Statistical Part-based Models for Object Detection in Large 3D Scans

Martin Sunkel

Max-Planck-Institut Informatik
Saarbrücken, Germany

Dissertation zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.–Ing.)
der Naturwissenschaftlich-Technischen Fakultäten
der Universität des Saarlandes

Eingereicht am 29.05.2013 in Saarbrücken.

Dekan — Dean

Prof. Dr. Mark Groves	Universität des Saarlandes Saarbrücken, Germany

Betreuender Hochschullehrer — Supervisor

Dr. Michael Wand	Max-Planck-Institut Informatik Saarbrücken, Germany

Gutachter — Reviewers

Prof. Dr. Hans-Peter Seidel	Max-Planck-Institut Informatik Saarbrücken, Germany
Dr. Michael Wand	Max-Planck-Institut Informatik Saarbrücken, Germany

Kolloquium — Examination

Datum – Date
17.09.2013

Vorsitz – Chair

Prof. Dr. Philipp Slusallek	Universität des Saarlandes Saarbrücken, Germany

Prüfer – Examiners

Prof. Dr. Hans-Peter Seidel	Max-Planck-Institut Informatik Saarbrücken, Germany
Dr. Michael Wand	Max-Planck-Institut Informatik Saarbrücken, Germany

Protokoll – Reporter

Dr. Kwang in Kim	Max-Planck-Institut Informatik Saarbrücken, Germany

Abstract

3D scanning technology has matured to a point where very large scale acquisition of high resolution geometry has become feasible. However, having large quantities of 3D data poses new technical challenges. Many applications of practical use require an understanding of semantics of the acquired geometry. Consequently scene understanding plays a key role for many applications.

This thesis is concerned with two core topics: 3D object detection and semantic alignment. We address the problem of efficiently detecting large quantities of objects in 3D scans according to object categories learned from sparse user annotation. Objects are modeled by a collection of smaller sub-parts and a graph structure representing part dependencies. The thesis introduces two novel approaches: A part-based chain-structured Markov model and a general part-based full correlation model. Both models come with efficient detection schemes which allow for interactive run-times.

Kurzfassung

Die Technologie für 3-dimensionale bildgebende Verfahren (3D Scans) ist mittlerweile an einem Punkt angelangt, an dem hochaufglöste Geometrie-Modelle für sehr große Szenen erstellbar sind. Große Mengen dreidimensionaler Daten stellen allerdings neue technische Herausforderungen. Viele Anwendungen von praktischem Nutzen erfordern ein semantisches Verständnis der akquirierten Geometrie. Dementsprechend spielt das sogenannte "Szenenverstehen" eine Schlüsselrolle bei vielen Anwendungen.

Diese Dissertation beschäftigt sich mit 2 Kernthemen: 3D Objekt-Detektion und semantische (Objekt-) Anordnung. Das Problem hierbei ist, große Mengen von Objekten effizient in 3D Scans zu detektieren, wobei die Objekte aus bestimmten Objektkategorien entstammen, welche mittels gerinfügiger Annotationen durch den Benutzer gelernt werden. Dabei werden Objekte modelliert durch eine Ansammlung kleinerer Teilstücke und einer Graph-Struktur, welche die Abhängigkeiten der Einzelteile repäsentiert. Diese Arbeit stellt zwei neuartige Ansätze vor: Ein Markov-Modell, das aus einer teilebasierten Kettenstruktur besteht und einen generellen Ansatz, der auf einem Modell mit voll korrelierten Einzelteilen beruht. Zu beiden Modellen werden effiziente Detektionsschemata aufgezeigt, die interaktive Laufzeiten ermöglichen.

Summary

3D scanning technology has matured to a point where very large scale acquisition of high resolution geometry has become feasible. Using mobile LIDAR scanners, point clouds at centimeter resolution of complete countries can be captured at economically viable costs. Cost efficient approaches such as structure-from-motion reconstruction from community photo collections complement these efforts.

Having large quantities of 3D data poses new technical challenges. A key problem is semantic scene understanding: Almost any application beyond simple 3D rendering, such as mobile navigation, requires an understanding of the semantics of acquired geometry, e.g. finding roads or entrances to buildings. Acquiring these data in a large scale by human annotation is obviously infeasible.

From the variety of problems for semantic scene understanding this thesis studies two core topics: Object detection and semantic alignment.

We address the problem of detecting object instances (*shapes*) in 3D geometry (*point clouds*) according to semantic object classes. Shapes are defined as a set of distinctive parts prescribing local part appearance (*local shape*) as well as the spatial layout (*constellation*) of these parts. From only a small number of hand-annotated examples, a part-based statistical object model is derived to retrieve large quantities of further object instances, while the part-based object structure also encodes a natural correspondence structure for objects. This work is concerned with the construction and analysis of robust statistical models for this problem as well as with the derivation of efficient and robust inference and detection schemes for such models.

The first part of this thesis is devoted to give an introduction to, and definitions of the theoretical concepts used within this work. Further, there will be explanations and derivations how these concepts are related to each other.

The main contributions of this thesis will be given in Chapters 6 and 7. There, we will see how object correspondences can be encoded with graphical models and how such models can be trained with a minimal amount of user interaction. The chapters derive two different models which allow for efficient object detection in large point clouds.

Chapter 6 presents a statistical parts model based on a Markov chain and discusses the motivations for using a chain topology model. Besides model

definition (which relies on geometry with a predominant orientation), the chapter provides a detection scheme that contributes to a drastic reduction of the search space and explains how a large number of shapes can be detected simultaneously.

Chapter 7 depicts a more general part-based approach for the detection problem. We introduce a statistical graphical model which correlates all object parts in terms of local shape and constellation. It is fully rotational invariant and can be extended to a hierarchical version in order to efficiently model and detect more complex objects. Since fully connected graphical models cannot be exactly inferred in feasible time, the chapter provides an approximate but robust and efficient detection scheme – an approach which has not been utilized yet. Similar to the Markov chain model there is also a strategy, how search space can be further depleted for such general models.

This work will conclude with a final general discussion about different graphical models for the detection scenario of this thesis and give a prospect to future applications.

Zusammenfassung

Die Technologie für 3-dimensionale bildgebende Verfahren (3D Scans) ist mittlerweile an einem Punkt angelangt, an dem hochaufglöste Geometrie-Modelle für sehr große Szenen erstellbar sind. So können unter Verwendung von LIDAR-Scannern Punktwolken von ganzen Ländern in Zentimeterauflösung zu ökonomisch vertretbaren Kosten hergestellt werden. Auch kostengünstige Ansätze wie structure-from-motion Rekonstruktionen aus Bilderkollektionen von Photo-Gemeinschaften vervollständigen diese Bemühungen.

Große Mengen dreidimensionaler Daten stellen allerdings neue technische Herausforderungen. Viele Anwendungen von praktischem Nutzen erfordern ein semantisches Verständnis der akquirierten Geometrie. Dementsprechend spielt das sogenannte "Szenenverstehen" eine Schlüsselrolle bei vielen Anwendungen: Fast jede Anwendung jenseits von einfachem 3D-Rendern wie zum Beispiel mobile Navigationsanwendungen erfordert ein Verständnis der Semantik der erfassten Szene, um zum Beispiel Straßen oder Gebäudeeingänge zu finden. Diese Daten mittels Annotation durch Menschen im großen Stil zu beschaffen, ist offensichtlich nicht machbar.

Von der Vielfalt an Problemen beim semantischen Szenenverstehen studiert diese Arbeit zwei Kernthemen: dreidimensionale Objekt-Detektion und semantische (Objekt-) Anordnung.

Wir wenden uns dem Problem zu, Objektinstanzen (*Shapes*) in 3D Geometrie (*Punktwolken*) entsprechend semantischer Objektklassen zu finden. Shapes werden als eine Menge markanter Teile definiert, die sowohl das lokale Erscheinungsbild als auch die räumliche Anordnung (*Konstellation*) dieser Teile beschreiben. Unter Zuhilfenahme einer nur kleinen Anzahl handannotierter Beispiele wird ein teilebasiertes statistisches Objektmodell abgeleitet, um große Mengen weiterer Objektinstanzen zu finden. Dabei kodiert die teilebasierte Objektstruktur auch eine natürliche Korrespondenzstruktur für Objekte. Diese Arbeit beschäftigt sich mit der Konstruktion und Analyse stabiler statistischer Modelle für dieses Problem sowie mit der Herleitung effizienter und robuster Inferenz- und Detektions-Schemata für solche Modelle.

Der erste Teil der Arbeit ist dazu bestimmt eine Einführung in, und Definitionen der benötigten theoretischen Konzepte zu geben. Außerdem werden Erklärungen und Herleitungen aufgezeigt wie diese Konzepte zusammenhängen.

Der Hauptbeitrag der Dissertation ist verteilt auf die Kapitel 6 und 7. Dort

sehen wir, wie man Objektkorrespondenzen mit graphischen Modellen beschreiben kann und wie solche Modelle mit minimalem Benutzeraufwand trainiert werden können. In beiden Kapiteln werden zwei unterschiedliche Modelle vorgestellt, die effiziente Objektdetektion in großen Punktwolken ermöglichen.

Kapitel 6 stellt ein statistisches Teile-Modell vor, das auf einer Markov Kette basiert und diskutiert die Beweggründe, ein Modell mit Kettentopologie zu wählen. Neben der Modelldefinition (die sich auf Geometrie mit ausgewiesener Orientierung beschränkt) zeigt das Kapitel ein Detektionsschema auf, welches zu einer drastischen Reduktion des Suchraums beiträgt und erklärt, wie man eine große Anzahl von Shapes simultan detektieren kann.

Kapitel 7 zeigt einen allgemeineren, Teile-basierten Ansatz für das Detektionsproblem auf. Dort wird ein statistisches graphisches Modell vorgestellt, das alle Objektteile hinsichtlich des lokales Erscheinungsbildes als auch der räumlichen Konstellation miteinander korreliert. Es ist vollständig rotationsinvariant und kann zu einer hierarchischen Version ausgebaut werden, um komplexere Objekte effizient modellieren und detektieren zu können. Da vollständig konnektierte graphische Modelle nicht in machbarer Zeit exakt berechnet werden können, wird im Kapitel ein approximatives aber robustes und effizientes Detektionsschema vorgestellt – ein Ansatz, der zuvor noch nicht genutzt worden ist. Ähnlich dem Markov Kettenmodell gibt es ebenfalls eine Strategie, wie der Suchraum für solche Modelle weiter eingeschränkt werden kann.

Die Doktorarbeit schließt mit einer finalen, allgemeinen Diskussion über verschiedene graphische Modelle für das Detektionsszenario dieser Arbeit und gibt einen Ausblick auf zukünftige Anwendungen.

Acknowledgements

Just like any larger work, a dissertation is rarely possible without guidance, inspiration, or help of several individuals. In one way or the other, it is the entire environment that assists in and contributes to the preparation and completion of bigger projects such as this work. Accordingly, I would like to thank my advisers, colleagues, friends and family alike.

I would like to express my gratitude to my supervisors and advisers Hans-Peter Seidel and Michael Wand for providing an excellent research environment and for their ongoing support throughout the different phases of my research. I highly appreciate their effort, patience, availability, and above all the expertise they passed on to me.

I thank the people shouldering major administrative work which so often accompanies research in progress; especially I want to thank our administrative secretary, Sabine Budde, whose lasting effort helped reducing bureaucracy to a pleasant minimum. Likewise, I thank the helpers and employees of our Information Services and Technology department for their constant fight to maintain an excellent computer infrastructure and fulfill the permanently growing needs over the past years.

Further credit is due to my co-workers for their stimulating ideas and helping comments, and for directing my attention to interesting topics. Foremost I thank my co-authors Silke Jansen and Elmar Eisemann who have broadened my view and enriched the publications with their valuable help and insightful discussions. Furthermore, I am grateful to my former fellow students and colleagues in the Computer Graphics Department of the MPI and in the Statistical Geometry Processing Group of the MMCI, who it was a great pleasure to work with. Thank you for effective suggestions, motivating comments, and help in both my academic and social life.

I also wish to thank all the anonymous reviewers of our publications, for their crucial effort, pushing quality and providing valuable comments; and I wish to thank the many authors of the many software programs providing vital tools necessary for the research in preparation of this thesis. Special thanks I owe all the authors which helped building *XGRT*, a powerful software for efficient processing and rendering of large 3D point clouds – the foundations this work is built on. However, many ideas and concepts cannot prove its virtue without evaluation. Therefore I want to give credit to all providing test data for research purposes. In this place I wish to thank Claus Brenner

and the IKG Hanover for making their city scan data available, and Tobias Leppla and the LKVK Saarland for their scan of the “Ludwigskirche”.

Finally, I am indebted to my family, my parents Dorothea and Gustav as well as my brother Thomas, to my beloved Silke and to all my close friends who have always supported and encouraged me, and showed all the patience necessary in stressful periods.

Many thanks to all of you for accompanying me on my way and for making it a memorable time.

Martin Sunkel

Contents

List of Figures

1 Introduction

I think it would be a good idea.

rumors about his answer to a question:
Mahatma Gandhi (1869 - 1948)

3D scanning technology has matured to a point where very large scale acquisition of high resolution geometry has become feasible. Using mobile LIDAR scanners, point clouds at centimeter resolution of complete countries can be captured at economically viable costs (such as in the well-known projects by companies like Google [ADF+10] or Navteq). Cost efficient approaches such as structure-from-motion reconstruction from community photo collections [ASS+09, FGG+10, GAF+10] complement these efforts.

Having, at some point, accurate 3D models of our entire planet offers enormous opportunities, but it also poses new technical challenges. A key problem is semantic scene understanding: Almost any application beyond simple 3D rendering, such as mobile navigation, maintenance of public infrastructure, or planning for disaster preparedness, requires an understanding of the semantics of acquired geometry, such as finding roads, cars, street lights, entrances to buildings, and the similar. This information is not acquired by any 3D scanner, and human annotation of large scale data is obviously infeasible. Hence, the development of computer-aided techniques for this class of problems has become a very important research topic. From the variety of scene understanding problems two important categories have emerged – correspondence problems and semantic labeling problems, such as object detection.

The benefits of matching algorithms for scene understanding is obvious. They establish geometric and/or semantic correspondences between data, e.g. for parts of a bigger scene. This can be used for a variety of applications: cleanup and precision improvements of scanned data, auto completion and hole filling, meshing, instance highlighting and many more.

There is a large body of work on correspondence estimation for geometric data sets. Early techniques such as the classic iterated closest point (ICP) algorithm for rigid matching [BM92, CM92] are concerned with rigid transformations in order to align two or more objects. More sophisticated techniques such as deformable ICP [ACP03, HTB03, BR07] include more complex local transformations. However, they are based on local optimization and require a user-guided initialization. Furthermore, a variety for global optimization techniques has been proposed. They solve matching problems without pre-alignment [ASP+04, GMGP05, LG05, BBK06, HFG+06, CZ08, HAWG08, ZSCO+08, TBW+09, TBW+11]. Almost all algorithms for global correspondences (without known initialization) start by detecting geometric features that are invariant under the considered correspondence transformation. Consistent correspondences are found via global optimization techniques such as branch and bound, loopy belief propagation, or Hough-transform-based voting. Afterwards, dense correspondences can be estimated using local optimization techniques such as gradient descent (to refine transformation parameters) or region growing (to refine the corresponding area in partial matching scenarios).

However, the most simple but crucial prerequisite of all these approaches is that the instances to be matched must already be given. For bigger scenes that means, we first need to detect objects prior to establishing correspondences. Though, there are techniques which are concerned with finding correspondences within a scene. In fact, symmetry detection algorithms [MGGP06, PMW+08, BBW+09] yield geometric correspondences between re-occurring parts. But they mostly work with static symmetries, that means they do not capture semantics, and therefore they do not allow for bigger variations. Thus, in order to approach scene understanding, it is also important to come up with efficient and robust algorithms for 3D object detection which can also find instances belonging to a object category comprising semantically similar objects.

For the construction of an efficient detector design we need to distinguish between ‘thing’ and ‘stuff’ detectors. Detecting ‘things’ typically means that we want to find well-defined entities such as cars, pedestrians, or win-

dows in a wall, whereas 'stuff' detection more refers to amorphic objects or areas without clear structure, e.g. streets, bushes, walls. There are also object categories which comprise both types. For example detecting plants in a pot requires both, finding a structured entity (the pot) in addition to green 'stuff'. A prominent category of problems, where stuff detection is involved, are segmentation problems. Traditionally they are concerned with structuring objects into different pieces, typically by labeling regions [GG04, GF09, KHS10, HKG11, SvKK+11, vKTS+11]. Although, they can cope with unstructured regions and to some extend incorporate semantic knowledge, it is hard to find a particular object. Most of all, the majority is not applicable to bigger scenes. However, similar to segmentation, object detection can also be regarded as a labeling task – by labeling all pieces of geometry which are semantically similar to a particular object class with a common label [ATC+05, GKF09, VSS12]. That means detection can be understood as a binary labeling (or classification) task.

In the following the term *detection* will be associated with detecting entities comprising well recognizable structures and an object center. There is a variety of algorithms which are concerned with object detection in 2-dimensional images [VJ01, RPA03, DT05, FH05, WZFF06, LLS08, RGKG12]. However, the field of 3D object detection is only little explored with merely few algorithms approaching this problem [ATC+05, GKF09, VSS12]. These methods are typically employed for the setting of classification tasks which excel in detecting objects that can be easily segmented, but they are not suitable to detect smaller repetitive elements such as ornaments or windows in buildings. In general, detection problems for the 3D domain come with different issues than 2D approaches, inhibiting the direct transfer of 2D implementations to 3D geometry. Where object detection on images needs to cope with illumination problems, self-occlusions or scale issues, 3D data such as point clouds come with different challenges: points are not aligned to a regular grid, but they are irregularly sampled and subject to non-Gaussian noise artifacts. Further, there is no clear notion of direct neighbors, which implies additional computational challenges.

Towards 3D scene understanding, it would be a valuable contribution to have general algorithms which do not only detect single instances from a category of semantically similar objects, but also establish semantic correspondences. This problem can be approached naively in a brute force attempt: First an object detection algorithm is applied (e.g. using an implicit shape model [VSS12]), then a matching approach (such as [TBW+11]) consecutively tries to establish correspondences for all relevant regions. A benefit of

this approach is that one could choose those algorithms which perform best for the given task. The weak spot is that on one hand computation times might be high (for each detected instance another matching needs to be performed), and on the other hand, depending on the variety of the object class, it might be hard to find consisting correspondences for all of the shapes.

The main motivation of this work is to find general models which combine both worlds: models for object detection that additionally establish consistent and semantically meaningful correspondences. Further, in order to be applicable on large 3D scenes, we are looking for run-time efficient and scalable implementations. Having such tools allows for a rich number of future applications. It might be only a small step to come up with methods for decomposing large 3D scans into low resolution primitives for mobile applications. Other applications could aim at artifact removal and super-resolution for complex shapes (similar to [BBW+09]). But most desirable can be considered efficient tools capable of fitting (generative) morphable models to large 3D scenes. Such tools perfectly suit not only the before mentioned applications, they can also be used for smoothly filling in scan holes; beyond that they even allow for morphing in complex scenes.

1.1 Problem Statement

We address the problem of detecting object instances (*shapes*) in 3D geometry according to semantic object classes learned from sparse user annotation. Shapes are defined as a set of distinctive parts describing local geometry (*local shape*) as well as the spatial layout (*constellation*) of these parts. From a small number of hand-annotated examples, a statistical part-based object model is derived to retrieve large quantities of further object instances, while encoding object correspondences via the part-based object structure. This thesis is concerned with the construction and analysis of robust statistical models for this problem, as well as with the derivation of efficient and robust inference and detection schemes for such models.

Formally the detection problem is given as follows: Let $\mathcal{S} \subset \mathbb{R}^3$ be a smooth 2-manifold embedded in three-space. Typically $\mathcal{S}$ is represented by a sampled approximation (*point cloud*) $S = \{\mathbf{s}_1, ..., \mathbf{s}_n\}, \mathbf{s}_i \in \mathbb{R}^3$.

Given a shape model with k parts, each part i encodes a relative spatial arrangement $\mathbf{x}_i$ and a local shape description $\mathbf{d}_i$. The individual parts can be subsumed into an overall local shape D and an overall spatial layout X. The

shape model is then defined by parameters $\theta = (\theta_D, \theta_X)$ of (parametrized) probabilistic models for local shape and layout, and the goal is to find reasonable assignments $H = (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_k)$ for k parts to points in S, where $\mathbf{h}_i \in S$ denotes the position of part i in the point cloud.

Hence, the objective is to find assignments for H which maximize the model likelihood $p(D, X, H|\theta)$, and the detection problem itself can be formulated as a maximum a posteriori hypothesis search over the joint posterior distribution $p(H|D, X, \theta)$ of H.

Implementations for the model and the posterior search are subject of this thesis and can be found in chapters 6 and 7.

1.2 Organization and Contributions

Chapter 2 provides an overview over previous work which is most related to the topic of this thesis. Chapters 3,4,5 give an introduction to, and definitions of the theoretical concepts used in this thesis: Chapter 3 introduces basic definitions and notations from probability theory, in Chapter 4 we will see concepts for probabilistic graphical models, and Chapter 5 shows how such models can be computed. Further, we will highlight how these concepts are related to each other, since this is essential for a deeper understanding of this work. To that end, the idea for simultaneous object detection with message passing algorithms is explained in Section 5.1.4, and in Section 5.2.3 we find the derivation for an approximate inference scheme which can be used to compute maximal likelihoods for fully-connected parts models, such as the one defined in Chapter 7.

The main contributions of this thesis can be found in Chapters 6 and 7. There, we will see how semantic object correspondences can be encoded using graphical models and how such models can be trained with a minimal amount of user interaction. The chapters derive two different models which allow for efficient object detection in large point clouds.

Chapter 6 presents a statistical Markov chain approach and discusses the motivations for using a chain topology model. Besides introducing the model (which relies on geometry with a predominant orientation) we find a detection scheme that contributes to a drastic reduction of the original search space. The model as well as the detection performance are evaluated in a number of experiments.

Chapter 7 depicts a general part-based model for the detection problem. We introduce a statistical graphical model which correlates all object parts with respect to local shape and constellation. Unlike the Markov chain model used in Chapter 6 it is fully rotational invariant. Since optimal assignments of fully connected graphical models cannot be exactly computed in feasible time, there is given an approximate but robust and efficient inference scheme – an approach which is novel in literature. Similar to Chapter 6 there is shown how the search space can be depleted for such general models. Further, an approach is presented how the model can be extended to a hierarchical version in order to efficiently model and detect more complex objects. A number of quantitative and qualitative experiments evaluate model capabilities and the detection performance.

The last chapter (Chapter 8) gives a final general discussion about different models for the detection scenario of this thesis. It will show strengths and limitations, and depict approaches how to proceed. Finally, this work concludes with a prospect to future research.

List of Contributions:

- This thesis comprises (an implementation of) two novel statistical part-based models for interactive and robust object detection in point clouds: a Markov chain parts model and a rotational invariant, fully-correlated parts model.
- The detection schemes are designed such that found object instances automatically yield semantic part correspondences – even for object categories with high variability.
- The implementations comprise efficient pruning techniques that drastically lower the search space such that detection can be performed at interactive run times even for large 3D scenes.
- The inference algorithms are able to detect all object instances simultaneously in unprocessed point clouds.
- The implementations come with a scheme, how to define and refine semantic object classes with minimal user input.
- Solutions for the correlated parts model cannot be computed exactly. Therefore, the implementation comprises a novel, efficient approximate inference scheme.

- A hierarchical model extension is presented: a tool to create hierarchical models for detection of more complex object categories.
- The correlated parts model is fully rotational invariant, hence it can be used for detection tasks in arbitrary geometry.

1.3 List of Publications

The following research papers have been originated by or written in collaboration with the author during the preparation phase of this thesis. They have been published in the proceedings of international, peer-reviewed, leading venues in the area of computer graphics. The list is split in two parts:

The first part comprises the publications contributing to this thesis. Main text passages of the papers from this part have been utilized without being tagged individually. The coauthors have permitted the use of these passages. The same applies for figures.

The second part contains work which has been published in the proceedings of international, peer-reviewed, major conferences as well, but it is not part of this thesis.

Publications contributing to the thesis:

M. Sunkel, S. Jansen, M. Wand, E. Eisemann, H.-P. Seidel: "*Learning Line Features in 3D Geometry*", Computer Graphics Forum (Proc. Eurographics) (2011)

M. Sunkel, S. Jansen, M. Wand, H.-P. Seidel: "*A Correlated Parts Model for Object Detection in Large 3D Scans*", Computer Graphics Forum (Proc. Eurographics) (2013)

Additional publications:

K. Scherbaum, M. Sunkel, H.-P. Seidel, V. Blanz: "*Prediction of Individual Non-linear Aging Trajectories of Faces*", Computer Graphics Forum (Proc. Eurographics) (2007)

M. Sunkel, B. Rosenhahn, H.-P. Seidel: "*Silhouette Based Generic Model*

Adaptation for Marker-Less Motion Capturing", ICCV, Workshop on Human Motion (2007)

N. Hasler, C. Stoll, M. Sunkel, H.-P. Seidel: "*A Statistical Model of Human Pose and Body Shape*", Computer Graphics Forum (Proc. Eurographics) (2009)

2 Related Work

The significant problems we face
cannot be solved at the same level of thinking
we were at when we created them.

Albert Einstein (1879 - 1955)

In this chapter we briefly review prior concepts and ideas which helped in design and formation of the work presented in this thesis. Historically, object detection was first explored in image domain. For that reason the main concepts for part-based detection models evolved from there.

2.1 2D Scene Understanding

The problem studied in this thesis is closely related to "scene understanding" approaches for 2D images. While first object detection schemes aimed at rigid objects under changing viewpoints, current research strives to detect classes of non-rigid or deformable objects. Besides the development of sophisticated local shape descriptors [DT05], and bags of descriptors [LSP06, GD05] prescribing (and evaluating) part appearance, two main detection strategies have emerged: On one hand Markov random fields are used to model local appearance and consistency between neighboring pixels [KH03, HZRCP04]. Such techniques mostly aim at labeling amorphic categories such as "vegetation" or "buildings". On the other hand, hampered by misleading information due to clutter and occlusion, the focus has shifted from holistic approaches to

part-based models describing objects by individual parts linked with structural information. Such models are used to detect constellations of distinctive parts [RPA03, FH05, LLS04, FMR08], excelling in recognizing individual objects, such as cars or bikes. The following provides an overview of concepts for part-based 2D object detection which are most related to the approaches described in this thesis. A detailed exploration of "scene understanding" approaches for images in general is not subject of the overview.

Part-based models describe objects by a collection of parts embedded in a graph structure in order to encode contextual information. The single parts capture local appearance properties at a certain position which is dependent on neighboring parts. The models developed in this thesis also follow this paradigm. There is a number of different approaches which can be grouped mainly by the different graph topologies used:

Voting Schemes: Voting schemes typically represent star-like graph structures. Single detected parts independently vote for a designated object center, thus generating a discrete distribution of votes in the image. The local maxima of this distribution then represent detected object centers. There is a number of generalized Hough transform methods [Bal87, GL09, RGKG12] implementing such voting schemes. A special case of the generalized Hough transform is the implicit shape model of Leibe *et al.* [LLS04, LLS08]. Essentially, it is a codebook of feature point descriptors that are most discriminative for a given object class. After the codebook is created, each entry is assigned a set of offsets with respect to the corresponding object centroids. The offsets are learned from training data. At run-time, the interest point descriptors in the query image are matched against the codebook and the matches cast probabilistic votes about possible positions of an object center in the image. Another popular tool for tackling the problem of intra-category diversity for object detection is Felzenszwalb *et al.*'s so-called deformable parts model [FMR08, FGMR10]. It is based on HOG descriptors [DT05] for local appearance descriptions. The idea behind deformable parts is to represent an object model using a single lower-resolution 'root' template, and a set of spatially flexible smaller high-resolution 'part' templates. Each part captures local appearance properties of an object, and deformations are characterized by the part positions relative to the root center. Additionally, an object class can be partitioned into subcategories. There, the idea is to segregate object instances into disjoint groups, each with a simple attribute such as frontal versus profile view, and learning a separate model per group.

Pictorial Structures: The basic technique most similar to the approach described in Chapter 6 is 'Pictorial Structures' [FE73, FH05]. In Felzenszwalb and Huttenlocher's implementation they use a tree-structured MRF of parts to describe objects by decomposition into local appearances and spatial relations. Local appearances as well as spatial relations are estimated with single independent Gaussian distributions. The model is applied to two different scenarios: Face detection and pose estimation for an articulated body model, both for frontal views only. While conceptually related, our setting differs in two important aspects: First, we are dealing with general 2-manifolds in an irregularly sampled representation (point clouds) as inference domain, which provides more degrees of freedom and is more difficult to handle than regular pixel grids. Secondly, unlike in image understanding applications, we are aiming at finding many instances simultaneously rather than only the best explanation for an image given just very few training exemplars.

Constellation Model: The method introduced in Chapter 7 is related to the idea of constellation models in image understanding as described in Fergus *et al.* [RPA03]. They propose a part-based approach which models the pairwise spatial dependencies between all parts. They train a generative object class model. That means the parts are modeled by independent Gaussian distributions for part appearances and overall part alignment. However, the model does not consider pairwise relations of part appearances. Since it is infeasible to efficiently and exhaustively explore the distribution of such models, the state space is limited to a set of up to 30 precomputed features which can be used for parts. For detection a combinatorial deterministic (greedy) best-first search (A* search [YC98]) is utilized. Although Bergtholdt *et al.* [BKSS10] further improved the detection scheme by introducing more sophisticated branch-and-bound heuristics, they still rely on a small number of pre-defined features.

Correlating Appearance of Parts: The utility of appearance correlations has been shown in the context of bags-of-words models: Wang *et al.* [WZFF06] demonstrate that explicitly modeling the inter-dependencies of local patches yields more discriminative models. In the context of part-based models, pairwise geometric relations of lines have also proven to be helpful for recognition [LHS07, SGS09]. Leordeanu *et al.* [LHS07] use a set of angles and distances to represent the geometric relations between parts. Stark *et al.* [SGS09] enrich constellation models by pairwise symmetry relations between contour segments. In the 3D domain, these correlation can be

expected to be even more pronounced, as variability due to lighting, texture, and occlusion is not present.

2.2 3D Scene Understanding

In prior work 3D scene understanding mostly deals with 2-manifolds in different representations. We can distinguish between sampled representations such as point clouds, mesh representations which provide additional topology information, and parametrized continuous manifolds. Such data inherit tasks similar to 2D scene understanding, but with different challenges. For example in case of point clouds captured by LIDAR range scanners the most prominent issues are not only the lack of topological information but rather the irregular sampling and strong non-Gaussian noise artifacts. Therefore a simple transfer of 2D implementations to 3D geometry is not possible.

For disambiguation the following will give a brief overview of typical 3D scene understanding problems in general and show work which is related to the detection scenario of this thesis.

Matching: The methods described in this thesis can not only be viewed from the object detection perspective, but also as means to establish correspondences among semantically similar geometric features. There is a large body of work on correspondence estimation for geometric data sets. Early techniques such as the classic iterated closest point (ICP) algorithm for rigid matching [BM92, CM92] as well as the later deformable ICP techniques [ACP03, HTB03, BBK06, BR07] are based on local optimization and require user-guided initialization.

More recently, several global optimization techniques have been proposed that solve the problem without pre-alignment [ASP+04, GMGP05, LG05, HFG+06, CZ08, HAWG08, ZSCO+08, TBW+09]. An interesting variant is partial symmetry detection, where a single shape is decomposed into building blocks. A common approach is transformation voting, which detects symmetries under a fixed group of transformations such as similarity transformations [MGP06, PSG+06, PMW+08]. The use of a location independent voting space can, however, lead to problems if many symmetric parts are present simultaneously. Matching graphs of surface features [BBW+08] has been proposed to avoid this problem.

Learning Correspondences: Learning of feature matching is rare in geometry processing. Schoelkopf et al. [SSB05] apply regression techniques from machine learning to estimate correspondences, but they do not learn from user input. In follow-up work, Steinke et al. [SSB06] identify invariant feature regions in a morphable face model [BV99] by an oriented principal component analysis (PCA). Their method requires some amount of training data and outputs a weighting function that describes the invariance of local descriptors.

Retrieval: A large amount of work has been devoted to the recognition of isolated objects (see for example [MSS+06], and [DGD+11] for a survey), but not detecting instances within a larger scene. Most methods are primarily based on bags-of words approaches [LG07, BBGO11]. Furthermore, fixed models of deformations have been studied in order to match template shapes to data (typically isolated objects) under isometry or different types of elastic deformation (see [vKZHCO11] for a survey). In contrast, our methods aim at learning the variability from training data and detect occurrences in larger scenes.

Urban Scenes: In recent years major efforts have been made in acquiring large scale data of urban environments. For example from Google street view project [ADF+10] there is tons of data available. However, most data is collected as 2D image sequences, which first off all poses the demand for 3D scene reconstruction. Structure-from-motion techniques [ASS+09, FGG+10, GAF+10] tackle that problem. Early approaches for large scale reconstruction of the street view data included simple geometric constraints, namely that most street scenes can be roughly reconstructed by piecewise planar 3D models [MK09]. Based on such an approach Google showed that reconstruction quality can be improved by enriching image data with 3D laser range scans [ADF+10] of the scene. However most improvements augmenting visual appearance are on texture level, e.g. fusing multi-modal data into high resolution textures. Yet multi-modal data can also be used to reconstruct more detailed piecewise planar 3D models. Pylvänäinen *et al.* [PBK+12] show how to combine LIDAR range scans and panoramic image sequences into multi-view stereo reconstructions in order to reconstruct a 3D model of a bigger city scene.

Although the 3D reconstruction of isolated buildings in a scene yields obvious information about their whereabouts, none of these approaches include

any form of semantic correspondences. There are techniques which exploit fixed semantics typical for city scenes. Mesolongitis *et al.* [MS12] propose an approach for window detections in point clouds of urban scenes However, the main idea of their approach is to first segment the scenes in order to find facades and potential locations for windows. Then they hard-code prior knowledge: For window detection that is grid-like assumptions for the global alignment structure of all instances.

More general semantic information can also be extracted based on geometric information only. For example Kerber *et al.* [KBWS12] are concerned with symmetry detection for large city scans, i.e. the retrieval of reoccurring geometry in bigger blocks, such as windows in buildings. However, such information can only be computed for rigidly matching instances, not for bigger semantic object categories. Further, it is not clear beforehand which symmetry information can be extracted. Last but not least, the quality of the results depends on the quality of the scan. A broader overview over symmetry detection in 3D geometry is given in a state of the art report of Mitra *et al.* [MPWC12].

Labeling and Classification: Object detection can be interpreted as a labeling and classification problem where parts of the geometry are labeled according to geometric or semantic classes, which is a classical segmentation task. Segmentation and classification of geometric objects according to semantic categories is often tackled using conditional random fields [ATC+05, KHS10, ZLZ+10]. However, the application to large scenes is often limited, because all labels have to be estimated in a complex global optimization problem. Further, such approaches are mostly limited to data with reliable topology information, i.e. clean triangle meshes.

Kalogerakis et al. [KHS10] explore learning user segmentation by example. They use a Markov random field with learned local descriptors at nodes and learned edge compatibility costs and perform MAP estimation using iterated graph-cuts. This approach is not applicable to point clouds, it does not yield correspondences and will find only one global solution for a prior isolated object.

Anguelov *et al.* [ATC+05] address the problem of segmenting large scale data obtained from 3D scans into few object classes. Their segmentation framework is based on pairwise conditional random fields. Each variable is associated to a point in the scan, edges are installed to nearest neighbors. The task is to assign one of the class labels to each of the variables. Segmentation

is done by iteratively performing binary classification into the single classes using a graph-cut algorithm. They incorporate a set of diverse descriptors and enforce the preference that adjacent scan points have the same label. Obviously such an approach does not yield any form of correspondences.

A similar technique is proposed by [MVH09]. They show how a conditional random field can be efficiently extended to higher-order cliques and present a fast approximation scheme for online classification, onboard a mobile vehicle for environment modeling.

An approach which segments point clouds into instances of single object classes is presented by [PMD08]. Their technique relies on the assumption that an object instance can be well separated from the point cloud. They use a pure descriptor-based, example driven approach which looks for nearest neighbor matches in a trained data-base in a bottom-up and subsequent top-down manner.

Moosmann *et al.* [MS11] put a special focus on difficult, unstructured outdoor scenarios with object classes ranging from cars over trees to buildings. They do not not assume that a perfect segmentation is possible. They provide a fully automatic hierarchical training method to cope with typical outdoor scenarios, thus excluding user-defined semantics. The approach is similar to bag-of-words methods. It can be outlined as follows: The scene is hierarchically decomposed into previously segmented regions. For each hierarchical segment a set of features is calculated. The features are clustered and finally different object classes are defined using selected clusters.

The method of Stamos *et al.* [SHZF12] aims at a very special scenario. They aim at fast low-complexity detection and classification algorithms for online data processing during the acquisition process. They classify points into 5 distinct classes (vegetation, vertical, horizontal, car and curb regions) and determine the ground level without requiring training or parameter estimation. Key ingredient of the method is the use of heuristic summary statistics which reduce the dimensionality of the data. Although this is a fast technique for rough segmentation, it questionable if it can be used to classify arbitrary objects.

The methods examined so far are more or less based on holistic approaches. Also some part-based methods which are widely used in 2D computer vision have been adapted to recognizing 3D meshes [TCF09, KPW+10]. They represent the surface of an object with a part-based model, i.e. a set of local part descriptors together with their spatial configuration. The part-based

representation is robust against partial occlusions and can handle moderate deformations of object geometry. Toldo *et al.* [TCF09] adapted the bag-of-features method from 2D images to also recognize 3D shapes. An object is modeled as a histogram of 3D visual 'word' appearances in terms of its subparts. Based on that descriptor they describe an effective method for hierarchical 3D object segmentation on triangle meshes. Knopp *et al.* [KPW+10] also use an extended version of the SURF feature descriptor in the context of 3D shapes, but they enforce spatial consistency as well. They present an implicit shape model based on 3D surf feature descriptors and perform shape retrieval and object class recognition for single 3D models and on classes with rather low intra-class variability.

Another alternative is segmenting shapes using clustered geometry. A well-received work for object segmentation and categorization of 3D point clouds in large scenes is given by Golovinskiy *et al.* [GKF09]. Their algorithmic pipeline consists of 4 main steps: First background is removed. In case of urban geometry that is removing planar data such as the ground. The remaining data is segmented and clustered into sets of connected components of nearest neighbor graphs. The clusters form hypotheses for potential objects. Then, for each component a set of global features is generated (e.g. height, volume, spin image). In the last step, the vector of all features is classified using a support vector machine. Velizhev *et al.* [VSS12] extend this model by replacing the last step with an implicit shape model (i.e. a part-based representation voting for the object center) similar to the one of Knopp *et al.* [KPW+10]. They show that such a part-based classification approach yields better detections especially for objects which exhibit considerable intra-class variability. Thus, they not only yield a segmentation for detected objects but also a clear object center. However, these methods mainly suit the problem to extract shapes for which a first hypothesis is easy to compute. For example cars and lamp posts stick out of the ground and can therefore be well separated. It would not be possible to find smaller shapes within a bigger structure such as windows in a building.

In summary we can see that the objective of prior large scale approaches is either to segment scenes directly, using global conditional random fields, or to first cut geometry into smaller pieces, and classify these afterwards. Yet then they rely on easy data separability and robust segmentation (for background removal). There is no (part-based) method which directly approaches the detection problem without major pre-processing efforts that reduce a 3D scene to a number of cut-out hypotheses that need to be evaluated separately.

In fact, the method of Velizhev *et al.* [VSS12] comes closest to the detection problem defined in this thesis. In addition to a segmentation it yields an object center. Also, the Hough voting could be used to reconstruct part correspondences for single object instances. Unfortunately the method is only suited for detection scenarios with easy and well separable scene objects. Further it relies on automatic feature extraction and does not allow for semantic correspondences defined by the user. The flexibility of object classes is therefore limited to geometric similarities which can be automatically extracted.

3 Background

Do not worry about your difficulties in Mathematics.
I can assure you, mine are still greater.

Albert Einstein (1879 - 1955)

Our understanding of the world is and will always be limited by our observations. Accordingly we need models which can deal with our imperfect knowledge of the world. Statistical models are machine learning tools which can cope with example-based knowledge and explicitly take into account uncertainty.

This chapter briefly introduces underlying concepts and notations used in this thesis. We do not give attention to all details, the interested reader is referred to a wide range of textbooks such as [Bis06, KF09]. Throughout this thesis we try to keep notations in a simplified and more intuitive form in order to support the general readability and understanding, rather that cherishing absolute soundness with mathematical formalism.

3.1 Probability

Probability or *likelihood* is a measure for estimation of how likely something will happen. Probability theory can be used to describe underlying mechanisms in complex systems. In the following we will briefly introduce the most basic definitions.

Random Variable and Probability: A *random variable* X is a variable whose value is subject to the outcome of a given event. It can be assigned *values* (or *states*) $x \in \Omega$ from the set of all possible outcomes (*state space*). In the scope of this thesis we will only consider discrete random variables with a finite state space $\Omega = \{x_i\}_{i=1,\dots,k}$.

For a discrete random variable X, each state $x \in \Omega$ is associated with a probability $p(X = x) \in [0, 1]$, which, for the sake of simplicity, we will also denote by $p(x)$. Values between 0 and 1 represent the degree of certainty for variable X being in state x, with $p(x) = p(X = x) = 1$ expressing the maximal certainty. The set of all probabilities $p(X) := \{p(x)|x \in \Omega\}$ then defines the *probability distribution* of X. Every probability distribution must be normalized, i.e. $\sum_{x\in\Omega} p(x) = 1$.

Joint Probability: The *joint probability* measures the likelihood for a number of potentially interacting random variables (combined into one random vector). For a multivariate random variable $X = (X_1, \dots, X_n)$ of n random variables ranging over values $x_i \in \Omega_{X_i}$ the (multivariate) joint probability for an n-dimensional event $(X_1 = x_1, \dots, X_n = x_n)$ is given by $p(\mathbf{x}) = p(x_1, \dots, x_n) = p(X_1 = x_1, \dots, X_n = x_n) \in [0, 1]$. The *joint probability distribution* for X is denoted by $p(X)$.

Marginal Distribution: Given a joint probability distribution $p(X)$ with $X = (X_1, \dots, X_n)$, the marginal distribution $p(X_i)$ defines the distribution of a single variable X_i. It can be computed by

$$p(X_i) = \sum_{X_1} \cdots \sum_{X_{i-1}} \sum_{X_{i+1}} \cdots \sum_{X_n} p(X_1, \dots, X_n),$$

i.e. by summing over all states of all random variables except for X_i. Similarly the marginal distribution for a subset of variables $S \subset X$ with $S = X \backslash X_i$ (i.e. all variables except for X_i) can be computed by summing over all states of the random variable X_i only:

$$p(S) = \sum_{X_i} p(X_1, \dots, X_n)$$

Conditional Probability Distribution: If for a joint probability distribution $p(X)$ with $X = (X_1, \dots, X_n)$ the state of one random variable is already

known ($X_i = x_i$), then the joint probability distribution for the remaining random variables can be conditioned:

$$p(X_1 \dots X_{i-1}, X_{i+1} \dots X_n | X_i = x_i) = \frac{p(X_1 \dots X_n)}{p(x_i)} \tag{3.1}$$

Bayes' Theorem: As we will see later, Bayes law is of major importance for deriving inference schemes for probabilistic reasoning. Intuitively, for two multivariate random variables X and Y it helps reformulating probabilistic relationships by translating between $p(Y|X)$ and $p(X|Y)$.

From definition of conditional probability, Bayes' law can easily be derived:

$$p(X|Y) = \frac{p(Y|X) \cdot p(X)}{p(Y)}$$

3.2 Gaussian Distribution

Gaussian or *normal distributions* are parametrized distribution models and have many convenient properties. Random events with unknown distributions can often be assumed to be normally distributed. Although this is a dangerous assumption, the central limit theorem [Kal02] states that (under certain conditions) the mean observation of a sufficiently large number of independent random variables tends to be normally distributed: Many common measures or phenomena such as human body height, examination grades or noise effects roughly follow normal distributions. We can often (though not always) regard a single event as the average of a number of effects. In a more general sense the central limit theorem states, that the more a measurement behaves like the sum or average of independent random variables, the more it tends to be normally distributed. This justifies the use of the normal distribution as an approximation to stand in for the outcomes of random events.

In Chapters 6 and 7 we will derive statistical models for arbitrary, user-defined object classes. There we need to utilize parametrized distribution models in order to represent the user input. That means the underlying distribution is unknown, but we assume that it behaves like the average of many independent unmeasured effects. Therefore, we will estimate a Gaussian model from the set of exemplars.

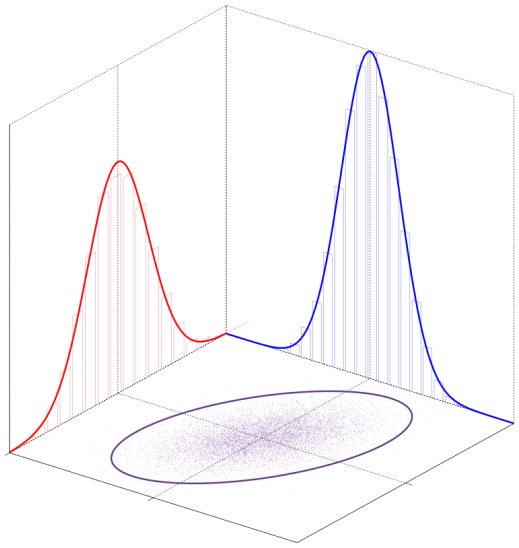

Figure 3.1: *Many samples from a 2D multivariate normal distribution, shown along with the 3-sigma ellipse, the two marginal distributions, and the two 1D histograms.*

Be $X = (X_1, \ldots, X_n)$ an n-dimensional multivariate random variable. We use the notation $X \sim \mathcal{N}\{\mu, \Sigma\}$ to indicate that X is normally distributed where μ is an n-dimensional mean vector, representing the center of the distribution, and Σ is an $(n \times n)$-dimensional covariance matrix specifying linear correlations between pairs of single random variables. Then, for any assignment $\mathbf{x} \in X$ follows:

$$p(\mathbf{x}) = (2\pi)^{-n/2} \cdot |\Sigma|^{-\frac{1}{2}} \cdot \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right) \tag{3.2}$$

In the following be $X = (X_1, X_2)$ a multivariate random variable which can be decomposed into two random variables of lower dimensionality. Be

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$$

mean and covariance matrix of a normal distribution for X.

Marginal distribution: The marginal distribution over a subset $X_1 \subset X$ is obtained by dropping the irrelevant variables from the mean vector

and the covariance matrix:

$$X_1 \sim \mathcal{N}(\mu_1, \Sigma_{11})$$

Conditional distribution: Given the marginal distributions of X_1 and X_2, the normal distribution of X_1 conditional on $X_2 = \mathbf{x_2}$ is given by

$$\tilde{X}_1 := (X_1 | X_2 = \mathbf{x_2}) \sim \mathcal{N}(\tilde{\mu_1}, \tilde{\Sigma}_{11})$$

with

$$\tilde{\mu}_1 = \mu_\mathbf{1} + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x_2} - \mu_2)$$

and

$$\tilde{\Sigma}_{11} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}.$$

The matrix $\tilde{\Sigma}_{11}$ is called the *Schur complement* (see [Eat83]) of Σ_{22} in Σ.

An interesting fact is, that although knowing the state of X_2 alters the covariance, $\tilde{\Sigma}_{11}$ does not depend on the specific value of $\mathbf{x_2}$, whereas the mean is shifted by $\Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x_2} - \mu_2)$. Furthermore, the random vectors X_2 and $\tilde{X}_1$ are statistically independent. This fact is used to implement an efficient inference scheme in Chapter 7.

Estimating μ and Σ: If neither the mean nor the covariance matrix of the normal distribution of a random vector $S = (X_1, \ldots, X_n)$ is known, it can be estimated from a set of m independent observations $\mathbf{x}_i \in \mathbb{R}^n$. The mean μ can be estimated by computing the observation mean

$$\mu = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i.$$

An estimation for the $(n \times n)$-dimensional covariance matrix is given by

$$\Sigma = \frac{1}{m-1}\sum_{i=1}^{m}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T.$$

3.3 Paradigm of Probabilistic Reasoning

Constructing a probabilistic model $p(X_1, \ldots, X_n)$ for all relevant interacting variables $X_1, \ldots, X_n$ of a given problem is the vital task probabilistic reasoning is concerned with. However, sometimes there is only information on

marginals or conditional probabilities available. The basic idea of reasoning (*inference*) is to infer joint probabilities based on conditional relations. Given evidence which sets some variables into known states, probabilities of interest can be subsequently computed using the conditional dependencies.

For example, given 4 random variables A, B, C, D with an unknown underlying joint distribution $p(A, B, C, D)$. With help of the rules for conditional probability (Equation 3.1) the probability for $p(A = a, B = b, C = c, D = d)$ can be computed by

$$\begin{aligned} p(a, b, c, d) &= p(a) \cdot p(b, c, d|a) && (3.3) \\ &= p(a) \cdot P(b|a) \cdot p(c, d|a, b) && (3.4) \\ &= p(a) \cdot p(b|a) \cdot p(c|a, b) \cdot p(d|a, b, c). && (3.5) \end{aligned}$$

For reasons of simplicity, in the following we will set aside correct notation for conditional probability distributions and present formulas with capital letters for random variables although evidence might be given.

3.4 Bayesian Inference

There are numerous ways to describe probabilistic models for a given problem. A popular family of problems is concerned with relating observed data $\mathcal{D}$ to parameters Θ of an underlying *model* (a data generating mechanism). The main goal is to find those model parameters Θ which are most likely, given some data $\mathcal{D}$. Thus, we are interested in the *posterior* distribution $p(\Theta|\mathcal{D})$ of the parameters Θ for observed data $\mathcal{D}$. Bayes' law now indicates the inference scheme:

$$p(\Theta|\mathcal{D}) = \frac{p(\mathcal{D}|\Theta)p(\Theta)}{p(\mathcal{D})}$$

This shows how, from assumed *prior* knowledge of $p(\Theta)$ (about which parameter values are appropriate), and from *model likelihood* $p(\mathcal{D}|\Theta)$, the posterior distribution is inferred. The likelihood is defined by the model and represents the probability of observing the data, given the model parameters are known. The term $p(\mathcal{D})$ is called *model evidence*. Since $p(\mathcal{D}) = \sum_\Theta p(\mathcal{D}, \Theta) = \sum_\Theta p(\mathcal{D}|\Theta)p(\Theta)$, it is also sometimes called the *marginal likelihood*. Regardless which value is assigned to Θ, $p(\mathcal{D})$ is constant. This means that this factor does not enter into determining the relative probabilities of the posterior and can be omitted during optimization.

Finally, we want to find the *most probable a posteriori (MAP)* setting which maximizes the posterior:

$$\Theta_* = \arg\max_{\Theta} \{\ p(\Theta|\mathcal{D})\ \} \propto \arg\max_{\Theta} \{\ p(\mathcal{D}|\Theta) \cdot p(\Theta)\ \}$$

Sometimes the prior $p(\Theta)$ is assumed constant. Then the MAP solution is equivalent to the *maximum likelihood*, namely the Θ maximizing the likelihood $p(\mathcal{D}|\Theta)$ of the model generating the observed data.

3.5 Statistical Learning

Some words on learning statistical models: In the context of this thesis, *learning* a model actually means estimating the multivariate distribution for a high-dimensional probabilistic model given a set of exemplars (*training data*). Those will be Gaussian distributions which are defined by a mean μ and a covariance matrix Σ. The covariance matrix describes the linear correlations between all pairs of random variables. In case of an n-dimensional multivariate distribution, it needs at least n exemplars to estimate a non-degenerated covariance matrix Σ. However, if n is large, there is often not enough training data available. In those cases, the estimation is regularized by adding a user-defined multiple of the identity matrix λI to the original estimation of Σ, thus lowering rank deficiency [HJ85]. However, finding robust values for λ is subject to manual adjustments.

4 Graphical Models

If your experiment needs statistics,
you ought to have done a better experiment.

Ernest Rutherford (1871 - 1937)

Graphical models [Bis06] provide an intuitive but powerful representation for joint probability distributions. They describe the probabilistic relationship of random variables using graphs. The nodes of the graph represent random variables. The edges describe conditional dependencies between the variables.

Two nodes which are not directly connected are conditionally independent. Figure 4.1 shows an example for a graphical model with 3 random variables X, Y and Z. The variable X is conditionally independent of Y given Z. That

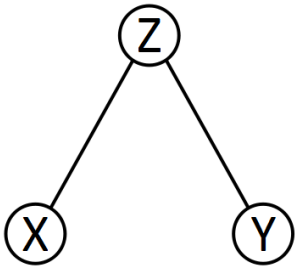

Figure 4.1: *A simple graphical model with* 3 *random variables. X and Y become conditionally independent, given Z is known.*

means, Y contains no additional information about X as soon as Z is known. Similarly, given Z, knowing X does not tell anything about Y. Technically, that means $p(X, Y|Z) = p(X|Z) \cdot p(Y|Z)$ and the joint probability can be inferred to

$$p(X, Y, Z) = p(X, Y|Z) \cdot p(Z) = p(X|Z) \cdot p(Y|Z) \cdot P(Z).$$

Two main forms for graphical models have emerged [Moo08, Bis06, KF09]: *Bayesian Networks*, which are defined on directed graphs and *Markov Random Fields* which come along with undirected graphs.

4.1 Bayesian Networks

Although this thesis is mostly concerned with undirected graphical models, we will briefly review the fundamentals of Bayesian networks.

A *Belief Network* or *Bayesian Network (BN)* represents the factorization of a multivariate distribution into conditional probabilities of variables dependent on *parental variables*. It is represented by a directed, acyclic graph. The nodes of the graph denote the random variables, and the edges describe conditional dependencies between the variables. For random variables $X_1, \ldots, X_n$ a BN defines a distribution of the form

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i|pa(X_i)),$$

where $pa(X_i)$ represents all parental variables of random variable X_i. This is captured by the graph structure – with arrows from parent variables to child variable. Variables without any parent-child connection are independent of each other. That also means, if there is no parent of the ith node, the ith factor is defined by $p(X_i)$.

Figure 4.2 shows a simple BN with variables $X_1, \ldots, X_4$. The variables X_2 and X_4 are independent variables. X_3 is dependent on both X_2 and X_4, whereas X_1 is only dependent on X_2. Then the joint probability distribution is given by

$$p(X_1, \ldots, X_4) = p(X_1|X_2) \cdot p(X_2) \cdot p(X_3|X_2, X_4) \cdot p(X_4).$$

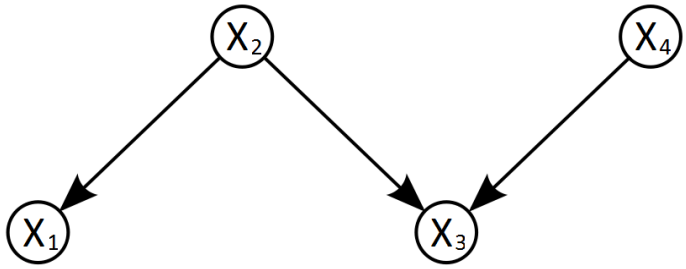

Figure 4.2: *A simple example for a Bayesian Network with variables $X_1, \ldots, X_4$. X_2 and X_4 are independent, X_3 is dependent on X_2 and X_4, whereas X_1 is only dependent on X_2.*

4.2 Markov Random Fields

Although Bayesian Networks are natural for representing *causal* relations (as defined by parental nodes) they are not capable of capturing all possible relations among variables. BNs lack the ability to express 'inter-dependencies' between random variables due to their definition based on directed, acyclic graphs. If those dependencies are of interest Markov Random Fields can be employed.

A *Markov Random Field (MRF)* (sometimes also referred as *Markov Network*) is a set of conditional distributions, one for each random variable. Given an undirected graph G and associated random variables $X_1 \ldots X_n$, the set of distributions $p(X_i|ne(X_i)) > 0$ is defined to form a MRF if

$$p(X_i|X_j, j \neq i) = p(X_i|X_j, j \in ne(i))$$

with $ne(X_i)$ representing all neighbors of variable X_i in graph G. Figure 4.3 shows a simple example for a chain graph MRF.

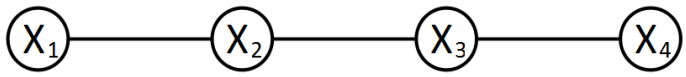

Figure 4.3: *A simple example for a chain graph MRF. This is the analog with undirected edges for the example BN shown in Figure 4.2. However, the dependency structure is different. Every connected pair of variables is dependent on each other.*

Given such a set of conditional distributions, in general there is no direct way to compute the joint probability distribution $p(X_1, \ldots, X_n)$ [Ibe08]. However, according to the *Hammerlsey-Clifford theorem* [HC71, HGW90], it is guaranteed that the probability distribution will factorize into a product of positive *potential functions* for all maximal cliques of G.

4.2.1 Factorized Formulation

A *clique* C of an undirected graph G is a subset of nodes such that every two nodes in the subset are connected in G. The clique C is called *maximal* if no more nodes of G can be added to it. That means C is not a strict subset of any other clique in G.

A function $\psi(X)$ is called a *potential function* of the random variable X if $\psi(X)$ is strictly positive ($\psi(x) > 0$, for all $x \in X$). For a set of random variables $X_1, \ldots, X_n$ the function $\psi(X_1, \ldots, X_n) > 0$ defines a joint potential function. Further, a potential function $\psi(X)$ is also a probability distribution if it satisfies $\sum_X \psi(X) = 1$.

With these definitions there is an alternative formulation for Markov Random Fields: For a graph G with maximal cliques C and a set of variables $S = \{X_1, \ldots, X_n\}$ associated with G, the Markov Random Field can be defined by the normalized product of potential functions $\psi_c(S_c)$ with clique variables $S_c \subset S$ for all maximal cliques C:

$$p(X_1, \ldots, X_n) = \frac{1}{Z} \prod_{c \in C} \psi_c(S_c) \tag{4.1}$$

The constant Z ensures that the distribution is normalized:

$$Z = \sum_{X_1, \ldots, X_n} \prod_{c \in C} \psi_c(S_c)$$

The chain example in Figure 4.3 then factorizes into a product of pairwise potential functions – one potential for each edge.

$$p(X_1, X_2, X_3, X_4) = \frac{1}{Z} \cdot \psi_1(X_1, X_2)\psi_2(X_2, X_3)\psi_3(X_3, X_4)$$

Remark: The Hammerlsey-Clifford theorem also ensures that this alternative, factorized definition for a Markov Random Field meets the conditions of

a *Gibbs Random Field*, i.e. the distribution is strictly positive [Li95]. Thus, the joint distribution can also be given in negative log-likelihood form

$$p(X_1, \ldots, X_n) = \frac{1}{Z} \exp\left(-\sum_{c \in C} E_c(S_c)\right)$$

with *joint energy functions* E_c for all cliques in C.

4.3 Conditional Random Fields

So far, we have defined graphical models encoding a joint distribution for a set of random variables $S = \{X_1, \ldots, X_n\}$. The same graph representation as for Markov Random Fields can also be used to define a (*discriminative*) conditional distribution $p(X_1, \ldots, X_n | Y_1, \ldots, Y_m)$. It defines the distribution for set a of random variables $\{X_i | i = 1, \ldots, n\}$ (also called *target variables*) given some disjoint set of observations $\{Y_j | j = 1, \ldots, m\}$ (see [KF09]). It is used to encode relationships between known observations and construct consistent interpretations.

By definition a *Conditional Random Field (CRF)* is an undirected graph G whose nodes correspond to the combined set of variables $S = \{X_i\} \cup \{Y_j\}$. Like for Markov Random Fields the CRF is annotated with a set of potential functions $\psi_c(S_c)$ with clique variables $S_c \subset S$. Then the conditional distribution is encoded by

$$p(X_1, \ldots, X_n | Y_1, \ldots, Y_m) = \frac{1}{Z(Y)} \prod_{c \in C} \psi_c(S_c) \tag{4.2}$$

with the normalization constant Z summing only over all observations X_j:

$$Z(Y) = \sum_{X_1, \ldots, X_n} \prod_{c \in C} \psi_c(S_c)$$

In fact, this definition for Conditional Random Fields can be derived from the definition of Markov Random Fields. Given two sets of random variables X and Y, we have

$$\begin{aligned} p(X|Y) &= \frac{p(X,Y)}{p(Y)} = \frac{p(X,Y)}{\sum_X p(X,Y)} \\ &\underset{\text{Def. MRF}}{=} \frac{\prod_c \psi_c(S_c)}{\sum_{X,Y} \prod_c \psi_c(S_c)} \cdot \frac{\sum_{X,Y} \prod_c \psi_c(S_c)}{\sum_X \prod_c \psi_c(S_c)} \\ &= \frac{\prod_c \psi_c(S_c)}{\sum_X \prod_c \psi_c(S_c)}. \end{aligned}$$

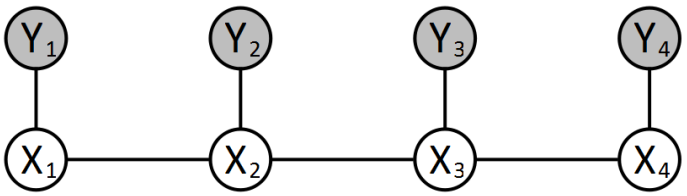

Figure 4.4: *An example for a pairwise CRF. The graph is built of pairwise cliques with white target variables X_i and gray observation variables Y_i. This example can also be interpreted as a Hidden Markov Model [KF09]. We try to make predictions about hidden states of the variables X_i using the knowledge about dependencies on observations.*

That means the distribution of a CRF equals the distribution of a MRF conditioned on some known observations.

Figure 4.4 shows an example for an often used, special type of CRF. There is an equal number of random variables and observation variables. Each observation variable is connected to exactly one random variable, that means, given X the observations become conditionally independent. Although the relations between the observations are not directly modeled, there might still be some dependencies. The fact that the joint probabilities of the observations are avoided allows to incorporate (into the model) a rich set of observed variables whose dependencies might be quite complex indeed. For that reason such models are often used for computer vision tasks such as image segmentation or object recognition [GG84, ATC+05, KHS10]. Though, the underlying graph for the target variables X_i is often a grid-like lattice structure instead of a chain.

4.4 Pairwise Markov Models

Within this section the focus will be set on *Pairwise Markov Models* [Bis06, KF09]. In this special case the underlying graph contains cliques of size 2 only. Thus, any distribution factorizes into a product of pairwise potentials defined on each link between two variables. Figure 4.4 shows such a model. When dealing with *Hidden Markov Models (HMM)* [KF09] we are usually given an additional set of observation variables Y_i pairwise linked to a set of *latent* or *hidden* variables X_i. In a scenario like that the goal is to gain some knowledge about the hidden states of the variables X_i given the con-

ditional dependencies to direct observations. Since every hidden variable is exclusively linked to one observation, it is convenient to specify the graph structure which only defines the conditional relations between hidden variables. Models like that are popular for Computer Vision tasks as well as in physics. A well studied example is the so-called *Ising Model* [Cip87]. There, the nodes of a lattice model represent 'mini-magnets' which prefer to be aligned in the same state, depending on the temperature. That means, given the observations, the set of potentials encourages neighboring variables to have the same state.

In case of pairwise Markov Models literature usually distinguishes between two types of potential functions [Bis06]. The potential function representing links between observations and unknown states for each hidden node i is called *evidence* and is commonly denoted by $\phi_i(X_i, Y_i)$. The dependencies $\psi_{ij}(X_i, X_j)$ between two hidden variables X_i and X_j is called *compatibility function*. Finally, there is a factor for each link in the model and the joint probability for 'hidden' scene and observations can be factorized as follows:

$$p(X_1, \ldots, X_n, Y_1, \ldots, Y_n) = \frac{1}{Z} \prod_i \phi_i(X_i, Y_i) \prod_{i \sim j} \psi_{ij}(X_i, X_j) \tag{4.3}$$

where $i \sim j$ denotes all pairs of nodes i and j (with $i \neq j$) which are connected by an edge. The Normalization is given by

$$Z = \sum_{i,j} \prod_i \phi_i(X_i, Y_i) \prod_{i \sim j} \psi_{ij}(X_i, X_j).$$

An interesting fact is, that this factorization also equals the definition of a Conditional Random Field because there is no direct conditional dependence between pairs of observation variables (Y_i, Y_j). Further, all links to hidden variables are exclusive. Thus, for normalization it is sufficient to sum over all pairwise combinations involving the X_i. That means, the distribution conditioned on observations as well as the joint probability distribution over all X_i and Y_i are the same. Finally, when all observations are given, the $Y_i = y_i$ are fixed and the joint distribution $p(X_1, \ldots, X_n, Y_1, \ldots, Y_n)$ also equals its marginalization $p(X_1, \ldots, X_n)$.

Thus, in this special case of a pairwise Markov Model, the MRF formulation equals the one for CRFs and we can simplify notation to specifying the joint probability only for the hidden variables:

$$p(X_1, \ldots, X_n) = \frac{1}{Z} \prod_i \phi_i(X_i) \prod_{i \sim j} \psi_{ij}(X_i, X_j) \tag{4.4}$$

5 Inference on Markov Random Fields

If all you have is a hammer,
everything looks like a nail.

Abraham Maslow: *The Psychology of Science* (1966)

In the scope of this thesis inference is the process of computing functions of some given distribution $p(X_1, \ldots, X_n)$. In case of a pairwise MRF the probability distribution is defined as shown in Equation 4.4. Direct evaluation of such a distribution is exponential in the number of variables. Assuming the domain of each variable X_i comprises K discrete states, the joint distribution is built of K^n possible combinations which is an infeasible problem in most cases. However, many problems are often addressed by finding solutions for a single variable - a marginal distribution.

In Chapters 6 and 7, i.e. when performing object detection, we will use Markov Random Fields to represent 3D object models in a graph- (part-) based manner. When it comes to the actual detection we are not interested in the joint distribution of all parts. Instead it might be sufficient to detect high values for only one single part variable. The conditional relationships to the other part variables then guarantee that we found one part of an instance of the whole model.

Marginal inference is concerned with the computation of the distribution for a subset of variables. For example, given we are interested in X_1, the

marginal distribution for $p(X_1)$ is

$$p(X_1) = \sum_{X_2,\ldots,X_n} p(X_1, X_2, \ldots, X_n). \tag{5.1}$$

5.1 Exact Inference

In general case, the exact inference of marginal distributions is still infeasible. It would require K^{n-1} operations. However, there are exceptions for non-loopy graph models [KF09, Bis06] such as pairwise chain-structured MRFs and pairwise tree-structured MRFs: When taking the special conditional independence structure into consideration the computation effort can be reduced to quadratic costs in the number of states.

5.1.1 Inference on Markov Chains

Given a chain graph model such as the example in Figure 4.4 for a so-called *Markov Chain*, each variable is conditioned on at most two neighbors. That means, each variable appears in at most two compatibility terms. The joint distribution can thus be inferred to

$$p(X_1, \ldots, X_n) = \frac{1}{Z} \prod_{i=1}^{n} \phi_i(X_i) \prod_{i=1}^{n-1} \psi_{i,i+1}(X_i, X_{i+1}). \tag{5.2}$$

The marginal distribution for a single random variable X_i is the given by

$$p(X_i) = \frac{1}{Z} \sum_{X_1 \ldots X_{i-1}} \sum_{X_{i+1} \ldots X_n} \prod_{i=1}^{n} \phi_i(X_i) \prod_{i=1}^{n-1} \psi_{i,i+1}(X_i, X_{i+1}). \tag{5.3}$$

Changing the order of summation and multiplication [Bis06] results in a recursive formulation:

$$p(X_i) = \frac{1}{Z} \cdot \phi_i \cdot \left[\sum_{X_{i-1}} \phi_{i-1} \psi_{i-1,i} \cdots \left[\sum_{X_1} \phi_1 \psi_{1,2} \right] \right] \left[\sum_{X_{i+1}} \phi_{i+1} \psi_{i,i+1} \cdots \left[\sum_{X_n} \phi_n \psi_{n-1,n} \right] \right]$$

With this decomposition we can reduce the computational costs from exponential to quadratic in the number of states. Given K discrete states per random variable, evaluating $\psi_{i,i+1}$ generates $O(K^2)$ costs; ϕ_i comes with costs

$O(K)$; there are $O(n)$ (actually $O(\#\text{edges})$) factors of ϕ and ψ. Thus $p(X_i)$ can be computed with a total cost of $O(n \cdot K^2)$.

This formulation for marginal inference on Markov Chains yields a recursive message passing algorithm which is given in Section 5.1.3. The node associated with the (marginalized) variable X_i is called the *root* node. Starting from the *leaves* each of the summed factors is stored and passed as a message to the next factor. Finally at root node all incoming messages are collected and multiplied with the root evidence.

5.1.2 Inference on trees

For pairwise tree models a similar but slightly more complex approach can be obtained. The main idea behind is that tree structures are cycle-free graphs. Then we can find a recursive reformulation for the marginal distribution. A derivation can be found in [Jan10]. Again, the marginal is factorized into a product of messages. Since messages are always passed along edges, the total computational effort is of order $O(\#\text{edges} \cdot K^2)$.

5.1.3 Message Passing Algorithm

In the previous sections we have seen that marginal inference for non-loopy, pairwise MRFs can be factorized into recursive structures. The recursion structure is implied by the structure of the underlying graph. Starting from the leaves in the graph 'local' *messages* are passed along the graph edges to the root node, thus generating the marginal distribution of the root. Thereafter, in order to compute the marginals for the remaining variables, the root message is propagated back to the leaves again [Jan10].

The Sum Product *Belief Propagation* algorithm introduces multidimensional variables $m_{i \to j}(X_j)$ which can intuitively be understood as a *message* from node i to node j about which state variable X_j should be in [Moo08, YFW01]. The size of $m_{i \to j}(X_j)$ will be of the same dimensionality as the number of states (domain size) of X_j. This is easiest encoded as a vector with each component representing the likelihood for 'node i believes node j is in the corresponding state'.

Finally, the marginal distribution for X_i is computed as follows:

$$p(X_i) = \frac{1}{Z} \cdot \phi_i(X_i) \prod_{j \in N(i)} m_{j \to i}(X_i) \tag{5.4}$$

with messages

$$m_{i \to j}(X_j) = \sum_{X_i} \phi_i(X_i) \psi_{ij}(X_i, X_j) \prod_{k \in N(i) \setminus j} m_{k \to i}(X_i), \tag{5.5}$$

where $N(i)$ represents all neighbors of node i.

In the special case of a Markov Chain the computation is a bit easier:

$$p(X_i) = \frac{1}{Z} \cdot \phi_i(X_i) \cdot m_{i-1 \to i}(X_i) \cdot m_{i+1 \to i}(X_i) \tag{5.6}$$

with

$$m_{i-1 \to i}(X_i) = \sum_{X_{i-1}} \phi_{i-1}(X_{i-1}) \psi_{i-1,i}(X_{i-1}, X_i) \cdot m_{i-2 \to i-1}(X_{i-1})$$

and

$$m_{i+1 \to i}(X_i) = \sum_{X_{i+1}} \phi_{i+1}(X_{i+1}) \psi_{i+1,i}(X_{i+1}, X_i) \cdot m_{i+2 \to i+1}(X_{i+1}).$$

At the ends of the chain, the messages $m_{0 \to 1}(X_1)$ and $m_{n+1 \to n}(X_n)$ are set to 1.

5.1.4 Most Likely Assignments

So far, we have dealt with estimating marginals of probability distributions. However, when focusing on detection scenarios we are less interested in estimating the distribution of a model. In fact, we are interested in finding the most probable assignment to all of the variables. In our case, i.e. having a pairwise Markov Model that means, we want to find the most likely joint assignment to the variables $X_1, \ldots, X_n$, given we observed some evidence $y_1, \ldots, y_n$. Thus, we are interested in finding $\xi = (x_1^*, \ldots, x_n^*)$ such that $p(\xi) = \max_{X_1 \ldots X_n} p(X_1, \ldots, X_n | y_1, \ldots, y_n)$. As discussed in Section 4.4, at this point conditional distribution and joint distribution are the same, so we can rewrite the problem to finding

$$\xi = \arg \max_{X_1 \ldots X_n} p(X_1, \ldots, X_n). \tag{5.7}$$

Please note finding the most likely joint assignment for the hidden variables is not the same as finding the set of states that individually are the most probable [Bis06]. The latter case can be solved by first running the Belief Propagation algorithm to find the marginal distributions for each variable X_i. Then the resulting marginals are maximized separately.

The problem of solving 5.7 can be approached similar to finding the marginal distribution in Section 5.1.1. We are looking for

$$p(\xi) = \max_{X_1} \cdots \max_{X_n} p(X_1, \ldots, X_n).$$

In case of a Markov Chain that is

$$p(\xi) = \frac{1}{Z} \max_{X_1} \cdots \max_{X_n} \left[\prod_{i=1}^{n} \phi_i(X_i) \prod_{i=1}^{n-1} \psi_{i,i+1}(X_i, X_{i+1}) \right]. \tag{5.8}$$

Similar to decomposing Equation 5.3 into a recursive form, Equation 5.8 can be reformulated as well (see [Bis06]):

$$p(\xi) = \frac{1}{Z} \cdot \max_{X_1} \left[\phi_1 \cdot \left[\max_{X_2} \phi_2 \psi_{1,2} \cdots \left[\max_{X_n} \phi_n \psi_{n-1,n} \right] \right] \right]$$

Likewise, tree-structured pairwise MRFs $p(\xi)$ can also be decomposed into a recursive structure. Therefore, $p(\xi)$ can be computed using the sum-product inference scheme - as implemented by the BP-algorithm - by just replacing the $\sum$ with the max operator, resulting in max-product message passing inference. Finally, all incoming messages (see Equation 5.4) are multiplied and maximized at the root node.

For example, given a Markov Chain of length n and taking root node X_n (the 'chain end'), the maximum of the joint distribution is inferred by:

$$p(\xi) = \frac{1}{Z} \max_{X_n} \left[\phi_n(X_n) \cdot m_{n-1 \to n}(X_n) \right] \tag{5.9}$$

$$m_{i-1 \to i}(X_i) = \max_{X_{i-1}} \left[\phi_{i-1}(X_{i-1}) \cdot \psi_{i-1,i}(X_{i-1}, X_i) \cdot m_{i-2 \to i-1}(X_{i-1}) \right]$$

Up to now, we can compute the global maximum $p(\xi)$ of the joint distribution. The result will be the same irrespective of which node is chosen as root. However, the main interest lies in finding the optimal assignment ξ. The solution to that can easiest be seen in the example for message passing inference on chains. Since X_n is the root, propagation starts at X_1 and ends

with maximizing incoming messages over X_n. Finding the maximizing assignment for X_n is easy. It is that one which maximizes 5.9. Z is constant, so we can compute ξ_{X_n} by

$$\xi_{X_n} = \arg\max_{X_n} [\phi_n(X_n) \cdot m_{n-1 \to n}(X_n)] \tag{5.10}$$

Now we need to determine the states ξ_{X_i} of the 'previous' variables that correspond to the maximizing configuration. Therefore a *back-tracking* [Bis06] approach can be applied, i.e. keeping track of which values of the variables gave rise to the maximum of their direct successors.

$$\xi_{X_{i-1}} = \arg\max_{X_{i-1}} [\phi_{i-1}(X_{i-1}) \cdot \psi_{i-1,i}(X_{i-1}, \xi_{X_i})] \tag{5.11}$$

In fact, there is also a way to avoid the back-tracking approach. However, it comes with the expense that the computation itself requires more memory. In this case, again we make use of the message passing algorithm but additionally we store a 'maximizing' path $tr_i(x_i)$ for every state $x_i \in X_i$. A path to node i comprises one state for each preceding node - those states which help maximizing the path to node i. That means, for states $x \in X_i$ and $y \in X_{i-1}$ of two linked nodes we store a path $tr_i(x) = [tr_{i-1}(y), x]$, if $y = \arg\max_{X_{i-1}} [\phi_{i-1}(X_{i-1}) \cdot \psi_{i-1,i}(X_{i-1}, x)]$. Accordingly, the best solution ξ is then given by $tr_n(\xi_{X_n})$.

Pairwise Tree Structures: For tree-structured graphs the computation of ξ is similar to chain graphs. Starting from the leaves, the propagation algorithm infers the maximum of the joint distribution stopping propagation at the root node. Then the assignment for ξ_{root} is computed and back-tracking yields the full joint assignment ξ.

Max-Sum Inference: Depending on the actual graph structure the max-product algorithm involves products of potentially very small factors. That can easily lead to arithmetical issues during computation. One way to circumvent those is to normalize each single message. Another approach is to compute potentials in logarithmic space. Since logarithm is a strongly monotonic function, max operator and log function can be interchanged. Finally, taking logarithm has the effect of simply replacing the products by sums yielding a max-sum algorithm. For chain structures it is called the *Viterbi* algorithm [RN93, Bis06]:

Be $\overline{\phi} := \log(\phi)$ and $\overline{\psi} := \log(\psi)$, then

$$p(\overline{\xi}) = \max_{X_n} \left[\overline{\phi}_n(X_n) + m_{n-1 \to n}(X_n) \right]$$

$$m_{i-1 \to i}(X_i) = \max_{X_{i-1}} \left[\overline{\phi}_{i-1}(X_{i-1}) + \overline{\psi}_{i-1,i}(X_{i-1}, X_i) + m_{i-2 \to i-1}(X_{i-1}) \right]$$

Here, the message $m_{0 \to 1}(X_1)$ is defined 0.

Spatially Local Maxima: Besides message passing there are also other inference approaches: sampling methods such as *Markov Chain Monte Carlo* algorithms draw a number of samples to compute the most likely assignments of a graphical model. However, message passing algorithms such as Belief Propagation come with a benefit, only little mentioned in literature: In contrast to sampling methods, Belief Propagation computes many potential solutions in parallel. When computing the Max-Product BP algorithm, we are able to not only extract the globally optimal joint variable assignment but we also get all solutions which are optimal for the root node, i.e. the *max-marginal* distribution of the root node variable. We just have to skip the last maximum operation from computation. For example for chains see equation 5.9.

That qualifies message passing algorithms for further applications. For example, in Chapters 6 and 7 we represent 3D objects using graphical models. In that case, the state space of each graph node is represented by a point cloud. The task is to detect all instances of the graphical model in the point cloud - not only the best fitting one. If we now compute the most likely assignments for the root node, each vertex in the point cloud is assigned a likelihood for being an instance of the root node. Thus, (after thresholding) we can extract all 'spatially local' maximum root assignments in the point cloud. These are then used to start back-tracking, yielding all model instances in the point cloud (see Figure 6.6).

5.2 Approximate Solutions

For many probabilistic models of practical interest, exact inference is intractable. As soon as it comes to computing most likely assignments (or probability distributions) for loopy graph models, we we need to resort to some form of approximation. For this purpose different strategies have evolved:

Two popular approaches are *Sampling*-based and *Propagation*-based methods. In the following we will shortly review both before we focus on a novel approach for approximate inference of a fully connected graphical model in Section 5.2.3.

5.2.1 Sampling Methods

Here, we consider approximate inference methods based on numerical sampling. Those techniques are also called *Monte Carlo* techniques [GG84, ADFDJ02]. They are handy, especially when complex, loopy graph structures are involved [KKKT12]. However, they often require a lot of computation time to generate good approximations, so they are not well suited for interactive tasks. Moreover, they yield only one result per computation. Thus, if for example the goal is to detect all model instances in a scene, a fair number of repeated runs would be required.

The idea behind Monte Carlo sampling is to generate a set of independent and identically distributed multidimensional data *samples* (sometimes also called *particles*) until it represents a good approximation for a target distribution. However, the target distribution is not explicitly given. In fact we only have the graphical model defining conditional dependencies of single variables in the joint distribution. However, for most applications the main interest lies not in computing the distribution itself. It rather lies in evaluating expectations for an objective function or to optimize some energy functional.

In general, the expected value of an arbitrary function $f(x)$ with respect to a probability function $p(x)$ is defined by

$$\mathbb{E}[f] = \int f(z)p(z)dz$$

In our case p is not directly known but its distribution is given in form of a set of N samples $x^{(i)}$. For large numbers N the expectation $\mathbb{E}[f]$ can be approximated [ADFDJ02]:

$$\hat{f} = \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})$$

With that approximation we can compute the probability of a new sample y using a so-called kernel estimate. If we define a gaussian kernel function $f = f_y(x) = \frac{1}{a\sqrt{\pi}} \exp\left(-(x-y)^2/a^2\right)$ then the expectation for f_y converges to

the probability p at a queried position y.

$$\hat{f}_y = \frac{1}{a\sqrt{\pi}N} \sum_{i=1}^{N} \exp\left(-(x^{(i)} - y)^2/a^2\right) \underset{\substack{N\to\infty \\ a\to\infty}}{\longrightarrow} p(y)$$

However, we cannot compute those probabilities as long as there are not enough samples available. Those can be generated by iterative sampling methods. *Markov Chain Monte Carlo (MCMC)* describes a class of algorithms which generate a sequence of samples of a complex distribution. A widely used implementation is the so-called *Gibbs sampler* [Bis06, KF09].

5.2.2 Loopy Belief Propagation

Although only guaranteed to converge on cycle-free graphs, Belief Propagation may also be used to approximate inference for general graphs. The algorithm is then typically referred to as *Loopy Belief Propagation* [Bis06, KF09].

The computation steps for marginal inference as well as for maximum likelihood estimates are the same. However, when applying Belief Propagation to graphs containing cycles, it is not possible to compute all messages before they are needed to update another message. There are cyclic dependencies. Therefore, all messages need to be initialized, for example with uniform values [Jan10].

Belief propagation is neither guaranteed to converge (it might even oscillate), nor is there any warranty to compute good approximations [Moo08]. This becomes even more crucial, the more loops are involved. Besides sophisticated updating strategies, i.e. the plan in which order single nodes are updated, there are several approaches which try to tackle the loops. For example Generalized Belief Propagation [YFW01] sends messages between groups of nodes. However, this requires to sum over the groups variables at once, which is intractable for problems with a big state space. Another form of approach is to approximate the graph with spanning trees [Jan10], thus omitting some conditional dependencies. In that case BP yields an exact result for the approximate model.

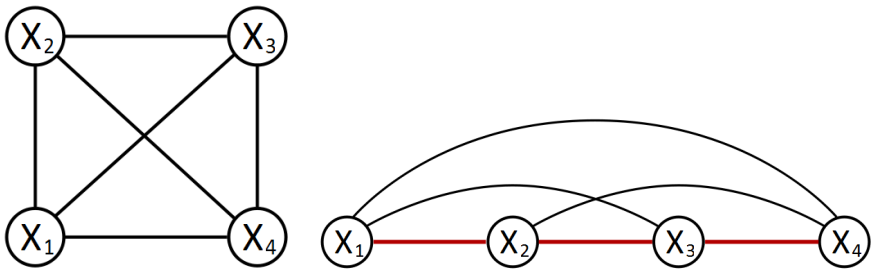

Figure 5.1: *Example for a fully connected graphical model with 4 random variables and its representation as a third order Markov Chain. The red edges show the chain order.*

5.2.3 Markov Chain Approximation

In Chapter 7 we will present a statistical model for object detection which is based on a fully connected graph. That means each edge is part of a loop. In our case the state space for each variable will be associated with point clouds, i.e. it is in the order of millions. Those reasons render inference methods based on loopy Belief Propagation inapplicable. On the other hand sampling approaches are not helpful, too. Although a Monte Carlo method might yield approximations close to the optimum, it works only for a very large number of samples which costs a lot of time. Furthermore, the detection scenario involves finding all spatially local maxima (see Section 5.1.4) of the joint distribution.

Hence, we need to find a formulation for the problem such that we can approximate it with a message passing scheme. The idea is that each fully connected graphical model can be represented as a Markov Chain of order $n-1$, if n is the number of nodes of the chain [Bis06]. We assume the cyclic dependencies can be neglected for approximate estimation of the maximal likelihood. More precisely, we assume that we find a chain inference, such that for each pair of successive nodes (X_i, X_{i+1}) there is a bi-directional conditional relation, but for all other pairs (X_i, X_j) with $j > i+1$ we can assume a one-directional condition such that X_j is dependent on X_i (but not vice versa).

Figure 5.1 shows an example for a fully connected graphical model with 4 random variables and its representation as a third order Markov Chain. The

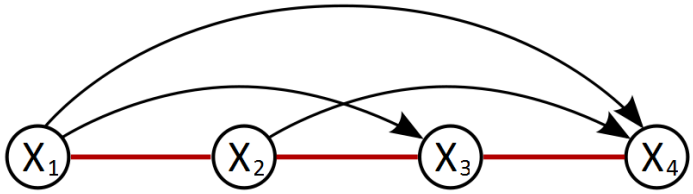

Figure 5.2: *Graphical model approximation for a fully connected graph with 4 random variables (see Figure 5.1). Please note neighboring variables in terms of chain order still own two-directional conditional relations.*

joint distribution is inferred by

$$p(X_1, X_2, X_3, X_4) = p(X_1) \cdot p(X_2|X_1) \cdot p(X_3|X_1, X_2) \cdot p(X_4|X_1, X_2, X_3).$$

The most likely estimate $p(\xi)$ for the fully connected model is given by

$$p(\xi) = \max_{X_1, X_2, X_3, X_4} \left[p(X_1) \cdot p(X_2|X_1) \cdot p(X_3|X_1, X_2) \cdot p(X_4|X_1, X_2, X_3) \right].$$

Unfortunately the loopy conditional structure of the graph does not allow to generate a recursive formulation similar to the one seen in Section 5.1.4. However, that is overcome if for every variable X_i dependency is reduced to dependency on all predecessors $X_1, \ldots, X_{i-1}$ in terms of chain order (see Figure 5.2). Then we can derive a recursive formulation with $p(\widetilde{\xi}) \approx p(\xi)$:

$$\begin{aligned} p(\widetilde{\xi}) \quad = \quad & \max_{X_4} \Big[\max_{X_3} [\ p(X_4|X_1, X_2, X_3) \cdot \\ & \max_{X_2} \Big[p(X_3|X_1, X_2) \cdot \max_{X_1} [\ p(X_2|X_1) \cdot p(X_1)\ ] \Big] \Big] \Big] \end{aligned}$$

Similar to computing the arg max for a pairwise Markov Chain we yield the following recursion:

$$\begin{aligned} \widetilde{\xi}_{X_1}(x_2) &= \arg\max_{X_1} [\ p(x_2|X_1) \cdot p(X_1)\ ] \\ \widetilde{\xi}_{X_2}(x_3) &= \arg\max_{X_2} \left[\ p\left(x_3|X_2, \widetilde{\xi}_{X_1}(X_2)\right) \cdot p\left(\widetilde{\xi}_{X_1}(X_2), X_2\right)\ \right] \\ \widetilde{\xi}_{X_3}(x_4) &= \arg\max_{X_3} \left[\ p\left(x_4|X_3, \widetilde{\xi}_{X_2}(X_3), \widetilde{\xi}_{X_1}(\widetilde{\xi}_{X_2}(X_3))\right) \cdot p\left(\widetilde{\xi}_{X_1}(X_3), \widetilde{\xi}_{X_2}(X_3), X_3\right)\ \right] \\ \widetilde{\xi}_{X_4} &= \arg\max_{X_4} \left[\ \widetilde{\xi}_{X_3}(X_4) \cdot p\left(\widetilde{\xi}_{X_1}(X_4), \widetilde{\xi}_{X_2}(X_4), \widetilde{\xi}_{X_3}(X_4), X_4\right)\ \right] \end{aligned}$$

$\widetilde{\xi}_{X_i}(x_{i+1})$ denotes a vector representing the approximate optimal assignment for X_i, given $(X_{i+1} = x_{i+1})$ is chosen in the next iteration.

In general the recursion can be easily expressed using the BP message passing formalism (see Equation 5.9) in combination with the assignment tracking:

$$\begin{aligned}
p(\xi) \approx p(\widetilde{\xi}) &= \max_{X_n} [\ m_{n-1\to n}(X_n)\ ] \\
m_{i\to i+1}(X_{i+1}) &= \max_{X_i} \left[p(X_{i+1}|X_i, \widetilde{\xi}_{X_1,\ldots,X_{i-1}}(X_i)) \cdot m_{i-1\to i}(X_i)\right] \\
m_{1\to 2}(X_2) &= \max_{X_1} [\ p(X_1|X_2) \cdot p(X_1)\ ] \\
\widetilde{\xi}_{X_1,\ldots,X_{i-1}}(X_i) &= (\widetilde{\xi}_{X_1}(X_i), \ldots, \widetilde{\xi}_{X_{i-1}}(X_i)) \\
\widetilde{\xi}_{X_j}(X_i) &= \arg\max_{X_j} \left[p(X_i|X_j, \widetilde{\xi}_{X_1,\ldots,X_{j-1}}(X_j)) \cdot m_{j-1\to j}(X_j)\right]
\end{aligned}$$

In this way we generate a Markov Chain approximation for a fully connected graphical model. However, that works only for joint distributions for which it is clear how the conditional probability is computed. For example the Gaussian distribution is such a distribution and will be used for the model described in Section 7.

Another important point is that the inference order must be chosen carefully such that the omitted conditional relations are of lower importance than the remaining ones. For inference we assume conditional independence between the first variable and the other ones (except for its direct successor). If that assumption turns out to be wrong we might introduce potential approximation errors already at the beginning and we might not find a solution which is close to the optimum. Though the model in Section 7 will take that into account.

6

A Chain Model for 3D Object Detection

Make everything as simple as possible,
but not simpler.

Albert Einstein (1879 - 1955)

This chapter examines the idea of learning objects represented as a chain configuration of geometric parts. We introduce a formal model for chain constellations based on a Markov model. Further, we propose an efficient algorithm that simultaneously detects a large number of instances including semantic part correspondences. After a short user-guided training stage, in which one or a few exemplars of semantically similar objects are sketched directly onto point cloud data, the algorithm automatically finds all instances of the learned object category. In particular, the algorithm is able to recognize broader classes of semantically similar but geometrically varying shapes, which is very difficult using unsupervised techniques. In a number of experiments, we apply the technique to point cloud data from 3D scanners. The algorithm is able to detect object instances with low rates of false positives and negatives.

The work presented in this chapter has been published at Eurographics 2011 [SJW+11]. With permission of the co-authors main text passages and figures are utilized without tagging them individually.

Figure 6.1: *We propose an interactive learning algorithm for 3D object detection. Object classes can be intuitively defined with user strokes. In this example, 2-3 instances have been sketched on the model for each class (indicated by different colors). Further instances are found automatically using inference on a Markov chain model.*

6.1 Introduction

Approaching the detection problem defined in this thesis we can roughly distinguish between global models, i.e. modeling the whole object as one piece of geometry, and part-based models. Whenever constructing a global object model there is a major issue: Most objects can be decomposed into more or less distinct object parts and it becomes difficult to model strong variations within single parts using a global model. For example there are (generative) statistical models for faces such as in [BV99]. They construct a complete high-dimensional face space which covers some variety of faces. However, the coverage is only as complete as the set of training data. If that approach would be extended to full body-models, we would not only need a tremendous amount of data to capture a reasonable, fine grained space of all joint variations, the dimensionality of the search space would simply explode.

Hence, in order to be able to model complex objects we need to derive a model for 3D objects which combines statistical models for smaller parts, a so-called *part-based* model. However, detecting single parts separately does not yield a recombination scheme, and using simple combinatorial heuristics

is unreliable. Graphical models (see Chapter 4) represent a good means to define models which combine single parts in a joint objective function, i.e. a joint distribution.

Though, when modeling there are different graph topologies which must be taken into account. The optimal case would be to compute a fully connected graphical model. However, as discussed in Section 5 there exist no exact inference methods to compute those distributions in feasible time. In Chapter 7 we will see such a model but we need to apply an approximate inference scheme in order to find assignments.

For the approach in this chapter we use a chain topology. It models the relations between pairs of different parts - in contrast to star-like topologies which model only relations to a designated center. In fact, part-based object models with star-topology such as implicit shape models [LLS04, LLS08, VSS12] (or Hough-transform models [KPW+10, RGKG12] in general) are well known and yield robust approaches for detection applications. Their strong point is that they yield high detection rates for object centers. However, there is also a weak spot. Star topologies are prone to false positive detections. The main reason for that is that all non-center parts are conditionally independent of each other. That can be explained more intuitively: Models of that topology are mostly implemented by voting schemes. Every object part votes (independently from all other parts) for an assumed object center. That results in a histogram (a discretized, marginal distribution) of center positions where the local maxima represent detected object centers. But, the votes are only consistent in terms of the center positions, and there is no guarantee that all votes for one object center come consistently from one existing object instance. For that reason there is an issue with false positive detections in complex scenes (e.g., where multiple model instances are close by), and part correspondences might be flawed. This issue arises with Hough-voting models in general [RGKG12]. However, this problem can be overcome by constructing a model which incorporates dependencies between pairs of different parts (other than relating all to one center). An intuitive way to do so is to implement a graphical model with chain topology.

In order to define an object class, the approach is particularly designed to run with minimal and simple user input, which is crucial for making it a useful interactive tool. Typically, a single ‘stroke‘ in a point cloud is enough to train a simple model, where several exemplars can span more complex classes with significant geometric variation. In particular, we can detect semantically similar parts, which permits applications that are currently mostly inaccessible

to unsupervised global correspondence algorithms.

6.2 Model Definition

In the following we present a part-based model for 3D object detection. The goal is to detect 3D object instances (*shapes*) via chain-like configurations of local geometry (*parts*) such that the spatial alignment (*constellation*) of these parts as well as the parts' local appearance (*local shape*) coincide with a semantic object class. That means, we represent objects as a chain graph with nodes describing single parts. In this way the part-based graph structure also induces a correspondence structure between shapes. The current implementation works on manifolds approximated by finite point clouds, therefore being applicable to raw 3D scanner data. An implementation for other representations (e.g., triangle meshes) would require only minimal modifications. The training data is expected to be given in the same chain-structured format, with nodes being placed at corresponding locations. Figure 6.2 shows an example definition for a window object. The aim is to learn the statistical characteristics of these chain graphs and subsequently find all occurrences of similar constellations in an input data set.

Formally, we assume that we are given an input surface $\mathcal{S} \subset \mathbb{R}^3$ that is a 2-manifold of arbitrary topology. From the perspective of the algorithm it will be given (as a point cloud) in a sampled representation $S = \{\mathbf{s}_1, ..., \mathbf{s}_n\}, \mathbf{s}_i \in \mathbb{R}^3$. We encode shapes on S as a chain of k local parts $H = (\mathbf{h}_1, ..., \mathbf{h}_k)$, where $\mathbf{h}_i \in S$. Each part $\mathbf{h}_i$ represents both, the local shape $\mathbf{d}_i = D(\mathbf{h}_i)$ of this part, as well as the position $\mathbf{x}_i = X(\mathbf{h}_i)$ of the part itself (i.e. $\mathbf{x}_i = \mathbf{h}_i$).

A specific object class is characterized by statistical models for both the local part shapes $D = (\mathbf{d}_1, \ldots, \mathbf{d}_k)$ with model parameters θ_i^ϕ and the spatial alignment $X = (\mathbf{x}_1, \ldots, \mathbf{x}_k)$ with parameters θ_i^ψ for each chain segment. The score for an object belonging to a given class can be computed by a probability function $p : S^k \to [0, 1]$ which we define over the joint distribution of a Markov Chain model, and thus it can be inferred exactly (see Chapter 5). It is defined by the normalized product of two potential functions: Singleton evidence functions ϕ_i that prescribe local shape $\mathbf{d}_i$ of an object part and pairwise compatibility functions ψ_i that encode the relative alignment of two neighboring part positions $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. Given the parameters $\theta = (\theta_i^\phi, \theta_i^\psi)_i$,

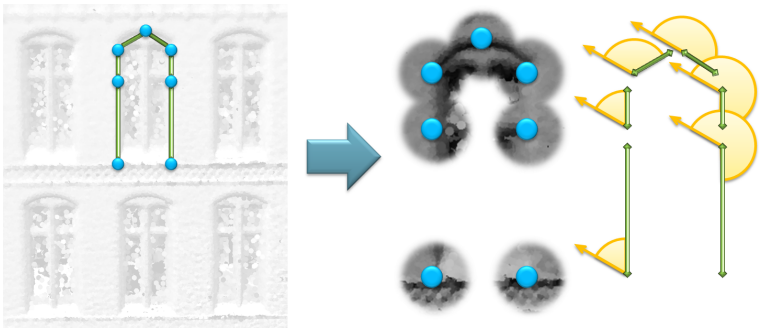

Figure 6.2: *Example for a window model: The object parts are aligned along the outline of a window. Starting from the lower left corner, we chose distinct parts at edges and corner points. Blue marks the variables in the point cloud space and green the edges of the graphical model. The evidence function ϕ yields matching scores for local shapes and ψ defines the compability of the parts, i.e. the relative alignment, given a global orientation (yellow arrows).*

we can infer a specific model class:

$$p(D, X, H|\theta) = \frac{1}{Z} \prod_{i=1}^{k} \phi_i(D(\mathbf{h}_i)) \prod_{i=1}^{k-1} \psi_i(X(\mathbf{h}_i), X(\mathbf{h}_{i+1})) \tag{6.1}$$

As already mentioned the part variables $\mathbf{h}_i$ coincide with their positions $\mathbf{x}_i$, furthermore local shapes can also be identified by the part positions $\mathbf{x}_i$. Accordingly we simplify notation of Equation 6.1 using solely the position variables $\mathbf{x}_i$:

$$p(D, X, H|\theta) = \frac{1}{Z} \prod_{i=1}^{k} \phi_i(\mathbf{x}_i) \prod_{i=1}^{k-1} \psi_i(\mathbf{x}_i, \mathbf{x}_{i+1}) \tag{6.2}$$

This means that an object class is completely characterized by specifying the set of potential functions $\{(\phi_i)_i, (\psi_i)_i\}$. In the following section, we will look at how to define and learn these functions from user input.

6.3 Learning

For representing both the ϕ_i and the ψ_i, we use simple Gaussian models. Obviously, there are more sophisticated learning techniques such as support vector machines or boosted classifiers that are frequently applied at this point in computer vision applications [VJ01, PJC+10]. However, we are aiming at

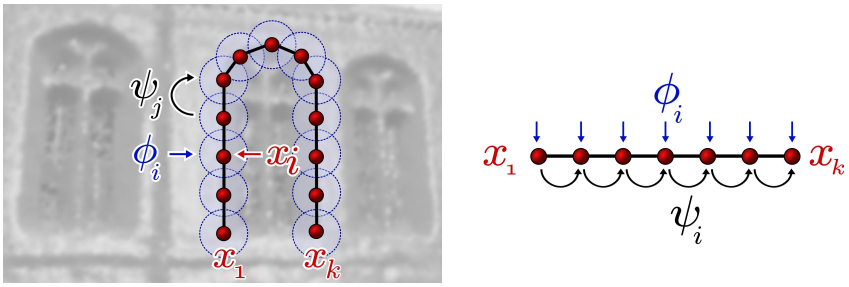

Figure 6.3: *Objects are modeled as Markov chains: Each node is controlled by a potential function ϕ that matches local shape. In addition, pairwise potentials ψ encode the variations of the spacial alignment of parts.*

learning from a minimal amount of training data (sometimes even only one or two examples). Complex non-linear methods are of little use here as the training data itself does not support many degrees of freedom. Thus, a simple linear statistical model is an adequate choice for our scenario.

To simplify, we assume that all models have a designated upward direction. That is a convenient assumption since there is a fair number of tasks (e.g. finding windows in city scans) that involve data for which there is a natural orientation. For models such as architectural scenes, statues, as well as many other man-made and natural artifacts, this is the case and can be easily specified interactively. We will denote this direction as **up** in the following. In conjunction with the surface normal $\mathbf{n}(\mathbf{x}), \mathbf{x} \in S$, this gives us a unique coordinate frame for each surface point, removing a rotational degree of freedom from the detection problem. In the following, we use $(\mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{x}), \mathbf{n}(\mathbf{x}))$, or in short $(\mathbf{u}, \mathbf{v}, \mathbf{n})$ when clear from the context, to denote an orthogonal, right-handed coordinate frame where $\mathbf{n}$ is the surface normal and $\mathbf{u}$ is a vector orthogonal to $\mathbf{n}$ that is closest to **up**.

User Interaction: For the learning phase, we ask the user to sketch one or more poly-lines (i.e. a series of points successively connected by straight lines) on the surface - a natural way of user input for object definition of a part-based chain model. We have implemented a tool that combines a 3D point-cloud viewer with a surface curve editor to specify such poly-lines interactively. Each click leaves a control point on the surface, thus defining the center positions of local object parts. If multiple object instances are specified for one model class, the algorithm expects the user to put the nodes in

corresponding positions, thus establishing semantic correspondences between all training instances. This is not a big problem in practice as users usually tend to chose salient features as node points that are rather easy to locate.

Learning Local Shape: We define the potential $\phi_i(\mathbf{x}_i)$ by a Gaussian model with parameters $\theta_i^\phi = (\mu_i^\phi, \Sigma_i^\phi)$ in the shape space of local pieces of geometry in the vicinity of each data point: We cutout consistent regions around each point of the model and form local depth images (height fields) along the normal axis. We will refer to these as *descriptors*. The choice for height fields is reasonable: There is a number of more sophisticated and robust histogram-based shape descriptors, e.g. Shape Contexts [BMP02] or adaptations of HOG-descriptors [DT05] or SIFT descriptors [Low03] for 3D objects. However, our focus will be on evaluating strengths and weaknesses of the model itself, hence we are looking for robust but simple shape descriptors. Descriptors other than height fields are not part of the analysis. Last but not least we want to limit the computational effort for descriptor computations, since we aim at interactive computation times.

The descriptor space is high dimensional: every depth image pixel represents one degree of freedom. We learn a low dimensional subspace as Gaussian distribution by a principal component analysis (PCA) of corresponding training points. For a single training example, this automatically degrades to learning a constant mean depth image.

Technically, we parametrize the neighborhood of each point $\mathbf{x} \in S$ as local height fields [PG01]: We fix a radius parameter r_ϕ, specified by the user. It specifies the "local coverage" of the parts, i.e. how far the algorithm should reach out around each node to compare local geometry. We then cut out all points within a radius of r_ϕ from $\mathbf{x}$ and project them in normal direction onto the tangent plane in $\mathbf{x}$ spanned by $(\mathbf{u}, \mathbf{v})$. Let (u_j, v_j, n_j) denote the respective coordinates of the points in the local tangent frame. We collect all points that fall within a rectangular region $u_j, v_j \in [-r_\phi, ..., r_\phi]$ and store their normal direction n_j. We finally splat these values into a $d \times d$ pixel grid (we use a fixed $d = 16$) and use a push-pull algorithm to fill potential holes. We precompute these descriptors for all points $\mathbf{x} \in S$ in the input and compress them to more compact d_c-dimensional vectors using PCA (in experiments, we keep the $d_c = 16$ dominant of 256 dimensions of the PCA). This reduces computational costs but it will not significantly reduce the discriminability of the descriptors because the projection will well preserve distances in the descriptor space [DG03]. We denote these values $\mathbf{d_x}$ (for all $\mathbf{x} \in S$).

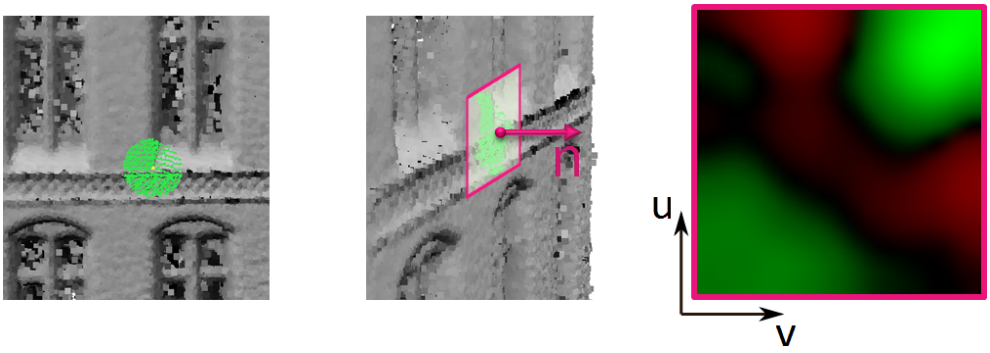

Figure 6.4: *Local shape descriptor: For a point we extract all neighbors within a given radius form the point cloud. Given the surface normal n, a tangent frame is computed. Finally, a smooth depth image is generated by splatting all extracted points onto the tangent plane. The right picture shows the noise-reduced result after PCA compression.*

Given this database of local geometry, learning a Gaussian model of $\phi_i(\mathbf{x})$ is straightforward. Assume that the user has specified m input instances, each consisting of k nodes with positions $\mathbf{x}_i^{(j)}, i = 1..k, j = 1..m$. We compute a d_c-dimensional mean μ_i^ϕ and a $d_c \times d_c$ covariance matrix Σ_i^ϕ of the sets $\{\mathbf{d}_{\mathbf{x}_i^{(j)}}\}_j$ for each node $i = 1..k$ independently across the chain using PCA. The so-defined subspaces of significant variance will typically be much lower dimensional than the original descriptor space because they were computed from only very few and presumably highly correlated examples. We store these statistics to characterize the training set. Then we can evaluate ϕ_i as multivariate normal distribution:

$$\phi_i(\mathbf{x}_i) \sim \exp(-\frac{1}{2}(\mathbf{d}_{\mathbf{x}_i} - \mu_i^\phi)^T \overline{\Sigma_i^\phi}^{-1} (\mathbf{d}_{\mathbf{x}_i} - \mu_i^\phi)) \tag{6.3}$$

where

$$\overline{\Sigma_i^\phi} := \Sigma_i^\phi + \lambda \mathbf{I}. \tag{6.4}$$

Here, we do not use the original covariance matrices but add an additional uniform noise term, specified by the user parameter λ. This term models uniform Gaussian noise in the data. Typically, it should be set according to the measurement accuracy. Often, this is not known exactly and has to be estimated roughly. Increasing this parameter also increases the tolerance towards shape variations that are not captured in the training set. In the user interface, the parameter λ can be specified as fraction of the overall variance (maximum eigenvalue of the covariance matrix of the complete descriptor

space) of the input patches. This choice facilitates the definition of λ as it becomes independent of scaling and to some extent invariant to different input data sets. Please note that λ is not learned automatically from the user-specified examples. Using only few examples might not be sufficient to estimate full rank covariance matrices Σ_i^ϕ, and a statistically reliable estimation of the full matrix would require prohibitively large amounts of training data. This simple mixture model avoids this problem by separating the noise floor from one or two main directions of variation in the descriptor space.

Learning Spatial Layout: Learning the spatial layout parameters $\theta_i^\psi = (\mu_i^\psi, \Sigma_i^\psi)$ of the chain nodes proceeds analogously: For each user specified instance, we measure the length and the *twist* of each edge (chain segment) $\{\mathbf{x}_i, \mathbf{x}_{i+1}\}$. The length is just the Euclidean distance of the endpoints. The twist is a vector of two angles that describes how to rotate around the normal $\mathbf{n}(\mathbf{x}_i)$ and out of the tangent frame in order to align the next part position $\mathbf{x}_{i+1}$ relative to $\mathbf{x}_i$, thereby encoding the orientation. For each triple of length and orientation parameters, we again independently estimate means μ_i^ψ and 3×3 covariance matrices Σ_i^ψ for corresponding graph edges in the user-defined training set. We scale the length and the two angle parameters with three constant factors to obtain a unit variance for each chain segment over all object instances. This whitening transform is necessary because angles and length are measured in different units so that the joint probability might be affected by the different measures rather than data characteristics. In order to form the final potential function $\psi_i(\mathbf{x}_i, \mathbf{x}_{i+1})$, we again add isotropic Gaussian noise to account for general inaccuracies. In the following, let $\mathbf{c}_{\mathbf{x}_i,\mathbf{x}_{i+1}}$ denote the scaled length and orientation parameters. Using this notation, we obtain:

$$\psi_i(\mathbf{x}_i, \mathbf{x}_{i+1}) \sim \exp(-\frac{1}{2}(\mathbf{c}_{\mathbf{x}_i,\mathbf{x}_{i+1}} - \mu_i^\psi)^T \overline{\Sigma_i^\psi}^{-1} (\mathbf{c}_{\mathbf{x}_i,\mathbf{x}_{i+1}} - \mu_i^\psi)) \tag{6.5}$$

with

$$\overline{\Sigma_i^\psi} := \Sigma_i^\psi + \begin{pmatrix} \lambda_{length}^2 & 0 & 0 \\ 0 & \lambda_{angle}^2 & 0 \\ 0 & 0 & \lambda_{angle}^2 \end{pmatrix}. \tag{6.6}$$

The parameters λ_{length} and λ_{angle} describe the anticipated variation of edge length and orientation. In the user interface, length variations are specified relative to the average segment length (in percent) and angular variation by the expected standard deviation in degrees.

Furthermore, we add an option to scale all covariance matrices that are learned from data (for both local shape and spatial layout) by a factor > 1

so that more variation in the learned direction is allowed. This is useful as small training sets might not capture the full variance, so the user can emphasize learned correlations.

6.4 Inference

The task is now to find all solutions that match the probabilistic model defined in Equation 6.2. We are interested in maximizing the posterior distribution $p(H|D, X, \theta)$ for all part assignments. Using Bayesian inference that means we need to find assignments $H \in S^k$ which maximize

$$p(H|D, X, \theta) \propto p(D, X|H, \theta) \cdot p(H|\theta) = p(D, X, H|\theta).$$

The technical challenge is that Equation 6.2 defines a probability function on a high-dimensional space: For k chain points, we have to consider $|S|^k$ different assignments. A brute-force evaluation is clearly infeasible. In general there would be no exact solution for this task. However, for pairwise tree-structured MRFs (and thus for chains in particular), the computation can be performed in polynomial time using the max-product belief propagation algorithm (see Sections 5.1.1 and 5.1.4).

As a representation for assignments, we use the n-dimensional max-marginals ν_i of the distribution p with respect to fixed part positions $\mathbf{x}_i = \mathbf{y}$:

$$\nu_i(\mathbf{y}) = \max_{\substack{\mathbf{x} \in S^k \\ \text{with } \mathbf{x}_i = \mathbf{y}}} p\left(\mathbf{x}_1, .., \mathbf{x}_i, .., \mathbf{x}_k\right) \tag{6.7}$$

This means, all chain constellations that go through $\mathbf{y}$ at node i are considered and the maximum probability value is kept.

The max-product belief propagation algorithm produces exact max-marginals in $\mathcal{O}(kn^2)$ time for k nodes (of a chain graph) and n states (discrete points in S in this case) [YFW01]. We will now first briefly recap how belief propagation is used to compute a single globally optimal solution (Subsec. 6.4.1). Afterwards, we describe a modified algorithm that finds several locally optimal solutions simultaneously (Subsec. 6.4.2). Derivations and theoretical details can be found in Chapter 5. Finally, we describe how we can augment the algorithm to further reduce the $\mathcal{O}(n^2)$ complexity (Subsec. 6.4.3) of the BP algorithm.

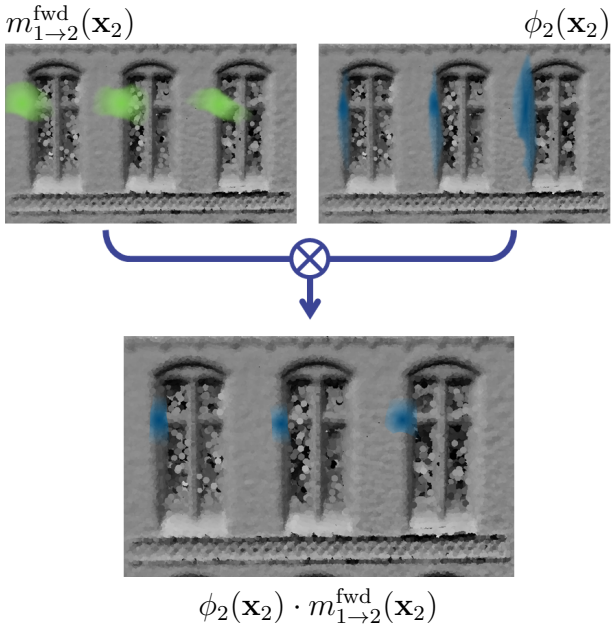

Figure 6.5: *This example shows how the max-marginal distribution for the second chain node is constructed, given the window model example of Figure 6.2. After the message* $m^{\text{fwd}}_{1\to 2}(\mathbf{x}_2) = \max_{\mathbf{x}_1} \phi_1(\mathbf{x}_1)\psi_{1,2}(\mathbf{x}_1, \mathbf{x}_2)$ *from node* 1 *to node* 2 *is computed, the marginal distribution* $\phi_2(\mathbf{x}_2)$ *can be updated to* $\phi_2(\mathbf{x}_2) \cdot m^{\text{fwd}}_{1\to 2}(\mathbf{x}_2)$.

6.4.1 Belief Propagation

Belief propagation on chains works in two passes [YFW01]: First, information is propagated from the start node through all nodes of the chain until the end node. The end node then has gathered enough information to compute the correct max-marginals. In a second pass, information is propagated back to the start to obtain correct max-marginals everywhere. The messages combine knowledge from the evidence term ϕ, the compatibility term ψ and the incoming belief about the state of the node:

$$m^{\text{fwd}}_{i\to i+1}(\mathbf{x}_{i+1}) = \max_{\mathbf{x}_i} \phi_i(\mathbf{x}_i)\psi_i(\mathbf{x}_i, \mathbf{x}_{i+1})m^{\text{fwd}}_{i-1\to i}(\mathbf{x}_i) \tag{6.8}$$

The message passed to the first node is $m^{\text{fwd}}_{0\to 1}(\mathbf{x}_1) = 1$.

The backward messages $m^{bwd}_{i\to i-1}$ are constructed analogously, just passing information in the opposite direction. The max-marginals are afterwards given by

$$\nu_i(\mathbf{x}_i) = \phi_i(\mathbf{x}_i)m^{\text{fwd}}_{i-1\to i}(\mathbf{x}_i)m^{\text{bwd}}_{i+1\to i}(\mathbf{x}_i). \tag{6.9}$$

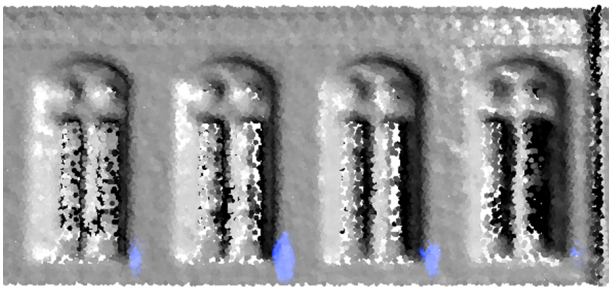

Figure 6.6: *After computation of the max-marginal distribution for the last node of the chain graph, we can extract all spatially local maxima. Those are the starting points for back-tracking.*

If we want to compute the globally optimal solution, we have to perform backtracking on the solution. It can be conveniently combined with the backward message passing, where each step yields the needed correct max-marginal at that node, as follows. We start backtracking from the k-th node and output a maximum likelihood assignment $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$ by choosing the maximum of the max-marginal of node k and then combining this solution with the best match at $k-1$ and then iterating backwards:

$$\mathbf{z}_k = \arg\max_{\mathbf{x}_k} \nu_k(\mathbf{x}_k); \quad \mathbf{z}_i = \arg\max_{\mathbf{x}_i} \phi_i(\mathbf{x}_i)\psi_i(\mathbf{x}_i, \mathbf{z}_{i+1}) \tag{6.10}$$

6.4.2 Computing Several Local Optima Simultaneously

In order to detect a large number of objects having only one output constellation is not enough. We therefore augment the standard algorithm as follows:

Instead of starting backtracking at a single point $\mathbf{z}_k$ (the one with maximum max-marginal probability), we extract a solution for each local maximum of the max-marginal distribution ν_k. The restriction to local maxima is necessary since otherwise (when starting backtracking for all positions $\mathbf{x}_k$ with a $\nu_k(\mathbf{x}_k)$ over a threshold) we would obtain very similar instances with just minimally perturbed positions of the last node. Therefore, we only accept maxima that are the largest value within a window of radius $2r_\phi$ around each point (where r_ϕ is again the descriptor radius). We then start backtracking at each of these solutions. This will extract all at most $\mathcal{O}(n/r_\phi^2)$ locally optimal solutions for which the distance at node $\mathbf{x}_k$ is at least $2r_\phi$.

In cases where solutions overlap at the end node, it is possible to miss locally optimal assignments that could diverge in two or more different directions (think of detecting the boundary of a single leaf of a four leaf clover; only one object instance will be found if the center is the end node). We can fix this problem by starting the local maximum backtracking for each node of the chain (after computing the respective max-marginals). Then we obtain an arrangement of overlaid chain instances that can be combined arbitrarily by cutting segments between crossings of the chains and recombining. In practice, however, such situations are rare; we did not observe this ambiguity in experiments. We therefore restrict the practical implementation to a single backtracking pass for the max-marginal distribution of the last chain node.

6.4.3 Reducing the complexity

So far, the algorithm is too slow to achieve interactive detection times in practice: In order to compute a single message in Equation 6.8, we have to consider all states of node $\mathbf{x}_{i+1}$ and for each we maximize over all possible states $\mathbf{x}_i$, requiring $\mathcal{O}(n^2)$ computations where n is the number of discretization points, i.e. the number of samples in S representing the states of each chain node (typically in the range of millions). This has to be repeated at least $k-1$ times to compute ν_k.

The computational effort can be reduced significantly by taking the properties of the potentials ϕ and ψ into account. We know that ψ_i describes a chain segment originating at position $\mathbf{x}_i$ with a length that is constrained by a Gaussian distribution. In addition, the descriptors $\mathbf{d}_i$ are typically non-matching for large portions of the data.

We first replace the Gaussian distributions ϕ_i, ψ_i by truncated versions, were we clamp the value to zero if the score falls below 5% of the maximum. This has the additional advantage of avoiding spurious local extrema of very low score so that we can run the previously described peak extraction algorithm without need for an additional threshold value to filter out promising local extrema.

Given this truncated potentials, we can now restrict the message passing computations: First, in the loop over possible positions $\mathbf{x}_i \in S$, we only consider points for which the messages computed so far are non-zero. In the first iteration, we consider the evidence term ϕ_1 to make this decision. Secondly, we do not try to combine it with all points $\mathbf{x}_{i+1} \in S$ to evaluate

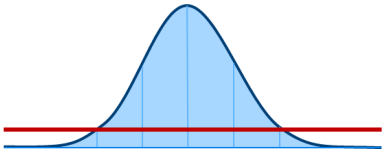

Figure 6.7: *We prune all points which yield a score of less than* 0.05*. Thus, the Gaussian potentials are truncated at* 2σ*.*

$\psi(\mathbf{x}_i, \mathbf{x}_{i+1})$ but restrict ourselves to points $\mathbf{x}_{i+1}$ for which $\psi(\mathbf{x}_i, \mathbf{x}_{i+1})$ is potentially non-zero. In order to estimate this, we compute the maximum length l_{max} of the distance $\|\mathbf{x}_i - \mathbf{x}_{i+1}\|$ for which $\psi(\mathbf{x}_i, \mathbf{x}_{i+1})$ is non-zero. This can be achieved by looking at the covariance matrix Σ_i^{ψ} (which correlates angels and distances) and extracting the maximum variance in distance direction. Having l_{max}, we now extract only points $\mathbf{x}_{i+1}$ from S that are located within a sphere of radius l_{max} around $\mathbf{x}_i$. We retrieve these points efficiently by using a hierarchical range query. For this, we precompute an octree for the points S. During runtime, we traverse the octree top down, only following boxes that overlap with the spherical range and outputting the points in the leaf nodes that are within the range. In typical applications, the maximum radius l_{max} is very small in comparison to the size of the complete scene so that we get very significant speedups by this optimization.

6.5 Results and Implementation

We have implemented the described object detection framework in plain, single threaded C++ and performed experiments on an Intel Core-2 Quad 2.4GHz PC with 8GB main memory. In order to realistically evaluate the performance in practice, we applied the technique to a number of LIDAR range scans (and two synthetic test scenes, Figure 6.18). We used the raw data without any preprocessing other than a simple sub-sampling for the very dense point clouds. In particular, no smoothing, hole-filling, or outlier detection has been performed. Four of the benchmark data sets are taken from the well-known Hannover city scan collection (available online at http://www.ikg.uni-hannover.de). In addition, we used a scan of the "Ludwigskirche", a baroque church from the 18th century located in Saarbrücken, provided by the LKVK Saarland. The LIDAR data suffers from significant noise artifacts. In particular, reflective surfaces such as windows create local clouds of structured, non-Gaussian noise artifacts. The scan quality of the

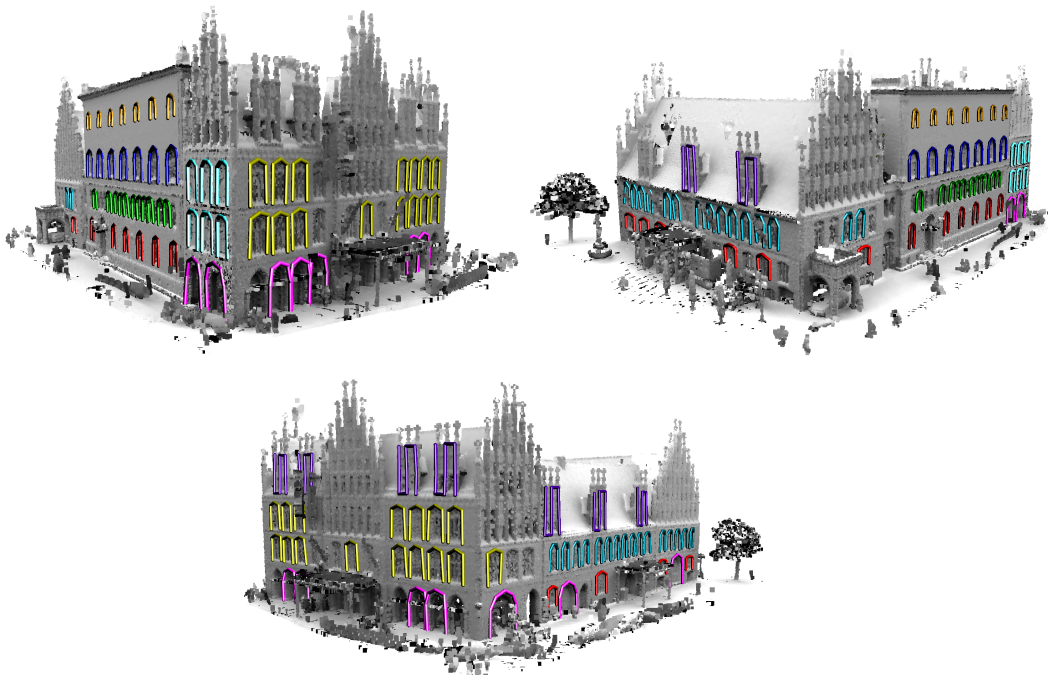

Figure 6.8: *Multi-class learning: Decomposing the old town hall into partially symmetric elements. Up to two examples are used per class.*

church data set is slightly better, probably due to more modern acquisition equipment.

In the following, we conduct a number of experiments: For each data set, we have annotated structural elements such as windows, and the different types of these. Afterwards, we use our algorithm to identify the learned classes. We perform different types of experiments: In multiple class learning, the objective is to find elements of different categories. In single class learning, we try to build a single general class for semantically similar objects. In a third experiment type, we perform multiple class learning and restrict ourselves to only a single example per category to study how far we can get with an absolute minimum of user supervision. Further, we study the effect of using truncated potentials for pruning during inference.

Multi-class Learning: Figure 6.8 shows the result of a multi-class learning experiment performed on the Hannover "old town hall" data set. We train separate object classes for several types of windows, the small roof towers,

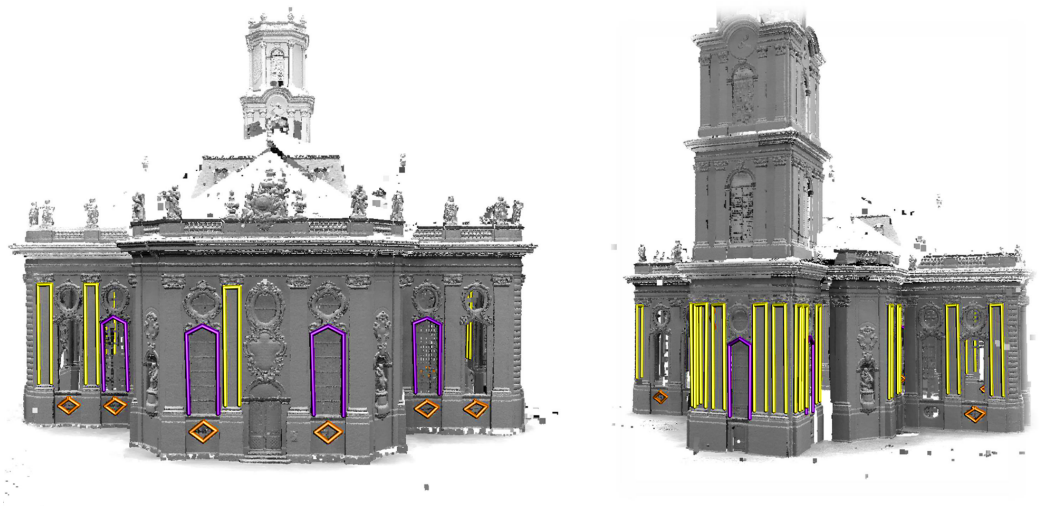

Figure 6.9: *Learning classes by using just one example each.*

and archways. The detected structures include rounded, pointy, and even complex shapes. For each class, we use at most two examples, sketched by the user directly onto the model. By setting conservative global variance parameters per class, we can avoid any false positives: No wrong matches are present and members of different classes are not mixed up. At the same time, we obtain a false negative rate of 24% (32 out of 133). Unrecognized elements include several severely distorted pieces such as elements with large scale acquisition holes or strong clutter due to outliers.

We repeat the multi-class experiment on the new town hall data set, obtaining roughly comparable results (see Figure 6.10).

To illustrate the robustness of our approach, we perform another multi-class experiment on the museum data set (Figure 6.11). In this case, we use exactly the same isotropic noise parameters for all classes, nevertheless obtaining good matching results. Figure 6.9 shows the results for the baroque church. Here, we perform multi-class learning with only a single instance specified per class. This reduces the recognition results a bit, but we nevertheless retrieve more than 70% correctly and do not observe false negatives. Figure 6.1 shows results of another interactive session where 2 examples per class are used (the round windows marked in green use three examples capturing different types of shape interior: empty, solid and grid-structured).

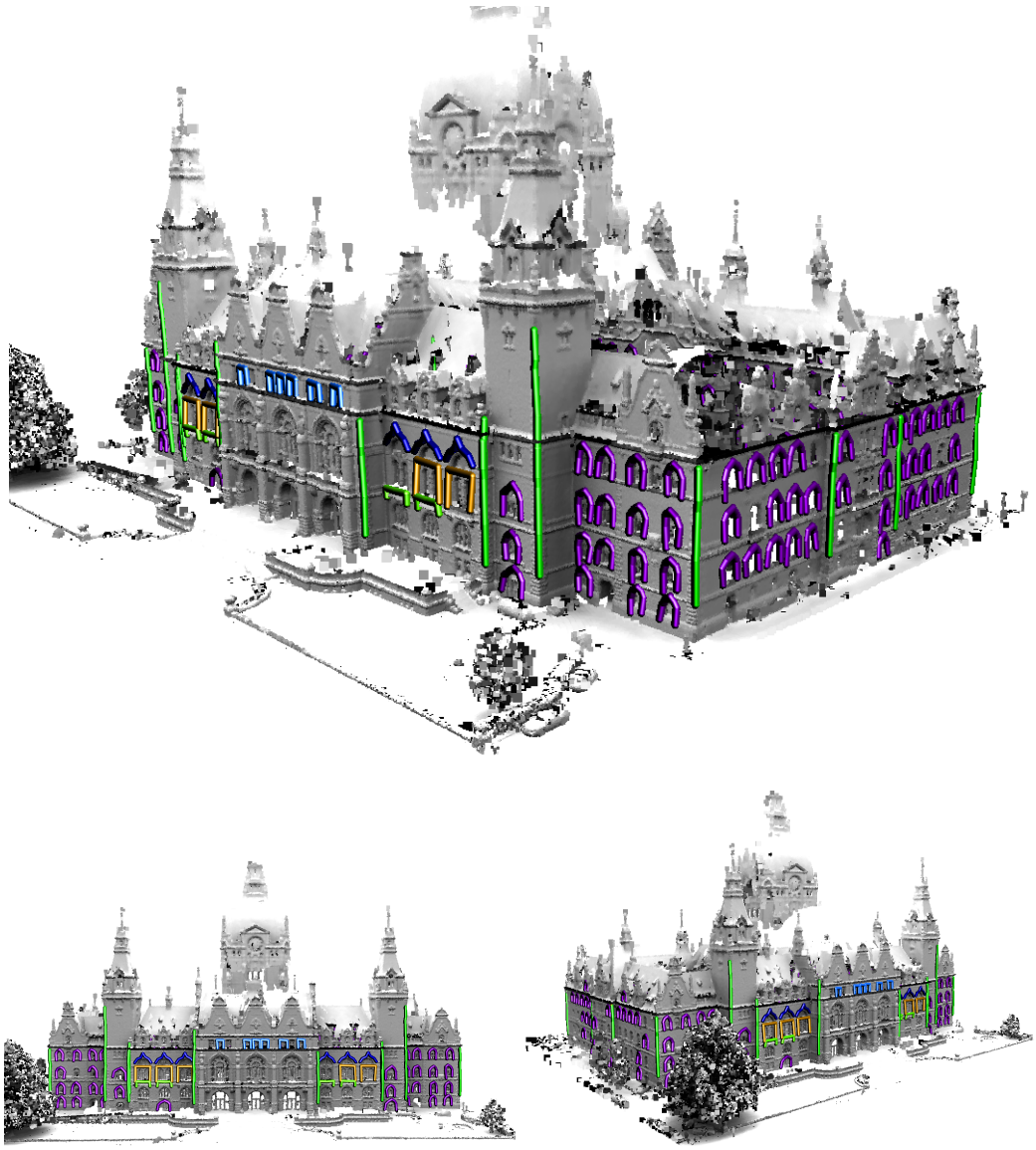

Figure 6.10: *Decomposition into several classes.*

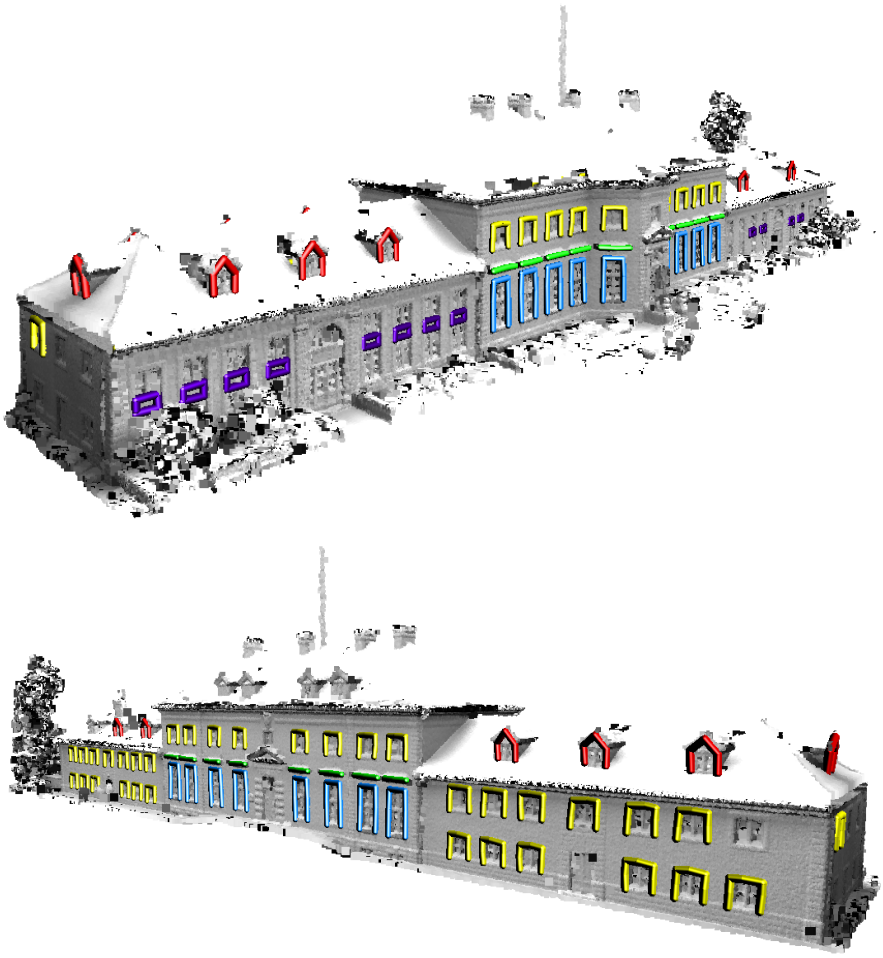

Figure 6.11: *5 different classes extracted. Same global parameters for each class. Two training instances clicked per class.*

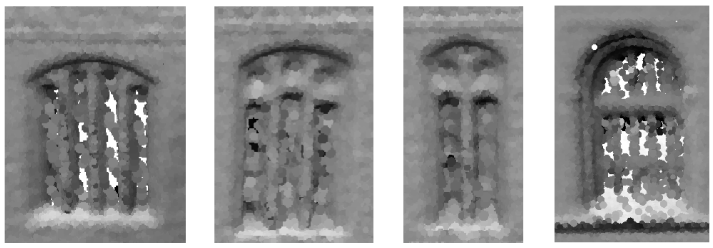

Figure 6.12: *Different types of windows recognized as one class in the experiment shown in Figure 6.13.*

Figure 6.13: *Single-class learning: Detecting "all windows". Training data (windows with blue part positions) consists of 4 exemplar windows. Figure 6.12 shows a closeup of different window types detected by this class. In the example of Figure 6.8 these types have been trained individually.*

Single-class Learning: Next, we perform a single-class learning experiment, where a set of windows are comprised in one class. The results are shown in Figure 6.13 and the variation in the input is illustrated in Figure 6.12. In this case, we recognize all but 7 out of 84 windows in the model. Out of the 7, three contain large holes such that a recognition is obviously impossible. Again, false negatives can be avoided. In order to obtain good results, it is important to specify a large enough variance of the learned model. It is obviously problematic to estimate the variance from only very few training examples. Therefore, we have used the option to scale up the learned variance of descriptors as well as spatial layout. Figure 6.14 shows the difference between using unmodified and scaled values. We obtain much higher recognition rates without introducing false positives even if we scale up the standard deviation significantly, for both the trained shape model and the descriptors. Consequently, we used these settings for all examples.

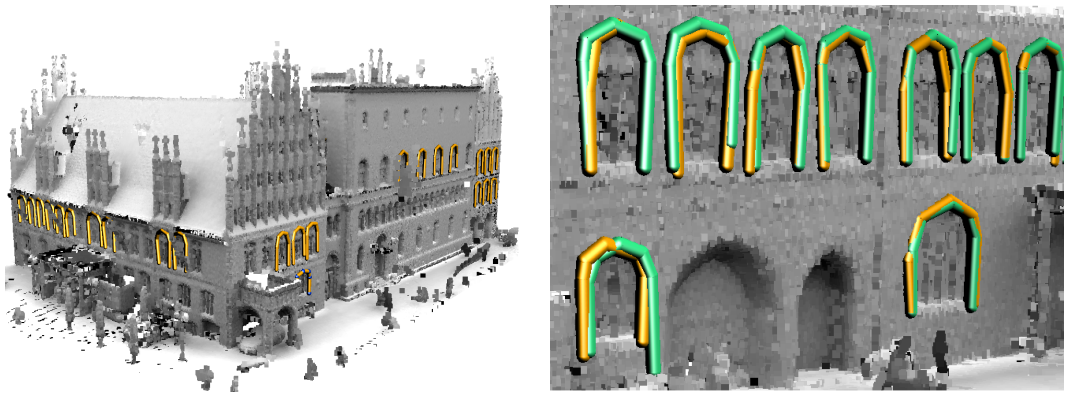

Figure 6.14: *Single-class learning: Detecting "all windows". Training data (blue spheres) consists of only 4 exemplar windows. Left: Without scaling the learned covariance matrices the results deteriorate compared to Figure 6.13. Right: Detail of overlay: green (unscaled), yellow (scaled). When using scaled versions of the learned covariance matrices, the extracted instances adopt better to the underlying geometry. Without scaling the noise level exceeds learned correlations.*

Effect of pruning: We study the effect of the truncated potentials by comparing the marginal distributions of marginals and max-marginals for succeeding inference steps. Similar to the previous experiments we model a window, with first chain node representing the lower left corner (red) and last chain node representing the lower right corner (blue). Like Figure 6.15 shows, the state space of the max-marginals decreases rapidly during inference. Already the first truncated potential function limits the state space such that we can expect run times which are significantly lower than the theoretical $O(kn^2)$, if n is the number of states per node.

Running time: As most descriptor-based shape analysis techniques, our technique needs some time to precompute the descriptors. This has to be done only once per data set. Preprocessing times are normally in the range of a few minutes (Table 6.1). Please note that this is an embarrassingly parallel task that can probably be sped up significantly by an optimized implementation, but this is not the focus of this work. The interaction is much faster. Training a model is very fast but some auxiliary tasks such as estimating the global descriptor variance can take up to 5 seconds in our examples. Finding all object instances is usually also interactive, in the range from below one to approximately 10 seconds. However, it is possible to create

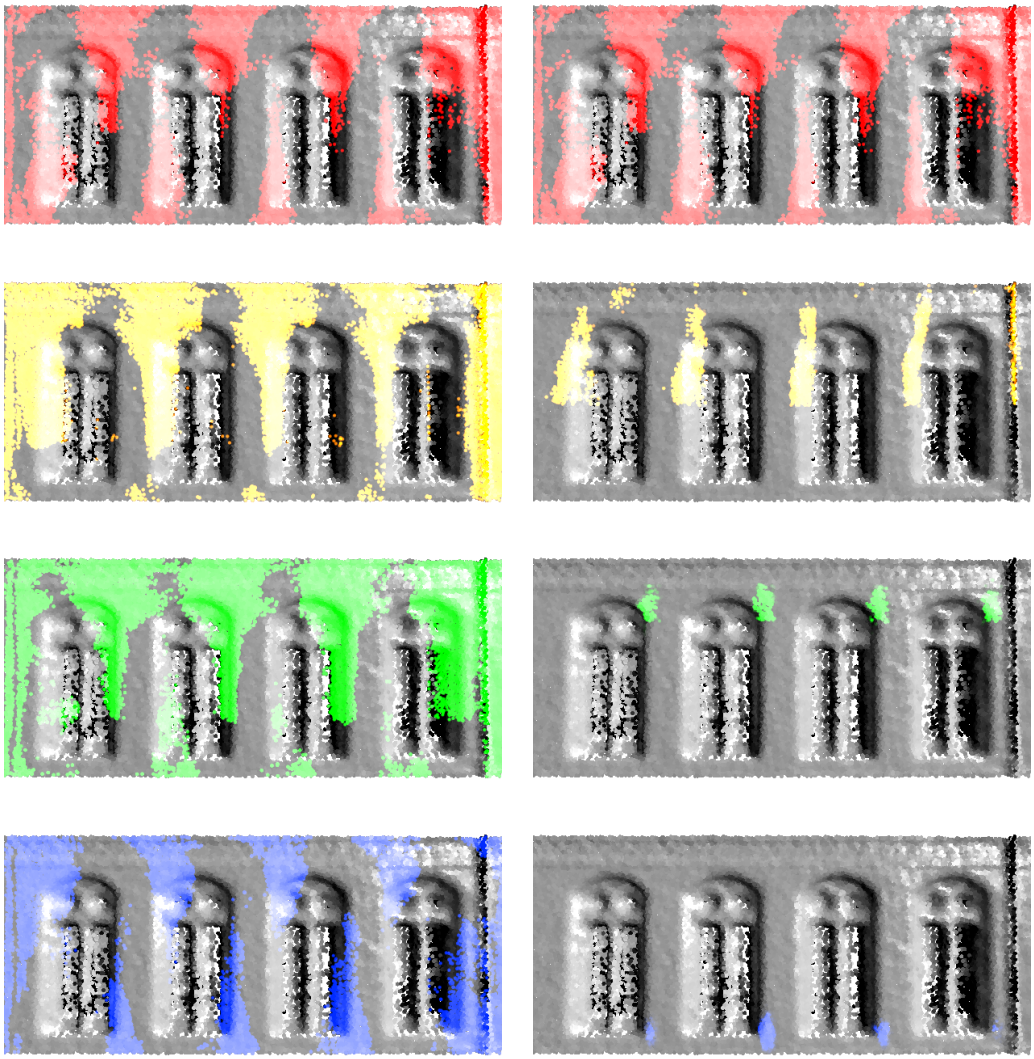

Figure 6.15: *Distribution of the max-marginals during inference with the truncated potentials. Left: Truncated potentials ϕ_i. Right: Max-marginals $\phi_i \cdot m^{\text{fwd}}_{i-1\to i}$. The model is defined in the sense of Figure 6.2. From top to down: The state space of the max-marginals decreases rapidly during inference.*

model	#points	#preproc. [s]
old town hall	744,567	516
new town hall	1,271,777	908
museum	2,119,367	3,741
baroque church	2,618,385	2,432
stars	313,236	326
relief	289,509	336
street (training/all)	198,767/1,056,158	436/999

Table 6.1: *Precomputation times for the example scenes: timings are measured for descriptor radius* 0.01.

"broken" training models for which the algorithm takes a very long time (up to 20 minutes) to finish. This happens for example if object definitions are sketched on non-descriptive, planar regions where pruning cannot be effectively performed. For practical applications, these degenerate cases are not relevant as no reasonable matching can be expected.

6.6 Discussion

In any detection task, there is an inherent trade-off between generalization performance and precision. Our model aims at learning robustly from very few examples, which limits the precision for very large training sets. Figure 6.13 shows that, nevertheless, quite some variance can be captured. And for very different geometry, the user can anytime create multiple object classes with parts modeling more complex mixtures.

An important point is a comparison with models of different topologies, e.g. star-like topologies as implemented by implicit shape models. It will be discussed in Chapter 7, There we will implement a fully connected graphical model and conduct experiments which compare the performance of star- and chain topologies. The results indicate that chain topologies outperform star topologies (at least in the setting of the experiments) .

Model Transfer: In Figure 6.16 we evaluate the ability to transfer object models from one data set to another, depicting a row of different houses from one street in the Hannover data set: Only the small part of one house

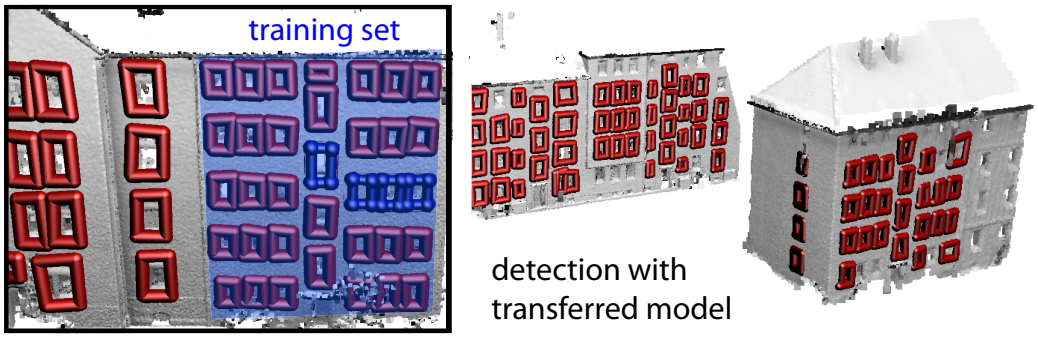

Figure 6.16: *Transferring learned models: Four exemplars were trained on the blue area, detections on the same and on a different data set are marked red.*

marked in blue is used for initial learning (4 windows). The model is stored and reused on the second independent data set. A large fraction of windows in the other houses is detected. Of course, this is limited to rather similar geometry. It is impossible to infer very different geometry that is not within the span of the example space.

Sensitivity to User Input: Figure 6.17 illustrates various quality levels of user input. For a single input instance, accuracy is not an issue and all matched objects will be detected with the same imperfect shape. Only when combining several instances in a single model, accuracy plays a role. Nonetheless, potential imprecision is implicitly handled by the model, since it accounts already for inaccurate surface descriptions. Further, as the feedback is almost immediate, it is easy to add further data only when needed or to see and correct the negative impact of an imprecise input. Figure 6.17 illustrates the effect of different levels of noise in the specification of two training instances. It turned out that we had to deliberately click off target to obtain artifacts.

The bad-input example of Figure 6.17 also reflects that we do not explicitly impose closed chains. In principle, it is possible to formulate the belief propagation algorithm for closed chains, but the computation cost would increase by a polynomial factor. The choice of a cycle-free chain allows us to perform a very efficient search procedure. Most matched instances are almost closed because the descriptors snap them to the right locations during the search. We could easily add a culling step to reject chains that are not 'closed enough', but, in practice, this was not necessary. In the examples, only the

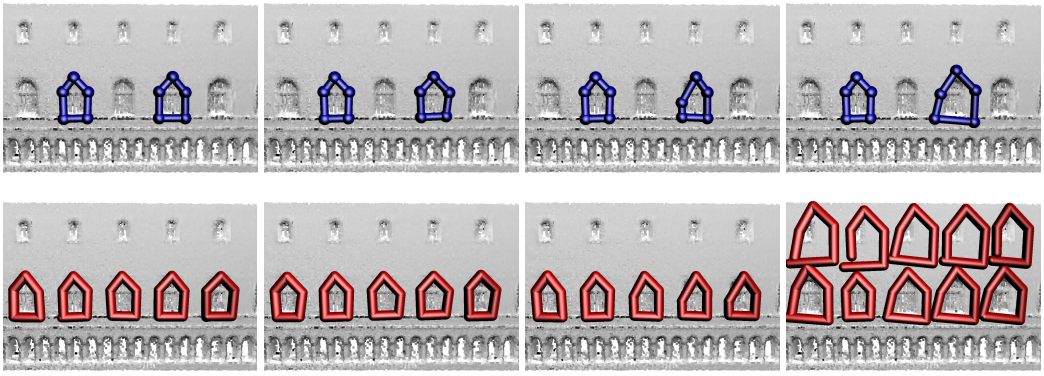

Figure 6.17: *The method is robust to imprecision in user input (left, center), even for very noisy input, the model still finds reasonable matchings (right). The top row shows the noisy training samples, the bottom row the detection results. Data set: old town hall.*

unrealistically bad user input benefits from this.

Scale Invariance: The model is not necessarily scale invariant: Only if the user provides examples with scale differences, the linear model will automatically span the whole subspace of scaled examples. However, the current descriptors are not scale-invariant, thereby limiting the scale range; a design choice to make the detection more reliable. We could achieve full scale-invariance using scale space descriptors such as SIFT [Low03, LG05]. Still, our solution is flexible enough to handle significant variations (Figure 6.18). By learning different scales its impact in the detection step is reduced (although we still use scale-dependent descriptors). The algorithm finds even instances that were not directly trained, but it fails for strong scale differences which is desired behavior.

Object Alignment We qualitatively evaluated the correspondences established with detected instances for their usability in a object alignment experiment. Therefore, we roughly designed simple primitives for 3D objects which were to be aligned with their corresponding instances in a point cloud. We defined 4 different classes of windows in a point cloud. For each class we manually established correspondences between the parts of the models and their corresponding 3D primitives. These primitives have been hand-crafted with editing tools such as *3D Studio Max*. After detecting all windows in

Figure 6.18: *Although the descriptors are not scale invariant to increase precision, our method is scale tolerant (green) when learned (red) and supports complex layouts (synthetic data sets; top: stars, bottom: embossed shapes).*

the point cloud we applied rigid transformations [Sor09] to the primitives in order to align them with the corresponding instances in the point cloud. The results are presented in Figure 6.19.

The detected window instances show up small variations due to variations in the point cloud, but the simple 3D primitives can be aligned to point cloud quite well. This experiment is just a simple example which indicates future applications. Detecting correspondences allows for creating super resolution models of point clouds which comprises repetitive shapes that are not necessarily aligned to a grid. Another application might be to generate reduced versions of a scan such that it can be decomposed into simpler 3D objects than the ones found in the point cloud. It is also interesting to further improve this idea: Instead of fitting static objects to a scene one might use generative representations in the sense of morphable models [BV99, HSS+09] such that the scene can be reconstructed in a most flexible way. Then applications for perfectly filling holes in scans or even morphing parts of a scene come into range.

Limitations: The user-guided detection scheme has currently some obvious limitations.

A major limitation is the topology structure of the model: We can only find chain-like object models, we cannot find complex graphs with cycles. These have to be composed out of several, independently-trained chain models. Further, the pairwise dependencies of a Markov chain model do not allow Belief Propagation to infer marginals which are consistent to all previously assigned marginals. For example, assume we are given an object model which

Figure 6.19: *The detected object correspondences are used to align synthetic 3D models with detected objects of different (color coded) categories. The upper left shows the training examples and the matching primitives. The lower left shows all detected shapes. The right side shows the aligned primitives.*

comprises scaled versions of one object instance. When computing the last inference steps we cannot decide whether the first matched parts were assigned for a small scale instance or a big one. The decision only depends on the direct neighbor-part and we possibly yield an inconsistent constellation of parts. In this context, an extension to construct more general, tree-structured graphical models is possible. In Chapter 7 we examine more complex graphs: fully correlated models.

Furthermore, the current approach assumes a fixed upward direction, mostly motivated by applications to architectural models and similar man-made structures. This is no principal limitation. However, a straightforward solution to this problem (such as evaluating the match for different base orientations covering 360° degrees) would increase the runtime. We will approach this limitation in Chapter 7, too.

One further limitation we have observed in practice is that the model is not robust to missing data; if an object part maps to a hole in the surface it cannot be detected because no discretization points are available (using a robust ϕ model is therefore not sufficient).

6.7 Summary

We have presented a technique that learns statistical models for semantic similar shapes from user input. It is designed such that it can handle scenarios with only few training examples. Object detection can be performed at interactive run times and yields robust results with only few false positives and negatives on scanner data suffering from significant noise artifacts. The presented approach is conceptually straightforward and easy to implement. We can not only simultaneously detect all instances of an object class, we also yield semantic correspondences between those instances which qualifies the technique for a wider range of applications.

The weak form of user supervision for this approach is a powerful tool for object detection problems, in particular, if care is taken to minimize the necessary work for the user. Nonetheless, even a small user interaction makes matching significantly more robust.

Similar results in accuracy and robustness are hard to achieve with unsupervised techniques. Only setting up their parameters often takes more time than to specify a few object parts with our solution. Also, a supervised approach can learn broader classes of *semantically* similar objects, which is a very hard problem without user input. Consequently, user guidance offers a viable way to higher-level "shape understanding", approaching semantic rather than purely geometric interpretations.

Since there is no prior work in the field of 3D object detection which also produces reliable correspondences for parts, the approach described in this chapter can be seen as a first step into this broad but little explored venue of geometry processing.

7

A Correlated Parts Model for 3D Object Detection

Problems worthy of attack
prove their worth by fighting back.

Paul Erdos (1913 - 1996)

This chapter addresses the problem of detecting objects in 3D scans according to semantic object classes learned from sparse user annotation. Extending the model presented in Chapter 6, we model objects belonging to a class by a set of fully correlated parts. We encode dependencies between local shapes of different parts as well as their relative spatial arrangement. For an efficient and comprehensive retrieval of instances belonging to a class of interest, we introduce a new approximate inference scheme and a corresponding planning procedure. Further, we show how to extend the technique to hierarchical composite structures, thus reducing training effort. Finally, we evaluate the method on a number of real-world 3D scans and demonstrate its benefits as well as the performance of the approximate inference algorithm.

The work this chapter is based on has been published at Eurographics 2013 [SJWS13]. With permission of the co-authors main text passages and figures are utilized without tagging them individually.

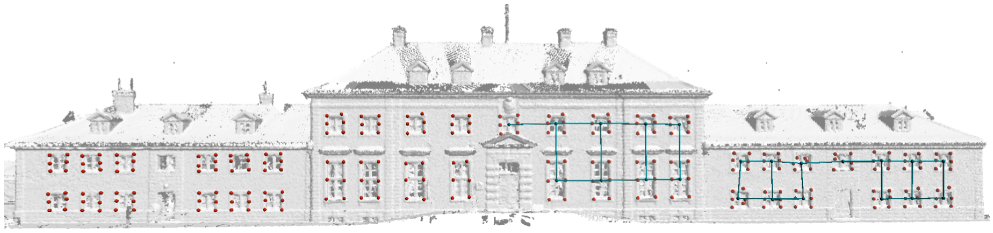

Figure 7.1: *Detection results for the Wilihelm Busch Museum in Hannover. A general object class (6 parts) for all windows is trained. Additionally a composite model detects arrangements of windows (overlayed on the right side).*

7.1 Introduction

We address the problem of detecting object instances (*shapes*) according to semantic object classes. Shapes are defined as a set of distinctive parts describing local geometry (*local shape*) as well as the spatial layout (*constellation*) of these parts. From a small number of hand-annotated examples, a part-based shape model is derived to retrieve large quantities of further instances, including their correspondences (see Figure 7.2). Our goal is to find a general model which also overcomes limitations of the Markov chain approach presented in Chapter 6. There, a major limitation is that a simple Markov chain model does not encode part relations bigger than pairwise. That means, the pairwise dependencies of a chain do not allow the message passing algorithm to 'remember' all previously passed messages. For a category of shapes, which comprises large spatial variations, it is hard to consider only those parts (during inference) that generate a constellation which is semantically consistent. For example during training we might have observed scaled instances of one single instance. The pairwise dependencies would allow to detect a constellation that blends small and big instances. The same holds for local shapes. The chain model presented in the previous chapter does not even comprise pairwise dependencies of neighboring parts' local shape descriptions.

In general, detecting constellations of parts in 3D geometry poses a number of unique challenges and opportunities. In Chapter 6 we have seen a simple but efficient model which shows how to approach two important issues:

Semantic symmetry: In a large 3D scan, a large number of instances of the same object class such as cars or windows show up simultaneously. Thus

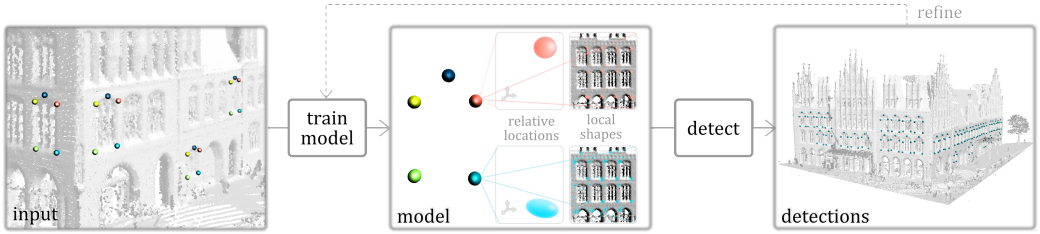

Figure 7.2: *Outline of the method. Given sparse user annotations defining some shapes and their correlations, a shape model characterizing these shapes is constructed. Each node in the model is associated with corresponding distributions for the local shapes (descriptors) and the relative location within the shape. The efficient detection allows for refining the model interactively.*

it is important to detect all instances in a single pass.

Correspondences: In 3D computer graphics, detecting semantic correspondences can be an intermediate step towards building generative models such as morphable shape spaces. Therefore, obtaining accurate and consistent part correspondences is desirable. A chain model is a simple solution to that. However, higher accuracy and semantic consistency requires more complex dependencies. Using fully correlated parts improves both.

In this chapter we will in addition approach the following challenges:

Frame invariance: Detection should be invariant under translations and rotations. In contrast to Chapter 6, we will show how to model full rotational invariance.

Structuring scenes: Objects can be arbitrarily complex. Parts themselves might be better modeled by sub-parts. Thus, an extension of the model to a hierarchical version is helpful. Composite models can learn higher order co-occurrence patterns to structure the input.

Correlations: The local shape of object parts is typically strongly correlated, e.g. for window- or car models we want to not only ensure that parts are aligned rectangular, they should also share the same visual appearance. These general correlations have not been captured yet by previous work.

Efficiency: Since full correlations are modeled with a fully connected graph, it can not be inferred exactly. That poses the demand for an efficient approximation scheme.

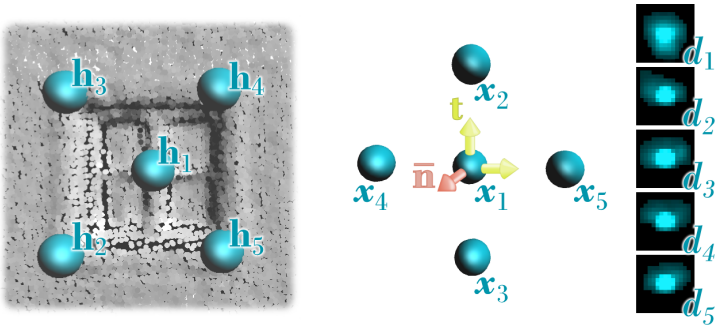

Figure 7.3: *Each shape H is associated with a spatial layout X and local shapes $\mathbf{d}_i$. The layout is given by relative coordinates $\mathbf{x}_i$ of the individual parts $\mathbf{h}_i$ in a coordinate frame centered at the first part. The local shape is a collection of the local shape descriptors of the shape parts.*

We analyze the method empirically on a number of benchmarks, evaluating the detection performance as well as the run-time costs. This includes a large city scan with 4GB of input data, for which training is interactive and detecting all instances of an object class takes less than 2 min, using single-threaded, unoptimized C++ code.

7.2 Shape Model

Each shape model characterizes an object class by encoding similarities of shapes from this class. As illustrated in Figure 7.3, we define shapes as a set of correlated parts. Each part i consists of a relative position $\mathbf{x}_i$ and the local shape description $\mathbf{d}_i$. The individual parts are subsumed into local shape D and their overall spatial layout X. The shape model $\theta = (\theta_D, \theta_X)$ is then defined by Gaussian models $\theta_D \sim \mathcal{N}(\mu_D, \Sigma_D)$ and $\theta_X \sim \mathcal{N}(\mu_X, \Sigma_X)$ over D and X, thus encoding dependencies (*correlations*) of the individual parts.

7.2.1 Probabilistic Model

In the following, let $\mathcal{S} \subset \mathbb{R}^3$ be a smooth manifold embedded in three-space. We use $\mathbf{n}(\mathbf{x})$ to denote the surface normal at point $\mathbf{x} \in \mathcal{S}$. Typically $\mathcal{S}$ is represented by a sampled approximation (*point cloud*) $S = \{\mathbf{s}_1, ..., \mathbf{s}_n\}, \mathbf{s}_i \in$

$\mathbb{R}^3$, acquired by 3D scanners and thus subject to noise artifacts.

Given a shape model θ with k parts, the goal is to find reasonable assignments of the k parts to points in S: $H = (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_k)$, where $\mathbf{h}_i \in S$ denotes the position of part i. The detection problem can be formulated as a maximum a-posteriori hypothesis search over the joint posterior distribution of H and manifold evidence S. Hence, the objective is to maximize the probability $p(D, X, H|\theta)$. The model is designed such that local shape D and layout X behave independently of each other. Hence, we can infer:

$$\begin{aligned} p(D, X, H|\theta) &= p(D, X|H, \theta) \cdot p(H|\theta) && (7.1) \\ &= \underbrace{p(D|H, \theta_D)}_{Local\ Shape} \cdot \underbrace{p(X|H, \theta_X)}_{Layout} \cdot \underbrace{p(H|\theta)}_{Prior} && (7.2) \end{aligned}$$

It consists of a term accounting for the descriptors (local shape) and their constellation in 3D (layout). In addition, a prior distribution $p(H|\theta)$ can be used to model additional constraints on the detection. In the experiments, it is assumed to be uniform over $\mathcal{S}$.

Spatial Layout

Since Gaussian models cannot represent rotations well, we encode the spatial layout X relative to a local coordinate frame $\mathbf{T}$. The reference frame $\mathbf{T}$ is spanned by the first two parts $\mathbf{h}_1, \mathbf{h}_2$ and the smoothed surface normal $\overline{\mathbf{n}}$ at $\mathbf{h}_1$, as illustrated in Figure 7.3. For reasons of robustness the normals are smoothed over the whole region the local part covers. We compute a tangent vector $\mathbf{t} = (\mathbf{h}_2 - \mathbf{h}_1) \times \overline{\mathbf{n}}$ and set $\mathbf{T} = (\overline{\mathbf{n}} \times \mathbf{t}\ ,\ \mathbf{t}\ ,\ \overline{\mathbf{n}})$.

The $3(k-1)$-dimensional layout vector $X(H)$ is then given by:

$$X(H) = (\mathbf{T}(\mathbf{h}_2 - \mathbf{h}_1), \mathbf{T}(\mathbf{h}_3 - \mathbf{h}_1), ..., \mathbf{T}(\mathbf{h}_k - \mathbf{h}_1)) \qquad (7.3)$$

After this (orientation-) normalization, we model the spatial layout of model parts as a joint Gaussian distribution over the relative coordinates of $X(H)$, with mean μ_X and covariance Σ_X:

$$p(X|H, \theta_X) \sim \exp(-\frac{1}{2}(X(H) - \mu_X)^T \Sigma_X^{-1} (X(H) - \mu_X)) \qquad (7.4)$$

Local Shape

Local shape D is modeled by a joint Gaussian function on $(d \cdot k)$-dimensional vectors $D(H)$ composed of all k d-dimensional local shape descriptors:

$$p(D|H, \theta_D) \sim \exp(-\frac{1}{2}(D(H) - \mu_D)^T \Sigma_D^{-1}(D(H) - \mu_D)) \tag{7.5}$$

Again, μ_D represents the (learned) mean and Σ_D the covariance of the descriptors, again including all cross-correlations between the local shapes captured by descriptors of all of the different parts.

The model parameters are the mean μ_D and covariance matrix Σ_D. The framework is independent of the actually chosen shape descriptor. Any function $S \rightarrow \mathbb{R}^d$ can be used. The descriptors used in the experiments are described in Section 7.4.1.

7.2.2 Learning

In our scenario the model is manually defined: For a number of shapes the user labels all correspondences between the shapes – thus defining the parts. The ability to manually select meaningful parts is important for many applications that can use these coarse correspondences as input, like replacing found objects with a template. Whereas, parts automatically chosen by some objective function might not be the ones desired by the user.

The model parameters $\theta = (\mu_D, \Sigma_D, \mu_X, \Sigma_X)$ are learned by supervised training. This is done by specifying a sparse set of corresponding points on objects of interests. The correspondences are defined using the same sketching technique as in Chapter 6. Given the training instances, mean and covariance are estimated for both local shape and spatial layout.

For small sets of training instances, the learned covariance matrices are rank deficient. For example, a principal component analysis (PCA) extracts at most a 3-dimensional space from 4 examples. Hence, the variability is underestimated, and whole subspaces are falsely assumed to be noise-free in such cases. Even if in theory the actual class is described sufficiently by a few dimensions, inaccuracies such as scanner noise typically make a covariance of full rank inevitable in practice. We model these unmeasured effects by a uniform Gaussian noise model, i.e. by adding $\lambda \mathbf{I}$ to the covariance matrix, where λ is a user-controllable parameter.

In certain scenarios it might also be necessary to amplify (or attenuate) the variations learned from the training data. For instance, if we want to detect objects of different sizes but have only observed two sizes so far. Similar to Chapter 6 this is implemented by scaling the observed covariances.

The final covariances are given by:

$$\Sigma_X = \gamma_X \Sigma_X^{obs.} + \lambda_X \mathbf{I} \quad \text{and} \quad \Sigma_D = \gamma_D \Sigma_D^{obs.} + \lambda_D \mathbf{I}.$$

7.2.3 Hierarchical Shape Models

We propose a simple extension of the fully correlated model, which allows the detection to be performed hierarchically, using detected constellations as parts of higher level constellation models.

A weakly related idea is modeled in [DPP09]. They present a tree-structured graphical model to decompose objects into a hierarchical constellation of parts with simple descriptors for the leave parts. The potential functions are represented in a non-parametric form. Hence they need to apply a sampling technique for the detection.

The motivation of the new approach is two-fold: First, many complex shapes can be described by constellations of simpler base shapes. By training part models separately, fewer training examples are required for estimating good model parameters. Secondly, finding constellations of constellations allows us to recognize structural relations between parts (e.g. windows arranged on a regular grid), which permits the extraction of additional information.

As shown in Figure 7.4, the hierarchical extension is straightforward: First, several different base models are constructed as described in the preceding sections. Then, further instances are detected in the model and offered to the user as additional feature points to be selected as parts in composite models. The position is set to the centroid of the found instance and the local shape is obtained by concatenating all local shape descriptors of the base instance (followed by a reduction to the original d dimensions of a local shape via principle component analysis). There is also the possibility to comprise instances of different categories into the composite model. Therefore, the class of the base model is also used to discriminate the feature points (using labels for each class). Thus, during detection of higher stages in the hierarchy, we look only for those parts which correspond to the labels defined in the composite model.

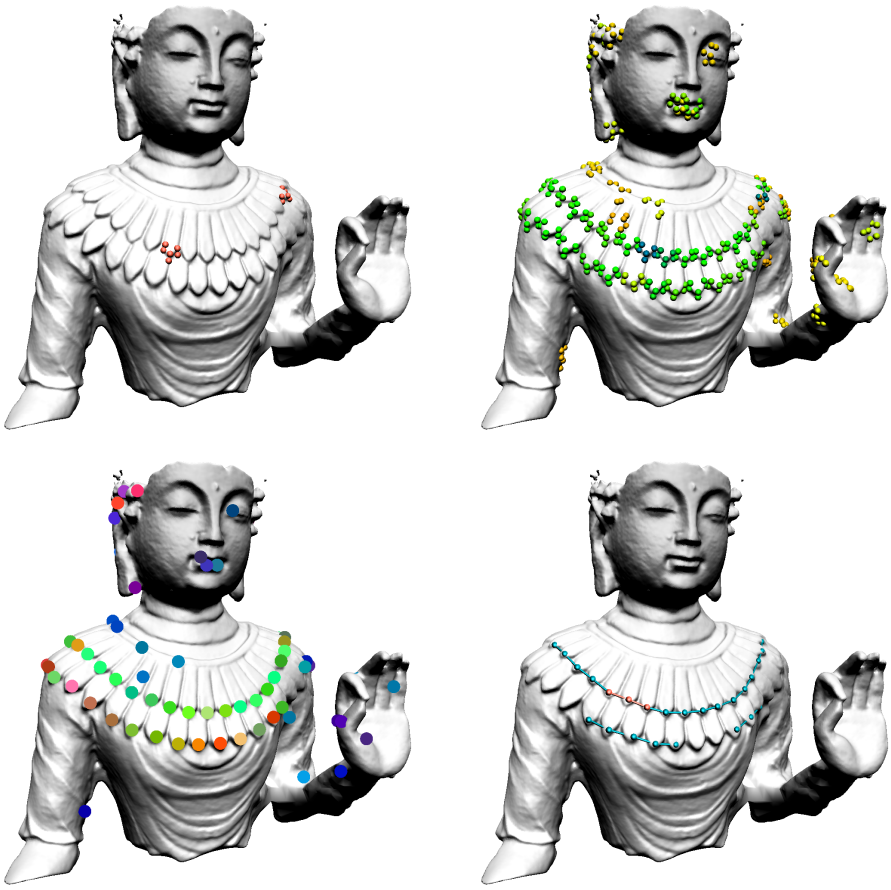

Figure 7.4: *Using sparse user annotations (upper left) a shape model is learned. The detected instances (upper right; colors indicate the quality of the match (blue: perfect; red: bad)) are transformed into descriptors (lower left, similar colors indicate similar descriptors) for the second hierarchy level. Hierarchical detections shown on the lower right are obtained only using the single, red training exemplar.*

7.3 Shape Inference

We now need to find instances of the model defined in the previous section. Given an input point cloud S, we want to retrieve all local maxima with a significant score for $p(H|D, X, \theta)$. Obviously, the log-likelihood of this probability function is non-convex in any non-trivial case.

Traditionally, inference in constellation models is done using Markov-chain Monte-Carlo sampling ($MCMC$) [SGS09] or sometimes expectation maxi-

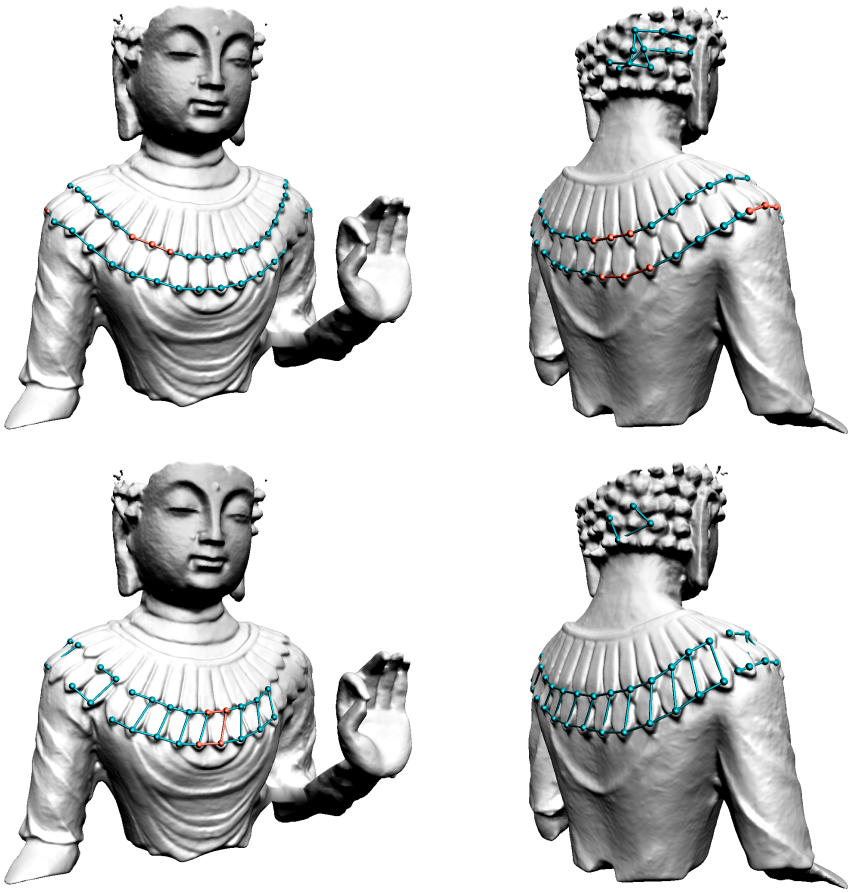

Figure 7.5: *More hierarchical detections for the statue. The hierarchical descriptors shown in Figure 7.4 were used. Instances used to train the model are marked red.*

mization (EM) [RPA03], but these techniques are slow and can only compute one solution at a time. Another option is a restriction to tree-structured models [FH05], allowing for exact inference, or even further restrictions to star [LLS04, VSS12] or chain models like the one presented in Chapter 6.

We use a greedy dynamic programming scheme for approximately retrieving the local maxima of $p(H|D, X, \theta)$. Similar to Chapter 6 (with help of Bayesian inference) we need to find assignments $H \in S^k$ which maximize Equation 7.2. In order to do so efficiently, we use an inference scheme which is related to belief propagation for chain-structured models. However, instead of only considering direct predecessors, we incorporate dependencies to all predecessors. Omitting these dependencies in the model would result

in an ordinary Markov Chain Model which can be inferred exactly, but at the cost of a weaker model. We examine the effect of including these global dependencies in Section 7.4.

Like for chain-structured models, we successively compute sets of partial assignments. Since complete enumeration is not feasible (i.e., of exponential effort), the key idea is to restrict evaluation to those assignments which are likely to be part of an actual instance. A detailed explanation and derivation for the approximate inference is shown in Section 5.2.3.

Given a set of candidate assignments for the first i parts of the model, denoted by $\mathcal{H}_i \subset S^i$, we form augmented assignments for the first $i+1$ model parts. For each possible value $\mathbf{h}_{i+1}$ of part $i+1$, we search for the best partial assignment (in terms of maximizing Equation 7.2) to be combined with $\mathbf{h}_{i+1}$:

$$\mathcal{H}_i(\mathbf{h}_{i+1}) = \underset{(\mathbf{h}'_1,\ldots,\mathbf{h}'_i)\in\mathcal{H}_i}{\arg\max}\; p(\mathbf{h}'_1,\ldots,\mathbf{h}'_i,\mathbf{h}_{i+1}) \tag{7.6}$$

We form the set of candidate assignments $\mathcal{H}_{i+1}$ by combing all $\mathbf{h}_{i+1} \in S$ with their corresponding best matching assignment $\mathcal{H}_i(\mathbf{h}_{i+1})$:

$$\mathcal{H}_{i+1} = \left\{ (\underbrace{\mathbf{h}_1,\ldots,\mathbf{h}_i}_{=\mathcal{H}_i(\mathbf{h}_{i+1})},\mathbf{h}_{i+1}) \middle| \mathbf{h}_{i+1} \in S \right\}; \quad \mathcal{H}_1 = S \tag{7.7}$$

Once the candidate assignments for part k have been computed, we perform a local maxima search to retrieve the final detections.

By just keeping track of the current best estimates, we might lose track of a desired instance in favor of a seemingly better but wrong match. Luckily, real-world data is typically benign, as demonstrated in Section 7.4, since fixing the first few parts imposes substantial restrictions on the remaining ones. Accordingly, we will optimize for the order in which the parts are processed, as detailed in Section 7.3.2.

7.3.1 Efficiency

Even though we have reduced the complexity from exponential (for the naive but exact evaluation) to quadratic costs in $|S|$, the algorithm is still too slow for large, real-world scenes with several million points.

The goal is to retrieve instances with significant probabilities; accordingly we can discard all values $\mathbf{h}_i$ for part i if their local shape or relative position does

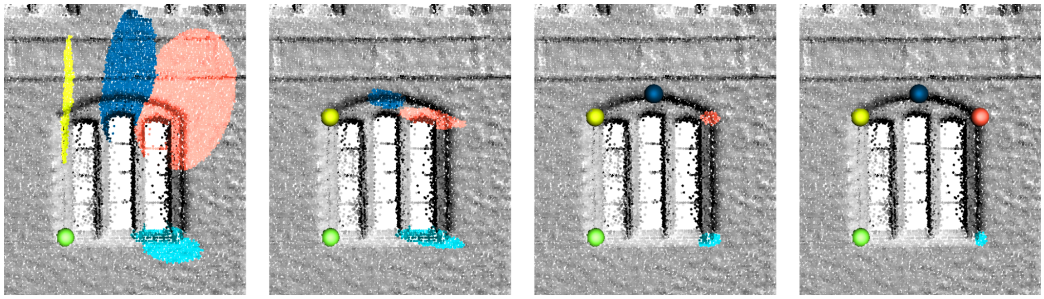

Figure 7.6: *Effect of covariance updates. Potential locations for the remaining parts after fixing preceding parts (for purpose of illustration the orientation of the shape was fixed). Fixing the second part of the shape already decreases the horizontal variance drastically.*

not match the model at all. We regard $\mathbf{h}_i$ as a potential value for part i if its local shape has a Mahalanobis distance of at most 2 to the local mean shape of part i, thus reducing the set of initial candidates for each part drastically, see Figure 7.7.

Similarly, $(\mathbf{h}_1, \ldots, \mathbf{h}_i)$ is only included in the set of candidates $\mathcal{H}_i$ during detection, if $\mathbf{h}_i$ adds at most 2 to the overall Mahalanobis distance to each model mean. In order to prune efficiently, we first need to reduce the search space of potential candiates for h_i: Given the tentative assignment $(\mathbf{h}_1, \ldots, \mathbf{h}_{i-1})$, we compute $p(X|\mathbf{h}_1, \ldots, \mathbf{h}_i, \theta_X)$, i.e. the conditional marginal distribution for the layout of part i. That is a Gaussian distribution, and the corresponding 2σ ellipsoid yields an upper bound for the set of candidates which add at most 2 to the overall Mahalanobis distance. Following the procedure described in Chapter 6 we again use a hierarchical data structure to efficiently extract the candidate positions for h_i.

Another improvement concerns the evaluation of Equation 7.2: The incremental algorithm requires repetitive evaluation for partial assignments $(\mathbf{h}_1, \ldots, \mathbf{h}_i)$. We can reduce computational effort from $O(i^2)$ to $O(1)$ if we update the model using the Schur complement:

$$Y = \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \right)$$

$$Y_2|Y_1 \sim \mathcal{N}\left(\tilde{\mu}, \tilde{\Sigma}\right)$$

with

$$\begin{aligned} \tilde{\mu} &= \mu_2 + \Sigma_{21}\Sigma_{11}^{-1}\left(Y_1 - \mu_1\right) \\ \tilde{\Sigma} &= \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12} \end{aligned}$$

Figure 7.7: *Effect of planning. Upper left: User input representing the model class. After the planning step we obtain an inference order for the parts: dark blue, red, cyan. The set of initial candidates for each part is shown.*

The update incorporates restrictions caused by the previously assigned parts to the updated model. Further, the updated covariance matrices are independent of the ongoing inference and thus can be precomputed. Since the dependencies on previous parts are encoded in the updated models, they can be used for a more efficient pruning. The effect of these model updates (for the spatial layout) is illustrated in Figure 7.6, giving a hint of the effectiveness of these updates.

The last step to further improve efficiency concerns the evaluation of the descriptor score for part candidates. The full model for local shape is a Gaussian distribution, also the local shape for a single part is a Gaussian. Since the local shape for part i is evaluated for all potential candidates h_i, it is advisable to reduce the number of necessary operations. That can be achieved by compressing the Gaussian using principle component analysis.

7.3.2 Planning Inference Order

As already discussed in Section 5.2.3, it is advisable to plan the inference order, i.e. the order in which part hypothesis are tested. Our goal is to improve the performance of the inference, both in terms of accuracy and speed. Therefore, the strategy is to first search for those parts for which the location has the least uncertainty. This has two benefits:

- The search space is reduced, due to the pruning of unlikely hypothesis (Section 7.3.1), thereby improving the run-time.
- By fixing the parameters for the part with the least uncertainty first, the risk of propagating wrong information to later stages of the search is reduced. This is important since the approximate inference does not use backtracking or backward propagation of information to part hypothesis tested earlier.

in Section 7.4 we will demonstrate empirically that planning has a significant impact on both of these aspects, improving both on run-time costs and accuracy.

Planning itself is easy: We pick those parts first for which the uncertainty in localization is minimal. Again, we use a greedy optimization algorithm to make the choices:

The first part is selected by descriptor uniqueness: We match the descriptor model against the whole training set and choose the part with the lowest matching frequency, i.e., whose descriptor matches the fewest other points (again using a Mahalanobis distance of 2 as threshold).

For choosing subsequent parts, we now need to model the influence of the previous choices. Specifically, both the ambiguity in terms of descriptor match as well as variation in spatial localization should be minimized. For the locality, we compute the marginal of the Gaussian model: we estimate a marginal covariance in position, given we fix the previous plan-points. Please note this can be done using the Schur complement of the covariance matrix. For the descriptor, the improvement depends both on how frequent the descriptors are expected in the input as well as on the correlations with previously detected part descriptors. We therefore multiply the frequency of the part descriptor by the relative change in volume in descriptor space due to correlations. This shrinking of descriptor volume is modeled by the ratio of

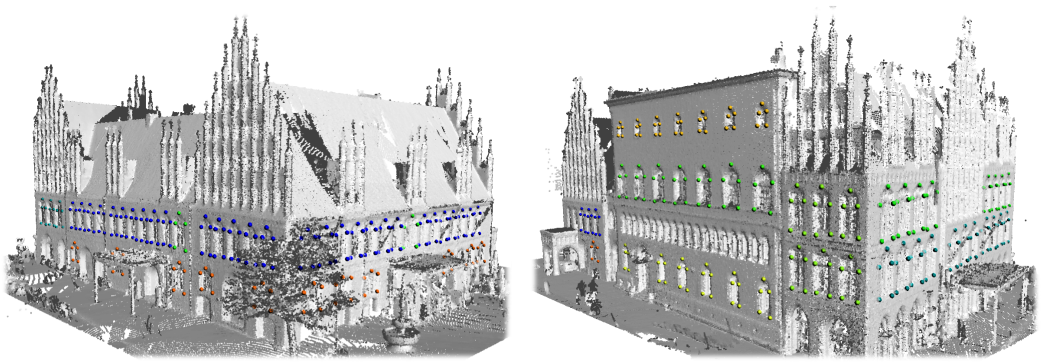

Figure 7.8: *Manual annotation of the old town hall used as ground truth for evaluation. For experiments with a general class all different window types are subsumed in one class, while the specific class experiments only comprise the windows marked in blue.*

the determinants of the unconstrained marginal descriptor covariance and the marginal descriptor covariance obtained after fixing the already selected plan-points. A typical result obtained from the planning step is shown in Figure 7.7.

7.4 Results and Evaluation

Similar to the previous chapter, we evaluate the performance of the correlated parts model and inference using LIDAR range-scans from the Hannover city scan collection (http://www.ikg.uni-hannover.de, courtesy of C. Brenner, IKG, University of Hannover) which is subject to significant (non-Gaussian) noise artifacts. In addition, we also use 3D scans of figurines to demonstrate rotational invariance as well as the hierarchical variant of the method. Experiments with large and memory demanding scenes have been performed using an unoptimized single-core C++ implementation on a dual socket workstation with two Intel Xeon X5650 (2.6GHz) and 48GB of RAM.

7.4.1 Local Shape Descriptors

The local shape descriptor is used to characterize the geometry within the r-neighborhood $N_r(\mathbf{x}) = \{\mathbf{y} \in S | \, \|\mathbf{x} - \mathbf{y}\| \leq r\}$ of a point $\mathbf{x}$ in S.

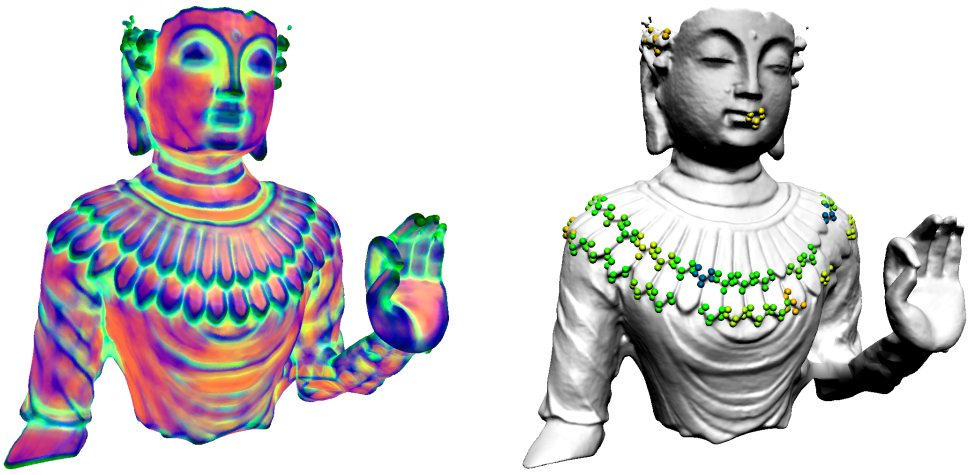

Figure 7.9: *Shapes under arbitrary rotations. Projections of the descriptors onto the first three principle components of descriptor space are illustrated on the left. The shape model was constructed using the exemplars shown in Figure 7.4, however allowing for less variations. Detection results are shown on the right. Colors indicate the quality of the match (blue: perfect; red: bad).*

In order to effectively assess the correlated parts model, we do the experiments without the use of sophisticated shape descriptors such as [FHK+04, CSM+06, KPW+10]. The experiments on large point cloud data require descriptors which come with low computation costs and yield a robust descriptor for small, almost planar, surface regions (which does not hold for spin images [JH99]). We have implemented the following shape descriptors:

- **Normal histograms**: For every point $\mathbf{y} \in N_r(\mathbf{x})$, we express the normal direction $\mathbf{n}(\mathbf{y})$ in polar coordinates with respect to a coordinate frame defined by the smoothed normal $\overline{\mathbf{n}}(\mathbf{x})$ and $\mathbf{y} - \mathbf{x}$. We then build a joint histogram of the two angles using 15×15 bins.

  The motivation behind is that we try to measure curvature in radial and the corresponding tangential directions and build a (discrete) distribution of both. The idea is that on one hand such shape descriptors are rotational invariant but still sensitive enough to recognize fine shape variations.

- **Oriented normal histograms:** Here, we augment the normal histograms by using a fixed reference frame given by $\overline{\mathbf{n}}$ and a fixed global upward direction $\mathbf{u}$.

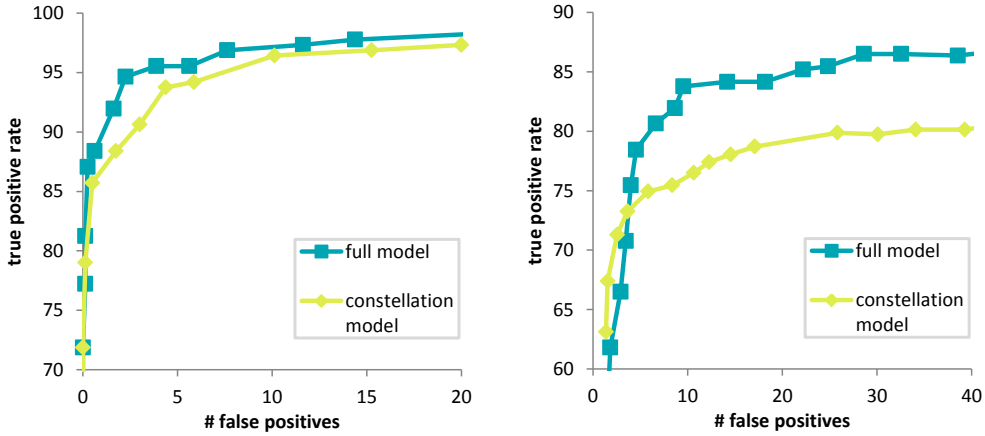

Figure 7.10: *Comparison with constellation model: The detection rate improves by including descriptor correlations. The left diagram shows results for the specific class of rigidly similar windows, the right for the more general class containing more varying geometry.*

For efficiency reasons and in order to smooth out noisy data, the set of high-dimensional descriptors in an input scene is projected onto a low-dimensional subspace using principal component analysis. In Figure 7.9 we demonstrate the robustness on a complex shape: All descriptors are mapped onto the first 3 principal components. Local shapes with similar appearance yield similar scores.

7.4.2 Quantitative Evaluation

For a quantitative evaluation of different aspects of the method we manually annotated two different test sets on the old town hall, as shown in Figure 7.8, ranging from a very specific class, capturing the most prominent window type, to a general class, comprising all types of windows present in the building. We count the number of false positives and negatives by a coarse criterion that measures the distance of the centroids of the detection hypotheses to the centroids of ground truth data. A detected instance only counts as a true positive if this distance to the nearest ground truth data is smaller than the descriptor radius r employed to compute the local shapes. We use cross-validation for measuring the performance; we always use 3 examples per type and average over 8-10 stratified random samples; we precompute a random partition that guarantees to cover all examples at

Figure 7.11: *Hierarchical detections for the crocodile with base model as shown in Figure 7.7. Upper left: projections onto the first 3 principle components of the base descriptor space. Upper right: detections of the base model (3 parts: nick-tip-nick). Lower left: projections onto the first 3 principle components of the descriptors obtained from 1st level hierarchy (at centroid positions of the base shapes). Lower right: Results for a hierarchical model comprising strands of base shapes. Structuring also helps reducing false positive detections of the base model.*

least once. Curves are measured by varying one of the model parameters while keeping the others fixed.

7.4.3 Shape Model Experiments

Rotation invariance and hierarchical shape model: We demonstrate the rotation invariance of the approach in Figure 7.9. The statue model features a deformed, regular pattern in different orientations, not captured by a common "upward direction", which is needed for the method of Chapter 6 to work. We also examine the use of hierarchical models (see Fig-

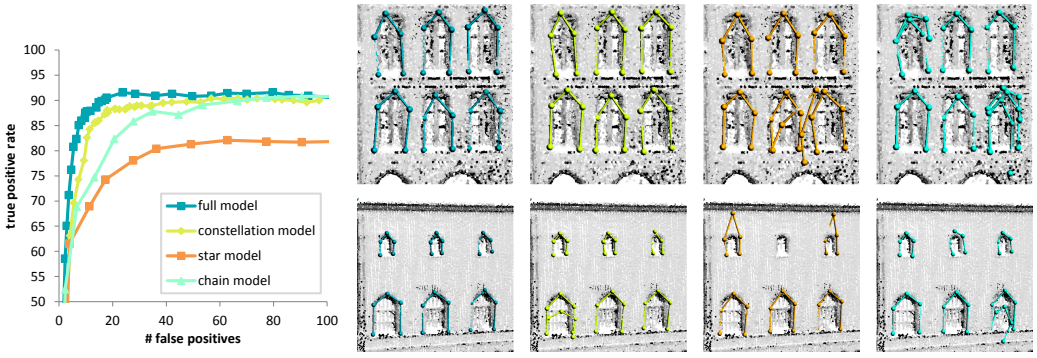

Figure 7.12: *Evaluation of different model structures. The excerpts show typical results for the different model types. Results are shown for similar numbers of false positives. The colors represent the different models.*

ures 7.4, 7.5, 7.11), improving the recognition accuracy and yield the structuring of the instances in terms of a regularly repeating grid. Figure 7.22 shows a hierarchical decomposition for the old town hall scan.

Effect of correlations between parts: We first evaluate the effect of including all pairwise correlations of the parts' local shapes into the model (which are not included in the traditional constellation model [RPA03]). Curves are measured by varying the descriptor noise parameter λ_D, describing the tolerance of a fit to noise and unmodeled effects. The results are shown in Figure 7.10: When learning a complex class for different windows in the old town hall, the detection performance improves by including these correlations. For the class of rigidly similar windows, this effect is less pronounced.

Secondly, we compare to different underlying graph structures, i.e. dependencies of relative locations of parts: Both, the full model as well as constellation style model include all pairwise correlations of parts locations. Star shaped models, as used in [FMR08, LLS04], only consider spatial relations to one center part. Chain-structured models, as employed in the previous chapter, merely consider pairwise correlations between a part and its predecessor. In order to compare the different model structures, we restrict the general, correlated model appropriately by removing accordant pairwise interactions, i.e. by setting the corresponding entries of the covariance matrix to 0. The results are shown in Figure 7.12. We varied covariance scale γ_D, to observe the behavior under shape models of different flexibility. The underlying structure of the shape is captured better the more dependencies are included.

Figure 7.13: *Comparison to the Markov chain model in Chapter 6. Because of the lacking global correlations, lower boundaries of the windows do not match up (left), which the fully-correlated model avoids (right). For both examples identical training instances are used. Edges encode the chain used in the previous approach.*

We also compare the method qualitatively to the chain model presented in the previous chapter (Figure 7.13). The new method yields more accurate correspondences. The global correlations avoid drift over the course of the chain. For applications in computer graphics, beyond pure detection, this is an additional benefit.

7.4.4 Inference Experiments

Effect of approximate inference: In order to quantify the error imposed by the approximate inference, we have implemented an exact version of the algorithm. To make exact inference feasible, a very specific model (small variances) is required since this allows for efficient pruning: We use the specific class (Figure 7.8) and assume very little noise. The results are shown in Figure 7.14 (middle). As expected, the exact version yields slightly better results, but the gain of the exponential algorithm is below 3% (please note the scale!).

Effect of Planning: We also study the effect of planning (see Figure 7.14). We compare to the average of a random order for inference, again for finding windows in the "old town hall". The recognition rate improves consistently by up to 4%. The effect on the run-time is more dramatic: As shown in Figure 7.14 (right), we obtain high detection rates much more rapidly than without planning.

exact vs heuristic inference

planning vs random order

effect of planning on timings

Figure 7.14: *Evaluation of the inference scheme. For a very narrow class (where exact inference is feasible) exact inference only performs slightly better than the approximate inference (left). Curves are measured by varying the noise parameter λ_D, describing the tolerance of a fit to noise and unmodeled effects. Effect of planning: the recognition performance improves over the average of a random order (middle). The runtime improves significantly (right). Curves are measured by varying λ_D.*

Scalability: We apply the method to all facades from the Hannover data set (126 million sample points, 4GB binary data). Since we do not want to compute descriptors for the complete set, we reduce the point cloud to a set of interest points. This can be done efficiently by extracting points of high curvature, using the technique of Gumhold *et al.* [GWM01], i.e., using the smallest eigenvalue of a PCA-analysis of local neighborhoods as curvature measure. Only these points are considered as candidates for points in H, the rest of the method remains unchanged. We denote the reduced point set by $\tilde{S}$.

Figure 7.15 shows an example where 35 windows are used for training, modeling a class with high variability, and retrieving a large subset of the actual windows in the rest of the town with few false positives (Figures 7.16, 7.17). False negatives (i.e. missed instances) arise mainly due to the facts, that considerable parts of the scan suffer from severe noise artifacts which the simple descriptors were not able to handle. Also in some regions the LIDAR scanners captured only little data (due to insufficient scan resolution for points far away) which results in missing interest points. The inference algorithm runs in less than 2 minutes (on the reduced set), statistics are shown in Figure 7.18. We also vary the scene size by cutting out excerpts: the run-time scales almost perfectly linear with scene size (Figure 7.18).

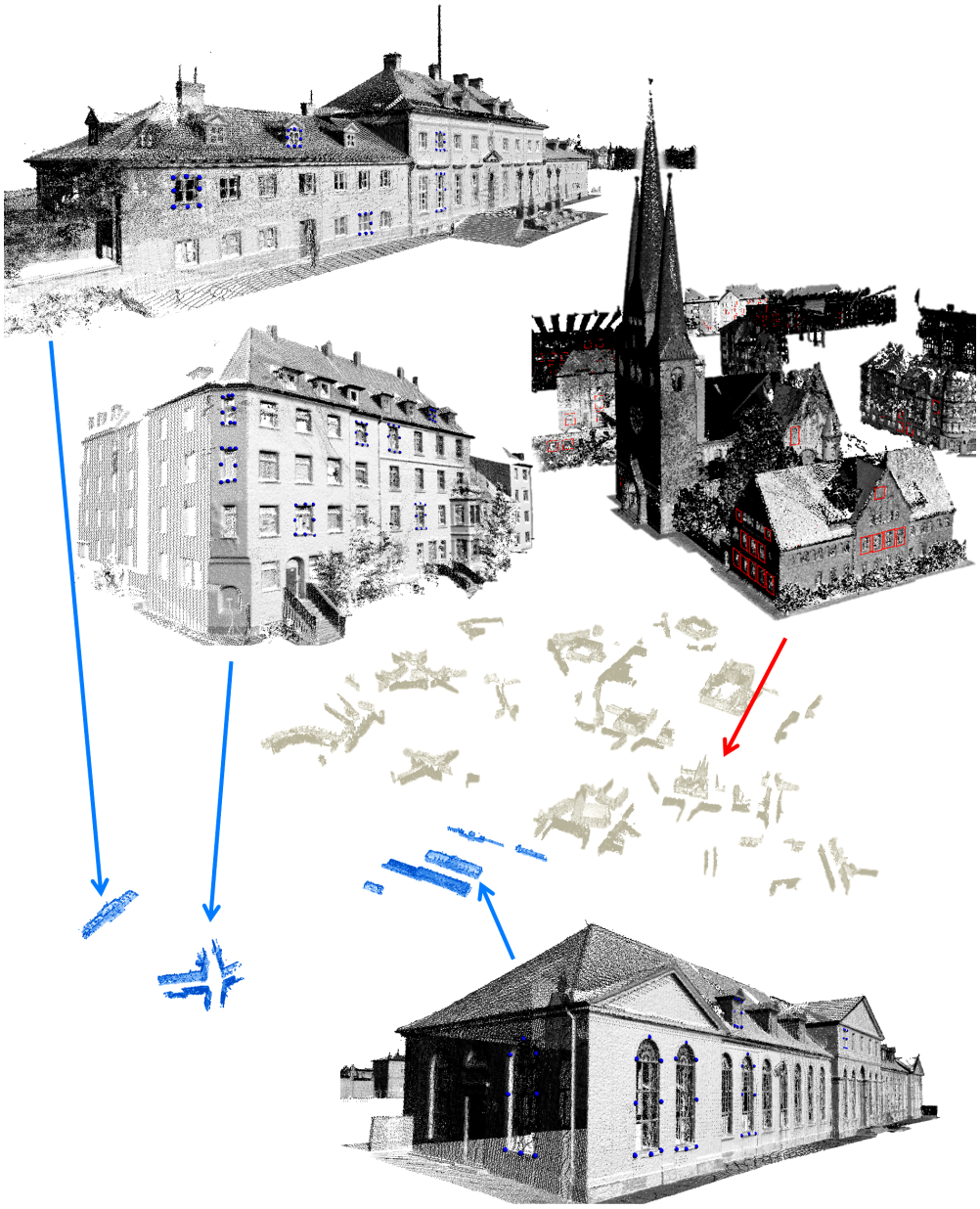

Figure 7.15: *Large scale result: The complete set of Hannover scans (126 million points, 4GB of raw data). We train a class of considerably varying shapes using 35 example windows, all taken from the buildings marked in blue (16 million points). We are able to detect a substantial fraction of further instance of the "window" class on the rest of the data set. The computation time is approx. 2 minutes. Close-ups show trained instances (marked blue) and one detection excerpt (marked red). More detection close-ups are shown in Figures 7.16 and 7.17.*

Figure 7.16: *Detection results for the big Hannover scene (see Figure 7.15). The top shows the Wilhelm Busch museum which is part of the training region. The bottom shows detected windows in a street of houses in the test region.*

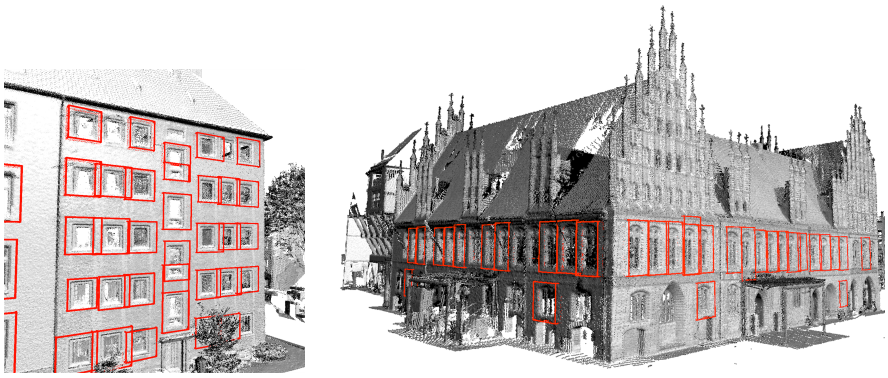

Figure 7.17: *Further detection results for the big Hannover scene (see Figure 7.15). A considerable percentage of all windows is detected.*

	$\lvert S\rvert$	$\lvert\tilde{S}\rvert$	# found
train	16,193,592	1,013,301	249
test	110,763,979	8,645,128	672
all	126,957,571	9,658,429	921

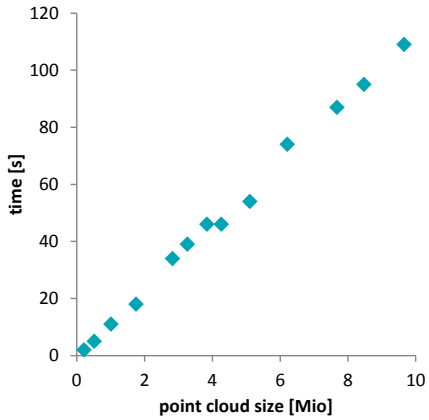

Figure 7.18: *Statistics for the complete Hannover scan. The point cloud was cut in parts of different sizes. The measurements of the computation times for detection show the linear scalability to big data.*

Figure 7.19: *Multi-class learning results for the old town hall. Each class was trained with 3 exemplars.*

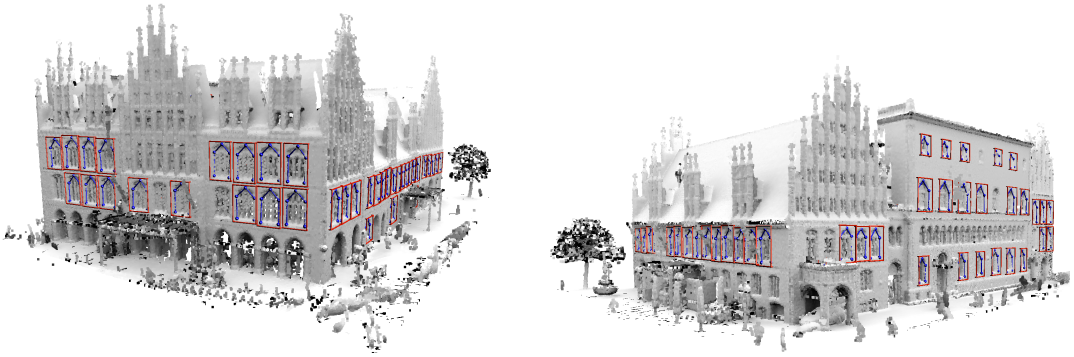

Figure 7.20: *Single-class learning results for the old town hall. The model is trained with 3 windows of each category as defined in Figure 7.8*

7.4.5 Experiments on Old Town Hall Scan

Similar to Chapter 6 we conduct window detection experiments on the scan of the old town hall in Hannover, for which we used the oriented normal histogram as descriptors. The challenge with this scan is that it is of rather low quality. We used the ground truth annotations shown in Figure 7.8. Figures 7.19 and 7.20 show qualitative evaluations of detection experiments with the multi-class and the single-class learning. Multi-class training included 3 random exemplars of a chosen category and single-class learning combined 3 training examples of each window category into one detector, thus generating a class with high variability. Except for few misses, the detection yields good results with semantically correct part correspondences.

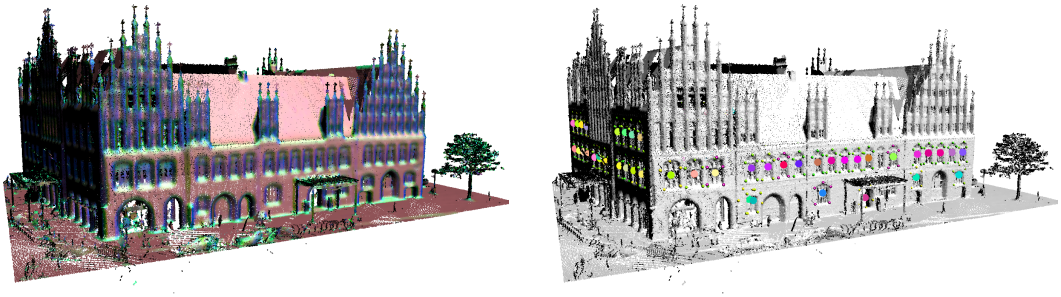

Figure 7.21: *Structure retrieval – 1st step: We use the directed orientation histogram descriptor (left) and use 4 windows (each form a different class) to train a multi-class window model. For each found window instance we generate the descriptor for the next hierarchy level. The colored centroids represent the resulting descriptors after projection onto the first principal components. Please note windows of the same type inherit similar colors.*

We also applied a hierarchical decomposition and extracted grid-like window alignments. In a first step we trained a multi-class window model out of 4 training exemplars. From the detected instances we generate new descriptors for the 2nd hierarchy level. With help of those a grid-like structure is extracted. The results can be seen in Figures 7.21 and 7.22.

7.5 Summary

We have implemented an approach for a general graphical model for object detection in large 3D scans. The model correlates all parts for both layout and local shape. We introduced an efficient approximate inference algorithm which robustly detects all object instances, including their correspondences. On data with significant artifacts we have shown that it yields robust detection results for whole shapes as well as their part correspondences. The approach can handle more complex shapes. It is rotational invariant and can decompose objects hierarchically into composite models with parts representing simpler models themselves – which can be used to obtain a natural structuring of the scene.

In experiments we have deliberately chosen to employ simple, basic descrip-

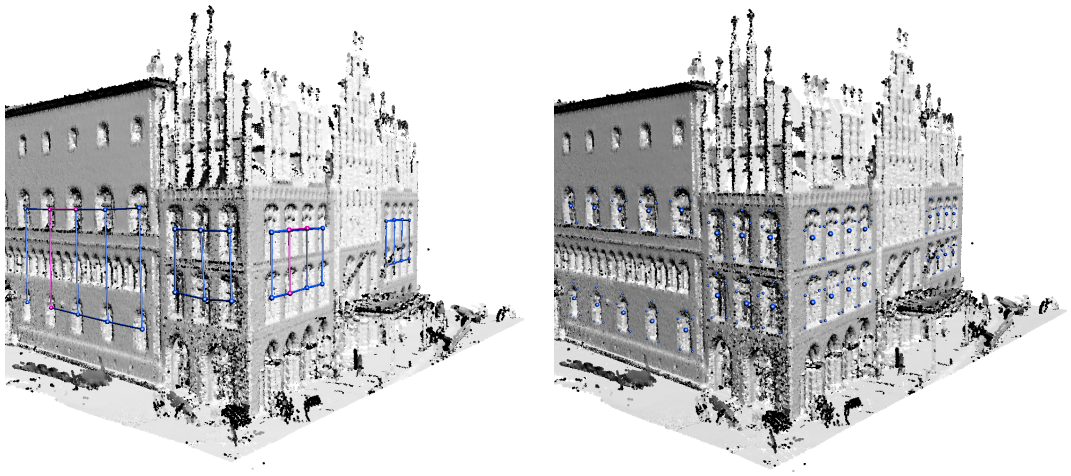

Figure 7.22: *Structure retrieval – 2nd step: We extract the grid-like structure of windows in the old town hall scan. The red shapes (left image) denote the training examples extracted from the 1st hierarchy level (Figure 7.21) of the hierarchical model. The right side shows the detections of the base model which coincide with the detection results of the second level. That way false positives of the 1st level can be filtered out. Please note we actually define corner-like shapes in one orientation only, but we also find rotated instances. The 2nd level descriptors for windows are tuned less sensitive compared to the influence of spacial layout.*

tors to focus the study on the effect of the correlated parts model and the approximate inference. Even then, we already obtain remarkable results, such as discovering a large number of windows in a city scene with few false positives from a rather small training set, that have not been demonstrated previously. The algorithm is fast and scalable; a single-threaded non-optimized implementation can retrieve sets of object instances in large 3D scans with more than 100 million points within 2 min, a figure that has also not yet been shown in literature.

In comparisons with other graphical models the correlated model clearly outperforms. In general the results indicate that pairwise chain structures are more suited for the detection scenario than star topologies, especially if part correspondences are wanted. A pairwise chain structure performs better, but it still allows for semantic flaws in the alignment. Those can be mostly overcome if we use full constellation models. However, those are still not robust enough when applied to object classes with high shape variability.

Nonetheless there are still some limitations. The presented detection scheme is (just as the chain model in Chapter 6) sensitive to missing data such as holes in a scan, an issue which still needs further consideration. Another limitation is that the overall detection could be further improved. First experiments with integrating various different descriptors as in [KHS10] indicate potential for improvement. Especially experiences with Bag-of-features (or Bag-of-words) implementations which are used for implicit shape models [LLS04, LLS08, VSS12] yield a positive prospect. Since in the experiments we have shown that a fully-correlated model is superior over star topologies, integrating according descriptors might yield even better performance.

8 Conclusion

Don't let it end like this.
Tell them I said something.

rumors about the last words of
Pancho Villa (1877 - 1923)

We will conclude this thesis with a summarizing discussion about graphical models for 3D object detection. We start with some final words regarding the different graphical models. Then we will summarize the contributions of this work to scene understanding and review existing limitations, and depict approaches how to proceed further. Finally, this work concludes with a prospect to the 'future'.

8.1 Final Discussion on Graphical Models

Besides results from prior work this thesis has shown that 3D object detection problems can be and should be approached utilizing part-based graphical models. However, there is a number of decisions to take care of. The most important one is the choice of the dependency structure. Each comes with its benefits and drawbacks.

Star Topology: The star topology is one of the common implementations of graphical models used for Computer Vision. They can be exactly inferred with easy message passing or voting algorithms. But their strongest point

is also the weakest. There is no part which is not dependent on a node other that the root. That means, in case of missing geometry (i.e. holes or heavy artifacts) those approaches can easily be made insensitive (up to some extend) by just introducing robust potential functions. If geometry is missing the model can simply assume a low score for the missing part. Then, for reconstruction it can be, either be left out, or represented by the mean of this part. However, the conditional independence structure of star topologies implies that, as soon as the root node is assigned, all parts become independent and the overall constellation of all parts might be semantically inconsistent. This fact is observed in experiments.

Chain Topology: The chain implementation is a dual of the star topology. In contrast to linking all nodes to a root there are single, exclusive links between pairs. That approach makes the assignment for parts dependent on their direct neighbors. Experiments indicate that this approach is better suited for the detection goal defined in this thesis, i.e. consistent semantic correspondences between all detected object instances. For models with low part variations we can capture good results, but as soon as the model allows for bigger variations, there might be certain 'drifts' and inconsistencies in layout. Further, in contrast to star topologies it is unclear how to make this approach insensitive to missing data without giving up the efficient inference scheme.

Tree Topology: Tree topologies are a hierarchical combination of star models. Hence, they also inherit strengths and weaknesses of normal star structures. But every path from root to a leaf is also a chain. Thus, they combine both star and chain topologies. Detection can be made insensitive to holes since we might utilize robust potentials for full branches. Further each chain structure would yield a better dependence structure in order to provide higher semantic consistency. However, the consequence is that each branch itself is sensitive to holes or we must include early back-tracking capabilities at the cost of efficiency. Although in terms of semantic consistency the tree topology is superior to star- or chain structures it sill leaves out some possibly important dependencies.

Constellation model: Constellation models encode a fully connected graph for the part positions, hence they encode full spatial consistency. But those models cannot be exactly inferred in feasible time. Prior to this work the most common approach is to use slow sampling approaches. Utilizing the approximate inference scheme presented in Chapter 7 overcomes that issue. However, experiments for models with strong variations in the local part

appearance show that the final localization of parts might be not as precise as desired.

Correlation model: The novel correlation model is also a fully connected graphical model. But in comparison to a constellation model it also takes into account all pairwise part dependencies for local shape. Thus, in comparison to constellation models problems with part localization can be diminished. The joint distribution can be efficiently approximated, but likewise the chain inference it is sensitive to missing data.

8.2 Contributions and Achievements

The concepts and methods discussed and evaluated in this thesis yield a clear and desirable contribution towards the goal of semantic 3D scene understanding: General object detection and automatic semantic alignment for large quantities of object instances in complex scenes are a key tool to facilitate sophisticated future applications on 3D data. We have seen how to define statistical 3D object models with only a small amount of user interaction. The main idea behind robust and efficient detection is to utilize part-based graphical models which also encode natural correspondences for all objects by just using the parts. We implemented models for two different graph types and presented highly efficient detection schemes which allow for interactive applications. The simple chain model can be exactly computed and yields robust detections, however the model still allows for flaws in the final parts alignment. The correlation model clearly outperforms the chain approach (as well as other approaches). It comes with a hierarchical scheme that constructs composite models which are capable to easily model and detect more complex shapes – decomposing them into a hierarchical structure. Further, the model is invariant on orientation of the input and comes with a novel approximate message passing inference that allows for fast and robust detections even for large city scans. Last but not least there is no comparable prior approach also tackling the given general detection scenario. On one hand prior work is mainly focused on classification. On the other hand these approaches either rely on shapes which can be easily separated from scenes or they need to encode prior knowledge like finding repetitive patterns within facades. Thus, this thesis can be considered as a pioneer step towards general scene understanding in 3D.

8.3 Current Limitations

Besides all the strengths of the presented methods and models, there are still some limitations. The following shows the major problems and indicates how they could be addressed in future research.

Shape descriptors: On descriptor side the current approaches still have room for improvements. For example, results using simple star-structured implicit shape models exhibit good detection results. A major reason for that is the utilization of more sophisticated descriptors.

The part-based detection scenario of this thesis raises the demand for precise, well locatable shape descriptors. Popular techniques, such as histogram of oriented gradients [DT05], yield descriptors which are insensitive to (high frequency) shape variations, yet their principle is to decompose local shapes into regions of blocks. Thus, they are robust but also less discriminative, i.e. they might not yield sharply peaked predictions. Therefore they are less suited when in comes to finding accurate shape correspondences.

The secret behind implicit shape models [LLS04] most likely hides in the Bag-of-features approach. The main idea behind those implementations is the following: Each point in a geometric data set can be described by its matching score with the local shapes of a number of feature points (*code book*), given some descriptor function. Most important however is, that these feature points are generated such that their representations in shape space are well separated from each other. Consequently, any point in the data can be described in a most discriminative way: by the vector of distances (in shape space) to all of the feature shapes. This fundamental idea also found application in deformable matching implementations such as in [TBW+11]. Hence, it is advisable to extent the current implementations to a bag-of-features inspired approach in a first step and afterwards explore that direction further.

Parameters: For both of the approaches presented in this thesis the models contain some meta parameters (e.g. 'covariance scale' or 'sample noise') which need to be fixed manually. That is not a big issue since single detection passes can be performed efficiently. Also, it might be desirable to have some influence on the final model. Yet estimating the parameters which yield the best performance might be cumbersome. This issue can be overcome by applying a scheme which discriminatively learns all model parameters. However, this is a difficult problem, given the user only annotated few positive

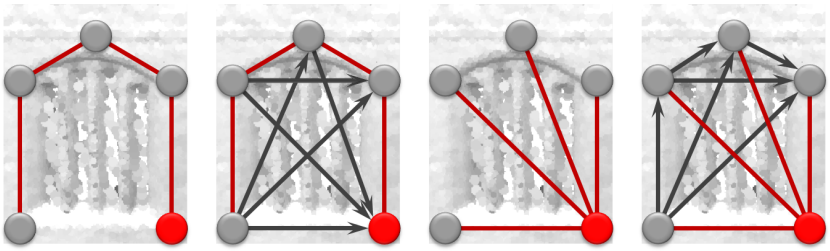

Figure 8.1: *Graphical models for a window. The red node depicts the root node. The red edges represent bidirectional dependencies, the black arrows show one-directional dependencies. From left to right: chain model, fully correlated model with 'chain' approximation, star model, fully correlated model with 'star' approximation.*

training data since we also need negative training exemplars. In general, it is necessary to gain as much training data as possible in order to reliably estimate high dimensional distributions.

Discriminative training can help to generate these data utilizing an Expectation-Maximization algorithm: For a given scene, the user first specifies all occurring instances of the object class which is to be trained, as well as a certain tolerance measure. Then we can sample further positive training data which is within the tolerance in addition to random negative training samples which are clearly out of bounds. Similar to support vector training we can estimate an ellipsoidal separating hyperplane representing the detection threshold of the Gaussian model. The volume of the ellipsoid then represents the space from which we accept valid assignments. Now we can iteratively refine the hyperplane by further adding false positive detections to the negative training data until there is no more improvement possible.

This is a rather rough sketch for a discriminative but promising training approach. However, there is a main issue. The quality of the final model highly depends on the generated training samples as well as on their number. In order to estimate a true non-rank deficient covariance matrix we need to generate $O(d^2)$ many samples, where d is the dimensionality of the probability distribution. Thus, we need to find a sophisticated strategy for that problem.

Sensitivity to holes: As already discussed, the presented algorithms are sensitive to missing data. However, for correlation models that might be overcome by modifying the approximate inference scheme. The intuition behind this is that dual to the chain approximation (see Section 5.2.3), the

original fully connected graphical model could also be approximated by a star model with additional one-directional dependencies. Figure 8.1 depicts the dependency structures for both models. The red edges indicate the message passing routes (always towards the root node). For the star-like approximation messages are not recursively computed like for the chain approximation, but they are directly sent to the root node, separately and in a successive order. Each time a message is passed we can update the max-marginal distribution of the root variable and (similar to the chain approximation) update the overall probability distribution for all remaining variables. In that way the recursive message passing structure can be avoided, thus allowing to apply techniques with robust potentials that make the inference less sensitive to holes. An implementation and evaluation of such an approximation is subject to future work.

Learning Parts: Sometimes the need for supervision in order to train a semantic model is considered a limitation, even if it is only little. In those cases it is necessary to not only estimate all model parameters but also to automatically find consistent correspondences for all training data. That is obviously a hard problem. A general solution for a fully connected model is infeasible. However, an approach might be to utilize the inference scheme that is used for detection also for learning the model, i.e. planning the number and position of parts. Hence, it is most important that the inference can be efficiently computed – which can be done with the presented schemes. A benefit of such an approach is that we would also yield an inference order that suits the detection scheme best.

8.4 Future Directions

Having sophisticated and robust high-performance tools like the ones presented in this thesis opens the prospect to a rich number of future applications. Most obvious tasks might be decomposing 3D scans into low resolution primitives, e.g. for mobile or navigation applications. On the other hand, techniques for artifact removal or super-resolution based on semantic knowledge might result in a growing availability of high resolution models captured with low-budget scanning devices or structure-from-motion techniques. Even further, with the availability of robust semantic correspondences for a large number of scans and with the help of additional techniques for dense correspondence estimation, the effort for constructing generative models for complex object classes can be drastically lowered since wearisome

manual alignment becomes obsolete. This might entail a number of tools for easy generation of morphable models, and with a growing number of such generative models we should expect very desirable and powerful applications that not only allow for automatic seamless hole filling, they also empower semantic morphing in large complex 3D scenes. Eventually, a general solution for each of such applications demands general tools for basic semantic understanding, for which this work shows a direction future research should follow.

Bibliography

[ACP03] B. Allen, B. Curless, and Z. Popović. The space of human body shapes: reconstruction and parameterization from range scans. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 587–594, New York, NY, USA, 2003. ACM. 2, 12

[ADF+10] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver. Google street view: Capturing the world at street level. *Computer*, 43, 2010. 1, 13

[ADFDJ02] C. Andrieu, N. De Freitas, A. Doucet, and M. I. Jordan. Introduction to mcmc for machine learning. *Machine Learning*, 2002. 42

[ASP+04] D. Anguelov, P. Srinivasan, H.-C. Pang, D. Koller, S. Thrun, and J. Davis. The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In *Proc. NIPS*, 2004. 2, 12

[ASS+09] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, 2009. 1, 13

[ATC+05] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *CVPR*, 2005. 3, 14, 32

[Bal87] D. H. Ballard. Readings in computer vision: issues, problems, principles, and paradigms. In M. A. Fischler and O. Firschein, editors, *Pattern Recognition*, chapter Generalizing the hough transform to detect arbitrary shapes, pages 714–725. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987. 10

[BBGO11] A. M. Bronstein, M. M. Bronstein, L. J. Guibas, and M. Ovsjanikov. Shape Google: Geometric words and expressions for invariant shape retrieval. *ACM Trans. Graph.*, 30, February 2011. 13

[BBK06] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Science (PNAS)*, 103(5):1168–1172, 2006. 2, 12

[BBW+08] A. Berner, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. A graph-based approach to symmetry detection. In *Proc. Symp. Point-Based Graphics 2008*, 2008. 12

[BBW+09] M. Bokeloh, A. Berner, M. Wand, H.-P. Seidel, and A. Schilling. Symmetry detection using line features. *Computer Graphics Forum (Eurographics 2009)*, 2009. 2, 4

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. 19, 27, 28, 32, 33, 36, 39, 40, 43, 44

[BKSS10] M. Bergtholdt, J. Kappes, S. Schmidt, and C. Schnörr. A study of parts-based object class detection using complete graphs. *Int. J. Comput. Vision*, 87(1-2):93–117, March 2010. 11

[BM92] P. J. Besl and N. Mckay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2), 1992. 2, 12

[BMP02] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24:509–522, 2002. 53

[BR07] B. Brown and S. Rusinkiewicz. Global non-rigid alignment of 3-d scans. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 26(3), 2007. 2, 12

[BV99] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proc. SIGGRAPH*, pages 187–194, 1999. 13, 48, 71

[Cip87] B. A. Cipra. An introduction to the ising model. *Am. Math. Monthly*, 94(10):937–959, December 1987. 33

[CM92] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155, 1992. 2, 12

[CSM+06] S. R. Correa, L. Shapiro, M. Meila, G. Berson, M. Cunningham, and R. Sze. Symbolic signatures for deformable shapes. In *PAMI*, 2006. 89

[CZ08] W. Chang and M. Zwicker. Automatic registration for articulated shapes. *Computer Graphics Forum (Proc. of SGP)*, 27(5):1459–1468, 2008. 2, 12

[DG03] S. Dasgupta and A. Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures and Algorithms*, 22(1):60–65, 2003. 53

[DGD+11] H. Dutagaci, A. Godil, P. Daras, A. Axenopoulos, G. C. Litos, S. Manolopoulou, K. Goto, T. Yanagimachi, Y. Kurita, S. Kawamura, T. Furuya, and R. Ohbuchi. Shrec '11 track: Generic shape retrieval. In *EG 3DOR Workshop*, 2011. 13

[DPP09] R. Detry, N. Pugeault, and J. Piater. A probabilistic framework for 3d visual object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1790–1803, 10 2009. 81

[DT05] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 3, 9, 10, 53, 106

[Eat83] M. L. Eaton. *Multivariate Statistics: a Vector Space Approach.* John Wiley and Sons, 1983. 23

[FE73] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Comput.*, 22(1):67–92, January 1973. 11

[FGG+10] J.-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day. In *ECCV*, 2010. 1, 13

[FGMR10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. 10

[FH05] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *Intl. J. Computer Vision*, 61(1):55–79, 2005. 3, 10, 11, 83

[FHK+04] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *Proc. Europ. Conf. Comp. Vision (ECCV)*, 2004. 89

[FMR08] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008. 10, 92

[GAF+10] M. Goesele, J. Ackermann, S. Fuhrmann, R. Klowsky, F. Langguth, P. Muecke, and M. Ritz. Scene reconstruction from community photo collections. *IEEE Computer*, 43(6), 2010. 1, 13

[GD05] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, 2005. 9

[GF09] A. Golovinskiy and T. Funkhouser. Consistent segmentation of 3D models. *SMI*, 33(3):262–269, June 2009. 3

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6(6):721–741, 1984. 32, 42

[GG04] N. Gelfand and L. J. Guibas. Shape segmentation using local slippage analysis. In *Proc. Symp. Geometry processing*, pages 214–223, 2004. 3

[GKF09] A. Golovinskiy, V. G. Kim, and T. Funkhouser. Shape-based recognition of 3D point clouds in urban environments. *ICCV*, September 2009. 3, 16

[GL09] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *In Proceedings IEEE Conference Computer Vision and Pattern Recognition*, 2009. 10

[GMGP05] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann. Robust global registration. In *Proc. Symp. Geometry Processing*, pages 197–206, 2005. 2, 12

[GWM01] S. Gumhold, X. Wang, and R. MacLeod. Feature extraction from point clouds. In *Proc. Meshing Roundtable*, 2001. 94

[HAWG08] Q.-X. Huang, B. Adams, M. Wicke, and L. J. Guibas. Non-rigid registration under isometric deformations. *Computer Graphics Forum (Proc. SGP)*, 27(5):1449 – 1457, 2008. 2, 12

[HC71] J. M. Hammersley and P. Clifford. Markov field on finite graphs and lattices. http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf, 1971. 30

[HFG+06] Q.-X. Huang, S. Flöry, N. Gelfand, M. Hofer, and H. Pottmann. Reassembling fractured objects by geometric matching. *ACM Trans. Graphics*, 25(3):569–578, 2006. 2, 12

[HGW90] J. Hammersley, G. Grimmett, and D. Welsh. *Disorder in physical systems: a volume in honour of John M. Hammersley on the occasion of his 70th birthday*. Oxford science publications. Clarendon Press, 1990. 30

[HJ85] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985. 25

[HKG11] Q. Huang, V. Koltun, and L. Guibas. Joint shape segmentation with linear programming. *ACM Trans. Graph.*, 30, 2011. 3

[HSS+09] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn, and H.-P. Seidel. A statistical model of human pose and body shape. *Computer Graphics Forum (Proc. Eurographics)*, 28(2), 2009. 71

[HTB03] D. Häehnel, S. Thrun, and W. Burgard. An extension of the icp algorithm for modeling nonrigid objects with mobile robots. In

Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI), pages 915–920, 2003. 2, 12

[HZRCP04] X. He, Zemel, R.S., and M. Carreira-Perpinan. Multiscale conditional random fields for image labeling. In *CVPR*, 2004. 9

[Ibe08] O. Ibe. *Markov Processes for Stochastic Modeling*. Stochastic modeling. Elsevier Science, 2008. 30

[Jan10] S. Jansen. Symmetry detection in images using belief propagation. Master's thesis, Universität des Saarlandes, Saarbrücken, July 2010. 37, 43

[JH99] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE PAMI*, 21:433 – 449, 1999. 89

[Kal02] O. Kallenberg. *Foundations of Modern Probability*. Springer, 2002. 21

[KBWS12] J. Kerber, M. Bokeloh, M. Wand, and H.-P. Seidel. Scalable symmetry detection for urban scenes. *Computer Graphics Forum*, 2012. 14

[KF09] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*, volume 2009. MIT Press, 2009. 19, 28, 31, 32, 36, 43

[KH03] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *ICCV*, 2003. 9

[KHS10] E. Kalogerakis, A. Hertzmann, and K. Singh. Learning 3d mesh segmentation and labeling. *ACM Trans. Graph.*, 29(3), 2010. 3, 14, 32, 101

[KKKT12] Y. Kwon, K. I. Kim, J. Kim, and C. Theobalt. Efficient learning-based image enhancement: Application to super-resolution and compression artifact removal. In *Proceedings of the British Machine Vision Conference*, pages 14.1–14.12. BMVA Press, 2012. 42

[KPW+10] J. Knopp, M. Prasad, G. Willems, R.Timofte, and L. V. Gool. Hough transform and 3d surf for robust three dimensional classification. In *ECCV*, 2010. 15, 16, 49, 89

[LG05] X. Li and I. Guskov. Multiscale features for approximate alignment of point-based surfaces. In *Symp. Geometry Processing*, pages 217–226, 2005. 2, 12, 70

[LG07] X. Li and I. Guskov. 3d object recognition from range images using pyramid matching. In *ICCV, Workshop on 3D Representation for Recognition*, 2007. 13

[LHS07] M. Leordeanu, M. Hebert, and R. Sukthankar. Beyond local appearance: Category recognition from pairwise interactions of simple features. In *CVPR*, 2007. 11

[Li95] S. Z. Li. *Markov random field modeling in computer vision.* Springer-Verlag, London, UK, UK, 1995. 31

[LLS04] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Proc. ECCVŠ04 Workshop on Statistical Learning in Computer Vision*, 2004. 10, 49, 83, 92, 101, 106

[LLS08] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *Int. J. Comput. Vision*, 77(1-3):259–289, May 2008. 3, 10, 49, 101

[Low03] D. Lowe. Distinctive image features from scale-invariant keypoints. In *Int. J. Computer Vision*, volume 20, pages 91–110, 2003. 53, 70

[LSP06] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 9

[MGGP06] N. J. Mitra, L. Guibas, J. Giesen, and M. Pauly. Probabilistic fingerprints for shapes. In *Symposium on Geometry Processing*, pages 121–130, 2006. 2

[MGP06] N. J. Mitra, L. J. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Trans. Graph.*, 25(3):560–568, 2006. 12

[MK09] B. Micusik and J. Kosecka. Piecewise planar city 3d modeling from street view panoramic sequences. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 2906–2912. IEEE, 2009. 13

[Moo08] J. Mooij. *Understanding and improving belief propagation.* PhD thesis, Radboud University Nijmegen, 2008. 28, 37, 43

[MPWC12] N. J. Mitra, M. Pauly, M. Wand, and D. Ceylan. Symmetry in 3d geometry: Extraction and applications. In *EUROGRAPHICS State-of-the-art Report*, 2012. 14

[MS11] F. Moosmann and M. Sauerland. Unsupervised discovery of object classes in 3d outdoor scenarios. In *ICCV Workshops*, pages 1038–1044, 2011. 15

[MS12] A. Mesolongitis and I. Stamos. Detection of windows in point clouds of urban scenes. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 17–24, 2012. 14

[MSS+06] B. Matei, Y. Shan, H. Sawhney, Y. Tan, R. Kumar, D. Huber, and M. Hebert. Rapid object indexing using locality sensitive hashing and joint 3d-signature space estimation. *PAMI*, 28:1111 – 1126, 2006. 13

[MVH09] D. Munoz, N. Vandapel, and M. Hebert. Onboard contextual classification of 3-d point clouds with learned high-order markov random fields. In *IEEE International Conference on Robotics and Automation*, May 2009. 15

[PBK+12] T. Pylvänäinen, J. Berclaz, T. Korah, V. Hedau, M. Aanjaneya, and R. Grzeszczuk. 3d city modeling from street-level data for augmented reality applications. In *3DIMPVT*, pages 238–245, 2012. 13

[PG01] M. Pauly and M. Gross. Spectral processing of point-sampled geometry. In *Proc. Siggraph*, 2001. 53

[PJC+10] A. Perina, N. Jojic, U. Castellani, M. Cristani, and V. Murino. Object recognition with hierarchical stel models. In *Proc. Europ. Conf. Comp. Vision (ECCV)*, 2010. 51

[PMD08] A. Patterson, P. Mordohai, and K. Daniilidis. Object detection from large-scale 3d datasets using bottom-up and top-down descriptors. In D. A. Forsyth, P. H. S. Torr, and A. Zisserman, editors, *Proc. European Conference on Computer Vision (ECCV)*, volume 5305 of *Lecture Notes in Computer Science*, pages 553–566. Springer, 2008. 15

[PMW+08] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. Guibas. Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics*, 27(3), 2008. 2, 12

[PSG+06] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, and T. Funkhouser. A planar-reflective symmetry transform for 3D shapes. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 25(3), 2006. 12

[RGKG12] N. Razavi, J. Gall, P. Kohli, and L. J. V. Gool. Latent hough transform for object detection. In *ECCV (3)*, pages 312–325, 2012. 3, 10, 49

[RN93] M. S. Ryan and G. R. Nudd. The viterbi algorithm. Technical report, University of Warwick, Coventry, UK, UK, 1993. 40

[RPA03] F. R., P. P., and Z. A. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages 264–271, 2003. 3, 10, 11, 83, 92

[SGS09] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, 2009. 11, 82

[SHZF12] I. Stamos, O. Hadjiliadis, H. Zhang, and T. Flynn. Online algorithms for classification of urban objects in 3d point clouds. In *3DIMPVT*, pages 332–339, 2012. 15

[SJW+11] M. Sunkel, S. Jansen, M. Wand, E. Eisemann, and H.-P. Seidel. Learning line features in 3d geometry. In *Proc. Eurographics*, volume 30, April 2011. 47

[SJWS13] M. Sunkel, S. Jansen, M. Wand, and H.-P. Seidel. A correlated parts model for object detection in large 3d scans. *Computer Graphics Forum (Proc. EUROGRAPHICS)*, 32(2):1–8, May 2013. 75

[Sor09] O. Sorkine. Least-squares rigid motion using svd. Technical Notes (published on web), February 2009. 71

[SSB05] B. Schölkopf, F. Steinke, and V. Blanz. Object correspondence as a machine learning problem. In *Proc. Int'l Conf. on Machine Learning*, pages 777–784, 2005. 13

[SSB06] F. Steinke, B. Schölkopf, and V. Blanz. Learning dense 3d correspondence. In *Advances in Neural Information Processing Systems (NIPS)*, volume 19, pages 1313–1320, 2006. 13

[SvKK+11] O. Sidi, O. van Kaick, Y. Kleiman, H. Zhang, and D. Cohen-Or. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. *Proc. SIGGRAPH Asia*, 30, 2011. 3

[TBW+09] A. Teves, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. Isometric registration of ambiguous and partial data. In *Proc. Conf. Computer Vision and Pattern Recognition (CVPR)*, 2009. 2, 12

[TBW+11] A. Tevs, A. Berner, M. Wand, I. Ihrke, and H.-P. Seidel. Intrinsic Shape Matching by Planned Landmark Sampling. In O. Deussen and M. Chen, editors, *Computer Graphics Forum (Proc. EUROGRAPHICS)*, pages 543–552, Llandudno, UK, 2011. Eurographics, Blackwell. 2, 3, 106

[TCF09] R. Toldo, U. Castellani, and A. Fusiello. A bag of words approach for 3d object categorization. In *Proceedings of the 4th International Conference on Computer Vision/Computer Graphics CollaborationTechniques*, MIRAGE '09, pages 116–127, Berlin, Heidelberg, 2009. Springer-Verlag. 15, 16

[VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE Conf. on Comp. Vision and Pattern Recognition (CVPR)*, 2001. 3, 51

[vKTS+11] O. van Kaick, A. Tagliasacchi, O. Sidi, H. Zhang, D. Cohen-Or, L. Wolf, , and G. Hamarneh. Prior knowledge for part correspondence. *Computer Graphics Forum (Proc. Eurographics)*, 30(2):553–562, 2011. 3

[vKZHCO11] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011. 13

[VSS12] A. Velizhev, R. Shapovalov, and K. Schindler. An implicit shape model for object detection in 3d point clouds. *22nd ISPRS Congress, Melbourne, Australia*, 2012. 3, 16, 17, 49, 83, 101

[WZFF06] G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *CVPR*, 2006. 3, 11

[YC98] A. L. Yuille and J. M. Coughlan. An a* perspective on deterministic optimization for deformable templates. *Pattern Recognit*, 33:603–616, 1998. 11

[YFW01] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *IJCAI 2001 Distinguished Lecture track*, 2001. 37, 43, 56, 57

[ZLZ+10] H. Zhao, Y. Liu, X. Zhu, Y. Zhao, and H. Zha. Scene understanding in a large dynamic environment through a laser-based sensing. In *ICRA*, 2010. 14

[ZSCO+08] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliasacchi. Deformation-drive shape correspondence. *Computer Graphics Forum (SGP 2008)*, 27(5):1431–1439, 2008. 2, 12