MASTERARBEIT

Efficient CPU-based Direct Volume Rendering For CT-Angiography

Ausgeführt am

Institut für Computergraphik und Algorithmen
der Technischen Universität Wien

unter Anleitung von

Univ. Doz. Dipl. Ing. Dr. Miloš Šrámek
Österreichische Akademie der Wissenschaften

und

Dipl. Ing. Dr. Matúš Straka
Österreichische Akademie der Wissenschaften

durch

Matthias Michael Bernhard
Hirschkellerstr. 8
D-88171 Weiler-Simmerberg

Wien, 30.11.2006

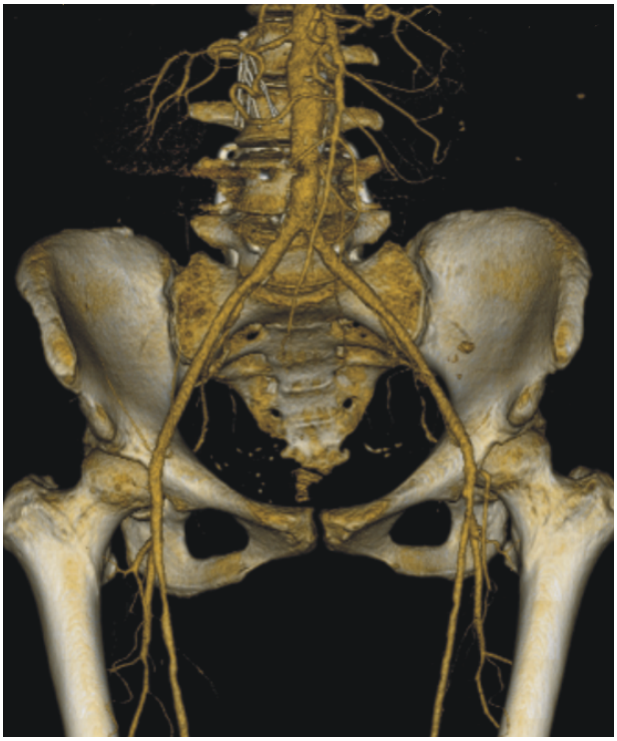
_____

Unterschrift

Matthias Michael Bernhard

Efficient CPU-based Direct Volume Rendering For CT-Angiography

Master's Thesis



Supervised by

Univ. Doz. Dipl. Ing. Dr. Miloš Šrámek

and

Dipl. Ing. Dr. Matúš Straka

Abstract

This work reports the development of a CPU-based Direct Volume Rendering (DVR) software, which was implemented for medical visualisation in the field of CT-Angiography. An interactive DVR-plugin was finally added to the so called AngioVis-Toolbox, a software for post-processing and visualisation of CTA datasets. This work was focusing on finding adequate solutions to provide an interactive behaviour of the DVR-tool. This was accomplished by investigating several acceleration strategies, which were assessed with respect to their effectivity. Further this work sketches an alternative efficient solution for vessel-tree related focus and context visualisation referring to the so-called concept of "VesselGlyph", which was proposed before by members of the AngioVis-Project.

Zusammenfassung

Diese Arbeit befasst sich mit der Entwicklung einer Direct Volume Rendering (DVR) Software, die für die medizinische Visualisierung im Bereich der CT-Angiographie entworfen wurde. Es wurde letztendlich ein interaktives DVR-Plugin in die sogenannte AngioVis-Toolbox, ein Softwarepaket zur Verarbeitung und Visualisierung von CTA-Datensätzen, integriert. Diese Arbeit hat sich hauptsächlich damit befasst passende Lösungen zu finden, um ein interaktives Verhalten des DVR-Moduls zu ermöglichen. Dies wurde erreicht durch die Erprobung und Anwendung verschiedener Beschleunigungsverfahren, deren Effektivität schließlich evaluiert wurde. Ausserdem wird ein alternativer kostengünstiger Ansatz vorgeschlagen, zur Realisierung einer Fokus und Kontext Visualisierung die sich am Verlauf der Gefäße orientiert und dessen Konzept bereits vorher im Rahmen des AngioVis-Projekts unter der Notation "VesselGlyph" vorgeschlagen wurde.

Aknowledgements:

I would like to thank:

- Matúš Straka for introducing me into the AngioVis-Project and his invalueable support for programming the software.
- My supervisor Miloš Šrámek for his support, openness and the time he spent for helpful discussions, letting me benefit from his great experiences.
- Andrej Varchola, Leonid Dimitrov and the other people from ÖAW-Viskom for giving me useful feedback on my presentations.

Contents

1 Introduction **1**

1.1 Preface 1

1.2 The AngioVis Project 2

1.2.1 Peripheral Occlusive Aterial Disease 2

1.2.2 Vessel Visualisation Purposes 3

1.3 Digital Angiography and Visualisation Techniques 4

1.3.1 Digital Subtraction Angiography 5

1.3.2 Visualisation for 3D-Angiography 5

2 Fundamentals of Direct Volume Rendering **12**

2.1 Classification 13

2.2 Reconstruction 14

2.2.1 Reconstruction Theory 14

2.2.2 Simple Reconstruction Filters 17

2.3 Shading 19

2.3.1 Derivatives of Discrete Volume Data 19

2.3.2 Shading Model 21

2.4 Compositing 22

2.4.1 Optical Model 22

2.4.2 Discrete Compositing 24

3 Technical Solutions for DVR **28**

3.1 Major Approaches 29

3.1.1 Raycasting 29

3.1.2 Splatting 30
3.1.3 Object-order Raycasting: Shear and Warp 32
3.1.4 Texture Mapping 33
3.1.5 Utilising GPU 34
3.2 Re-sampling and Classification 35
3.2.1 Re-sampling Pipelines 35
3.2.2 Transfer Function Design 36
3.2.3 Reconstruction Filters 39
3.2.4 Gradient Estimation 40
3.3 Accelerating Raycasting of Large Datasets 43
3.3.1 Bricked Memory Layout 43
3.3.2 On-the-fly Processing 44
3.3.3 Parallelisation 45
3.3.4 Early Ray Termination 47
3.3.5 Empty Space Leaping 47
3.3.6 Adaptive Sampling 50

4 Implementation and own solutions 53
4.1 Circumstances and Limitations 53
4.2 Main Approach 54
4.2.1 Memory Structure 54
4.2.2 Transfer Function 55
4.2.3 Rendering Passes 56
4.2.4 Resampling and Compositing 57
4.3 Acceleration 57
4.3.1 Block-by-block Processing 57
4.3.2 Ray Traversal Templates 58
4.3.3 Quadtree Space Leaping 61
4.3.4 Caching 64
4.4 Interactive Mode 64
4.5 Results 68
4.5.1 Evaluation of Acceleration Techniques 68

4.6 Future Improvements 78

5 Focus and Context Visualisation by the Concept of VesselGlyph 81

5.1 Motivation and Idea 81

5.2 Realisation 82

5.2.1 VesselGlyph DVR 84

5.2.2 VesselGlyph DVR+CPR 85

5.3 Implementation of VesselGlyph DVR 87

5.3.1 Algorithm 87

5.3.2 Efficient Solution 89

5.3.3 Current State of Development 90

6 Conclusion and Future Work 92

6.1 Conclusion 92

6.2 Own Ideas and Proposals 93

Chapter 1

Introduction

1.1 Preface

The work presented in this thesis is a contribution to the AngioVis project, which addresses to visualisation techniques for *Computer Tomographic Angiography*(CT-Angiography,CTA). The AngioVis-Group is an interdisciplinary team of medical and computer science experts, who are working on the visualisation of peripheral CT-angiographic datasets. Vascular imaging techniques support physicians, particularly vascular surgeon and cardivascular specialists, in reading and interpreting large CT angiographic datasets for clinical decision making and treatment planning.
Since this diploma-thesis belongs to the field of medical visualisation, it will present an overview on angiographic visualisation techniques. This will be introduced by a brief description of the AngioVis-Project, its' motivation and the software AngioVis-Toolbox. The contribution of this work is the development, implementation and evaluation of an efficient CPU-based *Direct Volume Rendering* (DVR) software for the purposes of CT-Angiography, which is integrated as an interactive visualisation tool in the AngioVis-Toolbox. In recent years DVR has become a well known rendering technique and thus a lot publications about various approaches and improvements are avaible. This work will present the theoretical fundamentals and discuss several practical

solutions for DVR, especially acceleration techniques, in order to implement the most suitable ones for our purposes. In addition an approach for focus- and context-visualisation by the concept of VesselGlyph will be presented as extended functionality of our DVR software.

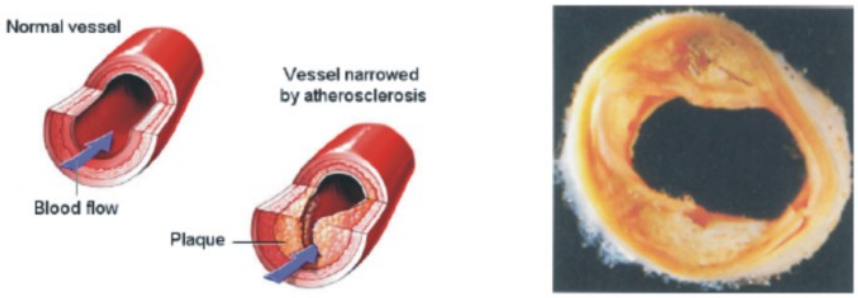
1.2 The AngioVis Project

3D-Imaging techniques as CT- or *Magnet Resonance Imaging* (MRI) scanning have become a common practice in radiology. These techniques allow a non-invasive insight into the human body by the use of various 3D-visualisation tools. Medical visualisation of 3D-data challenges to generate clinical valuable 2D-projections. The *AngioVis-Project* (AVP) is concerned with post processing and visualisation of CT-Angiographic data in order to support radiologists and physicians in diagnosis and treatment-planning for patients suffering from *Peripheral Occlusive Aterial Disease* (POAD).

1.2.1 Peripheral Occlusive Aterial Disease

Pathologic changes in the vascular system are often results of an unhealthy lifestyle, including bad habits as excessive smoking and drinking, unbalanced fatty nutrition, insufficient exercise or even psychological exposures like stress. Further risk factors are an innate hazard and chronic diseases as diabetes, adipositas or hypertension, which may be also an intermediate result of unhealthy habits. Artheriosclerosis has become a civilisation disease with an increasing incidence in developed countries. Its' pathogenesis is often a mute slowly progressing and cumulative process, which may have a duration over several decades. The disease is caused by artheriosclerotic plaques induced by the deposit of lipids, connective tissues, thrombi or calcifications at the inner vessels walls, which protrude into the vessels flow lumen and narrow the blood flow. When plaques are narrowing the arterial blood-flow significantly (over 50% of the diameter), they cause clinical symptomes, as e.g. pain in the limbs or a restricted ability to walk long distances, in the

beginning. Advanced stages may cause severe complications as not healing wounds, strokes or even heart attacks. The scope of AVP are narrowings and occlusions of peripheral arteries only, which are the clinical manifestations of artheriosclerosis within the lower extremity arteries. Atherosclerotic plaque in peripheral arteries causes early symptoms like pain while walking, namely "intermittent claudication". In advanced stages of this disease, patients may suffer from severe complications, as rest pain or even tissue loss. Short segment luminal stenoses can be treated with catheter techniques (balloon angioplasty), but long segments of complete occlusion may even require a surgical bypass-graft to reestablish flow.



(a) (b)

Figure 1.1: Arterial narrowing caused by the deposit of plaques (a). Photograph of the traverse section of a calcified vessel (b).

1.2.2 Vessel Visualisation Purposes

CT-Angiography requires a powerful medical image post processing workstation to visualise the aquisited data in a valuable manner. In order to perform CTA on a regular basis, a standardised workflow and visualisation protocols are necessitated as well [2]. The research of AVP is concerned with the development, implementation and evaluation of various preprocessing tools, which make lesions, luminal narrowings or occlusions, and their degree, length and

location in the peripheral vessel-tree, visible. Physicians shall be supported to detect, analyse and localise them for purposes of their clinical activities. For instance, it is clinically important to distinguish between calcified plaque, versus significant narrowing or complete occlusion.
The process of vessel visualisation includes also preparation steps for segmentation and classification to introduce high-level information into the finally visualised data. This enables an enhancement of important vascular details and a suppression or removal of less interesting or obscuring objects (e.g. bones). These tools and visualisation techniques are assembled in the post-processing software *AngioVis-Toolbox*(AVTB). Its' goal is to provide an usable functionality to perform a standardised post-processing workflow in order to obtain interpretable and informative images of the peripheral vessel tree. The selection of appropriate visualisation tools depends on the the inspected patients' disease and further circumstances, but also on the purpose of visualisation: interpretation or documentation. For interpretation tasks a fast navigation and flexible visualisation is preferred. An interactive exploration software with a well designed user interface and time-efficient performance, enabling to switch and navigate fast through different visualisations, satisfies this demand. For the purpose of documentation one needs to create a protocol-driven set of standardised static images and measurements from predefined views. Typical protocols for CTA post processing include *Curved Planar Reformations* (CPR), various kinds of *Maximum Intensity Projections* (MIP), a bone removal, and *Volume Renderings* (VR) of different parts of the body [2].

1.3 Digital Angiography and Visualisation Techniques

The most common angiography techniques is X-ray angiography. In recent times, *Magnet Resonance Imaging* (MRI),*Computer Tomography* (CT) and/or ultrasound diagnostic vascular imaging techniques have been devel-

oped for diagnostic purposes. All X-ray techniques require contrast medium injections during acquisition, because blood vessels do not contrast sufficiently with the surrounding tissues. Today, digital x-ray imaging techniques as intra-arterial *Digital Subtraction Angiography* (CTA) and CT-Angiography (CTA) are commonly used for the angiography of peripheral arteries. As both, conventional Angiography and DSA, require arterial catheterisation, the CTA and *MR-Angiography* (MRA), which require intravenous contrast medium injection only, are less invasive and less costly alternatives. Thus and due to technical progression, both vascular 3D-imaging techniques, MRA and CTA, are ongoing to replace the diagnostic catheter-based DSA.

1.3.1 Digital Subtraction Angiography

The term *Digital Subtraction Angiography* (DSA) refers specifically to techniques which subtract two images that are obtained before and after contrast media was administered to the patient. It was developed to improve vessel contrast to overcome the imaging problems when bones and other dense tissues obscure and interfere the display of vessels. The idea of subtraction images was first proposed by the Dutch radiologist Ziedses des Plantes in the 1930's, when he was able to produce subtraction images using plain film. The image which was recorded before the contrast agent was administered is called the mask image. The other image taken with contrast medium is called opacification image. The anatomical structures that are the same in the mask and opacification image can be removed, so the resulting image shows the vessels only. Finally the subtraction is performed arithmetically with digitised images from different frame shots.

1.3.2 Visualisation for 3D-Angiography

Volume visualisation is an important challenge for post processing medical 3D-images from devices like CT or MRI. To represent 3D-image data preferably informative, several post processing and visualisation techniques have

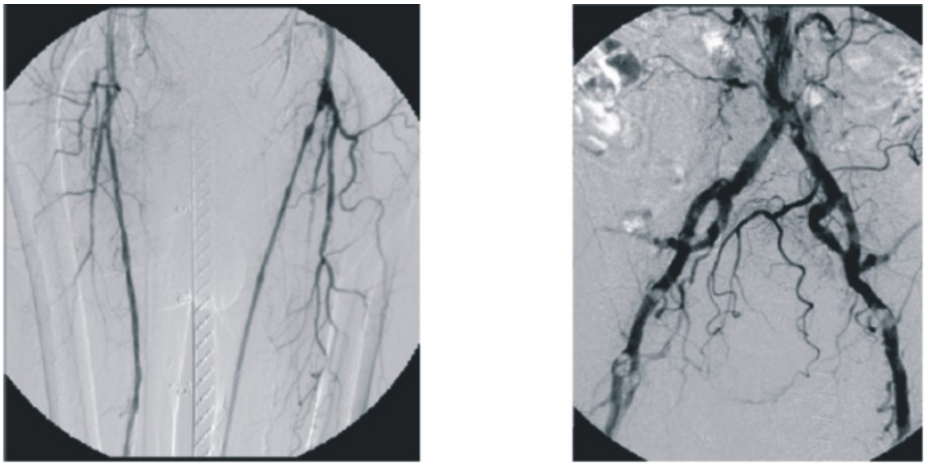


Figure 1.2: DSA is a powerful visualisation technique and provides highly resolved images. This is a DSA of the lower extremity arteries. The black branches are the contrast-medium enhanced vessels.

been developed and introduced into the AVTB.
The AVP has chosen to focus on CTA only, this has following reasons: In comparison to DSA, a 3D data representation provides new and more opportunities for information extraction and visualisation. On the other hand, CT devices are currently more available in clinics, have a simpler and faster acquisition, and also a higher spatial resolution than common MRI devices. However, the AVP regards the assessment and improvement of the clinical potential and usability of CTA as its' primary challenge at the moment [6]. Adaption to MRA is planned as future projects.

Segmentation, Classification and the Vessel Tree Model

The visualisation is the final step of a sophisticated post-processing pipeline. In order to model, enhance, suppress or extract certain informations from the input data, higher level information has to be extracted from the density grid representing the CT-recording. The AVTB contains segmentation and classification tools to label voxels with the kind of tissue they belong to. Although the sole distinction between vessels, bones and other tissues

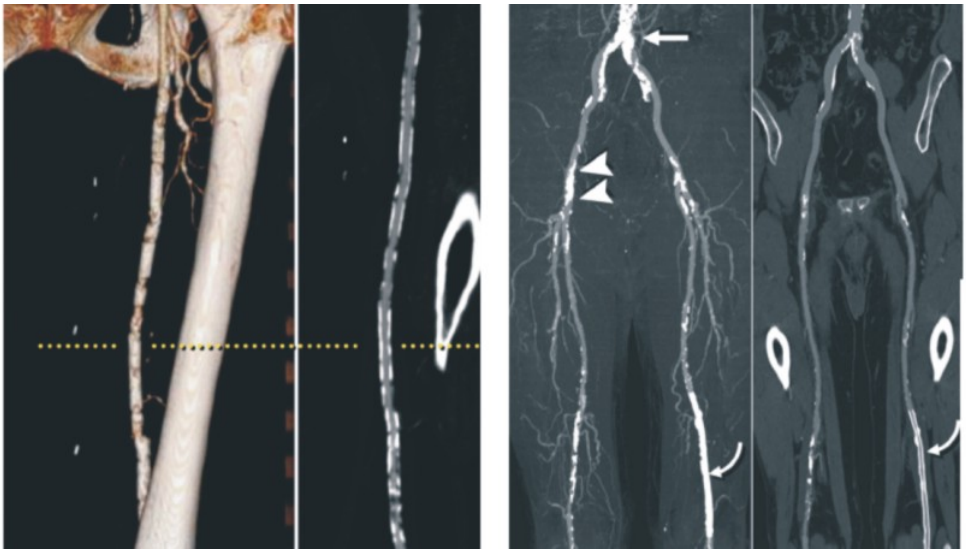


Figure 1.3: Examples from angiographic documentation: visualisation of a calcified vessel with VR and CPR (left), visualisation of a diseased vessel tree with MIP and CPR.

is sufficient for the purposes of CTA, this process remains a difficult task due to overlapping densities, the presence of noise and artifacts, insufficient contrast, spatial distortion, inter- and intra-patient variabilities. Hereby the major difficulty is the differentiation between vessel and bone tissues, mainly because of spatial proximity and overlapping densities. Since CTA deals with information of clinical relevance, a high reliability of any processing step is indispensable. Hence, this procedure must perform with high accuracy, as a false classification may have severe consequences - e.g. important vessels may disappear when out-segmenting the bones. As thresholding or region-growing algorithms fail to perform this challenging task, either manual, user-driven approaches or more complex segmentation algorithms have to be employed. The most suitable methods for vessel-tree extraction and bone segmentation, comprising a combination of watershed transform and the probabilistic atlas approach, were assessed by Straka et al. [8, 9, 6]. If the segmentation of vessels succeeds, high level information about the vessel-tree can be retrieved in order to model the vessel-tree and its' properties. This process of modeling

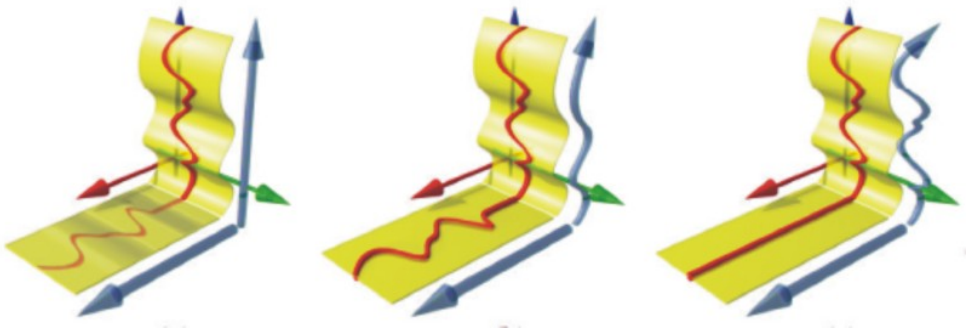
the vessel tree involves three main steps: estimation of the centerline and the radius of the cylindrical segments, building of the vessel tree hierarchy and precise modeling of the bifurcations[6]. To estimate the vessel-trees centerline, a so-called 'vessel tracking' algorithm was developed (described by Kanitsar et al. in [1]), which requires a manual specification of start- and end-points of the vessel-path to be tracked. Since the process of modeling the vessel-tree is also a very complex task, it requires several investigations (presented in the PhD-thesis of LaCruz [7]) to overcome various problems, especially segmentation inaccuracies and the presence of vascular abnormalities caused by diseases. Finally the retrieved vessel-tree model is essential to generate vessel surface meshes, CPR and context/focus-visualisations by the concept of VesselGlyph. The main visualisation techniques and their purpose will be summarised below. The concept of VesselGlyph will be elucidated in Chapter 5.

Maximum Intensity Projections

Maximum Intensity Projections (MIP) display the maximum densities of all voxels that contribute to the same pixel in a 2D projection (two images are shown in Figure 1.5 (a) and (b)). In comparison to VR images, MIP images are less overloaded and in an appropriate usage they display only important details of the data [46]. MIP images are quite similar to conventional angiography and display also very thin collateral vessels. But to visualise all vessels, obscuring bones have to be removed. Those can be cut out, if the dataset was segmented before.

Curved Planar Reformation

Volumetric data from medical acquisition may contain many objects of less or no diagnostic interest. In case of CTA the diagnostic focus lies on vascular detail. As the vessel-tree extends a very small volume in relation to the whole dataset, visualisation techniques apart from conventional volume rendering strategies, which provide incomplete clinical information only, are



(a) (b) (c)

Figure 1.4: CPR projection methods: (a) parallel projected, (b) stretched, (c) straightened [4].

needed. To highlight important details of the vessels, a technique giving an accurate inside view of the vessels is desired. *Curved Planar Reformation* (CPR) is a method to visualise vascular structures with small diameters by resampling a longitudinal curved planar cut throughout the volume, embedding the vascular centerline (example in Figure 1.5 c). Therefore the vesseltree centerline path is required, which is a high level information obtained from the process of vesseltree-modeling. Kanitsar et al. [4] presented several strategies to project and sample the data by CPR (Figure 1.4). To define the sampling method they introduced a *vector of interest (vi)* and a *line of interest (li)*. The first defines the sampling-direction in one slice and the latter is the line, defined by *vi*, along each voxel being touched is sampled. The simplest visualisation can be performed by a parallel projection of the *li* samples onto the image plane with the desired viewing angle. A rotation is performed around each single point of the vascular centerline, keeping the *vi* permanently orthogonal to the viewing axis. To retain isometry, which is distorted by parallel projection, a stretched projection can be applied. This is performed by a longitudinal stretching of the curved plane to a non curved plane, which can be directly mapped to the image plane. When one want to measure extension of vascular abnormalities, straight linear aligned vessels might be preferably, what is provided by straightened CPR.

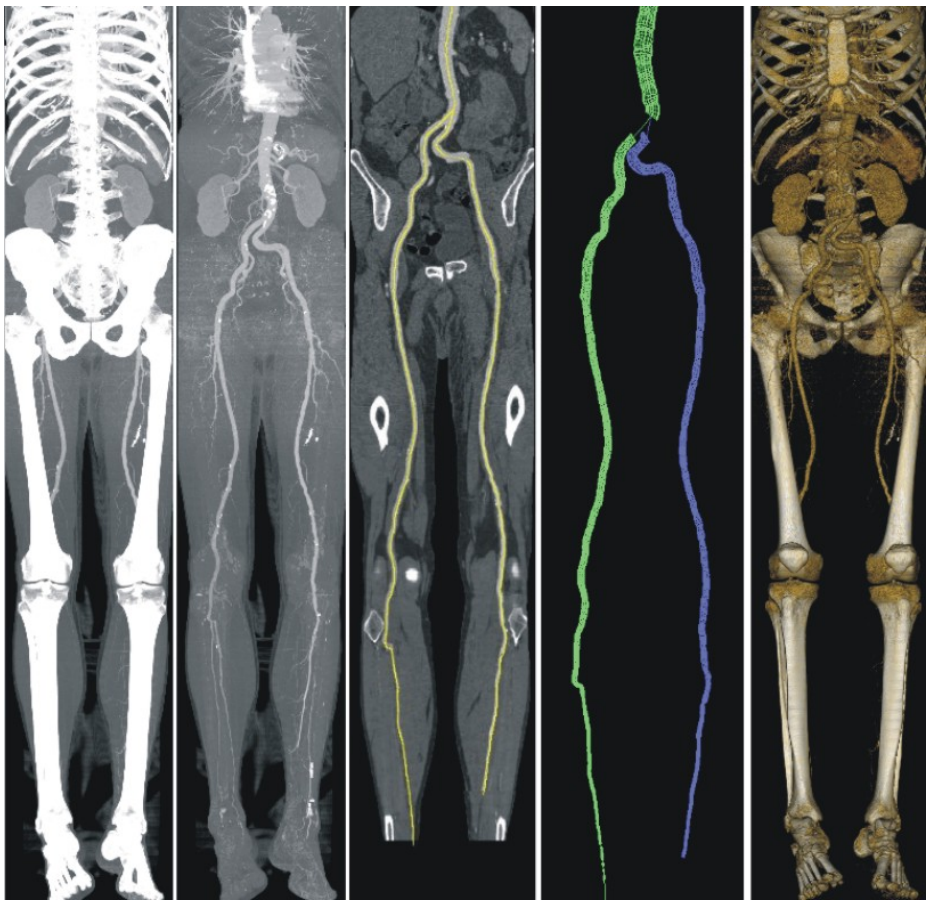
CPR emerged as a capable method to display most valuable information about the vessel tree and its pathological changes. Its' main advantage is its' ability to show the vessels flow lumen, even in the presence of strong calcifications or stents. But it suffers from deformation of the context information, which has to be obtained by other visualisation methods. To provide both goals at the same time, visualisation of clinical relevant information and preserving local orientation by context visualisation, hybrid techniques were investigated, which combine different visualisations for context and focus regions in one image. These approaches refer also to the notation VesselGlyph (Chapter 5).

Volume Rendering

Generally the term *Volume Rendering* (VR) can be described as a method to project a volume dataset onto a 2D image plane. This is also done when a MIP is performed, thus in the strict sense MIP methods are some special kind of VR. But usually the term VR refers to a method which displays surfaces and tissues in a manner that the 3D information is preserved by depth cues (example in Figure 1.5 e). The notation *Direct Volume Rendering* (DVR) rather refers to the method which is used to obtain VR images (see Section 2).

Because VR images provide depth cues, bone editing is not mandatory, as it is for MIP. Instead of bone removal, relevant details can be exposed by interactive adjustment of the viewing direction and clipping planes. Exquisite vascular detail can be blended in or carved out by interactively manipulating the opacity transfer function [2]. In clinical practice one can use VR as a valuable tool for interactive exploration, which especially supports the observers spatial orientation, since the VR images provide an intuitive comprehensible and realistic impression. Further, they are well suited for educational and illustration purposes, as "snapshot views" in medical publications for example. But the disability to display coloured images on the high-resolution greyscale monitors of the *Picture Archivating and Communication Systems* (PACS) is

a notable restriction for the usage of this visualisation technique. However, the main limitation of VR, and these applies also for MIP, is its' inability to visualise the vascular flow channel when it is obscured by calcifications or stents. Due to this fact VR remains in the majority of cases a subsidiary visualisation technique in the field of CTA [3]. But it has to be remarked, that every kind of visualisation technique has its limitations, and thus only a conjunction of different techniques may satisfy all goals.



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

Figure 1.5: Images obtained by different visualisation techniques: (a) MIP, (b) MIP with outsegmented bones, (c) parallel projected CPR, (d) vessel tree model as polygon mesh, (e) DVR

Chapter 2

Fundamentals of Direct Volume Rendering

In fact, the notation Volume Rendering is commonly used for two distinct approaches: *Indirect Volume Rendering* (IDV) and *Direct Volume Rendering* (DVR) [13]. IVR uses an intermediate conversion step which transforms the volume in a surface representation, like a polygonal mesh or binary volume. This is done by isosurface extraction methods like a straightforward binary classification or advanced techniques, which define a function over a neighbourhood of each voxel and classify the results. Very popular became the Marching Cubes algorithm [12], which creates a polygonal mesh by the use of mesh primitives being selected according to a joint classification of all eight voxels at the corners of each cell.
DVR has a rendering pipeline without an intermediate conversion step. It was invented to overcome several accuracy problems of isosurfacing techniques. These problems are related to making a decision for every volume element whether or not the surface passes through it. This can produce false positives or false negatives, particularly in the presence of small or poorly defined features [56]. Because DVR does not use intermediate geometrical representations, it has the capability to display weak or fuzzy surfaces. This frees one from the requirement to make a decision whether a surface is

present or not [10]. Instead, an object's surface is associated with a range of values, and an opacity function can make a range of values opaque or translucent. Levoy [10] presented the first DVR approach in 1988. This involved the following steps: the transformation of the density values into colors and opacities (Classification), reconstruction of a continuous function from the discrete grid of values (Reconstruction), and projecting it onto the 2D viewing plane from the desired point of view (Compositing). The output image is generated by sampling and compositing the RGBAs in viewing direction, what can be performed e.g. by ray-casting. To convey depth effects the surfaces light reflection behaviour has to be modelled, what is actually done by shading.

2.1 Classification

To make a data volume visible, one first have to assign renderable optical properties to its content. Optical properties in DVR are at least defined by an opacity value (α) and a chromacity, which is usually represented with the *Red-Green-Blue* (RGB) colour model. Further properties might concern reflectance, emittance or even non-photorealistic effects. To obtain optical properties from numerical data, a *Transfer Function* (TF) has to be specified. It can be generally defined as a function

$$\tau : P_1 \times P_2 \times \ldots \times P_n \rightarrow O_1 \times O_2 \times \ldots \times O_m \ , \tag{2.1}$$

which maps a cartesian product of scalar fields of input parameters P_i to a cartesian product of optical properties O_i. In a strict sense, this definition also includes the process of shading into the TF-mapping. But in the most usual and simplest case, TFs only map from densities to chromacities and opacities ($n = 1$,$m = 2$). TFs with several input parameters are called multidimensional TFs. Input parameters contain certain qualifying informations extracted from the data volume. This can be pure segmentation information or simply the densities. Between those extrema of high and low level information, several other qualifiers can be obtained by 3D-preprocessing tools,

like gradient filters, for instance. As those information extracting functions contribute also to the behaviour of the TF, it was generalised by Fang et. al [23] to a set of preprocessing tools and a mapping functions. But conventional forms of TF-representation are piecewise linear functions, polynomials and splines, which are determined by a set of interpolation-point/value tupels $(s,t)^{(i)} = \left((p_1,\ldots,p_n)^{(i)},(o_1,\ldots,o_n)^{(i)}\right)$. In order to visualise exactly details of interest, one have to specify an elaborate TF. That is a very difficult task, even for a simple TF model ($n = 1$,$m = 2$). Hence, various approaches focus just on assisting and easing the design of TFs (see Section 3.2.2).

2.2 Reconstruction

A volume dataset is an array of density values, which represents samples at discrete positions in a three dimensional grid. The term *Reconstruction* refers to the process of obtaining a density function which is defined throughout the volume, given a set of discrete samples [35]. Reconstruction has a major impact on the quality of the output image and is also applied to obtain the gradients, which are required for shading. The theory about the ideal reconstruction function actually belongs to the fundamentals of digital signal processing.

2.2.1 Reconstruction Theory

When a real-world object is scanned, it is sampled in (each dimension) with a certain frequency f_S, whereas each sample belongs to a discrete position. In abstract terms, this process of sampling can be modeled by a signal multiplication. To simplify matters, we reduce this consideration to an one dimensional case:

Let $g(t)$ be a periodical signal which meets the Dirichlet conditions [1] and contains the continuous information of the object's space [2] . Shall $k : \Re \rightarrow \{0, 1\}$ be the sampling function defined as an infinite set of unit Dirac impulses shifted to each sampling position $n\frac{1}{f_S}$, with $n \in Z$.

Sampling corresponds to the signal multiplication $g_s = gk$. The product $g_s(t)$ equals $g(t)$ for all $t = \frac{n}{f_S}$ and is zero for all other t. To issue the reconstruction of g from the product g_s, we regard the frequency domain, where the multiplication of g and k corresponds to the convolution of their Fourier transforms $\hat{g}$ and $\hat{k}$. The convolution with $\hat{k}$, which is also an impulse grid in the frequency domain, folds the spectrum of $\hat{g}$ to an infinite number of identical spectra shifted by integer multiples of the angular sampling frequency $\omega_S = 2\pi f_S$. The spectrum centered at zero is the primary spectrum, the others are alias spectra. If f_S is at least twice the highest frequency occurring in the input signal g, there is no interlace between the alias spectra and thus it can be separated without a loss of information by virtue of aliasing [3]. The region used to separate the primary spectrum is called Nyquist region. By multiplying $\hat{g}$ with a function $\hat{h} : C \rightarrow \{0, 1\}$ which is one inside the Nyquist region $[-\omega_N, \omega_N]$ and zero elsewhere $\hat{g}$ can be obtained from $\left(\hat{g} * \hat{k}\right)$.

$$\hat{h}(\omega) = \begin{cases} 1 & if \quad |\omega| \leq \omega_N \\ 0 & else \end{cases} \tag{2.2}$$

In the spatial domain this corresponds to a convolution of g_s with h, the inverse Fourier transform of $\hat{h}$.

[1] Diriclet Conditions:

- $\int |o(t)|\, dt < \infty$.
- $o(t)$ can be divided in a finite number of continuous sub-intervals.
- If t_0 is a point of discontinuity, $o(t_0 + 0)$ and $o(t_0 - 0)$ are existent.

[2] Only if these conditions are met, a fourier transform of the signal exists for sure. In that we have a finite object space, we can simulate the periodicity by repeating the signal. Diriclet conditions are always met when a realistic object is scanned.

[3] Shannon-Nyquist Sampling Theorem

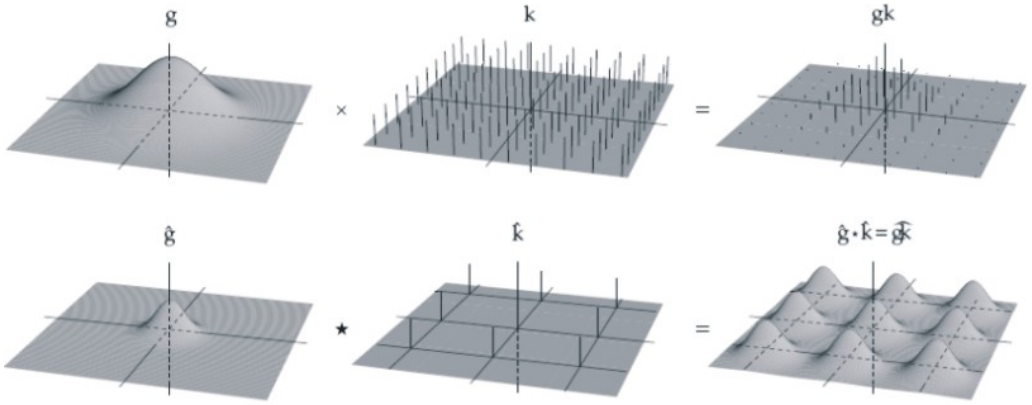


Figure 2.1: Illustration of the sampling and reconstruction of a 2D function in spatial (top) and frequency (bottom) domain.[35]

$$g(t) = g_s * h = \int_{\infty}^{\infty} g_s(s)h(t-s)ds \tag{2.3}$$

This means h is the optimal reconstruction filter. It becomes a *sinc* function in the spatial domain:

$$\begin{aligned} h(t) &= \frac{1}{\sqrt{1}} \int_{-\infty}^{\infty} \hat{h}(\omega) e^{jt\omega} d\omega = \int_{-\omega_N}^{\omega_N} e^{jt\omega} d\omega \\ &= \frac{1}{jt} \left[e^{j\omega_N t} - e^{-j\omega_N t} \right] = \frac{2\sin\omega_N t}{t} \\ &= 2\omega_N \mathrm{sinc}(\omega_N t) \end{aligned} \tag{2.4}$$

To consider the discrete representation of the equations above we let $g[n] : Z \rightarrow \Re$ be a digital signal of the discrete samples from g ($g[n] = g(fracnf_S)$). Further let $T = \frac{1}{f_S}$ be the sampling period. Then we derive the reconstructed function $g(t)$ (a "continuous-dicrete convolution") from Equation 2.3.

$$g(t) = \sum_{n=-\infty}^{\infty} g_s[n]h(t-nT) \tag{2.5}$$

Inherently, this model can not be implemented because it assumes a periodical, infinite input signal and a reconstruction filter with an infinie extend.

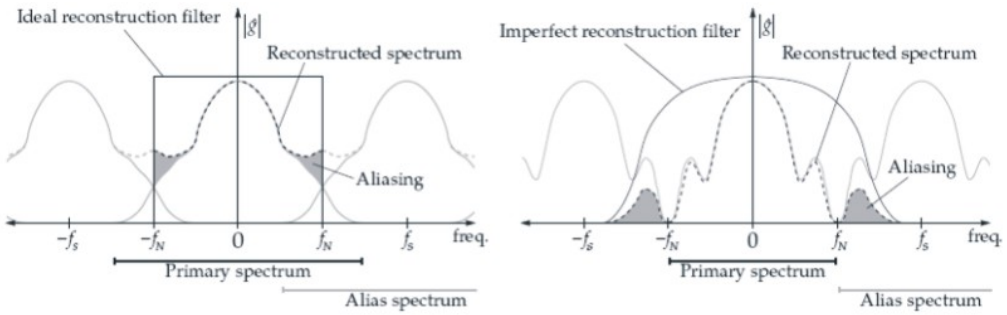


Figure 2.2: Pre-aliasing caused by a sampling frequency not fulfilling the Nyquist criterium (left). And Post-aliasing due to an imperfect reconstruction (right).[35]

On the other hand, especially in the case of 3D processing, the computational cost shall be kept low, and thus a small reconstruction filter radius is preferred. Thus h is only approximated poorly in many cases, particularly for 3D rendering. Due to an imperfect filter design, the frequency response inside the Nyquist region is not homogeneous, what causes smoothing when the high frequencies are suppressed or overshoot when high frequencies get amplified. Figure 2.2 illustrates that there are in fact two sources of aliasing artefacts: pre-aliasing because of a non-sufficient sampling frequency and post-aliasing due to an imperfect reconstruction filter.

2.2.2 Simple Reconstruction Filters

Although the process of reconstruction can be optimised according to the theoretical consideration above, typical DVR approaches use relatively simple reconstruction filters to avoid a substantial slow down of the finally implemented program. Commonly approved is the employment of a *next neighbour interpolation*, suited for fast renderings with low resolution, and *trilinear interpolation*, for high quality renderings, often with a sampling distance below voxels' size, in moderate time. The mathematical descriptions of these filters are:

Next Neighbour Interpolation:

$$f(x,y,z) = f\left[x_{nn}, y_{nn}, z_{nn}\right] \tag{2.6}$$

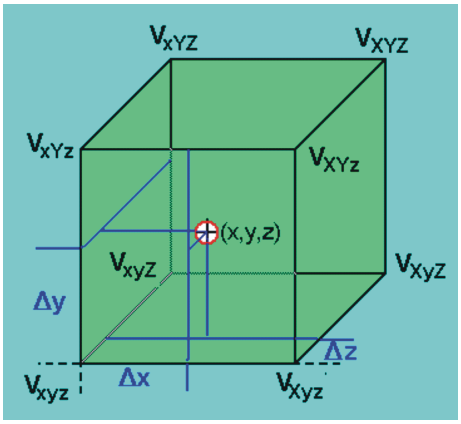
with:

$$x_{nn} = \begin{cases} \lfloor x \rfloor & \text{if } x - \lfloor x \rfloor \leq \lceil x \rceil - x \\ \lceil x \rceil & \text{else} \end{cases}$$

$$y_{nn} = \begin{cases} \lfloor y \rfloor & \text{if } y - \lfloor y \rfloor \leq \lceil y \rceil - y \\ \lceil y \rceil & \text{else} \end{cases}$$

$$x_{nn} = \begin{cases} \lfloor z \rfloor & \text{if } z - \lfloor z \rfloor \leq \lceil z \rceil - z \\ \lceil z \rceil & \text{else} \end{cases}$$

Trilinear Interpolation:



$$\begin{aligned} f(x,y,z) = {} & V_{xyz}(1-\Delta x)(1-\Delta y)(1-\Delta z) + \\ & V_{Xyz}\Delta x(1-\Delta y)(1-\Delta z) + \\ & V_{xYz}(1-\Delta x)y(1-\Delta z) + \\ & V_{xyZ}(1-\Delta x)(1-\Delta y)\Delta z + \\ & V_{XyZ}\Delta x(1-\Delta y)\Delta z + \\ & V_{xYZ}(1-\Delta x)\Delta y\Delta z + \\ & V_{XYz}\Delta x\Delta y(1-\Delta z) + \\ & V_{XYZ}\Delta x\Delta y\Delta z \end{aligned} \tag{2.7}$$

2.3 Shading

VR projects 3D-data to a 2D screen. To convey the observer of the image a three dimensional impression, a variety of depth cues can be provided by modeling the light reflectance behavior on the surfaces. This requires to have the normals of the surfaces reflecting the light. For VR in general, there exist two major approaches to obtain surface normals: image-space methods, which estimate them e.g. from gradients of the depth field of iso-surfaces (obtained from the z-buffer for instance) and object-space methods, which estimate them according to the 3D neighbourhood of a voxel [49]. As DVR does not depict clear surfaces, a common practice is to interpret the first derivatives as the contours' normals.
In order to choose an appropriate illumination model, one has to specify the limitations and requirements of the visualisation technique. In most cases, a visualisation of 3D scalar data shall preferably result in high informative images. E.g. DVR for CTA addresses to a visualisation of clinical relevant details. Endeavours to provide a photorealistic imaging becomes obsolete as far as complex realistic light interactions do not necessarily contribute to the value of (medical) information visualisation. A reduction to an essential complexity is also desired in order to keep the processing costs as low as possible. Thus most DVR approaches just apply a very simple shading model, which, nevertheless, provides a fully sufficient 3D impression for our purposes.

2.3.1 Derivatives of Discrete Volume Data

To simulate light interactions when rendering volumetric data, we need to estimate the unknown surface normals. Therefore we assume that gradients correspond to the normals of the depicted surface contours.
The gradient of a n-dimensional scalar field $\varphi(\vec{v})$ is defined as:

$$\nabla\varphi = \begin{pmatrix} \frac{\delta\varphi}{\delta v_1} \\ \vdots \\ \frac{\delta\varphi}{\delta v_n} \end{pmatrix} \tag{2.8}$$

In our case $\varphi(\vec{v})$ becomes the reconstruction $g = g_s * h$ of the discrete data volume g_s (see Section 2.2). This leads to:

$$\nabla(g_s * h) = \nabla g_s * h = g_s * \nabla h \tag{2.9}$$

The gradients are derivatives of a convolution, so we can also convolve them with derivative of one of the operands instead. Hence, there are three ways to obtain gradients. Naturally, we always have to deal with approximations of the gradients, because the optimal gradient filter is the first derivate of the three dimensional equivalent of the optimal reconstruction filter presented in Equation 2.2 and is thus a *cosc* function. Widely used gradients estimates are *Central Difference* and *Intermediate Difference*, which are interpolated afterwards with a reconstruction filter to obtain a continuous function throughout the volume. Actually this course of action corresponds to '$\nabla g_s * h$'. Alternative solutions will be discussed in Section 3.2.4. Below, a representative representation for one dimension (k) of these gradient estimates:

Intermediate Difference:

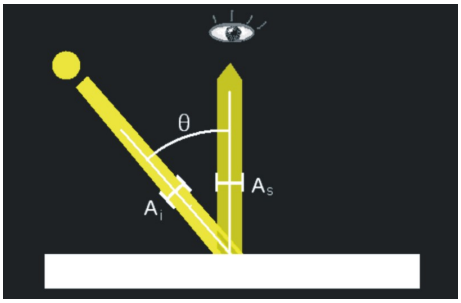
$$\frac{\delta\varphi}{\delta v_k}^{(i)} \approx \frac{v_k^{(i+1)} - v_k^{(i)}}{2} \tag{2.10}$$

Central Difference:

$$\frac{\delta\varphi}{\delta v_k}^{(i)} \approx \frac{v_k^{(i+1)} - v_k^{(i-1)}}{2} \tag{2.11}$$

2.3.2 Shading Model

Shading refers to the process of altering a surface's color with respect to its orientation towards the beams of light it is reflecting. A simple physical model, which assumes a uniform light reflection, is depicted in the left figure.



The amount of light energy being reflected to the observers direction corresponds to the proportion of the light beams cross-sectional area A_i and the area A_s, over which it is spread out when striking the surface. By virtue of simple trigonometric laws we get:

$$cos(\theta) = \frac{A_i}{A_s} \tag{2.12}$$

The light intensity I_s is altered by weighting the illumination I_0 with the cosine of θ, which equals to the dot product of the vector the light beam $\vec{l}$ and the surface normal $\vec{n}$.

$$I_s = I_0 cos(\theta) = I_0 \left(\vec{l} \cdot \vec{n}\right) \tag{2.13}$$

It is reasonable that this equations are only valid for $\theta \in \left[-\frac{\pi}{2}, +\frac{\pi}{2}\right]$, elsewise we get negative values. Usually the dot product is implemented within a maximum function ($max(\vec{l} \cdot \vec{n}, 0)$) to avoid this. But this means that antipodal surfaces have zero light reflection and appear due to the deficiencies of a underlying local illumination model as black regions. The addition of a slight ambient illumination is a simple trick, but common practice, to prevent this artificial impression. But for medical DVR this is not mandatory as the most valuable visualisation is obtained by defining the direction of illumination equal to the viewing direction. Some approaches (e.g.[38][39]) employ also a specular light reflection. But for the purposes of CTA this is inappropriate, as specular highlights are likely to be confused with calcifications.

2.4 Compositing

2.4.1 Optical Model

DVR approaches regard resampled values as volumetric particles having certain optical properties. This means each particle has a chromacity, an absorbance behavior and one or more reflectance properties, which together determine the amount and color of light being irradiated to (emission), or absorbed from and diffracted to (reflection or scattering) a certain direction. Thus all light irradiating towards a pixel has to be accumulated with respect to all particles scattering it. An entire model for light interaction and absorbance is given by Kajiya's Rendering Equation [54]:

$$I(x,x') = g(x,x')\left[\epsilon(x,x') + \int_S \rho(x,x',x'')I(x',x'')dx''\right] , \tag{2.14}$$

where $I(x,x')$ denotes the intensity of light reaching position x and being emitted at x'. The geometry term $g(x,x')$ is the visibility function and weights the intensity of irradiations from object x' towards object x. The emission from particle's x' own light being irradiated towards x is represented by $\epsilon(x,x')$. The scattering or reflection of light, passing from x'' to x through particle x', is expressed by $\rho(x,x',x'')$.

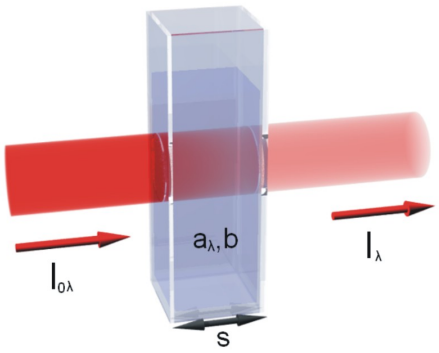
As this equation integrates the reflecting and scattering light on x' over the whole scene and recursively, there exists no analytical solution of it. An economical simplification of this model with respect to the most important properties of the particles, opacification and emission, is sufficient. Therefore we first reduce this global illumination model to a local model with a low albedo case:

To model a viewers light perception, let us assume that a pixel x is located on the screen of a rudimental camera obscura. The accumulation of all light contributing to this pixel x relates to an integration of $I(x,x')$ over all x'. Assuming the camera model, all visible x' ($g(x,x') > 0$) are striked by a ray which passes from pixel x through the cameras' pinhole. Thus the integra-

tion can be performed just in a one dimensional manner. Scattering effects which diffract light beams are discarded now. We further reduce the optical properties of each x' to an emission of light $I_\lambda(s)$ with frequency λ and an absorbance rate $\mu_\lambda(s)$ for light passing through particles at s coming from $s + \Delta s$ ($\Delta s > 0$). s is a scalar value and represents the position on the ray, and the emitted spectrum $I_\lambda(s)$ comprises all kinds of light reflections and emissions at s.

The Beer-Lambert-Bouguer law models in a physical manner the absorbance of a beam with a certain wavelength λ traversing a homogeneous volume of thickness s:

$$A_\lambda = -\ln \frac{I_\lambda}{I_{0\lambda}} = a_\lambda b s \Leftrightarrow I_\lambda = I_{0\lambda} e^{-A_\lambda} \tag{2.15}$$



The absorbance A_λ, which is the logarithmical ratio of incoming ($I_{0\lambda}$) and outgoing light energy (I_λ) with wavelength λ, equals to the product of the absorbtivity a_λ, the density of the absorbing substance b, and the distance s. As we can not differentiate density and absorbtivity of materials, the parameters a and b can be merged into one "absorbance rate" μ_λ. For a further simplification we assume a frequency independent absorbance and thus μ becomes unique for all λ. This simplification reduces spectral absorbance properties to a pure opacification. Further we have a heterogeneous volume and the opacity is a function of s. This means A becomes an integral of $\mu(s)$:

$$\begin{aligned} I_\lambda &= \lim_{\Delta s \to 0} I_{\lambda 0} \prod_{i=0}^{s/\Delta s} e^{-\mu(i\Delta s)\Delta s} \\ &= I_{\lambda 0} e^{-\int_0^s \mu(t)dt} \end{aligned} \tag{2.16}$$

The product shall illustrate the successive use of the Beer-Lamberts law for infinitesimal homogeneous subvolumes.
The emittance (= density of emission) of a particle at s is also modulated with its' opacity and thus the intensity of light being emitted can be expressed as: $I_\lambda(s) = \mu(s)C_\lambda(s)$. The chromacity $C_\lambda(s)$ was introduced to model the spectrum of light emissions and is usually implemented as RGB-model. To obtain the amount of light being emitted from a particle located at s which reaches the given pixel after being hazed by all particles in between, we have to multiply the emitted light $I_\lambda(s)$ with the translucency $e^{-\int_0^s \mu(t)dt}$. Finally we have to integrate all light emissions along the ray to obtain the composited light intensity at a given position x from (ray-)direction r. This leads to the volume density scattering model which was introduced by Blinn [55]:

$$I_\lambda(x, r) = \int_0^L C_\lambda(s)\mu(s)e^{-\int_0^s \mu(t)dt}ds \ , \tag{2.17}$$

where L denotes the distance along which the compositing is performed.
A numerical solution of this equation is given by the discrete compositing equation presented below.

2.4.2 Discrete Compositing

To derive a numerical solution of the volume density scattering model presented in Equation 2.17 one can reduce it to a series of discrete sampling positions s_i with a constant sampling density Δs. We assume that $C_\lambda(s_i)$ comprises the whole light emission energy and $\alpha(s_i) = \int_{\Delta s} \mu(s)ds$ the whole opacification of the particle's volume having a thickness of Δs. Then both integrals can be replaced by sums (compare to Equation 2.16). For simplicity we further assume $\Delta s = 1$ and generalise it for $\Delta s \neq 1$ later. Since the argument of exponential function then becomes a sum, we can transform the exponential function into the product $\prod_{j=0}^{i-1} e^{-\alpha(s_j)}$. If $e^{-\alpha(s_j)}$ is finally approximated by the first two terms of its' Taylor series, we get the following equation, which is well known as as the discrete compositing equation:

$$I_\lambda(x,r) = \sum_{i=0}^{L/\Delta s} C_\lambda(s_i)\alpha(s_i) \prod_{j=0}^{i-1} (1-\alpha(s_j)) \tag{2.18}$$

In order to implement this equation as an iterative algorithm, we need to analyse the progression of the sum. Therefor let $I_\lambda^{(k,N)}$ represent the compositing of all light emissions between the sampling steps $k \leq N$ and $N = L/\Delta s$:

$$I_\lambda^{(k,N)} = \sum_{i=k}^{N} C_\lambda(s_i)\alpha(s_i) \prod_{j=k}^{i-1} (1-\alpha(s_j))$$

If we process in back-to-front order we need the induction from $I_\lambda^{(k,N)}$ to $I_\lambda^{(k-1,N)}$:

$$I_\lambda^{(k-1,N)} = I_\lambda^{(k,N)} (1-\alpha(s_{k-1})) + C_\lambda(s_{k-1})\alpha(s_{k-1}) \tag{2.19}$$

When compositing in back-to-font direction, only the intensities have to be accumulated. This is done by blending over successively all $C_\lambda(s)$ with the according opacities.

A particle can not emit any light to the observers eye if its' light is absorbed by obscuring particles completely. According to the compositing equation this does actually not happen if all voxels are classified with an opacity value below one, which is a common case. But even when the translucency $\prod_{i=0}^{k} (1-\alpha(s_i))$ just approaches zero, compositing does not significantly change the values. In other words, further particles behind k need not to be sampled or can be sampled with reduced precision, because they are less or not perceptible. Hence, if processing is done in a front-to-back order, one can save processing time by an early ray termination.

For processing in front-to-back order we need the induction from $I_\lambda^{(0,k)}$ to $I_\lambda^{(0,k+1)}$:

$$\begin{aligned} I_\lambda^{(0,k+1)} &= I_\lambda^{(0,k)} + C_\lambda(s_{k+1})\alpha(s_{k+1}) \prod_{i=0}^{k} (1 - \alpha(s_i)) \\ &= I_\lambda^{(0,k)} + C_\lambda(s_{k+1})\alpha(s_{k+1}) \left(1 - \alpha_{acc}^{(k)}\right) \end{aligned}$$

where:

$$\begin{aligned} \alpha_{acc}^{(k+1)} &= \sum_{i=0}^{k+1} \alpha(s_i) \prod_{j=0}^{i-1} (1 - \alpha(s_j)) \\ &= \alpha_{acc}^{(k)} + \alpha(s_{k+1}) \left(1 - \alpha_{acc}^{(k)}\right) \end{aligned} \tag{2.20}$$

The term α_{acc} denotes the accumulated opacity. This transformation seems circuitous at first glance since it can be represented quite simpler [4]. But, if one regards the product $\alpha(s_{k+1})(1-\alpha_{acc}^{(k)})$, one can see that it appears in both formulas, the iterative compositing equation and the opacity accumulation. Thus this representation of the discrete compositing equation is the most efficient front-to-back calculation rule, because it can be implemented with two multiplications only.

[4] A straightforwardly derived rule, which needs at least three multiplications, would be:

$$\begin{aligned} I_\lambda^{(0,k+1)} &= I_\lambda^{(0,k)} + C_\lambda(s_{k+1})\alpha(s_{k+1})\rho^{(k)} \\ \rho^{(k+1)} &= \rho^{(k)} (1 - \alpha(s_k)) \end{aligned}$$

Since rendering is performed with different object sampling distances, we need a normalisation to avoid biases in the opacification. Therefor we regard the Taylor approximation of the exponential function for discrete opacity accumulation and generalise it by introducing the sampling distance Δs:

$$e^{-\alpha \Delta s} \approx (1 - \alpha)^{\Delta s}$$

If we sample with a $\Delta s \neq 1$, we need to adjust α accordingly:

$$\begin{aligned} (1 - \alpha_{corr}) &= (1 - \alpha)^{\Delta s} \\ \Leftrightarrow \alpha_{corr} &= 1 - (1 - \alpha)^{\Delta s} \end{aligned} \tag{2.21}$$

Chapter 3

Technical Solutions for DVR

DVR is very costly and practical applications tend to be time critical. For CTA of peripheral vessels, we have to deal with large datasets (about 0.7 GB in voxel representation), and the cost of DVR increases linearly with the amount of input data ($O(n)$) and cubical with the resampling frequency ($O\left(f_i^2 f_o\right)$, with image sampling frequency $f_i = \frac{1}{\Delta s_i}$ and object sampling frequency $f_o = \frac{1}{\Delta s_o}$). By reason of that, one major goal when designing DVR software is efficiency, which can be improved with several acceleration strategies.

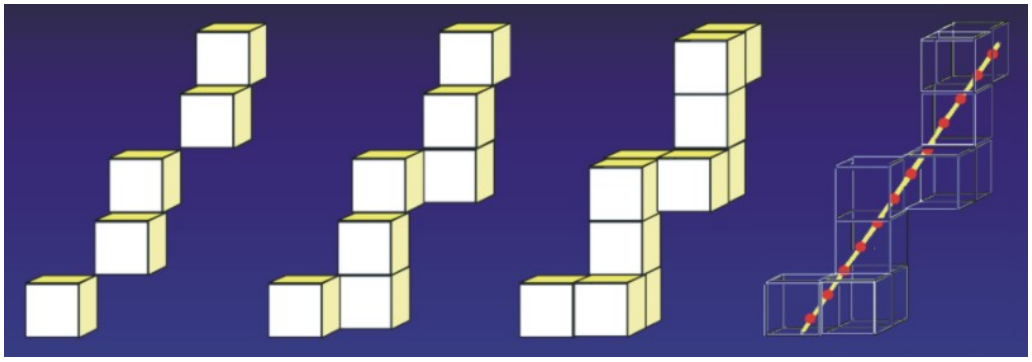
The purpose of this chapter is to give a survey on the state-of-the-art of DVR and to discuss several approaches with respect to our implementation. Thus the content will progress from general to specific solutions. Techniques which are employed, seem to be adequate or shall be at least considered for our challenges are discussed in detail.

3.1 Major Approaches

The following approaches emerged as the most popular solutions for designing DVR algorithms:

3.1.1 Raycasting

Raycasting is a simplified raytracing strategy and was applied in the first DVR approaches. But due to its' advantages, it is still a widely used technique (for example [11, 36, 45, 20]). Raycasting is an image order approach, what means that only those voxels which contribute to a given pixel are resampled and composited. A raycasting algorithm samples and composites along a ray shot through the volume with the direction determined by the positions of the observers eye and the given pixel on the display. This can be done either by following an analytical line and sampling consecutive positions on this line, or traversing and sampling the grid following a path of connected voxels. Those paths can be created following either 6-,18- or 26-neighbourhood connections between voxels (see Figure 3.1). The most conservative and costly among those is a 6-line traversal, which samples actually each voxel being pierced by the ray. However, discrete ray-traversal can avoid aliasing when no voxels being pierced by the ray are leaped (6-line), but analytical ray-traversal can be performed with a consistent sampling rate and its' costs do not vary with the viewing direction.
Raycasting allows to composite in front-to-back order and to apply early ray termination when additional samples will not significantly change the corresponding pixel's value. Further strategies to accelerate raycasting will be described in section 3.3). However, raycasting is a kind of brute force strategy and the quintessential algorithm is not efficiency optimal. When sampling in a particular cell, one has to access all corner voxels inside the radius of the reconstruction filter, and their gradient estimates have to be computed respectively. Thus each voxel being relevant to the output-image is often processed multiple times, especially when the sampling distance is



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

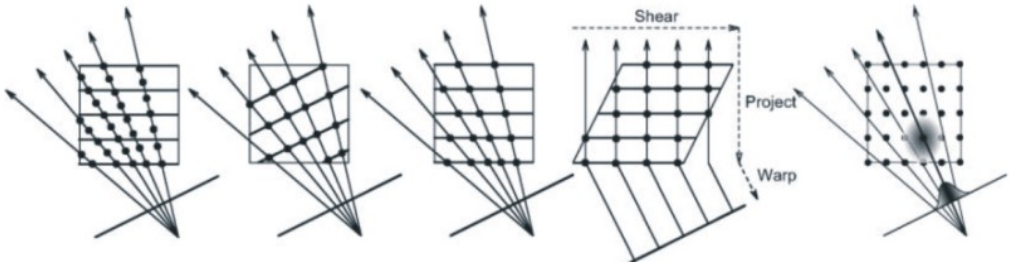
Figure 3.1: Ray-traversal techniques: (a) 26-path, (b) 8-path and (c) 6-path traversal. (d) Sampling along an analytic line.

below the voxels' size. Due to sequential raycasting, this operations suffer from reduced temporal and spatial locality, what is a general drawback of image order approaches.
The main advantage of the ray-casting approach is that it enables to reach maximum quality, because it strictly follows the basic physical model of light transport. It provides also a high flexibility by reason of its' straightness and simplicity. Functionality extensions can be integrated easily in the raycasting pipeline, what will be utilised in our case for focus and context rendering (Chapter 5).

3.1.2 Splatting

Rather than sampling in image order by raycasting, one can also sample in object order by data related process traversal. This can be done by iterating through the data and projecting and compositing each sample to the pixels of the output image. This approach allows to process the data in storage order and gains that way speed by reducing the computational overhead. In contrast to image order processing, the sampling positions are not related to the pixels. Thus one has to determine the screen space contribution of

each sample to the output image. Westover [57] presented a method which integrates a chosen reconstruction kernel (commonly Gaussian approximations) along viewing direction to obtain a 2D footprint function, defining the contribution of a certain sample to each pixel (see Figure 3.2(e)). Intuitively, this can be considered as "separated convolutions" which are performed for each voxel independently and "merged" afterwards. For orthographical projections, the footprint function remains constant for all samples in a certain view and can thus be pre-computed as a discrete 2D look-up table. While processing a certain sample, the footprint is weighted with the voxels values (RGBA) and mapped onto the image screen. One major advantage of this method is that each voxel is accessed only once and it has to be splatted only if its relevant to the image. With respect to this fact, one can enormously reduce the amount of data being processed and stored. Early splat elimination is an acceleration strategy presented by Mueller et. al. in [63], which corresponds to early ray termination. But this method requires still a projection transformation to determine if a voxel is visible. Splatting is an efficient approach, but the pre-integration of the reconstruction kernel causes inaccuracies. The extend of the kernel is composited as a whole and this results in a violated compositing order, which becomes apparent as colour bleeding artifacts. This problem was addressed by accumulating footprints from all voxels belonging to a slice being aligned with the most image-plane-parallel volume face in a sheet-buffer, before mapping and compositing them on the image plane. But since the slices of the sheet-buffer are not exactly parallel to the image plane and each voxel spreads its' entire energy only within one slice instead to several, the discrete compositing equation is still violated. A non-image-parallel sheet buffer causes also severe brightness variations in some viewpoint transitions (when the sheet-buffer switches between two slices), what is undesirable for interactive viewing. A solution to this problem is image-aligned splatting, which presented Mueller et. al. later in [64]. This algorithm processes data in slabs being parallel to the image plane. Voxel kernels overlapping the current slab are clipped and the clipped energy is



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

Figure 3.2: Illustration of popular DVR methods: the (a) ray-casting,(b) 2D-texture mapping,(c) 3D-texture slicing,(d) shear-warp and (e) splatting approaches.[68]

saved by adding it to the sheet-buffer of consecutive slabs.

3.1.3 Object-order Raycasting: Shear and Warp

The principle idea of the shear and warp approach is maintaining speed gain of object order algorithms by processing in storage order, while employing an ray-casting like compositing scheme at the same time, what allows to apply early ray termination, which is a very potential acceleration strategy. The essential shear and warp transformation is based on a factorisation of the viewing matrix into a 3D shear parallel to the slices of the volume data, a projection to form a distorted intermediate image, and a 2D warp to produce the final image [14] (see Figure 3.2(d)). Lacroute and Levoy [14] presented an implementation of this hybrid order approach, which applies additional run-length encodings of transparent space in rows in the data volume and in intermediate images. As rows in the intermediate image are aligned parallel to the rows of the data volume, they can be processed in synchrony, taking the advantages of spatial coherence present in both. The shear and warp approach has been considered as one of the fastest DVR strategies, but its' straight-forward implementation does not allow to apply 3D reconstruction filters and the sampling rates vary with the view direction. Thus it can not keep up with the quality achieved by conventional raycasting algorithms.

3.1.4 Texture Mapping

The principle of the shear and warp factorisation can be also applied by utilising texture compositing capabilities of graphics hardware. Texture mapping stands for the process of projecting the volume content as textures on representative planes, represented e.g. by polygons, and composite them with commodity hardware functionality. As this functionality is provided in every nowadays common graphics hardware, it was employed to involve GPU processing into the volume rendering pipeline. With 2D and 3D texture mapping there exist two different approaches, which both decompose a volume graphic to a stack of texture slices, being rendered like surface graphics by compositing with alpha blending in a back-to-front order. 2D texture mapping approaches ([19, 15]) pass a stack of object aligned 2D-textures to the graphics hardware. Therefor a stack of object aligned slices for each of the three major viewing directions is stored. While compositing, the textures are interpolated bilinearily. As this method corresponds to a shear-warp factorisation, it suffers from the same drawbacks in sampling and reconstruction. An improvement was achieved with the multitexture approach from Salama et al. [15]. To extend reconstruction to the third dimension, they implemented an on-the-fly interpolation between subsequent slices with pure graphics hardware functionality [1]. Actually, this is a hybrid solution between the 2D and 3D texture mapping approaches. In contrast to 2D textures, 3D texture mapping methods ([16, 19, 17, 18]) sample and align the textured planes parallel to the image plane, and for each single view direction the textures have to be rebuild. These methods pass the volume as 3D texture to the graphics hardware. Image aligned polygons with 2D textures are then obtained by sampling with trilinear interpolation and consistent rate by hardware. However, this approaches are dependent on the size of video memory and are suitable for medium sized datasets only. Decomposing the

[1] They applied alpha blending to substitute the linear interpolation operations. They also utilised the GPU to perform shading. Therefor they proposed to ”abuse” RGB-datatypes to encode gradient vectors.

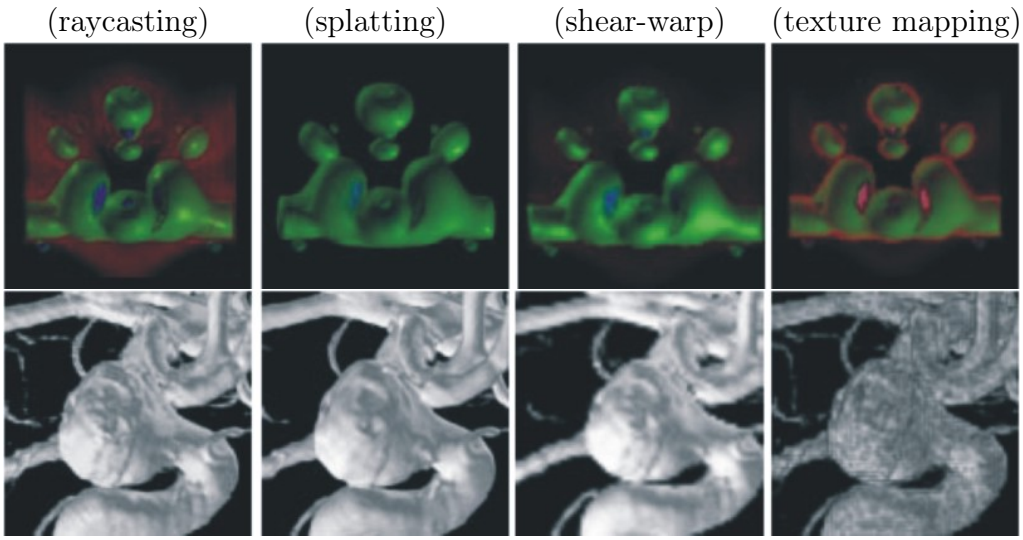


Figure 3.3: A comparison of popular DVR algorithms was presented by Meißner et al. [13]. Two different targets were rendered with different algorithms. Raycasting handles well with colours (top: neghip), while splatting conveys very smooth surfaces - even though with some loss of detail (bottom: details of a blood vessel in head). Both other techniques do obviously not keep up to their quality.

data leads to significant speed penalties due to an increased data traffic over a bandwidth limited databus, which becomes a main bottleneck.

3.1.5 Utilising GPU

Hardware based DVR approaches progress in two ways. On the one hand new functionality being introduced into hardware for general purposes can be utilised for specific tasks of the DVR pipeline. This results in a hardware determined software development. But as far as the development of commodity graphics hardware was primarily focused on surface graphics, as used in computer games, their functionality has to be utilised in a less conventional manner. On the other hand proven innovations for DVR, being investigated in former software approaches, are reimplemented on hardware basis to provide them as further functionality on the GPU. This refers to a special purpose hardware development. A comparison of special purpose

DVR hardware was presented in [65]. Nowadays special purpose hardware provides DVR with framerates that never can be achieved with pure software solutions. In contrast to approaches utilising general purpose hardware (e.g. texture mapping approaches), they also assure a high-quality standard, because their functionality and architecture is adequate for volume processing. Today, even volume raycasting approaches are implemented on hardware. The major drawback of software that utilises special purpose hardware is its' dependence on the workstations equipment. Especially for multi-functional applications like the AngioVis toolbox, which shall be portable to different workstations and operating systems, a hardware independent performance is advantageous.

3.2 Re-sampling and Classification

3.2.1 Re-sampling Pipelines

Obtaining visual properties at a particular point in the volume involves three main operations which have to be consired for choosing a suitable order:

- **Classification**- to receive chromacity and opacity information from density values.
- **Shading**- to assign the illumination properties.
- **Reconstruction**- to retrieve a continuous function from discrete values.

Actually there are six permutations of these operations. To perform shading, that is altering the values according to the surface normal, before classification is incorrect. Hence, there are only three orders left which are acceptable. Under presence of sharp transitions in the data, a classification before reconstruction seems more stable than vice versa. This has two main reasons: the partial volume effect, which only occurs in a post-classification,

and further reconstruction errors, which are passed to the classification procedure and are not predictable then. Thus if one focus on quality only, the pre-classification approach might be the better choice [2]. But this approach requires on the one hand an intermediate representation of the data as classified values, being re-sampled afterwards. On the other hand, re-sampling involves the reconstruction of four channels (opacity weighted RGB and α) and is thus significantly more costly. If shading is also performed before reconstruction, one can omit the costly gradient reconstruction since the shaded chromacities are interpolated directly. But this results in a loss of quality when poor gradient estimates are used (e.g. intermediate difference). But finally, the pipeline we choosed in our implementation is: reconstruction - classification - shading.

3.2.2 Transfer Function Design

Since the design of a TF with desired results often tends to be a frustrating trial and error process, there exist various approaches aiming to avoid these drawbacks. To increase usability one has to make TF space less confusing by restricting excessive flexibility. Further support of the user can be provided by valuable information, guidance, semi-automation or automation. TF design has even become an own field of research in recent years. Major approaches for TF design can be summarised under the following terms:

Trial and Error: This means to edit the TF manually and enforces to learn by experience. This method should only be used by the experienced developer of the rendering software. Due to simplicity, using this approach can save implementation efforts, but only if sufficient TFs can be found in time. Hence it makes sense to use this approach for testing prototypes of the rendering

[2] But it has to be implemented correctly. Wittenbrink et. al. [62] approved that it is incorrect to reconstruct voxel colors and opacities separately and then compute the product $C(s)\alpha(s)$. Instead, one must weigh color with opacity at each voxel before the reconstruction.

software, or under the assumption that all input datasets can be rendered sufficiently with a small set of predefined TFs. The AngioVis-toolbox uses preprocessed and density-equalised datasets. So different datasets can be rendered with the one TF and a similar appearance can be expected. To compensate slight variations, one can introduce a restricted interactive adjustment by a global opacity modulator or an adjustable offset shift the whole opacity TF slightly forward or backward (in the density domain).

Spatial Feature Detection: This refers to a multidimensional TF design that includes spatial features of the image as additional determinants. Those approaches (e.g. [23, 27, 24, 25]) extend the conventional TF model to a multi-dimensional one, including spatial features, which have to be extracted by a set of 3D processing tools. Such features include amongst others gradients, curvatures, silhouettes or halos. This makes the classification very flexible, as surfaces can be detected more precisely and it is less dependent to the according density distributions. Overall, these approaches ease the search for a TF with desired results, but they introduce expensive 3D-processing into the rendering pipeline. This problem can be addressed by outsourcing the feature extraction to offline processing and storing the feature information in additional bits in the voxel data. Generally regarded, involving segmentation information into the classification corresponds to this approach, as well as focus and context rendering by the VesselGlyph (see Chapter 5).

Image-centric Search: The process of TF design can be also regarded as a search in a large multidimensional parameter space. Thus the designer can be supported by guiding, semi-automated or automated searching methods. Parameter searching problems are well known from many scientific and technical disciplines, where they occur in many cases as cost-optimizing parameter tweaking tasks and various algorithms for automation or clustering have been developed (Self Organizing Maps, Genetic Algorithms,etc.). Since all searching tactics require a kind of cost-function, each result of a selected or proposed TF-design has to be evaluated. This can be done by a manual

assessment of the renderings by the user (supervised), or automatically by a cost-function, evaluating some features of the rendered images (unsupervised). However, manual assessment is indispensable, at least to make the final selection of adequate TFs. It can be involved by displaying preselected and clustered design galleries[22], judging thumbnail renderings between the iterations of a TFs breeding genetic algorithm [21], or exploration, guided by a visual history of renderings with logically connected varying TF settings [28]. Unsupervised assessment, which is used in the automated selection steps of the search algorithm, can evaluate objective features as entropy, energy or variance.

Data-centric Analysis: TFs can be also specified with respect to the data itself. A representative data-centric approach was presented by Kindlmann et al. [26], which is based on a surface detection by analysing binned 3D-histograms of densities, their first and second order derivatives in direction of maximum gradients. Instead detecting contours in the spatial domain, they detect them in the range of data values. The employed histograms are used to find characteristics of significant boundaries being present in the data volume. The zero crossing of the second derivative corresponds to maximum of the first derivative, and this corresponds to an edge in the density grid (see Figure 3.4). The main idea of this approach is to relate the densities, first and second derivatives with each other, and thus be able to project out spatial information. The bins being related to boundaries in the data can be automatically identified by a tool, detecting contour-characteristical curves of the three functions. After identifying characteristical bins for major boundaries, one can visualise samples within contour transitions by classifying them with respect to the bins they belong to. This method benefits from an automated identification of salient contours with an empirical detection method.

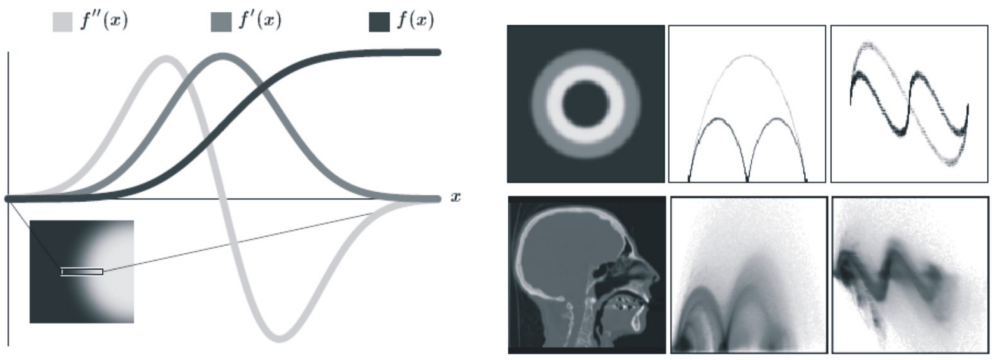


Figure 3.4: Left: An analytical consideration of a boundary transition. Right: Scatterplots of the densities versus the first derivatives and first derivatives versus the second derivatives. [26]

3.2.3 Reconstruction Filters

Reconstruction filters are applied for re-sampling and gradient estimation. For gradient estimation the reconstruction is extended to three channels. As mentioned in Section 2.2.1, the quality of reconstruction depends on the similarity of the employed filter to the optimal reconstruction filter. But under practical consideration, efficiency is in many cases of major importance and reconstruction imperfections are readily accepted. Next neighbour interpolation (Equation 2.6) is a useful low-cost solution and it is by far the most efficient approach. It can be applied for rendering with moderate quality in short time (in our implementation 0.7 - 2 seconds), required for interactive use. According to the cubical cost-function, rendering can be speeded up enormously by increasing the sampling distance. If it is mandatory to sample above voxels' size, a smoothing reconstruction filter is needed to reduce alias-artifacts. In Section 4.4 we briefly describe an efficient solution, which uses the densities being accessed for gradient estimation by central difference, anyway, for computing the mean value of the nearest neighbour voxel and its neighbours in x-,y- and z-direction. Obviously coarse filters are not suitable for reconstructing when sampling with sub-voxel accuracy. A common filter method for this challenge is trilinear interpolation (Equation 2.7), which is

used in almost all DRV approaches referenced in the bibliography of this work. When large datasets are rendered, one can assume a high definition of the object space in relation to the image resolution. Hence when using trilinear interpolation together with sampling distances below the voxels' size, one obtains high-quality images being almost optimal, although it is only a first order interpolation filter with radius one. Image quality suffers also less from poor reconstruction when transitions from transparent to opaque samples are smooth. Assuming a smooth classification (TF) one can expect smooth contour transitions for all data sets obtained by alias-frequencies suppressing sampling. However, high-quality rendering of datasets with sharp contures and high frequencies, or even strong zooming operations, necessitate the use of higher order filters to suppress artifacts, becoming apparent under those circumstances. Superior reconstruction filters were assessed by Marschner et al. [35]. They examined separable and spherical symmetric filters with respect to smoothing, post-aliasing and overshoot (q.v. 2.2.1). Therefor they employed a circularly spreading 2D sine-wave as test signal to be rendered (see Figure 3.5). In comparison to more expensive filters like b-spline interpolation or a windowed sinc function, the performance of a trilinear interpolation on this target was poor due to a high level of post-aliasing and smoothing. Superior reconstruction filters require a larger filter radius, what raises the costs for data-access tremendously. But, the chosen test signal, whose energy lies near the Nyquist frequency, represents a too artificial and challenging case in comparison to CT-Angiographic datasets. Furthermore, they assessed the performance with an isosurface reconstruction, which has of course no smooth contour transitions and amplifies the visibility of reconstruction artifacts in the rendered images.

3.2.4 Gradient Estimation

Möller et al. [48] states the quality of gradient estimation a significantly higher impact on the image quality than the reconstruction of the density function. Due to a 2D-projection, the spatial aberrations of an imperfect

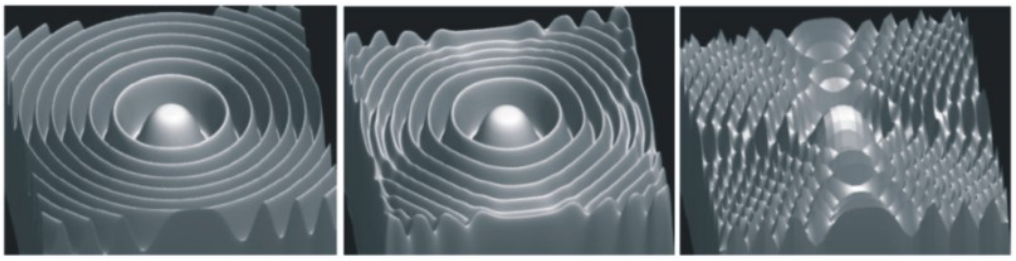


Figure 3.5: Evaluation of reconstruction filters on a target signal (left). The function was first sampled and than reconstructed with a windowed sinc function (center) and a trilinear interpolation (right). [35]

density reconstruction are less visible. But when visualising depth cues by modeling light reflections, aberrations in each dimension become apparent. Those reflectance imperfections are caused by aberrations of the surface normals and get in sight as starecase artefacts, for instance. As the surface normals are represented by gradients estimates, the quality of gradient estimation has a major impact on the 3D-impression of a rendered image.
The process of obtaining gradients from a density grid involves two steps: the reconstruction and the derivation. Referring to equation 2.9 this can be performed in three different manners. Möller et al. examined different approaches for gradient estimation and categorised them the following way:

- **Reconstruction First (RF)**- derivation of reconstructed densities in the neighbourhood of the current sample : $\nabla(g_s * h)$
- **Derivative First (DF)**- reconstruction of the derivated density grid : $\nabla g_s * h$
- **Continuous Derivative (CD)**and **Analytic Derivative (AD)** - both convolve g_s with the derivative of the reconstruction filter : $g_s * \nabla h$

The last approach ($g_s * \nabla h$) can be implemented by creating a filter by either convolving a digital derivative filter with a continuous reconstruction filter (CD) or an analytical derivation of a reconstruction filter (AD). They ([48]) approved that each implementation of the DF, RF and CD approaches

is numerical equivalent when the same reconstruction filter and derivative filter is used. For the AD method was shown that it is the most efficient one, but it has a questionable accuracy. For instance, when using filters which are not at least two times differentiable, like the trilinear interpolation. It was also shown that, with respect to the quantity of samples per cell, the DF approach is increasingly less costly than the RF method if the DF algorithm is optimised by taking the chance of caching gradients of grid points being computed already ([48][36]). The caching mechanism is only avaible for the DF approach. Assuming effective savings by gradient caching, the CD approach, which has an increased filter radius due to the convolution, is also less efficient.

Popular gradient estimation methods are intermediate difference (Equation 2.10) and central difference (Equation 2.11), which both are equally costly, but differ in their radii. The first has more specificity, due to involving the density of current voxel, and the latter is better balanced, because of a larger extend and its' symmetry. Central difference derivatives cause less staircase artifacts, but require data access in a larger radius. However, both approaches are designed under efficiency criteria and have a relatively small radius, what makes them prone to inaccuracies and unstable in the presence of strong noise. Sophisticated gradient estimates tend to be multi times more expensive due to large radii and many mathematical operations. An economical high quality gradient estimation method was presented by Neumann et al. [49], which retrieves gradients from computing a linear regression of the densities of the voxels 26-neighbourhood. Since the coefficients of a linear regression correspond to the derivatives of the regression function (in 1D: $\frac{\delta(Ax+c)}{\delta x} = A$), gradients correspond to the normal of the 3D regression plane. Moreover, using this method yields also a reconstructed density, which is given by the constant value of the regression function. The advantage of this method is that it can be implemented with cheap convolutions and significantly reduces staircase artifacts, even when sampling with a large sampling distance. But still a major drawback is the large radius (26-neighbourhood,

involves with trilinear interpolation a 63-neighbourhood), which makes this method very costly in a sparse memory layout.

3.3 Accelerating Raycasting of Large Datasets

The AngioVis project concerns CTA from abdomen to entire legs. For instance, 512x512x1200 voxels is a typical size of the density grid according to angiographic datasets. Rendering datasets in this magnitude challenges developers to find appropriate techniques to achieve acceptable speed.

3.3.1 Bricked Memory Layout

In general, the input of a volume visualisation system is an array of voxel values. With rising amount of data, the memory access to such a representation becomes increasingly costly. To accelerate the memory access on a large data-set, a decomposition in a bricked layout is a common practice (e.g. [53, 37, 36]) - also for other volume visualisation methods ([50, 51]). A promising solution is a data division scheme which supports a cache coherent processing. Grimm and Bruckner [37, 36] examined a cache coherent processing and data addressing scheme for DVR by raycasting. They aimed to address and process the data in such manner that the data being currently processed remains in the top level CPU-caches, what yields to a high cache hits rate. Going up the cache hierarchy of a CPU, the caches get smaller but faster. Hence, cache coherent processing becomes a gainful acceleration technique. It is realised via processing with temporal and spatial locality. Temporal and spatial coherency can be maintained by loading data as small bricks and rendering them separately by processing each brick at once. To assure a processing order which does not violate the compositing order (front-to-back traversal), they proposed to distinguish eight different block traversal schemes for eight major viewing directions. After loading once, one bricks' data shall remain in the (level 2) CPU-cache during whole processing time. However, the assessment of the optimal brick-size leads to

a trade-off between cache coherency and an increasing overhead due to separated processing. To evaluate the optimal brick size, they tested the speed up when bricking the data with several sizes [3]. With a speed-up factor of 2.8 a brick-size of $64KByte$, what corresponds to a side length of 32 voxels, gave the best performance. A main problem of a bricked data layout is sampling cells at the bricks boundaries, because the reconstruction and gradient filters necessitate to access voxels of neighboring bricks. This problem can be solved by introducing an overlap and store the brick borders' neighbour voxels redundantly, like it is done by our loading system (see Section 4.2.1). They (Grimm and Bruckner) also proposed to load the whole data in one array in a brick-by-brick storage order. For this representation one can reduce addressing times by an efficient mechanism, because voxels in neighbouring blocks can be addressed with a systematic selection from a small set of constant offsets. If the brick sizes are a power of two, costly address computing operations (modulo,division,multiplication) can be accelerated by the use of bitshifts and bitwise logical operations. An efficient strategy to exploit the capabilities of neighbour addressing by constant offsets from look-up-tables was also presented in this work ([37]).

3.3.2 On-the-fly Processing

To avoid redundant operations, a lot of processing can be done offline. The classification and the gradient estimation are independent of any manipulation of view direction and sampling distance. But precomputing this information introduces additional data and is not suitable for applications to render large datasets, since memory consumption is a critical factor anyway. When processing a bricked memory layout, computing all gradients of one brick only in one pass might be reasonable. But as far as most voxels do not contribute to the image anyway, an on-the-fly execution of the whole sampling pipeline was proposed to be the most efficient solution [36]. When sampling with sub voxel accuracy, the amount of redundant operations increases enor-

[3] Their system did only support brick dimensions being a power of two

mously with such strategy. Each sampling operation using a first or higher order reconstruction filter necessitate to access gradients of all voxels within the filter radius. Those redundant operations can be reduced effectively by caching on brick level. Grimm et al. [36] recommended to apply caching for gradient estimates and also to mark cells being classified as transparent already. Both caches have the same length as the array containing data of one brick, and they are used to cache the gradient, or rather visibility information for those voxels being accessed. When the current block is processed, the caches are initialized and reused for processing the next block. They stated that, assuming a modest block size (they choose actually 32), the overhead of memory consumption does not significantly interfere cache coherent processing on nowadays commodity hardware. Especially the cell transparency cache is very cheap, since it needs only one bit per cell. The gradient cache is more costly, but the memory management can be improved by using an additional binary cache to mark gradients being already computed.

3.3.3 Parallelisation

DVR is easily parallelisable because the results for all pixels in the output image can be obtained independently from each other. Parallelisation can increase the performance in two different ways: processing parallel instructions on multiple CPUs and parallelising multiple instructions for one CPU by exploiting its' capacities of simultaneous operations per cycle. Reasonably, the first approach gains speed-up only if the software runs on a multi-processor system or network. The latter technique refers to the instruction level parallelism of modern CPUs, having increasing operation capacities per cycle, which enables them to process independent instructions in parallel. Instructions which can be processed simultaneously can be issued on the basis of thread level parallelism (TLP), with so called hyper-threading. That is to represent one physical CPU as multiple logical units, receiving threaded instruction sequences [4]. Hyper-threads are actually processed simultaneously

[4]Further explanation of hyper-threading can be found in [60]

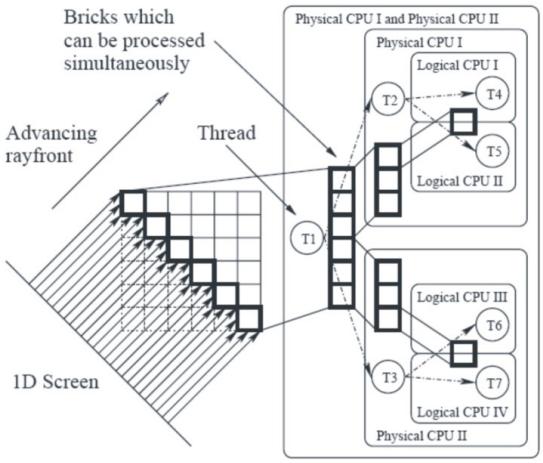


Figure 3.6: A Ray-casting system exploiting thread level parallelism and multiprocessor parallelisation. [37]

on one (physical) CPU and thus share the same caches. This is the essential difference to threading on a multiprocessor architecture and has to be taken into account when parallelising DVR with hyper-threading. When processing a bricked volume layout, both threading methods can be applied adequately. Grimm et al. [37] proposed to parallelise processing of a single block with hyper-threading and to parallelise the processing of different blocks on multiple physical processors (see Figure 3.6). By that all hyper-threads of one physical CPU access the data of the same block and the shared caches are utilised optimally. Moreover, intermediate results, like gradients, can be shared by caching. Threading on parallel hardware involves mainly the management of distributed processing and load balancing. Various approaches to manage parallelisation of VR on multi-processor architectures can be e.g. found in [20, 61]. According to a block wise rendering pipeline, a load balanced block-processing traversal scheme to avoid interferences and mistakes in the parallelized ray-traversal has to be involved. Those problems are due to the fact that each ray strikes several blocks and hence the block wise parallelisation must not violate the front-to-back compositing equation.

3.3.4 Early Ray Termination

If a back-to-front raytracing approach, early ray termination can be applied when the opacification reaches a value of one. Due to the front-to-back compositing equation (2.20) the value one is just asymptotical when each sample has an opacity $\alpha < 1$. This applies usually for common TF designs. Nevertheless, the opacity will reach soon a value close to one and thus a threshold slightly below one (e.g. 0.95) can avoid senseless compositing of irrelevant samples, but this solution is not correct in a strict sense and introduces a bias. An interesting method to compensate this bias would be the *Russian Roulette* strategy, originally proposed in [44] for Monte-Carlo particle-tracing methods, which terminates the rays with a probability equal to the accumulated opacity [40]. But this method is not a real remedy, as it just transforms the bias into noise.

3.3.5 Empty Space Leaping

The quantity of opaque voxels in CTA data is relatively small compared to the total amount of voxels. For a reasonable TF, one can readily expect 90 percent of transparent space. This means a large fraction of data does not contribute to the final image and can be skipped while processing. A straightforward and self-evident strategy is to leap single transparent voxels. This can be further optimised by cell transparency caching as proposed in Section 3.3.2. But since transparent or opaque voxels are in most cases gathered in coherent regions, gainful acceleration strategies base on estimating and localising opaque regions in the dataset in advance. There are two categories of empty space leaping approaches:

Volume Subdivision:

Volumes can be subdivided in two ways: fixed sized regular subdivision (bricking) and hierarchical subdivision. The first attempt corresponds to a bricked memory layout and does not adapt to the shape of empty regions

at all. Hence it is more suitable for a coarse volume decomposition. However, a hierarchical subdivision can adapt to the localisation and shape of opaque regions and can be performed even down to single voxels. Its' efficiency comes from the use of tree-structures, which enable to treat larger opaque or transparent regions as a whole, because the classification of subregions is propagated to their parent nodes. In general, a volume subdivision can be applied in two ways: analytical subdivision of space or decomposing the data according to the subdivision of space. The latter approach can be used to save memory space by excluding irrelevant data, but it is bound to a consistent classification (e.g. fixed TF) and the memory layout may not be shared with other (visualisation) tools in a multi functional environment (e.g. AngioVis-Toolbox). But also under such circumstances it is reasonable to apply a memory subdivision in order to exclude data being defined as irrelevant globally, namely background voxels [5] (see Section 4.2.1). But the general goal of volume decomposition is to separate opaque and transparent regions. Therefor one first has to classify the sub-volumes. A straightforward and 100 percent precise solution is a classification of every single voxel in an offline pre-rendering pass. But this requires long processing times and has to be repeated everytime the opacity TF changes. A conservative approach to avoid repeated pre-rendering when modifying the TF interactively is to use an opacity TF which covers all possible modifications. This leads to a trade-off between strong restrictions for the TF modification, which are even preferable in most applications, and significantly reduced precision in the visibility estimation, reducing performance of empty space leaping strategies. A solution that provides an efficient online opacity estimation is an extraction of TF independent features in the offline pre-rendering step, which are then interpretated while rendering. Those features can be binary histograms ([36]) or min-max densities ([43, 42, 36]). Of course the complexity of this approach depends on the input parameter space of the TF and is not suitable

[5] Those voxels which do not contribute to a CT recording at all, like samples of the surrounding air, are labeled as background to save memory and processing time.

for all TF models. The drawback of this method is that those features require additional data storage. But simple features like (binned) binary histograms [6] can be also applied for speeding up the pre-rendering pass significantly, and reduce its' time consumption to an acceptable - even though not interactive - degree. In this case only the binary information 'visible' or 'transparent' has to be stored, what can be further optimised by the use of hierarchical compression according to the tree structure. Although hierarchical volume subdivision can be performed with balancing by using adaptive sizes (e.g. median-cut mechanism [41]), fixed size octree approaches ([11, 43, 42, 36]) have become well-tried solutions. Their advantage is that templates can be applied to determine the rays' entry positions. Those templates can be realised as 3D cubes, with one representative for each level of the octree. The ray-entry positions in each opaque cube are determined by translating the templates to the volumes' position. Since our application needs to perform one dimensional rotation around a grid aligned axis only, we simplified the octree approach to a slice-wise quadtree approach with 2D templates (see Section 4.3.3).

Distance Transform:

Hierarchical space decompositions are inflexible and align more or less poor with coherent opaque regions. Thus one can assume that quite a lot coherent transparent space inside static subregions being classified as opaque. Distance transforms (DT) provide information about the localisation of opaque regions by assigning a proximity measure to every cell. The proximity is a metric value, namely an efficient approximation of the euclidean distance, denoting the distance to the closest opaque object. The costs for DTs depend on the dimensionality (2D,3D) and the metric which is applied. A general survey on different techniques to perform DTs is given in [33]. The first approach for accelerating raytracing by DT was presented by Zuiderfeld et al.

[6] Binary histograms are generated by marking each apparent density value in a bit array.

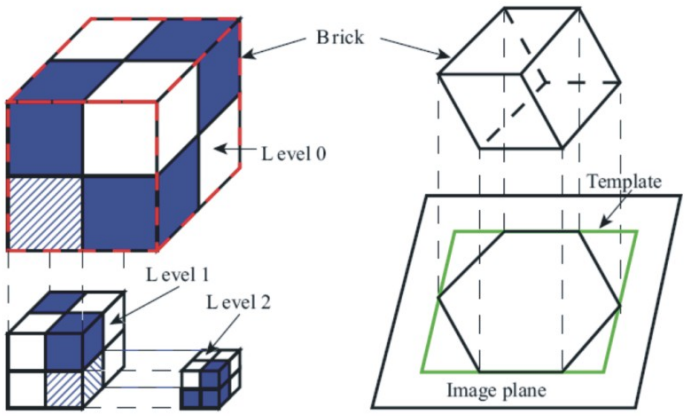


Figure 3.7: Left: Decomposition of a bricks' volume by a 2 level octree. Right: Templates for octree subvolumes. [36]

[34]. They used the proximity information in the raycasting loop to advance the sampling position accordingly. A Chessboard Distance technique was presented by Sramek et al. [30, 29]. Their method skips cubic macro regions of empty space and provides higher accuracy by advancing the rays exactly to the entry positions of opaque cells. These entry positions are located on the faces of the closest macro-region and can be determined efficiently with the algorithm they proposed. The chessboard distance approach benefits also from its' generality, as it is applicable on rectilinear grids with varying voxel sizes. When using parallel projections, further acceleration can be achieved by image-space oriented leaping. For instance, using the minimum distance discovered by a ray which traversed the volume without striking opaque cells to leap all neighbouring rays being closer.

3.3.6 Adaptive Sampling

The ideas of early ray termination and empty space leaping can be generalised to an adequate adaption of the sampling costs, which are determined by the number of samples and the costs for reconstruction and shading, subject to the relevance of particular data for the final image. In short that is

sampling less significant details with low costs. The term significance refers in this case to the relevance of a detail for the rendered image or its global significance in object space. Refering to the contribution to the image, all details which are totally obscured have zero relevance, and details which emit light being opacified by particles in front of them have reduced significance according to their perceptibility. In object space, voxels in homogeneous regions have actually low information due to a low information entropy, and transparent regions are insignificant at all. Details which are transparent or obscured can be excluded from processing by empty space leaping, or rather early ray termination (Section 3.3.5 and 3.3.4). But we focus here on economical sampling of less significant, but visible, details. Danskin and Hanrahan [67] stated that sampling density can be reduced linearly with the opacity-distribution and intensity integral. However, they also showed that the optimal sampling density can not be determined in advance, but it can be estimated roughly by linear functions of α_{acc} and the accumulated intensity (brightness). They also proposed to adapt the sampling density to the homogeneity of the region being sampled. To classify the homogeneity of particular regions, they used a pyramid, but other kinds of hierarchical volume decomposition could be applied also.

A quite interesting approach, involving a memory efficient volume decomposition, was presented by Bajaj et al. [66]. They applied a wavelet compression for parallelised rendering, to minimise the communication between processing elements in a distributed computing network. Since wavelet compression yields a multiresolution decomposition with several levels of detail, insignificant information (e.g. noise, transparent voxels) is skipped efficiently by zero encodings and thus excluded automatically. With a multi-resolutional representation of the data one can adapt the sampling-rate to the highest significant frequency, and homogeneous or smooth regions can be sampled efficiently with an almost optimal sampling rate. In the main, wavelet transformations can be utilised to remove irrelevant information by lossy compression, in frequency and spatial domain at the same time. Further, it

introduces a hierarchical decomposition by (zero-)tree encoding and efficient data access, compression, empty space leaping and adaptive sampling are rolled into one.

Chapter 4

Implementation and own solutions

4.1 Circumstances and Limitations

The DVR software is integrated as a plugin in the system of AngioVis toolbox (the description of the overall software is presented in the PhD thesis of Matúš Straka [6]). This system is implemented in C++ and thus the DVR plugin as well. The software shares classes with other plugins, which are assembled in a systems' library. Hence the software was adapted to some conventions of the systems classes. This applies especially to the data layout and memory of the volume, which is shared by several visualisation tools even during runtime. The purposes of the DVR-plugin are: displaying DVR images of the entire data interactively and generating high-quality renderings for documentation. For our purposes viewing with a rotation with one degree of freedom around a horizontal aligned axis is fully sufficient. This is reasonably due to the large height of CTA data volumes. So we decided to simplify the DVR procedure with this restriction for gaining further efficiency. This simplification is a major difference to common DVR approaches and lead us to a novel software design with additional capacities for acceleration (e.g. templates for ray-traversal).

4.2 Main Approach

4.2.1 Memory Structure

Clinical datasets are submitted in DICOM format to the workstation. For further processing the DICOM files are converted into the f3d format [1], which was especially designed for representation of 3D volume data. When the data volume is loaded into main memory, it is decomposed in a block-by-block memory layout. The notation "block" instead of "brick" was chosen because we also divide and allocate the data in particular arrays according to the blocks. The content of each block is addressed by a pointer to its' data (Figure 4.1). With this layout main memory space can be saved by skipping blocks comprising background voxels only. Those are represented by one common dummy block instead. The sparse data allocation is a main difference to a system which loads the whole volume in one array by using just a brick-by-brick storage order - as it was proposed by Grimm et al. ([37, 36]). Thus we are not able to employ an efficient addressing scheme as they proposed, based on a constant set of offsets being used to "jump" between voxels of neighbouring blocks. To overcome the drawbacks of costly addressing between neighbouring voxels at the blocks borders, one voxel overlapping was introduced to enable gradient computing and trilinear interpolation by accessing to one blocks' data only. Details about the block loading and addressing system can be also read in [6]. Direct voxel access is supported by a block grid manager, which addresses the blocks data by pointer arithmetics. But actually this functionality is not fully utilised by our DVR-plugin. For efficiency reasons, we just use the block grid manager to obtain the pointer to a specific block. The access to single voxels is performed by direct addressing with the actual offsets of single voxels in the one dimensional density array, instead using their 3D grid positions therefor.
The size of the blocks is 34x34x34, whereas 2 voxels units in each dimension (these are $34^3 - 32^3$ voxels) are stored redundantly due to two-sided

[1] About this format see [32]

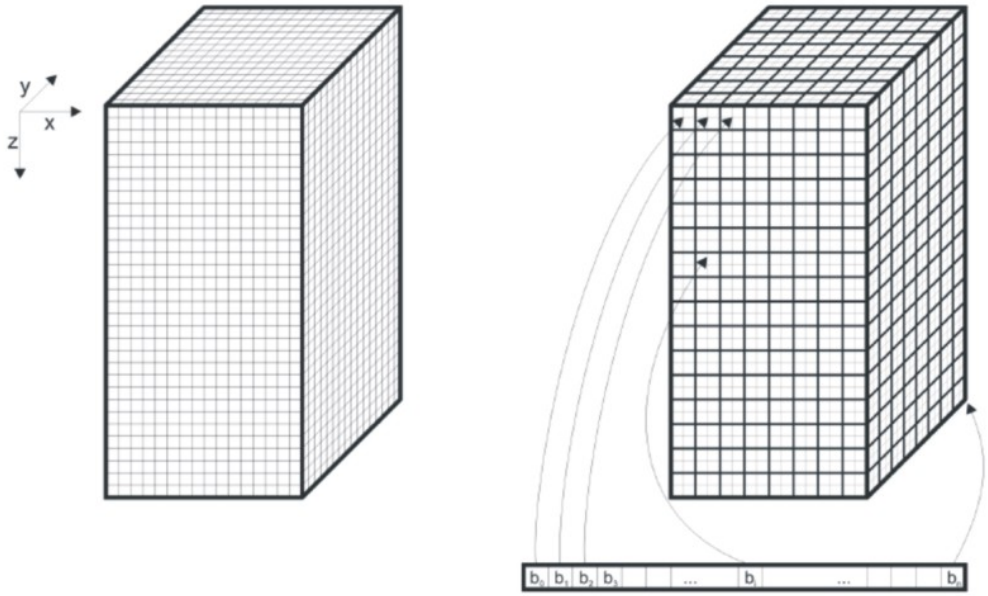


Figure 4.1: The density grid (a) and its' decomposition into a blocked memory layout (b). [6]

overlappings. A single voxel has 2Bytes (12 bit for density and 4 bits for segmentation keys), so the data of a whole block requires 39304x2 Bytes, what is about 80 KByte. For up-to-date CPUs one can expect a Level 2 cache size of 512KByte (e.g. Intel Pentium 4), which is quite enough to keep even several blocks loaded in parallel, so the block system assures a level 2 cache coherent processing.

4.2.2 Transfer Function

Currently we use a straight-forward trial-and-error TF approach, which is more or less a prototype. The TF is designed by specifying a set of control points, consisting of density, chromacity and opacity. These values are then interpolated linearly along the densities and translated to a discrete look-up table comprising 4096 elements. To provide a fast classification of density

ranges, the table is extended by the integrated density-opacity TF. This is done just by accumulating consecutively the entries of the opacity channel in the look-up table. If a min-max range of a quadtree node shall be classified for instance, the opacity integrals according to maximum and minimum densities are subtracted. Results being unequal to zero indicate that the given range contains at least one opaque density and the according volume has to be processed.
To ease the use of our DVR software, we currently provide a set of predefined TF that issue good and suitable visualisations among all datasets. Later we plan to offer an interface to manipulate the TF in a save manner. For example a global opacity modulator for manipulating transparency interactively or a gradient threshold for surface enhancement.

4.2.3 Rendering Passes

Our implementation involves a rendering pipeline with an offline and an online pass:

Loading and Preprocessing: First, the f3d-density grid has to be loaded as a block-by-block representation, what is performed by the system itself. Next, transparent and opaque regions of blocks and their quadtree decomposition (see 4.3.3) have to be pre-classified for empty space leaping. Of course, the TF look-up table has to be generated before. The pre-classification of blocks and quadtrees is done by the use of the proposed estimation method which uses min/max-densities therefor. The quadtrees are build by decomposing each block in slices, and those into quadtree subregions.

Rendering: This involves one step to determine a process-order block list that does not violate the correct ray-traversal (see in Figure 4.4). Then all variables being invariant during the raytracing loop - templates, sinus function of the viewing angle ectr. - are precomputed. Next, an array of rays is initialized according to the dimensions of the output image. The process-

order block list is traversed, while each block being opaque is rendered at once. Rendering a single block is performed slice-by-slice, whereas each slice is fragmented by an opacity quadtree. Regions according to opaque nodes are rendered one after another by raycasting with equidistant sampling. Rendering of opaque quadtree subregions has to be performed also in a suitable order (see Figure 4.4). Finally, the output image is generated by writing the accumulated colour of each ray to the according pixel.

4.2.4 Resampling and Compositing

As we use raycasting, compositing is performed in front-to-back order following equation 2.19. For reconstruction we used three different filters, what will be discussed in Section 4.4. Gradients are estimated with central difference or intermediate distance. Shading was performed following equation 2.13. The sampling pipeline, which was extended by a cell invisibility cache (Section 4.3.4) - is illustrated in Figure 4.6 (right). In this illustration, the "Pre-Classification" refers to an additional step being introduced in the beginning of the pipeline to identify a whole cell - that are the eight voxels at its' corners - as transparent or rather opaque. This is a good tactic to avoid the senseless, but costly, reconstruction (trilinear interpolation) of samples being transparent.

4.3 Acceleration

The basic DRV approach is easy to implement as a straightforward solution. But optimisations introduce momentous complexity and affect the overall design.

4.3.1 Block-by-block Processing

Block-by-block processing has two main advantages: a higher cache-hits rate due to spatial and temporal locality, and a subdivision of the volume, which

enables to exclude large empty regions from processing by skipping empty blocks. Its' implementation challenges managing the processing of entire blocks in a suitable order and rendering of the partial volume according to a single block only. A suitable process order that does not violate the ray-traversal scheme (front-to-back) must be constituted. For a rotation of the viewing direction with one degree of freedom, we need to distinguish only four main directions that require distinct block traversals (see Figure 4.4). The process-order is fixed in a pre-rendering step by sorting the blocks in a list according to the quadrant where the current viewing direction belongs to. Processing a block involves at first a test whether it is transparent, and can be skipped, or not. This test is performed by the min/max range test. When a block is finally rendered, all rays that strike this block have to be determined. This is done by projecting a 2D window on the image plane and selecting each ray inside (see as top view in Figure 4.3). In effect, a block is rendered slice-by-slice, what enhances cache coherency as well. When a quadtree subdivision of the blocks slices is applied, the ray selection is actually done when a particular opaque quadtree region is rendered. The selection window is then obtained from the projection of the quadtree region onto the image plane (only 1D) and the top and bottom of the according slice. Processing partial volumes separately also issues the determination of entry and exit positions of rays traversing them. The solution can be also reduced to 2D space as sketched in Figure 4.3. We introduced a template based ray traversal, whereas templates are generated for one particular stack of blocks only. Hence entry and exit positions are only required to generate the ray templates for one stack of blocks (see4.3.2).

4.3.2 Ray Traversal Templates

Due to the restriction to a rotation with one degree of freedom around an z-direction aligned axis only and a parallel projection, all rays are running parallel to each other and to the slices as well. This means all rays from the same column have an equal path in a projection onto the xy-plane. This

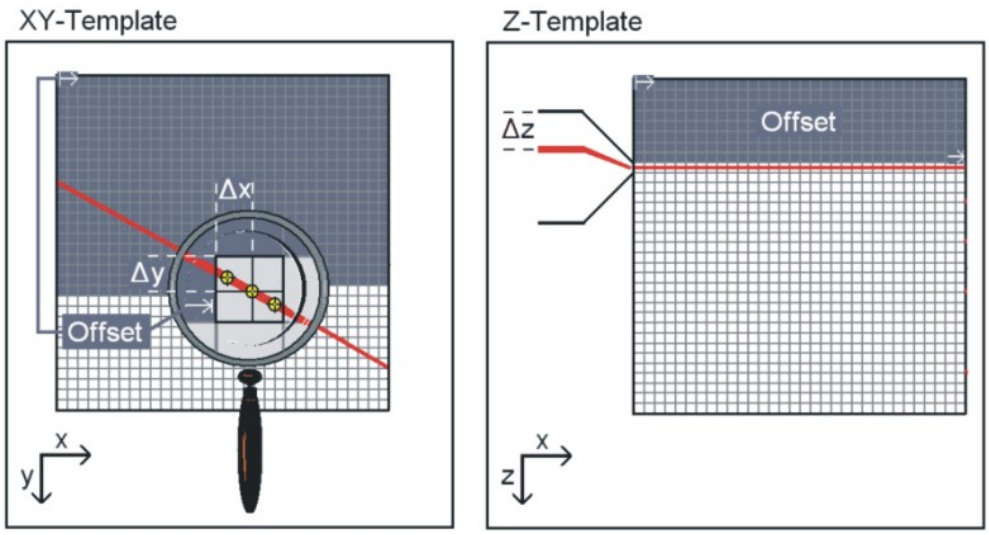


Figure 4.2: The parameters stored for each sampling position in the ray-template table.

fact can be utilised for addressing the voxels' data and determining the sampling positions within a cell efficiently. The densities of a block are stored in xyz-order in an one dimensional array. A voxels' density is addressed by adding its' offset ($\lfloor x \rfloor + \lfloor y \rfloor * blocksize + \lfloor z \rfloor * blocksize^2$) to the address of the arrays' first element. Since all rays of one column have the same path in 2D, they have a similar pattern of *xy-offsets* ($\lfloor x \rfloor + \lfloor y \rfloor * blocksize$), the offsets within one slice of the block. The entire offsets are only shifted by adding the according *z-offsets* ($\lfloor z \rfloor * blocksize^2$), the offset of all slices under the current slice in the block. Those z-offsets are again constant for all rays within one slice. We implemented the template for ray-traversal as a pattern of xy-offsets for each column (*xy-template*) and z-offsets (*z-template*) for each row (in image space), which are stored as simple look-up-tables. Both templates - xy-template and z-template - contain also the sampling positions inside the according voxels ($x - \lfloor x \rfloor, y - \lfloor y \rfloor, z - \lfloor z \rfloor$), which are required for trilinear interpolation. With this templates, ray advancing and positioning is reduced to an iteration on an array of pre-computed values. Only one addition is needed to sum the slice offset from the z-template, which is constant

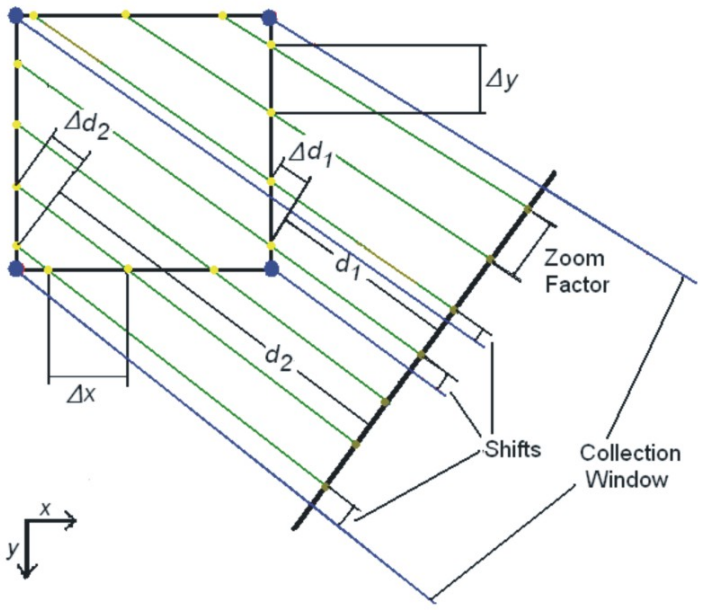


Figure 4.3: How to compute intersections of a rectangular region by a few pre-computed parameters (deltas). Only the shifts between corners and the next ray intersection have to be computed accordingly.

during ray-traversal, and the xy-offset from the xy-template for obtaining the entire offset in the current blocks' data. To save memory, the xy-template is generated for the 2D-region of one block only and reused for all blocks in the same stack. By virtue of that, the block processing order list has to feature also a stack-by-stack traversal. The xy-template actually represented as a two dimensional ($[raycolumn, sample\_index]$) look-up-table. Rather the z-template requires only one dimension, so we compute it for the whole density grid once in the beginning. Due to block-sized tiling, the additional memory consumption for the xy-template look-up-table can be kept in an almost insignificant magnitude. E.g. let us specify a image sampling distance of 0.5 and an object sampling distance of 0.3, what yields extreme high resolution sampling. Then we have in average about 64x96 samples, which need 2Bytes for the offset value and 8 Bytes (2 floats) for the position information. These are finally 30KB, what does not cause a significantly increased flushing of

data in level 2 cache (≥256KB even in out-of-date CPUs). The memory consumption for z-template table is not critical at all, because it needs to be accessed only once per ray per subregion traversal.
To generate the xy-template, we need the entry and exit position of the rays striking the according block in 2D. Figure 4.3 illustrates how positions and depths of ray intersections with rectangular subregions can be computed (in a 2D projection on xy-plane) with using just a few constant slope values (deltas). Depths are needed to determine accurately the first and last sampling position inside a blocks' region. This is necessary to avoid artefacts due to compositing biases, being introduced by inaccurate sampling distances at transitions between two subregions. The according algorithm starts at the first intersection (x_0, y_0) of a ray right hand to the front left corner of the block and computes the position and depth of all consecutive entries just by adding constant slopes $(\Delta d, \Delta x, \Delta y)$ and stops before the next corner is reached. This algorithm has to be performed for the four edges between the corners of the 2D projection of a the subvolume to obtain the entries (left corner to front corner, front corner to right corner) and exits (left corner to rear corner, rear corner to right corner). In actual fact, this optimisation for fast intersection determination was introduced just for optimising ray-tracing without templates. Introducing ray-templates outsourced this computation to the template generation, which is performed only few times, and the improvement is not that relevant any more.

4.3.3 Quadtree Space Leaping

Usually, empty space leaping approaches using a hierachical subdivision employ an octree subdivion strategy to decompose volume data. In our special case, we do not need a 3D decompostition since we can render slice-by-slice. Thus we decided to simplify hierarchical subdivision to a 2D case and employ quadtrees for each single slice instead. At every parent node of the quadtree, the according subspace is splitted in four equal sized child subspaces. The classification of each node is done in the offline pass (pre-rendering) with the

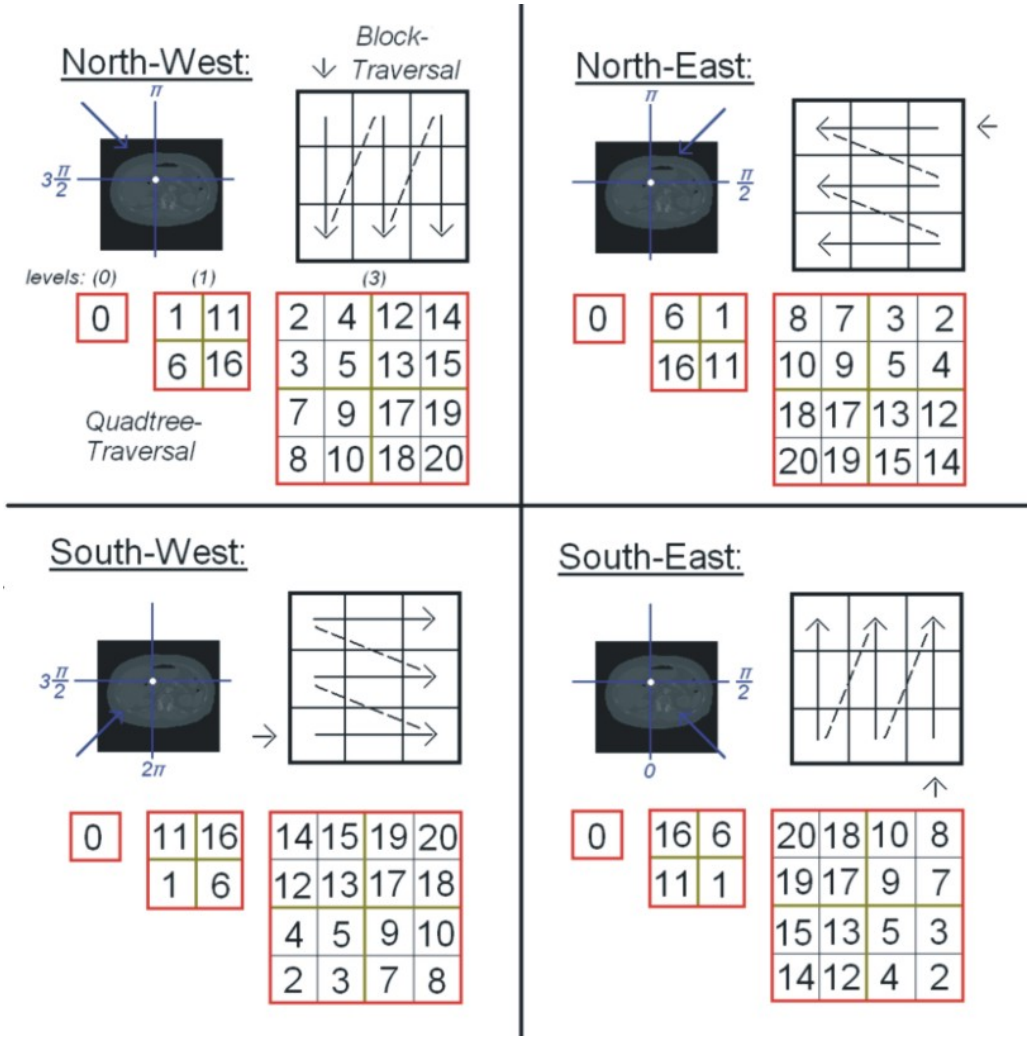


Figure 4.4: A valid traversal-scheme for processing (stack of) blocks and quadtree-nodes. As we rotate only in one dimension, we have to distinguish only four different cases in 2D: viewing from the North-West, North-East, South-East or South-West quadrant.

min-max estimation method suggested e.g. in [36]. It is accelerated by the use of the density-opacity TF integral (Section 2.1). Each node is classified with one of three attributes: ”opaque” , ”transparent” and ”heterogenous”. The subregions represented by opaque nodes are processed as a whole and the subregions of a transparent nodes can be skipped respectively. If the tree

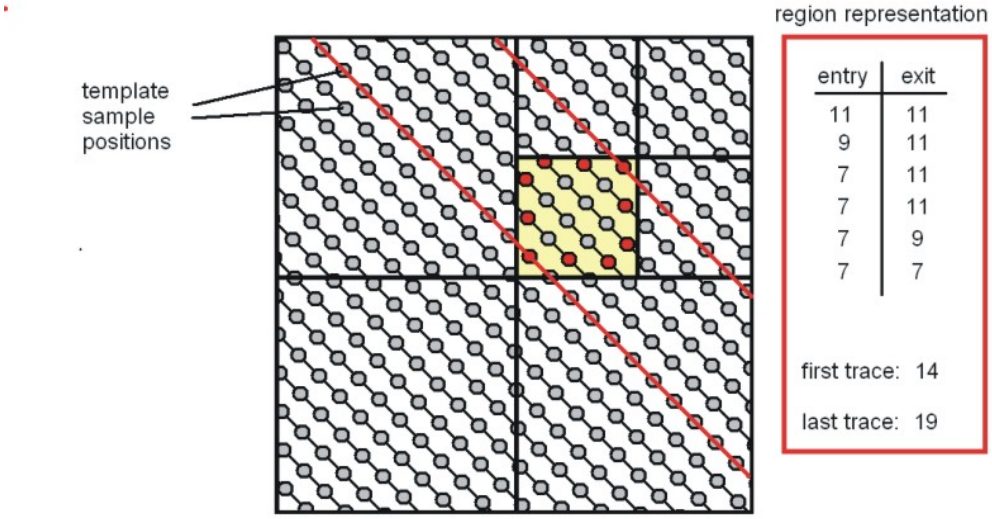


Figure 4.5: The representation of a Quadtree subregion. According to predefined sampling positions, a subregion can be defined on the basis of integer indeces referring to the template.

traversal algorithm reaches a heterogenous node, it has to evaluate the child nodes. For not violating the front-to-back ray traversal scheme, all opaque subregions have to be processed in an adequate order. Thus we have one ranking scheme for all nodes of the trees, which determines the process order for each of the distinct quadrants of viewing direction. The four distinct rankings being used for processing opaque subregions are illustrated in Figure 4.4. When processing an opaque subspace, we need to determine the rays piercing it, as well as their entry and exit positions. Since a ray traversal template is employed and implemented as 2D look-up table, we need only the corresponding indeces of the samples within a quadtree sub-region. Those can be represented by the range of indeces correponding to the traces passing the sub-region, which can be obtained by determining the left- and right-most one. Further we require only the indeces corresponding to the first and last sampling position within the sub-region on each trace. Figure 4.5 sketches

a structure that represents the information needed to render an exemplary subregion of the quadtree. All subregions corresponding to equivalent nodes in quadtrees of different slices of the same block require the same pattern of ray-template indeces for sampling. Thus one can use again a template structure to store and reuse those informations. This contains one representation of each distinct quadtree subregion and is even used for a whole stack of blocks.

4.3.4 Caching

Referring to the work of Grimm et al. [36], we applied a gradient cache and *Cell Transparency Cache* [2] (CTC) in a direct manner as they proposed. The gradient cache and the CTC are implemented as arrays having exactly the same length as the density array of one block. Even their content is addressed with the same offsets as used to address voxels. When a new block is processed, the content of the cache is re-initialized and the allocated space is reused. The left drawing in Figure 4.6 shows how the amount of redundant operations increases with the number of multiple samples per cell. Applying a CTC leads also to a small modification of the raycasting pipeline (Figure 4.6 right), but it has a high impact on the rendering time as the evaluation in Section 4.5.1 shows. Essentially CTC is the last refinement of empty space leaping as it resolves single empty cells. But both caching strategies are only useful when cells are actually sampled multiple times.

4.4 Interactive Mode

We choosed to use three different quality levels to afford an interactive behaviour of our software. This involves two preview modes: a fast low quality rendering for interactive rotation and medium quality rendering in short, but not interactive, time. The final level is a high quality rendering, which

[2] They ([36]) used the term "cell invisibility cache". We replaced it, because it might be misunderstood as invisibility of hidden voxels.

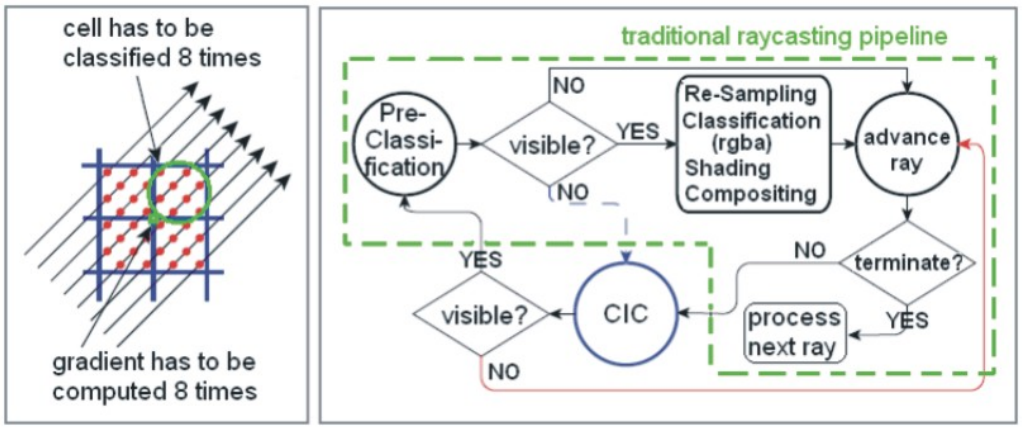


Figure 4.6: Left: Redundant operations which may be reduced by caching. Right: Sampling pipeline extended by Cell Invisibility Cache (CIC).[36]

is accomplished within a couple of seconds (10-20). The parameters for low quality renderings are configured in such manner that an almost runny display during an interactive rotation is possible. A frame-rate of at least 3 fps (frames per second) turned out to be indispensable to meet this demands. To achieve such framerates we need to select an adequate setting of the sampling parameters. This means we had to find a suitable setting of image and object sampling distance. Figure 4.9 shows an exemplary table of rendering times with respect to different sampling distances and two efficient reconstruction filters (mean filter, next neighbour interpolation). Among those settings we choosed a suitable candidate for configurating the preview mode. The exemplary setting we choosed has an image sampling distance of 3.0 voxel units (distance between rays) and an object sampling distance of 1.0 voxel unit (distance between samples in ray direction). Due to this under-sampling of the density grid, a high amount of alias frequencies occur when the filter radius encloses one voxel only. The resulting images suffer from colour jittering, which becomes especially apparent while rotating interactively. To suppress this, a low-cost smoothing reconstruction filter was applied. The cost of a reconstruction filter is mainly driven by the complexity of the used reconstruction function and the number and locality of

voxels to be accessed. We propose a smoothing filter which is a mean function over the next neighbouring voxel and all its adjacent voxels having a chessboard distance of one to it. This design was chosen since it can be implemented in a very efficient way by re-using all voxels being accessed for the gradient estimation by central-differences. Since the gradient computation is executed in all relevant (visible) samples anyway, accessing the same neighbourhood for density reconstruction comes almost for free (see the only slight differences between next neighbour interpolation and mean filtering in Figure 4.9). If an image sampling distance of 3.0 is chosen, only four out of nine voxels - instead eight out of nine when using next neighbour interpolation - do not contribute to the output image. Tests showed that this filter reduces the amount of colour flicker while rotating. However, the resulting images have a low resolution (approx. width/height of the volume divided by the image sampling distance) and are only sufficient to give a rough orientation while moving the viewing angle interactively. To display a fully resolved image, we need to render with an image sampling distance of at least 1.0 together with an object sampling distance of 1.0 to guarantee that each visible voxel is processed at least once, and thus contributes to the output image. If our mean filter is used an object sampling distance of 1.5 is also still adequate. But smoothing is not always desired and due to the sufficient image resolution we can also apply a next neighbour interpolation for efficient rendering. When employing a next-neighbour interpolation for gradient and density reconstruction, our software rendered full datasets with up to 1.5 fps. The resulting images have a resolution of about 1200x500, what is almost the limit to be displayed on nowadays monitors. Their reconstruction imperfections are almost not noticeable in a non-zoomed display. Hence we can display high resolution preview images with sufficing quality, being rendered in just a moment. For detailed inspection a high quality and definition image is rendered with oversampling and trilinear interpolation in the next 10-12 seconds (for 2000x4500 pixels, with image sampling distance: 0.5, object sampling distance: 0.3). This images are suitable for zooming op-

erations and they can be generated especially for documentation purposes, which tend to be not such time-critical anyway. More details about rendering times can be seen in the evaluation presented in section 4.5.1. The images according to the proposed quality levels are shown in Figure 4.8.
In our interactive DVR tool, the images are displayed in progressing quality. When some parameter (e.g. view direction) is modified, a fast low-quality rendering is performed and images of the next two quality levels are rendered successively as long the parameters are not changed by the user. This was implemented with a procedure starting the process of rendering images with progressing quality in a thread, which is interrupted and restarted when a new "parameters changed!"- event occurs in the GUI driving loop. A safe interruption function was implemented by introducing a volatile flag, which is always checked before a new block is processed. If this flag is set to "interrupt!", the rendering function returns after processing the actual block (see Figure4.8).

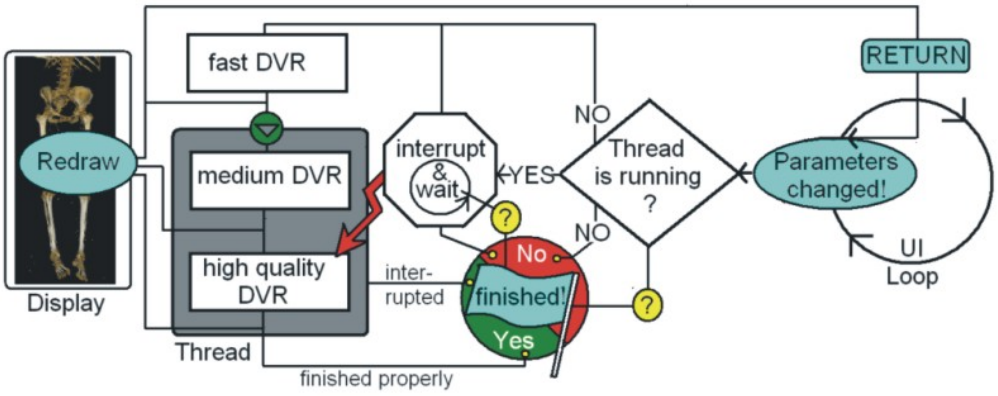


Figure 4.7: The interactive mode with a progressing image pipeline, which is parallelised with a thread.

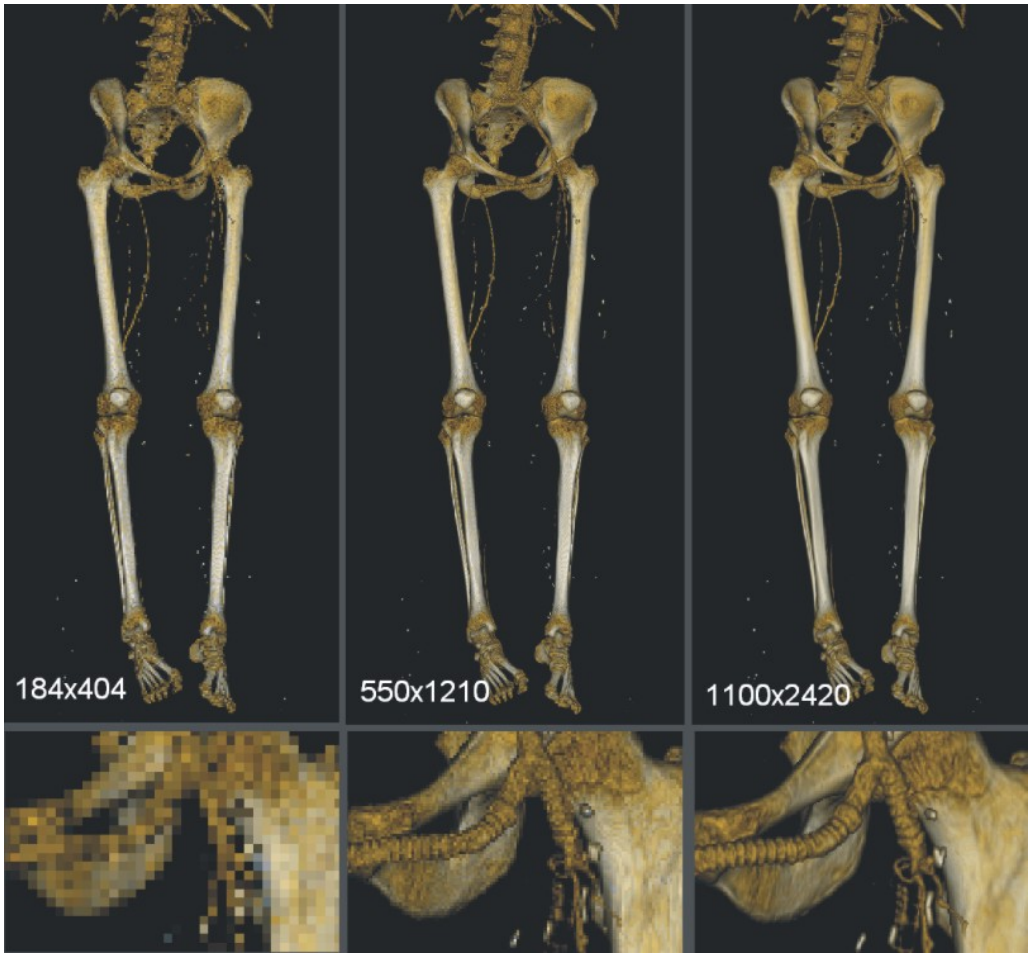


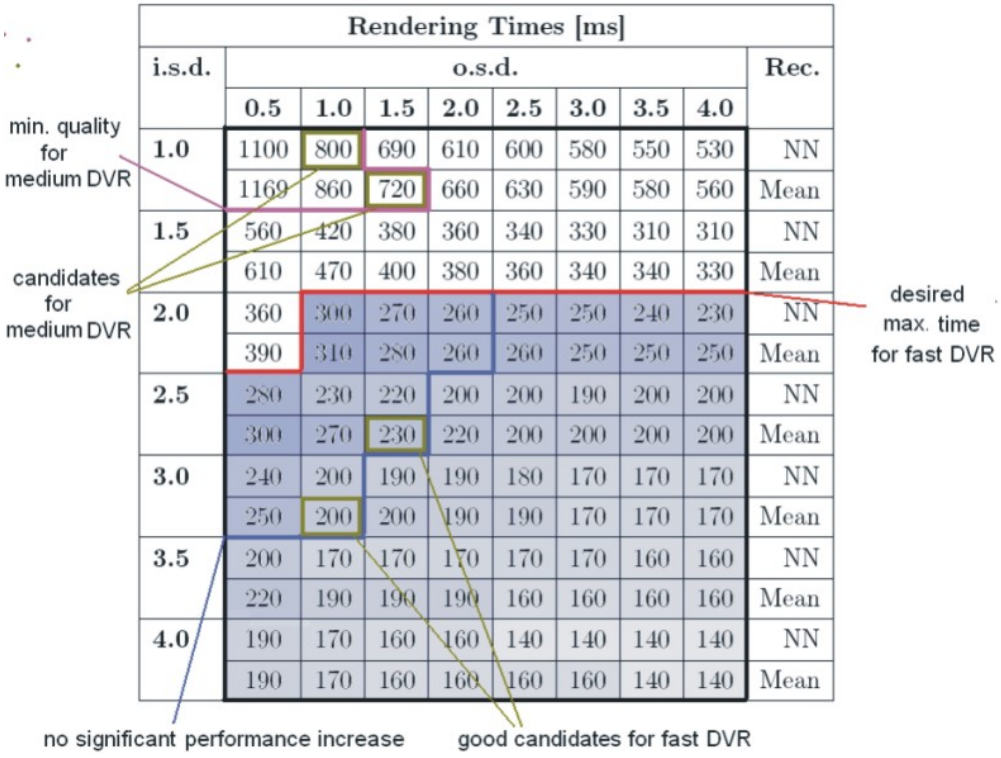
Figure 4.8: The dataset being used for our evaluation rendered with three different quality levels: fast rendering with image sampling distance 3.0, object sampling distance 1.0 and mean filter in 0.3 s (right). Medium quality rendering with image sampling distance 1.0, object sampling distance 1.0 and next-neighbour interpolation in 0.8 s (center). High quality rendering with image sampling distance 0.5, object sampling distance 0.3 and trilinear interpolation in 11 s (right).

4.5 Results

4.5.1 Evaluation of Acceleration Techniques

Circumstances

The following evaluations focus on acceleration strategies which are not innate due to our systems' data loading and representation design. This means



Rendering Times [ms]									
i.s.d.	o.s.d.								Rec.
	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	
1.0	1100	800	690	610	600	580	550	530	NN
	1169	860	720	660	630	590	580	560	Mean
1.5	560	420	380	360	340	330	310	310	NN
	610	470	400	380	360	340	340	330	Mean
2.0	360	300	270	260	250	250	240	230	NN
	390	310	280	260	260	250	250	250	Mean
2.5	280	230	220	200	200	190	200	200	NN
	300	270	230	220	200	200	200	200	Mean
3.0	240	200	190	190	180	170	170	170	NN
	250	200	200	190	190	170	170	170	Mean
3.5	200	170	170	170	170	170	160	160	NN
	220	190	190	190	160	160	160	160	Mean
4.0	190	170	160	160	140	140	140	140	NN
	190	170	160	160	160	160	140	140	Mean

Figure 4.9: Selecting candidates for interactive mode. Comparing different image and object sampling distances (i.s.d.,o.s.d.) and different reconstruction methods: mean filtering (mean) and next neighbour interpolation (NN). We intuitively choosed the highest quality providing a significant performance increase. We estimate the quality decrease linearly against the o.s.d. and quadratically against the i.s.d. The brightness of the blue colours is related to the rendering times to give a better visual orientation.

that one of the major impacts, which is the cache coherent processing, is actually not evaluated. The reason is that cross-comparisons with a not block-by-block processing algorithm with the other acceleration techniques would be severely biased, since an overall redesign of several algorithms, which are strongly adapted to the environmental system and the concept of block-by-block processing, would be necessitated for this purpose. We also pass on an explicit evaluation of the effects of background- or transparent block skipping, since this is a self-evident strategy and also innate to a block-by-block

processing scheme. An estimate of 85% of blocks being totally skipped, when using a typical TF for our purpose, should give a sufficient idea of its' effect. The evaluation presented below was performed on a representative dataset containing a density grid of 512x512x1211 voxels, which was rendered with a representative TF for purposes of angiographic visualisation. With this TF the fraction of voxels being classified as opaque was only 1.5%. The evaluation was done on a personal computer with an Intel Pentium 4 CPU with a clock speed of 3.2GHz, a level 2 cache of 512 KByte and a main memory of 2GB. The resulting images are shown in Figure 4.8.
To be able to analyse each acceleration technique independently we used precompiler commands to exclude/include all lines of code belonging to certain technique. We tried to accomplish this with as much reliability as possible and therefor we also had to add some additional code. For example the absence of ray-templates are simulated by (re-)introducing a function which computes the ray position. To provide a reasonable comparison, we tried to find the most efficient line traversal algorithm, that works without any flooring operations or rather integer typecasts, which tend to be very costly. All acceleration techniques were implemented as described above, but for the sake of completeness some performance influencing facts have to be remarked: the early ray termination is executed with a opacity threshold of 0.9 and the applied visibility quadtree has a depth of two levels only (three levels may follow in future work).

Definitions

The performance of a particular acceleration technique is interfered by other acceleration techniques being applied at the same time. Thus we wanted to assess the performance of a set of various acceleration strategies in particular and with respect to each other. A brute force evaluation approach would involve to test each combination of switching on and off single acceleration techniques, and this would require to assess 2^N different combinations, whereby N denotes the number of different acceleration strategies to be eval-

uated in comparison. To reduce this large and confusing evaluation space, we decided to define two views to assess and interpret the performance of a certain technique:

Potential: We want to assess the *Potential* of a certain acceleration technique by measuring the performance when it is employed solely. Therefor we compare the times for renderings with and without this technique, while all other acceleration techniques are switched of.

Contribution: We regard the *Contribution* of a particular acceleration strategy as its' impact on performance when employing it additionally to a given set of other acceleration techniques. We propose to compare rendering times when rendering with and without this technique, while all other acceleration techniques are used in both cases. To have a general consideration, we will later define the "Contribution" related to a set of acceleration strategies. We introduce the term "contributional speed-up" for the additional acceleration which is gained by involving a certain acceleration strategy in addition to others. It is actually computed by dividing the rendering time for runs involving all acceleration techniques with the time for rendering without the analysed technique.

Effects on Fast Rendering

For fast rendering we choosed to employ the mean filter for density reconstruction together with a next neighbour interpolation for gradient estimation. Sampling was done with an image sampling distance 3.0 and an object distance of 1.0 (See candidates in Figure 4.9). As this configuration leads to maximal one sample per voxel and the according gradient estimate, acceleration by Gradient- and Cell Transparency Caches is without any effect and is not applied. Thus we evaluated only the performance of the strategies *Early Ray Termination*(ERT), *Quadtree Space Leaping*(Quadtree) and *Raytraversal Templates*(Templates) in comparison to each other.

Potential				
Absolut time(ms)	1080	330	950	730
ERT	-	-	-	x
Templates	-	-	x	-
Quadtree	-	x	-	-
Speed-up	1.0	3.3	1.1	1.5

image size: 184x404 o.s.d. = 1.0

Contribution				
Absolut time(ms)	200	620	250	260
E.R.T	x	x	x	-
R.Templates	x	x	-	x
Quadtree	x	-	x	x
Speed-up	5.4	1.7	4.3	4.2
Contr.Speed-up	1.0	3.1	1.2	1.3

Employing all acceleration techniques together leads to a total speed-up factor of 5.4 for this configuration. Obviously, the main acceleration is achieved with applying Quadtrees for empty space leaping. But also the other techniques provide a significant and valuable speed-up.

Effects on Medium Quality Rendering

For efficient rendering with full resolution (every relevant voxel is sampled) one can apply next neighbour interpolation, which is the most efficient reconstruction function. For this evaluation we choosed a typical configuration for rendering a precise preview image: an image sampling distance 1.0 and a sampling distance of 1.0 (see candidates in Figure 4.9). Thus when rendering an image each relevant voxel is sampled once in average. Thus there are no redundant operations due to multiple sampling of single, and we consider the same set of acceleration techniques as above.

Potentials				
time(ms)	8400	1700	7200	5700
ERT	-	-	-	x
R.Templates	-	-	x	-
Quadtree	-	x	-	-
Speed-up	1.0	4.8	1.2	1.5

image size: 550x1210 o.s.d. = 1.0

Contributions				
time (ms)	750	4600	1200	1100
ERT	x	x	x	-
R.Templates	x	x	-	x
Quadtree	x	-	x	x
Speed-up	11.2	1.8	7.0	7.6
Contr.speed-up	1.0	6.2	1.6	1.5

This test showed that we could speed-up renderings with medium quality settings with a factor of 11.2. The effect of acceleration techniques is considerably higher with this resolution. The number of samples was about 800% more than for fast rendering, but the rendering time increased only about 270%. Again Quadtree space leaping acceleration is the most potential one and also yields the highest contribution. This let us conclude that we might gain more speed by introducing further levels into the Quadtree.

Effects on High-Quality Rendering

For a full performance evaluation, we tested the behaviour of certain acceleration techniques with respect to the image and object sampling distance. The filter for gradient and density reconstruction was a trilinear interpolation, which is noticeable more costly than a next neighbour interpolation. To assure smooth shading, an object sampling distance of 0.3 is recommended for high quality rendering. This setting was used to test with varying image sampling distances, what is shown in Figure 4.10. Renderings with image sampling distances 0.5 or higher are suitable for high-quality reconstruction, what is required for zooming. Those renderings might be valuable for documentation or printing, as they allow to zoom on details after rendering, what can be even done with standard 2D imaging software. We choosed such a high resolution setting, which yields images of 1099x2420, to test the performance according to a varying object sampling distance. The results of this testings are shown in 4.11. Our software renders with the proposed high quality setting of 0.3,0.5 (i.s.d.,o.s.d.) on our PC (Intel Pentium 4) in about 11 seconds - a time which certainly does not extend someones' patience. The absolute timings of the fully accelerated software are sketched with the white graph in the top left diagrams of both figures (4.10,4.11).

Gradient caching seems without any significant effect. This is contrary to the suggestions in [36] and [48], and needs to be discussed: Indeed this evaluation is performed for a representative TF for angiographical visual-

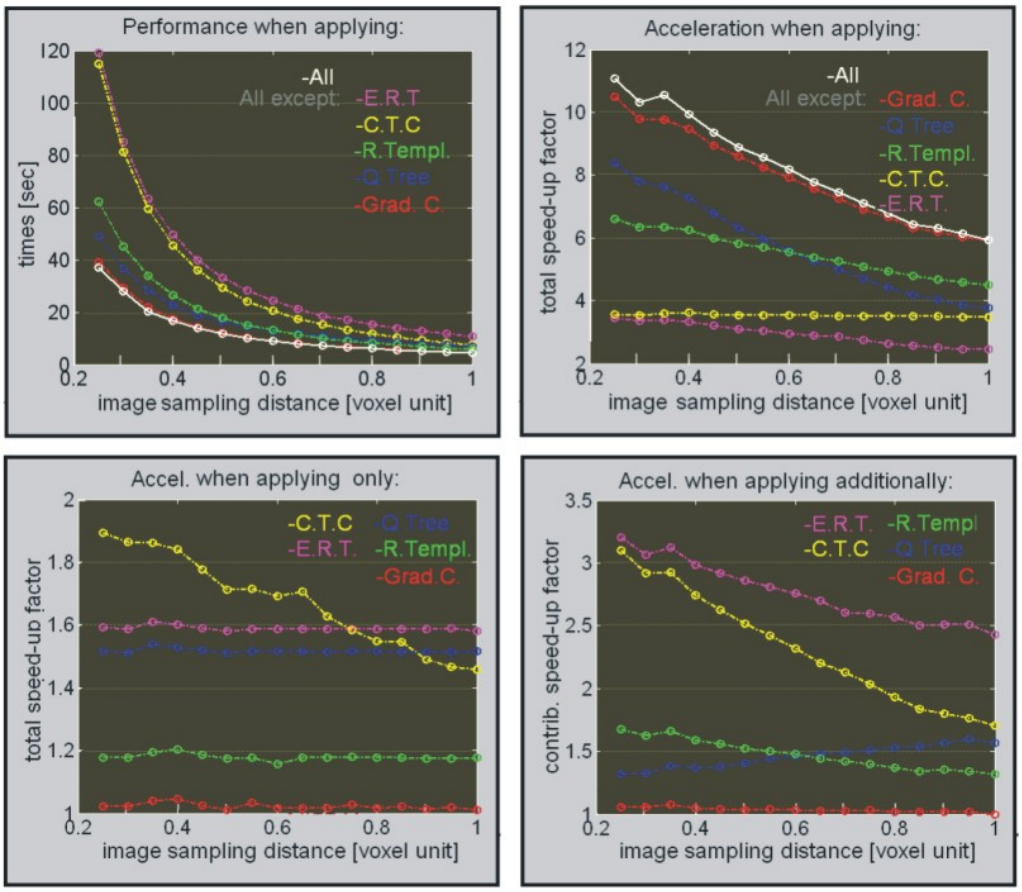


Figure 4.10: Evaluating performance behaviour with respect to varying image sampling distance. We used a constant object sampling distance of 0.3 .The down left diagram shows the speed-up when employing a particular technique solely ("Potential"), whereas the right-hand one shows the speed-up when using all acceleration techniques without and with a particular among those ("Contribution").

isation, nevertheless this is a TF having special speed relevant properties. The amount of required gradient computations is very small when rendering a "bones and vessels only"- image. Thus we can not directly compare to the behaviour of more common TF designs, as e.g. Grimm and Bruckner [38, 39, 36] examined. Another reason might be, that we are using only central difference gradients estimates, which are not that costly. However, this issue has to be clarified earnestly in future to rule out any fundamental

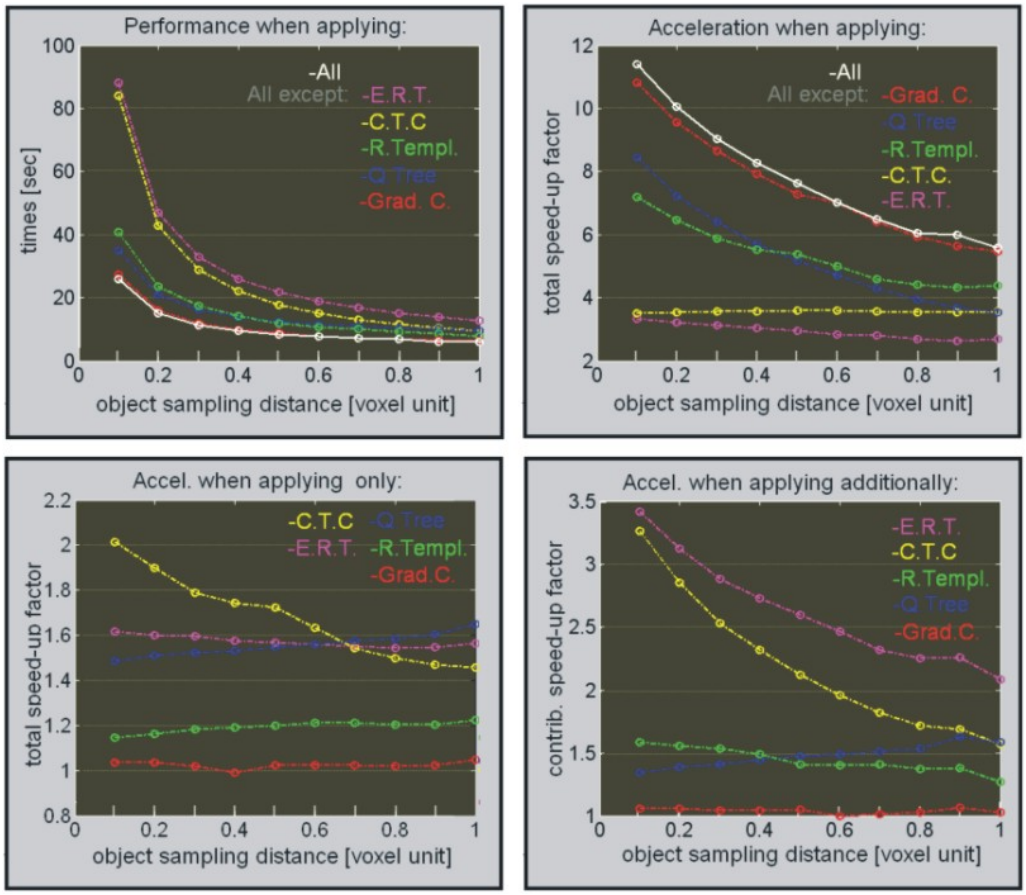


Figure 4.11: Evaluating performance behaviour with respect to varying object sampling distance. Here we used a constant image sampling distance of 0.5 .The down left diagram shows the speed-up when employing a particular technique solely ("Potential"), whereas the right-hand one shows the speed-up when using all acceleration techniques without and with a particular among those ("Contribution").

mistakes in our implementation which might cause this weakness.

CTC yields almost amazing effects and turns out to be one main accelerator. This shows that empty cell leaping is of enormous importance, what is reasonable for such a TF (about 1% opaque cells in the grid, 10% in processed blocks). Nevertheless this fact concludes us to consider a refinement of empty space leaping by **Quadtrees**, which might exploit the presence of

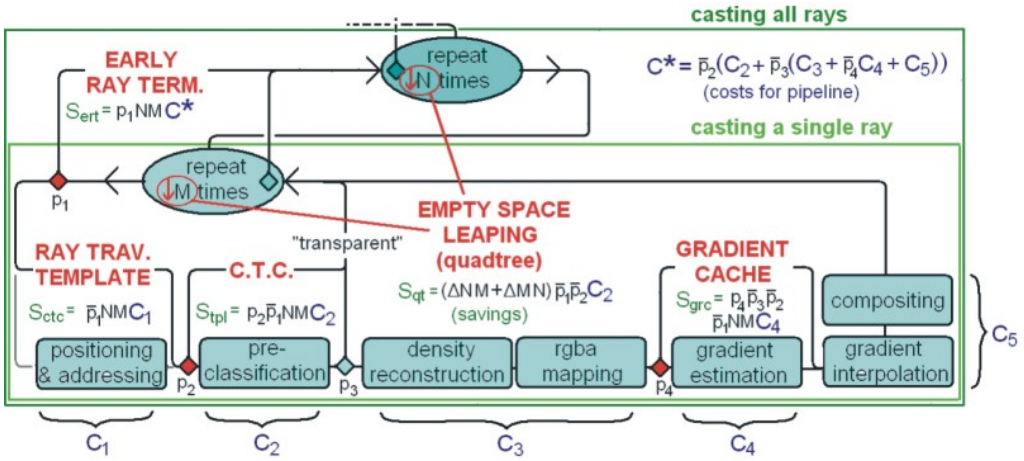


Figure 4.12: A probabilistic model how acceleration techniques affect the rendering pipeline and which cost savings they yield. We used probabilities $(p_1, \ldots, p_4)$ to model the execution of alternative paths in the pipeline. The savings (e.g. S_{ctc}) are retrieved from a probabilistic modeling of the costs $(C_1, \ldots, C_5, C^*)$ being saved by a particular acceleration technique. But we have to take into account, that the probabilities have some relevant dependencies: the probability for ERT p_1 increases with an increasing probability that a cell is opaque $\overline{p_2 p_3}$; the probability ,p_2, that a transparent cell is marked in the gradient cache increases with the ratio of samples per cell and decreases with the amount of savings by empty space leaping tactics such as Quadtree or empty block skipping.

empty space even more effectively. To illustrate the effects of the examined acceleration techniques, we attempted to model their effectiveness in Figure 4.12. The model assumes that the savings of Ray Templates, CTC and Gradient Caching acceleration are stochastical independent to each other. Of course ECT and Quadtrees interfere with each other, as they both attempt to save costs for exactly the same operation - the pre-classification. Thus an increasing effectivity of Quadtrees reduce the propability (p_2) that a cell is transparent and cached in the CTC. The Quadtree lacks due to its' coarseness, whereas the effectivity of CTC suffers from an overhead, being introduced by the fact that the pre-classification of a cell has to be executed at least one time before the caching mechanism works. As also Quadtrees

introduce an overhead, due to fragmenting the ray traversal, the best balance of both methods should be found to achieve the optimum acceleration. The current results suggest that a refined Quadtree (more than 2 levels) would lead to more gain in speed. Actually the Contribution of Quadtree acceleration is directly related to the overhead of leaping cells alternatively by CTC. Consequently it decreases with the increasing number of samples per cell, which determines the ratio of executed pre-classifications and CTC-avoided pre-classifications.

Ray Templates show a more or less constant behaviour. They have a solid effect, which gets amplified with a cost-reduction for other operations by other acceleration techniques, since the relations change then.

ERT show a major influence on rendering speed. The testings in Figures 4.10 and 4.11 were done with a constant opacity threshold of 0.9 for ERT. A further evaluation with a varying opacity threshold is presented in the plots on top of Figure 4.13. Although rendering can be speeded-up by reducing the opacity threshold, it introduces a bias and a significant quality decrease ($\alpha < 0.7$), which comes apparent as dark noisy artifacts. Despite its' great effectivity in the evaluated settings, ERT has one major drawback since its' performance decreases with an increasing translucency of the TF. This has to be taken into account, because translucency of bones is often desired for angiographic visualisation. The bottom plots in Figure 4.13 present the results for various modulations of our representative TF, by just multiplying a factor with its' opacities. In fact, the performance of ERT converges to zero acceleration when the translucency is increased.

4.6 Future Improvements

The presented software is integrated into the system of the AVTB, but several improvements and functionality extensions are planned. This might involve further acceleration by parallelisation, or an optimisation of particular core operations (e.g. reconstruction) with SIMD instructions or self-written assembler code replacements. The current DVR software does not support segmented rendering. Since segmentation information is available anyway in the density grids being delivered to the DVRenderer, an extension featuring different TFs for bones and other tissue is planned. Further progression on our software will focus on interactivity and usability. Especially interactive manipulators of the TF is required to provide an interactive adjustable classification. Therefor only a small degree of freedom in manipulation should be provided in order to hide the complex task of TF configuration from the user. Our ideas concern the manipulation of predefined TF by global opacity modulation or slight shifts of the opacity TF. Also useful would be an interactive conventional clipping functionality apart from the VG features.

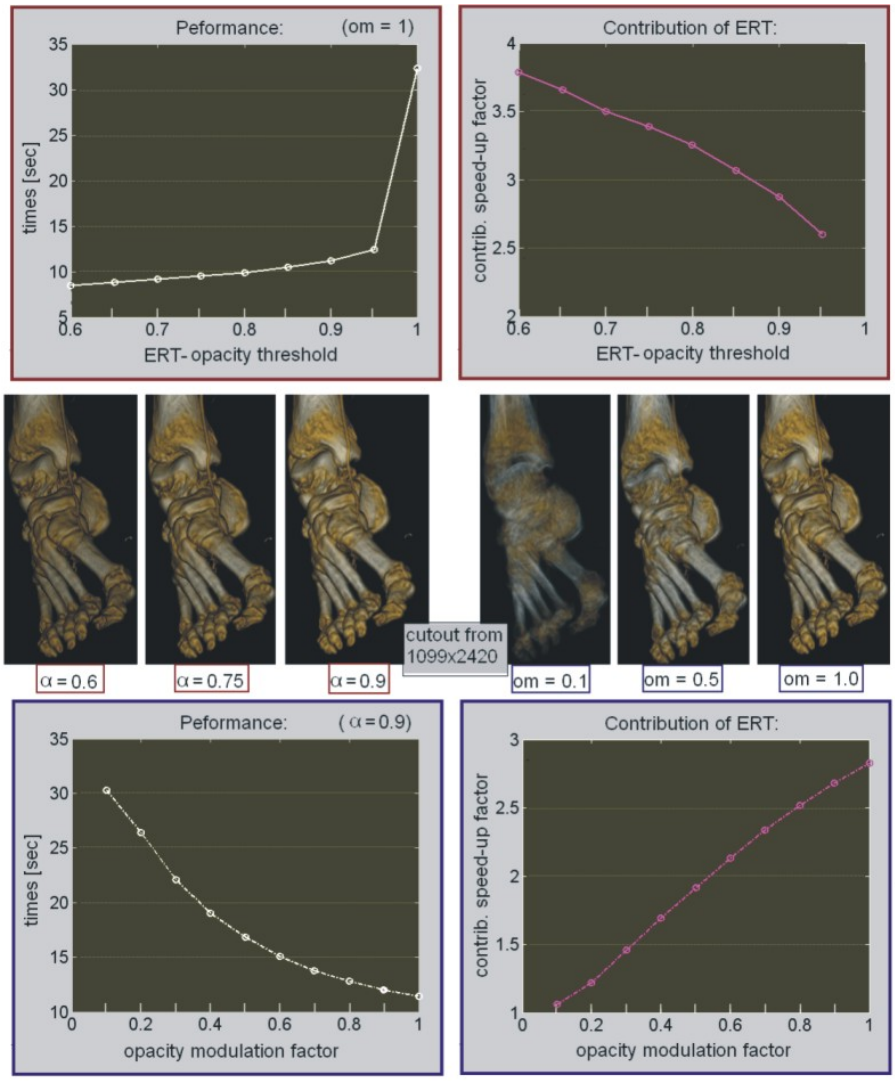


Figure 4.13: Performance and Contribution of ERT when varying its' opacity threshold or changing the TF with a global opacity modulation factor: An i.s.d. of 0.5 and an o.s.d. of 0.3 was used to render a full dataset of 512x512x1211. The left sides show the absolute timings and the right hand graphs illustrate the contributional acceleration ERT with varying opacity thresholds (top), or translucencies (bottom) respectively. Nevertheless rendering can be speeded-up by reducing the opacity threshold, it introduces a bias and significant quality decrease which comes apparent as dark noisy artifacts. Obviously the effectivity of ERT is decreasing with higher translucency.

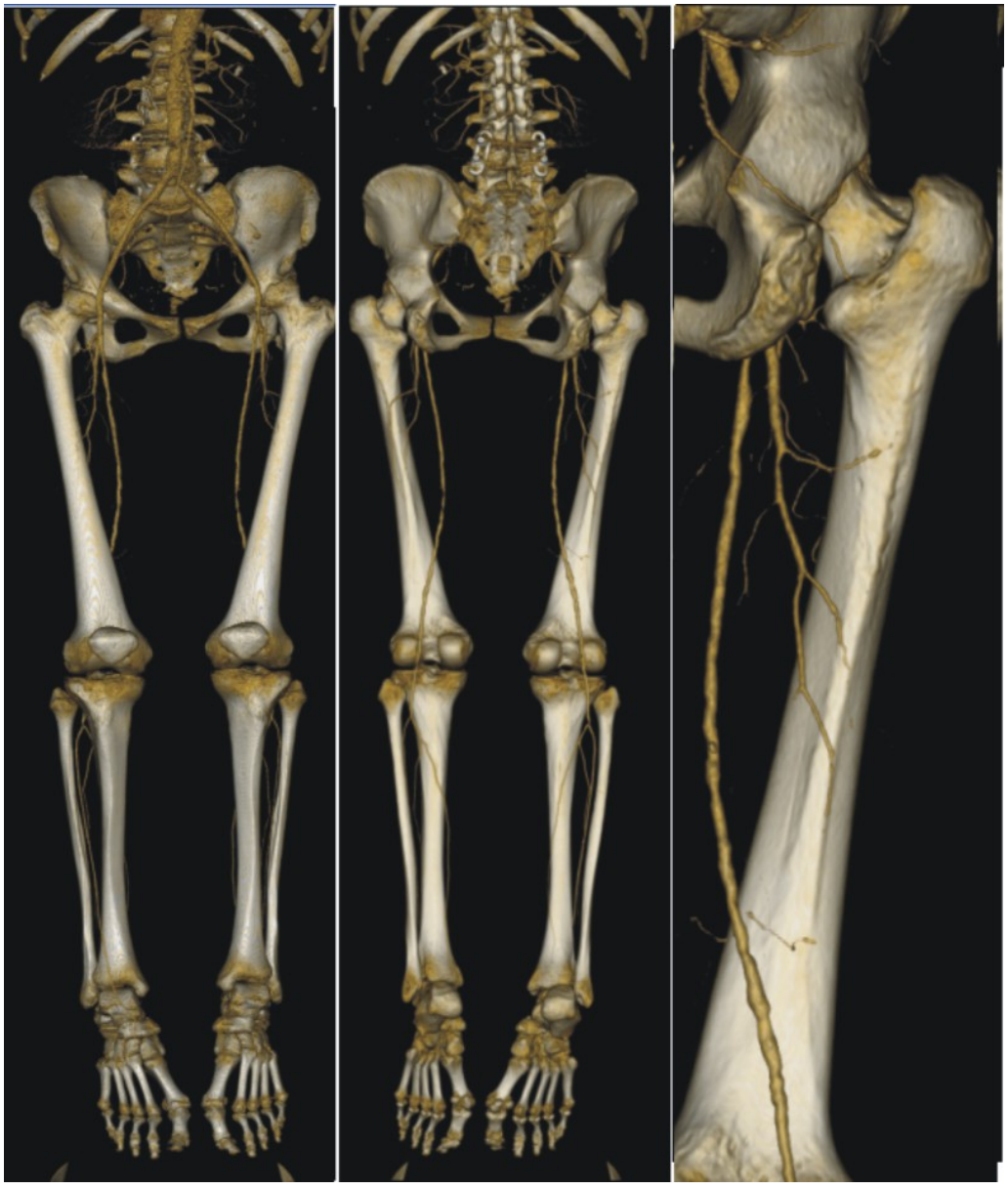


Figure 4.14: High-quality renderings for vessel visualisation. The right image is a zoomed cutout of the image in the center.

Chapter 5

Focus and Context Visualisation by the Concept of VesselGlyph

5.1 Motivation and Idea

Every kind of visualisation method has its own limitations and can not fulfill all visualisation goals at the same time. Those limitations concern either disabilities of a certain technique itself or the configuration of the visualisation parameters making some goals contrary to each other. CPRs, for instance, can display the flow lumen of vessels, but due to spatial distortion they fail in giving an intuitive and interpretable image of the vessels context. Overlapping densities, which are used as features for DVR classification, make some visualisation goals contrary to each other. For example, it is not possible to classify focus regions, namely vessels and their close neighbourhood, with high opacity and making the less important context data translucent at the same time.

The notation *VesselGlyph* (VG) was introduced by Straka et al. [70] as a novel term for a context and focus visualisation of tubular structures like the vesseltree. The context and focus visualisation approach is well-known from information visualisation ([69]). A comparable volume visualisation approach was presented in [72], with hybrid rendering (DVR and non-photorealistic

VR) of a lens-like focal region - actually a sphere - and its' outside region. In [71] they presented a two-level rendering approach for rendering different objects, which are represented as arrays of member voxels, with MIP or DVR. Rather the VG-concepts refers to a two-level rendering of focus and context regions which are both vessel-tree related. Therefor one takes advantage of partially segmented data with approximately identified vessel centerlines to define such regions being rendered differently. The *focus* region comprises objects of primary interest, namely the vesseltree and its' close neighbourhood. The *context* region represents the space of less interest, being only visualised to support the observers' orientation. Separating those regions enables to apply contrary visualisation parameter settings or even distinct visualisation techniques to render one image. The clinical appreciation of VG-Visualisation is on the one hand the general ability to visualise vascular details within an anatomically correctly depicted context. On the other hand it may enable to improve the visualisation of collateral vessels by utilising the features of VG for accurate vessel-related two-level clipping. The augmented presence of collateral vessels is a hint for significant occlusions in the main vessels' flow-lumen, since those vessels are enlarging by-and-by with increasing pressure due to congestion in the main vessel path.

5.2 Realisation

Since a vessel tree model provides an estimation of the vessels centerline, a well suited focus region can be derived easily by specifying a neighbourhood (e.g. circle) of this centerline. The context region can be related to this centerline as well, by specifying a larger neighbourhood and excluding the focus region. Straka [70] proposed a set of symbols to represent different classes of VG specification(Figure 5.1). This classification is based on two dimensions: the shape of the context and focus regions (Figure 5.2) and the kind of visualisation technique being applied for each region (Figure 5.1). Since CPR provide a highly informative visualisation of the vessels flow lumen,

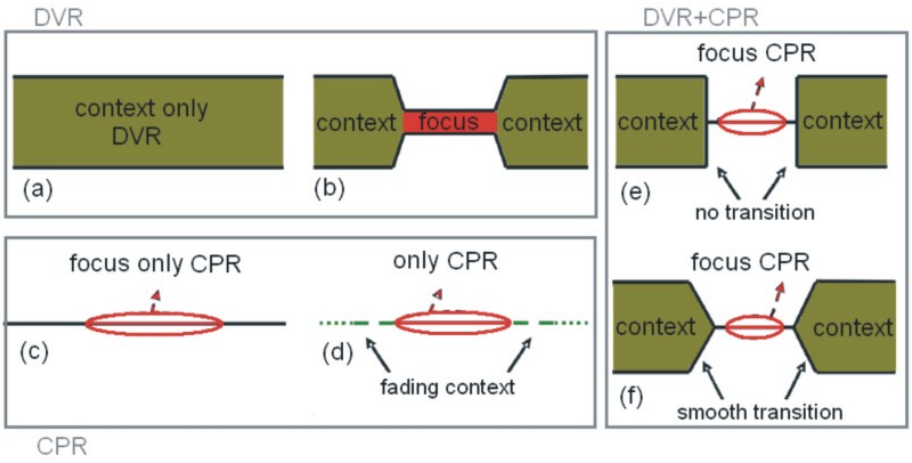


Figure 5.1: Exemplary VG modality symbols: Symbol (a) exemplifies VG-DVR as generalisation of conventional DVR, whereas symbol (b) represents what we propose actually as VG-DVR with an existing focus and context region. Applying the VG-concept to CPR-Visualisation only is represented in symbols (c) and (d). Hybrid applications with and without smooth transitions between the VG-regions is symbolised in figures (e) and (f). These figures should just sketch a new terminology of symbols which can be extended arbitrarily. It could be applied later for the GUI for instance.

but suffer from a distortion of the contextual information, they are suited for focus visualisation. Whereas DVR and MIP are well suited for context visualisation, since they image surrounding tissues in a very intuitive manner. Consequently this combination compensates the major drawbacks of those visualisation techniques by applying them adequately with respect to their specific purposes. Another alternative is the use of DVR with contrary parameter settings for both regions. This enables e.g. to visualise the focus content with a high opacity, while the contextual tissues are sketched with translucent clouds, which do not obscure hidden vessels as much. VG rendering also features volume clippings related to the vessel-paths, what is in fact done by excluding all voxels outside focus and context regions. Clipping insignificant regions can also enormously speed up the rendering procedure

and that without loosing information of interest. VG-DVR features also an enhanced visualisation of collateral vessels. We propose therefor an image parallel slab being sweeped along the vessels' centerline, which enables to carve out collateral vessels right and left handed to the vessel-tree. By rotating the viewing-direction one can then explore the localisation of collateral vessels around it.

For this work we want to point out two important kinds of VG usage which involve DVR:

5.2.1 VesselGlyph DVR

We propose to specify pure VG-DVR by the shape of focus and context regions. Therefor we propose the figuration presented in Figure 5.2.

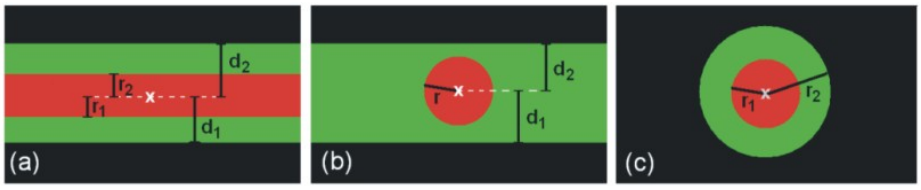


Figure 5.2: How to specify the VG-regions: (a) thick context and thin focus slab, (b) tubular focus and context slab, (c) tubular context and focus regions.

We suggest to specify the VesselGlyph regions by two simple geometric primitives: a circle and a slab. These 2D objects are then sweeped along the vessels' centerline path to retrieve 3D regions finally, whereas a slab is always oriented parallel to the image plane. VG-regions can be specified by selecting a primitive and configurating parameters to control its' extend. Reasonably a circle region is specified by the radius. This can be a constant value or even a factor to modulate the radius according to the vessel radius, which is retrieved from the vesseltree-model. To provide a high degree of flexibility the slab can be configured with the distance to the front boundary (d_1) and the distance to back boundary (d_2), respectively. A interesting configuration

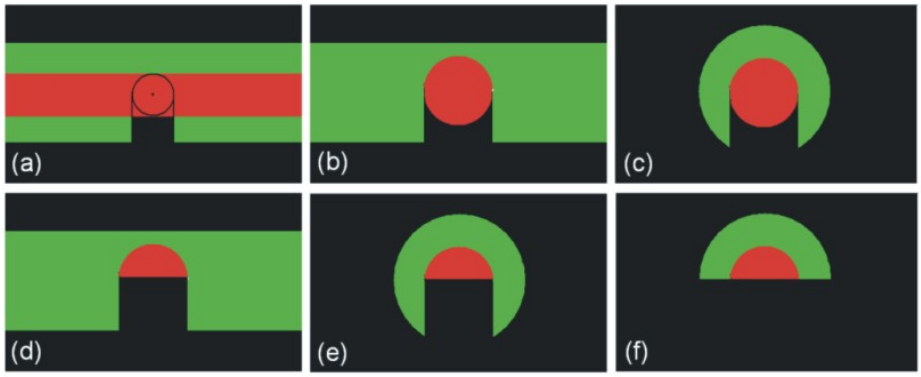


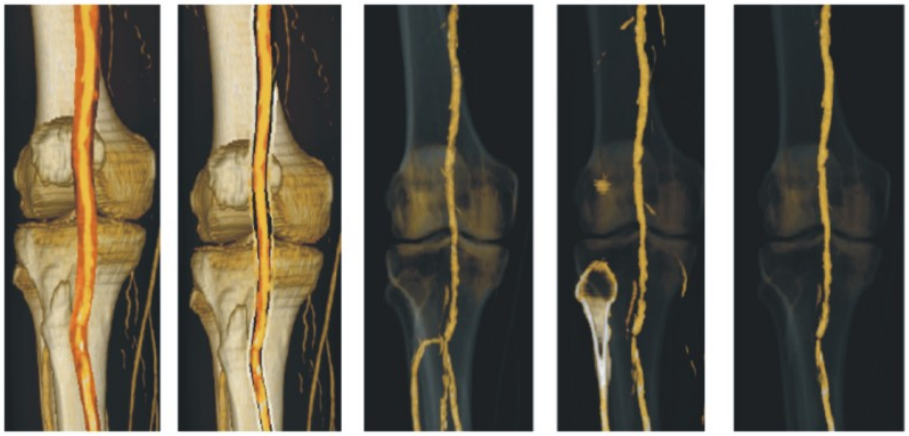
Figure 5.3: Proposals for foreground clipping: (a) shows slab regions for focus and context. The focus can be clipped for a circle-sized sub-focus region. Figure (b) and (c) show a focus-foreground cleft in the context regions. Figures (d),(e) and (f) propose extended foreground clipping related to the image parallel curved plane through the vessels centerline. Assuming an accurate centerline approximation, these approaches allow also to render an inside view of the main vessels.

would be to set d_1 to zero and clip the vessel along the centerline. If a total cutout of objects which occlude the focus region is desired, we propose e.g. to make a foreground cleft. Proposals for focus related foreground clipping are presented in Figure 5.3.

5.2.2 VesselGlyph DVR+CPR

The VG-DVR approach can be easily extended to a hybrid visualisation of DVR and CPR. This just involves to perform DVR with a cut-out of the focus region in image space. Accordingly CPRs are generated for the focus region only (see Figure 5.1 (c)). After registration of both output images with each other, they can be composited as textures. To make the visualisation more aesthetical the CPR images can be coloured with colours being harmonised with the colours appearing in the DVR image. Figure 5.4 shows some proposals how this approach could be configurated.

Figure 5.4: Proposals for hybrid rendering by compositing DVR (with focus cutout in image space) and focus only CPR: Figures (a) and (b) illustrate DVR rendering of the context region only and leaving a cleft according to the focus region. Figure (c) proposes a clefted focus and context DVR with slabs. The cut-out is related to an imaginated circle shaped focus region.



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

Figure 5.5: Images retrieved from a prototype and presented in [70]: (a) and (b) show CPR+DVR (symbol (a) in Fig. 5.4) with no transition or rather smooth transition (related symbols (e) and (f), resp., in Fig. 5.1), (b-d) show a pure DVR-VG visualisation (symbol (b) in Fig. 5.1). Only the regions definition vary: (c) shows a full-vessel rendering with a Tubular VG (symbol (c) in Fig. 5.2), (d) shows a half-vessel rendering with a thick slab VG (symbol (a) in Fig. 5.2 with $r_1 = 0$), (e) a half-vessel rendering with a Tubular VG (symbol (f) in Fig. 5.3)

5.3 Implementation of VesselGlyph DVR

5.3.1 Algorithm

Straka et al. [70] proposed to implement VG-DVR by applying a transparency modifier, which is assigned to every voxel of the density grid, and actually the proximity to the closest vessels centerline. This proposal has two main drawbacks: On the one hand only the opacity can be modified, this means that actually no fully independent classification (e.g. different colour mappings) can be applied for both regions. On the other hand the according algorithm involves a distance transform and the storage of an additional value for each voxel. This is not that economical when rendering datasets for peripheral CTA, which are very large, and causes long computational times [1]. However, this method provides the advantage to specify context and focus regions in arbitrary complexity, with even continuous transitions for instance.

Instead, we implemented the VG-concept with an image-space approach now, based straight-forward on the analytical representation of the VG. We propose to perform separate rendering by dividing actually the rays in context and focus regions. Hence this approach translates an analytical VG model to the ray-space by computing for each ray scalar intervals according to the VG-regions. Those intervals are related to the distance to the image plane (depth). Thus each ray object contains the information being needed to decide if the current sampling position is in a context or a focus region. The translation requires one additional pre-rendering step that computes the intersections of rays and region boundaries. Since our software renders only with viewpoints which allow a slice-parallel ray traversal, those intersections can be determined in 2D by solving simple equations analytically. To translate the VG to the ray-space, an algorithm follows the vessel path slice-by-slice and computes the intersections of rays traversing this slice with

[1] In [70], they reported 10 min with a non-optimised CPU algorithm and even 0.5 min with a GPU implementation

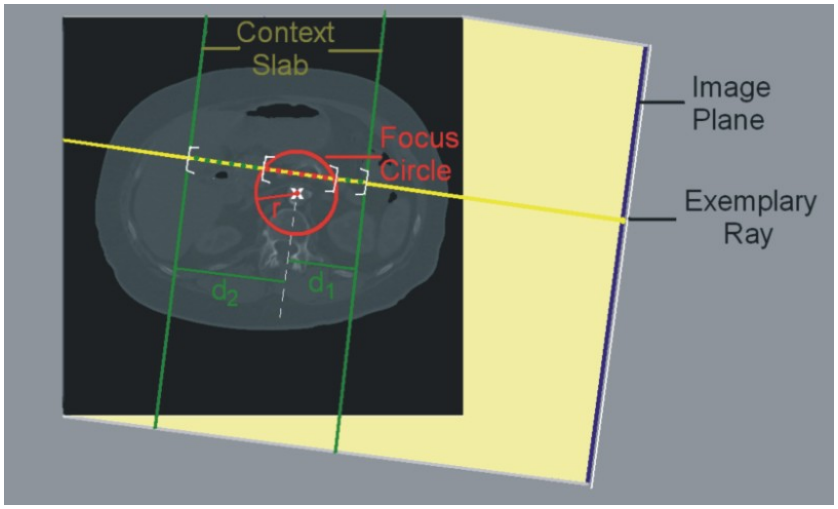


Figure 5.6: Example for a ray passing context and focus regions. In this instance the focus regions was specified as a sweeping circle having a radius of r. The context region was defined as a slab, which has a distance of d_1 to the front-side boundary and d_2 to the rear boundary related to the vessels centerline. The white brackets illustrate the scalar intervals to be stored in the ray structure.

this primitives in 2D space. The positions of the regions are determinated by the list of vessel center points belonging to the current slice. The size (radius,diameter) can be specified as constant value or as modulation factor of the vessel-radius according to the given vessel center point (as far it is available in the applied vesseltree model). For an image parallel slab region, all rays in one row have equal entry and exit depths when striking the slab. Thus the slab representing intervals are computed once (even in 1D) and copied for each ray in the same slice. Intersections with circle regions can be determined by the use of the quadratic formula. To eliminate unnecessary computations of ray-circle intersections, only those rays have to be evaluated which really strike the circle. These are only those rays that are closer than the circle radius to the centerline in the projection on the image plane. To avoid ambiguities due to multi path centerlines, a simplified kind of z-buffer algorithm is used to specify begin and end of regions even in presence of overlapping circles. Additional clips and cut-outs as proposed in Figure 5.3

can be implemented with the same trivial method as used for specifying a slab entry. A cleft requires just an additional selection of its' region in image space. Needless to say, this also a trivial problem.

5.3.2 Efficient Solution

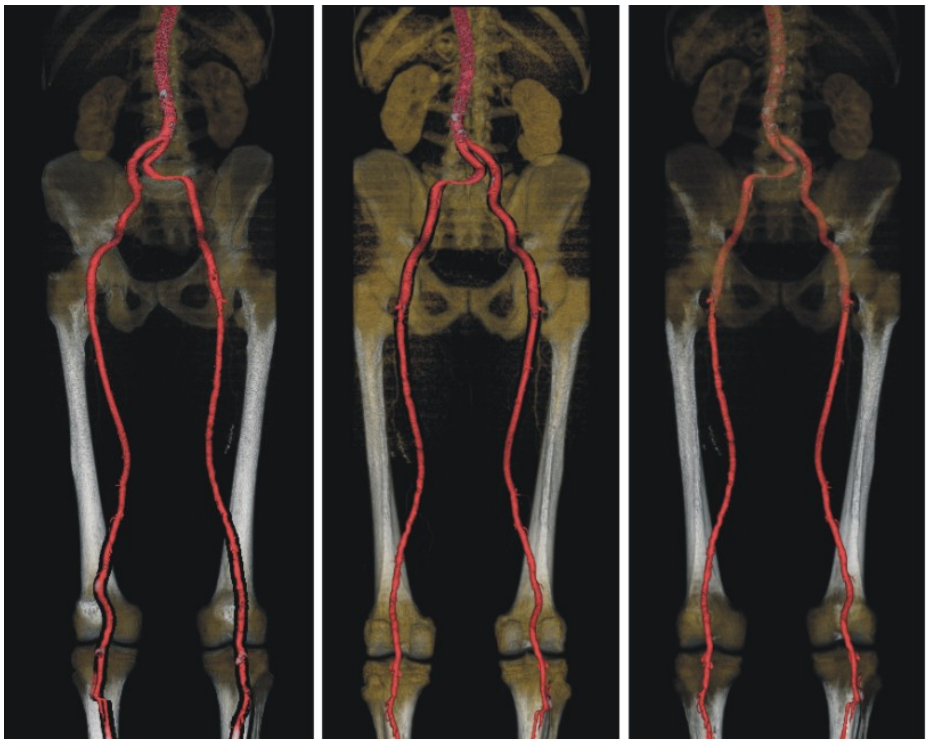
The first goal was to minimise the computational overhead being introduced by hybrid raycasting. This is mainly achieved by adapting the implementation in such manner that we can tap the full potential of the applied acceleration strategies of our conventional DVR design. The most modifications have to be done in order to support the empty space leaping by quadtree subdivision. Since VG-DVR enables to specify two TFs, we have to evaluate the visibility of certain subregions with respect to either one of them or both. Thus we introduced a procedure in the pre-processing pipeline, which assigns a flag to every block, which marks whether it potentially [2] contains a focus or context region, or rather both or non of them. This information is then used to build the visibility quadtrees of each block by employing the according TFs. Since VG concept allows to specify somehow the regions to be rendered, a lot of space in the volume may be skipped. Hence we aimed to speed-up raycasting of the VG by skipping the space outside the focus and context regions efficiently by excluding rays (in image space) and regions (in object space). To speed-up skipping of regions outside the VG, a function which evaluates whether a given rectilinear space is in- or outside the VG in a given slice. So is decided whether a quadtree sub-region has to be rendered or not. Unfortunately this information can not be pre-computed offline in any case, since slab-regions are changing according to the view direction. However, due to the simple geometries such a decision can be performed with a lowly quantity of comparisons, when the according logical decision tree is optimised. This kind of decision is also made for whole blocks, whereas also the

[2] When an image parallel slab is used, the slab orientation changes with view direction. This has to be taken into account when "predicting" the VG regions which may fall in a certain block in an offline pass.

region marking flag (mentioned above) is involved. The VG raycasting algorithm itself gets also not significantly more costly, since only few additional operations are introduced into its core. Those comprise just an evaluation whether the current sampling position is in the focus or context region. As this information is stored as simple 1D intervals within the ray-objects, this evaluation needs only a few ”>”- or”<”-comparisons.

5.3.3 Current State of Development

Our implementation of VG-DVR has honestly still the state of a prototype, since there still remain a few problems which have to be solved. For example the sweeping of a 2D primitive is performed with ignoring the vesselpath's direction, due to the simplifying assumption that all vessels run more or less in vertical directions. This approach is very efficient and easy to implement, but causes bad results when the vessels run still horizontally in some cases. In Figure 5.7 one can see such a case under the bifurcation of the abdominal vessel tree or below the knees. This deficiency causes that a small region of the main vessel is only included partially in the focus region. As an efficient solution to this problem, we think about a decision threshold according to the orientation of the vessel's normal, which let us treat only critical cases with a more costly, but correct sweep, performed orthogonally to the current directions along the vessel path. Another problem are the use of focus slabs which are not bounded on their left and right sides. They cause strange artefacts in bones which should rather belong to the context, but are intersected by the focus slabs. This is especially problematic in the case of multiple vessel paths. Better solutions therefor are also planned. Finally, the VG approach needs a lot further investigations to provide its' valuable usage. The next challenge is a suitable user interface to control the display of the VG interactively. This is a common challenge for all kinds of focus/context visualisation approaches and of major importance.



(a) (b) (c)

Figure 5.7: Images rendered with our current implementation by using a circular focus region with a constant radius of 15. Images (a) and (b) were obtained with a foreground-cleft. Image (c) was rendered with a very tranlucent TF and depicts the vesseltree even though the view direction is from the back side.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This work discussed and reported the implementation of a DVR-tool which renders angiographical images interactively. Adequate solutions were presented to accomplish the goals interactive behaviour and efficient rendering of high quality images. Interactive DVR enables the user to navigate through the dataset and find the best view direction. High-quality renderings are suitable for documentation and educational illustration. Our software is now able to render high-quality DVR-images for angiography in the magnitude of 2400x1000 pixels in a moderate time of 10 to 20 seconds. To achieve this efficiency, we designed an optimised software, which handles well in processing of large datasets as used in peripheral CTA. The optimisation was done by selecting a set of known acceleration techniques, adapting them to our implementation and evaluating the resulting effects. Further we decided to simplify our implementation to an one dimensional rotation, which is fully sufficient for the purpose of CTA. Due to this limitation we could apply special purpose acceleration techniques, as ray-templates and quadtrees instead octrees, and reduce the complexity of optimised ray-casting algorithms. Additionally, we presented an image centered approach to implement the idea of independent context and focus visualisation by the concept of VesselGlyph.

This new feature may be a valuable support for the purposes of angiographic visualisations. Of course user studies in clinical practice have to follow in order to assess its' clinical usefulness. However, we sketched a quite simple and low cost extension of our optimised DVR algorithm to perform independent context and focus rendering with the VG. Due to the geometrical simplicity of the slice-parallel ray-casting approach, it could be designed in a manner that it does not introduce time-significant complexity into the rendering process.

The presented approach is a pure software solution and thus provides a high degree of hardware independence. But it can not keep up to the efficiency of nowadays GPU-based implementations, although our implementation provides almost satisfying rendering times. Hence improving our software just with the goal to accelerate is of subsidiary importance for the purposes of technical research, since the problem of efficiency has been solved in fact more effectively with up-to-date GPU-based solutions. Rather one should focus on progressing particularly the visualisation potentials of DVR in the field of CTA. This might challenge to find new visualisation features and explore new purposes to employ DVR in CTA. This has to be considered, because today DVR is only applicable for illustration of the anatomical context and to provide easy interpretable images, what is mainly achieved by conveying just a photorealistic impression. Finally, in my opinion DVR has more potentials, which can be explored to find further fields of application in CTA. This view makes a pure software solution also more reasonable, since CPU-based ray-casting provides maximum flexibility to introduce new features directly.

6.2 Own Ideas and Proposals

Angiographic visualisation aims on depicting the major arteries, their flow lumen, their anatomical context and the collateral vessels. The first goals are met by conventional MIP,CPR and the presented DVR approaches includ-

ing VG-DVR. Still challenging is the visualisation of thin collateral vessels, which are not easily detectable, as contrast agents are less or not present and further uncertainties, due to partial volume effects and a low signal to noise ratio, are complicating this task.

Promising results were achieved by detecting tubular structures by Hessian filtering (see [6]). Hessian filtered datasets contain a probability for each voxel that it belongs to a vessel. These datasets require also a kind of visualisation. Therefor MIP is applied currently, but one could also adapt DVR for this purpose and test its' value for a visualisation of these probability volumes.

But also an economical extension, to enhance the visualisation of thin collateral vessel with DVR directly, would be valuable. Even MIP perform better in this task, as they do not rely on an one-dimensional classification. Classification actually controls what and how details are displayed with DVR. Using a straight-forward TF approach (density mapping) enables to visualise vessels being enhanced by contrast agents and having thus distinct densities to their surrounding tissues. But an one dimensional density-TF fails to depict vessel structures which do not have salient densities. A more sophisticated design of TF is required and approaches like data-centric or spatial feature extraction TF (Section 3.2.2) shall be considered. A data-centric TF design may allow a more specific characterisation of vessel structures, which have more discriminable properties when regarding their first and second order derivatives also. Including spatial features as classifiers means to involve the densities context into the classification. For instance such features could be a 3D-texture analysis (e.g. with a co-occurence matrix) or a filter which extracts local maximum densities.

Of course additional classifiers and features are costly to compute, but the VG concept enables to apply a costly on-the-fly classification only there where it is actually needed. So we are able to specify relatively small regions of interest to be rendered with costly TFs. Thus we can consider the value of VG-based rendering also under the aspect of efficiency. Focus and context

rendering allows to adapt the rendering costs to the importance of particular regions and to compensate costs by rendering the much larger regions of secondary interest with low-cost sampling and classification.

Bibliography

[1] A. Kanitsar, R.Wegenkittl, P. Felkel, D. Fleischmann, D. Sandner, and E. Gröller. Computed tomography angiography: A case study of peripheral vessel investigation. In *Proceedings of IEEE Visualization 2001*, pages 477-480, 2001.

[2] Dominik Fleischmann, Richard L. Hallett and Geoffrey D. Rubin. CT Angiography of Peripheral Arterial Disease. *Journal of Vascular Interventional Radiology*, 1 (17): 3-26 2006.

[3] Koechl A, Kanitsar A, Lomoschitz E, et al. Comprehensive assessment of peripheral arteries using multi-path curved planar reformation of CTA datasets. *Europ Radiol 2003*; 13:268 - 269.

[4] Armin Kanitsar, Dominik Fleischmann, Rainer Wegenkittl, and Meister Eduard Gröller: Diagnostic Relevant Visualization of Vascular Structures. *citeseer.ist.psu.edu/634416.html*.

[5] Armin Kanitsar, Dominik Fleischmann, Rainer Wegenkittl,Petr Felkel and Meister Eduard Gröller: CPR - Curved Planar Reformation. *Proceedings of IEEE Visualization 2002*, 2002.

[6] Matus Straka. Processing and Visualization of Peripheral CT-Angiography Datasets. PhD Thesis, Technical University Vienna, Institute for Computergraphics and Algorithms, 2006.

[7] Alexandra La Cruz. 3D Modelling and Reconstruction of Peripheral Vascular Structure. PhD thesis, Technical University Vienna, Institute for Computergraphics and Algorithms, 2006.

[8] M. Straka, A. La Cruz, A. Köchl, L. I. Dimitrov, M. Sramek, D. Fleischmann, E. Gröller. Bone Segmentation in CT-Angiography Data Using a Probabilistic Atlas. In *Proceedings of Vision*, Modeling and Visualization 2003 Conference, Munich, Germany 2003, p. 505-512

[9] M. Straka, A. Cruz, A. Köchl, M. Sramek, D. Fleischmann, E. Gröller. 3D Watershed Transform Combined With a Probabilistic Atlas For Medical Image Segmentation. In *Journal of Medical Informatics and Technologies*, Szczyrk, Poland, 2003, p. IT69-IT78

[10] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29 - 37, 1988.

[11] M. Levoy, Efficient ray tracing of volume data, *ACM Trans. Comp. Graph.*, vol. 9, no. 3, pp. 245-261, 1990.

[12] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. In *Proceedings of SIGGRAPH'87*, pp. 163-169.

[13] M. Meißner, J. Huang, D. Bartz, K. Mueller, and R. Crawfis. A practical evaluation of popular volume rendering algorithms. In *Proceedings of the Symposium on Volume Visualization 2000*, pages 81-90, 2000.

[14] P. Lacroute and M. Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. *Computer Graphics, 28(Annual Conference Series)*:451 - 458, 1994.

[15] C. Rezk Salama, K. Engel, M. Bauer, G. Greiner, and T. Ertl, Interactive volume rendering on standard pc graphics hardware using multitextures and multi-stage rasterization. In *Siggraph/Eurographics Workshop on Graphics Hardware*, pp. 109–119, 2000.

[16] R. Westermann and T. Ertl. Efficiently using graphics hardware in volume rendering applications. In *Proceedings of SIGGRAPH 1998*, pages 169-178, 1998.

[17] M. Meißner, U. Hoffmann, and W. Straßer. Enabling classification and shading for 3D texture mapping based volume rendering using OpenGL and extensions. In *Proceedings of Visualization 1999*: pages 207-214,1999.

[18] A. Van Gelder and K. Kim. Direct volume rendering with shading via three-dimensional textures. In *Proceedings of the Symposium on Volume Visualization 1996*, pages 23-30, 1996.

[19] Brian Cabral, Nancy Cam and Im Foran. Accelerated Volume Rendering and tomographic reconstruction using texture mapping. In *Proceedings of the 1994 IEEE Symposium on Volume Visualization*, pages: 91 - 98. ACP SIGGRAPH, 1994.

[20] Peter Schröder and Wolgang Krüger. Data parallel volume-rendering algorithms for interactive visualization. *The Visual Computer 9*: pp. 405 - 416, Springer Verlag 1993

[21] Taosong He, Lichan Hong, Arie Kaufman, and Hanspeter Pfister. Generation of Transfer Functions with Stochastic Search Techniques. In *Proceedings Visualization '96*, pages 227 - 234, 1996.

[22] J. Marks, and et al. Design galleries: Ageneral approach to setting parameters for computer graphics and animation. In *Computer Graphics (SIGGRAPH '97 Proceedings)*, pp 389 - 400, 1997.

[23] Shiaofen Fang, Tom Biddlecome, Mihran Tuceryan. Image-Based Transfer Function Design for Data Exploration in Volume Visualization. In *Proceedings of IEEE Visualization'98 Conference*, pp. 319-326, 1998.

[24] Jirí Hladuvka and Meister Eduard Gröller. Salient Representation of Volume Data. In *Proceedings of VisSym'01, Joint Eurographics - IEEE*

TCVG Symposium on Visualization, May 28 - May 30, 2001, Ascona, Switzerland, pages 203-211,351

[25] Jirí Hladuvka, Andreas König, Meister Eduard Gröller. Curvature-Based Transfer Functions for Direct Volume Rendering. In *Proceedings of Spring Conference on Computer Graphics and its Applications 2000 (SCCG 2000)*, Budmerice, Slovakia, May 3rd-6th, 2000, pp. 58-65

[26] Gordon Kindlmann and James W. Durkin. Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering. *IEEE Symposium on Volume Visualization*, 1998

[27] Penny Rheingans and David Ebert. Volume Illustration: Nonphotorealistic Rendering of Volume Models. *IEEE Transactions On Visualization And Computer Graphics*, Vol. 7, N0. 3, July-September 2001

[28] J. Patten and Kwan-Liu Ma. A Graph-based Interface for Representing Volume Visualization Results. In *Proceedings of Graphics Interface '98*

[29] Milos Sramek and Arie Kaufman. Fast Ray-tracing of Rectilinear Volume Data Using Distance Transforms. *IEEE Transactions on Visualization and Computer Graphics 3(6)*, pp. 236-252, 2000

[30] M. Sramek. Fast surface rendering from raster data by voxel traversal using chessboard distance. In *Visualization'94*, pages 188-195, October 17-21, 1994.

[31] Milos Sramek. Visualization of Volumetric Data by Ray Tracing. PhD Thesis, ISBN 3-85403-112-2, Austrian Computer Society, Austria 1998.

[32] Milos Sramek, Leonid I. Dimitrov. f3d - A File Format and Tools for Storage and Manipulation of Volumetric Data Sets. *1st International Symposium on 3D Data Processing, Visualization and Transmission*, pp 368 - 372, Padova, Italy, June 19 - 21, 2002.

[33] Mark Jones, J. Andreas Bærentzen, and Milos Sramek. 3D distance fields: A survey of techniques and applications. *IEEE TVCG 2006*, accepted.

[34] K.J. Zuiderveld, A.H.J. Koning, and M.A. Viergever. Acceleration of ray-casting using 3D distance transforms. In Visualization in Biomedical Computing II. *Proc. SPIE 1808*, pages 324-335, Chapel Hill, NC, 1992.

[35] S. R. Marschner and R. J. Lobb. *An evaluation of reconstruction filters for volume rendering*. In Proceedings of Visualization 1994, pages 100-107, 1994.

[36] S. Grimm, S. Bruckner, A. Kanitsar, and E. Gröller. Memory Efficient Acceleration Structures and Techniques for CPU-based Volume Raycasting of Large Data. In *IEEE/SIGGRAPH Symposium on Volume Visualization and Graphics*, pages 1 - 8, 2004.

[37] S. Grimm, S. Bruckner, A. Kanitsar, and E. Gröller. A refined data addressing and processing scheme to accelerate volume raycasting.*Computers & Graphics*, 28(5), 2004.

[38] Sören Grimm. Real-Time Mono- and Multi-Volume Rendering of Large Medical Datasets on Standard PC Hardware. PhD thesis, Technical University Vienna, Institut for Computergraphics und Algorithms, 2005.

[39] Stefan Bruckner. Efficient Volume Visualization of Large Medical Datasets. Master Thesis, Technical University of Vienna, Institute for Computer Graphics and Algorithms, 2004.

[40] Balazs Csebfalvi. Interactive Volume-Rendering Techniques for Medical Data Visualization. PhD-Thesis, Technical University of Vienna, Institute for Computer Graphics.

[41] K. R. Subramanian and D. S. Fussell. Applying space subdivision techniques to volume rendering. In *Proceedings of IEEE Visualization '90*, pages 150-159, 1990.

[42] J. Wilhelms and A. Van Gelder. Octrees for faster isosurface generation. *ACM Transactions on Graphics*, 11(3):201 - 227, 1992.

[43] B. Mora, J. Jessel, and R. Caubet. A new object order ray-casting algorithm. In *Proceedings of IEEE Visualization*, pages 107 - 113, 2002.

[44] L. Szirmay-Kalos. Theory of Three Dimensional Computer Graphics. Akademia Kiado, Budapest, 1995.

[45] H. Ray, H. Pfister, D. Silver, and T. A. Cook. Ray casting architectures for volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):210 - 223, 1999.

[46] H. Hauser, L. Mroz and E. Gröller. 2 level volume rendering - fusing MIP and DVR. In *IEEE Transactions on Visualization and Computer Graphics*, vol. 7 (3), pp 242 - 252, 2001.

[47] R. Yagel and D. Cohen and A. Kaufman. Normal Estimation in 3D Discrete Space. *The Visual Computer 8(5-6)*,pages 278 - 291, 1992.

[48] T. Möller, R. Machiraju, K. Mueller, and R. Yagel. A comparison of normal estimation schemes. In *Proceedings of Visualization 1997*, pages 19-26, 1997.

[49] L. Neumann, B. Csebfalvi, A. König, and M. E. Gröller. Gradient estimation in volume data using 4D linear regression. In *Proceedings of Eurographics 2000*, pages 351-358, 2000.

[50] Georgios Sakas, Marcus Grimm, and Alexandros Savopoulos. Optimized maximum intensity projection (MIP). In *Eurographics Rendering Workshop 1995*. Eurographics, June 1995.

[51] Michael B. Cox and David Ellsworth. Application-controlled demand paging for Out-of-Core visualization. In *Proceedings of Visualization '97*, October 1997, pp. 235 - 244.

[52] S. Parker, M. Parker, Y. Livnat, P.-P. Sloan, Ch. Hansen, P. Shirley. Interactive Ray Tracing for Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 5, No. 3, pp. 238 - 250, 1999.

[53] A. Law and R. Yagel. An optimal ray traversal scheme for visualizing colossal medical volumes. In *Proceedings of Visualization in Biomedical Computing 1996*, pages 43-52, 1996.

[54] James T. Kajiya. The Rendering Equation. *Computer Graphics, 20(4)*:143 - 150, 1986.

[55] James F. Blinn. Light reflection functions for simulation of clouds and dusty surfaces. *Computer Graphics, 16(3)*:21 - 29,1982.

[56] http://www-static.cc.gatech.edu/scivis/tutorial/tutorial.html. *Scientific Visualizations Tutorial* by Georgia Tech, Scientific Visualization Labratory.

[57] L. Westover. Footprint evaluation for volume rendering. *Computer Graphics, 24(4)*:367 - 376, 1990.

[58] M. Meißner, J. Huang, D. Bartz, K. Mueller, and R. Crawfis. A practical evaluation of popular volume rendering algorithms. In *Proceedings of the Symposium on Volume Visualization 2000*, pages 81-90, 2000.

[59] Meissner M., Grimm S., Strasser W., Packer J., Latimer D. Parallel volume rendering on a single-chip SIMD architecture. *Symposium on Parallel and Large Data Visualization and Graphics*, 2001. p. 107-13.

[60] D. Marr, F. Binns, D. Hill, G. Hinton, D. Koufaty, J. Miller, and M. Upton. Hyper-threading technology architecture and microarchitecture. *Intel Technology Journal*, 6(1):4-15, 2002.

[61] Craig M. Wittenbrink. Survey of Parallel Volume Rendering Algorithms. In *Proceedings of International Conference on Parallel Distributed Pro-*

cessing Techniques and Applications (PDPTA '98), Las Vegas, Nevada, July 1998.

[62] C.Wittenbrink, T. Malzbender, and M. Goss. Opacity-weighted color interpolation for volume sampling. *1998 Symposium on Volume Visualization*, pp. 135-142, 1998.

[63] K. Mueller, N. Shareef, J. Huang, and R. Crawfis, High-quality splatting on rectilinear grids with efficient culling of occluded voxels. *IEEE Transactions for Visualization and Computer Graphics*, vol. 5, no. 2, pp. 116-134, 1999.

[64] K. Mueller and R. Crawfis. Eliminating popping artifacts in sheet buffer-based splatting. In *Proceedings on Visualization'98*, pp. 239-245, 1998.

[65] H. Ray, H. Pfister, D. Silver, and T. A. Cook. Ray casting architectures for volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):210 - 223, 1999.

[66] Chandrajit Bajaj and Insung Ihm and Sanghun Park and Dongsub Song. Compression-Based Ray Casting of Very Large Volume Data in Distributed Environments. *The Fourth International Conference on High-Performance Computing in the Asia-Pacific Region-Volume 2*, p. 720, 2000.

[67] J. Danskin and P. Hanrahan. Fast algorithms for volume ray tracing. *Symposium on Volume Visualization Proceedings of the 1992 workshop on Volume visualization*, pp 91 - 98, 1992.

[68] Hanspeter Pfister. Moderne Volumen Visualisierung. *Information Technology, Heft 3/2004.*

[69] S. Card, J. Mackinlay and B. Shneiderman. Readings in Information Visualization: Using Vision to Think. *Morgan Kaufmann Publishers*, 1999.

[70] M. Straka, M. Cervenansky, A. La Cruz, A. Köchl, M. Sramek, E. Gröller, D. Fleischmann. The VesselGlyph Focus and Context Visualization in CT-Angiography. In *Proceedings of the IEEE Visualization 2004 Conference*, Austin, Texas, 2004, p. 385-392

[71] H. Hauser, L. Mroz, G.-I. Bischi, and E. Gröller. Two-level rendering. In *Journal IEEE Transactions on Computer Graphics and Visualization (IEEE TVCG)*, pages 242-252, 2001.

[72] J. Zhou, M. Hinz, and K. D. Tönnies. Hybrid-Focal Region Based Rendering Of Medical Data. In *Bildverarbeitung für die Medizin 2002*, Leipzig, pages 113-116, 2002.