DISSERTATION

Feature Centric Volume Visualization

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften unter der Leitung von

Ao. Univ. Prof. Dipl.-Ing. Dr. techn. Eduard Gröller
Institut für Computergraphik und Algorithmen
Abteilung für Computergraphik

eingereicht an der Technischen Universität Wien

Fakultät für Informatik

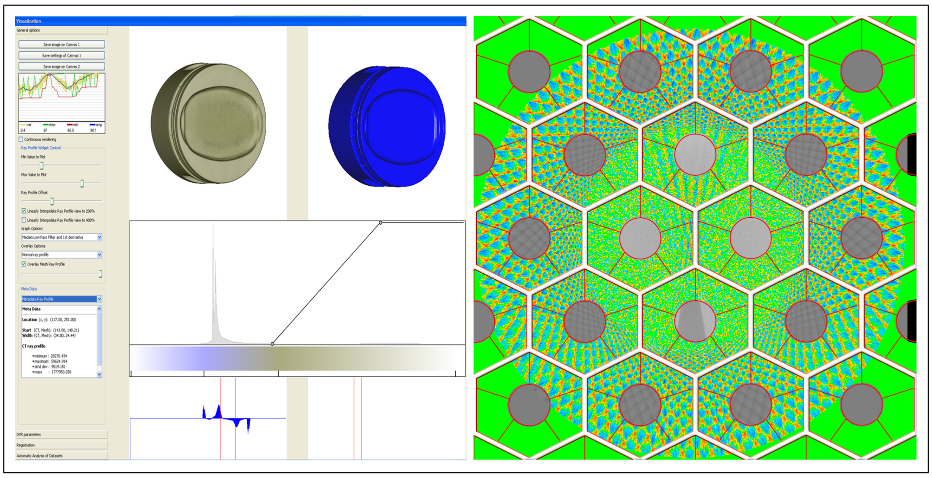
von

Muhammad Muddassir Malik
Matrikelnummer 0527542
Favoritenstraße 9–11
1040 Wien
Österreich

Wien, im Oktober 2009

delivered by
EUROGRAPHICS
DIGITAL LIBRARY
www.eg.org
diglib.eg.org

Feature Centric Volume Visualization



Muhammad Muddassir Malik

Institute of Computer Graphics and Algorithms
Vienna University of Technology, Austria

http://www.cg.tuwien.ac.at/staff/MuhammadMuddassirMalik.html
http://www.cg.tuwien.ac.at/research/publications/2009/malik-thesis

Abstract

This thesis presents techniques and algorithms for the effective exploration of volumetric datasets. The visualization techniques are designed to focus on user specified features of interest. The proposed techniques are grouped into four chapters namely feature peeling, computation and visualization of fabrication artifacts, locally adaptive marching cubes, and comparative visualization for parameter studies of dataset series. The presented methods enable the user to efficiently explore the volumetric dataset for features of interest.

Feature peeling is a novel rendering algorithm that analyzes ray profiles along lines of sight. The profiles are subdivided according to encountered peaks and valleys at so called transition points. The sensitivity of these transition points is calibrated via two thresholds. The slope threshold is based on the magnitude of a peak following a valley, while the peeling threshold measures the depth of the transition point relative to the neighboring rays. This technique separates the dataset into a number of feature layers.

Fabrication artifacts are of prime importance for quality control engineers for first part inspection of industrial components. Techniques are presented in this thesis to measure fabrication artifacts through direct comparison of a reference CAD model with the corresponding industrial 3D X-ray computed tomography volume. Information from the CAD model is used to locate corresponding points in the volume data. Then various comparison metrics are computed to measure differences (fabrication artifacts) between the CAD model and the volumetric dataset. The comparison metrics are classified as either geometry-driven comparison techniques or visual-driven comparison techniques.

The locally adaptive marching cubes algorithm is a modification of the marching cubes algorithm where instead of a global iso-value, each grid point has its own iso-value. This defines an iso-value field, which modifies the case identification process in the algorithm. An iso-value field enables the algorithm to correct biases within the dataset like low frequency noise, contrast drifts, local density variations, and other artifacts introduced by the measurement process. It can also be used for blending between different iso-surfaces (e.g., skin, and bone in a medical dataset).

Comparative visualization techniques are proposed to carry out parameter studies for the special application area of dimensional measurement using industrial 3D X-ray computed tomography. A dataset series is generated by scanning a specimen multiple times by varying parameters of the scanning device. A high resolution series is explored using a planar reformatting based visualization system. A multi-image view and an edge explorer are proposed for comparing and visualizing gray values and edges of several datasets simultaneously. For fast data retrieval and convenient usability the datasets are bricked and efficient data structures are used.

Kurzfassung

Diese Arbeit beschäftigt sich mit Algorithmen und Techniken zur effektiven Exploration von volumetrischen Daten. Die Visualisierungstechniken sind mit dem Fokus auf benutzerdefinierte Merkmale entwickelt worden. Die vorgestellten Techniken sind in vier Kapitel eingeteilt: Feature Peeling, Berechnung und Darstellung von Produktionsfehlern, lokal adaptives Marching Cubes Verfahren, und vergleichende Visualisierung für Parameterstudien von Datensatzreihen.

Feature Peeling ist ein neuer Darstellungsalgorithmus, der auf der Analyse von Sichtstrahlprofilen basiert. Die Strahlenprofile werden bei sogenannten Übergangspunkten also bei Hochpunkten und Tiefpunkten unterteilt. Die Empfindlichkeit dieser Übergangspunkte wird mittels zweier Schwellwerte kalibriert. Der Steigungsschwellwert basiert auf der Länge des Vektors, der einen Tiefpunkt und den darauffolgenden Hochpunkt verbindet. Der Peeling-Schwellwert bestimmt die Tiefe des Übergangspunktes im Verhältnis zu den Übergangspunkten der benachbarten Sichtstrahlen. Durch diese Technik kann ein Datensatz in eine Reihe von Merkmalsebenen zerlegt werden.

Produktionsfehler sind für die Qualitätskontrolle bei der Erstinspektion einer industriellen Komponente von primärer Wichtigkeit. In dieser Arbeit werden Techniken zum direkten Vergleich eines CAD-Modells mit einem 3D Röntgen Computertomographie-Scan präsentiert. Informationen aus dem CAD-Modell werden verwendet, um korrespondierende Punkte im Volumen zu identifizieren. Anschließend werden verschiedene Vergleichsmetriken ausgewertet, um Unterschiede (Produktionsfehler) zwischen dem CAD-Modell und dem Volumen zu messen. Die Metriken können entweder global (geometry-driven) oder lokal (visual-driven) ausgewertet und angezeigt werden.

Der lokal adaptive Marching Cubes Algorithmus ist eine Modifikation des ursprünglichen Marching Cubes Verfahrens, bei dem anstatt eines globalen Isowertes ein Isowert pro Voxel verwendet wird. Das dadurch definierte Isowertfeld modifiziert den Klassifikationsschritt des ursprünglichen Algorithmus. Das Isowertfeld ermöglicht die Korrektur von systematischen Messabweichungen im Datensatz, wie zum Beispiel Rauschen mit niedriger Frequenz, Kontrastverschiebungen, lokale Dichteveränderungen, etc. Außerdem kann der Algorithmus verwendet werden, um verschiedene Isoflächen (z.B. Haut und Knochen in einem medizinischen Datensatz) miteinander zu verschmelzen.

Für den speziellen Anwendungsbereich der 3D Messung mittels industriellem Röntgen Computertomographen werden im Rahmen dieser Arbeit vergleichende Visualisierungstechniken vorgeschlagen. Durch das wiederholte Abtasten eines Messkörpers mit veränderten Messparametern wird eine Datensatzreihe erzeugt. Im Rahmen dieser Arbeit wurde ein Visualisierungssystem entwickelt, das sich der planaren Reformattierung bedient. Dieses System kann verwendet werden um eine hoch aufgelöste Datensatzreihe zu explorieren. Eine Mehrbildansicht und ein Werkzeng zur Kanteninspektion werden zur Verfügung gestellt, um mehrere Datensätze gleichzeitig miteinander vergleichen zu können. Die Datensätze sind in Blöcke unterteilt, um mit effizienten Datenstrukturen schnell auf die Volumendaten zugreifen zu können und eine hohe Benutzerfreundlichkeit zu gewährleisten.

Contents

Preface **xi**

1 Introduction **1**
- 1.1 Computed Tomography 2
  - 1.1.1 General Visualization Techniques for CT 3
  - 1.1.2 Application Specific Visualization Techniques for CT 4
- 1.2 Magnetic Resonance Imaging 4
- 1.3 Scope of the Thesis 7

2 Feature Peeling **11**
- 2.1 Related Work 12
- 2.2 Feature Peeling 13
  - 2.2.1 Locating Transition Points 14
  - 2.2.2 Relevance Across Neighboring Rays 15
- 2.3 Implementation and Performance 19
- 2.4 Results 22
- 2.5 Conclusion 23

3 Computation and Visualization of Fabrication Artifacts **29**
- 3.1 Related Work 31
- 3.2 Computation & Visualization 32
  - 3.2.1 Geometry-Driven Comparison 33
  - 3.2.2 Visual-Driven Comparison 38
- 3.3 Results 40
  - 3.3.1 Artifacts and Errors 40
  - 3.3.2 Performance and Evaluation 44
- 3.4 Conclusion 45

4 Locally Adaptive Marching Cubes **47**
- 4.1 Related Work 48
- 4.2 Locally Adaptive Marching Cubes 49
  - 4.2.1 The Marching Cubes Algorithm 49
  - 4.2.2 The Modified Algorithm: Support of Iso-Value Fields 49
- 4.3 Implementation 51
- 4.4 Results 52
  - 4.4.1 Simple Objects 52
  - 4.4.2 Correcting the Iso-surface of a Biased Volumetric Dataset . . 53
  - 4.4.3 Blending Between Iso-surfaces 55
  - 4.4.4 Surface in the Interior of a Cell 55

4.5 Conclusion 55

5 Comparative Visualization for Parameter Studies of Dataset Series 59
5.1 Related Work 61
5.2 Comparative Visualization 62
5.2.1 Multi-image View 65
5.2.2 Edge Explorer 71
5.3 User Feedback and Results 74
5.3.1 First Evaluation - Workflow Comparison 75
5.3.2 Second Evaluation - Multi-image View vs. Attribute Blocks . 76
5.4 Implementation and Performance 80
5.5 Conclusion 82

6 Summary and Conclusion 83

A Questionnaire for First Evaluation 85
A.1 General Questions 85
A.2 Tasks 86
A.3 General Questions about the Technique 87

B Questionnaire for Second Evaluation 89
B.1 General Questions 89
B.2 Tasks 90
B.3 General Questions about the Two Techniques 92

Bibliography 95

Curriculum Vitae 103

> People’s hearts are like wild animals. They attach themselves to those who love and train them.
>
> Ali ibn Abi Talib

Preface

This thesis is the final work of my Ph.D. study at the Institute of Computer Graphics and Algorithms, Faculty of Informatics, Vienna University of Technology. It serves as a documentation of my research work between December 2005 and October 2009. The research has been funded in part by the Higher Education Commission of Pakistan (HEC).

I dedicate my thesis to my parents and family. Their love, continuous help in education, and prayers made it possible for me to complete the Doctoral studies. I am grateful to my supervisor M. Eduard Gröller for his immense scientific guidance, encouragement, and providing a wonderful working environment. I like to thank Torsten Möller, Christoph Heinzl, and Michael Reiter for scientific discussions, joint research, and collaboration in drafting scientific publications.

The Vis-group at our institute is home to research as well as fun and traditions. I enjoyed the fruitful scientific discussions with my colleagues at work (Andrej Varchola, Erald Vuçini, Martin Haidacher, Maurice Termeer, Peter Kohlmann, Peter Rautek, and Stefan Bruckner) and around the globe (Danial Patel, Ivan Viola, Jean-Paul Balabanian, and Matej Mlejnek). I will always remember fair walking up the stairs, chanting Visibile facimvs qvod ceteri non possvnt at meetings, and whistling to remember departing guests and friends.

The carp, head angiography, MRI head, and hand datasets used in this thesis are courtesy of the University of Erlangen, Özlem Gürvit from the Institute for Neuroradiology, Siemens Medical Systems, and Tiani Medgraph respectively. The present dataset and the rest of the industrial datasets were scanned at the Upper Austria University of Applied Sciences, Wels Campus.

Muhammad Muddassir Malik
Vienna, Austria, October 2009

> 1961: The purpose of computing is insight, not numbers.
> 1997: The purpose of computing numbers is not yet in sight.
>
> Richard Wesley Hamming

1

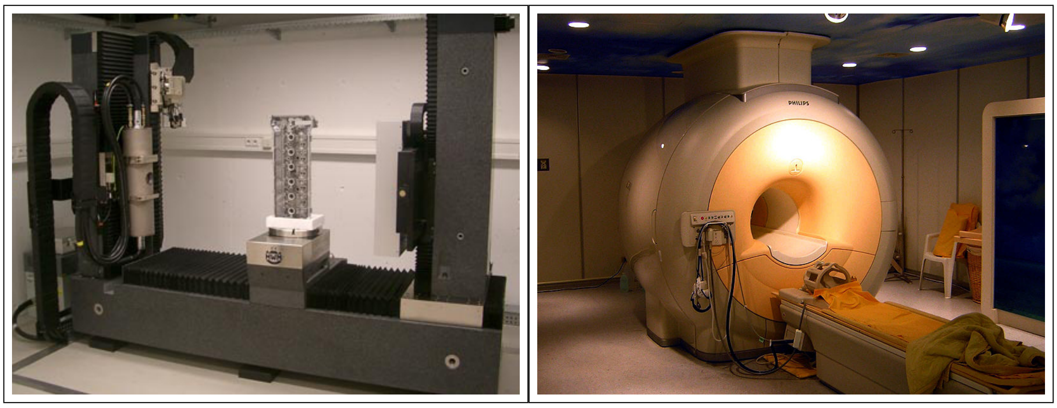
Introduction

The concept of X-ray imaging was introduced by an Italian radiologist Alessandro Vallebona in the early twentieth century [82]. This was one of the first methods proposed to look inside opaque objects. Plenty of machines such as computed tomography (CT), magnetic resonance imaging (MRI), positron emission tomography (PET), and ultrasound have been invented since then to help professionals plan surgery, perform nondestructive testing (NDT) of industrial components, search for oil, locate fish schools and examine the proper development of fetuses etc. All the above devices produce a volumetric dataset as an output as opposed to a 2D image generated from X-ray imaging.

These machines are used on a daily basis in the fields of medicine and industrial manufacturing. Radiologists and doctors use them for *diagnosis* like finding defective positioning of a body part, *treatment planning* like marking regions for therapy, and *intra-operative support* like monitoring life critical structures etc [59]. Quality control engineers capitalize on them by performing NDT, first part inspection, and measurement of internal features.

In most cases, the output of a scanning device is a raw dataset which is composed of scalar values. Each scalar value describes the specimen at a distinct spatial location in terms of the scanning technique employed. For example, a medical CT uses Hounsfield units to define the densities of the specimen, MRI measures the content of hydrogen atoms and ultrasound records the reflected echo of sound waves to calculate distances and densities of the tissue. A 3D dataset produced by such scanners can be very large and complex to be of any value in its raw format. The purpose of scanning is not to produce meaningful numbers in a high quantity but to be able to get insight into the specimen and to be able to deduce reliable results efficiently.

Communication of information to a target user can be carried out efficiently and in a focused manner by harnessing the qualities of the perceptual system of the user. Human beings have the capacity to combine vision, visual clues, and multiple viewpoints into a single 3D image [69]. The pre-attentive property of vision is the processing of displayed data even before a focused attention is



(a) 3DCT (b) MRI

Figure 1.1: (a) Rayscan 250XE industrial 3D X-ray computed tomography device [74]. (b) three tesla clinical magnetic resonance imaging scanner [81].

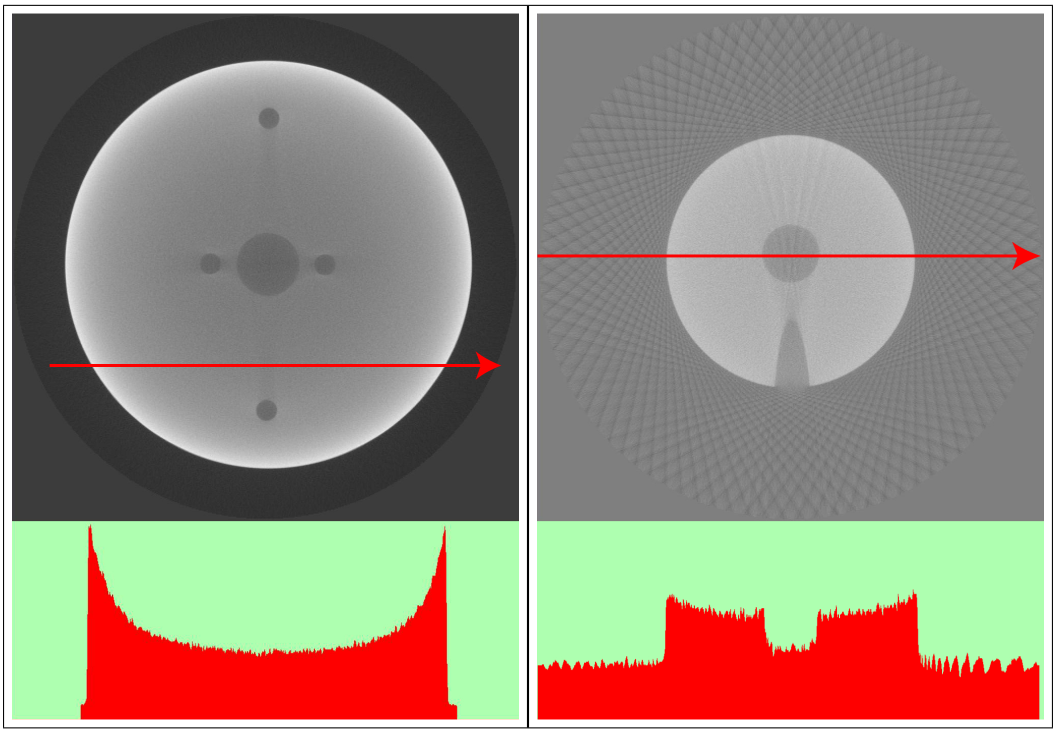
achieved. A visualization task capable of utilizing pre-attentive vision can be solved in merely 200 to 250 milliseconds. A large amount of visual information can be processed by the brain instantly [24]. Visualization techniques can exploit the human perceptual system for better and faster communication of voluminous data.

Visualization techniques can be applied to transform large and complex datasets into clear and easy to understand expressive images. The user may have the ability to control the display parameters and to interact with the rendered images. Visualization can reduce the time taken to understand data, to find trends, and to extract valuable conclusions from the dataset. We will discuss various visualization techniques for CT and MRI datasets in the rest of the thesis.

1.1 Computed Tomography

The first commercial CT machine was produced in 1972 by considerably enhancing the X-ray imaging methodology [82]. A CT produces a series of 2D X-ray attenuation measurements, which are used to mathematically reconstruct a 3D grid of scalar values (voxels) corresponding to the spatial densities of the specimen. The voxel values in CT are generally calibrated and are called Hounsfield units or density values. In case of an industrial 3D X-ray computed tomography (3DCT) scanner the values are not calibrated and are thus called gray values (figure 1.1(a)). The energy of the X-rays used in CT is directly related to their penetration power and their harmful affects on the patients. 3DCT scanners use higher energy X-rays and produce higher resolution volumetric datasets compared to CT machines engineered for scanning patients.

Volumetric datasets obtained from a CT are prone to a range of artifacts such as contrast drifts, beam hardening, streaking artifacts, scattered radiation, and ring artifacts. A contrast drift is a slow change in the density values of a given material during the course of scanning. Beam hardening is a common artifact in



(a) Beam hardening

(b) Streaking artifacts

Figure 1.2: (a) shows the beam hardening artifact. The gray-value profile of the red ray is plotted below the image. The gray values are higher near the edges and decrease with an increase in penetration length. (b) streaking artifacts are visible as bands of gray values inside and outside the material. The dataset was recorded by taking 90 projections in a single 360 degree turn of the rotary plate.

3DCT. Materials with a greater ability to absorb X-rays result in lower gray values in the center of the material (figure 1.2(a)).

Streaking artifacts are bands of gray values produced due to, e.g., insufficient sampling of the specimen. Figure 1.2(b) shows a slice of a dataset affected by streaking artifacts. Scattered radiation is caused by the deflection of X-rays. Neighboring detector pixels of the CT can be inhomogeneous due to the rotation of the dataset on a rotary plate. This results in ring artifacts.

1.1.1 General Visualization Techniques for CT

Over the years a lot of visualization techniques have been proposed to visualize CT data. Initially the aim was to develop a basic system that can visually represent large volumetric datasets. This was achieved to a considerable degree through direct volume rendering (DVR) and multi-planar reformatting (MPR), i.e, 2D planar slices of the data taken at arbitrary angles. Afterwards some researchers started working on application specific visualization algorithms. The intention was to further automate 3D volume exploration and processing and to make it robust.

1.1.2 Application Specific Visualization Techniques for CT

DVR renders a volumetric dataset by modulating the scalar values with a user specified transfer function (figure 1.3(a)). The user can interact with the transfer function and the updated image appears in the viewport. DVR is a basic technique for volume visualization. It is neither aware of the features inside the dataset, nor does it take into account the application specific goals of performing volume rendering. It treats a dataset as a cluster of scalar values, no matter if the dataset is a human brain, a car engine, or a tooth.

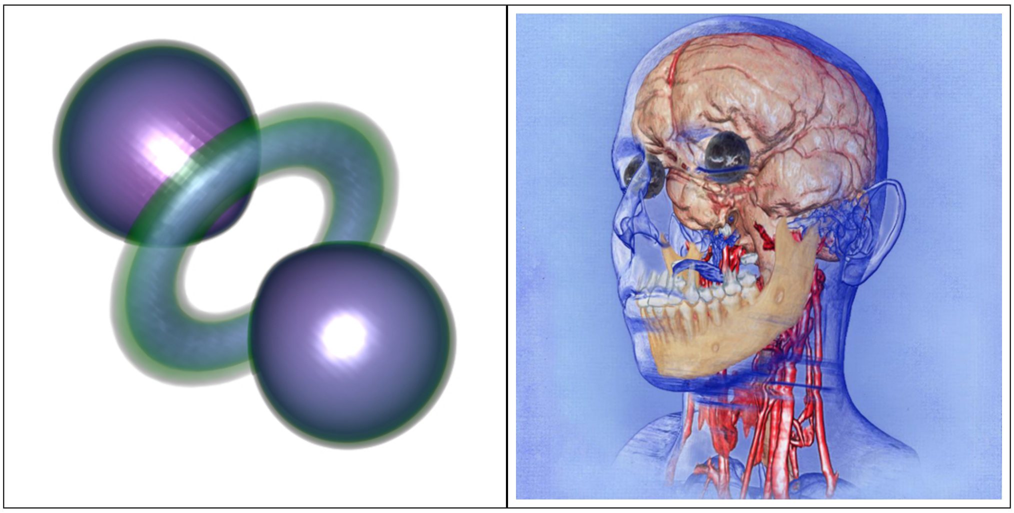
The parameters of DVR change according to the type of the dataset and the goals of the user. A doctor operating upon a complex fracture is primarily interested to view the bones. Skin and arteries far away from the fracture are irrelevant. A quality control engineer may be only interested in ensuring that the piston of an engine does not have any cracks. It is important to propose innovative visualization techniques and algorithms that are tailored for specific application areas.

Techniques like cut-away views, ghosted displays and exploded views are proposed to adapt DVR to importance driven visualization [8, 77]. A good viewpoint for a feature of interest can be automatically evaluated [40, 76]. Users can store, visualize, and compare their workflows and the rendered images [67]. The production quality of an industrial part is evaluated by comparing the surface model extracted from its 3DCT volume with the corresponding computer aided design model (CAD) [28] (figure 1.3(c)). Datasets from dual energy 3DCT can be combined to minimize artifacts and to maximize contrast between features [26]. Figure 1.3(b) shows a rendering of a segmented human head using style transfer functions [9]. In this thesis we propose four innovative visualization systems for CT datasets. The presented algorithms visualize the volume data on the basis of the features that are of interest for the user. Specific user goals were considered while designing the techniques (section 1.3).

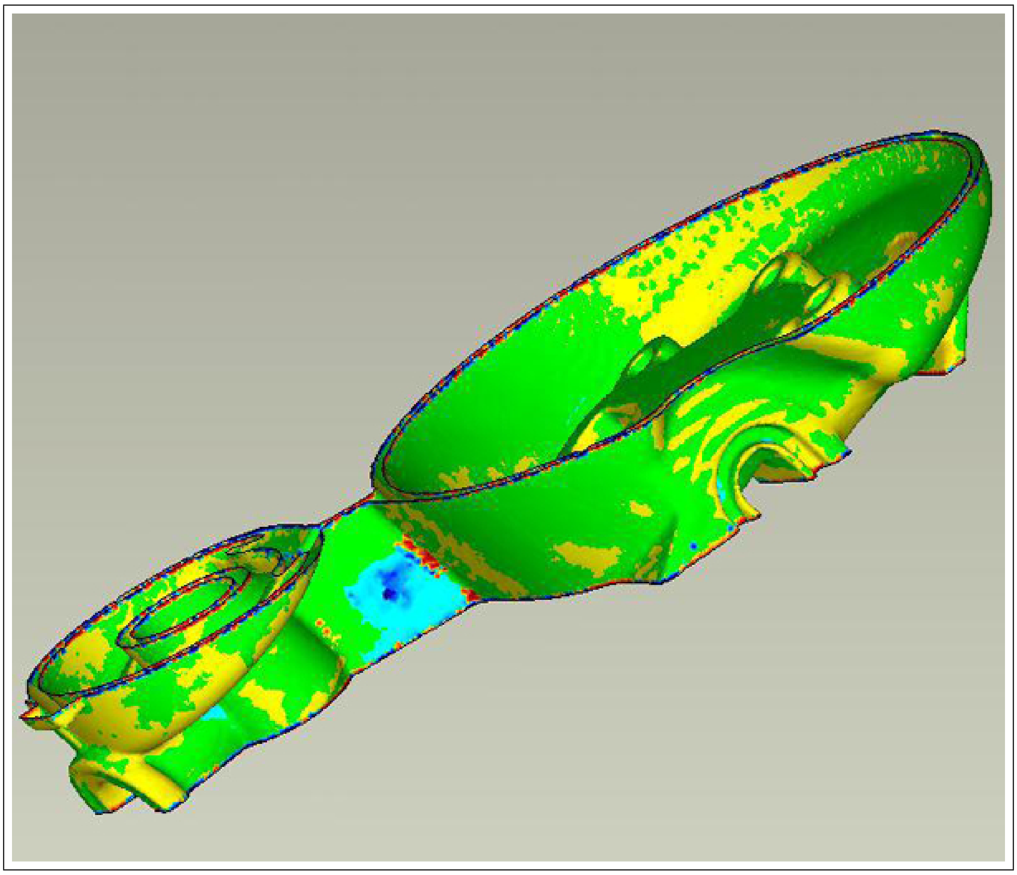
1.2 Magnetic Resonance Imaging

The origin of MRI can be traced back to 1930 when Isidor Rabin discovered the fundamentals of the magnetic behavior of protons. The first MRI image was produced in 1973 [81]. MRI uses a magnetic field to measure the hydrogen content of a specimen. Due to the use of a magnetic field in the scanning process, MRI can only be employed to scan objects that cannot be magnetized and that contain hydrogen. MRI is primarily operated in the medical field (figure 1.1(b)).

MRI datasets have high spatial resolution but they are prone to high frequency noise and low frequency noise. High frequency noise affects image clarity and low frequency noise alters the signal intensity globally. The noise makes it difficult to apply contemporary 3D visualization techniques like DVR on MRI data. Multiplanar reformatting (MPR) is a commonly used method to inspect MRI data for diagnosis.



(a) hydrogen dataset (b) CT head



(c) 3DCT specimen

Figure 1.3: (a) DVR of the hydrogen dataset. (b) feature weighted rendering of a human head using a multi-dimensional style transfer function [9]. (c) comparison between a surface model and a CAD model of a 3DCT specimen [28].

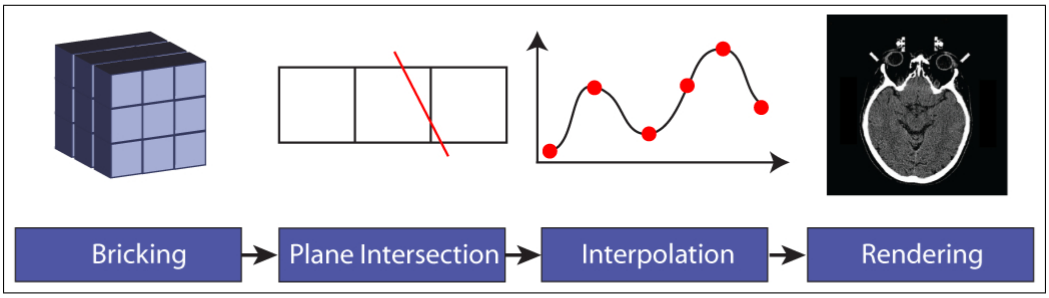
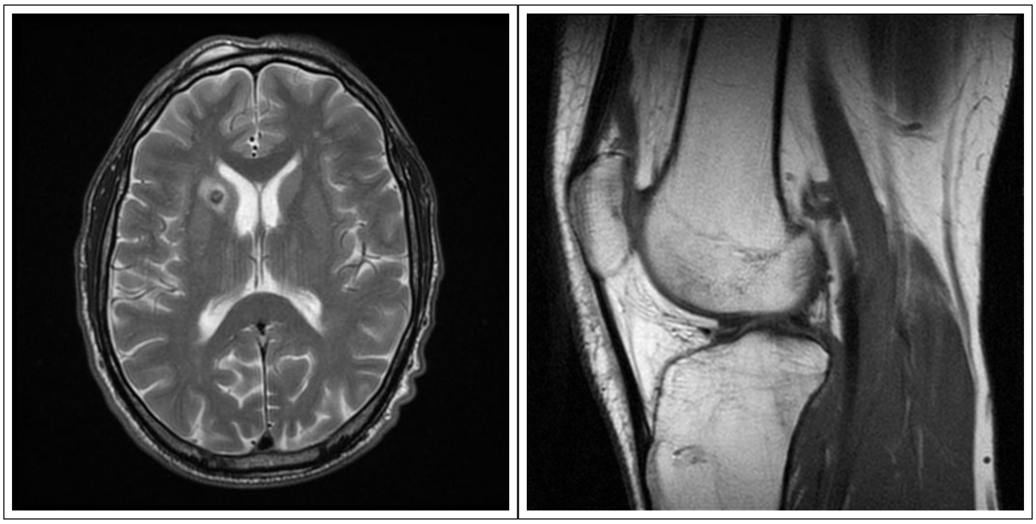


Figure 1.4: MPR pipeline. The intersection between a bricked volume and a cutting plane is calculated. Later interpolation is performed to generate images.



(a) Brain (b) Knee

Figure 1.5: (a) and (b) show MRI slices from the brain and the knee respectively [81].

Visualization Techniques for MRI:

Generally, in the first stage, MPR divides a monolithic volume into bricks (figure 1.4). The bricks which intersect a user placed cutting plane are loaded into the main memory and then an interpolation method is employed to calculate an image. Figures 1.5(a) and (b) display slices from an MRI brain dataset and an MRI knee dataset respectively.

Rezk-Salama and Kolb present an opacity peeling technique for visualizing MRI datasets [64]. They argue that DVR for MRI requires classification due to overlapping scalar values assigned to different tissues. Classification is a time consuming process that may require considerable user interaction. Their method is automated and is tailored for time critical scenarios like intra-operative support. Opacity peeling uses opacity to classify datasets. Layers of data are peeled off depending on the accumulated opacity along the rays in a DVR environment. Opacity peeling is a view dependent technique. Figure 1.6 shows an MRI head dataset rendered using the opacity peeling technique. Bone and skin are peeled off to reveal the brain.

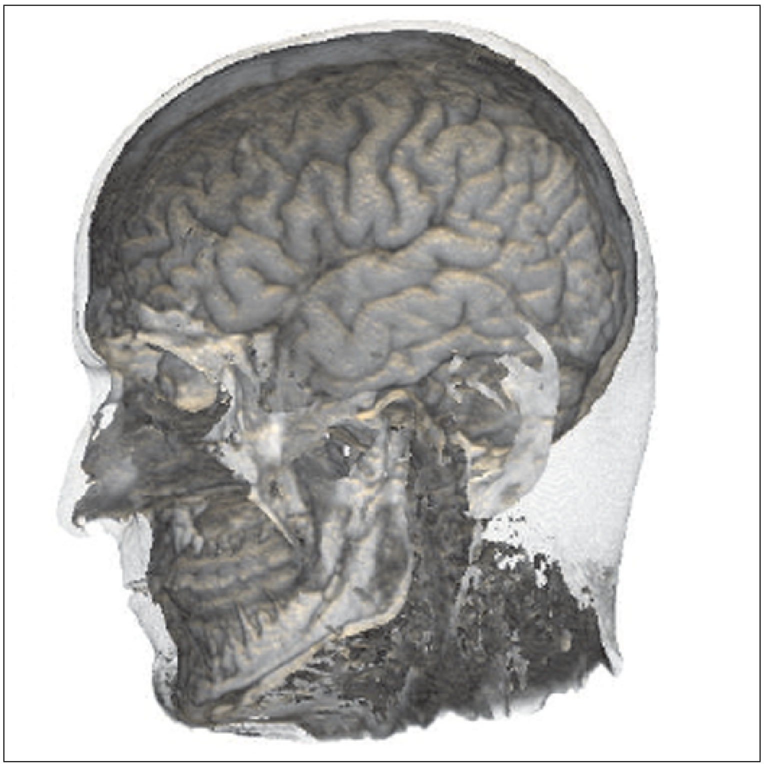


Figure 1.6: Brain uncovered from an MRI head dataset using opacity peeling [64].

1.3 Scope of the Thesis

This work is an attempt to strengthen feature specific volume visualization. A feature is a property or an object in a dataset which is of prime importance for analysis and decision making. The proposed techniques are designed to meet the following guidelines:

- Features of interest are identified in an application area.
- Innovative visualization algorithms are proposed.
- Parameters which require user interaction are minimized.
- Utility and usability of the presented algorithms is established.

In chapter 2 feature peeling is proposed to peel off layers of data from MRI and CT datasets to reveal underlying structures [48]. Layers are identified by casting view dependent rays and processing the encountered scalar values. The visualization of MRI data is enhanced. The technique also speeds up the exploration of CT and MRI datasets. The user does not have to perform the laborious task of specifying a transfer function. Instead just two thresholds need to be specified. The algorithm is compared with the opacity peeling technique [64]. Figure 1.7 shows an MRI head dataset with skin and bone peeled off. The brain surface is clearly visible.

3DCT datasets are often used for first part inspection of industrial components. The inspection is typically carried out by extracting a surface of the volume data and

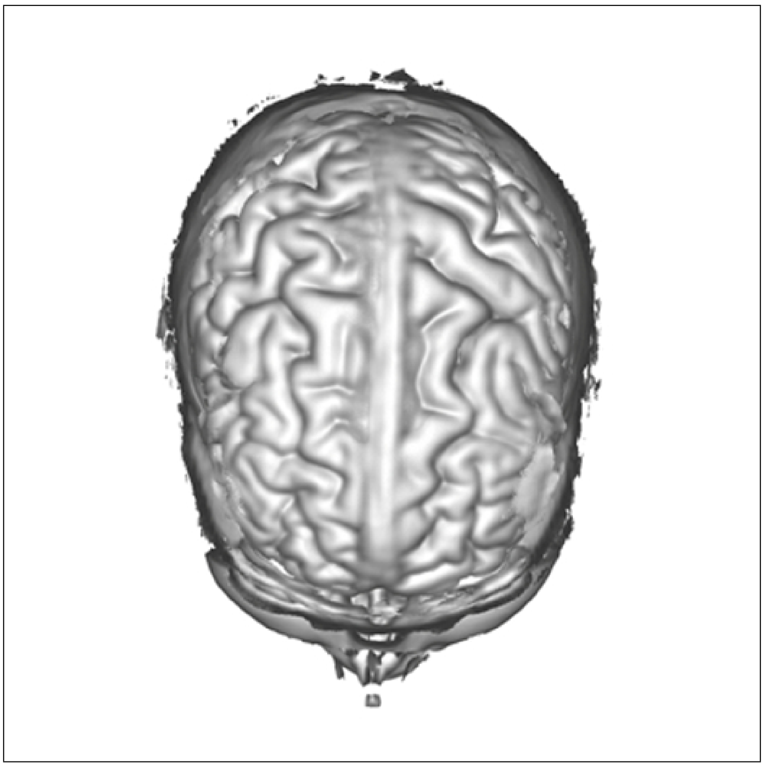


Figure 1.7: A rendering of a human head scanned with MRI. The skin and bone is peeled off and the brain surface is clearly visible.

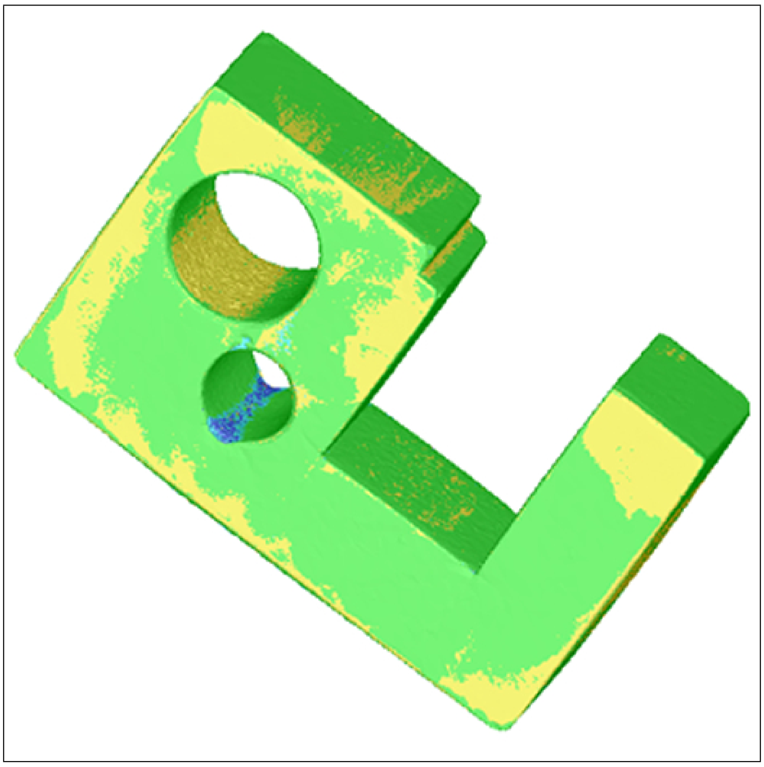


Figure 1.8: An industrial specimen color coded to show differences between its 3DCT volume and the corresponding CAD model.

then comparing the extracted surface model with the CAD model of the specimen. Fabrication artifacts are the most interesting features in such a comparison. They directly influence the production quality of an industrial component. The two step process of comparison is error prone and time consuming (chapter 3). A direct comparison between a CAD model and the corresponding 3DCT volume can be performed interactively [47]. It avoids artifacts introduced by the surface extraction process and it reduces the number of user specified parameters. Figure 1.8 shows a surface model of an industrial component. The dataset is color coded after performing a direct comparison between its 3DCT volume and the corresponding CAD model.

The surface extraction from a volumetric dataset is an error prone task. The errors are introduced either due to the extraction process or due to bias in the dataset. Surface extraction algorithms have considerably improved since the marching cubes algorithm [44] was developed in 1987. Not enough work, however, has been done to remove bias in the datasets. Bias in the scalar values might occur due to various CT artifacts and can severely affect the shape of various features in the specimen. The surface model of a specimen can be enhanced by using iso-value fields (chapter 4). The techniques presented in this thesis are useful in various other scenarios as well like smooth blending of multiple surfaces [20]. Enamel and dentine are both parts of a tooth but have different densities. In figure 1.9 iso-surfaces of enamel and dentine are both stitched together into one triangular mesh using an iso-value field.

Comparison of datasets is a usual task for professionals working with 3DCT. The comparison is performed to inspect and compare artifacts in different datasets (chapter 5). Artifacts are features of interest in such an application scenario. The users want a system which is able to highlight artifacts efficiently and which can handle multiple large datasets [46, 63]. Two studies are presented to show the affect of the proposed system on the workflow of domain specialists. Four slices from different datasets are compared in figure 1.10. The color coding displays the relative difference between the scalar values of the datasets in the sectors and the reference dataset displayed in the central circle (220 kV).

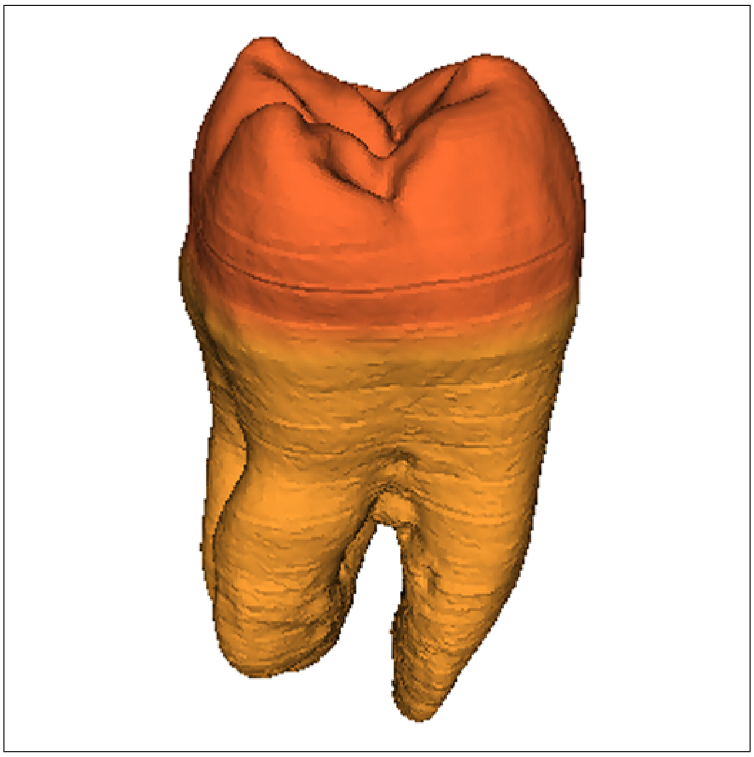


Figure 1.9: Locally adaptive marching cubes is utilized to generate a surface model of a human tooth dataset.

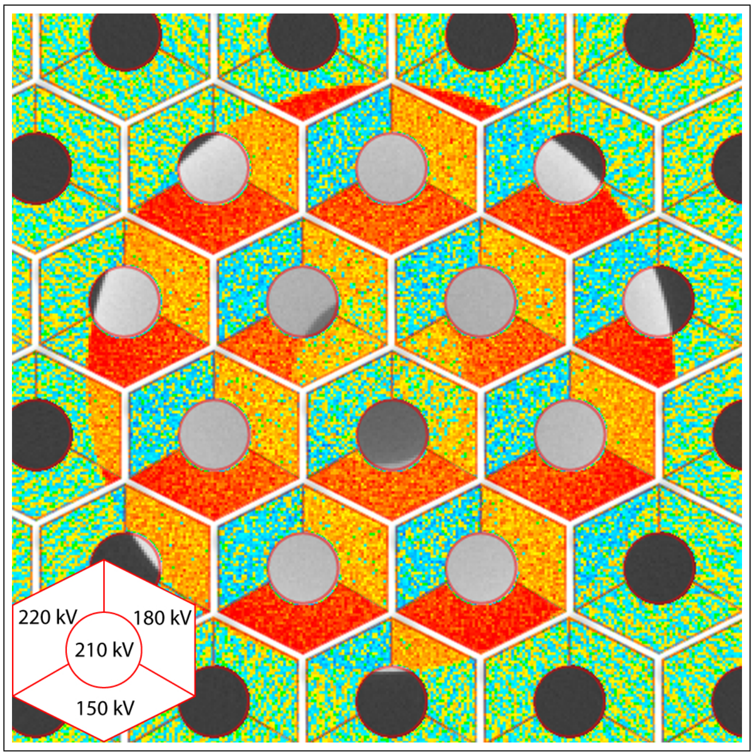


Figure 1.10: Comparison of four 3DCT datasets. The 220 kV dataset is the reference dataset. All the other datasets are color coded by computing the local difference between their scalar values and the scalar values of the reference dataset.

Great works are performed, not by
strength, but by perseverance.

Samuel Johnson

2

Feature Peeling

Transfer functions are used in 3D visualization to assign user defined optical properties to a volumetric dataset based on scalar values. The specification of the optical properties should be able to highlight defective tissues or features that are of interest for a particular medical or industrial study. This is a non-trivial task and often requires considerable time and expertise to achieve desired results. While one might be able to set up a system which can be reused for several patients, this is not always possible.

Magnetic Resonance Imaging (MRI) datasets are more difficult to handle than and are quite different from Computed Tomography (CT) datasets. Hounsfield numbers give a good and patient independent indication of the tissue types in CT. In contrast, the variance of tissue response between different patients in MRI datasets is too large to use pre-defined transfer functions for the detection of features. Transfer function specification has to be performed every time a new MRI dataset is generated. This fact makes transfer function specification a difficult and time consuming task.

Additionally, MRI typically contains a considerable amount of noise that makes it harder and more challenging to produce insightful visualizations. There is high frequency noise that affects the clarity of the images and there is low frequency noise that slowly changes the intensity of a signal. 3D visualization techniques are rare for MRI datasets and often medical personnel use manual exploration of the datasets through slice-based inspection (MPR).

We propose a novel rendering technique that identifies interesting features along viewing rays based on a ray profile analysis. For each particular view point, the algorithm allows the user to browse through the layered features of the dataset (see section 2.2). This technique can be used without specifying any transfer function and henceforth is suitable for time critical applications. Further, we include a de-noising step to be able to deal with MRI datasets. We successfully apply this technique to a variety of medical and synthetic datasets (section 2.4).

This work aims at similar goals as the work of Rezk-Salama and Kolb [64]. We intend to show the entire dataset to the user, in a layered manner without putting effort in setting up a complicated transfer function. We detail the differences as well as the relationship to other work in the next section.

2.1 Related Work

In volume rendering, transfer function specification is the main tool for the user to define optical properties. The transfer function guides the user to detect features in a volumetric dataset. The 1D transfer function is the simplest example which maps scalar values to opacity and color. More effective, but complex transfer functions that require user training and experience have been proposed.

A number of interesting enhancements, meant to give the user insight into the data, have been developed for the 1D transfer function. Simple histograms [7], the contour spectrum [2], and Laplacian-weighted histograms [56] have been suggested. Potts and Möller [58] investigate the usage of a logarithmic scale that eases transfer function specification.

Multi-dimensional transfer functions [34] have been introduced which assign optical properties based on data values but also first and second derivative information thereof. Kindlmann and Durkin [32] automate the generation of a transfer function. They use the relationship between scalar values and their first and second order derivative to highlight boundaries in the dataset.

Kniss et al. [35] describe how probing and clipping can enhance the understanding of the features that exist in a dataset. Patel et al. use innovative moment curves to specify feature aware transfer functions [55]. Bergner et al. [4] use a spectral representation of colors instead of the RGB model to enhance details and to allow an effective exploration of datasets. Transfer function specification is also tailored specifically to the visualization of medical datasets by making use of metadata [12].

Gao and Shen [18] efficiently extract isosurfaces for a given viewpoint. They divide the dataset into spherical partitions and store these in a binary tree designed for fast access. They propose a method for extracting a single iso-surface and thus their technique is not generally applicable to CT and MRI based medical datasets.

Höhne et al. [29] extend the Marr-Hildreth segmentation operator to 3D. They apply their technique in a view-independent manner. The work requires correction of errors through user input and produces a segmentation of the dataset as output. In our approach, we do not intend to perform segmentation but instead want to reveal features in a view-dependent manner. As we neither require human intervention nor any pre-processing of the entire dataset, our technique is interactive and we can control the level of peeling in real-time. A detailed description of volume graphics techniques and their applicability to medical visualization is given by Engel et al. [17].

While most of these techniques require user intervention, it would be preferable to cut this step from the visualization process to provide a quick insight into the dataset. This is the idea of the opacity peeling approach of Rezk-Salama and Kolb [64]. It allows a layered browsing of the dataset. The layers are defined through accumulated opacity and basically all the information in the dataset is visible. As the layers are based entirely on visibility (as opposed to features), objects of interest might be split and distributed among several layers. Instead of

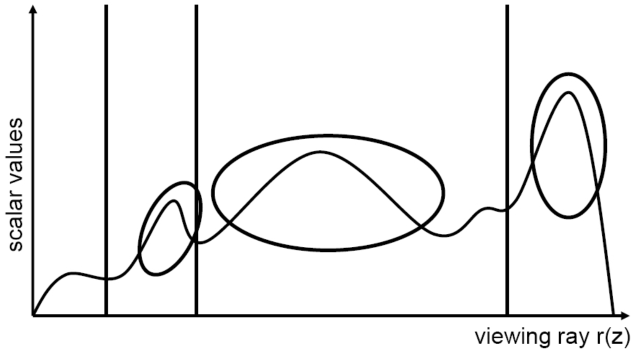


Figure 2.1: A ray profile showing three features as prominent density peaks. Features are marked with ovals and vertical lines show the transition points. Transition points split a ray profile into different layers.

peeling different layers of opacity, we propose to do an analysis of ray profiles and split the rays not according to opacity thresholds, but rather at possible feature transition points.

2.2 Feature Peeling

Along a ray in a dataset, we find valleys, peaks, and homogeneous regions. Areas of interest are, by default, regions where the data field is changing. If there are detectable transitions from one tissue to another, then these transitions will be present in regions with changing data values. These transitions can be very useful and the transition points are the places where interesting features inside the dataset start or end. Whether we wish to look at objects occluded by others or want to search for a disorder inside a single organ, such transitions will help us to explore and to locate information fast and easily.

A ray cast through a medical dataset produces a ray profile based on the scalar values encountered. This ray profile is used by various techniques in a specific way to generate images. For example, direct volume rendering (DVR) performs front to back or back to front compositing along the ray profile. Considering a linear ramp as an opacity transfer function, peaks in the ray will contribute most to the volume rendering integral. Averaging on the other hand calculates an average of all the scalar values encountered by a ray, as is common in X-Ray rendering or Fourier volume rendering [49].

Maximum intensity projection (MIP) only displays the highest peak in a ray profile [43, 52, 65]. Similarly, local maximum intensity projection (LMIP) [66] searches for the first peak above a specified threshold along the ray.

In figure 2.1, a ray profile with three features is shown. These features are prominent density peaks in the ray profile as compared to the rest of the profile. Feature peeling is separating these features into different layers by locating transition points between them.

We present a high level explanation of feature peeling in figure 2.2 by a 2D illustration. We use three concentric circular layers shown at the top left of figure 2.2 as dataset. A ray is also shown, which passes through the center of the dataset. The ray profile and the transition points corresponding to the ray are

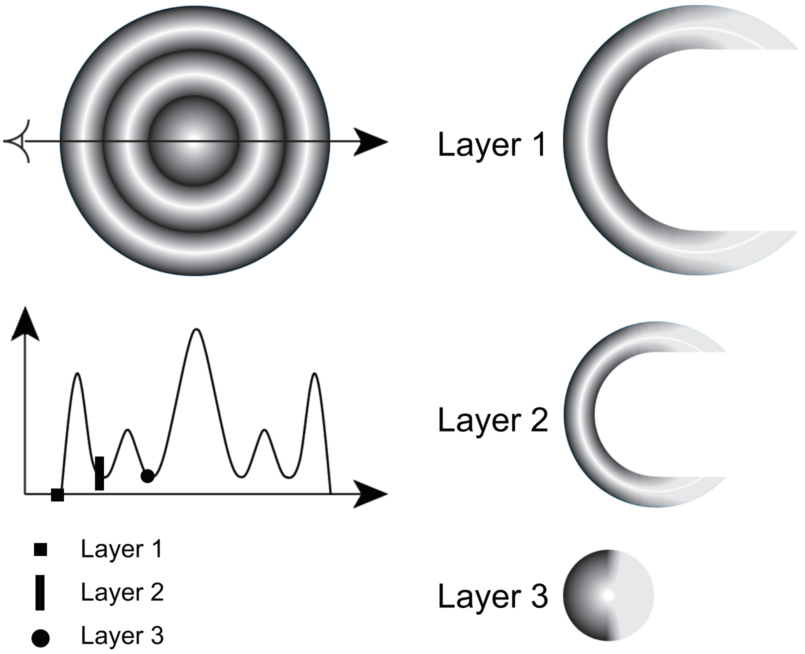


Figure 2.2: Three concentric layers of a dataset are shown at the top left of the figure. A ray is depicted in the dataset and the corresponding ray profile is drawn below. Small marks on the ray profile are shown which depict the transition points. On the right we show the first three layers that will be generated from this dataset as a result of feature peeling.

drawn below the dataset. The first three transition points of the ray are depicted with a square, a rectangle and a circle respectively. These transition points mark the start of features. On the right we show the first three resulting layers from the current view point. The example in figure 2.2 illustrates the importance of finding transition points that are representative of the features in a dataset.

2.2.1 Locating Transition Points

The search for the transition points is based on the first and second derivatives of the ray profile. Local minima, i.e., locations along the ray where the first derivative is zero and the second derivative is positive are considered as the transition points.

The number of transition points can vary from ray to ray based on the number of features along a ray and the amount of noise present. In the case of MRI datasets, a large number of transition points might be present due to high frequency noise. We remove high frequency noise by using a low pass filter. This will smooth the profile, remove the transition points that exist because of noise, and enhance the transition points that are representative of the dataset.

Low frequency noise in the MRI datasets may also generate false transition points. These transition points are removed by calculating the slope between the transition point and the local maximum of the peak immediately following the transition point. If the slope is less than some user specified threshold, it is discarded and the search for another transition point is carried on from the local maximum onward. We call this threshold *slope-threshold* henceforth.

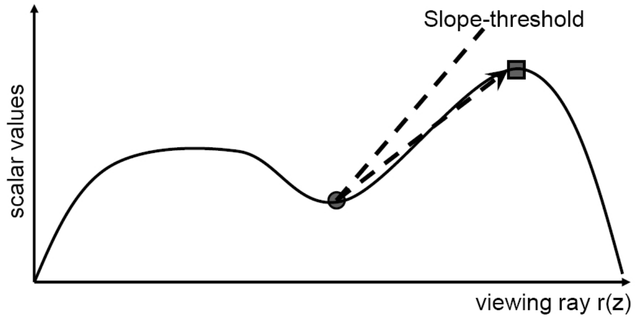


Figure 2.3: The gray circle shows the local minimum and the gray square shows the first local maximum after this local minimum. The dashed line depicts the *slope-threshold*. The slope between the local minimum and the local maximum is shown by a dashed arrow.

Figure 2.3 illustrates the usage of the *slope-threshold*. A local minimum and a local maximum are depicted as a circle and a square respectively. The dashed line shows the *slope-threshold* specified by the user. A dashed arrow depicts the calculated slope. This local minimum will not be considered a transition point as the slope between the local minimum and the local maximum is below the *slope-threshold*.

Figure 2.4(top) shows an original ray profile from an MRI head dataset. High frequency noise is visually recognizable. Transition points for this ray profile are shown in figure 2.4(center) as vertical lines, calculated after removing high frequency noise. Arrows point at the local minima that are the result of low frequency noise and which are discarded using the *slope-threshold*. Figure 2.4(bottom) shows a slice from the MRI dataset with a line indicating the ray whose profile is given in figures 2.4(top) and 2.4(center). The *slope-threshold* was set to 1.0 (45 degrees).

2.2.2 Relevance Across Neighboring Rays

The transition points are generated with no input from neighboring ray profiles. This may subdivide the volume data into non-smooth feature layers because of the difference in depths of the transition points of neighboring rays. Figure 2.5 shows three neighboring rays and their first and second transition points. A dashed curve is drawn by connecting the first transition points of all the three rays. Similarly, the dotted curve connects the second transition points of the rays. The distance from the start of a ray to the location of the transition point along the ray is called transition depth. The transition points have variable transition depths in all the rays shown.

The part of the dataset that lies between its boundary and the dashed curve in figure 2.5 is called layer 1. Similarly, data between the dashed curve and the dotted curve is called layer 2 and so forth. We cannot expect constant transition depths across neighboring rays as the main point of feature peeling is to generate images from the individual irregular feature layers. In order to accommodate inaccuracies of our de-noising procedure, we however, want to avoid large changes in the transition depths from one ray to its neighbors. This is achieved by assigning each transition point an importance or a relevance value that is based on its similarity

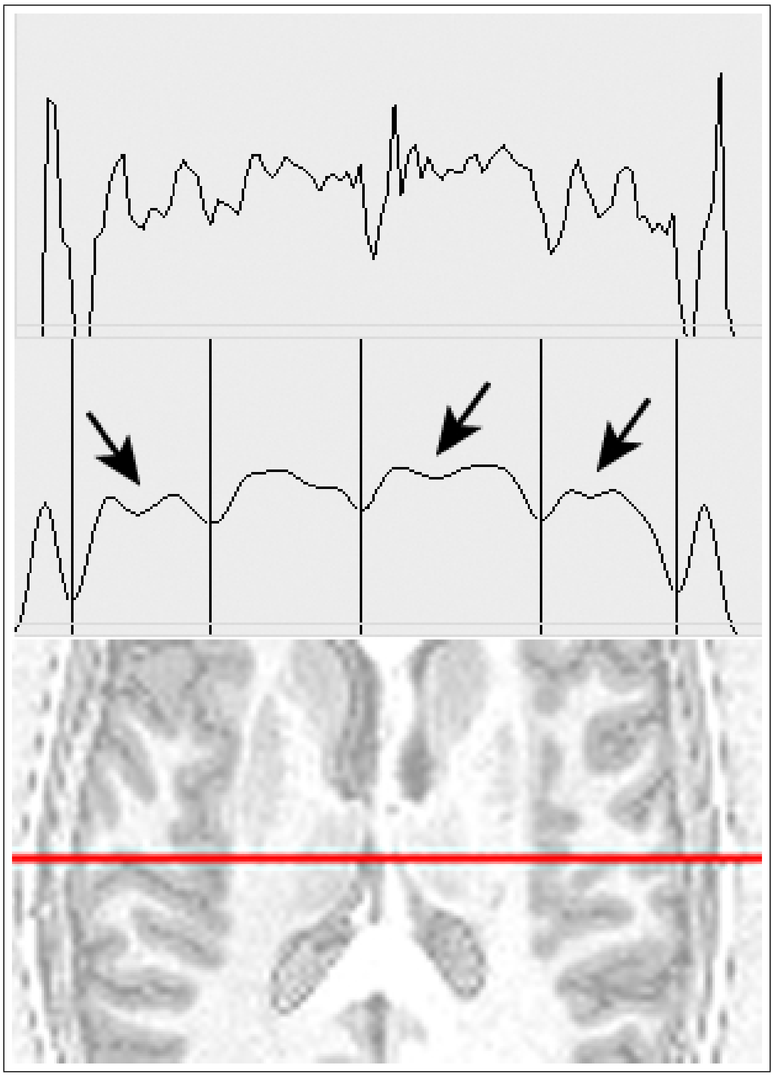


Figure 2.4: Original ray profile of an MRI dataset (top) shows a lot of high frequency noise. The profile is filtered with a median low pass filter (center). Vertical lines depict the transition points and arrows indicate the local minima skipped based on the *slope-threshold*. At the bottom a slice from the MRI dataset is shown and the horizontal line indicates the path of the ray.

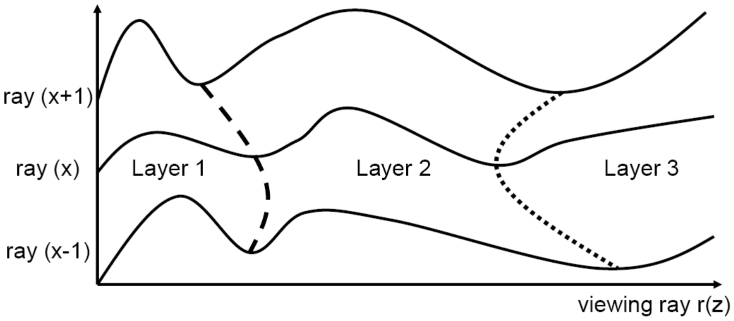


Figure 2.5: Three rays with position x-1, x and x+1 are shown. A dashed curve is drawn by connecting the first transition point of each of the three rays. It shows an interface between layer 1 and layer 2. A dotted curve shows an interface between layer 2 and layer 3 and is drawn by connecting the second transition points of all the rays.

to corresponding transition points in the neighboring rays. This will help to interactively control the level of peeling, i.e. to specify how many layers will be generated.

According to equation 2.1 we calculate the importance value Ip_{xy} of a transition point by taking the absolute difference between the transition depth $depth_{x,y}$ at the current ray profile x, y and the average of the transition depths in the 3×3 neighborhood of the ray x, y. Finally we subtract the normalized absolute difference from 1.0. The normalized absolute difference is computed by dividing the absolute difference by $depthmax$. The maximum transition depth $depthmax$ can be set individually for each view-point or it may also be set globally, i.e., equivalent to the diagonal of the bounding box of the dataset.

$$Ip_{xy} = 1.0 - \left[\left| depth_{x,y} - \sum_{i=-1}^{1} \sum_{j=-1}^{1} \frac{depth_{x+i,y+j}}{9} \right| / depthmax \right] \tag{2.1}$$

$$\text{where } 0 \leq depth_{u,v} \leq depthmax$$

The importance value ranges from close to zero to one with zero being the lowest importance and one being the highest importance for a transition point. Transition points with a low importance are potential jumps in a layer and should be discarded. Similarly, the user can also control the level of peeling (i.e., number of layers) by means of a user specified threshold, called *peeling-threshold*. Transition points with importance less than the *peeling-threshold* are ignored by the algorithm.

Listing 1 shows the high level pseudo code for finding transition points using feature peeling. A function, which locates a transition point on a ray for a given layer is presented. We also show the pseudo code for finding a given layer by opacity peeling in Listing 2. Opacity peeling only takes visibility into consideration and is therefore insensitive to features. It performs front to back compositing until the accumulated opacity is above a certain threshold called *T_high* and the opacity for the current ray position has dropped below a threshold called *T_low*. Feature peeling on the other hand produces layers where each layer corresponds to a feature in the dataset.

Listing 1: Pseudo code of feature peeling for calculating transition points on a single ray. A transition point for a given layer is calculated on the basis of user specified thresholds.

lowPassFilter() /* filters a location using a user specified filter */
sampVol() /* returns a scalar value from a 3D dataset */
calSl() /* returns slope between the parameters */
calImp() /* returns an importance value for a transition point */

```
LocateTransitionPoint(layer, rayPos)
  counter = 0 /* counts number of feature layers */
  LocalMinFd = false /* local minimum found or not*/
  nextVal = lowPassFilter(sampVol(rayPos))

  while (NotEndofRay)
    curVal = nextVal
    nextVal = lowPassFilter(sampVol(rayPos + 1))
    slope = calSl(curVal, nextVal)

    if ((slope>0)&&(LocalMinFd == false))
      LocalMinFd = true, LocalMin = rayPos
    else if ((slope<0)&&(LocalMinFd == true))
      LocalMax = rayPos
      slope = calSl(sampVol(localMin),curVal)
      if (slope>slope_Threshold)
        IPxy = calImp (localMin)
        if (IPxy>peeling_Threshold)
          if (counter == layer)
            return LocalMin
          else
            counter++
      LocalMinFd = false
    rayPos = rayPos++
```

Listing 2: Pseudo code of opacity peeling. Features might be split among opacity layers.

FrontToBackComp() /* performs front to back compositing */
T_high /* threshold for the accumulated opacity */
T_low /* threshold for the opacity at current ray position*/

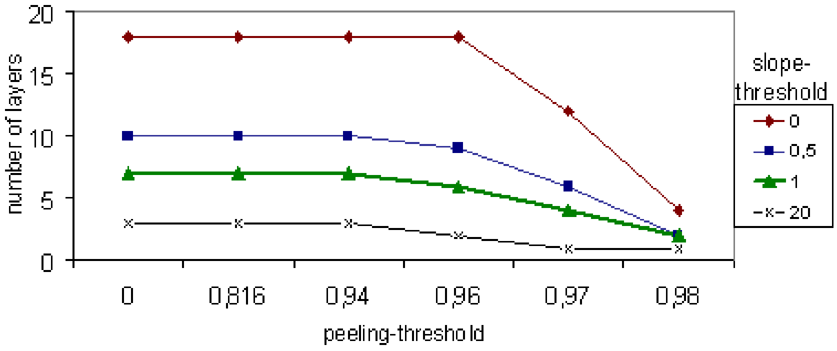
```
LocateLayerPostion (layer, rayPos)
  accOp = 0 /* accumulated opacity */
  curOp = 0 /* opacity at current ray position */
  counter = 0 /* counts number of opacity layers */

  while (NotEndofRay)
    FrontToBackComp()
    if ((accOp>T_high)&&(curOp<T_low))
      if (counter == layer)
        return accOp
      else
        counter++, accOp = 0
  rayPos = rayPos++
```

2.3 Implementation and Performance

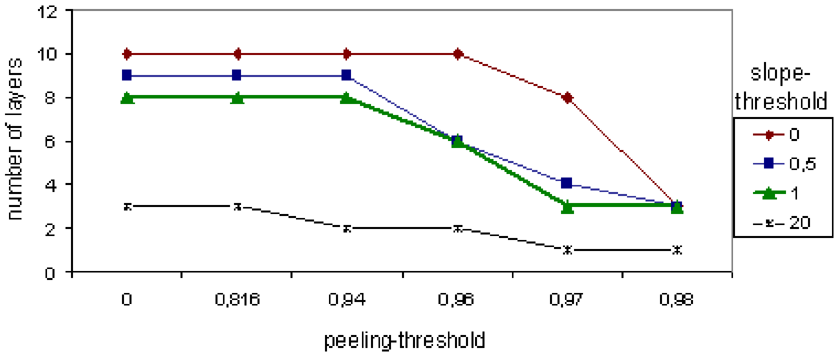
We implemented feature peeling on an AMD Athlon, 2.1 GHz CPU and an NVidia GeForce 6600 graphics board. Feature peeling is generic with respect to rendering. The separate layers can be rendered for example with DVR, MIP, LMIP. The images that we show in this chapter are generated by using DVR. We have used a median low pass filter of width five for de-noising in all of our test cases.

The result images are computed by controlling just two sliders. One slider specifies the *slope-threshold* and the second slider controls the *peeling-threshold*. Both of these thresholds affect the number of layers that are generated by the feature peeling algorithm. Figure 2.6(a) shows the number of layers generated from an MRI head dataset by using various combinations of these thresholds. The resolution of the head dataset is 256 × 256 × 109.

There is no change in the number of layers produced by feature peeling when the *slope-threshold* is zero and the *peeling-threshold* is between 0 and 0.96. There is a sharp decline in the number of layers produced through feature peeling when the *slope-threshold* is set to the lowest value of zero and the *peeling-threshold* is changed from 0.96 to 0.98. On the other hand, there is not a significant change in the number of layers when the *slope-threshold* is set to a high value of 20 and the *peeling-threshold* is changed from 0 to 0.98.



(a) MRI head dataset



(b) CT hand dataset

Figure 2.6: These graphs show the number of layers produced from (a) an MRI head dataset and (b) a CT hand dataset for different combinations of the *peeling-threshold* and the *slope-threshold*.

This shows that by assigning a low value to the *slope-threshold* the number of layers does not vary uniformly by changing the *peeling-threshold*. For a high *slope-threshold* the number of layers that can be produced by manipulating the *peeling-threshold* is limited.

Figure 2.6(b) shows the number of layers generated by different combinations of the *slope-threshold* and the *peeling-threshold* for a CT hand dataset. The dataset has a resolution of $244 \times 124 \times 257$. The variance in the number of layers with respect to the thresholds is similar as witnessed in figure 2.6(a). However, the variance in the number of layers is less than in figure 2.6(a) as there are fewer features in the hand dataset.

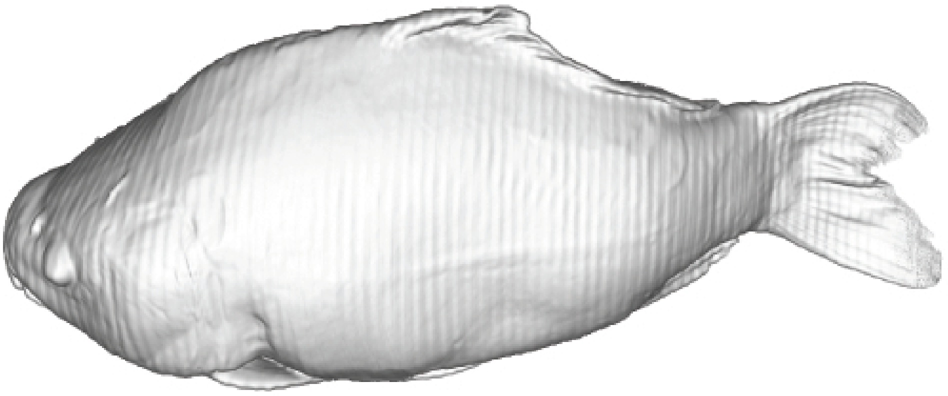
The *slope-threshold* can also affect how features show up in an image. The user can interactively change the value of the *slope-threshold* for each layer according to the requirements. Figure 2.7(a) shows the first layer of the carp dataset. Figure 2.7(b) and (c) both show the second layer of the carp with different *slope-thresholds*. In figure 2.7(b) we observe that noise and parts of the spinal cord are partially occluding the swim bladder of the carp. An increase in the *slope-threshold* can fade away the spinal cord and noise and display a clear view of the swim bladder as in figure 2.7(c). The swim bladder is visually highlighted by a rectangle.

We measure the performance of our system using an MRI head dataset. We also compare our results with an implementation of opacity peeling. Table 2.1 shows the rendering speed of feature peeling and opacity peeling for four different layers of the head dataset. Opacity peeling produces layers through direct volume rendering. In order to extract a certain layer, all the layers on top of that layer have to be rendered as well. Feature peeling defines layers based on the ray profile analysis and requires less computation to identify a feature layer. The *slope-threshold* was set to 1.0 (45 degrees) and the *peeling-threshold* was set to 0.965. The image resolution was 512×512.

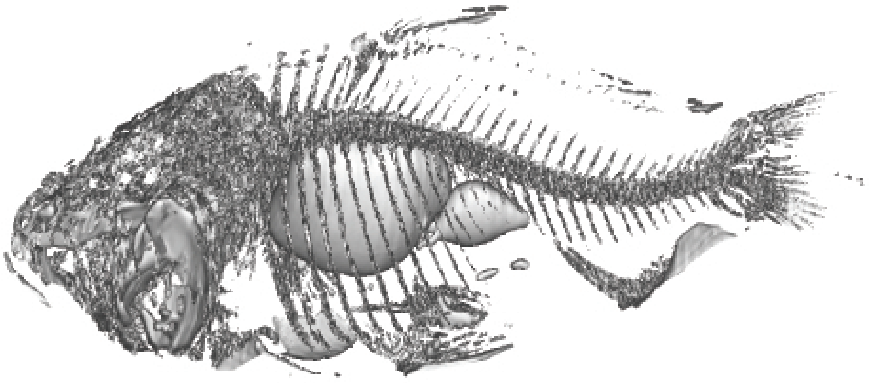
The performance decreases with an increasing number of layers. The frame rate is dependent on the depth of the transition points and also on the number of local minima being skipped. Therefore, the rendering time must increase to visualize layers of larger depth.

Table 2.1: This table shows the performance of feature peeling. Column 2 shows the rendering speed in frames per second if only the *slope-threshold* is used to decide if a local minimum will be considered a transition point. Column 3 shows the rendering speed for different layers when both thresholds are calculated. The last column shows the performance of opacity peeling.

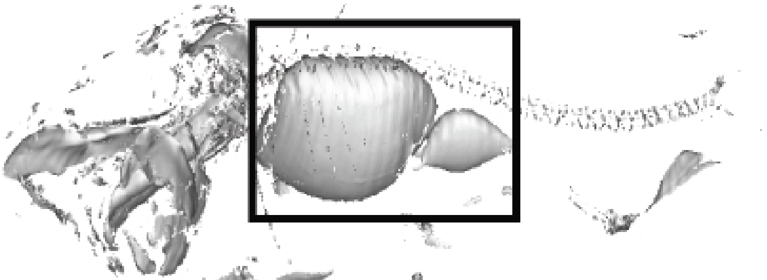
	Slope-threshold only	Slope-threshold and peeling-threshold	Opacity peeling
Layer1	8.2fps	7.8fps	4.3fps
Layer2	6.1fps	5.9fps	3.6fps
Layer3	5.6fps	5.2fps	2.3fps
Layer4	5.2fps	4.8fps	2.0fps



(a) Layer 1



(b) Layer 2



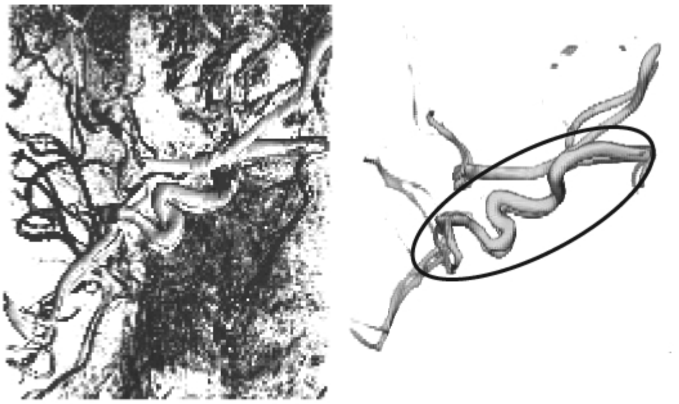
(c) Layer 2

Figure 2.7: First layer of the carp is shown in (a). (b) and (c) display the second layer of the carp with different values for the *slope-threshold*. The *slope-threshold* is 1.3 and 7.0 for (b) and (c) respectively.

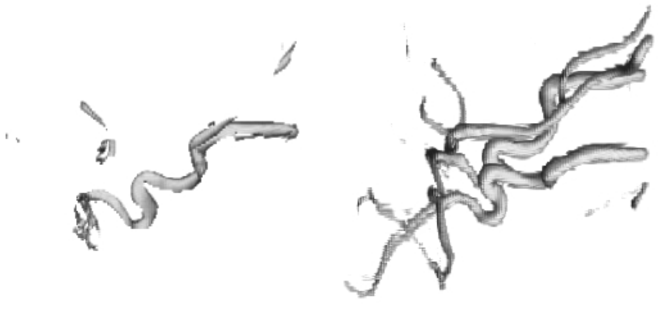
2.4 Results

We show results by using two MRI datasets (head angiography and head) and two CT datasets (present and hand). We also show the variance in the transition depth across the layers produced by feature peeling and compare the results with opacity peeling.

Figure 2.8 shows an MRI angiography dataset divided into two layers. Figure 2.8(a) shows the direct volume rendering of the dataset without feature peeling. A lot of high frequency noise is present and thus the veins are not clearly visible. Figure 2.8(b) shows the first layer of the MRI dataset, generated through feature peeling. All the high frequency noise is filtered out and we have a clear view to the veins. However, two veins of approximately the same shape are present in the area marked with an oval. The vein that is covering the other vein is removed in figure 2.8(c), to reveal the hidden vein by showing the second layer. Figure 2.8(d) again shows the first layer rendered through feature peeling after slightly rotating the dataset. Both the veins can now be seen to have almost the same shape.



(a) DVR (b) Layer 1



(c) Layer 2 (d) Layer 1

Figure 2.8: DVR of an MRI dataset (head angiography) is shown in (a). The first layer of the dataset is given in (b). Two veins with approximately the same shape exist in the region marked with an oval. The vein that occludes the other one is peeled away in layer 2 (c). Both veins are visible in (d). The *slope-threshold* is 3.0 and the *peeling-threshold* is 0.4.

Figure 2.9 shows six layers of the present dataset: the complete present as it looks from the outside (2.9(a)), the inner box (2.9(b)), the snow globle (2.9(c)), the St. Stephan's Cathedral (2.9(d)), the platform and the hidden towers of the St. Stephan's Cathedral (2.9(e)) and the mouse (2.9(f)) are all distinctively visible in separate layers.

In figures 2.10(a) to 2.10(c) and 2.10(e), we show four layers of an MRI head dataset. The first layer in figure 2.10(a) shows the outer layer, the second layer in figure 2.10(b) shows the brain surface, the third layer in figure 2.10(c) uncovers the eyeball and reveals parts of the ventricles and the fourth layer in figure 2.10(e) shows the corpus callosum, inner parts of the ventricles and the right eye. Figure 2.10(d) and 2.10(f) show the third and fourth layers of the same dataset produced using opacity peeling. Ventricles and corpus callosum are for example less recognizable in figure 2.10(d) and 2.10(f) compared to figure 2.10(e).

Figures 2.11(a) and 2.11(b) show the second and the third layer of a hand dataset produced by opacity peeling. Figure 2.11(a) shows bones and some parts of the veins. The third layer in figure 2.11(b) skips large parts of the veins, making it difficult to visualize them. Zoom-in highlights the region where veins have been skipped.

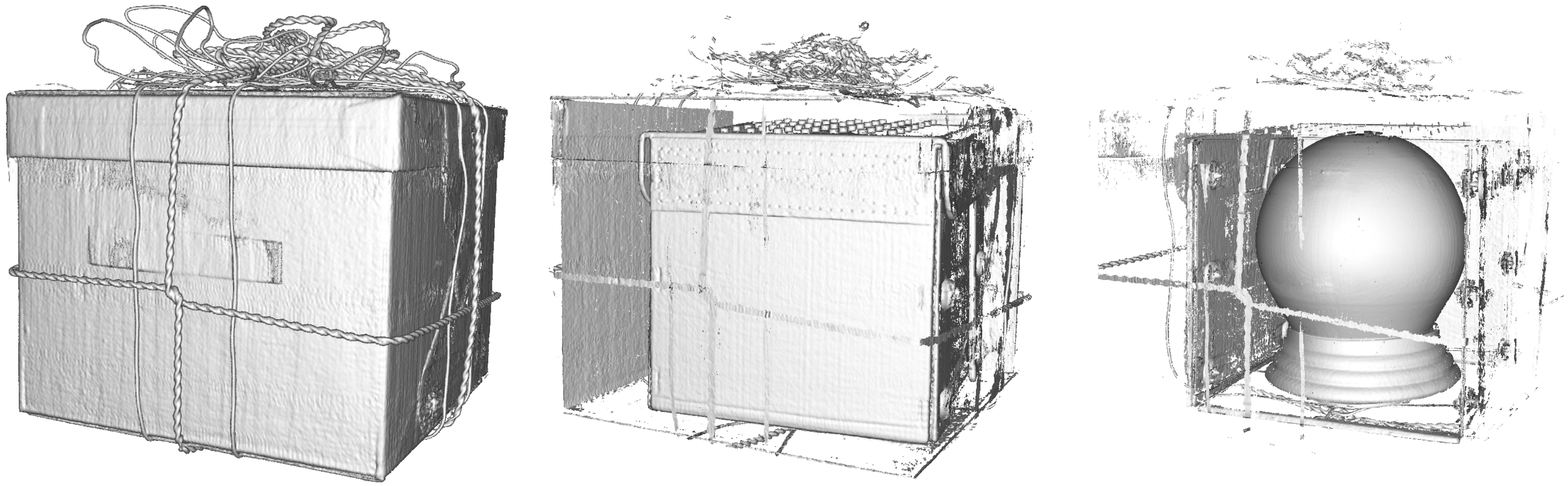
Figures 2.11(c) and 2.11(d) show the second and the third layer of the hand dataset generated using feature peeling. The second layer in figure 2.11(c) peels off the upper bone and shows some veins and the lower part of the bone. The third layer in figure 2.11(d) removes bone to show occluded veins and arteries.

Figures 2.12(a), 2.12(b) and 2.12(c) display the standard deviations for the second, third and fourth layer of the head dataset using both feature peeling and opacity peeling. Feature peeling consistently produces lower variance on the layer boundaries as compared to opacity peeling. Opacity peeling is concerned with visibility irrespective of feature boundaries, while feature peeling separates the volume data along smooth feature interfaces. We have used a *slope-threshold* of value 1.0 to generate these graphs.

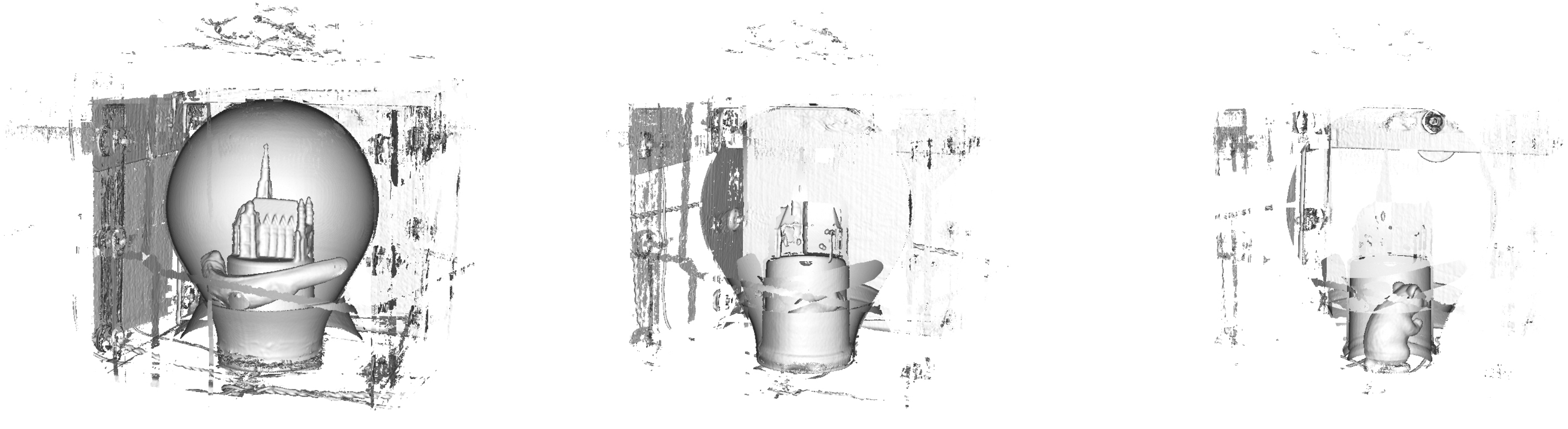
2.5 Conclusion

This chapter introduces feature peeling, which for a selected view-point enables browsing of volumetric data in feature layers. It has shown promising results for MRI datasets, which are hard to visualize using traditional 3D visualization techniques. While feature peeling requires the specification of two thresholds (a *slope-threshold* as well as a *peeling-threshold*), we believe that these thresholds can remain constant over a large number of patient studies. However, this needs further investigation. Further, a more thorough test needs to be done on the influence of the de-noising method. Currently we are simply using a low pass median filter. It is possible that a bi-lateral filtering in combination with Gaussian smoothing or similar approaches might improve the coherency of transition points.

It will be interesting to investigate if we can use feature peeling to dynamically select view positions. We would like to detect the regions inside the dataset where the data field is changing most rapidly. The algorithm can then calculate an optimal viewing position for these regions and perform feature peeling. This could provide a separate view for almost every feature of the dataset.

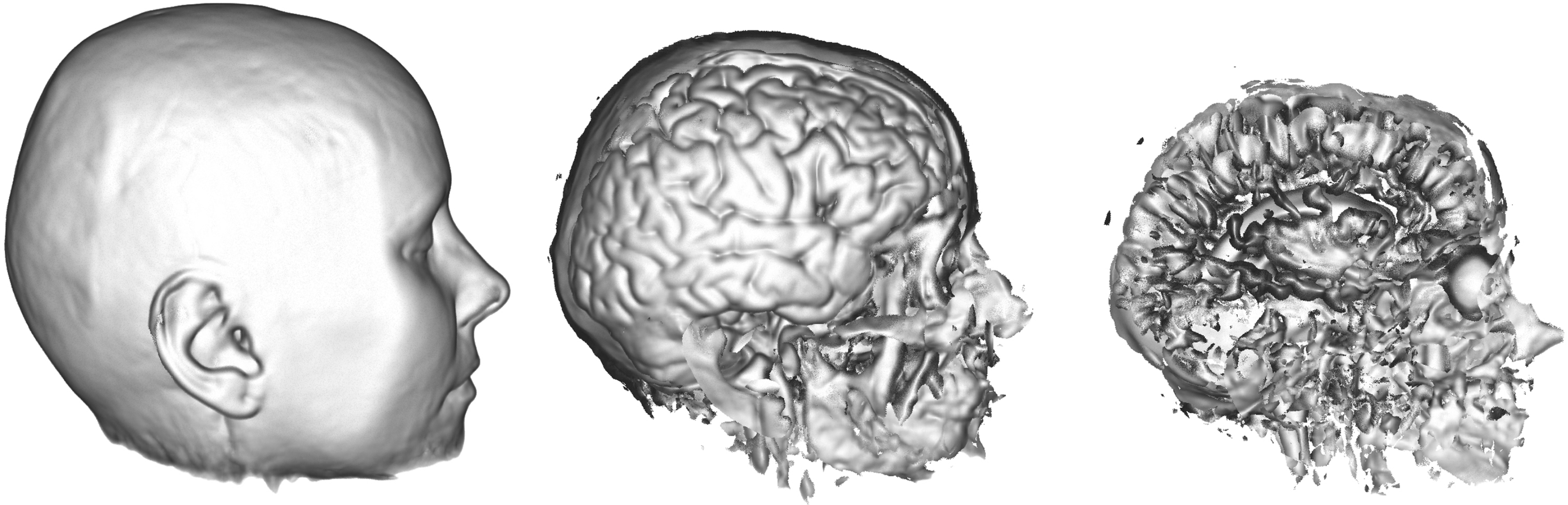


(a) Layer 1 (b) Layer 2 (c) Layer 3

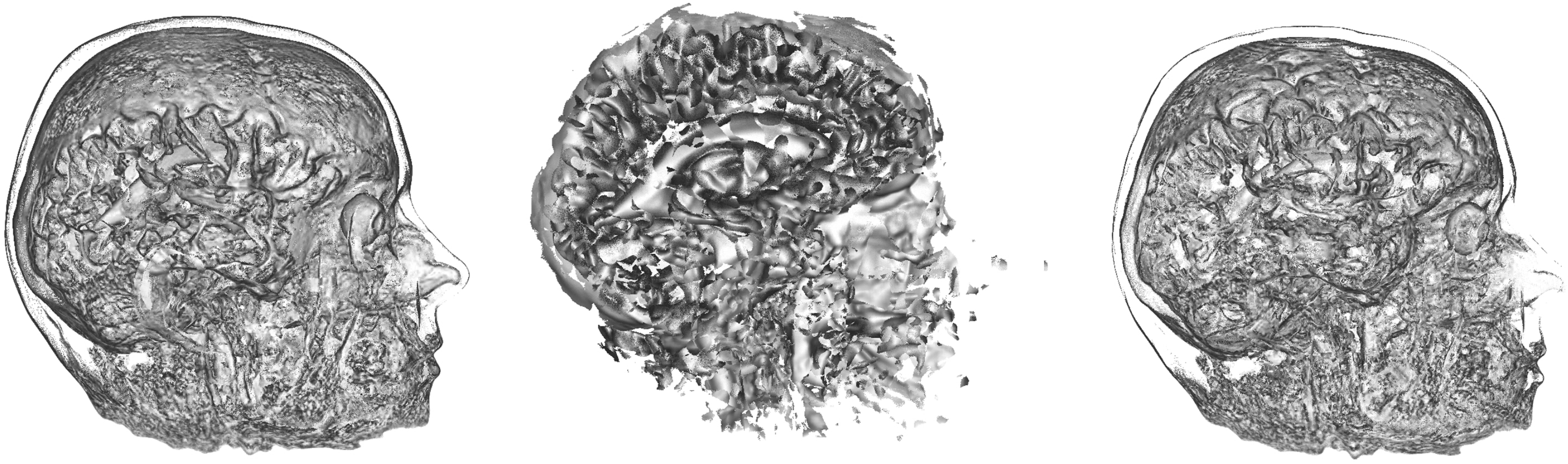


(d) Layer 4 (e) Layer 5 (f) Layer 6

Figure 2.9: From (a) to (f), six layers of the present dataset are shown in order. Present is a CT dataset with a resolution of 492 × 492 × 442. The *slope-threshold* is 2.0 and the *peeling-threshold* is 0.9.

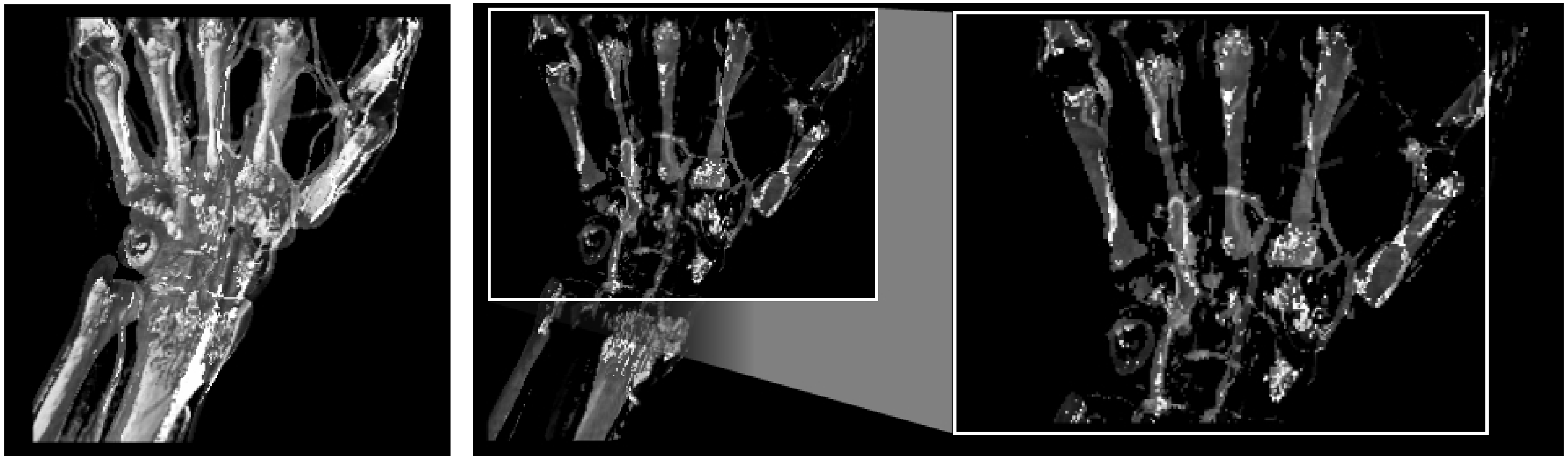


(a) Feature peeling (b) Feature peeling (c) Feature peeling



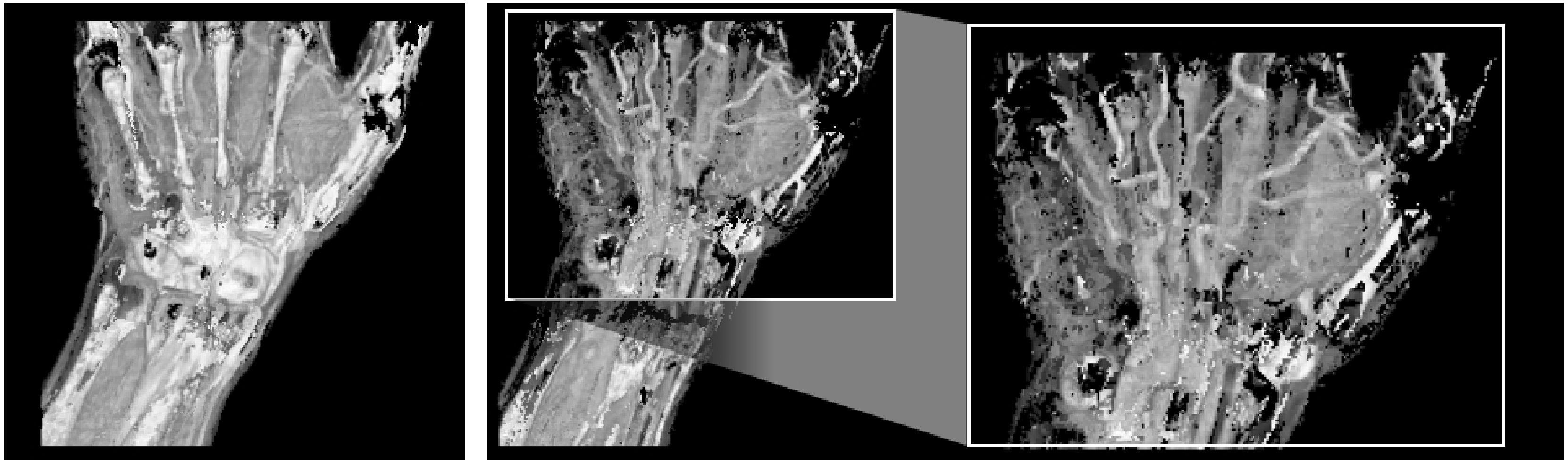
(d) Opacity peeling (e) Feature peeling (f) Opacity peeling

Figure 2.10: (a), (b), (c) and (e) show the first, the second, the third, and the fourth layer generated using feature peeling. The third and the fourth layer obtained using opacity peeling are shown in (d) and (f). (e) shows ventricles and the right eye as well as a clearly distinguishable corpus callosum. These features are not clearly visible neither in (d) nor in (f). The *Slope-threshold* is 1.0 and the *peeling-threshold* is 0.965.



(a) Opacity peeling

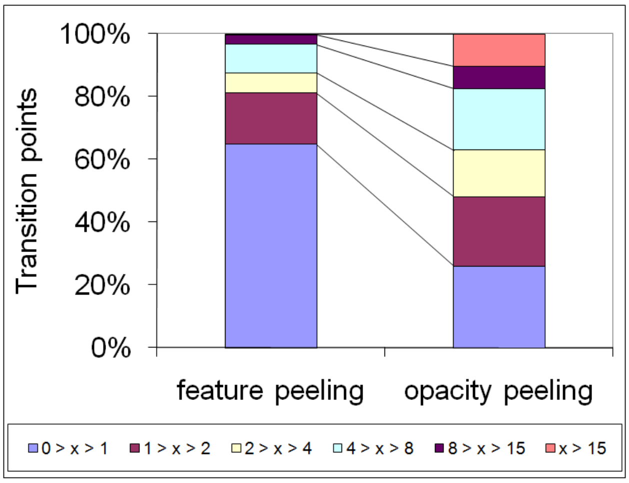
(b) Opacity peeling



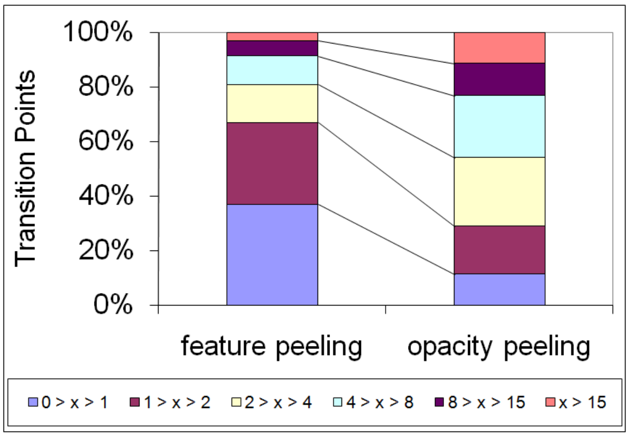
(c) Feature peeling

(d) Feature peeling

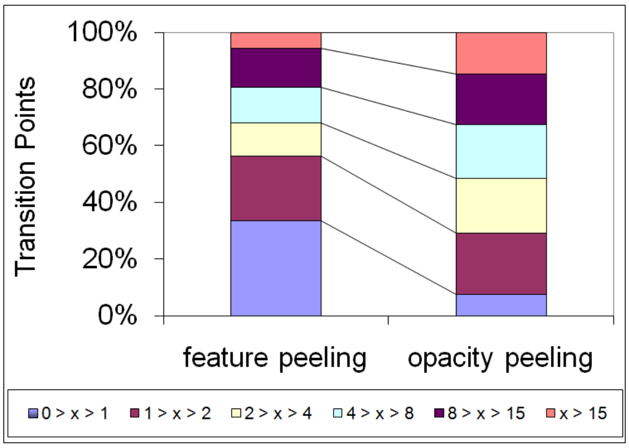
Figure 2.11: (a) and (b) show the second and the third layer of the hand dataset rendered using opacity peeling. (c) and (d) show the second and the third layer obtained using feature peeling. The veins in (d) are better visible through feature peeling. Zoom-ins are provided in (b) and (d). The first layer of the hand dataset is not shown as it is not relevant here. The *Slope-threshold* is 1.0 and the *peeling-threshold* is 0.97.



(a) Layer 2



(b) Layer 3



(c) Layer 4

Figure 2.12: The variance in the transition depth of a layer for feature peeling as well as for opacity peeling is shown for three layers. (a) shows variation of depth in the second layer of the MRI head dataset. (b) and (c) show results for the third and the fourth layers respectively. The legend includes ranges of standard deviations into which the transition points were categorized.

> Generosity is giving more than you can, and pride is taking less than you need.
>
> Kahlil Gibran

3

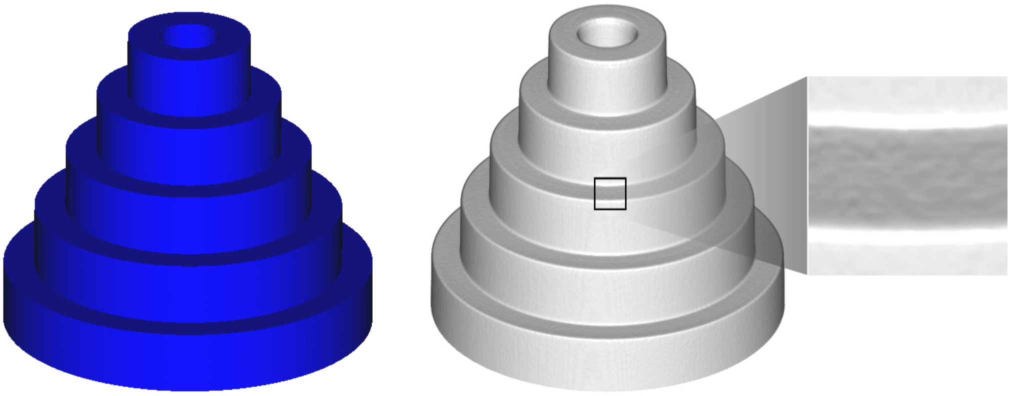
Computation and Visualization of Fabrication Artifacts

Comparison of two almost identical datasets is very important for the continuously rising demands of quality control in industrial engineering. Recently much work has been done in the area of mesh comparison. A high number of vertices and edges are hard to process in real-time due to the limited processing power available in hardware. This initiated research to simplify mesh datasets in such a way that the rendering speed is increased while the mesh distortion is limited. Distortions introduced through mesh simplification led to research on mesh comparison.

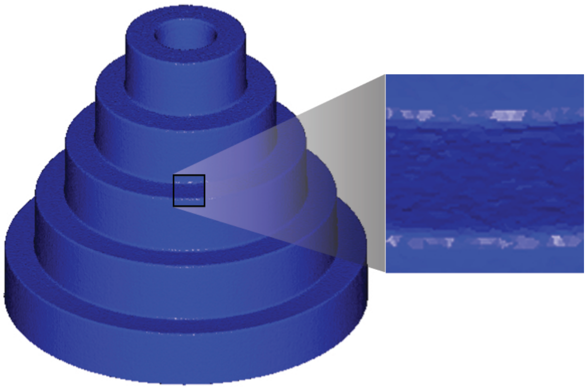
In the manufacturing industry, it is necessary to produce industrial components as close as possible to the computer aided design model (CAD) of the part. Engineers use CAD tools like AutoCAD, Pro Engineering etc. for designing. The CAD model is considered to be the ground truth during the manufacturing process. To verify the accuracy of the production process, manufactured components are scanned with an industrial 3D X-ray computed tomography (3DCT) machine. The volumetric dataset obtained from the 3DCT scan is then compared to the CAD model of the part. The comparison between the two datasets is supposed to clearly identify erroneous regions.

The comparison process uses various methods to measure differences between the two datasets. The differences present between the CAD model and the volume data are the result of fabrication, measurement, and surface reconstruction artifacts. We are primarily interested in detecting the fabrication artifacts as these are introduced in an industrial part during the production phase. The goal of the comparison process is to minimize all post-production artifacts so that the differences measured between the datasets mainly correspond to fabrication artifacts.

Datasets of industrial components, unlike medical datasets, mostly consist of materials with distinctive density values. There is a high signal to noise ratio and the interfaces in the volume data are easy to detect. For this reason the most



(a) CAD (b) 3DCT scan



(c) Iso-surface mesh

Figure 3.1: (a) CAD model of test-part-1 (CAD model: 200,000 triangles, volumetric dataset: 561 × 559 × 436 voxels). (b) direct volume rendering of the scan of test-part-1. (c) iso-surface mesh extracted from the volumetric dataset in (b).

common method for first part inspection is to generate an iso-surface mesh from the 3DCT scan and to compare it with the CAD model. In various cases this is not the ideal approach: First, the generation of a mesh from the 3DCT dataset requires a surface extraction algorithm. Industrial components have sharp edges and corners and therefore a lot of surface reconstruction artifacts are introduced [37]. Second, mesh generation for a given iso-value is not interactively possible during the comparison process. Therefore, the need to do a comparison with a higher or lower resolution mesh will lead to a delay in the examination process. Third, a 3DCT dataset goes beyond a CAD model and has information about the interior of the mechanical part as well. Losing this information limits the examination possibilities of the 3DCT dataset.

Figure 3.1 shows a CAD model in (a), direct volume rendering (DVR) of the 3DCT scan in (b) and an iso-surface mesh extracted from the 3DCT scan in (c). In figure 3.1(c), all the internal information of the volumetric dataset is lost. Areas marked with black rectangles in figure 3.1(b) and 3.1(c) are shown as zoom-ins. We observe surface reconstruction artifacts in figure 3.1(c).

In this chapter we present a novel approach to perform a comparison directly between the CAD model (which is the ground truth) and the volumetric dataset obtained from the 3DCT scan. We calculate the difference between the CAD model and an interface of the volume data and also compare the relative surface smoothness. We show that the differences we measure represent fabrication artifacts (section 3.3). The uncertainty of the measurement process is also evaluated and presented to the user.

Color coding, glyphs, ray profiles, and 3D box plots are provided for visualization and the results are also displayed quantitatively. The proposed method is implemented on the Graphics Processing Unit (GPU). It provides interactive comparison and visualization. We successfully avoid reconstruction artifacts by comparing the CAD model directly with the volume data. Delays in the examination process are also avoided by embedding the complete comparison and visualization pipeline in a single system.

3.1 Related Work

Large numbers of triangles are inefficient to render and also hard to stream over a network. Subsequently, algorithms are proposed to simplify meshes [13, 73]. Mesh simplification distorts the original shape to some extent and therefore techniques are proposed to measure the differences between the two meshes.

Metro [14] is a general tool designed to compare two surfaces. One of the surfaces is scan converted to a set of points and then the distance between each point and the other surface is measured. Aspert et al. [1] propose to use the Hausdorff distance for measuring differences which is computationally and memory wise efficient. Pichon et at. [57] propose to use the gradient of the Laplacian equation to locate corresponding points on the two surfaces. The corresponding points are then used to measure distances between the surfaces. They argue that the Hausdorff distance fails to consider the variation in shape. We calculate uncertainties in the datasets to compensate for the shape variations.

A variety of metrics and visualization techniques are proposed by Zhou and Pang [85] to measure mesh differences and present the information visually. The combination of metrics and visualization methods intends to help users test and calibrate various mesh simplification algorithms and find what suits their specific requirements.

Turk presents an algorithm [73] to create multiple levels of detail from a given iso-surface mesh. Samples are randomly distributed over the surface and subsequently a relaxation process spreads them uniformly. In the final step the user can choose a surface from a discrete set of surfaces to perform comparison. The generation of a surface model and variance comparison are two separate processes. The method proposed by Turk can be used to avoid delays in the variance comparison process by generating multiple surface models with different levels of detail in advance.

Weigle and Taylor [80] investigate visualization methods for distance and local shape comparison. Their study shows that glyphs are better in conveying deviation information between surfaces than color coding alone. They use intersecting surfaces with known alignment for their study.

The above mesh comparison techniques do not need registration. As the simplified mesh is extracted from the original mesh, both meshes are perfectly aligned. In the case of a CAD to a 3DCT dataset comparison, where both datasets originate from a separate process, there is a need to perform registration. We have included an iterative closest point algorithm [5, 54] for dataset registration. A survey of registration algorithms specific to medical datasets is provided by Wang et al. [45].

There has been some recent work on the variance comparison between a CAD model and a 3DCT dataset. These methods however introduce a preliminary step to the variance comparison process, where an iso-surface mesh is generated from the 3DCT dataset. Heinzl et al. [28] propose a technique for generating a feature preserving mesh from a 3DCT dataset. They use filtering to suppress noise and a watershed segmentation to create a binary dataset. In the final step a surface model is created using elastic surface nets, which is then used for variance comparison. There is a whole body of work in the area of surface extraction, which is considered to be out of scope for the work in this chapter.

Geomagic Qualify [61] is a well-known software product, used for quality control in industrial engineering. A CAD model and an iso-surface mesh of the volumetric dataset are inputs to this tool. It uses a point-set to point-set or a best fit alignment algorithm for registration and then performs distance analysis between the two datasets. Results are presented as a color coded surface. Methods for extracting an iso-surface mesh from a volumetric dataset [26, 28, 38, 78] have to be used in a preprocessing step for performing comparisons using Geomagic Qualify. Geomagic Qualify works independently from the surface extraction process and therefore assumes no reconstruction artifacts during the comparison process, although such errors are introduced in the pre-processing step.

3.2 Computation & Visualization

Our comparison system is divided into *geometry-driven* and *visual-driven* analysis techniques. *Geometry-driven* techniques provide an overall visualization of the differences between the CAD model and the volumetric dataset. *Visual-driven* techniques are used on top of the *geometry-driven* comparison techniques for a user guided analysis and for obtaining precise quantitative information.

An overview of the system is shown in figure 3.2. The Iterative Closest Point (ICP) algorithm performs rigid registration and produces a transformation matrix as output. The output matrix transforms the CAD model (moving dataset) through translation and rotation to closely orient it to the 3DCT dataset (fixed dataset). Registration is not the major scope of our work. We performed it with high accuracy (see section 3.3.1) using a well known algorithm in a semi-automatic way. Fully automatic registration techniques have not been investigated but might be applied.

Both types of comparison, i.e., *geometry-driven* and *visual-driven* comparison techniques (figure 3.2), query the registered CAD model for the necessary information but work completely independent from each other. The results of the query, i.e., meta data from the 3DCT dataset and the chosen visualization technique, are used to compute quantitative data and to produce images.

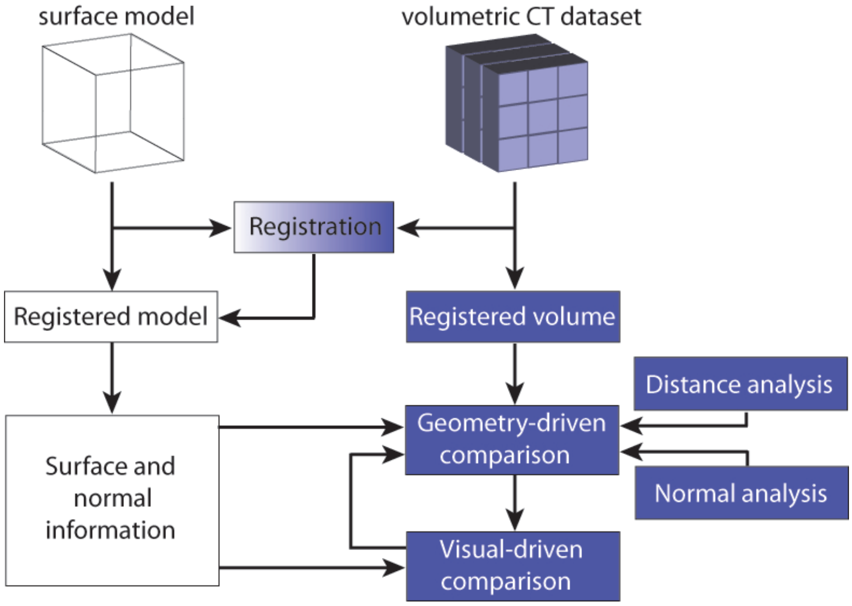


Figure 3.2: System overview. *Geometry-driven* comparison techniques color code the datasets and render glyphs. *Visual-driven* comparison techniques provide localized information about differences.

Geometry-driven comparison techniques consist of a *distance analysis* and a *normal analysis*. The *distance analysis* calculates the differences between the CAD model and an interface in the volumetric dataset as Euclidean distances. It also measures the uncertainty of the measurement process. The *normal analysis* precisely locates differences in curvature and compares the surface smoothness of the two datasets.

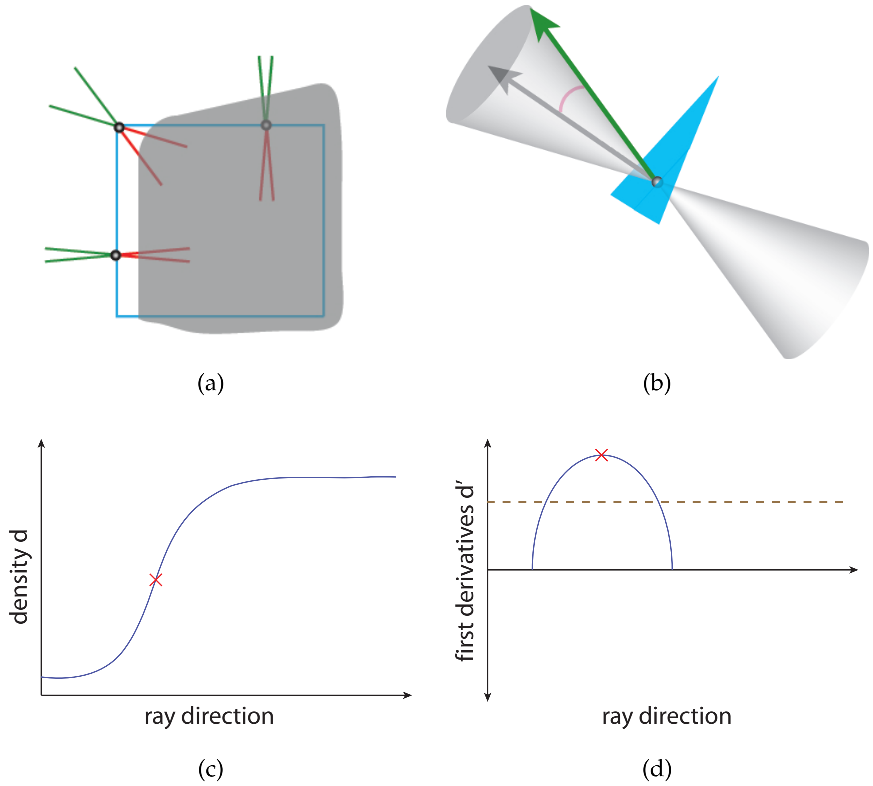
We provide a *ray-profile analysis* and a *magic lens* as building blocks of the *visual-driven* comparison. The *ray-profile analysis* visually presents the data and differences at a user specified location and also displays the information quantitatively. The *magic lens* extracts the differences between datasets at a user specified neighborhood and displays them using glyphs, i.e., box plots

3.2.1 Geometry-Driven Comparison

Distance and *normal analysis* methods require for each surface point on the CAD model, the specification of a corresponding point in the 3DCT dataset. Starting from a surface point we have to locate the corresponding point in the volumetric data. The search direction is approximately along the surface normal. In high curvature areas the search should be extended to nearby directions as well to ensure robustness.

Consider the blue rectangle and the gray object in figure 3.3(a) to be a CAD model and a volumetric dataset respectively. Black spheres represent surface points. A pair of red and green lines originating from each surface point indicates the conical space in which we search for a corresponding point in the volume data. The space is larger for surface points in high curvature regions (see the surface point at the corner in figure 3.3(a)).

For each triangle of the CAD model we evaluate the facet normal and the three vertex normals. The angle between the facet normal and each of the vertex



(a) (b)

(c) (d)

Figure 3.3: (a) pairs of red and green lines depict the space in which we search for each surface point (black sphere) on the CAD model (blue rectangle) a corresponding point in the volume data (gray object). (b) double cone representing the search space in 3D. A density profile and the first derivative of a density profile are illustrated in (c) and (d) respectively.

normals is computed and the maximum of the three angles (called search-angle henceforth) is stored. The search-angle indicates the local curvature of the CAD model. In areas of high curvature, a large search-angle will be calculated whereas the search-angle will approach zero in planar areas of the CAD model.

In figure 3.3(b) we indicate the search-angle as a red arc between the facet normal (black arrow) and one of the vertex normals (green arrow) of the blue triangle. Using the search-angle we can construct a double cone with the opening angle set to be twice the search-angle. The double cone is depicted in figure 3.3(b) with the apex placed on the surface of the triangle. We then extract the spatial locations and the normal vectors for a set of uniformly distributed surface points on the triangles of the CAD model. At each surface point the apex of a double cone is placed and the cone axis is oriented along the surface normal. A triangle therefore bisects the double cone at its apex (figure 3.3(b)). We call the nappe of the double cone that lies in front of the triangle as outside nappe, while the nappe on the back of the triangle is called inside nappe. The double cone defines a region in which we can search for an interface point in the volumetric dataset. An appropriate interface point found inside the double cone will be associated with the surface point of the triangle for further computations.

In order to search for an interface point in the volume data, we start from the surface point and traverse the volume data along several rays distributed inside the double cone. The rays originate from the surface point and are directed towards the two bases of the double cone. The density profile of each ray is used to identify the interface point as the position with highest gradient magnitude (first derivative is a maximum/minimum and the second derivative is zero). The gradient magnitude at a spatial location must be greater than a user specified threshold for that location to be considered an interface point. Thresholding is necessary to filter out small changes in gradient magnitude which do not represent an interface. We also apply a median filter to the density values to reduce noise. Among all the considered rays the interface point with minimum distance to the surface point is stored for further processing. The rays are distributed in concentric circles in the base of the double cone. The density of the rays is kept almost constant by taking more rays in the outer circles compared to inner circles.

A density profile of a ray is illustrated in figure 3.3(c). The graph of the first derivative of such a density profile is drawn as the blue curve in figure 3.3(d). The dashed brown line shows a threshold for the first derivative. The first peak or valley with absolute derivative value above the threshold is considered an interface point in the volumetric dataset. The interface point is indicated by a red cross in figures 3.3(c) and (d).

As we find an interface point in the volumetric dataset, we store its spatial location, the nappe (inside or outside) in which the interface point was found, and the gradient. The information extracted from the CAD model and the 3DCT dataset provides all the required parameters to evaluate the metrics for the *distance analysis* and the *normal analysis*.

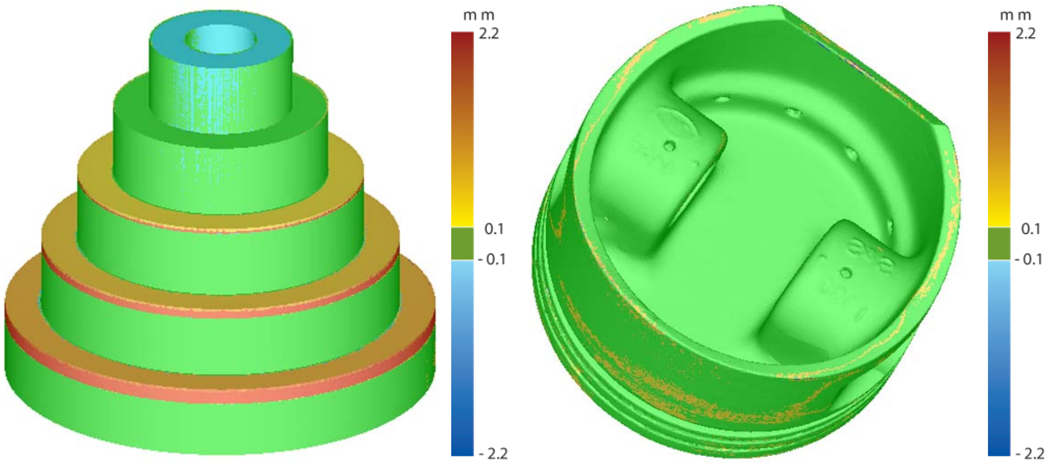
Distance Analysis:

The computationally intensive step of finding for each point on the CAD model a corresponding interface point in the volume data has already been done. The *distance analysis* shows the difference between the datasets as Euclidean distances. We compute the differences between the spatial locations on the CAD model and their corresponding interface points in the 3DCT dataset. We also have information about the nappe of the double cone in which the interface point was found. Using this information we color code the dataset for *distance analysis*.

Figures 3.4(a), (b), and (c) show test-part-1, test-part-2, and a synthetic dataset respectively. The test-parts are rendered using the *distance analysis* with distances measured in millimeters. The distance has positive sign if the interface point is found in the inside nappe of the double cone.

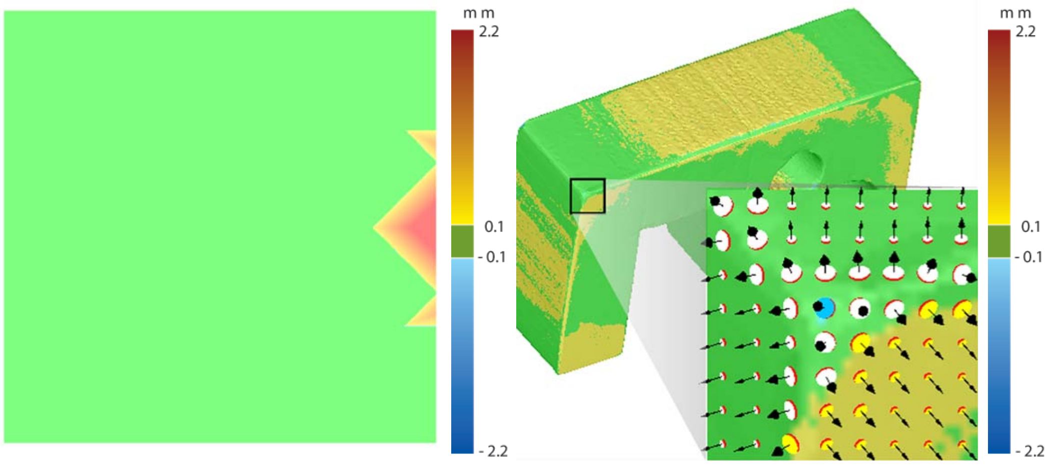
Figure 3.4(d) shows test-part-3 rendered using our *distance analysis* technique. We render distance glyphs on the zoom-in of the user specified area (black rectangle). The arrow of the distance glyph is aligned with the normal vector of the surface and the diameter of the disc is proportional to the diameter of the base of the double cone. The color of the disc indicates if the difference was found in the inside nappe (yellow), outside nappe (blue) or no difference was recorded (white).

So far we only consider the minimum distance between the CAD model and the interface of the volumetric dataset for *distance analysis*. The technique does not take the interface shape into consideration. The results have uncertainty in high curvature regions which needs to be highlighted. For a double cone the



(a) Test-part-1 (distance analysis)

(b) Test-part-2 (distance analysis)



(c) Synthetic dataset (distance analysis)

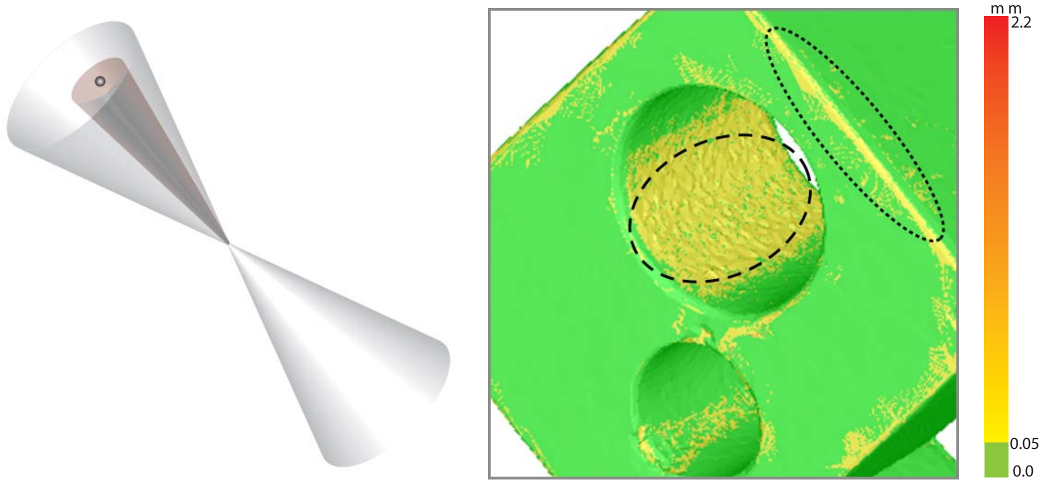
(d) Test-part-3 (distance analysis)

Figure 3.4: (a) test-part-1, (b) test-part-2 (CAD model: 152,054 triangles, volumetric dataset: $408 \times 351 \times 355$ voxels), (c) synthetic dataset (CAD model: 11,424 triangles, volumetric dataset: $256 \times 256 \times 256$ voxels) and, (d) test-part-3 (CAD model: 25,880 triangles, volumetric dataset: $329 \times 527 \times 181$ voxels) rendered using *distance analysis*. The image resolution is 512×512.

difference between the minimum and maximum distance from the CAD model to the volume data will be larger in high curvature regions compared to planar areas. Therefore the difference between the minimum and the maximum distance serves as the uncertainty value of the measurement process.

To determine uncertainty we look for the maximum distance from the surface point to the interface in the volume data. The search for the maximum distance is conducted in the neighborhood of the ray along which the minimum distance was found. The neighborhood for searching the maximum distance has a radius of one voxel. We choose this radius, as the search space should be smaller than the smallest feature in the dataset. Any feature less than the size of a voxel is not detectable in the volumetric dataset anyway.

Figure 3.5(a) illustrates the uncertainty measurement process. Let us assume that the closest interface point was found along the ray which starts from the



(a) Overview

(b) Test-part-3

Figure 3.5: (a) the maximum distance to the interface point is searched in the neighborhood (brown cone) of the ray (cone vertex to black sphere) along which the interface point was recorded. (b) uncertainty rendering for a zoom-in of test-part-3. A dotted and a dashed oval highlight areas of high curvature and rough surface respectively.

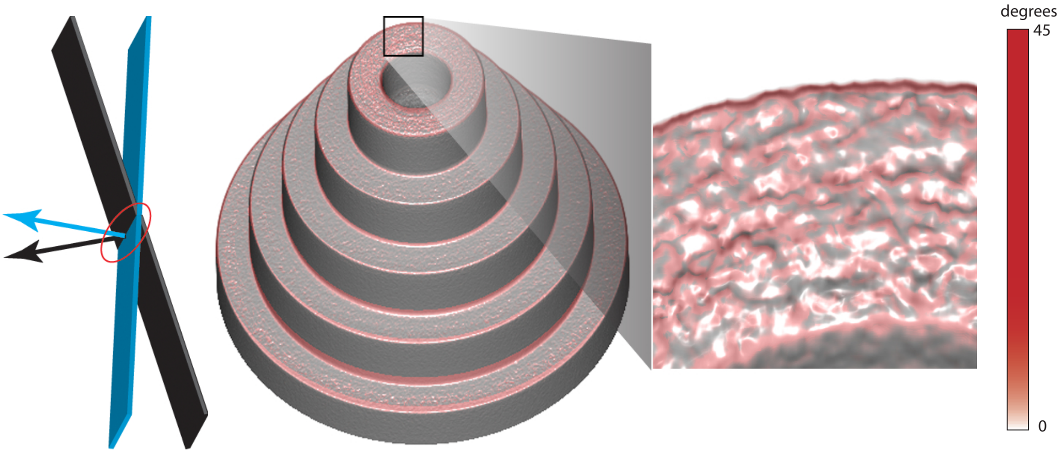
surface point (cone vertex) and extends towards the black sphere depicted on the base of the cone. In the neighborhood around this ray (brown cone), we search for an interface point with maximum distance to the surface point. The difference between the minimum and the maximum distance from the surface point to the interface in the volume data is considered the uncertainty of the measurement process.

The uncertainty in case of test-part-3 is shown in figure 3.5(b). It becomes apparent that areas of high curvature or high surface roughness, which are highlighted using a dotted and a dashed oval respectively, have higher uncertainty.

Normal Analysis:

Normal analysis is proposed as an efficient method to compare surface smoothness. *Normal analysis* compares the orientation of the normal vectors extracted from the CAD model with the gradients obtained from the 3DCT dataset. The angle between the normal vector and the gradient indicates the difference in the curvature of the CAD model and the interface of the 3DCT dataset. *Normal analysis* is easy and efficient to compute given that the surface points and the corresponding interface points are already evaluated.

The type of difference shown by a *normal analysis* may pass undetected by a *distance analysis*. Consider the black plane in figure 3.6(a) to be part of the CAD model with the normal vector indicated by a black arrow. The interface of the volume data (blue plane) overlaps the CAD model in the area marked with a red oval. The *distance analysis* will report no difference in such a case. However, there is a difference in the orientation of the two datasets as the normal vector and the gradient do not point in the same direction. Such differences can be emphasized



(a) Overview (b) Test-part-1

Figure 3.6: (a) *normal analysis* emphasizes differences in orientation. (b) test-part-1 rendered using *normal analysis*. The zoom-in shows roughness at the top of test-part-1.

using a *normal analysis*. The *Normal analysis* will report a constant difference along the entire surface in this example.

Figure 3.6(b) shows test-part-1 rendered using the *normal analysis*. The *Normal analysis* detects differences at edges and rough surfaces. As the volumetric dataset is generated from an industrially manufactured part, it does not match the smoothness and exactness of the CAD model, especially at the edges. The zoom-in in figure 3.6(b) shows that the top of test-part-1 is quite rough. The color scale can be changed dynamically by the user.

3.2.2 Visual-Driven Comparison

Visual-driven comparison techniques are grouped into *ray-profile analysis* and *magic lens* displays. *Ray-profile analysis* displays the differences between the datasets both as 2D plots and as quantitative numbers. A *magic lens* is used to zoom-in/out of the dataset and to view the differences graphically.

A ray-profile display is generated by plotting the first derivative of the density values encountered along the ray in the volume data. The peaks and valleys in the graph show the interface points. The location of the surface point is marked on top of the graph. The horizontal difference between the interface point and the surface point in the plot shows the local difference. This provides precise information about the differences in the datasets.

Figure 3.7 shows two ray profiles generated for positions on test-part-3 marked with black crosses. The vertical red lines depict the points on the CAD model. The blue graph shows the first derivatives of the density values encountered by the ray, along which the interface point in the volume data was found. The peaks in the blue graphs are the edges detected in the volume data. The horizontal distance between a peak and a red line indicates the local difference between the datasets.

In the ray profile in figure 3.7(left), we observe that the surface point (red line) and the interface point (peak) overlap and thus there is very little difference

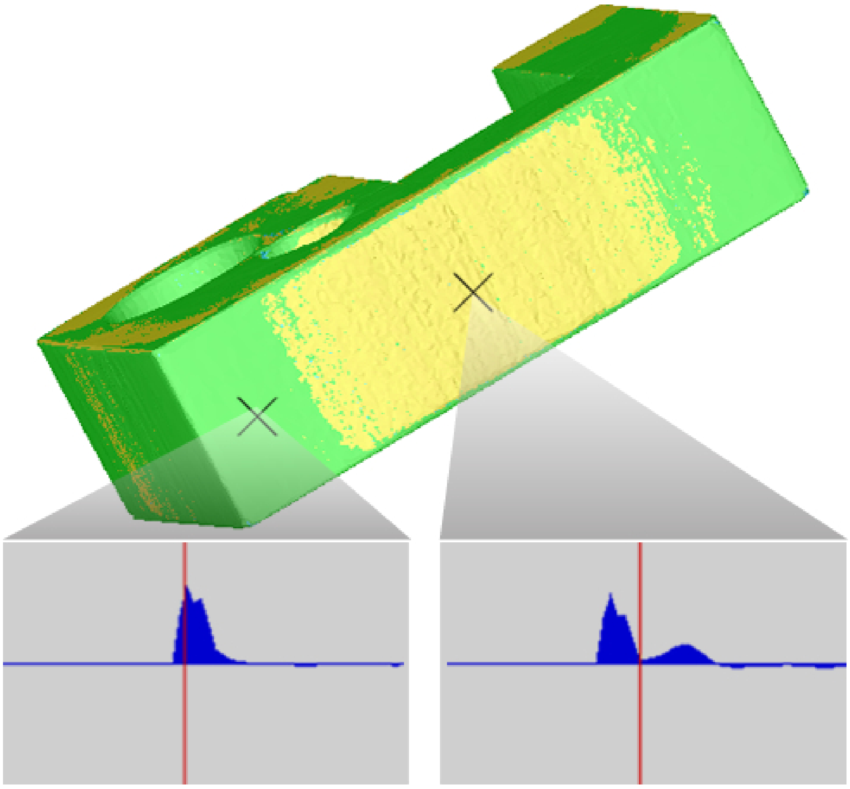
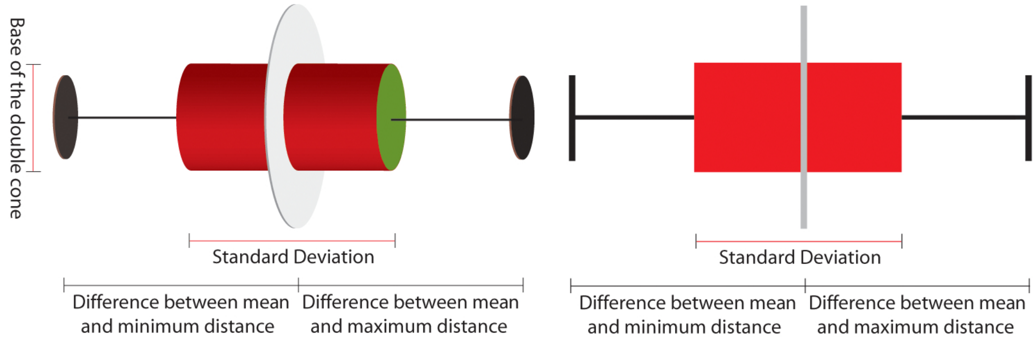


Figure 3.7: Two ray profiles are extracted for the locations marked with crosses on test-part-3. The horizontal differences between the red line (surface point) and the peaks in the graph (interface points) depict the dataset differences.

between the two datasets. The ray profile in figure 3.7(right) however shows a difference between the CAD model and the volumetric dataset as there is a horizontal difference between the red line and the nearest peak. Our system reported a difference of 0.2 *mm*.

A ray profile shows the distance at one specific position. The next approach shows differences in a small local neighborhood. A *magic lens* can provide a precise graphical view of the differences by means of 3D or 2D box plots (figure 3.8). The box plots are rendered in a user specified area. Each box plot shows the minimum, the maximum, the mean, and the standard deviation of the differences between the two datasets at each local neighborhood. Additionally, 3D box plots are oriented along the normal vectors of the CAD model. The diameter of a 3D box plot is directly proportional to the base of the double cone in which the interface point was searched. 3D box plots therefore encode distance values, uncertainty, and the dependent variables (normal vector and the base of the double cone) whereas 2D box plots only encode the distance values and the uncertainty of the measurement process (figure 3.8(a) and (b)).

Figure 3.9(a) shows 3D box plots over a user specified area (black rectangle) on test-part-3. 3D box plots similar to white planar discs indicate no difference in the minimum, maximum, and the mean distances recorded between the two datasets. The measurement is most certain in areas where "flat" 3D box plots are rendered.


Base of the double cone
Standard Deviation
Difference between mean and minimum distance
Difference between mean and maximum distance
Standard Deviation
Difference between mean and minimum distance
Difference between mean and maximum distance

(a) 3D box plot (b) 2D box plot

Figure 3.8: (a) diameter of a 3D box plot depicts the base of the double cone. (b) 2D box plot has a constant diameter.

3D box plots are less suited for a relative comparison as they are differently oriented along surface normals. 2D box plots make a relative comparison in a user specified area easier (figure 3.9(b)).

3.3 Results

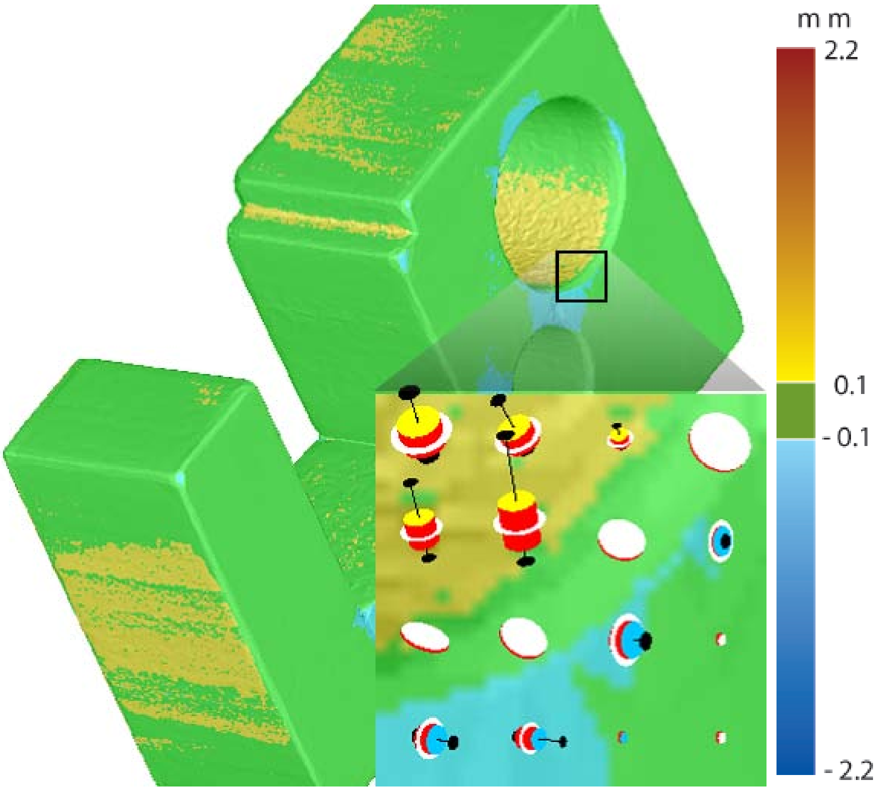
We implemented a prototype on a Pentium 4, 3.4 GHz CPU and an NVidia GeForce 8800 graphics board. We used C++ and OpenGL/GLSL as programming language. The system renders the volumetric data and the CAD model side by side in a volume view and a surface view. We maintain a central queue for the events performed in all views. An operation initiated in one view also pushes an event into the central event queue and releases a signal. The other view pops the event from the queue and executes it. We implement first-come, first-serve scheduling for the central queue.

3.3.1 Artifacts and Errors

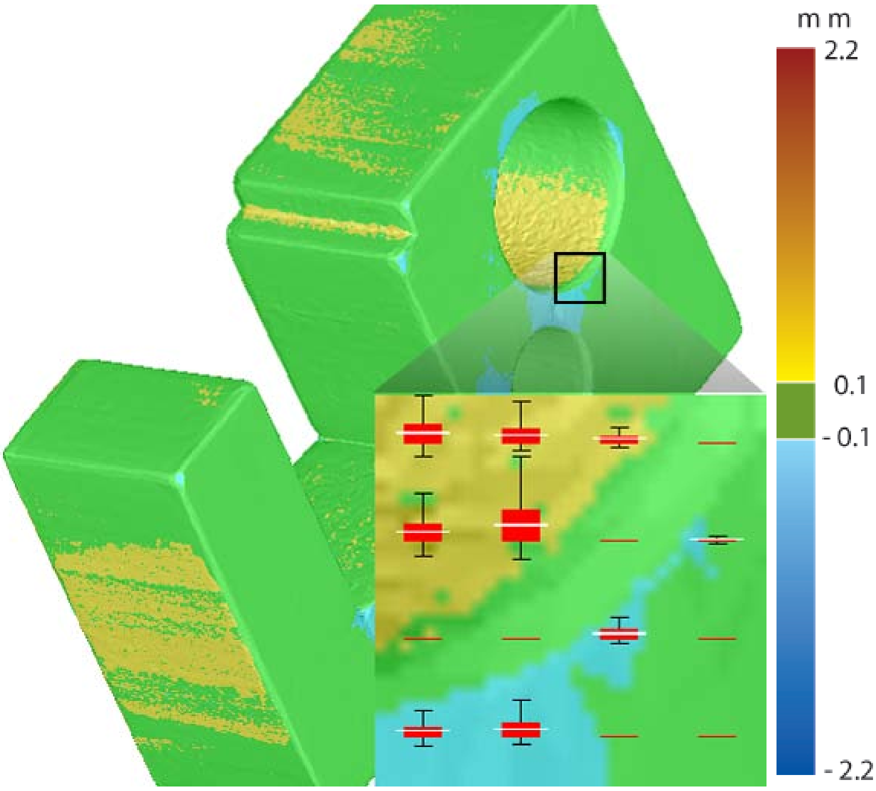
An industrial 3D X-ray computed tomography includes fabrication artifacts and measurement errors. Fabrication artifacts are introduced during each step of the manufacturing process whereas measurement errors are caused by the 3DCT machine. Additionally, two kinds of errors are generated by the software that is used to process the 3DCT scan. Surface reconstruction artifacts are introduced while extracting an iso-surface mesh from the volumetric dataset. Registration errors are caused by the registration algorithm.

Quality assurance engineers are primarily interested in measuring the fabrication artifacts. To accurately compute the fabrication artifacts, errors caused by the software should be minimized. We perform registration with high accuracy and unlike other contemporary techniques avoid surface reconstruction artifacts.

We evaluate the iterative closest point (ICP) registration algorithm by performing registration 20 times between test-part-3 and a feature preserving mesh [28] of test-part-3. We use a feature preserving mesh for testing purposes so that the fabrication artifacts and the measurement errors are minimized and we can



(a) 3D box plots



(b) 2D box plots

Figure 3.9: (a) and (b) show box plots over a user selected area. The user can interactively switch between 3D box plots and 2D box plots.

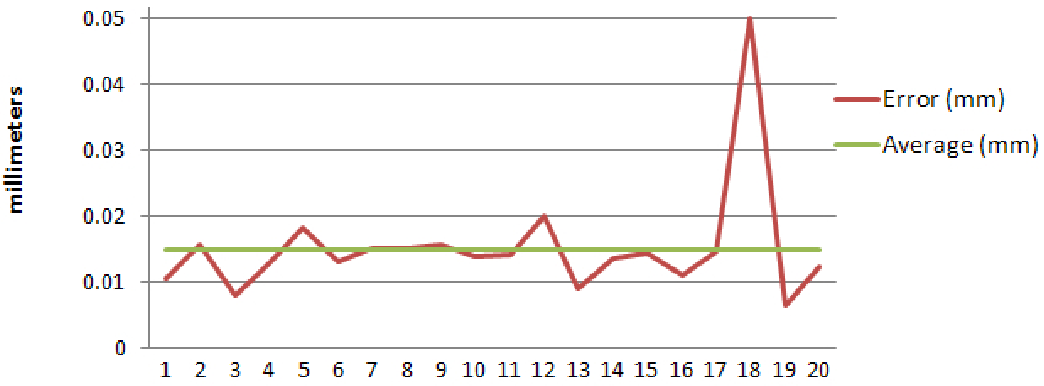


Figure 3.10: Mean square error produced by point-set to point-set registration on test-part-3 (size: $60 \times 100 \times 30$ *mm*).

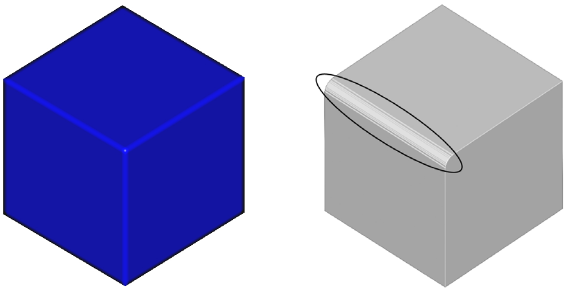
monitor just the registration error. We measure the mean square error between the mesh and the test-part-3 (see figure 3.10) and record an average registration error of 0.0152 *mm*. The registration algorithm converged in 3.5 iterations on average.

Experiment number 18 produced a high error compared to the rest of the experiments. The ICP registration algorithm requires user interaction and the large error in experiment 18 is due to a bad specification of control points. The maximum fabrication and measurement artifact found in test-part-3 is 1.93 *mm* and the mean difference recorded is 0.27 *mm*. Thus the average registration error introduced by the ICP algorithm is considerably lower than the mean and the maximum fabrication artifacts in the dataset.

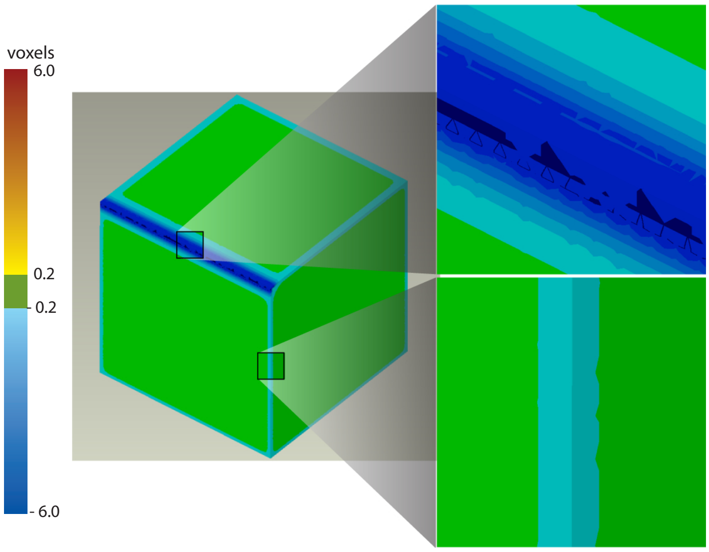
Reconstruction artifacts are introduced while extracting a mesh from a volumetric dataset. We use a synthetic dataset with known fabrication artifacts to evaluate our technique. Measurement and registration errors are not present in a synthetic dataset. This provides a good opportunity to analyze just the effect of surface reconstruction artifacts. Figure 3.11(a) shows a CAD model of a cube dataset and 3.11(b) shows a volume dataset with known fabrication artifacts. Fabrication artifacts are marked with an oval. The CAD model consists of 12,288 triangles and the volume data has a resolution of $256 \times 256 \times 256$ voxels.

We generated a feature preserving mesh [28] of the volumetric dataset and compared it with the CAD model using Geomagic Qualify (figure 3.11(c)). Surface reconstruction artifacts are visible in the differences shown by both zoom-ins. A difference is also reported at the vertical edge of the mesh (lower zoom-in) even though there should be no difference. The differences at vertical edges are purely caused by surface reconstruction artifacts and are not present in the dataset (see figure 3.11(b)). Figure 3.11(d) shows the comparison using our system. Our system correctly calculates no difference on vertical edges (lower zoom-in). The fabrication artifacts in the volumetric dataset are also reported correctly (upper zoom-in). The color coding is smooth and we do not observe any reconstruction artifacts.

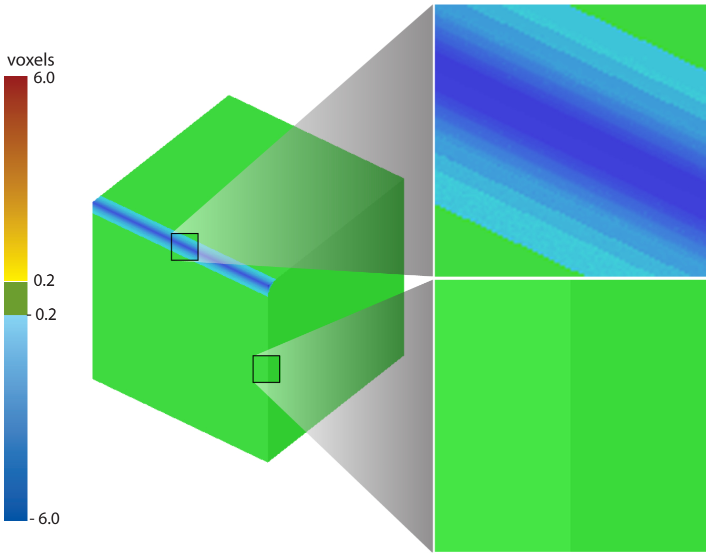
The comparison of the maximum and average differences evaluated by Geomagic Qualify and our technique is given in table 3.1. Our method calculates the differences very close to the ground truth. Geomagic Qualify reports the maximum difference close to the ground truth but the average difference has a large error. Reconstructing a mesh from the volume data introduces artifacts distributed over



(a) Triangular mesh (b) Volume data



(c) Comparison using feature preserving mesh



(d) Direct comparison using volume data

Figure 3.11: (a) triangular mesh. (b) volume data with known fabrication artifacts. Artifacts are highlighted with an oval. (c) comparison between the CAD model and a feature preserving mesh of (b). (d) direct comparison between a triangular mesh and the volume data (our approach).

the entire mesh. This is why the average error reported by Geomagic Qualify is very small compared to the ground truth. As we avoid reconstruction artifacts, our calculations are more accurate.

Table 3.1: Maximum and average difference reported by Geomagic Qualify and our system. Absolute difference is shown and the unit of measurement is voxel distance.

	Ground truth	Geomagic	Our technique
Maximum	8.485	7.95	8.91
Average	3.42	0.195	3.51

3.3.2 Performance and Evaluation

The earlier solutions proposed for comparison are divided into two major steps. For instance, Heinzl et al. [28] propose a robust surface detection pipeline for effective comparison. First, they extract a feature preserving mesh from the volume dataset. The mesh extraction part consists of a four step pipeline. In the first three steps, an anisotropic diffusion filter, a gradient filter, and a watershed segmentation are in turn applied to the volume dataset. In the final step constrained elastic nets are used. The mesh is then compared to the CAD model using some existing tool like Geomagic. We combine the entire comparison and visualization process into a single, interactive system. Table 3.2 shows the runtime performance of our system, in comparison to the robust surface detection pipeline [28] and Geomagic.

The bottle neck in earlier methods has been the surface extraction process. Due to parameter tweaking the surface extraction took very long as opposed to the actual comparison process. Our method is more automated and requires much less user interaction. A screen shot of our system is shown in figure 3.12.

Distance glyphs and the 3D box plots are additional visualization techniques for showing differences and uncertainties. Two domain experts who have used various mesh comparison systems in their professional capacity tested the usefulness of our visualization techniques. They were both quite interested in using distance glyphs and 3D box plots to visualize differences as compared to color coding alone. They acknowledged that they acquired more valuable information about the surface (surface normal), the measurement process (base of the double cone), and the differences using distance glyphs and 3D box plots. Showing glyphs in a user specified area is one of the features which the users are missing in conventional tools. The experts also appreciated the idea of showing the uncertainty of the measurement process along with the *distance analysis*. The robustness of the registration algorithm was satisfactory to them.

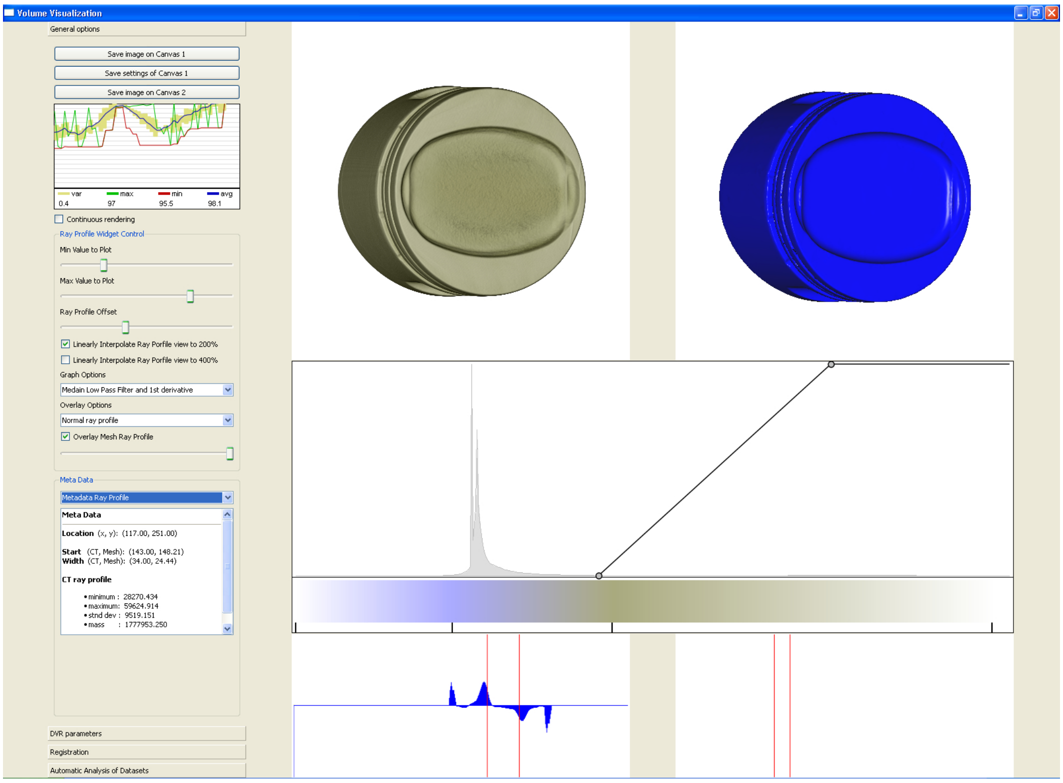
Table 3.2: Comparison of the performance of our system.

	Test-part-1	Test-part-3
Distance analysis (our method)	0.051 sec	0.033 sec
Robust surface detection pipeline	10.23 min	4.58 min
Distance analysis (Geomagic)	9.31 sec	8.51 sec

3.4 Conclusion

We have presented a variance comparison system that compares a CAD model directly to the 3DCT scan of specimens, especially in the preproduction phase and for first part inspection of new industrial products. We avoid intermediate steps for data enhancement and surface extraction. Two sets of tools, namely *geometry-driven* and *visual-driven* techniques provide comprehensive comparison opportunities. The system is implemented on graphics hardware and all the proposed methods work in real-time.

Future work might expand the toolset by plugging in more volume visualization and exploration techniques. For instance, feature extraction methods like region growing and 2D histograms can help inspect the concealed structures within the 3DCT data [31]. We use a semi-automatic algorithm for registration. We achieve high accuracy but fully automatic registration techniques can be further investigated. Feature-based registration algorithms can be modified to register a CAD model to a volumetric dataset.



var 0.4, max 97, min 95.5, avg 98.1

☐ Continuous rendering

☑ Linearly Interpolate Ray Porfile view to 200%

☐ Linearly Interpolate Ray Porfile view to 400%

Graph Options: Medain Low Pass Filter and 1st derivative

Overlay Options: Normal ray profile

☑ Overlay Mesh Ray Profile

Metadata Ray Profile

Meta Data

Location (x, y): (117.00, 251.00)

Start (CT, Mesh): (143.00, 148.21)
Width (CT, Mesh): (34.00, 24.44)

CT ray profile

- minimum : 28270.434
- maximum: 59624.914
- stnd dev : 9519.151
- mass : 1777953.250

Figure 3.12: Comparison system.

If you think in terms of a year, plant a seed; if in terms of ten years, plant trees; if in terms of 100 years, teach the people.

Confucius

4

Locally Adaptive Marching Cubes

THE EXTRACTION of a constant density surface from a dataset (contouring) is used in many disciplines. In medical environments it has opened new ways for radiologists and physicians to visualize and interact virtually with the human body. In metrology (the science of measurement) industrial 3D X-ray Computed Tomography (3DCT) scanners are used for specimen measurements and NDT.

The most well-known algorithm for extracting a polygonal representation (e.g., set of triangles) of a constant density surface (iso-surface) from a 3D dataset is marching cubes and was published by Lorensen and Cline in 1987 [44]. The marching cubes algorithm works with the *divide and conquer* principle. The volumetric dataset is divided into cells and the iso-surface within each cell is calculated. It uses a global threshold called iso-value to determine the interior and exterior of the iso-surface in each cell.

A volumetric dataset often is subject to biases. Such biases can be low frequency noise, contrast drifts, local density variations and other artifacts introduced by the measurement process. Examples are noise-induced streaks, aliasing, beam-hardening, scattered radiation effects, cupping etc. [26]. When performing contouring on a biased dataset with a global iso-value the result may not be satisfactory. The iso-surface may contain holes, thinned regions, or regions where volume is added. An iso-value field where every vertex has its own iso-value enables the contouring algorithm to compensate for flaws. We want to locally modify the iso-value and with it the surface in those regions of the dataset where measurement errors would cause an unsatisfactory surface to be generated. Additionally an iso-value field supports blending between various iso-surfaces by linearly interpolating between iso-values.

The *locally adaptive marching cubes* algorithm modifies the marching cubes algorithm and allows the specification of an iso-value field. The modifications to the marching cubes algorithm include a slightly different case identification process and a changed computation of intersections between the surface and cell edges. Marching cubes uses linear interpolation to compute intersections. In the modified algorithm we intersect two general line segments that are defined by two density values and two iso-values.

In the following we list some related work on iso-surface correction. Then we provide a brief overview on the marching cubes algorithm and present our modifications to transform it into the *locally adaptive marching cubes* algorithm. In section 4.3 we present a reference application that supports the specification of iso-value fields on volumetric datasets. Another application visualizes the surface in the interior of a single cell. We conclude by discussing simple and more complex iso-value fields on various volumetric datasets.

4.1 Related Work

Common approaches to correct iso-surfaces are either to modify the input dataset and perform contouring with a global iso-value or optimize the iso-surface after generation.

Heinzl et al. [26] use Dual Energy Computed Tomography (DECT) and image fusion to reconstruct a dataset before extracting an iso-surface. As input images they use a high-energy macro focus image, which is blurry but less affected by artifacts, and a low-energy micro focus image, which is precise but artifact-prone. The overall structure of the specimen is taken from the low precision dataset while the sharp edges are derived from the high precision dataset. For contouring, a local surface extraction approach is used. First an iso-surface is extracted from the dataset, which describes the real surface in good approximation. In the next step the iso-surface is corrected by moving each surface vertex along the direction of the surface normal until the gradient magnitude in the volume reaches a maximum. The gradient magnitude is computed from the low energy dataset.

Kobbelt et al. [38] developed an extended marching cubes algorithm to compensate for aliasing artifacts at sharp features of the extracted surfaces. First, their algorithm detects those grid cells that are intersected by a sharp feature of the surface. Then it computes additional sampling points that lie on the feature and inserts them into the mesh. Additional sampling points are retrieved from a local distance field and its gradient.

Matyas et al. [50] present an automated contouring approach with locally changing iso-values to determine anatomic branching structures. They first segment the dataset and determine the appropriate iso-value in each segment. Iso-values do not vary continuously but in discrete steps, which requires to blend together the extracted iso-surfaces of the segments. They named the resulting surface "metasurface".

Šereda [79] proposes local transfer functions to adapt to locally modified data values in the dataset. A transfer function (TF) is used to transform density values into optical properties such as color and opacity. Typically a global TF is defined. In his work he discusses how the definition of TFs can be automated and how local TFs produce better results on datasets with local data variations.

4.2 Locally Adaptive Marching Cubes

In the following subsections we first briefly describe the marching cubes algorithm and then present the necessary modifications to transform it into our *locally adaptive marching cubes* algorithm.

4.2.1 The Marching Cubes Algorithm

The algorithm [44] subdivides the volumetric dataset into cubical cells and processes each of them separately. To identify the triangulation of the iso-surface inside a cell, the density value at each vertex is compared with the global iso-value to check if the vertex lies inside or outside the iso-surface. The possible triangulations of a cell are stored in a case lookup table. There are 256 possible cases, which can be reduced by topological and rotational symmetry to 15 cases. In the original algorithm there exist ambiguous cases where more than one triangulation is possible. This can lead to holes in the iso-surface. A method to overcome this problem is the *asymptotic decider* [53]. A simple solution adds six additional cases to the case table to avoid holes [69]. After retrieving the triangles for a case from the lookup table, the positions where the triangle vertices lie on the cell edges have to be calculated. Roughly four steps are performed for each cell:

1. identify the appropriate index case
2. look up triangles in the case lookup table
3. interpolate intersection points along the cell edges
4. compute surface normals for triangle vertices (required for shading)

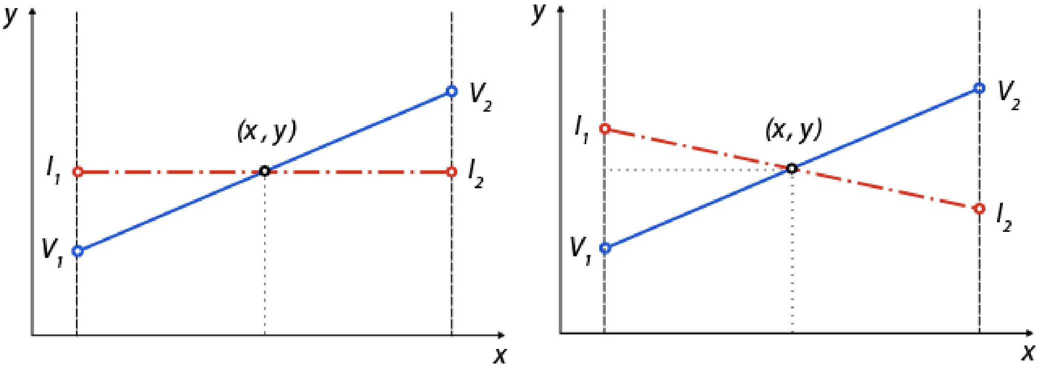
4.2.2 The Modified Algorithm: Support of Iso-Value Fields

Our *locally adaptive marching cubes* algorithm uses one iso-value per cell vertex instead of a single global iso-value. Each cell has eight iso-values, one at each vertex of the cube. Trilinearly interpolating these varying iso-values produces a continuous iso-value field. This alters how the index cases are identified and also how the intersection points along the cell edges are calculated.

In the case identification process the marching cubes algorithm compares every cell vertex value to the global iso-value. In the modified algorithm every vertex has its own iso-value to which the cell vertex value is compared. It is then decided whether the vertex is considered to be zero or one in the index, i.e., inside or outside. The remaining part of the case identification process is similar to the original algorithm.

The marching cubes algorithm uses linear interpolation to interpolate triangle vertices along the edges. Since both vertices of an edge have now different iso-values, the intersection point of two arbitrary line segments has to be calculated (figure 4.1).

We assume that the line segment of density values is defined by the two cube vertices V_1 and V_2 and the line segment of iso-values is defined by I_1 and I_2. In the following x corresponds to the spatial position along an edge and y corresponds



(a) Linear interpolation (b) Arbitrary line intersection

Figure 4.1: (a) linear interpolation in the original algorithm. (b) arbitrary line intersection in the modified algorithm. The dashed line (red) depicts iso-values and the solid line (blue) shows density values. In (a) the slope of the iso-value line is always zero. In (b) it can have any value.

to density values or iso-values. Line equations are of the form $A_j \cdot x + B_j \cdot y = C_j$, $j = 1, 2$. V_j^x refers to the x-component of V_j.
We assign:

$$A_1 = V_2^y - V_1^y$$
$$B_1 = V_1^x - V_2^x$$
$$C_1 = A_1 \cdot V_1^x + B_1 \cdot V_1^y$$

A_2, B_2 and C_2 are derived in the same way from I_1 and I_2.
When solving our linear system:

$$A_1 \cdot x + B_1 \cdot y = C_1 \tag{4.1}$$

$$A_2 \cdot x + B_2 \cdot y = C_2 \tag{4.2}$$

we get:

$$x = \frac{C_1 \cdot B_2 - C_2 \cdot B_1}{A_1 \cdot B_2 - A_2 \cdot B_1}$$

$$y = \frac{C_2 \cdot A_1 - A_2 \cdot C_1}{A_1 \cdot B_2 - A_2 \cdot B_1}$$

If $A_1 \cdot B_2$ equals $A_2 \cdot B_1$ the lines are parallel and do not intersect. Even if an intersection point between the two lines exists, it may not lie on the line segments themselves. Hence we have to check if the intersection point lies between the line segment's endpoints:

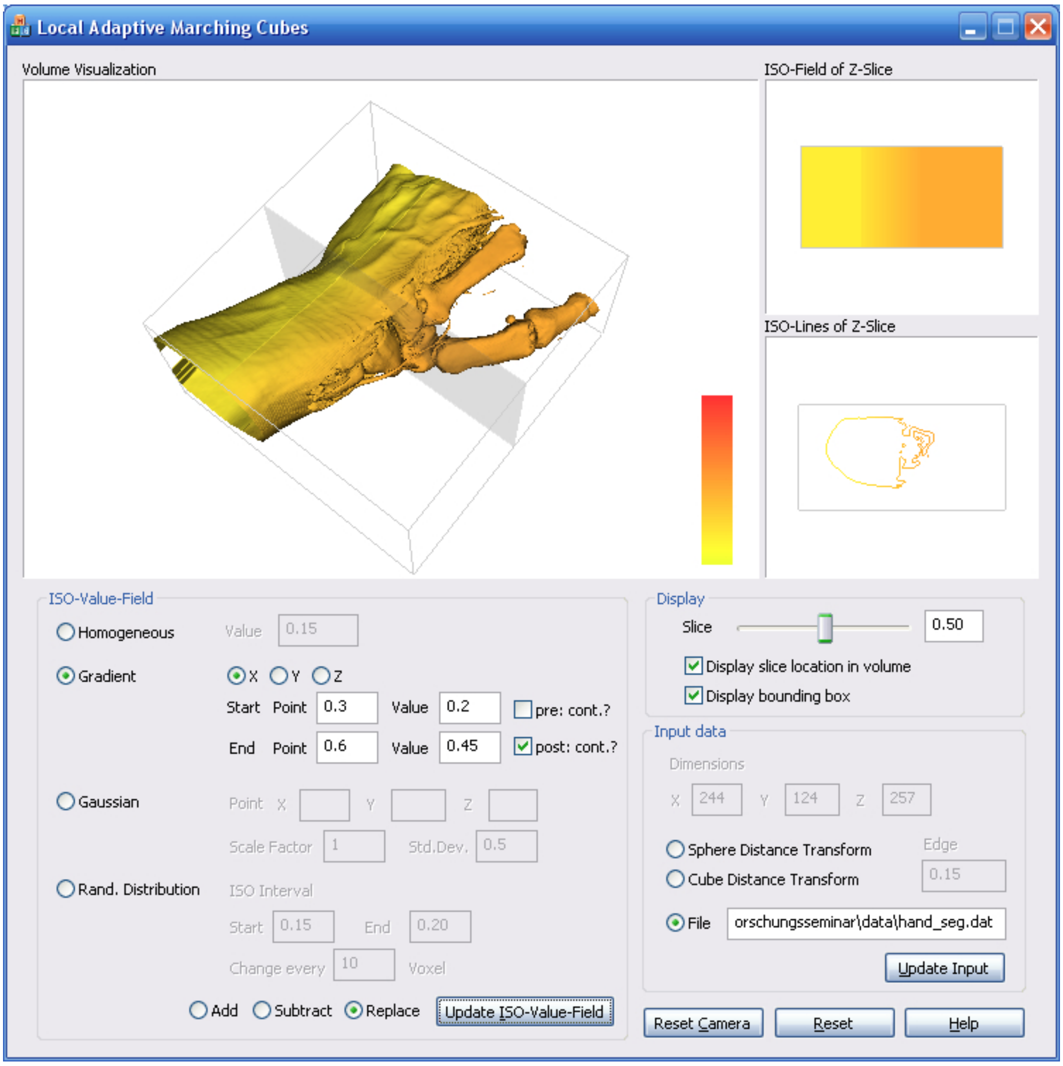
$$\min(V_1^x, V_2^x) \leq x \leq \max(V_1^x, V_2^x)$$

It is sufficient to check only the x-values of one line segment to decide if the intersection point is valid since the line segments are correlated (see figure 4.1).

4.3 Implementation

We implemented the algorithm with the proposed changes and created two basic applications to show the algorithm's possibilities. One application (*Locally Adaptive Marching Cubes*, figure 4.2) allows the specification of iso-value fields on volumetric datasets. The second one (*Cube Insight*) visualizes the interior of a single cubical cell using trilinear interpolation.

The graphics and visualization part of the applications is handled by *The Visualization Toolkit* (VTK) [68]. Using VTK for the reference applications has several advantages. First, the toolkit is popular in the visualization community and is open source. Second, an implementation of the marching cubes contouring filter is already available and can be easily modified.



Local Adaptive Marching Cubes

ISO-Value-Field

○ Homogeneous Value 0.15

● Gradient ● X ○ Y ○ Z

Start Point 0.3 Value 0.2 ☐ pre: cont.?

End Point 0.6 Value 0.45 ☑ post: cont.?

○ Gaussian Point X Y Z Scale Factor 1 Std.Dev. 0.5

○ Rand. Distribution ISO Interval Start 0.15 End 0.20 Change every 10 Voxel

○ Add ○ Subtract ● Replace

Display

Slice 0.50

☑ Display slice location in volume

☑ Display bounding box

Input data

Dimensions X 244 Y 124 Z 257

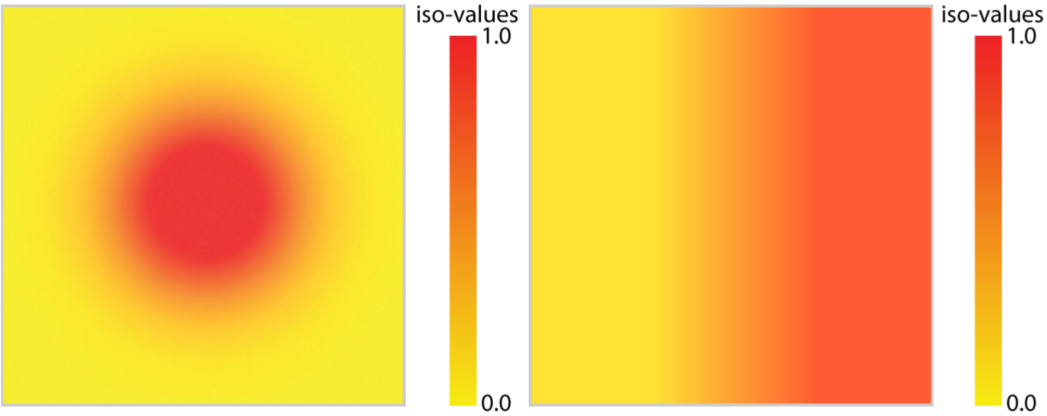
○ Sphere Distance Transform

○ Cube Distance Transform

Edge 0.15

● File orschungsseminar\data\hand_seg.dat

Figure 4.2: Locally Adaptive Marching Cubes allows the specification of iso-value fields on volumetric datasets. It is possible to specify homogeneous iso-value fields, fields with linear iso-value gradients in coordinate directions, and isotropic 3D Gaussian distributions of iso-values centered at specific sampling points of the dataset.



(a) Gaussian distribution (b) Linear gradient

Figure 4.3: (a) isotropic Gaussian distribution added to a homogeneous iso-value field. (b) linear gradient in x-direction with range [0.3, 0.7].

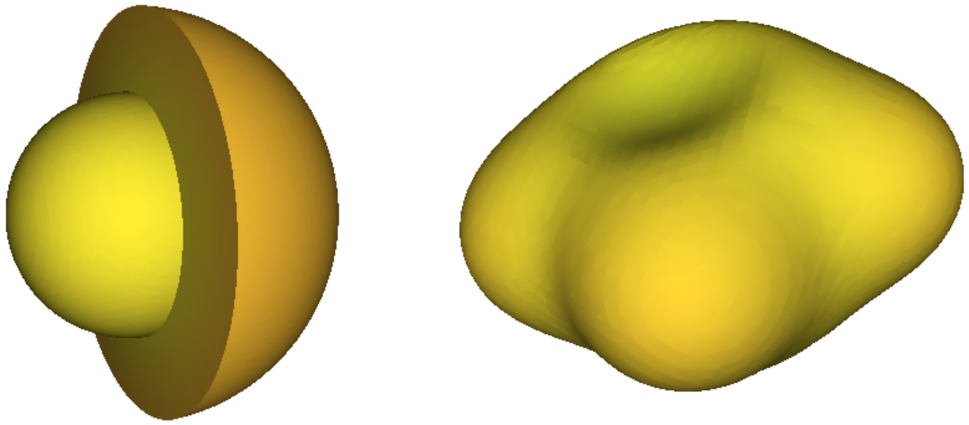
The generation of iso-value fields requires user interaction. There exist various possibilities to alter the iso-value field. Our application supports fields with linear iso-value gradients and isotropic 3D Gaussian distributions of iso-values centered at specific sampling points of the dataset (figure 4.3).

4.4 Results

We now discuss several iso-value fields on various volumetric datasets. First, we show how iso-value fields impact the appearance of simple objects. Then we present a biased dataset and discuss how the surface can be corrected with our method. Furthermore we show how our technique supports blending between different iso-surfaces and discuss how the surface inside a cell is modified when iso-values at the cell vertices are altered.

4.4.1 Simple Objects

In figure 4.4 the volume is a Euclidean distance transform of a sphere. The distance transform is generated by computing the Euclidean distance of each sample point from the center of the dataset. Distance transforms of simple objects like a sphere are well suited to show how different iso-value fields change the appearance of the surface. Continuous iso-value modifications must lead to continuous changes in the surface of the sphere (bulges and dents). Since we have a distance field, increasing iso-values lead to an increased sphere radius, decreasing iso-values lead to a decreased radius of the resulting surface. In figure 4.4a the iso-value field is divided into two halves, one with a lower and one with a higher iso-value which leads to two half-spheres. Figure 4.4(b) shows the sphere with a more complex iso-value field. Gaussian distributions of standard deviation 5 have been added to a homogeneous iso-value field at three locations and one Gaussian distribution has been subtracted from it. This leads to three bulges and one dent in the surface.



(a) Sphere dataset

(b) Sphere dataset

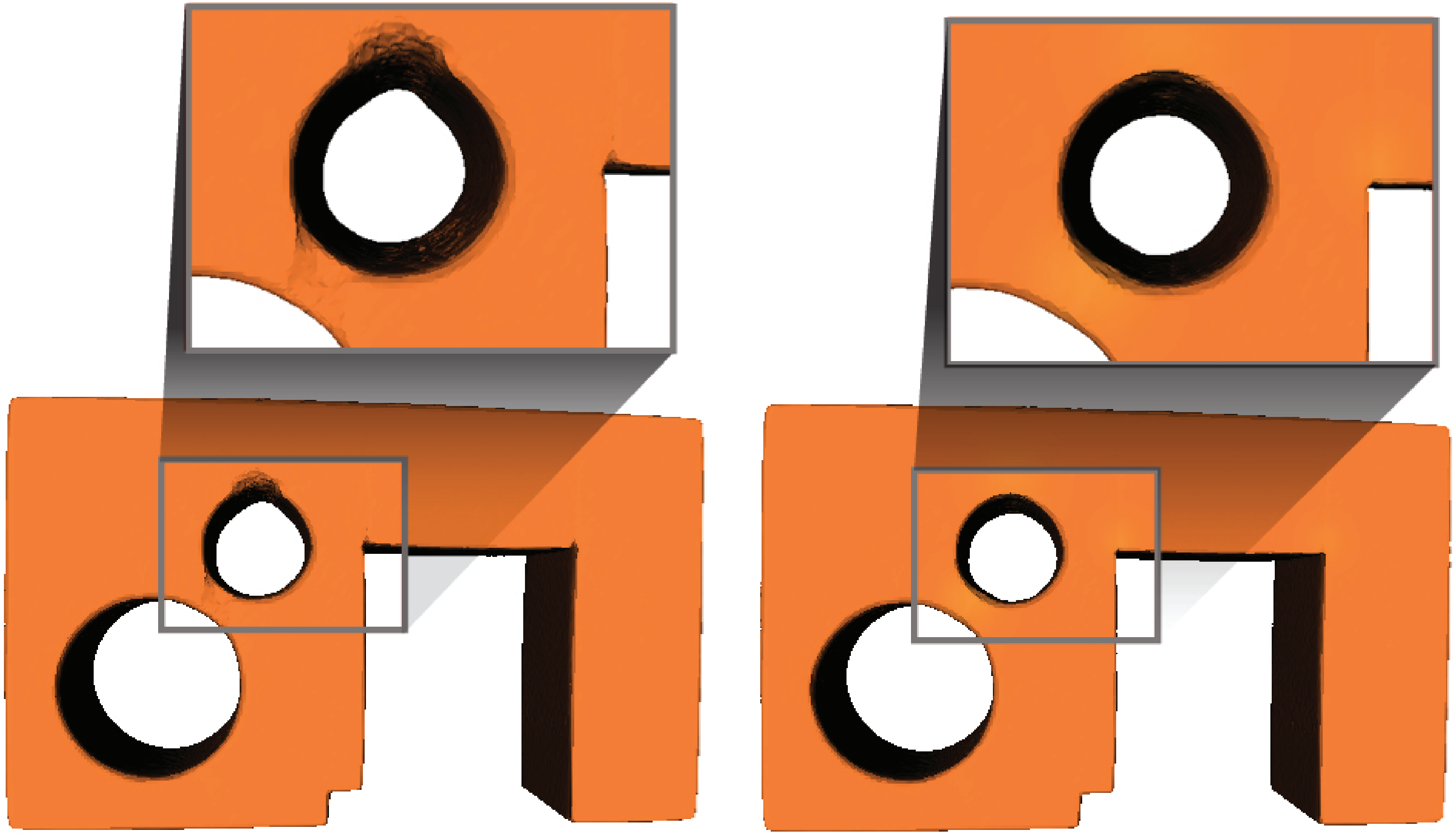
Figure 4.4: The data is a Euclidean distance transform of a sphere (resolution: $64 \times 64 \times 64$). In (a) the iso-value field is divided into two halves. In (b) a homogeneous iso-value field has been modified with a Gaussian distributions of standard deviation 5 at four locations.

Modifying the iso-value field in such a way can be considered as digitally painting in the 3D iso-value field, where the brush is given by the Gaussian distribution. Various extensions and variations to this painting metaphor and brush types are easily conceivable.

4.4.2 Correcting the Iso-surface of a Biased Volumetric Dataset

The specimen depicted in figure 4.5 has severe scattered radiation and beam hardening artifacts in the area of the drill holes and also at the rectangular milling. The iso-surface in figure 4.5(a) shows several errors especially at the smaller hole where the geometry of the object is changed. Given that there exists an iso-value field where the surface is appropriate in that area of the dataset, it is possible to correct the surface with our method. The iso-value field is modified in those areas where flaws are present. To assure a continuous transition between iso-values an isotropic 3D Gaussian distribution is used. Figure 4.5(b) shows the resulting surface. Several Gaussian distributions of various kernel sizes have been subtracted from the homogeneous iso-value field of value 0.65. The flaws at the drill holes and at the corners of the milling are thus corrected.

The approach described here requires user interaction to generate the iso-value field. The user introduces his knowledge about the object geometry into the process. One can think also of automated approaches to surface correction using iso-value fields. By generating special (e.g., data-dependent) iso-value fields the information that corrects artifacts can be passed to the extracting algorithm as input instead of changing/correcting the extracted surface afterwards.



(a) Global iso-value

(b) Iso-value field

Figure 4.5: A volumetric dataset (resolution: 164 × 263 × 90) that contains artifacts. (a) shows the iso-surface extracted with a global iso-value of 0.65. There are several flaws visible in the surface especially at the smaller hole. (b) depicts the surface that was extracted with a modified iso-value field and has most of the flaws corrected.

4.4.3 Blending Between Iso-surfaces

Linear gradients in iso-value fields can be used to blend between different iso-surfaces in one processing step. The interpolation of iso-values in a dataset region leads to a continuous transition between iso-surfaces. In figure 4.6 a CT hand is shown. A linear iso-value gradient in the x-range [0.25, 0.6] with the iso-value range [0.15, 0.4] is used to visualize the surface of skin on one half and the surface of bones and veins on the other half of the dataset. A contrast medium has been injected into the veins, since there is almost no difference between their density value and the density value of bone. In the result image it becomes visible how the skin gets thinner, dissolves and then exposes the bones in a smooth way.

In figure 4.7 a human tooth is depicted which roughly consists of the enamel (top part of the tooth) and the dentine (bottom part of the tooth). We assume that the enamel should be visualized with an iso-value of 0.7. With that value it is not possible to visualize the dentine because it starts to dissolve at iso-values greater than 0.6 (figure 4.7(a)). A solution for this problem could be to generate iso-surfaces for the dentine with iso-value 0.5 and for the enamel with iso-value 0.7 and then combine both iso-surfaces. With our method this is not necessary. A linear iso-value gradient in the z-range [0.7, 0.75] with iso-value range [0.5, 0.7] can be used to visualize the entire tooth in one processing step (figure 4.7(b)) without the need to combine two generated iso-surfaces.

4.4.4 Surface in the Interior of a Cell

Figure 4.8 shows the surface in the interior of a cell for three index cases. Generally modifications of iso-values result in the movement of intersection points on the cell edges. There can also be topology changes and the index case might change as well. Increasing the iso-value at the circled vertex in figure 4.8(a) alters the index case. An opening in the surface appears. In figure 4.8(b) the index case is not changed, but intersection points are moved. First both "wings" of the surface have equal height. After increasing the iso-value at the circled vertex, one "wing" is lower than the other. The modification of the iso-value in figure 4.8(c) connects both components of the surface and changes its topology while leaving the index case unchanged.

4.5 Conclusion

This chapter describes *locally adaptive marching cubes*, a modification of the marching cubes algorithm. It allows the usage of an iso-value field instead of a single global iso-value. Iso-value fields are generated by user specification and are independent from the volume data. Surfaces are extracted, which are a continuous blend between various iso-surfaces. The necessary modifications to the algorithm are simple. First, the case identification process has to be altered since we now have a different iso-value for each cell vertex. Second, when finding the intersection points of the surface with the cell edges, we now have to compute the intersection of two line segments. Modifications of iso-values can impact the surface in the interior of a cell in three ways. First, the modification of iso-values can change the index case as in figure 4.8(a). When the index case is not changed, the modification

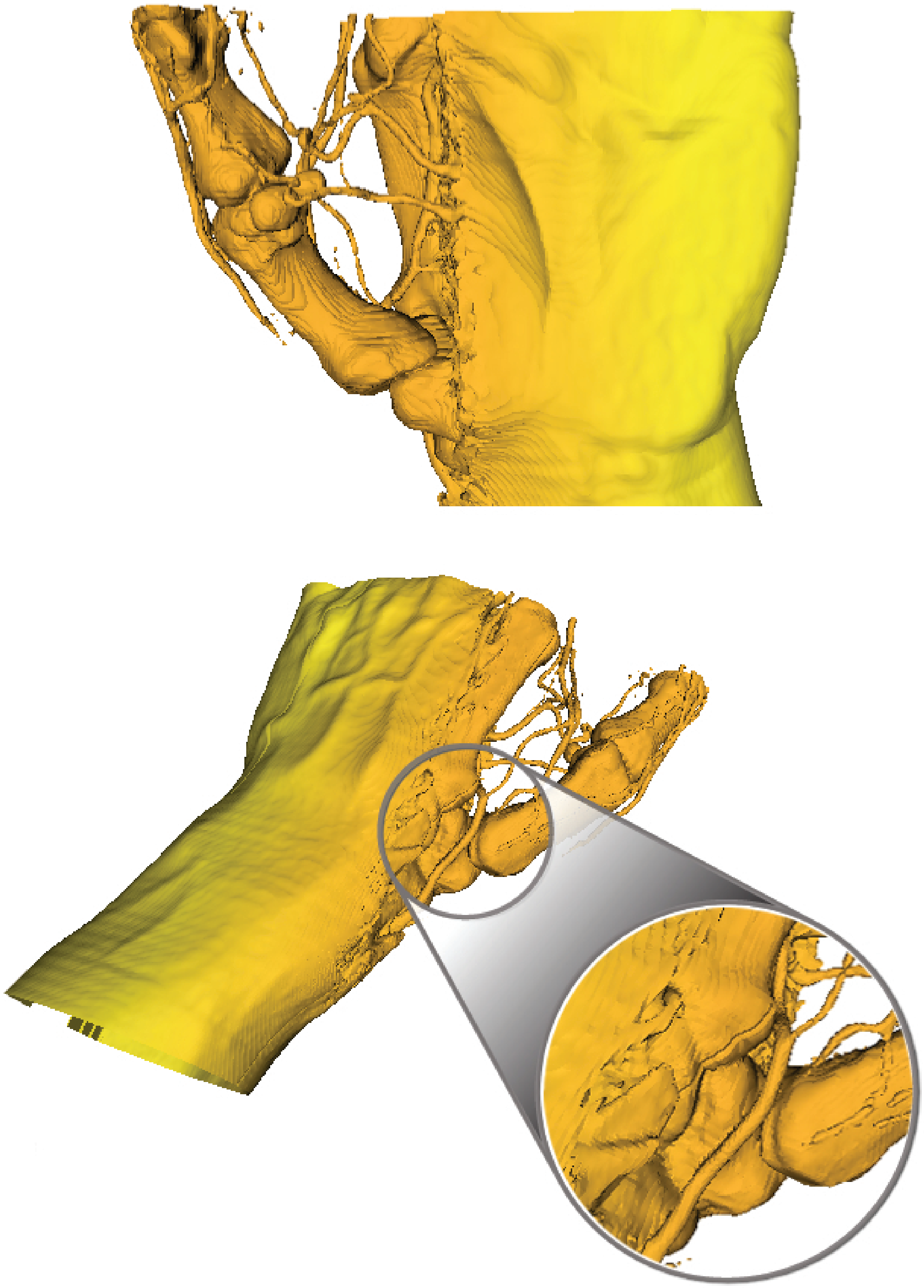


Figure 4.6: A human hand (resolution: 244 × 124 × 257) with linear blending between the iso-surfaces of skin and bone.

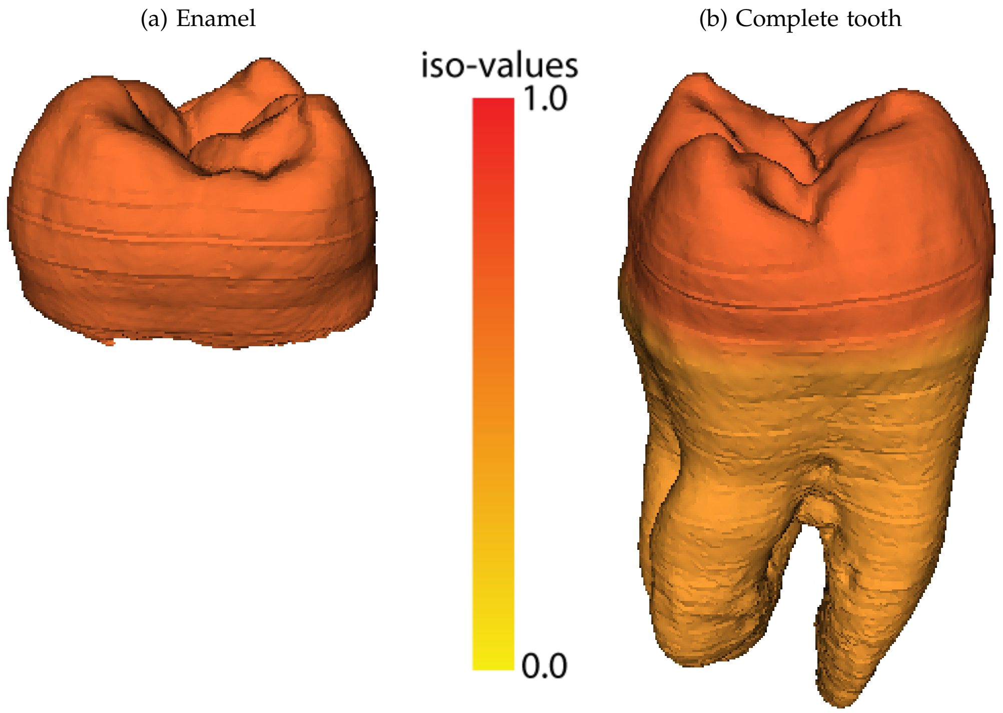


Figure 4.7: A human tooth dataset (256 × 156 × 161). (a) generated with a global iso-value of 0.7, only the enamel part of the tooth is displayed. (b) an iso-value field with a linear gradient in the z-range [0.7, 0.75] is used to blend between the iso-surface of the dentine part and the iso-surface of the enamel part of the tooth.

of iso-values moves intersection points of the surface with the cell edges as in figure 4.8(b). There can also be topology changes as in figure 4.8(c).

Locally adaptive marching cubes can be used for the correction of iso-surfaces with flaws like low frequency noise, contrast drifts, and local density variations. The iso-value field is appropriately modified in those regions of the dataset where flaws are present. Our technique also supports blending between different iso-surfaces by specifying iso-value gradients in the iso-value field. This is useful when the entire dataset can not be visualized with one global iso-value, or when different iso-surfaces should be visualized in different regions of the dataset. Blending between iso-surfaces normally requires to first extract all iso-surfaces and then combine them in an additional processing step. Our algorithm supports blending directly in the contouring process.

Two reference applications have been presented. One application allows the specification of iso-value fields and of volumetric datasets. A second application visualizes the interior of a single cell. Possibilities to enhance our system include improving the user interface and improving the process of defining modifications to the iso-value field. When correcting iso-surfaces with flaws due to artifacts there may be automated ways to specify the iso-value field.

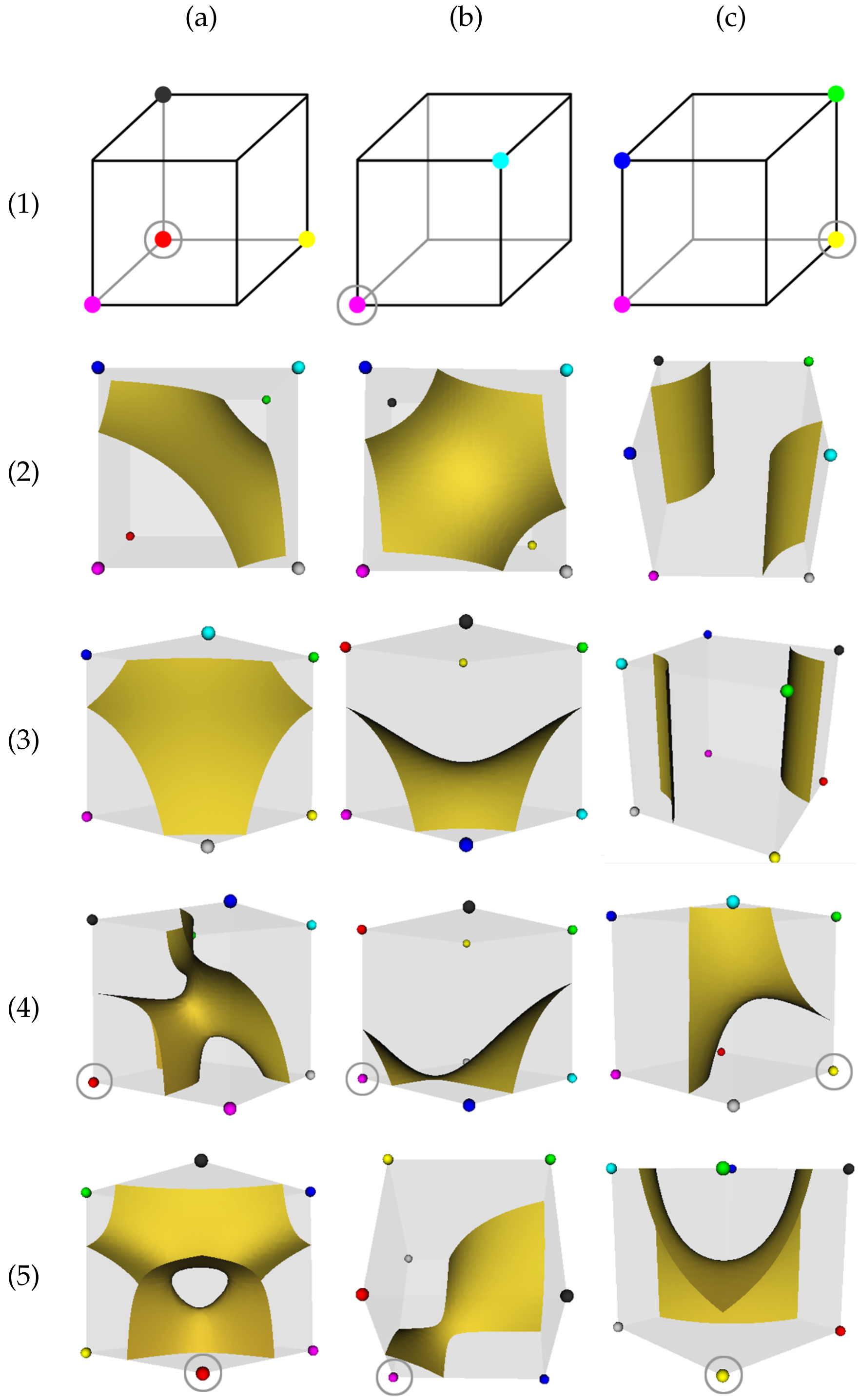


Figure 4.8: Visualization of the surface in the interior of a cell for three index cases. (1) depicts the index case. A cell vertex marked with a dot indicates that the density value exceeds its iso-value.The surface intersects those edges where one vertex is marked with a dot while the other one is not. (2) and (3) show the case from two different viewpoints while (4) and (5) show corresponding views where the iso-value of a vertex is modified. Vertices with modified iso-values are circled. Density values are set to 0.1 for non-marked vertices and are set to 0.25 for marked vertices. The iso-values are set to 0.15.

Whatever is produced in haste goes
hastily to waste.

Saadi

5

Comparative Visualization for Parameter Studies of Dataset Series

Industrial 3D X-ray Computed Tomography (3DCT) is a radiographic method to locate and measure volumetric details of industrial components. A 3DCT scanner produces a 3D grid of gray values (voxels) corresponding to the spatial density of the specimen. The voxels are typically stored with either 16 bit or 32 bit precision. Contemporary volumetric datasets typically have a high resolution and are quite large. A 3DCT dataset is not calibrated and is more prone to artifacts than a conventional radiography [30].

3DCT is used in industry for NDT and for first part inspection. Application areas include the automotive, aeronautic, electronic, and leisure industry, as well as many other industrial fields. Quality-control engineers compare the volumetric datasets scanned using a 3DCT with the design specifications of the industrial components, in order to evaluate the precision and accuracy of the production pipeline.

A wide variety of 3DCT scanners have been introduced since the first computed tomography (CT) machine was produced in 1972. Microtomography (MicroCT) [16] and Nanotomography (NanoCT) devices were produced in 1982 and 2005 respectively. The basic principle of data acquisition in MicroCT and NanoCT is the same as in early CT machines. The main difference is the size of the focal spot in the X-ray source and the size of the detector pixels. The size of the focal spot is in micrometers and nanometers in MicroCT and NanoCT respectively. A low energy micro focal spot has smaller penetration length but it results in higher resolution volumetric datasets. On the other hand, using a high energy macro focus X-ray source better captures the structure of the specimen as it has a higher penetration length. The resolution of the dataset however, is lower than the one produced by a low energy micro focus X-ray source due to the larger focal spot. Lately, dual energy CT devices have been introduced to combine the advantages of high energy and low energy X-ray sources [62].

Table 5.1: Various dataset series and their specifications. The number of datasets varies from series to series. The voxel size is isotropic.

Dataset series, parameter varied	Dataset resolution	No. of datasets	Series size	Voxel size
voltage	$1000 \times 1000 \times 882$	5	8.21 GB	122 μm
no. of projections	$1000 \times 1000 \times 882$	6	9.86 GB	136 μm
current	$856 \times 856 \times 882$	6	7.22 GB	122 μm
integration time	$800 \times 800 \times 882$	5	5.26 GB	136 μm
filter plates	$848 \times 848 \times 882$	15	17.72 GB	122 μm
mean value dataset	$1000 \times 1000 \times 882$	3	4.93 GB	136 μm

3DCT machines provide a large parameter space which has to be explored and specified before performing a CT scan. These parameters can severely influence the quality of a dataset depending on the material and the geometry of the specimen. The parameters which can be manipulated include voltage, current, integration time, number of projections (attenuation measurements), thickness and alloy of the X-ray filtering plates etc.

Unfortunately the parameters cannot be changed in real-time with live feedback from the 3DCT as a typical 3DCT scan takes approximately 30 to 45 minutes. The volumetric dataset is then examined to see if it meets the user's requirements. In case the volumetric dataset is not up to the user's expectations, the parameters are changed and the whole process is repeated again. This is a very costly procedure. It is therefore highly interesting for the 3DCT users to study the changes induced in the volumetric datasets by the parameter variations. It helps to predict those parameter settings which will introduce the least amount of artifacts in the volumetric dataset of a given specimen. We present an innovative comparative visualization system for the study of 3DCT parameters. Our system can be used for the comparative visualization of dataset series produced by a wide variety of imaging devices. In this chapter we specifically describe our experiments using 3DCT datasets.

We visualize the 3DCT datasets in collaboration with our company partners. A dual source HWM Rayscan 250XE device (voxel size: ≥ 5 microns, 225keV micro and 450keV macro focus X-ray source) is used to generate CT datasets. A dataset series is generated by varying a single parameter over the entire parameter range. Table 5.1 gives information on the various dataset series that were used to test and evaluate our techniques. All the datasets in a series were generated in a single scan session without removing the specimen from the 3DCT device. The datasets in a series are thus perfectly aligned and there is no need for registration.

Some of the parameters in table 5.1 have a known behavior. For example the X-ray source voltage may induce cupping at high penetration lengths due to beam hardening. Beam hardening is caused by high attenuation of X-rays moving through long stretches of highly absorbing materials. These so called high penetration lengths result in lower gray values in the center of the object. It may become hard to differentiate between different materials in such a scenario. Not enough projections in a single 360 degree turn of the rotary plate or insufficient X-ray source current may result in streaking artifacts. Streaking artifacts are bands of low or high gray values due to insufficient sampling. A mean value dataset is produced by measuring each projection image multiple times and evaluating a mean projection image. A mean value measurement is supposed to improve the signal-to-noise ratio of the volume data but it increases the time and cost of each scan.

The huge amount of data produced by a 3DCT makes it difficult to provide interactive comparison and 3D visualization for the volumetric datasets (table 5.1). The amount of main memory in the latest computer systems has increased considerably but the memory in the graphics hardware remains the bottleneck. Graphics hardware is crucial for interactive rendering of the volumetric datasets. We have a varying number of 3D datasets for every parameter and the amount of data that needs to be processed is rather large. Efficient management of the data and interactive rendering is however possible through planar reformatting, i.e., 2D planar slices of the data taken at arbitrary angles.

3DCT users are familiar with planar reformatting based systems for a long time. Many commercial tools like VGStudio MAX [78], which are used by 3DCT professionals, are designed using planar reformatting. Planar reformatting is fast and easy to use and our planar reformatting based system is scalable with the number of datasets in a series. Scalability can be a hard issue to resolve in 3D. 3D visualization has occlusion issues and also 3D visualization does not provide as much detail as we can extract using planar reformatting. The use of 3D visualization therefore reduces the advantage gained through the use of latest MicroCT machines.

The differences between the datasets of a series do not typically show up as distortions in shape. The differences are in terms of variable gray values caused by diverse parameter settings. This variability in datasets is better captured and analyzed in 2D. Planar reformatting suits our design requirements as it provides interactive and detailed exploration, comparison, and visualization of large dataset series.

We propose two techniques for comparative visualization in this chapter. The *multi-image view* allows the simultaneous viewing and comparison of the entire dataset series. The *edge explorer* provides comparative visualization of edges in a dataset series. Our system automatically scales according to the number of datasets under active comparison. We use efficient algorithms and data structures for fast retrieval of data from the main storage device.

5.1 Related Work

The simultaneous display of multiple datasets is an interesting topic in many fields. In the medical domain datasets from different modalities are first registered and then visualized by the radiologists for diagnosis. The aim of multi-modal visualization is to render different parts of a volume using the modality which best captures the respective object part [10]. Kniss et al. combine segmentations resulting from multiple algorithms into one probabilistic space for visualization [36].

In fields like weather simulation and sociology, the users are also interested in the simultaneous display of multiple parameters. There has been a whole body of work on techniques for the simultaneous visualization of multiple parameters [70]. Kirby et al. combine multiple values in a 2D flow image for a simultaneous display [33]. Laidlaw et al. use ellipsoids and brush strokes of varying intensity to visualize tensor images [42]. Ebert et al. use blobby objects for the simultaneous rendering of multiple parameters [15]. The blobby objects are 3D shapes and therefore the technique is prone to occlusion issues. Taylor uses color coding and orientation of simple shapes to visualize multiple datasets on a single surface [72].

Data is either replicated or fused in space for a concurrent display of parameters. Urness et al. introduce color weaving for flow visualization [75]. Gossett and Chen present the fusion of multiple parameters through color blending [21]. Hagh-Shenas et al. carried out a comparison of blending and weaving techniques [23] for visualizing multiple parameters. They find weaving to be better suited for conveying information. The weaving pattern's ability to convey information however reduces considerably when the number of parameters exceeds four. Healey and Enns propose to vary color and texture attributes for the simultaneous display of multiple parameters [25].

Miller proposes to display parameters side by side in attribute blocks [51]. His work is in the spirit of weaving patterns. Attribute blocks have a well defined structure and therefore the technique seems to be scalable with the number of parameters. These systems are however not specifically designed for the exploration of differences between datasets at a voxel level.

As our work focuses on planar reformatting, research on image comparison is of interest in this context. Gesú and Starovoitov propose a distance function for comparing images [19]. The difference between images is computed as a single number. Yang et al. [83] use the Hausdorff distance and the normalized gradient to evaluate the similarity between images. A survey of image comparison metrics and algorithms is provided by Zhou et al. [84] and Radke et al. [60]. Image comparison research has mainly focused on producing a similarity metric for images without focusing on visualization techniques. Visualization provides important spatial information about the differences between images.

Interactive rendering of multiple volumes is a challenging task. The amount of memory available in graphics hardware is limited. Grimm et al. [22] propose bricking and an efficient addressing scheme for large datasets that do not fit into the main memory. Kohlmann et al. [39] describe a bricked volume layout for planar reformatting based exploration of large medical datasets. Our implementation is based on the results achieved by Grimm et al. and produces interactive visualizations of large dataset series.

It is necessary to study the impact of our research on the workflow of the domain specialists. Kosara et al. [41] and Bartz et al. [3] provide a thorough description on the dynamics of user studies in computer graphics. We collect feedback from domain specialists for verifying the utility and robustness of our methods.

5.2 Comparative Visualization

We introduce two novel techniques for comparative visualization in this chapter, namely *multi-image view* and *edge explorer*. Furthermore, we also propose a variety of tools for user interaction.

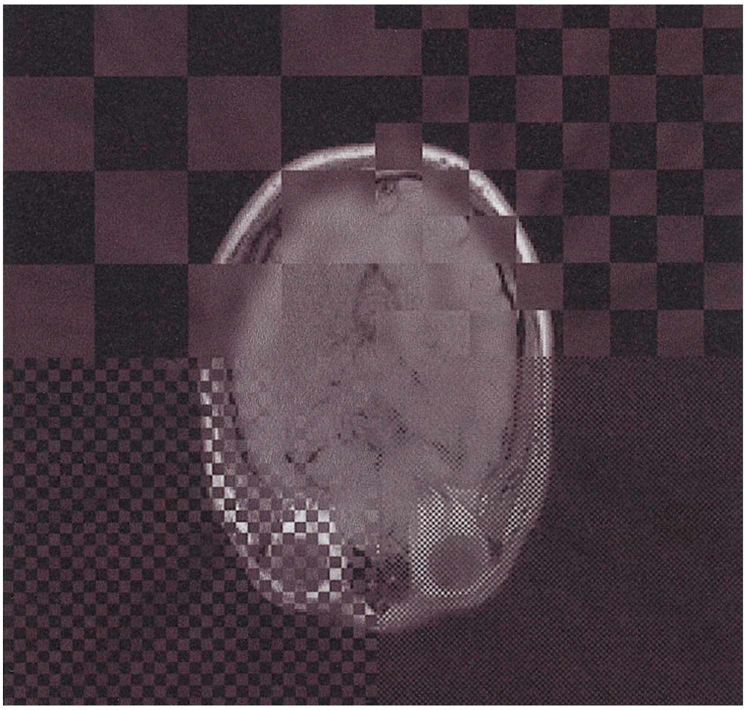
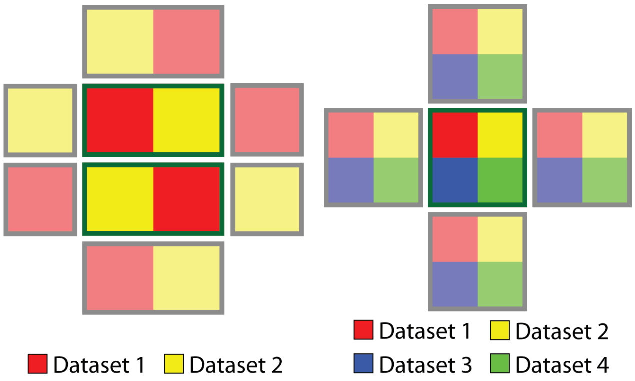


Figure 5.1: SPECT and MR slices displayed in a checkerboard pattern [71].

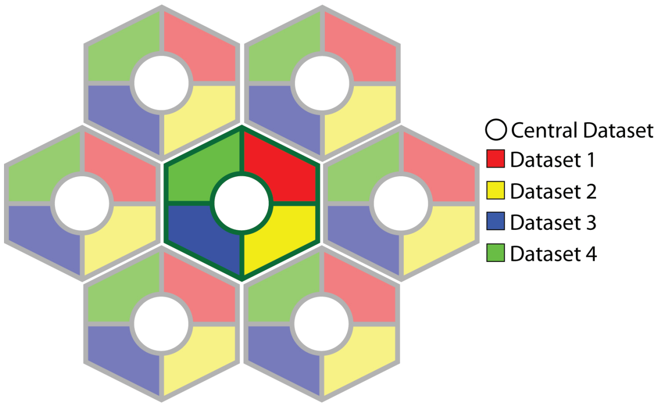
The natural choice for comparing two slices or images will be to either display them side by side or to display them in a single checkerboard display. The checkerboard display has been previously used for multi-modal datasets. For example Stokking et al. [71] display single photon emission computed tomography (SPECT) and magnetic resonance (MR) images in a checkerboard arrangement. The checkerboard may be used to simulate the transparency effect and to verify the registration of the datasets (figure 5.1).

We have a varying number of datasets to compare and visualize. The checkerboard display handles just two datasets and is unsuitable for our purposes (figure 5.2(a)). For the evaluation of a dataset series, a representation is needed which scales with the number of datasets. Attribute blocks can be used for simultaneously displaying multiple parameters or datasets [51]. The plane is thereby tiled with a rectangular cell pattern, where a cell is a square dedicated for displaying a specific parameter. We intend to perform comparative visualization and attribute blocks may not be ideal for our specific goal.

We want a scheme that is space filling and has a regular pattern like a checkerboard. A regular pattern is quite important for the user. It helps him to get acquainted with the spatial arrangement and to comfortably make use of it in everyday work. If the above two conditions are enforced, no cell of an attribute block shares a border with every other cell when the number of datasets being visualized is more than three (figure 5.2(b)). In such a case it is not possible to compare datasets across cell boundaries. Attribute blocks were specifically designed for parameter visualization. The structure of the attribute blocks therefore seems less scalable for comparative visualization. We want a visualization mechanism where multiple datasets can be displayed and compared easily.



(a) Checkerboard pattern (b) Attribute blocks



(c) *Base tile* pattern for *multi-image view* (5 datasets)

Figure 5.2: (a) checkerboard pattern for two datasets. (b) attribute blocks with four cells. No cell shares edges with all other cells. (c) *multi-image view* for comparative visualization.

In the *multi-image view* the idea is to divide the plane into circles rather than squares. A circle can be easily divided into multiple, equally sized sectors where each sector displays a specific dataset of the series. Circles however are not space filling. A good approximation to a circle, which also tiles the plane, is a hexagon. So we divide our screen into tightly packed hexagons, where the user can control the size and location of the hexagons.

Each hexagon has a circle in the middle and is divided into multiple sectors as shown in figure 5.2(c). Corresponding slices of the datasets from a series are displayed in each sector of the hexagon and in the central circle. The dataset which is displayed in the circle is called the *central dataset*. All other datasets are compared and visualized with respect to the *central dataset*. The *central dataset* is neighboring every other dataset in the series. Every hexagon provides a concurrent view into multiple datasets. We name our custom designed hexagon a *base tile*. The tiling of the plane with space filling *base tiles* is called *multi-image view*.

5.2.1 Multi-image View

At the beginning of the evaluation, the first dataset in a series is used as the *central dataset*. The rest of the datasets are initially assigned to the sectors of the hexagon in a clockwise orientation. The *central dataset* always displays the gray values of a planar reformatted slice of the corresponding dataset. The sectors of the *base tile*, which are compared to the *central dataset* either show gray values or color coded deviations. Deviations are calculated by computing the local difference between the gray values of the *central dataset* and the datasets in the sectors. The result is color coded according to a user provided color scale. The rendering mode, i.e., showing either gray values or deviations, can be switched in real-time.

Borland and Taylor present an interesting report on the use of different color scales [6]. They show that a rainbow color map lacks the perceptual ordering necessary for interpreting a color scale. We however choose the rainbow spectrum as the default color scale for the *multi-image view* due to two reasons. First, our company partners have been using the rainbow color scale for a long time and they are comfortable with its usage. Second, we have positive and negative values to display in the *multi-image view*. There is no uni-directional perceptual ordering in our scale and therefore we require multiple colors to show zero, positive values and negative values. We use the black body radiation spectrum for the *edge explorer*. The user has the freedom to set the color scale in our implementation.

Several other interaction possibilities are provided for effective comparison. For example, the placement of the datasets in the *base tile* is crucial for a comparative visualization. The user can rearrange the sectors in which the datasets are displayed by a simple drag and drop operation. The *central dataset* can also be interactively selected and changed by the user. Color coded deviations rendered in the sectors are updated according to the new *central dataset*. A legend helps the user to keep track of the current allocation of sectors to the datasets.

Any dataset, which is not interesting for the analysis or is an outlier can be removed from the comparison process. The number of sectors in each *base tile* will be reduced and rearranged by the system accordingly. A honeycomb frame of lines for visual distinction between the *base tiles* and to distinguish between sectors can be rendered over the result image. The transparency of the honeycomb is user controlled. The *base tile* pattern can be shifted freely over the plane with live update of the images. This uncovers otherwise hidden portions of the inspected slices. Small shifts allow a detailed comparison of the datasets across sector boundaries. Also, the size of the *base tiles* can be interactively controlled.

The user can choose to display just a single *base tile* instead of the space filling arrangement of multiple *base tiles* as in the *multi-image view*. The size and location of the single *base tile* is user specified. It is used for probing the dataset series and tracking artifacts. We found the use of a single *base tile* to be highly interesting for professionals when we evaluated our methods.

Figures 5.3 and 5.4 show the voltage dataset series. There are four datasets in the voltage series (150 kV, 180 kV, 210 kV and 220 kV), each recorded with 16 bits per voxel. We also use low-level fusion-techniques like a multi-scan fusion algorithm [27] to generate a fused dataset from the dataset series. Figure 5.3 shows a rendering of the gray values in a *multi-image view*. The fused dataset is the *central dataset*. We can observe differences in gray values of the datasets near object edges.

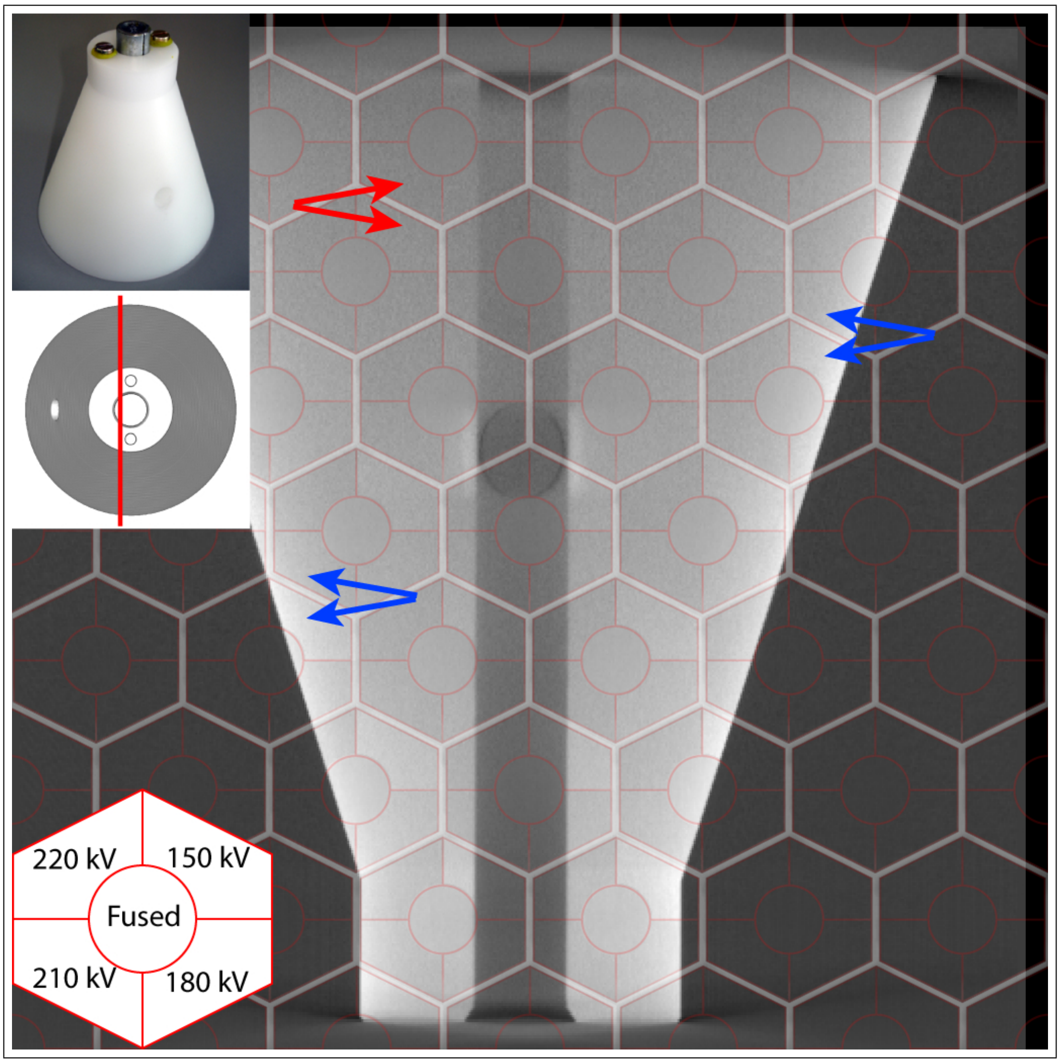
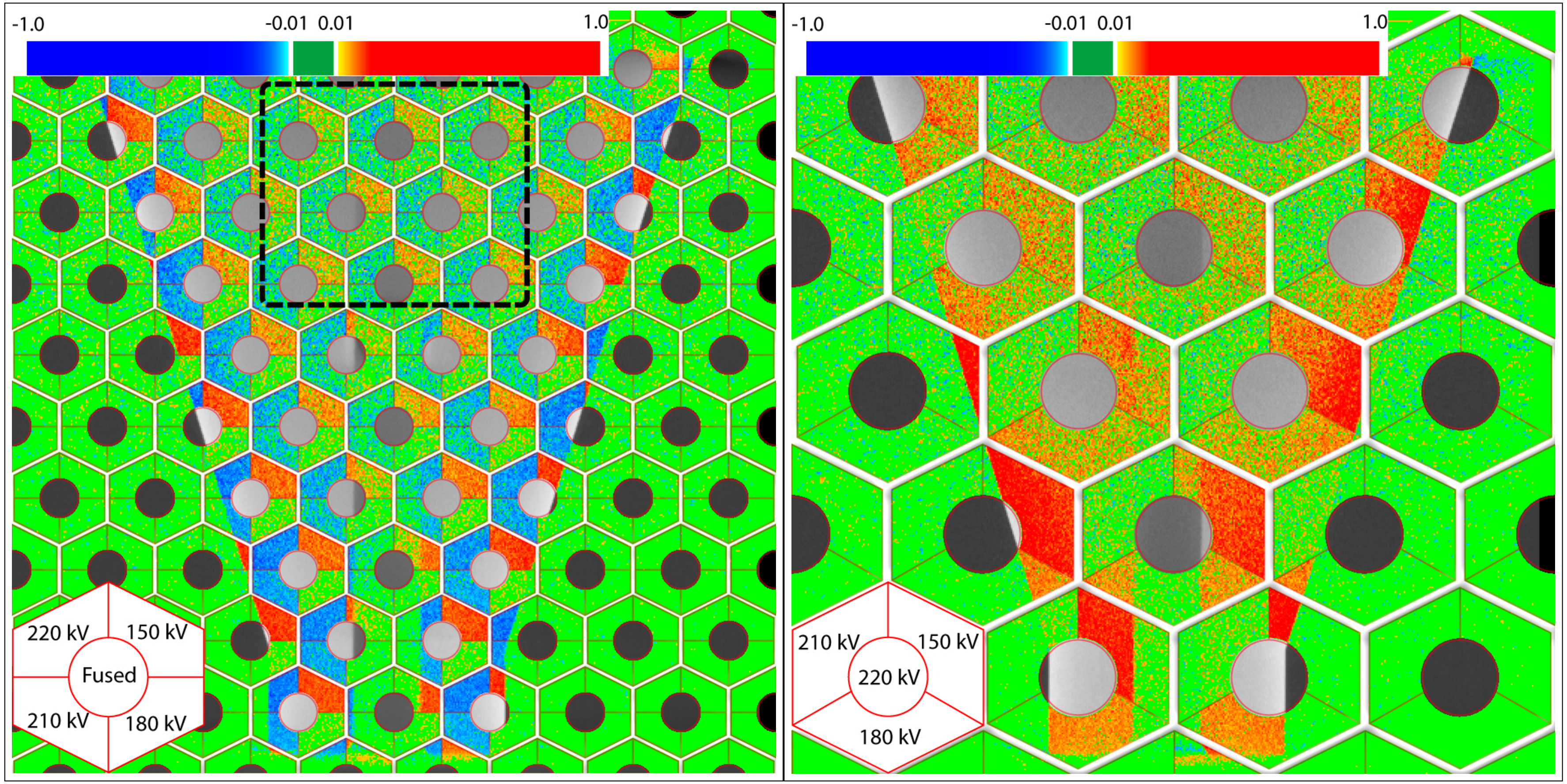


Figure 5.3: *Multi-image view* of the voltage dataset series rendered using gray values. An image of the specimen and the slice position is shown in the top left of the image.

Observe the difference between the 150 kV and the 210 kV datasets indicated by two pairs of blue arrows. The difference between the datasets however is not detectable in the center of the specimen (highlighted by a pair of red arrows).

In some cases one may not be sure whether the absence of differences between the datasets is real or due to our inability to visually differentiate between the shades of gray. Also, at the moment we are visually comparing gray values of spatially different locations and therefore only the border regions between sectors are interesting. In this case one can switch to render deviations.

We compute deviations in the sectors of the *base tile* by evaluating the local difference between the gray values of the dataset in the sector and the gray values of the *central dataset*. The sectors are then color coded differentially with respect to the *central dataset*. This makes the visualization more expressive and the solution is robust (figure 5.4(a)). Neighboring sectors in the tiling show deviations from



(a) Multi-image view

(b) Multi-image view

Figure 5.4: *Multi-image view* of the voltage dataset series. Our system uses normalized gray values. The color scale encoding the deviations is in the range of -1 to 1. (a) *multi-image view* with the fused dataset as the *central dataset*. The dashed rectangle highlights an area affected by beam hardening. (b) *multi-image view* with the 220 kV dataset as the *central dataset*. The fused dataset has been removed from the analysis process. The number of sectors in each *base tile* is reduced from four to three.

adjacent but different parts of the underlying dataset slices. This can be ameliorated by interactively shifting the *base tile*.

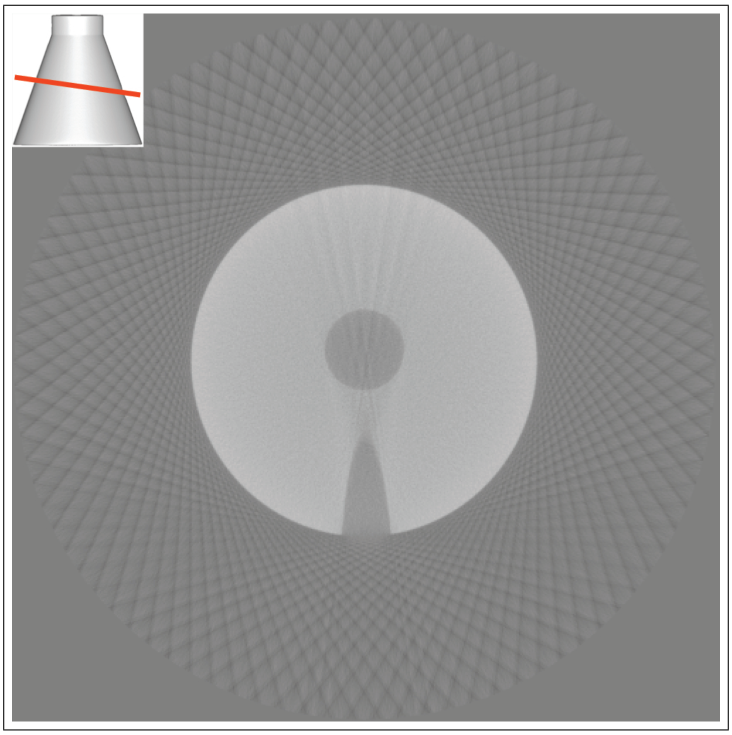
We observe in figure 5.4(a) that the differences between the datasets are well pronounced in *base tiles* which are in the lower half of the image and at the outer edges of the dataset (bright red and blue regions). The differences between the datasets are not clearly recognizable in areas of high penetration length (highlighted by a black dotted rectangle). This is due to beam hardening artifacts. All the voltages under study are affected by beam hardening. The dataset measured with 150 kV acceleration voltage of the X-ray source has higher contrast between air and material and thus produces stronger outer edges. This is obvious as the 150 kV dataset has higher gray values (red sectors) over the entire slice compared to all other datasets (green and blue color coding). Hence, a low voltage is better suited to capture edges at low penetration length. The fused dataset looks very similar to the dataset recorded with 180 kV as the corresponding sectors are mostly green. Our system reported the mean and the maximum difference between the fused and the 180 kV dataset as 238 and 1948 units. The gray values range from 0 to 65535 units in the voltage series.

The datasets measured with 210 kV and 220 kV seem to produce almost the same result. We can compare these datasets by making one of these datasets the *central dataset*. We drag the 220 kV dataset into the *central dataset* position. We also remove the fused dataset from the analysis. The number of sectors in the *base tile* automatically reduces from four to three (figure 5.4(b)). Although the difference between these two datasets is small, still the 210 kV dataset produces slightly higher gray values (sectors with reddish yellow and green color coding).

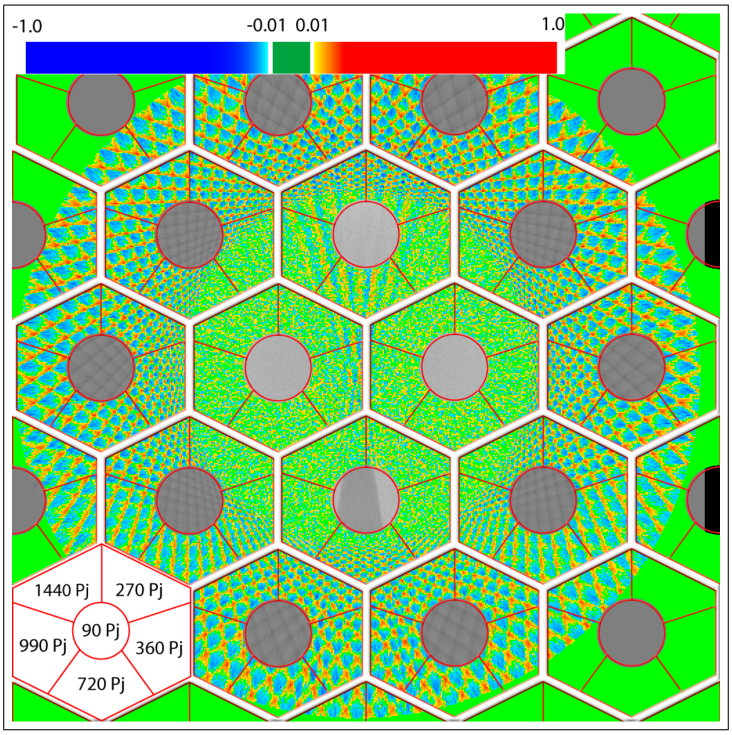
The grid of *base tiles* can be interactively switched to a single *base tile*. This somewhat corresponds to a single multi-faceted magic lens. The user can control the size and location of the single *base tile* and can probe specific regions of the dataset series. This is particularly helpful for comparative visualization of areas severely affected by artifacts. We can thus determine the datasets which contain the least artifacts. Quantitative data, e.g., mean and maximum deviations are only displayed for the area covered by the single *base tile*.

A comparative visualization can be difficult to interpret when the *central dataset* has a high variation in gray values or has a lot of salt and pepper noise. This is apparent when analyzing the dataset series with the varying number of projections. The number of projections dataset series has six datasets with 90, 270, 360, 720, 990, and 1440 projections for a single 360 degree turn of the rotary plate. If the number of projections is too low, the volumetric dataset computed from these projections has streaking artifacts due to undersampling (figure 5.5(a)). Figure 5.5(b) shows a *multi-image view* where the dataset with 90 projections is taken as the *central dataset*. We observe streaking patterns emerging in all the sectors of the *base tiles*. The patterns contain blue and red colors showing high variation in gray values of all the datasets. These patterns may have emerged due to two reasons. First, all the datasets are affected by streaking artifacts. Second, only the *central dataset* is affected by streaking artifacts. It is important to determine if the patterns are appearing due to a bad selection of the *central dataset*.

We use a simple method to quickly determine the homogeneity of the slices and visualize it using a *multi-image view*. For homogeneity rendering we color a pixel based on the following conditions: If the gray value of a pixel is within

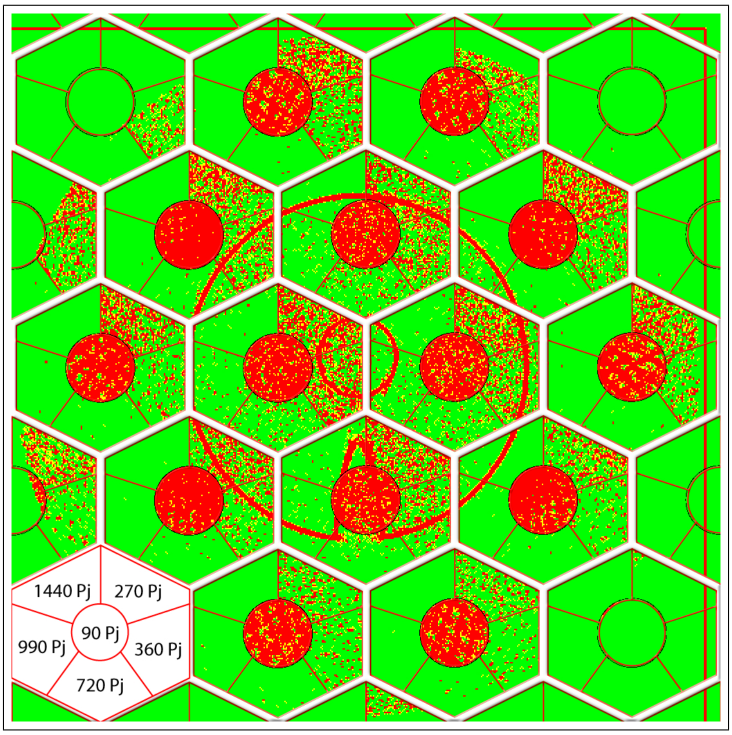


(a) Planar reformatting

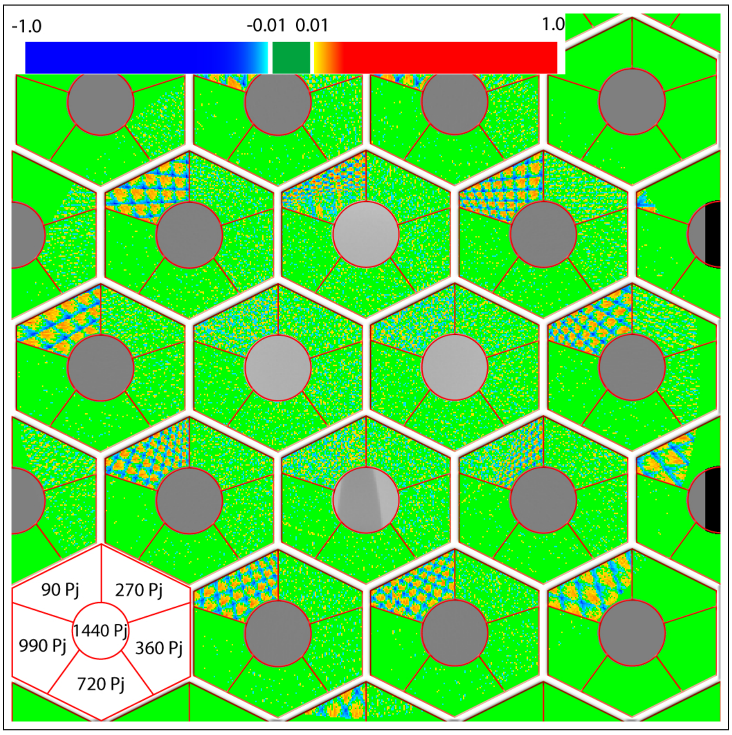


(b) Multi-image view

Figure 5.5: (a) slice from the dataset with 90 projections (Pj). The slice position is shown in the top left of the image. (b) *multi-image view*. The dataset with 90 Pj is the *central dataset*.



(a) Homogeneity rendering



(b) Multi-image view

Figure 5.6: (a) Homogeneity rendering depicts the local variation of gray values within a slice. (b) *multi-image view* for the number of projections series. The dataset measured with 1440 Pj is the *central dataset*.

a certain range of the gray values in its 8-neighborhood, the pixel is rendered green. If the difference between the gray values of a pixel and its neighbors in the 4-neighborhood is within a predefined limit, it is colored yellow. Otherwise the pixel is colored red. This provides a quick overview of the amount of gray value variation in a slice. Homogeneity rendering (figure 5.6(a)) shows the *central dataset* (90 projections) to be the most in-homogeneous one. It is therefore not a good choice to be used as a *central dataset*. Figure 5.6(b) is rendered by taking the 1440 projections dataset as the *central dataset*. We conclude that the streaking artifacts are greatly reduced when the number of projections is increased from 90 to 270. Streaking artifacts are even more reduced when the number of projections is increased to 720. A further increase in the number of projections does not have a considerable impact on the dataset quality.

5.2.2 Edge Explorer

Edge detection is an important topic in 3DCT as it is useful for segmentation, feature identification, feature selection, etc. Edges are blurred in a scanned volumetric dataset due to various artifacts like beam hardening, geometric blur due to the focal spot of the X-ray source or scattered radiation. Our company partners consider edge quality as a strong indication of scan quality. The *edge explorer* is designed to provide an overview of the edges in an entire set of slices or images. The edge information for a specific slice of the entire dataset series is thereby condensed into a vertical and a horizontal histogram. The histograms basically count outlier edges which are present in some dataset but not in some other. This guides the user to problematic areas on a slice. Furthermore the *edge explorer* includes a zoom-in functionality for a detailed inspection of a specific edge throughout the entire dataset series.

The *edge explorer* uses either the Sobel or the Canny edge detection algorithm [11]. The choice is based on the user preferences. The Sobel edge detector is faster than the Canny edge detector. It only requires the manipulation of a single threshold for edge detection whereas the Canny edge detector has two thresholds. The Canny edge detector provides better results in all our experiments. Two images are produced as a result of an edge detection algorithm. First, a binary image is generated which specifies the edges detected in an image. Second, the edge strength computed by the algorithm is returned as an RGB image.

Edge detection is performed on a slice of the *central dataset* and the result is displayed on the screen (figure 5.7). Edge detection is also performed on the other datasets in the series and the results are stored for further processing. As we have edge information from all the datasets we can easily test if an edge is present or not in all the datasets at a particular position in an image. Also we can compare the strength of the edges in different datasets.

We show two graphs with violet background in figure 5.7. The vertical graph has yellow bars and is displayed on the right of the image. The horizontal graph is plotted above the image and has red bars. We scan the rows of the slices between the two vertical yellow lines on the image and count the number of locations where an edge is present in one dataset but not in some other dataset. The number of such locations found in a row is shown on the vertical graph as a bar. Similarly

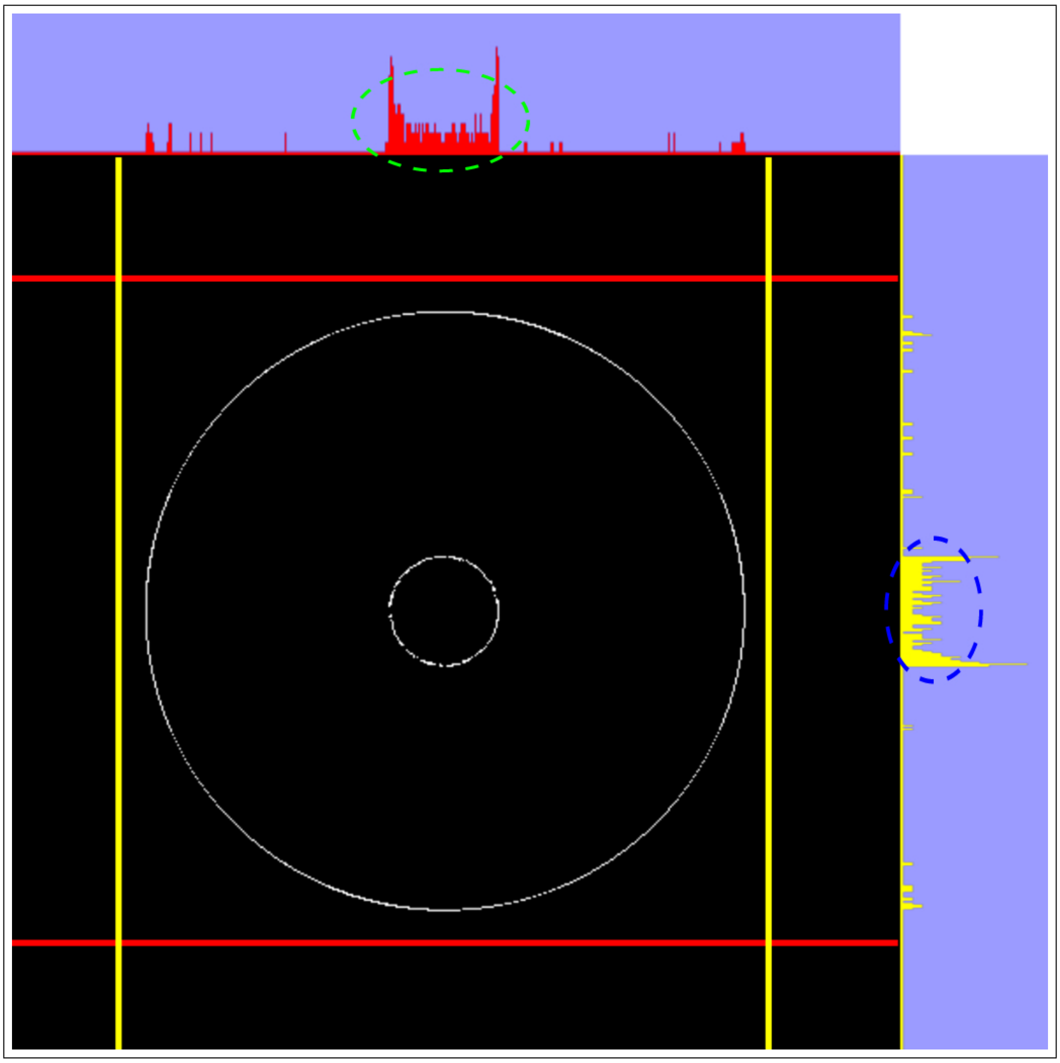
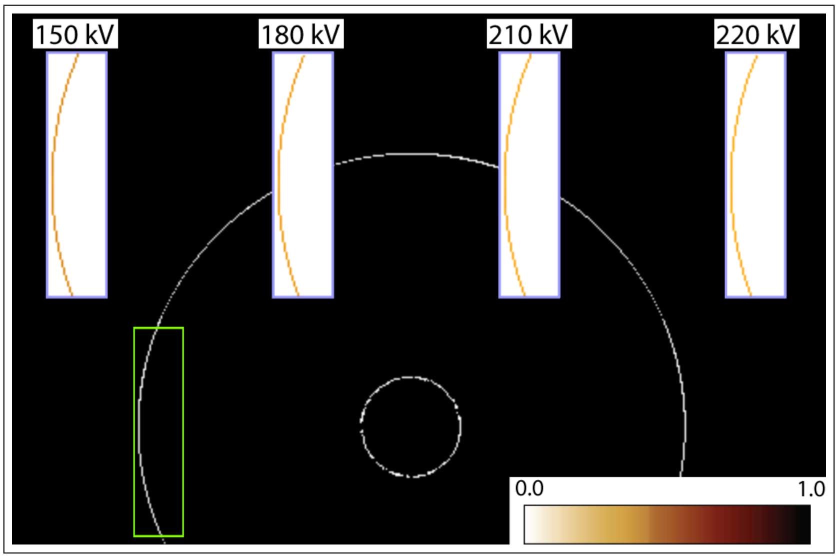


Figure 5.7: *Edge explorer* for the voltage series.

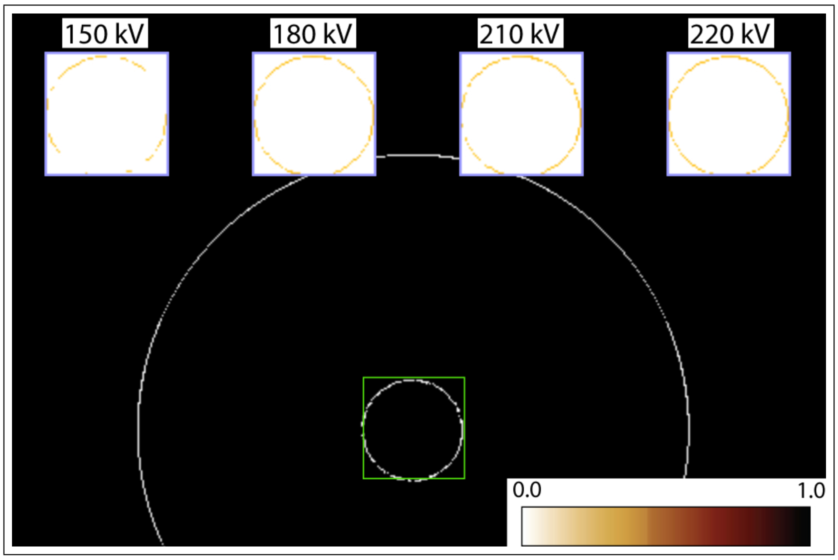
we scan the columns between the horizontal red lines and plot the result on the horizontal graph. Both graphs can be scaled separately.

The vertical and horizontal lines can be moved by the user with live update of the histograms. User interaction and the combined information from the two histograms help to find areas where edges are missing in some of the datasets. In figure 5.7, long bars on the horizontal graph (highlighted by a green ellipse) show an area where edges are missing in some of the datasets. The information from the vertical graph (dark blue oval) can be used to further localize the problematic area on the slice.

We also have a zoom-in tool in our *edge explorer*. The user selects a location on the image for zoom-in. The normal vector of an edge at that location (if an edge is present) or the gradient of the gray values automatically determines if the edge information is shown in a horizontally or vertically elongated rectangle. If the gradient direction is close to the horizontal axis, the zoom-in is given as a vertical rectangle. If the gradient direction is close to the vertical axis, the zoom-in is given



(a) Zoom-in



(b) Zoom-in

Figure 5.8: (a) zoom-in of all the datasets in a user selected area. Zoom-ins are displayed side by side for comparison. (b) zoom-in of central drill hole.

as a horizontal rectangle. Otherwise a square is chosen for the zoom-in. Further shapes can easily be added to the zoom-in tool based on the gradient direction.

Figure 5.8(a) shows a zoom-in using a vertical rectangle. The zoom-in shows each dataset of a series in a separate window. The edge strength in the zoom-in region (outer edge) is color coded according to the black body radiation spectrum. As we found out earlier by analyzing figure 5.4(a) the outer edge in the 150 kV dataset is the best pronounced. The 150 kV dataset captures the outer edges best but it is affected most severely by beam hardening. The zoom-in of the edges at a high penetration length (central drill hole) shows the datasets with 210 kV and 220 kV voltage to produce the strongest edges (figure 5.8(b)).

We conclude that the choice of the voltage is dependent on the penetration length of the specimen. In case of a dual energy CT machine, the low voltage setup must be used to analyze the outer edges of the specimen whereas the high voltage setup will be better to explore the inner parts of the component. We compare the robustness of investigating edges using *multi-image views* and the *edge explorer* in the next section.

5.3 User Feedback and Results

We have proposed two visualization techniques namely *multi-image view* and *edge explorer* for the comparative visualization of multiple datasets. Salient features of the *multi-image view* are:

- Concurrent viewing of multiple datasets.
- Efficient comparative visualization.
- Scalable with the number of datasets.
- Interaction tools for enhanced usability.
- Single *base tile* for user guided analysis.

Salient features of the *edge explorer* are:

- Designed specifically for comparing edges.
- Exploration of missing edges and comparison of edge strength.
- Histograms for fast localization of problematic areas.
- Interaction tools for enhanced usability.

We collected feedback on our system in collaboration with company partners to evaluate the robustness and practical value of the proposed visualization techniques. We conducted two evaluations of the system. In the first evaluation the methods are compared to the current workflow of the domain specialists. In the second evaluation we ask the same domain specialists to compare *multi-image view* with attribute blocks.

The users who participated in the evaluation were domain specialists working professionally with 3DCT. Table 5.2 shows how often they work with 3DCT, their professional experience in handling 3DCT data, the maximum number of slices they have compared, and the type of comparison they have performed. The type of comparison can be a quantitative comparison, a qualitative comparison, or a combination of both.

Table 5.2: Profiles of the domain specialists who participated in the evaluation. User 2 has never compared slices. User 3 and user 4 have experience of comparing more than three slices.

	Works with 3DCT	Professional experience	Maximum slices compared	Comparison type
User 1	everyday	4.5 years	2	combination
User 2	everyday	0.5 years	Nil	-
User 3	everyday	3.0 years	≥ 4	quantitative
User 4	everyday	4.0 years	≥ 4	quantitative

5.3.1 First Evaluation - Workflow Comparison

In the first evaluation (questionnaire provided in Appendix A) we gathered the feedback in four parts. In the first part we obtained information from the users about their professional experience with 3DCT and the comparison of multiple datasets (table 5.2). In the second part, we explained the respective techniques and afterwards the users performed a sequence of tasks with our software prototype. They were provided with assistance in completing the tasks. This part was meant to familiarize them with the system.

In the third part, they were given five tasks to accomplish with our system. We measured the time they took to complete each task. The users were asked several forced-choice questions (FCQ) after each task. In the fourth part, we did a short interview (FCQs and anectodical feedback) about our techniques and the earlier workflow they used to compare multiple datasets.

The users do not have a standardized workflow to compare 3DCT datasets. Typically, they load all the datasets into VGStudio MAX [78]. Then the corresponding slices from the datasets are extracted. Afterwards, the slices are compared either visually, using VGStudio MAX, and/or quantitatively using Matlab. A workstation with a huge amount of RAM is required to perform the comparison of large dataset series. Our work provides a standardized workflow for comparative visualization.

The five tasks that the users performed in the evaluation were:

1. Use *multi-image view* to find the dataset that best preserves the edges.
2. Use *edge explorer* to find the dataset that best preserves the edges.
3. Mark areas on a slice (middle slice in axial direction) where the variation between the datasets is minimal and maximal.
4. Find the dataset(s) which has the fewest beam hardening artifacts.
5. Find the dataset(s) which has the most and the fewest streaking artifacts.

The voltage series was used for the first four tasks. The last task was performed on the number of projections series. The reason for taking a different dataset series in the last task was that streaking artifacts vary considerably with the number of projections. The information regarding which parameter was varied in the dataset series was hidden from the users to ensure that they could not employ their insider knowledge while completing the tasks. User 3 and user 4 were given task 2 before task 1.

Both the techniques presented in this chapter (*multi-image view* and *edge explorer*) allow a successful comparison of the edges in the volumetric datasets (table 5.3). *Edge explorer* is specifically designed for comparing edges and therefore it gives a better performance as compared to a *multi-image view*. It took the users 3 minutes on average to find the dataset that preserves the edges best with the *edge explorer* whereas the average time using the *multi-image view* was 5.6 minutes.

Task 3 shows the ability of the *multi-image view* to convey spatial information about the deviations present in the dataset series. The users were also able to detect and compare streaking artifacts in a dataset series with ease. Finding the dataset with the smallest amount of beam hardening took the longest amongst all the tasks.

Compared to their earlier workflow, the users indicated the biggest advantages of our technique to be the ability to see and compare multiple slices in a single image and the good performance of the implemented system. Loading large monolithic volumes is very time consuming. The users felt that each of the above tasks would have taken them approximately 30 minutes using their earlier workflow. User 3 and user 4 suggested that gray value histograms should also be incorporated in the system. They are used to them from their previous workflow.

Table 5.4 shows the subjectively expressed impact that our system has on the quality and speed of comparative visualization for the domain specialists. Choices that the user could select from were: greatly improves, somewhat improves, no impact, somewhat deteriorates, greatly deteriorates, and do not know. User 2 had no prior experience of comparing multiple datasets and is therefore not included.

5.3.2 Second Evaluation - Multi-image View vs. Attribute Blocks

There is a large body of work on the visualization of multiple parameters. Fields like weather simulation and anthropology for example require the simultaneous display of multiple parameters. Spatial fusion techniques like color blending [21] and space-sharing methods like color weaving [75] and attribute blocks [51] are common for such applications. Attribute blocks have a structured approach compared to color weaving. All the parameters are displayed side by side in every attribute block on the plane. In the following a comparison between attribute blocks and the *multi-image view* is discussed. We slightly modify the attribute blocks to show color coded deviations with respect to one reference dataset.

We propose a set of interaction tools for comparative visualization. In order to make a fair comparison between the two methods we provided the domain specialists with a set of images to complete four tasks. By providing just images we separate the effect of user interactions from the visual impact of the *base tiles*. The purpose of this evaluation is to compare the structure of the hexagonal *base tile* of the *multi-image view* with the rectangular *base tile* of the attribute blocks.

Table 5.3: Results achieved and the time (minutes) needed by each user for every task performed. The results (correct, incorrect, or do not know) show if the user was able to answer the question given in the task. The user could choose “do not know” if he could not figure out how to perform the task. The results show the applicability of our techniques. User 2 did not know about beam hardening and therefore did not participate in task 4.

	Task 1 result achieved	Task 1 time	Task 2 result achieved	Task 2 time	Task 3 result achieved	Task 3 time	Task 4 result achieved	Task 4 time	Task 5 result achieved	Task 5 time
User 1	do not know	0.5	correct	4	correct	2	correct	7	correct	3
User 2	correct	6	correct	3	correct	1	-	-	correct	5
User 3	correct	5	correct	3	correct	1	do not know	4	correct	4
User 4	correct	6	correct	2	correct	0.5	correct	9	correct	4
Mean time		5.6		3.0		1.1		8.0		4.0

Table 5.4: Overall impression of our technique on the users. The users rated the impact of our system on the quality and speed of comparing multiple datasets.

Impact on	User 1	User 3	User 4
quality	greatly improves	somewhat improves	somewhat improves
speed	greatly improves	greatly improves	greatly improves

We ask the professionals to complete the following tasks:

1. Find the dataset that best preserves the edges.
2. Mark areas on a slice of the voltage series where the variation between the datasets is minimal and maximal.
3. Mark areas on a slice of the projections series where the variation between the datasets is minimal and maximal.
4. Find the dataset(s) which has the most and the fewest streaking artifacts.

The feedback was collected from the users through FCQs and open ended questions (questionnaire provided in Appendix B). Each task was performed with both the techniques. User 1 and user 2 tested *multi-image view* first to complete the tasks whereas user 3 and user 4 tested attribute blocks first. The time taken to complete the tasks using *multi-image view* is considerably less than the time spent using attribute blocks (table 5.5).

All the users appreciated the idea of having a reference dataset in the center. They felt that it was very helpful for the comparison. User 1 and user 3 were of the view that the *base tiles* of the *multi-image view* are a lot better than the *base tiles* of the attribute blocks. The other two domain specialists felt that the structure of the *base tiles* of the *multi-image view* is only a little bit better than that of the attribute blocks. All agreed that given the interaction tools and the efficient memory management of our tool, it is much more practical for their work compared to attribute blocks.

The domain specialists mentioned that a major advantage of the *multi-image view* is that it displays each dataset in a sector with a unique shape. It was easy for the users to track and locate the individual datasets in the image. In the attribute blocks all the parameters were displayed in squares and hence the users had to refer to the legend more often. The main advantage of the attribute blocks was that it was relatively easy to learn and use. User 1 also mentioned that the attribute blocks could become confusing with an increasing number of datasets.

Figure 5.9 compares our method with an implementation of attribute blocks. Hagh-Shenas et al. found that the robustness of color weaving deteriorates when the number of datasets is more that four [23]. Therefore we have excluded color weaving from the evaluation and just compare *multi-image view* with attribute blocks. Figure 5.9(a) shows a slice from the voltage series. We perform a comparative visualization of the voltage series using a *multi-image view* in figure 5.9(b). The 210 kV dataset is the *central dataset*. The 150 kV and 180 kV datasets produce red and reddish yellow colors respectively. They contain gray values higher than the 210 kV dataset. We easily notice that the differences between the datasets are well

Table 5.5: Results achieved and the time (minutes) needed by each user for every task performed. The results (correct, incorrect, or do not know) show if the user was able to answer the question given in the task. The user could choose "do not know" if he could not figure out how to perform the task.

	Task 1 result achieved	Task 1 time	Task 2 result achieved	Task 2 time	Task 3 result achieved	Task 3 time	Task 4 result achieved	Task 4 time
User 1 - Multi-image view	correct	1	correct	1	correct	1	correct	2
User 1 - Attribute blocks	correct	1	correct	2	correct	1	correct	2
User 2 - Multi-image view	correct	1	correct	1	correct	1	correct	2
User 2 - Attribute blocks	correct	2	correct	4	correct	2	correct	2
User 3 - Multi-image view	correct	1	correct	1	correct	1	correct	3
User 3 - Attribute blocks	correct	2	correct	2	correct	2	correct	3
User 4 - Multi-image view	correct	1	correct	1	correct	1	correct	2
User 4 - Attribute blocks	do not know	3	correct	3	correct	1	correct	3
Mean - Multi-image view		1		1		1		2.25
Mean - Attribute blocks		2		2.75		1.5		2.5

pronounced near the edges, compared to the inner part of the specimen. The 150 kV dataset has the highest contrast between material and air. All the datasets are neighboring the *central dataset*. The *base tile* structure is easily scalable as we add or remove datasets from the active comparison process.

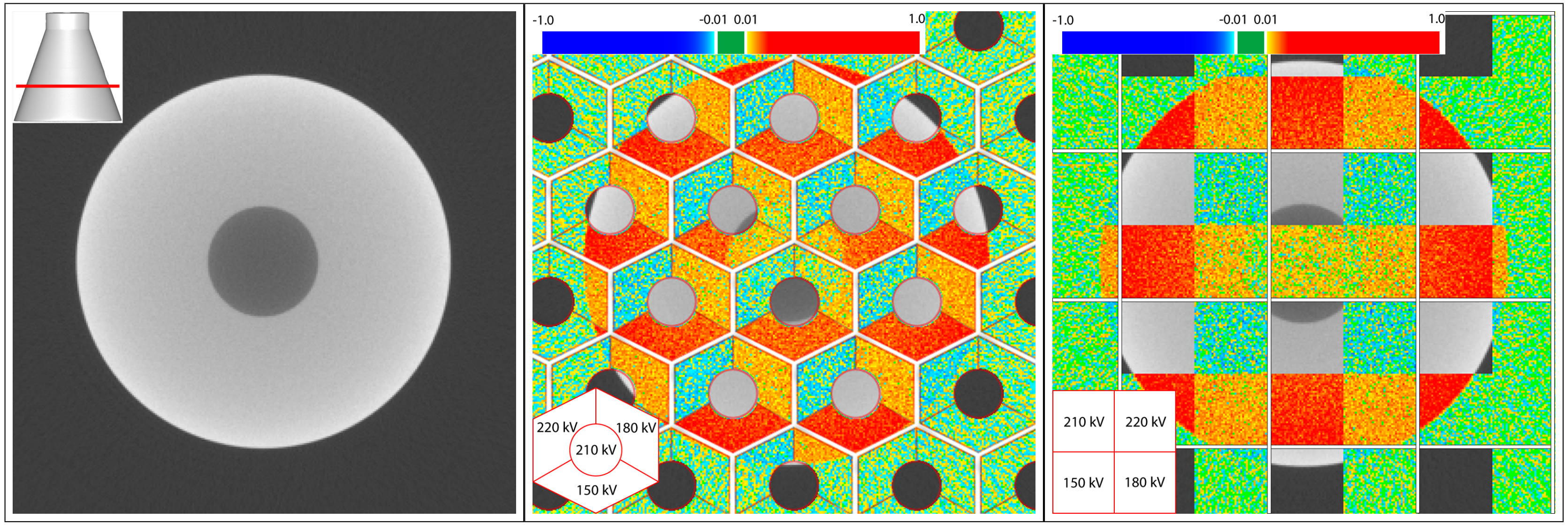
Figure 5.9(c) shows the voltage series rendered with attribute blocks. The 210 kV dataset is displayed in gray. The other cells render the differences between the 210 kV dataset and the respective dataset of the cell. Space-sharing techniques like the attribute blocks are specifically designed for the simultaneous display of multiple parameters. Attribute blocks are scalable for displaying more datasets but they seem less suitable for comparison of multiple parameters. Attribute blocks have a grid like structure (figure 5.2(b)) and hence they are not specifically designed for comparisons across sector boundaries. For *multi-image view* we propose a set of tools and interaction techniques to support the comparison process. Our method provides a direct comparison of the datasets. The differences between the datasets are apparent by just a fast visual inspection. Furthermore our system is purpose built to efficiently process large and high resolution dataset series (section 5.4).

5.4 Implementation and Performance

We implemented a prototype on a Pentium 4 3.4 GHz CPU and an NVidia GeForce 8800 GTS graphics board. The datasets to visualize are quite large (table 5.1). In a pre-processing step each volume of the dataset series is subdivided into equal-sized rectangular sub-volumes (bricks). Each element of a brick is a list containing a gray value from each dataset of the series. Brick fetching is fastest when the brick size is about 64 KBytes [22]. The pre-processing step dynamically chooses brick dimensions so that the brick size is close to 64 KBytes. We only fetch those bricks into memory that are required for rendering.

To speed up the overall performance of the application we also implemented pre-fetching of the bricks. We have three threads in our application. The first thread runs in the background and loads those bricks into main memory that might be needed in the near future. Based on the orientation of the slice and the direction of navigation, we can calculate the next set of bricks that may be required. The second thread which has the highest priority is responsible for fetching those bricks that are needed for rendering the current *multi-image view*. The second thread will only be executed when the currently required set of bricks has not already been fetched into memory. The third thread performs rendering of images and handles user interaction with the system.

Based on the available storage in the system we keep three sets of bricks in memory. These sets of bricks are in the order of importance: first, the set required for rendering the current *multi-image view*, second, the next set of bricks required in the future, and third, the set of bricks most recently used. The performance of our system is given in table 5.6. The timings shown include fetching the data from the main storage and rendering. We observe approximately a 26% performance drop for the axial case because of bricking. We gained about 61% and 21% rendering speed-up by bricking in the sagittal and the coronal cases respectively. The loss in performance in the axial case is due to the optimal alignment of the monolithic volume with the slicing direction.



(a) Planar reformatting

(b) Multi-image view

(c) Attribute blocks

Figure 5.9: Comparison of *multi-image view* and attribute blocks. (a) slice through a dataset of the voltage series. The slice position is shown in the top left. (b) displaying the dataset series through *multi-image view*. (c) displaying the dataset series through attribute blocks.

Table 5.6: Performance: Average time (in milliseconds) required for rendering a gray value slice using a monolithic volume (i.e., non-bricked), a bricked volume, rendering a *multi-image view*, and rendering an *edge explorer*.

slice	monolithic	bricked	multi-image view (bricked)	edge explorer (bricked)
axial	19	24	26	29
sagittal	90	35	38	44
coronal	35	28	31	36

5.5 Conclusion

We have presented two visualization techniques for the comparative visualization of multiple datasets. A *multi-image view* helps the user to track artifacts and also provides interaction tools to find datasets that have a better contrast or stronger edges. The *edge explorer* is specifically designed for simultaneously visualizing edges and edge strengths in a dataset series.

We use bricking for efficient memory management and threading for prefetching of the bricks. We take advantage of the graphics hardware to achieve a considerable rendering speed-up. Our system is interactive and works by loading only those parts of the datasets that are required for rendering. We are therefore able to perform comparative visualization on large dataset series. The system has been evaluated by domain specialists for 3DCT. They found our techniques quite helpful for their work. Our techniques are generic and seem applicable for dataset series from other imaging devices as well.

Learning without thought is labor lost; thought without learning is perilous.

Confucius

6
Summary and Conclusion

The idea of assisting users to efficiently and to easily deduce valuable results from data is an inherent part of visualization. The user requirements and the purpose of performing the visualization have to be considered during the design and development of a visualization algorithm. The users are often not well acquainted with all the parameters that are used in an algorithm. The usability of a technique usually declines with an increase in the number of parameters that the user has to interact with. The parameters can be reduced to a feasible number if the visualization techniques are application specific and if they are designed according to the user requirements.

The user is often concerned about a specific feature in a volumetric dataset. We design techniques that are tuned to visualize these features of interest. The rest of the data is considered to be context information. Feature centric visualization techniques are robust and focused. They require less user interaction, the visualization is more expressive and objective, and the software design is customized for the target users.

Feature Peeling:

Peeling layers of data is a successful attempt to efficiently search for features inside a volumetric dataset. The user is no more required to specify a transfer function or to perform laborious segmentation. Instead, two linear parameters can be set to generate feature layers. The user can browse through the layers in real-time. Our algorithm is particularly useful in time critical medical scenarios, in using contemporary volume visualization techniques for MRI data, and in exploration of volumetric datasets.

It will be interesting to monitor the variability of the two linear parameters with respect to different types of datasets. We believe that these parameters will not require intense interaction when used for a single type of dataset, like MRI.

Computation and Visualization of Fabrication Artifacts:

3DCT datasets are prone to various artifacts. The fabrication artifacts are interesting for the engineers while the errors introduced by the measurement machine and by the post processing software are detrimental. They negatively affect the accuracy with which fabrication artifacts can be concluded. We try to separate the fabrication artifacts from the reconstruction artifacts which are introduced by the surface extraction algorithm. A single system to execute the entire pipeline for the computation and visualization of fabrication artifacts is proposed.

The research system suggested in this thesis is already in use by our company partners. They particularly appreciate the idea of rendering box plots in a zoom-in of a user specified area of interest. In the future the measurement errors might be taken into account while calculating the fabrication artifacts. The measurement errors can be incorporated into the visualization pipeline as uncertainties. This however requires a comprehensive study on the measurement errors.

Locally Adaptive Marching Cubes:

Measurement errors affect the shape of the objects inside a volumetric dataset. The shape is modified due to bias in the scalar values. This is corrected by introducing an iso-value field in the marching cubes algorithm. The algorithm can also be used to stitch together iso-surfaces or to generate a surface model that has a smooth transition from one feature of interest to another. We have tested this method on various medical and industrial specimens and the results are promising.

Comparative Visualization for Parameter Studies of Dataset Series:

The thesis describes a planar reformatting based technique for comparative visualization of volumetric datasets. The method is also applicable for comparison of images. The proposed system has been developed in close collaboration with our company partners. We conducted two evaluations to study the utility and robustness of our algorithms.

Comparison of multiple datasets is an important task for the 3DCT domain experts. They perform it regularly in their professional work. The continuous increase in the precision, the resolution, and the size of the 3DCT datasets makes comparative visualization an emerging topic. There is demand for efficient and high quality comparisons but the limited main memory in CPUs and GPUs is a bottle neck. Comparative visualization will be an interesting field for further research in the future.

The work presented in this thesis provided immense opportunities for learning and critical thinking. Four software systems have been developed as a proof of concept for the research results that we present. Latest graphics hardware and shading languages have been used to achieve real-time performance. Most of the work has been completed in collaboration with our company partners. This gave us a chance to evaluate our work with the help of domain experts, to use latest microCT and NanoCT devices, and to set research goals that make a difference in the work of professionals.

> The least flexible component of any system is the user.
>
> Lowell Jay Arthur

A

Questionnaire for First Evaluation

A.1 General Questions

Question 1.1: How often do you work with Industrial CT?

⊙ At least once a year
⊙ At least once a week
⊙ Never
⊙ At least once a month
⊙ At least once a day

Question 1.2: How much experience do you have with Industrial CT?

_______________ months

Question 1.3: Have you ever compared multiple slices?

⊙ Yes
⊙ No (Please go to **Section A.2: Tasks**)

Question 1.4: How do you compare multiple slices?

⊙ I compute quantitative measures like "difference of images" using MATLAB etc
⊙ I use a specialized software to visualize the differences between the slices
Please name the software: _______________

Question 1.5: Does your comparison provide spatial information?

⊙ Yes
⊙ No
⊙ Do not know

A.2 Tasks

Please use the provided software to answer the following questions:

Task 2.1: Which dataset preserves the edges best? (use multi-image view)
- ⊙ Dataset 1
- ⊙ Dataset 2
- ⊙ Dataset 3
- ⊙ Dataset 4
- ⊙ Do not know

Task 2.2: Which dataset preserves the edges best? (use edge explorer)
- ⊙ Dataset 1
- ⊙ Dataset 2
- ⊙ Dataset 3
- ⊙ Dataset 4
- ⊙ Do not know

Task 2.3: Mark areas in a given slice where all the datasets
- ⊙ have almost similar grey values
- ⊙ have most variation in grey values

Task 2.4: Which dataset in the areas of high penetration depth:
- ⊙ has minimal beam hardening artifacts?
  - ⊙ Dataset 1
  - ⊙ Dataset 2
  - ⊙ Dataset 3
  - ⊙ Dataset 4
  - ⊙ Do not know
- ⊙ has maximal beam hardening artifacts?
  - ⊙ Dataset 1
  - ⊙ Dataset 2
  - ⊙ Dataset 3
  - ⊙ Dataset 4
  - ⊙ Do not know

Task 2.5: Which dataset: (You can select multiple choices)
- ⊙ has least streaking artifacts?
  - ⊙ Dataset 1
  - ⊙ Dataset 2
  - ⊙ Dataset 3
  - ⊙ Dataset 4
  - ⊙ Dataset 5
  - ⊙ Dataset 6
  - ⊙ Do not know
- ⊙ has most streaking artifacts?
  - ⊙ Dataset 1
  - ⊙ Dataset 2
  - ⊙ Dataset 3
  - ⊙ Dataset 4
  - ⊙ Dataset 5
  - ⊙ Dataset 6
  - ⊙ Do not know

A.3 General Questions about the Technique

Question 3.1: Will you like any change in the hexagonal structure?

⊙ No
⊙ Yes. Please mention ____
____

Question 3.2: Will you like any additional interaction techniques to be incorporated in the system?

⊙ No
⊙ Yes. Please mention ____
____

Proceed further only if you answered yes to **Question 1.3** in **Section A.1**:

Question 3.3: Approximately how much time will it take you to complete Task 2.1 using your earlier work flow?

⊙ Time: _______________ minutes
⊙ Do not know

Question 3.4: Approximately how much time will it take you to complete Task 2.3 using your earlier work flow?

⊙ Time: _______________ minutes
⊙ Do not know

Question 3.5: Approximately how much time will it take you to complete Task 2.4 using your earlier work flow?

⊙ Time: _______________ minutes
⊙ Do not know

Question 3.6: Approximately how much time will it take you to complete Task 2.5 using your earlier work flow?

⊙ Time: _______________ minutes
⊙ Do not know

Question 3.7: Will *multi-image view* and *edge explorer* improve the quality of the work flow of comparing multiple datasets?

⊙ Greatly improve
⊙ Somewhat improve
⊙ Somewhat deteriorate
⊙ Greatly deteriorate
⊙ Unchanged
⊙ Do not know

Question 3.8: Will *multi-image view* and *edge explorer* increase the speed of the work flow of comparing multiple datasets?

⊙ Greatly increase
⊙ Somewhat increase
⊙ Somewhat reduce
⊙ Greatly reduce
⊙ Unchanged
⊙ Do not know

Question 3.9: What is the biggest advantage that *multi-image view* and *edge explorer* has over your earlier work flow?
⊙ There is no advantage
⊙ The biggest advantage is: ____________________________________
____

Question 3.10: What is the biggest advantage of your earlier work flow over *multi-image view* and *edge explorer*?
⊙ There is no advantage
⊙ The biggest advantage is: ____________________________________
____

Open ended discussion follows ...

> One is alone, two are eleven.
>
> Pakistani Proverb

B

Questionnaire for Second Evaluation

B.1 General Questions

Question 1.1: How often do you work with Industrial CT?

⊙ At least once a year
⊙ At least once a month
⊙ At least once a week
⊙ At least once a day
⊙ Never

Question 1.2: How much experience do you have with Industrial CT?

_______________ months

Question 1.3: Have you ever compared multiple slices?

⊙ Yes
⊙ No (Please go to **Section B.2: Tasks**)

Question 1.4: How do you compare multiple slices?

⊙ I compute quantitative measures like “difference of images” using MATLAB etc
⊙ I use a specialized software to visualize the differences between the slices
Please name the software: _______________

Question 1.5: Does your comparison provide spatial information?

⊙ Yes
⊙ No
⊙ Do not know

B.2 Tasks

We intend to compare the structure of the base tile of the *multi-image view* (MV) with the base tile proposed in attribute blocks (AB).

Task 2.1.1: Which dataset preserves the edges best? (use MV)
⊙ Dataset 1 ⊙ Dataset 2
⊙ Dataset 3 ⊙ Dataset 4
⊙ Do not know

Task 2.1.2: Which dataset preserves the edges best? (use AB)
⊙ Dataset 1 ⊙ Dataset 2
⊙ Dataset 3 ⊙ Dataset 4
⊙ Do not know

Question 2.1 Which base tile looks better for comparing the edge strengths in a set of slices?
⊙ MV is a lot better
⊙ MV is a little better
⊙ AB is a little better
⊙ AB is a lot better
⊙ No difference
⊙ Do not know

Task 2.2.1: Mark areas in the two given slices where all the datasets
⊙ have almost similar grey values (use AB)
⊙ have most variation in grey values (use AB)

Task 2.2.2: Mark areas in the two given slices where all the datasets
⊙ have almost similar grey values (use MV)
⊙ have most variation in grey values (use MV)

Question 2.2 Which base tile looks better for recognizing uniform areas?
⊙ MV is a lot better
⊙ MV is a little better
⊙ AB is a little better
⊙ AB is a lot better
⊙ No difference
⊙ Do not know

Task 2.3.1: Which dataset: (You can select multiple choices)
⊙ has least streaking artifacts? (use MV)
  ⊙ Dataset 1 ⊙ Dataset 2 ⊙ Dataset 3
  ⊙ Dataset 4 ⊙ Dataset 5 ⊙ Dataset 6
  ⊙ Do not know
⊙ has most streaking artifacts? (use MV)
  ⊙ Dataset 1 ⊙ Dataset 2 ⊙ Dataset 3
  ⊙ Dataset 4 ⊙ Dataset 5 ⊙ Dataset 6
  ⊙ Do not know

Task 2.3.2: Which dataset: (You can select multiple choices)

- ⊙ has least streaking artifacts? (use AB)
  - ⊙ Dataset 1
  - ⊙ Dataset 2
  - ⊙ Dataset 3
  - ⊙ Dataset 4
  - ⊙ Dataset 5
  - ⊙ Dataset 6
  - ⊙ Do not know
- ⊙ has most streaking artifacts? (use AB)
  - ⊙ Dataset 1
  - ⊙ Dataset 2
  - ⊙ Dataset 3
  - ⊙ Dataset 4
  - ⊙ Dataset 5
  - ⊙ Dataset 6
  - ⊙ Do not know

Question 2.3 Which base tile is better for comparing streaking artifacts?

- ⊙ MV is a lot better
- ⊙ MV is a little better
- ⊙ AB is a little better
- ⊙ AB is a lot better
- ⊙ No difference
- ⊙ Do not know

B.3 General Questions about the Two Techniques

Question 3.1 Will MV improve the quality of the work flow of comparing multiple datasets?

⊙ Greatly improve
⊙ Somewhat improve
⊙ Somewhat deteriorate
⊙ Greatly deteriorate
⊙ Unchanged
⊙ do not know

Question 3.2 Will AB improve the quality of the work flow of comparing multiple datasets?

⊙ Greatly improve
⊙ Somewhat improve
⊙ Somewhat deteriorate
⊙ Greatly deteriorate
⊙ Unchanged
⊙ do not know

Question 3.3 Will AB increase the speed of the work flow of comparing multiple datasets?

⊙ Greatly increase
⊙ Somewhat increase
⊙ Somewhat reduce
⊙ Greatly reduce
⊙ Unchanged
⊙ do not know

Question 3.4 Will MV increase the speed of the work flow of comparing multiple datasets?

⊙ Greatly increase
⊙ Somewhat increase
⊙ Somewhat reduce
⊙ Greatly reduce
⊙ Unchanged
⊙ do not know

Question 3.5: Does it help to have the reference dataset in the neighborhood of all the rest of the datasets in the series?

⊙ Yes
⊙ No

Question 3.6 Which of the following best describes the comparison of the hexagonal base tile used in MV and the rectangular base tile proposed in the AB?

⊙ MV is a lot better
⊙ MV is a little better
⊙ AB is a little better
⊙ AB is a lot better
⊙ No difference
⊙ Do not know

Question 3.7: What is the biggest advantage that AB has over MV?

⊙ There is no advantage
⊙ The biggest advantage is: ________________________________
____

Question 3.8: What is the biggest advantage that MV has over AB?

⊙ There is no advantage
⊙ The biggest advantage is: ________________________________
____

Open ended discussion follows ...

Bibliography

[1] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. Mesh: Measuring errors between surfaces using the Hausdorff distance. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, volume I, pages 705–708, 2002. (Cited on page 31.)

[2] C. L. Bajaj, V. Pascucci, and D. R. Schikore. The contour spectrum. In *Proceedings of the IEEE Visualization*, pages 167–175, 1997. (Cited on page 12.)

[3] D. Bartz, D.W. Cunningham, J. Fischer, and C. Wallraven. The role of perception for computer graphics. In *Proceedings of the Eurographics State-of-the-Art-Reports*, pages 65–86, 2008. (Cited on page 62.)

[4] S. Bergner, T. Möller, M. S. Drew, and G. D. Finlayson. Interactive spectral volume rendering. In *Proceedings of the IEEE Visualization*, pages 101–108, 2002. (Cited on page 12.)

[5] P. J. Besl and N. D. McKay. A method for registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992. (Cited on page 32.)

[6] D. Borland and R. M. Taylor II. Rainbow color map (still) considered harmful. *IEEE Computer Graphics and Applications*, 27(2):14–17, 2007. (Cited on page 65.)

[7] D. Brian, D. Brian, and H. Carr. On histograms and isosurface statistics. *IEEE Transactions on Visualization and Computer Graphics (in IEEE Visualization '06)*, 12(5):1259–1266, 2006. (Cited on page 12.)

[8] S. Bruckner and M. E. Gröller. Exploded views for volume data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1077–1084, 2006. (Cited on page 4.)

[9] S. Bruckner and M. E. Gröller. Style transfer functions for illustrative volume rendering. *Computer Graphics Forum*, 26(3):715–724, 2007. (Cited on pages 4 and 5.)

[10] W. Cai and G. Sakas. Data intermixing and multi-volume rendering. *Computer Graphics Forum*, 18(3):359–368, 2001. (Cited on page 61.)

[11] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 8(6):679–698, 1986. (Cited on page 71.)

[12] S. Castro, A. König, H. Löffelmann, and M. E. Gröller. Transfer function specification for the visualization of medical data. Technical report, Institute

of Computer Graphics and Algorithms, Vienna University of Technology, March 1998. (Cited on page 12.)

[13] P. Cignoni, C. Montani, and R. Scopigno. A comparison of mesh simplification algorithms. *Computers and Graphics*, 22(1):37–54, 1998. (Cited on page 31.)

[14] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum*, 17(2):167–174, 1998. (Cited on page 31.)

[15] D. S. Ebert, R. M. Rohrer, C. D. Shaw, P. Panda, J. M. Kukla, and D. A. Roberts. Procedural shape generation for multi-dimensional data visualization. *Computers & Graphics*, 24(3):375–384, 2000. (Cited on page 62.)

[16] J. C. Elliott and S. D. Dover. X-ray microtomography. *Journal of Microscopy*, 126:211–213, 1982. (Cited on page 59.)

[17] K. Engel, M. Hadwiger, J. M. Kniss, C. Rezk-Salama, and D. Weiskopf. *Real-time Volume Graphics*. A. K. Peters, 2006. (Cited on page 12.)

[18] J. Gao and H. Shen. Hardware-assisted view-dependent isosurface extraction using spherical partition. In *Proceedings of the Joint Eurographics-IEEE TCVG Symposium on Visualization*, pages 267–276, 2003. (Cited on page 12.)

[19] V. D. Gesú and V. Starovoitov. Distance-based functions for image comparison. *Pattern Recognition Letters*, 20(2):207–214, 1999. (Cited on page 62.)

[20] M. Glanznig, M. M. Malik, and M. E. Gröller. Locally adaptive marching cubes through iso-value variation. In *Proceedings of the International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pages 33–40, 2009. (Cited on page 9.)

[21] N. Gossett and B. Chen. Paint inspired color mixing and compositing for visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 113–118, 2004. (Cited on pages 62 and 76.)

[22] S. Grimm, S. Bruckner, A. Kanitsar, and M. E. Gröller. A refined data addressing and processing scheme to accelerate volume raycasting. *Computers & Graphics*, 28(5):719–729, October 2004. (Cited on pages 62 and 80.)

[23] H. Hagh-Shenas, S. Kim, V. Interrante, and C. Healey. Weaving versus blending: A quantitative assessment of the information carrying capacities of two alternative methods for conveying multivariate data with color. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1270–1277, 2007. (Cited on pages 62 and 78.)

[24] C. G. Healey. Perception in visualization - (September 30, 2009). `http://www.csc.ncsu.edu/faculty/healey/PP/index.html`. (Cited on page 2.)

[25] C. G. Healey and J. T. Enns. Large datasets at a glance: Combining textures and colors in scientific visualization. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):145–167, 1999. (Cited on page 62.)

[26] C. Heinzl, J. Kastner, and E. Gröller. Surface extraction from multi-material components for metrology using dual energy CT. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1520–1527, 2007. (Cited on pages 4, 32, 47, and 48.)

[27] C. Heinzl, J. Kastner, T. Möller, and M. E. Gröller. Statistical analysis of multi-material components using dual energy CT. In *Proceedings of Vision, Modeling and Visualization*, pages 179–188, 2008. (Cited on page 65.)

[28] C. Heinzl, R. Klingesberger, J. Kastner, and E. Gröller. Robust surface detection for variance comparison and dimensional measurement. In *Proceedings of the Eurographics/IEEE-VGTC Symposium on Visualization*, pages 75–82, 2006. (Cited on pages 4, 5, 32, 40, 42, and 44.)

[29] K. H. Höhne, M. Bomans, A. Pommert, M. Riemer, C. Schiers, U. Tiede, and G. Wiebecke. 3D visualization of tomographic volume data using the generalized voxel model. *The Visual Computer*, 6:28–36, 1990. (Cited on page 12.)

[30] J. Hsieh. *Computed Tomography: Principles, Design, Artifacts, and Recent Advances*. SPIE Optical Engineering Press, 2003. (Cited on page 59.)

[31] R. Huang, K. Ma, P. McCormick, and W. Ward. Visualizing industrial CT volume data for nondestructive testing applications. In *Proceedings of the IEEE Visualization*, pages 547–554, 2003. (Cited on page 45.)

[32] G. Kindlmann and J. W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of the IEEE Symposium on Volume Visualization*, pages 79–86, 1998. (Cited on page 12.)

[33] R. M. Kirby, H. Marmanis, and David H. Laidlaw. Visualizing multivalued data from 2D incompressible flows using concepts from painting. In *Proceedings of the IEEE Visualization*, pages 333–340, 1999. (Cited on page 62.)

[34] J. Kniss, G. Kindlmann, and C. Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proceedings of the IEEE Visualization*, pages 255–262, 2001. (Cited on page 12.)

[35] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002. (Cited on page 12.)

[36] J.M. Kniss, R. Van Uitert, A. Stephens, G.-S. Li, T. Tasdizen, and C. Hansen. Statistically quantitative volume visualization. In *Proceedings of the IEEE Visualization*, pages 287–294, 2005. (Cited on page 61.)

[37] L. P. Kobbelt and M. Botsch. Feature sensitive mesh processing. In *Proceedings of the 19th Spring conference on Computer graphics*, pages 17–22, 2003. (Cited on page 30.)

[38] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H. Seidel. Feature-sensitive surface extraction from volume data. In *Proceedings of the SIGGRAPH*, pages 57–66, 2001. (Cited on pages 32 and 48.)

[39] P. Kohlmann, S. Bruckner, A. Kanitsar, and M. E. Gröller. Evaluation of a bricked volume layout for a medical workstation based on java. *Journal of WSCG*, 15(1-3):83–90, January 2007. (Cited on page 62.)

[40] P. Kohlmann, S. Bruckner, A. Kanitsar, and M. E. Gröller. Livesync: Deformed viewing spheres for knowledge-based navigation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1544–1551, 2007. (Cited on page 4.)

[41] R. Kosara, C. G. Healey, V. Interrante, D. H. Laidlaw, and C. Ware. Thoughts on user studies: Why, how, and when. *IEEE Computer Graphics and Applications*, 23(4):20–25, 2003. (Cited on page 62.)

[42] D. H. Laidlaw, E. T. Ahrens, D. Kremers, M. J. Avalos, R. E. Jacobs, and C. Readhead. Visualizing diffusion tensor images of the mouse spinal cord. In *Proceedings of the IEEE Visualization*, pages 127–134, 1998. (Cited on page 62.)

[43] G. A. Laub and W. A. Kaiser. MR angiography with gradient motion refocusing. *Journal of Computer Assisted Tomography*, 12(3):377–382, 1988. (Cited on page 13.)

[44] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the SIGGRAPH*, pages 163–169, 1987. (Cited on pages 9, 47, and 49.)

[45] J. Maintz and M. Viergever. A survey of medical image registration. *Medical Image Analysis*, 2(1):1–36, 1998. (Cited on page 32.)

[46] M. M. Malik, C. Heinzl, and M. E. Gröller. Visual exploration for parameter studies of industrial CT dataset series. Technical report, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2008. (Cited on page 9.)

[47] M. M. Malik, C. Heinzl, and M. E. Gröller. Computation and visualization of fabrication artifacts. *Journal of WSCG*, 17(1):17–24, 2009. (Cited on page 9.)

[48] M. M. Malik, T. Möller, and M. E. Gröller. Feature peeling. In *Proceedings of Graphics Interface 2007*, pages 273–280, 2007. (Cited on page 7.)

[49] T. Malzbender. Fourier volume rendering. *ACM Transactions on Graphics*, 12:233–250, 1993. (Cited on page 13.)

[50] N. M. Matyas, L. Linsen, and B. Hamann. Metasurfaces: Contouring with changing isovalue. In *Proceedings of Vision, Modeling, and Visualization*, pages 147–154, 2005. (Cited on page 48.)

[51] J. Miller. Attribute blocks: Visualizing multiple continuously defined attributes. *IEEE Computer Graphics and Applications*, 27(3):57–69, 2007. (Cited on pages 62, 63, and 76.)

[52] S. Naple, M.P. Marks, R.D. Rubin, R.B. Jeffrey, M.D Dake, D.R. Enzmann, and McDonnell. CT angiography using spiral CT and maximum intensity projections. In *Proceedings of the Radiology*, pages 607–610, 1992. (Cited on page 13.)

[53] G. M. Nielson and B. Hamann. The asymptotic decider: resolving the ambiguity in marching cubes. In *Proceedings of the IEEE Visualization*, pages 83–91, 1991. (Cited on page 49.)

[54] N. Nikolaidis and I. Pitas. *3-D Image Processing Algorithms*. Wiley-Interscience, 2000. (Cited on page 32.)

[55] D. Patel, M. Haidacher, J. Balabanian, and M. E. Gröller. Moment curves. In *Proceedings of the IEEE Pacific Visualization Symposium 2009*, pages 201–208, 2009. (Cited on page 12.)

[56] V. Pekar, R. Wiemker, and D. Hempel. Fast detection of meaningful isosurfaces for volume data visualization. In *Proceedings of the IEEE Visualization*, pages 223–230, 2001. (Cited on page 12.)

[57] E. Pichon, D. Nain, and M. Niethammer. A Laplace equation approach for shape comparison. In *Proceedings of SPIE Medical Imaging*, pages 373–382, 2006. (Cited on page 31.)

[58] S. Potts and T. Möller. Transfer function on a logarithmic scale for volume rendering. In *Proceedings of the Graphics Interface*, pages 57–63, 2004. (Cited on page 12.)

[59] B. Preim and D. Bartz. *Visualization in Medicine: Theory, Algorithms, and Applications*. Morgan Kaufmann, 2007. (Cited on page 1.)

[60] R.J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: A systematic survey. *IEEE Transactions on Image Processing*, 14(3):294–307, 2005. (Cited on page 62.)

[61] Raindrop. *Raindrop: The magic of making it simple*. March 2007. `http://www.geomagic.com`. (Cited on page 32.)

[62] V. Rebuffel and J.-M. Dinten. Dual-energy X-ray imaging: benefits and limits. In *Proceedings of the European Conference on Non Destructive Testing*, 2006. (Cited on page 59.)

[63] M. Reiter, M. M. Malik, C. Heinzl, D. Salaberger, M. E. Gröller, H. Lettenbauer, and J. Kastner. Improvement of x-ray image acquisition using a GPU based 3DCT simulation tool. In *International Conference on Quality Control by Artificial Vision*, 2009. (Cited on page 9.)

[64] C. Rezk-Salama and A. Kolb. Opacity peeling for direct volume rendering. *Computer Graphics Forum*, 25:596–606, 2006. (Cited on pages 6, 7, 11, and 12.)

[65] S. Rossnick, D. Kennedy, G. Laub, G. Braeckle, R. Bachus, D. Kennedy, A. Nelson, S. Dzik, and P Starewicz. Three dimensional display of blood vessels in MRI. In *Proceedings of the IEEE Computers in Cardiology*, pages 183–196, 1986. (Cited on page 13.)

[66] Y. Sato, N. Shiraga, S. Nakajima, S. Tamura, and R. Kikinis. Local maximum intensity projection (LMIP): A new rendering method for vascular visualization. *Journal of Computer Assisted Tomography*, 22(6):912–917, 1998. (Cited on page 13.)

[67] C.E. Scheidegger, H.T. Vo, D. Koop, J. Freire, and C.T. Silva. Querying and creating visualizations by analogy. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1560–1567, 2007. (Cited on page 4.)

[68] W. J. Schroeder, K. M. Martin, and W. E. Lorensen. The design and implementation of an object-oriented toolkit for 3D graphics and visualization. In *Proceedings of the IEEE Visualization*, pages 93–101, 1996. (Cited on page 51.)

[69] W. J. Schroeder, K. M. Martin, and W. E. Lorensen. *The Visualization Toolkit, An Object-Oriented Approach To 3D Graphics, Third Edition*. 2002. (Cited on pages 1 and 49.)

[70] R. Sisneros, C. R. Johnson, and J. Huang. Concurrent viewing of multiple attribute-specific subspaces. *Computer Graphics Forum*, 27(3):783–790, 2008. (Cited on page 61.)

[71] R. Stokking, I. G. Zubal, and M. A. Viergever. Display of fused images: Methods, interpretation, and diagnostic improvements. *Seminars in Nuclear Medicine*, 33(3):219–227, 2003. (Cited on page 63.)

[72] R. Taylor. Visualizing multiple fields on the same surface. *IEEE Computer Graphics and Applications*, 22(3):6–10, 2002. (Cited on page 62.)

[73] G. Turk. Re-tiling polygonal surfaces. *ACM SIGGRAPH Computer Graphics*, 26(2):55–64, 1992. (Cited on page 31.)

[74] FH OOE Forschungs und Entwicklungs GmbH. Computer-Tomograph HWM Rayscan 250XE - (September 30, 2009). `http://www.3dct.at/index.php?show=ctanlage`. (Cited on page 2.)

[75] T. Urness, V. Interrante, I. Marusic, E. Longmire, and B. Ganapathisubramani. Effectively visualizing multi-valued flow data using color and texture. In *Proceedings of the IEEE Visualization*, pages 115–121, 2003. (Cited on pages 62 and 76.)

[76] I. Viola, M. Feixas, M. Sbert, and M. E. Gröller. Importance-driven focus of attention. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):933–940, 2006. (Cited on page 4.)

[77] I. Viola and M. E. Gröller. Smart visibility in visualization. In *Proceedings of EG Workshop on Computational Aesthetics / Computational Aesthetics in Graphics, Visualization and Imaging*, pages 209–216, 2005. (Cited on page 4.)

[78] Volume-Graphics. *VGStudio Max 1.2-User's Manual*. 2004. `http://www.volumegraphics.com/products/vgstudiomax`. (Cited on pages 32, 61, and 75.)

[79] P. Šereda. *Facilitating the Design of Multidimensional and Local Transfer Functions for Volume Visualization*. 2007. PhD thesis at Eindhoven University of Technology. (Cited on page 48.)

[80] C. Weigle and R. M. Taylor. Visualizing intersecting surfaces with nested-surface techniques. *IEEE Visualization, 2005.*, pages 503–510, 2005. (Cited on page 31.)

[81] Wikipedia. Magnetic resonance imaging - (September 30, 2009). `http://en.wikipedia.org/wiki/Magnetic_resonance_imaging`. (Cited on pages 2, 4, and 6.)

[82] Wikipedia. X-ray computed tomography - (September 30, 2009). `http://en.wikipedia.org/wiki/X-ray_computed_tomography`. (Cited on pages 1 and 2.)

[83] C. T. Yang, S. Lai, and L. Chang. Hybrid image matching combining Hausdorff distance with normalized gradient matching. *Pattern Recognition*, 40(4):1173–1181, 2007. (Cited on page 62.)

[84] H. Zhou, M. Chen, and M. F. Webster. Comparative evaluation of visualization and experimental results using image comparison metrics. In *Proceedings of the IEEE Visualization*, pages 315–322, 2002. (Cited on page 62.)

[85] L. Zhou and A. Pang. Metrics and visualization tools for surface mesh comparison. In *Proceedings of the SPIE Visual Data Exploration and Analysis VIII*, volume 4302, pages 99–110, 2001. (Cited on page 31.)

> Strive not to be a success, but rather to be of value.
>
> Albert Einstein

Curriculum Vitae

Personal Details

Name	Muhammad Muddassir Malik محمّد مدّثر ملک
Address	Favoritenstrasse 9-11 / E186, A-1040 Vienna, Austria, Europe
Phone number	+43 (1) 58801 18659 +92 (997) 301448
Email address	malik@cg.tuwien.ac.at muddassir.malik@gmail.com
Date of Birth	21st January, 1980
Nationality	Pakistani
Languages	Urdu(اردو), English, Hindko(هندکو)
Interests	Computer Graphics, Soccer, Trekking

Educational Background

since 2005 PhD study in Computer Science at the Vienna University of Technology. Research conducted in the field of scientific visualization with emphasis on volume visualization. The topic of the doctoral dissertation is "Feature Centric Volume Visualization". Supervised by Prof. M. Eduard Gröller.

2002 - 2004 Masters of Science in Computer Science from the Lahore University of Management Sciences, Lahore. The title of the master's thesis is "Restructuring the IT Maturity Models".

1999 - 2002 Bachelors of Computer Science with honors from the Bahria University, Islamabad. The final project was "Parallel Processor Matrix Multiplication and Load Balancing"

1985 - 1998 O' levels and high school graduation from Burn Hall School and College, Abbottabad.

Working Experience

since 2005 Research Assistant at the Institute of Computer Graphics and Algorithms, Vienna University of Technology. Responsibilities include conducting research in scientific visualization, supervising projects, organizing computer graphics seminars, and assisting faculty in teaching and examinations. I was part of the research team working on the SimCT (simulation of industrial 3D computed tomography) project. The project was a collaboration between the Institute of Computer Graphics and Algorithms, Research & Development Ltd., Institute of Solid State Physics, and Carl Zeiss.

I was part of the teaching staff in the following courses:

1. SS06 - Teacher's assistant for information visualization course
2. WS07 - Organization and supervision of computer graphics seminar
3. WS08 - Organization and supervision of computer graphics seminar
4. WS09 - Organization and supervision of computer graphics seminar

I supervised the following student projects:

1. Locally adaptive marching cubes
2. Efficient voxelization of surface models
3. Determining optimal specimen orientation for 3DCT scan
4. GPU based reconstruction of industrial CT data

2004 - 2005 Lecturer at the Comsats Institute of Information Technology. Responsibilities included delivering lectures to graduate and undergraduate students and supervising projects.

2001 - 2004 Freelance Programmer. Design and development of small scale softwares and modules.

Professional Activities

Referee for Journals: IEEE Transactions on Visualization and Computer Graphics (TVCG), Journal of WSCG.

Referee for Conferences: International Conferences in Central Europe on Computer Graphics and Visualization (WSCG), Pacific Graphics, Spring Conference of Computer Graphics (SCCG), and Vision, Modeling, and Visualization (VMV).

Invited talks:

1. Quality Control for Industrial 3D X-ray Computed Tomography *in Visgroup-VrVis Symposium* on February 05, 2008.
2. Techniques for Variance Comparison *in Visgroup-Oslo Mini Symposium* on April 24, 2009.

Publications

M. M. Malik, T. Möller, M. E. Gröller, "Feature peeling," in proceedings of *Graphics Interface 2007*, pp. 273–280, 2007, `http://www.cg.tuwien.ac.at/research/publications/2007/malik-2007-FP/`.

M. M. Malik, C. Heinzl, M. E. Gröller, "Computation and Visualization of Fabrication Artifacts," in *Journal of WSCG*, 17(1-3), pp. 17–24, 2009, `http://www.cg.tuwien.ac.at/research/publications/2009/malik-2009-CVFA/`.

M. Glanznig, M. M. Malik, M. E. Gröller, "Locally Adaptive Marching Cubes through Iso-value Variation," in proceedings of *International Conference in Central Europe on Computer Graphics and Visualization 2009*, pp. 33–40, 2009, `http://www.cg.tuwien.ac.at/research/publications/2009/glanznig-2009-LAMC/`.

M. Reiter, M. M. Malik, C. Heinzl, D. Salaberger, M. E. Gröller, H. Lettenbauer, J. Kastner, "Improvement of X-Ray image acquisition using a GPU based 3DCT simulation tool," in proceedings of *International Conference on Quality Control by Artificial Vision 2009*, pp. 123–456, 2009, `http://www.cg.tuwien.ac.at/research/publications/2009/Reiter_2009_IXIA/`.

M. M. Malik, C. Heinzl, M. E. Gröller, "Visual Exploration for Parameter Studies of Industrial CT Dataset Series," *Technical Report* 2009, `http://www.cg.tuwien.ac.at/research/publications/2008/TR-186-2-08-12/`, (submitted)