Of Assembling Small Sculptures and Disassembling Large Geometry

Jens Kerber

Max-Planck-Institut für Informatik
Universität des Saarlandes
Saarbrücken, Germany

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

Eingereicht im Juli 2013 in Saarbrücken.

Dekan — Dean

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

Datum des Kolloquiums — Date of Defense

17. September 2013 in Saarbrücken

Prüfungsausschuss — Board of Examiners

Chair	Prof. Dr. Thorsten Herfet	Universität des Saarlandes Saarbrücken, Germany
Examiner	Prof. Dr. Hans-Peter Seidel	Max-Planck-Institut für Informatik Saarbrücken, Germany
Examiner	Prof. Dr. Alexander Belyaev	Heriot-Watt University Edinburgh, United Kingdom
Examiner	Dr. Michael Wand	Max-Planck-Institut für Informatik Saarbrücken, Germany
Reporter	Dr. Stefanie Wuhrer	Universität des Saarlandes Saarbrücken, Germany

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.
Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Saarbrücken, am 12. Juli 2013

(Jens Kerber)

Abstract

This thesis describes the research results and contributions that have been achieved during the author's doctoral work. It is divided into two independent parts, each of which is devoted to a particular research aspect.

The first part covers the true-to-detail creation of digital pieces of art, so-called relief sculptures, from given 3D models. The main goal is to limit the depth of the contained objects with respect to a certain perspective without compromising the initial three-dimensional impression. Here, the preservation of significant features and especially their sharpness is crucial. Therefore, it is necessary to overemphasize fine surface details to ensure their perceptibility in the more complanate relief. Our developments are aimed at amending the flexibility and user-friendliness during the generation process. The main focus is on providing real-time solutions with intuitive usability that make it possible to create precise, lifelike and aesthetic results. These goals are reached by a GPU implementation, the use of efficient filtering techniques, and the replacement of user defined parameters by adaptive values. Our methods are capable of processing dynamic scenes and allow the generation of seamless artistic reliefs which can be composed of multiple elements.

The second part addresses the analysis of repetitive structures, so-called symmetries, within very large data sets. The automatic recognition of components and their patterns is a complex correspondence problem which has numerous applications ranging from information visualization over compression to automatic scene understanding. Recent algorithms reach their limits with a growing amount of data, since their runtimes rise quadratically. Our aim is to make even massive data sets manageable. Therefore, it is necessary to abstract features and to develop a suitable, low-dimensional descriptor which ensures an efficient, robust, and purposive search. A simple inspection of the proximity within the descriptor space helps to significantly reduce the number of necessary pairwise comparisons. Our method scales quasi-linearly and allows a rapid analysis of data sets which could not be handled by prior approaches because of their size.

Kurzfassung

Die vorgelegte Arbeit beschreibt die wissenschaftlichen Ergebnisse und Beiträge, die während der vergangenen Promotionsphase entstanden sind. Sie gliedert sich in zwei voneinander unabhängige Teile, von denen jeder einem eigenen Forschungsschwerpunkt gewidmet ist.

Der erste Teil beschäftigt sich mit der detailgetreuen Erzeugung digitaler Kunstwerke, sogenannter Reliefplastiken, aus gegebenen 3D-Modellen. Das Ziel ist es, die Objekte, abhängig von der Perspektive, stark in ihrer Tiefe zu limitieren, ohne dass der Eindruck der räumlichen Ausdehnung verloren geht. Hierbei kommt dem Aufrechterhalten der Schärfe signifikanter Merkmale besondere Bedeutung zu. Dafür ist es notwendig, die feinen Details der Objektoberfläche überzubetonen, um ihre Sichtbarkeit im flacheren Relief zu gewährleisten. Unsere Weiterentwicklungen zielen auf die Verbesserung der Flexibilität und Benutzerfreundlichkeit während des Enstehungsprozesses ab. Der Fokus liegt dabei auf dem Bereitstellen intuitiv bedienbarer Echtzeitlösungen, die die Erzeugung präziser, naturgetreuer und visuell ansprechender Resultate ermöglichen. Diese Ziele werden durch eine GPU-Implementierung, den Einsatz effizienter Filtertechniken sowie das Ersetzen benutzergesteuerter Parameter durch adaptive Werte erreicht. Unsere Methoden erlauben das Verarbeiten dynamsicher Szenen und die Erstellung nahtloser, kunstvoller Reliefs, die aus mehreren Elementen und Perspektiven zusammengesetzt sein können.

Der zweite Teil behandelt die Analyse wiederkehrender Stukturen, sogenannter Symmetrien, innerhalb sehr großer Datensätze. Das automatische Erkennen von Komponenten und deren Muster ist ein komplexes Korrespondenzproblem mit zahlreichen Anwendungen, von der Informationsvisualisierung über Kompression bis hin zum automatischen Verstehen. Mit zunehmender Datenmenge geraten die etablierten Algorithmen an ihre Grenzen, da ihre Laufzeiten quadratisch ansteigen. Unser Ziel ist es, auch massive Datensätze handhabbar zu machen. Dazu ist es notwendig, Merkmale zu abstrahieren und einen passenden niedrigdimensionalen Deskriptor zu entwickeln, der eine effiziente, robuste und zielführende Suche erlaubt. Eine simple Betrachtung der Nachbarschaft innerhalb der Deskriptoren hilft dabei, die Anzahl notwendiger paarweiser Vergleiche signifikant zu reduzieren. Unser Verfahren skaliert quasi-linear und ermöglicht somit eine rasche Auswertung auch auf Daten, die für bisherige Methoden zu groß waren.

Acknowledgements

First, I would like to express my gratitude to Kristina Scherbaum and Hans-Peter Seidel for providing an excellent research environment. Their respective administrative and IT support staff have ensured unobstructed and pleasant working conditions over the past few years.

I owe special thanks to my advisors, Alexander Belyaev and Michael Wand, for productive discussions and helpful suggestions during the preparation period of this thesis. I highly appreciate their effort, patience, availability, and above all the expertise they passed on to me. The stimulating ideas of my collaborators, Jens Krüger and Meili Wang, have broadened my view by directing my attention to related research fields.

Furthermore, I am grateful to my former fellow students and colleagues in the Computer Graphics Department at MPI and in the Statistical Geometry Processing Group at MMCI, with whom I had the pleasure to work. Thank you for effective suggestions, motivating comments, and concrete help in both my academic and social life. Deserving special mention here are Art Tevs, Martin Bokeloh, Rhaleb Zayer, and Zhao Dong (in alphabetical order).

Special thanks also to Theodora Popova and Pascal Schmitt for their particular contributions to the relief prototype, and to Krista Ames and Margaret De Lap for proofreading this thesis.

Finally, I am indebted to my family and friends who have always supported and encouraged me, although they often suffered because of my absence.

Thanks to all of you for accompanying me on this journey and for making it a memorable time.

Jens Kerber

Contents

List of Figures

List of Tables

Part I

User-Friendly Relief Art Design

Chapter 1
Introduction

I saw the angel in the marble
and carved until I set him free.

Michelangelo (1475 - 1564)

Relief generation is a young branch of computer art which combines elements from geometric shape deformation with properties of the human visual system. This research discipline deals with the problem of finding a transformation from a three-dimensional input into a more planar counterpart without perceivably affecting the initial appearance.

In fine art, reliefs belong to a category that occupies an intermediate stage between two-dimensional painting and three-dimensional sculpting. On the one hand, relief motifs are not free-standing, but rather are based on a background. On the other hand, they are plastic because the underlying medium is eroded or carved, or additional material is applied to it.

Although inherently a complanate representation of a scene, a relief simulates the appearance of objects of full spatial extent by inducing an artificial depth impression. This suggestive effect is achieved by deriving a surface that mimics the shading properties of the original. Reliefs have a long tradition throughout almost all epochs and cultures, they occur in varying nature and scales, on diverse substances, and for purposes that range from decoration to practical use [Fla29].

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 1.1: Selected relief examples: Marmoreal bas-relief by Michelangelo (a), Egyptian sunken relief in granite (b), Indian high relief in stone (c), mid-relief on a Byzantine ivory leaf (d), bas-relief on a copper-nickel coin (e), bronze high relief (f), vitreous Roman vase (g), carved onyx gem (h), Persian bas-relief (i). All images shown are available at Wikipedia in the public domain or under a Creative Commons license [Mis13a].

We distinguish four main forms of reliefs depending on their situation and the degree of elevation[1]:

- **High reliefs:** larger sculptures perceptibly detached from a surface, e.g., story-telling stone artworks on religious sites, and antique monuments
- **Bas-reliefs:** very shallow shapes that only project negligibly, e.g., coinage, matrices for printmaking, or engravings on jewelry
- **Mid-reliefs:** bridge the gap between bas- and high reliefs, e.g., adornments for ceilings, furniture, glass, and pottery
- **Sunken reliefs:** are worked into the substance rather than protruding, e.g., ancient chiseled Egyptian illustrations, or cave art

A very rare related form is known as **counter-relief**[2], in which a motif is hollowed negatively into a surface such that an imprint in soft material like wax, or a casting with liquid metal, produces the actual desired relief. We find such counter-reliefs on gems of signet rings, for example. This type of relief can be derived from one of the other forms and is therefore not further considered here.

Nowadays, we find reliefs applied in industrial printing, as on business cards and packaging, and even in the manufacturing of candy formed from chocolate, marzipan, or sugar icing [Her99]. Furthermore, they are used to adorn digital shapes or to design virtual modern artworks. Figure 1.1 shows a variety of differing types of reliefs [Mis13a].

Up to now, crafting reliefs has been a laborious, demanding, and time-consuming task that has the drawbacks of lacking a preview option and being hard to correct or replicate with regard to large-scale manufacturing. This part of the thesis shall describe how computers can make a contribution to facilitate the relief generation process and to overcome these troublesome issues.

1.1 Problem Statement

Since we essentially speak of digital reliefs, we assume that the components of a desired scene are already available as virtual 3D models. If this is not the case, real-world objects can be captured by a laser scanner and directly serve as input. As we will show later, the derived relief can be printed directly, e.g., by a 3D printer or a milling device, in order to obtain a tangible exemplar.

[1] http://en.wikipedia.org/wiki/Relief, last visited: January 24th 2013
[2] http://en.wikipedia.org/wiki/Intaglio_(jewellery), last visited: January 24th 2013

(a) (b)

Figure 1.2: The Lucy statue after a naïve global compression (a) and our bas-relief result (b) in the same pose and under the same lighting conditions.

Given a three-dimensional scene with original proportions, the task is to shrink its elongation with respect to a certain perspective. Depending on the application, the limited available material depth is the main motivation. In the case of engravings or embossments, where the substance is only several millimeters thick, this becomes most apparent. But the mapping of reliefs on other virtual surfaces also requires a depth adaption to retain the proportions [POC05], [LTLZ11].

A naïve approach would be to squeeze the entire content to a plane perpendicular to the viewing direction by a straightforward linear rescaling of the depth dimension. This would indeed achieve the aspired compression, but would unfortunately fade perceptually salient shape features. Thus, this shrinking only works when no significant compression is required. In the case of a bas-relief, where the depth has to drop to a fractional amount of its initial size, the perceptibility of small and visually important details would drastically be impaired and the overall plastic impression would suffer to a great extent. Therefore, more sophisticated techniques are needed to retain the visibility of features and to ensure plausible outcomes. Figure 1.2 contrasts the result of a global linear shrinking with a relief generated with one of our methods. Note the emphasis of the garment wrinkles and the torch. The structure of the rock and her facial expression are much more clearly visible. In both cases the model was reduced to 2% of its former depth.

Most of the existing algorithms differ mainly in the way this crucial feature-aware compression is attained. Some methods do not yield accurate or satisfying results, whereas others are so complex or slow that only experts can achieve convincing reliefs in a reasonable time span.

Moreover, all other approaches are so far limited to scenes with static content and a fixed vantage point. Changing a model or adjusting the perspective requires a re-computation of the input, which further increases the time required to produce satisfying results.

1.2 Contributions

Our goal is to tailor suitable solutions that make this field of application accessible to even untrained enthusiasts by enabling any user to quickly create expressive and visually pleasing reliefs with little effort. Neither outstanding artistic skills nor substantial computer expertise is required. Aside from computational efficiency, it is therefore essential to keep the concepts intuitive and to focus on ease of use.

Our major achievements in research can briefly be summarized as follows:

- We have developed two conceptually simple and semi-automatic algorithms to address the problem of achieving a significant depth compression without compromising the quality of surface details. In both cases, our methods yield lifelike results even though the necessary user intervention was reduced to a minimum [KTB+09] [KTB+10].

- We have implemented our solutions on graphics hardware. In addition to a boost in performance, this also allows us to directly access required information from the graphics card. Thus, we are able to interconnect two major steps of the design pipeline, which results in more flexibility than ever before. Our system is the first one capable of processing digital reliefs in real-time and of handling dynamic scenes as input. These include animated models or interactive changes, like a camera motion [KTB+10].

- We have set up a graphical user interface to encapsulate all the advantages in one single application. It allows one to witness the effect of changing parameters, the scene content, or the perspective without delay. This represents enormous progress with regard to user-friendliness and the overall development time [KTB+10].

- We have achieved more artistic freedom by extending our method to model continuous and seamless relief artworks of scenes that are assembled from multiple objects or perspectives. Cubism-like portraits or overlapping geometric collages are potential examples of application [KTB+09] [KTB+10].

Altogether, these steps fulfill our aspired goals to enrich this field by offering a facile and rapid relief generation tool for every user.

Apart from our focus on designing elevated relief sculptures, we have also been involved in the research on sunken relief generation [WKCZ11] [WCKZ12] that contributed to the doctoral thesis of our collaborator Meili Wang [Wan11]. Additionally, we have compiled a survey on computer-assisted relief generation techniques in general, which also delineates and reviews techniques from related research fields [KWC+12].

When not stated otherwise, this first part of the thesis is based on the content presented in [KTB+09], [KTB+10], and [KWC+12].

Chapter 2
Fundamentals

All our knowledge has its origin in our perceptions.

Leonardo da Vinci (1452 - 1519)

This chapter imparts essential background knowledge about the interplay between the human visual system and the properties of reliefs. We introduce the necessary terminology for the upcoming chapter and explain some key components of our relief design methods. Finally, prior work in related research areas is reviewed.

2.1 Human Visual Perception

For relief generation, the goal is to dupe the eye of a beholder by creating a complanate representation of a three-dimensional scene, while at the same time conveying the appearance of fully extended objects. This false impression is achieved by inducing the shading in such a way that a difference from lifelike models is hard or impossible to discover, as long as an observer contemplates the relief from a certain perspective. This phenomenon of human perception is known as the bas-relief ambiguity [BKY99], which says that there exist several deformed modifications of an object whose appearance is almost indistinguishable from the one of the initial shape.

To be precise, under the assumption that the viewpoint does not vary, there exists a three-parameter family of transformations, under which the shading remains unchanged although the shape is distorted. In other words, multiple, differently formed shapes can cause the same impression to the human eye. Slight motions

of the viewer or marginal tilting of the relief allow the suggestion to still hold true, but if an off-axis vantage point is taken, the illusion is revealed.

The advantage of this ambiguity is that it allows one to artificially create nearly planar variations of 3D objects, for which the depth impression does not suffer. This fact has been known and exploited by artists for a long time. The negative aspect of this phenomenon is that most shape reconstruction algorithms (see Section 2.3.2) encounter the drawback that their solutions are not unique in general.

Edges along silhouettes, and large steps on a surface, are not visible from an orthogonal vantage point and only indicate low frequency transitions between distinct elements or height levels by casting a shadow. Nevertheless, they occupy a lot of "unused" depth range that could be compressed. These areas are characterized as local gradient extrema of the shape.

The effect of occluding contours for human perception and their correlation with principal curvatures along a surface has been investigated in [Koe84]. This has inspired other research areas to extract, e.g., so called suggestive contours [DFRS03], or stylized yet expressive line drawing [SP03], directly from a given geometry. For a human observer, the visually important clues about the constitution and the characteristics of a surface are contained in its ridges and valleys [JDA07]. The specular reflection along these crease lines is remarkably high, since they correspond to curvature extrema. As shown in [OBS04], even just the information at ridges and valleys is sufficient to automatically restore the underlying shape reliably.

2.1.1 High Dynamic Range Compression

The problem of transforming a shape into a more planar representation can be regarded as a geometric analogue to the task in *high dynamic range imaging* (henceforth HDR). In HDR, also known as *tone mapping*, a very large luminance interval has to be compressed such that it can be displayed on regular screens without compromising visually significant features like contrast and fine details. For relief generation, this corresponds to squeezing the depth interval range of a scene and preserving the perceptibility of ridges, valleys, as well as low- and high-frequency structures on the surface at the same time. Since image and shape features are of very different natures, a straightforward adaption of HDR methods is not possible. For reliefs we face additional problems like foreground-background transition and self-occlusions. Moreover, HDR produces 2D results for which the viewpoint of an observer does not matter, which opposes the desired properties of sculptures. Nevertheless, most relief design methods can be regarded as variants of, or are at least inspired by, solutions from tone mapping. For deeper insight into this related research topic, we refer to [DR06].

2.2 Concept Introduction

Relief generation is essentially a mapping from one surface to another. Nevertheless, several algorithms in this research area, including ours, rely on appropriate 2D representations of shapes and borrow well-established techniques from image processing that are adapted to the specific needs of this area. To follow, we formally introduce the notations that are used later on, describe mandatory operations, and explain the visualization.

2.2.1 Preprocessing

Given a 3D scene, we first convert the scene content to a so-called *height field*. As a height field, we denote a two-dimensional function h which stores distance information in regular discrete grid cells.

$$\begin{aligned} h : \mathbb{N} \times \mathbb{N} &\rightarrow \mathbb{R} \\ h(x, y) &= z \end{aligned}$$

The entries of the lattice implicitly encode the shapes in a scene relative to the viewing plane up to occlusion. This is why it is also denoted as a 2.5D representation. In practice, we capture height fields by rendering a scene and reading the depth buffer of the GPU. This is very fast and provides a preview such that camera position, situation of the objects, and height field resolution can be interactively adjusted beforehand. Henceforth, we use the terms *depth map* and *range image* as synonyms for height field.

We assume that the background pixels of h, those outside the clipping planes, are set to a default value δ and derive a binary *background mask* b that allows us to distinguish foreground and background content.

$$\begin{aligned} b : \mathbb{N} \times \mathbb{N} &\rightarrow \{0, 1\} \\ b(x, y) &= \begin{cases} 0, & \text{if } h(x, y) = \delta \\ 1, & else \end{cases} \end{aligned}$$

Then, the height field is normalized in such a way that the background lies on the zero-level and the foreground entries range from zero to a positive value. This maximal foreground entry is denoted as $R_{T=0}$ and indicates the initial depth interval range. The properties of the derivatives of h are important for several steps of our algorithms. They would be negatively affected if foreground objects coincide with the height field boundary. To prevent this, we extend the boundary of both functions h and b by 2 pixels of value 0. This concludes the pre-processing step and defines the input for our compression methods.

From now on, we alternatively use all two-dimensional functions with a unary pixel index i or j to achieve more compactness and readability in the notation.

2.2.2 Feature Enhancement

The main idea of relief generation is the preservation of small and high frequency details throughout the compression step. Therefore, it is important to distinguish visually important structures from those that are less important or coarse enough to "survive" the shrinking anyway. After representing the input appropriately, it is decomposed into a fine component, which is more sensitive to the compression, and a more robust coarse part. The details are then artificially boosted such that they also remain perceivable in the shrunken relief.

Unsharp Masking

The concept of *unsharp masking* is a key ingredient in our compression techniques. This term describes a feature enhancement technique that was initially designed to emphasize certain properties in images [LCD06] by manipulating the interplay between different frequency components of an underlying signal. Among others, such a technique can also be applied to exaggerate characteristics of meshes [RSI+08]. In our scenario, the height field is decomposed into a coarse and a fine component. Then, the relative importance of fine features is enhanced at the expense of coarse structures.

Technically, an input image I is convolved with a low pass kernel K, resulting in a smooth version L of I. Subtracting L from I leads to a high-frequency image H, containing sharp peaks at small scale details. Adding a multiple of H back to L leads to a relative emphasis of fine structures in the newly reassembled image $\overline{I}$.

$$\begin{aligned} L &= I \otimes K \\ H &= I - L \\ \overline{I} &= \lambda \cdot H + L \end{aligned}$$

Boosting with a sufficiently large λ ensures that even the fine details also remain perceivable in the compressed result. Whenever the relation between those components is influenced, it leads to a distortion of the shape. As described above, this effect is intended to fulfill the desired properties of the relief.

Bilateral Filtering

For the decomposition in the unsharp masking step, it was our idea to apply a bilateral filter. This is a tool commonly used in image processing. It permits edge-preserving smoothing of signals [TM98]. In contrast to regular Gaussian blurring, which leads to a smearing around transition areas, a bilateral filter overcomes this issue by diminishing the influence of pixels with differing entries.

Thus, it ensures the sharpness of local discontinuities and, at the same time, smoothes regions of similar values. Among other things, these properties are also provided by a diffusion filter but it is computationally more complicated and time-consuming. Gaussian as well as diffusion filtering have been used in earlier works. The particular differences as well as their advantages and disadvantages compared to the bilateral filter are described in Section 3.1.4

Technically, this is achieved by a convolution with a kernel that consists of a product of two non-linear weighting functions. The first one penalizes the spatial distance, whereas the second mapping only takes deviations of the values into account. Here, we introduce the bilateral weighting function ω which is used in the actual filtering process later on.

$$\begin{aligned} \omega(f,i,j) &= G_{\sigma_s}(||i-j||) \cdot G_{\sigma_v}(|f(i)-f(j)|) \\ f &: \mathbb{N}^2 \rightarrow \mathbb{R} \\ i,j &\in \mathbb{N}^2 \end{aligned} \tag{2.1}$$

As parameters, it takes a mapping f, a center pixel i, and a neighbor pixel j, both within the domain of f. The symbols G_{σ_s} and G_{σ_v} each denote one-dimensional Gaussian kernels with corresponding standard deviations. As the default, we apply the same adaptive adjustment for both parameters as proposed in the implementation of [PD06][1]. To be precise, σ_s is chosen as $\frac{1}{16}$ of the smaller resolution of f (in x- or y-dimension), whereas σ_v corresponds to 10% of its value range.

A bilateral filter is not separable but can be approximated [DD02, PD06]. Its computation is simple and can be done in parallel. The idea is independent of the domain and, among other things, also finds application in the denoising of meshes [FDCO03]. To gain deeper insight into the characteristics and the concept of bilateral filtering, we refer to [PKTD08].

2.2.3 Presentation

The output of our algorithms is again a depth map. In order to display a static result in a figure, we triangulate the range image according to the 2D lattice and displace each vertex by its corresponding entry along the z-direction. The resulting mesh is then rendered using Phong shading. We vary the material colors, but always use white specular reflections and one single white light source.

[1] http://people.csail.mit.edu/jiawen/software/bilateralFilter.m

Model sources

The models that we use to depict the effects and capabilities of our algorithms throughout this part of the thesis have been provided by courtesy of the sources listed below:

- Stanford 3D Scanning Repository: Lucy statue, armadillo, bunny
- XYZ RGB Inc.: Thai statue, dragon
- Aim@Shape: Lion-dog, lion vase, filigree, Caesar, cup, pharaoh mask, Greek statue
- Google 3D Warehouse: Cinderella castle
- Cyberware: Satva statue
- The Digital Michelangelo Project: David head
- Graphics and Geometric Computing Group, Tsinghua University: Robot 3D animation sequence

2.3 Related Work

We classify the existing approaches according to their input. Sometimes a unique assignment is not possible because some methods make use of several representations, and so the transitions between different classes are continuous. We restrict ourselves to introducing methods that yield proper shape information as output, and hence do not cover bump-mapping or other approaches that cause pseudo-relief effects on images.

2.3.1 Reliefs from Geometry

In this subsection, we present techniques that are, without exception, designed for relief generation from a given 3D scene. The concepts mainly differ in their domains. One class manipulates differential properties (gradient domain) only, whereas the others also include the shape information (range domain) directly to achieve the desired goal. Most of the techniques are specifically devoted to detailed and lifelike reliefs, whereas others aspire a stylization of the content.

Pioneering Work

In [CMS97], the authors put forth the idea of using a height field by projecting the geometry of a scene to the viewing plane. They distinguish the depth map pixels according to their saliency with respect to the current vantage point.

In order to produce a bas-reliefs, they apply a compression function that is inversely proportional to the height value. This results in a stronger diminishing of scene elements that are far away from an observer and has less effect on the more salient parts. In other words, regions at a similar depth level are treated the same way, regardless of what type of feature they belong to.

For high reliefs, they propose to first decompose the scene into a near and a far region. Then, the more distant parts are compressed as described above and added back to the unmodified foreground layer. This has the benefit that changes in the viewing angle on the relief can make hidden or partly occluded objects visible.

Although this idea works fine for slight compression ratios, in terms of the visibility of details, this method hardly does better than linear rescaling when it comes to a significant shrinking. The authors note that such perspective foreshortening even relatively enlarges edges on a surface, and so a significant amount of the depth range remains wasted if these regions are not specifically treated. Unfortunately, this work only operates in the range domain and disregards the differential properties, which are characteristic at surface singularities and visually important structures like steps, ridges, and valleys.

Gradient Domain Techniques

Instead of projecting the shape to the viewing plane (for capturing a height field), the approach by [SBS07] first measures the saliency on the surface of a given mesh [LVJ05] under a certain viewpoint and then describes the obtained and projected saliency values in differential coordinates. They subsequently use unsharp masking with a Gaussian kernel to enhance fine features. After reconstructing the new height field, a finalizing linear rescaling is applied to achieve the desired depth range. [SBS07] were the first to investigate the importance of derivatives for bas-relief generation in order to distinguish between large and small surface features. Nevertheless, their method appears slightly complicated overall and their results do not look lifelike enough to justify the effort it takes.

Our previous work, presented in [KBS07] and [Ker07], adapts the idea of exploiting gradient information. A global, absolute thresholding is performed to eliminate extraordinarily large gradients as they appear on silhouettes and along occlusion boundaries. This results in flat but obvious transitions that encircle and emphasize different regions but that no longer occupy unused depth range.

Unsharp masking with a Gaussian filter is applied to boost fine and visually important details. After such strengthening, their perceptibility is preserved even for very high compression ratios. This approach is very simple, fast, and produces results of reasonable quality. Nevertheless, the outcomes tend to appear unnaturally exaggerated because a Gaussian filter is not feature-sensitive and can cause undesired peaks during the enhancement step.

This problem can be avoided if a more elaborate filtering is applied. [WDB+07] make use of a silhouette preserving isotropic diffusion filter, which preserves the sharpness at gradient discontinuities. The authors propose a multi-scale approach that enables an artist to steer the relative importance of features at different frequency bands. Besides offering more artistic freedom, this makes it possible to selectively suppress noise, for example. To date, this approach produces the most successful, high-quality results in terms of sharpness, precision, richness of detail, and naturalness. The quality and flexibility of this method are attained at the cost of user-friendliness and performance. It requires a great deal of intervention, as there are many (sometimes non-intuitive and model-dependent) parameters to be set. In addition, it actually requires several minutes to compute a result. This can make the production of satisfying reliefs a very time-consuming process, if a user is less familiar with the approach.

The five publications that have been named so far [CMS97, SBS07, KBS07, Ker07, WDB+07], mark the only approaches that were available in this class at the beginning of the author's PhD phase. Chapter 3 shall report on our own progress and the achievements we were able to bring to the entire field since then. The following geometry-based methods have arisen concurrently to our further development.

Range Domain and Hybrid Approaches

The bas-relief generation method presented in [SRML09] operates directly on the height field but makes use of gradient information for additional re-weighting during the compression. It allows to distinguish features on multiple scales and relies on the concept of adaptive histogram equalization [PAA+87], which is primarily used for local contrast enhancement in images. The algorithm is suitable for bas-relief generation and produces very natural and detailed results, competitive to [WDB+07]. Unfortunately, adaptive histogram equalization is computationally expensive, and their initial implementation is very time-consuming. In addition, a user can influence the outcome by adjusting up to six parameters, and almost every one of them requires an entire re-computation.

The algorithm presented in [BH11] uses both domains as well. It triangulates the height field first and then applies a smoothing on the derived mesh to extract the details by subtracting both surfaces [KCVS98]. These details are then described in Laplacian coordinates and stored for later reuse [SCOL+04]. The gradient field of the smoothed surface is computed and compressed using a non-linear mapping. After that, the new, thin height field is reconstructed from its manipulated gradients. The previously extracted small and high-frequency features can then be transferred back to the surface. The motivation for this hybrid approach is to ensure that details remain completely unchanged, rather than being boosted to visually survive the gradient compression as it has been done by other approaches. On the other hand, it makes the method more vulnerable to noise on the initial model. The authors also describe how Laplacian sharpening, as an optional post-processing, can be used to further emphasize details in the generated relief.

In terms of user-friendliness and performance, the work of [ZZZY12] follows the same goals as we do in this thesis. The authors do not work with height fields, but rather operate on the input mesh directly. Using a bilateral filter, the surface is first split into a base mesh and a detail part. The base mesh is then mapped to a view-dependent plane and compressed with an adaptive linear function. After that the details are added back. A user can control the amount of fine features and the desired height limit. Depending on the number of triangles their implementation achieves real-time performance.

Sunken Reliefs

The work of [WKCZ11] focuses on the generation of sunken reliefs. Motivated by ancient, chiseled examples they strive for a suggestive stylization of a scene. First, a binary line drawing is derived from a given 3D model [RDF05]. Repeated morphological operations are proposed to clear the image from very small undesired crease lines. Alternatively, a smoothing of the initial input should be used as a pre-processing step to get rid of too-high-frequency responses. After producing a tidy line image, they project it on a planar mesh and set undercuts at the appropriate locations. The method is intuitive and demonstrates that, for a not-too-complex scene, a reduction to just a few coarse feature strokes is sufficient for producing convincing sunken reliefs. This approach contrasts with the other techniques in that it does not aim at achieving highly detailed and curvy results.

Recently, in [WCKZ12], an algorithm was presented that extends the work of [WKCZ11] and bridges the gap between different representations of the model. In addition to the projected line drawing, which extracts features directly from the 3D representation, a depth map (2.5D) is generated from the same viewpoint.

This depth map is compressed by attenuating its gradients in a non-linear way and acts as a base layer. Finally, an additional 2D image of the model with a Lambertian surface is rendered. If a single light source coincides with the camera position, the resulting image allows one to track the behavior of the surface normal. Hence, it contains information about very smooth, yet visually important and view-dependent, transitions that help to recover the height deviation. The knowledge from 3D, 2.5D and 2D is then assembled by a weighted energy minimization approach. The observation that geometric contours and visual cues have to be extracted and treated in different ways makes it possible to add more suggestive power since a larger variety of features is contained in a particular outcome. The fact that the depth information is taken into account here leads to a curved surface in areas where no lines were detected. Their results look promising and appear much more plastic and convincing compared to the sunken reliefs produced by their prior algorithm.

Gist

The methods presented so far differ mainly in terms of user friendliness, efficiency, and visual quality with respect to plausibility, detail preservation, and sharpness. All in all, it is not surprising that algorithms with a high degree of flexibility, artistic freedom, a high demand for user intervention, and long computation times yield the most impressive results. This is a grievance which this thesis hopes to address.

The advantage of a given 3D scene is that the perspective, and even the composite, can be adjusted. Neither much artistic imagination nor additional practical skills are required from the user. This does not hold for the following categories which rely on two-dimensional input or incorporate completely interactive techniques.

For the sake of completeness, we describe additional, in part distantly related, methods based on 3D input in the remainder of this subsection.

More Geometry-Based Work

Since relief generation from given shapes is a narrow field, we also cover the following publications. They are very recent and, apart from the abstract, only available in Chinese. For this reason, we have to rely on translations and summaries of their content. These have kindly been provided by Meili Wang.

The system presented in [ZL10] appears to be concurrent with our real-time approach [KTB+10]. It operates in the gradient domain and uses a non-linear global compression technique based on the arc tangent. It seems that the authors do not split the signal, and so they omit the feature enhancement. The tool is

implemented on the GPU such that parameter adjustments and viewpoint changes can be witnessed in real time. Handling animated models is left for future work.

In [LLZX11], the visible part of a 3D model is decomposed into three components using the Laplace operator. A base layer and two components of different frequency bands are compressed with an individual compression function before they are reassembled to form the result. This work operates entirely in the range domain and seems very related to our range-domain method which is presented in the next chapter [KTB+10].

The technique presented in [LLL12] is based on manifold harmonics spectral analysis. Discrete mesh vertices are transformed into the frequency domain and split into a high frequent, a low frequent, and a noisy part. The noise is ignored and the other frequencies are each compressed in a different way. Mapping the modified frequencies back to the range domain via reverse manifold harmonics transformation results in the desired relief.

The following three papers specifically focus on mapping reliefs from a given 3D model onto a second curved surface. In [PZ10] the foreground part of a height field is extracted by detecting its silhouette. This boundary condition, together with manipulated gradient entries, is used to reconstruct the relief right on a desired shape. The work of [LLZ12] additionally uses a bilateral filter to enhance and preserve the details. The algorithm described in [HX10] has similar goals, but they also treat the second surface as a depth map. The gradients of both surfaces are compressed, blended, and integrated accordingly.

Relief Extraction

If a virtual shape (or a scan of a real-world object) contains a relief on its surface, there are algorithms specifically designed to isolate it from its background. Since this idea of obtaining a relief is dissimilar to a proper creation, we only describe these approaches briefly at the end of this subsection.

A segmentation method for reliefs on triangle meshes is presented in [LMLR06]. They start with a rough hand-drawn polygon on the surface and contract it until it coincides with the boundary of the relief. The extraction becomes more challenging if the underlying surface is not smooth, but textured. Solutions for this case are detailed in [LMLR07b].

Snake-like approaches, like the above ones, are able to adapt to concave sculptures, but they fail to detect the background within a relief if the foreground surrounds it. This problem is dealt with in [LMLR07a]. First, a continuous background is estimated by fitting a B-spline surface patch to the area surrounding the relief which refined afterwards. Thus, a distinction between the relief and the surface becomes possible everywhere.

An alternative way to decouple a base layer and a height function defining the relief is described in [ZTS09]. First, an adaptive Gaussian low-pass filtering is applied to the surface normals, and then the corresponding base layer is estimated from these new normals. After that, the relief offset function is computed by solving a global minimization problem. A thresholding step is used to assign regions to the foreground or the background. Finally, a refinement step is proposed to overcome issues introduced by noise or very sharp edges.

In [CCL+11], the authors describe a different way to detach a relief from a background. They make use of differential coordinates. The underlying smooth and continuous surface is fitted by reconstructing it from the given normals, whereas those with significant changes (along a boundary) are re-estimated such that all normals in a local neighborhood share a common orientation. This idea marks an improvement compared to the previous approach, where differently oriented normals contributed to the result. Furthermore, the authors demonstrate that editing operations like global transformations or local deformations of the relief can be performed directly. In this case, such modifications directly benefit from the representation in differential coordinates.

These examples conclude the related works in the category of geometry based relief generation.

2.3.2 Reliefs from Images

All algorithms in this class aim to reverse-engineer a 3D surface that has produced a given 2D input. In general, this is an ill-posed problem. One reason for this is the bas-relief ambiguity. Although it appears to be a blessing for artisans, this ambiguity is found to be a curse for scientific disciplines like computer vision and shape reconstruction. Furthermore, the luminance entry in an image usually does not correspond to geometric shape properties.

Assumptions about the camera setup, lighting conditions, the type of model, surface reflectance properties, or even depth information need to be included in order to resolve the ambiguity and ensure an appropriate solution [AMK07, TMQ+07, CKK05]. In some cases, researchers must rely on human observation and knowledge to guide the generation of suitable shapes, e.g, by providing additional visual cues.

Shape from Shading

One scientific discipline which has intensively studied this problem is known as *shape-from-shading* [Hor70, HB89, ZTCS99].

Traditional shape-from-shading methods are not intentionally designed for relief generation. Nevertheless, more recent algorithms, which may include user intervention, can be used for this purpose. Among the huge number of publications that address this problem, we pick two examples to illustrate possible solutions for the creation of reliefs.

In [ZMQS05], the authors propose an interactive approach that efficiently resolves the bas-relief ambiguity by adopting human knowledge. Their method requires a user to set a reasonable surface normal first. Shape from shading is then applied locally to reconstruct each surface patch, and then the local solutions are combined to form a smooth global surface.

[WSTS08] describes an interactive system for reconstructing surface normals from a single image. First, the previous shape-from-shading algorithms are improved by reconstructing a faithful normal for local image regions. Then, low-frequency errors are corrected using a simple mark-up procedure. However, there is a high demand for user intervention to achieve bas-reliefs of reasonable quality. In the case of high reliefs, the effort increases drastically. There are some other limitations, as it works well for simple materials but manifests problems when using colored images or ones that contain a complex texture, as input.

Other Inverse Problems

The method described in [WCPZ10] occupies an intermediate stage. Given a 2D image as input, the authors implicitly regard it as shape information. The image is first converted to greyscale, and then the pixel luminance values are considered as entries of a height field. After that, they proceed as in [Ker07] to produce a three dimensional bas-relief. Instead of a final linear rescaling, they propose applying gamma correction to further equalize the visibility of features in areas of different depth levels. The method is also limited to images with a low texture complexity because varying colors can lead to undesired distortions in the outcome.

The automatic approach presented in [AM10] follows a somewhat converse idea. Instead of making sure that a relief looks faithful under one constant lighting condition, they investigate designing it in such a way that the appearance differs when it is illuminated from different directional light sources. They achieve this goal by placing small pyramids at the center of each image pixel and deforming them according to the desired reflectance properties. This algorithm is capable of producing bas-reliefs that contain information about a pair of input images in one single piece of art. Moreover, it can also transfer the color information of a given image to the relief representation if directional color light sources are applied. This method is the first to exploit the nature of reliefs and their ambiguity to use them as a type of display.

Their subsequent approach [AM11, Ale12] makes use of pits instead of pyramids. Each such cylindrical hole corresponds to an image region of several pixels. The depth of a pit, which is responsible for the amount of light to be consumed, depends on the brightness of the respective patch. Their results can easily be machined and lead to an interesting novel representation.

The work presented in [WMR+13] is specifically devoted to the generation of bas-reliefs of human faces from a given frontal photograph. Their technique consists of two components. First, they perform a learning step by analyzing the interplay between a rendered image of a 3D model of a human head and the image of a corresponding bas-relief, achieved with [SRML09]. After that, an input photograph is relit under different conditions. Using the knowledge from the first step, these differently relit images are transformed to images of reliefs. They then serve as input to a shape-from-shading approach in order to produce flat surfaces. In the end, they are properly combined to a single bas-relief. Once the learning is done, their method produces results in less than five minutes.

A reverse engineering problem for the purpose of cultural heritage was investigated in [LWYM12]. Given a single imprint, the goal is to reconstruct the chiseled relief that was used as a printing block. To achieve the rough structure, the authors detect object contours first, and then extract their skeleton. The height at these locations is estimated by taking into account the local extension. After that, the information is transferred to a mesh representation. A diffusion between the values at the skeleton and the background concludes the low-frequency base layer. The high-frequency details are directly contained in the initial image and are added to the low-frequency part in order to assemble the final relief. Aside from real imprints, the method is capable of computing virtual stamps from arbitrary pictures.

A related, traditional type of art, known as *Choshi*, is presented in [TMH10]. Given a colored input image, it is first segmented in same-color patches. Then, the algorithm yields templates for cutting several differently-colored layers of paper and explains how to overlay them in order to create a representation with a stylized, yet similar, impression. Although this method produces very coarse cartoons which omit details, it can be very useful for relief generation purposes since the different layers can be regarded as a counterpart to discrete iso-height levels.

Sketch-based Approaches

Using hand-drawn or automatically-derived line sketches as templates lets the relief generation process shift into the area of image-based modeling [OCDD01] or sketch-based modeling [OSSJ09, CA09]. In the first case, different regions of a 2D input are manually assigned a depth order to reconstruct the underlying geometry.

Two ways to derive 3D offset functions, given simple 2D contours, are described in [PSS01]. One idea is to convert an implicit polygon sketch to a monotone formula. This is achieved by composing the descriptions of convex and concave polygon parts in a set theoretic manner. The second way is to convert a scattered line drawing with varying gray scales to a depth function by approximating it using finite element methods. The resulting formulas can then be evaluated, and a relief can then be mapped onto an arbitrary surface. Their results show that these methods are only useful for non-complex reliefs.

Recently, [KLST11] provided a semi-automatic tool that processes line drawings for mapping reliefs to a base surface. First, they extract curves from the input [Ste98] and detect junctions and margins from them. A graph-based approach is used to determine the height levels at transitions between adjacent elements. The Laplacian of the relief layer is used to reconstruct the entire relief by smoothly fitting it on the base shape. The authors describe an additional manual fine-tuning step for post-processing the automatically generated result in order to fix misinterpreted curves.

Gist

Since an image is already given, no further artistic imaginativeness is necessary, but changes in the viewpoint or the scene content are not possible. This is why the methods depending on 2D input are less flexible than their 3D counterparts. In general, additional user intervention and practical skills are helpful, and in some cases necessary, to achieve satisfying outcomes.

2.3.3 Reliefs from Scratch

Another way to achieve a relief is direct fully-interactive modeling. Common 3D modeling tools like 3DS Max, Maya, Catia, Blender, or SketchUp, just to name a few, allow a user to create, combine, manipulate, and edit surfaces. Such modeling is a laborious and time-consuming process with multiple steps and it requires an experience on the part of the user to achieve visually pleasing results. This is because the above mentioned tools belong to the category of computer-aided design software, which serves more general needs rather than being specially developed for artistic purposes.

By way of comparison, computer-aided manufacturing software, like ArtCAM, JDPaint, Type3, or 3Design, provides special tools or templates which tend to assist in the construction of a relief-like geometry [WCZ10].

Freeform Sculpting

Interactive virtual sculpting is a discipline in computer art which models a variety of tools like hammers, prickers, carving knives, or differently shaped gouges and their particular impact on virtual surfaces in multiple different ways.

The work of [Coq90] proposes to use freeform deformations of lattices to manipulate an underlying shape. In [WK95], a solid material block and multiple tools are represented on a discrete voxel grid, and the deformations are considered as boolean operations. In the real-time system presented in [MOT98], the initial material is a wooden block, described as a constructive solid geometry. The tools are represented as ellipsoids, and an artist can individually control their elongation. The carving takes place at intersections of material and tool in 3D space. As an application, the authors demonstrate how a woodcut can be used as a printing block to do virtual print-making with the previously designed carving. Aside from carving, all sculpting methods above can attach material as well by using each operation inversely, which marks a drastic improvement compared to manual crafting.

A sculpting framework which introduces digital clay was developed in [PF01]. It is intentionally designed for creating virtual characters for the entertainment industry. The key ingredient to modeling the behaviour of clay is the concept of adaptively-sampled distance fields [FPRJ00]. This efficient representation is a scalar field which contains information about signed distances between points and a shape. Many samples are taken in detailed regions, and a coarser sampling is applied in smooth areas. Hence, the necessary memory usage is reduced without compromising precision. An additional organization into an octree data structure further accelerates the operations. The algorithm also accepts range data as input.

In [Sou01b] and [Sou01a], the surface and the tools are both described by mathematical functions [PSS01]. Modifications, like undercuts or bulges and their transitions, are represented as offset or set-theoretic operations. In contrast to the above-mentioned sculpting systems, which start with a solid block of material, the author focuses on flat sheets of metal or wood to produce virtual pieces of art by free-form carving and embossment.

Gist

All techniques in this subsection operate directly in 3D space to manually design a plastic object. The advantage over crafting is that the virtual tools allow one to undo modifications that have already been made, and that it is easy to edit and combine intermediate results or to replicate a final outcome.

One drawback that all these methods have in common is that the entire production process is time-consuming and needs painstaking user intervention. The quality of the outcomes depends heavily on the skills, experience, creativity, and imaginativeness of the artist.

Chapter 3
Easy and Rapid Relief Design

Simplicity is a great virtue but it requires hard work
to achieve it and education to appreciate it.
And to make matters worse: complexity sells better.

Edsger Wybe Dijkstra (1930 - 2002)

In this chapter we present our main extensions and contributions to the research in the field of relief generation from three-dimensional models. These methods have been developed to allow an accessible and quick creation of visually pleasing results, even for laymen. To achieve this convenience, the main goal is the reduction of required user intervention by saving parameters and finding adaptive settings. Providing a graphical user interface and speeding up the computation further improve the ease of use and allow new applications, like reliefs of animated models. In addition, we offer more artistic freedom by demonstrating how to design seamless multi-perspective and multi-object reliefs.

3.1 Relief Generation Using Bilateral Filtering

To follow, we will give an in-depth description of two relief generation algorithms both of which exploit the properties of a bilateral filter in different ways.

3.1.1 Gradient Domain Approach

As described in Section 2.1, ridges and valleys are vital features for a human observer. Their sharp preservation is mandatory for generating convincing reliefs.

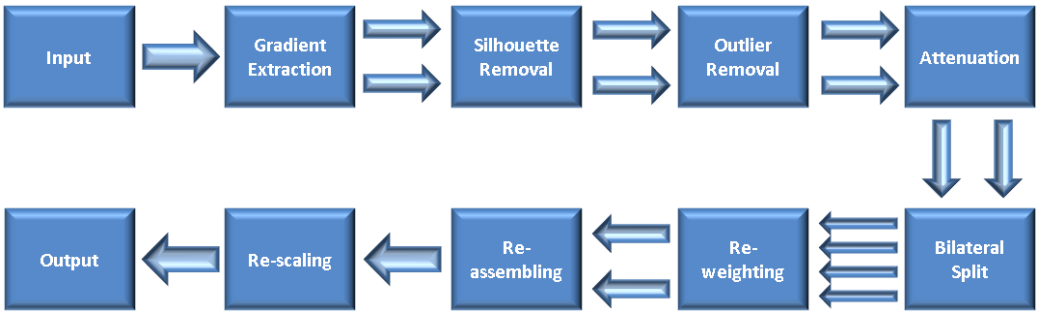


Figure 3.1: A flow chart of our gradient domain approach.

These visually important regions of a surface occur at its local curvature extrema (second derivative). In other words, they coincide with the edges of the first derivative. Hence, the edge-preserving capability of a bilateral filter transforms to a *ridge-and-valley-respecting* smoothing, when it is applied to the gradient field of a signal. This observation marks the key idea of our first approach.

Algorithm Overview

Given the normalized input height field, we first compute its partial derivatives. After that, we derive masks that allow us to mark and specifically treat locations with singular properties that would negatively affect the upcoming computations. An attenuation step leads to an initial non-linear compression of the content. Then, the gradient signals are split by applying a modified bilateral filter which disregards the previously detected falsifying regions. At that stage, the visually important fine details are enhanced using unsharp masking such that they remain perceivable in the complanate result. Given the new modified gradient signals, we have to solve a partial differential equation to compute the relief. Subsequently, its extent is then adapted by a finalizing linear rescaling. A graph of the different algorithm stages and their interplay is depicted in Figure 3.1.

Detailed Description

Gradient extraction: As a first step, we extract the x- and y-gradient fields from the input.

$$g_x = \frac{\partial h}{\partial x}$$
$$g_y = \frac{\partial h}{\partial y}$$

In practice, these derivatives are computed by a forward difference. For the purpose of compactness and to increase readability, we henceforth use the subscript $k \in \{x, y\}$ to express that an operation is executed for both gradient dimensions.

In general, this gradient field exhibits artificial responses at the boundary where the background area meets foreground objects. These extraordinarily high entries would negatively affect the upcoming steps and impair the quality of the outcome, since they would remain perceivable at the cost of smaller details. Hence, these regions have to be detected and ignored in order to avoid singular behavior.

Binary masks: Fortunately, the gradient of the previously extracted background mask can be directly used to acquire the necessary information automatically. We extract a second binary *silhouette mask* s that marks these contours. It is equal to zero if and only if a foreground pixel has at least one direct background neighbor and vice versa:

$$\begin{aligned} s &: \mathbb{N} \times \mathbb{N} \rightarrow \{0, 1\} \\ s(i) &= \begin{cases} 0, & \text{if } |\frac{\partial b(i)}{\partial x}| + |\frac{\partial b(i)}{\partial y}| > 0 \\ 1, & else \end{cases} \end{aligned}$$

Right after that, we set the corresponding entries of the gradient fields to 0 by component-wise multiplication.

$$g_k = g_k \cdot s$$

Now that the silhouettes are removed, we still face the problem of steps directly on an object's surface. Moreover, self-occlusions and regions where several objects overlap can cause large gradients as well. Again, keeping those would lead to undesired behavior. The presence of such coarse transitions in the final relief would be too dominant compared to the perceptibility of small-scale features. To detect those areas, we opted for a relative thresholding that depends on the value range of the remaining foreground pixels. It leads to a third binary *outlier mask* o with null entries at those pixels that deviate too much:

$$\begin{aligned} o &: \mathbb{N} \times \mathbb{N} \rightarrow \{0, 1\} \\ o(i) &= \begin{cases} 0, & \text{if } |g_k(i) - \mu_k| > t * \sigma_k \\ 1, & \text{otherwise} \end{cases} \end{aligned}$$

Here, μ_k represents the mean value, and σ_k stands for the respective standard deviation of all foreground pixels in g_k (excluding the silhouette). The tolerance parameter t is adjusted by the user and allows control over the strictness of this criterion. It answers the question: *What shall not be preserved?*

For large values of t, no pixels will be marked, and this step will not have an effect; hence, larger steps will still be contained in the result. If it is chosen to be too small, almost everything will be regarded as an outlier. This would lead to large completely planar regions in the outcome. Our experiments have shown that a setting between 3 and 5 is a reasonable choice for t. The detected outliers will also be immediately removed from the gradient field.

$$g_k \quad = \quad g_k \cdot o$$

An important positive aspect of this approach is the fact that zero gradients at those discontinuities lead to flat transitions. They encircle distinct scene elements in the outcome and thus emphasize the impression of a step without wasting any depth range. In other works, such a thresholding was based on an absolute value which could vary from model to model. In contrast to our relative thresholding, a meaningful initial setting could not be proposed.

Since the values in the detected troublesome pixels have been artificially influenced, they must not affect the subsequent steps. Therefore, we finally combine the three masks into one, in order to distinguish those pixels later on. This mask m contains zeros at the background, the silhouette and all outlier positions; the entries are equal to one everywhere else.

$$\begin{aligned} m \quad &: \quad \mathbb{N} \times \mathbb{N} \rightarrow \{0, 1\} \\ m \quad &= \quad b \cdot s \cdot o \end{aligned}$$

Attenuation: After removing silhouettes and outliers, the remaining entries in g_k now contain only the visually important information about the constitution of the foreground surfaces. As a next step, we need to reduce the amplitude of the gradient values in order to achieve a first compression of the interval range. Therefore, we apply the following polynomial attenuation function [FLW02]:

$$g_k(i) \quad = \quad g_k(i) \cdot \left(\frac{a}{|g_k(i)|} \cdot \left(\frac{|g_k(i)|}{a} \right)^b \right) \tag{3.1}$$

The parameter a marks the magnitude for which this mapping behaves neutrally. Entries below it are slightly boosted, whereas those above it are diminished. The two parameters a and b are chosen according to the values proposed in [FLW02]. Here, a is determined adaptively as 10% of the average absolute value of all unmasked pixels in g_k. Parameter b steers the intensity of the attenuation. In practice, a constant value of 0.8 is applied for all the results of our gradient domain approach. Obviously, pixels with entry 0 remain unchanged.

All in all, this step has two benefits. It leads to a shrinking of the depth interval and relatively strengthens smaller values at the same time. Nevertheless, it operates globally and does not take local features into account. A deeper analysis of this attenuation function and a detailed evaluation in comparison to alternative approaches for non-linear compression are given in Section 3.1.4.

Unsharp masking: Now that the gradient signal is free of singularities and has undergone a first compression, we want to decompose the remaining signal into low- and high-frequency components in order to boost the fine details appropriately. As mentioned above, we exploit the edge-preserving capabilities of the bilateral filter for this step. The fact that we operate in the gradient domain ensures that the sharpness of ridges and valleys on the surfaces is not compromised. Moreover, the information contained in the mask m helps us to ignore pixels which would falsify the result. We denote the low-frequency part of g_k by L_k and the high-frequency part by H_k:

$$\begin{aligned} L_k(i) &= \frac{m(i) \cdot \int_j m(j) \cdot \omega(g_k, i, j) \cdot g_k(j) dj}{\int_j m(j) \cdot \omega(g_k, i, j) dj} \\ H_k &= g_k - L_k \end{aligned} \tag{3.2}$$

This equation describes a convolution. Here, $m(i)$ ensures that masked entries remain unchanged, whereas $m(j)$ guarantees that they do not influence the computation. The function ω (see Equation 2.2) yields the bilateral weights and the denominator is used to normalize the result. In the end, L_k contains only the coarser features of g_k. Subtracting both leads to the fine details which are now contained in H_k.

We then assemble the new gradient components $\overline{g}_k$ by changing the relation between L_k and H_k:

$$\begin{aligned} \overline{g}_k &= \lambda \cdot L_k + H_k \\ 0 < \lambda &< 1 \end{aligned}$$

Since we diminish the coarser structures rather than increasing the details, this step inherently results in a depth compression. The user-defined parameter λ controls the amount of the enhancement. It answers the question: *How strongly will the details be emphasized?* Choices for λ close to zero lead to a loss of low-frequency features and hence to slightly exaggerated results. A value close to 1 has almost no effect at all and the perceptibility of fine details will suffer. Obviously, this setting depends on the desired compression ratio. In practice, we found out that for a reduction to 2% of the initial depth extent, a choice of $\lambda = 0.2$ leads to convincing reliefs. In other words: for a compression of 1:50 a detail enhancement of factor 5 is suitable.

Poisson reconstruction: Now that the new feature-enhanced gradient components are derived, the task is to reconstruct the corresponding height field out of them. Here, we face the problem that after a modification the gradient field is, in general, no longer integrable. To get back to the range domain, it is therefore necessary to compute the height field $\overline{h}$ for which the deviations of its gradient to $\overline{g} = \overline{g_x} + \overline{g_y}$ are minimal in a least-square sense:

$$\underset{\overline{h}}{argmin} \int_{\Omega} (\nabla \overline{h} - \overline{g})$$

Using the above-mentioned energy functional in the Euler-Lagrange equation leads us to the well-known partial differential Poisson equation:

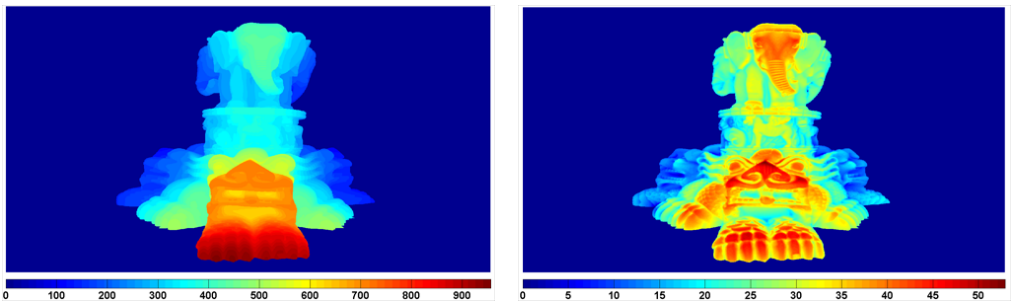
$$\begin{aligned} \Delta \overline{h} &= \nabla \overline{g} \\ \nabla \overline{g} &= \frac{\partial \overline{g}_x}{\partial x} + \frac{\partial \overline{g}_y}{\partial y} \end{aligned}$$

Please note that this is in fact an equalization of second derivatives. The solution for this diffusion problem is well-studied and requires solving a sparse system of linear equations with respect to the boundary conditions in $\overline{g}$ [Bra01]. Due to our preprocessing, we can be sure that the entire frame of $\overline{g}$ is equal to zero. This fact simplifies the solution and ensures that the outline of the relief remains flat and is not bent, as would have happened when there were nonzero entries in the boundary.

Post-processing: The intermediate height field $\overline{h}$ is already close to the final outcome. All visually important features have been enhanced and the depth is drastically reduced. Unfortunately, its remaining extent is not predictable beforehand, so we need to adjust it by applying a finalizing linear rescaling. Due to the fact that the required shrinking is much smaller than it would have been at the beginning, a linear compression is acceptable at this stage. Now, even the fine details will remain perceivable.

As we did in the preprocessing, $\overline{h}$ is normalized by setting the background and the smallest foreground entry to zero with the rest of the foreground protruding from it. The achieved depth range $R_{T=1}$, is then equal to the maximal foreground value after this normalization. Let $R_{T=2}$ denote the third and last user-defined parameter that determines the desired and the maximum allowed height. It depends on the specific application or intention and answers the question: *How strong will the compression be in the end?* Thus, we compute the actual relief as follows:

$$\begin{aligned} \overline{h}(i) &= b(i) \cdot (\overline{h}(i) - min(\overline{h})) \\ R_{T=1} &= max(\overline{h}) \\ \overline{h} &= \frac{R_{T=2}}{R_{T=1}} \cdot \overline{h} \end{aligned}$$



(a) (b)

Figure 3.2: The range image of the XYZRGB Thai-statue socket before (a) and after (b) our gradient domain relief generation method.

Eliminating unwanted discontinuities, manipulating the gradient magnitudes, and affecting the interplay of different frequencies of the signal has lead to heavy distortions of the surface. These deformations are intended in order to achieve the desired ambiguity effect. Figure 3.2 contrasts an input height field and the final depth map of the corresponding relief which has been attained using our gradient domain algorithm. Note the strong emphasisation of small details and the very different interval ranges.

3.1.2 Range Domain Approach

Inspired by our first experiments with depth compression operating only in the range domain, which led to results of relatively poor quality, we have decided to further investigate this concept. Rather than using only two components, as has been done in [KTB+09], we now add an additional split such that three layers with different properties can be re-weighted.

Algorithm Overview

Given a normalized range image as input, it is first decomposed into a base layer and a detail layer with the help of a bilateral filter. The latter contains information about the constitution of the surfaces and is further divided into a coarse and a fine component. Therefore, we use a modified version of the Laplacian diffusion that is adapted to our needs. A user can now reassemble the shape by influencing the relative importance of the three layers. This is done in a twofold unsharp masking process. Since the sensitive features have been enhanced at this stage, a finalizing linear rescaling can be used to achieve the desired compressed outcome. Figure 3.3 illustrates our range domain relief generation pipeline.

Figure 3.3: A flow chart of our range domain approach.

Detailed Description

Base layer extraction: As we have mentioned earlier, edges are not visible from an orthogonal vantage point. The fact that they cover a huge amount of the entire initial depth range offers a high compression potential. Nevertheless, these steps contain essential information about the depth order of different patches, and they convey a more plastic overall impression by casting small shadows. Therefore, our first goal is to decompose the given signal h into a base layer h_{base} which roughly describes the shapes in the scene and a detail layer h_{detail} with information about the constitution of the object surfaces. To do so, we apply a bilateral filter which smoothes the range image but preserves discontinuities like edges and larger self-occlusions. Again, background pixels must not have an effect on the upcoming steps. Therefore, we use the derived background mask b to distinguish and ignore these entries.

$$\begin{aligned} h_{base}(i) &= \frac{b(i) \cdot \int_j b(j) \cdot \omega(h,i,j) \cdot h(j) dj}{\int_j b(j) \cdot \omega(h,i,j) dj} \\ h_{detail} &= h - h_{base} \end{aligned} \tag{3.3}$$

In this convolution, the function ω (see Equation 2.2) computes the appropriate weights. Masked entries remain unchanged. This results in the desired, piecewise almost constant base layer h_{base}, and h_{detail} which contains the more sensitive features.

Laplacian-like diffusion: Given the detail component, we want to extract a layer h_{high} which contains the small-scale high-frequency details and a component h_{low} which contains slightly coarser and lower-frequency structures. Again, for the purpose of relief generation it is vital to preserve ridges and valleys, and especially their sharpness. This is why we opted for a Laplacian-like diffusion. The Laplacian operator Δ is widely used as a convolution mask for the high-pass filtering of an image.

$$\Delta = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

In our scenario, this means that the difference between a signal before and after smoothing contains the fine details as well as the desired responses at ridges and valleys. Nevertheless, we face the problem that h_{detail} still contains smaller edges and steps. Their relatively high gradient would inherently lead to a stronger compression. This is an issue because these components are not meant to be contained in h_{high} but should rather be preserved during the filtering and reflected in h_{low}. As a remedy to restrict the undesired influence, we have developed a slightly modified Laplacian operator by adding an appropriate denominator. Altogether, this step reads as follows:

$$\begin{aligned} h_{low} &= \begin{cases} h_{t=0} &= h_{detail} \\ h_{t+1}(i) &= h_t(i) + \alpha \cdot \frac{b(i) \cdot \sum\limits_{n \in N(i)} b(n) h_t(n) - h_t(i) \cdot \sum\limits_{n \in N(i)} b(n)}{1 + |\nabla h_t(i)|^2} \end{cases} \\ h_{high} &= h_{detail} - h_{low} \end{aligned} \tag{3.4}$$

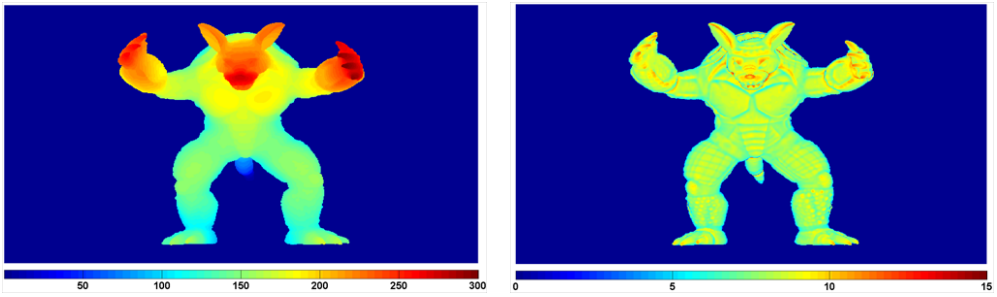
This describes an iterative process with repeated application of the modified Laplacian operator. Here, $N(i)$ stands for the set of the four direct neighbors of pixel i in the x- and y-directions. The binary mask b ensures that background pixels remain unchanged and do not have an influence on the foreground computation. In the case where all neighbors belong to the foreground, the numerator in this equation is identical to the standard Laplacian expression Δ. The symbol ∇ denotes the gradient. One can see that the additional denominator $1 + |\nabla|^2$ diminishes the effect of entries with a larger first derivative and thus prevents edges from causing undesired responses in h_{high}. The parameter α sets the time step size. It controls how fast the diffusion converges and how strong the smoothing will be. Smaller adjustments for α lead to more accurate but slower approximations. In practice, we always use a maximum of 20 iteration steps.

So far, we have split the initial shape into a very coarse base signal and two components each of which contains information about different aspects of the surface consitution.

Unsharp masking: We will now boost the influence of the high-frequency small-scale features of the surface relative to the low-frequency details. At the same time, we suppress the base layer linearly such that the importance of the entire detail layer is further enhanced.

$$\begin{aligned} \overline{h} = \lambda_1 \cdot I_{base} &+ \lambda_2 \cdot I_{coarse} + I_{fine} \\ 0 < \lambda_1 &< 1 \\ 0 < \lambda_2 &< 1 \end{aligned}$$

This results in a new height field $\overline{h}$ with adjusted weights for the three different components.



(a) (b)

Figure 3.4: The height field of the armadillo model before (a) and after (b) our range domain relief generation technique.

Smaller values for both parameters imply that more details will remain perceivable in the relief. For $\lambda_1 = \lambda_2 = 1$ the result would be the same as a linear rescaling of the input height field. Setting one of the parameters to 0 completely eliminates the respective component. In the case of the base layer this would, among other things, result in losing the correct impression of the depth order of scene elements.

Post-processing: As in the gradient domain counterpart, the intermediate depth range is not predictable since it depends on the model and the parameter setting. We proceed in the same way as above and linearly scale the relief $\overline{h}$ to the desired extent. Thanks to the enhancement that has been done so far, such a shrinking no longer harms the visibility of fine structures. Let $R_{T=1}$ denote the achieved depth interval size and $R_{T=2}$ the desired one, then the final outcome $\overline{h}$ is computed as follows:

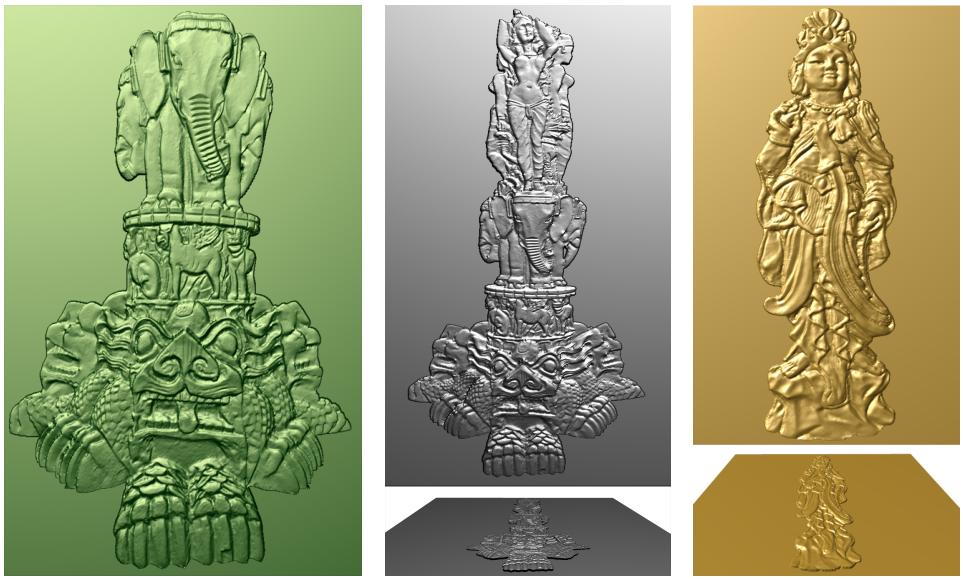
$$\overline{h} \quad = \quad \frac{R_{T=2}}{R_{T=1}} \cdot \overline{h}$$

Figure 3.4 illustrates the effect of our range domain algorithm. The input height field is compressed to 5% of its former depth. Note that the tail and the head are mapped to a similar range and that features like muscles can be distinguished.

3.1.3 Results

Digital Reliefs

The quality of a relief result cannot be objectively measured in terms of correctness. Nevertheless, an outcome can be evaluated by rating its naturalness, plausibility, depth impression, richness of detail, sharpness and preservation of features at different scales.



(a) (b) (c)

Figure 3.5: Rendered reliefs attained with our gradient domain technique. The Thai statue base (a), the entire Thai statue (b), and the Satva statue (c).

The reliefs we show here have all been achieved from a static input and have been compressed such that their depth does not exceed 2% of their largest extension in x- or y-direction. As an example: A height field of 800x1000 pixels is mapped to a maximal height of 20 times the pixel spacing. For the computation, all parameters have been chosen in the default ranges that were proposed above.

Figure 3.5 contains three results generated with our gradient domain algorithm. Part (a) shows the rendered version of the Thai statue pedestal relief (see Figure 3.2). One can see that sharp and small features like the dragon's teeth and scales are faithfully reproduced. The relief on the center pillar as well as the cavities of the elephant's trunk are clearly visible. In Figure 3.5 (b) the full Thai statue is displayed together with a tilted version to demonstrate the flatness of the relief. Finally, part (c) contains the gradient domain outcome for the Satva statue. Various important characteristics like the garment wrinkles, the jewelry and the facial expression are all well preserved.

In Figure 3.6 (a) we show a fourth gradient domain result. Here, the lion vase model was flattened without compromising the visibility of the high-frequency hair in the mane and low-frequency features like the nose, the muzzle, and the spherical adornments at the bottom. The other two images display outcomes of the range domain technique. Figure 3.6 (b) contains the rendered counterpart to the result for the armadillo shown in Figure 3.4. The step between the jaw

(a) (b) (c)

Figure 3.6: The lion vase model compressed using the gradient domain algorithm (a). Results of our range domain method: the complanate armadillo (b) and a relief of the filigree model (c).

and the breast as well as the self-occlusions on the ears are emphasized. The surface structure and certain features like muscles and fangs are readily visible. The ornament in Figure 3.6 (c) demonstrates that the depth in particular order remains correctly preserved.

More results of other models are shown as comparisons in the discussion section 3.1.4.

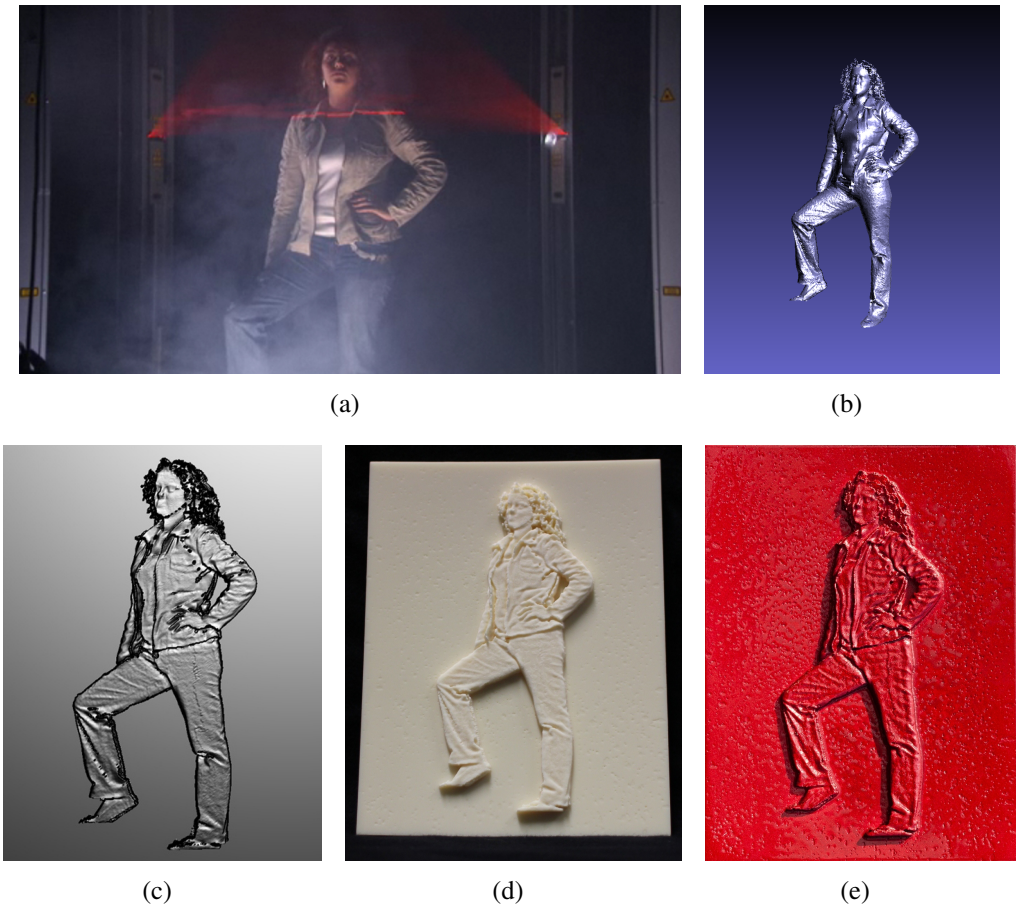
Relief Prototyping

To demonstrate the practical relevance and capabilities of our techniques we have added a small extension at the beginning and the end of our pipeline to rapidly produce tangible reliefs of real-world objects, e.g. as a souvenir.

In our example we use a 3D full body scanner to quickly achieve a mesh representation of a human model which then directly serves as input for our gradient domain technique. This yields a virtual relief which needs to be transformed to a watertight mesh by adding a thin frame before it can be printed. The 3D printer we actually used to produce the prototype (in 2010) operates with numerous thin layers (0.1 mm) of photo polymer that are separately cured using UV light.

Figure 3.7 depicts the different stages of the extended prototyping pipeline. The top row contains an image of the capturing process[1] and the obtained 3D model. The bottom row shows the virtual relief, the untreated real-world counterpart right after printing and the result after the use of varnish, clear paint and a final polishing. The relief perceptibly contains features on a variety of scales like

[1] still image of the video at http://www.exzellenz-initiative.de/saarbruecken-multimodal-computing-interaction, last visited January 24th 2013



(a) (b)

(c) (d) (e)

Figure 3.7: The laser scanning of a human body (a) and the corresponding 3D model (b). The rendered digital relief generated with our gradient domain technique (c), an untreated 3D printout (d), and the result after additional manual work (e) (Photograph by Bertram Somieski).

the large steps along the open jacket, the high frequency parts on the curly hair, the wrinkles on the trousers, the fingers and even the facial expression. The ground plate is of size 15x20 cm and the relief rises to a maximum of 0.8 cm above it. The small holes on the right part of the jacket and along the chin which can be observed in the virtual result are due to acquisition artifacts of the scanner.

Scanning and creating the virtual relief together took less than two minutes, whereas the printing took about two hours. This can of course be accelerated if a more recent or different type of printer like an engraving laser or a CAM milling device is used. More information about the runtimes of our algorithms is given in Section 3.2.

3.1.4 Discussion

In this subsection, our algorithms are critically evaluated. Aside from comparing the results to the ones of related methods, we also contrast some intermediate steps with concepts that were used in other approaches and discuss the particular pros and cons.

General Remarks

As alluded to above, there is no real objective quality measure for reliefs. The evaluation is always in the eye of the beholder.

By design, our methods can only work effectively if the object surface contains fine structures. If the model is very smooth or contains only coarse spherical elements, the decomposition and the boosting will not significantly affect the outcome.

The resolution of the depth map is crucial, especially for the gradient domain relief generation pipeline. Not only does it allow one to better distinguish very fine details, but especially on an object's contour it is important that such features are at least three or four pixels thick, since the silhouette detection erodes one pixel on each side that touches the background. On the other hand, only a sufficiently large number of foreground pixels allows a meaningful and representative outlier detection.

Our gradient domain approach can achieve almost arbitrary compression ratios because steps are eliminated and details can be boosted to the desired level. It is therefore suitable for very shallow bas-relief generation. Only the reflectance properties of the material and the lighting conditions can be a limitation, because minor deformations might not be perceivable any longer.

For its range domain counterpart, the base layer contains vital information about the depth order. The problem is that it consumes a relatively large range, since it still contains steps. This layer either impairs the visibility of fine features if it remains too prominent, or it is compressed so much in the favor of details that the overall plastic impression is compromised. This balancing act is the reason why the range domain technique is better suited for mid- and high relief generation and becomes less effective if the desired compression ratios are too high. So the decision between the two techniques should be driven by the constitution of the scene and the desired depth range.

Bilateral Filter

Here, we will contrast the bilateral filter, as we have used it, with two filtering techniques that have found application in other approaches.

(a) (b) (c)

Figure 3.8: An example case illustrating the individual principles of the Gaussian (a), the bilateral (b), and the diffusion filter (c).

Using a binary pattern as an example, Figure 3.8 illustrates the different principles of a Gaussian filter, a bilateral filter, and a diffusion filter . Each image describes the scope of those values that are taken into account for the weighted averaging. In each case, the pixel in the center marks the kernel origin. The dark and bright stripes indicate regions of different values. The colored area, from yellow to red, describes the respective weight at the corresponding position. One can see that the Gaussian filter averages only by the spatial distance, regardless of the underlying values. Dark and bright pixels at equal distance have the same influence, although the center pixel is bright. In the case of a bilateral filter, the effect of entries with differing values is prevented, as illustrated by the gaps. However, it continues to include regions that are not spatially connected to the origin of the kernel. In contrast, for a diffusion filter the high gradients at the transitions act as a barrier, preventing the area of influence from spreading to non-connected patches.

Simple Gaussian blurring, as it was used in [KBS07] and [Ker07], turned out to be not suitable for relief creation purposes because it lacks the feature sensitivity which is reflected in unnaturally exaggerated results. This is due to the fact that sharp edges, and in fact ridges and valleys, are washed out during the decomposition, which leads to undesired peaks in the high-frequency part. An additional boosting by unsharp masking artificially introduces artifacts around ridges and valleys, which is counterproductive as it wastes height that could be compressed. Applying a bilateral filter instead overcomes this issue and marks a significant improvement in comparison to [KBS07, Ker07], because it preserves the sharpness of a step and thus guarantees a precise signal analysis. Figure 3.9 depicts the difference for a 1D example.

On the one hand, the very precise iterative silhouette-preserving diffusion filter applied in [WDB+07] is even “more correct”, since potentially falsifying distant regions are also excluded.

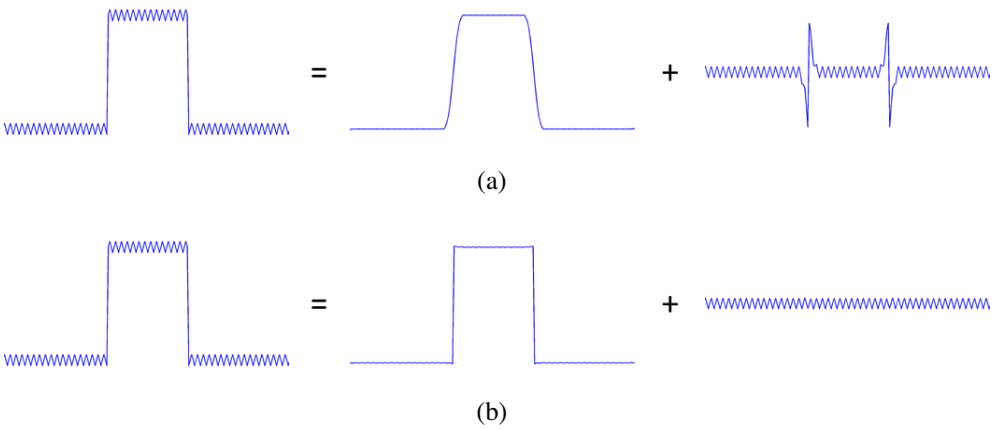


Figure 3.9: An example signal decomposed using Gaussian blurring (a) and a bilateral filter (b).

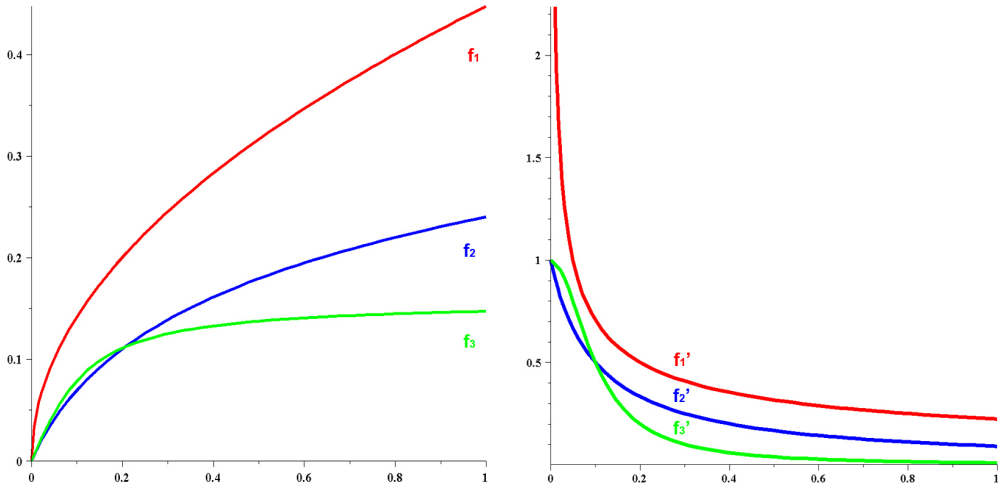
On the other hand, it is complex and computationally very expensive, which noticeably slows down the entire process. Our experiments have shown that, in practice, a bilateral filter is sufficient to produce convincing results. Moreover, its parameters can be set adaptively and it can be computed in parallel. This is the reason why we opted for the bilateral filter as a good and convenient compromise in terms of feature sensitivity, simplicity and efficiency.

Attenuation

During the manipulation of the gradient field, the non-linear compression of the amplitudes has been a relevant step. The goal is to achieve a higher compression for large entries than for small values, which inherently leads to a relative enhancement of fine details. Please recall that this is a global operation which is not sensitive to any features like edges, ridges or valleys. Such an attenuation would not make sense for the range domain, since features of like size would be treated differently if they were situated on different height levels. In the relief generation literature we find three different types of non-linear attenuation mappings.

The polynomial method we apply was originally used in [FLW02] for HDR purposes. Aside from our approach, it also found application in [Ker07, WCPZ10, WCKZ12]. We slightly reformulate the relevant expression from Equation 3.1:

$$\begin{aligned} f_1(x) &= \frac{x^b}{a^{b-1}} \\ \frac{\partial f_1}{\partial x} &= \frac{b}{a^{b-1}} \cdot x^{b-1} \end{aligned}$$



(a) (b)

Figure 3.10: A graph of three different types of attenuation functions (a) and their respective first derivatives (b).

In [WDB+07, SRML09], the authors make use of a logarithmic rescaling:

$$\begin{aligned} f_2(x) &= \frac{1}{\alpha} \cdot log(1 + \alpha \cdot x) \\ \frac{\partial f_2}{\partial x} &= \frac{1}{1 + \alpha \cdot x} \end{aligned}$$

Finally, [BH11] use an alternative mapping based on the arc tangent:

$$\begin{aligned} f_3(x) &= \frac{arctan(\alpha \cdot x)}{\alpha} \\ \frac{\partial f_3}{\partial x} &= \frac{1}{1 + (\alpha \cdot x)^2} \end{aligned}$$

In f_2 and f_3, the user-defined parameter $\alpha > 0$ steers the compression ratio. The attenuation function applied in [ZL10] is comparable to f_3 but uses two different parameters for the numerator and denominator. Let us now investigate the properties of those functions in more detail.

Figure 3.10 (a) contains a plot of the these attenuation mappings and their particular derivatives (b). We want to stress that the parameters have been chosen such that the asymptotic behavior becomes obvious, and that different settings might be used in practice.

By applying function f_1, small-scale details (below the threshold a) are not compressed, but rather are additionally boosted, which further emphasizes them

relative to the larger values. This behavior might be unwanted if noise is contained in the original data set. Another problem is the fact that this function flattens out slowly, as can be seen by inspecting its derivative. In contrast, mapping f_2 compresses the entries at all scales. The attenuation linearly becomes stronger for larger values. Attenuating a signal using f_3 affects large entries much more strongly because its derivative exhibits a quadratic drop-off. This means that gradient values in a wider range are almost equalized.

Figure 3.11 shows multiple reliefs of the Lucy statue. The first one (a) is achieved without any attenuation; it is followed by reliefs for which the above-mentioned functions have been applied. As a common basis, we have used a MATLAB implementation of our gradient domain approach [KTB+09]. All depth intervals are shrunken to 10 units in depth and they are rendered under the same lighting conditions and in the same pose. It is difficult to see a difference in this side-by-side comparison. Therefore, the images in the second row indicate the difference between (a) and each of the other results.

The regions around the leg and the robe exhibit the greatest differences in all cases because this is where the larger gradients at the wrinkles are situated. Those have been affected most. The individual effects become apparent by the color distribution at the contours of the wings, the chest and the rock. The difference between (a) and the results achieved with f_2 and f_3 look similar, but note that the different scale illustrates the stronger compression.

We have to admit that the stronger compressions achieved with the attenuation functions f_2 and f_3 is a great advantage over our method. Nevertheless, we are convinced that introducing another non-intuitive model-dependent parameter is not in line with our goals of simplifying the overall procedure and reducing the need of user intervention. The polynomial mapping f_1, on the other hand, has the benefit that we can propose a reasonable parameter choice (a adaptively, b fixed) which turned out to require no further tweaking to achieve plausible outcomes. On top of that, this function additionally enhances fine details in only a single step.

Comparison

From our point of view, the technique of [WDB+07] produces the most impressive results and can be regarded as the current state of the art in terms of visual quality. This is why we first contrast our results with theirs and discuss the individual strengths and weaknesses. Relief meshes of two different models were provided by courtesy of the authors. Figure 3.12 shows a side-by-side comparison. The left column was achieved with our gradient domain method, the center column contains the results of [WDB+07] and the reliefs attained with our range domain approach are displayed on the right.

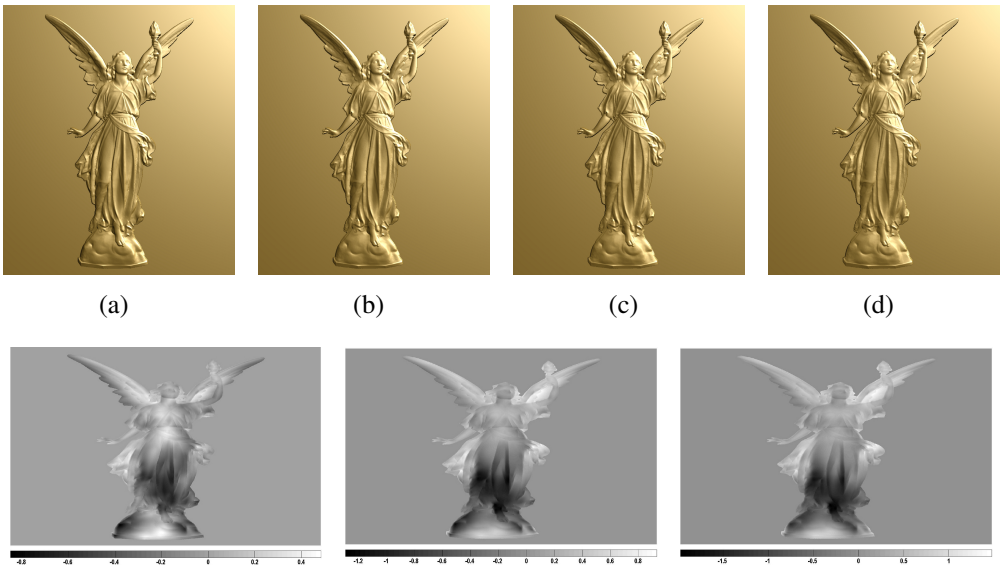


Figure 3.11: Reliefs achieved with varying attenuation functions: no attenuation (a), using function f_1 (b), applying f_2 (c) and f_3 (d). The lower row depicts the following height differences: (b)-(a), (c)-(a) and (d)-(a), from left to right.

For the Cinderella castle, it becomes obvious that the range domain approach produces the better result compared to the gradient method. This is due to the fact that the model does not exhibit many high-frequency structures, but instead consists of planar regions and large steps which are better reproduced because they are covered in the base layer. In contrast, the eyeball and the claws of the dragon are reproduced more faithfully in the gradient domain method and do not appear as flat as it does in the range domain counterpart.

We have to admit that in both cases the results of [WDB+07] convey a more salient appearance. Their approach gives the opportunity to decompose the gradient field into an arbitrary number of frequency levels. Among other things, this allows distinguishing between very small details and noise, and a selective suppression of specific bandwidths. Such a level of quality and control, however, comes at the price of long computation times (diffusion filter for each gradient component, see above) and the necessity for laborious re-iterations during design. This is because a number of, in part, non-intuitive and model-dependent parameters have to be adjusted with consideration given to their interplay.

Figure 3.13 demonstrates how the relief generation techniques have evolved. We use the armadillo model as an example and show the results in chronological order. All images are shown exactly as they appear in the respective publications.

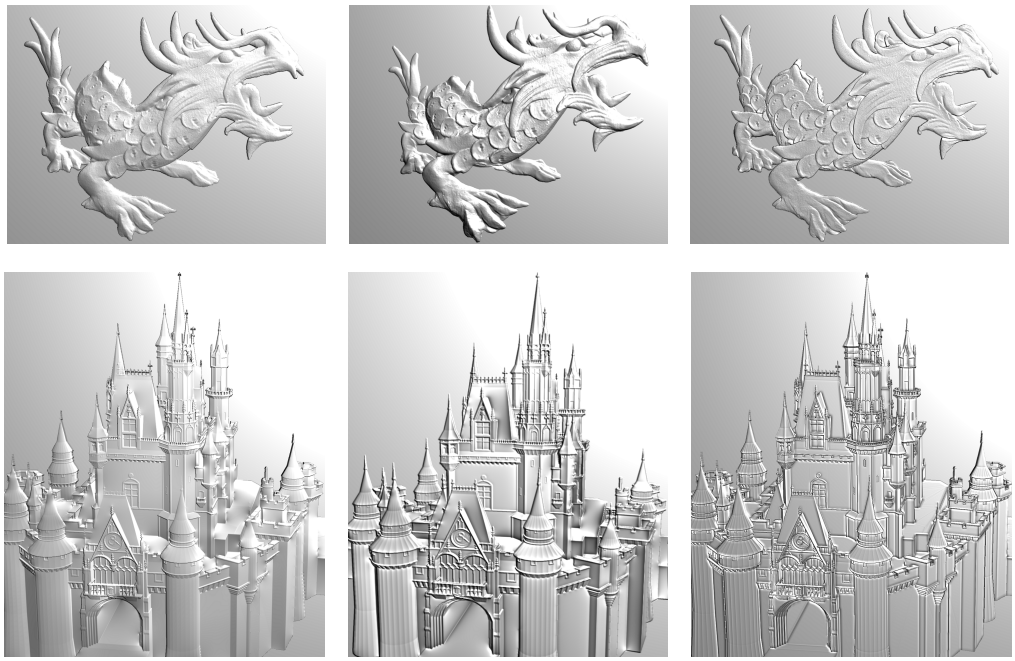
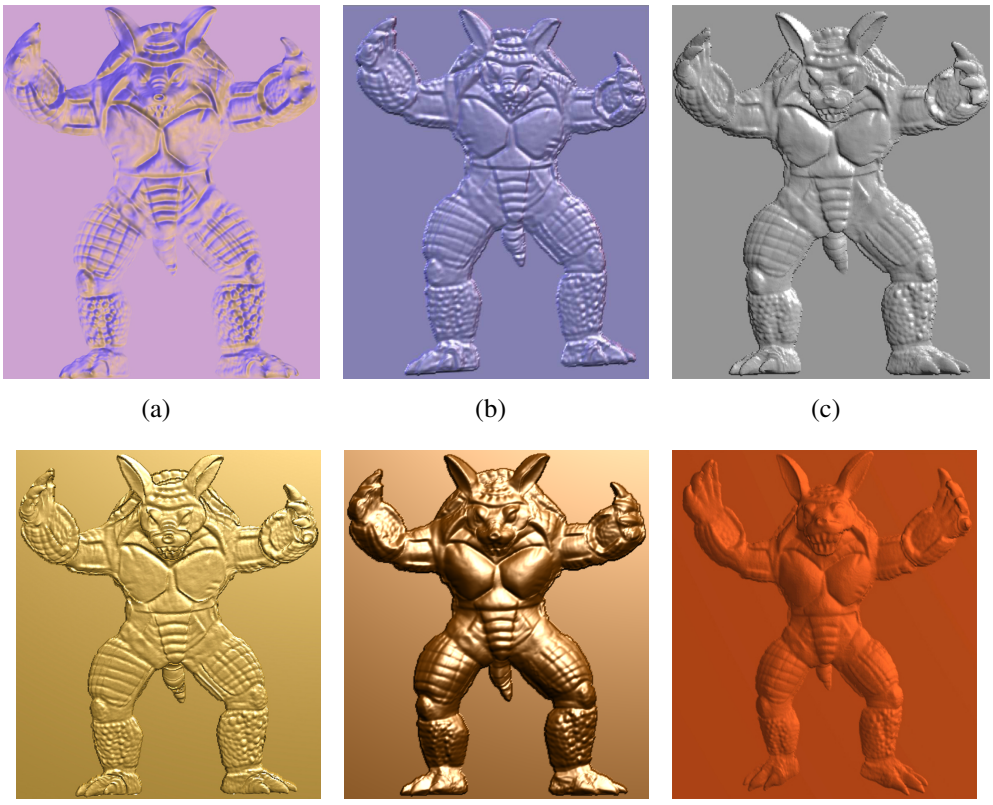


Figure 3.12: A comparison between results of our gradient domain approach (left), the technique of [WDB+07] (middle) and our range domain method (right). The XYZRGB dragon (top row) and the Cinderella castle (bottom row) have been used as examples.

This is the reason why material and pose differ. The figure contains the result of [SBS07] (a). Here, the inner parts on the lower legs and the heels appear to melt into the background. The jaw is hard to see and all in all it does not look very plausible. Our earlier work [Ker07] in (b) faced the problem that the contours of the head and the transitions of the teeth could not be preserved. The result of [SRML09] (c) is followed by our range domain outcome (d) as has already been shown in Figure 3.6. Both reliefs are comparably sharp and detailed. Part (e) shows the result of our gradient domain technique and in (f) the armadillo relief of [BH11] is shown. Here, a fair comparison is difficult, since the perspective differs noticeably from that of the other examples.

Altogether, we can say that by accepting small, yet smart, compromises and restrictions, we have managed to increase the simplicity and convenience of relief generation without sacrificing the visual quality of its outcome. The conceptual advances have been achieved by keeping the overall pipelines as intuitive as possible and reducing the user intervention to a necessary minimum at the same time. This contributes to our desired goal of providing algorithms that allow even untrained users to easily and successfully produce visually pleasing reliefs.



(a) (b) (c)

(d) (e) (f)

Figure 3.13: Reliefs of the armadillo model achieved with different methods in chronological order.

3.2 Relief Computation in Real-Time

Now that the concept is reworked, the next step is to provide an appropriate tool to further increase the ease of use. Here, we focus on efficiency such that the effect of the remaining adjustments immediately becomes apparent.

In previous works, the depth map capture and the processing pipeline are not connected. This has led to static applications, which require tedious work to generate satisfying results. Every time a user decides to adjust the pose of a model, the camera parameters or the resolution, it is then necessary to capture a new height field, store it, and run the relief generation pipeline again. Moreover, a new input usually demands the tweaking of multiple parameter values by trial and error, which also requires a complete recomputation. Depending on the algorithm, every such run can take several minutes.

In our application, we capitalize on the highly parallel nature of our methods and exploit the properties of modern graphics hardware. Besides drastically increasing the performance, this approach offers access to the z-buffer data which can be directly used for further processing. This allows us to devise an OpenGL application that implements the full algorithmic pipeline of our techniques. In this way, we avoid the overhead of reading the depth data back to the CPU or writing it to a file as input for an external application, as it was necessary in former approaches.

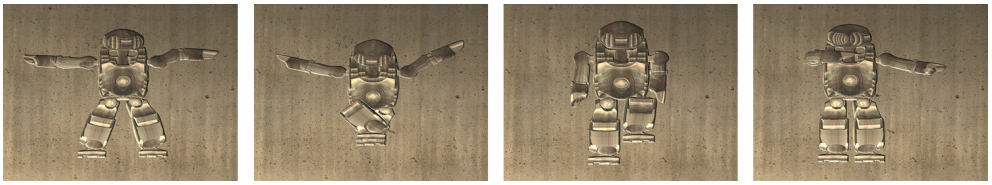
In our implementation, all intermediate results are stored as floating-point textures, such that the entire pipeline remains on the GPU. This results in an interactive tool that achieves real-time performance and allows a user to witness the effect of controlling parameters on the fly. Thus, it facilitates finding appropriate settings. Furthermore, editing the scene by transforming objects or adding more models has an effect on the relief without noticeable delay. As a direct result of this performance boost, our implementation also makes it possible the generation of reliefs of dynamic scenes.

3.2.1 Graphical User Interface

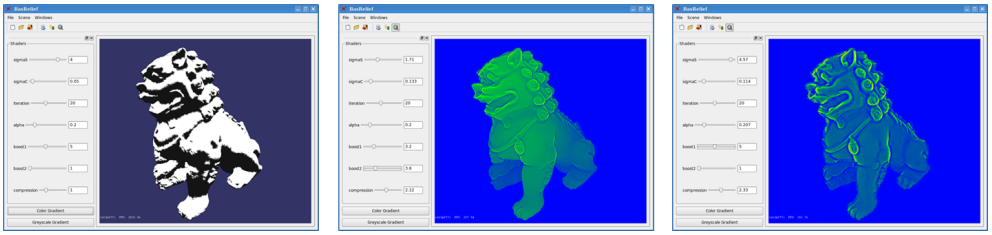
To be precise, we have set up an individual graphical user interface on top of the hardware implementation of each of the two presented techniques. Although they have most of the functionality in common, we adapt the usability to each approach. Below we briefly list the properties of these interfaces. The difference in the provided features should not be regarded as a restriction but rather lets us rate their utility in practice. All functionality can easily be transferred into the other scenario.

Gradient Domain Approach: The current interface supports the import and export of meshes, height fields, and 3D animations in well-established file formats. A user can also load a file with a set of control points for an animated path of an orthogonal or a perspective camera. Since in this special case there are only 3 parameters to be set, we opted for letting a user increase or decrease their magnitudes in small step sizes via the keyboard. Other than the static results shown above, we can also render the reliefs with arbitrary shaders and under predefined lighting conditions right in the interface. Figure 3.14 (a) contains 4 screenshots, each of which shows a different time step of an animated dancing robot scene. The relief is based on a material with a wooden texture.

Range Domain Approach: Here, we provide a QT viewing interface with two different modes. In the first mode, a user can import meshes and arrange the scene content by transforming the objects accordingly. The camera, which can be toggled between orthogonal and perspective projection, either remains fixed or can be rotated around an arbitrary axis.



(a)



(b)

Figure 3.14: Rendered reliefs of a dynamic scene in the gradient domain user interface (a). The input model and two colorcoded reliefs (achieved with different parameter settings) in the interface of the range domain GPU implementation (b).

In the second mode, the viewer displays the acquired height field which corresponds to a relief of the scene under the current perspective. We provide a number of sliders that allow the setting of the input parameters in meaningful ranges. The default setting for the standard deviations (see Equation 2.2) turned out to yield good results in general. Nevertheless, their proper adjustment can be more crucial here than in the gradient domain counterpart. Therefore, we offer the possibility to influence σ_s and σ_v as well. The relief is not rendered, but the depth is colorcoded. This makes the effect of controlling parameters become obvious in another way. Figure 3.14 (b) shows the initial lion-dog model and two outcomes achieved with different parameter settings. The result can be exported as a depth map or a triangular mesh.

3.2.2 Performance

In general, the computation time is independent of the scene complexity. The parameter choice theoretically affects the performance, but only plays a minor role in practice. Only the resolution and the number of background pixels can influence the runtime noticeably.

The Poisson equation can be solved via fast Fourier transformation [SS88, Bra91][2]. For our gradient domain method, we therefore considered it best to include NVidia Cuda, and especially its Fast-Fourier-Transform library *CUFFT*[3], for an elegant and very fast solution. The downside of this is the fact that the transfer from OpenGL textures into Cuda space consumes a significant percentage of the overall computation time, and thus marks the main bottleneck of the pipeline. The computation of the average values and standard deviations for the filtering, the outlier removal, and the attenuation step are achieved via mip mapping. Since we need multiple textures, we have to consider the available graphics memory as a limited resource. Regarding the specification of common modern hardware, this restricts us to reliefs of a resolution up to 9 megapixels for a graphics card equipped with 1.5 GB of memory.

The range domain approach uses fewer textures and can therefore handle much higher resolutions. Moreover, its implementation operates entirely in OpenGL since no Poisson reconstruction is required. Hence, no additional time-consuming Cuda texture transfer is necessary. This makes it very fast. Both GPU pipelines could further be accelerated, for example, by using a bilateral filter that works with Gaussian KD-trees [AGDL09].

Table 3.1 shows the average computation times for a relief of 1024x1024 pixels (Lucy statue), acquired with default parameters. We have measured the time on the hardware that was used in 2010 and compared it with more recent graphics hardware to demonstrate how this development has affected the performance. In the first case, an NVidia GeForce 8800 Ultra with 800MB of memory was used. In 2013, we repeated the same experiment on an NVIDIA GeForce GTX 580 equipped with 1,5 GB of memory. In order to point out the advantages of the GPU, we also list the timings for a single-thread CPU MATLAB implementation of our gradient domain method on a Intel Xeon X5650 with 2.66 GHz. The code makes use of a fast approximation of the bilateral filter [PD06]. This implementation was kindly provided by Jiawen Chen[4]. For this experiment, a static input height field was captured externally. The file IO duration is excluded.

In the earlier gradient domain implementation we used Cuda 2.0. In this case, the texture transfer consumed up to 70% of the overall workload. This problem has been addressed in Cuda 4.0, which was applied in 2013, and leads to a drastic speed-up. Nevertheless, one can see that the range domain approach is significantly faster because this step can be omitted. The GPU-CPU comparison shows that the pipeline on graphics hardware can achieve frame rates that are at least two orders of magnitude faster than its counterpart's.

[2] http://www.physics.buffalo.edu/phy410-505-2004/Chapter6/ch6-lec2.pdf
[3] http://docs.nvidia.com/cuda/cufft/index.html
[4] http://people.csail.mit.edu/jiawen/software/bilateralFilter.m

Method	GPU 2010	GPU 2013	CPU 2013
Gradient domain	25 fps	70 fps	8.0 sec
Range domain	180 fps	220 fps	–

Table 3.1: Average runtimes on different hardware configurations.

With such a real-time application we further increased the usability and efficiency of the design process. To the best of our knowledge, the processing of dynamic scenes in such a way could not be accomplished before. The accessible user interfaces and their functionality let us achieve our aim of providing a user-friendly solution for the rapid generation of convincing reliefs.

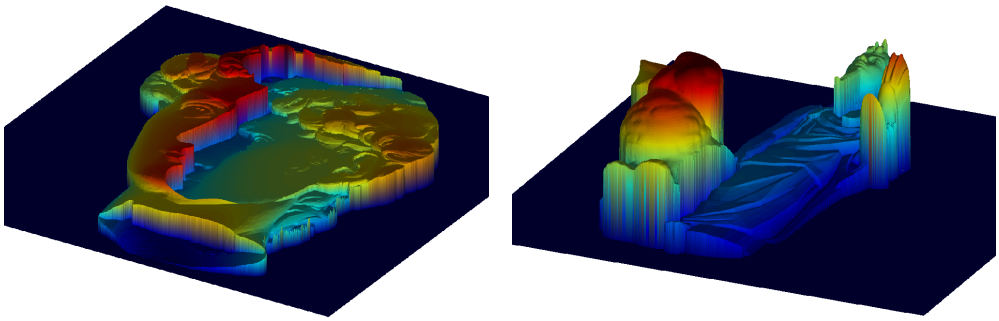
3.3 Assembled Reliefs with Seamless Overlap

In this section we demonstrate how a simple modification of our gradient domain method can assist in modeling more sophisticated relief sculptures. The simple setup of a single object is generally satisfactory for standard applications. Here, we want to increase artistic freedom by providing a solution for combined inputs. Such a branch of digital relief generation has not been investigated until now. In particular, our focus is on seamless transitions between different objects or viewpoints. One application is the generation of geometric collages for which a user can easily assemble the content by positioning multiple overlapping objects. Another field is cubism. In this type of art, multiple perspectives of one and the same object are merged into a single scene.

Again, simplicity and ease of use are the most important aspects here. This is why a composition of the input can take place directly in 3D space by arranging objects in a drag-and-drop manner, or by using a regular image editing tool to mosaic the individual height fields. In practice, we use a 32-bit TIFF format for the still depth maps. Today, most programs support this format and also provide dialogs for layers and transparency. Altogether, this makes it very convenient to generate multi-object and multi-perspective input.

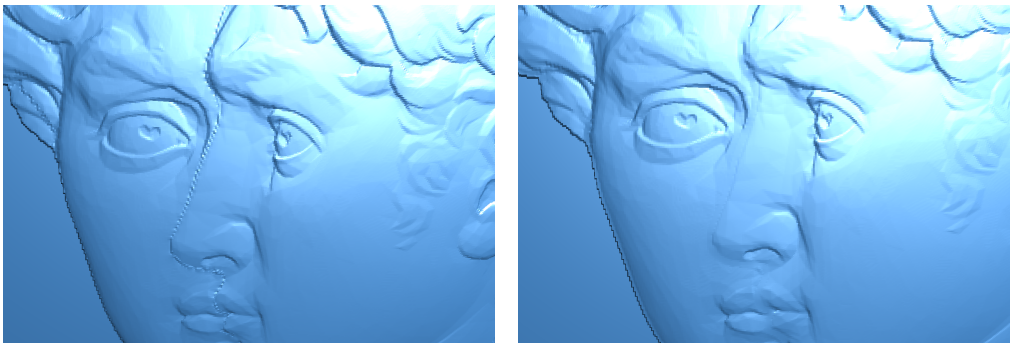
3.3.1 Challenge

Different shapes, or even different views on the same object, can exhibit very different depth ranges. Figure 3.15 illustrates the problem. In part (a) one can see that the front view is much more salient than the side-face. The collage (b) of 4 different objects (Greek statue, bunny, cup, pharaoh mask) shows that the objects themselves inherently exhibit very different extensions. These differences reflect in large jumps along the areas where two or more elements occlude each other.



(a) (b)

Figure 3.15: An assembled height fields of two different perspectives of the David head (a) and a depth map of collage with four different shapes.



(a) (b)

Figure 3.16: A relief with obvious transitions (a) and a seamless result (b).

The great benefit of our gradient domain approach is that the outlier detection finds these transitions automatically throughout the pipeline, without the need for further intervention. The remaining challenge is that setting those areas to zero results in a flat transition which emphasizes the impression of having distinct scene elements. This behavior was intended in the general pipeline, but in this scenario, it would definitely undermine the desired impression. Figure 3.16 (a) depicts the result of the unmodified algorithm. One can see that the step transforms to a visible seam along the transition area. Our experiments revealed that using a diffusion process or blurring those pixels in the spatial domain may lead to even worse results, as they introduce additional steps between modified and unmodified entries. It is therefore necessary to treat these singularities by appropriately manipulating the corresponding gradients before the reconstruction takes place.

3.3.2 Solution

An outlier affects its direct neighbors during the Poisson reconstruction step. Thus, we extent the outliers mask so that adjacent pixels are also considered. Let us denote this new binary transition mask by t. It covers all non-reliable positions and can easily be derived from o using a morphological dilation of the null entries. In other words, every binary pixel is mapped to the product of its 3x3 neighborhood.

Now we need to proceed by refilling the areas of t with meaningful values by a weighted average of the surrounding area. Therefore, we apply a Gaussian smoothing with a large support, but ignore all other masked pixels. This step reads as follows:

$$\begin{aligned} t(i) &= \begin{cases} 1, & \forall j \in N(i) : o(j) = 1 \\ 0, & else \end{cases} \\ m &= b \odot s \odot t \\ \forall i : t(i) = 0,\ g_k(i) &= \frac{\int\limits_{j \in g_k} G_{\sigma_s}(i-j) \cdot m(j) \cdot g_k(j)\, dj}{\int\limits_{j \in g_k} G_{\sigma_s}(i-j) \cdot m(j)\, dj} \end{aligned}$$

Here, $N(i)$ stands for pixel i and its direct neighbors. For the standard deviation σ_s of the Gaussian kernel G_{σ_s} a constant value of 8 has been chosen for all of the experiments in this section. Applying this additional modification to the gradient field before the reconstruction leads to the effect which is shown in Figure 3.16. The unsightly seam from part (a) has successfully been removed, which results in a continuous relief (b).

We want to mention that all detected outliers are refilled. This means that areas of coarser steps on an object itself are also affected and no longer lead to flat encircling lines. To be able to distinguish between the treatment of overlapping areas and surface edges, another threshold value would be necessary. Nevertheless, as our experiments have shown, such a distinction is hardly required in practice. This is demonstrated by the following results which have been achieved without another such parameter.

3.3.3 Results

Figure 3.17 contains four example reliefs achieved with this slightly extended pipeline. Parts (a) and (c) correspond to the the input height fields shown in Figure 3.15. The results demonstrate that the described method nicely blends together the outlines of different models or perspectives and thus supports the impression of watching a continuous scene.

The input for the first three exemplars has been assembled using an image editing tool. In contrast, the second collage in Figure 3.17 (d) has been set up right in 3D space. We want to stress that the three objects (Caesar, lion vase and bunny) do not touch in 3D. The head and the bunny are simply placed in front and over the vase such that they appear to touch in this fixed perspective. No mesh editing or similar actions have been necessary.

This simple, yet powerful, extension of the initial algorithm has proven to reliably remove disturbing transition artifacts from assembled scenes. Since the detection of singular areas comes basically for free, no additional user intervention is necessary. The described approach allows even hobby-artists to design more elaborate relief artworks in which multiple objects or perspectives appear to melt into each other.

(a) (b)

(c) (d)

Figure 3.17: Multi-perspective reliefs of the David head (a) and (b), followed by two results of geometric collages assembled from a number of different shapes (c) and (d).

Chapter 4
Conclusion

Art is a lie that makes us realize truth.

Pablo Picasso (1881 - 1973)

In keeping with this quotation, we have described the illusive nature of reliefs and the corresponding phenomenon of human perception. The combination of the two can cause a depth illusion and thus allows us to transfer a shape to a more complanate one without this being noticed.

We have presented two semi-automatic tools to generate relief sculptures from 3D models. The methods can achieve a depth compression without sacrificing the visibility of important features. In some cases, our results are significantly better than those of other methods. In comparison to the state-of-the-art technique [WDB+07], we can produce competitive results faster and with less user intervention.

With a simple modification, we have shown how to produce continuous digital sculptures with seamless transitions, even if their content is assembled from multiple views or a number of overlapping objects with different properties.

Implementing our methods on graphics hardware, and thereby streamlining the entire pipeline, marks the most important contribution. We have set up a graphical user interface and limited the number parameters to a necessary minimum. Our system is the first one capable of handling dynamic input and allows the user to witness interactive changes of any kind on the fly. This results in an intuitive real-time application, which is paramount, since we aim at enabling even enthusiasts who are not familiar with computer art to successfully and rapidly design convincing digital reliefs.

Regarding the manufacturing of reliefs from real-world objects, we have demonstrated that input from a 3D laser scanner can directly be processed, and a finalizing 3D printing step can yield life-like touchable exemplars.

4.1 Future Prospects

An interesting next step would be to integrate our system into an online shape repository in order to extend the range of services it offers. A simple dialog to choose the pose of an object and to set the few remaining parameters would be sufficient to create and download individual reliefs of the models contained in the database. In such a context where scenes commonly consist of only one object, it would also be useful to combine our methods with achievements from research on good view selection [SLF+11] to further facilitate the utility.

Recently published more advanced edge-aware image filtering techniques [PHK11, GO11, XLXJ11, SCHP12] are an alternative to the bilateral filter. Including them in our pipeline could further improve the relation between achievable sharpness and required computation time.

So far, the outcomes of most methods in this research area remain limited to a uniform substance. The effect of combining material layers of different colors or textures in one model has not yet been investigated.

Beyond our experiments with cubism, the influence of multiple or more complex perspectives has not yet gained interest. We see great potential in applying different camera models [YM04] for capturing the height field. Among other things, this could assist in generating input for large panoramic reliefs which can surround a beholder or decorate very long walls [RL06]. Elements from anamorphosis could further stress the relativity of perspective and lead to more complex, ambiguous, or "impossible" sculptures.

With regard to the increasing number of available animated 3D models and the advent of low-cost real-time depth sensors, like Microsoft Kinect, we see a rising demand to process entire range videos or to directly compute digital reliefs of a moving human actor. Artistic installations like an interactive relief mirror or the production of unique souvenirs with the help of a 3D-printer are be imaginable examples. Home-entertainment and home-fabrication are other potential areas of application. Against the background of our contributions, we regard the field of computer assisted relief generation to be well-prepared to face these upcoming challenges.

Part II

Scalable Symmetry Analysis

Chapter 5
Introduction

So that I may perceive whatever holds
The world together in its inmost folds

Faust [Part I, v. 382 f.]
J. W. v. Goethe (1749-1832)

The phenomenon of symmetry occurs in a number of natural forms, for example in the anatomy of organisms, cells, crystalline structures, and proteins. It is crucial for our aesthetic understanding and remains indispensable to our daily life, as it is present, among other things, in architecture, industrial design, fine arts, music, and geometry. Figure 5.1 shows various examples of objects which exhibit different symmetric properties [Mis13b].

The second part of this thesis investigates the analysis of symmetries in digital shapes, represented as massive three-dimensional point clouds. The task is to decompose a data set into smaller entities such that man and machine can draw conclusions on what the input is composed of and how it is actually assembled. In this context, symmetries can be regarded as a synonym for repetitive patterns, self-similarities, structural regularities, or redundancies within a given geometry.

This information is essential for a large and growing number of applications. The discovered redundancy can be utilized for compression [MGP06] and data cleanup by averaging matching parts [GSH+07, PMW+08, BBW+09, ZSW+10]. Furthermore, symmetry-based invariants have been exploited for symmetry preserving editing [GSMCO09, BWKS11, WXL+11, ZFCO+11], for semi-automatic creation of related shapes [MP08, BWS10, LCOZ+11], and to reason about the functionality of shapes [MYY+10]. For more detailed information about symmetries and their applications, we refer to this very recent survey [MPWC13].

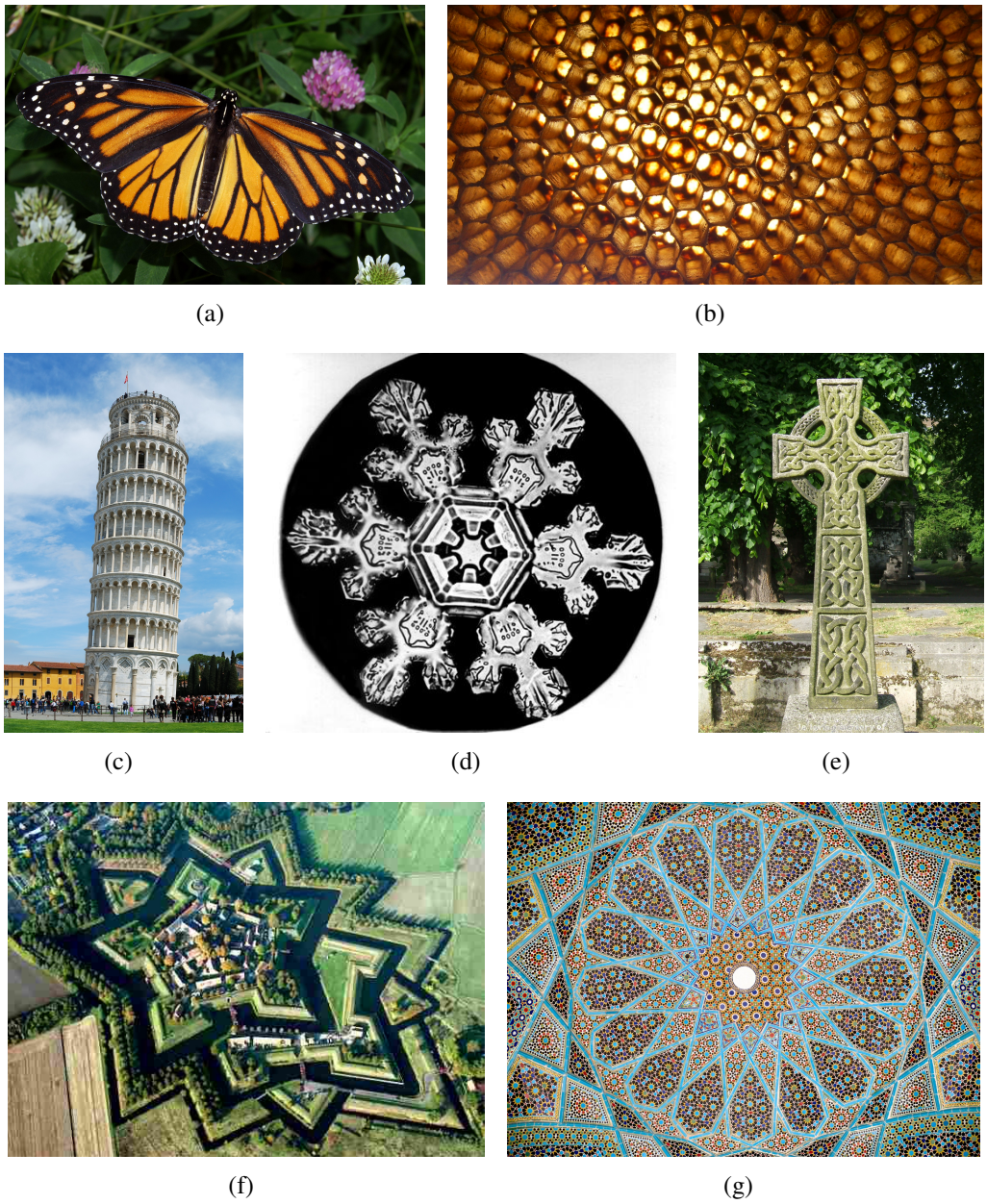


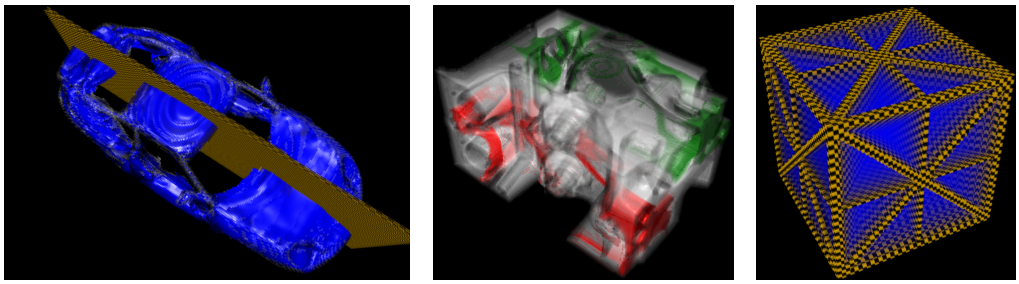
Figure 5.1: Selected examples of objects with different symmetric properties: A butterfly (a), a closeup of honeycombs (b), the leaning tower of Pisa (c), a snowflake (d), a Celtic cross decorated with Celtic knots (e), a fortress (f), a mosaic (g). All images shown are available at Wikipedia in the public domain or under a Creative Commons license [Mis13b].

5.1 Problem Statement

Symmetry detection algorithms published so far all share an important limitation. Current approaches do not scale to large quantities of geometry. The largest scenes for which results have been reported in the literature are in the range of a few hundred megabytes in size [MGP06, PMW+08, BBW+09]. However, there are application scenarios where the data sets to be handled are several orders of magnitude larger. For example, there are ongoing large-scale urban scanning campaigns, such as Google Street View, producing enormous quantities of 3D point clouds depicting most urban areas on this planet [MWA+12]. The rising interest in such projects and the progress in quality and density of 3D laser scanners will lead to a tremendous increase of data in the future. For such massive geometric data sets, there is an even higher demand for discovering redundancy through symmetry analysis than for small scenes. In addition to the obvious need for obtaining a compact encoding, applications such as non-local scan consolidation [ZSW+10] or structure learning approaches would clearly benefit from a larger database and could be expected to perform better. However, available symmetry detection algorithms are fundamentally limited in the input size that can be handled.

Many previous techniques are based on transformation voting [MGP06]. These approaches consider pairs of points with matching local neighborhood, compute a canonical transformation between them, and vote for them in transformation space, disregarding the spatial location. For a large scene with many symmetric elements, the transformation space is cluttered by the overlay of simultaneous matches and the structured noise that comes with them. Hence, discrimination among the various symmetries becomes increasingly difficult. Analyzing a city-scale scan in a single transformation space is clearly infeasible. Even larger buildings already require a hierarchical decomposition for disambiguation [MGP06].

An alternative approach is to match local feature constellations [BBW+08, BBW+09]. These techniques are more restricted as they require the presence of suitable features in the model. However, they can handle large amounts of symmetries with arbitrary structure. Nevertheless, the feature-based approaches essentially work by comparing all pairs of feature constellations explicitly. Some speedup is obtained by randomized sampling, but the scaling behavior is quadratic in the size of the scene. Furthermore, all of the processing is performed in-core and with random access to the data, so that the potential scene size is limited by available main memory. At moderate scene sizes, this is not yet an issue, but handling very large data sets is not possible. Other techniques, such as computing robust pairwise auto-alignments [SKS06] or moment analysis [MSHS06] are also not applicable to large scenes.



(a) (b) (c)

Figure 5.2: Reflective Symmetry of a car model with respect to the indicated plane (a), two repeating non-overlapping structures within an engine block model (b), and an indicated decomposition of a cube into 48 equally size tetrahedra by analyzing all its reflective symmetries.

5.2 Contribution

In this part of the thesis, we make a first step towards symmetry analysis in very large point cloud data sets. We present a new, scalable symmetry detection method that is designed to handle massive amounts of input geometry.

In order to obtain a scalable algorithm, we need to avoid explicit pairwise comparisons of matching geometry, which leads to unacceptable, quadratic complexity. Instead of that, we have developed a local descriptor that is directly mapped into a feature space such that similar geometry coincides. Hence, we can map the detection problem to a much simpler nearest-neighbor problem in a low-dimensional feature space, and follow this with a cascade of tests for the clustering of matches. By aggregating information in an appearance rather than a transformation space, it is easier to differentiate a large number of symmetric parts. As a result, we obtain an algorithm that can detect symmetries in very large data sets efficiently both in terms of asymptotic scaling behavior as well as in terms of quality.

We evaluate the accuracy, robustness, and performance of our proposal on real-world 3D scans that suffer from noise, outliers, and incomplete acquisition. The processing can be done in a single, linear scan, allowing for external data storage on hard disk. Our method can detect the symmetries within a scene of 14GB of raw data size in less than 70 minutes on a dual-socket commodity PC. Nevertheless, we maintain a recognition accuracy comparable to that of the current state-of-the-art technique [BBW+09], which is probably the most scalable technique for this task but still is not able to compute a result on the full data set.

In order to study scalability further, we artificially create much larger scenes with up to 500GB of raw data. Even in this case, symmetry detection can be performed overnight on a single PC.

As part of the doctoral work we have also explored structures in volume data sets like CT or MRI scans [KBW+10, KWKS11]. In this case the domain consists of density values on a discrete 3D voxel grid. The focus has been placed on extracting and visualizing information about salient features, characteristics, and symmetries within the volume. Figure 5.2 shows detected symmetries for three selected models [KWKS11]. Since this is a very different scenario with more degrees of freedom, we do not describe these works in more detail here.

Moreover, we have also been involved in research on a related correspondence problem, known as *animation reconstruction* [TBW+12], which contributed to the doctorate of Art Tevs [Tev11].

When not stated otherwise, this second part of the thesis is based on the content presented in [KBWS12], and [KBWS13].

Chapter 6
Fundamentals

Symmetry is what you see at a glance ...

Blasie Pascal (1623 - 1662)

This quotation puts the problem in a nutshell. In an urban scene, the formation of e.g. windows, balconies and roofs, as well as the hierarchy or the patterns in which they occur, is obvious for a human observer [Pal77]. Our brain is used to grasping such information quickly, since we have been faced with such tasks in our daily life for years. For a computer, the provided input is a loose set of points without any semantical context. It is a computationally complex matching problem to establish correspondences between different features and shape patches and thus extract a basic understanding of the content.

Research on symmetry detection in digital 3D shape models [GCO06, MGP06, PSG+06, SKS06, Mar07, PMW+08, OSG08, BBW+09, LTSW09, LCDF10, Liu10, MBB10, RBBK10, STS10, BWM+11, XZJ+12, FS13, MPWC13] has recently gained much interest. The goals and applications as well as the inspected properties are diverse. Nevertheless, to the best of our knowledge, the scalability aspect has not been taken into account so far.

For the specific case of urban scenes, and facades in particular, there exists a great deal of literature on extraction, visualization, or learning regularities based on 2D images. Since our domain is proper 3D geometry, these methods are not in the scope of this thesis and therefore not further considered here.

Before discussing our approach in detail, we would first like to formalize the notion of symmetry detection so that it becomes clear what we are aiming at computing and why it is possible to speed this computation up. After that, we describe the data set which is used later on and briefly introduce important basic concepts.

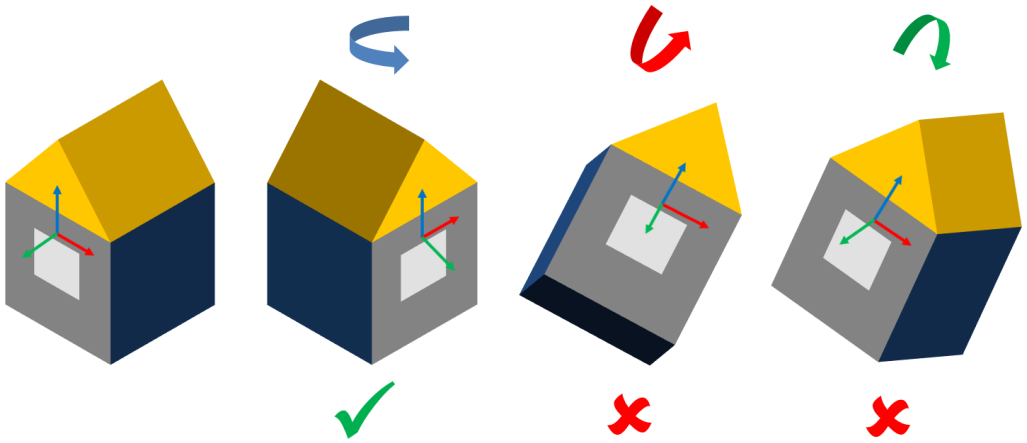


Figure 6.1: The original model (left) may only be rotated around the global upward direction (blue). A tilting by rotating around the tangential direction (red) and letting a facade be oriented uphill by rotating around the surface normal (green) is not meaningful for urban architecture. In the latter two cases the orientation of the former local upward direction has been affected.

6.1 Symmetry Detection

To be precise, we essentially investigate the detection of partial symmetries. This means that portions of an object match back to themselves under a constraint class of mappings. This is in contrast to global symmetries which have to hold true for the entire model.

A detection algorithm for partial symmetries takes a piece of geometry $\mathcal{S} \subset \mathbb{R}^3$ as input and considers a group of transformations $\mathcal{T}$ acting on it, which is a subgroup of the bijections of $\mathbb{R}^3$. Most frequently, these are rigid motions $\mathcal{T} = E(3)$ [BBW+09], or similarity transforms [MGP06]. Here, we consider a restricted class of true rigid motions $SE(3)$ that excludes reflections and keeps a global upward direction $\mathbf{u}$ of the scene invariant, i.e., $\mathbf{Tu} = \mathbf{u}$ for all $\mathbf{T} \in \mathcal{T}$. Hence, we only take translations and rotations around the up vector into account. This is only a minor restriction for architectural models, but simplifies the construction of our descriptors substantially. The restrictions arising from this assumption are visualized in Figure 6.1.

We now consider all subsets of $\mathcal{S}$ that are mapped back to $\mathcal{S}$ under transformations $\mathbf{T} \in \mathcal{T}$. The set of *pairwise correspondences* $\mathcal{C}$ with respect to input $\mathcal{S}$ and a transformation group $\mathcal{T}$ is defined as:

$$\mathcal{C}(\mathcal{S}, \mathcal{T}) = \{(\mathcal{P}, \mathbf{T}) | \mathcal{P} \subseteq \mathcal{S}, \mathbf{T} \in \mathcal{T}, \mathbf{T}(\mathcal{P}) \subseteq \mathcal{S}\}$$

A single correspondence consists of a subset $\mathcal{P}$ of the scene $\mathcal{S}$ and a transformation $\mathbf{T} \in \mathcal{T}$ that matches this piece to an alternative location. For a globally symmetric surface or a regular infinite model, like a crystallographic grid, the set of patches and the transformations form a mathematical group [MPWC13]. Nevertheless in our case with partial symmetries and finite shapes, this property does not hold because any repeated translation would harm the closeness criterion.

Obviously, there still is a large number of correspondences. For example, each subset of $\mathcal{P}$ itself is again a valid correspondence under $\mathbf{T}$. We can remove this ambiguity by considering only maximal sets $\mathcal{P}$. Another issue is that of continuous symmetries: a plane, for example, has an infinite set of symmetry transformations that map portions of it back to itself. To avoid computational costs, such continuous symmetries are usually computed a priori using a differential analysis [GG04] and can be excluded from further detection [BWS10]. Our algorithm follows this idea and excludes continuous symmetries by computing matches only at non-slippable features [BBW+09]. Finally, we also do not want to enumerate spurious matches that lead to very small subsets $\mathcal{P}$. A requirement should be that the detected parts $\mathcal{P}$ have a certain minimum extent. Our method implements this filter by demanding a match of local neighborhoods rather than just matching points.

Enforcing all of the mentioned restrictions still does not solve the complexity problem of symmetry detection. We still obtain a large number of overlapping subsets $\mathcal{P}$ that map back to $\mathcal{S}$ under various transformations $\mathbf{T}$. Explicitly computing all of these sets by a pairwise matching of geometry is possible for small scenes [BWS10] but prohibitively expensive for large inputs.

However, a computation of all pairwise matches is not required. Quadratic complexity is caused by overlapping matches. If we consider a single point on the model, the set of all points reached by a symmetry transformation forms a clique [LCDF10, KBW+12]. This is because correspondence, as we have defined it, is an equivalence relation. Consequently, it suffices to walk a linear-sized subgraph of the full clique to discover all symmetric points.

6.2 Test Data

The input to our algorithm is a raw point cloud from a 3D scanner, stored as a long list of 3D points, potentially amounting to several hundreds of gigabytes of data. As a benchmark data set, we have chosen the Hannover city scan collection [Bre07]. This data set features a diverse collection of many different buildings all over the city of Hannover, Germany, collected by 3D time-of-flight scanning and subsequent scan registration.

As a real-world data set, it contains significant noise artifacts, as well as clutter and outliers. Furthermore, many of the buildings are acquired only partially and the sampling density varies considerably. The full collection consists of 463 million points, or roughly 14GB in binary format. As we are not aware of publicly available data sets of even larger size, we use replications of parts or all of the data to study the scaling behavior for even larger scans.

6.2.1 Preprocessing

Our first task is to organize the data for efficient further processing. We use the data structure proposed by [WBB+08], for which an open source implementation is available[1].

The first step is to partition the data into blocks that fit into main memory. This is done by a hashing scheme. The scene is overlaid with a regular grid and each grid cell is associated to a file on hard disk. Then, in a single scan through the input data, the point set is split up into manageable parts. After that, an octree is built to organize the pieces hierarchically, with the leaf nodes storing the subsets of the input data. Each leaf contains at most 64K points, which is the blocking size for further processing. The employed system also builds a multi-resolution rendering datastructure that is not necessary for the processing, but is used for visualizing the results. We also use the provided processing system to estimate normals by local PCA fitting and orient them according to the scanner positions, giving us an unambiguous surface orientation everywhere.

6.3 Basic Concepts

Before going into the details of our method we would like to sketch some fundamental tools which are used later on.

Line features: Crease lines can capture the most important part of the shape structure at small representational costs. From only sparse crease lines, a surprisingly exact surface representation can be reconstructed [OBS04]. We employ slippage-based line features [GG04, BBW+09] as local primitives to achieve a more abstract description of the scene content. These features group adjacent points along edge-like structures which share a common tangential direction. Line features have proven to be both compact and expressive at the same time and thus serve a first step towards reducing the amount of information to a manageable size.

[1] http://www.mpi-inf.mpg.de/∼mwand/XGRT/index.html

Orientation histograms: This type of descriptor is widely used in image processing for a variety of purposes [FR95, Low03, DT05, MS05]. Among other things, it is invariant under small local shifts, which turns out to be a useful property for our purpose. For its computation, an image is divided into overlapping patches. For each such patch the distribution of gradients or tangents at local points of interest are described by binning their orientation. This leads to a remarkably, discriminative, and robust local description of the content. The idea has also been adapted to describe and match features on meshes [MFK+10, DK12].

Principal component analysis: We use this technique (henceforth PCA) for the dimensionality reduction of vectors. During the process, the set of vectors is treated like points in an n-dimensional space. After that, an n-dimensional ellipsoid is fitted to this set of points in a least-squares sense. One can easily determine the n principal axes of this ellipsoid. Those directions with the longest elongation are the most discriminative. We now regard only an m-dimensional orthogonal subspace, with $m < n$, spanned by the "best" components. By projecting all points to this subspace, we can shrink the dimensionality of the corresponding vectors without sacrificing significant information.

Image alignment: The technique presented in [AGP08] is designed to rapidly align a pair of images. This means that images of similar subjects but differing points of view are slightly shifted such that the content matches in an overlapping area. The method is conceptually simple yet very effective. For each image the content with horizontal orientation is projected to the vertical axis leading to a histogram with characteristic distribution. To align the two images vertically one has to slide both images up and down, relative to each other, until their histograms reach a maximal correlation. Horizontal alignment is achieved accordingly. For rotational alignment, the histograms projected to the diagonals are taken into account.

Normalized cross-correlation: This approach (henceforth NCC) is a way to measure the similarity of two images. Technically, it is an inner product of the normalized inputs. In other words, if the images are regarded as a long vector, the normalized cross-correlation computes the cosine of the angle between them. The output is a value between -1 and 1 which expresses the analogy.

Chapter 7

Large Scale Symmetry Detection

Today we live in a chaos of straight lines,
in a jungle of straight lines.

Friedensreich Hundertwasser (1928-2000)

Our method consists of three major steps as illustrated in Figure 7.1. First we need to achieve an abstraction of the content. Therefore, we reduce the given geometry to salient line features and select a sparse set of interest points to compactly characterize the underlying shape.

In order to describe the neighborhood of these key points, we capture information about the constellation of feature lines in their local neighborhood. For the descriptor we only consider the distribution of their orientation and later on reduce the dimensionality for the sake of efficiency. The main observation is that key points of symmetric shape patches are mapped to similar positions in the descriptor space and thus indicate a potential correspondence. This fact drastically reduces the search space for matching candidates.

In this descriptor space, we build a k-nearest-neighbor graph. By walking along this graph, we will with high likelihood encounter matching geometry. Using k nearest neighbors ensures that both approximate matches and overlaps in descriptor space are handled correctly. Since the proximity of descriptors does not guarantee a match, we need another verification step. This is done by pairwise comparisons between candidates. We align local feature line configurations and evaluate their similarity.

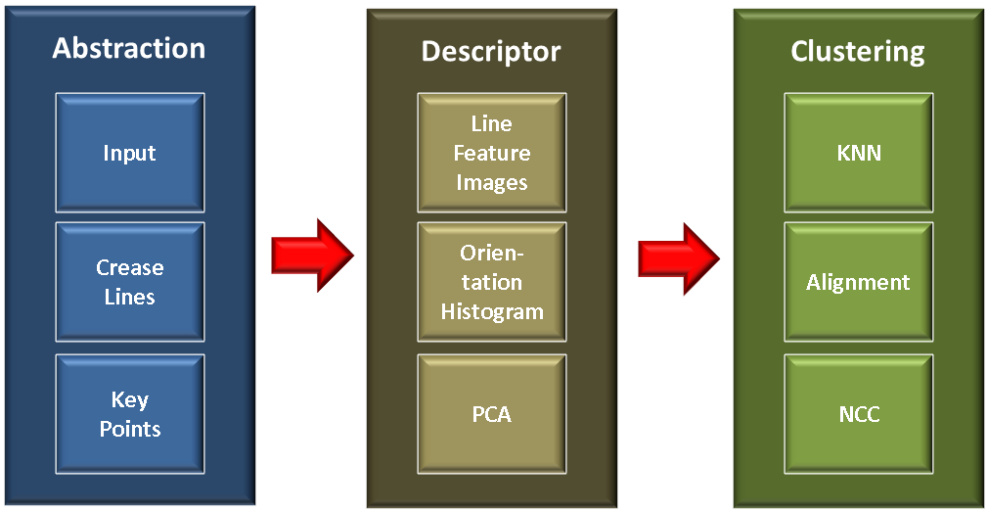


Figure 7.1: A flow chart of our scalable symmetry detection algorithm.

7.1 Line Features and Key Points

The first goal of our processing pipeline is to reduce the amount of data to a more manageable set while maintaining most of the important shape information. We follow the previous work of [BBW+09] and compute a line-feature representation that extracts the salient crease lines from the input geometry as an abstraction of the full point cloud. In addition to the line features, we extract a relatively sparse set of sample points from the input that will be equipped with descriptors later on, and for which we will actually compute symmetry information. We will refer to these as *key points*.

Line Features: In practice, slippage-based line features [BBW+09] yield a storage reduction of about a factor of 190 (from 14GB down to 72MB for the Hannover data set). Assuming that at least 8GB of main memory are available in most off-the-shelf PCs today, this implies that this representation can compress the original input data of more than a Terabyte so that it fits into main memory, where more flexible processing is possible.

Key Points: We restrict the scene to feature points $\mathcal{F} \subseteq \mathcal{S}$ that are non-slippable, which means that tey do not belong to a continuous symmetry. Therefore we use junctions of crossing line features (called "bases" in [BBW+09]) as samples. Since many junctions can occur in a small neighborhood, we remove points that are too close by Poisson disc sampling. As detailed below, our descriptors will cover a radius r. They are designed with robustness towards local shifts so that we can rather aggressively sparsen the sample set. We use a sample spacing of 20% of the descriptor radius r in practice.

Implementation: We implement a parallel out-of-core algorithm for computing the line features efficiently. The feature extraction method in [BBW+09] retrieves points with linear slippability [GG04], corresponding to a general notion of line features. After that, a projection operator is used that draws surface points onto the closest point of maximal curvature in direction orthogonal to the line direction. To compute the set of line features from the input, sample points are randomly distributed over the surface using Poisson disk sampling with spacing σ and each sample is projected individually onto a line segment, never moving by more than σ. Since every projection is independent from other samples, we can process many samples in parallel.

From the octree, we extract all child nodes, which we subsequently treat as processing blocks. For each child node, we retrieve the direct neighbors to provide context at the boundaries and cut out an extended region. The projection operator requires a local neighborhood of surface points (to be exact, a sphere with radius σ) in order to compute surface properties such as curvature or slippage analysis [GMGP05]. Furthermore, a projection can move a point up to σ away from its original position. This means for each process block we need to add all points within a 2σ boundary to the process block in order to compute correct projections. Further, in order to maintain a uniform sampling density of line features that is not affected by process block boundaries, we remove all line features outside their initial process block boundary.

We now use a single thread to load these blocks from disk and extract the neighborhoods. We traverse the tree coherently in a depth-first manner and use an LRU cache for blocks. The extraction thread then dispatches work packages to several processing threads that perform the line feature extraction for several blocks in parallel. To speed up the range queries of the projection algorithm, local spatial hierarchies are used. As the computational costs for the projection are much higher than for just loading and clipping the data, we usually obtain full CPU utilization on our reference machine. Only for small scenes consisting of just a few blocks, we observe a suboptimal CPU utilization because the load-balancing is quantized to whole blocks.

At the end of this first step, we are given an abstract and compact representation of the input geometry with a number of special points of interest at locations with characteristic properties.

7.2 Descriptor

From the construction of the key points, we can assume that matching pairs of symmetric instances will contain sample points that are nearby with respect to the symmetry transformation but vary in their exact position.

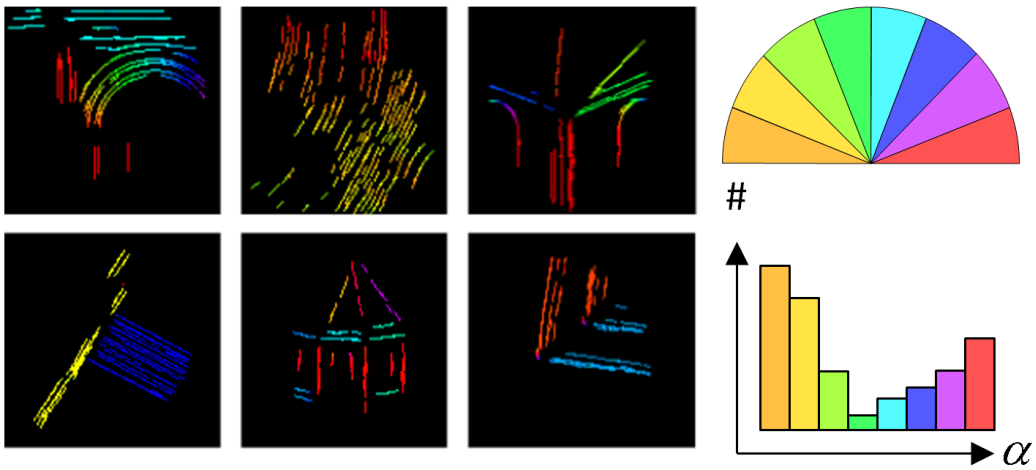


Figure 7.2: Examples of 6 different line feature images from the Hannover data set (left) and an illustration of how a corresponding orientation histogram is built (right).

For example, a window might receive a sample on the corner, while a symmetric window gets a sample more towards its center. Therefore, we have to ensure that the descriptor is invariant under translations up to a tolerance of at least the sample spacing $0.2r$.

Such invariance is a common problem in computer vision, where robustness towards small transformations as well as local distortions is crucial for effective image descriptors. A widely employed and well-performing standard solution to this problem is the use of orientation histograms [Low03, DT05, MS05]. which provides local shift invariance.

This idea can be transferred to the geometric domain by using crease lines on the geometry as the analogue of image gradients, both of which contain the most important information about the data. Our situation is different, as we are dealing with noisy and partial point cloud data, and are targeting large data sets and thus aiming at high throughput. We build our algorithm upon the previously computed sparsely sampled line feature representation. The distribution and constellations of line feature directions are examined in local neighborhoods to build orientation histograms. Figure 7.2 illustrates the concept. The color coding of the lines, in six images the left part, depicts their angle with the horizontal axis, ranging from $0°$ to $180°$. For each image, the histograms capture the distribution of line directions without considering spatial location.

7.2.1 Line Feature Images

We assume that we have an urban scene with a consistent upward orientation $\mathbf{u}$. Mappings between symmetric instances cannot change this upward direction. To build descriptors, we consider all line features in a sphere of radius r around each key point. For this set of features, we fix a local coordinate frame aligned with the upward orientation and the average surface normal that we approximate by performing principal component analysis on the line features. From our experience, it is sufficient to consider the projection of line features onto the plane orthogonal to the average surface normal, because architectural facades mostly resemble reliefs with relatively small depth. As the projection of matching feature points is consistent, our descriptor will still work for objects with arbitrary depth, but the overlay of features might be a bit less discriminative. The 2D projections of the line features form *line feature images*, from which we derive the descriptors. We use 128×128 pixel images, in which we render the computed line samples by drawing short segments of length σ.

7.2.2 Orientation Histograms

The orientation histograms encode the distribution of orientations discretized into a set of orientational bins. In this work, we use a relatively low resolution of typically $B = 8$ different undirected angles, which has the advantage of compactness and also acts as a low pass filter of orientations under the assumption of random noise in the line feature orientations. In order to increase the expressiveness of our descriptors, we compute multiple orientation histograms with overlapping but different spatial regions, on a $H \times H$ grid. In practice, we have used 4×4 histograms to achieve all the results shown within this thesis. Our evaluation in the upcoming section will justify this parameter setting. To avoid aliasing, we use a Gaussian window function with standard deviation r/H in the spatial domain and $180°/B$ degree in the orientation domain and continuously distribute contributions to overlapping bins correspondingly. As shown in [DT05], such a proper filtering is instrumental to good results.

7.2.3 Dimensionality Reduction

The number of dimensions increases with spatial and orientational binning. We therefore use PCA to project the descriptor vector of size $4 \times 4 \times 8 = 128$ to a lower dimensional basis with D dimensions. We will later show experimentally that $D = 8$ is a good choice where the dimensionality reduction has only minimal impact on recognition performance.

The reason for seeking a low-dimensional representation is the "curse of dimensionality" [Bel03]. The upcoming clustering requires retrieving similar descriptors by proximity queries in descriptor space. For high dimensions of this space, most traditional exact and even approximate range query techniques become inefficient. There are approaches like locality sensitive hashing that can achieve sub-linear query times in high dimensions [AI08], relying on the Johnsen-Lindenstrauss Lemma [DG03]. However, these techniques come at considerable costs in terms of runtime constants and super-linear memory requirements that again impede scalability [GPB05]. In summary, the more we are able to discriminate geometry by descriptors with few degrees of freedom, the easier and more efficient the search problem, and thereby the clustering, becomes.

At the end of this second step, we have computed a descriptor vector $descr(\mathbf{x}) \in \mathbb{R}^d$ for each key point $\mathbf{x} \in \mathcal{F}$. This descriptor maps the local neighborhood of radius r , $N_r(\mathbf{x}) := \{\mathbf{y} \in \mathcal{S} | dist(\mathbf{x}, \mathbf{y}) < r\}$, of points $\mathbf{x} \in \mathcal{S}$, to a short vector that summarizes the local geometry. This mapping is of course not injective, but will ensure that similar geometry maps to similar descriptor values while dissimilar geometry, with high likelihood, is assigned dissimilar descriptors.

7.3 Clustering

We now use the computed descriptors to cluster matching geometry, thereby computing the symmetry transformations for each feature point. Two points $\mathbf{x}, \mathbf{y} \in \mathcal{F}$ are considered symmetric if there exists a $\mathbf{T} \in \mathcal{T}$ such that $\mathbf{T}(N_r(\mathbf{x})) \approx N_r(\mathbf{y})$. By construction of the descriptors, we know that this implies $descr(\mathbf{x}) \approx descr(\mathbf{y})$, but not the other way round. Similar geometry will have similar descriptors, but multiple different pieces of geometry can still map to the same descriptor. Therefore, we need an additional verification step for effective clustering. The key idea is to consider nearby descriptors as potential matching candidates and verify the quality of the match by an explicit geometric alignment.

7.3.1 Rapid Geometric Alignment

At this point, we need a fast algorithm to align two feature line images. Obviously, we could use the iterative closest lines (ICL) algorithm originally employed in [BBW+09]. However, the absolute costs are high. We therefore propose an efficient approximation that is at least two orders of magnitude faster. Practical experiments have shown that the run time of rapid alignment decreased to 4 minutes from 480 minutes for standard ICL. The new technique also has the advantage of a larger convergence radius, as it performs a global search in contrast to the local search of ICL.

We employ the technique from [AGP08] for rapid image alignment. In our scenario, a rotational alignment is given a priori by the coordinate frame defined by the upward direction and the mean normal, which is robust as it has been averaged over a large area. In contrast to the original approach, we perform the correlation in the Fourier domain using FFT, for further speed-up.

With the estimated alignment, we now compare the line feature images by comparing shifted images, which we compute on the fly to save memory. The comparison is done applying NCC, and after blurring with a small Gaussian kernel that again avoids aliasing and sensitivity to alignment errors that are expected to be in the range of one or two pixels.

7.3.2 Geometric Clustering

Our clustering algorithm is a simple region growing technique. We consider a graph with feature points as nodes and edges connecting the k-nearest-neighbors in descriptor space, using Euclidean distances. The edges connect different candidates of matching geometry, which we now verify during the clustering. We traverse the graph breadth first. For each new node encountered, we compute an alignment of the feature line image of the new node to that of the original start node using the rapid alignment technique. We continue the search towards the unprocessed neighbors of this node if and only if the images match, which means that the NCC score is above a user-chosen threshold ϵ.

We always compare to the start node to avoid drift that could be caused by long chains of pairwise matches that overall lead to a large difference in geometry. We want to find an approximate equivalence class where all elements have similar geometry. Only comparing neighbors could lead to an accumulation of errors, permitting large dissimilarities within one and the same cluster. Using the first cluster member as reference avoids this problem, at the expense of making the result dependent on the first random choice of this center piece. This problem will be addressed in a post-processing step.

It is also important to stress that we utilize two comparison measures at this point. On the one hand, similarity of descriptors serves as the initial filter to obtain candidate matches. On the other hand, the actual match of geometry is used as the criterion to steer and stop the traversal of the descriptor graph.

Filtering by descriptors reduces the number of pairwise comparisons from naive $O(n^2)$ to $O(kn)$ for n key points. If the descriptor is effective, k is a small number. We will show empirically that small values for k, between 10 and 50, are sufficient to obtain good results on complex scenes with a large amount of geometric variation. As long as k is large enough, we will be able to retrieve the whole fully-connected clique by performing only a linear number of comparisons.

The only performance-critical part that remains is the computation of nearest

neighbors in descriptor space. The design of our descriptors allows us to use rather low-dimensional feature vectors with only 8 dimensions. In addition, the reduction to sparse samples permits storing key points in-core even for very large scenes. We therefore currently employ the approximate nearest-neighbor library [AMN+98], which performs well for spatial queries with medium dimensionality.

So far, we have found a clique of matching points for each key point. This information is only linear in size and encodes all correspondences $\mathcal{C}(\mathcal{S}, \mathcal{T})$ of the whole scene implicitly. Unlike traditional approaches [MGP06, BBW+09], we do not segment the scene into symmetric pieces, but output the matching for each point [BWS10]. This representation provides more information than an ad-hoc partitioning [MPWC13] as it encodes all overlapping symmetries.

7.3.3 Dynamic Area Queries

The clustering algorithm described above concludes the preprocessing of symmetry information. The output is a partition of the set of all feature points into fully connected clusters of matching geometry.

We now discuss how this information can be utilized to derive more precise symmetry information. We assume that the user, or an outer-loop algorithm, wants to retrieve all pieces of geometry within the whole scene $\mathcal{S}$ that are symmetric to a given query piece $\mathcal{P} \subset \mathcal{S}$. In this situation, we have stronger cues because we can combine the information of *all* feature points $\mathcal{F}_\mathcal{P}$ within $\mathcal{P}$ *simultaneously*.

We first enumerate all clusters to which the feature points in $\mathcal{F}_\mathcal{P}$ belong. From this, we retrieve a set of transformations to symmetric instances. Since the clusters can still contain outliers, we align all line features within the query region to the potential symmetries using iterative closest line (ICL) [BBW+09]. Now that the number of comparisons has drastically decreased, this approach is feasible. We use the resulting ICL score to validate a symmetry, and display the symmetry if at least 50 percent of all line features find a correspondence.

We recursively repeat this process with the transformations we can find in the symmetric regions we match. In other words, we build the transitive closure via region growing and verification by ICL to the original instance $\mathcal{P}$. As shown below, this transitive-closure step significantly improves the results.

In our experiments, we let the user choose rectangular bounding boxes to mark example regions $\mathcal{P}$. In the corresponding images these are shown as wire-frame boxes. Please note that the region queries are dynamic, not precomputed. Therefore, the costs are not included in the processing costs given. The queries always search the whole scene, not only the visible portion. Nevertheless, response times are interactive: in the range of a few seconds for the full Hannover test scene.

7.4 Parameter Evaluation

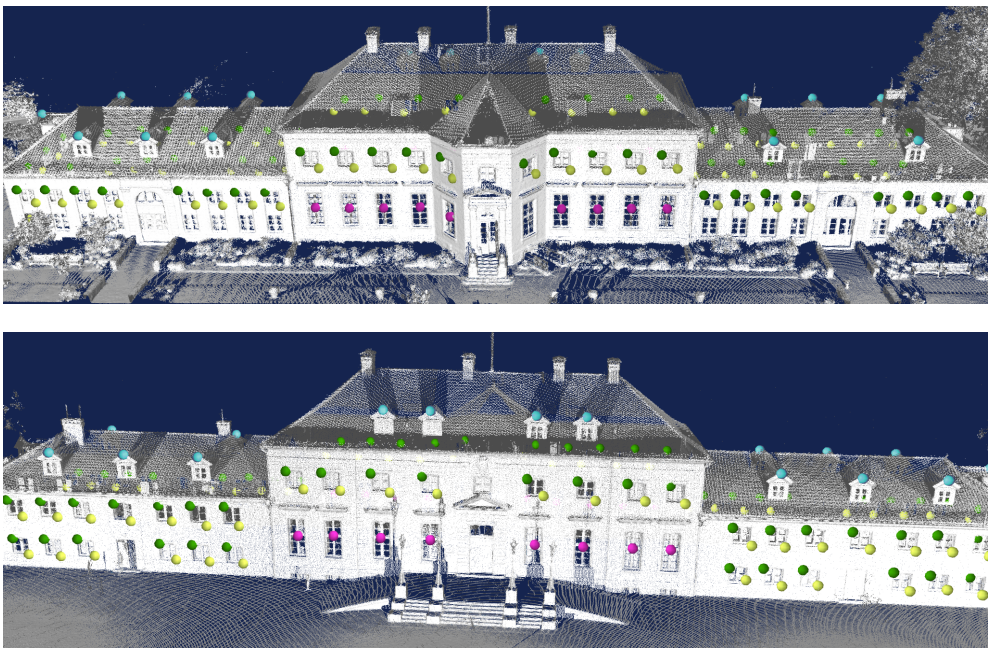


Figure 7.3: Manually annotated benchmark data set. Four different types of features are labeled at the front and back of the museum. All spheres with the same color indicate symmetric geometry.

We now evaluate our method by studying the influence of algorithmic and complexity parameters.

Our method has a number of parameters that might significantly affect the performance in practice. Therefore, we systematically study the influence of these parameters. We proceed in two steps. First, we look at only the recognition performance of the descriptor and choose good parameter values. Second, we study the full pipeline and its behavior under different configurations.

We chose the Wilhelm Busch Museum, which is part of the Hannover scans, as a benchmark and manually annotated the data. We label points on the object's surface according to salient partial symmetries. As shown in Figure 7.3, we have picked four different types of features at different windows that show up repeatedly within the building and marked corresponding points, indicated by different colors. We restrict the annotation to these four unambiguous types of symmetric building blocks and ignore other matches for this test.

7.4.1 Descriptor Test

For this test, we retrieve the key point closest to each annotated point and match it against all other samples using their descriptors. A good descriptor should match these descriptors to other descriptors close to an annotation point of the same class, and only to them. We define closeness by a sphere of radius $0.5r$, which is the maximum distance at which our alignment scheme could possibly find a match of the descriptor images. According to this definition, we compute false negative rates (averaged over all features and descriptors), and the absolute number of false positives, normalized by dividing by the number of tests (i.e., overall number of annotated points).

The results are shown in Figure 7.4. The curves correspond to different threshold values in matching the descriptors. The x-axis is the false positive rate as a percentage of all descriptors, and the y-axis is the true positive rate, averaged over all annotations. We first test different parameters for the spatial and the orientational binning Figure 7.4 (top).

It turns out that 4×4 spatial histograms yield a good performance, with only marginal improvements for larger values. For orientation histograms, dividing the angular range of $180°$ into 8 bins leads to the best results. 16 bins as well as 4 bins (not shown to avoid clutter) are worse. We have also tried in-between values and settled for 8 bins as the best value.

Next, we examine the effect of dimensionality reduction in Figure 7.4 (bottom). The reduction from 16 to 8 dimensions via PCA impedes the recognition rate only marginally, while providing substantial benefits for an efficient nearest-neighbor search. To demonstrate the necessity for normalization, we also include a comparison of normalized vs. unnormalized descriptors (dashed lines) in this diagram. Normalization means that we scale the histogram entries by the inverse of the l_2 norm of the full histogram. This significantly improves the recognition performance and holds true for all other parameter combinations as well, which we omit to avoid clutter. This is why we only use normalized histograms in all our experiments.

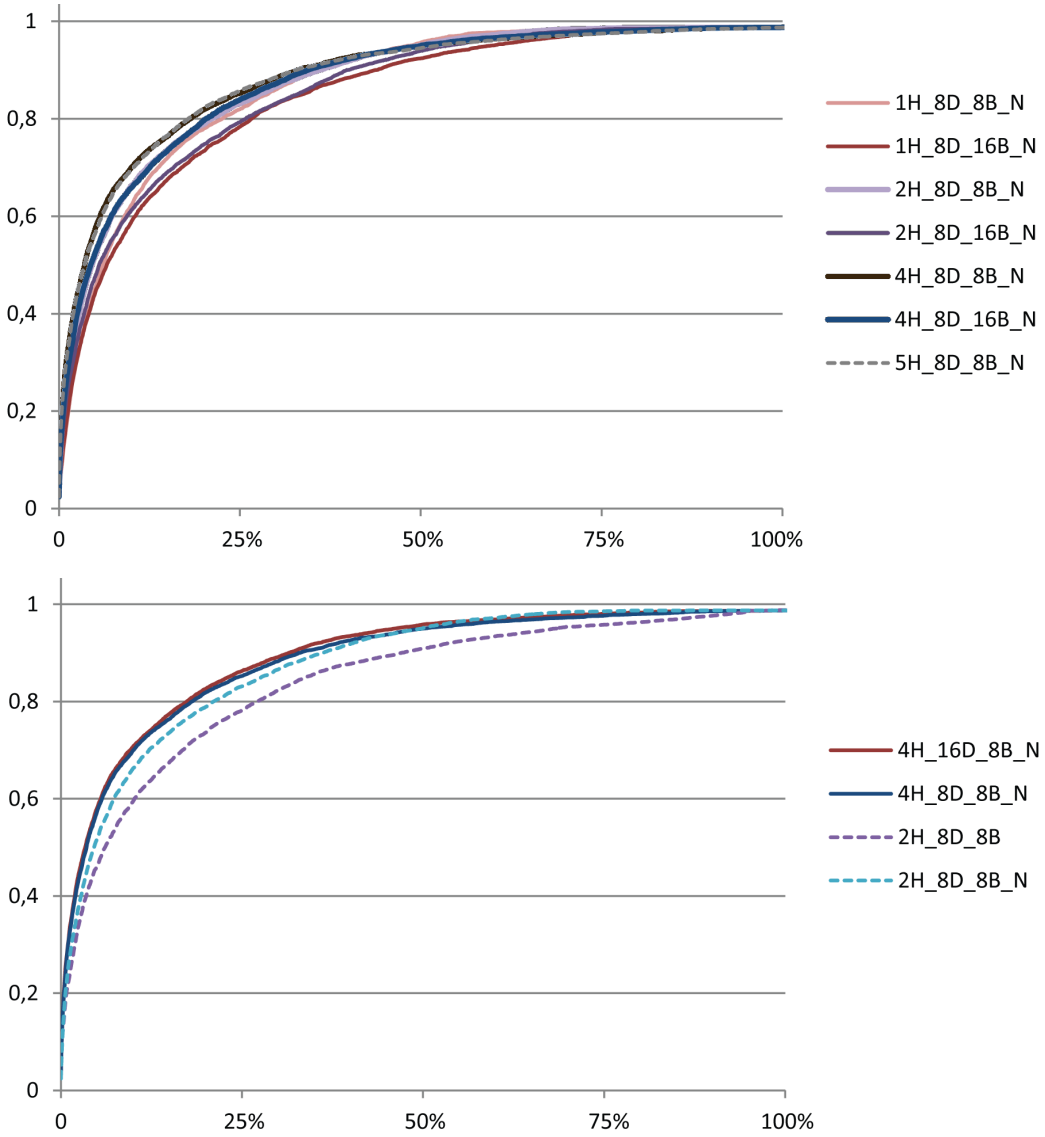


Figure 7.4: Precision recall curves for the descriptor test. In the top graph the number of histogram patches and orientation bins is varied, whereas the bottom graph demonstrates the effect of dimensionality reduction and normalization. The annotations represent the following: H = number of spatial histograms per dimension, D = dimensionality after reduction by PCA, B = number of orientation bins. The suffix N denotes a normalization of the histograms prior to comparison.

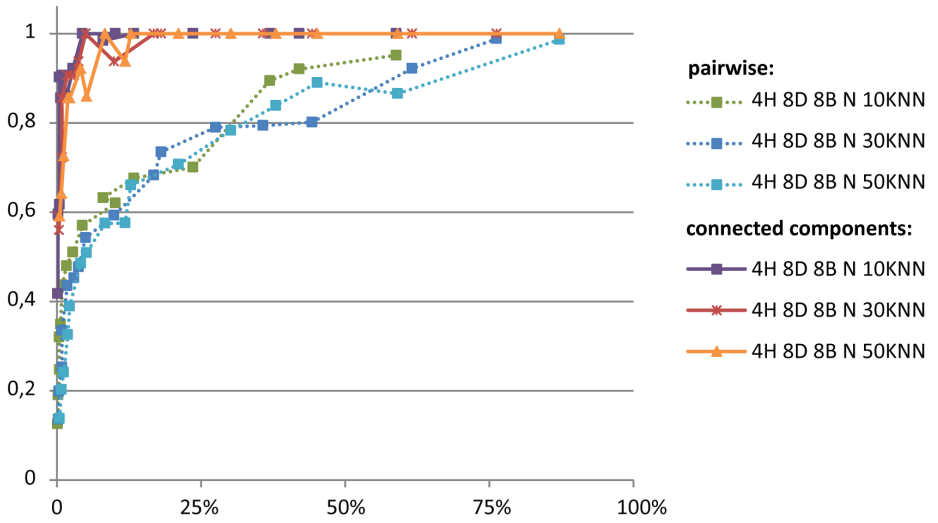


Figure 7.5: Precision recall curves for the full pipeline. The lower three curves show the pairwise recognition rate when considering 10, 30, and 50 nearest neighbors. The upper three make the more realistic assumption of extracting the connected components of the transitive closure. To avoid clutter, we only consider the case of 4 by 4 histograms (H), with 8 orientation bins (B), reduced to 8 dimensions (D) and normalization (N).

7.4.2 Full Pipeline Benchmark

Here, we test the full pipeline, including the rapid geometric alignment. We again use the annotated data for the Museum scene and apply the same criteria as in the previous test, but this time with a tolerance radius for a match of $2r$. For counting true and false positives, we use two separate criteria. First, we just evaluate the accuracy of the pairwise matches, as before. In addition, we also examine the transitive closure of the matches, which is done using the dynamic query algorithm.

Figure 7.5 shows the results for the corresponding precision recall tests of the full pipeline, varying the threshold value ϵ of the NCC-based image comparison. The different curves correspond to varying the number of nearest neighbors (indicated by the annotation **KNN**) in the the clustering step. Within the accuracy of this test, the parameter does not seem to affect the results, and 10 nearest neighbors appear to be sufficient. However, the test scene is still rather small. In very large scenes, an increase might still provide an advantage. Experimenting with the full Hannover data set, we found that 30 nearest neighbors are usually sufficient. A full assessment would require large-scale annotations on large quantities of data, which we have to leave for future work.

We have also varied the other descriptor parameters (not shown in the figure) and confirmed the results from the previous test, namely that 4×4 spatial histograms with 8 bins and a reduction to 8 dimensions leads to the best performance with respect to both accuracy and efficiency.

In terms of absolute recognition performance, the results are quite encouraging. When taking the transitive closure of the matches into account, the recognition rate rapidly approaches 100% at small false-positive rates. For these further tests, we read off an NCC threshold value of 0.6 that gives almost 100% accuracy at minimal false positive rates, and use it for all further experiments.

7.5 Results

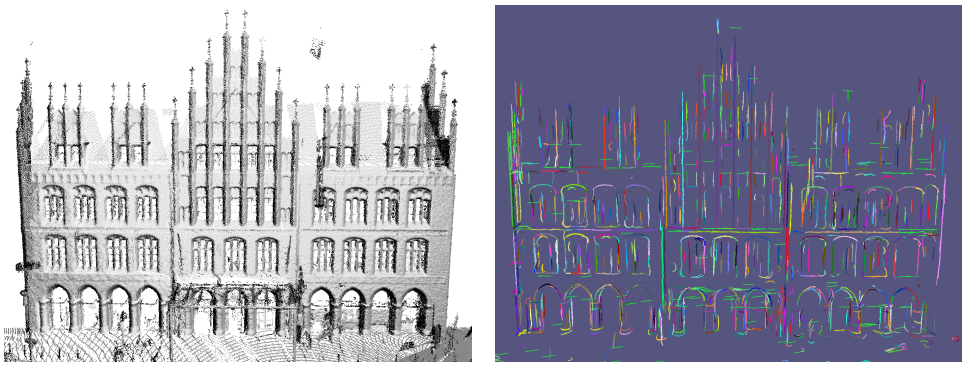
Here, we present some practical outcomes of our method applied to the Hannover scene and analyze the overall performance in terms of accuracy and scalability.

7.5.1 Detection

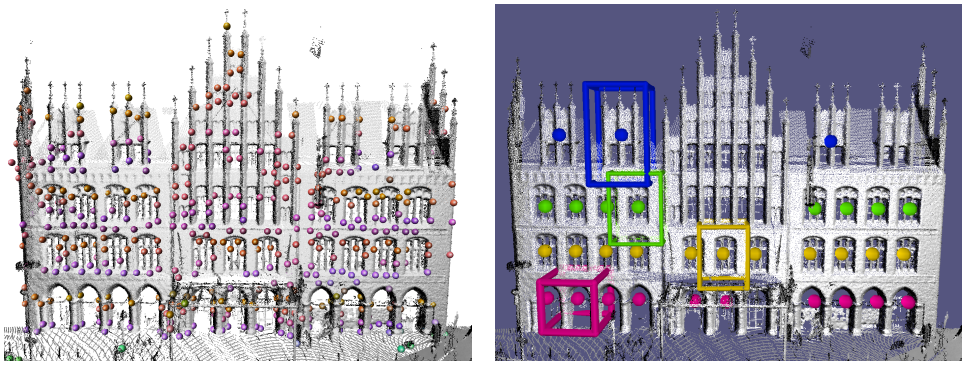
For clarity, we only show a selection of dynamic queries, such that not all symmetry information is displayed simultaneously in the results. We indicate the selected symmetries by a colored sphere where each user query has an individual random color. The user-marked bounding box is accordingly shown in the same color.

For purposes of illustration, Figure 7.6 depicts the intermediate results of the detection pipeline for the "old town hall" scan, demonstrating how the line features capture important geometric aspects, robust to partial and noisy data, and how the descriptors at sparse key points serve as a prefilter for the clustering and geometric validation. Finally, a dynamic query extracts area-based symmetries. Please note that the color of the lines is randomly chosen and does not correspond to their orientation here. The descriptors are shown in a three-dimensional projection color coded in RGB. They are actually 8-dimensional and thus more discriminative than they appear in this projection.

In Figure 7.7 we show a set of example results, all obtained from a single symmetry detection pass on the Hannover scan collection. Please note that this test uses a single parameter set for the entire data set. The search for both clustering and dynamic queries is performed globally and simultaneously on the whole city.



(a) (b)



(c) (d)

Figure 7.6: Visualization of the symmetry detection pipeline stages. Input data (a), line feature abstraction (b), color coded descriptors at key points (c), visualized symmetric instances of selected parts (d).

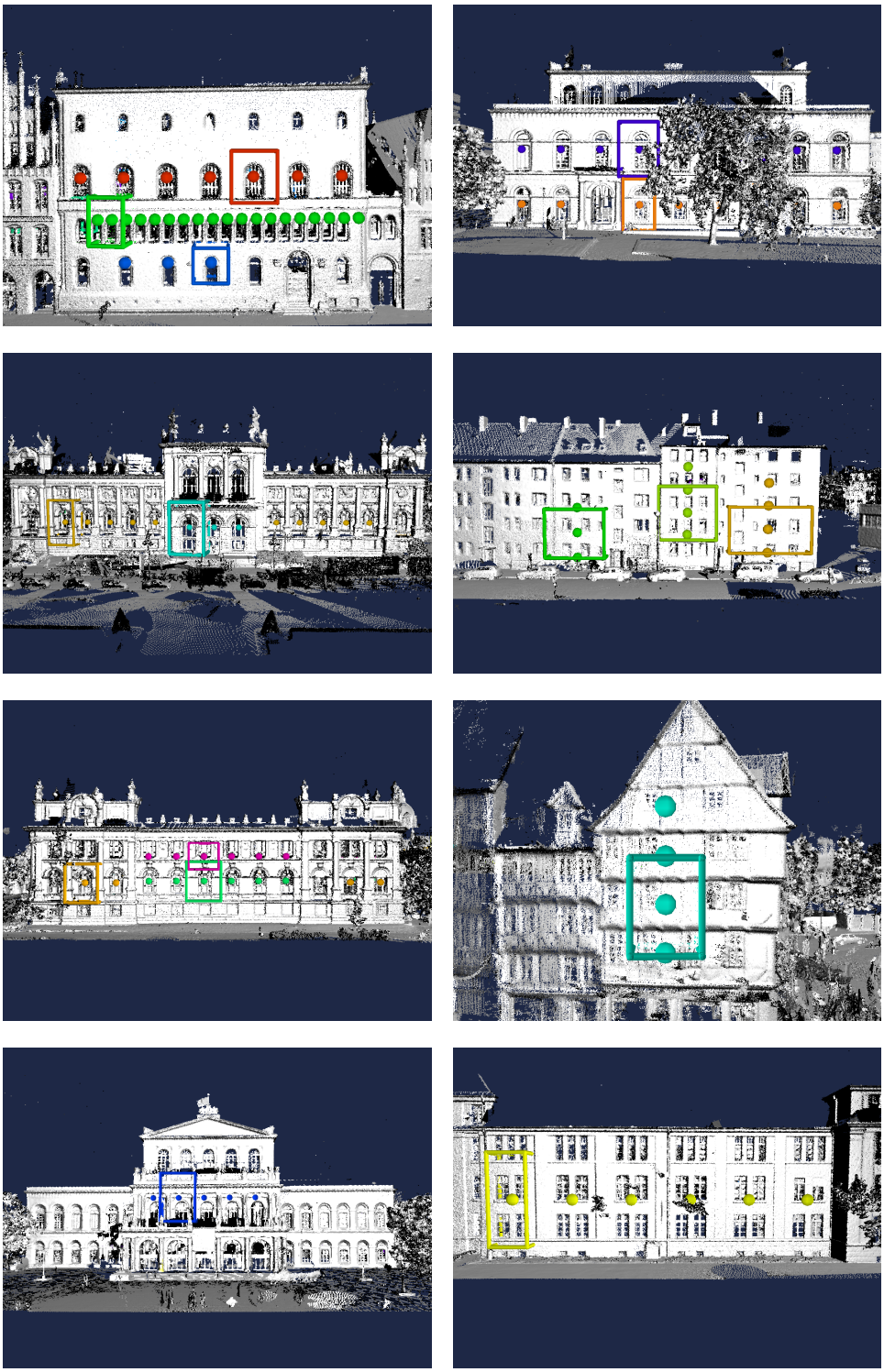


Figure 7.7: A selection of symmetries found across the city of Hannover.

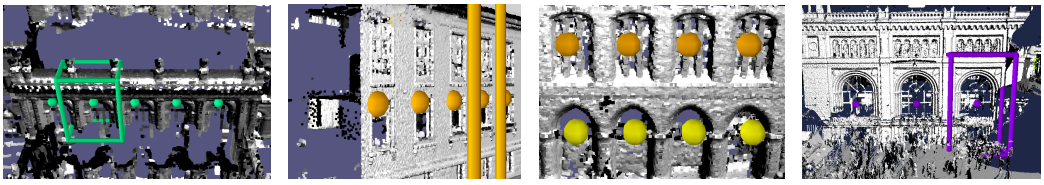
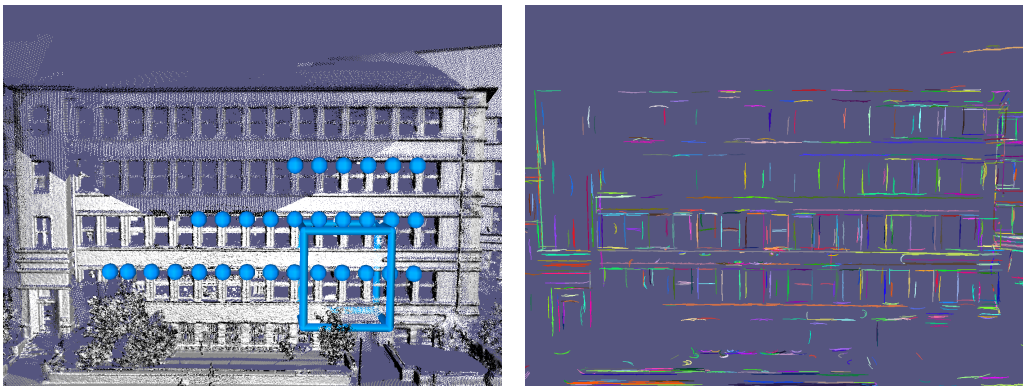


Figure 7.8: A collection of challenging spots handled by our method.



(a) (b)

Figure 7.9: In case of highly deviating sampling, a reliable line feature computation is not possible and thus only works in areas of similar acquisition density.

Our results are robust even in the presence of substantial noise, clutter, and missing data, as highlighted in Figure 7.8. This is due to the fact that we rely on partial matches of crease line patterns. However, if these features cannot be found, a reliable matching is no longer possible (see Figure 7.9).

7.5.2 Scalability

Our experiments are performed on a dual socket Intel Xeon X5650 2.66 GHZ (6 physical cores + hyper threading) equipped with 48 GB of main memory and an NVIDIA GeForce GTX 480. The code for line feature extraction uses multi-threading with 24 threads. The remainder of the code is currently single threaded, as feature extraction dominates the runtime.

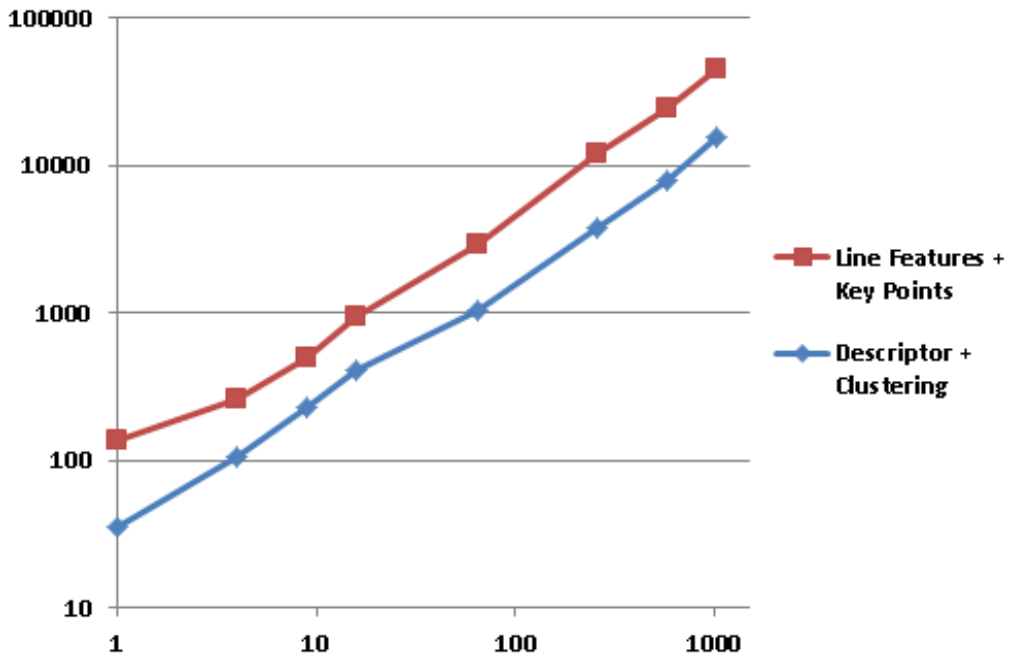


Figure 7.10: Log-log plot showing the scaling behavior of our pipeline with increasing scene size.

Figure 7.10 depicts the scaling behavior of our method in a log-log plot. For this series, we replicate the museum scene on a regular grid to obtain up to 500GB of input data. The x-axis depicts the number of instances (15.3M points each, corresponding to 488MB). The y-axis shows the elapsed time in seconds. The timings for both steps scale almost linearly with the scene size. When going from the smallest to the next bigger scene, the increase is even sublinear because of uneven load balancing in the parallel line feature computation that is negatively affected by small workloads. The absolute running times are also quite satisfactory. Computing symmetries on a 490GB input scene took less than 17 hours overall, and the single Hannover scan collection required just 66 minutes. About two thirds of the time is spent on computing line features and feature points, whereas one third is spent in the actual clustering-based symmetry detection. Table 7.1 summarizes the statistics and timings of all of the test scenes. For one instance of the museum we extracted more than 20k line features and almost 1k key points. For one instance of the entire city, these values increased to 500k line features and almost 24k key points. In all cases, the computation of a dynamic query required a few seconds each.

Model	#Instances	Size	#Points	Line Features + Key Points	Descriptors + Clustering
Busch-	1	0.5GB	15.3M	2m 18s	35s
Museum	4	2GB	61.3M	4m 20s	1m 45s
	9	4GB	138.0M	8m 22s	3m 46s
	16	7GB	245.4M	15m 43s	6m 43s
	64	30GB	981.6M	48m 52s	17m 32s
	256	122GB	3.9B	3h 23m 27s	1h 2m 19s
	576	276GB	8.8B	6h 49m 18s	2h 12m 17s
	1024	490GB	15.7B	12h 29m 2s	4h 15m 47s
Hannover-	1	14GB	463.4M	43m 57s	23m 12s
Sights	9	128GB	4.2B	5h 19m 4s	3h 50m 14s

Table 7.1: Runtimes of the two algorithm stages for multiple instances of two different models.

7.6 Discussion

In this section we contrast our method with the current state-of-the-art approach and address open problems.

7.6.1 Comparison

For this purpose, we use the method of [BBW+09] as it can be regarded as the current state of the art. A quantitative comparison is difficult because the authors compute symmetric parts by greedy region growing. Depending on parameter settings, this will yield quite different results in terms of detected symmetries. In contrast, our method precomputes the richer set of all point-wise symmetry cliques for all feature points. However, we then use these for a dynamic query with specified area, a piece of information that the previous technique cannot rely on.

Nevertheless, it is worth attempting a qualitative comparison as shown in Figure 7.11. For the old town hall scene (top), we can observe the differences from greedy region growing. The previous method detects some of the windows but omits other details in favor of a global reflective symmetry of the whole facade, which our approach does not recognize by design. However, our method collects all symmetry information for the details and reports it when queried, with only one top piece missing. Unlike the previous method, we can also differentiate the different window variants correctly.

Figure 7.11: Symmetry detection results of [BBW+09] (left) and our outcomes (right) for the old town hall (top) and the new town hall (bottom).

For the new town hall (bottom), different parameter settings for the region growing in [BBW+09] lead to a more favorable, fine-grained decomposition of the facade. Our technique cannot capture very small details individually, such as the small windows in the top row, but otherwise leads to comparable results.

Please recall that our method processed the *entire* Hannover data set at once with one set of parameters. In [BBW+09] parameters were set to each data set individually. Furthermore, our new method is in general less parameter-dependent. Only the sensitivity and scale of the line-feature detection need to be adapted to the input once, further parameters are kept constant.

7.6.2 General Remarks

Recognition results: The recognition results of our method for architectural scenes are comparable with the previous state of the art in symmetry detection restricted to small data sets. However, our method is restricted in that it does not directly output dense correspondences, but only matches at feature points that sample the model less densely.

Dense correspondences still have to be reconstructed from this information, as is done for example in the presented dynamic query algorithm. Precomputing such information, as was done in previous work, is not practical for large scenes. Finding efficient algorithms and data structures for computing and storing such overlapping dense correspondence information is still an open problem. Nevertheless, we would argue that the main problem is finding symmetric instances in the first place, which is what this thesis addresses, rather than estimating the coverage of symmetric area.

Scalability: Our method is currently designed for handling large but not extremely large scenes. A key limitation is that the line features and key points need to fit into main memory. Observing an empirical compression factor of about 170, this would in principle allow us to handle scenes up 1-10TB using an affordable 5-50 GB of main memory. However, the computation of the line features then becomes a bottleneck. Our system requires roughly 6.5h per 100GB for the computation. In addition, data partitioning takes twice the amount of time, but this could be optimized by removing the multi-resolution rendering capability, which currently dominates the preprocessing times [WBB+08].

Another concern is the effectiveness of the descriptor in large scenes with a large variety of geometry. While we were not able to examine this on Terabyte-sized scenes due to the lack of publicly available data, our experience with the Hannover test set was rather positive. Despite significant geometric variety and severe noise, missing data and clutter, recognition results and clustering costs were satisfactory, with results comparable to the state of the art in small scenes.

Noise and undersampling: As demonstrated by the results on real-world city scans, our method is quite robust to noisy data. Two aspects are important in this context: First, the primary source of information is the set of line features, which is computed by a moving-least-squares procedure that averages over local neighborhoods of radius σ. This suppresses high-frequency noise and variations in sampling density up to the scale of the averaging radius σ, which has to be chosen appropriately. Secondly, further robustness to large scale artifacts, like outliers and missing data, is obtained by matching line features.

As demonstrated by [BBW+09], pairwise alignment of feature lines is impressively insensitive to missing data. Their paper even shows results of ICL alignments of objects with no overlap, which is impossible with classical criteria of minimizing geometric distance, such as point-to-point ICP. Matching of crease lines and their orientation gives strong cues for geometric alignment that are less likely to be coincidental than the mere proximity of surface points. The descriptor that our method uses in addition to direct alignment is based on statistics of crease line orientations, which inherits this robustness.

Limitations: The discussion in the previous paragraph also shows a major remaining limitation of our current method: Similar to previous work [BBW+09], we still extract crease lines at a single level of detail, meaning that σ is fixed for the entire data set. Therefore, our method cannot locally adapt to varying sampling density and varying noise levels. A multi-resolution implementation would probably make the method more robust, and remove the need for the user to specify the parameter σ, which is currently the most important parameter that has to be chosen manually. Being based on line features, our method also requires geometry that has a sufficient amount of such features. Shapes which are extremely smooth at all scales, like spheres or cylinders, are problematic. However, for architectural scenes, which we are targeting, this is in practice hardly ever the case.

Further, our method is limited by design to detect symmetries that share a fixed upward orientation, and does not consider reflections. In the context of city scanning this is acceptable, since common architectural building blocks meet this requirement. However, this might be an issue for other data sets, such as large scale data sets with content of a more general nature. The main challenge for obtaining full rotational invariance is reformulating the descriptor for full coordinate frame invariance.

Chapter 8
Conclusion

Science cannot solve the ultimate mystery of nature. And that is because, in the last analysis, we ourselves are a part of the mystery that we are trying to solve.

Max Planck (1858 - 1947)

We have presented a symmetry detection method specifically designed for large-scale point clouds of 3D city scans. The key is to design a low-dimensional feature space in which nearby points have symmetric geometry with high likelihood, and then to perform clustering in that space, which involves only local comparisons, such that the method is able to scale to large data sets. We have introduced a new descriptor that is optimized for low dimensionality and fast computation.

Furthermore, we have designed a very fast approximate geometry matching scheme that is optimized for architectural objects and is two orders of magnitude faster than previous line-feature alignment, and thereby four orders of magnitude faster than plain pairwise ICP of the original data points. A user can interactively select a surface patch locally and gets the global correspondences as a result within seconds.

In combination, we obtain a symmetry detection system that can handle very large point sets on a single PC. We have demonstrated symmetry detection, with computations taking less than a day, in scenes up to 500GB of raw input data, which is three orders of magnitude larger than the largest scene for which a symmetry detection result had been reported in any previous work.

8.1 Future Prospects

In the future, we would like to examine scalability to even larger data sets, beyond several TB in size. This would require a distributed implementation on a cluster. A CUDA implementation could help in further improving the absolute throughput. Existing distributed algorithms for out-of-core clustering of image data [LRR07, FGG+10] could serve as an inspiration for a generalization in this direction.

Finding a more compact encoding and an intuitive visualization of actual matching geometry is still an interesting challenge for the entire research field. Here, we have only investigated symmetric patches of maximal extent. Following the idea of [KBW+12], we could apply our method recursively on the previously detected symmetries in order to decompose the model into canonical building blocks and extract knowledge about hierarchies.

Although our descriptor is very compact yet expressive enough for the task of processing an urban scene, there is still room for further improvement. It would be great if the proximity in feature space not only indicates similar geometry but directly implies it, such that a second verification step for the clustering would no longer be necessary. This would further speed up the overall computation times.

It is obvious that the flood of available information will further increase in the future. As a consequence, there will be a high demand for similar solutions also in other fields. By adapting the descriptor appropriately, our pipeline can easily be transferred to very different domains for which massive data has been collected over the past decades. Possible and interesting examples would be the analysis of reoccurring patterns in climate and weather conditions, or similarities in music data, but also correlations or peculiar behavior of stock-market prices. Our advances in detecting symmetries in volume data [KWKS11] could e.g. directly be applied to a video in order to investigate the periodicity of motions in time.

Many experts in computer vision and cognitive sciences expect and hope that from the understanding of how to teach a computer to analyze the structure in a given data, one can draw conclusions on how the human brain comprehends its environment. Despite the achievements and effort in the entire area, we are yet aware that there is still a long way to such an ambitious goal. In this regard, we would like to leave the reader with the Max-Planck-quotation at the beginning of this chapter.

Bibliography

[AGDL09] Andrew Adams, Natasha Gelfand, Jennifer Dolson, and Marc Levoy. Gaussian KD-trees for fast high-dimensional filtering. *ACM Transactions on Graphics*, 28(3):21:1–21:12, July 2009. 50

[AGP08] Andrew Adams, Natasha Gelfand, and Kari Pulli. Viewfinder alignment. *Computer Graphics Forum*, 27(2):597–606, 2008. 71, 79

[AI08] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117–122, 2008. 78

[Ale12] Marc Alexa. Synthetic images on real surfaces. In *Computational Design Modelling*, pages 79–88. Springer Berlin Heidelberg, 2012. 22

[AM10] Marc Alexa and Wojciech Matusik. Reliefs as images. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 1–7. ACM, 2010. 21

[AM11] Marc Alexa and Wojciech Matusik. Images from self-occlusion. In *Proceedings of the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*, pages 17–24. ACM, 2011. 22

[AMK07] Neil Alldrin, Satya Mallick, and David Kriegman. Resolving the generalized bas-relief ambiguity by entropy minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2007. 20

[AMN+98] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 45:891–923, 1998. 80

[BBW+08] Alexander Berner, Martin Bokeloh, Michael Wand, Andreas Schilling, and Hans-Peter Seidel. A graph-based approach to symmetry detection. In *Symposium on Volume and Point-Based Graphics*, pages 1–8, Los Angeles, CA, 2008. Eurographics Association. 63

[BBW+09] Martin Bokeloh, Alexander Berner, Michael Wand, Hans-Peter Seidel, and Andreas Schilling. Symmetry detection using line features. *Computer Graphics Forum*, 28(2):697–706, 2009. 61, 63, 64, 67, 68, 69, 70, 74, 75, 78, 80, 90, 91, 92, 93

[Bel03] Richard Ernest Bellman. *Dynamic Programming*. Dover Publications, Incorporated, 2003. 78

[BH11] Zhe Bian and Shi-Min Hu. Preserving detailed features in digital bas-relief making. *Computer Aided Geometric Design*, 28:245–256, May 2011. 17, 43, 46

[BKY99] Peter N. Belhumeur, David J. Kriegman, and Alan L. Yuille. The bas-relief ambiguity. *International Journal of Computer Vision*, 35(1):33–44, 1999. 9

[Bra91] Bert Larue Bradford. *Fast Fourier transforms for direct solution of poisson's equation*. PhD thesis, University of Colorado at Boulder, Boulder, CO, USA, 1991. UMI Order No. GAX91-28875. 50

[Bra01] D. Braess. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, 2001. 32

[Bre07] Claus Brenner. Hannover city scan database. http://www.ikg.uni-hannover.de/index.php?id=413, 2007. Last accessed on 2012-09-01. 69

[BWKS11] Martin Bokeloh, Michael Wand, Vladlen Koltun, and Hans-Peter Seidel. Pattern-aware shape deformation using sliding dockers. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2011)*, 30(6):123:1–123:10, 2011. 61

[BWM+11] Alexander Berner, Michael Wand, Niloy Mitra, Daniel Mewes, and Hans-Peter Seidel. Shape analysis with subspace symmetries. *Computer Graphics Forum (Proceedings EUROGRAPHICS)*, 30(2), April 2011. 67

[BWS10] Martin Bokeloh, Michael Wand, and Hans-Peter Seidel. A connection between partial symmetry and inverse procedural modeling. *ACM Transactions on Graphics*, 29:104:1–104:10, July 2010. 61, 69, 80

[CA09] Matthew T. Cook and Arvin Agah. A survey of sketch-based 3-D modeling techniques. *Interacting with Computers*, 21:201–211, July 2009. 22

[CCL+11] Yin Chen, Zhi-Quan Cheng, Jun Li, Ralph R. Martin, and Yan-Zhen Wang. Relief extraction and editing. *Computer-Aided Design*, 43:1674–1682, December 2011. 20

[CKK05] Manmohan Krishna Chandraker, Fredrik Kahl, and David J. Kriegman. Reflections on the generalized bas-relief ambiguity. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '05, pages 788–795, 2005. 20

[CMS97] Paolo Cignoni, Claudio Montani, and Roberto Scopigno. Computer-assisted generation of bas- and high-reliefs. *Journal of Graphics Tools*, 2(3):15–28, 1997. 15, 16

[Coq90] Sabine Coquillart. Extended free-form deformation: A sculpturing tool for 3D geometric modeling. In *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '90, pages 187–196. ACM, 1990. 24

[DD02] Frédo Durand and Julie Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *SIGGRAPH '02: ACM SIGGRAPH 2002 Papers*, pages 257–266. ACM, 2002. 13

[DFRS03] Doug DeCarlo, Adam Finkelstein, Szymon Rusinkiewicz, and Anthony Santella. Suggestive contours for conveying shape. *ACM Transactions on Graphics*, 22:848–855, July 2003. 10

[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. 78

[DK12] T. Darom and Y. Keller. Scale-invariant features for 3-D mesh models. *IEEE Transactions on Image Processing*, 21(5):2758–2769, 2012. 71

[DR06] Paul Debevec and Erik Reinhard. High-dynamic-range imaging: Theory and applications. SIGGRAPH 2006 Course #5, 2006. 10

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *International Conference on Computer Vision & Pattern Recognition*, volume 2, pages 886–893, Washington, DC, USA, June 2005. IEEE Computer Society. 71, 76, 77

[FDCO03] Shachar Fleishman, Iddo Drori, and Daniel Cohen-Or. Bilateral mesh denoising. *ACM Transactions on Graphics*, 22(3):950–953, July 2003. 13

[FGG+10] Jan-Michael Frahm, Pierre Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, and Marc Pollefeys. Building Rome on a cloudless day. In *European Conference on Computer Vision*, Berlin, Heidelberg, 2010. Springer-Verlag. 96

[Fla29] John Flaxman. *Lectures on sculpture*. London: J. Murray, 1829. 3

[FLW02] Raanan Fattal, Dani Lischinski, and Michael Werman. Gradient domain high dynamic range compression. In *ACM Transactions on Graphics (Proceedings SIGGRAPH 2002)*, pages 249–256, 2002. 30, 42

[FPRJ00] Sarah F. Frisken, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 249–254. ACM Press/Addison-Wesley Publishing Co., 2000. 24

[FR95] W. T. Freeman and M. Roth. Orientation histograms for hand gesture recognition. In *International Workshop on Automatic Face and Gesture- Recognition*, pages 296–301. IEEE Computer Society, 1995. 71

[FS13] Sam Friedman and Ioannis Stamos. Online detection of repeated structures in point clouds of urban scenes for compression and registration. *International Journal of Computer Vision*, 102(1-3):112–128, 2013. 67

[GCO06] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics*, 25(1):130–150, 2006. 67

[GG04] Natasha Gelfand and Leonidas Guibas. Shape segmentation using local slippage analysis. In *Proceedings Symposium on Geometry Processing*, New York, NY, USA, 2004. ACM. 69, 70, 75

[GMGP05] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann. Robust global registration. In *Proceedings Symposium on Geometry Processing*, pages 197–206, Aire-la-Ville, Switzerland, 2005. Eurographics Association. 75

[GO11] Eduardo S. L. Gastal and Manuel M. Oliveira. Domain transform for edge-aware image and video processing. *ACM Transactions on Graphics*, 30(4):69:1–69:12, 2011. 58

[GPB05] Jonathan Goldstein, John C Platt, and Christopher J C Burges. Redundant bit vectors for quickly searching high-dimensional regions. *Deterministic and Statistical Methods in Machine Learning*, 3635:137–158, 2005. 78

[GSH+07] Ran Gal, Ariel Shamir, Tal Hassner, Mark Pauly, and Daniel Cohen-Or. Surface reconstruction using local shape priors. In *Proceedings Symposium on Geometry Processing*, Aire-la-Ville, Switzerland, 2007. Eurographics Association. 61

[GSMCO09] Ran Gal, Olga Sorkine, Niloy Mitra, and Daniel Cohen-Or. iWIRES: An analyze-and-edit approach to shape manipulation. *ACM Transactions on Graphics*, 28(3):33:1–33:10, 2009. 61

[HB89] B. K.P. Horn and M.J. Brooks. *Shape from shading*, volume 2. MIT Press, Cambridge, Massachusetts, 1989. 20

[Her99] Lesley Herbert. *Bas Relief & Applique*. Merehurst, 1999. 5

[Hor70] B. K.P. Horn. Shape from shading: A method for obtaining the shape of a smooth opaque object from one view. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1970. 20

[HX10] He Huizhen and Yang Xunnian. Relief generation on surfaces. *Journal of Computer-Aided Design & Computer Graphics*, 22(7):1132–1137, July 2010. 19

[JDA07] Tilke Judd, Frédo Durand, and Edward H. Adelson. Apparent ridges for line drawing. *ACM Transactions on Graphics*, 26(3):19, 2007. 10

[KBS07] Jens Kerber, Alexander Belyaev, and Hans-Peter Seidel. Feature preserving depth compression of range images. In *Proceedings of the 23rd Spring Conference on Computer Graphics*, pages 110–114. Comenius University, Bratislava, April 2007. 15, 16, 41

[KBW+10] Jens Kerber, Martin Bokeloh, Michael Wand, Jens Krüger, and Hans-Peter Seidel. Feature preserving sketching of volume data. In *15th International Workshop on Vision, Modeling and Visualization*, pages 195–202. Eurographics Association, November 2010. 65

[KBW+12] Javor Kalojanov, Martin Bokeloh, Michael Wand, Leonidas Guibas, Hans-Peter Seidel, and Philipp Slusallek. Microtiles: Extracting building blocks from correspondences. *Computer Graphics Forum*, 31(5):1597–1606, August 2012. 69, 96

[KBWS12] Jens Kerber, Martin Bokeloh, Michael Wand, and Hans-Peter Seidel. Symmetry detection in large scale city scans. Research Report MPI-I-2012-4-001, Max-Planck-Institut für Informatik, April 2012. 65

[KBWS13] Jens Kerber, Martin Bokeloh, Michael Wand, and Hans-Peter Seidel. Scalable symmetry detection for urban scenes. *Computer Graphics Forum*, 32(1):3–15, February 2013. 65

[KCVS98] Leif Kobbelt, Swen Campagna, Jens Vorsatz, and Hans-Peter Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 105–114. ACM, 1998. 17

[Ker07] Jens Kerber. Digital art of bas-relief sculpting. Master's thesis, Universität des Saarlandes, August 2007. 15, 16, 21, 41, 42, 46

[KLST11] M. Kolomenkin, G. Leifman, I. Shimshoni, and A. Tal. Reconstruction of relief objects from line drawings. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 993–1000, June 2011. 23

[Koe84] J.J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13:321–330, 1984. 10

[KTB+09] Jens Kerber, Art Tevs, Alexander Belyaev, Rhaleb Zayer, and Hans-Peter Seidel. Feature sensitive bas relief generation. In *International Conference on Shape Modeling and Applications*, pages 148–154. IEEE Computer Society Press, June 2009. 7, 8, 33, 44

[KTB+10] Jens Kerber, Art Tevs, Alexander Belyaev, Rhaleb Zayer, and Hans-Peter Seidel. Real-time generation of digital bas-reliefs. *Computer-Aided Design and Applications (Special Issue: CAD in the Arts)*, 7(4):465–478, May 2010. 7, 8, 18, 19

[KWC+12] Jens Kerber, Meili Wang, Jian Chang, Jian J. Zhang, Alexander Belyaev, and Hans-Peter Seidel. Computer assisted relief generation - A survey. *Computer Graphics Forum*, 31(8):2363–2377, December 2012. 8

[KWKS11] Jens Kerber, Michael Wand, Jens Krüger, and Hans-Peter Seidel. Partial symmetry detection in volume data. In *16th International Workshop on Vision, Modeling and Visualization*, pages 41–48. Eurographics Association, October 2011. 65, 96

[LCD06] Thomas Luft, Carsten Colditz, and Oliver Deussen. Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics*, 25(3):1206–1213, July 2006. 12

[LCDF10] Yaron Lipman, Xiaobai Chen, Ingrid Daubechies, and Thomas Funkhouser. Symmetry factored embedding and distance. *ACM Transactions on Graphics (SIGGRAPH 2010)*, pages 103:1–103:12, July 2010. 67, 69

[LCOZ+11] Jinjie Lin, Daniel Cohen-Or, Hao (Richard) Zhang, Cheng Liang, Andrei Sharf, Oliver Deussen, and Baoquan Chen. Structure-preserving retargeting of irregular 3D architecture. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2011)*, 30(6):183:1–183:10, December 2011. 61

[Liu10] Yanxi Liu. Computational symmetry: Past, present, and future. ECCV 2010 Tutorial, 2010. 67

[LLL12] Bo Li, Shenglan Liu, and Zhang Liyan. Bas-relief generation using manifold harmonics analysis. *Journal of Computer-Aided Design & Computer Graphics*, 24(2):252–261, February 2012. 19

[LLZ12] Bo Li, Shenglan Liu, and Liyan Zhang. Detail-preserving bas-relief on surface from 3D scene. *Journal of Computer-Aided Design & Computer Graphics*, 24(6), June 2012. 19

[LLZX11] Bo Li, Shenglan Liu, Liyan Zhang, and Xu Xiaoyan. Bas-relief generation algorithm based on Laplace operator decomposition of 3D model. *Computer Integrated Manufacturing Systems*, 17(5):946–951, May 2011. 19

[LMLR06] Shenglan Liu, Ralph R. Martin, Frank C. Langbein, and Paul L. Rosin. Segmenting reliefs on triangle meshes. In *Proceedings of the 2006 ACM Symposium on Solid and Physical Modeling*, SPM '06, pages 7–16. ACM, 2006. 19

[LMLR07a] Shenglan Liu, Ralph R. Martin, Frank C. Langbein, and Paul L. Rosin. Background surface estimation for reverse engineering of reliefs. *International Journal of CAD/CAM*, 7(4), 2007. 19

[LMLR07b] Shenglan Liu, Ralph R. Martin, Frank C. Langbein, and Paul L. Rosin. Segmenting geometric reliefs from textured background surfaces. *Computer-Aided Design and Applications*, 4(5):565–583, 2007. 19

[Low03] D. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, volume 20, pages 91–110, 2003. 71, 76

[LRR07] Ting Liu, Charles Rosenberg, and Henry A. Rowley. Clustering billions of images with large scale nearest neighbor search. In *Proceedings of the Eighth IEEE Workshop on Applications of Computer Vision*, WACV '07, Washington, DC, USA, 2007. IEEE Computer Society. 96

[LTLZ11] Shenglan Liu, Zhengxiang Tang, Bo Li, and Liyan Zhang. Relief pasting algorithm based on normal vector adjustment. *Journal of Computer Applications*, 31(1):33–36, January 2011. 6

[LTSW09] R. Lasowski, A. Tevs, H.-P. Seidel, and M. Wand. A probabilistic framework for partial intrinsic symmetries in geometric data. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 963–970, 2009. 67

[LVJ05] Chang Ha Lee, Amitabh Varshney, and David W. Jacobs. Mesh saliency. In *ACM Transactions on Graphics (Proceedings SIGGRAPH 2005)*, pages 659–666, 2005. 15

[LWYM12] Zhuwen Li, Song Wang, Jinhui Yu, and Kwan-Liu Ma. Restoration of brick and stone relief from single rubbing images. *IEEE Transactions on Visualization and Computer Graphics*, 18:177–187, 2012. 22

[Mar07] Aurélien Martinet. *Structuring 3D Geometry based on Symmetry and Instancing Information*. PhD thesis, INP Grenoble, March 2007. 67

[MBB10] Niloy J. Mitra, Alex Bronstein, and Michael Bronstein. Intrinsic regularity detection in 3D geometry. In *Proceedings of ECCV'10*, pages 398–410. Springer-Verlag, 2010. 67

[MFK+10] C. Maes, T. Fabry, J. Keustermans, D. Smeets, P. Suetens, and D. Vandermeulen. Feature detection on 3D face surfaces for pose normalisation and recognition. In *Biometrics: Theory Applications and Systems (BTAS)*, 2010. 71

[MGP06] Niloy J. Mitra, Leonidas J. Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3D geometry. *ACM Transactions on Graphics*, 25(3):560–568, 2006. 61, 63, 67, 68, 80

[Mis13a] Miscellaneous. Website, 2013.
(a) http://en.wikipedia.org/wiki/File:Buonarotti-scala.jpg;
(b) http://en.wikipedia.org/wiki/File:Luxor_temple_15.jpg;
(c) http://en.wikipedia.org/wiki/File:GraniteElephant.jpg;
(d) http://en.wikipedia.org/wiki/File:Diptych_Barberini_Louvre_OA3850.JPG;
(e) http://en.wikipedia.org/wiki/File:Planck2mark.jpg;
(f) http://en.wikipedia.org/wiki/File:St_GaudensShaw_Mem.jpg;
(g) http://en.wikipedia.org/wiki/File:Portland_Vase_BM_Gem4036_n4.jpg;
(h) http://en.wikipedia.org/wiki/File:Cameo_gonzaga.jpg;

(i) http://en.wikipedia.org/wiki/File:Persepolis_stairs_of_the_Apadana_relief.jpg; last visited on January 24th 2013. 4, 5

[Mis13b] Miscellaneous. Website, 2013.
(a) http://en.wikipedia.org/wiki/File:Monarch_In_May.jpg;
(b) http://en.wikipedia.org/wiki/File:Apis_florea_nest_closeup2.jpg;
(c) http://en.wikipedia.org/wiki/File:Leaning_Tower_of_Pisa_(April_2012).jpg;
(d) http://en.wikipedia.org/wiki/File:Bentley_Snowflake11.jpg;
(e) http://en.wikipedia.org/wiki/File:Bromptoncross.jpg;
(f) http://en.wikipedia.org/wiki/File:Fortbourtange.jpg;
(g) http://en.wikipedia.org/wiki/File:Roof_hafez_tomb.jpg;
last visited on April 13th 2013. 61, 62

[MOT98] Shinji Mizuno, Minoru Okada, and Jun-ichiro Toriwaki. Virtual sculpting and virtual woodcut printing. *The Visual Computer*, 14:39–51, 1998. 24

[MP08] Niloy J. Mitra and Mark Pauly. Symmetry for architectural design. In *Advances in Architectural Geometry*, pages 13–16, 2008. 61

[MPWC13] Niloy J. Mitra, Mark Pauly, Michael Wand, and Duygu Ceylan. Symmetry in 3d geometry: Extraction and applications. *Computer Graphics Forum*, 2013. 61, 67, 69, 80

[MS05] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005. 71, 76

[MSHS06] Aurélien Martinet, Cyril Soler, Nicolas Holzschuch, and François Sillion. Accurate detection of symmetries in 3D shapes. *ACM Trans. on Graphics*, 25(2):439 – 464, 2006. 63

[MWA+12] Przemyslaw Musialski, Peter Wonka, Daniel G Aliaga, Michael Wimmer, Luc van Gool, and Werner Purgathofer. A survey of urban

reconstruction. In *EUROGRAPHICS 2012 State of the Art Reports*, pages 1–28. Eurographics Association, 2012. 63

[MYY+10] Niloy J. Mitra, Yong-Liang Yang, Dong-Ming Yan, Wilmot Li, and Maneesh Agrawala. Illustrating how mechanical assemblies work. *ACM Transactions on Graphics*, 29(3):58:1–58:12, 2010. 61

[OBS04] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Transactions on Graphics*, 23:609–612, August 2004. 10, 70

[OCDD01] Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. Image-based modeling and photo editing. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 433–442. ACM, 2001. 22

[OSG08] Maks Ovsjanikov, Jian Sun, and Leonidas Guibas. Global intrinsic symmetries of shapes. In *Eurographics Symposium on Geometry Processing (SGP)*, pages 1341–1348, Aire-la-Ville, Switzerland, 2008. Eurographics Association. 67

[OSSJ09] Luke Olsen, Faramarz F. Samavati, Mario Costa Sousa, and Joaquim A. Jorge. Sketch-based modeling: A survey. *Computers & Graphics*, 33(1):85 – 103, 2009. 22

[PAA+87] Stephen M. Pizer, E. Philip Amburn, John D. Austin, Robert Cromartie, Ari Geselowitz, Trey Greer, Bart Ter Haar Romeny, and John B. Zimmerman. Adaptive histogram equalization and its variations. *Computer Vision, Graphics, and Image Processing*, 39:355–368, September 1987. 16

[Pal77] Stephen E. Palmer. Hierarchical structure in perceptual representation. *Cognitive Psychology*, 9(4):441 – 474, 1977. 67

[PD06] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. In *Proceedings of the European Conference on Computer Vision*, pages 568–580, 2006. 13, 50

[PF01] Ronald N. Perry and Sarah F. Frisken. Kizamu: A system for sculpting digital characters. In *ACM Transactions on Graphics (Proceedings SIGGRAPH 2001)*, pages 47–56, 2001. 24

[PHK11] Sylvain Paris, Samuel W. Hasinoff, and Jan Kautz. Local Laplacian filters: Edge-aware image processing with a Laplacian pyramid. *ACM Transactions on Graphics*, 30:68:1–68:12, August 2011. 58

[PKTD08] Sylvain Paris, Pierre Kornprobst, Jack Tumblin, and Frédo Durand. A gentle introduction to bilateral filtering and its applications. ACM SIGGRAPH class, 2008. http://people.csail.mit.edu/sparis/bf_course/. 13

[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):43:1–43:11, 2008. 61, 63, 67

[POC05] Fábio Policarpo, Manuel M. Oliveira, and João L. D. Comba. Real-time relief mapping on arbitrary polygonal surfaces. *ACM Transactions on Graphics*, 24(3):935–942, July 2005. 6

[PSG+06] Joshua Podolak, Philip Shilane, Aleksey Golovinskiy, Szymon Rusinkiewicz, and Thomas Funkhouser. A planar-reflective symmetry transform for 3D shapes. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2006)*, 25(3):549–559, July 2006. 67

[PSS01] Alexander A. Pasko, Vladimir Savchenko, and Alexei Sourin. Synthetic carving using implicit surface primitives. *Computer-Aided Design, Elsevier*, 33(5):379–388, 2001. 23, 24

[PZ10] Zhao Peng and Bian Zhe. The bas-relief on curved surface from 3D meshes. *Journal of Computer-Aided Design & Computer Graphics*, 22(7):1126–1131, July 2010. 19

[RBBK10] Dan Raviv, Alexander M. Bronstein, Michael M. Bronstein, and Ron Kimmel. Full and partial symmetries of non-rigid shapes. *International Journal of Computer Vision*, 89(1):18–39, August 2010. 67

[RDF05] Szymon Rusinkiewicz, Doug DeCarlo, and Adam Finkelstein. Line drawings from 3D models. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05. ACM, 2005. 17

[RL06] Augusto Román and Hendrik P. A. Lensch. Automatic multiperspective images. In *Rendering Techniques 2006: Eurographics Symposium on Rendering*, pages 161–171. Eurographics Association, June 2006. 58

[RSI+08] Tobias Ritschel, Kaleigh Smith, Matthias Ihrke, Thorsten Grosch, Karol Myszkowski, and Hans-Peter Seidel. 3D unsharp masking for scene coherent enhancement. *ACM Transactions on Graphics*, 27(3), 2008. 12

[SBS07] Wenhao Song, Alexander Belyaev, and Hans-Peter Seidel. Automatic generation of bas-reliefs from 3D shapes. In *SMI '07: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2007*, pages 211–214. IEEE Computer Society, 2007. 15, 16, 46

[SCHP12] Chih-Tsung Shen, Feng-Ju Chang, Yi-Ping Hung, and Soo-Chang Pei. Edge-preserving image decomposition using l1 fidelity with l0 gradient. In *SIGGRAPH Asia 2012 Technical Briefs*, pages 6:1–6:4. ACM, 2012. 58

[SCOL+04] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, pages 175–184. ACM, 2004. 17

[SKS06] Patricio Simari, Evangelos Kalogerakis, and Karan Singh. Folding meshes: Hierarchical mesh segmentation based on planar symmetry. In *SGP '06: Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, pages 111–119, Aire-la-Ville, Switzerland, 2006. Eurographics Association. 63, 67

[SLF+11] Adrian Secord, Jingwan Lu, Adam Finkelstein, Manish Singh, and Andrew Nealen. Perceptual models of viewpoint preference. *ACM Transactions on Graphics*, 30(5), October 2011. 58

[Sou01a] Alexei Sourin. Functionally based virtual computer art. In *SI3D '01: Proceedings of the 2001 Symposium on Interactive 3D Graphics*, pages 77–84, 2001. 24

[Sou01b] Alexei Sourin. Functionally based virtual embossing. *The Visual Computer*, 17(4):258–271, 2001. 24

[SP03] Mario Costa Sousa and Przemyslaw Prusinkiewicz. A few good lines: Suggestive drawing of 3D models. *Computer Graphics Forum (Proceedings of EUROGRAPHICS*, 22:381–390, 2003. 10

[SRML09] Xianfang Sun, Paul L. Rosin, Ralph R. Martin, and Frank C. Langbein. Bas-relief generation using adaptive histogram equalisation.

IEEE Transactions on Visualization and Computer Graphics, pages 642–653, 2009. 16, 22, 43, 46

[SS88] Ulrich Schumann and Roland A Sweet. Fast fourier transforms for direct solution of poisson's equation with staggered boundary conditions. *Journal of Computational Physics*, 75(1):123–137, 1988. 50

[Ste98] C. Steger. An unbiased detector of curvilinear structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2):113 –125, February 1998. 23

[STS10] Jiwon Shin, R. Triebel, and R. Siegwart. Unsupervised discovery of repetitive objects. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 5041–5046, 2010. 67

[TBW+12] Art Tevs, Alexander Berner, Michael Wand, Ivo Ihrke, Martin Bokeloh, Jens Kerber, and Hans-Peter Seidel. Animation cartography - Intrinsic reconstruction of shape and motion. *ACM Transaction on Graphics, TOG*, 31(2):12:1–2:15, April 2012. 65

[Tev11] Art Tevs. *Deformable Shape Matching*. PhD thesis, Universität des Saarlandes, 2011. 65

[TM98] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the 1998 International Conference on Computer Vision*, ICCV '98, pages 839–846. IEEE Computer Society, 1998. 12

[TMH10] Natsuki Takayama, Shubing Meng, and Takahashi Hiroki. Choshi design system from 2D images. In *Proceedings of the 9th International Conference on Entertainment Computing*, ICEC'10, pages 358–365. Springer-Verlag, 2010. 22

[TMQ+07] Ping Tan, Satya Mallick, Long Quan, David Kriegman, and Todd Zickler. Isotropy, reciprocity and the generalized bas-relief ambiguity. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2007. 20

[Wan11] Meili Wang. *3D Digital Relief Generation*. PhD thesis, Bournemouth University, July 2011. 8

[WBB+08] Michael Wand, Alexander Berner, Martin Bokeloh, Philipp Jenke, Arno Fleck, Mark Hoffmann, Benjamin Maier, Dirk Staneker, Andreas Schilling, and Hans-Peter Seidel. Special section: Point-based graphics: Processing and interactive editing of huge point clouds from 3D scanners. *Computers & Graphics*, 32:204–220, April 2008. 70, 92

[WCKZ12] Meili Wang, Jian Chang, Jens Kerber, and Jian Zhang. A framework for digital sunken relief generation based on 3D geometric models. *The Visual Computer*, 28:1127–1137, 2012. 8, 17, 42

[WCPZ10] Meili Wang, Jian Chang, Junjun Pan, and Jian J. Zhang. Image-based bas-relief generation with gradient operation. In *The Eleventh IASTED International Conference on Computer Graphics and Imaging*, February 2010. 21, 42

[WCZ10] Meili Wang, Jian Chang, and J.J. Zhang. A review of digital relief generation techniques. In *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on*, volume 4, pages 198–202, April 2010. 23

[WDB+07] Tim Weyrich, Jia Deng, Connelly Barnes, Szymon Rusinkiewicz, and Adam Finkelstein. Digital bas-relief from 3D scenes. *ACM Transactions on Graphics*, 26(3):32, 2007. 16, 41, 43, 44, 45, 46, 57

[WK95] Sidney W. Wang and Arie E. Kaufman. Volume sculpting. In *Proceedings of the 1995 Symposium on Interactive 3D Graphics*, I3D '95, pages 151–ff. ACM, 1995. 24

[WKCZ11] Meili Wang, Jens Kerber, Jian Chang, and Jian J Zhang. Relief stylization from 3D models using featured lines. In *Proceedings of the 27th Spring Conference on Computer Graphics*, pages 63–68. Comenius University, Bratislava, April 2011. 8, 17

[WMR+13] J. Wu, R.R. Martin, P.L. Rosin, X.-F. Sun, F.C. Langbein, Y.-K. Lai, A.D. Marshall, and Y.-H. Liu. Making bas-reliefs from photographs of human faces. *Computer-Aided Design*, 45(3), 2013. 22

[WSTS08] T.P. Wu, J. Sun, C.K. Tang, and H.Y. Shum. Interactive normal reconstruction from a single image. *ACM Transactions on Graphics (TOG)*, 27(5):1–9, 2008. 21

[WXL+11] Y. Wang, K. Xu, J. Li, H. Zhang, A. Shamir, L. Liu, Z. Cheng, and Y. Xiong. Symmetry hierarchy of man-made objects. *Computer Graphics Forum*, 30(2):287–296, 2011. 61

[XLXJ11] Li Xu, Cewu Lu, Yi Xu, and Jiaya Jia. Image smoothing via L0 gradient minimization. *ACM Transactions on Graphics*, 30(6):174:1–174:12, December 2011. 58

[XZJ+12] Kai Xu, Hao Zhang, Wei Jiang, Ramsay Dyer, Zhiquan Cheng, Ligang Liu, and Baoquan Chen. Multi-scale partial intrinsic symmetry detection. *ACM Transactions on Graphics*, 31(6):181:1–181:11, November 2012. 67

[YM04] J. Yu and L. McMillan. General linear cameras. In *Proceedings of ECCV 2004*, pages 14–27. Springer, May 2004. 58

[ZFCO+11] Youyi Zheng, Hongbo Fu, Daniel Cohen-Or, Oscar Kin-Chung Au, and Chiew-Lan Tai. Component-wise controllers for structure-preserving shape manipulation. *Computer Graphics Forum*, 30(2):563–572, 2011. 61

[ZL10] Shizhe Zhou and Ligang Liu. Realtime digital bas-relief modeling. *Journal of Computer-Aided Design & Computer Graphics*, 22(3):434–439, March 2010. 18, 43

[ZMQS05] Gang Zeng, Y. Matsushita, Long Quan, and Heung-Yeung Shum. Interactive shape from shading. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 343–350, June 2005. 21

[ZSW+10] Qian Zheng, Andrei Sharf, Guowei Wan, Yangyan Li, Niloy J. Mitra, Daniel Cohen-Or, and Baoquan Chen. Non-local scan consolidation for 3D urban scenes. *ACM Transactions on Graphics*, 29(3):94:1–94:9, 2010. 61, 63

[ZTCS99] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape from shading: A survey. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 21(8):690–706, 1999. 20

[ZTS09] Rony Zatzarinni, Ayellet Tal, and Ariel Shamir. Relief analysis and extraction. *ACM Transactions on Graphics*, 28:136:1–136:9, December 2009. 20

[ZZZY12] Yu-Wei Zhang, Yi-Qi Zhou, Xiao-Feng Zhao, and Gang Yu. Real-time bas-relief generation from a 3D mesh. *Graphical Models*, 2012. In print. 17

Appendix A
List of Author's Publications

The following research papers and articles were 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 conferences and major journals in the area of computer graphics. They are listed in chronological order according to date of appearance.

[1] **Jens Kerber**, Art Tevs, Alexander Belyaev, Rhaleb Zayer, Hans-Peter Seidel: Feature sensitive bas relief generation. *In Proceedings of the IEEE International Conference on Shape Modeling and Applications (SMI)*, pages 148–154, June 2009.

[2] **Jens Kerber**, Art Tevs, Alexander Belyaev, Rhaleb Zayer, Hans-Peter Seidel: Real-time generation of digital bas-reliefs. *Computer-Aided Design and Applications (Special Issue: CAD in the Arts)*, 7 (4), pages 465–478, May 2010.

[3] **Jens Kerber**, Martin Bokeloh, Michael Wand, Jens Krüger, Hans-Peter Seidel: Feature preserving sketching of volume data. *In Proceedings of the 15th International Workshop on Vision, Modeling and Visualization (VMV)*, pages 195–202, November 2010.

[4] Meili Wang, **Jens Kerber**, Jian Chang, Jian J Zhang: Relief stylization from 3D models using featured lines. *In Proceedings of the 27th Spring Conference on Computer Graphics (SCCG)*, pages 63–68, April 2011.

[5] **Jens Kerber**, Michael Wand, Jens Krüger, Hans-Peter Seidel: Partial symmetry detection in volume data. *In Proceedings of the 16th International Workshop on Vision, Modeling and Visualization (VMV)*, pages 41–48, October 2011.

[6] Art Tevs, Alexander Berner, Michael Wand, Ivo Ihrke, Martin Bokeloh, **Jens Kerber**, Hans-Peter Seidel: Animation cartography - Intrinsic reconstruction of shape and motion. *ACM Transactions on Graphics*, 31 (2), pages 12:1–12:15, April 2012.

[7] **Jens Kerber**, Martin Bokeloh, Michael Wand, Hans-Peter Seidel: Symmetry Detection in Large Scale City Scans. *Technical Report, Max-Planck-Institut für Informatik*, MPI-I-2012-4-001, April 2012. Content corresponds to [10].

[8] Meili Wang, Jian Chang, **Jens Kerber**, Jian J Zhang: A framework for digital sunken relief generation based on 3D geometric models. *The Visual Computer*, 28 (11), pages 1127–1137, November 2012.

[9] **Jens Kerber**, Meili Wang, Jian Chang, Jian J Zhang, Alexander Belyaev, Hans-Peter Seidel: Computer assisted relief generation - A survey. *Computer Graphics Forum*, 31 (8), pages 2363–2377, December 2012.

[10] **Jens Kerber**, Martin Bokeloh, Michael Wand, Hans-Peter Seidel: Scalable symmetry detection for urban scenes. *Computer Graphics Forum*, 32 (1), pages 3–15, February 2013. Journal version of [7].