-Experten gibt es 25 Meta-Regeln und 30 Objekt-Regeln und im Kategorie-Experten 30 Objekt-Regeln.

Fuzzy-Logik

Die Fuzzy-Logik erweitert das Konzept der Boolschen Logik indem sie mehr Werte als nur wahr (1) und falsch (0) zulässt. Die Werte werden in der Fuzzy-Logik nicht mehr numerisch, sondern linguistisch belegt. Das heißt, sie haben Ausprägungen wie „gering“ oder „mittel“. Jeder Variablen wird eine Zugehörigkeitsfunktion (ZG), auch Fuzzy-Set genannt, zugeordnet, die besagt, zu welchem Prozentsatz die Variable zu einer Ausprägung gehört. Hierzu werden häufig fünf oder sieben verschiedene Ausprägungen verwendet. Der Zugehörigkeitsgrad kann verschiedene symmetrische oder asymmetrische Formen annehmen. Z.B. können die Formen Dreieck, Trapez oder Glocke aus Abbildung 2.2 verwendet werden.

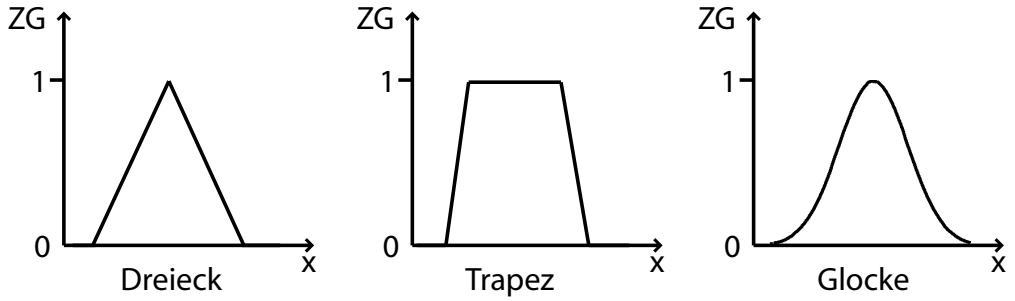


Abbildung 2.2: Zugehörigkeitsfunktionen in den Formen Dreieck, Trapez und Glocke.

Verwendet man Fuzzy-Logik für oben beschriebene Expertensysteme, so erhält man Fuzzy-Expertensysteme. Hierbei wird wieder ein Regelsystem mit Regeln „Wenn Prämisse, dann Konklusion“ aufgestellt, wobei die Variablen jetzt linguistische Werte annehmen. Die Schlussfolgerung in einem Fuzzy-Expertensystem setzt sich aus den Schritten Fuzzifizierung, Inferenz und Defuzzifizierung zusammen.

Die Fuzzifizierung übersetzt die numerischen Eingaben in Ausprägungen der linguistischen Variablen. Für die Inferenz werden die Vorbedingungen der Regeln ausgewertet. Der Grad, zu dem die Schlussfolgerung wahr ist, ergibt sich aus dem Grad, zu dem die Bedingung wahr ist. Bei mehreren Bedingungen werden diese aggregiert, d.h. bei UND-Verknüpfungen wird das Minimum und bei ODER-Verknüpfungen das Maximum verwendet. Liefern Regeln verschiedene Zugehörigkeitsgrade für eine Ausprägung, so werden diese akkumuliert, also das Maximum verwendet.

Bei der Defuzzifizierung müssen ein oder mehrere Ausprägungen der linguistischen Variablen, die mit Zugehörigkeitsgraden versehen sind, zu einem numerischen Wert kombiniert werden. Dazu werden die Fuzzy-Sets zunächst entweder am ermittelten Zugehörigkeitsgrad abgeschnitten oder mit diesem Faktor multipliziert. Zur Abbildung auf einen Wert wird der Schwerpunkt der Fläche der resultierenden Fuzzy-Sets berechnet.

Brzank und Heipke (2007) verwenden die Fuzzy-Logik für die Klassifikation von Wasser und Watt aus luftgestützten Laserdaten. Hierbei wird jeder Flugstreifen einzeln bearbeitet um die tiedebedingte unterschiedliche Wasserhöhe in den einzelnen Streifen zu umgehen. Für jeden Punkt wird dann ein Zugehörigkeitswert zur Klasse

Wasser berechnet. Dieser basiert auf den Parametern Höhe, Intensität und Punktdichte, die gewichtet werden können. Die Zugehörigkeit berechnet sich nach folgender Formel:

$$\mu(h,i,p,\alpha) = \frac{\delta_H \mu_H(h) + \delta_I(\alpha)\mu_I(i,\alpha) + \delta_P(\alpha)\mu_P(p,\alpha))}{\delta_H + \delta_I(\alpha) + \delta_P(\alpha))},$$

wobei h, i, p, α Höhe, Intensität, Punktdichte und Auslenkungswinkel sind, $\delta_H, \delta_I(\alpha), \delta_P(\alpha)$ die entsprechenden Gewichte zu den Werten und $\delta_H\mu_H(h), \mu_I(i,\alpha), \mu_P(p,\alpha)$ die Zugehörigkeitswerte zur Klasse Wasser bezüglich der einzelnen Parameter.

Für den Klassifizierungsprozess müssen die folgende Parameter bestimmt werden: Jeweils zwei Grenzwerte, die die Werte der Parameter Höhe, Intensität und Punktdichte in Werte der Fuzzy-Zugehörigkeit überführen, und Gewichte für die drei Zugehörigkeitsfunktionen. Hierbei ist das Gewicht für die Höhe konstant, die Gewichte für Intensität und Punktdichte hängen von dem Auslenkungswinkel ab. Als Letztes müssen noch Grenzwerte für die Überführung des Fuzzy-Zugehörigkeitswert in die Klassen Wasser oder Watt bestimmt werden.

Dazu werden zuerst Trainingsgebiete für beide Klassen bestimmt. Für die Zugehörigkeitsfunktion werden obere und untere Grenze bestimmt, die ihren Bereich begrenzen. Der untere Wert ist der Mittelwert der Klasse Watt, hier hat die Funktion den Wert 0. Den oberen Grenzwert bildet der Mittelwert der Klasse Wasser mit dem Wert 1. Zwischen den beiden Punkten ist die Zugehörigkeitsfunktion eine Gerade.

Die Gewichte werden zwischen 0 und 1 gesetzt. Es werden für beide Klassen Mittelwert und Standardabweichung berechnet und die Verteilung als Normalverteilung modelliert. In Bereichen, in denen sich die Verteilungen überlappen, ist der Parameter nicht so hilfreich für die Klassifizierung, deshalb sollte das Gewicht hier klein sein und in Bereichen ohne Überlappung groß.

Um den unteren und oberen Grenzwert zu bestimmen, werden wieder die Gaußverteilungen der gesamten Zugehörigkeitsfunktion berechnet. Der Benutzer muss jetzt zwei Verhältnisse von Wahrscheinlichkeitsdichten angeben (typisch 0,1 und 10), die als Grenzwerte verwendet werden.

Semantische Netze

Semantische Netze sind Modelle des menschlichen Gedächtnisses. Sie bestehen aus Knoten, die Konzepte repräsentieren, und Kanten. Ein Konzept ist dabei ein Tripel aus Konzeptname, einer Menge aller Objekte, die zum Konzept gehören und Extension genannt werden, und der Intension, die die Konzeptmerkmale darstellt. Die Kanten stellen assoziative Verbindungen zwischen den Knoten dar.

Es werden zwei Typen von Netzen unterschieden, die einen enthalten nur eine Art von Kanten. Diese sind ungerichtet und können gegebenenfalls durch einen numerischen Wert gewichtet werden. Diese Art von Netzen werden assoziative Netzwerke genannt. Die zweite Art besitzt verschiedenartige, gerichtete Kanten, die nicht gewichtet werden. Sie ist viel aussagekräftiger und wird deshalb hauptsächlich verwendet und auch als semantisches Netzwerk bezeichnet.

Im Bereich der Logik entsprechen die Knoten den Termen und die Kanten sind Prädikate. Häufige Beziehungen in semantischen Netzen sind „Ist-ein“- und „Hat-Teil“-Beziehungen. Abbildung 2.3 zeigt beispielhaft die Repräsentation der Klasse Seerose in einem semantischen Netz. Neben Beziehungen können Kanten auch Eigenschaften wie im Beispiel den Schutzstatus oder die Höhe ausdrücken. Dies wird durch eine Verbindung zu einem Eigenschaftsknoten modelliert.

Die Konzeptknoten können entweder für eine Klasse von Knoten stehen oder für eine einzelne Instanz. Eine einzelne Instanz wird von einem Individualkonzept wie Seerose-1 repräsentiert und die Klasse von Knoten von einer Konzeptklasse Seerose. Knoten, die Beziehungen zu einem Knoten einer Konzeptklasse haben, gelten für die gesamte Klasse. Dies ist im semantischen Netz aus Abbildung 2.3 unter anderem die Eigenschaft schwach giftig zu sein.

Koch u. a. (1997) nutzen für die Interpretation von Luftbildern und Karten semantische Netze, um die Szene zu beschreiben. Das Netz besteht aus Knoten, die Objekte und sensorspezifische Eigenschaften sein können, und Relationen zwischen diesen. Weiterhin wird das Netz in zwei Bereiche aufgeteilt. Es gibt das Konzept-Netz und das Instanz-Netz, wobei das Konzept-Netz die Konzeptklassen beinhaltet und den grundsätzlichen Aufbau einer Szene beschreibt und im Instanz-Netz die tatsächlich vorhandenen Objekte in Form von Individualkonzepten modelliert werden. Objekte im Netz können einen der vier Zustände „Hypothese“, „fehlende Instanz“, „teilweise vorhandene Instanz“ und „vollständige Instanz“ haben.

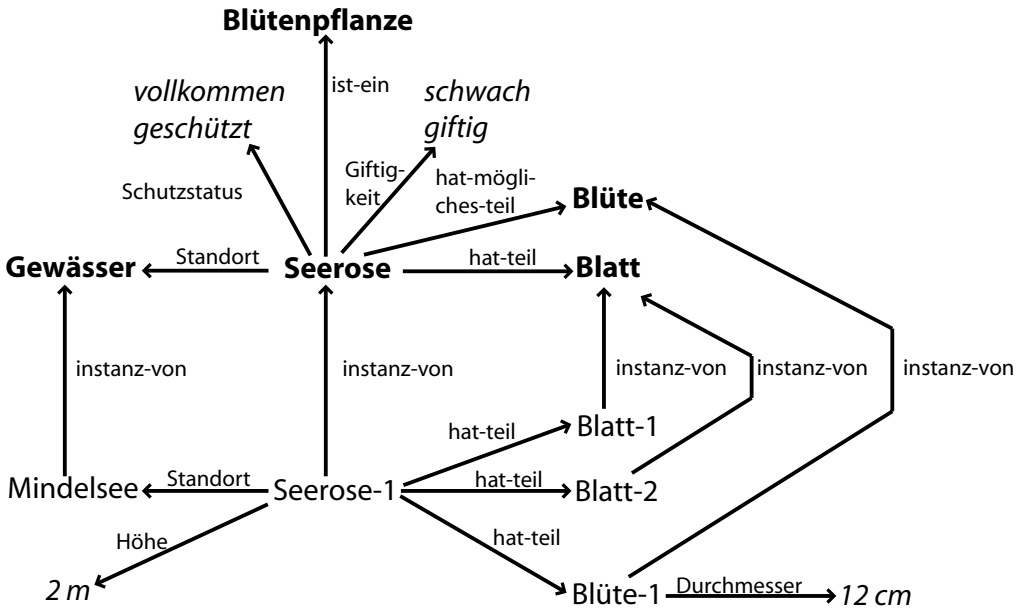


Abbildung 2.3: Modellierung der Konzeptklasse Seerose mit zugehörigem Individualkonzept Seerose-1 (nach Reimer, 1991).

Zwischen den Knoten kann es neben den oben verwendeten Relationen *instance-of, is-a* und *part-of* die Relationen *cdpart-of, con-of, data-of* geben, die eine kontextabhängige „Teil-von"-Beziehung, eine konkrete Realisierung und Instanzen, die direkt in den Daten segmentiert werden können, darstellen. Zusätzlich können Attribute vergeben werden.

Der Ablauf des Verfahrens wird regelbasiert gesteuert. Die Regeln besagen z.B. „Sind alle Instanzen, die eine *part-of* Beziehung zu einer weiteren Instanz haben, vollständig, so ist auch diese weitere Instanz vollständig".

Die Interpretation ist vollständig, wenn eine Instanz gefunden wurde, die dem Modell und den im Bild segmentierten Merkmalen entspricht. Um diese zu finden, wird ein Suchbaum mit konkurrierenden Lösungen aufgestellt, in dem mit dem A*-Algorithmus die beste Lösung für das weitere Vorgehen gesucht wird.

Zu Beginn der Interpretation liegt im semantischen Netz eine Beschreibung der Szene durch Vektordaten vor. Die Struktur dieser Daten wird im ATKIS[1]-Layer abgebildet. Im weiteren Vorgehen werden diese Daten anhand eines Luftbildes verifiziert. Um Konflikte zu vermeiden, werden die Zustände der Knoten im Szene-Layer auf Hypothese gesetzt. Dann kommt es zu einer top-down-Übertragung vom ATKIS-Layer in den Luftbild-Layer. Für die Verifikation wird ein Bildbearbeitungsoperator verwendet, der auf lokaler Varianz und lokalem Kontrast basiert. Nach diesem Prozess sind dann aktuelle Position und Form bekannt. Um den Prozess zu verbessern, kann man auch noch andere Sensoren hinzunehmen. Zusätzlich kann man im Netz noch räumliche Relationen hinzufügen, um die Interpretation zu verbessern. So können auch Relationen modelliert werden wie „eine Brücke führt über einen Fluss".

Das Verfahren eignet sich ebenso für die Interpretation eingescannter Karten. Damit lassen sich mit Karten von verschiedenen Zeitpunkten langfristige Änderungen in der Landnutzung erkennen.

In Liedtke u. a. (2001) wird das wissensbasierte System geoAIDA als Erweiterung von AIDA (Liedtke u. a., 1997) zur Interpretation komplexer Szenen aus Daten verschiedener luft- und satellitengestützer Sensoren vorgestellt. Mit diesem System sollen Daten aus Geoinformationssystemen verifiziert werden. Als Eingabedaten dienen die Sensordaten, GIS-Daten und vorhandenes Wissen, das in einem semantischen Netz modelliert wird. Das System AIDA führt gleichzeitig Bildverarbeitung und symbolische Verarbeitung durch, wobei die Bildverarbeitung der symbolischen Verarbeitung die extrahierten Features liefert und diese der Bildverarbeitung die Bedingungen liefert.

Im Bereich der Bildverarbeitung werden z.B. Gebäude als ebene Flächen im Laserscan detektiert. Anhand von Bilddaten werden Textur-Klassifizierungen vorgenommen oder Straßen extrahiert.

[1] Amtliches Topographisch-Kartographisches Informationssystem

Im System geoAIDA wird das Wissen in einem semantischen Netz gespeichert, welches im Gegensatz zu AIDA hierarchisch gestaltet ist. Die „Ist-Teil-von“-Relation spielt hier also eine wichtige Rolle. Außerdem können die Operatoren in jedem Level des semantischen Netzes angewandt werden. So muss nicht jedes Haus einzeln erkannt werden, um eine Siedlung zu extrahieren.

Bayes-Netze

Ein Bayes-Netz besteht aus einem gerichteten Graphen, dessen Knoten Variablen enthalten. Diese Variablen können eine endliche Menge von Werten annehmen (Jensen und Nielsen, 2007). Sie haben zu einer Zeit genau einen Wert, der aber nicht immer bekannt sein muss. Führt eine gerichtete Kante von A nach B, so bedeutet das, dass der Wert der Variable A die Variable B beeinflusst. In dem gerichteten Graphen dürfen deshalb keine Zyklen auftreten. Zusätzlich zum Graphen gibt es für jede Variable A mit Eltern $B_1, \ldots, B_n$ eine Tabelle mit bedingten Wahrscheinlichkeiten $P(A|B_1, \ldots, B_n)$. Für Variablen A ohne Eltern wird die unbedingte Wahrscheinlichkeit $P(A)$ verwendet. Abbildung 2.4 zeigt beispielhaft ein Bayes-Netz, welches modelliert, wie die Klasse eines Objektes von dessen Farbe und Form abhängt. Für die Variablen Objektfarbe und -form sind unbedingte Wahrscheinlichkeiten angegeben, da sie keine Eltern haben. Die Wahrscheinlichkeit der Objektklasse ist durch dessen Form und Farbe bedingt.

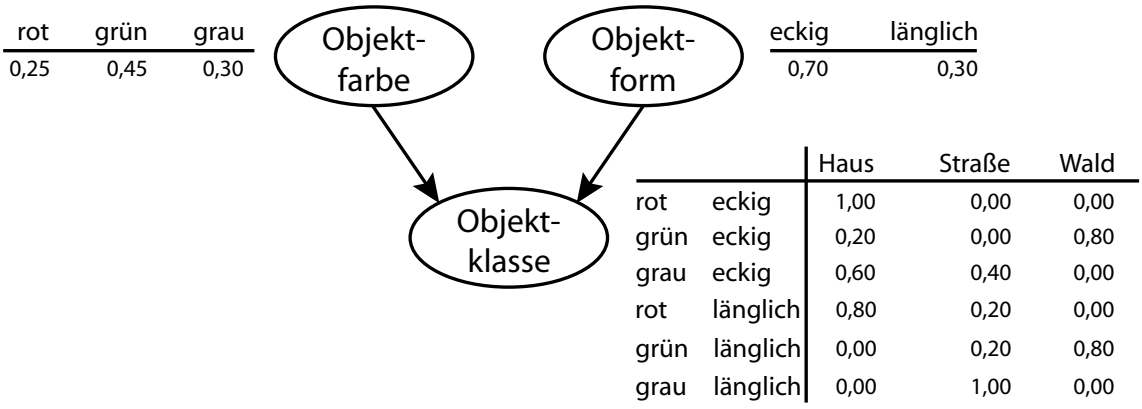


Abbildung 2.4: Beispiel eines Bayes-Netzes.

Ist $\mathcal{U} = \{A_1, \ldots, A_n\}$ die Menge an Variablen eines Bayes-Netzes, so kann man mit der Tabelle über die gemeinsame Wahrscheinlichkeit $P(\mathcal{U})$ alle einzelnen Wahrscheinlichkeiten $P(A_i)$ und $P(A_i|e)$ bei gegebener Evidenz e einiger Variablen des Netzes berechnen. Da $P(\mathcal{U})$ aber mit der Anzahl der Variablen exponentiell wächst, wird die Tabelle nicht direkt angegeben, sondern aus den bedingten Wahrscheinlichkeiten der einzelnen Variablen berechnet.

Für ein Bayes-Netz BN über den Variablen $\mathcal{U}$ sind die bedingten Wahrscheinlichkeiten der Variablen unter gegebenen Eltern angegeben. Aus diesen kann mit der Kettenregel für Bayes-Netze $P(\mathcal{U})$ berechnet werden. Sei $\mathcal{U} = \{A_1, \ldots, A_n\}$. Dann legt das Bayes-Netz eindeutig eine gemeinsame Wahrscheinlichkeitsverteilung $P(\mathcal{U})$ durch das Produkt aller bedingten Wahrscheinlichkeiten fest.

$$P(\mathcal{U}) = \prod_{i=1}^{n} P(A_i|pa(A_i)),$$

wobei $pa(A_i)$ die direkten Vorgänger von A_i bezeichnet. So kann im obigen Beispiel die Wahrscheinlichkeit eines roten, eckigen Objektes mit der Objektklasse Haus über die bedingten Wahrscheinlichkeiten berechnet werden.

$$P(\text{Objektfarbe=rot,Objektform=eckig,Objektart=Haus}) =$$

$$P(\text{Objektart=Haus}|\text{Objektfarbe=rot,Objektform=eckig}) \cdot P(\text{Objektfarbe=rot}) \cdot P(\text{Objektform=eckig}) =$$

$$1,00 \cdot 0,25 \cdot 0,70 = 0,175.$$

Liegen nicht alle Informationen vor, so können die Wahrscheinlichkeiten unter den gegebenen Bedingungen berechnet werden. Die Evidenz in Form von „Die Variable A kann nur im Zustand i oder j sein“ wird durch

einen Vektor **e** der Länge n dargestellt, der an Stelle i und j 1 und sonst 0 stehen hat, wobei n die Anzahl möglicher Zustände ist. Dieser Vektor wird auch Finding genannt.

Ist $P(A) = (x_1, \ldots, x_n)$, wobei x_i die Wahrscheinlichkeit angibt, dass die Variable A im Zustand i ist, so ist die Wahrscheinlichkeit unter der Evidenz e

$$P(A, e) = (0, \ldots, 0, x_i, 0, \ldots, 0, x_j, 0, \ldots, 0),$$

was einer elementweisen Multiplikation mit **e** entspricht. Gleiches gilt für $P(\mathcal{U}, e)$, hier wird in $P(\mathcal{U})$ an alle Stellen, an denen A nicht im Zustand i oder j ist, eine 0 geschrieben. Dies entspricht ebenfalls einer elementweisen Multiplikation mit **e**. Damit ergibt sich dann für Findings $\mathbf{e}_1, \ldots, \mathbf{e}_m$

$$P(\mathcal{U}, e) = \prod_{A \in \mathcal{U}} P(A|pa(A)) \cdot \prod_{i=1}^{m} \mathbf{e}_i.$$

Kulschewski (1997) verwendet Bayes-Netze zur Erkennung von Gebäudetypen aus Luftbildern. Er stellt ein geometrisches und semantisches Modell der Gebäude auf. Im Bayes-Netz werden aus dem Bild extrahierte Features, mögliche Gebäudearten und Vorwissen gespeichert. Neben der Konstruktion des Bayes-Netzes wird noch eine Strategie vorgestellt, die die Reihenfolge festlegt, in der die Features in das Netz gegeben werden.

In der Arbeit werden sechs verschiedene Gebäudetypen berücksichtigt, die einen rechteckigen Grundriss haben und sich in der Dachform unterscheiden (Flachdach, Satteldach, Walmdach, Teilwalmdach, Pultdach und Mansardgiebeldach). Der erste Knoten des Netzes (siehe Abbildung 2.5) ist der Gebäude-Knoten (g) und kann diese sechs verschiedenen Werte annehmen. Der zweite Knoten ist der Aspekt-Knoten (a), der Werte für die 2D-Ansicht des 3D-Objektes beinhaltet. Durch die Einschränkung auf einen 90°-Öffnungswinkel und das Zusammenfassen von Aspekten, die die gleiche Anzahl an Flächen und den gleichen Adjazenzgraphen mit korrespondierender Flächendarstellung (Dreieck, Trapez,...) haben, schränkt sich die Anzahl der möglichen Werte für diesen Knoten auf 25 ein.

Vorgänger des Aspekt-Knotens sind neben dem Nachbarschafts-Knoten (N), der mit der Anzahl der Flächen und dem Adjazenzgraphen 23 verschiedene Werte annehmen kann, die Face-Knoten (f_i). Die Anzahl dieser Knoten wird je nach Anzahl der sichtbaren Flächen in einem Aspekt dynamisch angepasst. Mögliche Werte für diesen Knoten sind Dreieck, Parallelogramm, Trapez, Fünfeck, Sechseck oder Siebeneck. Vorgänger der Face-Knoten sind Feature-Knoten (l_j). Jeder Face-Knoten besitzt eine feste Anzahl an Feature-Knoten als Vorgänger. Diese stehen für geometrische Eigenschaften wie „Es gibt eine Symmetrieachse“, „Es gibt parallele Kanten“ oder semantischen Eigenschaften wie „Ist ein Teil vom Dach“.

Zusätzlich zu den beschriebenen Knoten wird noch Vorwissen über den Knoten V in das Bayes-Netz gebracht. Hier gehen Informationen über die Häufigkeit von Gebäudetypen und Aspekten sowie über verdeckte Flächen ein.

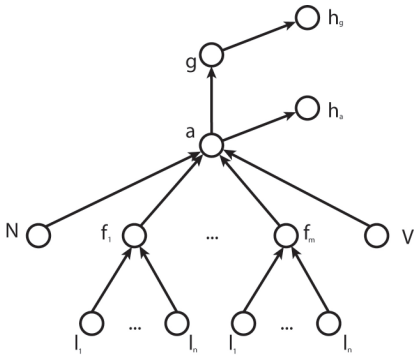


Abbildung 2.5: Bayes-Netz zur Gebäuderekonstruktion (aus Kulschewski, 1997).

Das Rekonstruktionsverfahren startet mit der Bildverarbeitung, die Regionen und deren Nachbarschaften aus den Bilddaten berechnet und daraus den Flächenadjazenzgraphen aufstellt. Zusätzlich sind Kanten, Punkte und deren Bezug zur Fläche bekannt. Aus Kanten und Regionen können weitere Features berechnet werden. Aus dem digitalen Höhenmodell können zusätzliche Informationen entnommen werden.

Aus dem Flächenadjazenzgraphen einer Szene werden die Faces zu einem Aspekt kombiniert, der einen Gebäudetyp darstellt. Hierzu werden nach und nach die Faces in das Bayes-Netz eingefügt. Um die Reihenfolge der Faces

festzulegen, gibt es verschiedene Möglichkeiten. Die erste ist, ein Face zu verwenden, das zu einem Dach gehört. Diese Information kann aus dem Höhenmodell gewonnen werden. Weitere Möglichkeiten sind, eine signifikante Fläche zu wählen oder eine Fläche, die aufgrund ihrer Features eine hohe Wahrscheinlichkeit hat. Die erste Bedingung ist die beste und so werden die anderen nur verwendet, um aus mehreren Dachflächen auszuwählen.

Nach der gewählten Fläche wird der Nachbarschaftsknoten instantiiert. Damit kann die erste Wahrscheinlichkeit im Gebäudeknoten berechnet werden. Für das weitere Vorgehen werden jeweils alle verbleibenden Flächen getestet und die Fläche, die den größten Wahrscheinlichkeitszuwachs im Gebäudeknoten verursacht, wird dem Netz hinzugefügt. Dies ist ein teures Vorgehen und kann alternativ durch eine Prädiktion über die Wahrscheinlichkeit ersetzt werden. Es werden so lange Flächen zum Netz hinzugefügt, bis sich die Wahrscheinlichkeit im Gebäudeknoten nicht mehr verbessert. Ein Vorteil dieses Verfahrens ist die Flexibilität des Netzes.

Künstliche Neuronale Netze

Ein Künstliches Neuronales Netz (KNN) ist ein parallel arbeitendes System, das der Funktionsweise des menschlichen Gehirns nachempfunden ist (Bodendorf, 2006). Es besteht aus Neuronen oder verarbeitenden Elementen, die durch gerichtete und gewichtete Kanten verbunden sind (siehe Abbildung 2.6). Ein Neuron sammelt die eingehenden Signale und sendet abhängig von seinem Aktivierungszustand Signale an nachfolgende Neuronen. Ein KNN ist in mehrere Schichten unterteilt. Dies sind die Eingabeschicht, eine oder mehrere Zwischenschichten und die Ausgabeschicht. Durch die Netzstruktur und die Gewichte der Kanten kann das Verhalten des KNN beeinflusst werden.

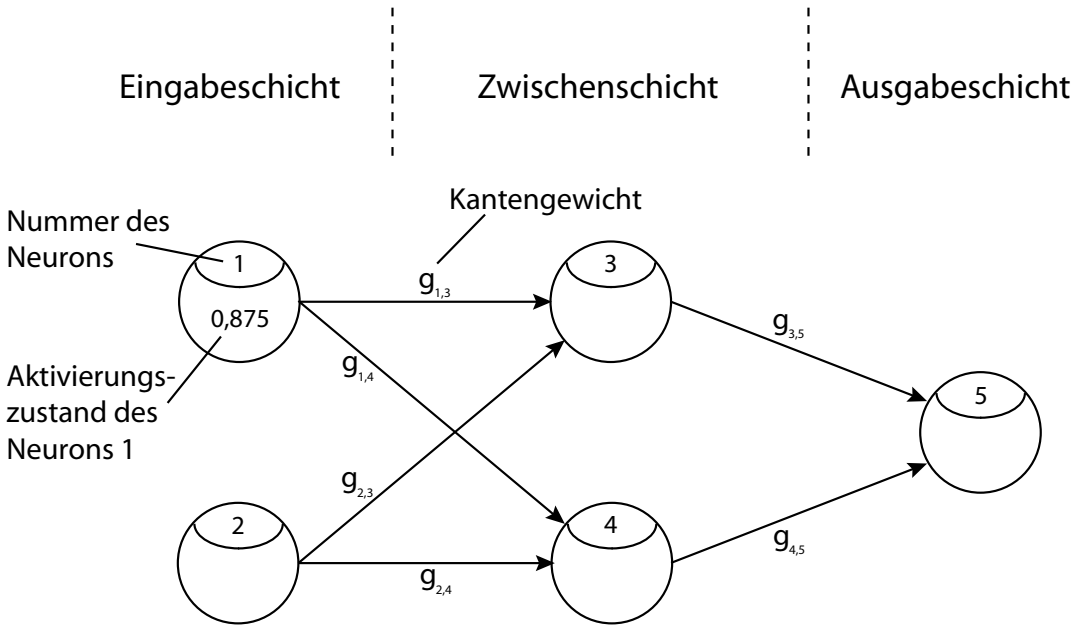


Abbildung 2.6: Künstliches Neuronales Netz (nach Bodendorf, 2006).

Ein KNN kann in zwei Phasen arbeiten. In der Arbeitsphase werden zu den Eingabedaten, die in die Eingabeschicht geschrieben werden, Ausgabedaten berechnet. Dazu werden in jedem Neuron die Eingangssignale gewichtet aufsummiert und daraus das Aktivierungspotential berechnet und an die nachfolgenden Knoten geschickt. Dieser Vorgang setzt sich bis in die Neuronen der Ausgabeschicht fort, in der das Ergebnis steht.

Um dem KNN das richtige Verhalten beizubringen, durchläuft es eine Lernphase, in der Trainingsdaten in Form von Eingabe-Ausgabe-Paaren prozessiert werden. Für diese Phase gibt es verschiedene Lernverfahren. Ein weit verbreitetes Verfahren ist die Backpropagation. Dabei berechnet das KNN zu einer Eingabe die Ist-Ausgabe und vergleicht diese mit der gegebenen Soll-Ausgabe. Die Differenz beider Werte wird als Netzfehler bezeichnet. In einem Rückwärtsdurchlauf durch das KNN werden nun in Abhängigkeit des Netzfehlers die Gewichte der Kanten angepasst. Dies wird mit einer großen Zahl an Trainingsdaten so lange wiederholt, bis der Netzfehler klein genug ist.

Ziel ist es hierbei nicht das Netz so gut auf die Trainingsdaten anzupassen, dass der Netzfehler zu null wird. Dies würde zu schlechteren Resultaten bei anderen Daten führen. Um das zu vermeiden, wird die Menge der

Trainingsdaten in Trainingsdaten und Validierungsdaten unterteilt. Es werden so viele Lernzyklen mit den Trainingsdaten ausgeführt bis der Fehler für die Validierungsdaten sein Minimum erreicht. Dann ist das Training optimal.

Murai und Omatu (1997) entwickelten ein dreistufiges Verfahren zur Klassifizierung von Gebieten aus Satellitenbildern. Im ersten Schritt, der Vorverarbeitung, werden die Daten mit dem k-Means-Verfahren (Hartigan und Wong, 1979) in Cluster eingeteilt, um Trainingsdaten für ein Neuronales Netz zu erhalten. Dieses wird dann zur Klassifizierung der Gebiete verwendet. Hierbei können Fehlklassifizierungen auftreten, da die Daten nur eine geringe Auflösung von 30 m besitzen und in Kombination mit der Abschattung durch Wolken die Datenwerte eine Kombination der Werte für Erdoberfläche und Wolken sein können. Um diese Fehlklassifizierungen anschließend zu erkennen und zu beheben, wird geographisches Wissen eingeführt.

Die Konstruktion des Netzes hängt von den Parametern der Aufgabenstellung ab. Aus den gegebenen Daten werden drei Eingabelayer verwendet und die Zahl der möglichen Klassen, in die unterteilt werden soll, ist zehn. Somit ergibt sich eine Anzahl von drei Knoten in der Eingabeschicht und zehn Knoten in der Ausgabeschicht. Schwieriger ist es, die Anzahl der Knoten in der Zwischenschicht festzulegen. Zu wenige Knoten reduzieren die Information aus der Eingabe und verhindern somit den Lernprozess. Durch zu viele Knoten verliert das Netz seine Generalisierungskapazität, die aber für verrauschte Daten nötig ist. In (Murai und Omatu, 1997) wird die Zahl der Knoten der Zwischenschicht auf 10 festgelegt.

Die in Cluster aufgeteilten Daten werden in der Lernphase als Trainingsdaten verwendet. In der Arbeitsphase auftretende Fehler in der Klassifizierung werden mit einer wissensbasierten Fehlerkorrektur behoben. Diese nutzt geographisches Wissen und betrachtet jeweils die Umgebung eines Bildpunkts. Betrachtet man eine große Umgebung, so ist komplexes Wissen nötig und die Berechnungen dauern sehr lange. Deshalb wurde in der Arbeit eine 3×3-Umgebung gewählt. In einer so kleinen Maske wäre es zwar schwierig zu klassifizieren, aber hier müssen nur typische Fehler bei der Klassifizierung behandelt werden. Dazu werden Pixelmuster angegeben, die im ersten Schritt den Fehler erkennen. Die fehlerhaften Pixelmuster entsprechen Anordnungen von Klassen, die in der Realität nicht auf diese Weise angeordnet sein können. Für die Korrektur im zweiten Schritt werden Regeln aufgestellt, wie die Pixel-Klassifizierung ersetzt wird, um eine korrekte Lösung zu erhalten.

Würde das geographische Wissen für diese Korrektur im Neuronalen Netz enthalten sein, so würde dieses sehr groß und komplex werden. Deshalb wurde hier eine wissensbasierte Fehlerkorrektur dem Klassifizierungsprozess nachgeschaltet.

2.1.2 Weitere Methoden zur Objekterkennung

Neben den wissensbasierten Methoden wurden auch viele Heuristiken zur Klassifikation und Objekterkennung bzw. -rekonstruktion entwickelt. Einige davon werden im Folgenden vorgestellt.

CAD-basierte Rekonstruktion

CAD-Modelle können für die Rekonstruktion verwendet werden. Hoffman u. a. (1989) verbessern die Objekterkennungsmethoden, indem sie für die Lernphase CAD-Modelle statt der Sensordaten verwenden. Dies ist für Bild- und Entfernungsdaten möglich. Für Bilddaten sind Kanten das Hauptmerkmal, diese können über die üblichen Kantendetektoren ermittelt werden. Für Entfernungsdaten spielen Flächen die wichtigste Rolle. Die Entfernungsdaten lassen sich mit Raytracing-Verfahren direkt aus dem CAD-Modell gewinnen.

Hansen und Henderson (1989) stellen eine weitere Methode vor, CAGD-Modelle (Computer Aided Geometric Design) in einer Szene zu erkennen. Sie verwenden geometrisches Wissen, um einen Strategie-Baum zu erstellen. Hiermit können Objekte in einer Szene erkannt und lokalisiert werden, die auch teilweise verdeckt sein dürfen.

Das Verfahren teilt sich in drei Schritte. Der erste ist die Repräsentation des geometrischen Wissens. Hierzu wurden Methoden entwickelt, die die CAD-basierte Darstellung der Modelle in eine Computer Vision Repräsentation umwandelt. Im nächsten Schritt werden Merkmale ausgewählt, die zuverlässig erkannt werden und die Lage des Objekts möglichst genau bestimmen. Hier werden hauptsächlich geometrische Merkmale wie Ecken, Kanten, Ebenen, Punkte und andere verwendet. Es können aber auch z.B. Textureigenschaften oder Reflexivität verwendet werden. Der Prozess der Merkmals-Auswahl wird auch als Filter bezeichnet, wobei die Merkmale, die nicht den Anforderungen entsprechen, ausgefiltert werden. Es kann nach folgenden fünf Eigenschaften gefiltert werden: Seltenheit, Robustheit, Kosten, Vollständigkeit und Konsistenz. Die Reihenfolge der angewandten Filter ist bei diesem Verfahren entscheidend und wird durch Wissen über den Erkennungsprozess und durch Erfahrung gebildet. Da ein Hintereinanderausführen der Filter dazu führen könnte die leere Menge als Merkmalsmenge

zu erhalten, löschen die Filter nicht alle Elemente, die nicht der Bedingung entsprechen, sondern ordnen sie in einer Reihenfolge an.

Nachdem diese Merkmale bestimmt sind, wird automatisch eine Suchstrategie entwickelt, die auf den besten Merkmalen und dem Einfluss dieser auf die Beschränkung der Lage des Objekts basiert. Dazu wird ein Strategie-Baum erstellt, der den Erkennungsprozess abbildet. Im Wurzelknoten steht hierbei das zu erkennende Objekt. In der nächsten Ebene stehen die einflussreichen Merkmale, die blickwinkelunabhängig sind und eine schnelle Erkennung des Objekts und seiner Position erlauben. In der nächsten Ebene stehen die Merkmale, die die Hypothese aus der höheren Ebene bekräftigen können.

Aus den gemessenen Daten werden Merkmale extrahiert und aus den Merkmalen und dem Strategiebaum wird eine Hypothese generiert. Diese kann dann mithilfe von Merkmals- und Pixelkorrelation validiert werden.

Kantenbasierte Extraktion

Bei der Extraktion von Objekten aus Luftbildern spielen Kanten eine große Rolle. Viele Verfahren extrahieren zunächst Kanten im Bild und aus diesen werden dann auf verschiedene Weise Objekte generiert.

Fischer u. a. (1998) zeigen einen modellbasierten Ansatz zur automatischen 3D-Extraktion aus Luftbildern. Sie verwenden das Wissen über Gebäude, um den Prozess zu kontrollieren. Sie verwenden gleichzeitig ein generisches 3D-Objekt-Modell, das die räumliche Erscheinung und die Charakteristik der Komponenten darstellt, und ein 2D-Bild-Modell, das ein Sensor- bzw. Sicht-Modell ist und das projizierte Erscheinungsbild darstellt.

Der Ansatz ist, eine Hypothese zu erzeugen und diese dann zu testen. Er kombiniert bottom-up (datengetriebenes) und top-down (modellgetriebenes) Vorgehen. Das Objektmodell ist eine mehrschichtige *Teil-von*-Hierarchie, wobei die verschiedenen Schichten verschiedene semantische Abstraktionen darstellen. Die Schichten beinhalten Merkmale, zusammengesetzte Merkmale, Gebäudeteile und Gebäude. Diese Klassen sind Generalisierungen, Unterklassen von Merkmalen sind Punkt, Linie und Region. Gebäudeteile können Terminale oder Konnektoren sein. Das 2D-Bild-Modell bildet die gleichen Schichten ab und stellt das Aussehen im Bild dar.

Eingabedaten für das Verfahren sind digitale Rasterdaten. Aus diesen werden zunächst Punkte, Linien und Regionen extrahiert und die Beziehungen zwischen diesen berechnet. Das eigentliche Extraktionsverfahren ist eine Iteration der drei Schritte Rekonstruktion von 3D-Ecken, Generierung von Hypothesen und Verifikation der Hypothesen.

Im ersten Schritt sollen schon früh 2D- und 3D-Schlussfolgerungen gezogen werden. 2D-Ecken-Modelle helfen dabei, in 3D die Ecken zu rekonstruieren. Diese 3D-Ecken werden verwendet, um in einer Datenbank von parametrisierten Gebäudeteilen, die die Ecken erklären, nachzuschlagen. Aus den Ecken werden dann Gebäudeteile zusammengesetzt. Die so erhaltenen Gebäudehypothesen werden zur Verifikation ins 2D-Bild projiziert und hier durch einen Abstimmungs-Prozess abgelehnt oder angenommen. Durch geometrische Schlussfolgerungen können mithilfe der projektiven Geometrie im Bild Merkmale gefunden werden. Diese legen freie Parameter der Gebäudehypothese fest. Diese Schritte werden iterativ ausgeführt und vergrößern so schrittweise das Wissen über die Gebäude, bis keine weiteren Hypothesen aufgestellt werden können.

Ein weiteres auf Kanten basierendes Verfahren zur automatischen Gebäuderekonstruktion aus Luftbildern beschreiben Baillard u. a. (1999). Dies wurde im Projekt IMPACT (IMage Processing for Automatic Cartographic Tools) entwickelt. Das Verfahren teilt sich in zwei Aufgaben. Zunächst werden automatisch Liniensegmente aus verschiedenen Bildern zugeordnet. Hierzu werden sowohl geometrische als auch fotometrische Bedingungen verwendet. In dieser Arbeit wird das Verfahren von Schmid und Zisserman (1997), welches für drei Bilder konstruiert wurde, auf eine beliebige Anzahl an Eingabebildern erweitert.

Im zweiten Teil werden aus den Liniensegmenten stückweise planare Rekonstruktionen erstellt. Hierbei können Hypothesen aus einer einzigen 3D-Linie erstellt werden. Zuerst werden verlässliche Halbebenen, die durch eine Linie definiert sind, berechnet. Hierzu werden Ähnlichkeitsbewertungen durch Korrelation korrespondierender Pixel in verschiedenen Bildern berechnet. Im zweiten Schritt werden Linien und Halbebenen gruppiert und vervollständigt. Kollineare Halbebenen werden zusammengefasst und Linien, deren Halbebenen koplanar sind, werden vereinigt. Zusätzlich werden auf Schnittkanten von Ebenenpaaren neue Linien eingefügt.

Im dritten Schritt werden vollständige Ebenen aus den Linien geformt. Zunächst müssen die Begrenzungen der Ebenen festgelegt werden. Dazu werden die Linien, die die Ebene initiiert haben, verwendet. Abgeschlossen werden die Ebenen durch heuristische Regeln. So werden z.B. Begrenzungslinien angepasst, wenn diese sich

schneiden oder ihre Endpunkte dicht beieinander liegen. Dann wird die konvexe Hülle der Endpunkte der begrenzenden Linien berechnet. Ebenen, die nur eine Begrenzungslinie haben, werden eliminiert. Die so gefundenen Ebenen werden nach der Ähnlichkeit der Intensität bewertet und somit fallen falsche Ebenen weg.

Gebäudededektion basierend auf Höhenmodellen

Andere Verfahren nehmen digitale Höhenmodelle (DHM) zur Hilfe, um die Bereiche zu selektieren, in denen Gebäude vorkommen. Abbildung 2.7 zeigt, dass Gebäude und Vegetation im DHM als hellere Bereiche zu erkennen sind. Die zusammenhängenden Bereiche werden auch als Blobs bezeichnet. Im Höhenmodell werden sie als Gebäudehypothesen extrahiert. Zheltov u. a. (2001) entfernen zunächst die Gebäude mit Grauwertmorphologie aus dem Höhenmodell. Zieht man das Ergebnis vom Höhenmodell ab, so erhält man die interessanten Bereiche. Niederöst (2000) bildet zur Blob-Erkennung aus dem DOM Schichtbilder, die je einer Höhenschicht von 1,5 m zugeordnet sind. Ihre Pixel stellen binäre Bins dar, die anzeigen, ob in dieser Höhenschicht Punkte liegen. Um überlappende Schichten zu erhalten, beginnen diese im Meterabstand. Ist nun ein Pixel in mehreren aufeinander folgenden Schichten belegt, so wird dies als Blob detektiert.

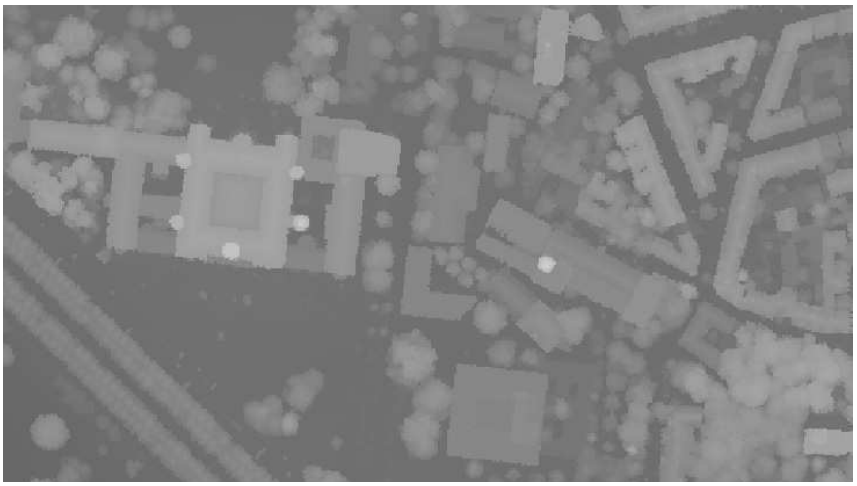


Abbildung 2.7: Im digitalen Höhenmodell sind Gebäude und Vegetation gut als helle Bereiche zu erkennen.

Niederöst (2000) entwickelt im Projekt ATOMI (Automated reconstruction of Topographic Objects from aerial images using vectorised Map Information) ein Verfahren zur Objekterkennung aus Luftbildern. In diesem Projekt werden Vektordaten von Verkehrswegen und Gebäuden auf Basis von digitalen Luftbildern weitgehend automatisch aktualisiert (Eidenbenz u. a., 2000). Im Vordergrund steht hier die Zuverlässigkeit. Das System erkennt Fehler automatisch und diese Objekte müssen dann manuell nachgemessen werden. Verkehrswege und Gebäude werden zunächst getrennt betrachtet. Für beide werden aber gleiche Strategien angewandt. Es werden verschiedene Hinweise für die Existenz von Objekten und existierende Datenquellen zur Objekterkennung verwendet. Hierbei werden nicht nur komplementäre, sondern auch redundante Informationen genutzt, um Fehler und unvollständige Ergebnisse der einzelnen Komponenten auszugleichen. Als Hinweise werden unter anderem Farbe, Schatten, Textur und Kanten verwendet, aber auch Hinweise wie fahrende Autos auf der Straße oder der Kontext zwischen Straße und Gebäude. Des Weiteren wird vorhandenes Wissen über die zu extrahierenden Objekte genutzt.

Niederöst (2000) verwendet sein Verfahren zur Aktualisierung von Karten. Als Vorwissen werden Vektordaten verwendet, die evtl. nicht dem aktuellen Stand entsprechen. Es können auch Gebäude fehlen oder Gebäude beinhaltet sein, die nicht mehr existieren. Zusätzlich werden das digitale Oberflächen Modell und das digitale Geländemodell verwendet. Aus den Luftbildern wird ein Orthophoto berechnet.

Da die Vektordaten nicht aktuell sind, werden zusätzlich zu diesen noch mit zwei Methoden weitere Gebäudehypothesen aufgestellt. Die erste ist die oben beschriebene Blob-Erkennung und die zweite eine Klassifizierung. Zur Klassifizierung werden mehrere Eingabekanäle verwendet. Der S-Kanal im HSI-Farbraum wird verwendet, um Schattenbereiche auszulassen. Eine Kombination aus Rot- und Grün-Kanal gibt Aufschluss darüber, wie anthropogen ein Gebiet ist. Auf der Basis dieses Kanals sowie der Textur wird der Bereich mittels K-means-Clustering in zwei Gebiete geteilt. Eine Kombination aller einzelnen Klassifikationen liefert Gebäudehypothesen.

Zur Gebäuderekonstruktion werden die Ergebnisse aus der Blob-Erkennung, der Klassifizierung und die Vektordaten verwendet. Diese werden anschließend durch Translation, Rotation und Skalierung verbessert, wobei die Parameter jeweils in einem gewissen Bereich um den Ursprungswert variiert werden und mithilfe einer Bewertungsfunktion der beste Wert ermittelt wird. Die Höhe wird aus dem DOM aus dem Median aller Punkte innerhalb des Gebäudes berechnet.

Die Bewertung findet anhand von drei Bereichen statt. Diese sind ineinander geschachtelte Rechtecke. Der äußere Bereich liegt außerhalb des Gebäudes, der mittlere Bereich ist die Randregion und der dritte der innere Bereich. Der erste Bereich sollte eine geringe Höhe haben und wenig anthropogen sein. In beiden inneren Bereichen sollten diese Werte hoch und ähnlich sein.

Zheltov u. a. (2001) schränken in ihrem Verfahren das Gebäudemodell durch folgende Bedingungen ein. Die Gebäude sollen einen annähernd rechteckigen Grundriss und vertikale Wände haben, Nischen und Bögen im Grundriss werden nicht modelliert und die Dächer sind Flachdächer oder haben eine Firstlinie, wobei ein eventueller Überstand vernachlässigt werden kann. Die Gebäuderekonstruktion teilt sich in die Auswahl der Bereiche durch Blobs im Höhenmodell und das Erkennen, Gruppieren und die Analyse der Kanten im Bild.

Zur Gebäudeextraktion werden gerade Liniensegmente mit dem Algorithmus von Burns u. a. (1986) extrahiert. Zu den erhaltenen Segmenten werden Eigenschaften wie Anfangs- und Endpunkt, Länge, Geradengleichung und der Winkel zur horizontalen Achse berechnet. Durch Regeln basierend auf Maßen wie Nähe, Kollinearität, Parallelität usw. werden Rechtecke mit Firstlinie aus den Segmenten gefiltert.

Außerdem wird die 3D-Rekonstruktion anhand des Höhenmodells durchgeführt. Hierbei wird die Höhe direkt aus dem Höhenmodell entnommen. Somit hängt die Genauigkeit des Modells von der Genauigkeit des Höhenmodells ab. Alternativ kann die Höhe durch Stereobildzuordnung bestimmt werden.

Dieses Verfahren ist auf ein Gebäudemodell spezialisiert und für weitere Modelle muss jeweils ein neues Extraktionsverfahren entwickelt werden. Für zylindrische Objekte wird z.B. die Hough-Transformation verwendet. Als Verfahren für Gebäudeextraktion mit verschiedenen Dachformen wurde ein halbautomatisches Verfahren entwickelt, bei dem der Benutzer jeweils die Dachform aus einer Liste auswählt und Schlüsselpunkte im Bild markiert. Dieses sind beim rechteckigen Dach z.B. Punkte in der Nähe der kurzen Seiten.

Gebäuderekonstruktion auf Basis von Grundrissen

Falls von dem Gebiet, in dem Gebäude extrahiert werden sollen, Daten der Grundrisse zur Verfügung stehen, können diese verwendet werden, um die Lage der Gebäude zu erhalten. Brenner u. a. (2001), Stilla und Jurkiewicz (1999) und Suveg und Vosselman (2002) wenden diese Daten für die Gebäuderekonstruktion aus Laserscans bzw. Luftbildern an.

Brenner (2000) verwendet Grundrisse und Laserscandaten für die 3D-Rekonstruktion von Gebäuden. Zuerst werden die Gebäudegrundrisse in Rechtecke zerlegt, in die Gebäude-Primitive gesetzt werden. Aus der Zahl möglicher Primitive werden die passenden nach geometrischen Kriterien wie Winkel und Flächeninhalt der Dachfläche gewählt. Bleiben nach den gesetzten Schwellwerten mehrere Möglichkeiten, so wird das Primitiv mit dem kleinsten Fehler aus der anschließenden Parameterschätzung durch Ausgleichung gewählt. Die Parameter werden je nach Dachform durch eine Ausgleichung geschätzt.

Da dieses Verfahren Strukturen, die nicht im Grundriss enthalten sind, nicht modellieren kann, wurde noch eine semiautomatische Erweiterung entwickelt. Hier kann der Benutzer zusätzliche Strukturen wie Gauben modellieren oder Primitive hinzufügen oder löschen. Ebenso können die geschätzten Parameter manuell verändert werden. Das Verfahren bietet den Vorteil, dass im ersten Schritt automatisch eine große Anzahl an Modellen erzeugt werden kann. Interaktion ist erst bei einer möglichen Verfeinerung des Modells nötig.

Die hier verwendeten Dächer beschränken sich auf solche, die sich aus Primitiven zusammensetzen lassen. Deshalb wurde ein weiteres Verfahren entwickelt, bei dem das Dach in einzelne Dachflächen segmentiert und zu diesen der Normalenvektor berechnet wird. Mit einem grammatikalischen Ansatz werden dann die Regionen aus der Segmentierung bestimmt, die zur Dachrekonstruktion verwendet werden können. Benachbarte Flächen werden dazu nach ihrer Normalenvektor-Kompatibilität gekennzeichnet. Abbildung 2.8 zeigt die möglichen Label „kompatibel“, „kompatibel mit dem Vorgänger“, „entgegen dem Nachfolger“, „senkrecht nach links“ usw. und Beispieldächer mit entsprechenden Labeln. Um zu beschreiben, welche Label-Folgen akzeptiert werden, werden Regeln mithilfe von Regulären Ausdrücken definiert. Alle Muster, die nach der Anwendung „kompatibel“ sind, werden akzeptiert. Folgen mehrere akzeptierte Muster aufeinander, so werden sie durch Regeln zu Gruppen

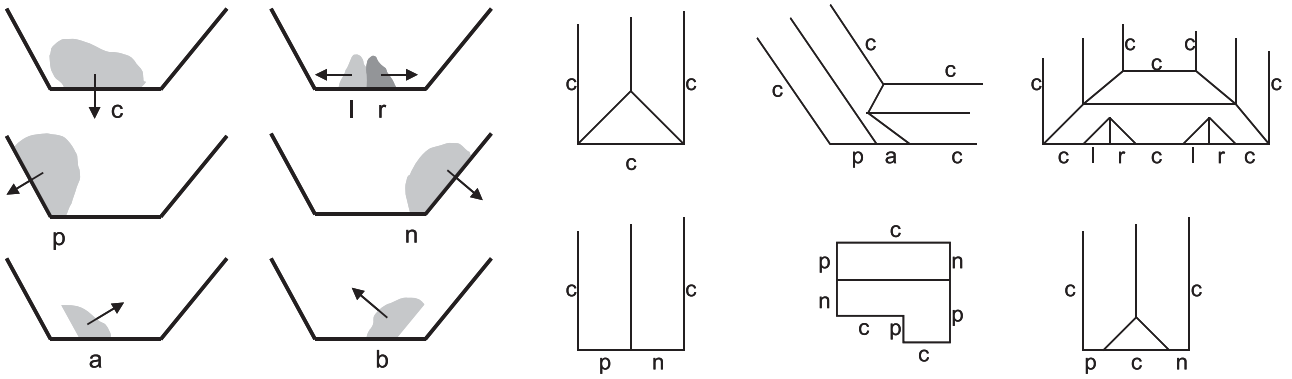


Abbildung 2.8: Label der Dachflächen basierend auf dem Normalenvektor und einige Beispiele zu typischen Dachflächen. Kompatibel (c), senkrecht links (l) und rechts (r), Vorgänger (p) und Nachfolger (n), entgegen zu Vorgänger (a) und Nachfolger (b) (aus Brenner, 2000).

zusammengefasst. Auf Basis der akzeptierten Gruppen wird die Dachtopologie bestimmt und anschließend werden die Ebenenparameter neu geschätzt.

Stilla und Jurkiewicz (1999) nutzen Karten zur Aufteilung in einzelne Gebäude, modellieren die Dachstrukturen aber unabhängig hiervon aus den Höhendaten und können somit auch Strukturen modellieren, die nicht in der Karte vorkommen.

Die Gebäudebeschreibung entsteht in mehreren Schritten. Zuerst werden prismatische Objekte aus einer Karte im Maßstab 1:5000 erzeugt. Eigenschaften wie Verbundenheit, Nähe und Enthaltensein werden mit einem Produktionsnetz eines generischen Modells getestet, um die Gebäudeteile aufzuteilen und zu gruppieren. Basierend auf der Arbeit von Stilla und Michaelsen (1997) wird eine hierarchische Beschreibung der Gebäude oder Gebäudekomplexe erzeugt.

Der zweite Schritt ist die Rekonstruktion der Dachform aus den Höhendaten. Diese stützt sich auf das Histogramm über die Höhenwerte, das für jede Dachform eine charakteristische Form besitzt. Für einige Dachformen zeigt Abbildung 2.9 die zugehörigen Höhenhistogramme. Je nach Histogrammform wird eine Dachhypothese aufgestellt. Möglich sind hierbei Flachdächer, Flachdächer mit Aufbauten und geneigte Dächer. Ein Flachdach zeigt sich durch eine einzelne Häufung im Histogramm. Mehrere Häufungen unterstützen die Hypothese „Flachdach mit Aufbauten“. Der Wert der niedrigsten Häufung ist die Dachhöhe und weitere Häufungen stellen Aufbauten dar. Diese werden modelliert, wenn ihre Fläche genügend groß ist.

Rechteckige oder trapezförmige Histogramme deuten auf geneigte Dachformen hin. Da bei realen Daten die exakte Form nicht aus dem Histogramm gelesen werden kann, wird zusätzlich ein Gradientenfeld berechnet. Mit dem Histogramm über die Orientierung der Gradienten wird die Dachfläche segmentiert. Durch Erosion und Dilatation werden kleine Löcher geschlossen und kaum verbundene Bereiche getrennt. Da immer noch verschiedene Steigungen in einem Segment vorkommen können, wird mithilfe eines Steigungshistogramms eventuell noch weiter unterteilt. Aus den Punkten eines Segments wird durch Kleinste-Quadrate-Schätzung eine Ebene bestimmt. Aus den Ebenen werden schließlich Schnittkanten und -punkte berechnet.

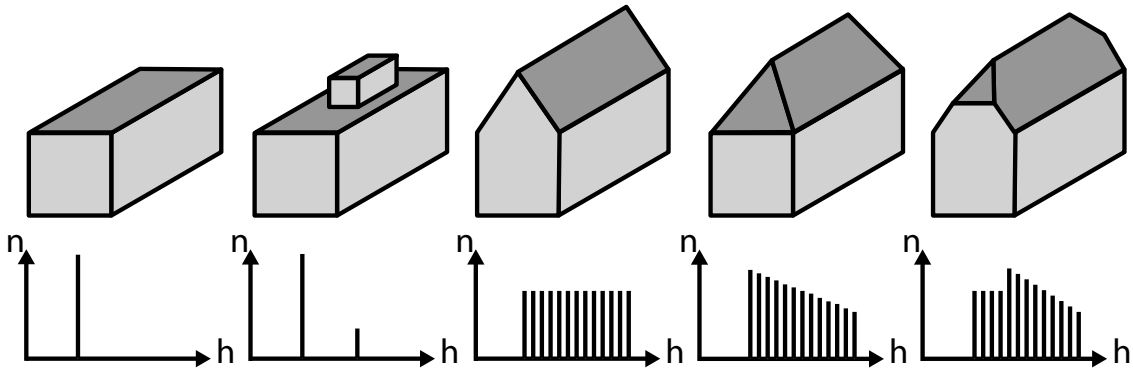


Abbildung 2.9: Charakteristische Höhenhistogramme für unterschiedliche Dachformen (nach Stilla und Jurkiewicz, 1999).

Suveg und Vosselman (2002) rekonstruieren 3D-Gebäude aus Luftbildern mithilfe von Grundrissen aus 2D-GIS-Daten und weiterem Wissen über Gebäude. Hierzu verwenden sie eine Bibliothek von Gebäudeprimitiven. Ihr Modell geht davon aus, dass komplexe Gebäude sich aus einfachen Gebäudeteilen mit Flach-, Sattel- oder Walmdach zusammensetzen lassen.

Das Verfahren findet zuerst Gebäude und rekonstruiert diese anschließend. Dieses geschieht durch eine vielschichtige Generierung und Verifikation von Hypothesen im Suchbaum. Als Grundlage wird der Grundriss des Gebäudes in den GIS-Daten verwendet. Dieser wird in Rechtecke partitioniert. Die erste Ebene des Suchbaums enthält alle Möglichkeiten, das Gebäude zu partitionieren. Diese werden in eine Rangfolge gebracht, wobei die Reihenfolge dadurch bestimmt wird, wie gut die Partitionierung zum Bild passt. Die zweite Schicht enthält zu jeder möglichen Partitionierung die zugehörigen Partitionen. In der dritten Schicht befinden sich die Gebäude-Hypothesen aus einer Datenbank. Zur Verifikation wird die Hypothese in das Bild projiziert. Die Zuordnung zu Informationen aus dem Bild steuert die Suche im Baum durch eine Bewertungsfunktion. Diese basiert zum Einen auf dem ins Bild projizierten Gebäudeumriss. Die Kanten werden mit Gradienten in den Bildern abgeglichen. Zum Anderen werden Helligkeitsunterschiede in korrespondierenden Bereichen der unterschiedlichen Bilder untersucht. Findet sich in diesem Prozess keine passende Partition, so wird eine neue auf dem Bild basierende Partition berechnet und das Verfahren von neuem gestartet. Zusätzlich können geometrische Bedingungen, wie „Zwei Gebäude haben die gleiche Orientierung“ oder „Zwei Gebäude teilen sich eine Kante“, das Ergebnis verbessern.

2.2 Arbeiten zur Fassadenrekonstruktion

Da die Anforderungen an 3D-Modelle stetig wachsen, werden auch höhere Ansprüche an den Detailgrad der Modellierung gestellt. Um diesen zu genügen, müssen auch einzelne Fassaden mit ihren Elementen wie Fenstern, Türen usw. modelliert werden. Als Datengrundlage können terrestrische Scans und auch Bilddaten dienen. Viele der Rekonstruktionsverfahren basieren darauf, dass die Glasscheiben der Fenster nicht richtig erfasst werden, da der Laserstrahl nur teilweise reflektiert wird. Somit können Fenster an Löchern in der Punktwolke erkannt werden. Für Bilddaten gilt eine ähnliche Aussage, hier werden Fenster als dunkle Bereiche auf der Fassade dargestellt.

Pu (2008) stellt ein Verfahren zur automatischen Rekonstruktion von Fassaden aus Laserscan-Daten vor. Hierbei handelt es sich um ein Bottom-up-Verfahren. Im ersten Schritt wird die Punktwolke segmentiert. In der Arbeit werden verschiedene Klassen wie Wand, Dach, Tür oder Vorbau definiert. Die Segmente werden den Klassen zugeordnet, wenn die bestimmte Bedingungen erfüllen. Diese Bedingungen basieren auf der Größe, der Position, der Richtung und der Topologie der Segmente. Sie beinhalten Informationen wie „Das Dach liegt über der Wand“ und „Eine Wand ist eine vertikale Fläche, die sich möglicherweise mit dem Boden schneidet“. Fenster sind in dieser Klassifikation zunächst nicht enthalten. Sie werden später als Löcher in der Wand erkannt.

Nachdem den Flächen ihre Klasse zugewiesen ist, wird der Umriss bestimmt. Dies ist nicht nur abhängig von der Punktwolke, die das Segment bildet, sondern auch von der dem Segment zugewiesenen Klasse. Für die einzelnen Fassadenelemente werden verschiedene Methoden vorgestellt, den Umriss zu bestimmen. Hierbei werden ebenfalls Bedingungen berücksichtigt. Beispielsweise müssen sich Dach und Wand schneiden. Dies ist in der Punktwolke aufgrund von Verdeckungen nicht immer der Fall. Dementsprechend wird der Umriss des Daches angepasst.

Einen weiteren datengetriebenen Ansatz zur Fassadenrekonstruktion aus Scan- und Bilddaten stellen Becker und Haala (2008) vor. Sie gehen davon aus, dass schon ein grobes Modell des Gebäudes, z.B. aus Luftscans generiert, vorhanden ist. Die terrestrisch erfasste Punktwolke wird mit dem vorhandenen Modell registriert und für dessen geometrische Verfeinerung verwendet. Zunächst werden die Daten in ein lokales Koordinatensystem gebracht, welches in der Fassadenebene liegt. Dadurch reduziert sich das Problem auf eine 2,5D-Fragestellung. Im nächsten Schritt soll die Fassade in Zellen unterteilt werden. Hierzu werden die Umrandungen der Fenster verwendet. Diese werden anhand der geringeren Punktdichte erkannt. Rechte und linke Fensterbegrenzungen verursachen vertikale Unterteilungen und obere und untere Begrenzungen horizontale Unterteilungen über die ganze Fassade (siehe Abbildung 2.10 (links)). Die so entstandenen Zellen werden anhand der enthaltenen Punktdichte, die in der so genannten *point availability map* (Abbildung 2.10 (rechts)) gespeichert ist, nach festen Schwellwerten als Fenster oder Wand klassifiziert. Bei Werten zwischen diesen beiden Schwellwerten wird die Klasse der Nachbarzellen in die Entscheidung mit einbezogen.

Das so entstandene Fenstermodell kann anschließend mit Informationen aus Bildern verfeinert werden. Durch eine Kantenextraktion können Sprossen in den Fenstern detektiert werden. Diese werden dann ins Modell

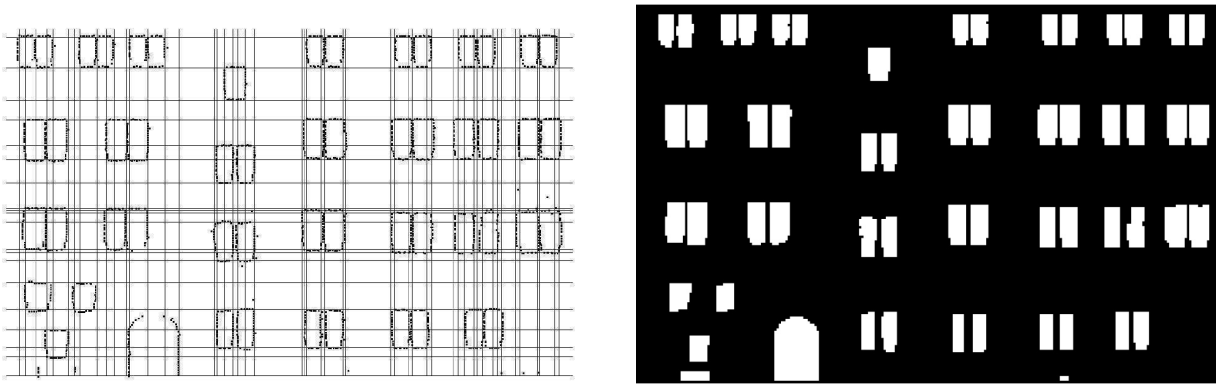


Abbildung 2.10: Unterteilung der Fassade entlang der Fensterränder (links) und point availability map (rechts) (aus Becker und Haala, 2007).

integriert. Bei manchen Fenstern ist es aufgrund von Verdeckungen durch einen extrem schrägen Blickwinkel nicht möglich, die Sprossen zu erkennen. In diesem Fall wird wieder auf die Information aus anderen gut sichtbaren Fenstern zurückgegriffen. Es wird angenommen, dass Fenster der gleichen Größe dieselbe Struktur haben, wenn sie in derselben Zeile oder Spalte liegen.

Die Autoren verwenden die so gewonnenen Modelle später für die Generierung von Fassaden in den Bereichen, die nicht vom Laserscanner erfasst werden können (Becker u. a., 2008). Hierzu wird die rekonstruierte Fassade zum Aufbau einer Wissensbasis verwendet. Sie wird in die einzelnen Stockwerke und dann weiter in Kacheln unterteilt. Diese können Wand- und Geometrieelemente sein und bilden zusammen mit ihren Beziehungen untereinander das Vokabular einer Grammatik. Aus der Abfolge werden dann Grammatikregeln abgeleitet und deren Wahrscheinlichkeiten bestimmt. Mit dieser Grammatik können anschließend nicht erfasste Gebäudeteile konstruiert werden.

Mayer und Reznik (2006) entwickeln einen wissensbasierten Ansatz zur Fassadenrekonstruktion aus Bildsequenzen. Da hier Bilddaten verwendet werden, muss zuerst eine 3D-Rekonstruktion und die Bestimmung von Fassadenebenen und -punkten durchgeführt werden. Anschließend werden Hypothesen für Fensterpositionen aufgestellt. Dieses geschieht nicht auf der Basis der oben genannten dunklen Regionen auf der Fassade, da diese Information nicht immer zuverlässig ist. Stattdessen werden die Fensterhypothesen mithilfe von Implicit Shape Models aufgestellt. Das Aussehen von Fenstern wird aus Trainingsdaten gelernt, indem Fenster aus Bildern ausgeschnitten werden. Auf diesen werden Interest-Punkte nach dem Ansatz von Förstner und Gülch (1987) berechnet und manuell die Mittelpunkte markiert (siehe Abbildung 2.11). Der Differenzvektor von den Förstner-Punkten zum Mittelpunkt wird gespeichert und bildet so das Implicit Shape Model.

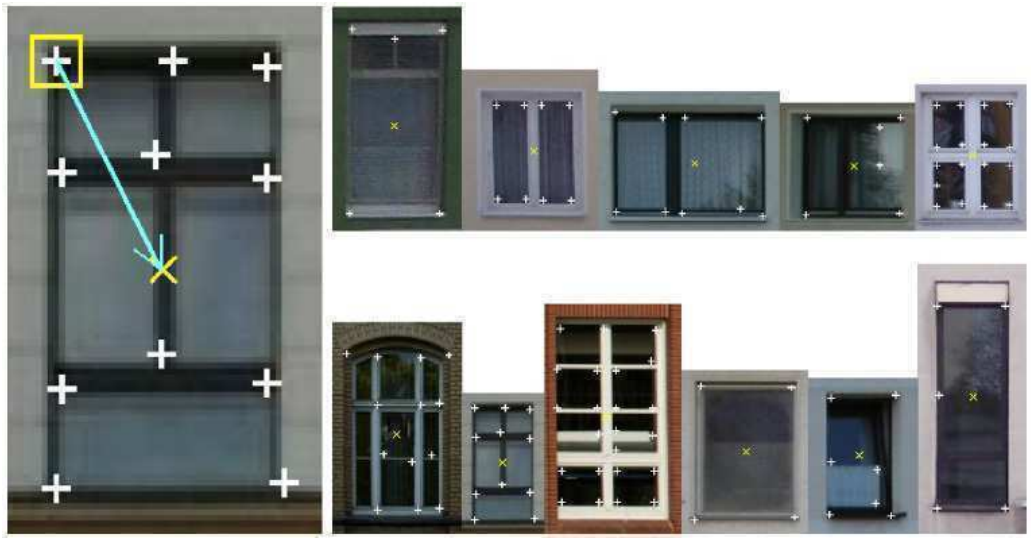


Abbildung 2.11: Förstner-Punkte und Mittelpunkte der Fenster bilden das Implicit Shape Model (aus Mayer und Reznik, 2006).

Auf der zu modellierenden Fassade werden ebenfalls die Förstner-Punkte berechnet. Mit Kreuzkorrelation werden diese mit den Implicit Shape Models verglichen. Ab einem Schwellwert wird eine Evidenz für den Mittelpunkt gespeichert. Die Summe aller Evidenzen wird geglättet und alle Maxima über einem Schwellwert bilden die Hypothesen für Fenstermittelpunkte. Diese Hypothesen dienen als Start für ein Markov Chain Monte Carlo Verfahren, welches durch Verändern der Fensterbreite, -höhe und -position das Modell der Fassade optimiert.

Da Fenster in Zeilen und Spalten oft die gleiche Struktur haben, wird in der Arbeit zusätzlich vorgeschlagen, das Verfahren auf rjMCMC so zu erweitern, dass ganze Zeilen von identischen Fenstern eingefügt oder gelöscht werden können.

In Müller u. a. (2007) wird die Struktur der Fassaden noch stärker genutzt. Es wird ein Verfahren vorgestellt, aus gering aufgelösten Luftbildern bzw. höher aufgelösten terrestrischen Bildern von Fassaden ein Gebäudemodell zu rekonstruieren. Aus dem Modell können anschließend Regeln der in Wonka u. a. (2003) und Müller u. a. (2006) entwickelten Split-Grammatik (siehe Abschnitt 3.2.4) gewonnen werden. Die Methode erzeugt aus einem Bild ein texturiertes 3D-Modell mit semantischer Struktur. Bei gering aufgelösten Bildern ist ein weiteres Ziel, eine optisch ansprechende Visualisierung zu erzeugen.

Das Verfahren analysiert zunächst die Fassadenstruktur und unterteilt die Fassade in Stockwerke und diese anschließend in Kacheln. Um die Fassadenstruktur zu untersuchen, wird die Mutual-Information als Ähnlichkeitsmaß zwischen zwei Regionen definiert. Diese verwendet die Kullback-Leibler-Distanz (Kullback, 1959) zwischen der gemeinsamen Verteilung zweier Zufallsvariablen $p(A = a, B = b)$ und dem Produkt ihrer Randverteilungen $p(A = a) \cdot p(B = b)$, wobei die Wahrscheinlichkeiten auf Intensitätshistogrammen basieren. Um die Fassade in Stockwerke und diese weiter in Kacheln zu unterteilen, werden adjazente Regionen gleicher Größe untersucht. Regionen mit maximaler Ähnlichkeit werden gewählt. Hierbei wird die Höhe der Regionen sehr gut bestimmt. Es kann jedoch vorkommen, dass die Aufteilung in Teilregionen eine Fensterreihe teilt.

Um das Problem zu beheben, wird die so genannte irreduzible Fassade bestimmt, in der keine Wiederholungen vorkommen. Wiederholte Bereiche werden durch den Mittelwert der Pixel in jedem Bereich dargestellt. In dieser irreduziblen Fassade können jetzt die Trennlinien bestimmt und auf der vollständigen Fassade fortgesetzt werden. Die Fassade ist also in Kacheln unterteilt, wobei ähnliche Kacheln einander zugeordnet werden können.

Die Kacheln werden weiter durch horizontale und vertikale Linien in Rechtecke unterteilt. Um Rauschen im Bild auszugleichen, werden die Kacheln nicht einzeln, sondern synchron behandelt. Hier gibt es lokale und globale Untersuchungen zu Unterteilungen. Im lokalen Fall werden alle Kacheln eines Typs gemeinsam untersucht. Im globalen Fall sollen alle Kacheln berücksichtigt werden. Hierbei werden zu einer Kachel für die horizontalen Trennlinien alle Kacheln in der Spalte betrachtet und für die vertikalen Linien alle Kacheln in der Zeile der Kachel.

In die so erhaltenen Regionen sollen jetzt 3D-Objekte aus einer Objektdatenbank eingesetzt werden. Hierzu werden zunächst zu den 3D-Objekten 2D-Projektionen erzeugt. Zwischen diesen und den Regionen wird wieder die Mutual-Information berechnet, um das geeignete Objekt zu finden. Hierbei werden wieder zur Unterdrückung von Rauschen alle Regionen einer Klasse gleichzeitig betrachtet.

Das erzeugte Modell ist als Baum ohne Tiefeninformation gegeben. Zu dem flachen Modell kann der Benutzer interaktiv die Tiefe der einzelnen Kacheln editieren, so dass ein texturiertes 3D-Modell entsteht. Über die Baumstruktur können abschließend die entsprechenden Regeln der Split-Grammatik abgeleitet werden.

2.3 Verfahren zur Strukturerkennung

Für die Rekonstruktion von Fassaden kann die Untersuchung ihrer Struktur ein wichtige Rolle spielen. Hier sollen einige Arbeiten zur allgemeinen Untersuchung von Strukturen vorgestellt werden. Die Erkennung von Symmetrien und Wiederholungen kann unter anderem zur Verbesserung der Daten durch eine Erhöhung der Punktdichte und das Ersetzen fehlender Teile verwendet werden.

Zur Detektion von Regelmäßigkeiten gibt es Ansätze, die auf verschiedenen Ideen basieren. Eine Möglichkeit ist es, Merkmale zu extrahieren. Ein Modell beschreibt ein Einzelteil eines regelmäßigen Musters und aus den Merkmalen werden diejenigen mit großer Ähnlichkeit zum Modell gesucht. Hierbei ist es allerdings wichtig, auffällige Merkmale im Bild zu haben. Andere Methoden verwenden Autokorrelation, die Fourier-Transformation oder Periodizitäts-Maße, die über Kookkurenz-Matrizen definiert sind.

Liu u. a. (2004) stellen einen Ansatz vor, der auf der Analyse kristallographischer Gruppen basiert. Dabei handelt es sich um eine mathematische Theorie, die periodische Muster analysiert. Für das Erkennen von

	Frieze-Muster	Punkt	Mittellinie	rechtwinklige Linie	Gleitreflexion
$\mathcal{F}_{\infty}$	LLLLLLL				
$\mathcal{F}_{\in}$	NNNNNNN	x			
$\mathcal{F}_{\ni}$	DDDDDDD		x		x
$\mathcal{F}_{\triangle}$	VVVVVVV			x	
$\mathcal{F}_{\triangledown}$	HHHHHHH	x	x	x	x
$\mathcal{F}_{/}$	VΛVΛVΛV	x		x	x
$\mathcal{F}_{\shortmid}$	LΓLΓLΓL				x

Tabelle 2.1: Frieze-Muster mit ihren Symmetrien (aus Cederberg, 2001).

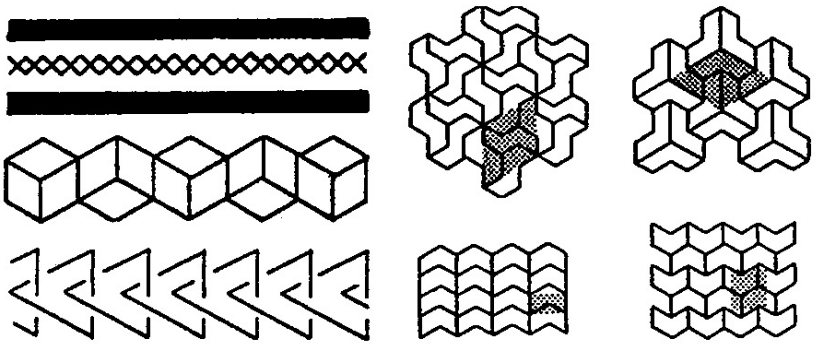


Abbildung 2.12: Einige Frieze- (links) und Wallpaper-Muster (rechts) (aus Cederberg, 2001).

Regularitäten in Bildern geht es um die Frieze-Gruppen, die Wiederholungen in einer Dimension beinhalten, und Wallpaper-Gruppen, die Wiederholungen in zwei Dimensionen enthalten.

Eine Frieze-Gruppe mit Achse c ist eine Gruppe von Transformationen, die eine Linie c unverändert lässt (Cederberg, 2001). Eine Punktmenge, die unter der Frieze-Gruppe unverändert bleibt, heißt Frieze-Muster mit Achse c und wird $\mathcal{F}_{\rfloor}$ geschrieben. Die Transformationen einer Frieze-Gruppe können neben Translationen auch andere Symmetrien sein. Dies sind Punktsymmetrien, Symmetrien entlang der Linie c, Symmetrien an Linien senkrecht zu c und so genannte Gleitreflexionen. Letztere sind Translationen entlang einer Achse mit gleichzeitiger Spiegelung an dieser Achse. Aus diesen Möglichkeiten ergibt sich, dass es sieben Frieze-Gruppen gibt. Tabelle 2.1 listet dazu die sieben Frieze-Muster mit den vorkommenden Symmetrien auf. Die Zeichen sind so gewählt, dass sie der jeweiligen Symmetrie genügen. Die Erweiterung auf zwei Dimensionen ergibt die Wallpaper-Gruppen bzw. -Muster. Die Wallpaper-Gruppen setzen sich aus Translationen entlang zweier nicht paralleler Linien zusammen. Insgesamt gibt es 17 mögliche Wallpaper-Muster. Einige Beispiele von Frieze- und Wallpaper-Mustern zeigt Abbildung 2.12.

Das Verfahren von Liu u. a. (2004) basiert auf der Peak-Erkennung aus Autokorrelation. Hierbei wird der Peak aber nicht nach seiner Höhe, sondern nach der Dominanz der Region, in der er liegt, bewertet. Dieses Vorgehen hilft dabei, wichtige Peaks am Rand von denen, die relativ zentral liegen, aber keine gute Korrelation liefern, zu trennen. Die besten der so sortierten Peaks werden als Kandidaten verwendet. Zwischen diesen müssen jetzt die kürzesten unabhängigen Translationsvektoren gefunden werden. Das wird durch eine Hough Transformation (Hough, 1962) realisiert. Hierbei kann zusätzlich die Bedingung integriert werden, dass der Winkel der beiden Vektoren zwischen 60 und 90 Grad liegt. Dieses gilt für alle Wallpaper-Gruppen.

Aus dem so gewonnenen Gitter wird dann eine Median-Kachel berechnet. Dazu wird eine Kachel nach der Gittervorschrift verschoben und anhand der Summe der quadratischen Distanzen registriert. Dann wird zu jedem Pixel der Median berechnet. Daraus ergibt sich die Median-Kachel.

Mit dieser Median-Kachel werden dann die einzelnen Symmetrie-Typen geprüft. Das sind für die Frieze-Gruppe die Rotation um 180°, die horizontale Spiegelung, die vertikale Spiegelung und die gleitende Spiegelung. Aus Kombinationen dieser Symmetrie-Typen ergeben sich sieben verschiedene Symmetrie-Gruppen. Für die Wallpaper Gruppen gibt es acht Symmetrie-Typen, die sich zu 17 Symmetrie-Gruppen kombinieren lassen. Mit Korrelation und der Summe der quadratischen Differenzen wird bestimmt, ob die Symmetrie zutrifft. Dies kann dann mit der Liste der Symmetrie-Gruppen abgeglichen werden.

Da die Symmetrien auch verschachtelt vorkommen können und nicht immer eindeutig sind, weil einige Symmetrie-Gruppen Untergruppen anderer sind, wird ein Distanzmaß zwischen einem gegebenen Muster und einem Frieze-

bzw. Wallpaper-Muster definiert. Dazu wird das geometrische Akaikes Information Criterion (AIC) (Akaike, 1973) verwendet. Hierbei werden die generellen Gruppen den spezielleren vorgezogen.

Pauly u. a. (2008) entwickeln eine Methode zur Erkennung regulärer Strukturen in Punktwolken oder Gittermodellen. Sie basiert darauf, zunächst paarweise Ähnlichkeitstransformationen zu bestimmen, und aus diesen anschließend Gitterstrukturen zu erkennen. Das resultierende Modell enthält die wiederholte Geometrie und die Transformation und Wiederholungen der Geometrie. Ein Vorteil des Verfahrens ist, dass kein Vorwissen über die Form, Größe oder Orientierung der Objekte vorhanden sein muss. Diese Werte werden automatisch während der Optimierung bestimmt.

Zu den Eingangsdaten soll ein generatives Modell erstellt werden, das möglichst große Bereiche der Eingangsdaten abdeckt, aber eine kleine Anzahl an Wiederholungen besitzt. Es werden also große Strukturen bevorzugt.

Um Kandidaten für die wiederholten Bereiche zu finden, wird ein zufälliges Sampling auf der Oberfläche durchgeführt. Der Abstand der Punkte entspricht dabei der minimalen Größe der wiederholten Struktur, die erkannt werden soll. Jedes Sample liefert einen Bereich und diese werden durch einen auf Gausskrümmung basierenden lokalen geometrischen Deskriptor, der invariant unter Ähnlichkeitstransformationen ist, in Ähnlichkeitsklassen aufgeteilt. Jede Ähnlichkeitsklasse wird getrennt weiter behandelt. Dadurch kann eine quadratische Komplexität vermieden werden. Die Ähnlichkeitsklassen werden in der Reihenfolge der Anzahl ihrer Elemente abgearbeitet.

Für die einzelnen Bereiche werden die Parameter der Transformation durch den Iterative Closest Point (ICP) Algorithmus (Besl und McKay, 1992) bestimmt und schlecht passende Paare werden verworfen. Die Transformationsparameter von gültigen Paaren werden aufgezeichnet. Hieraus soll anschließend das generative Modell der regulären Struktur geschätzt werden. Dazu werden die Translationsvektoren im Transformationsraum betrachtet (siehe 2.13 links). Um hier Regelmäßigkeiten zu finden, werden folgende Informationen ausgenutzt. Alle Transformationen einer Gruppe liegen in einer 2D-Ebene durch den Ursprung im Raum der affinen Transformationen. Zusätzlich bilden sie ein Gitter mit regelmäßigen Abständen. Abbildung 2.13 (rechts) zeigt die Transformationen in einer Ebene. In diesen muss ein regelmäßiges Gitter von Clustern gefunden werden. Probleme hierbei sind, dass auch falsche Elemente in der Ähnlichkeitsklasse vorkommen können, die das Ergebnis verfälschen. Und ebenso können einige Elemente fehlen, da sie nicht in der Punktwolke enthalten sind oder beim Sampling nicht gewählt wurden.

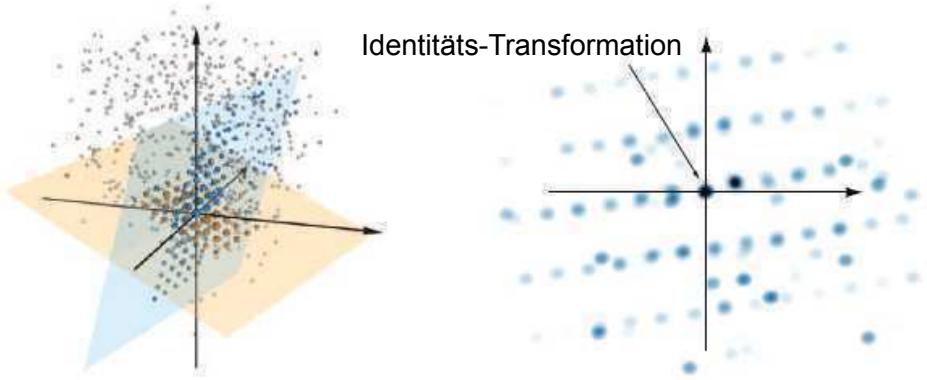


Abbildung 2.13: Translationsvektoren zwischen kompatiblem Paaren im Transformationsraum (links). Die zwei Ebenen zeigen die Transformationen der zwei dominanten regelmäßigen Strukturen. Die rechte Seite zeigt die Dichte der Vektoren in der blauen Ebene. Diese weisen die typische Gitterstruktur auf (aus Pauly u.a., 2008).

Da die Identität als Transformation immer vorhanden ist, ist bekannt, dass das gesuchte Gitter durch den Ursprung gehen muss. Die fehlenden Parameter lassen sich dann durch Minimierung einer Funktion aus vier Energiefunktionen schätzen. Diese ergeben sich aus den quadratischen Distanzen von Gitter zu Cluster und von Cluster zu Gitter sowie zwei Funktionen, die die Anzahl gültiger Korrespondenzen zwischen Gitter und Cluster erhöhen sollen. Dies wird durch das Gauß-Newton-Verfahren gelöst.

In einem abschließenden Schritt werden die Ergebnisse durch eine Aggregation zusammengeführt. Dabei wird eine gleichzeitige Registrierung ausgeführt, da es im kleinen Maßstab zu Ungenauigkeiten kommen kann, die im großen Maßstab ausgeglichen werden müssen.

Bei den bisher verwendeten Verfahren wird das Transformations-Voting eingesetzt, es werden also die am höchsten bewerteten Transformationen gewählt. Bei mehreren überlagerten Symmetrien ist es schwer, diese

voneinander abzugrenzen, denn die einzelnen Transformationen können sich überlagern und bilden dann keine regelmäßige Gitterstruktur mehr. Deshalb eignen sich diese Verfahren eher für großmaßstäbige Symmetrien. Um auch kleinere verschachtelte Symmetrien zu erkennen, stellen Bokeloh u. a. (2009) eine weitere Methode zur Symmetrieerkennung in Punktwolken vor. Diese basiert auf Linienmerkmalen und ermöglicht es, viele Symmetrien im Datensatz zu erkennen und zu unterscheiden.

Das hier verwendete Verfahren arbeitet mit Linienmerkmalen, die aus den Daten extrahiert werden. Auf diesen wird ein Nachbarschaftsgraph berechnet. Anschließend werden zunächst die Symmetrien auf den Linienmerkmalen bestimmt. Die so gefundenen Symmetrien werden dann noch einmal anhand der gesamten Punktwolke geometrisch verifiziert. Der Vorteil dieses Verfahrens ist, dass der Vergleich der Linienmerkmale viel effizienter ist als ein Vergleich der gesamten Daten.

Das Verfahren besteht im Wesentlichen aus drei Schritten: der Detektion von Merkmalen, der Zuordnung der Linien und der geometrischen Verifikation. Zuerst werden Linien detektiert und passende Merkmale zu so genannten Basen zusammengesetzt. Im zweiten Schritt wird nach weiteren Vorkommen von Teilen dieser Basen in der Nachbarschaft gesucht. Um hier passende Linien zu finden, wird ein Iterative Closest Line Algorithmus verwendet. Da die Genauigkeit bei der Zuordnung der Linienmerkmale nicht so hoch ist, muss das Ergebnis noch anhand der Punktwolke validiert werden. Diese beiden Schritte der Linienzuordnung und der Validierung werden in einer Schleife für jeden Kandidaten wiederholt.

Stehen alle Kandidaten fest, so wird das Ergebnis noch einmal verfeinert, indem in einem weiteren Zuordnungsschritt zusätzliche Instanzen der Symmetrie gesucht werden. Hierzu werden weitere Basen gesucht, indem Überlappungen mit den berechneten Punkten aus dem Validierungsschritt überprüft werden. Anschließend wird erneut eine geometrische Validierung durchgeführt. Das Ergebnis dieses Schrittes ist eine Symmetrie mit all ihren Instanzen. Der gesamte Prozess wird mehrfach ausgeführt, um verschiedene Klassen von Symmetrien zu finden.

3 Einführung in die Modellierung mit formalen Grammatiken

3.1 Theoretische Grundlagen formaler Grammatiken

3.1.1 Formale Grammatiken

Formale Grammatiken sind ein Konstrukt der theoretischen Informatik, Sprachen zu definieren, die automatisch verarbeitet werden können (Hedtstück, 2004). Noam Chomsky definierte sie und strukturierte sie in der nach ihm benannten Chomsky-Hierarchie (Chomsky, 1956). Formale Sprachen betrachten die Syntax und gehen nicht auf die Semantik der Sprache ein. Hier wird zunächst der Begriff der Sprache und dann der der Grammatik definiert. Die Bezeichnungen lehnen sich an das Vokabular der natürlichen Sprachen an.

Definition 3.1

Ein **Alphabet** V ist eine endliche, nichtleere Menge von Symbolen. Eine endliche Folge von Symbolen $x_1 \dots x_k$ mit $x_i \in V$ für $i = 1, \dots, k$ heißt **Wort** über V der Länge k. Das Wort der Länge 0 wird mit ϵ bezeichnet. V^* ist die Menge aller Wörter über V und $V^+ := V^* \setminus \{\epsilon\}$ die Menge aller nichtleeren Wörter über V. Jede Teilmenge von V^* wird als **formale Sprache** bezeichnet.

Um die Menge der formalen Sprachen zu strukturieren, teilt man sie in Klassen auf. Hierzu werden die Begriffe Aufzählbarkeit, Abzählbarkeit und Entscheidbarkeit verwendet.

Definition 3.2

Eine Menge M heißt **abzählbar**, wenn es eine surjektive Funktion $F : \mathbb{N} \to M$ gibt oder M leer ist. Ist M nicht abzählbar, so heißt sie **überabzählbar**.

Definition 3.3

Eine Menge M heißt **aufzählbar** (auch rekursiv aufzählbar oder semi-entscheidbar), wenn es eine surjektive Funktion $F : \mathbb{N} \to M$ und einen Algorithmus, der für jedes $n \in \mathbb{N}$ den Funktionswert $f(n)$ berechnet, gibt oder wenn M leer ist. Die Folge $f(0), f(1), \dots$ heißt dann **Aufzählung** von M.

Definition 3.4

Eine Sprache $L \subseteq V^*$ heißt **entscheidbar**, wenn es einen abbrechenden Algorithmus gibt, der für jedes $v \in V$ feststellt, ob $v \in L$ gilt oder nicht.

Aufzählbarkeit ist also eine Einschränkung der Abzählbarkeit. Hierbei wird zusätzlich gefordert, dass ein Algorithmus existiert, der die Abfolge der Elemente konstruiert. Die Entscheidbarkeit gibt an, ob in endlicher Zeit gesagt werden kann, ob ein Wort zu der Sprache gehört. Die drei Sprachklassen bilden die folgende Hierarchie: Die entscheidbaren Sprachen sind eine Teilmenge der aufzählbaren Sprachen. Diese sind wiederum Teilmenge der abzählbaren Sprachen:

$$\text{entscheidbare Sprachen} \subset \text{aufzählbare Sprachen} \subset \text{abzählbare Sprachen}$$

Eine Möglichkeit Sprachen zu beschreiben, bilden Grammatiken. Sie sind ein Regelwerk, das die Struktur einer Sprache wiedergibt.

Definition 3.5

Eine **Grammatik** ist ein Quadrupel $G = (N, \Sigma, P, S)$, wobei N und Σ endliche, nichtleere, disjunkte Mengen sind. N ist die Menge der nichtterminalen Symbole und Σ die Menge der terminalen Symbole. P ist eine endliche Menge von Grammatikregeln der Form $\alpha \to \beta$, wobei $\alpha \in (N \cup \Sigma)^* N (N \cup \Sigma)^*$ und $\beta \in (N \cup \Sigma)^*$.

Eine Grammatik heißt in **Normalform**, wenn für alle Regeln gilt $\alpha \in N^+$. Die Regeln einer Grammatik werden auch **Produktionen** genannt. $S \in N$ ist das Startsymbol.

Beim Erzeugen von Wörtern mit einer Grammatik werden Symbole mittels der Grammatikregeln schrittweise ersetzt.

Definition 3.6

Seien $x, y \in (N \cup \Sigma)^*$, dann sagt man y ist **direkt ableitbar** aus x ($x \Rightarrow y$), genau dann, wenn $x = \gamma\alpha\delta$, $y = \gamma\beta\delta$ und $\alpha \to \beta \in P$, wobei $\gamma, \delta \in (N \cup \Sigma)^*$.

Werden mehrere Ableitungsschritte benötigt, so sagt man y ist aus x **ableitbar** ($x \overset{*}{\Rightarrow} y$), genau dann, wenn es eine endliche Folge $(w_0, \ldots, w_n)$ gibt mit $x = w_0$ und $y = w_n$ und für alle $i = 1, \ldots, n$ w_i direkt aus w_{i-1} ableitbar ist. Die Folge heißt Ableitung von y aus x und wird so geschrieben: $w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_n$.

Definition 3.7

Die von einer Grammatik erzeugte **Sprache** ist die Menge aller aus dem Startsymbol S ableitbaren Wörter, die nur aus terminalen Symbolen bestehen. $L(G) = \{x \in \Sigma^* | S \overset{*}{\Rightarrow} x\}$.

Zwei Grammatiken G_1 und G_2 heißen **äquivalent**, wenn sie die gleiche Sprache erzeugen, d.h. wenn gilt $L(G_1) = L(G_2)$.

Die Grammatiken lassen sich in verschiedene Klassen einteilen. Noam Chomsky entwickelte eine Hierarchie verschiedener Grammatiktypen.

Definition 3.8

Typ-0-Grammatiken sind die Grammatiken, die wir bisher betrachtet haben ohne irgendeine Einschränkung.

Typ-1-Grammatiken oder auch **kontextsensitive** heißen die Grammatiken, deren Produktionen die Form $\alpha_1 A \alpha_2 \to \alpha_1 \beta \alpha_2$ haben, wobei $A \in N$, $\alpha_1, \alpha_2 \in (N \cup \Sigma)^*$ und $\beta \in (N \cup \Sigma)^+$. Zusätzlich kann es die Produktion $S \to \epsilon$ geben, wenn S auf keiner rechten Seite eine Produktion vorkommt.

Die Regeln der **Typ 2** oder **kontextfreien Grammatiken** haben die Form $A \to \beta$ mit $A \in N$ und $\beta \in (N \cup \Sigma)^+$. Für das leere Wort ϵ gilt die gleiche Ausnahmeregelung wie für Typ 1 Grammatiken.

Typ-3-Grammatiken, **reguläre** oder **rechtslineare Grammatiken** haben Produktionen der Form $A \to aB$ oder $A \to a$ mit $A, B \in N$ und $a \in \Sigma$. Zusätzlich gibt es die Ausnahmeregelung wie bei Typ 1.

Der Typ einer Sprache ist durch den Typ der erzeugenden Grammatik bestimmt. Eine Sprache L ist also vom Typ i, $(i = 0, 1, 2, 3)$, wenn es eine Grammatik G vom Typ i gibt mit $L(G) = L$.

Sprachen bzw. Grammatiken eines speziellen Typs sind gleichzeitig auch vom allgemeineren Typ. So gilt, dass für $0 \leq i \leq j \leq 3$ eine Grammatik bzw. Sprache vom Typ j gleichzeitig auch vom Typ i ist. Das ergibt eine Hierarchie der Sprachklassen, die Chomsky-Hierarchie genannt wird.

Des Weiteren unterscheidet man zwischen deterministischen und nicht deterministischen Grammatiken.

Definition 3.9

Eine Grammatik vom Typ 2 oder 3 heißt **deterministisch**, wenn es für jedes nichtterminale Symbol A nur eine Regel $A \to \beta$ mit $\beta \in V^*$ gibt und somit die Ableitung eindeutig ist. Ansonsten heißt die Grammatik nicht deterministisch.

Bei nicht deterministischen Grammatiken besteht die Möglichkeit, jeder Regel eine Wahrscheinlichkeit zu geben, mit der sie angewendet wird.

Definition 3.10

Eine **stochastische Grammatik** ist eine Grammatik $G = (N, \Sigma, P, S)$, bei der jeder Regel $\alpha \to \beta \in P$ eine Wahrscheinlichkeit $p(\alpha \to \beta)$ zugeordnet ist, mit der die linke Seite α durch β ersetzt wird. Für jedes Symbol α gilt $p(\alpha \to \beta) \geq 0$ und $\sum_{\beta : p(\alpha \to \beta) \in P} p(\alpha \to \beta) = 1$.

Des weiteren ist es möglich, den Symbolen kontextfreier Grammatiken Eigenschaften zu geben. Knuth (1968) definierte dafür die attributierten Grammatiken.

Definition 3.11

Eine **attributierte Grammatik** ist eine kontextfreie Grammatik $G = (N, \Sigma, S, P)$, bei der allen Symbolen $X \in N \cup \Sigma$ eine Menge $A(X)$ von Attributen zugeordnet ist. Diese Menge teilt sich in die disjunkten Mengen der **synthetisierten Attribute** $A_0(X)$ und der **geerbten Attribute** $A_1(X)$. Das Startsymbol hat keine geerbten Attribute und die terminalen Symbole besitzen keine synthetisierten, d.h. die Mengen $A_1(S)$ und $A_0(X)$ für $X \in \Sigma$ sind leer. Jedes Attribut α hat eine möglicherweise unendliche Wertemenge V_α.

G habe m Produktionsregeln und die p-te Regel sei $X_{p0} \to X_{p1}X_{p2}\dots X_{pn_p}$ mit $n_p \geq 0, X_{p0} \in N$ und $X_{pj} \in N \cup \Sigma$ für $1 \leq j \leq n_p$. **Semantische Regeln** sind Funktionen $f_{pj\alpha}$ für alle $1 \leq p \leq m, 0 \leq j \leq n_p$ und $\alpha \in A_0(X_{pj})$, wenn $j = 0$, $\alpha \in A_1(X_{pj})$, wenn $j > 0$. Jede dieser Funktionen ist eine Abbildung von $V_{\alpha_1} \times V_{\alpha_2} \times \dots \times V_{\alpha_t}$ nach V_α für ein $t = t(p, j, \alpha) \geq 0$, wobei jedes $\alpha_i = \alpha_i(p, j, \alpha)$ ein Attribut von einem X_{pk_i} ist, mit $0 \leq k_i = k_i(p, j, \alpha) \leq n_p, 1 \leq i \leq t$.

3.1.2 Lindenmayer-Systeme

Zur Modellierung von Pflanzen werden häufig Lindenmayer-Systeme (Prusinkiewicz und Lindenmayer, 1990) verwendet. Der Hauptunterschied zu den vorher beschriebenen Grammatiken besteht in der Art, in der Ableitungen gebildet werden. Während die oben beschriebene Ableitung symbolweise abläuft, geschieht dies bei Lindenmayer-Systemen streng parallel, d.h. alle Symbole werden gleichzeitig abgeleitet. Zusätzlich entfällt die Unterscheidung in terminale und nichtterminale Symbole.

Definition 3.12

Ein **Lindenmayer-System (**L-System) ist durch ein Tupel $G = (V, P, w)$ gegeben, wobei V ein Alphabet, $P \subseteq V \times V^*$ eine Menge von Produktionen und $w \in V^+$ das Startwort (Axiom) ist. Für die Menge der Produktionen muss für jedes $a \in V$ ein $(a, \alpha) \in P$ mit $\alpha \in V^+$ existieren. Anstelle von $(a, \alpha) \in P$ schreibt man üblicherweise $a \to \alpha$.

Für $x, y \in V^*$ mit $x = a_1 \dots a_r$, $a_i \in V$, $i \in [1, \dots, r], r \geq 0$ heißt y aus x ableitbar ($x \Rightarrow_G y$), wenn es Produktionen $a_i \to \alpha_i$ mit $y = \alpha_1, \dots, \alpha_r$ gibt. Wenn klar ist, um welches L-System es sich handelt, schreibt man auch $x \Rightarrow y$.

Definition 3.13

Sei $G = (V, P, w)$ ein 0L-System und $x \in V^*$. Dann ist $L_0(G, x) := \{x\}$ die Menge der Wörter, die sich in 0 Schritten aus x ableiten lassen. Rekursiv ist dann $L_{n+1}(G, x) := \{z : \exists y \in L_n(G, x) \wedge y \Rightarrow z\}$ definiert.

Es werden definiert

$$x \Rightarrow_G^n y :\Leftrightarrow y \in L_n(G, x),$$
$$x \Rightarrow_G^+ y :\Leftrightarrow \exists n \geq 1 : y \in L_n(G, x) \text{und}$$
$$x \Rightarrow_G^* y :\Leftrightarrow \exists n \geq 0 : y \in L_n(G, x).$$

Die von G erzeugte **Sprache** ist dann $L(G) = \{x : w \Rightarrow_G^* x\} = \bigcup_{n=0}^{\infty} L_n(G)$. Für $x = w$ schreibt man auch einfach $L_n(G)$ statt $L_n(G, x)$.

Mithilfe der so genannten Turtle Graphik (Prusinkiewicz und Lindenmayer, 1990) lassen sich mit L-Systemen Bilder zeichnen. Hierzu werden die Symbole des L-Systems als Zeichenanweisungen interpretiert. Der Zeichenstift hat einen Zustand (x, y, α), wobei (x, y) die Position und α die Richtung angibt. Das System hat zusätzlich zwei Parameter, die Schrittweite d und die Drehweite δ. Das Alphabet des L-Systems besteht aus den Zeichen $F, f, +$ und $-$. F steht für eine Vorwärtsbewegung des Stiftes, bei der eine Linie von der alten Position (x, y) zur neuen (x', y') gezogen wird. Der neue Zustand ist (x', y', α) mit $x' = x + d\cos\alpha$ und $y' = y + d\sin\alpha$. Das Zeichen f steht für die gleiche Stiftbewegung ohne eine Linie zu zeichnen. $+$ und $-$ verändern die Richtung des Stiftes und setzten den neuen Zustand auf $(x, y, \alpha + \delta)$ bzw. $(x, y, \alpha - \delta)$.

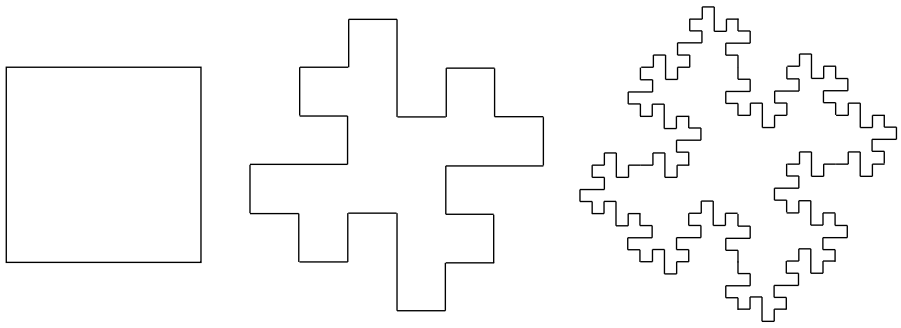


Abbildung 3.1: Turtle-Interpretation des Axioms und der ersten beiden Ableitungsschritte eines L-Systems.

Mit diesem System erhält man ein Quadrat durch die Zeichenfolge $F + F + F + F+$, wenn $\delta = 90°$ ist. Ersetzt man jede Linie F durch $F + F - F - FF + F + F - F$, so erhält man die mittlere Zeichnung in Abbildung 3.1. Die Regeln des L-Systems sind $P = \{F \rightarrow F + F - F - FF + F + F - F, + \rightarrow +, - \rightarrow -\}$. Der zweite Ableitungsschritt ist in der rechten Zeichnung in Abbildung 3.1 zu sehen.

3.2 Beispielanwendungen und Erweiterungen formaler Grammatiken

3.2.1 Strukturbeschreibung mit formalen Grammatiken

Alegre und Dallaert (2004) stellen ein Verfahren zur Fassadenmodellierung aus Bilddaten mithilfe einer stochastischen kontextfreien Attribut-Grammatik vor. Anhand der Grammatik werden Partitionierungen der Fassade erzeugt, die auf Basis von Intensitätswerten im Eingabebild bewertet werden. Mit einem Markov Chain Monte Carlo (MCMC)-Verfahren (siehe Abschnitt 4.2) wird die Partitionierung bestimmt.

Die Grammatik unterteilt die Fassade in rechteckige Blöcke konstanter Farbe. Diese Vorgehensweise ist insbesondere für moderne Bürogebäude geeignet, in denen die Fenster strikt in Zeilen und Spalten angeordnet sind. Für die Rekonstruktion wählt der Benutzer anhand des Fassadenfotos eine passende Grammatik aus. Die Grammatik beinhaltet zum einen Wissen über Fassaden, zum anderen kann sie die Detailgenauigkeit des Ergebnisses beeinflussen.

Die Symbole der Grammatik sind Regionen und Operatoren. Regionen sind rechteckige Gebiete, die Form, Farbe, Textur und das Aussehen beschreiben. Operatoren stellen verschiedene Wege dar, Regionen in Teilregionen zu unterteilen. Sie dürfen in einer Regel nur an erster Stelle der rechten Seite stehen. Alle anderen Symbole in einer Regel sind Regionen.

Es gibt zwei Arten von Produktionsregeln. Die erste ist die Kopier-Regel, sie gibt einer Region ein neues Label. Die zweite Art ist die Unterteilungs-Regel. Hiervon gibt es die Regeln *split* und *div* jeweils in horizontale und vertikale Richtung. Der *split*-Operator unterteilt die Region in eine feste Anzahl an Blocks, wohingegen der *div*-Operator den zur Verfügung stehenden Raum mit einer variablen Anzahl an Blocks auffüllt.

Bei der Grammatik handelt es sich um eine attributierte Grammatik. Die Attribute, die für das Verfahren verwendet werden, beziehen sich auf den Stil und die Geometrie der Regionen. Der Stil wird in jedem Ableitungsschritt an die Kinder weitergegeben, ist also in jedem Knoten gleich. Deshalb muss er auch nicht in die Grammatik aufgenommen werden. Die geometrischen Attribute sind in den Operator-Knoten enthalten. So genannte semantische Regeln legen die Parameter für die Operatoren fest. Der *split*-Operator beinhaltet die Verhältnisse, die besagen, wie die gesamte Region unterteilt wird. Der *div*-Operator enthält neben den Verhältnissen noch die Anzahl an Teilblöcken, in die unterteilt werden soll.

Die gesamte Bildpartitionierung läuft in zwei Schritten ab. Im ersten wird durch die Produktionsregeln der Ableitungsbaum aufgestellt. Im zweiten legen die semantischen Regeln die Parameter der Operatoren fest.

Mitchell (1990) beschreibt die in der Architektur verwendeten Strukturen und formuliert Regeln für diese. Er entwickelt beispielsweise Grammatiken für Kombinationen aus Tischen der Form eines halben Sechsecks, aber auch für kompliziertere Strukturen wie Grundrisse paladianischer Villen. Bei der Anwendung dieser Grammatiken zur Beschreibung von vorhandenen Strukturen werden die Ableitungen meist manuell vorgenommen, da die Regeln zu komplex für eine automatische Modellierung sind.

3.2.2 Stadtmodellierung mit Lindenmayer-Systemen

Parish und Müller (2001) verwenden L-Systeme zur Modellierung von Städten. Das entwickelte Programm *CityEngine* arbeitet ohne die Eingabe von Bild- oder Entfernungsdaten und generiert aus einer Auswahl aus verschiedenen geographischen und statistischen Datensätzen, wie z.B. einer Wasserkarte, einem Höhenmodell oder einer Karte der Bevölkerungsdichte, ein Stadtmodell mit Straßen und Gebäuden. Dies geschieht in drei Schritten. Zunächst wird ein Straßennetz generiert. Basierend hierauf werden anschließend Grundstücke für die Gebäude erzeugt. Im dritten Schritt wird für jedes Grundstück ein Gebäude konstruiert und abschließend texturiert.

CityEngine verwendet zwei verschiedene Lindenmayer-Systeme. Das erste wird für die Generierung des Straßennetzes verwendet, während das andere Gebäude verschiedenen Typs entwirft. Für die Erstellung solcher komplexen Szenen wird eine große Anzahl an Parametern und Bedingungen im L-System benötigt. Bei Änderungen am L-System müssten schlimmstenfalls Änderungen in allen Regeln vorgenommen werden. Um dem entgegenzuwirken, erweitern Parish und Müller (2001) das Konzept von L-Systemen. Sie verwenden statt der Regeln generische Templates, so genannte *ideal Successors*. Die Parameter dieser Templates werden über externe Funktionen verändert. Diese werden über globale Ziele und lokale Bedingungen bestimmt. Die globalen Ziele bewirken, dass die Straßen nach typischen Straßen-Mustern aufgebaut werden. Lokale Bedingungen überprüfen dann, ob die Straßen auch in gültigen Bereichen liegen und untersuchen Schnittpunkte von Kanten. Liegen die Straßen nicht in gültigen Bereichen, also z.B. im Wasser, so kann dies über Änderung der Länge oder des Winkels behoben werden. Bei der Schnittuntersuchung werden Anordnungen von Straßen korrigiert, bei denen Straßen sich nicht ganz schneiden, eine Straße über die andere hinaus ragt oder zwei T-Kreuzungen sehr dicht beieinander liegen. Die Straßen werden leicht verändert, um eine korrekte Kreuzung herzustellen.

Das Straßennetz liefert eine Unterteilung des Gebietes in Blöcke. Diese werden unter der Bedingung, dass die Flächen meist konvex und rechtwinklig sind, weiter unterteilt bis eine von Benutzer definierte Größe unterschritten ist. Aus den entstandenen Parzellen werden die zu kleinen und die, die keinen Zugang zu einer Straße haben, herausgefiltert. Die restlichen Parzellen dienen als Grundrisse für die Gebäude.

Um aus den Grundrissen Gebäude zu erzeugen, wird ein weiteres L-System erstellt. Dies verändert den Grundriss durch die verschiedenen Module für Transformation, Extrusion, Verzweigung und Terminierung und beinhaltet verschiedene Templates für Dächer und Dachaufbauten wie z.B. Antennen. In der *CityEngine* gibt es die Gebäudetypen Wolkenkratzer, Gewerbegebäude und Wohnhaus, die je nach Lage des Gebäudes gewählt werden. Für jeden der drei Typen wird ein unterschiedlicher Regelsatz des L-Systems verwendet. Für die abschließende Texturierung werden die Fassaden der Gebäude in möglicherweise verschachtelte Gitter unterteilt und die einzelnen Zellen werden texturiert.

Marvie u. a. (2005) stellen eine Erweiterung der L-Systeme vor, mit der komplexe Stadtmodelle erzeugt werden können, die eine Vielzahl verschiedenartiger Gebäude und auch Vegetation und andere städtische Elemente wie z.B. Straßenlaternen enthalten. Ein weiteres Ziel war, komplexe Modelle mit geringem Platzbedarf zu speichern und schnell wieder rekonstruieren zu können.

Das L-System wird zu einem FL-System erweitert, indem die terminalen Symbole durch Funktionen ersetzt werden. Diese Funktionen erlauben es, VRML-Szenen-Graphen und Geometrien zu erzeugen. Es gibt zwei Arten von Funktionen. Die terminalen Funktionen erzeugen oder modifizieren den Inhalt von Objekten. Sie geben keinen Wert zurück. Parameter-Funktionen erzeugen ein Objekt und geben eine Referenz darauf zurück.

Um ein FL-System auf 3D-Modelle anzuwenden, werden zwei Erweiterungen vorgenommen. Die erste ist die algebraische Erweiterung, die terminale Funktionen wie Translation, Rotation, Skalierung oder auch die Matrixmultiplikation beinhaltet. Die zweite Erweiterung ist die VRML-Erweiterung. Sie beinhaltet zwei Funktionen für das Erzeugen und Verändern von VRML-Knoten. Diese zwei Funktionen erlauben es, einen VRML-Szenengraphen zu erzeugen.

Damit kann ein FL-System zur Erzeugung von Gebäuden generiert werden. Es erzeugt verschiedene Gebäude eines Gebäudetyps, wobei die Veränderungen durch unterschiedliche Parameter des Axioms (Grundriss, Anzahl der Stockwerke und Höhe) oder durch stochastische Wahl der Regeln hervorgerufen werden können. Beim Entwurf des FL-Systems werden zunächst die terminalen Elemente definiert. Diese sind Fassadenelemente wie Fenster und Türen. Anschließend werden die Produktionen bestimmt, welche aus Dekomposition, wie der Unterteilung in Stockwerke, und Wachstum, wie dem Erstellen eines Volumens aus dem Grundriss eines Gebäudes, bestehen.

Die so entstandene Grammatik kann in zwei Richtungen optimiert werden. Eine Grammatik wird für das Real-Time-Rendering eingesetzt und die andere erzeugt Modelle, die sehr kompakt gespeichert werden können.

Mit den oben beschriebenen Erweiterungen können L-Systeme für die Modellierung von Gebäuden verwendet werden. In ihrer Grundfunktion eignen sie sich jedoch nicht so gut dafür, denn die Struktur von Gebäuden unterscheidet sich grundsätzlich von der von Pflanzen oder Straßen. Pflanzen wachsen in den freien Raum, während die Modellierung von Gebäuden eher eine Partitionierung als ein Wachstumsprozess ist. Besser geeignet sind daher Erweiterungen der formalen Grammatiken auf höhere Dimensionen, welche im Folgenden dargestellt werden.

3.2.3 Die Shape-Grammatik als Erweiterung formaler Grammatiken auf höhere Dimensionen

Eine Erweiterung der bisher zeichenbasierten Grammatiken auf zwei- und mehrdimensionale Formen veröffentlichten Stiny und Gips (1972). Die von ihnen vorgestellten Shape-Grammatiken bilden eine generative Methode, „gute“ Kunst in Form von Bildern oder Skulpturen zu erzeugen und ebenso Verständnis dafür zu entwickeln, was „gute“ Kunst ist.

Eine Klasse von Bildern ist hierbei gegeben durch ein Tupel (S, M), wobei S eine Shape-Grammatik und M das Material ist.

Definition 3.14

Eine **Shape-Grammatik** ist ein Quadrupel $SG = (V_T, V_M, R, I)$, wobei V_T und V_M zwei endliche Mengen von Shapes sind, für die gilt: $V_T^* \cap V_M = \emptyset$. V_T enthält die terminalen Symbole und V_M enthält so genannte Marker, die kennzeichnen, an welcher Position Symbole ersetzt werden können. R ist eine Menge von Paaren (u, v), wobei u ein Shape-Element aus V_T^* kombiniert mit einem Element aus V_M ist. Für v ergeben sich drei verschiedene Möglichkeiten:

(A) v besteht aus dem Element aus V_T^*, das in u enthalten ist,

(B) v besteht aus dem Element aus V_T^*, das in u enthalten ist, kombiniert mit einem Element aus V_M,

(C) v besteht aus dem Element aus V_T^*, das in u enthalten ist, kombiniert mit einem weiteren Element aus V_T^* und einem Element aus V_M.

Das Startshape I besteht aus Elementen von V_T^* und V_M.

Elemente aus V_T^* werden aus Elementen aus V_T gebildet. Diese dürfen beliebig oft vorkommen und beliebig skaliert und rotiert werden. Auch hier gibt es die Begriffe der terminalen und nicht terminalen Symbole. Elemente aus V_T^*, die in $(u, v) \in R$ oder in I auftreten, werden terminale Shape-Elemente genannt. Die Elemente von V_M werden nichtterminale Shape-Elemente oder Marker genannt. (u, v) aus R werden Shape-Regeln genannt und auch $u \rightarrow v$ geschrieben. Der Ableitungsprozess beginnt mit dem Startshape I, welches üblicherweise ein u enthält, für das es ein (u, v) in R gibt. Auf I werden die Shape-Regeln angewandt. Eine Regel $u \rightarrow v$ kann ausgeführt werden, wenn die linke Seite u im Shape vorkommt. Hierzu wird zuerst nach einem zur linken Seite gleichen Teil gesucht, wobei Skalierung, Rotation, Translation und Spiegelung erlaubt sind. Dann wird die Transformation bestimmt und die linke Seite wird durch den mit diesen Parametern transformierten rechten Teil ersetzt. Da in den Shape-Regeln die terminalen Symbole auf der linken und rechten Seite gleich sind, kann ein einmal gesetztes terminales Symbol nicht wieder entfernt werden. Die Ableitung ist dann zu Ende, wenn keine Regel mehr angewendet werden kann.

Definition 3.15

Die **Sprache** $L(SG)$, die von einer Shape-Grammatik definiert wird, ist eine Menge von Shapes, die von SG erzeugt werden und kein nicht terminales Symbol mehr enthalten.

Hier kann man noch weiter differenzieren und das Prinzip der Selektionsregeln einführen. Dieses basiert darauf, den terminalen Shape-Elementen Stufen zuzuweisen. Das passiert während der Ableitung nach folgendem Schema. Zuerst bekommen alle terminalen Elemente des Startsymbols Stufe 0 zugewiesen. Wird eine Regel angewandt, so sei die höchste Stufe eines Elementes der linken Seite der Regel N. Ist die Regel vom Typ (A), so wird jedem vom Marker umgebenen Element die Stufe N zugeordnet. Ist die Regel vom Typ (B), so wird jedes vom Marker umgebene Element auf der linken Seite mit N gekennzeichnet und jedes vom Marker umgebene Element auf der rechten Seite mit der Stufe $N + 1$. Bei Regeln vom Typ (C) wird den neu hinzugefügten Elementen Stufe $N + 1$ zugewiesen.

Mit dieser Regel können Elementen auch mehrere Stufen zugeordnet werden. Eine Selektionsregel ist jetzt ein Paar (m, n), wobei m die minimale Stufe und n die maximale Stufe bezeichnet. Zusätzlich kann mit Malregeln in Abhängigkeit der Stufe der Elemente definiert werden, wie welches Gebiet dargestellt werden soll.

3.2.4 Gebäudemodellierung mit einer Split-Grammatik

Eine räumliche Grammatik für die Beschreibung von Gebäuden wird in Wonka u. a. (2003) vorgestellt. Diese ist eine dreidimensionale Design-Grammatik, die das gesamte Gebäude in einem betrachtet.

Definition 3.16

Ein **Shape** ist eine begrenzte Zusammensetzung aus geraden Linien im dreidimensionalen euklidischen Raum.

Ein **beschriftetes Shape** $< s, P >$ besteht aus einem Shape s und einer Liste von beschrifteten Punkten, wobei ein beschrifteter Punkt $p : A$ ein mit einem Symbol A versehener Punkt p ist.

Definition 3.17

Ein **Basis-Shape** $b =< s, P, V >$ besteht aus einem einfachen Shape, das im Ursprung zentriert ist, einer Menge von drei beschrifteten Punkten und einem Symbol V. Die drei Punkte stellen die positiven Schnittpunkte mit den Koordinatenachsen dar.

In der Arbeit wird zunächst eine generelle Grammatik definiert. Diese ist allgemeiner als die Zeichenketten-Grammatiken aus Abschnitt 3.1.1 und wird über einer Algebra $< u, +, -, F, \leq >$ von Objekten definiert, die abgeschlossen unter den Operationen + und − und den Transformationen F ist. $f(u) \leq v$ bedeutet, dass u in v auftritt.

Definition 3.18

Eine **allgemeine Grammatik** ist ein Viertupel $G = (N, T, R, I)$, wobei $N, T, I \subset U$ gilt und $R \subseteq U \times U$ eine oder mehrere Regeln enthält. Eine Regel $a \rightarrow b$ kann auf u angewendet werden, wenn es ein $f \in F$ und eine Zuweisung g mit $f(g(a)) \leq u$ gibt. Unter Anwendung der Regel wird ein Objekt $v = (u - f(g(a))) + f(g(b))$ erzeugt.

Definition 3.19

Eine **Set-Grammatik** über dem Alphabet B ist eine Grammatik mit $U = \mathcal{P}(\mathcal{B})$, der Potenzmenge von B. + ist die Vereinigung, − die Mengendifferenz und $\leq$ die Teilmengenrelation. Transformationen für Teilmengen $S \subseteq B$ sind folgendermaßen definiert: $f(S) = \{f(s)|s \in S\}$. Es werden also die enthaltenen Shapes transformiert.

Die bei Wonka u. a. (2003) verwendete Split-Grammatik ist eine Set-Grammatik über dem Alphabet $B = \{f(b)|b \text{ ist Basis-Shape}, f \in F\}$. Ein Split ist hierbei die Zerlegung eines Basis-Shapes in Shapes aus B. Die Split-Grammatik enthält die folgenden zwei Regeltypen.

1. Split
   $a \rightarrow b$: a ist eine verbundene Teilmenge von B und b enthält die gleichen Elemente wie a außer einem, das zerteilt wird.
2. Umwandlung
   $a \rightarrow b$: a ist eine verbundene Teilmenge von B, die ein Basis-Shape enthält. b enthält dieselben Elemente wie a, nur das Basis-Shape wird durch ein anderes ersetzt. Dabei muss das Volumen von b in a hineinpassen.

Ein Unterschied der zwei Regelarten ist also, dass bei einem Split das gleiche Volumen wieder ausgefüllt wird, was bei einer Umwandlung nicht der Fall sein muss. Hier kann z.B. aus einem Quader eine Dachform geschnitten werden.

Eine Menge von Shapes $B' \subseteq B$ abgeleitet mit der Split-Grammatik heißt Gebäudeentwurf, wenn in ihr nur terminale Elemente enthalten sind, das heißt allen $b \in B'$ ist ein $V \in T'$ zugeordnet.

Die Shapes haben zusätzlich noch Attribute, die ihr Aussehen bestimmen und gleichzeitig den Ableitungsprozess steuern. Zusätzlich zu der Split-Grammatik wird bei diesem Ansatz eine Kontroll-Grammatik entwickelt, die anhand der Attribute bestimmt, welche Regeln angewendet werden.

4 Grundlagen der Monte Carlo Simulationsverfahren

In diesem Kapitel werden Monte Carlo Simulationsverfahren vorgestellt. Zunächst werden grundlegende Methoden erläutert. Anschließend wird die reversible jump Markov Chain Monte Carlo (rjMCMC) Methode beschrieben. Einen Überblick über Monte Carlo Verfahren gibt das Buch von Liu (2001).

4.1 Monte Carlo Simulation

Monte Carlo Methoden können als Alternative zu rein analytischen Untersuchungen von Funktionen verwendet werden. Diese Methoden finden besonders Anwendung bei analytisch schwer oder gar nicht analysierbaren Funktionen. Monte Carlo Methoden erlauben somit die Untersuchung von Zufallsvariablen mit nicht vollständig bekannter Verteilung und die Simulation komplexer Prozesse.

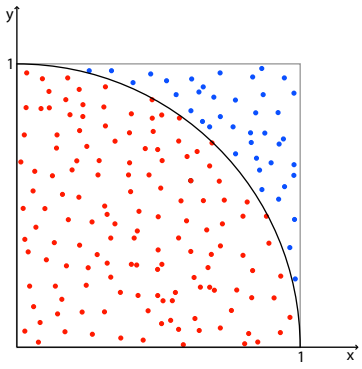
Eine Anwendung der Monte Carlo Methode ist z.B. die Bestimmung der Zahl π. Dazu betrachtet man den ersten Quadranten des Einheitskreises und das Quadrat, das von den Vektoren $(1,0)^T$ und $(0,1)^T$ aufgespannt wird (siehe Abbildung 4.1a). Als Stichprobe dient eine Menge gleichverteilter Punkte in diesem Quadrat. Aus den Flächeninhalten des Viertelkreises und des Quadrats ergibt sich das Verhältnis der Anzahl aller Punkte innerhalb des Einheitskreises zur Gesamtzahl der Punkte zu $\frac{\pi}{4}$.

Tabelle 4.1b zeigt die für π berechneten Werte für verschieden große Stichproben. Es ist zu erkennen, dass sich das Ergebnis bei wachsender Stichprobe an die Zahl $\pi = 3.14159\ldots$ annähert. Die Monte Carlo Methode nutzt also das Gesetz der Großen Zahlen aus, indem der wahre Wert durch Berechnung aus einer hinreichend großen Stichprobe gewonnen wird.

Monte Carlo Methoden werden häufig für die Berechnung von Integralen verwendet. Für ein Integral $I = \int_D g(x)dx$, wobei D in einem hochdimensionalen Raum liegt, kann das Integral durch

$$\hat{I}_m = \frac{1}{m}(g(x^{(1)}) + \cdots + g(x^{(m)}))$$

bestimmt werden, wenn man unabhängig und gleichverteilt zufällig eine Stichprobe von $g(x)$ ziehen kann. Das Gesetz der großen Zahlen besagt dann, dass $\lim_{m\to\infty} \hat{I}_m = I$ gilt. Dies liefert eine zweite Möglichkeit, die Zahl π zu berechnen. Die Fläche des ersten Quadranten des Einheitskreises lässt sich durch das Integral $\int_0^1 \sqrt{1-x^2}dx$ bestimmen. Aus einer Menge von m Stichproben $x^{(i)}$ einer gleichverteilten Variable zwischen 0 und 1 kann $\hat{\pi}_m = 4 \cdot \sum_{i=1}^{m} \sqrt{1-x^{(i)2}}$ als Näherung für π bestimmt werden. Die Methode lässt sich aber auch für andere Problemstellungen anwenden, in denen die Berechnung durch eine zufällig gezogene Stichprobe angenähert werden kann.



(a)

n	$\tilde{\pi}$
1000	3.24000
5000	3.14480
10000	3.12240
50000	3.13072
100000	3.14712
500000	3.14044
1000000	3.14214

(b)

Abbildung 4.1: Gleichverteilte Stichprobe an Punkten (a) und Annäherungen an die Zahl π mit dem Monte Carlo Verfahren bei wachsender Stichprobengröße n (b).

Der grundlegende Schritt bei Monte Carlo Verfahren ist das Erzeugen der zufälligen Stichprobe von einer Wahrscheinlichkeitsverteilung $\pi(x)$. Die einzelnen Elemente der Stichprobe müssen unabhängig sein, da sonst ihre effektive Größe deutlich kleiner ist als die reale Größe. Für bestimmte einfache Verteilungen ist es möglich, diese durch Transformation aus der Gleichverteilung zwischen Null und Eins zu gewinnen. Diese ist in vielen Software-Umgebungen enthalten. Für kompliziertere Verteilungen, die in der Praxis auch oft nur bis auf eine Konstante bekannt sind, ist diese Transformationstechnik nicht möglich und daher müssen andere Methoden verwendet werden. Das Rejection-Sampling erzeugt die Stichprobe anhand einer Testverteilung $p(x)$, die sich zwar von π unterscheidet, ihr aber ähnlich sein sollte. Bei den MCMC Verfahren werden Markov-Ketten konstruiert, die die gewünschte Stichprobe erzeugen.

4.1.1 Rejection-Sampling

Ein grundlegendes Verfahren zum Erzeugen einer Stichprobe ist das Rejection-Sampling (von Neumann, 1951). Es soll eine Stichprobe aus der Verteilung $l(x)$ gezogen werden, was nicht direkt möglich ist. Die Verteilung $l(x) = c\pi(x)$ ist auch nur bis auf eine Konstante c bekannt und für jedes x lässt sich $\pi(x)$ berechnen. Für das Verfahren wird eine Auswahlverteilung $g(x)$, aus der man eine Stichprobe gewinnen kann, und eine begrenzende Konstante M benötigt, für die gilt: $Mg(x) \geq \pi(x)$ für alle x. Damit kann man die Methode von von Neumann folgendermaßen anwenden: Im ersten Schritt wird eine Probe x_0 von $g()$ gezogen. Dann wird eine weitere Zufallszahl u_0 aus der Gleichverteilung auf $[0, Mg(x_0)]$ gezogen. Das Paar (x_0, u_0) ist gleichverteilt auf der Fläche unterhalb der Verteilung $Mg(x)$. Ist $u_0 > \pi(x_0)$, so wird die Probe verworfen und es geht im ersten Schritt weiter. Andernfalls wird die Probe akzeptiert und der Menge der Stichproben hinzugefügt. Die akzeptierten Proben lehnen sich an die Verteilung l an. Dieses Vorgehen entspricht dem Warten auf ein Sample, das $l(x)$ genügt. Alle nicht passenden Samples werden verworfen. Die Akzeptanzwahrscheinlichkeit entspricht dabei $\frac{\pi(x)}{Mg(x)}$.

Ein Beispiel ist das Ziehen einer Stichprobe aus der Gamma-Verteilung $\mathrm{Gam}(z|a,b) = \frac{b^a z^{a-1} \exp(-bz)}{\Gamma(a)}$ (Bishop, 2006). Das Ziehen der Stichprobe kann nicht einfach über die Transformationstechnik realisiert werden. Da die Gamma-Verteilung eine glockenförmige Verteilung ist (siehe Abbildung 4.2), bietet sich die Cauchy-Verteilung $p(y) = \frac{1}{\pi}\frac{1}{1+y^2}$ als Auswahlverteilung an. Sie ist ebenfalls glockenförmig und man kann leicht Stichproben aus ihr ziehen. Um sicher zu stellen, dass die Auswahlverteilung immer oberhalb der Gamma-Verteilung liegt, wird noch eine kleine Veränderung vorgenommen, so dass die Stichproben schließlich aus $g(x) = \frac{k}{1+\frac{(x-c)^2}{b^2}}$ gezogen werden. Um eine möglichst geringe Ablehnungsrate zu bekommen, werden $c = a - 1$ und $b^2 = 2a - 1$ gewählt. Das k wird so klein wie möglich mit $Mg(x) \geq \pi(x)$ gewählt.

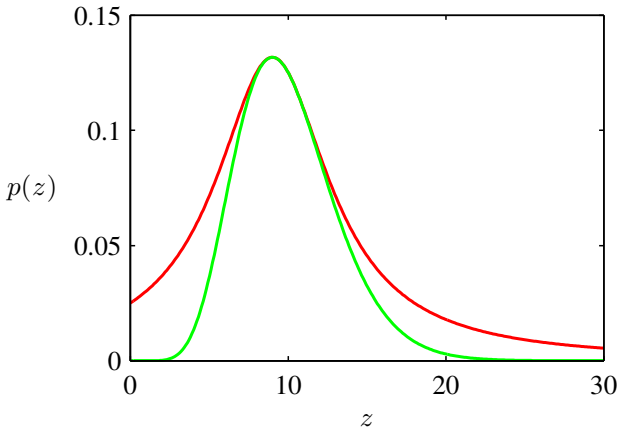


Abbildung 4.2: Skalierte Cauchy-Verteilung (rot) als Auswahlverteilung für die Gamma-Verteilung (grün) beim Rejection-Sampling (aus Bishop, 2006).

Um zu zeigen, dass die Methode korrekt ist, betrachtet man eine Indikatorfunktion I. I ist gleich 1, wenn die Probe X gezogen aus $g()$ akzeptiert wird, und sonst 0. Dann ist $P(I = 1) = \int P(I = 1|X = x)g(x)dx = \int \frac{c\pi(x)}{Mg(x)}g(x)dx = \frac{c}{M}$. Damit ist $p(x|I = 1) = \frac{c\pi(x)}{Mg(x)}g(x)/P(I = 1) = \pi(x)$. Da die Anzahl an Operationen, um eine akzeptierte Probe zu bekommen, M ist, sollte man eine gute Testverteilung $g(x)$ wählen, die ein kleines M hervorruft. Für hochdimensionale Monte Carlo Simulationen ist es schwierig Rejection-Sampling anzuwenden, da eine passende begrenzende Verteilung nur schwer gefunden werden kann.

4.1.2 Importance-Sampling

Marshall (1956) publizierte die Idee, sich beim Ziehen von Stichproben aus hochdimensionalen Räumen besonders die interessanten Regionen anzusehen. Die Bereiche, die nicht 0 sind, decken nur einen kleinen Teil des gesamten Raumes ab und somit kann durch eine Gewichtung Rechenzeit gespart werden. Das Verfahren stellt nicht direkt eine Möglichkeit dar, eine Stichprobe aus der gegebenen Verteilung zu ziehen, sondern berechnet direkt den Erwartungswert, indem die Samples der Testverteilung gewichtet werden.

Es soll der folgende Erwartungswert bestimmt werden. $\mu = E_\pi[h(x)] = \int h(x)\pi(x)dx$. Der Importance-Sampling-Algorithmus verwendet eine Testverteilung $g(.)$ und funktioniert folgendermaßen:

1. Ziehe $x^{(1)}, \cdots, x^{(m)}$ von der Testverteilung $g(.)$.
2. Berechne das Gewicht $w^{(j)} = \pi(x^{(j)})/g(x^{(j)})$ für $j = 1, \cdots, m$.
3. Approximiere μ durch
$$\hat{\mu} = \frac{w_1 h(x^{(1)}) + \cdots + w_m h(x^{(m)})}{w_1 + \cdots + w_m}. \tag{4.1}$$

Um den Schätzfehler möglichst klein zu halten, möchte man $g(x)$ möglichst nah an $\pi(x)h(x)$ wählen. Ein Vorteil der Approximation nach Formel 4.1 ist, dass man hier nur das Verhältnis $\pi(x)/g(x)$ bis auf eine Konstante kennen muss. Würde man die Formel $\tilde{\mu} = \frac{1}{m}(w_1 h(x^{(1)}) + \cdots + w_m h(x^{(m)}))$ verwenden, so müsste das Verhältnis genau bekannt sein.

4.2 Markov Chain Monte Carlo Sampling

Die bisher vorgestellten Methoden sind nicht geeignet, Stichproben aus hochdimensionalen Räumen zu generieren. Deshalb werden hier Markov Chain Monte Carlo (MCMC) Methoden vorgestellt. Sie bieten die Möglichkeit, Stichproben aus einer nicht vollständig bekannten, möglicherweise hochdimensionalen Verteilung zu ziehen. Dazu wird eine Markov-Kette generiert, die eine Stichprobe der gesuchten Verteilung erzeugt. Der Metropolis-Algorithmus (Metropolis u. a., 1953) und die Generalisierung von Hastings (1970) bilden die Grundlage der MCMC Methoden. Bevor diese beschrieben werden, werden vorher noch allgemein Markov-Ketten und einige ihrer Eigenschaften definiert.

4.2.1 Markov-Ketten

Eine Markov-Kette ist eine Folge von Zufallsvariablen, die durch eine spezielle Abhängigkeit gekennzeichnet ist. Die hier angegebenen Definitionen sind an die Darstellung von Krengel (2000) angelehnt. Zunächst sollen die Begriffe σ-Algebra und messbarer Raum definiert werden und mit deren Hilfe dann der Begriff des stochastischen Prozesses.

Definition 4.1

Sei $\Omega \neq \emptyset$ beliebig. Eine Familie $\mathcal{A}$ von Teilmengen von Ω heißt **σ-Algebra**, wenn gilt: $\Omega \in \mathcal{A}$, $A \in \mathcal{A} \Rightarrow A^c \in \mathcal{A}$ und $A_1, A_2, \ldots \in \mathcal{A} \Rightarrow \bigcup_{i=1}^{\infty} A_i \in \mathcal{A}$.

Dabei bezeichnet $A^c = \Omega \backslash A$ das Komplement der Menge $A \subset \Omega$.

Definition 4.2

Ein **messbarer Raum** ist ein Paar $(\Omega, \mathcal{A})$, bestehend aus einer nichtleeren Menge Ω und einer σ-Algebra $\mathcal{A}$ von Teilmengen von Ω. Ein **Wahrscheinlichkeitsmaß** P ist eine auf $\mathcal{A}$ definierte Funktion mit Werten in $[0, 1]$, welche den folgenden Bedingungen genügt:

$$P(\mathcal{A}) \geq 0 \text{ für alle } A \in \mathcal{A},$$

$$P(\Omega) = 1,$$

$$P \text{ ist } \sigma\text{-additiv, d.h. für disjunkte } A_1, A_2, \ldots \in \mathcal{A} \text{ ist } P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i).$$

$(\Omega, \mathcal{A}, P)$ heißt dann **Wahrscheinlichkeitsraum**, P auch **Wahrscheinlichkeitsverteilung**. Teilmengen $A \subset \Omega$, die zu $\mathcal{A}$ gehören, heißen **Ereignisse**.

Definition 4.3

Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum, T eine beliebige nichtleere Indexmenge und $(I, \mathcal{I})$ ein messbarer Raum. Dann heißt eine Familie $\{X_t, t \in T\}$ von Zufallsvariablen mit Werten in I **stochastischer Prozess** mit Parameterbereich T und Zustandsraum I.

Im Folgenden sei I endlich oder abzählbar unendlich und $\mathcal{I}$ bezeichnet die Potenzmenge von I. Mit diesen Definitionen kann der Begriff der Markov-Kette definiert werden.

Definition 4.4

Eine **Markov-Kette** ist ein stochastischer Prozess $\{X_n, n \in \mathbb{Z}^+\}$ mit abzählbarem Zustandsraum I, der die folgende **Markov'sche Eigenschaft** besitzt: Für alle $n \in \mathbb{Z}^+$ und für alle $i_0, \ldots, i_{n+1} \in I$ mit $P(X_0 = i_0, \ldots, X_n = i_n) > 0$ ist

$$P(X_{n+1} = i_{n+1} | X_0 = i_0, \ldots, X_n = i_n) = P(X_{n+1} = i_{n+1} | X_n = i_n).$$

Die Markov'sche Eigenschaft besagt also, dass der Zustandswechsel nicht von der gesamten Kette, sondern nur vom letzten Zustand und dem Zeitpunkt n abhängt. Man sagt auch, dass die Kette kein Gedächtnis hat.

Definition 4.5

Eine Markov-Kette heißt **homogen** oder Kette mit **stationären Übergangswahrscheinlichkeiten**, wenn für alle $i, j \in I$ gilt, dass $P(X_{n+1} = j | X_n = i) =: p_{ij}$ unabhängig von n ist.

Bei der speziellen Klasse der homogenen Markov-Ketten ist die Übergangswahrscheinlichkeit also unabhängig vom Zeitpunkt n, d.h. sie hängt nur vom letzten Zustand ab. Eine homogene Markov-Kette kann man dann durch eine stochastische Matrix $T = (p_{ij})$ beschreiben. Diese nennt man auch Übergangskern, da die einzelnen Einträge die Übergangswahrscheinlichkeiten zwischen den Zuständen sind. Für eine stochastische Matrix gelten die folgenden Eigenschaften. $p_{ij} \geq 0$ $(i, j \in I)$ und $\sum_{j \in I} p_{ij} = 1$. Für eine homogene Markov-Kette nennt man $p_{ij}^{(m)} := P(X_{n+m} = j | X_n = i)$ die m-Schritt-Übergangswahrscheinlichkeit von i nach j.

Im Folgenden werden nur homogene Markov-Ketten betrachtet. Es folgen Definitionen zu Eigenschaften von Markov-Ketten, die für das spätere Verfahren wichtig sind.

Definition 4.6

Eine Verteilung π heißt **stationäre** oder **invariante Verteilung** einer Markov-Kette mit Übergangskern T, wenn gilt $\pi T = \pi$.

Definition 4.7

Ein Zustand i führt in n Schritten zu Zustand j, geschrieben $i \rightsquigarrow j[n]$, wenn $p_{ij}^{(n)} > 0$. Gibt es ein $n \geq 1$ mit $i \rightsquigarrow j[n]$, dann sagt man i führt zu j und schreibt $i \rightsquigarrow j$. Die Relation $\rightsquigarrow$ ist transitiv. Man sagt, i **kommuniziert** mit j, $i \leftrightsquigarrow j$, wenn $i \rightsquigarrow j$ und $j \rightsquigarrow i$. i heißt **wesentlich**, wenn jeder Zustand j, zu dem i führt, zurück zu i führt, also mit i kommuniziert. Auf der Teilmenge der wesentlichen Zustände ist $\leftrightsquigarrow$ eine Äquivalenzrelation.

Definition 4.8

Die **Periode** von einem Zustand i mit $i \rightsquigarrow i$ ist der größte gemeinsame Teiler der potenziellen Rückkehrzeiten

$$d_i = ggT\{n \geq 1 : i \rightsquigarrow i[n]\}.$$

Gilt nicht $i \rightsquigarrow i$, so ist $d_i = \infty$. Zustände mit $d_i = 1$ heißen **aperiodisch**. Die Markov-Kette heißt **aperiodisch**, wenn alle Zustände aperiodisch sind, und **periodisch mit Periode** d, wenn alle $d_i = d \geq 2$ sind.

Definition 4.9

Eine Markov-Kette heißt **irreduzibel**, wenn die Äquivalenzrelation $\leftrightsquigarrow$ sie in nur eine Klasse aufteilt. Das heißt, dass alle Zustände miteinander kommunizieren.

Definition 4.10

Ein Zustand i einer Markov-Kette heißt **rekurrent**, wenn $P(X_n = i \text{ für irgendein } n \geq 1 | X_0 = i) = 1$ ist. Sonst heißt der Zustand transient. Eine Markov-Kette heißt **rekurrent**, wenn jeder Zustand rekurrent ist.

Definition 4.11

Eine Markov-Kette heißt **ergodisch**, wenn sie irreduzibel, aperiodisch und rekurrent ist.

Definition 4.12

Eine Markov-Kette heißt **reversibel**, wenn gilt

$$P(X_0 = i_0, \ldots, X_n = i_n) = P(X_0 = i_n, \ldots, X_n = i_0)$$

Reversibilität ist oft nicht leicht zu zeigen. Leichter zu zeigen ist die stärkere Bedingung $\pi(x)A(x,y) = \pi(y)A(y,x)$, die *detailed balance* genannt wird und die Reversibilität impliziert.

Neal (1993) gibt einen Beweis für den folgenden Satz:

Satz 4.1

Hat eine Markov-Kette mit endlichem Zustandsraum und dem Übergangskern $T(x, x')$ die invariante Verteilung π und ist

$$\nu = \min_x \min_{x':\pi(x')>0} \frac{T(x,x')}{\pi(x')} > 0, \tag{4.2}$$

dann ist die Markov-Kette ergodisch. D.h. unabhängig von den Ausgangswahrscheinlichkeiten $p_0(x)$ gilt für alle x

$$\lim_{n\to\infty} p_n(x) = \pi(x). \tag{4.3}$$

Eine Grenze für die Konvergenzrate ist durch

$$|\pi(x) - p_n(x)| \leq (1-\nu)^n \tag{4.4}$$

gegeben.

Die Aussage über die Ergodizität gilt ebenfalls für homogene Markov-Ketten, die Bedingung 4.2 nicht direkt erfüllen, bei denen diese aber für Übergänge mit k Schritten, also über T^k, gilt. In 4.4 muss entsprechend der Exponent durch $\lfloor \frac{n}{k} \rfloor$ ersetzt werden.

Mit dieser Erweiterung besagt Satz 4.1, dass eine irreduzible, aperiodische Markov-Kette mit endlichem Zustandsraum, die eine invariante Verteilung π besitzt, ebenfalls gegen π konvergiert. Ersetzt man in Bedingung 4.2 das T durch T^k ergibt sich $\nu = \min_x \min_{x':\pi(x)>0} \frac{T^k(x,x')}{\pi(x')} > 0$. Für irreduzible, aperiodische Markov-Ketten gibt es immer ein k, das diese Bedingung erfüllt. Denn die Irreduzibilität besagt, dass die Markov-Kette nicht in mehrere Teile unterteilt ist. Und die Aperiodizität stellt sicher, dass es eine Anzahl Schritte k gibt, mit der von jedem Zustand aus jeder andere Zustand erreicht werden kann.

Für die folgenden Verfahren ist es also wichtig, dass die Markov-Kette irreduzibel und aperiodisch ist. Zusätzlich muss eine stationäre Verteilung existieren.

4.2.2 Metropolis-Algorithmus

Metropolis u. a. (1953) veröffentlichten einen Algorithmus, auf dem ein Großteil der MCMC Verfahren aufbaut. Das Verfahren nutzt eine Markov-Kette im Zustandsraum der zu simulierenden Verteilung. Die stationäre

Verteilung der Kette ist die Zielverteilung $\pi(x)$. Für den Zustandswechsel von $x^{(t)}$ nach $x^{(t+1)}$ wird folgender Algorithmus angewandt:

1. Schlage eine zufällige Änderung von $x^{(t)}$ vor und generiere dadurch einen neuen Zustand y. Dies geschieht durch einen symmetrischen Übergangskern $T(x, y) = T(y, x)$.
2. Ziehe eine gleichverteilte Zufallszahl zwischen 0 und 1: $U \sim \text{Uniform}[0, 1]$.
3. Setze das neue $x^{(t+1)}$:
$$x^{(t+1)} = \begin{cases} y & \text{wenn } U \leq \frac{\pi(y)}{\pi(x^{(t)})} \\ x^{(t)} & \text{sonst.} \end{cases}$$

Die Einschränkung hierbei ist die Bedingung, dass der Übergangskern symmetrisch ist. Es können also nur Systeme simuliert werden, bei denen für alle Zustände x und y die Übergänge von x nach y und umgekehrt gleich wahrscheinlich sind.

4.2.3 Metropolis-Hastings-Algorithmus

Hastings (1970) erweiterte den Metropolis Algorithmus auf nicht symmetrische Übergangskerne. Er veränderte die Akzeptanzbedingung so, dass auch nicht symmetrische Übergangsfunktionen verwendet werden können. Einzige Bedingung ist hierbei, dass ein Übergang von x nach y nur dann möglich ist, wenn auch ein Übergang von y zu x möglich ist. $T(x, y) > 0 \Leftrightarrow T(y, x) > 0$ für alle x und y. Der Algorithmus arbeitet dann folgendermaßen: Das System befindet sich im Zustand $x^{(t)}$.

1. Ziehe mithilfe des Übergangskerns $T(x^{(t)}, y)$ einen neuen Zustand y.
2. Ziehe eine gleichverteilte Zufallszahl zwischen 0 und 1: $U \sim \text{Uniform}[0, 1]$.
3. Setze das neue $x^{(t+1)}$:
$$x^{(t+1)} = \begin{cases} y & \text{wenn } U \leq r(x^{(t)}, y) \\ x^{(t)} & \text{sonst,} \end{cases}$$
wobei $r(x, y)$ folgendermaßen definiert ist:
$$r(x, y) = \min\Big\{1, \frac{\pi(y)T(y, x)}{\pi(x)T(x, y)}\Big\}.$$

Im Folgenden wird gezeigt, warum die zwei vorgestellten Verfahren gegen die Zielverteilung konvergieren. Die Markov-Ketten Theorie sagt, dass eine Markov-Kette gegen ihre invariante Verteilung konvergiert, wenn sie irreduzibel und aperiodisch ist und eine invariante Verteilung besitzt. Nach Liu (2001) ist die Markov-Kette beim Metropolis-Algorithmus fast sicher aperiodisch. Unter der Annahme, eine irreduzible Markov-Kette konstruiert zu haben, bleibt zu zeigen, dass die Kette eine invariante Verteilung besitzt. Dazu betrachten wir die tatsächliche Übergangsfunktion $A(x, y)$ die sich aus Übergangskern und der Akzeptanzwahrscheinlichkeit zusammensetzt. $A(x, y) = T(x, y) \min\Big\{1, \frac{\pi(y)T(y,x)}{\pi(x)T(x,y)}\Big\}$. Folgende Gleichung muss gezeigt werden:

$$\sum_{x \in I} \pi(x)A(x, y) = \pi(y) \tag{4.5}$$

Dies ist direkt nicht so leicht zu zeigen, deshalb zeigt man detailed balance $\pi(x)A(x, y) = \pi(y)A(y, x)$ die Gleichung 4.5 impliziert, denn es gilt

$$\sum_{x \in I} \pi(x)A(x, y) = \sum_{x \in I} \pi(y)A(y, x) = \pi(y) \sum_{x \in I} A(y, x) = \pi(y).$$

Folgende Berechnung zeigt, dass detailed balance für Metropolis-Hastings gilt:

$$\pi(x)A(x, y) = \pi(x)T(x, y) \min\Big\{1, \frac{\pi(y)T(y, x)}{\pi(x)T(x, y)}\Big\} = \min\{\pi(x)T(x, y), \pi(y)T(y, x)\}$$

Diese Funktion ist symmetrisch in x und y, somit ist die Eigenschaft der detailed balance gegeben. Die Rechnung zeigt auch, dass andere Akzeptanzfunktionen verwendet werden können, solange die tatsächliche Übergangsfunktion $A(x, y)$ die Form $A(x, y) = T(x)\delta(x, y)$ hat, wobei $\delta(x, y)$ symmetrisch in x und y ist.

4.2.4 Reversible jump Markov Chain Monte Carlo Sampling

Ursprünglich konnten Markov Chain Monte Carlo Methoden nur für Probleme verwendet werden, bei denen die Anzahl der Variablen über den gesamten Prozess konstant war. Der zu bestimmende Parametervektor musste also eine feste Dimension haben, Veränderungen der Dimension waren nicht möglich. Green (1995) stellt eine Methode vor, die diese Dimensionsänderungen zulässt. Dabei teilt sich das Problem in die Wahl aus einer diskreten Menge von Modellen, Bestimmung eines Parametervektors, der in Abhängigkeit vom gewählten Modell interpretiert wird, und die Verwendung von Daten, die als Basis für die Vorhersage verwendet werden. Die verschiedenen Modelle können Parametervektoren unterschiedlicher Länge besitzen. Diese liegen in verschiedenen Unterräumen, zwischen denen das Verfahren hin und her springen kann. Diese Wechsel der Unterräume werden als „jumps“ bezeichnet.

Bei der rjMCMC Methode kann der Parametervektor θ, der das Modell beschreibt, unterschiedliche Dimension besitzen. Die Zustände können also aus unterschiedlichen Unterräumen sein. Somit gibt es unterschiedliche Typen von Veränderungen. Vom Zustand x wird mit einer Änderung vom Typ m in den Zustand dx' mit der Wahrscheinlichkeit $q_m(x, dx')$ gewechselt. Es gilt also $\sum_m q_m(x, \mathcal{C}) \leq 1$, wobei $\mathcal{C}$ der kombinierte Parameterraum ist, und zu einer Wahrscheinlichkeit $1 - \sum_m q_m(x, \mathcal{C})$ wird keine Änderung vorgenommen. Von einem Zustand x sind nicht alle Veränderungen möglich und deshalb ist auch für alle x für einige m $q_m(x, \mathcal{C}) = 0$.

Wie beim Algorithmus von Hastings wird auch hier die Veränderung in Abhängigkeit einer Akzeptanzwahrscheinlichkeit $\alpha_m(x, x')$ akzeptiert.

$$\alpha_m(x, x') = \min\left\{1, \frac{\pi(dx')q_m(x', dx)}{\pi(dx)q_m(x, dx')}\right\} \tag{4.6}$$

Es wird angenommen, dass $\pi(dx)q_m(x, dx')$ eine endliche Dichte $f_m(x, x')$ bezüglich eines symmetrischen Maßes ξ_m auf $\mathcal{C} \times \mathcal{C}$ hat.

Anschaulicher erklärt Green (1995) das an einem Beispiel mit zwei Unterräumen $\mathcal{C}_1$ und $\mathcal{C}_2$, wobei $p(\theta^{(1)}|k = 1)$ und $p(\theta^{(2)}|k = 2)$ geeignete Dichten in $\mathbb{R}^{n_1}$ und $\mathbb{R}^{n_2}$ sind. Es gibt nur einen Typ von Veränderung, so dass immer zwischen den Unterräumen gewechselt wird. Es ist also $q(x, \mathcal{C}_1) = 0$ für $x \in \mathcal{C}_1$ und $q(x, \mathcal{C}_2) = 0$ für $x \in \mathcal{C}_2$. Wobei auf den Index m verzichtet werden kann, da es nur eine Art der Veränderung gibt.

Der Übergang von $\mathcal{C}_1$ zu $\mathcal{C}_2$ kann jetzt durch einen zusätzlichen Vektor $u^{(1)}$ der Länge m_1 definiert werden. Dieser wird unabhängig von $\theta^{(1)}$ erzeugt und dann kann aus $\theta^{(1)}$ und $u^{(1)}$ der Wert für $\theta^{(2)}$ deterministisch berechnet werden. Ebenso kann man mit einem Vektor $u^{(2)}$ der Länge m_2 und $\theta^{(2)}$ einen Wert für $\theta^{(1)}$ berechnen. Für das Dimensions-Matching ist es wichtig, dass es eine Bijektion zwischen $(u^{(1)}, \theta^{(1)})$ und $(u^{(2)}, \theta^{(2)})$ gibt. Insbesondere müssen die Längen der Vektoren folgendes erfüllen: $n_1 + m_1 = n_2 + m_2$.

Damit wird die Vorschlagswahrscheinlichkeit $q(x, dx')$ über die Verteilung der Vektoren $u^{(1)}$ und $u^{(2)}$ ausgedrückt, welche über geeignete Dichten q_1 und q_2 bezüglich des Lebesguemaßes auf $\mathbb{R}^{n_1}$ und $\mathbb{R}^{n_2}$ gegeben sind.

Hiermit kann das symmetrische Maß ξ definiert werden. Für $A \subset \mathcal{C}_1$ und $B \subset \mathcal{C}_2$ ist

$$\xi(A \times B) = \xi(B \times A) = \lambda\{(\theta^{(1)}, u^{(1)}) : \theta^{(1)} \in A, \theta^{(2)}(\theta^{(1)}, u^{(1)}) \in B\},$$

wobei λ ein $(n_1 + m_1)$-dimensionales Lebesguemaß ist. Für allgemeine $A, B \subset \mathcal{C}$ ist

$$\xi(A \times B) = \xi\{(A \cap \mathcal{C}_1) \times (B \cap \mathcal{C}_2)\} + \xi\{(A \cap \mathcal{C}_2) \times (B \cap \mathcal{C}_1)\}$$

Die Dichte f wird für $x = (1, \theta^{(1)}) \in \mathcal{C}_1$ und $x' = (2, \theta^{(2)}) \in \mathcal{C}_2$ definiert

$$f(x, x') = p(1, \theta^{(1)}|y)j(1, \theta^{(1)})q_1(u^{(1)}),$$

$$f(x', x) = p(2, \theta^{(2)}|y)j(2, \theta^{(2)})q_2(u^{(2)}) \left|\frac{\partial(\theta^{(2)}, u^{(2)})}{\partial(\theta^{(1)}, u^{(1)})}\right|$$

und sonst ist $f(x, x') = 0$. Damit ist $f(x, x')$ für alle $x, x' \in \mathcal{C}$ die Dichte bezüglich ξ der gemeinsamen Vorschlags-Verteilung $\pi(dx)q(x, dx')$.

Damit ergibt sich als Akzeptanzwahrscheinlichkeit $\alpha(x, x')$ für einen Wechsel von $x = (1, \theta^{(1)})$ nach $x = (2, \theta^{(2)})$

$$\alpha(x, x') = \min\left\{1, \frac{p(2, \theta^{(2)}|y) j(2, \theta^{(2)}) q_2(u^{(2)})}{p(1, \theta^{(1)}|y) j(1, \theta^{(1)}) q_1(u^{(1)})} \left|\frac{\partial(\theta^{(2)}, u^{(2)})}{\partial(\theta^{(1)}, u^{(1)})}\right|\right\}.$$

Oft ist es auch so, dass nur bei Veränderungen in eine Richtung ein Vektor $u^{(i)}$ benötigt wird und eines der m_1, m_2 Null ist. Für $m_2 = 0$ vereinfacht sich die Akzeptanzwahrscheinlichkeit folgendermaßen:

$$\alpha(x, x') = \min\left\{1, \frac{p(2, \theta^{(2)}|y) j(2, \theta^{(2)})}{p(1, \theta^{(1)}|y) j(1, \theta^{(1)}) q_1(u^{(1)})} \left|\frac{\partial(\theta^{(2)})}{\partial(\theta^{(1)}, u^{(1)})}\right|\right\}$$

Dick u. a. (2004) verwenden für ihr Verfahren zur automatischen Modellierung von 3D-Modellen aus Bildserien ein rjMCMC Verfahren. In diesem Ansatz werden Gebäude wie in einem Lego-Baukasten aus Primitiven zusammengesetzt. Ein Haus setzt sich aus mehreren Wänden zusammen und für jede Wand wird die Fassadenstruktur aus Primitiven wie Fenster und Türen kombiniert.

Für jedes Primitiv geht Vorwissen in den Prozess ein und daraus kann mithilfe eines Maximum A Posteriori (MAP) Schätzers das Modell bestimmt werden. Informationen hierfür kommen teils aus Trainingsdaten und werden teils von Architekten bestimmt. Zusätzlich soll Wissen über den räumlichen Zusammenhang der einzelnen Primitive formuliert werden. Die Einflussgrößen sind hier der Maßstab, die Form, die Ausrichtung der Primitive zueinander und andere Relationen wie z.B. Symmetrien. Da diese Bedingungen nur schwer zu formulieren sind, verwenden Dick u. a. (2004) ein rjMCMC Verfahren, bei dem die Bewertungsfunktionen die oben genannten Kriterien verwenden. Der Übergangskern enthält Wahrscheinlichkeiten für das Erzeugen, Löschen und Verändern von Formen, Wänden oder Fenstern. Als Startzustände für das Verfahren werden die Ergebnisse des MAP-Schätzers verwendet.

In der Arbeit wird das rjMCMC Verfahren zunächst unabhängig von den Daten angewandt, um die Wahl des Shape-Priors zu überprüfen. Um es dann in Abhängigkeit der Bildserien laufen zu lassen, wird die Bewertungsfunktion um einen Datenterm erweitert. Dieser bewertet das Modell anhand der Daten durch einen Texturvergleich beider. Das Ergebnis des Verfahrens sind mehrere benannte Modelle, aus denen der Benutzer das beste auswählen kann.

4.3 Simulated Annealing

Die MCMC Verfahren liefern eine Folge von Zufallszahlen entsprechend einer gegebenen Verteilung. Um das Verfahren gegen das Maximum der Wahrscheinlichkeitsdichte konvergieren zu lassen, kann man es noch mit dem Optimierungsverfahren Simulated Annealing verbinden.

Das von Kirkpatrick u. a. (1983) vorgestellte Simulated Annealing Verfahren ist eher ein Optimierungs- als ein Simulationsverfahren. Es verwendet die Vorgänge beim Schmelzen von Metall als Motivation. Hierbei werden Partikel bei steigenden Temperaturen beweglicher. Übertragen entspricht das den Zustandsänderungen der Markov-Kette. Bei höheren Temperaturen sind größere Änderungen der Zustände möglich. Kühlt die Temperatur ab, werden die Teilchen unbeweglicher, also werden die Zustandsänderungen begrenzt.

MCMC Verfahren werden mit dem Simulated Annealing kombiniert (Andrieu u. a., 2003), um nicht nur eine Menge von Stichproben aus einer unbekannten Verteilung zu ziehen, sondern das Maximum der Dichtefunktion zu ermitteln. Das Maximum kann auch aus dem Maximum der Wahrscheinlichkeiten der Stichproben bestimmt werden. $\hat{x} = \arg\max_{x^{(i)}; i=1,\ldots,N} \pi(x^{(i)})$. Bei dieser Methode werden allerdings viele Vorschläge gemacht, die nicht in der Nähe des Maximums liegen. Dadurch werden viele unnötige Vorschläge gemacht, die das Verfahren ineffizient machen. Um dies zu verhindern, wird beim Simulated Annealing bei sinkender Temperatur die Masse um das Maximum konzentriert. Hierzu wird eine Markov-Kette simuliert, die in der Iteration i nicht mehr $\pi(x)$ sondern $\pi_i(x) \equiv \pi^{\frac{1}{T_i}}$ als invariante Verteilung hat.

Zu Beginn wird die Temperatur $T_0 = 1$ gesetzt. Die Markov-Kette befindet sich im Zustand $x^{(t)}$. Der angepasste Algorithmus läuft dann folgendermaßen ab:

1. Ziehe mithilfe des Übergangskerns $T(x^{(t)}, y)$ einen neuen Zustand y.
2. Ziehe eine gleichverteilte Zufallszahl zwischen 0 und 1: $U \sim \text{Uniform}[0, 1]$.

3. Setze das neue $x^{(t+1)}$:

$$x^{(t+1)} = \begin{cases} y & \text{wenn } U \leq r(x^{(t)}, y) \\ x^{(t)} & \text{sonst,} \end{cases}$$

wobei $r(x, y)$ folgendermaßen definiert ist:

$$r(x, y) = \min\Big\{1, \frac{\pi^{\frac{1}{T_i}}(y)T(y, x)}{\pi^{\frac{1}{T_i}}(x)T(x, y)}\Big\}.$$

4. Setze T_i entsprechend dem Abkühlungsschema

Man kann zeigen, dass mit Simulated Annealing das globale Maximum mit Wahrscheinlichkeit 1 erreicht wird, wenn man T_i genügend langsam abkühlt, d.h. in einer logarithmischen Größenordnung. In der Praxis wird aber keine so langsame Abkühlung verwendet. Es werden eher lineare oder sogar exponentielle Funktionen verwendet. Hierbei kann allerdings keine Konvergenz mit Wahrscheinlichkeit 1 zum globalen Optimum garantiert werden.

Lafarge u. a. (2009) verwenden rjMCMC in Kombination mit Simulated Annealing für die Gebäuderekonstruktion aus digitalen Höhenmodellen. Im ersten Schritt werden aus den Daten 2D-Regionen annähernd bestimmt, die Gebäudeteile darstellen. In diesen werden dann aus einer Bibliothek von 3D-Blöcken 3D-Modelle platziert. Für die Bestimmung der Lage gibt es eine automatische und eine interaktive Methode. Das automatische Verfahren (Lafarge u. a., 2008) erstellt zunächst eine grobe Approximation an das Gebäude durch Rechtecke. Das geschieht mithilfe eines Marked-Point-Prozesses. Dieser benötigt für jede Anordnung von Rechtecken eine Energiefunktion. Die Rechtecke werden weiter verarbeitet, indem benachbarte Rechtecke verschmolzen werden. Das Ergebnis des Prozesses ist ein Menge von Vierecken. Diese werden dann nach Dachhöhen-Diskontinuitäten partitioniert. Beim interaktiven Ansatz werden die Eckpunkte eines Vierecks manuell gesetzt. Der automatische Ansatz liefert nicht ganz so genaue Ergebnisse wie der interaktive, ist dafür aber deutlich schneller. Beide Ergebnisse können für die Erstellung der 3D-Gebäude verwendet werden.

Die 3D-Gebäude werden aus einer Bibliothek aus 3D-Blöcken zusammengestellt. Diese beinhaltet verschiedene Dachformen, wie z.B. Flachdach, Satteldach oder Mansardendach, die in unterschiedlichen Varianten vorkommen können. Diese Varianten eines Satteldachs sind dann z.B. ein einfaches Satteldach, ein Walmdach oder ein l-förmiges Satteldach. Um die Blöcke zu bewerten, werden Likelihood und Prior definiert. Der Likelihood ist ein Distanzmaß zwischen dem Objekt und dem digitalen Höhenmodell. Für den Prior wird die Interaktion zwischen benachbarten Objekten betrachtet. Dazu werden Objekte als „verbindbar“ definiert, wenn ihre Orientierungen kompatibel sind und an ihrer gemeinsamen Kante kein Höhensprung im Höhenmodell auftritt. Der Prior bevorzugt „verbindbare“ Objekte, lässt aber auch nicht passende zu.

Für die Optimierung wird dann ein rjMCMC Sampler verwendet und mit Simulated Annealing die beste Konfiguration gesucht.

5 Entwicklung einer formalen Grammatik zur Modellierung von Fassaden

Betrachtet man Gebäudefassaden, so fällt auf, dass ihr Aufbau gewissen Regeln folgt. In der Architektur treten oft gleiche Teile wiederholt oder symmetrisch auf. Die Platzierung von Fassadenelementen, insbesondere von Fenstern erfolgt auch nach regelmäßigen Kriterien. Sie sind oft in regelmäßigen Gitterstrukturen angelegt. Das Wissen über die Struktur von Fassaden kann für die Rekonstruktion genutzt werden. Deshalb wird in dieser Arbeit eine Fassadengrammatik definiert, die die Struktur von Fassaden beschreibt und die oben erwähnten Regeln, nach denen eine Fassade aufgebaut ist, wiedergibt. Hierbei geht es um typische Fassaden, wie sie beispielsweise in Wohngebieten zu finden sind. Einige Beispiele zeigt Abbildung 5.1.

Abbildung 5.1: Typische Fassaden in einem Wohngebiet.

Die im Folgenden entwickelte Fassadengrammatik unterteilt die Fassade schrittweise, ähnlich wie die Split-Grammatik aus Wonka u. a. (2003). Für die Fassadenrekonstruktion genügt es allerdings, zweidimensionale Shapes zu betrachten, da aufgrund der Erfassungsmethode nur die Daten der Fassade und nicht des vollständigen Hauses vorliegen. Shapes sind also hier Kompositionen aus Linien im zweidimensionalen Raum. Die Regeln der Fassadengrammatik sind Split- und Umwandlungsregeln.

Jede modellierbare Fassade ist ein Wort der Grammatik, das durch den zum Wort gehörenden Ableitungsbaum beschrieben wird. Der Rekonstruktionsprozess ist also der Ableitungsprozess eines Wortes. In jedem Schritt soll die Fassade weiterentwickelt werden. Ein Ableitungsschritt unterteilt die Teilfassade, die dem linken Symbol entspricht. Die rechte Seite kann aus einem oder mehreren Symbolen bestehen. Die Ableitung ist also eine Partitionierung der Fassade, die mit der leeren Fassade als Startsymbol beginnt.

Die Strukturinformationen können schon im Rekonstruktionsprozess genutzt werden. In einer Untersuchung verschiedener Rekonstruktionsverfahren zeigen van Gool u. a. (2007) den Nutzen von sich wiederholender Struktur. Objekte, die durch Verdeckung nicht erfasst wurden, können mithilfe von Wissen über Strukturen modelliert werden. Zusätzlich müssen wiederholte Strukturen nur einmal modelliert werden und erlauben so eine kompakte Speicherung der Modelle. Des Weiteren eröffnet die im Ableitungsbaum enthaltene strukturelle Information zusätzliche Möglichkeiten für die Generalisierung.

5.1 Aufbau der Grammatik

Um die Grammatik zu definieren, werden zuerst die Symbole bestimmt und dann die Ableitungsregeln formuliert. Die Grammatik wurde auf Basis von Fassadenbildern erstellt. Es wurden Bilder aus einer Fassadendatenbank analysiert und nötige Strukturelemente und -regeln extrahiert. Die verwendeten Regeln sind parametrisierbar und somit beschreibt jede Regel eine Vielzahl tatsächlicher Ausprägungen.

5.1.1 Definition der Symbole

Um Fassadenstrukturen zu modellieren, werden zwei verschiedene Arten von Symbolen verwendet. Es gibt zum einen Symbole ohne Strukturinformation, die für die Aufteilung der Fassade in einheitliche Bereiche verwendet werden, zum anderen Symbole, die Information über die Struktur beinhalten.

Startsymbol der Grammatik ist das Symbol Facade. Es stellt den Umriss einer rechteckigen Fassade dar und beinhaltet keine weiteren Informationen als seine Breite und Höhe. Daneben gibt es noch weitere strukturfreie Elemente, die zur Aufteilung der Fassade dienen. Dies sind PartFacade, FacadeRow, FacadeColumn und FacadeElement. PartFacade stellt einen beliebigen Teilbereich der Fassade dar, FacadeRow und FacadeColumn sind Zeilen bzw. Spalten in der Fassade und FacadeElement ein Teilbereich der Fassade, der nur ein Fassadenelement enthält. Die modellierbaren Fassadenelemente sind Window, Door, ShopWindow und DoorWay. Diese Fassadenelemente sind terminale Symbole. SplitPartFacade ist ein geteilter Teil einer Fassade. Dieses Symbol wird dann verwendet, wenn sich Strukturen über eine Fassade erstrecken, aber durch eine Spalte mit anderer Struktur unterbrochen werden.

Eine wichtige Rolle in der Fassadengrammatik spielen die Symbole, die strukturelle Eigenschaften beschreiben. Dabei handelt es sich um verschiedene Arten von Gittern und die Symbole SymmetricFacadeSide, RepeatedFacade und SplitPartFacade. SymmetricFacadeSide stellt die linke Seite einer Fassade oder Teilfassade dar, wobei die rechte Seite implizit symmetrisch vorhanden ist. RepeatedFacade beschreibt einen sich wiederholenden Teil einer Fassade. Hierbei wird ein Teil modelliert und ein Parameter gibt an, wie oft dieser wiederholt wird.

Des Weiteren gibt es verschiedene gitterförmige Strukturen. Diese beinhalten Fenster und optional Türen. Da Fenster nicht immer in einem Gitter mit gleichen Abständen angeordnet sind, gibt es eine Unterscheidung zwischen Gittern mit äquidistanten Spalten, die als regelmäßig (engl. regular) bezeichnet werden, und Gittern, in denen die Spaltenabstände variieren können. Da häufig die gleiche Fensterart in einer Fassade vorkommt, es aber manchmal auch Abweichungen gibt, wird als zweites Kriterium unterschieden, ob die Gitter identische (engl. identical) Fenster enthalten oder nicht. Daraus ergeben sich die vier Gittertypen RegularIdenticalWindowGrid, IdenticalWindowGrid, RegularWindowGrid und WindowGrid. Zu diesen kommen noch die entsprechenden Gittertypen mit Tür hinzu. Dabei kann die Tür ein Fenster in der untersten Reihe ersetzen oder zusätzlich an einer beliebigen freien Position am unteren Rand der Fassade stehen. Die Gitter mit Tür werden durch die Symbole RegularIdenticalWindowDoorGrid, IdenticalWindowDoorGrid, RegularWindowDoorGrid und WindowDoorGrid dargestellt.

Tabelle 5.1 listet die nichtterminalen Symbole der Grammatik auf, in Tabelle 5.2 sind einige terminale Symbole gegeben und Tabelle 5.3 zeigt die terminalen Symbole, die Gitterstrukturen abbilden. Diese 20 Symbole bilden das Alphabet der Fassadengrammatik. Im folgenden Abschnitt werden die Produktionsregeln definiert.

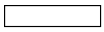
Facade		PartFacade	
FacadeRow		FacadeColumn	
FacadeElement		SymmetricFacadeSide	
SplitPartFacade		RepeatedFacade	

Tabelle 5.1: Nichtterminale Symbole der Grammatik.

Window		Door		ShopWindow		DoorWay	

Tabelle 5.2: Terminale Symbole der Grammatik.

5.1.2 Beschreibung der Regeln

Die Regeln sollen ausgehend vom Startsymbol Facade die Fassade partitionieren und den einzelnen Partitionen Strukturinformationen geben. Dazu gibt es zwei Arten von Regeln. Die einen unterteilen das Symbol auf der linken Seite in mehrere Bereiche. Das Ziel hierbei ist, die Fassade in Bereiche zu unterteilen, die jeweils eine einheitliche Struktur aufweisen. Hierbei werden zunächst die oben beschriebenen Symbole ohne Strukturinformation verwendet. Die zweite Art von Regeln ersetzt ein Symbol mit oder ohne Strukturinformation durch ein Symbol mit Strukturinformation.

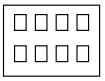
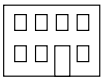
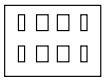
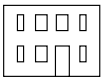
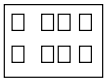
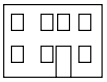
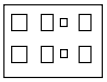
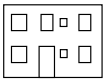
RegularIdenticalWindowGrid		RegularIdenticalWindowDoorGrid	
RegularWindowGrid		RegularWindowDoorGrid	
IdenticalWindowGrid		IdenticalWindowDoorGrid	
WindowGrid		WindowDoorGrid	

Tabelle 5.3: Gittersymbole der Grammatik.

Zu den Unterteilungsregeln gehört Facade → PartFacade FacadeRow. Sie unterteilt eine Fassade in Erdgeschoss und einen oberen Teil. Diese Konfiguration tritt häufig auf, wenn im unteren Stockwerk Läden oder Restaurants sind. Ihre Fensterstruktur weicht häufig stark von der im oberen Bereich ab. In dieser Situation tritt auch häufig ein Wechsel der Fassadenfarbe vom unteren zum oberen Bereich auf. Der untere Bereich kann dann durch die Regeln FacadeRow → FacadeElement ... FacadeElement weiter in Bereiche unterteilt werden, die jeweils ein Fassadenelement wie z.B. Fenster enthalten können. Eine weitere Unterteilungsregel ist Facade → SplitPartFacade FacadeColumn. Hier wird eine Spalte aus einem zusammenhängenden Bereich ausgeschnitten. SplitPartFacade symbolisiert zwar einen räumlich nicht verbundenen Bereich, die Strukturen, die später definiert werden, sind aber durchgängig. Das tritt häufig dann auf, wenn eine regelmäßige Struktur auf einer Fassade z.B. durch ein Treppenhaus unterbrochen wird. Liegt das Treppenhaus am Rand der Fassade, wird die Regel Facade → FacadeColumn PartFacade bzw. Facade → PartFacade FacadeColumn verwendet.

Die Regel Facade → SymmetricFacadeSide FacadeColumn enthält etwas von beiden Regeltypen. Sie unterteilt eine Fassade in eine Mittelspalte FacadeColumn und zwei Seitenbereiche SymmetricFacadeSide, die symmetrisch sind. Hierbei wird nur der linke Seitenbereich modelliert. Der rechte Teil ist durch die Symmetrie gegeben. Der mittlere Teil wird jedoch noch einmal vom Rest getrennt, da er nicht zur Struktur passt. Dies ist der Fall, wenn die Symmetrieachse durch Fassadenelemente hindurch geht. Würde dann keine separate Mittelspalte modelliert werden, müssten im Seitenteil halbe Fenster modelliert werden. Außerdem ist diese separat modellierte Mittelspalte sehr nützlich, da sie häufig um ein halbes Stockwerk in der Höhe verschoben ist.

Eine reine Ersetzungsregel ist Facade → SymmetricFacadeSide, bei der die Fassade durch ihre linke Hälfte dargestellt wird und die rechte dazu symmetrisch ist. Diese Regel ähnelt der oben beschriebenen, mit der Ausnahme, dass keine Fassadenelemente auf der Symmetrieachse liegen. Die Regel Facade → RepeatedFacade modelliert einen Teil einer Fassade und enthält Informationen darüber, wie oft dieser Teil wiederholt wird. Außerdem gibt es noch Regeln, die die Fassade auf Gitter von Fassadenelementen abbilden. Hier gibt es für jeden der acht Gittertypen eine Regel. Ein Beispiel ist Facade → RegularWindowDoorGrid. Alle diese Strukturregeln existieren nicht nur für Facade auf der linken Seite, sondern auch für PartFacade, SymmetricFacadeSide und teilweise für RepeatedFacade. Eine Liste aller Regeln zeigt Tabelle 5.4.

5.1.3 Der Ableitungsprozess

Im Ableitungsprozess werden ausgehend vom Startsymbol Facade verschiedene Regeln angewandt. Da die Grammatik nicht deterministisch ist, werden jeder Regel Wahrscheinlichkeiten zugewiesen. Die Wahrscheinlichkeiten wurden auf Basis von Häufigkeiten festgelegt, welche durch die Analyse von Fassadenbildern ermittelt wurden. Bei der Ableitung wird dann in jedem Schritt anhand dieser Wahrscheinlichkeiten eine Regel ausgesucht.

Während der Ableitung wird ein Ableitungsbaum aufgestellt. Dieser liefert das Modell der Fassade. Er enthält alle Strukturinformationen und Parameter, so dass die Fassade vollständig beschrieben ist. Abbildung 5.2 zeigt beispielhaft den Aufbau eines Ableitungsbaums. Dabei wurden nacheinander folgende Regeln verwendet:

- Facade → SymmetricFacadeSide FacadeColumn,
- SymmetricFacadeSide → RegularIdenticalWindowGrid,
- FacadeColumn → RegularIdenticalWindowDoorGrid.

Facade	→	PartFacade FacadeRow
Facade | PartFacade	→	SymmetricFacadeSide FacadeColumn
Facade | PartFacade	→	SymmetricFacadeSide
Facade	→	PartFacade FacadeColumn
Facade	→	FacadeColumn PartFacade
Facade	→	SplitPartFacade FacadeColumn
Facade | PartFacade | SymmetricFacadeSide	→	RegularIdenticalWindowDoorGrid
FacadeRow |FacadeColumn | RepeatedFacade	→	RegularIdenticalWindowDoorGrid
SplitPartFacade	→	RegularIdenticalWindowDoorGrid
Facade | PartFacade | SymmetricFacadeSide	→	RegularIdenticalWindowGrid
FacadeRow |FacadeColumn | RepeatedFacade	→	RegularIdenticalWindowGrid
SplitPartFacade	→	RegularIdenticalWindowGrid
Facade | PartFacade | SymmetricFacadeSide	→	RegularWindowDoorGrid
FacadeRow |FacadeColumn | RepeatedFacade	→	RegularWindowDoorGrid
SplitPartFacade	→	RegularWindowDoorGrid
Facade | PartFacade | SymmetricFacadeSide	→	RegularWindowGrid
FacadeRow |FacadeColumn | RepeatedFacade	→	RegularWindowGrid
SplitPartFacade	→	RegularWindowGrid
Facade | PartFacade | SymmetricFacadeSide	→	IdenticalWindowDoorGrid
FacadeRow |RepeatedFacade	→	IdenticalWindowDoorGrid
SplitPartFacade	→	IdenticalWindowDoorGrid
Facade | PartFacade | SymmetricFacadeSide	→	IdenticalWindowGrid
FacadeRow |RepeatedFacade	→	IdenticalWindowGrid
SplitPartFacade	→	IdenticalWindowGrid
Facade | PartFacade | SymmetricFacadeSide	→	WindowDoorGrid
FacadeRow | RepeatedFacade	→	WindowDoorGrid
SplitPartFacade	→	WindowDoorGrid
Facade | PartFacade | SymmetricFacadeSide	→	WindowGrid
FacadeRow | RepeatedFacade	→	WindowGrid
SplitPartFacade	→	WindowGrid
Facade | PartFacade |	→	RepeatedFacade
SymmetricFacadeSide | FacadeRow	→	RepeatedFacade
FacadeRow	→	FacadeElement ... FacadeElement
FacadeElement	→	Window
FacadeElement	→	Door
FacadeElement	→	ShopWindow
FacadeElement	→	DoorWay

Tabelle 5.4: Regeln der Fassadengrammatik.

Im ersten Schritt muss die Breite der mittleren Spalte FacadeColumn festgelegt werden. Anschließend werden die Seitenteile SymmetricFacadeSide weiterbearbeitet. Hier werden die Anzahl an Zeilen und Spalten sowie die Größe der Fenster bestimmt. Zuletzt wird die mittlere Spalte abgeleitet. Dazu wird die Anzahl an Zeilen festgelegt. Durch das Symbol FacadeColumn ist bereits festgelegt, dass es nur eine Spalte geben kann. Zusätzlich werden die Größen von Fenstern und Tür gewählt.

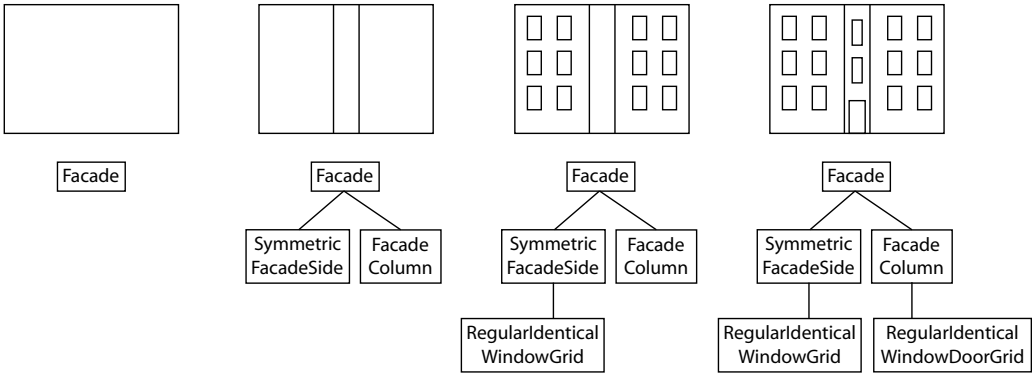


Abbildung 5.2: Beispiel einzelner Schritte der Ableitung einer Fassade mit der Fassadengrammatik.

5.2 Untersuchung modellierbarer Fassaden

In diesem Abschnitt wird untersucht, wie viele unterschiedliche Fassadenarten sich mit der oben beschriebenen Grammatik modellieren lassen. Wie bereits erwähnt, steht eine Regeln für eine Gruppe von Regeln mit unterschiedlichen Parametern. So kann eine Regeln Facade → IdenticalWindowDoorGrid verschiedene Ergebnisse produzieren. Zum einen variiert die Zahl der Zeilen und Spalten, in denen die Fenster angeordnet sind. Hierbei ist die maximale Anzahl auch von Breite und Höhe der Fassade abhängig. Zum anderen kann die Position der Tür wechseln. Außerdem sind die Größen von Fenstern und Tür festzulegen. Diese Freiheitsgrade sollen bei dieser Überlegung aber nicht betrachtet werden. Fassaden, die sich nur in diesen Eigenschaften unterscheiden, werden nicht unterschieden. Als unterschiedliche Typen werden dann z.B. verschiedene Gittertypen oder abweichende Unterteilungen angesehen. Die verschiedenen Typen ergeben sich also aus der Struktur des Ableitungsbaums.

Grammatiksymbol	Anzahl direkt ableitbarer Worte	Anzahl ableitbarer Worte
Facade	15	1760
PartFacade	11	96
SymmetricFacadeSide	9	16
FacadeRow	9	9
FacadeColumn	4	4
RepeatedFacade	8	8
SplitPartFacade	8	8
FacadeElement	4	4

Tabelle 5.5: Anzahl ableitbarer Worte.

Tabelle 5.5 listet für die nicht terminalen Symbole der Grammatik die Anzahl der direkt ableitbaren Wörter und der ableitbaren Wörter auf. Diese können aus der Liste der Regeln in Tabelle 5.4 entnommen werden. Es ergibt sich eine Zahl von 1760 verschiedenen Fassadentypen. Hierbei wurde nicht berücksichtigt, dass die Regel zur Ableitung von FacadeRow auf FacadeElement eine variable Anzahl von Elementen auf der rechten Seite zulässt. Für jedes Fassadenelement gibt es wiederum vier weitere mögliche Ableitungen. Damit ergäben sich für FacadeRow neben der Ableitung auf verschiedene Gittertypen $4^{\#\text{ FacadeElement}}$ weitere Ableitungsmöglichkeiten. Bei einer angenommenen mittleren Anzahl an FacadeElement von fünf würde das die Zahl unterschiedlicher Fassadentypen auf 99968 erhöhen. Da hierbei die Ähnlichkeit zwischen den einzelnen Typen aber sehr hoch ist, wurden diese Fälle als ein Typ gezählt. Abbildung 5.3 zeigt beispielhaft einige der modellierbaren Fassaden mit den zugehörigen Ableitungsbäumen.

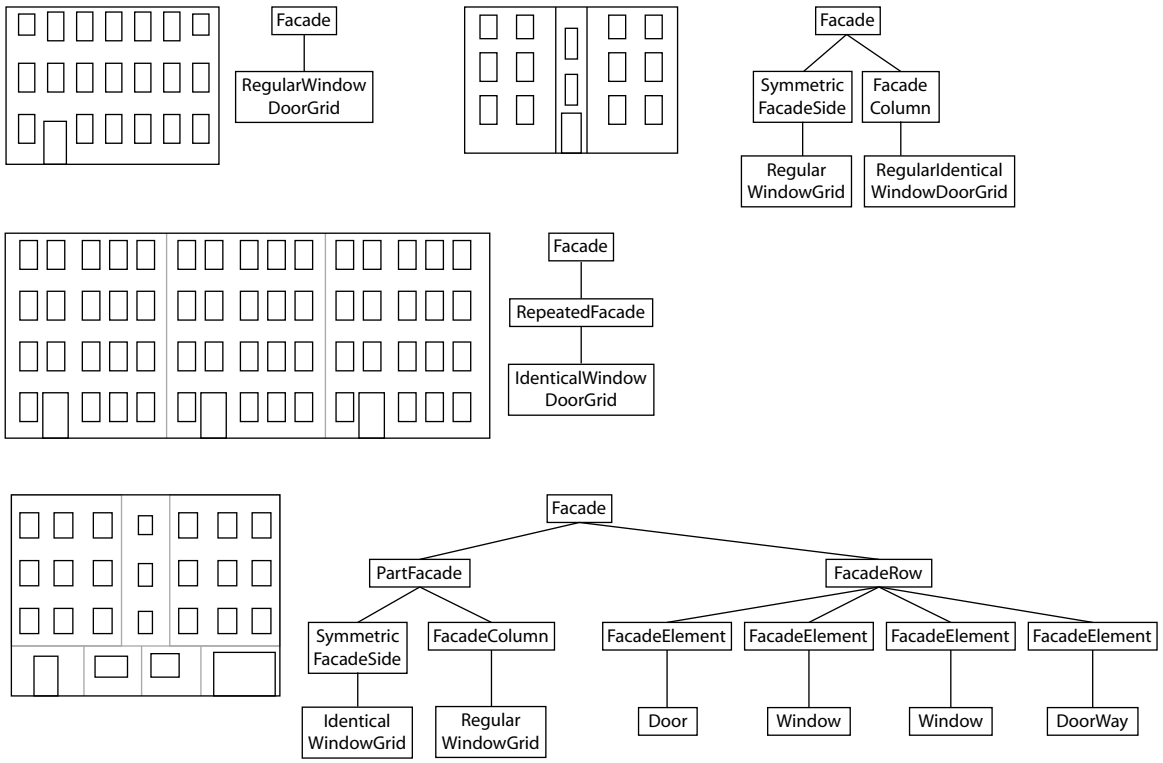


Abbildung 5.3: Verschiedene Fassaden mit Ableitungsbaum, die mit der Fassadengrammatik modelliert werden können.

5.3 Das Programm FacadeModeler

Um die Fassadengrammatik zu testen, wurde das Programm *FacadeModeler* entwickelt (Uden, 2008; Ripperda und Brenner, 2009a). Hiermit kann der Benutzer anhand eines Fassadenbildes die Fassaden manuell nach Regeln der Grammatik modellieren. Indem einige Testpersonen einige ausgewählte Fassaden modellierten, konnte untersucht werden, welche Schwächen die Grammatik noch aufzeigt.

Das Programm stellt eine graphische Benutzeroberfläche (siehe Abbildung 5.4) zur Verfügung, die dem Benutzer erlaubt, hauptsächlich durch Mausinteraktion Fassaden zu modellieren. Dabei unterstützt das Programm den Benutzer bei der Modellierung, indem es ihm in jedem Schritt die Regeln anzeigt, die im aktuellen Zustand des Ableitungsbaumes denkbar sind. Nach Auswahl einer Regel wird die Veränderung in der Fassade im Bild dargestellt. Das geschieht zunächst mit Standardwerten für die Parameter. Die richtigen Werte können anschließend eingestellt werden. Einige Parameter wie z.B. die Zahl der Zeilen und Spalten in einem Gitter, die Art des Gitters, oder ob eine Tür vorhanden ist, können in einem Spezifikationsfeld eingegeben oder ausgewählt werden. Die Position und Größe von Fassadenelementen kann direkt durch das Verschieben oder Verändern der Elemente auf dem Fassadenbild manipuliert werden.

Im Beispiel sind SymmetricFacadeSide und FacadeColumn modelliert. Der Teil SymmetricFacadeSide wird weiter zu einem Gitter modelliert. Im Bild ist die Möglichkeit zu sehen, über eingeblendete vertikale Linien die Abstände in x-Richtung zu variieren. Verschiebt man eine Linie auf einer Seite, so erfolgt dieselbe Veränderung symmetrisch auf der anderen, wie es die Regel SymmetricFacadeSide vorschreibt. Während der Modellierung im Bild wird der Ableitungsbaum ständig aktualisiert. Er ist unten links im Bild zu sehen[1]. Der Baum und das modellierte Bild können anschließend gespeichert werden.

Für dieses Programm wurden einige Regeln zu Gruppen zusammengefasst. So bilden z.B. die Regeln Facade → RepeatedFacade, PartFacade → RepeatedFacade und SymmetricFacadeSide → RepeatedFacade eine Gruppe. Für den Benutzer ist der Unterschied auf der linken Seite nicht entscheidend, ihm wird vom Programm vorgegeben, welche Regel er anwenden kann und der Effekt aller Regeln der Gruppe ist der gleiche. Ebenso wurden die Gitterregeln in zwei Gruppen zusammengefasst: eine Gruppe für Gitter mit regelmäßigen

[1] Das Symbol FacadeColumn hieß zur Zeit der Entstehung des *FacadeModeler*s noch SymmetricFacadeMiddle, wurde dann aber im Zuge einer Umstrukturierung umbenannt.

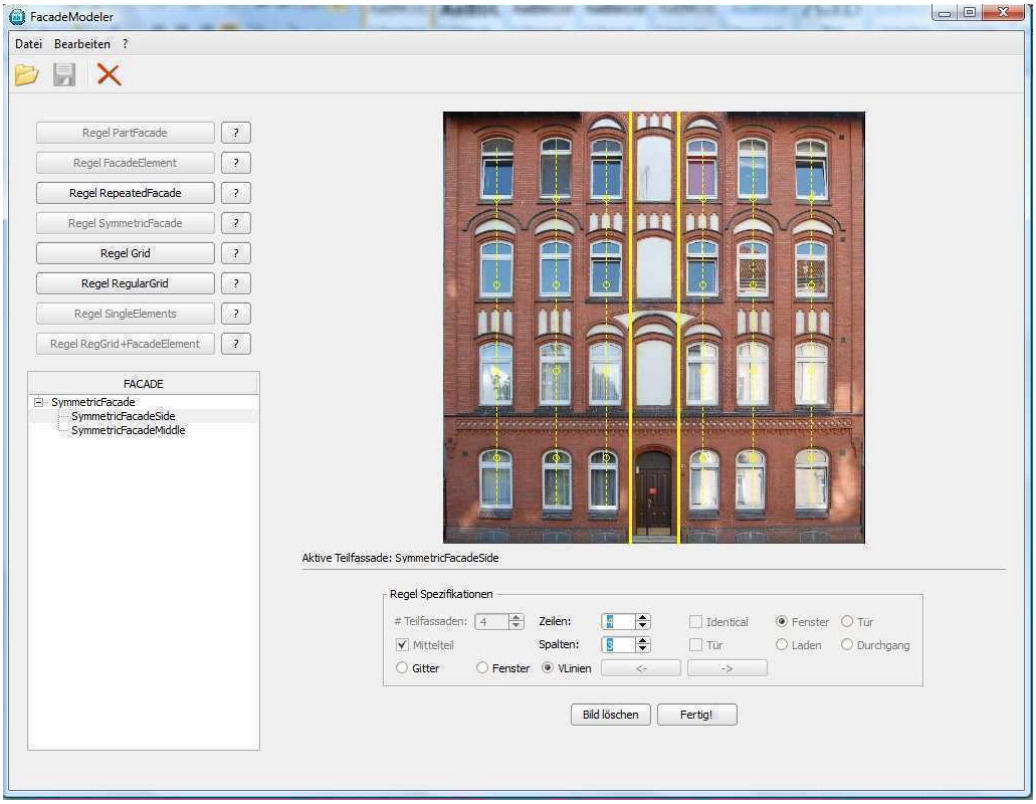


Abbildung 5.4: Graphische Benutzeroberfläche des Programms FacadeModeler.

Abbildung 5.5: Verschiedene Modelle einer Fassade.

Abständen und eine für unregelmäßige Gitter. Die weiteren Unterschiede kann der Benutzer über das Spezifikationsfeld eingeben. Hier kann er z.B. angeben, ob die Fenster identisch sind, oder eine Tür existiert.

Ein Test mit verschiedenen Nutzern hat gezeigt, dass verschiedene Personen die gleiche Fassade auf unterschiedliche Weise modellieren. Abbildung 5.5 zeigt drei mögliche Modelle von einer der Testfassaden. Einige Benutzer modellierten das Erdgeschoss getrennt und den oberen Teil als symmetrisch, wobei sie die zugemauerten Fenster außer Acht ließen. Andere haben eine symmetrische Fassade mit Mittelteil modelliert und in die einzelnen Bereiche Gitter mit Fenstern bzw. einer Tür gesetzt. Die dritte Möglichkeit ist ein unregelmäßiges Gitter von Fenstern mit einer Tür. Schon dieses Beispiel zeigt, dass es keine eindeutige Modellierung für eine Fassade gibt. Deshalb ist es wichtig, eine Methode zur Bewertung der Modelle zu entwickeln und diese für die automatische Modellierung zu verwenden.

6 Automatische Ableitung von Fassaden mittels reversible jump Markov Chain Monte Carlo

Die Fassadengrammatik ermöglicht es, verschiedene Fassaden zu generieren. Diese entstehen aber zunächst ohne Bezug zur Szene, welche in Form von Bild- bzw. Laserscanning-Daten vorliegt. Um das Modell zu finden, welches am besten zu den gemessenen Daten passt, wird die Erstellung des Ableitungsbaums in einen stochastischen Prozess integriert (Ripperda, 2008b). Die in Abschnitt 4.2.4 beschriebene rjMCMC Methode ermöglicht es, das Modell zu finden, welches die größte Wahrscheinlichkeit unter den gegebenen Scan- und Bilddaten hat.

6.1 Ablauf des Rekonstruktionsverfahrens

In diesem Abschnitt wird der Ablauf der Rekonstruktion skizziert. Er ist grob im Aktivitätsdiagramm in Abbildung 6.1 gegeben. Im Folgenden werden die einzelnen Schritte genauer beschrieben. Eingabedaten für die Rekonstruktion sind ein Tiefenbild und ein Orthophoto. Aus diesen wird in einem Vorverarbeitungsschritt ein Clusterbild berechnet, das später für die Rekonstruktion verwendet wird.

Für die Rekonstruktion nach dem rjMCMC Verfahren bilden die möglichen Ableitungsbäume der Grammatik den Zustandsraum der Markov-Kette. Die Übergänge zwischen den Zuständen werden durch die Regeln der Grammatik definiert. Als Startzustand wird der Ableitungsbaum verwendet, der nur aus dem Symbol Facade besteht. Aus den Regeln, die auf diesen Baum anwendbar sind, wird entsprechend der Vorschlagswahrscheinlichkeit eine Regel zufällig ausgewählt. Anschließend werden die für die Regel benötigten Parameter nach den entsprechenden Verteilungen gezogen. Dann wird die Regel angewandt und der veränderte Baum bewertet. Aus den Bewertungen des alten und neuen Ableitungsbaums wird die Akzeptanzwahrscheinlichkeit berechnet und entschieden, ob die Veränderung angenommen wird. Dementsprechend wird der veränderte Baum oder der alte Zustand für die nächste Iteration verwendet.

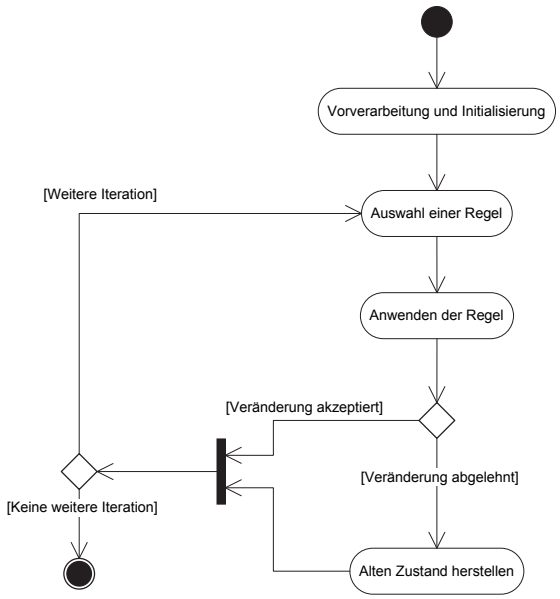


Abbildung 6.1: Ablauf der Rekonstruktion.

Die Rekonstruktion basiert auf Wissen über Fassadenstrukturen. Dieses geht einerseits in Form der Grammatik in das Verfahren ein, andererseits über die Verteilungen von Fassadenparametern, die durch die Analyse vieler Fassadenbilder gewonnen wurden.

6.2 Vorverarbeitung der Daten

Als Eingabedaten für die Rekonstruktion dienen ein Tiefenbild und ein Orthophoto einer Fassade. Die Daten werden mit einem terrestrischen Laserscanner, der zusätzlich mit einer Kamera versehen ist, aufgenommen. Die gemessenen Daten werden mit der zum Scanner gehörenden Software RiScanPro (Riegl, 2004) vorverarbeitet. Der Bereich der Fassade wird manuell ausgeschnitten. Aus dieser Punktwolke und den Fotos wird ein Orthophoto und ein Tiefenbild berechnet. Diese Daten dienen als Eingabe für das Rekonstruktionsverfahren.

In der Vorverarbeitung werden die Eingabedaten klassifiziert, damit später die Bewertungsverfahren auf die Daten angewandt werden können. Ergebnis dieses Schrittes sind klassifizierte Bilder, die nur eine geringe Anzahl an Klassen besitzen.

Für das Clustering wird das Verfahren k-Means (MacQueen, 1967) verwendet. Dabei handelt es sich um ein Clustering-Verfahren, das die Daten in höchstens k Klassen einteilt. Hierbei werden im ersten Schritt zufällig k Schwerpunkte gewählt. Dann wird jeder Datenpunkt dem Cluster zugeordnet, dessen Schwerpunkt er am nächsten liegt. Aus den so ermittelten Gruppen werden die neuen Schwerpunkte berechnet. Mit den neuen Schwerpunkten werden die Daten neu aufgeteilt. Solange sich bei der erneuten Aufteilung Änderungen ergeben, wird das Verfahren fortgesetzt.

Um die Eingabedaten des Rekonstruktionsverfahrens zu klassifizieren, werden vier Kanäle verwendet. Das sind die Farbwerte (R,G,B) und der Tiefenwert. Für die Zahl der Cluster hat sich drei als gut herausgestellt. Bei drei Klassen können zwei Bereiche in der Fassade und ein Bereich für Fenster und Türen auftreten. Wählt man eine größere Zahl an Klassen, so gibt es oft eine zu feine Unterteilung.

Abblidung 6.2 zeigt ein Orthophoto und Clusterbilder einer Fassade. Die Cluster wurden von links nach rechts aus Farbinformationen, dem Tiefenbild und einer Kombination aus beiden berechnet.

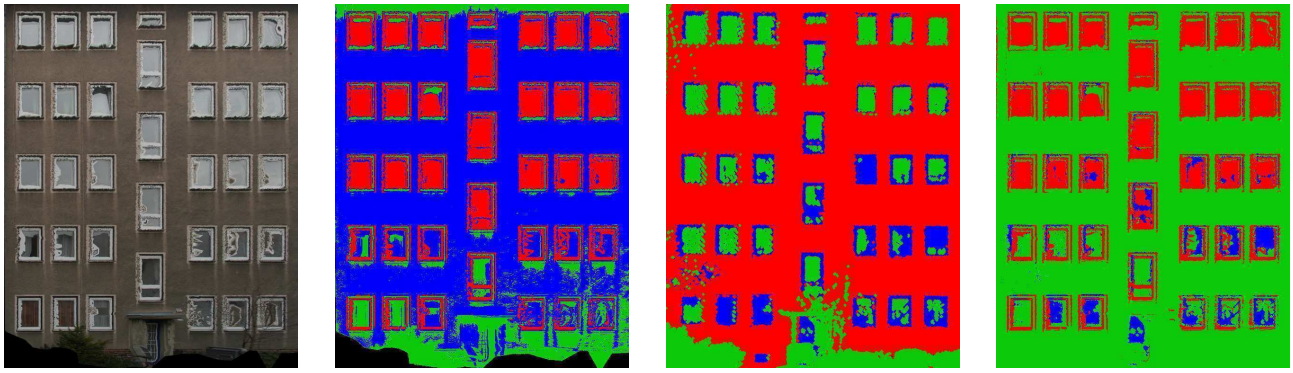


Abbildung 6.2: Beispielfassade und dazu berechnete Clusterbilder basierend auf Farbinformation, Tiefeninformation und einer Kombination aus beiden.

An Stelle des Clusterbildes können auch manuell segmentierte Fassadenbilder eingesetzt werden. In Abschnitt 7.1 wird diese Eigenschaft genutzt, um das Rekonstruktionsverfahren mit einer größeren Anzahl an Fassaden zu testen.

6.3 Integration von Fassadengrammatik und reversible jump Markov Chain Monte Carlo

Das Markov Chain Monte Carlo Verfahren steuert als stochastischer Prozess die Ableitung eines Fassadenmodells mithilfe der Grammatik so, dass das Modell mit der höchsten Wahrscheinlichkeit bezüglich der Daten gefunden wird. Dazu bilden alle Wörter der Grammatik und Zusammensetzungen aus terminalen und nicht terminalen Zeichen, also alle möglichen Zustände des Ableitungsbaums, den Zustandsraum der Markov-Kette. Abbildung 6.3 zeigt beispielhaft einen kleinen Teil des Zustandsraums der Markov-Kette. In den Kreisen sind verschiedene Zustände zu sehen. Oben an zweiter Stelle ist das Symbol FACADE, durch ein Rechteck dargestellt, zu sehen. Von hier aus gibt es Übergänge zu verschiedenen veränderten Fassaden. Übergänge zwischen den einzelnen Zuständen werden durch die Grammatik definiert. Mögliche Übergänge sind die Anwendung einer Grammatikregel, das Rückgängigmachen einer Grammatikregel oder das Verändern von Parametern. Der vollständige Zustandsraum ist wesentlich größer als hier dargestellt und kann nicht vollständig aufgezeigt werden. Die Größe des Zustandsraumes liegt vor allem an den Übergängen durch Veränderung der Parameter. Hier

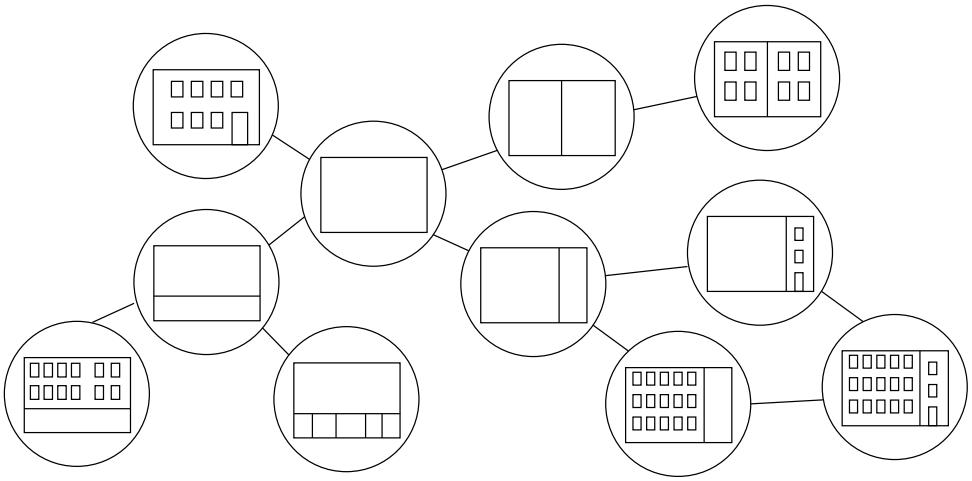


Abbildung 6.3: Ausschnitt aus dem Zustandsraum der Markov-Kette.

sind viele Kombinationen der Parameter Anzahl an Zeilen und Spalten im Fenstergitter sowie der Position und Größe der Fenster möglich.

Für die auf Grammatiken basierende Fassadenrekonstruktion ist es wichtig, dass die Dimension der Zustände sich bei Übergängen ändern darf, da sich die Anzahl der Struktur- und Fassadenelemente während der Rekonstruktion ständig verändert. Da dieses in einem normalen MCMC Prozess nicht vorgesehen ist, wird hier reversible jump MCMC verwendet, welches auch Dimensionsänderungen des Modells zulässt.

Damit die Rekonstruktion gegen die gewünschte Verteilung von Fassaden konvergiert, muss die Markov-Kette gewisse Anforderungen erfüllen. Satz 4.1 sagt, dass die Markov-Kette einen endlichen Zustandsraum haben muss, π, also die Verteilung der Fassaden als invariante Verteilung besitzen muss und irreduzibel und aperiodisch sein muss. Ein endlicher Zustandsraum ist durch die endliche Anzahl an Grammatiksymbolen gegeben, die auch nicht rekursiv miteinander verknüpft werden können. Die Kombination der Parameter mit diskretem Wertebereich, der durch die Dimension der Fassade eingeschränkt ist, führt auch nur zu endlich vielen Zuständen. Irreduzibel ist die Markov-Kette dadurch, dass jeder Zustand von jedem anderen mit positiver Wahrscheinlichkeit erreicht werden kann. Diese Eigenschaft muss also durch die Vorschlagswahrscheinlichkeiten gewährleistet werden. Die Aperiodizität ergibt sich daraus, dass sich durch die Übergänge, die die Parameterwerte verändern, mögliche Periodizitäten aufheben. Bleibt also die Existenz der invarianten Verteilung π zu zeigen. Dieses wird gewährleistet, indem die Akzeptanzwahrscheinlichkeit wie in Formel 4.6 gewählt wird. Damit ist die Eigenschaft der *detailed balance* gegeben, wodurch impliziert wird, dass π eine invariante Verteilung der Markov-Kette ist.

Für die Konstruktion der Markov-Kette müssen die Vorschlagswahrscheinlichkeiten für die Übergänge (Abschnitt 6.4) und die Akzeptanzwahrscheinlichkeit (Abschnitt 6.5) definiert werden. Anschließend werden mittels Simulated Annealing (Abschnitt 6.6) noch Verbesserungen des Verfahrens vorgenommen.

6.4 Bestimmung der Vorschlagswahrscheinlichkeiten

Ein Zustand, also ein Ableitungsbaum, wird mit θ bezeichnet. Um eine Markov-Kette mit Zuständen im Wertebereich von θ, also allen möglichen Ableitungsbäumen, zu erzeugen, muss man einen Übergangskern $J(\theta_t|\theta_{t-1})$ definieren. Dies ist eine $k \times k$-Matrix, wobei k die Anzahl aller möglichen Zustände ist. J weist jedem Wechsel vom Zustand θ_{t-1} in einen anderen Zustand θ_t eine Wahrscheinlichkeit zu. Im Rekonstruktionsprozess sind diese Übergänge durch die Regeln der Grammatik definiert. Mit dem Übergangskern werden Veränderungen vorgeschlagen, die dann anhand einer Akzeptanzwahrscheinlichkeit angenommen oder abgelehnt werden.

Im Rekonstruktionsverfahren gibt es verschiedene Arten von Übergängen. Die erste ist die Anwendung einer Grammatikregel. Hierbei werden zusätzlich Parameter vorgeschlagen. Diese können z.B. die Anzahl an Zeilen und Spalten in einem Gitter oder die Fenstergröße sein. Die Verteilungen dieser Parameter ist entscheidend dafür, ob die Veränderung angenommen wird. Diese Art von Übergängen wird auch Split-Operation genannt, da die Grammatikregel die Fassade unterteilt.

Um zu gewährleisten, dass die Markov-Kette reversibel bleibt, ist jede Regel auch in die entgegengesetzte Richtung anwendbar. Ein Übergang dieser Art wird als Reverse-Operation bezeichnet. Dieses Art Übergang

bildet einen Unterschied zur Vorgehensweise von Wonka u. a. (2003), sie ist aber notwendig für die Kombination der Grammatik mit dem rjMCMC Prozess. So wird es möglich, viele verschiedene Lösungen auszuprobieren, aber immer wieder zurückkehren zu können.

Die zweite Möglichkeit des Zustandswechsels ersetzt nicht die Symbole, sondern verändert die Parameter und wird deshalb Change-Operation genannt. Z.B. kann die Anzahl an Zeilen oder Spalten in einem Gitter von Fenstern oder die Fenstergröße verändert werden.

Um dieses Verhalten als Markov-Kette zu formulieren, werden zwei Arten von Verteilungen definiert. Zum einen die Wahrscheinlichkeit eine bestimmte Regel zu wählen und zum anderen die Verteilungen aller zugehörigen Parameter. Die folgenden Abschnitte beschreiben, wie diese Wahrscheinlichkeiten ermittelt werden.

6.4.1 Ermittlung der Regelwahrscheinlichkeiten

Die Wahrscheinlichkeiten für die Anwendung der jeweiligen Regeln werden aus Fassadenbildern gewonnen. Hierzu wurde das Auftreten der zu den Regeln korrespondierenden Strukturen in 451 Fassadenbildern gezählt und die relative Häufigkeit des Auftretens der Regel als Wahrscheinlichkeit zugeordnet. Die hierbei ermittelten Wahrscheinlichkeiten sind im Anhang in Tabelle A.1 gegeben. Die ermittelten Wahrscheinlichkeiten entsprechen den Wahrscheinlichkeiten für Split-Operationen. Der gegensätzlichen Anwendung der Regeln wird die gleiche Wahrscheinlichkeit zugeordnet, damit eine Umkehrung der Zustände gleichwahrscheinlich ist. Die Change-Operationen sollen mit höherer Wahrscheinlichkeit angewandt werden. Dies ist sinnvoll, weil viel mehr Iterationen benötigt werden, um die exakte Position von Objekten oder die richtige Anzahl an Fenstern zu finden.

6.4.2 Ermittlung von Fassadenparametern

Um die Verteilung der einzelnen Parameter in den Regeln zu bestimmen, wurden ebenfalls Fassadenbilder untersucht. Mithilfe von ArcGIS wurden in ca. 400 Bildern Fenster, Türen, Schaufenster, Einfahrten und bei farblichen Unterschieden auch Trennlinien zwischen Gebäudeteilen digitalisiert. Aus diesen Daten wurden Verteilungen für Fassadenparameter ermittelt (Ripperda, 2008a). Die Verarbeitungskette ist in Abbildung 6.4 dargestellt. Das Fassadenbild wird zunächst entzerrt und auf den Bereich der Fassade zugeschnitten. Über eine gemessene Referenzstrecke kann der Maßstab festgelegt werden. Anschließend werden die Fassadenelemente digitalisiert. Dabei werden für jedes Bild Vektordaten zu den Elementen Fenster, Tür, Schaufenster, Einfahrt, Umrandung, Trennlinie, Garage, Balkon, Vorsprung und Referenzlinie erzeugt. Aus den Vektordaten werden mittels eines Python-Skripts die Verteilungen der Fassadenparameter bestimmt. Das sind die Verteilungen der Fassadenelemente Fenster, Tür, Einfahrt und Schaufenster.

Hierzu werden die Höhe und Breite der jeweiligen Objekte sowie das Verhältnis der beiden Größen untersucht. Des Weiteren wird die Anzahl der Zeilen und Spalten pro Fassade und ihr Abstand untereinander untersucht. Um die Verteilung von Unterteilungen in Erdgeschoss und obere Stockwerke zu erhalten wurden vorhandene Farbwechsel digitalisiert. Diese gehen oft mit einem Wechsel der Struktur einher. Aus ihnen kann die Verteilung der Trennlinien abgeleitet werden, auch wenn diese nicht immer zusammen mit einem Farbwechsel auftritt.

Die Verteilungen der Parameter werden aus dem Histogramm über die Werte in den Fassadenbildern berechnet. Die relative Häufigkeit wird als Wahrscheinlichkeit verwendet. Abbildung 6.5 zeigt die ermittelten relativen Häufigkeiten der einzelnen Parameter. Die Diagramme 6.5a und 6.5b zeigen die Verteilung von Breite und Höhe von Fenstern. Für einen Großteil der Fenster liegt die Breite zwischen 0,5 m und 1,5 m und die Höhe zwischen

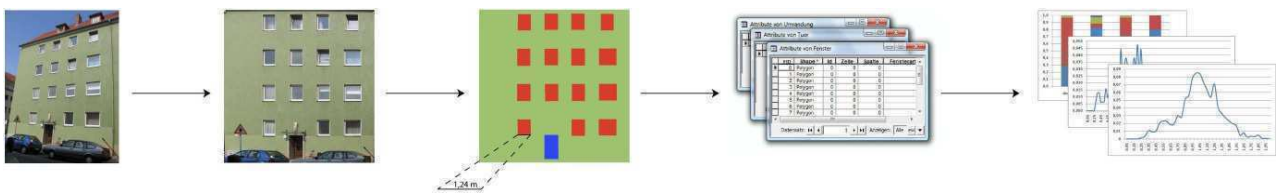
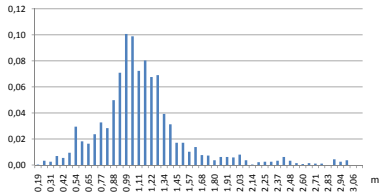
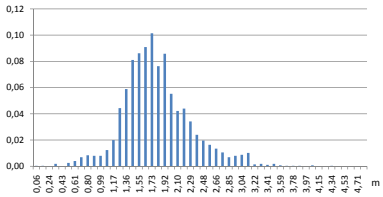


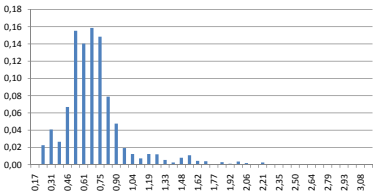
Abbildung 6.4: Ablauf der Bestimmung von Verteilungen aus Fassadenbildern. Zunächst wird das Fassadenfoto entzerrt und auf den Bereich der Fassade zugeschnitten. Anschließend wird es über eine Referenzstrecke mit einem Maßstab versehen und die Fassadenelemente werden in ArcGIS digitalisiert. Aus den resultierenden Vektordaten werden abschließend die Verteilungen berechnet.



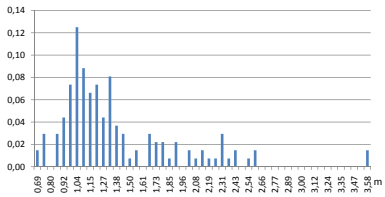
(a) Fensterbreite.



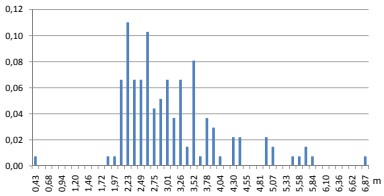
(b) Fensterhöhe.



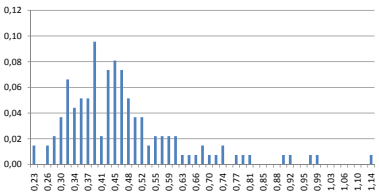
(c) Fensterbreite zu -höhe.



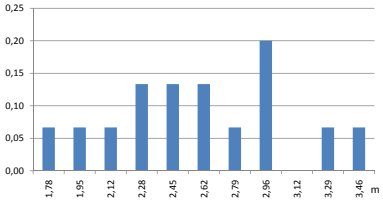
(d) Türbreite.



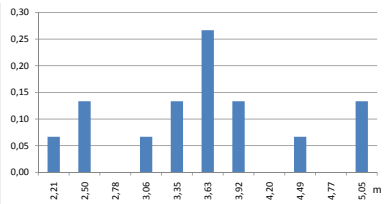
(e) Türhöhe.



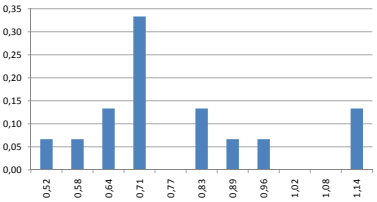
(f) Türbreite zu -höhe.



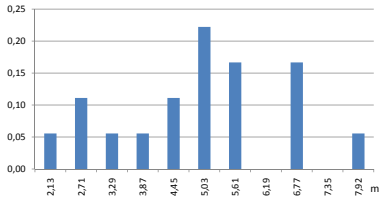
(g) Einfahrtbreite.



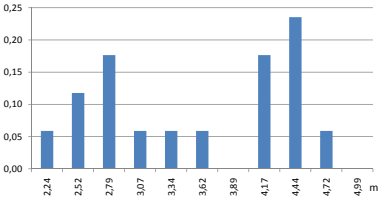
(h) Einfahrthöhe.



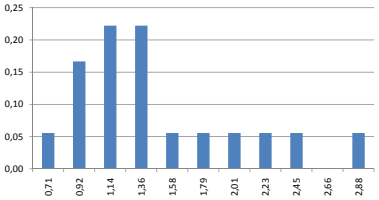
(i) Einfahrtbreite zu -höhe.



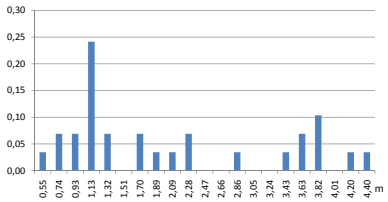
(j) Schaufensterbreite.



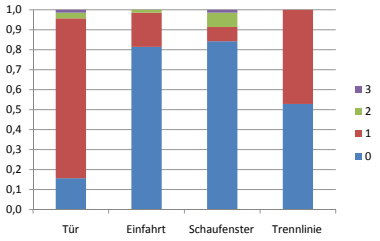
(k) Schaufensterhöhe.



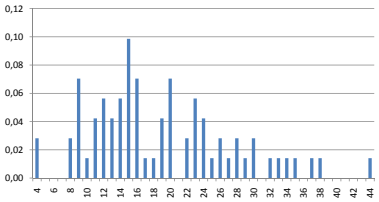
(l) Schaufensterbreite zu -höhe.



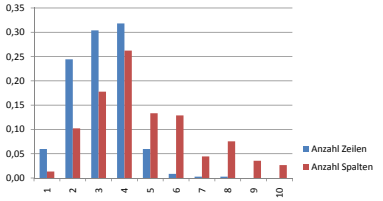
(m) Trennlinienhöhe.



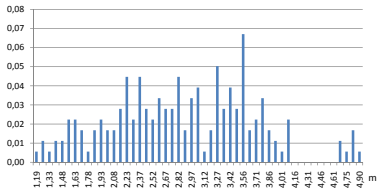
(n) Anzahl an Elementen auf einer Fassade.



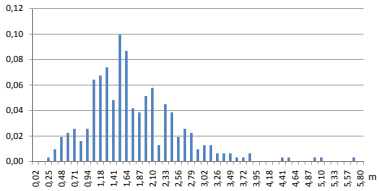
(o) Anzahl an Fenstern pro Fassade.



(p) Anzahl der Zeilen (blau) und Spalten (rot) in einer Fassade.



(q) Abstand zwischen Zeilen.



(r) Abstand zwischen Spalten.

Abbildung 6.5: Ermittlung von relativen Häufigkeiten anhand von manuell ausgewerteten Fassadenbildern.

1,2 m und 2,5 m. Für die Rekonstruktion ist das Verhältnis beider Größen ebenso wichtig. Diese Verteilung zeigt Abbildung 6.5c. In den meisten Fällen liegt das Verhältnis zwischen 0,3 und 0,9. Für die Fassadenelemente Tür,

Einfahrt und Schaufenster sind diese Verteilungen in den Diagrammen 6.5d-6.5l gezeigt. Für eine Unterteilung in obere und untere Stockwerke gibt Diagramm 6.5m die Verteilung der Höhe der Trennline an.

Des Weiteren wird die Anzahl der verschiedenen Fassadenobjekte auf einer Fassade untersucht. In Diagramm 6.5n ist für die Elemente Tür, Einfahrt, Schaufenster und Trennline zu sehen, wie häufig diese pro Fassade vorkommen. Diagramm 6.5o zeigt ein Histogramm über die Fensteranzahl pro Fassade. Zum Vorschlagen der Gittergröße wird allerdings nicht diese Verteilung verwendet, sondern die Verteilungen der Anzahl an Zeilen und Spalten. Diese sind im Diagramm 6.5p gegeben. Hier ist zu sehen, dass Häuser mit ein bis acht Stockwerken untersucht wurden. Die Mehrheit der Gebäude hatte drei oder vier Stockwerke. Die Zahl der Fensterspalten variiert zwischen eins und zehn, wobei hier die Mehrheit zwischen drei und sechs liegt.

Neben der Anzahl an Zeilen und Spalten muss auch der Abstand zwischen diesen vorgeschlagen werden. Die Verteilung dieser Werte ist in den Diagrammen 6.5q und 6.5r zu sehen. Die Abstände zwischen Zeilen liegen zum Großteil zwischen 1,2 m und 4,2 m, wobei auch wenige größere Abstände bis zu 4,9 m vorkommen. Bei den Spalten liegen die meisten Abstände zwischen 4,4 m und 3,0 m. Bei diesen Werten ist zu beachten, dass es sich um die Breite bzw. Höhe der Zwischenräume zwischen den Fenstern handelt. Auf diese Werte muss noch die Fensterbreite addiert werden, um den Abstand der Gitterpunkte vorzuschlagen.

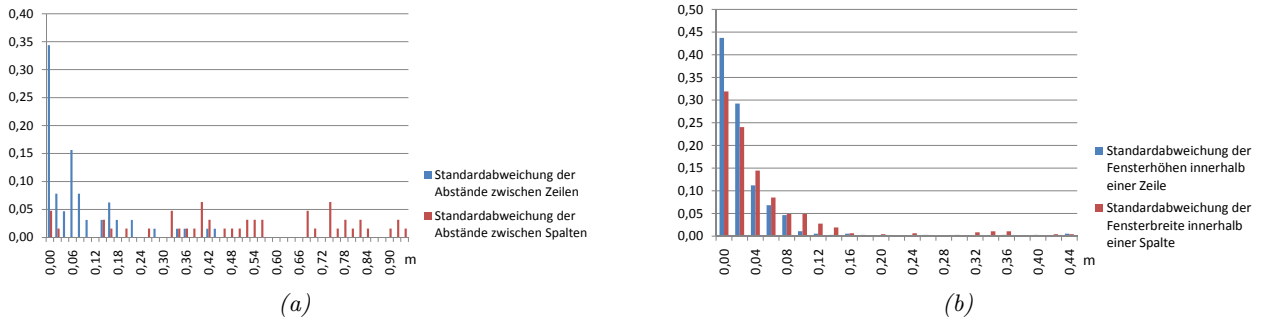


Abbildung 6.6: Histogramm der Standardabweichungen der Abstände zwischen Spalten (rot) bzw. Zeilen (blau) pro Fassade (links) und der Abweichungen von Fensterhöhen (blau) bzw. -breiten (rot) innerhalb einer Zeile bzw. Spalte (rechts).

Die hier ermittelten Werte fließen über die Vorschlagswahrscheinlichkeit in den Rekonstruktionsprozess ein. Die Untersuchung der Abstände zwischen Zeilen bzw. Spalten gibt jedoch noch weitere Informationen für die Rekonstruktion. Die Standardabweichungen der Abstände der Zeilen bzw. Spalten pro Fassade sind in Abbildung 6.6 gegeben. Sie zeigen, dass der Abstand zwischen Spalten innerhalb einer Fassade deutlich stärker schwankt als der Abstand zwischen den Zeilen. Die Standardabweichung für die Zeilen (blau) häuft sich bei Null und der Maximalwert ist 0,48 m. Im Vergleich dazu ist die größte Abweichung der Spalten (rot) 1,0 m. Das deutet darauf hin, dass der Abstand zwischen Spalten in einer Fassade deutlich flexibler ist als der Abstand der Zeilen. Diese Eigenschaft wird in der Auswahl an Gittersymbolen abgebildet. Es gibt Symbole für Gitter mit variablen Spaltenabstand, die Zeilen sind bei allen Symbolen äquidistant. Abbildung 6.6 (rechts) zeigt ein Histogramm der Abweichung der Fensterhöhen (blau) und -breiten (rot) innerhalb einer Zeile bzw. Spalte. Es zeigt, dass die meisten Abweichungen unter 5 cm liegen und dass die Breite der Fenster wieder stärker variiert als die Höhe.

Bei der Analyse der Fassadenbilder ergaben sich noch weitere Informationen. Abbildung 6.7a zeigt verschiedene Fensterarten, die in den Fassadenbildern vorkommen. Diese treten in unterschiedlichen Größen in den Fassaden auf und sind hier in ihrer Struktur gegeben. Das grau abgebildete Fenster 17 stellt ein zugemauertes Fenster dar. Abbildung 6.7b zeigt die relative Häufigkeit, mit der die Fenster auftreten. Die häufigsten Typen sind die Fensterarten 1, 3 und 7. Mit nur 1,36% sind die Typen 10 und 11 die seltensten.

6.5 Akzeptanzwahrscheinlichkeiten

Die Akzeptanzwahrscheinlichkeit wird so definiert, dass die Markov-Kette gegen die Zielverteilung $p(\theta|D_S D_I)$ konvergiert. Für eine irreduzible, aperiodische Markov-Kette mit dem Übergangskern J ist das der Fall, wenn die Akzeptanzwahrscheinlichkeit folgendermaßen definiert wird:

$$\alpha = \min\left\{1, \frac{p(\theta_t|D_S D_I) \cdot J(\theta_{t-1}|\theta_t)}{p(\theta_{t-1}|D_S D_I) \cdot J(\theta_t|\theta_{t-1})}\right\}.$$

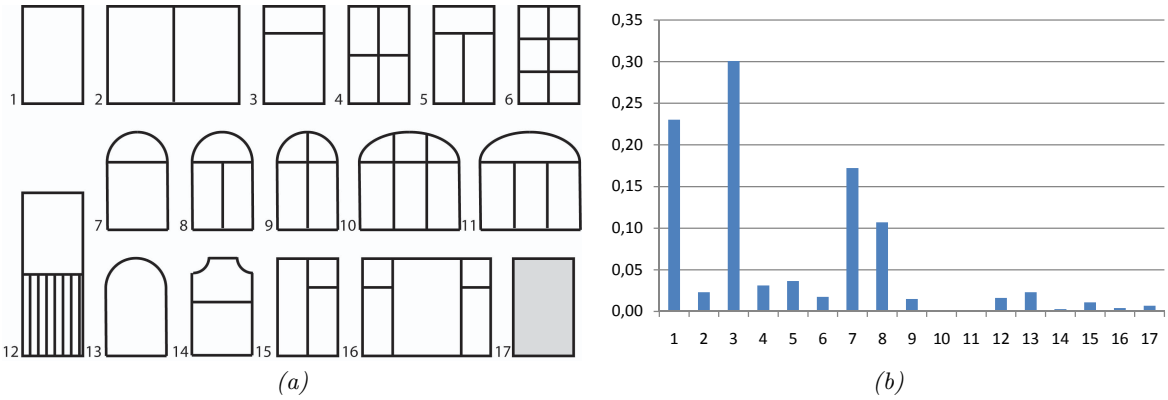


Abbildung 6.7: Verschiedene ermittelte Fensterarten (a) und ihre relativen Häufigkeiten (b).

Die Akzeptanzwahrscheinlichkeit ist abhängig von der unbekannten Verteilung $p(\theta_t|D_S D_I)$. Nach Bayes ist diese jedoch proportional zu $p(D_S D_I|\theta_t) \cdot p(\theta_t)$, dem Produkt aus Likelihood und Prior der Fassade. Der Term $p(D_S D_I)$ ist sowohl im Zähler als auch im Nenner vorhanden und kann daher gekürzt werden. $p(D_S D_I|\theta_t) \cdot p(\theta_t)$ wird mit einer Bewertungsfunktion bestimmt, die das Prinzip der Minimum Description Length (MDL) verwendet.

6.5.1 Bewertungsfunktion auf Basis der Minimum Description Length

Um zu bewerten, wie gut ein Modell zu den Daten passt, wird das Minimum Description Length (MDL) Prinzip verwendet, das Rissanen (1978) entwickelte. Es wird häufig für die Hypothesen-Auswahl verwendet. Dabei wird diejenige Hypothese gewählt, die die kürzeste Beschreibung von der Hypothese und den Daten unter Verwendung der Hypothese besitzt. Für den hier beschriebenen Rekonstruktionsprozess liefert MDL eine Bewertungsfunktion dafür, wie gut die Daten zum Modell passen und berücksichtigt dabei zusätzlich die Komplexität des Modells. Die Berücksichtigung der Modellkomplexität ist wichtig, denn wenn sie vernachlässigt würde, könnten einfache Fassaden durch überflüssig komplexe Strukturen modelliert werden. Eine Fassade mit einem einfachen regelmäßigem Gitter von Fenstern könnte beispielsweise als symmetrisch oder sich wiederholende Fassade modelliert werden. Durch Berücksichtigung der Modellkomplexität kann das verhindert werden.

Nach dem MDL-Prinzip ist das beste Modell θ für die Daten D das Modell mit der kürzesten Beschreibungslänge $L(\theta) + L(D|\theta)$. Dabei ist $L(\theta)$ die Länge, die zur Beschreibung des Modells benötigt wird, und $L(D|\theta)$ die Länge der Beschreibung der Daten D unter Verwendung des Modells θ. Es muss also die Codierungslänge des Modells und der Daten unter Verwendung des Modells bestimmt werden. Nach der Informationstheorie ist diese gleich des Informationsgehaltes des Modells bzw. der Daten bei gegebenem Modell. Der Informationsgehalt einer Nachricht x ergibt sich als negativer Logarithmus der Wahrscheinlichkeit der Nachricht $I(P(x)) = -\log_2 P(x)$. Nachrichten mit geringer Wahrscheinlichkeit haben also einen hohen Informationsgehalt. Für die Codierung von Nachrichten ist bekannt, dass der Code einer Nachricht x mit der Wahrscheinlichkeit $p(x)$ mindestens die Länge $-\log_2 p(x)$ hat. Für die Codelängen von Modell und Daten unter gegebenem Modell gilt also $L(\theta) = -\log_2 p(\theta)$ und $L(D|\theta) = -\log_2 p(D|\theta)$.

Die Bewertungsfunktion drückt die Ersparnis aus, die durch Verwendung des Modells entsteht. Da beim Rekonstruktionsverfahren geringe Codelängen besser bewertet werden sollen als längere, wird die Summe der Codelängen der Daten und des Modells von der maximalen Codelänge, also der Codelänge der Daten ohne Modell, subtrahiert. Es ergibt sich also $S = \frac{1}{s^2} F - \gamma G$ wobei $F = L(D|\text{ohne Modell}) - L(D|\theta)$ und $G = L(\theta)$ ist. s^2 und γ sind Gewichtungsfaktoren, die später bestimmt werden. Im Folgenden werden die Terme F und G genauer bestimmt. Für die Bewertung der Komplexität G werden zwei Methoden genannt.

Codierungslänge der Daten unter einem gegebenen Modell

Für die Berechnung von F wird die Wahrscheinlichkeit der Daten unter einem Modell $p(D|\theta)$ benötigt. Um diese zu bestimmen, wären sehr große Trainingsdatensätze nötig. Da diese nicht zur Verfügung stehen, wird eine andere Methode verwendet. Diese ähnelt dem Ansatz von Fua und Hanson (1989). Das Bild der Fassade hat $A = n \times k$ Pixel, die jeweils einem Cluster angehören. Es gibt N_C verschiedene Cluster.

Um die Daten ohne Modell zu codieren, werden $L(D|\text{ohne Modell}) = A \cdot \log_2 N_C$ Bits benötigt. In diesem Fall wird für jedes Pixel der Cluster gespeichert. Es gibt N_C verschiedene Cluster, deshalb werden $\log N_C$ Bits pro Pixel benötigt. Ein Modell θ ist durch den Ableitungsbaum gegeben und beschreibt, welches Pixel Wand, Fenster oder eine andere von der Grammatik modellierte Klasse ist. Um die Codelänge zu bestimmen, kann man folgende Überlegungen anstellen:

- Die Pixel unterteilen sich in n_0 Ausreißer und n_1 Pixel, die dem Modell genügen.
- Die Anzahl an Bits, die benötigt werden, um zu kennzeichnen, ob ein Pixel Ausreißer ist, ist durch die Entropie gegeben $E(n_1, n_0) = -(n_1 \log \frac{n_1}{A} + n_0 \log \frac{n_0}{A})$.
- Um die Ausreißer zu modellieren, werden zusätzlich $n_0 \log N_C$ Bits benötigt.
- Für die regulären Pixel werden keine weiteren Bits benötigt. Ein Gauß-Modell für ein Rauschen in den Daten, wie es Fua und Hanson (1989) verwenden, ist nicht sinnvoll für den hier verwendeten Ansatz, da dieser auf Clustern basiert und dadurch ein Pixel eindeutig zum Modell passt oder nicht.

Das ergibt für ein Modell θ für die Länge der Daten $L(D|\theta) = n_0 \log N_C + E(n_1, n_0)$ und damit eine Bewertung von $F = A \cdot \log N_C - (n_0 \log N_C + E(n_1, n_0)) = n_1 \log N_C - E(n_1, n_0)$. Zusätzlich wird noch ein Skalierungskoeffizient s verwendet, um die Größe der Eingabedaten zu kompensieren. Dieser stellt sicher, dass die Bewertung der Daten nicht abhängig von ihrer Größe ist. So bleibt F auch konstant, wenn das Bild vergrößert wird.

Die Zahl der Ausreißer wird anhand des Clusterbildes berechnet. Da Fenster und Türen sich farblich von der restlichen Fassade abheben und auch eine andere Tiefe haben oder teilweise in Fensterbereichen gar keine Punkte gemessen werden, gehören die Bereiche, in denen Fenster und Türen liegen, zu anderen Clustern als die Bereiche, in denen Wand liegt. Zusätzlich hat sich gezeigt, dass sich Bereiche unterschiedlicher Struktur, also z.B. mit einer unterschiedlichen Anordnung der Fenster, oft auch farblich voneinander abgrenzen. Deshalb kann man die Cluster gut als Vergleich für das Modell, das die Fassade ebenfalls in verschiedene Bereiche unterteilt, verwenden. Die Bewertungsfunktion benötigt die Zahl der abweichenden Pixel zwischen dem Modell und dem Clusterbild. Dazu werden die vom Hauptcluster abweichenden Pixel aufsummiert.

Abbildung 6.8 zeigt beispielhaft, wie die Ermittlung von Ausreißern abläuft. Gegeben sind ein Modell und ein Clusterbild einer Fassade. Das Modell definiert Bereiche, die als schwarz umrandete Rechtecke dargestellt sind. Im Beispiel gibt es einen Fensterbereich und einen Wandbereich außerhalb der Fenster. Für beide Bereiche wird der Hauptcluster bestimmt. Dabei handelt es sich um den Cluster, der die größte Fläche innerhalb des Bereichs einnimmt. Anschließend wird die Anzahl der Pixel bestimmt, die nicht zum Hauptcluster gehören. In Abbildung 6.8 sind die abweichenden Pixel des Fensterbereichs hellrot und die des Wandbereichs dunkelrot gekennzeichnet.

In komplexeren Fällen durchläuft die Funktion zur Ermittlung der Ausreißer den Ableitungsbaum. Teilt sich ein Symbol in verschiedene Symbole, so wird die Bewertung für alle Kinder berechnet und aufsummiert. Beispielhaft betrachten wir die Fassade aus Abbildung 6.10d. Der Baum wird vom Startknoten Facade durchlaufen. Dieser teilt sich in die Knoten PartFacade und FacadeRow. In beiden Knoten wird dann die Bewertung des Teilbaums nach obigem Muster unabhängig voneinander gestartet und das Ergebnis später aufsummiert. Besteht ein Teilbaum nur aus einem Symbol oder soll beispielsweise der Baum, der nur aus Facade besteht, bewertet werden, arbeitet die Methode prinzipiell gleich, allerdings wird hier nur der eine Bereich betrachtet. Der Hauptcluster des Bereichs wird bestimmt und anschließend werden die abweichenden Pixel gezählt.

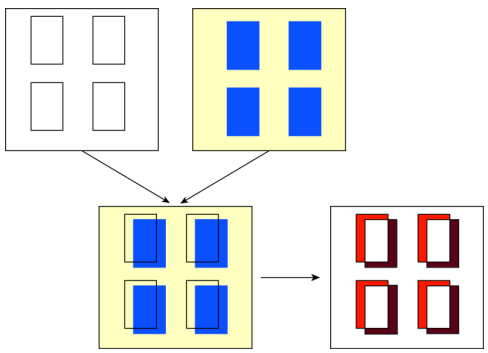


Abbildung 6.8: Skizze zur Ermittlung von Bildpunkten, die nicht zum Modell passen.

Codierungslänge des Modells

Um die Codierungslänge G des Modells θ zu bestimmen, werden zwei Ansätze vorgestellt. Für den ersten wird das Vorwissen über Fassaden verwendet, das auch schon für die Vorschlagswahrscheinlichkeiten der Regeln genutzt wird. Um die Codierungslänge $L(\theta) = -\log P(\theta)$ zu bestimmen, wird die Wahrscheinlichkeit des Modells benötigt. Das Modell θ ist durch den Ableitungsbaum mit zusätzlichen Attributen gegeben. Um die Wahrscheinlichkeit des gesamten Ableitungsbaumes $P(\theta) = p(S_0, \dots S_n)$, wobei die S_i die verwendeten Symbole sind, zu bestimmen, würde wieder eine sehr große Menge an Trainingsdaten benötigt werden. Deshalb wird die Gesamtwahrscheinlichkeit in die Wahrscheinlichkeiten der einzelnen Verzweigungen aufgeteilt, die den Regeln der Grammatik entsprechen. Die Wahrscheinlichkeit des Baumes ergibt sich also als Produkt der bedingten Wahrscheinlichkeiten, wobei ein Symbol immer von seinen Eltern abhängt. Die Wahrscheinlichkeit des Baumes in Abbildung 6.9 ergibt sich zu $p(\theta) = p(S_0, \dots, S_{31}) = p(S_0)p(S_{10}S_{11}|S_0)p(S_{20}|S_{10})p(S_{21}S_{22}|S_{11})p(S_{30}|S_{21})p(S_{31}|S_{22})$, wenn man die Parameter nicht berücksichtigt.

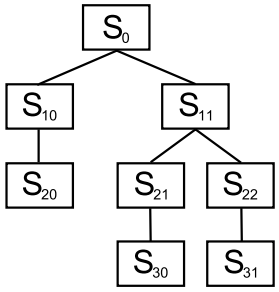


Abbildung 6.9: Beispielmodell zur Illustration der Berechnung der Wahrscheinlichkeit.

Die bedingten Wahrscheinlichkeiten, die für die Berechnung der Wahrscheinlichkeit des gesamten Baumes benötigt werden, entsprechen den Wahrscheinlichkeiten der Grammatikregeln. Die rechte Seite einer Regel ist abhängig von dem Symbol auf der linken Seite. Deshalb können die Regelwahrscheinlichkeiten verwendet werden, die in 6.4.1 bestimmt wurden.

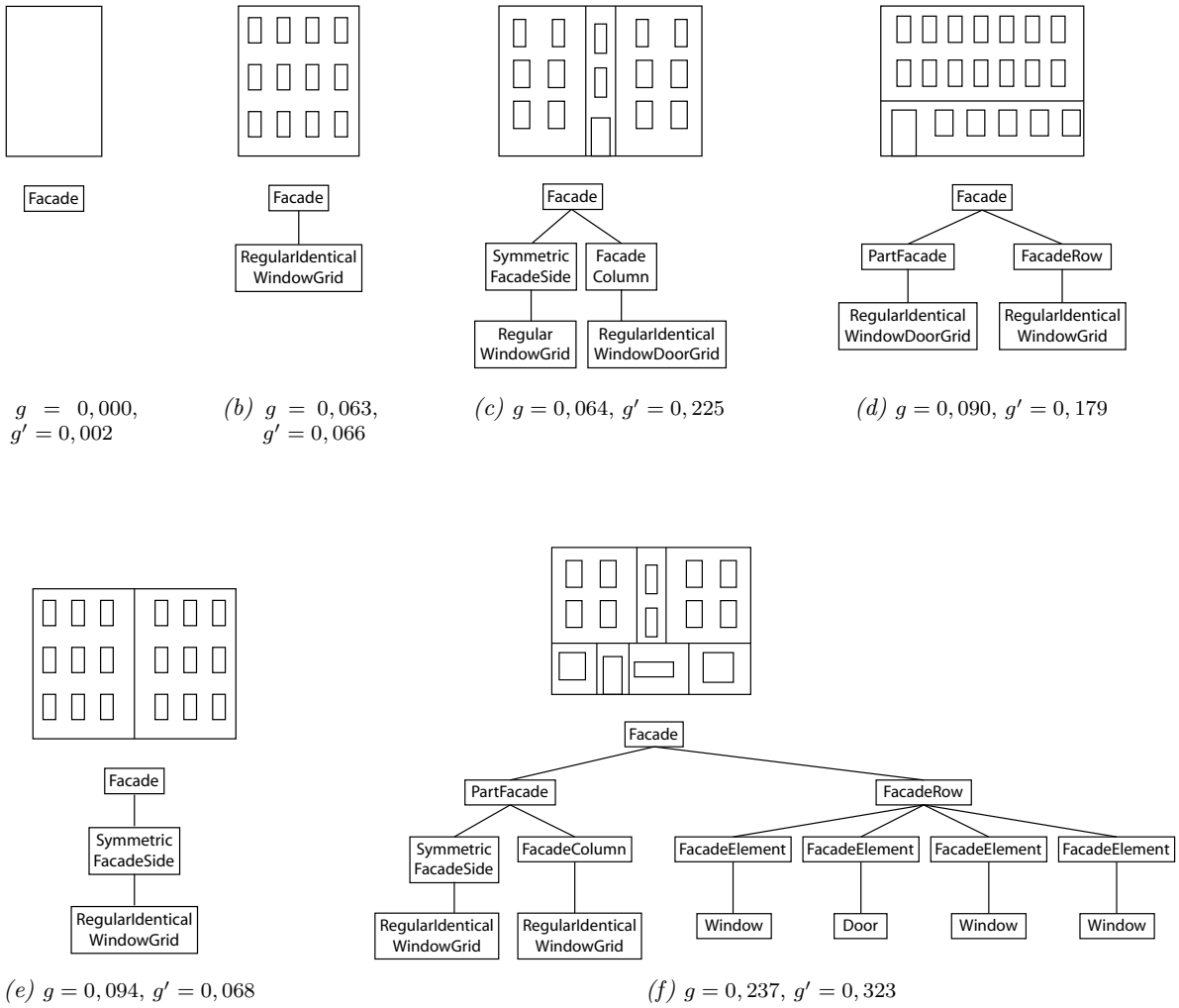
Abbildung 6.10 zeigt einige Fassaden angeordnet nach ihrer Komplexität $g = \gamma \cdot G$. Die Angabe g' bezieht sich auf ein unten beschriebenes weiteres Maß für die Komplexität. Es ist zu erkennen, dass dieses Komplexitätsmaß g Fassaden mit einfachem Baum eine geringe Komplexität zuweist. Ebenso haben Modelle mit stark verzweigtem Baum eine hohe Komplexität. Allerdings würde man intuitiv das Modell (e) als weniger komplex als die Modelle (c) und (d) einstufen.

Theoretisch sollte für die Bestimmung der Modelkomplexität auch die Wahrscheinlichkeit der Parameter in die Bewertung einfließen. Doch in dem Fall wäre es wichtig, Regelwahrscheinlichkeiten zusätzlich anhand der Parameter aufzuteilen. Dazu wäre eine sehr große Menge an Trainingsdaten nötig. Beispielsweise würde die Regel Facade → PartFacade FacadeRow in viele Klassen aufgeteilt werden, die die unterschiedlichen Möglichkeiten der weiteren Modellierung im unteren und oberen Bereich darstellen. Die Regeln hat in der hier verwendeten Version eine Wahrscheinlichkeit von 0,2337. Der Fall, dass oben und unten ein Fenstergitter mit identischen Fenstern und identischen Abständen modelliert ist, ist allerdings sehr unwahrscheinlich. So fein aufgeschlüsselte Wahrscheinlichkeiten können aber aus den vorhandenen Trainingsdaten nicht abgeleitet werden. Deshalb wird noch eine zweite Methode zur Bestimmung der Komplexität entwickelt.

Die zweite Möglichkeit der Bewertung der Komplexität verwendet die Anzahl der Symbole im Ableitungsbaum und deren Parameter als Maß. Dies ist eine Bewertung, wie man sie intuitiv vornehmen würde. Die benötigte Information kann direkt aus dem Parametervektor des Modells gelesen werden. Die Komplexität G' eines Modells ergibt sich dann als $G' = |\text{Nodes}(\theta)| \cdot \log_2 N_{Sym} + \sum_{N \in \text{Nodes}(\theta)} pb_N$, wobei Nodes($\theta$) die Anzahl der Knoten im Baum des Modells θ, N_{Sym} die Anzahl an Grammatiksymbolen und pb_N die Anzahl an Bits, die zur Darstellung aller Parameter des Symbols N benötigt werden, ist. Für jeden Parameter werden 16 Bit zur Speicherung berechnet.

Verwendet man diese Bewertung für die Komplexität eines Modells so verändert sich die Reihenfolge der Modelle in Abbildung 6.10. Die Werte sind dort als $g' = \gamma' \cdot G'$ angegeben. Modell (e) wird an die dritte Stelle vorgezogen und die Modelle (c) und (d) tauschen die Positionen. Dies entspricht auch eher der Reihenfolge, die man als Mensch vergeben würde.

Für das Rekonstruktionsverfahren wird in dieser Arbeit G' zur Bewertung der Modellkomplexität verwendet. Lägen größere Mengen an Trainigsdaten vor, so könnte die Bewertung auf G umgestellt werden.



(a) $g = 0,000$, $g' = 0,002$ *(b)* $g = 0,063$, $g' = 0,066$ *(c)* $g = 0,064$, $g' = 0,225$ *(d)* $g = 0,090$, $g' = 0,179$

(e) $g = 0,094$, $g' = 0,068$ *(f)* $g = 0,237$, $g' = 0,323$

Abbildung 6.10: Verschiedene Fassadenmodelle geordnet nach ihrer Komplexität.

Kombination der Codierungslängen für Modell und Daten

Um auf Basis der Codierungslängen zu einer einzigen Bewertungsfunktion zu gelangen, wird wie bei Fua und Hanson (1989) ein Koeffizient γ verwendet, um die Codierungslängen für das Modell und die Daten unter Verwendung des Modells zu gewichten. γ und der Skalierungsparameter s^2 werden im nächsten Abschnitt bestimmt. Als kombinierte Bewertungsfunktion erhält man schließlich unter Verwendung von G

$$S = \frac{1}{s^2}(n_1 \log N_C - E(n_1, n_0)) + \gamma \log P(\theta)$$

und unter Verwendung von G'

$$S' = \frac{1}{s^2}(n_1 \log N_C - E(n_1, n_0)) - \gamma\Big(|\text{Nodes}(\theta)| \cdot \log_2 N_{Sym} + \sum_{N \in \text{Nodes}(\theta)} pb_N\Big).$$

Diese Bewertungsfunktionen werden anstelle der Wahrscheinlichkeit $p(\theta|D_S D_I)$ in der Akzeptanzwahrscheinlichkeit verwendet.

Bestimmung der Parameter s^2 und γ

Die Parameter s^2 und γ sollen so gewählt sein, dass beide Größen ähnlichen Einfluss auf die Bewertung haben. Die Bewertung der Daten, also die Codierungslänge der Daten mit einem Modell, wird von der Größe der Eingangsdaten beeinflusst, da Pixel gezählt werden. Der Faktor $\frac{1}{s^2}$ ist also auch wichtig, damit die Bewertung skalierter Bilder gleich bleibt.

$F = n_1 \log N_C + n_1 \log \frac{n_1}{A} + n_0 \log \frac{n_0}{A}$ bewertet, welchen Vorteil man durch die Verwendung eines Modells hat. Abbildung 6.11 zeigt die Funktion in Abhängigkeit der Zahl, der zum Modell passenden Pixel n_1, beispielhaft für ein Bild mit 2 Mio. Pixeln und einer Clusteranzahl von drei. Es ist zu sehen, dass bei einer zu großen Zahl an Ausreißern die Funktion auch negativ werden kann. Das Maximum der Funktion von $A \cdot \log N_C$ wird erreicht, wenn alle Pixel zum Modell passen. Da s^2 das Maximum der Funktion auf 1 normieren soll, ergibt sich $s^2 = A \cdot \log N_C$.

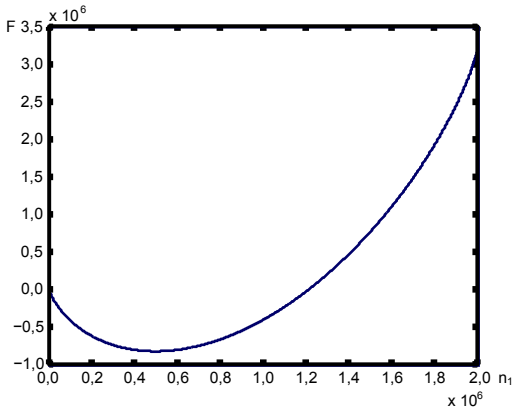
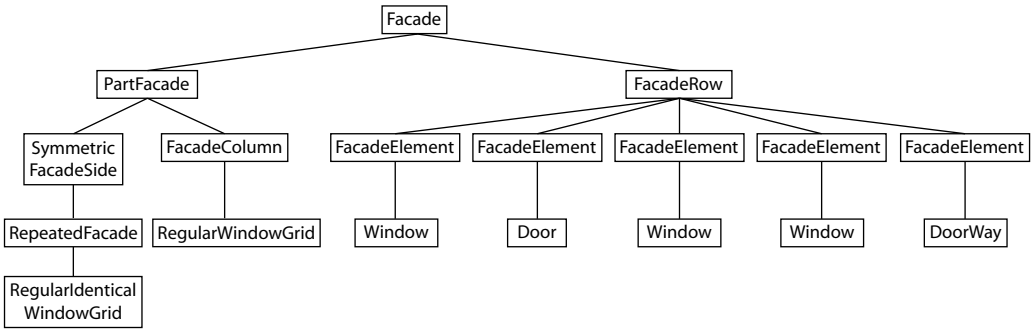
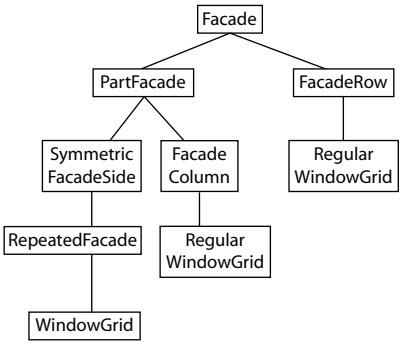


Abbildung 6.11: Bewertungsfunktion F für ein Bild mit 2 Mio. Pixeln und einer Clusterzahl von drei.

Mit γ kann man den Einfluss der Modellkomplexität steuern. Der Parameter regelt also wie unpassend das Modell sein darf, bevor ein komplizierteres Modell bevorzugt wird. Da F auf Werte kleiner als 1 normiert wurde, muss γ so angepasst werden, dass vernünftige Werte angenommen werden. Die Modellkomplexität G wird nach dem Logarithmus der Wahrscheinlichkeit berechnet. Das komplexeste Modell, das mit der Grammatik dargestellt werden kann ist in Abbildung 6.12a dargestellt. Die Zahl der Symbole FACADEELEMENT im unteren Bereich



(a) Bewertung nach G



(b) Bewertung nach G'

Abbildung 6.12: Die komplexesten Modelle, die bezüglich der jeweiligen Bewertungsfunktion mit der Grammatik dargestellt werden können.

kann je nach Breite der Fassade auch größer sein. Dadurch würde die Komplexität steigen. Hier wurde wie in Abschnitt 5.2 von einer Aufteilung in fünf Bereiche ausgegangen. Dieses Modell hat eine Wahrscheinlichkeit von $1,858 \cdot 10^{-13}$. Das ergibt eine Bewertung von -42,2913. Will man für ein sehr komplexes Modell etwa die Hälfte der besten Bewertung abziehen, so ergibt sich $\gamma = 0,012$. Wird die zweite Bewertungsart G' verwendet, so besitzt das komplexeste Modell (Abbildung 6.12b) neun Symbole und 63 Parameter. Dies ergibt eine Bewertung von 1047. In diesem Fall wird $\gamma' = 0,00048$ gewählt.

6.6 Integration von Simulated Annealing

Der rjMCMC Prozess erzeugt Fassadenmodelle als Zustände einer Markov-Kette. Man erhält das beste Modell durch Merken des bisher besten Modells und Vergleichen mit dem aktuellen Modell nach jeder Iteration. Vermutlich gute Fassaden treten im Prozess zwar häufig auf, aber bei diesem Vorgehen werden trotzdem sehr viele Fassadenmodelle mit geringerer Bewertung vorgeschlagen. Der Prozess durchläuft viele Iterationen, ohne gute Modelle vorzuschlagen. Um eine höhere Vorschlagsdichte an guten Modellen zu erreichen, wird das Simulated Annealing (Abschnitt 4.3) in den Prozess integriert.

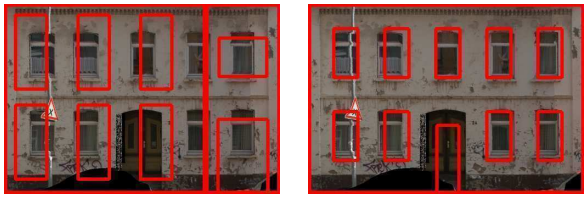
Beim Simulated Annealing wird die Akzeptanzwahrscheinlichkeit in Abhängigkeit von einem Temperaturparameter verändert. Sie sieht dann folgendermaßen aus:

$$\alpha(x,y) = \min\Big\{1, \frac{\pi^{\frac{1}{T_i}}(y)T(y,x)}{\pi^{\frac{1}{T_i}}(x)T(x,y)}\Big\}.$$

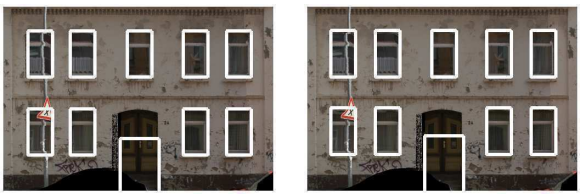
Zu Beginn des Prozesses ist die Temperatur $T_0 = 1$. Für den weiteren Verlauf muss eine Minimaltemperatur und ein Abkühlungsschema definiert werden. Theoretisch geht die Temperatur gegen 0. Da dies praktisch nicht möglich ist, wird hier eine Minimaltemperatur von $T_{Min} = 0,001$ festgelegt.

Schwieriger ist es, ein Abkühlungsschema zu definieren. Nach der Theorie soll die Temperatur sehr langsam abgekühlt werden, damit das globale Maximum erreicht wird. Sicher wird dieses erreicht, wenn die Temperatur sich logarithmisch abkühlt. Ein Abkühlungsschema wäre $T_i = \frac{1}{\log(i)}$, wobei i die aktuelle Iteration angibt. Um mit diesem Schema die Minimaltemperatur von 0,001 zu erreichen, wären 10^{1000} Iterationen nötig. Selbst wenn man das Temperaturschema skalieren würde, wäre die Anzahl an Iterationen zu groß. Deshalb werden hier andere Temperaturschemata verwendet. Es werden ein lineares Abkühlungsschema, eines, das nach der Quadratwurzel abkühlt, und eines, das nach der dritten Wurzel abkühlt, getestet. Die Temperaturen ergeben sich folgendermaßen: $T_{i,linear} = 1-(1-T_{min})\cdot\frac{i}{I}$, $T_{i,sqrt} = 1-(1-T_{min})\cdot\sqrt{\frac{i}{I}}$ und $T_{i,3teWurzel} = 1-(1-T_{min})\cdot\sqrt[3]{\frac{i}{I}}$, wobei I die Gesamtzahl der Iterationen ist. Da die Abkühlung zwischen 0 und 1 beschrieben wird, kann aus den Abkühlungsschemata keine Mindestanzahl an Iterationen bestimmt werden. Dazu werden die Funktionen $\frac{1}{i}$, $\frac{1}{\sqrt{i}}$ und $\frac{1}{\sqrt[3]{i}}$ betrachtet. Um eine Minimaltemperatur von 0,001 zu erreichen, werden für die Schemata 1000, 1 Mio. bzw. 1 Mrd. Iterationen benötigt. Da eine Rekonstruktion mit 1 Mrd. Iterationen eine zu große Laufzeit nach sich zieht, wird für den Test mit einer Abkühlung nach $T_{i,3teWurzel}$ die Minimaltemperatur auf 0,01 gesetzt, so dass dieser Test 1 Mio. Iterationen erfordert.

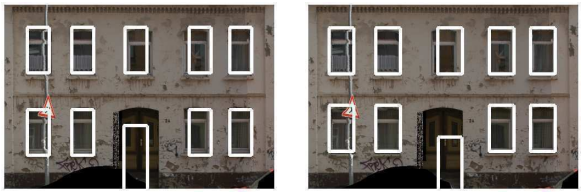
Das Ergebnis der Rekonstruktion mit integriertem Simulated Annealing kann nicht mit dem (unbekannten) theoretisch besten Fassadenmodell verglichen werden. Um das Ergebnis zu überprüfen, wurde während des Tests das beste aller vorgeschlagenen Modelle bestimmt. Dieses Modell dient zum Vergleich mit der Ergebnisrekonstruktion. Abbildung 6.13 zeigt das Ergebnis der Rekonstruktion (links) neben dem besten Modell des Test (rechts) für die verschiedenen Abkühlungsschemata. Für jedes Schema wurden mehrere Tests durchgeführt. Für das lineare Schema gab es einen Test, in dem das Verfahren zu keiner korrekten Lösung konvergierte. Das Ergebnis dieses Tests ist in (a) dargestellt. Weitere Tests mit diesem Schema führten aber auf eine korrekte Lösung. Ein Ergebnis ist in (b) dargestellt. Für die Abkühlung nach $T_{i,sqrt}$ (c) und nach $T_{i,3teWurzel}$ (d) konvergierte das Verfahren stets zu einer korrekten Lösung. Es ist zu sehen, dass die mit Simulated Annealing bestimmten Modelle nicht immer exakt mit dem besten Modell des gesamten Tests übereinstimmen. Sie kommen diesem aber sehr nahe. Für das Rekonstruktionsverfahren wird im Weiteren das Schema $T_{i,sqrt}$ verwendet. Dieses zeigte bei den Tests eine gute Konvergenz und lässt mit weniger Iterationen eine Minimaltemperatur von 0,001 zu.



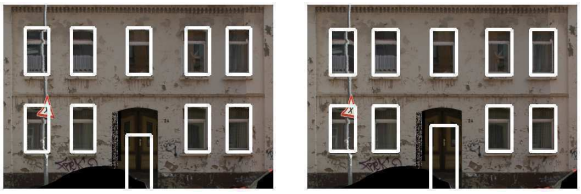
(a) Abkühlung nach $T_{i,linear}$ I



(b) Abkühlung nach $T_{i,linear}$ II



(c) Abkühlung nach $T_{i,sqrt}$



(d) Abkühlung nach $T_{i,3teWurzel}$

Abbildung 6.13: Vergleich der Ergebnisse von Simulated Annealing (jeweils links) und dem besten im Prozess vorkommenden Modell (jeweils rechts) für verschiedene Abkühlungsschemata. Farbgebung: Gitter mit konstantem Spaltenabstand sind rot dargestellt, Gitter mit variablem Spaltenabstand weiß.

7 Ergebnisse der Rekonstruktion

In diesem Kapitel wird das Rekonstruktionsverfahren evaluiert. Zunächst werden manuell segmentierte Fassadenbilder für die Evaluation verwendet. Ergebnisse aus diesem Test wurden verwendet, um die Grammatik sinnvoll zu erweitern. Erste Tests haben gezeigt, dass bestimmte Strukturen nicht modelliert werden konnten. Die Grammatik ist um diese Strukturen erweitert worden und damit auf den in Kapitel 5 beschriebenen Stand gebracht worden. Auf diesen beziehen sich die hier gezeigten Ergebnisse. Im zweiten Abschnitt wird eines der Fassadenbilder synthetisch verändert, um das Verhalten der Rekonstruktion bei steigender Verdeckung zu zeigen. Anschließend wird das Verfahren an Daten getestet, die mit einem terrestrischen Laserscanner aufgenommen wurden.

7.1 Strukturerkennung in eTRIMS-Fassadenbildern

Da die Anzahl an gescannten Fassaden recht gering ist, kann man mit ihnen das Rekonstruktionsverfahren und insbesondere die Allgemeinheit der Grammatik nicht hinreichend testen. Um trotzdem einen Test mit einer größeren Anzahl an Fassaden durchführen zu können, werden segmentierte Fassadenbilder aus einer Datenbank als Eingabe verwendet (Ripperda und Brenner, 2009b). Es handelt sich hierbei um die eTRIMS-Bilddatenbank (Korč und Förstner, 2009), die von dem EU-Projekt eTRIMS (e-Training for Interpretation of Manmade Scenes) zur Verfügung gestellt wird. Die Datenbank enthält Fassadenbilder und objekt- und klassenbasierte Segmentierungen. Die klassenbasierte Segmentierung steht für vier und acht Klassen zur Verfügung. Die Vier-Klassen-Segmentierung beinhaltet die Klassen Gebäude, Gehweg/Straße, Himmel und Vegetation. Bei der Acht-Klassen-Segmentierung wird in die Klassen Gebäude, Auto, Tür, Gehweg, Straße, Himmel, Vegetation und Fenster unterschieden. Zur Rekonstruktion ist die Acht-Klassen-Segmentierung geeignet, da sie Informationen über Fenster und Türen enthält. In der Datenbank befinden sich zur Zeit 60 annotierte Bilder. In der Zukunft soll sie noch erweitert werden. Von den 60 Bildern der Datanbank werden 56 verwendet. Vier Bilder werden weggelassen, da sie Gebäude wie z.B. Einfamilienhäuser zeigen, die sich nicht mit der Grammatik modellieren lassen.

Da das Rekonstruktionsverfahren für Entfernungsbilder kombiniert mit Orthofotos entwickelt wurde, müssen die Bilder aus der Datenbank vorverarbeitet werden. Die Bilder werden dazu manuell entzerrt und auf den Fassadenbereich zugeschnitten. Anschließend wird der Maßstab über die Anzahl an Stockwerken geschätzt. Hierbei wird eine Stockwerkshöhe von 3,30 m angenommen. Das Verfahren bekommt keine Informationen über den Zusammenhang zwischen Farben und Klassen. Das eTRIMS-Bild dient als optimale Segmentierung und kann anstelle der Clusterbilder eingesetzt werden.

Nach der Vorverarbeitung wird die Rekonstruktion mit jedem Bild durchgeführt. Abbildung 7.1 zeigt einige Rekonstruktionsergebnisse. Die Farbe, in der die Fenster dargestellt sind, gibt an, welche Gitterstruktur zugrunde liegt. Rote Fenster deuten auf äquidistante Abstände hin, in Gittern mit weißen Fenstern können die Abstände variieren. In der ersten Fassade werden Tür und Fenster richtig erkannt, nur der Größenunterschied der Fenster in der mittleren Reihe wird nicht korrekt modelliert. Das zweite Bild zeigt eine Rekonstruktion eines RegularIdenticalWindowDoorGrid, bei dem die Fassadenelemente relativ gut modelliert sind. In der dritten Fassade ist eine Unterteilung in Erdgeschoss und obere Stockwerke modelliert. Der obere Teil der Fassade wird als symmetrisch mit einem Fenstergitter modelliert. Im unteren Teil ist ein Fenstergitter mit einer Tür vorgeschlagen. Im Bild ist nur ein Fenster zu sehen, dieses hat allerdings eine deutlich größere Breite als in den Testdaten vorkommt. Deshalb ist der Breite die Wahrscheinlichkeit 0 zugeordnet und dieses eine breite Fenster wird durch drei dicht beieinander liegende Fenster modelliert. Die vierte Fassade zeigt ebenfalls eine horizontale Unterteilung. Im oberen Bereich ist das 2×3 Gitter exakt modelliert. Im unteren Bereich sind zwei Fenster und eine Tür modelliert. Die zweite Tür wurde nicht erkannt, da sie zum Großteil von einem Auto verdeckt ist. Die erste Fassade in der unteren Zeile ist durch ein Fenstergitter mit Tür modelliert. Die Elemente sind gut platziert. Die letzte Fassade lässt sich durch ein Gitter aus Fenstern modellieren. Diese sind trotz der teilweisen Verdeckung durch Bäume gut rekonstruiert.

In manchen Fällen wird die Struktur der Fassade nicht gut erkannt, insbesondere, wenn sie mit der verwendeten Grammatik nicht darstellbar ist. Abbildung 7.2 zeigt zwei solche Fälle, es ist jeweils das Fassadenfoto und die Rekonstruktion gezeigt. Die Ursachen dafür, dass die Struktur nicht in der Grammatik enthalten ist, sind hier

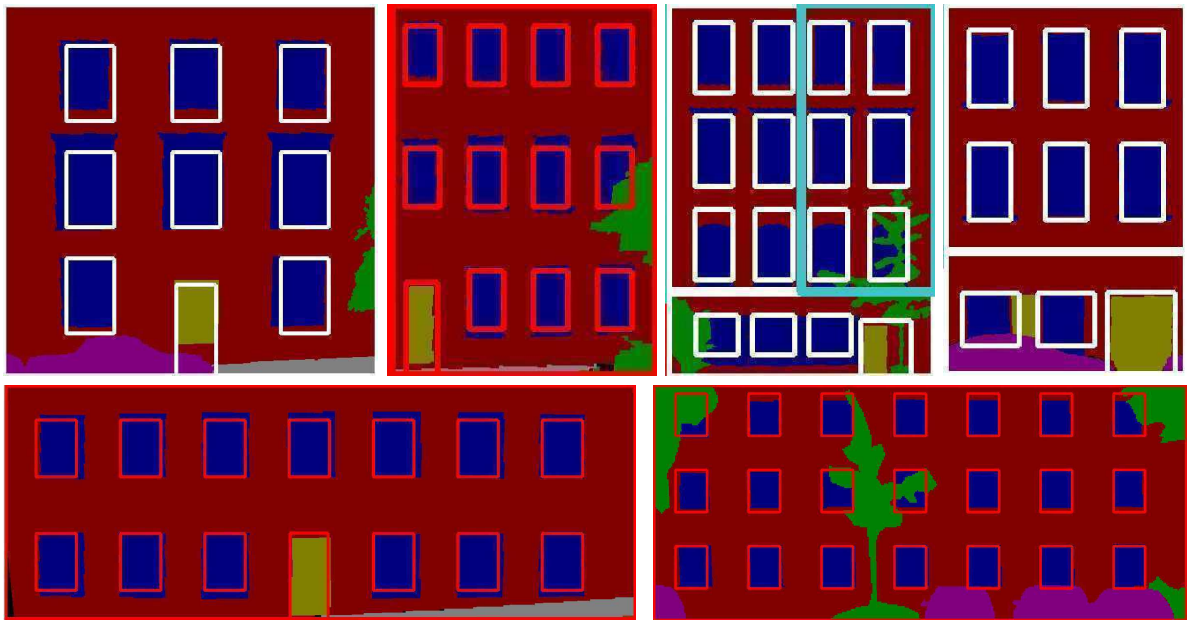


Abbildung 7.1: Ergebnisse der Strukturerkennung. Farbgebung: Gitter mit konstantem Spaltenabstand sind rot dargestellt, Gitter mit variablem Spaltenabstand weiß.

unterschiedlich. Bei der linken Fassade ist in der Segmentierung im Erdgeschoss nur eine Tür zu sehen. Dieser Fall wird von der Grammatik nicht abgedeckt, da bei den Untersuchungen von Fassadenbildern nie ausschließlich eine Tür im Erdgeschoss vorkam. Diese Struktur sollte der Grammatik auch nicht hinzugefügt werden, denn im Foto zur Segmentierung ist zu sehen, dass diese Struktur nicht der Wirklichkeit entspricht. Es befinden sich zusätzlich Schaufenster und Garagen im Erdgeschoss. Diese Elemente sind nicht in den acht Klassen der eTRIMS-Datenbank enthalten, deshalb wurden die entsprechenden Gebiete als Wand modelliert. Da die so entstandene Situation in der Grammatik nicht modelliert werden kann, erfindet das Rekonstruktionsverfahren eine Reihe von Fenstern, die zur Struktur der oberen Fenster passt.

Das zweite Beispiel zeigt eine Fassade mit einem Fenstergitter, in dem in der oberen Reihe das mittlere Fenster fehlt. Diese Segmentierung entspricht der Realität, kann aber ebenfalls nicht modelliert werden. Auch in diesem Fall wird ein zusätzliches Fenster erfunden, das sich der Struktur der anderen Fenster anpasst.

Auch wenn die Strukturen, die die Grammatik beschreibt, nicht in allen Fällen vorhanden sind, so zeigt sich doch häufig, wie hilfreich diese Annahmen von Regelmäßigkeiten sind. Besonders wenn Strukturen z.B. durch Vegetation verdeckt sind, zeigt sich der Vorteil dieser Vorgehensweise. Das Wissen über die Fassadenstrukturen erlaubt dann eine Vorhersage über die Struktur, die sich hinter der Vegetation verbirgt. Abbildung 7.1 zeigt unten rechts eine Fassade, bei der einige Fenster zumindest teilweise durch Bäume verdeckt sind. Das Wissen über die gitterförmige Anordnung von Fenstern hilft hier, die verdeckten Fenster korrekt zu modellieren. Abschnitt 7.2 untersucht, wie stark sie Verdeckung sein darf, ohne sich negativ auf das Ergebnis auszuwirken.

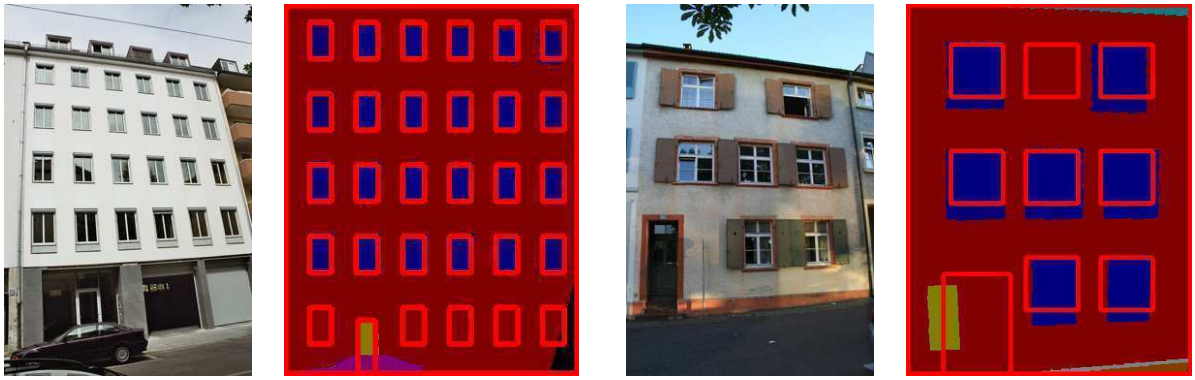


Abbildung 7.2: Beispielfassaden (jeweils Originalbild und Segmentierung) mit fehlerhafter Rekonstruktion. Farbgebung wie in Abbildung 7.1.

Bewertung der Ergebnisse

Zur Bewertung der Ergebnisse wurden diese zunächst visuell in Gruppen aufgeteilt. Diese Gruppen sind die korrekt rekonstruierten Fassaden (A), Rekonstruktionen, die Fehler in der Struktur haben, bei denen die Elemente aber gut platziert sind (B), Fassaden, die in Teilen gut modelliert sind (C), Fassaden mit korrekt rekonstruierter Struktur, die jedoch Fehler in der Position der Elemente haben (D) und schließlich die fehlerhaft rekonstruierten Fassaden (E). Um einen Eindruck der Klassen zu geben, zeigt Abbildung 7.3 je eine Fassade aus jeder Klasse. Die rekonstruierten Fassaden teilen sich folgendermaßen in die Gruppen auf 33,9% A, 25,0% B, 14,3% C, 5,4% D und 21.4% E.

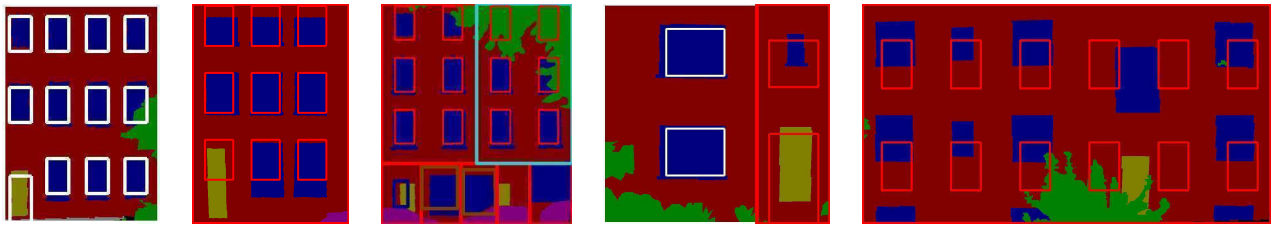


Abbildung 7.3: Beispielfassaden aus jeder der fünf Gruppen. Farbgebung wie in Abbildung 7.1.

Um die Ergebnisse genauer auszuwerten, werden die Rekonstruktionsergebnisse jetzt bildpunktweise untersucht. Hierzu wird aus der erhaltenen Struktur eine Segmentierung synthetisiert, welche bildpunktweise mit der originalen Segmentierung verglichen wird. Abbildung 7.4 (a) zeigt die klassenbasierte Segmentierung einer Fassade und (b) das synthetisierte Bild der zugehörigen Rekonstruktion.

Tabelle 7.1 (links) zeigt das Ergebnis des bildpunktweisen Vergleichs des originalen Bildes mit dem synthetisierten Bild aus Abbildung 7.4. Zu jeder Klassenkombination ist die Häufigkeit angegeben. In jeder Zeile steht eine Klasse der eTRIMS-Segmentierung bzw. die Klasse „nicht annotiert“. In den Spalten stehen die modellierten Klassen Wand, Fenster und Tür. Die anderen Klassen der eTRIMS-Segmentierung werden in der Grammatik nicht modelliert und weitere Symbole der Grammatik treten im Bild nicht auf. Um Pixel der nicht modellierten Klassen wie Vegetation nicht grundsätzlich falsch zu bewerten, wird der Begriff der neutralen Zuordnung genutzt. Es können alle Klassen von Vegetation verdeckt sein. Diese Zuordnungen werden als neutral bewertet. Für Straßen gilt, dass sie durch fehlerhafte Entzerrung des Bildes im unteren Bereich der Fassade vorkommen können. Deshalb sind Modellierungen als Wand oder Tür neutral. Fenster hingegen liegen höher in der Fassade und eine Zuordnung ist deshalb falsch. Die vollständige Einordnung der Kombinationen ist in Tabelle 7.1 (rechts) gegeben.

Tabelle 7.2 zeigt die Korrektheit $k_A = \frac{n_{A\to A}}{n_{M_A}}$ und die Vollständigkeit $v_A = \frac{n_{A\to A}}{n_{B_A}}$, $A \in \{W, F, T\}$ der drei vorkommenden Klassen in Prozent, wobei $n_{A\to A}$ die Anzahl an Pixeln ist, die in Bild und Modell der Klasse A angehören, n_{M_A} die Anzahl der Pixel der Klasse A im Modell und n_{B_A} die Anzahl der Pixel der Klasse A im Bild ist. Betrachtet man positive und neutrale Zuordnungen, so erhält man eine Korrektheit von 94,8% für Wand-Pixel, von 86,0% für Fenster-Pixel und von 70,2% für Tür-Pixel. Die Vollständigkeit liegt bei 93,2% für Wand, 86,1% für Fenster und bei 87,3% für Türen.

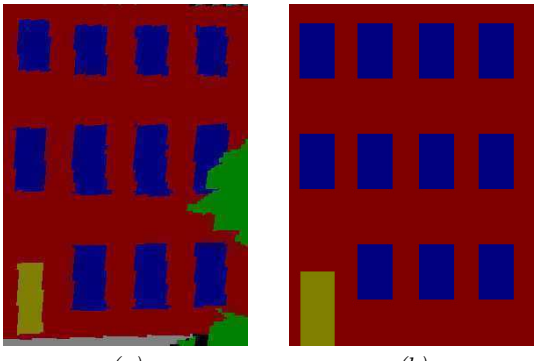


Abbildung 7.4: Beispielfassade (a) und aus der Rekonstruktion synthetisiertes Bild (b).

	Wand	Fenster	Tür
Wand	66524	3882	990
Fenster	3727	23121	0
Tür	291	0	2008
Vegetation	5197	717	0
Auto	0	0	0
Himmel	53	0	0
Gehweg	1689	0	317
Straße	0	0	0
nicht annotiert	294	0	0

	Wand	Fenster	Tür
Wand	c	f	f
Fenster	f	c	f
Tür	f	f	c
Vegetation	n	n	n
Auto	n	n	n
Himmel	n	f	f
Gehweg	n	f	n
Straße	n	f	n
nicht annotiert	n	n	n

Tabelle 7.1: Rekonstruktionsergebnis der Fassade in Abbildung 7.4 (links) und Definition korrekter, neutraler und falscher Zuordnungen (rechts).

		Wand	Fenster	Tür
Korrektheit	korrekt	0.855	0.834	0.606
	neutral	0.093	0.026	0.096
Vollständigkeit		0.932	0.861	0.873

Tabelle 7.2: Korrektheit und Vollständigkeit der Rekonstruktion der Fassade aus Abbildung 7.4.

Abbildung 7.5 zeigt die Korrektheit $k = \frac{\sum_{A\in\{W,F,T\}} n_{A\to A}}{\sum_{A\in\{W,F,T\}} n_{M_A}}$ und Vollständigkeit $v = \frac{\sum_{A\in\{W,F,T\}} n_{A\to A}}{\sum_{A\in\{W,F,T\}} n_{B_A}}$ pro Fassade für alle Fassaden der Datenbank. Die neutralen Zuordnungen sind extra aufgeführt. Man sieht, dass ohne sie die Korrektheit teilweise sehr gering ist. Das ist in Szenen, in denen große Bereiche der Fassade durch Vegetation verdeckt sind, der Fall. Die Fassaden sind in die oben genannten fünf Gruppen unterteilt. In Gruppe A liegt die Korrektheit zwischen 94.1% und 84.6% und die Vollständigkeit zwischen 93.9% und 81.8%. Gruppe B zeigt ebenfalls gute Ergebnisse mit einer Korrektheit zwischen 92.7% und 79.1% und einer Vollständigkeit zwischen 91.1% und 78.3%. In Gruppe C hängen diese Werte stark von der Größe der korrekt modellierten Teile ab. Deshalb variiert die Korrektheit zwischen 87.6% und 71.8% und die Vollständigkeit zwischen 86.6% und 65.6%. Die Rekonstruktionen der Gruppe D zeigen eine Korrektheit zwischen 87.7% und 82.9% und eine Vollständigkeit zwischen 86.6% und 75.6%. Bei den fehlerhaft rekonstruierten Fassaden in Gruppe E schwanken die Werte für die Korrektheit zwischen 87.8% und 57.3% und für die Vollständigkeit zwischen 85.8% and 55.0%. Diese Unterschiede liegen an der Zahl der Vegetations-Pixel. Sind große Teile der Fassade durch Vegetation verdeckt, so treten auch mehr Fehler in der Rekonstruktion auf.

Ebenso kann man auch untersuchen, wie viele Fassadenelemente korrekt erkannt wurden. Um zu bestimmen, ob ein Objekt korrekt erkannt wird, werden die Eingangsdaten mit der Rekonstruktion verglichen. Dabei wird

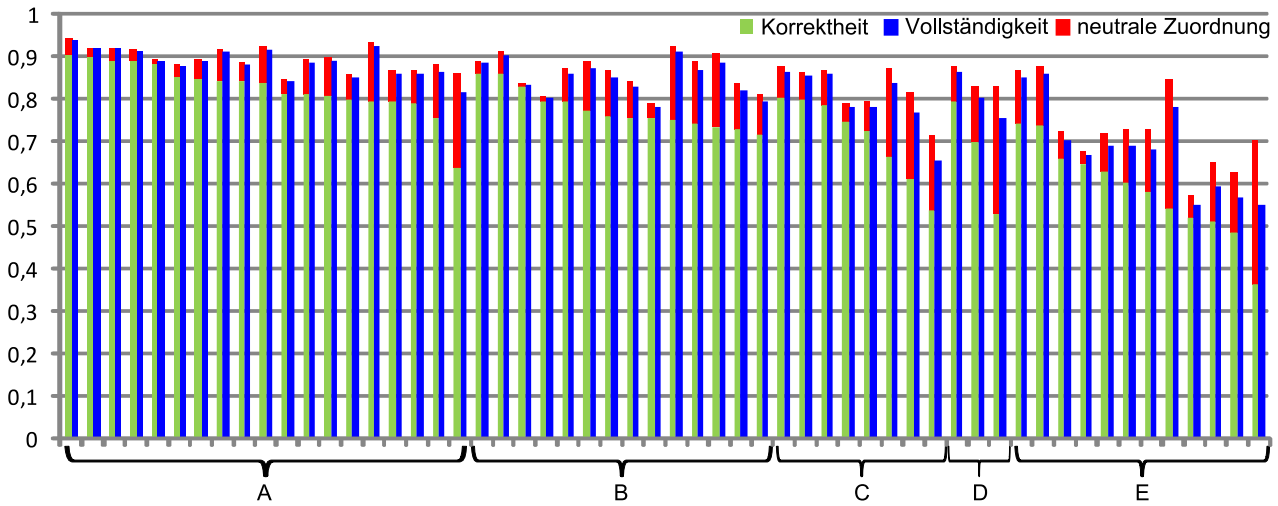
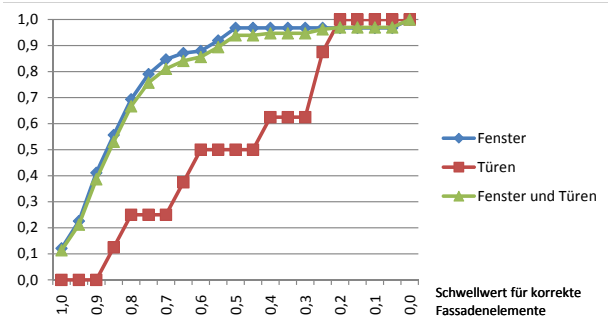
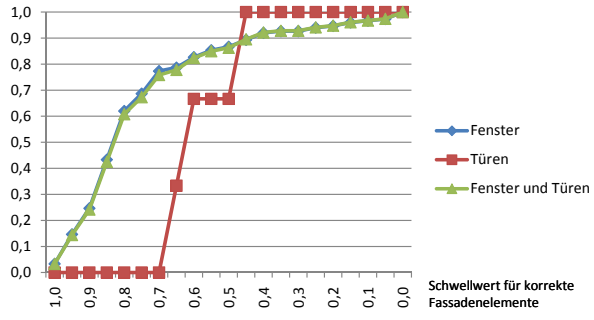


Abbildung 7.5: Korrektheit und Vollständigkeit der Rekonstruktion gruppiert nach der Qualität der Ergebnisses. A: Korrekt rekonstruiert, B: Strukturfehler aber gut positionierte Elemente, C: In Teilen korrekt rekonstruiert, D: Struktur korrekt rekonstruiert aber Fehler in der Position, E: Fehlerhafte Rekonstruktion.

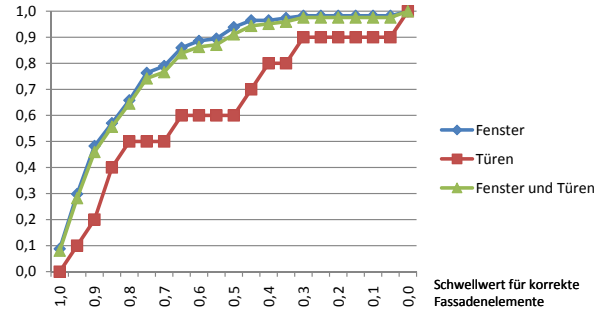
ein Fassadenelement als korrekt erkannt gewertet, wenn die Anzahl der Bildpunkte der Klasse des Objekts im Eingangsbild innerhalb des Objektbereichs in der Rekonstruktion $x\%$ beträgt. x ist eine Variable, die im Test in 5%-Schritten von 100% bis 0% läuft. Abbildung 7.6 zeigt den Anteil korrekt erkannter Elemente bei sinkendem x. Die Diagramme sind nach den oben definierten Gruppen A bis E der Rekonstruktionsergebnisse aufgeteilt. Jedes Diagramm zeigt den Anteil korrekter Fenster in blau, korrekter Türen in rot und die Summe beider in grün.



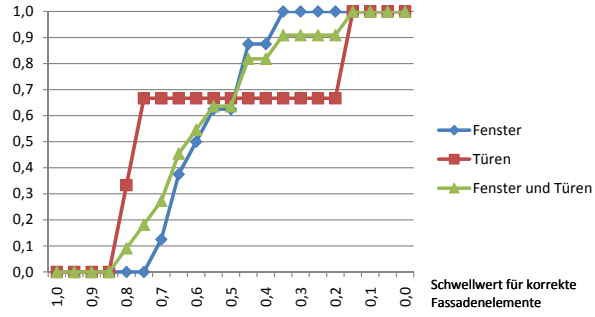
(a) Gruppe A



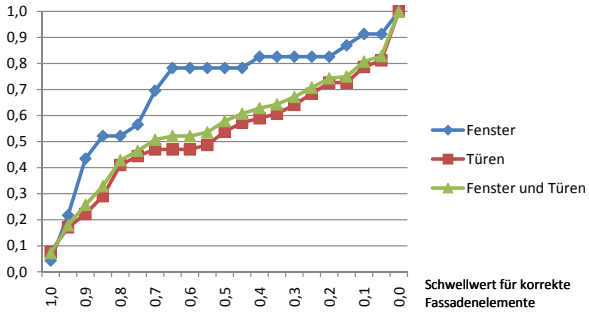
(b) Gruppe B



(c) Gruppe C



(d) Gruppe D



(e) Gruppe E

Abbildung 7.6: Anzahl korrekt erkannter Fassadenelemente in den Gruppen A bis E der Rekonstruktionen.

In den Diagrammen ist zu sehen, dass Fenster deutlich besser rekonstruiert werden als Türen. Dies ist damit zu erklären, dass im unteren Bereich der Bilder häufig Störungen durch Autos oder Vegetation vorkommen. Durch diese wird die Position der Tür nicht korrekt erkannt oder es wird eine Tür anstelle eines Fensters modelliert.

Die Diagramme (a) bis (c) gehören zu den Gruppen A bis C, also den Rekonstruktionen, die korrekt sind, Fehler in der Struktur aufweisen aber korrekt die Position der Elemente rekonstruieren oder in Teilen korrekt rekonstruiert sind. Für diese Rekonstruktionen steigt die Kurve für den Anteil korrekt modellierter Fenster zu Beginn steil an. Die Diagramme (d) und (e) stellen den Verlauf für die Gruppe der Bilder dar, die strukturell korrekt rekonstruiert wurden, aber Fehler in den Positionen haben, oder die fehlerhaft rekonstruiert wurden. Hier ist ein deutlich flacherer Anstieg der Kurve zu erkennen.

7.2 Analyse der Rekonstruktion bei teilweise verdeckten Objekten

Ein großer Vorteil der auf Struktur basierenden Rekonstruktion ist, dass auch teilweise verdeckte Objekte rekonstruiert werden können. Dies ist wichtig, da Vegetation häufig Teile der Fassade verdeckt und es nur mit großem Aufwand möglich oder teilweise auch unmöglich ist, die Fassade vollständig zu erfassen. Das in der Grammatik formulierte Vorwissen über Fassaden erlaubt es, an die verdeckten Stellen der Fassade Fassadenelemente so zu platzieren, wie es unter der Vorbedingung der sichtbaren Fassadenelemente am wahrscheinlichsten ist.

In diesem Abschnitt wird getestet, wie groß der verdeckte Bereich sein darf, um dennoch eine korrekte Rekonstruktion zu erhalten. Dazu wird ein Bild aus der eTRIMS-Datenbank verwendet, in dem Bäume Teile der Fassade verdecken. Es sind in diesem Bild nur Teile der Fenster verdeckt, bei der Rekonstruktion wird jedes Fenster korrekt modelliert. Die Fenster der Fassade sind ca. 35 Pixel × 50 Pixel groß, wobei hier ein Pixel etwa 3,4 cm entspricht. Für den Test wird auf den Baum in der Mitte des Bildes schrittweise eine Dilatation um jeweils fünf Pixel angewendet, wobei die maximale Ausdehnung 75 Pixel beträgt. Abbildung 7.7 zeigt die Bilder mit sich ausdehnendem Baum.

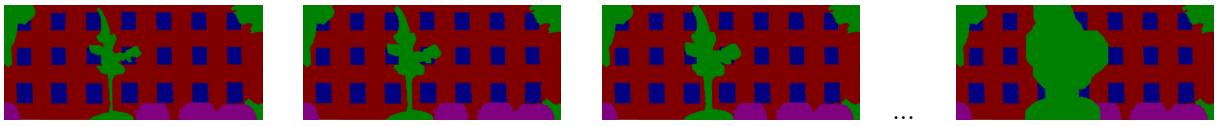
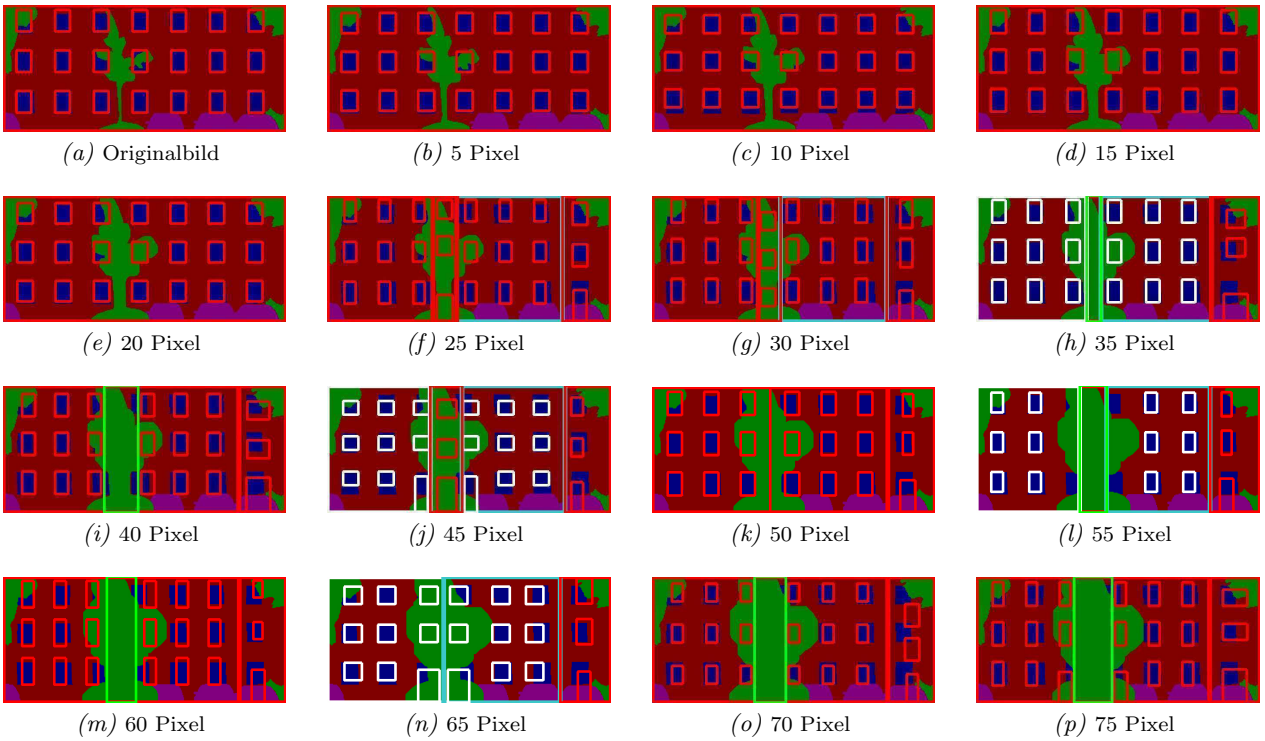


Abbildung 7.7: Testbilder zur Rekonstruktion bei wachsender Verdeckung durch Vegetation. Der Baum in der Mitte wird durch Dilatation schrittweise vergrößert.



(a) Originalbild *(b)* 5 Pixel *(c)* 10 Pixel *(d)* 15 Pixel

(e) 20 Pixel *(f)* 25 Pixel *(g)* 30 Pixel *(h)* 35 Pixel

(i) 40 Pixel *(j)* 45 Pixel *(k)* 50 Pixel *(l)* 55 Pixel

(m) 60 Pixel *(n)* 65 Pixel *(o)* 70 Pixel *(p)* 75 Pixel

Abbildung 7.8: Rekonstruktionen aus dem Originalbild und generierten Testbildern mit wachsender Verdeckung durch Vegetation. Farbgebung wie in Abbildung 7.1.

Abbildung 7.8 zeigt die Rekonstruktionsergebnisse zu den unterschiedlichen Verdeckungsgraden. Bis zu dem Bild mit einer Dilatation um 20 Pixel (e) werden alle Fenster und die Fassadenstruktur korrekt rekonstruiert. Ab einer Ausdehnung von 25 Pixeln (f) wird eine (komplexere) symmetrische Rekonstruktion bevorzugt. Da der Baum nicht mittig in der Szene steht, wird die rechte Spalte getrennt modelliert. Der symmetrische Teil wird teilweise mit Mittelteil modelliert, da dann der Bereich des Baumes nicht als fehlerhaft gewertet wird. Im Fall von 25 (f), 30 (g) und 45 (j) Pixeln Dilatation werden auch innerhalb des Mittelteils Fenster und Türen modelliert, die nicht im Bild vorkommen. Ein Großteil der Fenster wird trotz der überflüssigen Modellierung der Struktur korrekt modelliert. In der rechten Spalte wird eine Tür anstelle eines Fensters modelliert. Der

Grund hierfür sind die Stördaten eines Autos, welches dort zu sehen ist. Ein ähnlicher Effekt tritt in den Rekonstruktionen (j) und (n) auf. Hier wird durch die unteren Pixel des Baumes eine Tür anstelle des Fensters gesetzt. Die Pixel des Baumes gelten sowohl im Bereich der Wand als auch der Tür als falsch.

Abbildung 7.9 zeigt die Korrektheit und die Vollständigkeit der Rekonstruktionen bei unterschiedlichen Verdeckungsgraden. Für die Korrektheit sind zusätzlich die neutralen Zuordnungen angegeben. Die Rekonstruktion des Originalbildes hat eine Korrektheit von 0,779 und eine Vollständigkeit von 0,948. Mit neutralen Zuordnungen ergibt sich eine Korrektheit von 0,957. Sowohl die Korrektheit als auch die Vollständigkeit sinken mit steigender Verdeckung, aber selbst das Bild mit einer Dilatation von 75 Pixeln (p) lässt sich noch mit einer Korrektheit von 0,903 und einer Vollständigkeit von 0,849 rekonstruieren. Da bei steigender Verdeckung der Anteil der zum Baum gehörenden Pixel wächst, steigt auch die Anzahl neutraler Zuordnungen. Dies ist deutlich im Diagramm der Korrektheit zu sehen.

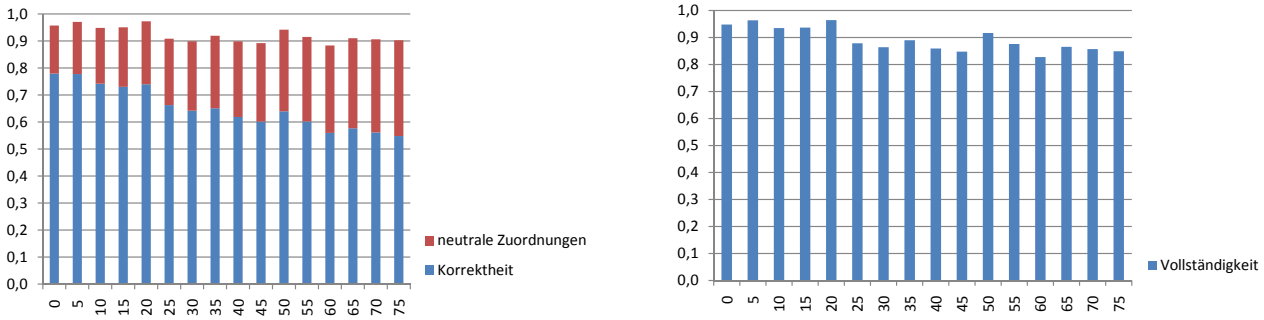


Abbildung 7.9: Korrektheit und Vollständigkeit der Rekonstruktion bei unterschiedlichen Verdeckungsgraden.

7.3 Rekonstruktion aus Scan- und Bilddaten

Für die Rekonstruktion wurden verschiedene Datensätze von Fassaden von Wohnhäusern in Hannover verwendet. Die Daten wurden mit einem terrestrischen Laserscanner, einem Riegl LMS Z360, und einer aufgesetzten Kamera erfasst. Die Vorverarbeitung der Daten wurde wie in Abschnitt 6.2 beschrieben durchgeführt, so dass ein Tiefenbild sowie ein Orthophoto der Fassade zur Verfügung stehen. In einem Datensatz sind nur Tiefeninformationen enthalten. Für diesen wird die Rekonstruktion ausschließlich auf Tiefeninformationen durchgeführt.

Die Daten stehen in einer Auflösung von 1 cm pro Pixel zur Verfügung. Um die Laufzeit zu verkürzen wurden die Daten für diesen Test nicht mit voller Auflösung verwendet, sondern auf die halbe Breite und Höhe skaliert. Der Test wurde mit Simulated Annealing und einer Zahl von 5 Mio. Iterationen durchgeführt. Bei dieser Konfiguration ergibt sich eine durchschnittliche Laufzeit von vier Stunden auf einem Intel Core2 2,66GHz.

Für jeden Test wird im Folgenden ein Bild gezeigt. Dies ist im Allgemeinen das Orthophoto. Der Datensatz Alemannstraße enthält nur Tiefeninformation, deshalb wird dort das Tiefenbild angegeben. In diesen Bildern sind die rekonstruierten Strukturen und Fassadenelemente eingezeichnet. Gitter mit konstantem Spaltenabstand sind dabei rot gezeichnet und Gitter mit variablem Spaltenabstand weiß. Neben den Bildern ist zusätzlich ein 3D-

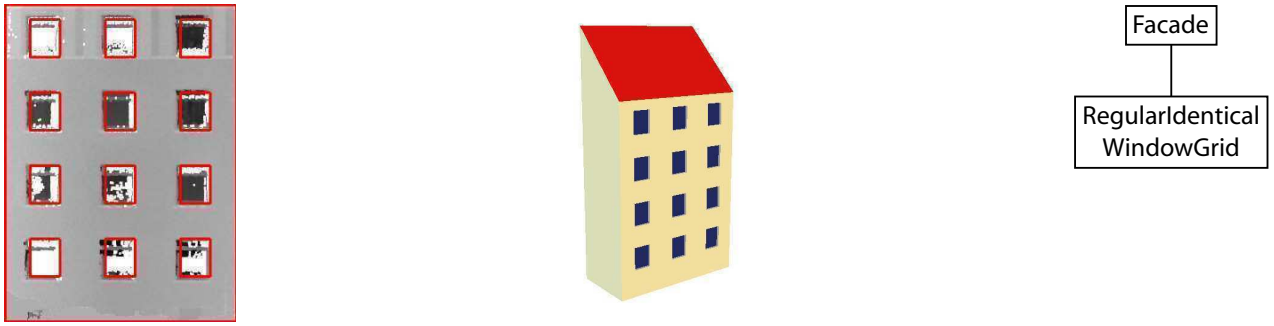


Abbildung 7.10: Rekonstruktion der Fassade Alemannstraße.

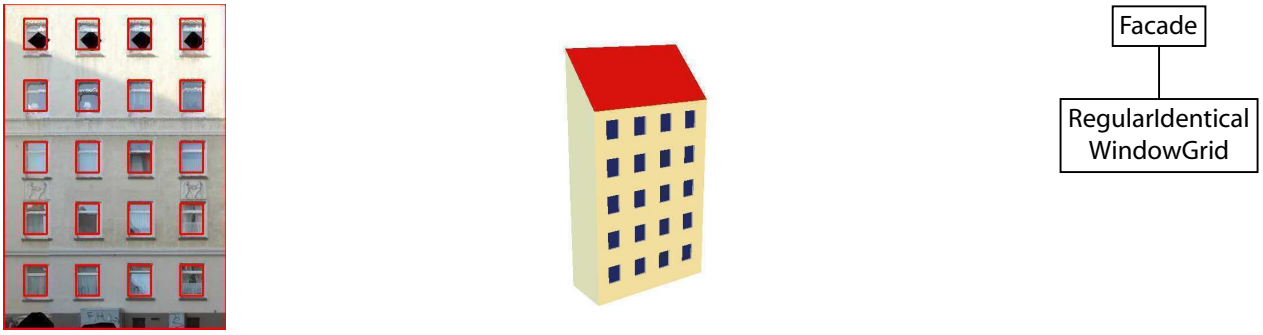


Abbildung 7.11: Rekonstruktion der Fassade Voltastraße.

Abbildung 7.12: Rekonstruktion der Fassade Schneiderberg I.

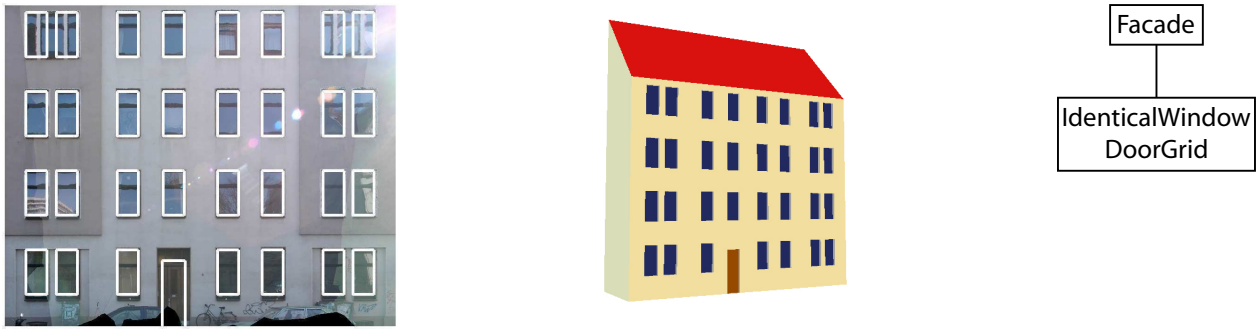


Abbildung 7.13: Rekonstruktion der Fassade Schneiderberg II.

Modell des Gebäudes zu sehen. Hierbei ist die Fassade aus den Daten rekonstruiert, die seitlichen Wände und das Dach sind hinzugefügt, um einen besseren optischen Eindruck zu bekommen. Zusätzlich ist der Ableitungsbaum der Grammatik dargestellt.

In den Ergebnissen zeigt sich, dass viele Symbole der Grammatik genutzt werden. Einfache Symbole, wie das RegularIdenticalWindowGrid, werden für die Datensätze Alemannstraße (Abbildung 7.10) und Voltastraße (Abbildung 7.11) verwendet. Hier sind alle Fenster der Fassade identisch und regelmäßig angeordnet. Im Datensatz Schneiderberg I und II (Abbildung 7.12 und 7.13) haben die Spalten keinen regelmäßigen Abstand, deshalb wird für diese Fassaden ein IdenticalWindowDoorGrid verwendet. Die Tür wurde in beiden Fällen an der richtigen Position modelliert. Allerdings ist sie für die Fassade Schneiderberg I deutlich zu niedrig. Im Datensatz Appelstraße (Abbildung 7.14) sind die Fenster regelmäßig angeordnet. Zwei Fenster im Erdgeschoss haben allerdings eine andere Größe als die übrigen Fenster. Diese Fassade wurde mit dem Symbol RegularWindowGrid modelliert. In dem Gitter sind zwei verschiedene Fensterarten enthalten. Die zwei vom Rest abweichenden Fenster werden durch einen Fenstertyp repräsentiert. Daraus ergibt sich, dass im Vergleich zum

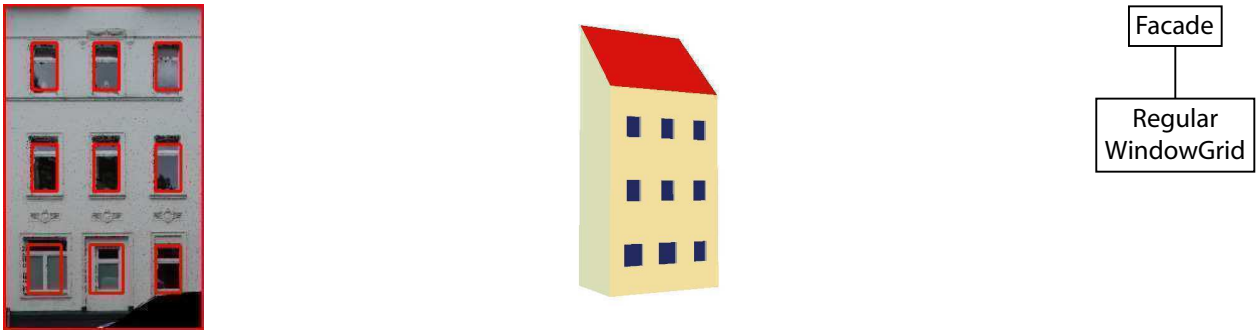


Abbildung 7.14: Rekonstruktion der Fassade Appelstraße.

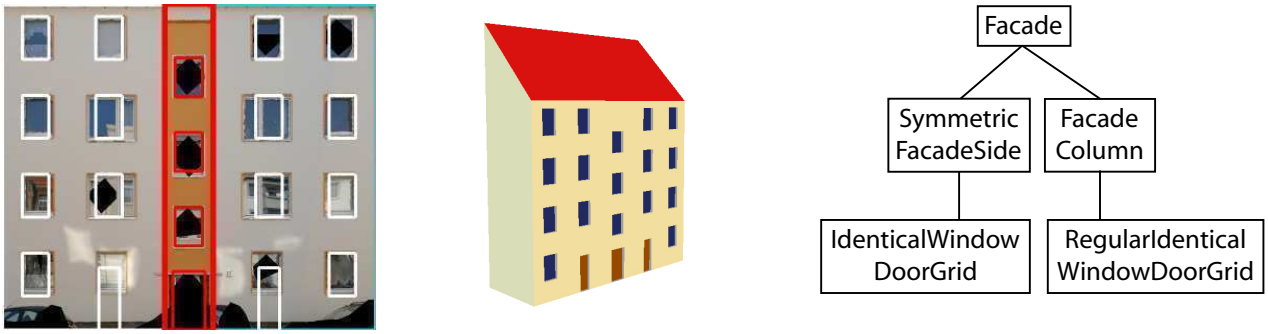


Abbildung 7.15: Rekonstruktion der Fassade Nelkenstraße.

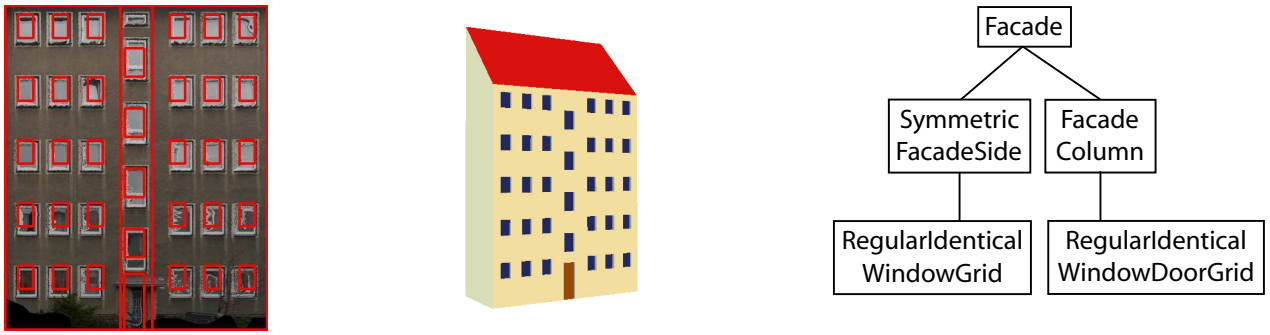


Abbildung 7.16: Rekonstruktion der Fassade Halketstraße.

Bild der Fassade das linke Fenster etwas zu klein und das rechte etwas zu groß modelliert ist. Betrachtet man allerdings das zur Rekonstruktion verwendete Clusterbild in Abbildung 7.17, so sieht man, das der Fensterbereich größer dargestellt ist, da um das Fenster ein zurückgesetzter Bereich der Fassade liegt. Somit liefert ein größeres Fenster an mittlerer Position im Erdgeschoss eine bessere Bewertung.

Verschachtelte Strukturen werden für die Datensätze Nelkenstraße (Abbildung 7.15) und Halketstraße (Abbildung 7.16) verwendet. Die Fassade Nelkenstraße wurde als symmetrisch mit Mittelteil modelliert. Das ist hier korrekt, da die mittlere Spalte nach unten verschoben ist. Die FacadeColumn wird weiter durch ein RegularIdenticalWindowDoorGrid modelliert. Hierbei sind die Fenster und die Tür korrekt modelliert. Der äußere, durch SymmetricFacadeSide dargestellte, Teil wird weiter durch IdenticalWindowDoorGrid modelliert. Hier tritt das Problem auf, das fälschlicherweise eine Tür modelliert wurde. Die Fassade Halketstraße wurde ebenfalls als symmetrisch mit Mittelteil modelliert. Auch hier ist die mittlere Spalte nach unten versetzt. Diese wurde als RegularIdenticalWindowDoorGrid modelliert und der symmetrische äußere Teil als RegularIdenticalWindowGrid.

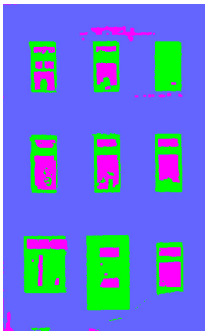


Abbildung 7.17: Zur Rekonstruktion der Fassade Appelstraße verwendetes Clusterbild.

	k_F	v_F	k_W	v_W	k_T	v_T	k	v
Alemannstraße	0,854	0,902	0,978	0,965	–	–	0,953	0,953
Voltastraße	0,872	0,893	0,972	0,965	–	–	0,950	0,950
Schneiderberg I	0,843	0,977	0,958	0,957	1,000	0,592	0,935	0,935
Schneiderberg II	0,955	0,804	0,920	0,985	0,973	0,600	0,950	0,950
Appelstraße	0,891	0,857	0,966	0,975	–	–	0,952	0,952
Nelkenstraße	0,881	0,662	0,918	0,952	0,297	0,935	0,888	0,888
Halketstraße	0,868	0,618	0,891	0,965	0,623	0,928	0,883	0,883

Tabelle 7.3: Korrektheit und Vollständigkeit der Rekonstruktion der einzelnen Datensätze.

Die hier vorgestellten Rekonstruktionsergebnisse wurden ebenfalls auf Korrektheit und Vollständigkeit untersucht. Tabelle 7.3 zeigt die Ergebnisse. Die Korrektheit für Fenster k_F liegt zwischen 0,841 und 0,955 und die Vollständigkeit v_F zwischen 0,618 und 0,977. Die Werte für die Vollständigkeit variieren stärker. Der geringe Wert für die Fassade Halketstraße ist dadurch zu erklären, dass alle Fenster etwas klein modelliert sind und somit viele Randpixel der Fenster nicht als Fenster modelliert sind. Für Wandpixel liegt die Korrektheit k_W zwischen 0,891 und 0,978. Die Vollständigkeit v_W liegt zwischen 0,940 und 0,985. Für Türen können die Werte nur teilweise angegeben werden, da in manchen Fassaden keine Türen vorkommen. Die Werte für die Korrektheit k_T und die Vollständigkeit v_T variieren sehr stark. Sie liegen zwischen 0,0 und 1,0 bzw. 0,935. Dies liegt daran, dass die Türen teilweise falsch modelliert wurden. Für die Fassaden Halketstraße und Nelkenstraße ist die Vollständigkeit gut, da die Tür an der korrekten Position modelliert wurde. Die Korrektheit für die Fassade Nelkenstraße ist mit 0,297 aber sehr gering, da zusätzlich zu der korrekten Tür im symmetrischen Bereich ebenfalls eine Tür modelliert wurde. Für die Fassade Schneiderberg I ist die Tür an der korrekten Position modelliert, aber sie ist deutlich niedriger als die Tür in den Daten. Daraus ergibt sich für die Rekonstruktion der Tür im Datensatz Schneiderberg I eine Korrektheit von 1,0 und eine deutlich geringere Vollständigkeit von 0,592.

Zusätzlich zu den Werten für Fenster, Wände und Türen sind Korrektheit und Vollständigkeit aller Objekte einer Fassade angegeben. Da sowohl im Ausgangsbild als auch in der Rekonstruktion nur Fenster, Türen und Wände modelliert wurden, ergibt sich für die gesamte Korrektheit derselbe Wert wie für die gesamte Vollständigkeit. Beide Werte liegen zwischen 0,870 und 0,953. Die Ergebnisse sind insgesamt etwas besser als die Rekonstruktionsergebnisse aus den eTRIMS-Fassaden. Das könnte unter anderem daran liegen, dass bei der Annotation der eTRIMS-Bilder andere Kriterien verwendet wurden. Zu Fenstern werden dort oft die Bereiche der Fensterbank und verzierender Umrandung hinzugezählt. Fensterläden hingegen gehören nicht zum Fenster und werden, auch wenn sie Teile der Umrandung überdecken, nicht als Fenster annotiert. Dadurch ergeben sich teilweise Fensterformen, die sich nicht mit dem Rechteckmodell der Grammatik modellieren lassen.

Insgesamt ist das Ergebnis der Rekonstruktion gut. Allerdings hängt das Ergebnis stark von der Qualität des Clusterbildes ab. Gerade im unteren Bereich sind Verdeckungen störend, da dort nicht mehr erkannt werden kann, ob ein Fenster oder eine Tür modelliert werden soll. In höheren Bereichen der Fassade ist das Verfahren robust gegenüber Verdeckungen, da dort die Struktur über die Grammatik geschätzt werden kann.

8 Technische Realisierung

Das in dieser Arbeit vorgestellte Rekonstruktionsverfahren ist in C++ implementiert. Hierbei wurde für das Datenmanagement der Bild- und Entfernungsdaten die freie Bibliothek OpenCV (Intel, 2000) verwendet. Der Ablauf der Rekonstruktion wurde schon in Abschnitt 6.1 und Abbildung 6.1 gezeigt. Die Klassenstruktur teilt sich in Klassen zur Prozesssteuerung, Grammatiksymbole, Regel-Klassen und einige Hilfsklassen. Die Grammatiksymbole stellen gleichzeitig Knoten im Ableitungsbaum dar. Deshalb bieten sie zusätzlich Verweise für eine Baumstruktur an. Auf die Hilfsklassen wird hier nur teilweise eingegangen. Nach der Beschreibung der Klassenstruktur wird die Interaktion zwischen den Objekten beispielhaft gezeigt.

8.1 Aufbau der Klassenstruktur

In diesem Abschnitt wird zunächst die grobe Struktur des Programms erklärt, indem das Zusammenspiel einiger Klassen beschrieben wird. Anschließend wird die Klassenstruktur der Grammatiksymbole beschrieben. Hierbei werden zunächst die in einer flachen Hierarchie angeordneten Symbole besprochen und dann auf die komplexere Hierarchie der Gittersymbole eingegangen.

Die Abhängigkeit der Steuerklassen ist im Klassendiagramm[1] in Abbildung 8.1 zu sehen. Die Klasse `RJMCMC` steuert den Ablauf und ist die einzige nach außen sichtbare Klasse. Über die Methoden `init()`[2] und `runRJMCMC()` kann der Ablauf gestartet werden. `RJMCMC` ist assoziiert zu den Klassen `FacadeData`, `TreeManager` und `RuleManager`. Die Klasse `FacadeData` dient als Containerklasse für die Daten. Zusätzlich dient sie zur weiteren Verarbeitung der Daten, was hauptsächlich die Berechnung der Clusterbilder beinhaltet. Nach außen hin sind im Wesentlichen die Methoden `init()` und verschiedene `get`-Methoden sichtbar. Über diese können Daten, berechnete Daten, wie das Clusterbild, und Parameter abgefragt werden.

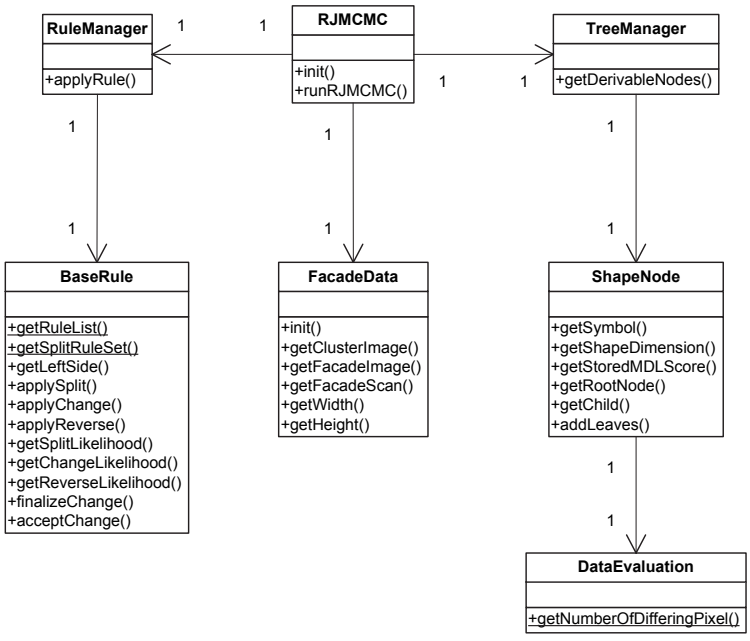


Abbildung 8.1: Gerüst der Klassen zur Fassadenrekonstruktion.

Die Klasse `TreeManager` verwaltet den Ableitungsbaum. Sie enthält einen Zeiger auf den Wurzelknoten, der vom Typ `ShapeNode` ist und von dem aus durch den Baum navigiert werden kann. Die Methode `getDerivableNodes()` ermittelt die Knoten, die aktuell für eine Veränderung im Baum zur Verfügung stehen.

[1] Die Auflistung der Methoden erfolgt nicht vollständig. Es werden nur die im Text behandelten Methoden genannt.
[2] Die Parameter von Methoden werden im gesamten Kapitel aus Gründen der Übersichtlichkeit nicht mit angegeben.

Für die Auswahl und Anwendung von Regeln ist die Klasse `RuleManager` verantwortlich. Sie ist assoziiert zu der Klasse `BaseRule`, die die verfügbaren Regeln bereitstellt. Die Methode `applyRule()` ermittelt mithilfe der Klasse `BaseRule`, welche Regeln anwendbar sind und wählt anhand der Vorschlagswahrscheinlichkeit hieraus eine aus. Die ausgewählte Regel wird angewandt und je nach Akzeptanzwahrscheinlichkeit angenommen oder verworfen.

`BaseRule` enthält eine statische Liste aller Regeln und kann so über die Methode `getSplitRuleSet()` alle auf einen Knoten anwendbaren Regeln ermitteln. Außerdem ist `BaseRule` die Oberklasse aller Regel-Klassen, die in flacher Hierarchie direkt von `BaseRule` abgeleitet werden und die Regeln der Grammatik implementieren. Dabei gibt es Klassen wie `HorizontalSplitRule`, die genau eine Regel abbilden, und Klassen, die mehrere Regeln zusammenfassen. Beispielsweise fasst `ToIdenticalWindowGridRule` drei Regeln zusammen, deren rechte Seite IdenticalWindowGrid ist. Mögliche linke Seiten sind dabei Facade , PartFacade und SymmetricFacadeSide. Das wesentliche Vorgehen ist für diese Regeln gleich. Innerhalb der Klasse kann bei Bedarf nach der linken Seite unterschieden werden.

Insgesamt sind 33 Regel-Klassen implementiert, die die 68 Regeln der Grammatik umsetzen. Allen Regel-Klassen gemein sind die in `BaseRule` deklarierten Methoden `applySplit()`, `applyChange()` und `applyReverse()`. Die Methode `applyChange()` berechnet die Vorschläge für Änderungen nicht selbst, sondern erhält diese von den betroffenen Grammatiksymbolen.

Die Grammatiksymbole sind von der Klasse `ShapeNode` abgeleitet. Die Struktur der abgeleiteten Klassen ist in Abbildung 8.2 gezeigt. Die Klassen sind nicht nur Symbole der Grammatik sondern gleichzeitig auch Knoten des Ableitungsbaums. Jedes Grammatiksymbol ist durch den Namen des Symbols, Position und Größe der beschriebenen Fläche und die zugehörigen Parameter gegeben. Die Struktur des Ableitungsbaums wird mithilfe von Zeigern zum Vorgänger und zu den Nachfolgern realisiert. Ein Teil der Symbole ist direkt von `ShapeNode` abgeleitet. Das sind die Klassen `FacadeElement`, `RepeatedFacade`, `SplitPartFacade`, `NonterminalShape` und `TerminalShape`. Die Klasse `NonterminalShape` fasst die Symbole Facade, FacadeColumn, FacadeRow, PartFacade und SymmetricFacadeSide zusammen und die Klasse `TerminalShape` stellt die Symbole Window, Door, DoorWay und ShopWindow dar.

Die Gitterklassen sind nicht wie die anderen Symbole flach von der Klasse `ShapeNode` abgeleitet. Sie haben unterhalb von `ShapeNode` eine eigene Hierarchie. Diese ist so aufgebaut, dass Vorschlags- und Überprüfungsmethoden nicht mehrfach implementiert werden müssen. Oberklasse der Gitter ist die abstrakte Klasse `AbstractGrid`. Sie enthält die Parameter, die allen Gittern gemein sind. Das sind die Anzahl an Zeilen und Spalten, die Position und der (immer äquidistante) Zeilenabstand. Für den Spaltenabstand gibt es zusätzlich die Möglichkeit, dass dieser variiert. Deshalb wird dieser Parameter erst in den Unterklassen deklariert. Von `AbstractGrid` werden die ebenfalls abstrakten Klassen `AbstractIdenticalWindowGrid` und `AbstractWindowGrid` abgeleitet. Hierbei wird nach der Eigenschaft identischer Fenster unterschieden und die Klassen enthalten Informationen über die Fensterbreite und -höhe. Für Gitter mit nicht identischen Fenstern werden zusätzlich Parameter für die Position verwendet. Diese ist bei identischen Fenstern im Gitterpunkt gegeben. In der nächsten Stufe teilen sich die Klassen nach äquidistantem und variablem Spaltenabstand im Gitter auf. Es gibt die instantiierbaren Klassen `IdenticalWindowGrid`, `RegularIdenticalWindowGrid`, `WindowGrid` und `RegularWindowGrid`, die einen Parameter für den Spaltenabstand besitzen. Je nach Klasse ist das ein Wert oder ein Array, das für jeden Zwischenraum die Breite beinhaltet. Alle Klassen dieser Struktur enthalten Methoden für Vorschläge zur Veränderung ihrer Parameter, beispielsweise `varyNumberY()`, und Methoden, die diese Vorschläge überprüfen und umsetzen wie `updateNumberY()`. Die Methoden sind jeweils an höchstmöglicher Stelle in der Klassenstruktur implementiert. Aus konzeptioneller Sicht wäre auch eine Unterteilung in anderer Reihenfolge oder Aufteilung durch Mehrfachvererbung möglich gewesen. Hier wurde aber diese Variante gewählt, da auf diese Weise die wenigsten Methoden mehrfach implementiert werden müssen.

Abschließend gibt es von jeder dieser Klassen eine Unterklasse, die zusätzlich von der Klasse `DoorParameter` erbt. Diese Klassen haben in der Gitterstruktur zusätzlich eine Tür. Diese kann entweder ein Fenster ersetzen oder zusätzlich auftreten, was durch die Methode `isAdditional()`abgefragt werden kann. Die Tür ist durch ihre Breite und Höhe sowie ihre Position gegeben.

Um die Bewertung zu realisieren, besitzt jede Unterklasse von `ShapeNode` eine Methode `getNumberOfDifferingPixel()`. Diese durchläuft den Baum bis zu den Blättern und veranlassen dort jeweils eine Bewertung des zugehörigen Teils der Fassade. Zur Berechnung der Bewertung wird die Klasse `DataEvaluation` verwendet. Sie berechnet in der Methode `getNumberOfDifferingPixel()` für die einzelnen Symbole die Bewertungsfunktion. Anschließend werden die einzelnen Teilbewertungen aufsummiert.

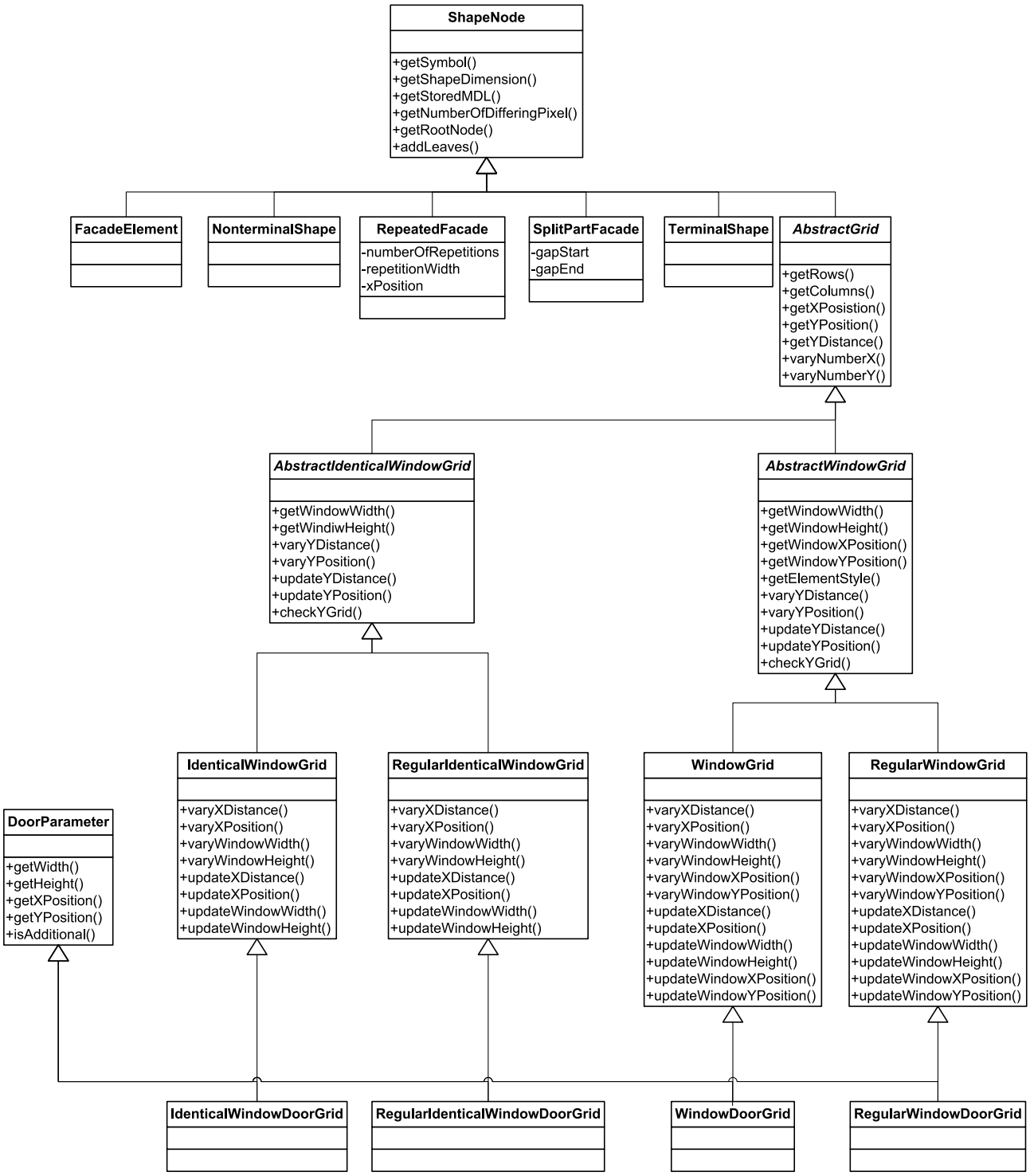


Abbildung 8.2: Die Klassenstruktur der Grammatiksymbole.

8.2 Interaktion zwischen Objekten

Das Zusammenspiel der einzelnen Klassen wird im Sequenzdiagramm in Abbildung 8.4 deutlich. Dort wird eine Iteration im Rekonstruktionsprozess gezeigt. Zu Beginn hat das System den Zustand, der in Abbildung 8.3 zu sehen ist. Die Fassade teilt sich in einen oberen und einen unteren Teil, wobei der obere Teil aus einem regelmäßigen Gitter aus identischen Fenstern besteht.

Die Iterationen werden in einer Instanz der Klasse `RJMCMC` in der Methode `runRJMCMC()` gesteuert. Um die Knoten zu ermitteln, die verändert werden können, fragt sie bei einer Instanz des `TreeManagers` über `getDerivableNodes()` diese Knoten ab. Der `TreeManager` delegiert die Anfrage an den Wurzelknoten `Facade` im Baum. Von hier aus wird der Baum mit dem Befehl `getLeaveList()` durchlaufen. Ergebnis der Anfrage sind in diesem Fall die Knoten `RegularIdenticalWindowGrid` und `FacadeRow`. Eine Split-Operation ist nur auf `FacadeRow` möglich, da für `RegularIdenticalWindowGrid` keine weiteren Regeln zur Verfügung stehen. Change- und Reverse-Operationen können nur auf `RegularIdenticalWindowGrid` ausgeführt werden, da `PartFacade` schon Kinder hat.

Nachdem die möglichen Regeln bekannt sind, ruft `RJMCMC` im `RuleManager` die Methode `applyRule()` auf. Diese wählt mithilfe der Methode `chooseNode()` aus den übergebenen Knoten, basierend auf den Regelwahrscheinlichkeiten, zufällig einen aus. Die Klasse `BaseRule` enthält eine statische Liste aller Regeln und liefert über die Methode `getSplitRuleSet()` alle auf den gewählten Knoten anwendbaren Split-Regeln. Die Change- und Reverse-Regeln stehen unmittelbar durch den gewählten Knoten fest. In diesem Fall handelt es sich um die Regeln *ToRegularIdenticalWindowGridRule*. Von der Instanz `ToRegularIdenticalWindowGridRule` werden über die Methoden `getChangeLikelihood()` und `getReverseLikelihood()` die Vorschlagswahrscheinlichkeiten abgefragt. Die Wahrscheinlichkeiten für die möglichen Split-Regeln werden bei diesen über die Methode `getSplitLikelihood()` abgefragt. In diesem Fall sind es die Regeln *FacadeRowToFacadeElement* und die Regeln, die FacadeRow auf alle möglichen Gitter abbilden. Anhand der Wahrscheinlichkeiten wird eine der Regeln ausgewählt. Für dieses Beispiel wird die Change-Operation der Regel *ToRegularIdenticalWindowGridRule* gewählt. Um sie auszuführen, wird in `ToRegularIdenticalWindowGridRule` die Methode `applyChange()` aufgerufen. Der weitere Verlauf ist der Übersichtlichkeit halber in einem weiteren Sequenzdiagramm in Abbildung 8.5 dargestellt.

Da eine Change-Regel in anderen Fällen auch über Knoten hinweg gehen kann, wie z.B. beim Verändern einer horizontalen Trennlinie, wird immer der Elternknoten an die Change-Methode übergeben. In diesem Beispiel findet die Veränderung innerhalb eines Knotens statt. Diesen Knoten erhält man, indem die Methode `getChild()` des Elternknotens `PartFacade` aufgerufen wird. Über die Methode `getRootNode()` kann der Wurzelknoten ermittelt werden. Das ist hier eine Instanz der Klasse `Facade`. Diese liefert über die Methode `getStoredMDL()` die Bewertung des aktuellen Baums.

Im Folgenden wird eine Veränderung vorgeschlagen und die Bewertung des neuen Baumes bestimmt. Um den alten Zustand zu sichern, wird zunächst eine Kopie des Knotens `RegularIdenticalWindowGrid` erstellt. Der Vorschlag für die Veränderung wird in der Klasse `RegularIdenticalWindowGrid` vorgenommen und über die Methode `change()` angestoßen. In dieser Methode wird aus verschiedenen Möglichkeiten eine Art der Veränderung vorgeschlagen. In diesem Beispiel sei es die Veränderung der Anzahl an Gitterpunkten in x-Richtung. Hierzu wird innerhalb der Klasse die Methode `varyNumberX()` aufgerufen. Diese schlägt eine Veränderung der Anzahl an Spalten vor. Die Methode `updateGridX()` prüft, ob die Änderung gültig ist und nimmt sie gegebenenfalls vor. Wurde in `RegularIdenticalWindowGrid` eine Änderung vorgenommen, so ruft `ToRegularIdenticalWindowGridRule` ihre Methode `finalizeChange()` auf. Diese ermittelt die Bewertung durch Aufruf der Methode

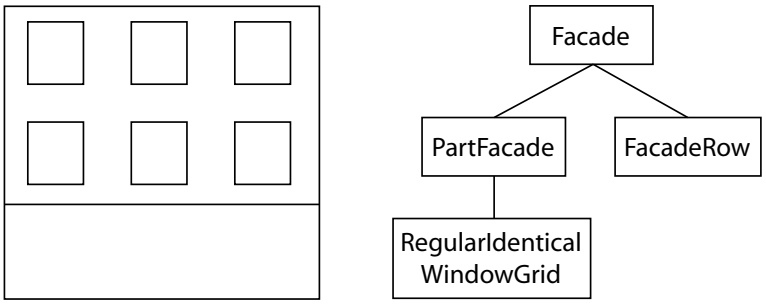


Abbildung 8.3: Zustand der Markov-Kette zu Beginn des Sequenzdiagramms.

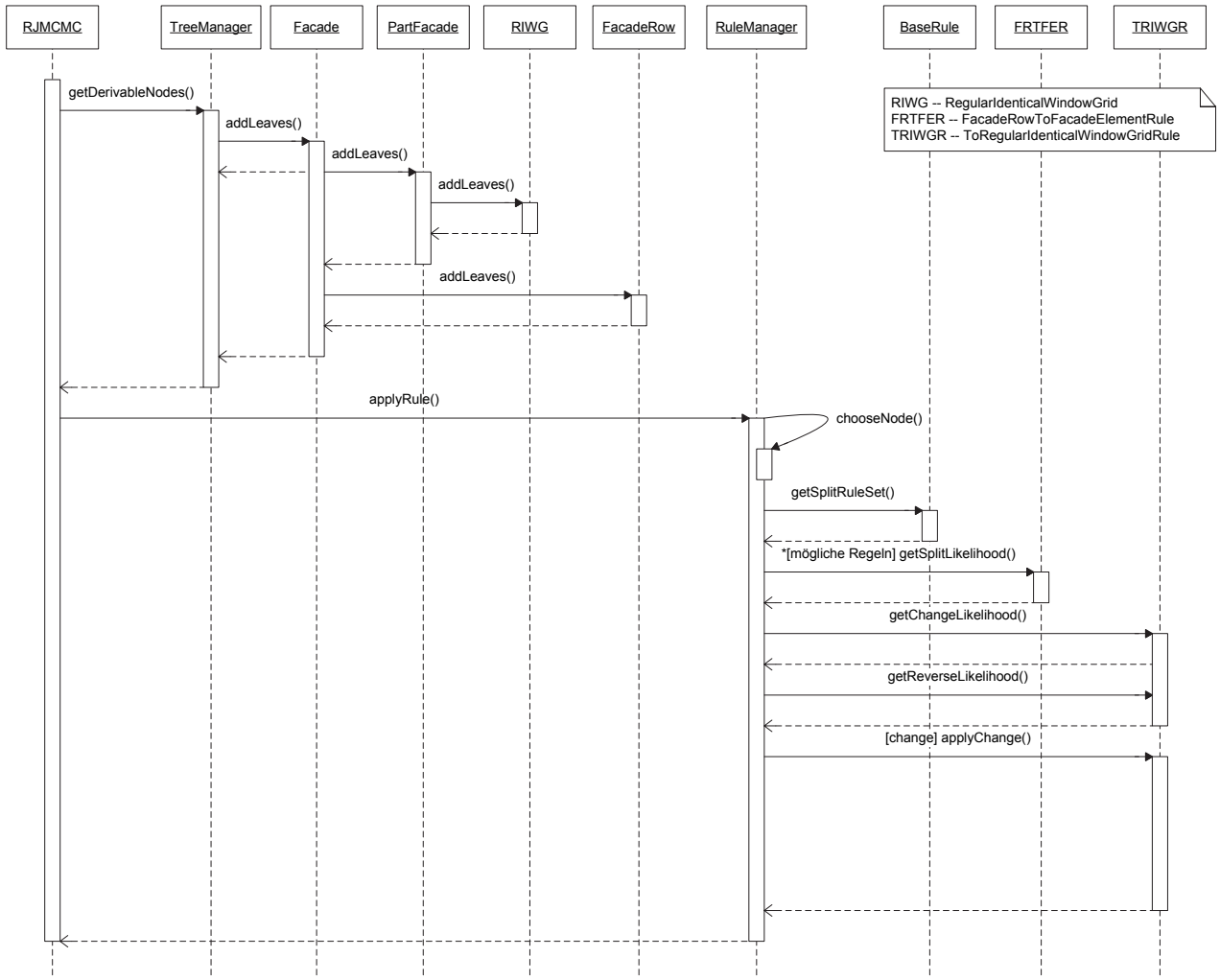


Abbildung 8.4: Beispielhaftes Sequenzdiagramm einer Iteration im Rekonstruktionsprozess.

`getMDLScore()` des Wurzelknotens `Facade`. Dieser propagiert die Methode `getNumberOfDifferingPixel()` bis in die Blätter des Baumes, wo die MDL-basierte Bewertung mithilfe eines `DataEvaluation`-Objektes berechnet wird. Im Beispiel wären zum einen der Knoten `PartFacade` betroffen, der wiederum diese Methode des Nachfolgers `RegularIdenticalWindowGrid` aufruft, und zum anderen der Knoten `FacadeRow`. In der Methode `accaptChange()` wird eine Zufallszahl zwischen 0 und 1 gezogen und die Veränderung in Abhängigkeit der Akzeptanzwahrscheinlichkeit angenommen. In diesem Beispiel wird sie angenommen und die Kopie des alten Zustands kann gelöscht werden. Andernfalls würde die Kopie mit dem alten Zustand in den Baum eingefügt und der aktuelle Zustand gelöscht werden. Damit ist der Durchlauf beendet und eine neue Iteration wird gestartet.

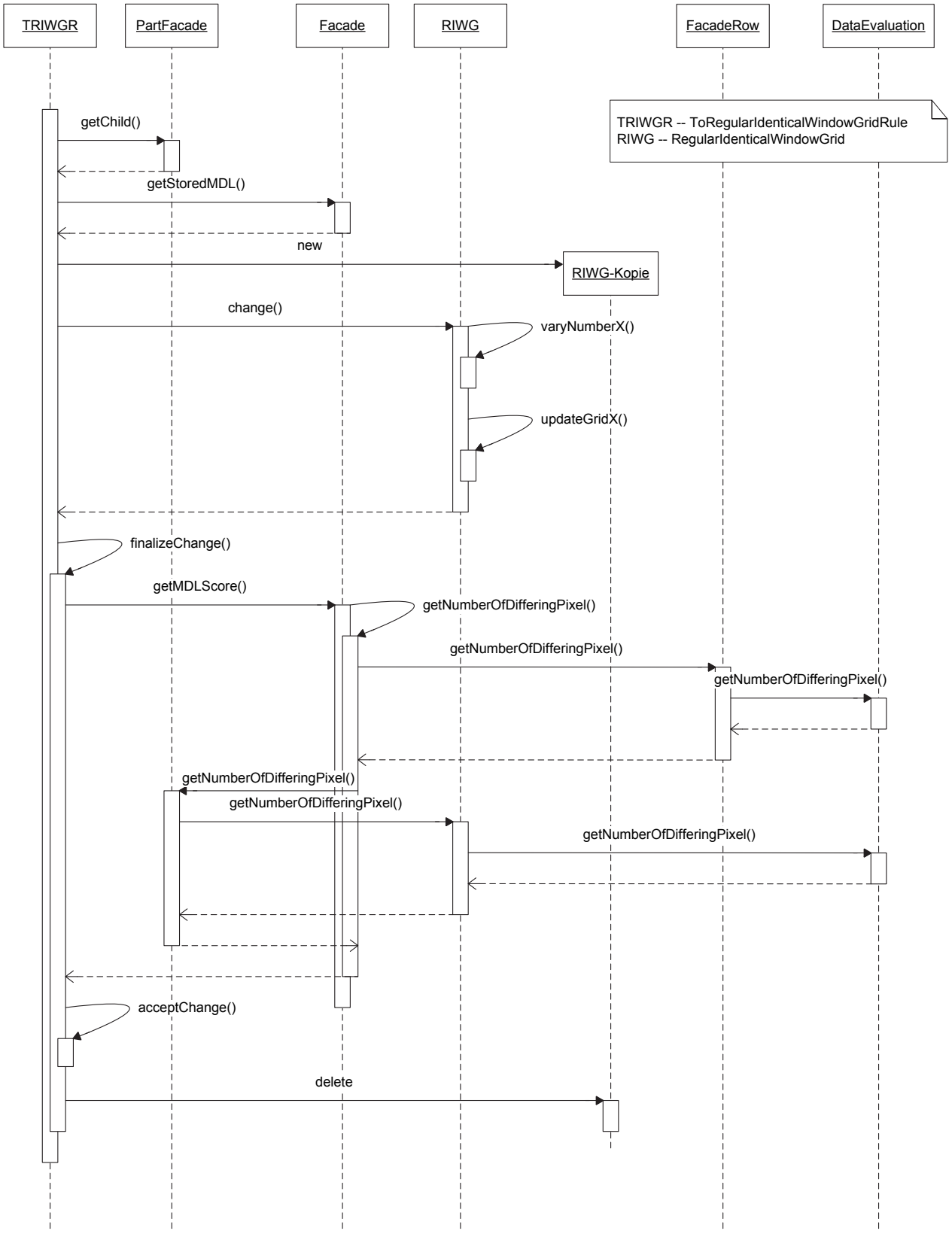


Abbildung 8.5: Sequenzdiagramm der Methode applyChange().

9 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde ein Verfahren zur automatischen Fassadenrekonstruktion aus Scan- und Bilddaten entwickelt. Zu Beginn wurden verschiedene Verfahren zur Objekterkennung aus luftgestützten Daten, sowie Arbeiten zur Fassadenrekonstruktion vorgestellt. Weiterhin wurden Grundlagen vorgestellt, die für das in dieser Arbeit entwickelte Rekonstruktionsverfahren benötigt werden. Zum einen handelt es sich dabei um formale Grammatiken und zum anderen um Markov Chain Monte Carlo Verfahren. Zu beiden Gebieten wurden Arbeiten beschrieben, die diese Ansätze für die Rekonstruktion bzw. Modellierung verwenden.

Das in dieser Arbeit entwickelte Verfahren nutzt die Struktur von Fassaden für die Rekonstruktion. Um Informationen über Fassadenstrukturen zu gewinnen, wurden Fotos von Fassaden aufgenommen und analysiert. Die daraus gewonnenen Informationen wurden in einer formalen Grammatik formuliert. Grammatiken, die bisher zur Fassadenrekonstruktion verwendet wurden, teilen die Fassade nach einem starren Schema zunächst in Zeilen und diese dann weiter in Kacheln auf. Eine Besonderheit der in dieser Arbeit entwickelten Grammatik ist, dass sie über dieses Schema hinausgeht. Sie berücksichtigt wiederholte und symmetrische Strukturen. Diese Strukturen erlauben neben Vorteilen bei der Rekonstruktion eine kompakte Speicherung der Modelle. Des Weiteren können Fassaden modelliert werden, die Spalten enthalten, deren Fenster z.B. um ein halbes Stockwerk verschoben sind. Ein weiterer Vorteil der Grammatik ist, dass durch die Struktur Aussagen über verdeckte Bereiche gemacht werden können. Ist beispielsweise in einem Fenstergitter ein Teil durch einen Baum verdeckt, so lässt sich aus den sichtbaren Fenstern schließen, an welchen Stellen sich wahrscheinlich weitere Fenster befinden.

Mit der Grammatik kann eine Vielzahl von Fassaden modelliert werden. Um eine automatische Rekonstruktion von Fassaden zu ermöglichen, wurde ein Verfahren entwickelt, das mithilfe eines reversible jump Markov Chain Monte Carlo Verfahrens die Ableitung eines Wortes aus den Grammatikregeln automatisch steuert. Dazu wurde eine Markov-Kette generiert, deren Zustände den verschiedenen Entwicklungsstufen der möglichen Ableitungsbäume der Grammatik entsprechen. Um Übergänge zwischen den Zuständen zu definieren, wurden den Regeln der Grammatik Wahrscheinlichkeiten zugewiesen. Zusätzlich wurde eine Bewertungsfunktion entwickelt, die steuert, ob eine vorgeschlagene Zustandsänderung vorgenommen wird, oder der alte Zustand erhalten bleibt.

Die Bewertungsfunktion wurde so definiert, dass sie zum einen Modelle bevorzugt, welche gut zu den Daten passen. Zum anderen wird aber auch die Modellkomplexität berücksichtigt, um nicht für eine kleine Verbesserung der Übereinstimmung von Modell und Daten deutlich komplexere Modelle zu verwenden. Um dies zu realisieren, wurde eine auf der Minimum Description Length basierende Bewertungsfunktion definiert. In dieser Arbeit wurden zwei Ansätze vorgestellt, die Modellkomplexität zu bewerten. Der eine basiert auf Wahrscheinlichkeiten, die als Grundlage für die Bewertung der Komplexität dienen. Ein sehr wahrscheinliches Modell kann mit wenigen Bits beschrieben werden, besitzt also eine geringe Komplexität. Im Gegenzug hat ein unwahrscheinliches Modell eine hohe Komplexität. Die hierbei entstehende Rangfolge der Modelle entspricht nicht immer der Reihenfolge, die ein Mensch empfindet. Der zweite Ansatz zur Bewertung der Modellkomplexität basiert auf der verwendeten Anzahl an Symbolen und Parametern. Wird dieser verwendet, so stimmt die berechnete Komplexität mit der vom Menschen empfundenen gut überein.

Das Verfahren wurde anhand von manuell segmentierten Fassadenbildern und an Entfernungs- und Bilddaten, die mit einem terrestrischen Lasercanner aufgenommen wurden, getestet.

Die eTRIMS-Datenbank stellt 60 manuell segmentierte Fassadenbilder zur Verfügung. Durch diese Bilder wurde ein Test des Rekonstruktionsverfahrens an einer größeren Zahl an Fassaden ermöglicht. Zusätzlich konnte an diesen Bildern getestet werden, wie allgemein die Grammatik einsetzbar ist. In einem weiteren Test wurde der Anteil an Vegetation in einem der Bilder synthetisch erhöht. Der Test hat gezeigt, dass die Rekonstruktion auch bei stärkerer Verdeckung erfolgreich durchgeführt werden kann.

Der Test mit den Entfernungs- und Bilddaten zeigte, dass die Fassaden auch hier erfolgreich rekonstruiert werden können. Auch Fassaden, für die nur Tiefeninformationen vorliegen, können mit diesem Verfahren korrekt rekonstruiert werden. Insgesamt sind die Ergebnisse der Rekonstruktion aus Tiefenbildern und Bilddaten sogar etwas besser als die, der Rekonstruktion aus manuell segmentierten Daten.

Die in dieser Arbeit verwendete Fassadengrammatik wurde aus Fassadenfotos aus Hannover entwickelt. Der Test mit eTRIMS-Bildern hat gezeigt, dass mit der Grammatik auch Fassaden aus anderen (überwiegend deutschen) Städten erfolgreich modelliert werden können. Es sind jedoch auch Situationen vorstellbar, in denen

die Fassadenstrukturen nach anderen Mustern aufgebaut sind. In diesem Fall müssten neue Grammatikregeln und gegebenenfalls auch neue Symbole definiert werden. Der zugrundeliegende Rekonstruktionsprozess ist aber derselbe und könnte übernommen werden.

In dieser Arbeit wird davon ausgegangen, dass der zur Fassade gehörende Bereich in den Daten bekannt ist. Für eine automatische Rekonstruktion von vollständigen Straßenzügen müsste ein Verfahren integriert werden, welches aus der Punktwolke die einzelnen Fassaden ausschneidet. Eine Schwierigkeit hierbei ist allerdings, dass die Fassaden der einzelnen Häuser sich oft nicht stark voneinander unterscheiden. Die Ebene, in der die Fassade liegt, ist in vielen Fällen die gleiche und auch farblich gibt es oft wenig Unterschiede.

Ein weiteres Problem bei der Rekonstruktion großer Datenmengen stellt die Laufzeit des Verfahrens dar. Durch die hohe Anzahl an benötigten Iterationen liegt die Laufzeit für die Rekonstruktion einer Fassade im Bereich einiger Stunden. Die Laufzeit könnte durch Parallelisieren des Verfahrens massiv verkürzt werden. Für die Konvergenz der Markov-Kette spielt es, nach einer Einschwingphase, keine Rolle ob ein Prozess n Iterationen durchläuft oder k Prozesse $\frac{n}{k}$ Iterationen durchlaufen und anschließend zusammengeführt werden. Durch Parallelisieren ließe sich der Prozess also deutlich beschleunigen. Einen weiteren Ansatzpunkt für die Beschleunigung des Verfahrens liefert die Bewertungsfunktion. Hierbei werden hauptsächlich Bilder erzeugt und verglichen. Dieser Prozess könnte auf die Grafikkarte ausgelagert werden und dort mithilfe der Graphics Processing Unit (GPU) extrem effizient ausgeführt werden.

A Wahrscheinlichkeiten der Grammatikregeln

Tabelle A.1 gibt die Wahrscheinlichkeiten aller Regeln der Fassadengrammatik an. Die Anzahl angewendeter Regeln wurde in Fassadenbildern gezählt und die relative Häufigkeit wird als Wahrscheinlichkeit der Regel verwendet.

Facade	→	PartFacade FacadeRow	0,2337
Facade	→	SymmetricFacadeSide FacadeColumn	0,0926
Facade	→	SymmetricFacadeSide	0,0121
Facade	→	PartFacade FacadeColumn	0,0182
Facade	→	FacadeColumn PartFacade	0,0091
Facade	→	SplitPartFacade FacadeColumn	0,0546
Facade	→	RegularIdenticalWindowDoorGrid	0,0622
Facade	→	RegularIdenticalWindowGrid	0,0273
Facade	→	RegularWindowDoorGrid	0,0470
Facade	→	RegularWindowGrid	0,0182
Facade	→	IdenticalWindowDoorGrid	0,0956
Facade	→	IdenticalWindowGrid	0,0258
Facade	→	WindowDoorGrid	0,2458
Facade	→	WindowGrid	0,0546
Facade	→	RepeatedFacade	0,0015
PartFacade	→	SymmetricFacadeSide FacadeColumn	0,0244
PartFacade	→	SymmetricFacadeSide	0,0183
PartFacade	→	RegularIdenticalWindowDoorGrid	0,0122
PartFacade	→	RegularIdenticalWindowGrid	0,2927
PartFacade	→	RegularWindowDoorGrid	0,0061
PartFacade	→	RegularWindowGrid	0,0487
PartFacade	→	IdenticalWindowDoorGrid	0,0061
PartFacade	→	IdenticalWindowGrid	0,2500
PartFacade	→	WindowDoorGrid	0,0122
PartFacade	→	WindowGrid	0,3232
PartFacade	→	RepeatedFacade	0,0061
SymmetricFacadeSide	→	RegularIdenticalWindowDoorGrid	0,0130
SymmetricFacadeSide	→	RegularIdenticalWindowGrid	0,3636
SymmetricFacadeSide	→	RegularWindowDoorGrid	0,0130
SymmetricFacadeSide	→	RegularWindowGrid	0,3116
SymmetricFacadeSide	→	IdenticalWindowDoorGrid	0,0260
SymmetricFacadeSide	→	IdenticalWindowGrid	0,0390
SymmetricFacadeSide	→	WindowDoorGrid	0,0260
SymmetricFacadeSide	→	WindowGrid	0,1948
SymmetricFacadeSide	→	RepeatedFacade	0,0130
SplitPartFacade	→	RegularIdenticalWindowDoorGrid	0,0277
SplitPartFacade	→	RegularIdenticalWindowGrid	0,1111
SplitPartFacade	→	RegularWindowDoorGrid	0,0277
SplitPartFacade	→	RegularWindowGrid	0,0282
SplitPartFacade	→	IdenticalWindowDoorGrid	0,0277
SplitPartFacade	→	IdenticalWindowGrid	0,1666
SplitPartFacade	→	WindowDoorGrid	0,0555
SplitPartFacade	→	WindowGrid	0,5555
RepeatedFacade	→	RegularIdenticalWindowDoorGrid	0,1250
RepeatedFacade	→	RegularIdenticalWindowGrid	0,1250
RepeatedFacade	→	RegularWindowDoorGrid	0,1250
RepeatedFacade	→	RegularWindowGrid	0,1250
RepeatedFacade	→	IdenticalWindowDoorGrid	0,1250
RepeatedFacade	→	IdenticalWindowGrid	0,1250

RepeatedFacade	→	WindowDoorGrid	0,1250
RepeatedFacade	→	WindowGrid	0,1250
FacadeRow	→	RegularIdenticalWindowDoorGrid	0,0795
FacadeRow	→	RegularIdenticalWindowGrid	0,0066
FacadeRow	→	RegularWindowDoorGrid	0,0265
FacadeRow	→	RegularWindowGrid	0,0066
FacadeRow	→	IdenticalWindowDoorGrid	0,0397
FacadeRow	→	IdenticalWindowGrid	0,0132
FacadeRow	→	WindowDoorGrid	0,2981
FacadeRow	→	WindowGrid	0,1060
FacadeRow	→	FacadeElement ... FacadeElement	0,4238
FacadeColumn	→	RegularIdenticalWindowDoorGrid	0,8374
FacadeColumn	→	RegularIdenticalWindowGrid	0,0488
FacadeColumn	→	RegularWindowDoorGrid	0,1057
FacadeColumn	→	RegularWindowGrid	0,0081
FacadeElement	→	Window	0,2432
FacadeElement	→	Door	0,3014
FacadeElement	→	ShopWindow	0,3664
FacadeElement	→	DoorWay	0,0890

Tabelle A.1: Wahrscheinlichkeiten der Grammatikregeln.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

Akaike, H., 1973. Information Theory and an Extension of the Maximum Likelihood Principle. In: Petrov, B. N., Csakis, F. (Hrsg.), *Proceedings of the Second International Symposium on Information Theory*. S. 267–281.

Alegre, F., Dallaert, F., 2004. A Probabilistic Approach to the Semantic Interpretation of Building Facades. In: *International Workshop on Vision Techniques Applied to the Rehabilitation of City Centers*. S. 1–12.

Andrieu, C., Freitas, N. d., Doucet, A., 2003. An Introduction to MCMC for Machine Learning. *Machine Learning* 50, S. 5–43.

Baillard, C., Schmid, C., Zissermann, A., Fitzgibbon, A., 1999. Automatic Line Matching and 3D Reconstruction of Buildings from Multiple Views. In: *ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery*. Bd. 32 von International Archives of Photogrammetry and Remote Sensing. S. 69–80.

Becker, S., Haala, N., 2008. Integrated LIDAR and Image Processing for the Modelling of Building Facades. *Photogrammetrie Fernerkundung Geoinformation* 2, S. 65–81.

Becker, S., Haala, N., Fritsch, D., 2008. Combined Knowlede Propagation for Facade Reconstruction. In: *International Society for Photogrammetry and Remote Sensing, XXI Congress*. Bd. 37 von International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. S. 423–429.

Besl, P. J., McKay, N. D., 1992. A Method for Registration of 3-D Shapes. In: *IEEE, Transactions on Pattern Analysis and Machine Intelligence*. Bd. 14. S. 239–254.

Bishop, C., 2006. Pattern Recognition and Machine Learning. Springer.

Bodendorf, F., 2006. Daten- und Wissensmanagement, 2. Auflage. Springer.

Bokeloh, M., Berner, A., Wand, M., Seidel, H.-P., Schilling, A., 2009. Symmetry Detection using Line Features. In: *Computer Graphics Forum (Proceedings of Eurographics)*. Bd. 28. S. 697–706.

Brenner, C., 2000. Dreidimensionale Gebäuderekonstruktion aus digitalen Oberflächenmodellen und Grundrissen. Dissertation, Universität Stuttgart, Deutsche Geodätische Kommission, Reihe C, Heft Nr. 530.

Brenner, C., Haala, N., Fritsch, D., 2001. Towards Fully Automated 3D City Model Generation. In: Baltsavias, E., Gruen, A., van Gool, L. (Hrsg.), *Automatic Extraction of Man-Made Objects From Aerial and Space Images (III)*. S. 47–57.

Brzank, A., Heipke, C., 2007. Supervised Classification of Water Regions from Lidar Data in the Wadden Sea using a Fuzzy Logic Concept. In: *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*. Bd. XXXVI von International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. S. 90–95.

Burns, J. B., Hanson, A. R., Riseman, E. M., 1986. Extracting Straight Lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (4), S. 425–443.

Cederberg, J., 2001. A Course in Modern Geometries, 2. Auflage. Undergraduate Texts in Mathmatics. Springer.

Chomsky, N., 1956. Three models for the description of language. *IRE Transactions on Information Theory* 2, S. 113–124.

Crevier, D., Lepage, R., 1997. Knowledge-Based Image Understanding Systems: A Survey. *Computer Vision and Image Understanding* 67 (2), S. 161–185.

Dick, A., Torr, P., Cipolla, R., Ribarsky, W., 2004. Modelling and Interpretation of Architecture from Several Images. *International Journal of Computer Vision* 60(2), S. 111–134.

Eidenbenz, C., Kaeser, C., Baltsavias, E., 2000. ATOMI – Automated Reconstruction of Topographic Objects from Aerial Images using Vectorized Map Information. In: *Proceedings of 19. ISPRS Congress*. Bd. XXXIII von International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. S. 462–471.

Fischer, A., Kolbe, T., Lang, F., Cremers, A., Förstner, W., Plümer, L., Steinhage, V., 1998. Extracting Buildings from Aerial Images Using Hierarchical Aggregation in 2D and 3D. *Computer Vision and Image Understanding* 72 (2), S. 185–203.

Förstner, W., Gülch, E., 1987. A Fast Operator for Detection and Precise Location of Distinct Points, Corners and Centers of Circular Features. In: *Proceedings of the ISPRS Intercommission Workshop on Fast Processing of Photogrammetric Data*. S. 281–305.

Fua, P., Hanson, A., 1989. Objective Functions for Feature Discrimination. In: *Proceedings of the Eleventh International Joint Conference on Artificial Intellegence*. Morgan Kaufmann, S. 1596–1602.

Goodenough, D., Goldberg, M., Plunkett, G., Zelek, J., 1987. An Expert System for Remote Sensing. *IEEE Transactions on Geoscience and Remote Sensing* GE-25 (3), S. 349–359.

Green, P. J., 1995. Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination. *Biometrika* 82 (4), S. 711–732.

Gröger, G., Kolbe, T. H., Czerwinski, A., 2007. Candidate OpenGIS® CityGML Implementation Specification. OGC Best Practices Document.

Hansen, C., Henderson, T., 1989. CAGD-based Computer Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11 (11), S. 1181–1193.

Hartigan, J. A., Wong, M. A., 1979. Algorithm AS 136: A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28 (1), S. 100–108.

Hastings, W. K., 1970. Monte Carlo Sampling Methods using Markov Chains and their Applications. *Biometrika* 57 (1), S. 97–109.

Hayes-Roth, F., Waterman, D., Lenat, D. (Hrsg.), 1983. Building Expert Systems. Reading. Addison-Wesley.

Hedtstück, U., 2004. Einführung in die Theoretische Informatik. Formale Sprachen und Automatentheorie. Oldenbourg.

Hoffman, R., Keshavan, H., Towfiq, F., 1989. CAD-Driven Machine Vision. *IEEE Transactions on Systems, Men and Cybernetics* 19 (6), S. 1477–1488.

Hough, P. V. C., 1962. Method and Means for Recognizing Complex Patterns. U.S. Patent 3.069.654.

Hunter, G., Cox, C., Kremer, J., 2006. Development of a Commercial Laser Scanning Mobile Mapping System - StreetMapper. In: *Proceedings of the second International Workshop of the Future of Remote Sensing*. S. 18–24.

Intel, 2000. Open Source Computer Vision Library, Reference Manual.
URL `http://developer.intel.com`

Jensen, F., Nielsen, T., 2007. Bayesian Networks and Decision Graphs. Bd. 2 von Information Science & Statistics. Springer Science + Business Media.

Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P., 1983. Optimization by Simulated Annealing. *Science* 220, S. 671–680.

Knuth, D., 1968. Semantics of Context-Free Languages. *Theory of Computing Systems* 2 (2), S. 127–145.

Koch, H., Pakzad, K., R.Tönjes, 1997. Knowledge Based Interpretation of Aerial Images and Maps Using a Digital Landscape Model as Partial Interpretation. In: Förstner, W., Plümer, L. (Hrsg.), *Semantic Modeling for the Acquisition of Topographic Information from Images and Maps*. Birkhäuser Verlag, S. 3–19.

Korč, F., Förstner, W., 2009. eTRIMS Image Database for Interpreting Images of Man-Made Scenes. Tech. Rep. TR-IGG-P-2009-01.
URL `http://www.ipb.uni-bonn.de/projects/etrims_db/`

Krengel, U., 2000. Einführung in die Wahrscheinlichkeitstheorie und Statistik, 5. Auflage. vieweg studium - Aufbaukurs Mathematik. vieweg.

Kullback, S., 1959. Information theory and statistics. John Wiley and Sons, New York.

Kulschewski, K., 1997. Building Recognition with Bayesian Networks. In: Förstner, W., L.Plümer (Hrsg.), *Semantic Modeling for the Acquisition of Topographic Information from Images and Maps*. Birkhäuser Verlag, S. 196–210.

Lafarge, F., Descombes, X., Zerubia, J., Pierrot-Deseilligny, M., 2008. Automatic Building Extraction from DEMs using an Object Approach and Application to the 3D-city Modeling. *Journal of Photogrammetry and Remote Sensing* 63 (3), S. 365–381.

Lafarge, F., Descombes, X., Zerubia, J., Pierrot-Deseilligny, M., 2009. Structural approach for building reconstruction from a single DSM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*To appear, preprint at http://www2.computer.org/portal/web/csdl/doi/10.1109/TPAMI.2008.281.

Liedtke, C.-E., Bückner, J., Grau, O., Growe, S., Tönjes, R., 1997. AIDA: A System for the Knowledge Based Interpretation of Remote Sensing Data. In: *Third International Airborne Remote Sensing Conference and Exhibition*. S. 7–10.

Liedtke, C.-E., Bückner, J., Pahl, M., Stahlhut, O., 2001. Knowledge based system for the interoperation of complex scenes. In: Baltsavias, E., Gruen, A., van Gool, L. (Hrsg.), *Automatic Extraction of Man-Made Objects From Aerial and Space Images (III)*. S. 3–12.

Liu, J. S., 2001. Monte Carlo Strategies in Scientific Computing. Springer Series in Statistics. Springer.

Liu, Y., Collins, R., Tsin, Y., 2004. A Computational Model for Periodic Pattern Perception Based on Frieze and Wallpaper Groups. *IEEE Transactions on Pattern Analysis and Machine Intellegence* 26 (3), S. 354–371.

MacQueen, J., 1967. Some Methods for Classification and Analysis of Multivariate Observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*. Bd. 1. S. 281–297.

Marshall, A., 1956. The Use of Multi-stage Sampling Schemes in Monte Carlo Computations. In: Meyer, M. (Hrsg.), *Symposium on Monte Carlo Methods*. Wiley, S. 123–140.

Marvie, J.-E., Perret, J., Bouatouch, K., 2005. The FL-system: A Functional L-System for Procedural Geometric Modeling. *The Visual Computer* 21(5), S. 329 – 339.

Mayer, H., Reznik, S., 2006. Building Facade Interpretation from Uncalibrated Wide-Baseline Image Sequences. *ISPRS Journal of Photogrammetry & Remote Sensing* 61, S. 371–380.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., Teller, E., 1953. Equations of State Calculations by Fast Computingmachines. *Journal of Chemical Physics* 21 (6), S. 1087–1091.

Mitchell, W. J., 1990. The Logic of Architecture : Design, Computation, and Cognition. The MIT Press.

Müller, P., Wonka, P., Haegler, S., Ulmer, A., van Gool, L., 2006. Procedural Modeling of Buildings. *Proceedings of ACM SIGGRAPH 2006 / ACM Transactions on Graphics* 25 (3), S. 614–623.

Müller, P., Zeng, G., Wonka, P., van Gool, L., 2007. Image-based Procedural Modeling of Facades. *Proceedings of ACM SIGGRAPH 2007 / ACM Transactions on Graphics* 26 (3), S. 85.

Murai, H., Omatu, S., 1997. Remote Sensing Image Analysis using a Neural Network and Knowledge-Based Processing. *International Journal of Remote Sensing* 18 (4), S. 811–828.

Neal, R. M., 1993. Probabilistic Inference using Markov Chain Monte Carlo Methods. Tech. Rep. CRG-TR-93-1, University of Toronto.
URL `citeseer.ist.psu.edu/neal93probabilistic.html`

Niederöst, M., 2000. Reliable Reconstructions of Buildings for Digital Map Revision. In: *International Archives of Photogrammetry and Remote Sensing*. Bd. 33. S. 635–642.

Parish, Y., Müller, P., 2001. Procedural Modeling of Cities. In: Fiume, E. (Hrsg.), *Proceedings of ACM SIGGRAPH 2001*. ACM Press, S. 301–308.

Pauly, M., Mitra, N. J., Wallner, J., Pottmann, H., Guibas, L., 2008. Discovering Structural Regularity in 3D Geometry. *ACM Transactions on Graphics* 27 (3), S. 1–11.

Prusinkiewicz, P., Lindenmayer, A., 1990. The algorithmic beauty of plants. New York, NY: Springer.

Pu, S., 2008. Advances in 3D Geoinformation Systems. Lecture Notes in Geoinformation and Cartography. Springer, Kap. Automatic Building Modeling from Terrestrial Laser Scanning, S. 147–160.

Reimer, U., 1991. Einführung in die Wissensrepräsentation. Leitfäden der angewandten Informatik. B.G. Teubner Stuttgart.

Reiter, R., Mackworth, A., 1989. A Logical Framework for Depiction and Image Interpretation. *Artificial Intelligence* 41 (2), S. 125–155.

Riegl, 2004. RiSCAN PRO, Manual.
URL `www.riegl.com`

Ripperda, N., 2008a. Determination of Facade Attributes for Facade Reconstruction. In: *International Society for Photogrammetry and Remote Sensing, XXI Congress*. Bd. 37 von International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. S. 285–290.

Ripperda, N., 2008b. Grammar Based Facade Reconstruction using rjMCMC. *Photogrammetrie Fernerkundung Geoinformation* 2, S. 83–92.

Ripperda, N., Brenner, C., 2009a. Application of a Formal Grammar to Facade Reconstruction in Semiautomatic and Automatic Environments. In: *Proceedings of 12th AGILE Conference on GIScience (auf CD)*.

Ripperda, N., Brenner, C., 2009b. Evaluation of Structure Recognition Using Labelled Facade Images. In: Denzler, J., Notni, G., Süße, H. (Hrsg.), *Pattern Recognition, 31st DAGM Symposium Proceedings*. Bd. 5748 von Lecture Notes in Computer Science. Springer, S. 532–541.

Rissanen, J., 1978. Modeling by Shortest Data Description. *Automatica* 14, S. 465–471.

Schmid, C., Zisserman, A., 1997. Automatic line matching across views. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. S. 666–671.

Sowmya, A., Trinder, J., 2000. Modelling and Representation Issues in Automated Feature Extraction from Aerial and Satellite Images. *ISPRS Journal of Photogrammetry & Remote Sensing* 55, S. 34–47.

Stilla, U., 1995. Map-Aided Structural Analysis of Aerial Images. *ISPRS Journal of Photogrammetry and Remote Sensing* 50 (4), S. 3–10.

Stilla, U., Jurkiewicz, K., 1999. Automatic Reconstruction of Roofs from Maps and Elevation Data. In: *International Archieves of Photogrammetry and Remote Sensing*. Bd. 32. S. 139–143.

Stilla, U., Michaelsen, E., 1997. Semantic Modelling of Man-Made Objects by Production Nets. In: Gruen, A., Baltsavias, E., Henricsson, O. (Hrsg.), *Automatic Extraction of Man-Made Objects from Aerial and Space Images (III)*. Birkhäuser Verlag, S. 43–52.

Stiny, G., Gips, J., 1972. Shape Grammars and the Generative Specification of Painting and Sculpture. Auerbach, Philadelphia, S. 125–135.

Strat, T., Fischler, M., 1991. Context-Based Vision: Recognizing Objects Using Information from Both 2-D and 3-D Imagery. *Pattern Analysis and Machine Vision* 13 (10), S. 1050–1065.

Suveg, I., Vosselman, G., 2002. Localization and Generation of Building Models. In: *International Archives of Photogrammetry and Remote Sensing*. Bd. 34. S. 356–360.

Uden, M., 2008. Entwicklung einer Benutzeroberfläche zur manuellen Fassadenmodellierung mittels Grammatiken. Bachelorarbeit, Institut für Kartographie und Geoinformatik, Leibniz Universität Hannover (unveröffentlicht).

van Gool, L., Zeng, G., van den Borre, F., Müller, P., 2007. Towards Mass-Produced Building Models. In: *Photogrammetric Image Analysis 2007*. Bd. 36 von International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. S. 209–220.

Velthuijsen, H., 1992. The Nature and Applicability of the Blackboard Architecture. PPT Research.

von Neumann, J., 1951. Various Techniques used in Connection with Random Digits. *National Bureau of Standards Applied Mathematics Series* 12, S. 36–38.

Wonka, P., Wimmer, M., Sillion, F., Ribarsky, W., 2003. Instant Architecture. *ACM Transaction on Graphics* 22(3), S. 669–677.

Zheltov, S., Sibiryakov, A., Bibitchev, A., 2001. Building extraction at the State Research Institute of Aviation Systems (GosNIIAS). In: Baltsavias, E., Gruen, A., van Gool, L. (Hrsg.), *Automatic Extraction of Man-Made Objects From Aerial and Space Images (III)*. S. 65–74.

Lebenslauf

Persönliche Daten

Name	Nora Ripperda
Geboren am	18. September 1978 in Oldenburg
Familienstand	ledig

Schulbildung

1985 – 1989	König-Christian-Grundschule Glückstadt
1989 – 1998	Detlefsenschule Glückstadt (Gymnasium)
25. Juni 1998	Allgemeine Hochschulreife

Studium

10/1998 – 09/2004	Diplomstudiengang Mathematik mit Studienrichtung Informatik, Leibniz Universität Hannover
02.09.2004	Diplom in Mathematik, Leibniz Universität Hannover

Beruf

10/2004 – 12/2009	wissenschaftliche Mitarbeiterin, Institut für Kartographie und Geoinformatik, Leibniz Universität Hannover

Danksagung

Die vorliegende Arbeit entstand während meiner Zeit als wissenschaftliche Mitarbeiterin in der Nachwuchsgruppe „Automatische Methoden zur Fusion, Reduktion und konsistenten Kombination komplexer heterogener Geoinformation“ am Institut für Kartographie und Geoinformatik der Leibniz Universität Hannover.

Ich danke meinem Betreuer Dr. Claus Brenner dafür, dass er mir die Mitarbeit in der Nachwuchsgruppe ermöglicht hat und ebenso für seine vielen Anregungen und Ideen sowie seine Unterstützung meiner wissenschaftlichen Arbeit. Ebenfalls möchte ich mich bei Frau Prof. Dr. Monika Sester für die Förderung meiner Arbeit bedanken.

Bei Herrn Prof. Dr. Helmut Mayer und Herrn Prof. Dr. Christian Heipke bedanke ich mich für die freundliche Übernahme des Korreferats.

Der Volkswagenstiftung danke ich für die finanzielle Unterstützung meines Projektes.

Dankbar bin ich auch meinen Kollegen, die für eine tolle Arbeitsatmosphäre gesorgt haben. Insbesondere danke ich Birgit Kieler und Christoph Dold für ihre fachliche und moralische Unterstützung.

Besonders dankbar bin ich meiner Familie, meinen Eltern Regina und Hermann sowie meinen Geschwistern Katrin und Sebastian für die beständige Unterstützung und ihren Rückhalt. Bei Katrin möchte ich mich zusätzlich für das Korrekturlesen dieser Arbeit bedanken.

Ganz besonders möchte ich mich bei Jennis Meyer-Spradow bedanken. Er hat mich während der Entstehung dieser Arbeit liebevoll unterstützt, mich immer wieder motiviert und stets in meiner Arbeit bestärkt.