Digital Fabrication of Shape: Abstraction, Data Structures and Optimization

vorgelegt von
Kristian Hildebrand,
Diplom Mediensystemwissenschaftler
aus Berlin

von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
– Dr.-Ing. –

genehmigte Dissertation

Promotionsausschuss:
Vorsitzender: Prof. Dr.-Ing Alexander Raake
Gutachter: Prof. Dr.-Ing. Marc Alexa
Gutachter: Prof. Niloy Mitra, Ph.D.
Gutachter: Dr. Bernd Bickel
Tag der wissenschaftlichen Aussprache: 28.03.2014

Berlin 2014
D83

Abstract

We are currently witnessing a transition from the age of mass production to a world of personalized manufacturing. The transition is enabled by the wide-spread availability of manufacturing devices that facilitate rapid customization and fabrication, revolutionizing the way we design, develop, fabricate, and consume products. Although there has been significant progress in the development of manufacturing devices recently, software that allows end users to intuitively create digital content is largely underdeveloped.

This thesis develops software for digital fabrication that explores the continuum between accurate and abstract reproductions of shapes. We propose consumer-level rapid prototyping applications, including computational models for next generation personalized toys and 3D puzzles, which are becoming increasingly popular. The work introduces three approaches that demonstrate improvements for manufacturing methods and discusses shape representations and fabrication-constraint processing.

Our first application proposes an optimization scheme that addresses accuracy issues in layered manufacturing. We show that fabrication can be more accurate when a shape is decomposed into parts. The approach provides significant benefits if the shape is larger than the available production volume of the manufacturing device or the process resolution is low.

Our second application proposes a computationally efficient method to partition a digital input shape into parts. The approach is designed to overcome limitations of existing 3-axis machining such as 3D printing and 3-axis CNC milling. We demonstrate that our technique reduces the amount of support material required for 3D printing. Also, we present that it supports the creation of 3D design prototypes using a 3-axis CNC milling machine with acceptable fabrication errors.

A third application provides an algorithm for the automatic generation of cardboard models from a given 3D input shape. Laser cutters or CNC milling machines are used to fabricate the parts of the cardboard model which is assembled manually.

Zusammenfassung

Im Moment befinden wir uns im Übergang vom Zeitalter der Massenproduktion in eine Welt der individuellen Fertigung. Dieser Übergang wird durch die Verbreitung von 3D Druckern ermöglicht, welche die Art und Weise wie wir Produkte entwerfen, entwickeln, fertigen und konsumieren, revolutionieren werden. Obwohl in den letzten Jahren gro e Fortschritte in der Entwicklung dieser Fabrikationsgeräte gemacht wurden, blieb die Entwicklung dazugehöriger Software, die für Endnutzer intuitiv genug ist, um personalisierte Inhalte zu erzeugen stark zurück.

Die vorliegende Arbeit ist eingebettet in den Kontext dieser sogenannten 'Digitalen Fabrikation'. Es werden softwareseitige sowie algorithmische Lösungen präsentiert, die aus digitalen, physische Objekte erzeugen. Diese werden in unterschiedlichen Auflösungsstufen präsentiert, deren Varianz von einer möglichst exakten Reproduktion der Eingabemodelle bis hin zu einer Abstraktion der erzeugten Formen reicht. Es werden Softwarelösungen für die automatische Erzeugung von Kinderspielzeug und 3D Puzzles gezeigt, die in den letzten Jahren immer populärer geworden sind. Diese Arbeit legt au erdem Verbesserungen für Fabrikationsmethoden dar und diskutiert Formrepräsentationen.

In einer ersten Anwendung wird eine Optimierungsmethode für additive Fertigungsverfahren gezeigt. Für diese Optimierung werden die dreidimensionalen Eingabemodelle in Teile zerlegt, wobei jedes für sich, mit grö tmöglicher Genauigkeit produziert werden kann. Dadurch wird deutlich, dass dieser Ansatz präziser ist, als die Modelle in einem Stück zu fertigen. Das Verfahren zeigt zudem klare Vorteile, wenn das Objekt grö er ist, als das Produktionsvolumen des 3D Druckers oder wenn die Fertigungsauflösung produktionsbedingt nur sehr gering ist.

In einer weiteren Anwendung wird eine effiziente Methode zur Zerteilung eines Eingabemodelles in zwei Teile vorgeschlagen. Durch die Fertigung der beiden Einzelteile können zwei unterschiedliche Fabrikationsmethoden profitieren. Zum einen wird demonstriert, dass durch die Zerteilung der Verbrauch von Stützmaterial beim 3D Druckproze verringert werden kann. Zum anderen kann mit Hilfe des gleichen Ansatzes die Erzeugung von dreidimensionalen Designprototypen, unter der Benutzung einer 3-Achs-Fräse, unterstützt werden.

In einer dritten Applikation wird ein Algorithmus zur automatischen Erzeugung von dreidimensionalen Steckmodellen erarbeitet, welche mit Hilfe von Laserschneidgeräten produziert und dann von Hand zusammengesteckt werden können.

Contents

List of Figures

Chapter 1

Introduction

> *The end of the 20th century was about information becoming digital. The 21st century is going to be about bringing the virtual world into closer alignment with the physical one.*
>
> — Lipson and Kurman [2012]

People have created physical objects since prehistoric times and have tried to reproduce shapes from the real world. In the beginning, they sculpted stones using simple tools. Over the centuries, the tools became more advanced and eventually objects were manufactured with increasing precision by machines. This development required a specialization of objects, design and production processes, moving the creation of everyday goods mostly into the hand of specialists. Nowadays, the creation and production of man-made shapes is typically completely digitalized and many hardware and software processes are needed to create a physical object that meets necessary real world standards such as accuracy, stability and haptics. However, people that are not directly involved in that process typically do not have any influence on the result.

The creation of objects has always been a very creative process resulting in many levels of representations ranging from very exact and detailed productions and reproductions to abstract representations. Involving people into the creation of objects more directly again will introduces new ways on how to design, develop and fabricate digital objects.

We believe that recent developments in rapid manufacturing technologies, e.g. 3D printers, will make such an involvement of the end user in the creation process (again) possible. To make this practical, two main requirements need to be fulfilled. First, one needs to provide a consumer centered fabrication that allows the user to be involved in the design process [Gross 2007]. Second, manufacturing devices need to provide the possibility to produce "anything", i.e. the user should have the freedom to control shape, appearance and behavior of objects as much as possible [Lipson and Kurman 2012]. We also believe that these requirements could potentially have a very large impact and galvanize a new industrial revolution that leads from from the current "age of mass production" to an "age of custom fabrication" [Anonymous 2012]. Figure 1.1 shows a

Figure 1.1: Compact summary of tools, techniques and their resulting shapes over the centuries on the example of a vase. Although the same techniques are still used today the complexity of shapes that can be fabricated increased with new tools and devices. From left to right: coil-built pottery, Vases created by a potter's wheel, Murano glas vase, CNC milled vase, and Snotty Vases, a 3d-printed vases based on scans of airborne snot molecules.

very selective view on this development going from ancient forms and the tools that have been used to new manufacturing technologies and the produced shapes.

This transformation from mass to customized production creates challenges for many areas of scientific research, such as material science, engineering, but also, and perhaps foremost, computer sciences. While there has been significant progress in the development of manufacturing devices, software and computational models that allow end users to intuitively create digital content that can be physically produced is largely underdeveloped.

Additional, with a change in these requirements there is a clear need for computational models that support production-centered thinking and allow intuitive design, efficient representation, fast simulation and visualization of physically realizable objects [Bickel and Alexa 2013].

This dissertation is embedded in the context of New advances in, what we refer to as, digital fabrication and computer graphics research. The computational approaches for realizing digital objects in physical real-world representations explored in this thesis examine different levels of shape abstraction. We propose consumer-level rapid prototyping applications, including computational models for the next generation personalized toys and 3D puzzle which are becoming increasingly popular. The work also demonstrates improvements for manufacturing methods and discusses shape representations and fabrication-constraint processing.

1.1 The Age of Digital Manufacturing

Digital manufacturing technologies are not new. The first 3D printers were developed in the 1980s and many other production processes, such as laser cutting or computerized numerical controlled (CNC) milling exist for many decades. Next to the widely used manufacturing devices, there also exist a variety of applications such as CAD and solid modeling, numerically controlled machine tools and simulation tools for injection molding to just name a few examples. The term *Digital Manufacturing* or *Computer Aided*

Manufacturing is commonly used in mechanical engineering to refer to the computer controlled process of production. This technology exists for more than fifty years and is still in active development. There are millions of parts in the automotive and aerospace industries produced by numerically controlled manufacturing. In other words, there is and has been a world of highly sophisticated technologies for the computer-controlled and computer-assisted production of complex engineering parts. However, in these applications the consumer is completely left out of the production process and the creation and manufacturing process is performed by experts using expert systems.

When we speak of the digital age in manufacturing we speak of the world of consumer products that are created by the consumers herself and are fabricated by output devices directly available to the consumers. The development is driven by improved hardware and new materials that are entering the mass market and enable users to design, develop, distribute, fabricate and consume products like never before.

Two major influences have been catalyzing the development of consumer-driven manufacturing. First, the influence of a large open-source community that helped the development of affordable 3D printers after the expiration of the patents on 3D printing technology. Second, Internet and mobile technologies and an increasing number of SAAS (software-as-a-service) platforms give people the opportunity to create, modify and order things online, using intuitive interfaces for creating or personalizing everyday things [Autodesk 2012] [Makerbot 2012]. Also businesses benefit significantly from the improved manufacturing hardware and software interfaces through the opportunity of enabling people to customize their products online, from shoes to cars, from glasses to cloth. The impact of this development could be tremendous, nothing less than a change of the production cycle through a second industrial revolution [Anonymous 2012]. The entire process of design, engineering and manufacturing becomes increasingly iterative, non-linear, localized and consumer-centric. The role of the passive consumer is transformed to that of an active participant in the product development cycle [Vilbrandt et al. 2008]. In other words, we might be in the process going from mass production to personalized manufacturing and customization [Malone and Lipson 2007] [Gross 2007].

1.2 Computer Graphics and Manufacturing

The influence of computer graphics research on the manufacturing processes has been large from the beginning. For example, Computer-Aided Design (CAD) and solid modeling revolutionized 2D and 3D drafting. Nevertheless, over the last 15 years the computer graphics pipeline was optimized for keeping data digital, outputting the content typically on displays for games and motion pictures.

There have been advances for digital 2D printing, e.g. PostScript simplifies and optimizes content creation. In contrast, 3D output devices, e.g. 3D printers, laser cutters or CNC milling machines, are hardly supported in software tools. This makes it difficult to take full advantage of their capabilities. For example, it is often not possible to simulate or preview the output of the devices, hampering the control of appearance and geometrical or mechanical properties. Moreover, no universal framework exists that ensures data transformation across different manufacturing output devices. Only recently, first steps for defining a general concept of a fabrication-oriented or process-based graphics pipeline for 3D printing have been made [Vidimče et al. 2013], [Chen et al. 2013], [Winkelmann 2014]. A look at the simplified computer graphics pipeline in Figure 1.2 shows that we are able to convert the real world into mathematical models in a computer but little standardized infrastructure exists converting them back to

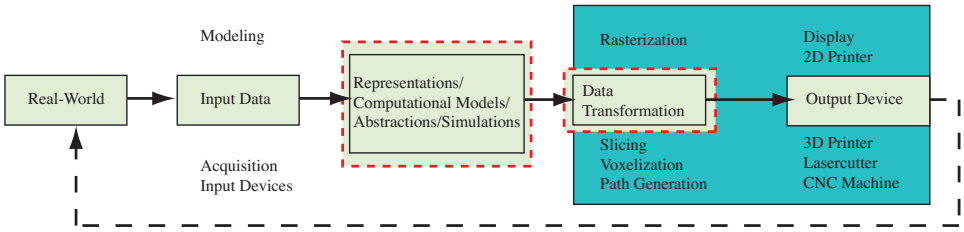


Figure 1.2: Simplified graphics pipeline with a focus on digital fabrication. By improving input devices and capabilities of output devices the loop between digital and real world can be closed. Top parts in green show the 2D rendering/printing pipeline, bottom parts the fabrication pipeline, algorithms and technologies. The areas of contributions described in this thesis are marked in red.

reality with adjusted personalized properties. One of the broader research goals of digital fabrication is to close this loop. Figure 1.2 marks in red the areas of contribution which are presented in this thesis.

Frequently used terms and and concepts:

Layered Manufacturing Layered manufacturing is the process of creating a three-dimensional object by successively depositing layers of material in different shapes. 3D printing is an layered manufacturing process and driven by machines with a controlled movement along three axes.

Additive Manufacturing Additive manufacturing is often used as a synonym for Layered Manufacturing but more correctly the generic term for 3D printing. It relaxes the constraint of depositing layers by the use of multi-axis controls for 3D printing technologies.

Support Material Support material is necessary in many 3D printing technologies. When material is deposited or fused to form a real world model of the digital input shape overhangs are strut by using so called support material. The material is usually different to the object material and can be washed away or taken off after the print-out is finished.

Subtractive Manufacturing Subtractive manufacturing summarizes many techniques, such as milling, cutting and drilling, that cut a final shape out of a raw material stock by a computer controlled removal process.

Production Volume The production or fabrication volume is the physical space inside the manufacturing device where the object is produced.

Fabrication Direction The fabrication direction is defined as the direction or machine axis along which the object is manufactured. Throughout this thesis we show that the rotation of the digital shape, and therefore the placement of the produced object inside the production volume, has influence on many aspects of the result.

1.3 Outline

In this thesis, we focus on the digital fabrication of shape at different levels of abstraction. We present three applications that vary from optimized reproduction to abstract representation of shapes using different manufacturing devices. The thesis is organized as follows:

Chapter 2 provides an overview of the manufacturing processes relevant for this thesis and presents a comprehensive introduction and computer graphics-centric classification of recent research in graphics and digital fabrication. Additionally, we organize the recent literature in a spectrum from exact reproduction to abstract descriptions and position our research in this taxonomy.

Chapter 3 and Chapter 4 briefly summarize a common set of algorithms, data structures and optimization methods that have been used throughout our applications.

Chapter 5 presents an optimization method for additive manufacturing. We provide a solution for three challenges that are solved in this work. First, we evaluate the choice of the object's rotation inside the production volume which has a large influence on the accuracy of the process. Second, an eligible data representation is presented to evaluate production errors. Third, derived from the data representation a meaningful object decomposition is proposed to segment a shape into parts that are fabricated most accurately. In simulation we show that this approach is superior to producing the whole shape in one direction only. It also has clear benefits if the shape is larger than the production volume.

Chapter 6 presents a case study for partitioning an input mesh into two parts and fabricate each part individually. This binary space partition approach is specifically useful to overcome limitations for 3-axis machining such as 3D printing and 3-axis CNC milling. In the case of 3D printing, we show that one can significantly decrease the usage of support material. In the case of 3-axis CNC milling, we demonstrate how to create physical 3D shapes with an acceptable fabrication error.

Chapter 7 provides an algorithm for the automatic generation of cardboard models from an input shape to a construction plan. Laser cutters or CNC milling machines are used to fabricate the parts of the cardboard model which is assembled manually in the final step. This work proposes solutions to the three main challenges. First, the choice of planes which are used to represent the abstracted input shape. Second, depending on the number of planes and the order of construction, we initialize a data structure necessary to spatially organize and guide the construction process algorithmically. Third, with a growing number of planes the choice of ordering gets complex. We use a discrete optimization method to maximize the compactness of the shape representation.

Chapter 8 draws conclusions, discusses limitations and provides directions for future research.

1.4 Publications

Most of the results presented in this thesis have been published in the following conference and journal papers.

- The presented approach on 3D models composed of interlocking planar pieces [Hildebrand et al. 2012], of Chapter 7 has been published at Eurographics in Cagliari, Italy in 2012. It has been developed in collaboration with Bernd Bickel and Marc Alexa from TU Berlin.
- The improvements on additive manufacturing [Hildebrand et al. 2013], of Chapter 5 was published at the Shape Modeling International Conference in Bournemouth, UK in 2013. The project was also a collaboration with Bernd Bickel and Marc Alexa from TU Berlin.
- Section 2.5 in Chapter 2 briefly presents a novel sketch-based application workflow for personalized manufacturing [Hildebrand and Alexa 2013] which has been published at the Design and Modeling Symposium 2013, Berlin, Germany. The work was joined effort with Marc Alexa from TU Berlin.

Chapter 2

Related Work

No one s going to print a working AK-47 with this, Sammy said.

— Cory Doctorow, Makers

The focus of this thesis is on the fabrication of shape at different levels of abstraction. In this chapter we give a comprehensive introduction to the underlying technologies and provide a taxonomy of recent manufacturing publications in the computer graphics literature. Additionally, we summarize work on intuitive interfaces that support users in their creativity. Recently, there has been a tremendous development in digital fabrication hardware and software to enable end users possible to create high-quality personalized or configurable products that embody effects in shape, appearance and behavior. Broadly speaking we categorize:

Computational models for the fabrication of shapes that propose geometric processing algorithms to model and modify digital shapes to achieve desired geometric properties or act within the constraints of manufacturing technologies or introduce construction plans to fabricate objects.

Computation models for controlling the variations of appearance and the interplay of light and material which a central aspect of graphics research. 3D printing and real-world materials offer interesting possibilities to produce or reproduce desired perceptional effects.

Computational models for reproductions of digital animation, deformation or kinematics that mimic real-world deformable behavior, moving mechanical parts and show perspectives for other research areas such as robotics.

We organize the recent literature using these objectives and arrange them within each class from exact representation to abstraction of the original input model. This differentiation naturally corresponds to our applications and also provides an insightful classification for other research projects. However, we begin by providing an overview of manufacturing technologies.

(a) (b) (c) (d)

Figure 2.1: (a) Fused Deposition Modeling 3D Printer ©MakerBot Industries. (b) Polyjet 3D Printer Object Connex 1000 ©Stratasys Ltd.. (c) CO_2 Laser Cutter ©Epilog Laser. (d) 3-axis CNC milling machine ©VK Technik GmbH.

2.1 Manufacturing Technologies

Manufacturing processes that create a desired real world object out of raw material can be distinguished by controlled material removal, known as subtractive manufacturing, and controlled material addition, referred to as additive manufacturing or 3D printing. We provide a very compact overview of the machining processes. The properties that we are mainly interested in in the context of this thesis are: accuracy of the machining processes, geometric complexity, and size of the producible objects.

Additive Manufacturing and 3D Printing Technologies

The central operation in Additive or Layered Manufacturing techniques is the discretization of a digital object into a set of layers with distinct height along the *fabrication direction*. The physical shape is then generated by bounding or depositing layer after layer of raw material to form a solid three-dimensional object. This allows rapid development of prototypes to be produced by engineers and designers. As 3D printers are becoming more capable and able to work with a broader range of materials, including production-grade plastics and metals, the machines are increasingly being used to fabricate final products [Anonymous 2011]. Additive manufacturing processes enable to fabricate any non-degenerated manifolds optimized in shape and weight.

Additive manufacturing methods are well evaluated and analyzed. A number of methods address the task of finding an optimal orientation of a single part [Alexander et al. 1998], considering surface finish, evaluate the surface roughness and part deposition time [Ahn et al. 2009], [Thrimurthulu et al. 2004], [Canellidis et al. 2006]. Danjou and Köhler [2009] suggest an optimization procedure based on a genetic algorithm to improve the printing orientation. [Masood et al. 2000] propose methodologies for computing the correct orientations based on the minimum volumetric error of basic primitives. However, none of these approaches considers segmenting the model into sub-parts with different orientations as proposed in Chapter 5.

In the following we will distinguish between two techniques, selective deposition printers and selective binding printers [Lipson and Kurman 2012].

Selective Deposition Printers These printers deposit raw material of the printing head into thin layers. The probably best known example is the *Fused Deposition Modeling (FDM)* (see Figure 2.1(a)) that deposits thin strings of fused plastic out of a hot end of a 3D printer head and fuses it layer by layer. Another method is called *Polyjet printing* that sprays liquid photopolymer in very thin layers (up to 16 microns) and

firms it up using ultraviolet light (see Figure 2.1(b)). Selective Deposition Printers can work with several printing heads and different materials for a single 3D print, e.g. to vary optical or rigidity properties. An alternative technology is the *Laser engineered net shaping (LENS)* technique that blows metal powder in the focal point of a laser. Particles that hit the focal point instantly melt onto the existing part surface. The advantage of this method is the ability of mixing metal powders to increase rigidity [Lipson and Kurman 2012]. *Layered Object Manufacturing (LOM)* laminates sheets of material into a 3D shape. Thin paper, plastic or metal is stacked, cut by a machine knife or laser cutter and glued together.

Selective Binding Printers This technology binds raw materials typically in powder form using a laser beam. The laser beam is located in the print head and moved over the printing volume. Thereby, a thin layer of powder is distributed over the printing surface and a laser beam focused on the slice of powder melts the raw material. *Stereolithography (SL)* uses liquid polymer in the production volume instead of powder. Ultra violet (UV) light is then used to sweep over the polymer to bond the fluid. This process is repeated layer by layer. Within a post-process the 3D printed object has to cured in an UV oven. *Laser sintering (LS)* is similar to Stereolithography but uses powder and also metal. In a post-process the solid object is infiltrated with epoxy resin. *Three-dimensional printing (3DP)* uses a print head that squeezes glue onto raw powder combining the different slices.

Materials Raw materials for 3D printing significantly depend on the process. Fused deposition modeling usually uses thermoplastics, such as polyactide (PLA), Acrylonitrile butadiene styrene (ABS) and Polyvinyl alcohol (PVA). However, every material which can be melted and formed by the extruder can be used to create the shape, e.g. sugar, chocolate and pastry. Other 3D printer techniques use polymers and resin that are sensitive to specific light wavelengths to bond. Selective binding printers offer different powders from plastic to metal. Currently, most of the objects generated with these materials are not rigid and persistent enough to replace production grade shapes.

Printing resolution and production volume size The printing resolution is dependent on the hardware, the material properties and software parameters. Overall 3D printers have a resolution, i.e. layer height, in the fabrication direction (which is the lowest process resolution) between 0.01mm and 0.5mm. Most of the 3D printers do not exceed a production volume size of $300 \times 300 \times 300$mm. There are many advances on mechanical engineering and architecture to build large scale 3D printers with lower resolutions [Monolite Ltd. 2012], [Bytes 2012] but large production volume.

Subtractive Manufacturing

Classic manufacturing methods, e.g. milling, drilling, sawing or cutting work by removing raw material to form a desired shape. In this thesis we use subtractive manufacturing methods such as CO_2 laser cutter (see Figure 2.1(c)) or a *Computerized Numerical Controlled (CNC)* 3-axis machine (see Figure 2.1(d)) to create the presented shapes.

CNC Milling encompasses a range of processes, one of them is milling. In milling, a tool mounted onto a movable arm removes raw material from a stock. Thereby, Computer-aided manufacturing (CAM) software is used to generate routes along an

input geometry that are travelled by the tool to remove material. The process is executed several times with an increasing resolution of tools. This iteratively approximates the original surface from a *roughing* to a *nishing* process. The process can work on any rigid material and is highly accurate (up to 0.001mm).

However, the number of axis the CNC operates on determines the complexity of the geometry that can be produced. For instance, a 3-axis CNC machine moves the tool on the horizontal 2D X-Y plane and along the vertical direction in the third axis. The geometry that can be generated is the projectional hull of the surface along the vertical axis. To create curved holes one needs at least 4-axes. Modern industrial machines have up to 11-axes. In Chapter 6 we use these restriction observations.

Laser cutting is a precise CNC process that can be used to cut, engrave or mark a variety of sheet materials, including plastic, wood, cardboard, glass and others. It works by focusing thermal energy to melt or vaporize material. The laser beams are guided through a series of mirrors to a nozzle that move with the X-Y plane traveling routes given from a 2D contour. The created cut is typically 0.1mm wide. All fabricated shape abstractions presented in Chapter 7 were created using a laser cutter.

Summarized comparison of both manufacturing techniques:

	Additive	**Subtractive**
Geometric Complexity	any non-degenerated two manifolds	restricted by number of axis, no holes and inclusions
Materials	very specific to the process; neither very rigid no long-living; thermoplastic, resin, different material powders	any rigid material
Accuracy	low (large-scale or consumer-printer); medium-high (industrial 3D printers)	very high
Production Volume Size	small-medium	small-large

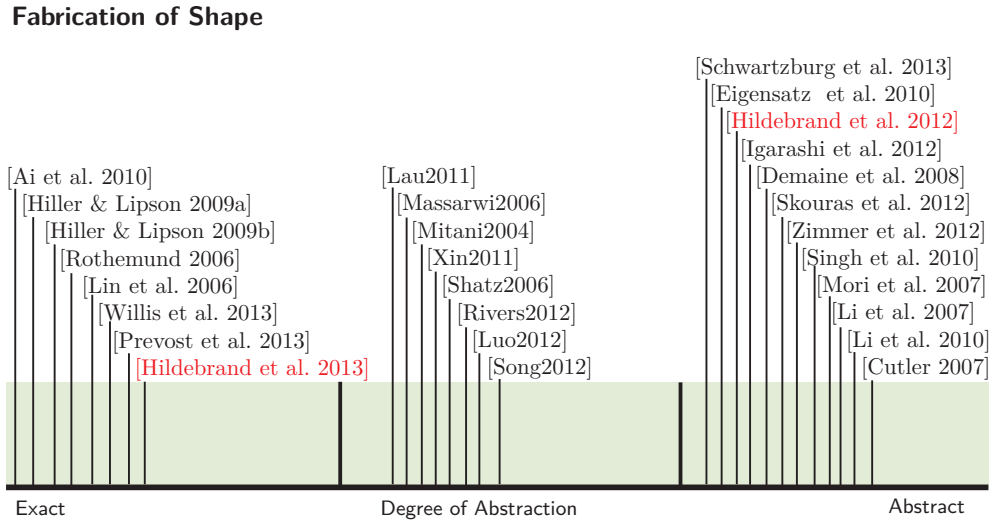


Figure 2.2: Illustration of recent research focusing on the fabrication of shape organized from an exact representation to an abstraction of the form.

2.2 Fabrication of Shape

The main goal of digital manufacturing is to bring the digital 3D shape content from a digital representation to a physical one. Various research initiatives have targeted this in various ways: different scales, degree of abstraction or manufacturing constraint assembly and production as shown in Figure 2.2. We will now look at these different works in more detail.

From Nano to Macro

In recent years, much effort has been devoted to creating algorithms for the control of amino acid sequencing to form a specified shape, also called DNA Origami [Rothemund 2006], [Lin et al. 2006]. On the other end of the scale, the construction of free-form surfaces became of practical importance for designing modern architecture. Cutler and Whiting [2007] and Eigensatz et al. [2010] propose methods to approximate free-form surfaces by developable surfaces that can be manufactured. Other works by Fu et al. [2010], Singh and Schaefer [2010] and Zimmer et al. [2012] focus on the problem of minimizing the number of individual construction tiles when building complex structures. The demands on producible complex large scale geometry led to a development of additive manufacturing technology such as D-shape [Monolite Ltd. 2012]. It follows the idea of increasing the possible 3D printing volume to be able to construct buildings.

Segmentation and Assembly

The decomposition of complex shapes into parts is central to many design and manufacturing processes. Several reasons exist for this, e.g. each part has to be fabricated with a different material or manufacturing device, objects do not fit as a whole in the production volume or efficient transportation requires smaller parts.

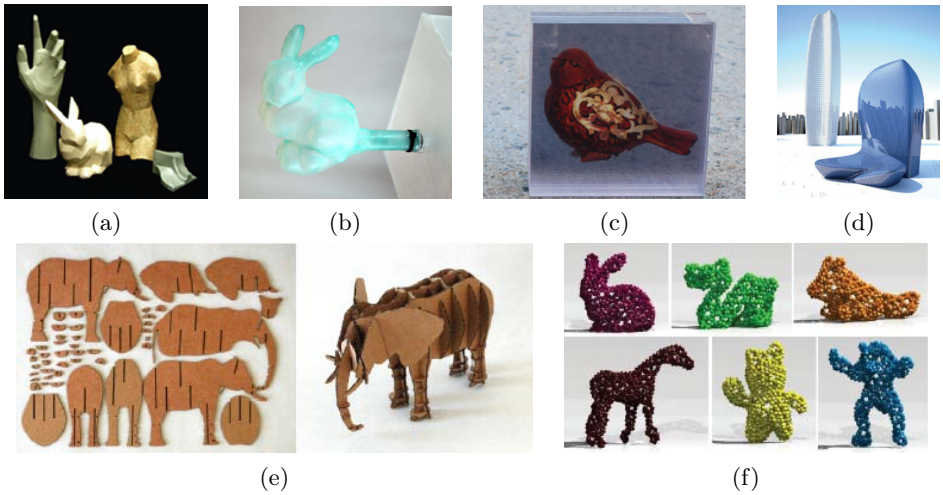


Figure 2.3: Shape Projects. Computational abstraction and physical reproduction of shapes for creating (a) papercraft models from meshes [Shatz et al. 2006], (b) rubber balloons [Skouras et al. 2012], (c) multilayer models [Holroyd et al. 2011], (d) paneled freeform surfaces [Eigensatz et al. 2010], (e) cardboard models [Hildebrand et al. 2012], and (g) interactive beadwork design [Igarashi et al. 2012] .

First advances in partitioning 3D shapes into pieces that fit in a 3D printing volume were presented by Luo et al. [2012] and are discussed as part of this thesis in Chapter 5.

Obviously, parts have to be connected and the object has to be constructible. Segmenting arbitrary digital objects into single manufacturable parts and designing connectors for the construction plan was proposed by Lau et al. [2011] for furnitures. Notably, Xin et al. [2011] and Song et al. [2012] propose systems to decompose a 3D shape into several interlocking parts, allowing to automatically generate burr puzzles. Here connectors are not necessary.

Shape Abstraction

The abstraction of shape plays an essential role in representing objects for various objectives [DeCarlo and Stone 2010], [Mi et al. 2009]. It is usually perceived as aesthetically pleasing and widely used as an artistic and scientific tool for illustration, stylization, and highlighting surface features. Our visual memory creates the impression of complete shape appearance yet giving space for shape interpretations and creativity. In computer graphics, numerous processes have been studied to compute efficient shape representations and simplifications [Luebke et al. 2002]. These representations provide inspiration for many fabrication approaches. An example of an abstraction is the idea of an exoskeleton as an abstraction of shapes [De Goes et al. 2011]. The exoskeleton as the external shell is a combination of geometry and a set of disk-like patches. In this spirit, Mehra et al. [2009] extract only the characteristic curves of 3D man-made shapes to provide a compact and representative version of the models. Pushing the level of abstraction even further, Décoret et al. [2003] suggest billboard clouds as an extreme simplified representation for 3D objects.

Recently, McCrae et al. [2011] and McCrae et al. [2013] proposed a powerful approach

for generating shape-proxies consisting of planar sections based on principles inferred from user studies. Also, planes are used as basic primitives, but are treated as an unstructured set and benefit from image-based impostors. We introduce this approach in the context of digital fabrication [Hildebrand et al. 2012]. We present the creation of cardboard models as potential application. Chapter 7 goes into detail that application. Similarly, Schwartzburg and Pauly [2013] add additional construction possibilities.

Recently, Demaine et al. [2008] proposed an algorithm for balloon twisting to create objects. Skouras et al. [2012] fabricated balloons in a desired shape. Given a target shape, they compute an optimal balloon that, when inflated, approximates the target as closely as possible. Igarashi et al. [2012] developed a framework to design and construct customized 3D beadworks. Igarashi and Demaine use every day goods and do not need a specific output device to create 3D shape abstractions. In contrast, Skouras and colleagues depend on a complete pipeline of output devices for the digital manufacturing process.

Computational models for constructing and designing physical models out of paper gained interest in the computer graphics community. Existing methods can be classified by the type of basic elements used for fabrication. While Li et al. [2007] have shown that models can be augmented with paper-cut patterns to support the perception of texture, there exist several approaches that address forming 3D shapes. Paper-crafting in art has a long history and dates back nearly 2000 years to the invention of paper. Origami, the art of paper folding, creates intricate structures from a flat piece without cutting or gluing. An overview and introduction to mathematical folding algorithms can be found in [Demaine and O'Rourke 2007].

There are several methods that approximate a 3D object with a set of paper strips that can be folded and glued. Mitani and Suzuki [2004] and Takahashi et al. [2011] segment the input mesh and represent it with a set of strips that can be crafted by bending the paper which also allows to represent smooth features. Massarwi et al. [2007] use a set of developable surfaces, each one being a generalized cylinder represented as a strip of triangles. [Shatz et al. 2006] follow a similar approach but restrict their elements to cones and planes.

Shape e ects and precision

Once the user has full control of the shape properties many desired effects can be embodied. For instance, Prévost et al. [2013] propose to assist users in producing properly balanced designs by interactively deforming an existing model and manipulating the center of mass. They formulate balance optimization as an energy minimization, improving stability by modifying the volume of the object, while preserving its surface details. Umetani et al. [2012] also experiment with balanced objects but in the physical valid scope of furniture design. Willis and Wilson [2013] embody information into a 3D printed shape. They define material-based passive tags that embed machine-readable information in the interior of physical objects. By varying material geometry within an object, hidden information can be used in applications ranging from inventory control, over real-time game interaction, to other systems linking physical objects to the digital world.

Optimizing the 3D printing process in terms of precision has a long history in machine engineering [Hiller and Lipson 2009], [Masood et al. 2000], [Canellidis et al. 2006]. None of these approaches work for general meshes. Chapter 5 elaborates our approach of optimizing the quality of an additive manufacturing process [Hildebrand et al. 2013].

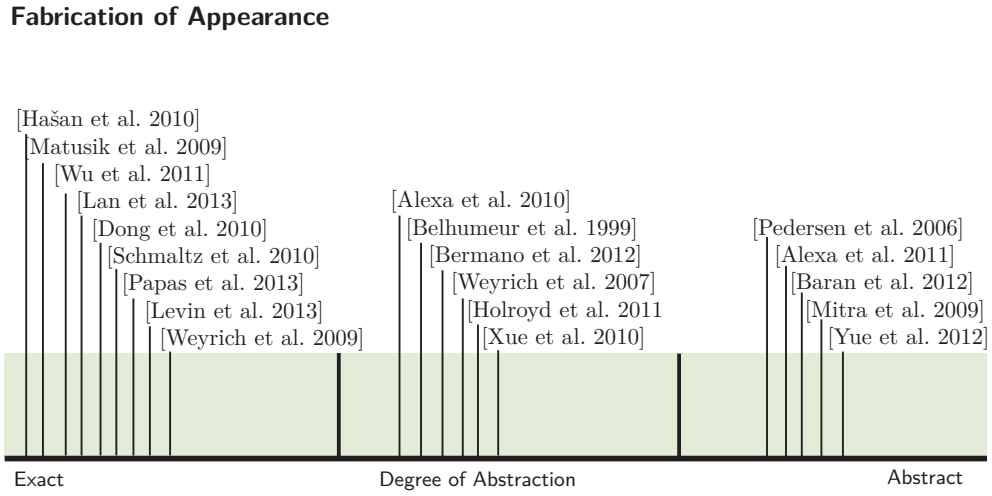


Figure 2.4: We summarize computational models for the reproduction of shape appearance. These are the classes we consider from exact to abstract models.

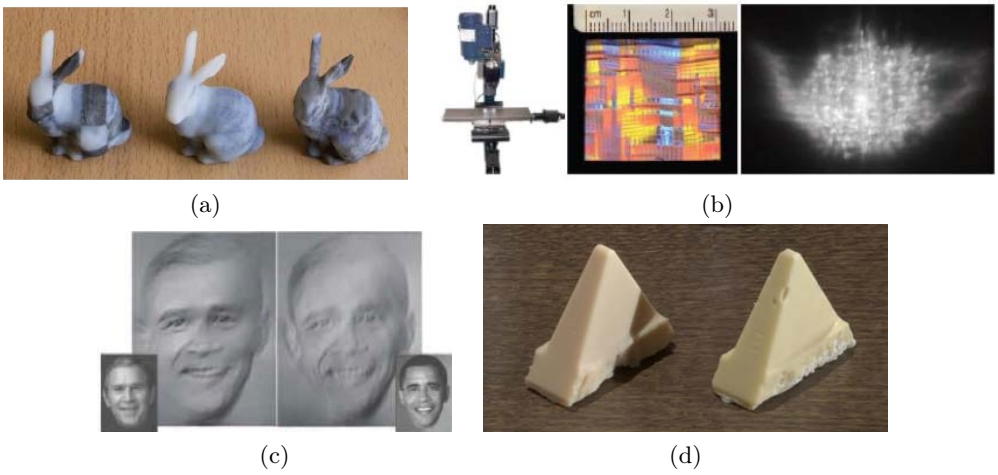
2.3 Fabrication of Appearance

Appearance is the result of the interplay of light, material and geometry. The scattering of light depends on its micro (on the order of a micron) and meso (on the order of 0.1 to 1 mm) scale structure. Appearance reproduction can be achieved by separately considering the design of the shape of an object and the design of the material it is made of. Appearance abstraction can be designed by considering shape and material together, along with the effect of light scattering from the object into the environment, see Figure 2.4.

Appearance Reproduction

Recently, several researchers have considered the control of small scale geometry to control the appearance of an object. Three different research directions can be identified. The reproduction of the bidirectional reflection distribution function (BRDF), the simulation and manufacturing of subsurface scattering behavior and goal-based caustics. Weyrich et al. [2009], Levin et al. [2013] and Lan et al. [2013] propose a system for manufacturing physical surfaces that show a desired surface reflectance. In two dimensions Matusik and colleagues use a 2D printer and a specialized set of ink to reproduce physically correct BRDFs [Matusik et al. 2009] metallic or glossy reflection properties.

Hašan et al. [2010] and Dong et al. [2010] suggest a reproduction pipeline for measuring and fabricating object materials with a specific subsurface scattering behavior. Papas et al. [2013] use a mixture of colored pigments to influence the scattering properties in liquid silicone to successfully mimic a variety of materials. Wu et al. [Wu et al. 2011] have recently developed a system that allows a user to interactively view the simulated macro-appearance of a material as the user edits the mesoscopic scale structure of a material. This facilitates, for the first time, the design of physically realizable materials, rather than simply physically plausible materials. Moreover, using the optical phenomenon of light refraction for designing and manufacturing surfaces that produce



(a) (b) (c) (d)

Figure 2.5: Shape appearance reproduction. Physical reproduction of materials with (a) subsurface scattering [Hašan et al. 2010], (b) micro-geometry for custom surface reflectance [Weyrich et al. 2009], (c) reliefs as images [Alexa and Matusik 2010] and (d) Pigment mixtures for fabricating translucent materials.

desired caustic images when illuminated by a light source was presented by Papas et al. [2011] and Yue et al. [2012].

Appearance Abstraction

By considering the effect of light, material and geometry together, one can create the illusion of a particular shape. Considering the human perception of shape and shape appearance during the design, modeling and machining of objects proposes a new concept for the creation of shape. While Rusinkiewicz et al. [2006] or Ritschel et al. [2008] show that exaggeration of shape features improve its perceived appearance in virtual environments, Xue et al. [2010] demonstrate that already the printing of patterns enhance shape perception. Working with the interplay of shape and its appearance leads to a major challenge: the optimization and manufacturing of real shapes based on likely viewing positions to convey important features. This is common approach in traditional sculpting.

One example is the creation of reliefs. The goal is to keep the notion of a 3D shape but compressing the height greatly yet still inducing the perception of depth. Belhumeur et al. [1999], Song et al. [2007] and Weyrich et al. [2007] developed optimized algorithms to create versions of a 3D shape with compressed height. Alexa and Matusik [2010] also construct a relief surface but do not create the notion of depth. Instead they optimize the relief to form two different images for different light directions. This method considers only the effect of diffuse shading and assumes no self-shadowing or shading effects. In contrast, shading due to self-shadowing is not a local effect: a surface point can cast a shadow on another point that is potentially far away. This non-locality makes it more challenging to control self-shadowing, but in return, it allows to encode more images into one surface and obtain sharper results. This was recently proposed by Bermano et al. [2012]. Similarly, the problem of computing a surface that depicts a given image

based on the occlusion of small embedded holes was proposed by Alexa and Matusik [2011]. The surface was fabricated using CNC milling machines.

The interaction of light and surfaces is one of the central topics in computer graphics. Computing a surface that creates a set of given images by taking light sources and its physical effects into account has gathered much interest in the community. Creating objects with a specific micro-surface geometry that uses light sources to create a certain image was presented by Baran et al. [2012]. They compute optimized attenuation masks which are then printed on transparent materials and stacked to form a single multi-layer attenuator similar to Holroyd et al. [2011] and Wetzstein et al. [2011] but with a completely different motivation. Computing shape from shadows (darkness) is a classic problem studied by computer vision researchers. Mitra and Pauly [2009] show the inverse process. They generate a volumetric structure that casts three different user-specified binary shadows onto planar surfaces that best approximate the input stencils.

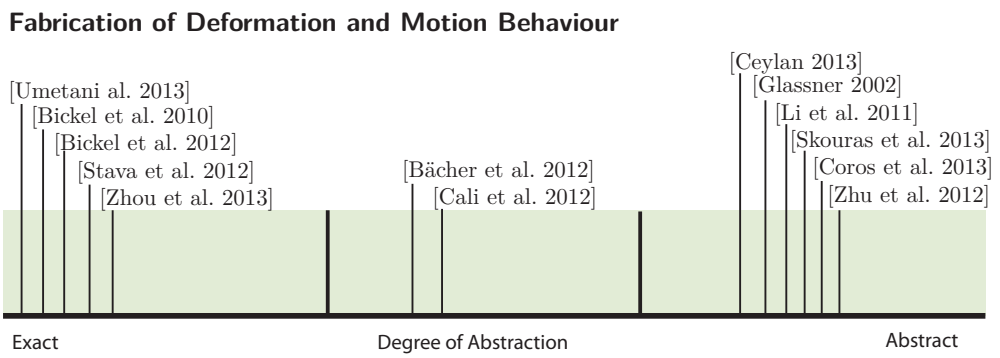


Figure 2.6: Fabrication of Deformation and Motion. In each class we consider a spectrum from exact reproduction to abstraction.

2.4 Fabrication of Dynamic Objects

Elastic deformations and rigid body motion are present in many objects in our everyday life. When we design representations of dynamic objects, either for animations in the computer or real world movements, we are faced with determining material properties and motion parameters such that the objects behave in a desired way. Figure 2.6 classifies recent research developments for fabricating dynamic objects from exact to abstract reproductions.

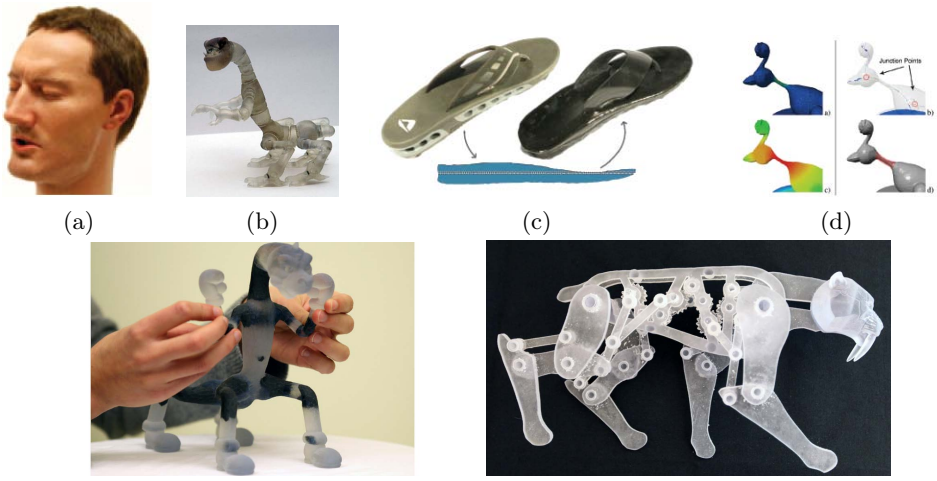
Reproducing Deformable Objects

A first approach by Bickel et al. [2010] demonstrates a goal-based design of deformable models with anisotropic and non-linear behavior. It enables the physical fabrication of these materials with the help of multi-material 3D printers such as the OBJET Connex series. The approach starts with measuring deformation properties of 3D printer base materials by acquiring a set of example deformations. The material is represented as a non-linear stress-strain relationship in a finite-element model. The material measurement process is validated by comparing simulations of arbitrary stacks of base materials with measured deformations of fabricated material stacks. After material measurements, an example deformation is given and an optimization is introduced to mimic that example by designing stacked layers of the base materials.

Next to the reproduction and deformation of solid objects another line of research is the design and computation of deformable shells. This approach has a large number of applications, e.g. for medical purposes [Glozman et al. 2010], soft robotics [Shepherd et al. 2011], or entertainment [Skouras et al. 2012].

Physical Objects and Approximate Motion

Creating a 3D hardcopy of an animated computer graphics game character is currently still well beyond the reach of consumers. Even for professionals it remains a difficult task to approximate the character's appearance and deformation behavior. Recently, researchers took a first step into automating this process [Bächer et al. 2012], [Calì et al. 2012] and [Skouras et al. 2013] by using multi-material 3D printers.



(a) (b) (c) (d)

(e) (f)

Figure 2.7: (a) Physical face cloning [Bickel et al. 2012], (b) Design of articulated characters [Bächer et al. 2012], (c) Fabrication of objects with designed deformation behavior [Bickel et al. 2010], (d) Improving structural strength of printable objects [Stava et al. 2012], (e) Actuated deformable characters [Skouras et al. 2013], (f) Design of mechanical characters [Coros et al. 2013]

A mechanical process using gear-wheels to drive motion is approached by Coros et al. [2013] and shown in Figure 2.7(f). They present the computational design of mechanical objects that given an articulated character and sketched motion curves as input creates an animated mechanical character as output. A similar goal but with motion capturing input data is pursued by Ceylan et al. [2013].

Driving this further to the creation of real articulated rigid structures brings us to the field of animatronics. The aim here is to develop physical robot characters that move and look like real humans. [Bickel et al. 2012] recently presented an animatronic figure that closely resembles either a given virtual character or even a human subject. The latter one is achieved by attaching a fabricated synthetic silicone skins to an articulated robot head.

As mentioned before, papercraft is a very popular low-tech fabrication method. Paper "pop-ups" can be designed with volvelles, flaps, pull-tabs, pop-outs or pull-downs to recreate animations and simple abstract movement to enrich non-rigid objects. Computational paper models that can be popped-up in a rigid and stable manner were recently presented by Li et al. [2010]. In this work, a three-dimensional building shape is approximated as a set of parallel planes. Another interesting class are v-style pop-ups, which can be opened and closed, i.e. moved into a flat state, without changing the rigidity of the structure or extra force except at two patches. Such pop-ups can be automatically generated [Li et al. 2011] and are used for books or cards [Glassner 2002]. While pop-ups are inherently intriguing and mathematically interesting, usually fabrication might require gluing.

In contrast to pop-up books where the structural strength is naturally given, in three dimensions there is no guarantee that a 3D printed model is structurally sound. The printed product often does not overcome cleaning, transportation, or handling, or it may

even collapse under its own weight. Stava et al. [2012], Wang et al. [2013] and Zhou et al. [2013] improve structural strength of 3D printable objects by automatically detecting and correcting structurally weak regions. The structural problems arise in areas of high structural stress. In an optimization step the model is corrected by combining three approaches: hollowing, thickening, and strut insertion [Stava et al. 2012]. Zhou et al. [2013] propose a stress optimization by deformation to solve the latter mentioned problem. Umetani and Schmidt [2013] analyze the fused deposition modeling process and determine a fabrication direction that increases the stability of the printout.

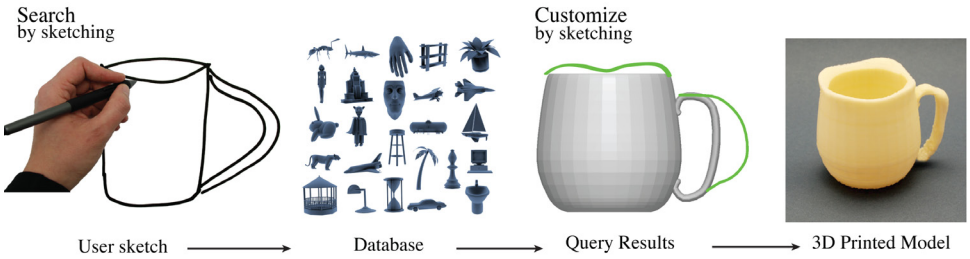


Figure 2.8: A sketch-based pipeline to facilitate personalized manufacturing is shown. The process starts by sketch-based retrieval of a similar 3D model from a user sketch in a large 3D model database. Sketch-based modeling is then used to customize the model from the database. Customized 3D shapes are manufactured using a 3D printer.

2.5 Personalized Manufacturing and Intuitive Interfaces

The transition from handcrafted personalized manufacturing to intuitive and easily accessible mass customization is one of the central challenges for the digital manufacturing age. A successful implementation of this process depends on the development of high quality interfaces for end users. [Autodesk 2012] offers one of the first manufacturing pipelines that is designed for the needs of everyday users. We present an alternative approach that employs sketching and sketches to make the manufacturing pipeline accessible to non-expert users. Because of its ability to act as a common means of visual communication, sketches provide an exceptionally well suited input for searching large image and 3D shape databases [Eitz et al. 2012], for modeling 3D surfaces [Zimmermann et al. 2007] and for the fabrication of functional mechanical devices [Mueller et al. 2012]. Works by Saul et al. [2011] and Lin et al. [2010] investigate the aspects of intuitive sketching and design for furniture design. We combine the above mentioned insights and propose a sketch-based manufacturing pipeline.

Sketch-Based Pipeline for Mass Customization

We present a novel application workflow to physically produce personalized objects [Hildebrand and Alexa 2013]. With our prototype we show for the first time an end-to-end workflow from a user drawn 2D sketch to a 3D printed, personalized object that takes manufacturing limitations into account. This is achieved by sketch-based retrieval [Eitz et al. 2012] and modeling [Zimmermann et al. 2007], and it enables the user to control the process with an intuitive and consistent sketch-based input metaphor. The components of our approach include:

1. A simple 2D view of a 3D shape is sketched by the user and used as a query image for a large 3D model database. The retrieval system returns a set of matching 3D models.

2. The user selects one model of the retrieval set, and modifies it using simple strokes along the shape's silhouette.

3. The final modified object is printed using an off-the-shelf 3D printer.

Shape Retrieval The presented retrieval method is based on a preprocess over the 3D models in the database. Important features of the models are extracted to match the sketched input images with the 3D shapes. To this end, non-photorealistic rendering algorithms are employed to render the object with selected feature lines from several virtual viewpoints. This results in the creation of a set of line renderings for each model. An image descriptor based on Gabor filters, which is specifically designed to match the sketch-based user input and rendered images is then used to generate a bag-of-features representation [Eitz et al. 2012]. This enables to search the shape database quickly and accurately even if the database contains millions of 3D models and the sketch is drawn by a non-expert user.

Shape Modeling and Fabrication The sketch-based modeling process enables the user to modify the model by sketching new silhouettes of the mesh. For each stroke drawn by the user, a corresponding feature-preserving deformation is computed. This deformation, however, might introduce changes that cannot be manufactured with a 3D printer, such as,

- thin and fragile structures, which produce unstable results,
- self-intersections,
- meshes that exceed the boundaries of the available printing volume.

To avoid these production difficulties, a simulation of the additive manufacturing process is performed after each modeling step. For the simulation, a set of intersections contours with the deformed shape is generated. The contours are used as 3D printing proxy shapes to be tested against the above fabrication constraints. The deformation is reduced stepwise using linear interpolation until the constraints are satisfied. For details we refer to Hildebrand and Alexa [2013].

Chapter 3

Discretization and Abstraction

An abstraction is one thing that represents several real things equally well.

— Edsger W. Dijkstra

In this thesis, we introduce computational models for different digital fabrication applications. Each of these applications map a given input geometry onto one of the hardware devices discussed in Chapter 2. Although each of the output mappings has different properties and purposes, we can derive a set of common principles that provides an effective working model for various of the algorithms and applications proposed in subsequent chapters. Figure 3.1 shows the spectrum of fabrication techniques, resolutions and scales that are examined throughout the following chapters.

A significant aspect of the digital fabrication process is mapping the digital model to the output device. This requires to convert from the shape representation used in the computer to a shape approximation that can be fabricated. The precision of the

(a) (b) (c) (d) (e)

Figure 3.1: The Stanford Bunny model fabricated with different output mappings developed in this thesis. (a) A consumer level 3D printer is used to print the object. (b-c) Accuracy decreases with increasing layer thickness. Chapter 5 shows a decomposition scheme to increase accuracy even for layers with substantial thickness.(d) A shape assembled out of two parts created by a 3-axis CNC milling. The manufactured shape introduces fabrication errors that are minimized by the choice of a partitioning plane as presented in Chapter 6 (e) The shape is represented by a sparse set of layers that are fabricated as presented in Chapter 7.

output mapping depends on many aspects, e.g. on axes precision limits, the height of each layer that can be deposited (3D printing), the radius of the milling tool (CNC milling) or the accuracy pre-defined by user-settings. In each case, the limited precision of the output device leads to a discretization of the shape. It is achieved by sampling the input surface along a specific constant direction, e.g. an axis of the devices, a process known as *slicing* (layer-based discretization). The sampling rate is thereby typically pre-defined by the output capabilities and each sample can be reconstructed by a piecewise constant function as shown in Figure 3.3. Furthermore, depending on the fabrication process and the hardware device, *direction bias* may be introduced through varying fabrication precisions along the different manufacturing axes.

3.1 From Exact to Abstract Representations

The main objective of fabrication techniques is to reproduce a digital model as faithfully as possible. However, the complexity of the input data almost always exceeds the available capabilities of output devices. Therefore, an output mapping of the input to the capabilities of the device or user specification is necessary. The mapping usually involves an unavoidable loss in accuracy although there are also cases where this is desired. Many existing applications in computer vision, digital fabrication and computer graphics work with shape abstractions. For instance, non-photorealistic rendering creates abstract 2D image content by various approaches [Kyprianidis et al. 2013]. Also, in 3D computer graphics techniques for the abstraction of shape have been proposed. Some of the related works have been cited in Chapter 2. For instance, the approaches of McCrae et al. [2011], McCrae et al. [2013] and Mehra et al. [2009] provide interesting ideas to find simpler geometry to represent shapes for various applications, for example, computational shape recognition, modern art and the creation of toys, interior and architectural design.

To achieve a high degree of shape abstraction one often defines so-called proxy-geometry, i.e. simple primitives such as planar elements that represent the shape. The main challenge of this approach is to identify parts of the original shape that can be represented by the proxy-shape.

Figure 3.1 shows the scope of representations throughout this thesis. Figure 3.1(a) shows a consumer level 3D print with some geometric artifacts; (b-c) illustrate that accuracy decreases with increasing layer thickness. By subdividing the shape into parts one can approximate the geometry in different directions more accurately. We will discuss the approach in detail in Chapter 5. This is useful in particular for low resolution 3D printers or when the object exceeds the size of the production volume. Figure 3.1(d) presents a shape assembled out of two parts created by a 3-axis CNC milling. The manufactured shape introduces fabrication errors that are minimized by the choice of the partitioning as presented in Chapter 6. Figure 3.1(e) shows an entirely different approach that results in a shape abstraction. We show in Chapter 7 how to find directions and represent the original geometry by sparse and irregular sampling. The result is a cardboard abstraction.

In the following, we introduce notation and definitions that are used throughout this thesis. We also discuss the main output mapping methods that are used to approximate a given input shape.

3.2 Preliminaries and Notation

We define a triangle mesh M in $\mathbb{R}^3$ as a collection of triangles that defines a piecewise linear surface. From a topological point of, the triangle mesh is a graph structure with a set of vertices

$$\mathcal{V} = \{v_0, \cdots, v_{\mathcal{V}}\} \tag{3.1}$$

that are connected by a set of triangular faces

$$\mathcal{F} = \{f_0, \cdots, f_{\mathcal{F}}\}. \tag{3.2}$$

Each face can be represented by a set of edges

$$\mathcal{E} = \{e_0, \cdots, e_{\mathcal{E}}\} \tag{3.3}$$

connecting the vertices. From a geometric point of view, each vertex has a position in $\mathbb{R}^3$ so that each face $f \in \mathcal{F}$ is specified by its three vertex positions [Botsch et al. 2010].

Important throughout this thesis will be the intersection of a plane $\mathcal{H}$ with the mesh M. We define the plane $\mathcal{H}$ as

$$Ax + By + Cz + D = 0 \tag{3.4}$$

where $\mathbf{n} = \langle A, B, C \rangle$ is the normal vector of the plane and D the signed distance from the origin to the plane center h_0. We call the intersection of $\mathcal{H}$ with M a *planar section* or *slice* p. It is formed by the intersection vertices $v_{i0}, \ldots, v_{in}$ on the faces $\mathcal{F}$ of mesh M (see Figure 3.2). The vertices are connected by a number of planar *contour polygons* c_i on $\mathcal{H}$ that define the slice

$$p = M \cap \mathcal{H} = \{c_0, c_1, ...c_n\}. \tag{3.5}$$

A common principle for the applications in the remainder of the thesis is to find a set of planes $\mathcal{H} = \{\mathcal{H}_0, \ldots, \mathcal{H}_n\}$ such that the cross sections $\mathcal{P} = \{p_0, \ldots, p_n\}$ approximate M. The planes can be placed equidistantly along one direction or distributed sparsely with varying orientations depending on the application. We define a *layer* $\mathcal{L}_p$ as a solid object that results from the extrusion of p along its normal $\mathbf{n}$ by height h (as illustrated in Figure 3.2). We often refer to $\mathbf{n}$ also as the fabrication direction $\mathbf{d}$.

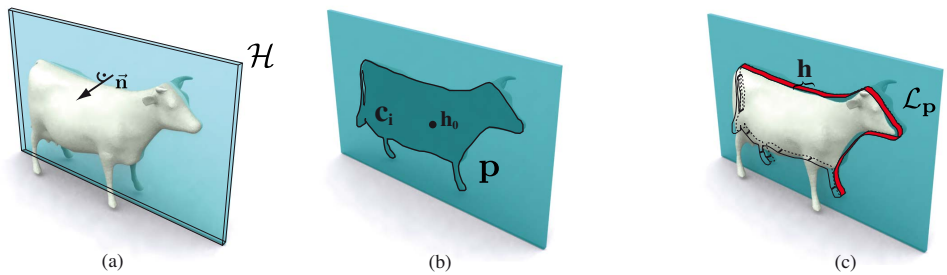


Figure 3.2: (a) A plane $\mathcal{H}$ with center point h_0 and normal $\mathbf{n}$ that intersects mesh M. (b) The intersection results in a slice p. Each slice consists of a set of contours c_i. (c) A layer $\mathcal{L}_p$ is defined by the extrusion of p in the direction of the plane normal $\mathbf{n}$ by height h.

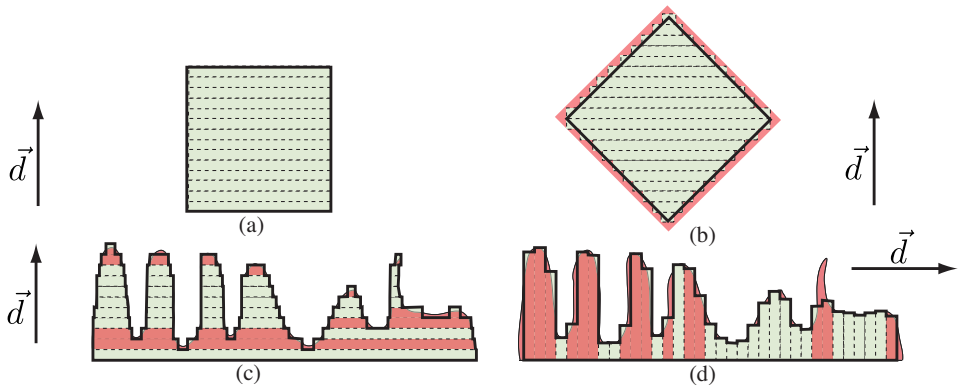


Figure 3.3: The effect of different sampling directions (columns) for two different surfaces (rows). In red we mark regions of the shape where high approximation errors occur. These regions depend on the sampling direction and the slope of the shape as seen in (a-b) and (c-d). Sampling artifact can also occur for very filigree parts of the input mesh as shown in (d).

3.3 Layer-based Discretization

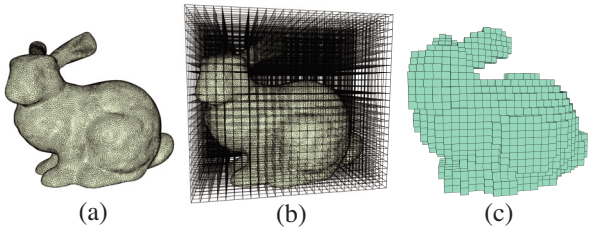
As discussed earlier, the output mapping leads to a discretization of the input mesh. Layered manufacturing techniques add layer to layer to fabricate a shape. Each layer is created by slicing, that is intersecting a plane at a specific height along the fabrication direction **d**. After slicing, the actual 3D printing hardware dependent mapping to the output devices takes place. One can decide between two approaches:

The 2D image-based approach rasterizes the contour elements into 2D bitmaps. Each pixel describes material properties at its position, e.g. the density or intensity. The bitmap is used to deposit a layer of fluid material (see Polyjet in Section 2.1). Note, that the 2D bitmaps can also be generated from a voxelization of the object.

Machine dependent contour paths are created from the contour polygon segments. These contour paths are translated to machine specific code, e.g. G-code. The code contains machine instructions for the printing head (see Section 2.1). The instructions control where to move to, how fast to move, and along what path and when to deposit or switch on the laser.

The physical fabrication direction is for todays 3D printing devices the up-direction inside the manufacturing volume. When we refer to a fabrication direction **d** it can potentially point in any direction. However, for the final fabrication step each part will be rotated to match the physical fabrication direction.

As discussed in the previous section, the plane intersection with the mesh faces results in a set of line segments. The orientation of each segment is thereby determined by the face normal of the face that was intersected and segments are connected to form contour polygons c_i which contain outer polygons and holes (see Figure 3.2). The layer height h is generally defined by the resolution of the additive manufacturing device (shown as the step function in Figure 3.3 that approximates the surface).



(a) (b) (c)

Figure 3.4: The distance to an input shape (a) is sampled at grid positions (b). Grid cells that are enclosed by the surface form a voxelized representation (c).

3.4 3D Grid-based Discretization

Most manufacturing processes work layer-based and hence define a natural layer height. A discretization into finer, discrete cells is however often helpful or even necessary to evaluate certain object properties. A 3D grid-based discretization rasterizes the volume enclosed by the surface in simple cubic elements, i.e. voxels, as shown in Figure 3.4. Therefore, one models the cubic cells with dimension $h \times h \times h$, where h is the lowest resolution of the process is h (e.g. layer height). The accuracy in the tangent directions is often better than h. To this end, we propose to sub-sample each voxel by a factor N and end up with a resolution of $h \times \frac{h}{N} \times \frac{h}{N}$. In order to voxelize the object consistently it has to be determined whether a voxel is inside or outside the object's boundaries. For this process one can consider different approaches:

The rules of 26-separating voxelization by Cohen-Or and Kaufman [1995] which results in a binary voxelization. In that case a voxel does not contain any further geometric information except its position.

An implicit geometric representation stores the signed distances to the input mesh in each voxel [Baerentzen and Aanaes 2005]. This additional information can be useful for manufacturing optimizations. The set of voxels that contains zero distance is called the *zero level set* and it describes the surface of the object. Moreover, additional geometry information can be inferred as needed, i.e. as gradient information.

Note that, the cubic elements naturally introduce three possible fabrication directions. Hence, we can evaluate errors and energy terms for the three orthogonal directions to decide which one best meets the application requirements. The voxel representation can also be used for 3D printing directly as proposed by Hiller and Lipson [2009] and Vidimče et al. [2013]. We demonstrate that grid-based discretization will help to improve the accuracy of the process in Chapter 5.

3.5 Approximation Errors

A significant advantage of additive compared to subtractive manufacturing is the possibility to manufacture general geometries. However, given the resolution of the manufacturing device and the physical size of the input object, approximation errors arise. To be more specific, the fixed resolution of the manufacturing device defines a layer thickness and thereby introduces staircase errors on any face that is not orthogonal to the fabrication direction $\mathbf{d}$, see Figure 3.3. Another problem is the user-specified physical size of the output object that, especially in very filigree areas, might introduce sampling artifacts or produce a mapping so that parts cannot be fabricated correctly. There are two possibilities to reduce these approximation errors.

1. We find a fabrication direction **d** that finds an improved alignment with the mesh to reduce the staircase effects as illustrated in Figure 3.3.

2. In regions where the fabrication process cannot correctly reproduce the shape we adjust the input geometry. This idea has not been exploited in this thesis, see however [Winkelmann 2014] for Fused Deposition Modeling.

As illustrated in Figure 3.3, the fabrication direction influences the accuracy of the output object. Figure 3.3 (a) and (b) show a square shape for different **d**, while (c) and (d) illustrate a surface including thin surface features. Figure 3.3 (a) shows a perfect orientation for the square shape along **d**. By rotating the square (or the fabrication direction) around 45 degrees the approximation error is maximized in (b). Also, the thin surface features in (d) cannot be represented very accurately or vanish completely. Sampling these regions along the orthogonal direction as shown in Figure 3.3(c) improves the accuracy. We go into more detail on how to improve the accuracy of the process in Chapter 5.

Chapter 4

Data Structures, Decomposition and Optimization

Divide each di culty into as many parts as is feasible and necessary to resolve it.

— René Descartes

We have seen in the last chapter that sampling, discretization and shape abstraction play important roles for digital fabrication. Another important aspect is the decomposition of an object into several parts. This is particularly motivated by two objectives. First, because of the limited production volumes of output devices, the size of objects that can be manufactured in one piece is limited. Therefore, large objects can only be fabricated when these are subdivided into smaller, producible parts. Second, a partitioning or segmentation can be useful to optimize the manufactured object with respect to factors such as rigidity, overall material usage and accuracy. The central question is hence how do we find a suitable decomposition of an object into parts? In the following chapters we will propose applications relying on such a decomposition of the input shape.

In the following, we will introduce broadly three main techniques that can be employed for the decomposition of an object with more details provided in the specific application chapters. First, we introduce principal component analysis which is a standard mathematical tool that can be used for the computation of fabrication directions, that is important to find suitable partitions. Second, we discuss binary space partitioning trees, a data structure that helps to organize the partitions in the mesh data. Third, we briefly talk about discrete optimization methods that come into play to find an optimal or near optimal partitions for an input object.

4.1 Principal Component Analysis

One standard mathematical tool that is used throughout this thesis is the Principal Component Analysis (PCA). We are specifically interested in using it to analyze geometric properties such as the distribution of vertex or face normals or vertex positions.

The goal of a PCA is to find an orthogonal transformation of a point cloud into a new coordinate system. The transformation is defined such that the first principal component (first coordinate axis) minimizes the error that arises under the projection of all data points onto that axis. The second principal component is chosen as the vector orthogonal to the first component that minimizes the projection error, and so forth. The number of principal components is equal to the dimension of the space over which the point cloud is defined, in our case typically $\mathbb{R}^3$. Note, that the first principal component is also the direction of greatest variance.

To perform PCA, we utilize the covariance matrix $Q = X^T X$ of the data matrix X

$$X = \begin{matrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{matrix} \tag{4.1}$$

where each row contains a data point x_i centered at the mean of the data set. The principal components w_i can now be found by an eigen-decomposition

$$Q = W \Lambda W^T \tag{4.2}$$

where Λ is the diagonal matrix of eigenvalues λ_i of $Q = X^T X$ and W is the orthogonal matrix whose columns are the eigenvectors w_i of $X^T X$. For our applications the first principal component corresponds often to the direction of minimum projection error of all vertex position or normals onto that axis, i.e. the fabrication direction $\mathbf{d}$. To give some intuition, finding this direction corresponds also to solving the maximization problem

$$\arg\max_{d=1} \quad Xd \quad ^2 \tag{4.3}$$

4.2 Decomposition and Binary Space Partitioning

Most layered or additive manufacturing processes work by viewing the digital input model as being monolithic. However, decomposing the model into a small number of pieces is helpful in many use cases. For this, the model is intersected with a suitable chosen plane separating the input object into two piece that are then manufactured separately and glued together. Repeating that intersection process recursively for each part calls for an organizing data structure, such as the *Binary Space Partitioning Tree* (BSP) which is already well known in computer graphics [Fuchs et al. 1980]. Many algorithms rely on it to store and organize geometric information, e.g. image synthesis, collision detection or physical simulation methods.

In this thesis we employ BSP tree representations over polygonal mesh data for various purposes. For us fast data access or storage efficiency are not of primary concern. Instead, our interest lies in a meaningful organization and partitioning of the object with respect to fabrication constraints. An approach similar to ours has recently been proposed by [Luo et al. 2012] who defined a linear combination of energy terms, e.g. based on aesthetics, structural soundness, symmetry and connector feasibility, to control the partitioning of an object with a binary space partitioning.

Given a triangular input mesh M, one constructs a binary space partitioning tree to organize its triangles. During the construction process a new tree node k is created by defining a plane $\mathcal{H}$ that intersects and thereby partitions M into two convex half-spaces,

i.e. a front- and backspace as shown in Figure 4.1. Each half-space and its triangles can then be accessed as child nodes of k in the tree, i.e. each node corresponds to a region of space. Triangles that lie on the plane $\mathcal{H}$ are also stored in the node k. The construction of the BSP Tree is usually done recursively until every subspace holds only one triangle.

To insert a new triangle into the BSP Tree one compares it against the plane at each node, propagates it to the right side and splits it if necessary. When the triangle reaches an empty cell, one creates a node with the triangles supporting plane. To access the triangles in the BSP Tree one needs to specify a query point and perform a tree traversal which is application specific. Basically, it begins at the root node and classifies the query point with respect to its partition plane and subtree. This procedure is repeated recursively for each subtree.

Important for our decomposition concepts is that the triangles contained in one half-space form a part of our object. An example for the choice of $\mathcal{H}$ in our applications is the previously discussed PCA over the face normals of M.

The usage of partitioning planes in the binary space partitioning results in a planar cut through the mesh data. In our digital fabrication context this has three advantages. First, the planar cuts fit naturally in the fabrication concept, because parts can squared up in the production volume. Second, the assembly of different parts is easy as the contact surface is planar. Third, the computation of a suitable partitioning plane can be done very efficiently.

The choice of the partition plane is highly application specific. One objective is to obtain a balanced tree, which is necessary to ensure performance of spatial classification tasks where it is necessary to store an equal number of primitives in each half-space. The construction of an optimally balanced tree is an NP-complete problem. For most common computer graphics applications the trade-off is between finding the *tree balancing* and *splitting minimization*. However, each partition plane may introduce new primitives because of splitting intersected polygons. These new primitives have also to be inserted and stored. Hence, minimizing the number of splits of primitives is also desirable for some application.

In general, one can distinguish three different strategies to choose splitting planes as shown in Figure 4.1.

1. For most graphics purposes it is appropriate to choose the partition plane from the input set of polygons (called an auto-partition). During the construction a triangle is selected. Its plane defines the splitting plane for the next subspace. The primitive is then stored in the node.

2. Some applications require an axis-aligned splitting of the underlying data as seen in Figure 4.1 (2).

3. The partition planes are defined by evaluating some energy function over the data shown in Figure 4.1 (3). The orientation of the planes is then typically arbitrary.

Our fabrication applications make use of different splitting strategies. We specifically use orthogonal splitting planes based on the evaluation of splitting errors (Chapter 5). We employ the distribution of vertex normals and vertex positions over the mesh as an important criteria for choosing a splitting plane (Chapter 6, although we do not construct a full BSP here). Additionally, we propose modified BSP tree construction rules to detect and avoid collisions during the simulated construction of cardboard models (Chapter 7).

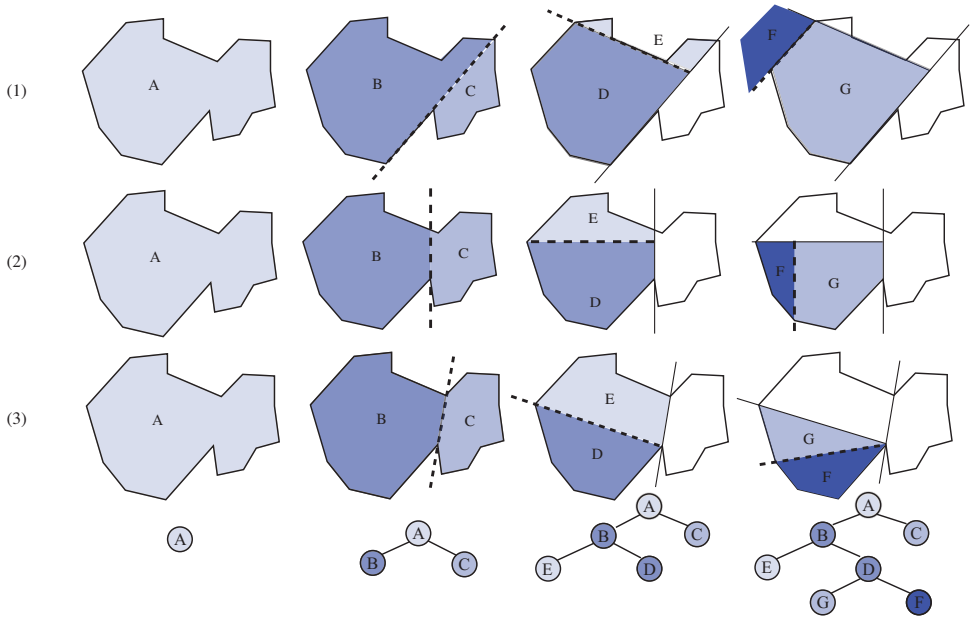


Figure 4.1: The binary space partitioning algorithm illustrated for a simple 2D scene. (1) auto partitioning; (2) partitioning planes are defined orthogonally to each other; (3) splitting planes are defined arbitrary according to some pre-defined energy function.

4.3 Discrete Optimization

Discrete optimization methods have a large number of applications. We are specifically interested in finding an optimal configuration out of a large set of possibilities. These possibilities arise either from decision or labeling problems. Our class of problems are known to be NP-hard optimization problems. That means, finding an optimal solution to the problem can take exponential time. We hence employ standard optimization strategies to solve this class of problems. Chapter 5 compares different labeling approaches using multi-label graph cut [Boykov et al. 2001] and branch-and-bound optimization [Land and Doig 1960]. Chapter 7 again uses branch-and-bound optimization to find an optimal or near optimal solution to an insertion ordering problem. Since these two algorithms are standard methods in computer science we only briefly summarize them conceptually with a simple example. We go into detail when we use the algorithm in the later chapters.

Branch and bound

In the following, we introduce the branch-and-bound method with the help of the well known Knapsack problem. The objective of the Knapsack problem is to find an optimal set of valuable items that fit into a bag with limited space capacity C. Each item has a value v_i and occupies some space s_i in the bag. The set of items in the bag with value v_i defines our objective function F, subject to a capacity constraint. We are hence interested in the following optimization problem:

$$F(x) = \arg\max \sum_{i \in I} v_i x_i \tag{4.4}$$

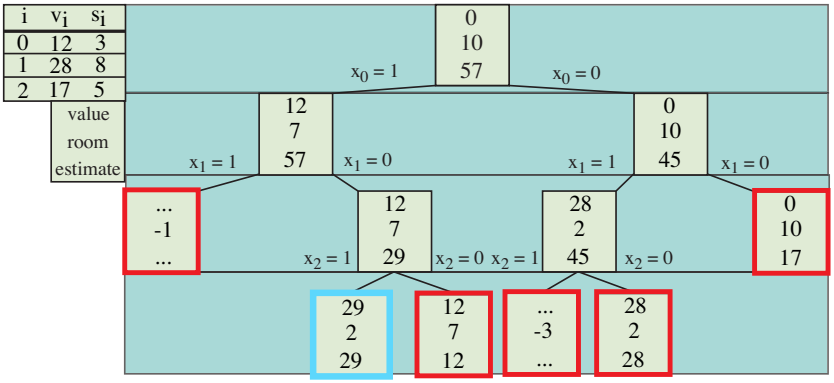


Figure 4.2: Illustration of the Knapsack problem decision tree. We show the optimal solution in blue and pruned subtrees and nodes in red. The capacity of $C = 10$.

subject to the constraints

$$\sum_{i \in I} s_i x_i \leq C, \quad x_i \in \{0, 1\} \tag{4.5}$$

where x_i determines if the i-th item is in the Knapsack or not. Since the Knapsack problem is NP-hard it becomes intractable for already a small number of items i because the set of possible combinations grows exponentially ($2^{|i|}$). Therefore, one tries to find a high quality solution instead. This can be illustrated with the binary decision tree shown in Figure 4.2. For each node in the tree one can ask the question: 'Do I take an item? Or do I not take an item?'. Each decision results in a new node in the tree and less space and more value inside the bag. The figure shows for each node the value of the bag, the room capacity left and the best solution that is still possible.

The basic goal of branch-and-bound is to avoid computations of as many suboptimal configurations as possible. That means, we prune away parts of the decision tree that will not contain an optimal solution. Therefore one iterates two step:

1. A branching step splits the problem into a number of subproblems and tries to explore the whole sub-tree of possibilities. So implicitly it will explore the whole tree. But there is the next step.

2. Bounding is about finding an optimistic estimate of the best solution of the subproblem. It evaluates nodes in the decision tree and prunes away entire subtrees that will never contain an optimal solution. In case of an maximization problem we call this estimate an *upper bound*, in case of a minimization problem a *lower bound*. All nodes in subtrees that are below that upper bound (lower bound respectively) do not need to be evaluated in the branching step.

The example shown in Figure 4.2 starts to evaluate the tree by taking the first item $x_0 = 1$ (left branch). We can ignore parts of the tree that exceed the capacity and eventually find a solution. Evaluating the right subtree leads to suboptimal subtrees where the best possible solution cannot get better than the solution we already found. We do not have to evaluate that subtree and hence prune it away. Figure 4.2 show the optimal solution in blue and invalid or suboptimal subtrees in red.

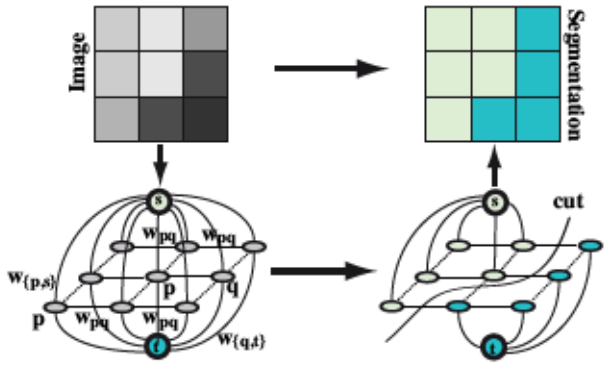


Figure 4.3: An input image is interpreted as a directed capacity graph with the terminals s and t. Each pixel has an edge to its neighbors weighted by the smoothness cost w_{pq}. Each pixel also connects to all terminals weighted by the data cost $w_{(pq\ st)}$. The energy in Equation 4.6 is solved using graph cut.

Depending on the application and optimization constraints different decision tree traversal strategies can be used, e.g. depth-first, breadth first or k-best traversals, where k means the number of the most valuable subsolutions that will be taken into account.

Multi-label graph cut

Another way of decomposing an object into a small number of pieces is by finding a segmentation of the given object. Such segmentation problems are often modeled as labeling problems [Kolmogorov and Zabih 2004]. One then wants to find a set of possibly large segments where all elements contained in one segment share the same label. Segmentation problems are standard problems in computer vision. In this case basic the elements are the pixels of an image. Here we use this example to introduce segmentation, see Figure 4.3.

The input is a set of pixels $\mathcal{P}$ encoding different intensities as grayscale image values and a set of two possible labels $\mathcal{L} = l_0\ l_1$. The goal for our example is to identify segments that share similar pixel intensity values. Labeling problems are formulated in terms of energy minimization. We hence minimize the following energy function:

$$E(x) = \sum_{p\ \mathcal{P}} \Phi_p(x_p) + \sum_{(p\ q)\ \mathcal{N}} \Theta_{(p\ q)}(x_p\ x_q) \tag{4.6}$$

The data term $\Phi_p(x_p)$ is a function of the pixel intensities that measure the cost of assigning a label x_p to pixel p. The smoothness term $\Theta_{(p\ q)}(x_p\ x_q)$ describes the cost of assigning the labels x_p and x_q to the neighboring pixels p and q in the pixel neighborhood $\mathcal{N}$. As the name suggests $\Theta_{(p\ q)}(x_p\ x_q)$ controls the smoothness of the labeling, i.e. the same label is assigned to elements which vary smoothly and different labels to elements at sharp discontinuities in the image. That means high gradients between image pixel intensities should lead to different labels between these neighboring pixels. In other words, neighboring elements tend to take the same label and have a geometric or location consistency [Kolmogorov and Zabih 2004].

Energy functions like $E(x)$ can be minimized efficiently using graph cut. For this, the relationship between elements is interpreted as a weighted graph. The graph also

contains two additional nodes that are connected to every node, i.e. a source s and sink t representing the two different labels, see Figure 4.3. The data term defines the edge weights from a terminal to a node and the smoothness term defines edges weights of neighboring pixels.

Once the graph is constructed, a segmentation is achieved by obtaining the minimum cut through the edges of the graph defining the boundaries between segments. In other words, we find a partition of the vertices in the graph into the disjoint sets S and T with $s \quad S$ and $t \quad T$ as shown in the bottom right of Figure 4.3. Finding the min cut through a graph is the dual problem to computing the maximum flow in that graph [Ford and Fulkerson 1956]. The maximum flow can be computed in polynomial time with small constant using algorithms shown by Goldberg and Tarjan [1988]. Boykov et al. [2001] and Boykov and Kolmogorov [2004] propose an extension to find approximate solutions for multi-label graph cut problems running in nearly linear time in practice. The resulting graph cut finds a binary labeling (i.e., two labels, so two segments) shown in Figure 4.3. Extensions to more then two labels are discussed by Boykov et al. [2001] who show a method to minimize an energy function for many labels by repeatedly minimizing an energy function for two labels.

Chapter 5

Orthogonal Slicing for Additive Manufacturing

Fast is fine, but accuracy is everything.

— Xenophon

In this chapter we introduce an optimization method for additive manufacturing that aims at an accurate shape reproduction. Our optimization is particularly interesting for low-resolution and large scale 3D printing methods. All additive manufacturing technologies work by layering, i.e. discretizing the shape into layers and manufacturing each layer independently. As discussed in Chapter 3, this introduces an anisotropy into the production process that typically exhibits itself as different precisions in the tangential and normal directions. Especially at low resolutions this results in sampling artifacts and staircase effects.

We try to minimize these staircase effects by decomposing a shape into a small number of pieces so that each piece can be consistently sliced with small geometric error and that by assembling the pieces one gets a replica with overall small error (see

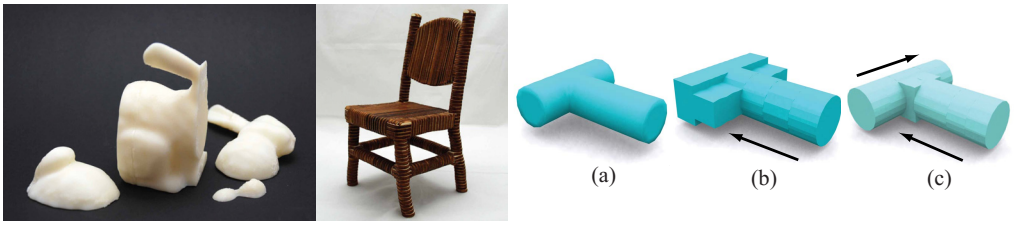


Figure 5.1: Left: We present a framework that improves additive manufacturing methods across different scales: 3D printing resolutions (left) medium to large scales that might exceed the machine manufacturing volume (right). Right: (a) T-shaped object. (b) sliced in one direction (c) decomposed into two partitions after optimization and sliced in two directions.

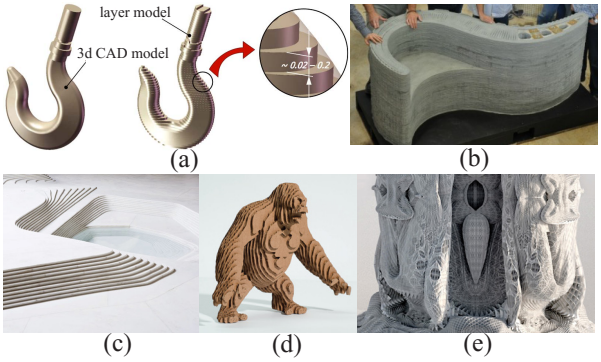


Figure 5.2: (a) Staircase effects created by an additive manufacturing process [Danjou and Köhler 2009] (b) Large scale cement 3D printing. (c) Architecture landscape models. (d) Fabrication of laser-cut stacked layered models ©Autodesk. (e) art sculpture "Subdivided Columns" consisting of several thousand slices by Michael Hansmeyer ©Michael Hansmeyer.

Figure 5.1). We formulate this objective as an optimization problem. The corresponding optimization thereby needs to avoid both extremes: only one direction is not flexible enough, creates a large error; while decomposition into many pieces can clearly make the error small, but the assembly or production becomes tedious or virtually impossible.

To describe the different precisions of the production process along different axes numerically we model it using anisotropic cubic elements. Our approach then determines a compromise between using an optimal manufacturing direction for each individual cubic element and one direction for the whole shape as shown in Figure 5.1(Right). We begin by computing an orthonormal basis and consider only the three basis vectors as slice normals (i.e. fabrication directions). Then we optimize a decomposition of the shape with respect to the basis so that each part can be consistently sliced along one of the basis vectors. While our approach can be generalized to all layered manufacturing methods from 2D slabs laser cutting over Layered Object Manufacturing to high resolution 3D prints, we wish to stress that the improvements one can obtain from slicing an object into different directions may depend on the resolution of the process, the size of the object, and the desired application. The potential benefits of our method for various applications are:

1. Staircase effects for 3D printing are reduced on average over the complete shape. This also has been focus of Danjou and Köhler [2009] and Nezhad et al. [2009], but does not work for general meshes (see Figure 5.2(a)).

2. Large scale 3D printing with low resolutions that become increasingly interesting for mechanical and civil engineering benefit with increasing precision (Figure 5.2(b)), see [Monolite Ltd. 2012] and [Khoshnevis et al. 2006].

3. Our method generates a partitioning of large objects that cannot be fabricated as a whole because they do not fit the production volume. Additionally, each part is fabricated to highest precision.

4. Multi-material objects often cannot be printed in one run because of printer limitations. Also partitioning is necessary in that case. Also here, parts can be fabricated to highest precision.

5. There is an increasing popularity for personalized 3D toys, puzzles and designs. We show an alternative approach to applications first introduced by Autodesk [2012] shown in Figure 5.2(c,d,e).

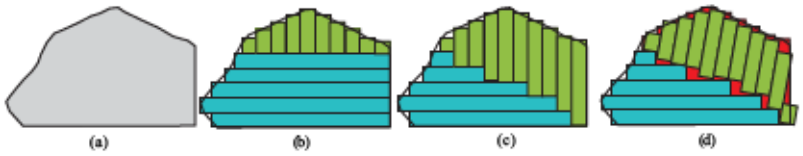


Figure 5.3: Composition of parts should be orthogonal to provide seamless assembly even for low resolutions. (a) shows an original 2D shape, (b) and (c) illustrate two possible orthogonal cuts (details in Section 5.4). (d) shows that non-orthogonal cuts can lead to non-planar boundaries. Especially for low resolutions this leads to assembly issues between these parts.

Preliminaries

Allowing only three orthogonal directions as fabrication directions is the result from early experiments. We found that with increasing layer thickness (e.g. $> 0\,5$ mm) an object printed of parts using non-orthogonal printing directions would result in a 'jaggy surface' along their aliased direction as shown in Figure 5.3. The assembly would then be difficult, often resulting in connections that cannot be glued together properly. While orthogonality could be achieved locally for some cuts we suggest to solve this problem globally by allowing only orthogonal printing directions Figure 5.3.

Consequently, our first modeling decision for this work is to *restrict the slicing directions* as well as the normals of the cutting planes to an *orthogonal basis* $B = [\mathbf{b}_0\mathbf{b}_1\mathbf{b}_2]$ $B^TB = I$. This approach allows to select an optimal slicing direction $\mathbf{b}_i$ for each part individually while guaranteeing planar connection areas without sampling artifacts between parts (see Section 5.2). In other words, because all cuts are orthogonal the model can be easily assembled.

We use an internal voxel representation with dimension $h \times \frac{h}{N} \times \frac{h}{N}$ as described in Chapter 3, to model the anisotropy in accuracy, i.e. the thickness of a layer is h, while the accuracy in the tangent directions is N times better than the thickness of a layer. The most natural choice for a smallest element for slicing direction $\mathbf{d}$ is a voxel cell of size h^3. Our idea is to pre-process the shape and find for each voxel its optimal slicing direction and a corresponding contour (see Section 5.3). Only for voxels containing parts of the shape's boundary the error will depend on the fabrication direction. Therefore, we will consider in the following only those voxels. Our objective is to optimize a partition of the voxel set along the voxel sides. The goal is to generate large sets of voxels that are processed along the same direction. To be able to improve the surface accuracy and to measure errors that result from slicing the voxel we compute three discrete volumetric differences (see Equation 5.14) between the input shape and the approximations for a certain direction:

We call the sum of smallest errors per voxel $e_{\min}(\Omega)$:

$$e_{\min}(\Omega) = \sum_{v \in \Omega} \min_{\mathbf{b}_i} e_{\mathbf{b}_i}(v) \tag{5.1}$$

with the three different slicing directions $\mathbf{b}_i$ over the complete set of voxels Ω. It models the least possible error over the whole mesh when we allow each voxel to have a slicing direction independent of its neighbors. We could achieve this result by fabricating each voxel individually along the direction with smallest error.

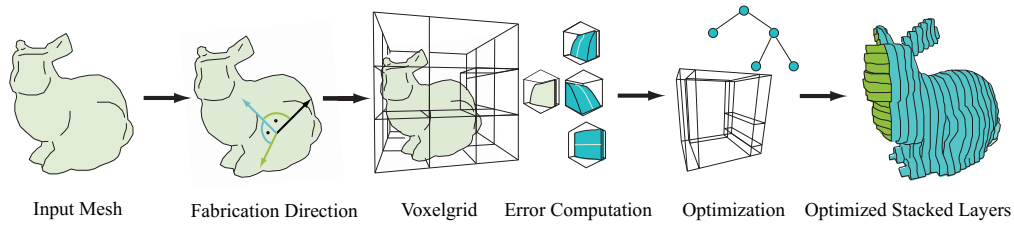


Figure 5.4: Overview of our method. We start with the input mesh and compute a set of orthogonal manufacturing directions. Afterwards we use these directions to perform a voxelization process where we divide our data into volumetric elements with the size h. For each voxel we compute slicing errors for each of the three directions. We then employ an optimization process to partition the voxel grid. We search for large segments with minimal slicing error while balancing the number of partitions. Our computed segmentation can be cut with a laser cutter or printed with a 3D printer.

We call $e_{slc}(\Omega)$ the minimal error resulting from choosing a consistent slicing direction out of the frame B. We compute $e_{slc}(\Omega)$ by adding the errors of each voxel $e_{\mathbf{b_i}}(v)$ for the three directions and then taking the minimum over the three possible directions:

$$e_{slc}(\Omega) = \min_{\mathbf{b_i}} \sum_{v \in \Omega} e_{\mathbf{b_i}}(v) \tag{5.2}$$

Note, $e_{min}(\Omega)$ and $e_{slc}(\Omega)$ define useful bounds for our optimization process. $e_{min}(\Omega)$ defines a lower bound, i.e. for the frame B we cannot get a partitioning with less error. $e_{slc}(\Omega)$ is a pessimistic upper bound for B, i.e. any reasonable partitioning should result in a lower error. Consequently, $e_{opt}(\Omega)$ minimizes the surface error by decomposing the object into smaller parts and allowing each part to have its individual slicing direction (see Section 5.4). We improve accuracy with an increasing number of parts. So, the two central questions we need to solve are thus.

1. Does the orthogonal basis B matter for the accuracy and how do we find an optimal one?

2. What is a meaningful partitioning the mesh that provides a balance between the number of parts and accuracy?

We address these questions in a computational pipeline that leads from an input mesh to an object that can be manufactured.

5.1 Overview

Let us summarize our pipeline to generate partitions that are sliced along directions that improve the overall accuracy of the input shape. We illustrate that in Figure 5.4.

1. Given a triangle mesh M, we compute a set of orthogonal directions $B = [\mathbf{b}_0\mathbf{b}_1\mathbf{b}_2]$ that are likely suited for a decomposition of the shape into small parts, each of which can be sliced along one of the three directions with small error (see Section 5.2). By rotation of the model with B^T we can now consider the canonical directions, i.e. x, y, z.

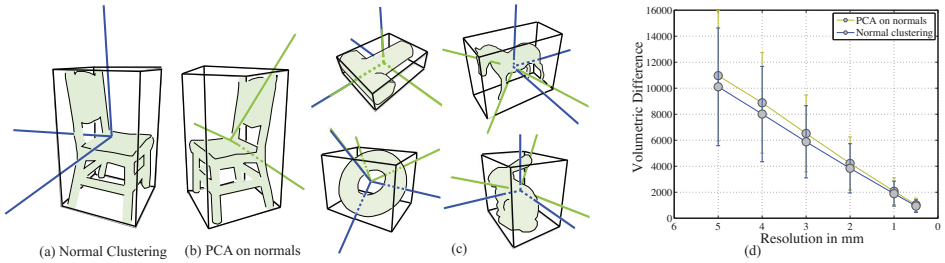


Figure 5.5: (a) We show the resulting manufacturing directions of the normal clustering (blue) (b) compared to a PCA over the set of normals (green). (c) We show different examples. Note, our method finds an optimal solution for the cylindrical T-shape and that the results of the clustering methods often correspond to the natural upright direction. (d) We evaluated the overall best approximation error over a set of test shapes. We show the mean minimal error and the standard deviation.

2. The shape is decomposed into voxels of size h^3, where h is the desired layer thickness (or, worst accuracy). Each voxel is then decomposed into N^3 sub-voxels, where N is the factor between the thickness of the slice and the accuracy in the tangent directions. We also use the sub-voxels to optimize the boundary of a voxel.

3. For each voxel, the errors for each of the three slicing directions are computed (see Equation 5.1 and Equation 5.2). We use the discrete volumetric difference between the input shape and each of three approximations for a certain direction, computed on the sub-voxel grid. This requires computing approximations that are constant in the direction normal to a slice, yet may vary in tangent direction with the sub-voxel resolution. We explain how to do this consistently for all voxels, yet using only the local information available in each voxel, in Section 5.3.

4. Based on the per-voxel errors, we compute a decomposition of the voxel grid so that each part can be sliced consistently with small error, yet the total number of pieces remains small. We also consider other factors in this process, such as the maximum size of each part. This process is explained in Section Section 5.4.

The result is a decomposition of the shape into a few pieces, as well as a slicing direction for each piece.

5.2 Selection of Manufacturing Directions

As discussed earlier, to improve the accuracy of the additive fabrication process we need to consider the rotation of the object inside the production volume. The fabrication direction $\mathbf{d}$, i.e. major direction in which the layers are placed, usually has the lowest resolution. We base our computations on a simple observation: a planar surface with normal direction $\mathbf{n}$ should be sliced in a direction *orthogonal* to $\mathbf{n}$ because the accuracy tangential to a slice is assumed to be significantly higher than normal to a slice. This leads to less approximation errors as seen in Figure 3.3. As explained with Figure 5.3 we require three orthogonal directions so that we can guarantee a seamless assembly of parts. In the following we will discuss three possible solutions to find an optimal orthogonal basis B.

Exhaustive Search

To evaluate the quality of three manufacturing directions B we voxelize the rotated object and compute the error e_{min}. The voxelization process is computationally expensive. An exhaustive search of the complete parameter space is therefore in general not feasible. However, to judge other optimization solutions we propose a ground truth B by searching over 10000 frames. For this, we sample 1000 uniform directions over the half-sphere and rotate around each direction 10 times. Table 5.1 shows the errors in the left most column. At the same time we evaluate the worst-case rotation (see right-most column) to show the spectrum of possible surface errors.

Main Normal Density Basis

A simple approach to find the main normal density basis is to use PCA over the face normals. Specifically, we define

$$C = \sum_{f \in M} a_i \mathbf{n_i} \mathbf{n_i}^T \tag{5.3}$$

be the covariance matrix of the faces of the mesh M. The matrix is a symmetric 3×3 matrix and so has real eigenvalues and orthogonal eigenvectors. We compute the eigenvectors and corresponding eigenvalues by the eigen-decomposition as discussed in Chapter 4

$$C = W \Lambda W^T \tag{5.4}$$

where Λ is the diagonal matrix of eigenvalues of C and W is an orthonormal basis B with the major eigenvector pointing in the direction the maximizes the scalar product over all area weighted face normals $\mathbf{n}$ of the mesh, i.e. it is the direction of the highest normal variance. The minor eigenvector points in the orthogonal direction of the lowest normal variance. The latter defines a direction $\mathbf{d}$ that introduces the least approximation errors as we want to slice orthogonal to $\mathbf{n}$. Umetani and Schmidt [2013] use a similar approach for computing a 3D printing direction but focus on the stability of an object.

Normal Clustering Basis

We propose another similar method for computing a set of orthogonal directions. Each triangle in the mesh corresponds to a planar piece with area a_i and normal $\mathbf{n_i}$. Our goal is to find three orthogonal directions $\mathbf{b_0}$, $\mathbf{b_1}$, $\mathbf{b_2}$. These can be represented conveniently as

$$B = [\mathbf{b_0} \mathbf{b_1} \mathbf{b_2}] \quad B^T B = I \tag{5.5}$$

such that the tangent space $T\mathbf{n_i} = (0\ 0)^T$ can be well approximated by two of the $\mathbf{b_c}$. Assume these are $\mathbf{b_0}$ and $\mathbf{b_1}$. Then, because B is orthogonal, the normal $\mathbf{n_i}$ is well approximated by $\mathbf{b_2}$. This means, it suffices to find an orthogonal B such that all normals are well approximated by one of the $\mathbf{b_c}$.

Note, it is not sufficient to perform a PCA over the set of the normals as done in the previous subsection. While this gives us one good direction to approximate all normals, it also gives us two more orthogonal directions that are not particular well suited for approximation of normals in the mesh. In other words, the solution is not well balanced and strongly biased in the average normal direction.

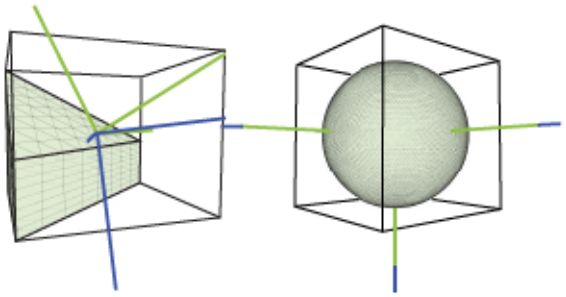


Figure 5.6: Left: A truncated prism shows that neither PCA directions (green) nor clustered normal directions (blue) are intuitive Right: Directions over the sphere.

Our approach is based on clustering of the normals, with the additional requirement of the cluster centers being orthogonal. We start with $B = I$ and then iteratively improve the current solution B as follows:

1. For each normal $\mathbf{n}$ find the closest direction $\mathbf{b_c}$ by maximizing $\mathbf{b_c n_i}$. This assigns each normal to one of the three directions and, thus, forms three sets of normal vectors.

2. For each cluster, we perform, similarly to the previous approach, PCA over the normal vectors *in this cluster* to find the directional center of the cluster. Specifically, let

$$N_c = \sum_{\mathbf{b_c n_i} > \mathbf{b_{c'} n_i} \;\; c' = c} a_i \mathbf{n_i} \mathbf{n_i}^T \tag{5.6}$$

   be the covariance matrix of the cluster c and a_i being the area of face with normal $\mathbf{n}$. The matrix is symmetric and so has real eigenvalues and orthogonal eigenvectors. We also compute the eigen-decomposition

$$E_c^T N_c E_c = \Lambda_c \tag{5.7}$$

   and use the eigenvector corresponding to the *largest* eigenvalue as the new cluster representative b_c.

3. The three vectors $B = (b_0 \; b_1 \; b_2)$ are generally not orthogonal. We use the SVD to compute the closest orthogonal matrix B from B:

$$B = UV^T \quad \text{where } B = U\Sigma V^T \tag{5.8}$$

4. We start over with the updated matrix B and repeat until convergence.

Figure 5.5 shows some results of our method. On the left we illustrate an example of the chair model and the resulting orthogonal basis (a) and the PCA over the set of normals (b). Note that the results of the clustering methods often correspond to the natural upright direction where the PCA averages the normals globally resulting in inefficient fabrication directions. Figure 5.6 shows that models that do not have a distinguished direction that would result in no or small fabrication error are still meaningful in the sense of minimizing the overall slicing error.

Since the differences between the three approaches to determine the directions are hard to judge in general, we illustrate the angle differences of the basis onto a circle and plot the scalar product with respect to the optimal B determined by exhaustive search in Figure 5.7.

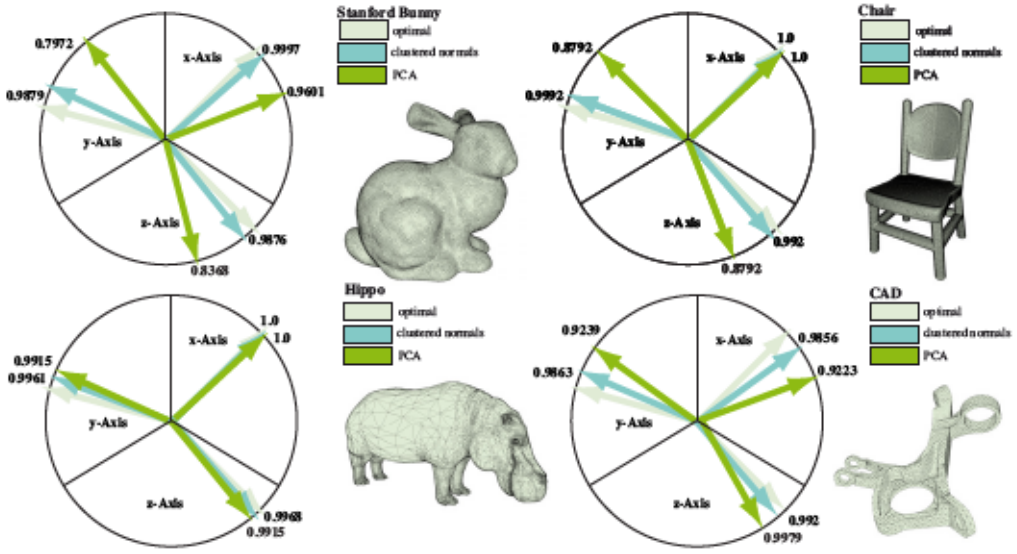


Figure 5.7: Evaluation of the optimal frames $\mathbf{b_0}$ $\mathbf{b_1}$ $\mathbf{b_2}$ determined with three different methods for a set of example shapes. Shown are the scalar products per axis to the optimal solution and an illustration that shows the differences of the axes of the frames laid out on a circle. This is done for four reference models. One can clearly see that the normal clustering basis lies very closely to the measured optimum. The PCA method varies significantly.

We also evaluated errors for all models as presented in Table 5.1. The results demonstrate that the overall accuracy can be improved and that the choice of the basis is a key for improvement. We evaluate the three direction methods and give a worst-case analysis. We show the corresponding e_{min} and e_{slc} in percentage to the volume over the voxels containing part of the boundary, i.e. the approximation errors normalized by the volume of all voxels of the zero level set. While the measured optimum shows the minimum error e_{min} for some of the case, our proposed normal clustering is often better or very close to the optimal solution. Note, due to under-sampling of directions the optimal case might not always be the true optimum. It can be in between non-optimal sampling points over the half-sphere. As expected, a single most accurate fabrication direction e_{slc} can also be found be using PCA over the face normals.

5.3 Boundary Voxel Optimization

Additionally to the selection of a suitable basis we consider optimizing the accuracy of the process by optimizing the boundary of the voxel, i.e. contour inside a single voxel. In the following, we explain the case of slices in the $x-z$ plane, and the y direction is normal to the slice. The other two directions can be computed similarly.

As a first step, we subdivide the voxel into N^3 sub-voxels. This follows from the resolution being N times higher in the tangent directions, which are a-priori unknown. For each sub-voxel we compute the signed distance to the original surface. Let the center of a sub-voxel be $\mathbf{s}_{ijk}$ and the corresponding closest point on the surface be $\mathbf{x}(\mathbf{s}_{ijk})$ with

Table 5.1: We evaluate the fabrication direction errors for several 3D model and direction methods. We compare the e_{min} and e_{slc} for the optimal orthonormal basis, the normal clustering, the PCA based method and the worst case basis and single direction. Each model has a size of 100mm and is sliced using a layer height of 2mm. We mark the minimal error in bold.

	Exhaustive %		n Clustering %		n PCA %		Worst Case %	
	e_{min}	e_{slc}	e_{min}	e_{slc}	e_{min}	e_{slc}	e_{min}	e_{slc}
Stanford Bunny	2.61	6.2	**2.59**	6.16	2.9	**6.13**	3.7	8.2
Chair	**0.99**	5.0	1.0	5.1	2.27	**4.97**	4.85	8.16
CAD model	**1.79**	6.8	1.92	6.6	2.5	**6.38**	4.7	8.36
Hippo	2.28	4.66	**2.24**	**4.51**	2.28	4.55	3.96	10.1
Kitten	**2.85**	6.62	2.89	**6.6**	2.94	6.62	3.42	7.84
Horse	**2.55**	6.1	2.65	6.04	3.02	**5.91**	3.56	8.67
Three Holes	**1.79**	5.55	1.79	5.55	1.79	**5.55**	4.0	8.13
Lion	2.94	5.8	**2.9**	**5.55**	2.96	5.6	3.88	8.88
Torus	2.8	6.0	**2.77**	**5.94**	2.86	6.1	3.66	10.6
Knot	**2.67**	6.94	2.76	7.0	3.22	7.1	3.4	8.25
Three Cylinders	**1.33**	6.5	2.8	6.67	2.6	**6.5**	3.8	8.8
Two Cylinders	**0.15**	4.6	0.15	4.6	0.15	**4.6**	3.9	10.2

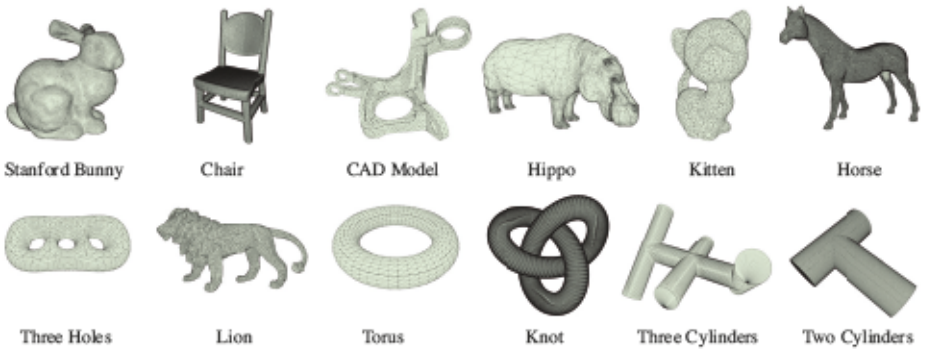


Figure 5.8: These are the reference 3D models we are using in the above table and throughout this thesis.

surface normal $\mathbf{n}(\mathbf{s}_{ijk})$. The signed distance is given by

$$d_{ijk} = sgn\left(\mathbf{n}(\mathbf{s}_{ijk}) \quad (\mathbf{x}(\mathbf{s}_{ijk}) - \mathbf{s}_{ijk}))\right) \quad \mathbf{s}_{ijk} - \mathbf{x}(\mathbf{s}_{ijk}) \tag{5.9}$$

Our objective is now to compute a new distance function for the $x - z$ plane approximating all distance values in the sub-voxels. Note that this process computes new values for each sub-voxel, so also for the corners, edges, and faces of the voxels. These elements are shared with neighboring voxels. The sign of the value has important topological consequences, namely if a point in space is inside or outside the shape. For topological consistency of the result it is necessary that the signs are identical for shared elements. The only local way to ensure this is to use the same values as input for each resulting value. This means, when we compute a certain value on the $x - z$ plane we can only use the varying y values in this column and no other sub-voxel in the current

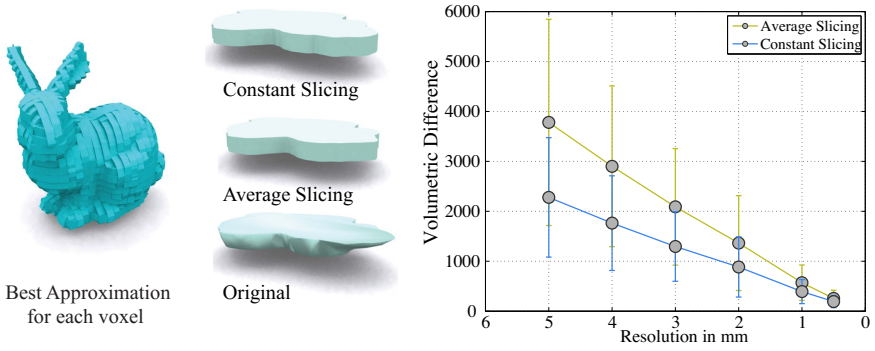


Figure 5.9: Left: Partitioning the model down to voxel-level would result in the minimal possible error for the manufacturing resolution. Middle: A single slice and two approximations that are constant along the normal to the slice. The first approach intersects the geometry at the center plane of the voxel. We suggest to rather compute average distance values in normal direction of the slice and then extracting the contour as the zero-set of the distance field. Right: the volumetric difference to the original shape as a function of voxel size.

voxel. We hence compute a new distance function

$$\tilde{d}_{ik} = f(d_{i1k}, \ldots, d_{iNk}) \tag{5.10}$$

where we still have freedom in our choice of f. Figure 5.11(Left) and Figure 5.10 illustrates how f is evaluated over the distance samples along the y-axis. This defines a new distance field in the x-z plane that is used to extract the final contour over the cell as the zero-set of the field.

The simplest choice would be to pick out a certain value from the column, i.e.

$$f(\gamma_1, \ldots, \gamma_N) = \gamma_{N/2} \tag{5.11}$$

This is equivalent to intersecting the original geometry with a plane at the height the center of the voxel as illustrated in Figure 5.10(a) and Figure 5.9(Middle: constant slicing). We suggest to rather compute a least squares solution for each column, which amounts to taking the average distance value (see Figure 5.10) and Figure 5.9(Middle: average slicing):

$$f(\gamma_1, \ldots, \gamma_N) = N^{-1} \sum_j \gamma_j \tag{5.12}$$

One important issue can arise for low resolutions or very thin input object structures. As illustrated in Figure 5.10(a,b) layers might be disconnected leading to topological and structural problems in the fabrication process. We propose to use the max function () for all y values. This guarantees topological correct output but significantly reduces the accuracy (see Figure 5.10(c)):

$$f(\gamma_1, \ldots, \gamma_N) = \max_N \gamma_j \tag{5.13}$$

As we show in Figure 5.9 optimizing the voxel boundary leads to significantly smaller volume differences even in high resolutions. The new distance field over the $x - z$ slice

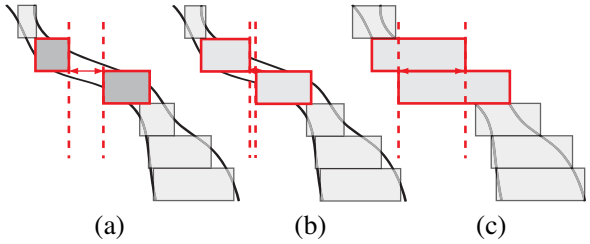


Figure 5.10: We show a sectional view of a thin structure in the background and the slices in the foreground. Different integration strategies are shown and their resulting layer sizes. (a) The layer is generated by y values at the midpoint of intersection Equation 5.11. (b) The layer is created by the average distance values Equation 5.12. (c) Always the maximum value is used Equation 5.13.

is extruded along the sub-voxels in y. We define the difference volume $e_{\mathbf{b_i}}(v)$ between the extruded 3D distance field and its original distances per direction $\mathbf{b_i}$ and voxel v as:

$$e_{\mathbf{b_i}}(v) = \sum_{ijk} d_{ijk} - \tilde{d}_{ijk} \tag{5.14}$$

For our optimization we store the difference volumes for each of the three directions for the remainder of the optimization process. For the fabrication step we extract the contour of a slice using the marching squares algorithm.

5.4 Shape Decomposition

It would be possible to fabricate each voxel individually along the direction with smallest error e_{min}. Figure 5.9(Left) shows a rendering of this solution for the Stanford Bunny model. However, assembling such a model would be very tedious. On the other hand, simply choosing one direction and slicing all voxels in the same direction is usually far from optimal. Our approach is to rather find a balance between the number of pieces that are sliced consistently and the volumetric error. We present two approaches to compute an optimal partitioning. First, a top-down approach that is conceptually based on a decomposition of half-spaces and a branch-and-bound optimization. Second, interpreting the problem of decomposition as a labeling problem we can employ a multi-label graph cut optimization over all voxels to find an optimal segmentation. We will explain these approaches now in detail.

Top-Down Decomposition with Axis-Aligned Parts

For finding clusters of consistently sliced voxels we choose a decomposition of the model into half-spaces. This results a binary space partitioning (see Chapter 4). We divide the shape by iterating over all possible locations for the split plane. This set is discrete because we consider splitting only on the faces between voxel cells. Our optimization will result in a number of cells. We define a cell ω (i.e. a box, $\omega \subset \Omega$) consisting of voxels and compute the error for a potential split along each of the planes. Each is consistently sliced along its optimal direction. We compute $e_{slc}(\omega)$ over each cell and define the error function $E_h(\Omega)$ for the cell, with h defining one half-space, as

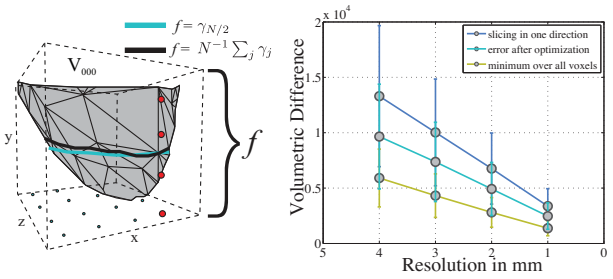


Figure 5.11: Left: We show a single voxel. f is evaluated along the y direction and influences the illustrated contour along the surface as shown in green and black. Right: We show that partitioning the shape significantly minimizes the volumetric error using just a small set (3-6) of partitions compared to standard one directional slicing.

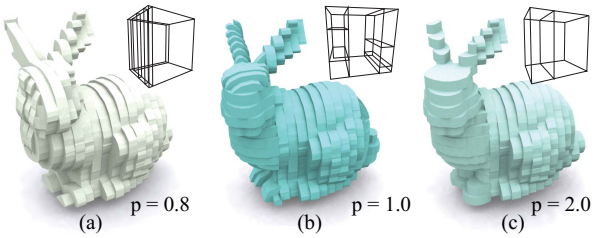


Figure 5.12: The splitting process divides partitions recursively in two half-spaces. We compute an error over each half-space and weight the errors using a p-norm. We show different p values resulting in different splits.

$$E_h(\omega) = e_{slc}(\omega) + \mathrm{T}(r) \tag{5.15}$$

where T ensures that the cell fits into the production volume of size $W(\mathbf{b_i})$:

$$\mathrm{T}(r_\mathbf{b}) = \begin{cases} error_{max} & \text{if } r_{\mathbf{b_i}} \geq (W(\mathbf{b_i})) \\ 0 & \text{if } r_{\mathbf{b_i}} < (W(\mathbf{b_i})) \end{cases} \tag{5.16}$$

with $r_{\mathbf{b_i}}$ being the size of the bounding box.

The error for a particular split plane can be computed from the errors $E_{h_l}(\omega)$ and $E_{h_r}(\omega)$ for the two half-spaces. We combine the errors using a particular p-norm,

$$E(\omega) = (E^p_{h_l}(\omega) + E^p_{h_r}(\omega))^{1/p}. \tag{5.17}$$

We find an approximately optimal solution over all possible split cells using a branch-and-bound approach. Thereby, we minimize the following objective function,

$$e_{opt} = \arg\min_E \sum_{|\omega|} E(\omega). \tag{5.18}$$

We solve the global optimization problem in a breadth first manner. The error in each cell is bounded by slicing the whole cell in one direction and then choosing the direction with smallest error which is e_{slc} (see Equation 5.2). We start with the voxel set of the overall bounding box of the input shape and take the k best options for the choice of the split plane and analyze the next level of splits. In other words, we iterate over all potential splits in ω computing the errors and keep the best k options in a priority queue. The maximum error of the k intermediate results define our upper bound. Per iteration we add one split to the set of existing cells in its branch. In the next level we analyze the set of cells again finding the best options to advance the next split thereby pruning suboptimal solutions. We define our relaxed lower bound as

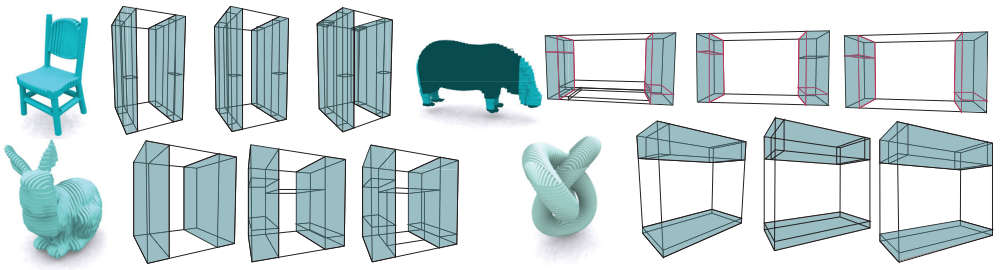


Figure 5.13: We show that the partitioning process is stable under a refinement of resolution (5mm, 2mm, 1mm) during optimization. Marked are the regions where the first splits occur independent of the voxel resolution.

$$E_{LB} = e_{\min}(1 + P/\tau) \tag{5.19}$$

where P is the number of partitions and τ is a user defined parameter that balances the number of parts with the allowed error. Recall, that $e_{\min}$ (in Equation 5.1) is the natural lower bound resulting from taking the smallest error in each voxel.

We found $\tau \in [2.0, 10.0]$ results in good approximations in a reasonable optimization time and in a desirable number of parts (between 4-11 parts) as shown in Figure 5.17. The optimization error e_{opt} is the sum over all partition errors.

p-norm Evaluation Figure 5.12 shows that the choice of the p value significantly influences the position of the splits and the convergence of the optimization. Values $p \leq 1.0$ tend to advance the splitting incrementally, whereas values $p \geq 1.0$ balance the partitioning and lead to fewer parts.

Stable partitioning We can show that the above decomposition process is stable to resolution changes. Figure 5.13 shows that with increasing resolution the first splitting regions stay stable during the optimization process over 1mm-5mm layer resolution (marked in green). Depending on the manufacturing goal we propose that optimizing and decomposing at lower resolutions improves the overall accuracy even if the machine resolution is higher. Additionally, this decreases the processing time and memory footprint of the complete process.

Shape Decomposition using Multi-label Graph Cut

The previously presented top-down approach has the advantage of simple fabrication and assembly. We take a bounding box of voxels, create layers along the computed fabrication direction and connect the solid parts. A disadvantage of the proposed method is that the planar boundaries between parts tend to be visually disturbing (see Figure 5.16).

To improve the visual quality, we propose a segmentation that creates parts with smooth, non-planar boundaries. At the time, we have to ensure that this still results in a producible shape decomposition. The previous decomposition method assumed planar boundaries at the partitioning cuts so that the inside of the part was well defined. With a segmentation that yields non-planar boundaries the assignment of an interior voxel

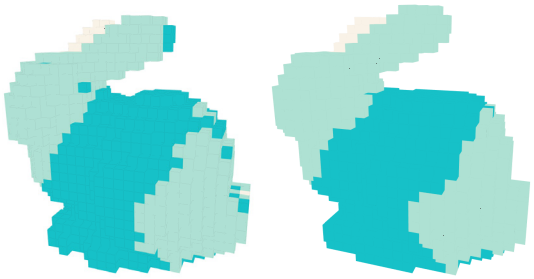


Figure 5.14: (a) Initial labeling after multi-label graph-cut. The circles mark the small segments that cannot be fabricated easily. We propose to filter these segments.(b) The final labeling after filtering.

to a specific part might result in non-constructability, e.g. interlocking configurations between parts. To avoid this, we propose to use an object shell given by a volumetric boundary representation. For this, we extend our surface voxelization towards the center of the shape by creating voxels also on the inside (for about $\sim 1/10$ of the size of the object). Since for these voxels no preferred fabrication direction exists their fabrication errors are set to zero.

With the shell representation, the segmentation process over the voxel grid can be modeled as a labeling problem, see Chapter 4 for an introduction to the idea. The voxels are our input elements and the three possible fabrication directions are labels, i.e. directions $\mathbf{b_0}, \mathbf{b_1}, \mathbf{b_2}$. Our objective is to identify segments that have similar voxel fabrication errors. This leads to an energy minimization problem which we solve using the graph cut based expansion move algorithm [Boykov et al. 2001], [Boykov and Kolmogorov 2004], [Kolmogorov and Zabih 2004]. We define our energy function very similar to Equation 4.6 as

$$E(x) = \sum_{v \in \mathcal{V}} \Phi_v(x_v) + \sum_{(v_i, v_j) \in \mathcal{N}} \Theta_{(v_i, v_j)}(x_{v_i}, x_{v_j}). \tag{5.20}$$

The data term $\Phi_v(x_v)$ describes the cost of assigning a label x_v to a voxel v based on the three voxel fabrication errors

$$\Phi_v(x_v) = \sum_{v \in V} e_{\mathbf{b_i}}(v). \tag{5.21}$$

The smoothness term $\Theta_{(v_i, v_j)}(x_{v_i}, x_{v_j})$ defines the relationship between neighboring voxels and measures the cost of assigning the labels x_{v_i}, x_{v_j} to the neighboring voxels v_i and v_j. The term hence penalizes different labels for adjacent voxels

$$\Theta_{(v_i, v_j)}(x_{v_i}, x_{v_j}) = \sum_{(v_i, v_j) \in \mathcal{N}} \begin{cases} 0 & \text{if } x_{v_i} == x_{v_j} \\ 1 & \text{otherwise} \end{cases} \tag{5.22}$$

The energy minimization results in a labeling for each voxel with one fabrication direction $\mathbf{b_i}$.

As explained in Chapter 4, we model the energy minimization as a graph cut problem. The graph is in our case given by the voxels $v \in V$ describing nodes and its neighboring voxels form the connected edges of the graph. Additionally, each label $x \in \{0, 1, 2\}$ is interpreted as a terminal node that connects to all nodes in V.

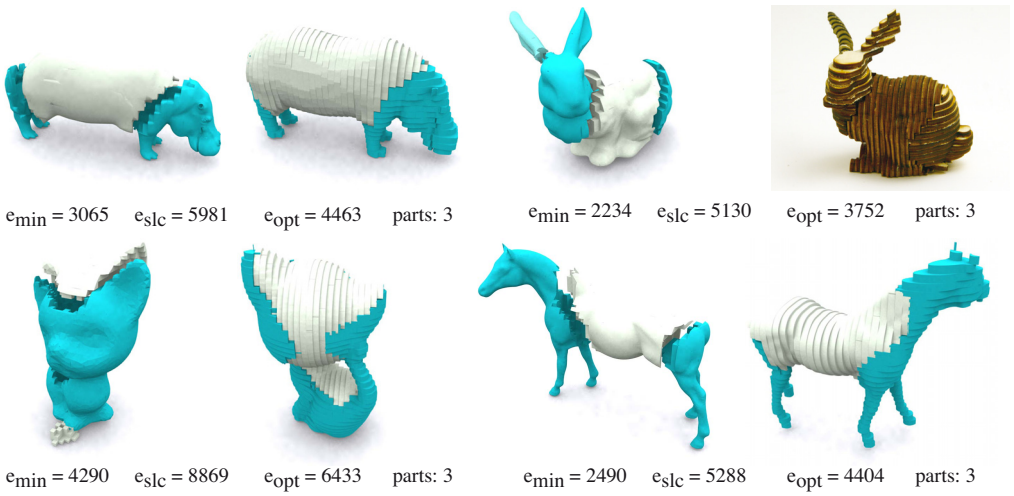


Figure 5.15: Results of the multi-label graph cut segmentation for various input meshes. We show the handcrafted results or renditions. In contrast to the top-down approach from Figure 5.17 segmentation boundaries are not planar but more natural. We also show the best approximation error and the error after optimization. Additionally, the number of resulting subparts is given. Note, overall error differences compared to top-down optimization are due to model size differences.

To set up neighborhood relations we use the 26-voxel neighborhood and compute three directional errors for each element, as discussed in the previous sections. In the graph this leads to a set of connected components where each component or segment matches their assigned labels. Figure 5.14(a) shows the results of a segmentation. We see that the resulting partitioning of the surface leads to large segments that change smoothly depending on the underlying error distribution. However, we also notice a set of very small segments, sometimes consisting only of a few voxels (marked with red circles). These parts would be tedious to fabricate, if possible at all. Therefore, we propose to filter the resulting segmentation. In particular, we compute the average voxel count over the segments and reassign all segments that are smaller than the average segment size to the neighboring segments. This is accomplished by assigning each voxel of these segments to the majority labeling of its neighbors. In ambiguous cases, we assign the voxel such that it minimizes the voxel error. Figure 5.14(b) shows the filtered result. Figure 5.15 presents the proposed segmentation method with more natural segmentation boundaries. One disadvantage of the segmentation is that global optimization constraints such as maximum part size cannot be easily incorporated (see Equation 5.16).

5.5 Evaluation and Results

To show that our framework significantly improves additive manufacturing processes we evaluated our results on the set of 3D objects shown in Figure 5.17, Figure 5.18 and Figure 5.15. All models were generated and analyzed for a size of approximately 150mm (chair is over 300mm in height) and resolutions from 5mm-0.5mm. The subsampling was computed with about 100dpi consistently over all resolutions. On a standard desktop computer the processing took from several seconds up to about an hour depending on

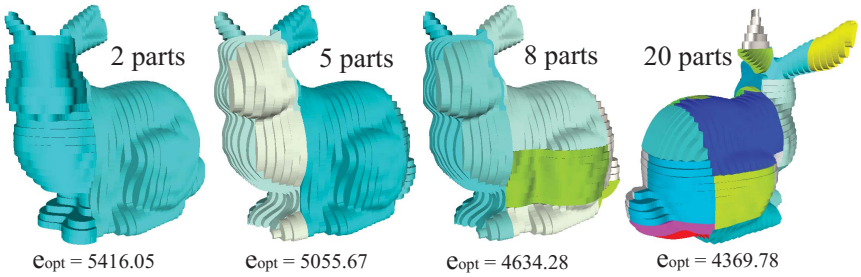


Figure 5.16: We show an increasing number of parts and the resulting optimization error e_{opt}. The best approximation error for this example is $e_{min} = 3178.04$. Slicing the object along its direction with the minimal error would result in $e_{slc} = 7462.34$. This example is generated for a material thickness of 3mm for a model size of 150mm.

the resolution. Printed objects were automatically rotated in its optimal position facing with the larger footprint towards the printing platform.

Optimal Slicing Direction Figure 5.7 and Table 5.1 show that the normal clustering method outperforms the PCA approach resulting in a lower best approximation error over all voxels. Figure 5.5 we also report mean and standard deviation over all models. the results demonstrate that we consistently achieve lower error rates by about 10% over the whole resolution scale. Interestingly, the results of our clustering method often correspond to the natural upright direction. This suggests that the approach can be used more generally for 3D printing.

Optimized Contour While standard additive manufacturing methods intersect the geometry at the center of a slice we propose an optimization by extracting the contour out of the distance field. We demonstrate that our method minimizes the volumetric difference error significantly in Figure 5.9. We plot the mean and standard deviation over increasing voxel resolutions showing that even for high resolutions up to 0.25mm the minimum voxel error improves between 20%-35%. To compute the volume difference over decreasing resolutions we need to account for the loss in sampling resolution. Therefore we use more subsamples for lower resolutions.

Branch and Bound Optimization Evaluation Our proposed top-down optimization process does not necessarily lead to a globally optimal partitioning. However, as shown in Figure 5.16 with the first five to eight parts the decomposition process lowers the volumetric difference error about 50% percent compared to slicing only in one direction on all our reference models. The error decreases only slowly when the number of parts is increased further. Figure 5.11 (right) also validates that we significantly improve accuracy compared to a standard one-directional slicing.

Figure 5.17 and Figure 5.18 shows additional a variety of results, including objects sliced in their best direction and their optimized decompositions. We provide the minimal volumetric error along the boundary of the shape e_{min} (Equation 5.1), the error that would result from one directional slicing along the best slicing direction e_{slc} (Equation 5.2) over the complete model and the error after optimization e_{opt} (Equation 5.18). We also show the user defined value τ used for the results. It can be seen that τ correspond to the number of parts generated.

In some cases our top-down splitting algorithm would prefer to cut through very thin connections that would be tedious to assemble, e.g. a 'toothbrush' shape cutted vertically along the 'brushes' instead of cutting through the toothbrush 'head' horizontally. We suggest to prevent this problem by adding an additional energy term C to Equation 5.15. We define C as the weighted connection cost of cutting area A over cutting perimeter P weighted by a user defined value κ

$$C = \kappa \;\; A \;\; P \tag{5.23}$$

Graph Cut Optimization Evaluation Figure 5.15 shows a variety of segmented 3D shapes using our proposed segmentation method. Additionally, we provide the minimal volumetric error along the boundary of the shape e_{min} (Equation 5.1), the error that would result from one directional slicing along the best slicing direction e_{slc} (Equation 5.2) over the complete model and the error after optimization e_{opt} (Equation 5.18). Similar to the top-down decomposition approach, we improve the surface error by about 50%.

Visual artifacts along the splits Our methods generate results optimized for accuracy but also introduces additional visual artifacts along the segmentation cuts. While the planar partitioning boundaries for the proposed branch-and-bound optimization (Figure 5.17) are visually very prominent, they are more naturally along the surface for the graph cut optimization (Figure 5.15).

With increasing layer thickness we have found that thin structures might also suffer visually from slicing in the wrong direction. For example, the horse model in the top row of Figure 5.17 is best sliced along the direction of its torso. This results in the legs being represented badly.

5.6 Discussion and Outlook

In this chapter we proposed different strategies to reduce direction fabrication bias. We evaluated the proposed techniques experimentally and demonstrated a significant reduction in error rate. However, for higher resolution 3D prints it is often not necessary to improve on the accuracy. There are other factors that make the distinction of the directions worthwhile: different tensile strength or strain [Bagsik and Schöppner 2011], [Umetani and Schmidt 2013] (i.e. one can increase the stability of the model by choosing the right orientation in each part), different build time [Danjou and Köhler 2009] (one can save production time by orienting the parts differently), different dimensions of the build volume [Luo et al. 2012], or changing amounts of support material (i.e. one can save and time by rotating the mesh) as we will discuss in the following chapter.

Figure 5.17: We show the input 3D shapes (left), a result that is sliced in its best direction (middle) and the handcrafted or 3D printed results or rendered images (right). We also show the best approximation error, the error that would result from one directional slicing and the error after optimization. Additionally parameter settings and the number of resulting subparts are annotated.

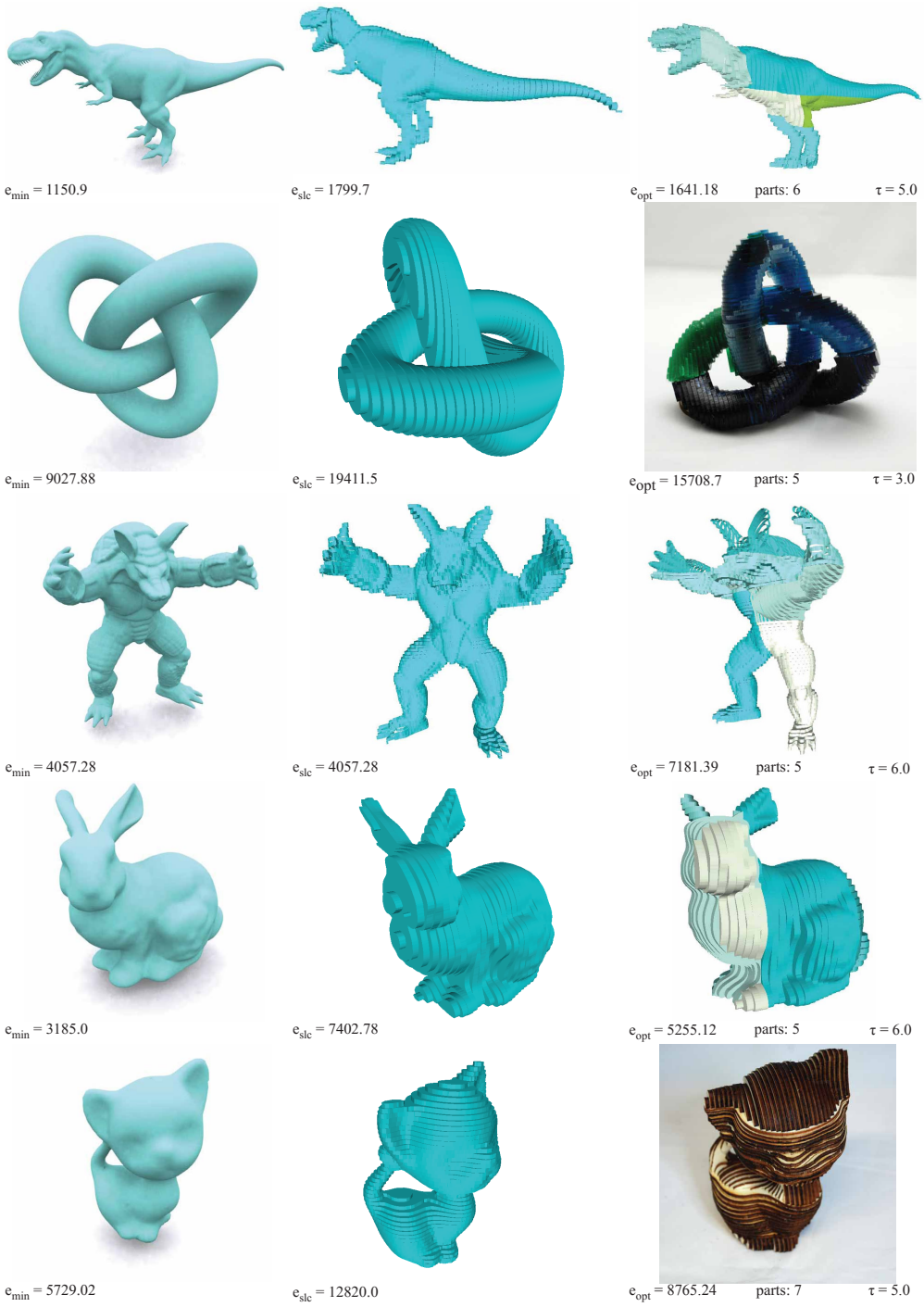


Figure 5.18: We show the input 3D shapes (left), a result that is sliced in its best direction (middle) and the handcrafted or 3D printed results or rendered images (right). We also show the best approximation error, the error that would result from one directional slicing and the error after optimization. Additionally parameter settings and the number of resulting subparts are annotated.

Chapter 6

Binary Space Partition For 3D Shape Manufacturing

Nothing s beautiful from every point of view.
— Quintus Horatius Flaccus

In the previous chapter we introduced methods to improve the accuracy of layered manufacturing. In this chapter we present a novel approach for 3-axis manufacturing that is based on the observation that the movement of the machine head of 3D printers and 3-axis CNC milling machines has limited operating range. Since both technologies have three degrees of freedom, the head can move horizontally in the x-y plane and vertically in the z-direction as shown in Figure 6.1.

Consequently, for 3D printing, overhanging parts of the shape need support material for fabrication. The amount of support material depends thereby on the volume of projection on the ground plane or subjacent parts of the input mesh. Figure 6.1(Middle) shows an example of the support material needed for Fused Deposition Modeling.

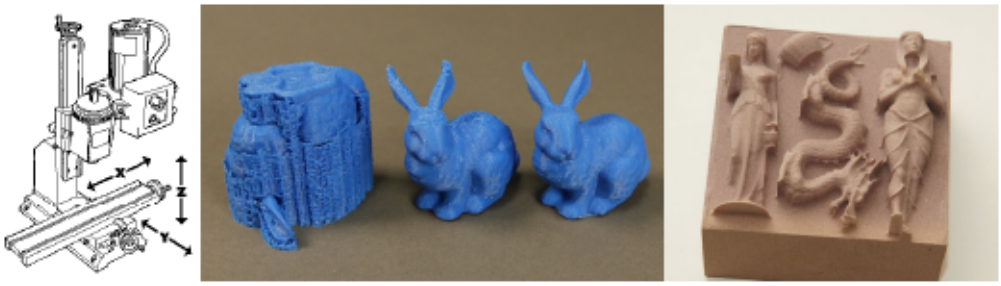


Figure 6.1: Left: 3-axis milling machine with limited degrees of freedom introduces limitations for additive and subtractive manufacturing. Middle: Most 3D printing techniques need support material to strut overhanging structures. Right: The geometric complexity of 3-axis CNC milling is limited.

In the case of a 3-axis CNC milling machine, the geometric complexity that can be manufactured is limited to the volume of the projective hull of the input geometry onto the ground plane because of the limited operating range of the tool head. This results in a relief-like structure where no undercut is possible, as shown in Figure 6.1(Right). To produce 3D shapes that cannot be fabricated from a single direction currently several complex semi-manual steps are necessary so that each part of the mesh surface can be milled separately.

Our objective is to ease both limitations by dividing the input shape into two parts using a splitting plane as shown in Figure 6.2. In the case of 3D printing, we show that this leads to a significant reduction in support material. In the case of 3-axis CNC milling, we demonstrate how to create physical 3D shapes with an acceptable fabrication error by splitting the shape in two parts. The main challenge for both uses cases is to find a suitable partitioning plane to divide the input shape so that both goals can be achieved. Note, we present here a first detailed explanation and solution of the problem but this is not a conductive investigation and fully explored idea.

6.1 Introduction

The design and engineering process of real-world objects often takes many iterations from a simple design prototype to the final product. Especially in the beginning of the development process many design prototypes are needed [Koberg and Bagnall 1976] and these should be obtained as easy and cost efficient as possible. Figure 6.2 shows two possible applications of our partitioning approach to support production development. As already mentioned, we distinguish between two use cases: 3D printing support material reduction (material cost and time optimization) and 3D shape manufacturing for 3-axis CNC milling with an acceptable fabrication error (time and material optimization).

Figure 6.2: Left: Two parts of the input shape are printed separately using stereolithography. The needed support material for an FDM print is decreased compared to an a single part along an up-right direction. Right: A 3D shape out of wood fabricated in two pieces using a CNC milling machine. The fabrication error is minimized by the choice of the partitioning plane.

3D Printing Support Material Optimization

Depending on the 3D printing technology saving material is not only a cost but also a time factor and even minor optimizations might have a large impact on the process efficiency. In general one can say that the projected volume of the overhanging geometry onto the building platform or subjacent geometry are an upper bound for the amount of material used. Recently, 3D printing software and research projects [Wang et al. 2013] provide first approaches to reduce the amount of material by computing optimal structures inside the projected volume. This can be done as an additional step to our proposed approach.

As already explained, support structure is needed when the object geometry has overhangs with respect to the fabrication direction. In convex regions no support material is needed to strut the printed structure. Hence, splitting an object smartly into overall convex subparts reduces the required support material. Finding an effective partitioning can be computationally expensive. Therefore, we propose a novel and very effective approach that splits the input shape into two parts with reduced overhangs. To the best of our knowledge, the only prior work related to this approach has been developed by Ilinkin et al. [2002]. Their goal is also to seek a decomposition into two polyhedra for which the total support requirement is minimized but, different to our approach, with respect to an already given partition plane normal,i.e. fabrication direction.

3D Shape Manufacturing for 3-Axis CNC Milling

There are many materials, such as wood, that cannot be used in a 3D printing process. CNC milling machines offer an alternative and are a common part of today's mass production manufacturing pipelines. The geometric complexity that can be manufactured depends on its the degrees of freedom, i.e. the number of axes a CNC machine has. Creating 3D shapes is a surprisingly complex and expensive process and there are only a few options to generate a three-dimensional sculpture:

1. The used CNC milling machine has enough axes, i.e. degrees of freedom, so that the tool head can be positioned at any point in the production volume and tangentially about the surface. These multi-axes machines are in general very expensive and hence not available to our target audience.

2. The machine has not enough degrees of freedom. The 3D object cannot be created in a single job and hence has to be manufactured iteratively. One has to manually interrupt the process, reposition the incomplete shape and register the intermediate state. The registration process is necessary to avoid collisions of the tool head with the already manufactured shape. This is a very complex and time consuming process.

3. The machine has not enough degrees of freedom but generates a shape consisting of parts simultaneously in a single job. The pieces fit together and can be assembled to form the final shape. The approach has the advantage that neither an expensive machine nor an expert-driven time-consuming manufacturing process is needed.

Our objective is to use the third approach and to do the decomposition computationally. We suggest to split the object once and mill both parts in a single milling job. As we try to find a split that reduces the projected volume, i.e. the volume that cannot be removed by the tool, we create an object that has a certain fabrication error but is still

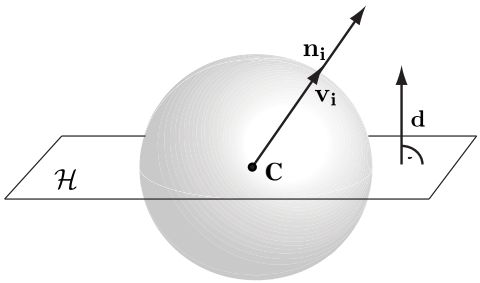


Figure 6.3: Our goal is to find a partitioning plane $\mathcal{H}$ with center C and normal $\mathbf{d}$ such that the mesh vertices $\mathbf{v_i}$ and vertex normals $\mathbf{n_i}$ correlate. This is the case in convex regions of the mesh surface as shown here for a sphere.

meaningful for many applications such as the rapid prototype design. To the best of our knowledge, this approach has not been considered before.

6.2 Selection of the Partition Plane

Given a triangle input mesh M, our goal and the main contribution of this chapter is the effective computation of a partitioning plane $\mathcal{H}$ that is specified through its normal $\mathbf{n}$ and offset o to the coordinate origin. $\mathcal{H}$ should minimize the volume of the projective hull of the input geometry onto plane and subjacent parts of the colliding mesh along the inverse fabrication direction $-\mathbf{n}$. We call the projective volume the fabrication error E_V which is defined in Section 6.3. The motivation behind our computation is to find correlations between vertex positions and its normals. For this, we require the input mesh to be positioned in the center of the coordinate system and scaled to fit a unit bounding sphere. This ensures that positions and normals of the shape are within the same bounds. Our goal is then to find a direction $\mathbf{d}$ over all vertex positions $\mathbf{v_i}$ and vertex normals $\mathbf{n_i}$ such that

$$\mathbf{d} \cdot \mathbf{n_i} \approx \mathbf{d} \cdot \mathbf{v_i}. \tag{6.1}$$

The intuition behind this is illustrated in Figure 6.3. The difference of the vertex positions and their corresponding vertex normals reflects the convexity of the shape. By computing the sum of differences between normals and positions we find the center C of their correlation and thereby the center of the partitioning plane which corresponds to its offset o

$$C = \sum_{i=0}^{N} \frac{n_i - v_i}{N}. \tag{6.2}$$

We normalize the vertex positions around the correlation center by subtracting C from v_i represented as V. N contains the vertex normals

$$V = \begin{pmatrix} v_1 - C \\ v_2 - C \\ \vdots \\ v_n - C \end{pmatrix}, \quad N = \begin{pmatrix} n_1 \\ n_2 \\ \vdots \\ n_n \end{pmatrix}.$$

Now, the strongest correlation between vertex positions and vertex normals can be computed by solving the following eigenvalue problem

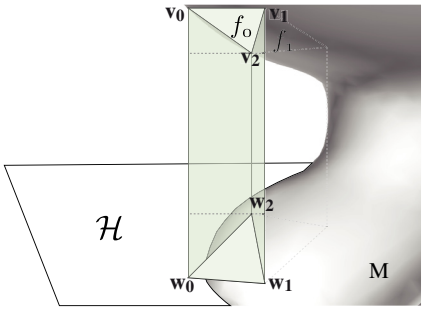


Figure 6.4: We show face f_0 of input mesh M projected on the partition plane $\mathcal{H}$ and in between parts of the mesh resulting in an irregular prism spanned by the face vertices v_i and the projected points w_i.

$$V^T V\mathbf{d} = V^T N\mathbf{d} \tag{6.3}$$

$$(V^T V)^{-1} V^T N\mathbf{d} = \lambda\mathbf{d} \tag{6.4}$$

with the correlation matrix X

$$X = (V^T V)^{-1} V^T N. \tag{6.5}$$

We find $\mathbf{d}$ as the major eigenvector of the eigendecomposition of X. It points in the direction of the strongest correlation regions, which correspond to the convexity of the surface. We use this eigenvector as our plane direction, i.e. fabrication direction, and C as the plane origin. By using this vector as the normal for the splitting plane we ensure that the tool head of the manufacturing device can reach a main parts of the surface on each side of the plane. That results in less projection volume.

6.3 Fabrication Error

To be able to evaluate the effectiveness of our proposed method we define the fabrication error E_V. We compute that error by evaluating the volumetric error E_f for each face of M. We do this for all triangles that face towards the partition plane normal, where $\mathbf{n_f} \cdot \mathbf{n}_\mathcal{H} < 0$. To determine E_f we projecting each triangle onto either the plane $\mathcal{H}$ or in-between faces of the mesh that point in the direction of the plane normal where $\mathbf{n_f} \cdot \mathbf{n}_\mathcal{H} > 0$. The triangle vertices v_f and the points w_f that arise from the projection form an irregular triangular prism shown in Figure 6.4. We follow the derivations of Prévost et al. [2013] to compute the volume of the prism using the divergence theorem by integrating over its 2D surface integrals

$$E_f = \frac{1}{18} \sum_{f \in \mathcal{F}} ((v_j - v_i) \times (v_k - v_i)) \cdot (v_i + v_j + v_k). \tag{6.6}$$

The overall projection volume E_V is then computed by summing over the prism volumes of all mesh faces with $\mathbf{n_f} \cdot \mathbf{n}_\mathcal{H} < 0$ on both half-spaces of $\mathcal{H}$.

$$E_V = \sum_{f}^{M} E_f. \tag{6.7}$$

Additionally, we define $E_\mathbf{d}$ as the error that arises by not partitioning, i.e. fabricating the object as a single piece or printing it in only one direction. To compute $E_\mathbf{d}$ we

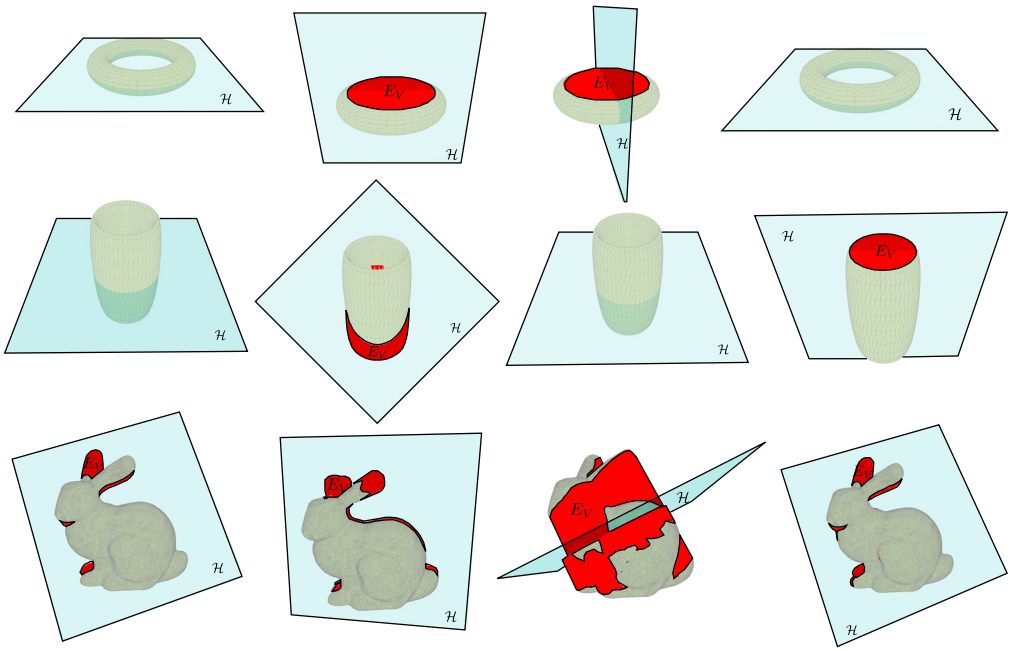


Figure 6.5: We illustrate the results of different partitioning planes for three different models and mark in red the fabrication error or support volume that arises. (a) Shows our proposed position and normal correlation approach. (b) Uses the basis with the smallest fabrication error that is proposed in Chapter 5. (c) Illustrates a partitioning plane that uses the first eigenvector of a vertex-position-based PCA as a normal and the area-weighted center of mass of the mesh as a center. (d) Shows corresponding to (c) a partitioning plane that uses the first eigenvector of a normal-based PCA.

first determine the splitting plane normal direction $\mathbf{n}_\mathcal{H}$ but position the plane at the extremal vertex in the $\mathbf{n}_\mathcal{H}$ and $-\mathbf{n}_\mathcal{H}$ direction. We compute the projection volume and define $E_\mathbf{d}$ as the minimum of both projection volumes.

We illustrate the prism and its approximated volume in Figure 6.4 and show the fabrication error in red in the result renditions in Figure 6.5 and Figure 6.7.

6.4 Fabrication, Results and Evaluation

We tested and evaluated our approach on a variety of 3D shapes. To create the manufactured results we use the Form1 stereolithography printer by Formlabs and a 3-axis milling machine by VK-Technik GmbH. All fabrication errors are measured in mm^3 over shapes with $100mm$ maximum width.

Figure 6.5 shows results of four different partitioning planes and fabrication directions for three different models. The fabrication error or support material volume E_V is shown in red. Figure 6.5(a) shows the results for our proposed plane selection method. This is compared to the direction with the smallest fabrication error of the approach presented in Chapter 5 in Figure 6.5(b) and the major eigenvector of a vertex position-based PCA

(c) and a area-weighted face normal-based PCA (d). The plane center is computed as the area-weighted center of mass of the mesh (c-d). It can be seen that the face normal-based PCA and our correlation method give very similar results. However, for the torus and the scaled torus in the first two rows our plane selection method is not biased towards large face regions pointing in possibly non-convex directions with respect to the fabrication direction. The superior performance of our plane selection method is also evident in Table 6.1 and Table 6.2 where we compare six-different approaches to compute the partitioning plane, i.e. the four methods used in Figure 6.5 and in addition a given upright direction of the original mesh and an exhaustive search over 450 different normal directions over the half-sphere and 30 different plane offsets along these directions. We compare the error E_V of the two-part partitioning and the error volume that arises by not partitioning, i.e. fabricating the object as a single piece or printing it in only one direction $E_{\mathbf{d}}$. Table 6.1 and Table 6.2 show the errors in percentage compared to the volume of the original input mesh. Note, the projection volume can be larger than the volume of the original mesh resulting in numbers larger than 100%.

Overall, our approach outperforms the other approaches for most of the shapes. As discussed before, the normal-based PCA and our proposed plane selection method perform almost equally well. For many models our approach results in a slightly better splitting plane. For models with higher genus, such as the Chair and CAD model our method does not perform very well. This is mainly because a single splitting plane is not sufficient enough to minimize the overall projection error for such complex geometry. We discuss possible solutions in the next section. Comparing all $E_{\mathbf{d}}$ errors results in similar findings. We can conclude that, for all our tested 3D shapes, the presented plane selection and partitioning method is suitable for the applications we have in mind. However, as can be seen from the results of exhaustive search, our method is still far from optimal.

Figure 6.7 illustrates resulting 3D models. The areas in red show the fabrication errors. For most models the error seems acceptable for creating physical 3D design prototypes but certainly not for final products. As shown in the bottom left result of the CAD model, with increasing geometric complexity of the input mesh a two-part partitioning is not sufficient to create satisfying results.

For the 3D printing case we performed an additional real world test without partitioning the object. In Figure 6.6 we compare a set of 3D models oriented in the production volume using two different methods. The upper row shows the renditions of the simulated 3D printing results using our correlation based approach and its generated support structure. The bottom row uses the auto-orientation function proposed by Preform, i.e the software package provided by Formlabs. The results show the estimated total material usage for the 3D print. Note, that the Form1 printer uses only one material (for the object and the support material). It can be seen that our approach can save up to 40% of material for the evaluated case. In general, this emphasizes again that the choice of the object's orientation inside the production volume has many consequences on the resulting object.

The computation of our partitioning plane can be implemented very efficiently and takes under 10 milliseconds even for meshes with over 130k vertices. The error computation takes $\subset$1sec on a MacPro for meshes with $\subset$100k vertices.

	Correlation E_V	**n**-Clust. E_V	**n**-PCA E_V	Upright E_V	**v**-PCA E_V	Exhaust. E_V
Armadillo	**31.1**	42.8	95.8	149.4	153.1	**21.1**
Bunny	**6.1**	13.3	6.2	56.2	67.1	**3.2**
CAD	211.6	158.7	**120.2**	242.6	368.0	**53.6**
Camelhead	8.9	12.4	**8.1**	50.6	40.2	**5.0**
Chair	147.1	166.0	**120.9**	193.4	232.2	**80.7**
Cow	**5.0**	5.0	5.0	26.5	69.0	**5.0**
Hippo	**4.1**	4.1	4.1	16.7	55.4	**3.8**
Kitten	6.9	20.1	**5.1**	68.7	98.0	**1.2**
Lion	27.7	27.6	**26.3**	47.0	179.0	**23.6**
Prism	6.3	40.3	56.6	**2.5**	92.2	**0.0**
RockerArm	**5.9**	107.8	8.1	8.9	153.8	**5.6**
Three Cylinders	**9.4**	66.5	55.5	83.7	62.0	**6.2**
Two Cylinders	**0.0**	59.8	0.0	0.0	53.5	**0.0**
ThreeHoles	**0.0**	0.0	25.8	25.8	81.8	**0.0**

Table 6.1: We evaluate the fabrication errors for several 3D models using our proposed correlation method, a normal clustering approach, a face normal-based PCA, a given upright direction, a vertex position-based PCA and a fabrication direction found by exhaustively searching over 13.5k planes. We compare the error E_V of the two-part partitioning. Errors are in percentage compared to the volume of the original input mesh. We mark the minimal error in bold.

	Correlation $E_{\mathbf{d}}$	**n**-Clust. $E_{\mathbf{d}}$	**n**-PCA $E_{\mathbf{d}}$	Upright $E_{\mathbf{d}}$	**v**-PCA $E_{\mathbf{d}}$	Exhaust. $E_{\mathbf{d}}$
Armadillo	**78.9**	81.1	149.8	247.5	283.4	**66.3**
Bunny	32.7	44.1	**31.1**	86.9	111.0	**29.7**
CAD	300.6	144.2	**124.1**	380.4	524.3	**129.3**
Camelhead	35.5	43.0	**35.3**	76.0	70.1	**34.6**
Chair	**202.2**	238.1	207.0	293.4	316.5	**128.0**
Cow	**34.6**	34.9	36.8	43.9	108.6	**36.8**
Hippo	36.6	37.0	37.3	**27.5**	111.8	**31.9**
Kitten	30.1	46.2	**28.9**	73.9	113.9	**27.5**
Lion	**49.7**	51.1	51.1	86.3	277.9	**56.5**
Prism	32.5	65.72	56.6	**0.0**	54.2	**0.0**
RockerArm	**29.8**	180.7	31.1	40.2	236.2	**31.6**
Three Cylinders	**35.8**	93.8	110.7	135.4	127.8	**76.1**
Two Cylinders	**14.1**	88.4	14.1	14.1	43.0	**14.1**
ThreeHoles	**21.5**	21.5	83.5	83.2	144.6	**21.7**

Table 6.2: We evaluate the fabrication errors for several 3D models using our proposed correlation method, a normal clustering approach, a face normal-based PCA, a given upright direction, a vertex position-based PCA and a fabrication direction found by exhaustively searching over 13.5k planes. We compare the error volume that arises by not partitioning, i.e. fabricating the object as a single piece or printing it in only one direction $E_{\mathbf{d}}$. Errors are in percentage compared to the volume of the original input mesh. We mark the minimal error in bold.

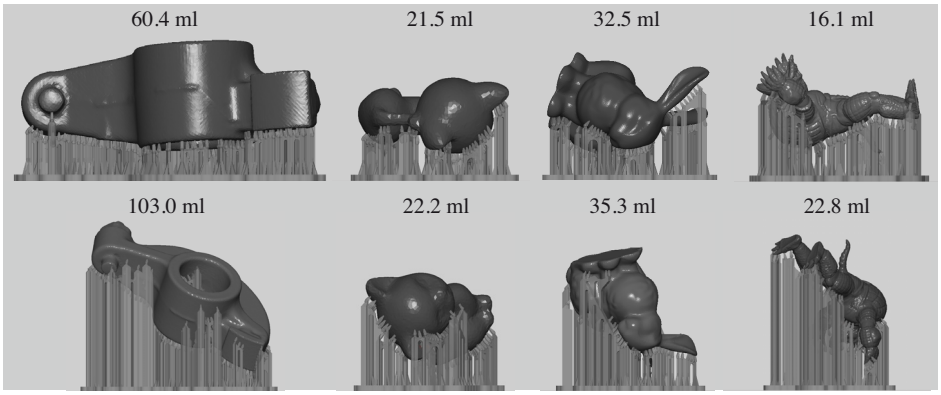


Figure 6.6: We show a set of simulated 3D prints generated with Preform, the software package of the Form1 Stereolithography printer by Formlabs. The upper row shows a set of 3D models oriented in the production volume using our correlation based approach and its generated support structure. The bottom row uses the auto-orientation function proposed by Preform. We also show the estimated total material usage for the 3D print.

6.5 Discussion

In this chapter we proposed an optimization method for fabricating design prototypes. We show that our promising technique can be useful for some 3D printing technologies such as Polyjet where most of the volume of projection is filled with expensive support material. We also show a simple approach for creating real-world 3D objects using 3-axis CNC milling. Nevertheless, the presented technique is a case study with much room left for improvements. We would like to discuss possibilities and limitations of the approach in the following.

Splitting of Complex Shapes The more complex and fragile the input geometry is the less useful is our single splitting plane. This is due to the fact that these models have higher genus and therefore might generate large errors in these inclusions and hole regions. The CAD model is a good example for this as shown in Figure 6.7. Note also, some input shapes that have inclusions can only created without fabrication errors using a 3D printer. CNC milling will always create fabrication errors.

Extended Partitioning The proposed splitting scheme can be extended to subdivide more than two parts of the shape. One would then have to find segments over the mesh surface that could be fabricated more efficiently in another direction than $\mathbf{d}$, similar to the top-down decomposition presented in Chapter 5.

Of course, we experimented with that machinery and implemented a similar approach to the mentioned top-down optimization using the splitting plane error E_V. Our hope was that the error falls under the user-defined value.

Unfortunately, that approach failed to generate more advanced results. That is mainly due to the following reason. An additional constraint for successive splitting planes is orthogonality which leads to insufficient partitions that do not necessarily lower the error for other fabrication directions. To overcome that problem one could use parallel splits and a voxelization to discretize the possible partitions.

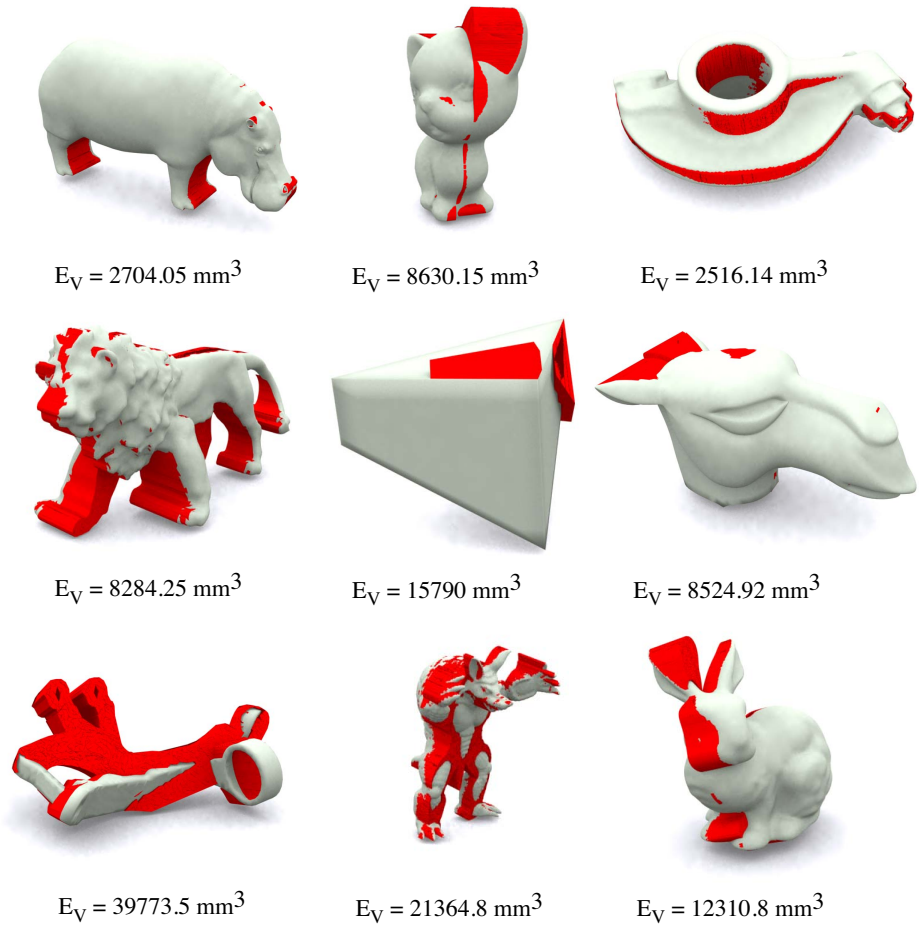


Figure 6.7: A set of results of our method. The regions in red show the fabrication error when splitting the input mesh, fabricating both parts using a CNC milling machine and assembling the two parts to the final approximated shape. Given is also the fabrication error for a 100mm sized object.

Overall, we found the single splitting approach already a good trade off between the number of parts where each part introduces visual cluttering and assembly effort. However, more research effort should be put into the extended partitioning to improve the results.

3D shape manufacturing using CNC milling We believe that our solution proposes a natural alternative to other approaches to create three-dimensional shapes using a 3-axis CNC milling machine. Since we offer to mill all parts of the shape in a single milling job given that all pieces fit the production volume, the process is fast, easy and cost-effective. Furthermore, all parts can easily be assembled. Depending on the desired accuracy we generate parts with little approximation error but with the visual appeal of an acceptable prototype (as shown in Figure 6.7). However, our solution does not necessarily fulfill industry production standards. Note, that we only consider 3-axis milling

machines in this chapter. For machines with more degrees of freedom these limitations typically do not exist.

Limitations Note, our binary partitioning does not ensure a global projection volume minimum. Additionally, our method does not incorporate machining and material properties. For example, some regions where the angle between face normal and the fabrication direction is small a support structure might not be not needed.

Chapter 7

Shape Fabrication by Sliding Planar Slices

There is no abstract art. You must always start with something. Afterwards you can remove all traces of reality.

— Pablo Picasso

The previous chapters were related to different manufacturing processes, their specific output mappings and their unavoidable approximation errors up to some degree of shape abstraction. The work presented in this chapter focuses specifically on the abstraction of shape. Our motivation is to create the very popular 3D shape abstractions which are colloquially called *cardboard sculptures* or *cardboard models*. The basic concept is the representation of an object by a sparse set of planar elements that define cross sections through the input object at different positions and orientations. The self-intersecting elements have prefabricated slits at their intersections and are assembled by sliding them together. We present a computational design to fabricate three-dimensional

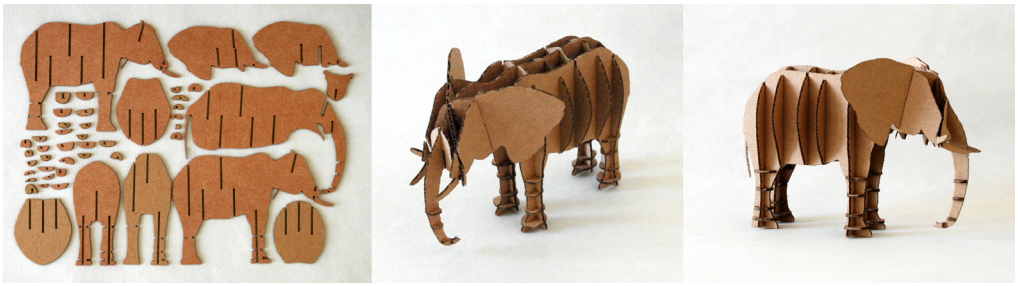


Figure 7.1: Cardboard models are abstractions of 3D models that consist of planar elements that can be easily assembled. Given a 3D model as input our algorithm automatically generates a set of elements (left) that can be fabricated and slid into each other. The photographs show a model created with our method.

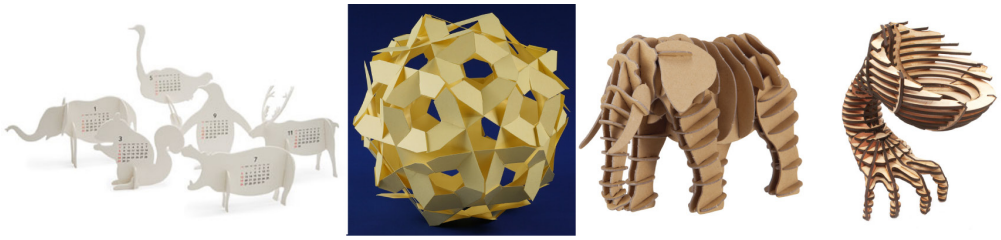


Figure 7.2: (Left to right) Abstraction of 3D models as cardboard cutout ©MOMA, H-Construction is a 'Slide-Together' geometric construction by George W. Hart ©George W. Hart, cardboard model by MUJI ©MUJI, cardboard model by 123D Make ©Autodesk.

shapes composed of these interlocking planar pieces. An example generated using our algorithm is shown in Figure 7.1.

Our work is motivated by an increasing demand of real-world prototypes for visualization. While 3D printers can be used to fabricate realistic replicas, they are expensive and the process is slow; printing a typical 3D model can take several hours. In contrast, our method allows for fast and easy fabrication of 3D shape approximations, and only requires equipment that is available in every office. Notably, while we share the goal of creating papercraft models from meshes, our approach significantly differs from related work that unfolds the surface [Mitani and Suzuki 2004] or tries to minimize the surface distance between 3D object and abstraction [Massarwi et al. 2007; Shatz et al. 2006]. In this sense, we trade realism with abstraction and fabrication complexity. The presented abstractions show a very popular approach for creating 3D toys and are also of interest for architecture and interior design as illustrated in Figure 7.2.

7.1 Introduction

A computational process to generate cardboard models faces three main challenges:

1. Given as input a 3D surface mesh, we need to obtain a set of planar sections p that perceptually approximates the object. They can be manually fabricated by sliding each of the planes onto one another.

2. Already simple planar abstractions face an elementary construction difficulty. Some pieces collide with other pieces during insertion and need to get clipped against them. For some pieces it cannot be ensured that they can be inserted completely as shown in Figure 7.5. Therefore we need to ensure constructability. We propose using a data structures to guide the process.

3. With a growing number of planes we face a combinatorial problem. As the ordering of element insertion during the construction significantly influences the appearance of the final physical model we need to define an optimization that maximizes the visual quality. As we show in Figure 7.15, a random order of planes is far from optimal due to necessity to clip colliding parts.

Automatically find an optimal set of planes covering all important geometric features is challenging, as geometric cover problems are NP-hard [Hochbaum 1997]. To address this problem, related work has suggested optimization heuristics [Décoret et al. 2003] and

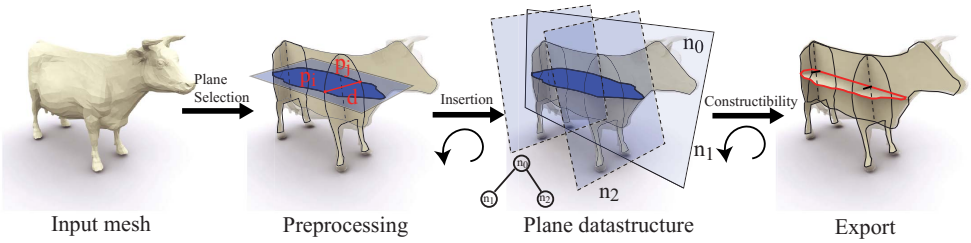


Figure 7.3: Overview of the construction process. Our pipeline starts with an input mesh. We then perform a preprocessing step which generates a set of polygons that are added iteratively to the cardboard model. By inserting the polygon into our data structure, we split the polygon into individual pieces and efficiently test if those can be physically slid into the cardboard model. If successful, the process is finalized by exporting the 2D fabrication plan.

very recently an approach that progressively selects planes to maximize feature coverage based on principles inferred from a user study [McCrae et al. 2011] and [McCrae et al. 2013].

Based on an analysis of construction rules, we propose an extended binary space partitioning (BSP) tree (see Chapter 4) that includes the insertion direction of planes as an efficient representation of such models. This data structure allows to evaluate the feasibility of newly added planar elements in the insertion process. We demonstrate the complete process by designing and fabricating cardboard models of various 3D shapes. The contributions of our research can be summarized as follows:

- We introduce a novel representation for *cardboard models* based on an extended binary space partitioning data structure.
- We propose a set of construction rules that respect physical constraints and guarantee that every piece can be slid onto the current construction.
- We present a complete automatic pipeline to generate constructible piece-wise planar shape abstractions and demonstrate its functionality with a variety of 3D models.

We will now explain each of the steps in our pipeline in more detail.

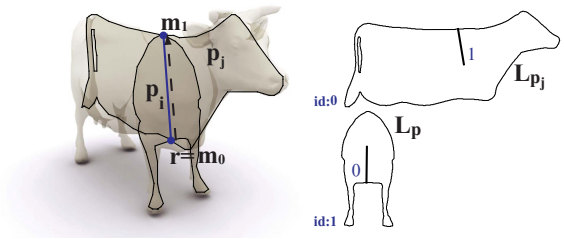


Figure 7.4: We define p_j as the *pivot polygon* of p_i when p_i is slit onto p_j and $\mathbf{m}$ as the *pivot direction* with the *pivot point* r. L_q and L_p define the extracted tiles in the output fabrication plan.

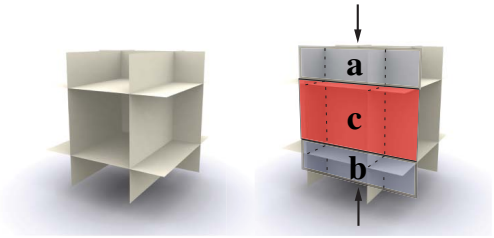


Figure 7.5: Insertion for a symmetric cardboard construction. Note, that the consecutive plane can neither from the top a nor from the bottom b slid into the model completely because part c is physically not insertable. Due to symmetry this means that this construction cannot be realized without losing parts of the planes.

7.2 Overview

Given a surface mesh M of a connected component in $\mathbb{R}^3$, our goal is to create a *cardboard model* abstraction that is physically constructible. The output is a set of planar contour sections $p = \{p_0, ..., p_n\}$ including intersection slits. We generate a construction plan for their assembly and the output layout, which can be printed and cut manually, sent to a cutting plotter, or laser cutter. The process is illustrated in Figure 7.3.

Our design process starts with sampling a set of candidate planar sections p by intersecting a plane $\mathcal{H}$ against the input object. For each of these polygons, we compute a quality measure that tries to quantify geometric features of the mesh which are covered by this plane. We define this set of planes as input for our construction algorithm. The construction is designed by iteratively adding planar sections. We specify for each element $p_i, i = 2, ..., n$ an already existing polygon $p_j, j < i$ to which it is physically connected to and slid onto (see Figure 7.4).

Physically inserting a planar section p_i from a specific direction is only possible if the insertion path is not blocked. To represent our abstracted object at any time and allow for efficient tests, we simplify the full insertion analysis to a straight line path for the plane to be inserted. This allows us to perform all tests based on a modified version of a *BSP tree* [Fuchs et al. 1980]. The full analysis of all possible insertion paths would be significantly more complex algorithmically, and would also not lead to a similarly elegant data structure. We argue that the number of cases in which a curved insertion path of elements would add to the aesthetics of a model are very small.

Linking a polygon p_i with another polygon p_j that is already part of the cardboard figure defines an intersection segment between p_i and p_j. Figure 7.4 shows p_j as the *pivot polygon* for p_i. In other words, the pivot polygon is the polygon we slide onto. The polygon intersection segment defines a *pivot point* r and a *pivot direction* $\mathbf{m}$ from which the insertion slit between p_i and p_j is derived. In addition to the data structure, we derive and analyze a set of construction rules that go along with the development of a cuttable layout as shown in Figure 7.4 and described in sections Section 7.4 and Section 7.5. Since the order of insertion is crucial for the outcome we use a branch-and-bound strategy to determine the best model. We test our pipeline on a large number of models. A subset is shown in Figure 7.16 and Figure 7.17.

It should be kept in mind, as cardboard models become more complex, the insertion ordering cannot be trivially solved. A fully area-preserving solution may not exist even

Figure 7.6: Left, middle: Given a set of planes generated by McCrae et al. [2011] we apply our insertion strategy in an optimal (left) and not optimal (right) construction case. Right: Cardboard model [McCrae et al. 2011]

for simple constructions (Figure 7.5). While in this example it is irrelevant which part is dropped for symmetry reasons, in most real-world models the insertion order leads to significantly different outcomes, see Figure 7.15.

We start the description of our algorithm with the selection of planes.

7.3 Plane Selection

Selecting a set of representative planes is challenging because the visual quality of the resulting figure is dependent on objective factors, such as coverage of geometric features and subjective factors, such as human perception and visual aesthetics. Our goal is to create a shape abstraction by finding a set of planes $\mathcal{H} = \quad H_0 \quad H_n$ such that their corresponding cross sections p approximate M. In theory, our construction method works in combination with any plane selection algorithm. For our pipeline, we incorporate various sampling strategies, including axis-aligned grid sampling, a simple yet powerful process that supports visual regularity patterns and the option for manual plane selection. We also generate results based on a sampling strategy learned from user data as recently proposed by McCrae et al. [2011] as can be seen in Figure 7.6. Additionally, several of these shape abstraction strategies were evaluated by Rosenberger [2012]. The work discusses various quality measures such as area, curvature, saliency and object silhouette length. It classifies approaches by grid-based, hierarchy-based and view based methods but leads to similar results as proposed by McCrae et al. [2011]. In the following we discuss the sampling strategies that are used to generate our final results in Figure 7.16 and Figure 7.17.

Optimized Axis-aligned Grid Sampling

We sample a number of equidistant planes along the axis-aligned bounding box. For each axis of the 3D shape we sweep the planes through the model searching for the largest overall cross section area. As this method does not take into account any knowledge about the shape itself, it usually results in a larger number of planes to represent the mesh faithfully compared to more sophisticated methods, but the resulting models look aesthetically pleasing, compact, and regular for many people. Rosenberger [2012] showed in a small user study of 18 participants that on average this is perceived most aesthetical. Regular, axis-aligned sampling works well for 'low-frequency' objects as shown in Figure 7.16, but for reproducing thin features, the number of required planes

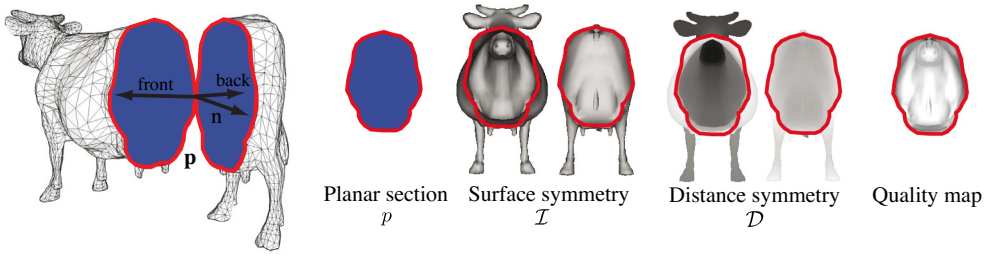


Figure 7.7: Evaluating the quality of the plane. Evaluation buffers: Distance in front and back of the plane, i.e. the distance symmetry term $\mathcal{D}$. Gouraud shading with the light source pointing opposite to the plane normal results in the surface symmetry term $\mathcal{I}$. We sum over the resulting quality map to compute our quality measure.

makes construction impractical. Furthermore, a grid-based method often fails when shape features are not axis-aligned.

Plane-Quality-Based Sampling

We propose a simple technique that can be combined with grid-based methods. This approach selects only the first few planes based on a simple importance sampling using a quality value μ. The remaining planes are selected by the grid based-sampling. There are two reasons behind the intuition of our quality value. We would like to favor planes:

1. that are oriented and positioned so that they cover only parts of the shape that are not already covered by other planes. The cover of the plane is thereby the total projected area of all mesh faces on the planar section p.

2. with normals oriented in its average surface direction which are additionally placed as close to a symmetry plane as possible.

We address the first issue by an iterative process that masks out parts of the mesh by marking faces as visited. Therefore, we evaluate all potential planes in each iteration. The set of planes is generated by uniformly sampling 4000 normal directions over a halfsphere and 50 offsets. Each plane has a normal directions $\mathbf{n}$ and the offset $D \in [0, b]$ to the origin, where b is the radius of the 3D model bounding sphere.

In the beginning, we take all faces of the mesh into account. This corresponds to a face visibility value of $V_f = 1.0$. We begin by selecting a plane and computing its quality value μ and repeat the process for all planes of our set. We choose the plane with the highest μ and mark all mesh faces with nonzero projection onto the plane as visited $V_f = 0.0$ for the next iteration. We repeat this process until the mesh is completely covered or a maximum number of planes is reached.

To solve the quality criterion we propose a measure based on two importance factors:

The *distance symmetry term* $\mathcal{D} \in [0, 1]$ encodes the difference of distances from two symmetric points $\mathbf{k_f}$ and $\mathbf{k_b}$ on the surface. The points lie in the front- and the backspace of the plane. The distances d_f and d_b from any points $\mathbf{k_f}$ and $\mathbf{k_b}$ to the plane with a normal $\mathbf{n}$ pointing in the direction of the front or back space and a plane offset D are defined by

$$d_{f,b} = (\mathbf{n} \cdot \mathbf{k_{f,b}}) - D. \tag{7.1}$$

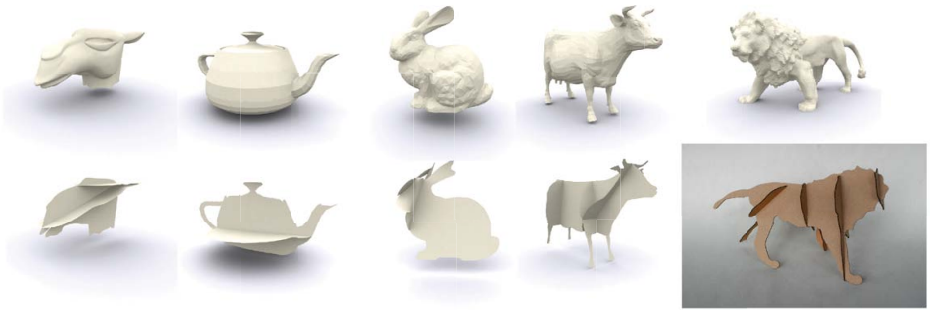


Figure 7.8: The original shape are shown in the top row. The first 2-5 planes are shown based on the proposed sampling strategy in the bottom row. The algorithm runs iteratively and stops when all faces have been projected on one of the planes.

The leads to the distance symmetry term

$$\mathcal{D} = 1 - (d_f - d_b). \tag{7.2}$$

The *surface symmetry term* $\mathcal{I} \in [0, 1]$ is a measure of correspondence between the mesh surface normals $\mathbf{s_f}$ and $\mathbf{s_b}$ at $\mathbf{k_f}$ and $\mathbf{k_b}$ respectively and the plane normal $\mathbf{n}$. It is computed as

$$\mathcal{I} = 1 - (\mathbf{n} \cdot \mathbf{s_f}) - (-\mathbf{n} \cdot \mathbf{s_b}) \tag{7.3}$$

The distance symmetry and surface symmetry terms are evaluated by:

$$\mu = \sum_{f \in M} V \cdot \mathcal{I} \cdot \mathcal{D} \tag{7.4}$$

This is accomplished by multipass rendering approach where an orthographic camera is placed in the center of p pointing in n and $-n$ respectively. The plane quality μ is computed by summing over the mesh faces and evaluating both terms in image space. This can be efficiently implemented using a graphics API which enables evaluation on the GPU (see Figure 7.7). The renditions for the distance symmetry term $\mathcal{D}$ are the depth maps in front and back space of the plane. The computation of the surface symmetry term $\mathcal{I}$ is accomplished using simple Gouraud shading with the light source vector $\mathbf{l}$ pointing opposite to the plane normal $\mathbf{n} \parallel \mathbf{l}$.

Figure 7.7 shows the resulting image buffers for the $\mathcal{D}$ and $\mathcal{I}$ term and the quality value of p. Figure 7.8 shows a set of example results generated with the proposed sampling. The algorithm results in just a few planar sections that are often visually not complex enough for a cardboard model or a 3D puzzle. But it can be used in combination with the grid-based sampling as shown in Figure 7.17.

Manual Plane Selection

What constitutes aesthetically pleasing planar elements for our abstraction can be very subjective. An automatic process often does not meet the visual standards of a user selection, especially when the 3D model is filigree and has very distinct geometric features that cannot be easily represented in a plane, or when it requires a higher-level

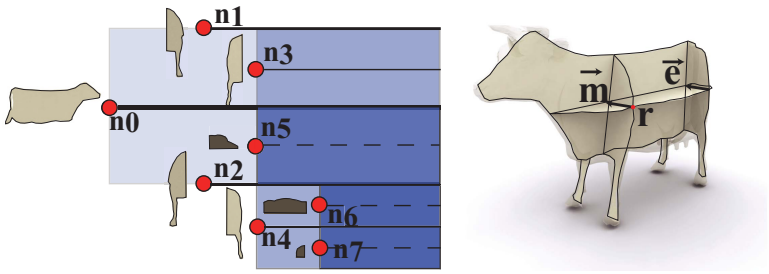


Figure 7.9: The BSP tree for the cardboard model on the right. The nodes n_i store their part of the polygon geometry and divide the scene in half-spaces. Left: A newly inserted part creates two parallel intersections.

understanding of construction. Therefore, we offer a very simple polygon insertion interface that allows to edit the cardboard model in a few minutes and can be used in combination with our automatic approach.

7.4 Constructability

With the set of selected planes $\mathcal{H}$ we now begin the construction process by simulating a physical cardboard model construction, i.e. we iteratively choose a plane and slide it into the current model. Consider a current construction state as shown in Figure 7.4. When we try to insert an additional slice p to a cardboard sculpture, there are three possible outcomes:

- The slice p_i can be directly put on the pivot polygon p_j and any other polygon with an intersection segment parallel to the insertion direction $\mathbf{m}$. There exists no polygon blocking the insertion path, so p can be inserted completely.
- We insert p and there is at least one already existing polygon that blocks the insertion path of p except the polygon p_j or any other polygon with an intersection segment parallel to $\mathbf{m}$. We then split p in parts and insert the parts separately from the directions $\mathbf{m}$ and $-\mathbf{m}$. Our data structure guides the splitting process (see Figure 7.9).
- Planar section p is split, but at least one of the resulting planar elements cannot be slit into the cardboard model due to its shape as shown in Figure 7.11 or the resulting intersection slits that would endanger the stability of the sculpture, as described in Section 7.4.

In the following subsections we describe each of these cases and our construction algorithm in more detail.

Cardboard Model Data Structure

Before we insert the polygon p into our data structure, we decide on the *pivot polygon* p_j with which we want to link to p. Recall, the pivot polygon is the polygon we slide onto. With the choice of p_j we compute the intersection segment between both polygons and identify its endpoints which define our pivot direction $\mathbf{m}$ and the pivot point r.

To maintain the cardboard model geometry we utilize a data structure. It is based on a *Binary Space Partitioning*, which we extend by modifying the insertion algorithm

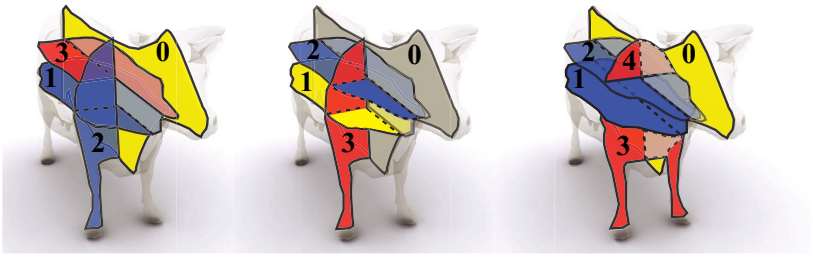


Figure 7.10: Insertion order makes a difference due to already existing polygons. In red we highlight the last inserted plane, in yellow we show its pivot polygon.

with a set of additional rules. We want to restrict the BSP insertion to geometry that can be slid in safely until an existing plane is blocking the insertion path. Note that this is different from a standard BSP tree (see Chapter 4): in our version planes or geometry are not necessarily inserted on both sides of each existing plane, but the behavior rather depends on the type of node. We also consider intersecting several existing polygons. This is possible if the intersection segment with other planes is parallel to the pivot insertion direction $\mathbf{m}$ of p.

Figure 7.9 illustrates the insertion process with the help of a simple example. The insertion of geometry into a BSP is done by recursively traversing the existing nodes n. We follow the traversal path and add new tree nodes:

- in both half-spaces: if node n_i contains the pivot polygon or if $\mathbf{m}$ is parallel to $\mathbf{e}$, where $\mathbf{e}$ is the of intersection segment with the polygon in n_i. Since we want to attach on it we can leave the polygon geometry as is.

- in both half-spaces: if the new polygon is not intersecting with the polygon in node n_i.

- only in the half-space that contains r. So, only the geometry in the half-space of the pivot point can be inserted.

We store r and $\mathbf{m}$ for each node n in the BSP. Furthermore n holds the part of the polygon representing the plane (see Figure 7.9).

The choice of the pivot polygon, the insertion order and insertion direction change the resulting cardboard model significantly because existing planes block parts in the shape for consecutive planes which potentially leads to discarding parts of these planes during insertion (see Figure 7.5). Finding an optimal solution is very complex. Therefore we apply the insertion order optimization described in Section 7.6. Figure 7.10 illustrates with a simple example the influence of the insertion order and the choice of the pivot polygon.

We now have a correctly clipped planar element p. In order to ensure that p is a valid element for our cardboard sculpture we need to check if it is *castable*. In manufacturing, *casting* is generally mentioned in the context of molding. In computational geometry the castability is usually referred to the removal test of a solidified object from a cavity [de Berg et al. 2000]. We borrow that term for an insertion test which we describe next.

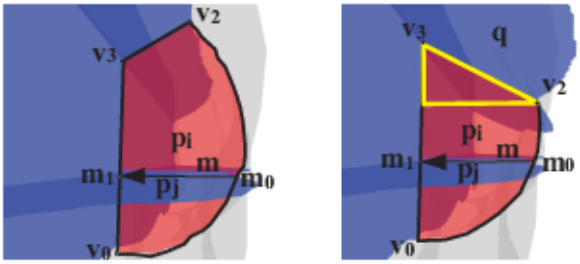


Figure 7.11: Left: This polygon p_i is is castable with respect to insertion direction $\mathbf{m}$. Right: The yellow part of polygon p_i indicates that it is not castable with respect to the insertion direction $\mathbf{m}$.

Castability of Planes

Inserting a polygon p into the BSP results in a clipping against other planes that block the insertion path. This clipping process generates a set of clipping points $\mathbf{c}_k$. Slice p can only be physically moved in direction $\mathbf{m}$ when the *signed* distance of all points $\mathbf{c}_k$ to the insertion direction is monotone along the clipping path (see Figure 7.11). We note that while the values of the distances may depend on the origin of the computation (e.g. the slit), their monotonicity just depends on the direction. Consequently, we can compute the projection vector $\mathbf{c}_k - (\mathbf{m}\ \ \mathbf{c}_k)\mathbf{m}$ of $\mathbf{c}_k$ onto $\mathbf{m}$ and then take the cross product with $\mathbf{m}$ to get a signed distance value:

$$\alpha_k = \mathbf{m} \times (\mathbf{c}_k - (\mathbf{m}\ \ \mathbf{c}_k)\mathbf{m}) = \mathbf{m} \times \mathbf{c}_k \tag{7.5}$$

We check that these values are monotone with respect to k.

Figure 7.11 shows a valid and invalid clipping configuration. The yellow triangle on the right figure indicates a part that violates the condition because the distance to $\mathbf{m}_1$ does not decrease for the clipping points $\mathbf{v}_2$ and the subsequent $\mathbf{v}_3$. We like to stress that casting is, unfortunately, a global condition, in the following sense: if a polygon p cannot be inserted by moving it in direction $\mathbf{m}$ it could potentially cut any other polygon in the model during the movement as well. This means we cannot fix the problem just by local modifications, such as cutting the interfering polygon q in the clipping path as shown in Figure 7.11.

Gathering Shape Features

So far, we defined the base elements of our cardboard sculptures as planar and within the volume of the input object. Therefore, abstractions of curved thin shells result in a large number of elements. To compensate for a possible loss of structural unity we propose a more efficient representation for such shapes. We assign an ϵ-neighborhood and consider all surface points within ϵ distance from the plane to be part of its intersection contour with the input mesh. Intuitively, one can think of it as the projection of these points onto the plane. Figure 7.12 shows a fabricated example where surface points are gathered for the manually selected elephant ears to complete its geometry.

7.5 Fabrication Plan

In order to physically construct the cardboard model, we need an additional insertion test. A cut line is the intersection slit between two polygons. It is divided into two parts, resulting in a halfway-cut for each intersecting polygon. In order to make sure that we can fabricate the polygons with slits true to scale to the thickness of the material it is

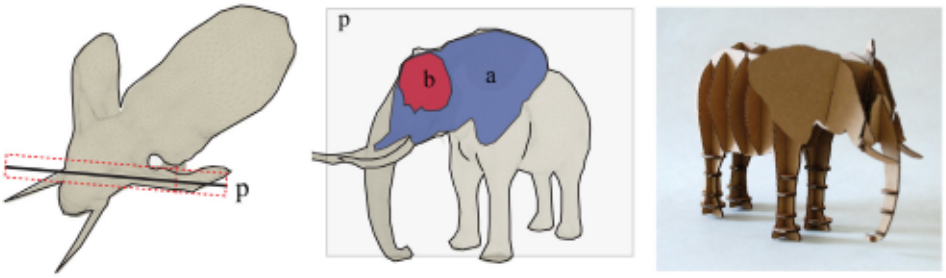


Figure 7.12: Left: Planar slice p through the curved ear of the elephant model viewed from above. Middle: Resulting planar slice with point gathering, a, and without, b. Right: Fabricated result including the complete elephant ear.

important to obey a set of simple conditions that are also tested during the insertion of an element:

- The intersection slit is not intersecting more than once with the contour of polygon p. Otherwise the tile could be disconnected.
- The slits need to have a minimum connection length with the pivot polygon to support physical stability of the tile and the resulting model.

Our algorithm exports a printable 2D fabrication plan including cut line annotations, as shown in Figure 7.4. We add additional annotations to provide step-by-step instructions for assembly of the cardboard sculpture. When physically constructing the cardboard model by hand one simply has to follow the linkage order. Recently, Schwartzburg and Pauly [2013] suggested cutting patterns dependent on the angle of the intersecting planes. That proposes an interesting extension to construction process.

7.6 Optimized Constructability and Evaluation

Once the basic construction constraints are in place, we need to decide on the order of the construction. For a given set of n planes there exists a large number of combinations to assemble the cardboard model. Specifically, there are $n!$ permutations for the order of the planes and the insertion directions given a specific plane order. Additionally, from an aesthetic point of view it is often desirable to construct the cardboard model under the constraint of preserving as much of the available slice area as possible. Some insertion directions allow only a fraction of the original polygon area to be inserted as already discussed. Some of the plane order permutations lead to the same outcome with respect to the remaining area even though their cardboard model tiles and the order of insertions are different.

In order to find an optimal construction, we propose a branch-and-bound approach (see Chapter 4) that utilizes a construction tree as shown in Figure 7.13 with depth n. From the very large combinatorial space we want to choose the configurations that maximize the number of slices p_n 0 1 and the overall score s_n [0 1]. Our objective function can hence be formulated as

$$\mathcal{S} = s_0 p_0 + s_1 p_1 + s_2 p_2 \quad s_n p_n \tag{7.6}$$

which we try to maximize as

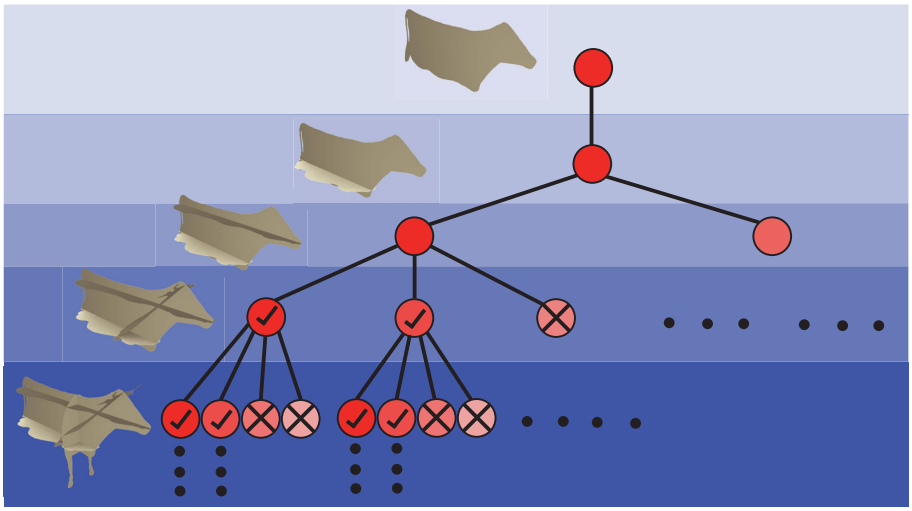


Figure 7.13: Given a new planar element we evaluate all possible insertions and sort the result by its projected area. We use branch-and-bound and proceed our evaluation with the best leaf models. This reduces the number of construction possibilities drastically.

$$\mathcal{S} = \arg\max_{p_n, A_p} \sum p_n s_n(A_p) \tag{7.7}$$

where $s(A_p)$ will be explained shortly.

We are trying to find the best possible insertion for each element but with an increasing number of elements finding the true optimum becomes intractable. The branch-and-bound optimization we employ helps us to avoid evaluating all possible configurations. For this, we iteratively add planes, considering all possible insertions at already existing polygons in the branch-and-bound tree. This leads to a number of intermediate cardboard models as nodes in the construction tree. Depth-first search quickly provides useful bounds and allows pruning of suboptimal solutions. We define our upper bound as the theoretically impossible score, i.e. we can insert all planes without the need to cut away parts. During the automatic construction the optimistic estimate is adapted to the best possible configuration found so far. We discard less significant intermediate construction results, working only with a set of k nodes for further insertion. This reduces our combination space drastically. We found that $k \in [2, 4]$ already gives good results even though it is not guaranteed that we will obtain the optimal solution.

We experimented with different score functions for each plane depending on its area $s(A)$. Note again, the area can change depending on the pivot plane. We also made the score dependent on the distance of the plane to the center of the model. For example, constructing the model starting with planes close to the surface of M can be formulated as

$$s(A) = \alpha A(p) + (1 - \alpha) d^2 \tag{7.8}$$

with d defining the normalized distance from the center of the model and $A(p)$ the normalized area that varies with the insertion direction and the choice of the pivot polygon. The normalization of $A(p)$ is done by dividing by the maximum area of all original polygons before insertion. The distance d is normalized by the radius of the

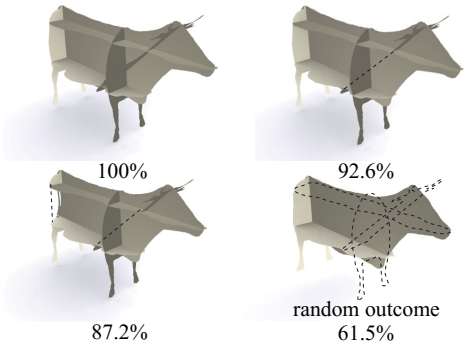


Figure 7.14: Given a set of planes a number possible outcomes and its insertion area in percentage for a specific insertion order is shown. The random outcome represents the median of 100 random insertion orders. Dashed lines indicate parts that could not be added because of a non-optimal insertion order.

bounding sphere of M. The linear interpolation coefficient α is user-defined. For that case, α is set to $\alpha \leq 0.1$. The values mentioned here were used throughout all the results and determined experimentally. The oppositely weighted construction from inside to outside, can be achieved by

$$s(A) = \alpha A(p) + (1 - \alpha)(1 - d^2) \tag{7.9}$$

using the same weighting term $\alpha \leq 0.1$. Maximizing the overall area by sorting the planes weighted strongly by their original plane area is defined as

$$s(A) = \alpha A(p) + (1 - \alpha)d^2 \tag{7.10}$$

with $\alpha \geq 0.1$. We can also define a random construction by setting

$$s(A) = rand() \tag{7.11}$$

Results of the various sorting weights are shown in Figure 7.15 for an axis-aligned plane sampling and different input models. Obviously, we can also just randomly insert planes without optimization. Figure 7.14 shows the outcome of 100 random insertion orders.

A set of possible outcomes of the different score functions is shown in Figure 7.15. In practice, we found that a score function, that rates planes close to or far from the center higher, results in overall more aesthetic constructions for conveying the silhouette. This holds even though the overall area might not be maximized using this approach.

7.7 Results and Discussion

We tested our approach on numerous 3D objects and fabricated several cardboard models. The fabricated objects contain between 7 (Armadillo in Figure 7.17) and 48 (Stanford Bunny in Figure 7.16) elements made out of standard paper, cardboard, plywood or plastic. They were manufactured using an Epilog Zing Lasercutter within 5 minutes and were assembled within 5 to 20 minutes. Side-by-side comparisons of the input model and the real fabricated cardboard models are shown in Figure 7.17 and Figure 7.16. We observed that even with a few polygons the shape of 3D models can be approximated quite well. Our models are thereby extremely low-cost, only require printing and cutting a layout which does not require a laser cutter, and could even be created by children. 3D Printing these objects would probably take ours for a reasonably sized object and cost about two-orders of magnitude more. Our algorithm is not restricted to a small

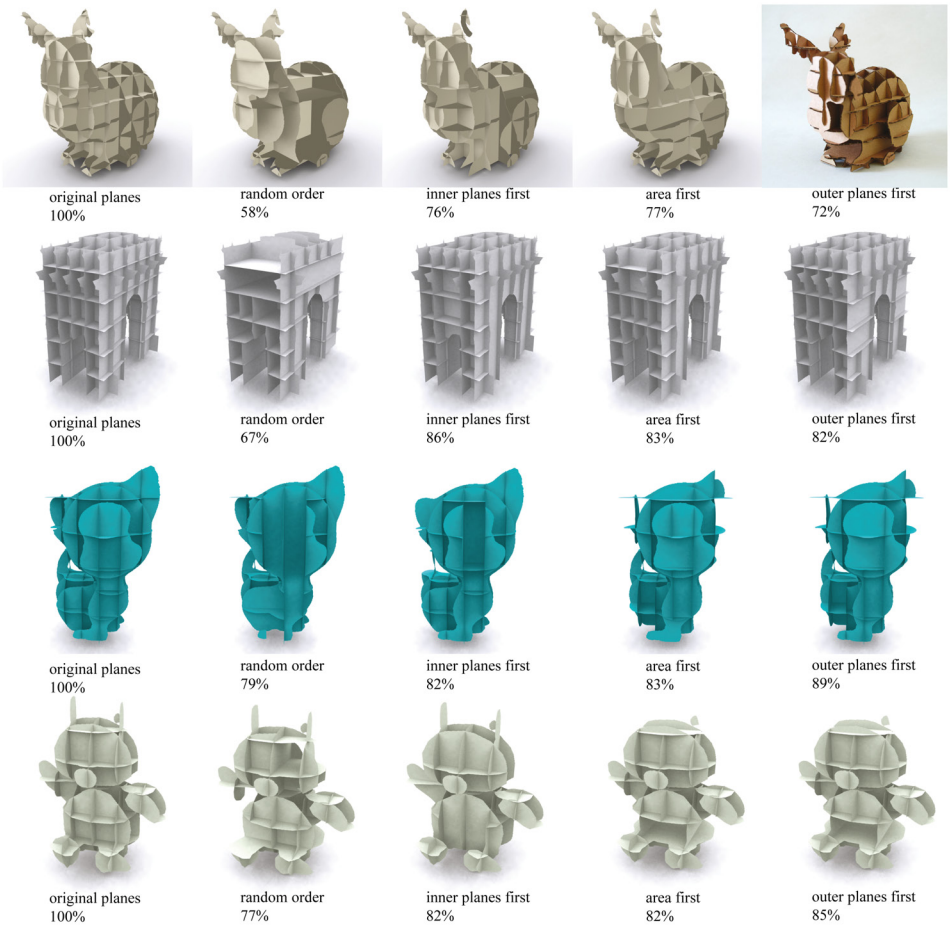


Figure 7.15: Axis-aligned plane sampling in an $8 \times 8 \times 8$ (first row) or $6 \times 6 \times 6$ (last three rows) grid. The order of insertion is determined by the decreasing score of the plane. The score is a weighting of area and the polygons distance from the surface. Thereby we control the construction order from inside to outside. If the score is equal for all polygons the insertion is random. It can be noticed that a construction starting with the larger inner planes reconstructs the silhouette much better with nearly the same overall area.

number of polygons as presented here, although physical assembly becomes impractical at some point.

For all automatically generated results shown in this paper we either used a grid layout of up to $8 \times 8 \times 8$ planes (see Figure 7.16) or the plane quality sampling or a combination of both (see Figure 7.17). For the plane quality evaluation we sampled the plane space with approximately 4000 normal directions and 50 distance offsets. Evaluating the quality of candidate polygons requires the majority of the computation time. For an input model complexity of about 100k triangles, our single-threaded algorithm evaluates about 130 candidates per second, resulting in a total processing time of about one hour on a MacPro using a GeForce GT 120 graphics card. Using the axis-aligned grid sampling we generate a cardboard model within seconds.

As shown in Figure 7.17, our approach for estimating the plane quality is robust and effective. However, it does not take into account higher level design goals or knowledge about the object itself. We therefore also provide a simple interface, allowing the user to indicate preferred samplings as done for the elephant ears in the teaser.

Limitations Currently, our plane quality estimation algorithm does not incorporate high level information about the input object such as salient features, symmetry, or texture. Although finding a general quality estimation that respects the aesthetics of the input and output object might be hard, for future work one might consider combining our representation and approach with work in symmetry detection and enhancement [Pauly et al. 2008] or shape perception.

Figure 7.16: We show the input meshes and the handcrafted cardboard models using axis-aligned grid sampling to find a set of planes. The process for all models is automatic except for the gorilla where we added an additional plane manually.

Figure 7.17: Resulting cardboard sculptures created from a completely automatic two-step process. The first step finds to best planar section based on the plane quality evaluation presented in Section 7.3. In the second step we additional sample axis-aligned planes. The Armadillo is created only of planar sections from the plane quality based sampling.

Chapter 8

Conclusions

This thesis presented novel computational approaches for realising digital objects in the real-world. Our work is embedded in the very active research field of digital fabrication and contributes to the advances in computer graphics. We introduced representations, data structures and application-specific optimizations to manufacture 3D shapes. We demonstrated the benefits of these findings using three consumer-level rapid prototyping applications with improvements for additive and subtractive manufacturing methods. Our work utilized different fabrication methods and tools from 3D printer technologies, over CNC milling machines to laser cutters. We presented methods that could be appealing to a large audience, and might have impact in areas such as architecture and design. Our proposed solutions are a step towards low-cost but widely applicable physical shape abstraction and optimized shape fabrication. We hope that our representations, optimizations and construction algorithms will inspire future work in the area of digital shape fabrication.

8.1 Summary

Intuitive Interfaces for Digital Shape Fabrication

Section 2.5 introduced our approach for an intuitive interface for digital shape fabrication. Such intuitive user interfaces have long been envisioned, [Gross 2007], [Malone and Lipson 2007] and Lipson and Kurman [2012] but our work is the first end-to-end prototype from a user drawn 2D sketch to a 3D printed, personalized object. The focus and main research contribution of this work is an integrated sketch-based pipeline for mass customization.

Optimized Fabrication of Shapes

Nonetheless the large body of research approaches in the rapid manufacturing engineering field [Danjou and Köhler 2009], [Raju et al. 2010], [Hiller and Lipson 2009], [Kulkarni and Dutta 1996], examining layered manufacturing technologies at all scales of resolution and size reveals that there is still room for improvements. In Chapter 5 we focused

on computational improvements for layered manufacturing and their precision by finding suitable fabrication directions. Given an input mesh, we computed an orthogonal basis and considered only the three basis vectors as slice normals (i.e. fabrication directions). Using a variation of branch and bound we optimized a decomposition of the shape along this basis so that each part can be consistently sliced along one of the basis vectors. Because all cuts are orthogonal the model can be easily assembled. In addition, we proposed a graph-cut based segmentation that allows non-planar boundaries to improve the visual quality of the decomposition. In simulation, we showed that this approach is superior to slicing the whole shape in one direction only. The approach also has clear benefits if the shape is larger than the build volume of the available equipment. We demonstrated the practicality of our technique on several models by cutting slices out of acrylic glass or wooden panels with a laser cutter and fused deposition modeling 3d printing.

Binary Space Partition For 3D Shape Manufacturing

Chapter 6 presented a case study for partitioning an input geometry into two parts using a splitting plane and then fabricating each part individually. To our knowledge, this is the first approach that is designed to overcome existing limitations of 3-axis machining such as 3D printing and 3-axis CNC milling.

3D printed geometries that have overhanging parts need support material to strut these overhangs. The amount of support material depends on the projected surface area on the ground plane or subjacent parts of the shape. We showed that it is possible to reduce the usage of support material by up to $\subset 40\%$ over a set of tested 3D objects by dividing the input shape into two parts compared to 3D printing the object in one piece. We also showed that even without splitting our approach can save expensive 3D printing material.

We employed the same computational machinery to improve the creation of 3D objects using 3-axis CNC milling machines. By using the same splitting plane we manufactured 3D shapes very easily and fast. However, this approach introduces fabrication errors. For rapid design prototypes this is typically acceptable.

Fabrication of Abstract Shapes

In Chapter 7 we presented a novel algorithm and representation for designing and fabricating cardboard sculptures from 3D models. The core component and main research contribution of our pipeline is the efficient construction and representation of such models that guarantees that they can be physically assembled. In combination with an optimization and sampling strategy for new elements, planar shape abstraction models are designed in an iterative process. Fabricating such models is extremely low-cost and simple.

8.2 Impact and Outlook

Digital fabrication is a young research area in computer graphics but there is currently a lot of research effort. Many research approaches might have larger impact then can be foreseen now. Our proposed applications focus specifically on the physical 3D output and their corresponding publications have already been cited many times within the last year.

The main impact of our applications that might also have industrial use is the possibility to create rapid design prototypes at different levels of shape precision using different manufacturing devices. Autodesk [2012] already introduced online services to create cardboard models, similar to the ones presented in Chapter 7, and objects composed of layered cardboard and plywood as introduced in Chapter 5, but of lower complexity then in our applications. Our approach can be seen as an improvement and extension for their use cases.

The choice of fabrication directions play an important role in all presented approaches and applications. As discussed, this is due to the limited degrees of freedom of the manufacturing devices and the inherent directional bias in their different axes. Therefore, depending on what should be achieved we show that the object's orientation in the production volume can increase the precision or save support material in a 3D printing process. The key here is to efficiently find a good orientation with small computational effort compared to exhaustive search. Our proposed contributions could be an interesting alternative and extension to various 3D printing software packages.

As also discussed by Luo et al. [2012], we believe that automatically decomposing complex or large geometry into parts that are fabricated individually will move into focus of research and industry applications. Again, our proposed methods show first advances to that problem and could be an interesting addition to digital fabrication software packages.

We hope that this thesis contributes to making digital manufacturing available to the masses.

Bibliography

D. K. Ahn, S. H. Lee, J. I. Song, and S. M. Kwon. Determination of part orientation to minimize post-machining in laminated object manufacturing using genetic algorithm. *Proceedings of the 8th WSEAS international conference on Arti cial intelligence, knowledge engineering and data bases*, pages 337 342, 2009.

M. Alexa and W. Matusik. Reliefs As Images. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4):60:1 60:7, 2010.

M. Alexa and W. Matusik. Images from Self-occlusion. In *Proceedings of the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*, CAe '11, pages 17 24, 2011.

P. Alexander, S. Allen, and D. Dutta. Part orientation and build cost determination in layered manufacturing . *Computer-Aided Design*, 30(5):343 356, 1998.

Anonymous. The printed world. *The Economist*, 398(8720):75 77, 2011.

Anonymous. A third industrial revolution. Special report: manufacturing and innovation. *The Economist*, 403(8781):1 15, 2012.

Autodesk. 123D. http://www.123dapp.com/, 2012.

M. Bächer, B. Bickel, D. L. James, and H. Pfister. Fabricating Articulated Characters from Skinned Meshes. *ACM Trans. Graph. (Proceedings of Siggraph 2012)*, 31(4): 47:1 47:9, 2012.

J. A. Baerentzen and H. Aanaes. Signed Distance Computation Using the Angle Weighted Pseudonormal. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):243 253, 2005.

A. Bagsik and V. Schöppner. Mechanical Properties of Fused Deposition Modeling Parts Manufactured with Ultem*9085. In *Proceedings of ANTEC 2011*, 2011.

I. Baran, P. Keller, D. Bradley, S. Coros, W. Jarosz, D. Nowrouzezahrai, and M. Gross. Manufacturing Layered Attenuators for Multiple Prescribed Shadow Images. *Computer Graphics Forum (Eurographics 2012)*, 31(2pt3):603 610, 2012.

P. Belhumeur, D. Kriegman, and A. Yuille. The bas-relief ambiguity. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 35 (1):1060 1066, 1999.

A. Bermano, I. Baran, M. Alexa, and W. Matusk. ShadowPix: Multiple Images from Self Shadowing. *Computer Graphics Forum (Eurographics 2012)*, 31(2pt3):593 602, 2012.

B. Bickel and M. Alexa. Computational Aspects of Fabrication: Modeling, Design, and 3D Printing . *IEEE Computer Graphics & Applications*, pages 24 25, 2013.

B. Bickel, M. Bächer, M. A. Otaduy, H. R. Lee, H. Pfister, M. Gross, and W. Matusik. Design and Fabrication of Materials with Desired Deformation Behavior. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4):63:1 63:10, 2010.

B. Bickel, P. Kaufmann, M. Skouras, B. Thomaszewski, D. Bradley, T. Beeler, P. Jackson, S. Marschner, W. Matusik, and M. Gross. Physical Face Cloning. *ACM Trans. Graph. (Proceedings of Siggraph 2012)*, 31(4):118:1 118:10, 2012.

M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. uno Levy. *Polygon Mesh Processing*. AK Peters, 2010.

Y. Boykov and V. Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124 1137, 2004.

Y. Boykov, O. Veksler, and R. Zabih. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23 (11):1222 1239, 2001.

B. Bytes. 3D Printed Bricks. http://buildingbytes.info, 2012.

J. Calì, D. A. Calian, C. Amati, R. Kleinberger, A. Steed, J. Kautz, and T. Weyrich. 3D-printing of Non-assembly, Articulated Models. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2012)*, 31(6):130:1 130:8, 2012.

V. Canellidis, V. Dedoussis, N. Mantzouratos, and S. Sofianopoulou. Pre-processing methodology for optimizing stereolithography apparatus build performance. *Computers in Industry*, 57(5):424 436, 2006.

D. Ceylan, W. Li, N. J. Mitra, M. Agrawala, and M. Pauly. Designing and Fabricating Mechanical Automata from Mocap Sequences. *ACM Trans. Graph.*, 32(6):186:1 186:11, 2013.

D. Chen, D. I. W. Levin, P. Didyk, P. Sitthi-Amorn, and W. Matusik. Spec2Fab: a reducer-tuner model for translating specifications to 3D prints. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4):135:1 135:10, 2013.

D. Cohen-Or and A. Kaufman. Fundamentals of Surface Voxelization. *Graphical Models and Image Processing*, 57(6):453 461, 1995.

S. Coros, B. Thomaszewski, G. Noris, S. Sueda, M. Forberg, R. W. Sumner, W. Matusik, and B. Bickel. Computational design of mechanical characters. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4):83:1 83:12, 2013.

B. Cutler and E. Whiting. Constrained Planar Remeshing for Architecture. In *Proceedings of Graphics Interface 2007*, GI '07, pages 11 18, 2007.

S. Danjou and P. Köhler. Determination of Optimal Build Direction for Different Rapid Prototyping Applications. In *Proceedings of the 14th European Forum on Rapid Prototyping*, 2009.

M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, second edition, 2000.

F. De Goes, S. Goldenstein, M. Desbrun, and L. Velho. Technical Section: Exoskeleton: Curve Network Abstraction for 3D Shapes. *Computers & Graphics*, 35(1):112 121, 2011.

D. DeCarlo and M. Stone. Visual Explanations. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering*, NPAR '10, pages 173 178, 2010.

X. Décoret, F. Durand, F. X. Sillion, and J. Dorsey. Billboard Clouds for Extreme Model Simplification. *ACM Trans. Graph. (Proceedings of Siggraph 2003)*, 22(3):689 696, 2003.

E. Demaine, L. Martin, and V. Hart. Computational Balloon Twisting : The Theory of Balloon Polyhedra. In *Proceedings of the 20th Canadian Conference on Computational Geometry*, pages 1 10, 2008.

E. D. Demaine and J. O'Rourke. *Geometric Folding Algorithms. Linkages, Origami, Polyhedra*. Cambridge University Press, 2007.

Y. Dong, J. Wang, F. Pellacini, X. Tong, and B. Guo. Fabricating Spatially-varying Subsurface Scattering. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4): 62:1 62:10, 2010.

M. Eigensatz, M. Kilian, A. Schiftner, N. J. Mitra, H. Pottmann, and M. Pauly. Paneling Architectural Freeform Surfaces. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4):45:1 45:10, 2010.

M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa. Sketch-based shape retrieval. *ACM Trans. Graph. (Proceedings of Siggraph 2012)*, 31(4):31:1 31:10, 2012.

L. R. Ford and D. R. Fulkerson. Maximal Flow through a Network. *Canadian Journal of Mathematics*, 8:399 404, 1956.

C.-W. Fu, C.-F. Lai, Y. He, and D. Cohen-Or. K-set Tilable Surfaces. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4):44:1 44:6, 2010.

H. Fuchs, Z. M. Kedem, and B. F. Naylor. On Visible Surface Generation by a Priori Tree Structures. *Computer Graphics*, 14(3):124 133, 1980.

a. Glassner. Interactive pop-up card design. 1. *IEEE Computer Graphics and Applications*, 22(1):79 86, 2002.

D. Glozman, N. Hassidov, M. Senesh, and M. Shoham. A Self-Propelled Inflatable Earthworm-Like Endoscope Actuated by Single Supply Line. *IEEE Transactions on Biomedical Engineering*, 57(6):1264 1272, 2010.

A. V. Goldberg and R. E. Tarjan. A New Approach to the Maximum-flow Problem. *J. ACM*, 35(4):921 940, 1988.

M. D. Gross. Now More Than Ever : Computational Thinking and a Science of Design. *Special issue of Japanese Society for Science of Design*, pages 1 6, 2007.

M. Hašan, M. Fuchs, W. Matusik, H. Pfister, and S. Rusinkiewicz. Physical Reproduction of Materials with Specified Subsurface Scattering. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4):61:1 61:10, 2010.

K. Hildebrand and M. Alexa. Sketch-Based Pipeline for Mass Customization. *Rethinking Prototyping: Proceedings of the Design Modelling Symposium Berlin 2013*, pages 465 477, 2013.

K. Hildebrand, B. Bickel, and M. Alexa. Crdbrd: Shape Fabrication by Sliding Planar Slices. *Computer Graphics Forum (Eurographics 2012)*, 31(2pt3):583 592, 2012.

K. Hildebrand, B. Bickel, and M. Alexa. Orthogonal Slicing for Additive Manufacturing. *Computers & Graphics*, 37(6):669 675, 2013.

J. Hiller and H. Lipson. Design and analysis of digital materials for physical 3D voxel printing. *Rapid Prototyping Journal*, 15(2):137 149, 2009.

D. S. Hochbaum, editor. *Approximation algorithms for NP-hard problems*. PWS Publishing Co., Boston, MA, USA, 1997.

M. Holroyd, I. Baran, J. Lawrence, and W. Matusik. Computing and fabricating multilayer models. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2011)*, 30(6):187:1 187:8, 2011.

Y. Igarashi, T. Igarashi, and J. Mitani. Beady: Interactive Beadwork Design and Construction. *ACM Trans. Graph.*, 31(4):49:1 49:9, 2012.

I. Ilinkin, R. Janardan, J. Majhi, J. Schwerdt, M. H. M. Smid, and R. D. Sriram. A decomposition-based approach to layered manufacturing. *Comput. Geom.*, (2):117 151, 2002.

B. Khoshnevis, D. Hwang, K.-T. Yao, and Z. Yeh. Mega-scale fabrication by Contour Crafting. *International Journal of Industrial and Systems Engineering*, 1:301 320, 2006.

D. Koberg and J. Bagnall. *The universal traveler: a soft-systems guide to creativity, problem-solving, and the process of reaching goals*. W. Kaufmann, 1976.

V. Kolmogorov and R. Zabih. What Energy Functions can be Minimized via Graph Cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147 159, 2004.

P. Kulkarni and D. Dutta. An accurate slicing procedure for layered manufacturing. *Computer-Aided Design*, 28(9):683 697, 1996.

J. E. Kyprianidis, J. Collomosse, T. Wang, and T. Isenberg. State of the Art: A Taxonomy of Artistic Stylization Techniques for Images and Video. *IEEE Transactions on Visualization and Computer Graphics*, 19(5):866 885, 2013.

Y. Lan, Y. Dong, F. Pellacini, and X. Tong. Bi-scale appearance fabrication. *ACM Trans. Graph.*, 32(4):145:1 145:12, 2013.

A. H. Land and A. G. Doig. An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, 28(3):497 520, 1960.

M. Lau, A. Ohgawara, J. Mitani, and T. Igarashi. Converting 3D Furniture Models to Fabricatable Parts and Connectors. *ACM Trans. Graph. (Proceedings of Siggraph 2011)*, 30(4):85:1 85:6, 2011.

A. Levin, D. Glasner, Y. Xiong, F. Durand, W. Freeman, W. Matusik, and T. Zickler. Fabricating BRDFs at high spatial resolution using wave optics. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4):144:1 144:14, 2013.

X.-Y. Li, C.-H. Shen, S.-S. Huang, T. Ju, and S.-M. Hu. Popup: Automatic Paper Architectures from 3D Models. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4):111:1 111:9, 2010.

X.-Y. Li, T. Ju, Y. Gu, and S.-M. Hu. A Geometric Study of V-style Pop-ups: Theories and Algorithms. *ACM Trans. Graph. (Proceedings of Siggraph 2011)*, 30(4):98:1 98:10, 2011.

Y. Li, J. Yu, K.-l. Ma, and J. Shi. 3D Paper-cut Modeling and Animation. *Comput. Animat. Virtual Worlds*, 18(4-5):395 403, 2007.

C. Lin, Y. Liu, S. Rinker, and H. Yan. DNA tile based self-assembly: building complex nanoarchitectures. *Chemphyschem : a European journal of chemical physics and physical chemistry*, 7(8):1641 1647, 2006.

J. Lin, T. Igarashi, J. Mitani, and G. Saul. A Sketching Interface for Sitting-pose Design. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, SBIM '10, pages 111 118, 2010.

H. Lipson and M. Kurman. *Fabricated: The New World of 3D Printing.* Wiley Press, 2012.

D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner. *Level of Detail for 3D Graphics.* Morgan Kaufmann, 2002.

L. Luo, I. Baran, S. Rusinkiewicz, and W. Matusik. Chopper: Partitioning Models into 3D-printable Parts. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2012)*, 31(6): 129:1 129:9, 2012.

Makerbot. Thingiverse. http://www.thingiverse.com/, 2012.

E. Malone and H. Lipson. Fab@Home: the personal desktop fabricator kit. *Rapid Prototyping Journal*, 13(4):245 255, 2007.

S. H. Masood, W. Rattanawong, and P. Iovenitti. Part Build Orientations Based on Volumetric Error in Fused Deposition Modelling. *International Journal of Advanced Manufacturing Technology*, 16:162 168, 2000.

F. Massarwi, C. Gotsman, and G. Elber. Papercraft Models Using Generalized Cylinders. In *Proceedings of the 15th Paci c Conference on Computer Graphics and Applications*, PG '07, pages 148 157, 2007.

W. Matusik, B. Ajdin, J. Gu, J. Lawrence, H. P. A. Lensch, F. Pellacini, and S. Rusinkiewicz. Printing Spatially-varying Reflectance. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2009)*, 28(5):128:1 128:9, 2009.

J. McCrae, K. Singh, and N. J. Mitra. Slices: A Shape-proxy Based on Planar Sections. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2011)*, 30(6):168:1 168:12, 2011.

J. McCrae, N. J. Mitra, and K. Singh. Surface perception of planar abstractions. *ACM Trans. Appl. Percept.*, 10(3):14:1 14:20, 2013.

R. Mehra, Q. Zhou, J. Long, A. Sheffer, A. Gooch, and N. J. Mitra. Abstraction of Man-made Shapes. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2009)*, 28(5): 137:1 137:10, 2009.

X. Mi, D. DeCarlo, and M. Stone. Abstraction of 2D Shapes in Terms of Parts. In *Proceedings of the 7th International Symposium on Non-Photorealistic Animation and Rendering*, NPAR '09, pages 15 24, 2009.

J. Mitani and H. Suzuki. Making Papercraft Toys from Meshes Using Strip-based Approximate Unfolding. *ACM Trans. Graph. (Proceedings of Siggraph 2004)*, 23(3): 259 263, 2004.

N. J. Mitra and M. Pauly. Shadow Art. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2009)*, 28(5):156:1 156:7, 2009.

Monolite Ltd. d-shape. http://www.d-shape.com, 2012.

S. Mueller, P. Lopes, and P. Baudisch. Interactive construction: interactive fabrication of functional mechanical devices. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, UIST '12, pages 599 606, 2012.

A. S. Nezhad, M. Vatani, F. Barazandeh, and A. R. Rahimi. Determining the optimal build directions in layered manufacturing. *Transactions on Applied and Theoretical Machanics*, 4:185 194, 2009.

M. Papas, W. Jarosz, W. Jakob, S. Rusinkiewicz, W. Matusik, and T. Weyrich. Goal-based Caustics. *Computer Graphics Forum (Eurographics 2011)*, 30(2):503 511, 2011.

M. Papas, C. Regg, W. Jarosz, B. Bickel, P. Jackson, W. Matusik, S. Marschner, and M. Gross. Fabricating translucent materials using continuous pigment mixtures. *ACM Trans. Graph.*, 32(4):146:1 146:12, 2013.

M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. J. Guibas. Discovering Structural Regularity in 3D Geometry. *ACM Trans. Graph. (Proceedings of Siggraph 2008)*, 27 (3):43:1 43:11, 2008.

R. Prévost, E. Whiting, S. Lefebvre, and O. Sorkine-Hornung. Make it stand: balancing shapes for 3D fabrication. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4): 81:1 81:10, 2013.

B. Raju, U. Chandrashekar, D. N. Drakshayani, and K. Chockalingam. Determining the influence of layer thickness for rapid prototyping with stereolithography (SLA) process. *International Journal of Engineering Science and Technology*, 2(7):3199 3205, 2010.

T. Ritschel, K. Smith, M. Ihrke, T. Grosch, K. Myszkowski, and H.-P. Seidel. 3D Unsharp Masking for Scene Coherent Enhancement. *ACM Trans. Graph. (Proceedings of Siggraph 2008)*, 27(3):90:1 90:8, 2008.

B. Rosenberger. Plane Selection Strategies for Shape Abstraction. Master's thesis, TU Berlin, 2012.

P. W. K. Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297 302, 2006.

S. Rusinkiewicz, M. Burns, and D. DeCarlo. Exaggerated Shading for Depicting Shape and Detail. *ACM Trans. Graph. (Proceedings of Siggraph 2006)*, 25(3):1199 1205, 2006.

G. Saul, M. Lau, J. Mitani, and T. Igarashi. SketchChair: an all-in-one chair design system for end users. In *Proceedings of the fth international conference on Tangible, embedded, and embodied interaction*, TEI '11, pages 73 80, 2011.

Y. Schwartzburg and M. Pauly. Fabrication-aware Design with Intersecting Planar Pieces. *Computer Graphics Forum (Eurographics 2013)*, 32(2pt3):317 326, 2013.

I. Shatz, A. Tal, and G. Leifman. Paper craft models from meshes. *The Visual Computer*, 22(9-11):825 834, 2006.

R. F. Shepherd, F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang, and G. M. Whitesides. Multigait soft robot. *Proceedings of the National Academy of Sciences*, 108(51):20400 20403, 2011.

M. Singh and S. Schaefer. Triangle Surfaces with Discrete Equivalence Classes. *ACM Trans. Graph. (Proceedings of Siggraph 2010)*, 29(4):46:1 46:7, 2010.

M. Skouras, B. Thomaszewski, B. Bickel, and M. Gross. Computational Design of Rubber Balloons. *Computer Graphics Forum (Eurographics 2012)*, 31(2pt4):835 844, 2012.

M. Skouras, B. Thomaszewski, S. Coros, B. Bickel, and M. Gross. Computational design of actuated deformable characters. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4):82:1 82:10, 2013.

P. Song, C.-W. Fu, and D. Cohen-Or. Recursive Interlocking Puzzles. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2012)*, 31(6):128:1 128:10, 2012.

W. Song, A. Belyaev, and H.-P. Seidel. Automatic Generation of Bas-reliefs from 3D Shapes. *IEEE International Conference on Shape Modeling and Applications 2007 (SMI 07)*, pages 211 214, 2007.

O. Stava, J. Vanek, B. Benes, N. Carr, and R. M ch. Stress Relief: Improving Structural Strength of 3D Printable Objects. *ACM Trans. Graph. (Proceedings of Siggraph 2012)*, 31(4):48:1 48:11, 2012.

S. Takahashi, H.-Y. Wu, S. H. Saw, C.-C. Lin, and H.-C. Yen. Optimized Topological Surgery for Unfolding 3D Meshes. *Computer Graphics Forum*, 30(7):2077 2086, 2011.

K. Thrimurthulu, P. M. Pandey, and N. V. Reddy. Optimum part deposition orientation in fused deposition modeling. *International Journal of Machine Tools and Manufacture*, 44(6):585 594, 2004.

N. Umetani and R. Schmidt. Cross-sectional Structural Analysis for 3D Printing Optimization. In *SIGGRAPH Asia 2013 Technical Briefs*, SA '13, pages 5:1 5:4, 2013.

N. Umetani, T. Igarashi, and N. J. Mitra. Guided exploration of physically valid shapes for furniture design. *ACM Trans. Graph.*, 31(4):86:1 86:11, 2012.

K. Vidimče, S.-P. Wang, J. Ragan-Kelley, and W. Matusik. OpenFab: a programmable pipeline for multi-material fabrication. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4):136:1 136:12, 2013.

T. Vilbrandt, E. Malone, H. Lipson, and A. Pasko. Heterogeneous Objects Modelling and Applications. chapter Universal Desktop Fabrication, pages 259 284. 2008.

W. Wang, T. Y. Wang, Z. Yang, L. Liu, X. Tong, W. Tong, J. Deng, F. Chen, and X. Liu. Cost-effective Printing of 3D Objects with Skin-frame Structures. *ACM Trans. Graph. (Proceedings of Siggraph Asia 2013)*, 32(6):177:1 177:10, 2013.

G. Wetzstein, D. Lanman, W. Heidrich, and R. Raskar. Layered 3D: Tomographic Image Synthesis for Attenuation-based Light Field and High Dynamic Range Displays. *ACM Trans. Graph. (Proceedings of Siggraph 2011)*, 30(4):95:1 95:12, 2011.

T. Weyrich, J. Deng, C. Barnes, S. Rusinkiewicz, and A. Finkelstein. Digital Bas-relief from 3D Scenes. *ACM Trans. Graph. (Proceedings of Siggraph 2007)*, 26(3), 2007.

T. Weyrich, P. Peers, W. Matusik, and S. Rusinkiewicz. Fabricating Microgeometry for Custom Surface Reflectance. *ACM Trans. Graph. (Proceedings of Siggraph 2009)*, 28 (3):32:1 32:6, 2009.

K. D. D. Willis and A. D. Wilson. InfraStructs: fabricating information inside physical objects for imaging in the terahertz region. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4):138:1 138:10, 2013.

M. Winkelmann. Slicing for Fused Deposition Modeling. Master's thesis, TU Berlin, 2014.

H. Wu, J. Dorsey, and H. Rushmeier. A Sparse Parametric Mixture Model for BTF Compression, Editing and Rendering. *Computer Graphics Forum (Eurographics 2011)*, 30 (2):465 473, 2011.

S. Xin, C.-F. Lai, C.-W. Fu, T.-T. Wong, Y. He, and D. Cohen-Or. Making Burr Puzzles from 3D Models. *ACM Trans. Graph. (Proceedings of Siggraph 2011)*, 30(4): 97:1 97:8, 2011.

S. Xue, X. Chen, J. Dorsey, and H. Rushmeier. Printed Patterns for Enhanced Shape Perception of Papercraft Models. *Computer Graphics Forum (Eurographics 2010)*, 29 (2):625 634, 2010.

Y. Yue, K. Iwasaki, B.-Y. Chen, Y. Dobashi, and T. Nishita. Pixel Art with Refracted Light by Rearrangeable Sticks. *Computer Graphics Forum (Eurographics 2012)*, 31 (2pt3):575 582, 2012.

Q. Zhou, J. Panetta, and D. Zorin. Worst-case structural analysis. *ACM Trans. Graph. (Proceedings of Siggraph 2013)*, 32(4):137:1 137:12, 2013.

H. Zimmer, M. Campen, D. Bommes, and L. Kobbelt. Rationalization of Triangle-Based Point-Folding Structures. *Computer Graphics Forum (Eurographics 2012)*, 31(2pt3): 611 620, 2012.

J. Zimmermann, A. Nealen, and M. Alexa. SilSketch: automated sketch-based editing of surface meshes. In *Proceedings of the 4th Eurographics workshop on Sketch-based interfaces and modeling*, SBIM '07, pages 23 30, 2007.