TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

DISSERTATION

Semantic Visualization Mapping for Volume Illustration

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

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

eingereicht an der
Technischen Universität Wien,
bei der Fakultät für Informatik

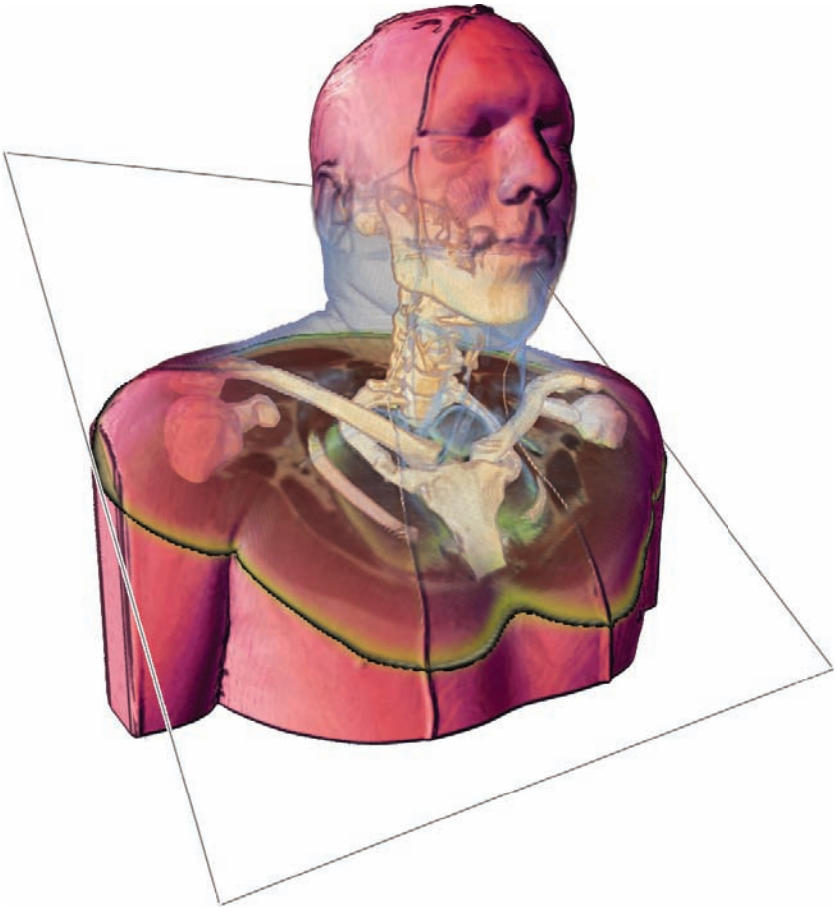
von

Dipl.-Ing. Peter Rautek
Matrikelnummer 9925695
Zehenthofgasse. 4/37
1190 Wien

Wien, im Dezember 2008

Semantic Visualization Mapping for Volume Illustration

DISSERTATION



Peter Rautek

supervised by
Meister Eduard Gröller
Institute of Computer Graphics and Algorithms
Vienna University of Technology

Kurzfassung

Das Gebiet der wissenschaftlichen Visualisierung beschäftigt sich mit der automatisierten Generierung von Bildern aus wissenschaftlichen Daten. Um relevante Informationen darzustellen, werden adäquate visuelle Abstraktionen benötigt. Visuelle Abstraktionen stellen gewöhnlich einen Kompromiss zwischen dem exakten Darstellen von Information und dem Verhindern einer visuellen Überlastung dar. Um visuelle Abstraktionen einsetzen zu können, wird eine Abbildung zwischen Datenattributen und visuellen Abstraktionen benötigt, die *Visualisierungsabbildung* genannt wird.

Diese Dissertation gibt einen Überblick über die Geschichte der visuellen Abstraktion und der Visualisierungsabbildung im Kontext der wissenschaftlichen Visualisierung. Danach wird eine neue visuelle Abstraktionsmethode - die *Karikaturistische Visualisierung* - vorgestellt. Das Prinzip der Übertreibung ist die in diesem Zusammenhang verwendete visuelle Abstraktionsmethode. Dieses Prinzip der Karikatur akzentuiert die markanten Details, während der Kontext nur schematisch dargestellt wird.

Die Abstraktionsmethoden, die in dieser Dissertation verwendet werden, sind von der visuellen Kunst insbesondere der traditionellen wissenschaftlichen Illustration inspiriert. Illustrationen sind gute Beispiele für handgezeichnete Visualisierungen. Allerdings ist die manuelle Anfertigung von Illustrationen sehr zeitaufwändig und verlangt umfangreiches künstlerisches Können. Um diese Techniken zu automatisieren, werden Algorithmen entwickelt, die einige Parameter zur Verfügung stellen, welche vom Benutzer eingestellt werden können. Im Rahmen dieser Dissertation ist eine Methode entstanden, die es ermöglicht, Semantiken explizit zu verwenden, um Abbildungen von Datenattributen auf visuelle Abstraktionen zu spezifizieren. Visualisierungsregeln können mittels semantischer Visualisierungsabbildung unter Verwendug von Domäne- und Visualisierungssemantik spezifiziert werden.

Das Verhalten der automatisch generierten *interaktiven Illustrationen* wird durch interaktionsabhängige Visualisierungsregeln festgelegt. Während Interaktionsmöglichkeiten wie die Manipulation der Blickrichtung den Standard in der Volumenvisualisierung darstellen, werden in dieser Dissertation umfangreichere Interaktionsmöglichkeiten vorgestellt. Das Verhalten der interaktiven Illustrationen wird von interaktionsabhängigen Regeln bestimmt, die in den semantischen Visualisierungsabbildungsansatz integriert werden.

Abstract

Scientific visualization is the discipline of automatically rendering images from scientific data. Adequate *visual abstractions* are important to show relevant information in the data. Visual abstractions are a trade-off between showing detailed information and preventing visual overload. To use visual abstractions for the depiction of data, a mapping from data attributes to visual abstractions is needed. This mapping is called the *visualization mapping*.

This thesis reviews the history of *visual abstractions* and *visualization mapping* in the context of scientific visualization. Later a novel visual abstraction method called *caricaturistic visualization* is presented. The concept of exaggeration is the visual abstraction used for caricaturistic visualization. Principles from traditional caricatures are used to accentuate salient details of data while sparsely sketching the context.

The visual abstractions described in this thesis are inspired by visual art and mostly by traditional illustration techniques. To make effective use of the recently developed visualization methods, that imitate illustration techniques, an expressive visualization mapping approach is required. In this thesis a visualization mapping method is investigated that makes explicit use of semantics to describe mappings from data attributes to visual abstractions. The *semantic visualization mapping* explicitly uses domain semantics and visual abstraction semantics to specify visualization rules. Illustrative visualization results are shown that are achieved with the semantic visualization mapping.

The behavior of the automatically rendered *interactive illustrations* is specified using interaction-dependent visualization rules. Interactions like the change of the viewpoint, or the manipulation of a slicing plane are state of the art in volume visualization. In this thesis a method for more elaborate interaction techniques is presented. The behavior of the illustrations is specified with interaction-dependent rules that are integrated in the semantic visualization mapping approach.

Contents

Preface **ix**

1 Introduction **3**
1.1 Visual Abstraction . . . 3
1.2 Visualization Mapping . . . 9
1.3 Overview of the Thesis . . . 13

2 Caricaturistic Visual Abstraction **17**
2.1 Introduction . . . 17
2.2 Related Work . . . 19
2.3 Caricature Space . . . 22
2.4 The Caricature Matrix . . . 24
2.5 Application Scenarios . . . 25
2.6 Results . . . 32
2.7 Summary . . . 34

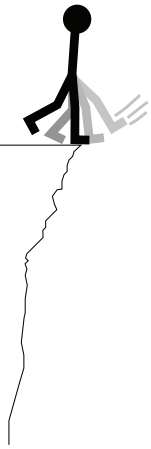
3 Semantic Visualization Mapping **37**
3.1 Introduction . . . 37
3.2 Related Work . . . 40
3.3 Overview of the Semantic Layers Concept . . . 42
3.4 Fuzzy Logic . . . 50
3.5 Rendering . . . 54
3.6 Results . . . 56
3.7 Summary . . . 60

4 Interactive Illustration **65**
4.1 Introduction . . . 65
4.2 Related Work . . . 68
4.3 Interaction-Dependent Semantics . . . 68
4.4 Fuzzy Logic on the GPU . . . 70
4.5 Flat Rendering . . . 73
4.6 Results . . . 77
4.7 Summary . . . 78

5 Summary and Conclusions **85**

Bibliography **86**

Curriculum Vitae **95**



However the work is a significant step backwards…

Anonymous Reviewer
IEEE Visualization

Preface

I dedicate this thesis to my girlfriend Anna and the tiny spot on the ultrasound image.
This thesis and the related publications are the outcome of my work at the Institute of Computergraphics and Algorithms, Vienna University of Technology, Austria. The welcoming atmosphere at the institute helped me to quickly get started with my work. For the hearty welcome and the nice working environment I want to thank the chair of our institute Werner Purgathofer, my supervisor Eduard Gröller, and also Sören Grimm, the supervisor of my diploma thesis.
My colleagues made the three years at the institute pass by very quickly. They provided me with a friendly and fruitful working atmosphere. I want to thank my colleagues Jean-Paul Balabanian, "Pope" Stefan "X" Bruckner, Ernesto Coto, Alexandra La Cruz, Raphael "der Bürger"- Fuchs, Martin "Bischt-Du-Da" Haidacher, Armin "moviestar" Kanitsar, Peter "the" Kohlmann, M^3 Muhammad Muddassir Malik, M^2 Matej "Mio" Mlejnek, Daniel "the Norwegian" Patel, Maurice "emty" Termeer, Ivan "Potaten" Viola, Erald(o) Vuçini, and many more visiting our group. I very much appreciated the stimulating discussions we had. I especially want to thank the *exvisation "taskforce"*: Eduard Gröller, Ivan Viola, and Stefan Bruckner. Without them this thesis would not have been possible. I also want to thank the Austrian Science Fund (FWF), that kindly supported the exvisation project (grant no. P18322).
Finally, I want to thank our beloved *Meister* (the researcher formerly known as Eduard Gröller), who unshakably believed in my work.

Thank you!

Peter Rautek

Vienna, December 2008

If you don't know where you are going,
any road will get you there.

Lewis Carroll

Introduction

Caricaturistic Visual Abstraction

Semantic Visualization Mapping

Interactive Illustration

Summary and Conclusions

In this Chapter the relevance of *visual abstraction* for scientific visualization will be highlighted. After motivating the need for *visual abstraction* techniques the necessity for expressive *visualization mappings* will be pointed out. Mapping data to *visual abstractions* in an expressive way is essential for advanced visualization methods. Lastly an overview of the thesis is given that relates the introduction to the remaining Chapters.
Parts of this Chapter were taken from a viewpoint paper on illustrative visualization in *ACM SIGGRAPH Computer Graphics Quarterly* [51] and an extended abstract published at the *Workshop on Knowledge-assisted Visualization* [52].

Chapter 1

Introduction

1.1 Visual Abstraction

In the eighties when supercomputers mass-produced data, the need arose for effective tools that aid human cognition for data exploration, hypothesis building, and reasoning. In 1987, the U.S. National Science Foundation Report "Visualization in Scientific Computing" [44] was published, stating the new challenges and proposing large-scale funding for scientific visualization. The field of visualization quickly started to evolve. The goal was (and still is) to generate images that show what is inside the data. Early attempts tried to establish a direct mapping between the data and optical material properties. For example, in volume visualization voxels are assigned color and transparency, and the simulation of light transport generates images following photorealistic principles. Since photorealism was an unquestioned paradigm of computer graphics, visualization also focused on the simulation of realistic light transport.

Later, when visualization was already a well-established field, people were questioning the sense of the simulation of light transport for the purpose of visualization. Photorealism in many cases prohibits the effective depiction of features of interest. Further, photorealism suggests that everything in the image is physically correct, potentially misleading the viewer. Visualization often illustrates aspects of the data and uses depiction techniques to convey information. For example shadows are used to enhance the perception of spatial relations between objects of interest. However using shadows in photorealistic imagery, for instance in a visualization of nano-scale structures such as nerve cells, might be misleading. Photorealism suggests that the image is physically correct and shadows occur the same way on nano-scale structures as on macro-scale structures, which is certainly not the case [46]. Using non-photorealistic techniques that inherently suggest that the depiction is illustrating the subject of interest helps in the correct interpretation of the imagery.

In Figure 1.1 two early examples of traditional scientific illustration can be seen. In Figure 1.1(a) sketches of Galileo Galilei are shown that illustrate scientific facts

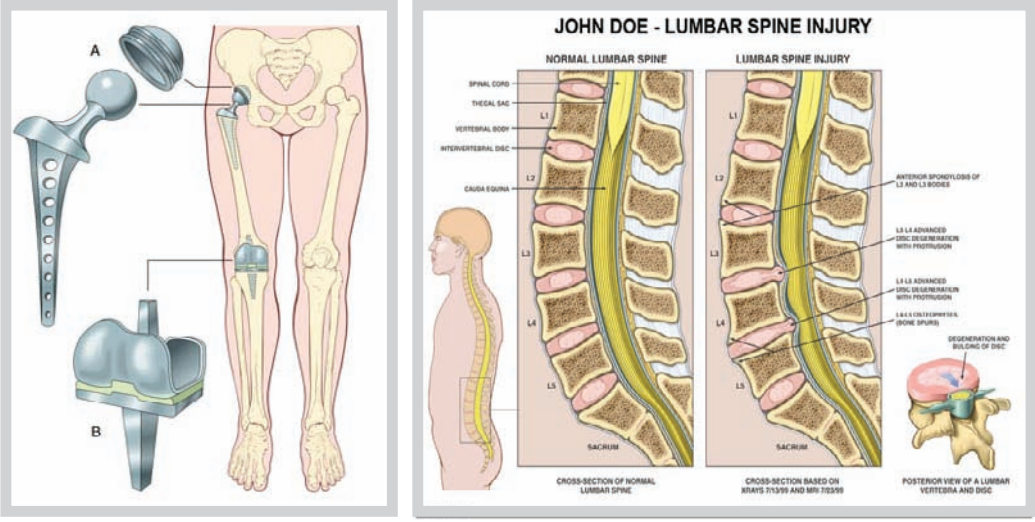
(a) (b)

Figure 1.1: *Examples of traditional scientific illustrations: (a) shows Galileo's sketches of the moon. They convey scientific information about different moon phases but do not focus on a realistic depiction of the moon itself. (b) shows Dürer's detailed illustration of a Rhinoceros that contains erroneous anatomical information.*

about the phases of the moon. Due to its sketchiness it also conveys the information that the moon itself is not the subject of this illustration. In Figure 1.1(b) a scientific illustration of Albrecht Dürer is shown that depicts a Rhinoceros which he never had seen himself. The anatomical details are not entirely correct (e.g. the small horn on the back of the Rhinoceros). Due to the very detailed depiction it was widely believed until the late 18th century to be an accurate representation of this species.

Since the strength of abstract visual representations was discovered, non-photorealistic rendering (NPR) models were adopted and used more frequently in visualization. NPR techniques are commonly inspired by artistic styles and techniques that do not focus on a realistic depiction of scenes and objects. They go beyond photorealism and are therefore free to express features that sometimes *cannot* - but more importantly *should not* - be shown using physically correct light transport. Visual art in general is an extensive source of visual abstractions that can be adopted for the purpose of scientific visualization. However, visualization has less artistic freedom than visual art. Visualization is bound to the depiction of the underlying data and cannot make use of all types of visual abstraction methods. The expressive way of depicting features of interest and the precision to accurately provide insight into the underlying phenomenon are requirements that are met by a discipline that is much older than visualization: it is traditional scientific illustration. Illustrators developed a toolbox with rendering techniques that depict

knowledge in an effective way. They carefully measure objects to correctly convey information about, and relations between features of interest.



(a) (b)

Figure 1.2: *Examples of modern scientific illustration: Illustration (a) shows shape and location of medical implants. An exploded view shows spatial relations and conveys assembly information. Illustration (b) shows a spine injury by comparing illustrations of cross sections of a normal and an injured case. Illustrations are by Prof. James A. Perkins [48])*

Two examples of modern illustrations are shown in Figure 1.2. Image (a) shows an example for an illustration of medical implants. The individual parts are relocated to show their location in the human body and to illustrate their assembly. The visual abstractions used in this illustration inherently point out that the object of interest is depicted in a non-veridical way. The human mind has outstanding abilities to correctly interpret this kind of visual abstractions. Illustration (b) in Figure 1.2 shows a spine injury. It illustrates the injured case by comparing it to a healthy spine. The caption of the injured spine illustration suggests that it is based on a real medical case and X-Ray and MRI images of a specific patient were used in the creation of the renderings.

Abstraction is the key that enables expressive rendering techniques beyond the depiction of reality. They are of immense value for the intuitive representation of phenomenas and the according data.

Information visualization researches the depiction of data without spatial correspondence. The abstract nature of this kind of data often inherently prevents photorealistic rendering. Therefore information visualization made use of visual

(a) (b)

Figure 1.3: *Examples of focus+context techniques: In image (a) regions of interest are drawn sharply while contextual regions are blurred [35]. In image (b) an example of uneven distribution of image space is shown [30].*

abstractions ever since. Focus+context techniques, that steer the viewers attention to the most important part of the image while still presenting context information, were explored.

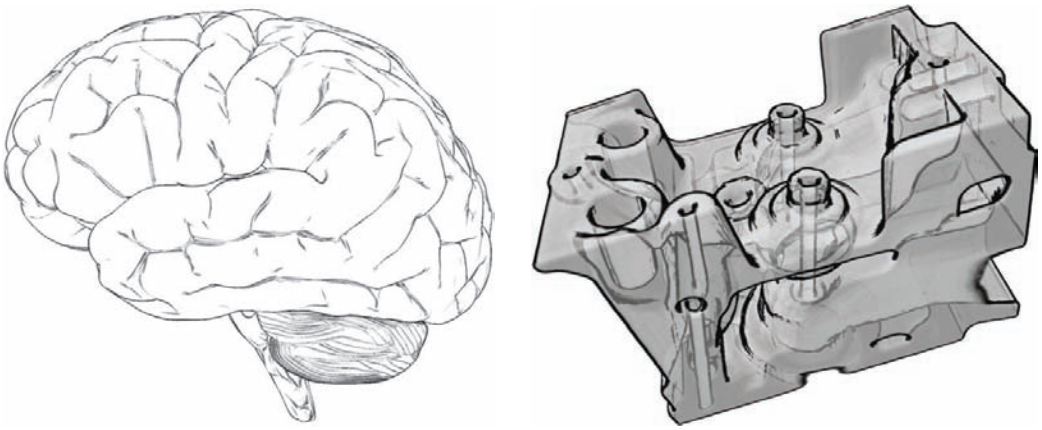
Examples from information visualization using focus+context include the semantic depth of field approach [35], where the sharpness of an object is related to its importance. Figure 1.3(a) shows an example where the most relevant features are drawn sharply, whereas context objects are blurred, to guide the observer's focus.

Another technique utilizing the focus+context concept is the magic lens metaphor. Magic lenses have been applied to a broad set of different data, such as maps [30], volume data [76], or abstract graphs [9]. In Figure 1.3(b) an example of a magic lens is shown that distorts a map to provide the most details at the point of focus. The overall principle of focus+context techniques is the uneven distribution of visual resources [22]. The levels of transparency, saturation, sharpness, or dedicated screen-space are typical examples of visual resources that are distributed among the features in the data. These examples from information visualization were early adopters of techniques that can be found in traditional illustration.

Flow and especially volume visualization, only recently started to explore the potential of illustration techniques. Subsequently, a vivid subfield (i.e., illustrative visualization) evolved, and algorithms were developed that automate the process of generating imagery using techniques from traditional illustration. These novel

approaches commonly utilize well-known methods that were carefully developed by traditional illustrators, and have shown to be very effective in conveying information. Traditional illustration techniques are a valuable source of inspiration for novel visualization methods. At the beginning, low-level visual abstractions, i.e., *how* to render features of interest, were explored. More recently, illustrative visualization deals with high-level visual abstractions or the question *what* to render. The low level visual abstraction techniques correspond to the tools of an illustrator. The toolbox is filled with drawing techniques like pencil, brush, or watercolor styles.

Line drawings are one of the most effective and difficult-to-master visual abstractions used in traditional illustration. In computer graphics, several techniques have been developed to represent mesh or volume data with feature lines to mimic hand drawn lines. These techniques are often based on local data properties such as first and second order derivatives. Typical representatives are ridge-and-valley lines [31], or view-dependent lines such as contours [56], suggestive contours [12], or apparent ridges [28]. In addition to line drawing, handcrafted shading techniques such as stippling [41], hatching, or toon shading [17, 37] have been simulated with computerized techniques. Examples of low-level abstractions are shown in Figure 1.4. Nowadays, computer-generated stylized depictions provide good results that are similar to but still distinguishable from handcrafted illustrations [43].



(a) (b)

Figure 1.4: *Examples of low-level visual abstractions: In image (a) a line rendering of a human brain is shown [12]. In image (b) a rendering of the engine dataset is shown using contours and cartoon shading.*

Low-level visual abstractions, such as those mentioned above, have been the primary focus of the NPR research. In addition to these low-level techniques, the illustrator works with expressive techniques that change the layout or deform

features to increase the communicative intent of the illustration. Expressive techniques such as cutaways, breakaways, close-ups, or exploded views relate to the focus+context concept. We refer to illustration-inspired focus+context techniques as high-level visual abstraction techniques. They map knowledge about the features to specific depictions, such that features are emphasized or suppressed according to their importance and the intent of the illustration.

Figure 1.5: *Examples of high-level visual abstractions: Image (a) shows an importance-driven rendering [75]. In image (b) an exploded view rendering is shown [7].*

High-level visual abstractions have been outside the main NPR research direction. These techniques usually require relevance information about the data. They have been in the main scope of illustrative visualization research as a continuation of the focus+context techniques. For instance Viola et al. [75] use an importance function as steering mechanism for the definition of view-dependent interactive cutaways. Other more explicit ways of focus definition have been used for interactive cutaways [5, 36], close-ups [6, 68], exploded views [7], or peel-aways [10] for volume data. In Figure 1.5 two examples for high-level techniques can be seen. Figure 1.5(a) shows a view-dependent interactive cutaway of the Gecko dataset and Figure 1.5(b) shows an exploded view of the Turtle dataset. Both illustrations allow a view on the interior parts of the depicted objects. The focus is defined either by a geometric region (e.g., sphere or cube) or by a segmentation mask. These approaches are also referred to as smart visibility techniques [73]. They can be seen as interactive focus+context methods, which have their static predecessors in traditional illustration.

In recent years many of the traditional illustration techniques were adopted for the purpose of direct volume visualization. The possibilities to map data to these visual abstractions are increasingly complex. Therefore novel ways to specify visualization mappings are needed.

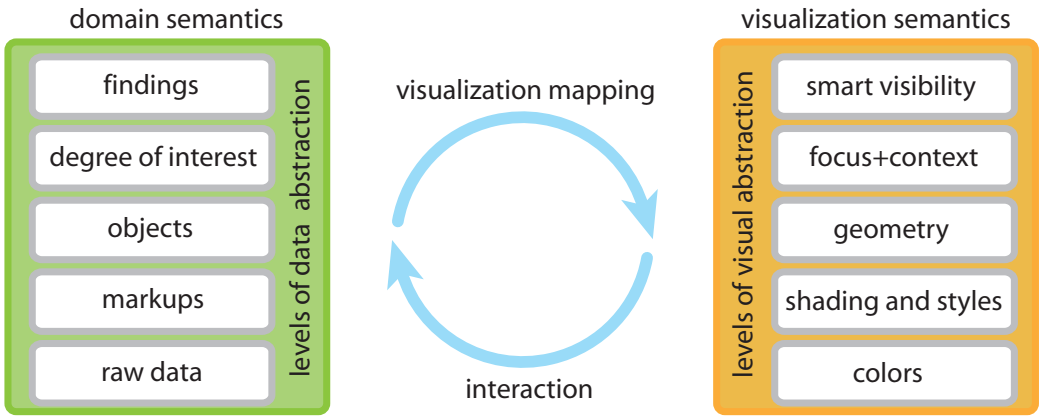


Figure 1.6: *Visualization mapping: Different levels of data abstraction are mapped to different levels of visual abstractions.*

1.2 Visualization Mapping

Visualization is the discipline dealing with the depiction of data. The assignment of visual abstractions to the data is referred to as the *visualization mapping*. Earlier examples of visualization mappings are described in Edward R. Tufte's book *Visual Explanations* [71]. During the Colera epidemic in London in 1854 Dr. John Snow tried to find out its cause. He suspected a relation between water wells and deaths from Colera. Testing the water did however not unveil suspicious impurities. Plotting cases of death on a map he found that one water pump was surrounded by a cluster of deaths. After further investigation he concluded that the one water pump was infected and the handle of the pump was removed. This instructive example of an early visualization mapping where data was manually mapped to visual abstractions for visual reasoning is however rather an exception. Visualization mappings were more often used to illustrate known facts than for visual reasoning.

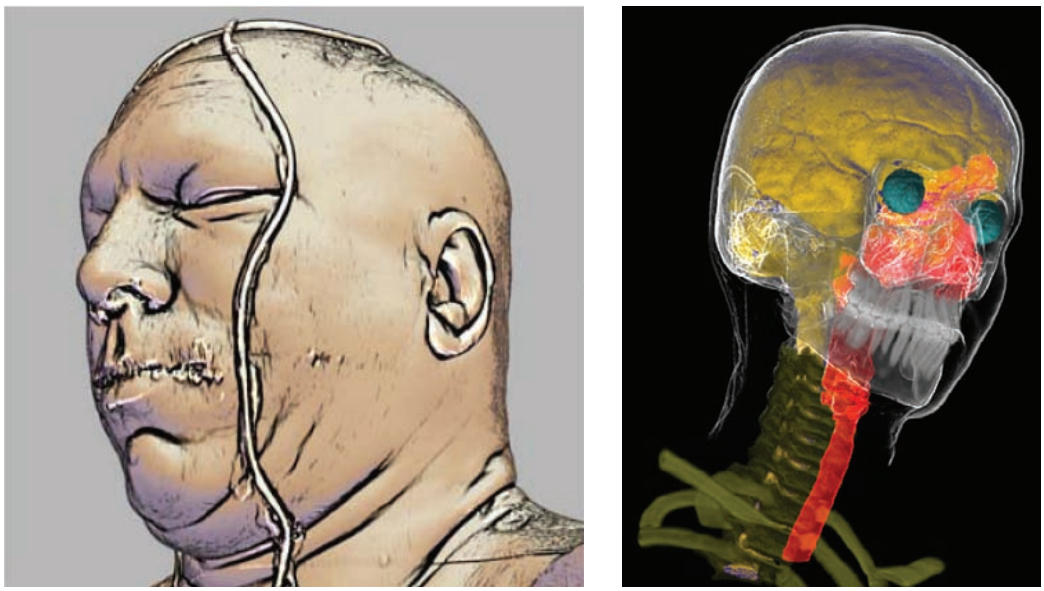
With small amounts of data often a simple examination of a table of numbers is sufficient. Therefore only when advanced measurement devices and supercomputers where available the overwhelming amount of data made it necessary to automatically map data to visual abstractions. Large tables filled with numbers representing the result of simulations or measurements needed to be mapped to visual abstractions. The examination and analysis of the data in many cases proofed to be more effective using expressive visualization mappings.

Figure 1.6 shows the general visualization mapping scenario. Different levels of data abstractions are mapped to different levels of visual abstractions. Conceptually, raw data (i.e., the lowest level of data abstraction) has no additional information available apart from the measured or simulated values and a descrip-

tion of the data types. An example for low level data abstraction is the scalar volumetric data obtained from computed tomography (CT), where each voxel defines a tissue density value. To define gradient or curvature information various image processing filters for smoothing and noise-reduction as well as other local operators can be applied. If another scan from the same spatial region is available from a different modality the data sets can be related using registration. Such data enhancements give more insights about the underlying raw data. These data-near semantics are referred to as markups in Figure 1.6. Unlike data markups the next level of data abstraction introduces domain specific semantics. For instance acoustic echo measurements are used for seismic exploration as well as in the medical domain. Performing filtering and shape analysis on acoustic echo data sets (from different domains) results in the same markups. However, for each domain these markups have a distinct meaning. Markups in the seismic domain identify geologic layers and faults, whereas in the medical domain the same markups identify vascular structures or organ boundaries. Such meaning is only derivable when models from a specific expert domain are introduced in the description of data. Faults, seismic layers, organs, or vessels are domain-specific concepts that define higher-level semantics. These objects are (unlike markups) coined terms in the respective domain. Object semantics can be derived with model- or atlas-based segmentation methods, or through a combination of markups that define the features of interest.

Data acquisition is usually motivated by a specific need of the respective domain. For instance, medical imaging is carried out to perform diagnosis or to identify the best treatment method. Each procedure looks for specific features and for relations to other features. For example, when planning tumor removal from the neck, the muscle also has to be dissected if the tumor tissue or metastatic lymph nodes are too close to a muscle. Vascular structures, however, must not be dissected. From such objective specifications, information about the importance of features from a neck CT scan is extracted. A degree of interest describes the relevance of objects, markups, or raw data. The degree of interest function, as a high-level semantics, can be defined as abstraction of particular domain procedures and gives information about structures the analyst wants to investigate in order to draw conclusions. The outcome of the visual analysis procedure and the according interpretation of raw data, markups, objects, and degree of interest specifications is the highest level of data abstraction. In Figure 1.6 this level of abstraction is labeled findings. Visualization in general is a tool to derive facts and findings from raw data. The data is mapped on visual abstractions. The user can interact with the visual abstractions to derive higher levels of data abstractions. At the end of the interactive process the user concludes the findings either in mind or a written report.

The common approach to model a visualization systems is to provide a set of parameters that control the visualization algorithm specifically tailored for the



(a) (b)

Figure 1.7: *Visualization mapping examples: Image (a) is a mapping from the raw data and the data markup curvature to visual properties [31]. Image (b) is a mapping from volumetric objects to individual visualization methods [19]. Tone shading (for the brain), contour enhancement (for the skin), shaded DVR (for the eyes and the spine), unshaded DVR (for the skull, teeth, and vertebrae), and MIP (for the trachea) are the used rendering techniques combined in this image.*

given type of data. The data attributes are then mapped to these visualization parameters. One of the most commonly used visualization mapping methods in volume visualization are transfer-functions [39]. A transfer function maps data attributes of a 3D volume such as density to visual representations such as color and opacity. Visualization mappings using higher order derivatives of the data are described in the work of Hladůvka et al. [25] and Kindlmann et al. [31]. In both works a markup (i.e., the curvature) is computed from the raw data and used for visualization mapping. In Figure 1.7(a) an example of a visualization mapping is shown that maps the markup *curvature* to the visual abstraction *contour*. A mapping from more than one attribute to visual properties is called a multi-dimensional transfer function [32]. Multi-dimensional transfer functions are typically used to map raw data from multiple imaging modalities or a combination of raw data and markups to visual abstractions. Hauser and Mlejnek [23] presented an approach using a degree-of-interest-function that maps the focus of the user to optical properties of 3D flow data. The degree-of-interest-function introduces

domain semantics into the visualization mapping process. These approaches are all examples of visualization mappings, that map different levels of data abstraction to visual abstractions. Transfer functions are a valuable approach to describe these different kinds of mappings. A more advanced visualization mapping approach was presented in the work of Hauser et al. [24] and Hadwiger et al. [19]. *Two-level volume rendering* was introduced to individually control the appearance of volumetric objects. In this approach different visualization strategies (such as direct volume rendering, maximum intensity projection, contour enhancements, etc.) are used for the visualization of segmented objects. This allows the combination of multiple visualization techniques in one rendering. Figure 1.7(b) shows objects that are rendered with individual visualization methods. Tone shading is used for the brain, contour enhancement for the skin, shaded DVR for the eyes and the spine, unshaded DVR for the skull, teeth, and vertebrae, and MIP rendering for the trachea. In this example the visualization mapping is controlled by object semantics.

One example of a knowledge presentation framework is described in the work of Wohlfart and Hauser [78]. They show a story telling application that provides a user interface for authoring and presentation of visualization stories. This approach is an example of a visualization mapping dealing with the highest level of data abstractions. The stories are interactive guides that direct the user attention towards the facts and findings previously discovered in the dataset.

These examples of visualization mappings on different levels of abstraction come from visualization literature. Other approaches also focusing on the specification of the visualization mapping are described in computer graphics literature. For instance Seligmann and Feiner [62] present a system for geometric objects

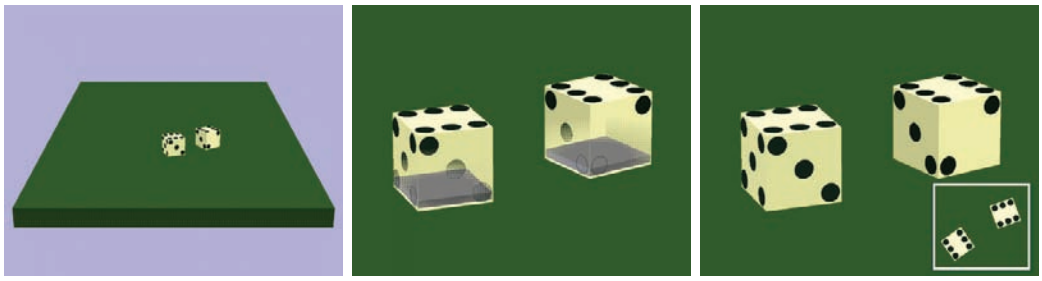


Figure 1.8: *Three renderings with different communicative intent of the same scene [63].*

that uses design rules to achieve the intended visualization. The communicative intent is formalized and mapped to stylistic choices. In Figure 1.8 three different renderings of a simple scene are shown. The communicative intent of Figure 1.8(a)

is to show the position of the dice in the scene. Figure 1.8(b) uses a ghosting technique to show the location of the weights in the dice. The communicative intent of Figure 1.8(c) is to show the dice and the value of the roll. This beautiful example demonstrates how the visualization changes if the communicative intent changes. The visualization mapping in this work is specified using design rules.

The work of Coyne and Sproat [11] is also an approach mapping data to the image domain using a linguistic description of scenes. The *WordsEye* system

Figure 1.9: *Comparison of a 1^{st} grade homework and the result of the WordsEye system [11].*

interprets a textual description and uses a large database of geometric objects to generate the described scene. In Figure 1.9 two interpretations of a scene description are compared. The left image shows the interpretation of a 1^{st} grade pupil and the right image shows the interpretation of the system.

The wide variety of visualization mapping methods in literature indicates the importance to map data representations from different levels of data abstractions to different levels of visual abstractions in an expressive way.

1.3 Overview of the Thesis

In this Chapter the importance of *visual abstractions* and the necessity of expressive and flexible *visualization mapping* methods was discussed. Visual abstractions help to make large amounts of data easier to examine and more intuitive to interpret. In Chapter 2 an example of a high-level visual abstraction method is investigated. Principles from traditional caricature are discussed and their applicability to scientific visualization is researched. Chapter 3 deals with a visualization

mapping approach that explicitly uses semantics from the data as well as from the visualization domain. The semantic visualization mapping approach is explained and application examples are shown. The visual abstractions that are used in this method are inspired by traditional technical and medical illustrations. In Chapter 4 the interactive behavior of illustrative visualizations is discussed. Unlike the static nature of traditional illustrations, illustrative visualizations have the ability to interact with the viewer. A semantic visualization mapping approach is presented that allows the definition of illustration behavior.

The capacity to be puzzled is the premise of all creation, be it in art or in science.

Erich Fromm

Introduction

Caricaturistic Visual Abstraction

Semantic Visualization Mapping

Interactive Illustration

Summary and Conclusions

In this Chapter an example of a high-level visual abstraction method is presented. *Caricaturistic visual abstraction* follows the principle of *exaggeration* found in traditional caricature to accent salient details of data. Caricatures are pieces of art depicting persons or sociological conditions in a non-veridical way. In both cases they exaggerate the deviations from a given reference model. The aim of caricaturistic visualization is an illustrative depiction of the characteristics of a given dataset by exaggerating the deviations to a reference model. The *Caricaturistic Visualization* approach was published in the *IEEE Transactions on Visualization and Computer Graphics* [53].

Chapter 2

Caricaturistic Visual Abstraction

2.1 Introduction

The high popularity of caricatures indicates the widespread ability of humans to identify outstanding features of faces. In addition, caricaturists have the ability to exaggerate these features and draw hyperbolized pictures. The exaggeration of features takes place in dependence to a reference model in the caricaturist's brain. A beholder of a caricature can interpret its meaning only if he has a similar reference model in mind. In Figure 2.1 an example of a reference model, the subject and the caricature of the subject are shown. The reference model can be seen as an idealized model within the domain of subjects. Each specimen within the domain is characterized by deviations to the reference model. The deviations of the specimen are the features of interest for the caricaturist. The caricature is the outcome of a hyperbolized depiction of the deviating features. It accents the essence of the depicted subject.

In Redman [54] the caricaturist is advised to differentiate between exaggeration and distortion: *"Exaggeration is the overemphasis of truth. Distortion is a complete denial of truth"*. Caricatures exaggerate but do *not* distort the deviations. This fact makes caricatures interesting for the purpose of visualization. The goal of traditional caricature is the entertainment of the beholder. For the purpose of visualization we follow the same principles but alter the goal. The aim of caricaturistic visualization is to accent the characteristics of the depicted object.

We found that many properties of caricatures correspond to specific techniques of illustrative visualization. Caricatures therefore provide a powerful metaphor in the context of visualization. We list some of the properties that are shared in visualization and traditional caricature.

Focus+Context techniques provide the user with detailed information at the focus of interest while the context is still present. Good caricatures accent the characteristics and salient details while sparsely sketching the context. The focus in caricatures is on the characteristics of the depicted object which are often the details of interest.

reference model specimen caricature

Figure 2.1: *Example of a non automatic caricature drawing: In the left image the head of Michelangelo's David statue is shown as an idealized model among the subjects of interest. In the middle image a specimen is shown. In the right image a hand drawn caricature of the specimen is shown. A caricature presumes the existence of a reference model.*

Effective Communication of Visual Content is a desired property commonly achieved by choosing good visual representations. Caricatures are expressive depictions of the content of interest simultaneously avoiding the depiction of details which are not of immediate interest. Therefore caricatures are well suited for the communication of visual content.

Augmentation of Images aids the viewer to correctly interpret the image. The augmentation is a descriptive visual information sparsely overlaid but not occluding the image. Therefore sparse visual representations are necessary to augment images. Caricatures are often line drawings which are extremely sparse representations and are therefore suitable for the augmentation of images.

Steering Attention to regions of interest is commonly done by visual cues. Caricatures provide intensive cues toward the details of interest. Highly exaggerated regions attract the user's attention. In contrast photorealistic rendering often fails to direct the attention to the focus of relevancy. Therefore caricaturistic visualization is especially suitable for datasets where deviations to a reference model are of interest.

Caricatures have many properties that are desirable in a wide variety of visualization applications. We give some ideas of potential application scenarios for caricaturistic visualization:

Quality Control aims to find subtle differences of workpieces to the reference model. Irregularities of surfaces are of immediate interest. The visual exaggeration of such irregularities leads to clear visible cues to the regions of interest.

Comparative Biology is concerned with the evolution and changes of species

over time as well as with the differences between species. Caricaturistic visualization helps to make the subtle differences visible even for lay persons and could for example also be used in education.

Case-based Education deals with learning by examination of different cases. Medical students for example have to learn different cases of diseases. Each case is a deviation from the reference model. By exaggerating these small deviations the learning process could be aided. The same approach can be used for patient communication. The patient as a layperson often fails to see the abnormalities in the data. Illustrative visualizations accenting the deviations can aid the patient to understand the diagnosis.

Deformation Surveillance is used to detect changes of objects over time. Small deformations are measured to estimate further deformations. For example the deformation of facades is monitored in order to guarantee the safety of a building. Caricaturistic visualization is able to exaggerate these deformations in order to make them visible and easily detectable.

The remainder of this Chapter is organized as follows: In Section 2.2 we briefly discuss related work. We derive a mathematical formulation of a feature and provide some simple guidelines for the design of features. In Section 2.3 we further illustrate the idea of caricaturistic visualization with some examples of simple caricaturistic operations using the provided mathematical framework. In Section 2.4 we present the *caricature matrix*, a technique for the visualization of divergences of datasets to each other. It is based on the caricaturistic visualization metaphor and exploits the feature based approach of caricaturistic visualization. In Section 2.5 we describe the implementation of application scenarios for caricaturistic visualization. We give ideas about feature design and a user interface for feature specification. In Section 2.6 we present the results of our caricaturistic visualization system and show examples of visual representations that are suitable for caricaturistic visualization. In Section 2.7 the Chapter is summarized and conclusions are drawn.

2.2 Related Work

Related work to this paper mostly focuses on facial caricatures. Computer aided facial caricature generation was addressed in several previous works [1, 2, 4, 57, 65]. The perception and recognition of faces in association to caricatures was an extensive subject of research [3, 20, 47, 57, 58, 59, 65]. While some works [3, 57, 58, 59] report an advantage in recognition or learning using facial caricatures, other works [20, 47] found no evidence that caricatures of people are better than photographs. Gooch et al. [18] present a more extensive discussion about human facial illustration and an evaluation of caricature techniques for face illustration.

For objects in general it was reported [15, 60] that stylized, accentuated drawings are more easily identified. They aid learning more than photographs of the same objects.

The work dealing with illustrative volume visualization focuses on imitating traditional illustration techniques. High level abstraction techniques as presented in the work of Viola et al. [74, 75] and Svakhine et al. [67] control the appearance of different features at varying degrees of sparseness and complexity. An illustrative visualization approach for time varying data was presented by Joshi et al. [27].

Weigle and Taylor [77] present a related technique for the comparison of different datasets. They investigate visualization techniques for intersecting surfaces and compare the performance of existing techniques and a novel glyph based approach. Wynblatt and Benson [80] present visual representations of web pages called caricatures. The caricaturization of web documents allows for fast browsing through a large number of documents. Liu et al. [40] present an approach to make subtle motions in video scenes clearly visible. The motions are accentuated by exaggerating the motion of objects in the video.

2.2.1 Feature Exaggeration

Caricaturists identify features and exaggerate certain properties of these features such as spatial extent, displacement, or angularity. We want to exaggerate the deviations of a specimen from the corresponding reference model. Therefore we measure the difference between the model and the specimen for each property. For example the displacement of the specimen's ear relative to the ideal model is a typical property in facial caricature.

2.2.2 Mathematical Framework

For each property we define a difference function over the domain of the property. The domain of property i is denoted as P_i and the difference function is denoted as $\ominus_i$. In a facial caricature a typical property is the angular offset of the ear to the reference model. Jug ears have a high value for the angular offset property while tight-fitting ears have a value close to zero. The defined domain of the angular offset of the ears could for example be $P_i = \{x | x \in (0, \frac{\Pi}{2})\}$. The difference operation for two values of P_i is the difference between the two angles.

Another example of a property is the three dimensional position of the ear. The domain of this property is a specific subspace of three dimensional space $P_i \subset \Re^3$. The distance measure for the position of the ear is simply the Euclidean distance.

A feature describes the characteristics of the specimen with respect to the reference model. A feature is therefore defined as a property vector. The property

vector space is defined as

$$P = P_1 \times P_2 \times \ldots \times P_{n-1} \times P_n \tag{2.1}$$

In the analogy of facial caricature a possible feature would be the ear given by its position, angular offset and spatial extent along its major axis. We define an exaggeration function for each property of the feature. This function describes the behavior of a feature as its properties are exaggerated. It is desirable that the deviating properties of the feature are even further deviated. In terms of facial caricatures the displacement of the ears would lead to even further displacement. We call this kind of exaggeration of a property *intra property exaggeration*. In contrast to that an *inter property exaggeration* is the exaggeration of a property caused by the deviation of another property. In the above example the inter property exaggeration of the displacement of the ears would also lead to an exaggeration of the scaling of the ears (i.e., the increase of the spatial extent of the major axis). We therefore define the exaggeration function e_i for property i as:

$$e_i(x_i, \delta) = x_i + (c_{i1}d_1(x_1, \widetilde{x_1}) + \ldots + c_{in}d_n(x_n, \widetilde{x_n}))\, \|x_i \ominus_i \widetilde{x_i}\|\, \delta \tag{2.2}$$

where δ is the exaggeration parameter, d_j is the distance function for property j, $\widetilde{x_j}$ is the value of the reference model for the property j, $c_{ij} \in \Re^+$ for $i, j = 1 \ldots n$ are the coefficients describing the inter and intra property exaggeration, and $\|x_i \ominus_i \widetilde{x_i}\|$ is given by

$$\|x_i \ominus_i \widetilde{x_i}\| = x_i \ominus_i \widetilde{x_i} \frac{1}{d_i(x_i, \widetilde{x_i})} \tag{2.3}$$

where $x_i, \widetilde{x_i} \in P_i$. The coefficient c_{ij} determines the influence of the deviation of property j on the exaggeration of property i. Intra and inter property exaggerations can be observed in real caricatures. In our approach we focus on intra property exaggerations. We therefore set all coefficients $c_{ij} = 0$ for $i \neq j$.

2.2.3 Guidelines for Features

Each feature consists of a set of properties. Simple features may only consist of few properties like position, orientation and elongation. More complicated features may consist of hundreds of properties describing the shape of the feature. Designing appropriate features is crucial for caricaturistic visualization. We designed our features to meet the following constraints:

Flexibility The set of properties is able to describe a wide variety of features.

Simplicity Each property is easy and fast to specify. Features which are complicated to specify may distract the user. Following the constraint of simplicity is not a restriction to the complexity of the feature. The automatically generated shape may be complicated while the user only specified few settings.

Measurability Each property is measurable and has a corresponding distance function. A pair of corresponding features differs only in the specified values of the properties. The distance between these values must be measurable.

While the first two constraints are guidelines to design good features the third constraint is a technical prerequisite for the caricaturization of features. The flexibility and simplicity constraints seem at first glance to result in a trade-off. On one hand the features should have the flexibility to describe the subject of caricaturization, on the other hand it should not be too complicated for the user to specify. To meet both constraints we propose to use automatic or semi-automatic approaches. In Section 2.5 we show an example of an automatic approach as well as examples of semi-automatic approaches.

2.3 Caricature Space

Based on the above framework we illustrate the idea of caricaturistic visualization and show an example of the caricature space. For the purpose of demonstration we define a three dimensional superquadric which is given by the implicit function

$$f(x,y,z) = \left(\frac{x}{s_x}\right)^{\frac{2}{\gamma}} + y^{\frac{2}{\gamma}} + z^{\frac{2}{\gamma}} \tag{2.4}$$

We define $s_x; \gamma \in \Re^+$ to be the properties of the implicit function. The property vector space $P = P_1 \times P_2$ of the implicit function is therefore defined as $\Re^+ \times \Re^+$. As a reference model we choose the superquadric with the property vector (1,1) which is a sphere. We define eight deviating objects with all combinations of the properties $s_x = 0.8, 1.0, 1.2$ and $\gamma = 0.6, 1.0, 2.5$. As visual representation for the implicitly defined function $f(x,y,z)$ we choose the iso-surface of the function

$$g(x,y,z) = \frac{1}{f(x,y,z)^2} \tag{2.5}$$

at an iso-value of 0.5.

In the inner square of Figure 2.2 eight deviating objects (i.e., the specimen) are shown. In the center of this square the reference model is shown. The vertical axis corresponds to the property γ which describes the actual shape of the iso-surface of the implicit function. The horizontal axis corresponds to the property s_x which describes the spatial extent of the iso-surface in x-direction. The object in the lower left corner of the inner square for example has the property values $s_x = 0.8$ and $\gamma = 0.6$. The inner square corresponds to a subspace of the property vector space which contains all occurring objects. The outer square in Figure 2.2 is the caricature space. The properties are exaggerated resulting in more distinctive

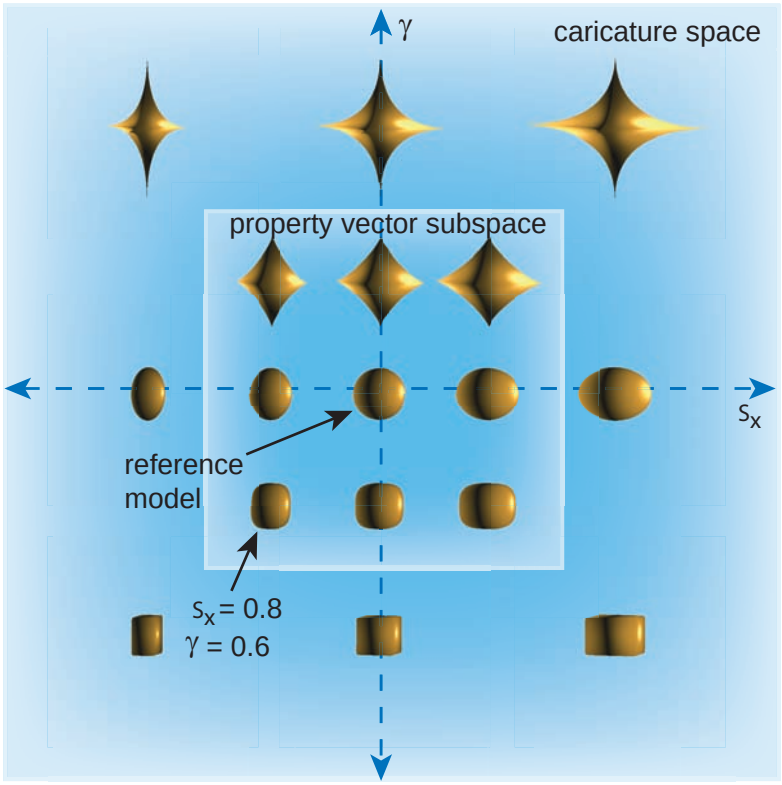


Figure 2.2: *Examples for caricaturistic operations. In the center of the inner square the reference model is depicted. The remaining eight objects in the inner square are examples of deviating specimen. The vertical axis corresponds to property γ which describes the actual shape of the iso-surface of the implicit function. The horizontal axis corresponds to the property s_x which describes the spatial extent of the iso-surface in x-direction. The outer square shows the caricatures of the corresponding inner square's objects.*

visual representations of the objects. The object in the lower left corner of the inner square differs in both properties from the reference model. Its visual representation is still close to the visual representation of the reference model. The caricature of this object makes use of a larger property vector space (i.e., the caricature space) and therefore results in a more distinct visual representation.

The objects in the upper row of the inner square are visually similar. The corresponding caricatures of these objects are shown in the upper row of the outer square. Due to the exaggeration of their descriptive properties they are visually more distinctive. The exaggeration of properties to make datasets more distinctive from each other is described in more detail in Section 2.4.

2.4 The Caricature Matrix

While artists drawing caricatures do not explicitly make use of a reference model (as illustrated in Figure 2.1), for caricaturistic visualization an explicit reference model is necessary. The exaggeration function assumes the existence of a difference function, which by itself assumes the existence of a reference model. Therefore caricaturistic visualization fails without a reference model. Collections of datasets about a given subject often lack the explicit existence of a reference model. In some cases this might be compensated by calculating the average of the available datasets. The average can then be used as reference model.

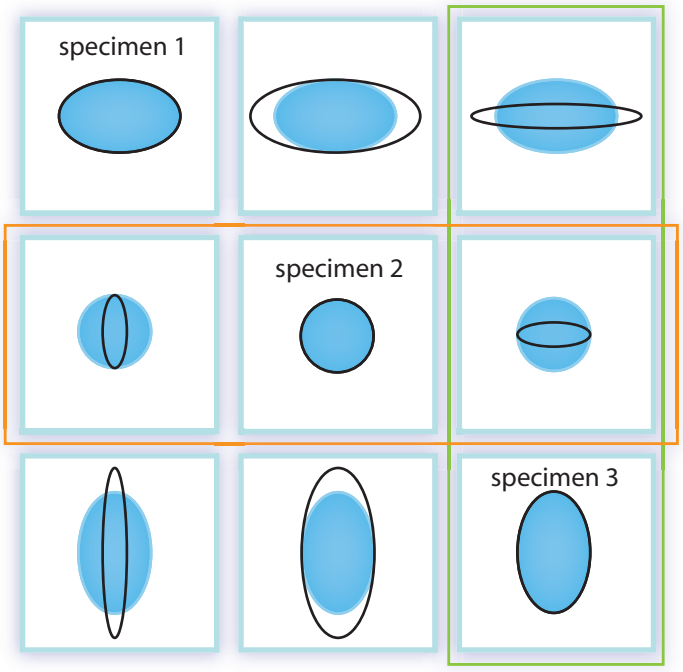


Figure 2.3: *Illustration of the caricature matrix. In the main diagonal the actual objects are shown in blue. Caricatures of the objects are drawn as black outlines. The rows of the matrix can be read as the caricatures of the object using the remaining objects as reference models.*

However, the direct visualization of differences between the datasets is a more expressive option. Each dataset from a given collection can be used as the reference model for all remaining datasets. A collection of n datasets leads to n^2 caricaturistic visualizations. We call this set of images the *caricature matrix*. In Figure 2.3 we illustrate the structure of the caricature matrix. The main diagonal is depicting the specimen. Row i of the matrix shows all caricatures of the object i using

the remaining objects as reference models. Column j of the matrix shows all caricatures using object j as the reference model. For example the second row in Figure 2.3 (outlined in orange) shows all caricatures of specimen two. The third column in Figure 2.3 (outlined in light green) shows all caricatures which use the third specimen as reference model. Therefore the element $(2, 3)$ of the matrix shows the caricature of the specimen 2 using the specimen 3 as the reference model. The caricature matrix is not necessarily meant to be completely shown to the user at once. It is a concept requiring further visualization and exploration techniques. While the average of datasets is distorted by outliers the caricature matrix depicts the direct comparison of all datasets to each other. Therefore we expect the caricature matrix to be more robust.

2.5 Application Scenarios

For a proof of concept for caricaturistic visualization we implemented three different systems. The system described in Section 2.5.1 is an approach to visualize differences in images of deformed facades. It follows a fully automatic feature specification approach. In Section 2.5.2 a system for visualizing differences in volumetric datasets is described. The feature specification in this system follows a user driven approach. In Section 2.5.3 a system for the specific case of CT angiography data is described. For the specification of features a semi-automatic approach was implemented. The aim of the implementation is to explore the abilities of caricaturistic visualization in different scientific areas and to demonstrate the applicability of the caricaturistic visualization concept on a variety of datasets. In Section 2.6 the results of the different systems are presented.

2.5.1 Caricaturistic Facade Deformation

The deformation of facades can be monitored by taking pictures at different points in time. These images are compared in order to find deformations that might require human intervention to guarantee the safety of the building. The subtle deformations are difficult to observe and therefore usually statistically evaluated. Caricaturistic visualization can enhance the subtle differences to make them clearly visible.

We tested our approach with simulated data from geodesists which they use to develop statistics for classifying facade deformations. In Figure 2.4 (left column) an image of the reference facade and of the deformed facade is shown. The feature specification is done fully automatically. The Harris [21] and the Förstner [16] interest operators are applied to find a set of relevant points in the reference image and in the deformed image (see Figure 2.4 middle column). A matching algorithm

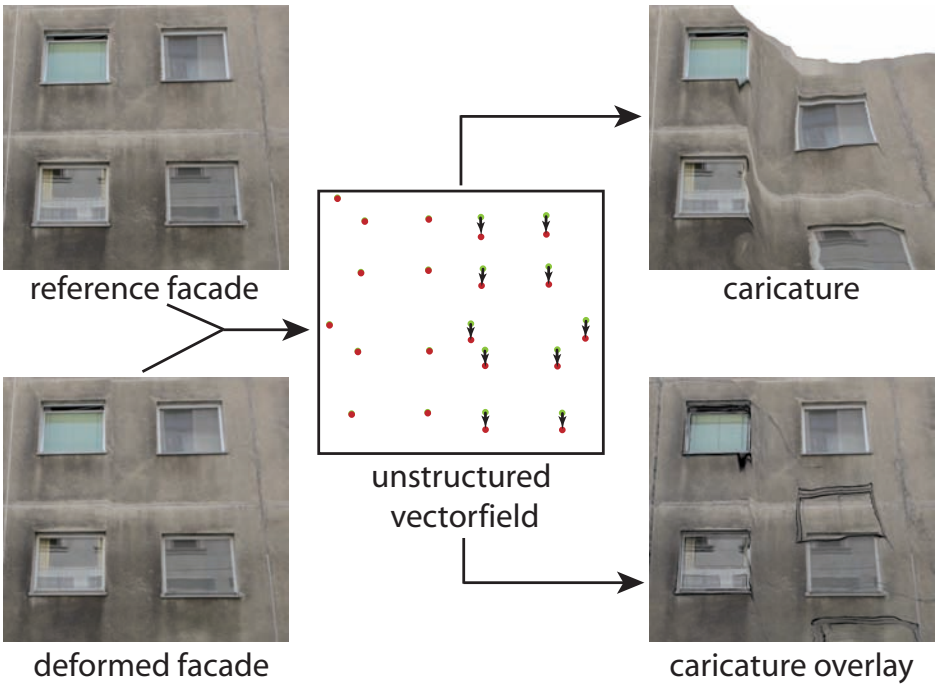


Figure 2.4: *Workflow for caricaturistic facade deformation. Left column: The reference model is shown on top and the deformed facade is shown below. Two irregular point sets are computed from the input images. Middle column: Illustration of two point sets and the corresponding unstructured vector field. Right column: Two visual representations of caricatures of the deformed facade. The upper caricature shows a deformed grid textured with the original image. The lower caricature shows the same deformed grid textured with an edge image.*

is applied to find corresponding pairs of points in the two images. Each pair of corresponding points specifies a deformation vector. The result of this procedure is an unstructured vector field describing the deformation of the facade.

We use this vector field to exaggerate the observed deformation. We first compute a Delaunay triangulation of the unstructured points of interest. This enables us to interpolate a vector for each position on the image plane. The user of the system controls only the exaggeration parameter which determines the interactive deformation of a textured grid. The deformed grid is textured either with the original picture of the deformed facade or with a more sparse representation of the image. For the sparser representation we chose an image showing only the edges of the deformed facade image. We automatically derive this image through an edge detector. The sparse edge image is overlaid onto the original image. Both examples are shown in the right column of Figure 2.4.

2.5.2 Caricaturistic Volume Visualization

We implemented a system for the generation of caricaturistic volume visualizations. To achieve a caricaturistic visualization the user has to specify a certain number of features in the reference model and corresponding features in the datasets of interest. Once the corresponding feature pairs are specified, the exaggeration function provides a feature vector for each value of the exaggeration parameter δ. This exaggerated feature vector is mapped to a visual representation. Caricaturistic visualization is not restricted to a specific visual representation. The exaggeration of features can be mapped to sparse representations such as contours, iso-lines, hatched surfaces, etc., or to dense representations such as polygonal surfaces or iso-surfaces. The possible visual representations also vary in the degree of abstraction and range from very tangible representations like iso-surfaces to high-level abstractions such as explanatory glyphs or automatically placed labels.

We implemented different approaches for feature specification and investigated different visual representations which widely vary in the level of sparseness. For each approach we describe the feature design and the visual representation used to generate the result images. The first approach, i.e., *caricature by visual augmentation*, augments direct volume rendering with NURBS surfaces in order to depict an exaggerated shape of the underlying feature. The second approach, i.e., *caricature by deformation*, takes the user specified features to deform the volume of the specimen. This results in a more distinct visualization of the specified features.

Caricature by Visual Augmentation

Our caricaturistic visualization system provides the user with an interface for feature specification. The user has to specify corresponding features in the reference model and in the specimen datasets respectively. The features used in our implementation consist of the following properties: a position, a major axis, and a minor axis. These properties implicitly define a local feature coordinate system. Further, the feature is defined by the spatial extent in the axis directions of the local coordinate system. These properties are specified and manipulated directly by the user. An additional property is derived automatically once the user has specified the other properties. This property describes the normal distance between the feature's major axis and a specific iso-surface in the volumetric object. In Figure 2.5 the local coordinate system of the feature is shown. On the left hand side the feature is shown in 3D. The blue circle in Figure 2.5 is the unit circle in a plane perpendicular to the major axis. x_f is a parameter varying along the major axis of the local coordinate system. In the example in Figure 2.5 x_f is set to x_0. $s(x_f, \theta)$ is the distance of point x_f on the major axis to the iso-surface in the direction θ. θ is the angular offset of the ray to the minor axis. Therefore, $s(x_f, \theta)$ is the normal

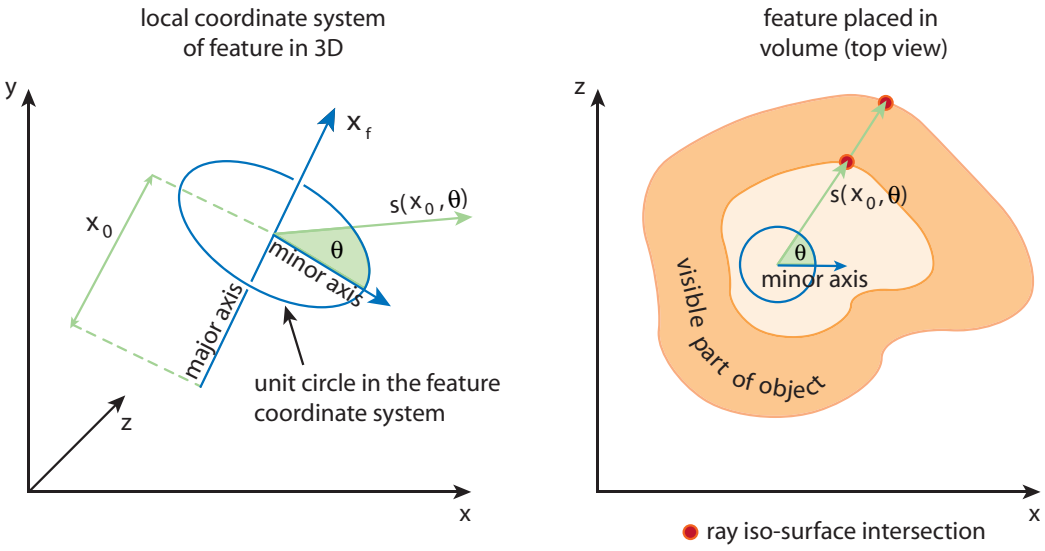


Figure 2.5: *Illustration of the feature's local coordinate system. On the left hand side the feature is shown in 3D. The direction and extent of the major and minor axis is specified by the user. The blue circle depicts a unit circle in the plane perpendicular to the major axis. The parameter x_f determines the position of the plane along the major axis. In this example the parameter is set to x_0. $s(x_0, \theta)$ is the distance of x_0 to the iso-surface in the direction θ, where θ is the angular offset of the ray to the minor axis. $s(x_f, \theta)$ corresponds to the normal distance of the major axis to the iso-surface. On the right hand side the feature is placed in the volume. The ray given by $s(x_0, \theta)$ intersects the volumetric object at two positions.*

distance of the iso-surface to the major axis. On the right hand side of Figure 2.5 the feature is shown in the volume (illustrated in 2D). The rays in general intersect many iso-surfaces. In our current prototype implementation the user can choose to either store the distance to the first or to store the distance to the last intersection point. We discretize the parameters x_f and θ in order to precompute the normal distance of the iso-surface to the major axis. The granularity of the discretization can be adjusted by the user.

For the specification of a feature in volumetric space the user has to specify values for properties like position and spatial extent of the major axis. Therefore it is necessary to provide a method for specifying a position in three dimensional space. By clicking on the image plane the user selects a ray in the viewing direction. The ray is intersected with the iso-surfaces of the volumetric object. In Figure 2.6 the ray intersects the iso-surfaces of the object at several positions. The user decides if the chosen ray specifies a point at the hit iso-surface, or a point in the middle between two consecutive iso-surface intersections. This approach allows

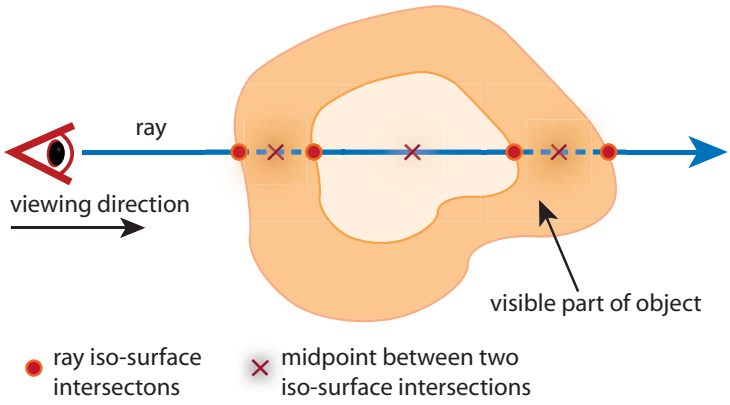


Figure 2.6: *Specification of a position in volumetric space. A ray is cast in the viewing direction intersecting the iso-surfaces of the volumetric object at several locations. Regions of homogeneous color in the figure correspond to regions of homogeneous visibility. The ray iso-surface intersections and the midpoints between two consecutive intersection points are candidates for the specified position.*

the placement of a feature in the middle of a homogeneous region or directly on the iso-surface. This spacial positioning of a point enables a wide variety of feature specification methods. In our approach the user sets the position of the feature as well as the direction and the spatial extent of the major axis by two consecutive mouse clicks. The first click specifies the position and the second click the other properties. The extent of the two remaining axes as well as their direction can be immediately manipulated by the user. When the local feature coordinate system is specified the normal distance to the major axis is derived automatically.

As visual representation for the caricature we chose NURBS curves and NURBS patches that are displaced from each other. We exaggerate the normal distance of the major axis to the iso-surface according to Equation 2.3. The exaggerated distances are taken to compute the control points of the NURBS patches and curves. An example of *caricature by visual augmentation* can be seen in Figure 2.9.

Caricature by Volume Deformation

Caricature by volume deformation is an approach based on the deformation of the volume during ray casting. Our approach is similar to the approach of Lerios et al. [38] who describe a technique for interpolating two volumetric models. In our method we extrapolate from the volumetric model through exaggerations in the feature coordinate system. This results in a volume deformation driven by the characteristics of the volume dataset. We describe the approach first for one

feature and later extend it to an arbitrary number of features.

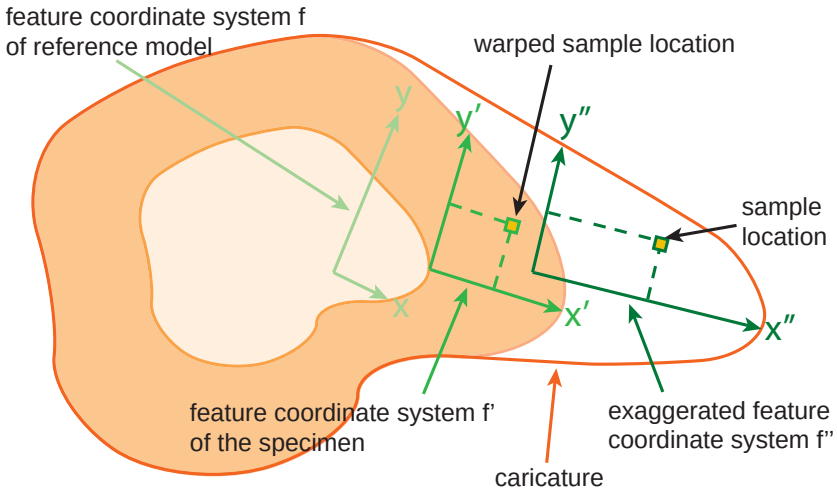


Figure 2.7: *Warping of a sample location. The feature coordinate system of the specimen dataset f' is exaggerated according to the feature coordinate system of the reference model f resulting in the exaggerated coordinate system f''. Each sample is warped from f'' to f'. The density value is derived by transforming the warped sample into volume space.*

Features are specified as described in Section 2.5.2. Each corresponding pair of features is defined by their local coordinate systems. During ray casting we warp the exaggerated feature coordinate system back to the original position of the feature's local coordinate system. The idea is sketched in Figure 2.7.

Figure 2.8: *Caricaturistic visualization of a facade deformation. The image of the deformed facade is overlaid by the caricatures. The exaggeration parameter is increased from left to right.*

The volume deformation is determined by the feature coordinate system of the reference model f and by the feature coordinate system of the specimen dataset f'. First the exaggerated feature coordinate system f'' is computed according

to Equation 2.3. In Figure 2.7 the three different feature coordinate systems are shown. We implemented a ray tracing approach where each sample position is transformed into the exaggerated feature coordinate system f''. The coordinates of the sample in the exaggerated feature coordinate system are then warped into f', the coordinate system of the specimen. The warped sample coordinates are finally transformed back into volume space to access the density value needed for ray-casting. The resulting caricature of the object is illustrated as an orange outline in Figure 2.7. Following the approach described by Lerios et al. [38], we extend our approach to more than one feature. We define a weighting function for each feature which is inversely proportional to the squared distance from the sample position to the position of the exaggerated feature. This allows a local control of the volume deformation specified by each feature. The above described warping calculation is done for each feature. The final density value for an arbitrary sample is computed as the weighted sum of all resulting density values. Examples of *caricature by volume deformation* can be seen in Figure 2.10.

2.5.3 Caricaturistic Angiography Visualization

For the specific case of angiography data we implemented an approach that caricaturizes the radius of blood vessels. Different vessel trees are compared to each other and are subject to caricaturistic visualization. The hierarchical structure of a vessel tree is given as a tree of segments. Each segment except the first one has exactly one predecessor segment and can have several successor segments. In order to compare two different vessel trees the hierarchical structures of the trees must correspond. The radius and the direction of the centerline are given on discrete points along the centerline of the vessel. This information is derived in a semi-automatic process described in the work of Kanitsar et al. [29]. To compare the radii along two segments we normalize the length of the segments to obtain corresponding values. With these corresponding values we exaggerate the radius and map it to a visual representation for the caricature. As visual representation we use small strokes placed at the boundaries of the exaggerated vessels.

In order to control occlusions of the vessel tree from different viewpoints we use a ghosting technique. The opacity of the volume is reduced in front of the vessel tree to a user defined value between zero and one. Zero corresponds to full occlusion of the vessel tree while one results in a full cut-out view on the vessel. We achieve this ghosting by first rendering spheres along the vessels into a depth texture. This depth texture is used during volume rendering to determine the entry points of the rays.

Further, the user can control the sparseness of the placed strokes. Occluded regions are automatically drawn sparser than non occluded regions. This is achieved by first rendering the dense set of strokes into a color buffer and into a depth texture.

This depth texture is used during volume rendering for the termination of the rays. All strokes behind opaque parts of the volume are occluded. As an overlay a sparse set of strokes is rendered sketching the shape of the caricature in occluded regions. Examples of *caricaturistic angiography visualizations* can be seen in Figure 2.11.

2.6 Results

All results were obtained in short interactive sessions. The feature specification and the visualization runs in real time on an AMD Athlon64 dual core 4400+ system with a GeForce 7800 GTX graphics processor with 256 MB memory. For the volume deformation approach we use a low quality mode for user interaction. The high quality image is rendered in about one second.

Figure 2.9: *Caricaturistic visualization of a carp. Left: reference model, Middle: direct volume rendering of a specimen augmented with a caricature of the diameter of its gas bladder. Right: caricature of the carp's shape.*

Figure 2.8 shows a caricaturistic facade deformation as described in Section 2.5.1. The deformation is exaggerated and overlaid over the original image of the deformed facade. The exaggeration parameter is increased from left to right.

Figure 2.9 shows a caricature by visual augmentation as described in Section 2.5.2. On the left of Figure 2.9 a fish of the species *Polypterus senegalus* is shown. It is used as reference model for the caricaturistic visualization of the carp dataset shown in the middle of Figure 2.9. The carp is augmented with NURBS surfaces which depict the exaggerated shape of the hollow space in the carp's interior. The right image in Figure 2.9 shows a caricature of the carp's shape. Both caricatures are derived using the feature specification approach described in Section 2.5.2. For each of the caricatures a feature is placed in the reference model as well as in the specimen. For the caricature in the middle of Figure 2.9 the feature is placed in the hollow space of the fishes. Rays are cast perpendicular to the major axis of the feature. The first ray iso-surface intersection is used for the estimation of the diameter. For the right image of Figure 2.9 the feature is placed inside the

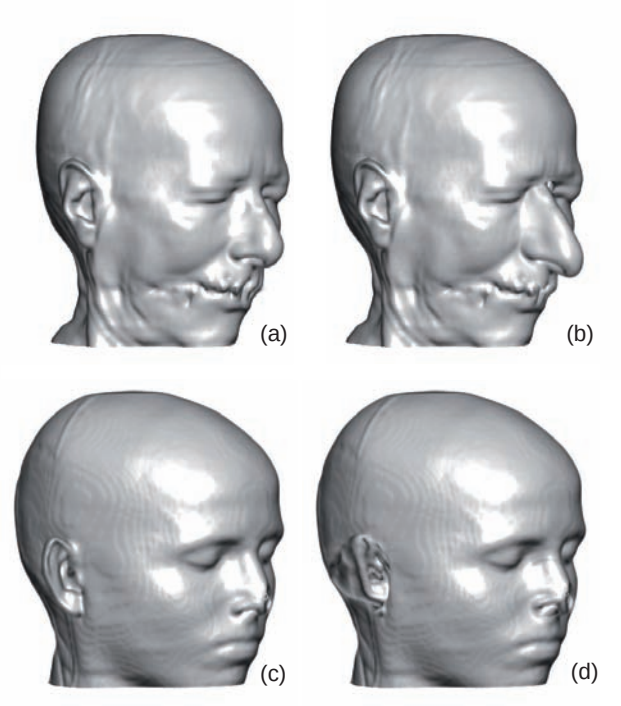


Figure 2.10: *A caricaturistic volume deformation. In (a) and (c) iso-surface renderings of the two datasets are shown. In (b) a caricature by volume deformation is shown using (c) as reference model. In (d) a caricature of (c) is shown using the features of (a) as reference model.*

fishes from the head to the caudal fin. For the estimation of the shape rays are cast perpendicular to the major axis where the last iso-surface intersection is stored.

In Figure 2.10 an example of a caricaturistic volume deformation specified by two feature pairs is shown. The features are specified to describe the extent and rotation of the nose and the right ear. The two depicted datasets where used as reference models for each other.

In Figure 2.11 results of the caricaturistic angiography visualization are shown. On the left of Figure 2.11 a caricaturistic angiography visualization is depicted. The vessel tree in green is augmented with black strokes representing the caricature. A ghosting technique was used to show the vessel in regions where it is behind the opaque bone. The caricature is sketched more sparsely (as dashed line) where it is occluded resembling the style of occluded objects in illustrations.

On the right of Figure 2.11 a caricature matrix for three different blood vessel trees is shown. Each vessel tree is presented together with a close-up of the lower *arteria femoralis*. The images in the main diagonal show the three different vessel trees.

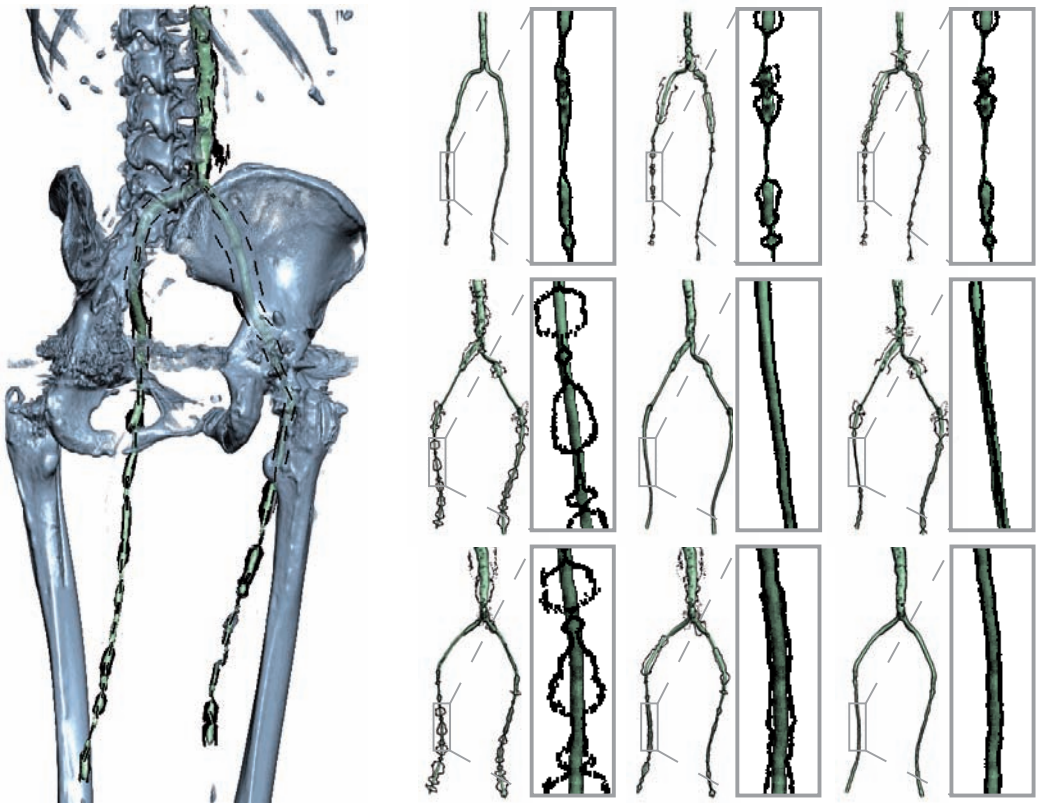


Figure 2.11: *Left: A caricaturistic angiography visualization. The occluded parts of the vessel tree shine through the opaque bones. The occluded strokes of the caricature are drawn more sparsely. Right: 3x3 Caricature matrix of three different vessel trees with a close-up of the caricatures.*

2.7 Summary

Caricatures exaggerate feature deviations between a specimen and a reference model. These deviations are the characteristics of the specific specimen. Caricatures depict the essence of a subject of interest. The caricature metaphor seems well suitable for visualization since caricatures have many properties in common with visualization. We presented a mathematical framework for caricaturistic visualization adequate for a wide variety of applications. Further, we introduced the caricature matrix, a technique based on the caricature metaphor. It makes subtle differences between datasets visible, without the need of an explicit reference model.

The limits of my language mean the limits of my world.

Ludwig Wittgenstein

Introduction

Caricaturistic Visual Abstraction

Semantic Visualization Mapping

Interactive Illustration

Summary and Conclusions

Mapping data abstractions to visual abstractions is inherently needed in visualization systems. In this Chapter a novel visualization mapping approach is introduced that explicitly makes use of domain and visualization semantics. The semantic layers concept allows a domain expert to specify the visualization mapping using coined terms of the domain. The mapping is specified by visualization rules that are evaluated with fuzzy logic. The work presented in this Chapter was published in *IEEE Transactions on Visualization and Computer Graphics* [49].

Chapter 3

Semantic Visualization Mapping

3.1 Introduction

Many popular direct volume rendering techniques use transfer functions that map the measured density values to colors and opacities. These visual properties are usually composited into the final image. More advanced techniques use other volumetric attributes, like gradient magnitude, curvature, or statistical properties and map these values via a multi-dimensional transfer function to visual attributes. All these techniques have in common, that they map attributes of the underlying data on visual appearance via a transfer function. Transfer functions are a powerful tool to achieve various visualizations. However, the specification of transfer functions is a complex task. The user has to have expert knowledge about the underlying rendering technique to achieve the desired results.

Especially the specification of higher-dimensional transfer functions is challenging. Common user interfaces provide methods to brush in two dimensions. While brushing is an intuitive method to select regions of interest or to specify features, user interfaces for higher-dimensions are more challenging and often non-intuitive.

We propose an alternative method to achieve meaningful mappings from volumetric attributes to visual appearance. Our method enables a multi-dimensional mapping from several volumetric attributes to multiple visual styles. We replace the complex task of multi-dimensional color transfer function design by introducing semantic layers. A semantic layer linguistically describes a mapping from a combination of volume attributes to one visual style.

Drebin et al. [14] suggested to use probabilistic classification algorithms to avoid the artifacts of binary classification. The resulting classification for descriptions of tissue types like *air*, *fat*, *soft tissue*, and *bone* correspond to membership functions of fuzzy sets. We extended this idea by explicitly defining semantic values using fuzzy sets, enabling a linguistic specification of renderings. The fuzzy sets are described by membership functions that are specified by the user. The

mapping of various volume attributes to a given visual style is achieved with fuzzy logic arithmetics. The volume attribute *density* is for example described by semantic values ranging from *zero*, *very low*, *low* and *middle* to *high*. The style *shading* is for instance described with the semantic values *soft*, *hard*, and *cartoonish*. A rule maps for instance the semantic values *high density* and *positive curvature* to the semantic value *cartoonish shading*. Fuzzy logic rules specify the mapping using the language of the application domain.

The objective of the semantic layers concept is a meaningful mapping from given volume attributes to given visual abstractions for the purpose of illustration. We identified two major challenges to make our novel concept applicable to scientific visualization. On one hand the mapping should not violate the property of bijectivity in order to be an adequate replacement of the well established color transfer functions. On the other hand it is desirable to achieve a semantic mapping by the use of meaningful values.

Bijective Mapping: A bijective mapping ensures that the generated image can be interpreted by the viewer. For the purpose of visualization it is necessary that the meaning of the image can be resolved. The viewer of the image has to be provided with details about the mapping procedure in order to resolve the meaning of the image. A common example of a bijective mapping is a visualization with color encoding. The legend accompanying the visualization ensures that the image can be resolved and has a meaningful interpretation in the underlying parameter domain.

Volume rendering with semi-transparency cannot guarantee to provide bijective visualizations. The compositing of the semi-transparent colors introduces ambiguities. However, volume rendering does potentially convey more information about the data than pure iso-surface renderings. Because of the semi-transparency inner structures are unveiled and can be explored. The ambiguities introduced by the compositing function can be resolved if the used colors are chosen carefully. Animation or interactive exploration provides further help to resolve the ambiguities. The use of visualization rules in our approach provides information about the visualization mapping and preserves the bijectivity property of visualizations. However, our approach does not solve the problem of ambiguity in volume rendering.

Semantic Mapping: The specification of the semantic mapping from volume attributes to visual styles in our approach is done with semantic values. Measured and simulated data have usually several meaningful intervals that are relevant to the user. For example a PET scan of a brain measures brain activity. It shows homogeneous regions of activity in the brain that are labeled by experts with semantic values such as *low activity* or *high activity*. Diffusion MRI data provides information about the healthiness of tissue regions and is classified by experts with semantic values like *healthy*, *diseased*, or *necrotic*. Medical CT data encode the

measured density values in Hounsfield units. Specific intervals of the Hounsfield scale refer to different tissue types like air, soft tissue, bone, contrast enhanced vessels, etc.

There are many more examples where semantic parameters are used by domain experts but are not necessarily used for visualization. Especially for direct volume rendering the transfer function prevents the use of these parameters for the description of visualization mappings.

Common direct volume rendering techniques also do not make use of semantics for the description of visual abstractions, although several semantic parameters are used naturally by illustrators. For example *shading*, *tone*, *rendering style*, *saturation*, *texture*, etc. are described using natural language. We propose to use semantic parameters from expert domain as well as from illustration in the specification of the visualization mapping. Figure 3.1 shows examples for semantics used in the antecedents, and the consequents of rules, as well as examples for visualization rules that can be defined with the semantic visualization mapping approach.

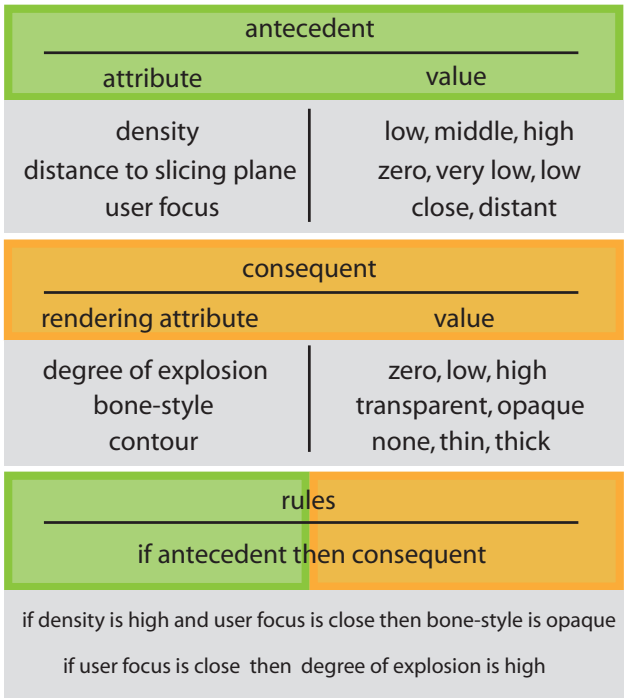


Figure 3.1: *Rules consist of an antecedent and an consequent part. The Figure shows examples for semantics typically used in the antecedent and the consequent as well as examples for semantic visualization mapping rules.*

Section 3.2 will review related work. In Section 3.3 the novel semantic visualizaton mapping concept is described in detail. We give an overview of the concept

and illustrate it with a simple example. In Section 3.4 we give details about the implementation of the fuzzy logic pipeline. Section 3.5 briefly discusses the used illustrative volume rendering algorithm. In Section 3.6 we show exemplary results of our system for different potential application areas. In Section 3.7 we conclude our work and give ideas for possible extensions of our approach.

3.2 Related Work

The related work about volume visualization and illustrative visualization is divided into four different categories. In Section 3.2.1 we give a brief overview of rule based interfaces for computer graphics. We review the related work about illustrative visualization in Section 3.2.2. In Section 3.2.3 we compare our work to other multi-dimensional visualization techniques that map multiple volume attributes to multiple visual abstractions. In Section 3.2.4 we describe related work in the area of medical visualization that deals with the semantic classification of volume data.

3.2.1 Rule Based Interfaces

The semantic visualization mapping approach is conceptually similar to the work of Coyne et al. [11] as well as to the work of Seligmann and Feiner [62]. Each of these approaches uses a textual description to specify the mapping from data to imagery. Coyne et al. [11] present a system that translates a description of a scene into a 3D representation using a large database of object models. The objects are arranged in the scene using relational expressions such as *above*, *below*, *in front of*, etc. This provides a non-expert user interface for modeling geometric scenes. The approach of Seligmann and Feiner [62] is more similar to our work as it does not focus on the description of scenes themselves but rather on the description of the visualization method. Design rules guide the rendering process. The rules are used to achieve the desired communicative intent. Different illustration strategies like *emphasis of features*, *ghosting*, *cutaways*, etc. are used to accent or unveil features. While their work focuses on a user interface for the illustrative depiction of manually modeled geometric objects, our approach implements a similar concept for volume visualization. The approach of Svakhine et al. [67] is targeting scientific visualization and is conceptually also similar to our work. They use illustration motifs to generate illustrative visualizations. The level of expertise of the viewer serves as input to adjust the illustration.

3.2.2 Illustrative Visualization

Many previous approaches for the selective application of visual abstractions have been presented. Yuan and Chen [82] present a method to enhance volume rendering with different styles of non-photorealistic surface rendering techniques. Hauser et al. [24] introduce two-level volume rendering that allows the selective combination of multiple rendering methods for different regions. The selective application of specific styles based on volume attributes was also used in other previous work [6, 42]. However, the focus of our work lies on the semantic visualization mapping being capable of selectively applying these illustrative effects.

For the description of many illustration styles we adapted the approach of Sloan et al. [64]. They present a technique to render pre-defined artistic styles that was later adopted for volume visualization in the work of Bruckner and Gröller [8]. This method allows the specification and use of different artistic styles such as contours, illustrative tissue styles, shading techniques, etc. in a uniform framework. We adapted this technique to serve as a basis for the parameterized representation of different styles.

3.2.3 Multi-Dimensional Volume Visualization

Our approach is capable of describing visualizations that map data of multiple dimensions to visual abstractions. We briefly review approaches that use multi-dimensional transfer functions for volume rendering. Kniss et al. [33] present an approach for multi-dimensional transfer functions. They also employ this technique to quantify statistical measures of multiple fuzzy segmentation volumes [34]. Hladůvka et al. [25] as well as Kindlmann et al. [31] used multi-dimensional transfer functions based on the density and a curvature measure. In the work of McCormick et al. [45] as well as Stockinger et al. [66] systems are presented that allow a formulation of the visualization mapping as mathematical expressions. Our approach hides the complexity of mathematical formulations with visualization rules using meaningful domain semantics. In the work of Doleisch et al. [13] a system for the interactive exploration of complex data is presented. Woodring and Shen [79] use set and numerical operators to visualize and compare multi-variate and time-varying data. Sato et al. [61] use rules to identify tissue structures in multi-modal data. Tzeng et al. [72] show a user interface to specify input for a neural network that classifies volume data in higher dimensions.

Our approach is similar to these approaches as it uses a rule based specification of features in the data. However the specification of the visualization mapping to visual attributes in our system is done using meaningful visualization rules. This allows the use of semantics from expert domain as well as from visualization domain.

3.2.4 Medical Visualization

Illustrative visualization for operation planning, patient briefing, and multi-modal visualization are potential applications of our approach. Others have presented medical visualization approaches related to our work.

Tappenbeck et al. [69] as well as Zhou et al. [83] modify the appearance of volumetric structures based on the distance to a predefined region. They present approaches for medical visualization implementing a specific visualization mapping method. Our system is general enough to provide similar functionality. The user defines a style that is used in dependence of the distance to a given region. The chosen style can be altered and modified interactively.

Rezk-Salama et al. [55] present a high-level user interface for the specification of transfer functions with semantics. We follow up on the idea of specifying a mapping from volume attributes to visual abstractions by meaningful parameters, but present a method that offers direct control over the specification of the visualization mapping.

3.3 Overview of the Semantic Layers Concept

A comparison between the traditional color transfer function based approach and the semantic visualization mapping approach is shown in Figure 3.2. The traditional approach takes multiple volume attributes as input and derives a color via the color transfer function for each sample position. The color is usually shaded and used for compositing to get the final color of the pixel. The semantic visualization mapping approach also takes multiple volume attributes as input but in contrast evaluates a set of rules to determine the visualization parameters applied to the current sample. The visualization parameters influence the appearance of a visual abstraction method.

To demonstrate the power of our approach we chose to use style transfer functions [8] as visual representation. Style transfer functions allow the description of illustration styles by artistic examples. Illustration semantics are defined for each style. For instance a style transfer function can describe a contour style making it possible to define semantic values like *thin* and *thick contours*. Another example is the description of different tissue styles like the *bone style*, *soft tissue style*, *skin style*, etc. using style transfer functions. Multiple styles are used for the visualization mapping and are blended to determine the color of a sample during ray casting. The colors of the samples along a ray are composited to determine the final color of the pixel.

Semantic values are used to described volume attributes and visual styles. In Section 3.3.1 semantic values are discussed. To achieve the mapping from semantic

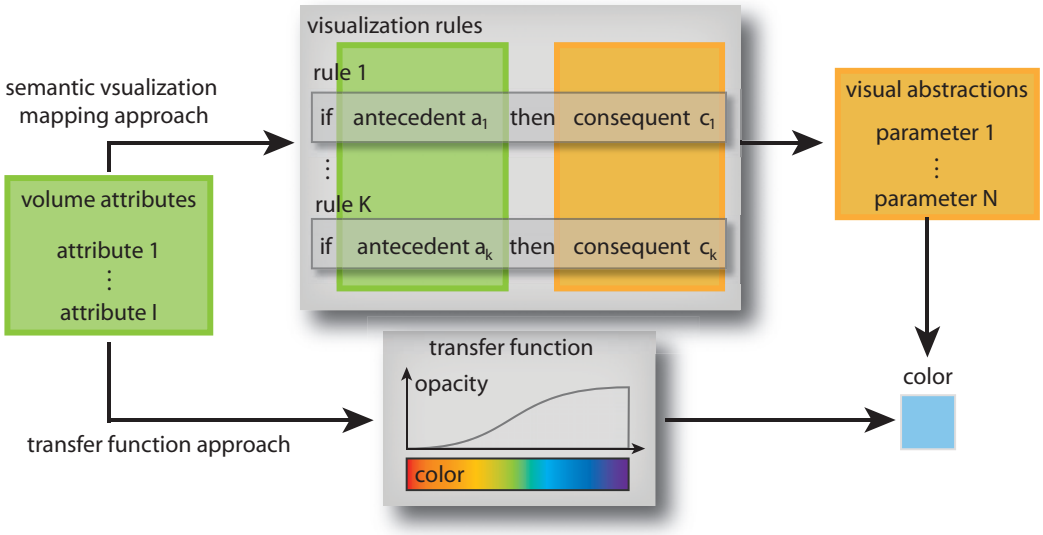


Figure 3.2: *Comparison of the traditional color transfer function and the novel semantic visualization mapping approach: In the traditional pipeline, multiple volume attributes are used to look up color in the color transfer function. The color is commonly shaded afterwards. The semantic visualization mapping approach uses visualization rules that are specified using semantic values of volume attributes in the antecedent part and semantic values of visual abstractions in the consequent part. The rules map data to parameters of visual abstractions.*

values of volume attributes to semantic values of visual abstraction parameters (i.e. the semantic visualization mapping) we use visualization rules. The rule base is described in Section 3.3.2. The evaluation of the rules results in a parameter for each visual style. Each style is parameterized and applied according to the resulting value. The styles are described in Section 3.3.3. The final visual appearance of a sample is determined by the composition of the layered visual styles. The visual abstraction concept of layered styles used in our framework is described in Section 3.3.4.

3.3.1 Semantic Values

A semantic value is a linguistic description of a value of a volume attribute or a visual style. It is defined as a fuzzy set given by its membership function.

Figure 3.3 shows an example of the specification of semantic values for the volume attributes *density* and *curvature*. The attribute *density* is described with the semantic values *low* and *high*. The attribute *curvature* is described with the semantic value *close-to-zero*. A semantic value is represented with its respective membership function. In Figure 3.3 only simple examples are shown to illustrate

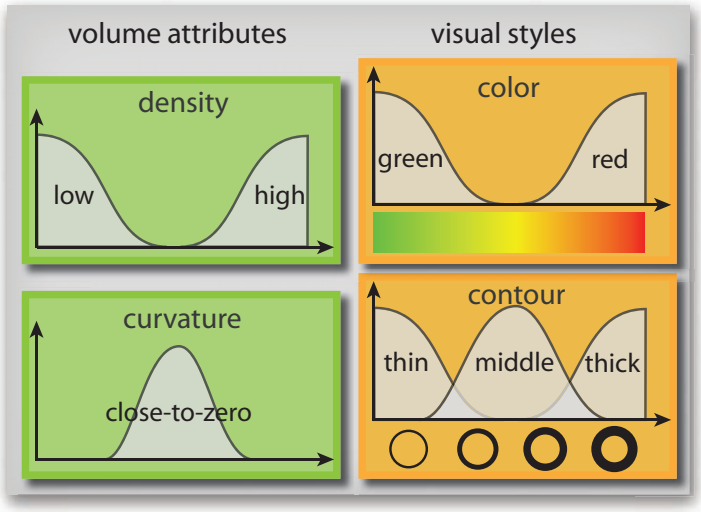


Figure 3.3: *Exemplary specification of semantic values. Simple semantic values for the volume attributes density, and curvature are shown as well as for the visual styles color and contour.*

the concept of membership functions and semantic value definition. The volume attribute *density* for example could additionally have the values *very low*, *low*, *middle* and *high* referring to meaningful types of tissue like *air*, *soft tissue*, *bone*, *contrast enhanced vessels*, etc.

The linguistic values for visual abstractions are also defined using membership functions. In Figure 3.3 simple examples for the visual styles *color* and *contour* are shown. For the visual style *color* the definition of the semantic values *green* and *red* are shown. The visual style *contour* is described by the semantic values *thin*, *middle* and *thick*. Other examples of visual styles include *shading* and *saturation*. *Shading* could be described with the semanitc values *none*, *soft*, *phongish*, *cartoonish*, etc. The style *saturation* could range from the value *no* to the semantic value *full*.

Membership functions model the semantics of volume attributes and visual styles. In Figure 3.3 smooth functions are used to describe the membership. Instead of the smooth functions also piecewise linear functions provide good results.

3.3.2 Rule Base

The central component of our system are the visualization rules which describe the desired visualization mapping. A rule states the premise in the antecedent part and the conclusion in the consequent part. The premise is a logical combination of semantic values of volume attributes. The conclusion is a list of styles that are affected by the rule. A rule could for example state *if density is middle and brain*

activity is very high then contour is thick. The result of the evaluation of one rule is another fuzzy set quantifying the membership to the antecedent. Antecedents of all rules that have implications on the appearance of one style are aggregated and defuzzified. The result of the defuzzification is a style volume describing the value of the visual style at each voxel position. Implication, aggregation, and defuzzification can for the moment be seen as black boxes that map the antecedents of the rules to parameters of visual abstractions. In Section 3.4 we give details about the used fuzzy logic methods.

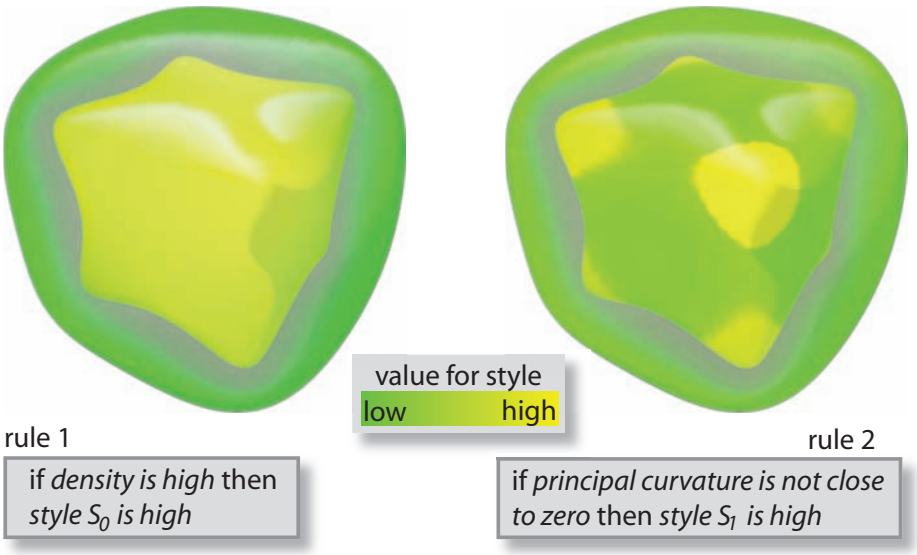


Figure 3.4: *Style volumes: Two iso-surfaces of the cube dataset for each style volume are rendered. A color-coding was used to encode the numeric values for each style. The left style volume shows the result of a density based rule. The right image shows the result of a curvature based rule. Green color means low values for the given style, yellow color means high values for the given style.*

In Figure 3.4 two style volumes are shown. A color-coding was applied on two nested iso-surfaces of the cube dataset to encode the value of the style. Green means low value for the style and yellow means high value for the style. The left style volume is the result of the rule *if density is high then style* S_0 *is high*. The right style volume is the result of the rule *if curvature is not close-to-zero then style* S_1 *is high*.

3.3.3 Styles

The style volume specifies the value for a style at each position. To apply a style to a sample the style needs to be parameterized. The parameterization of a style

ensures a continuous application of one style.

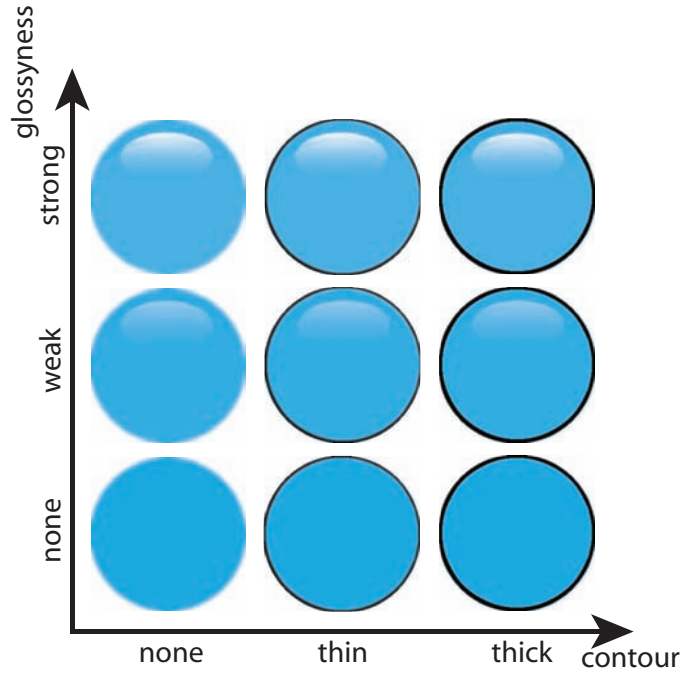


Figure 3.5: *Combination of three parameterized orthogonal styles: The style color is constantly set to blue. The styles contour and glossyness gradually vary over their specified domain.*

Parameterization: A common example of a parameterized style is a color scale. In traditional illustration however a greater variety of gradually varying styles exists. It is desirable that the application of one style volume results in a gradual application of the style. Parameterized styles are needed to achieve this gradual transition. An example of parameterized styles can be seen in Figure 3.5. The spheres in Figure 3.5 are drawn manually. The glossyness varies over the vertical axis from *none* over *weak* to *strong*. The style contour varies along the horizontal axis from *none* over *thin* to *thick*. In practice all three axis (the color, the glossyness, and the contours) vary continuously allowing a gradual change in each dimension. In Figure 3.5 the style *color* is constantly set to *blue*.

Orthogonality: We propose to use orthogonal styles to achieve meaningful mappings. Potentially the combination of different styles leads to ambiguities. It is desirable to achieve mappings of multiple styles that can be combined without leading to ambiguities. We define orthogonal styles as a set of styles that do not infer with each other. The concept of orthogonal styles can be seen in Figure 3.5. The example illustrates the combination of the three orthogonal styles *color*, *contour* and *glossyness*. Nine exemplary combinations of the gradually varying styles

contour and *glossyness* are shown. The *color* dimension is not shown in Figure 3.5, however it is orthogonal to the shown dimensions *contour* and *glossyness*. The orthogonality of visual styles does in general not solve the problem of ambiguities in volume renderings using semi-transparency. The use of orthogonal styles is (not sufficient, but) necessary to achieve meaningful and bijective volume visualizations.

3.3.4 Layered Styles

From an illustration point of view it is desirable to apply the styles incrementally and selectively. We chose to realize these illustration strategies with layered styles. Each style is assigned a priority and is applied accordingly.

Incremental Application of Styles: Our approach for the incremental application of multiple styles resembles the work-flow of a traditional illustrator. Traditional illustrations are drawn in layers. Starting with a background style the successive layers are applied on top of each other. For the case of traditional pencil illustration this procedure is described in *The Guild Handbook of Scientific Illustration* [26]. Layered styles for shading, contours, and specular highlights are applied on top of a basic style.

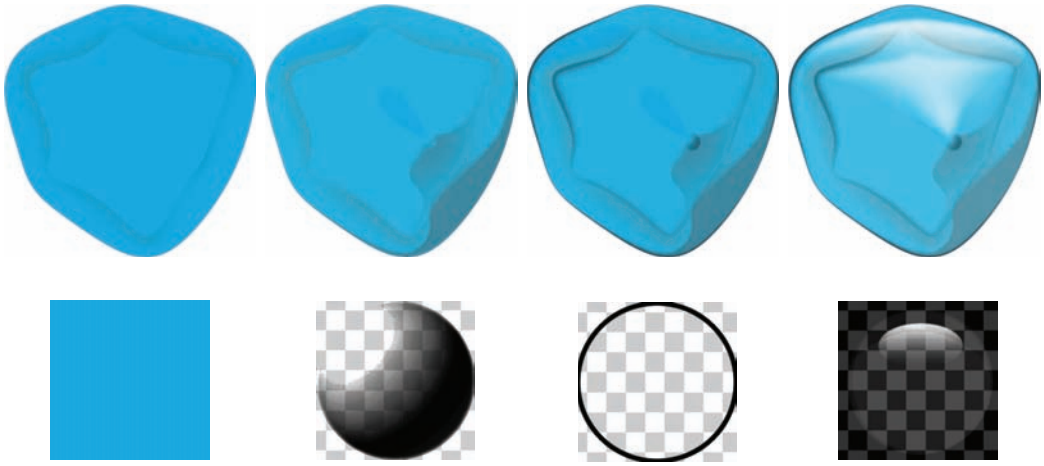


Figure 3.6: *Incremental application of styles: Two nested iso-surfaces of the cube dataset are shown. From left to right the styles are shown that are applied as incremental layers: blue background style, subtle cartoonish shading, contours, and glossy highlights.*

Figure 3.6 shows the incremental application of layered styles. Two nested iso-surfaces of the cube dataset are rendered applying the styles shown below. The styles are given as manually drawn spheres that are automatically applied during rendering to achieve a similar appearance. Details about the application of pre-defined styles during rendering are given in Section 3.5. The styles typically

contain transparency not to fully occlude each other. To clearly make the transparency of the used styles visible a checkerboard texture is used as background in Figure 3.6. The different layers respectively the different styles are blended, starting with a background style applied to all regions. The leftmost image in Figure 3.6 was rendered using the unshaded blue background style. The second image in Figure 3.6 is drawn applying a subtle cartoon shading style. Shading is another example of an orthogonal style that is not shown in Figure 3.5 as it is a fourth dimension. The third image in Figure 3.6 is rendered adding a contour. Finally the layer that resembles a glossy highlight is applied. The result is shown in the rightmost image of Figure 3.6. Since the glossy highlight style is transparent white the background in Figure 3.6 was inverted to make it clearly visible. Each style can be applied gradually and individually, however in Figure 3.6 all styles are fully applied to show the effect of an incremental application of several layered styles.

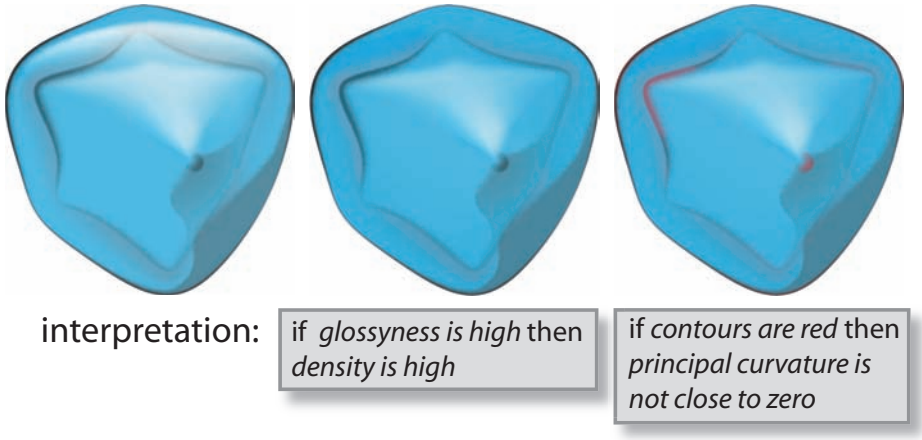


Figure 3.7: *Selective application of styles: The left image shows two iso-surface of the cube dataset. All styles are fully applied. In the middle image the style glossyness is determined by a density based rule. Glossy highlights are only drawn in regions of high density. In the right image the style contour is determined by a curvature based rule. The contours are drawn in red in regions of high absolute principle curvature. The rules specifying the mapping can be read backwards as an interpretation of the images.*

Selective Application of Styles: Illustrators are taught to avoid mixtures of too many styles. A selective application of styles to specific regions can aid to differentiate the individual regions. Each semantic layer defines the mapping to one style according to a set of visualization mapping rules. The rules allow for a differentiation of regions and for a selective application of the styles. Each layer is applied in specific regions according to its style volume, resulting in a selective application of the style. Note that a completely opaque style overdraws all styles

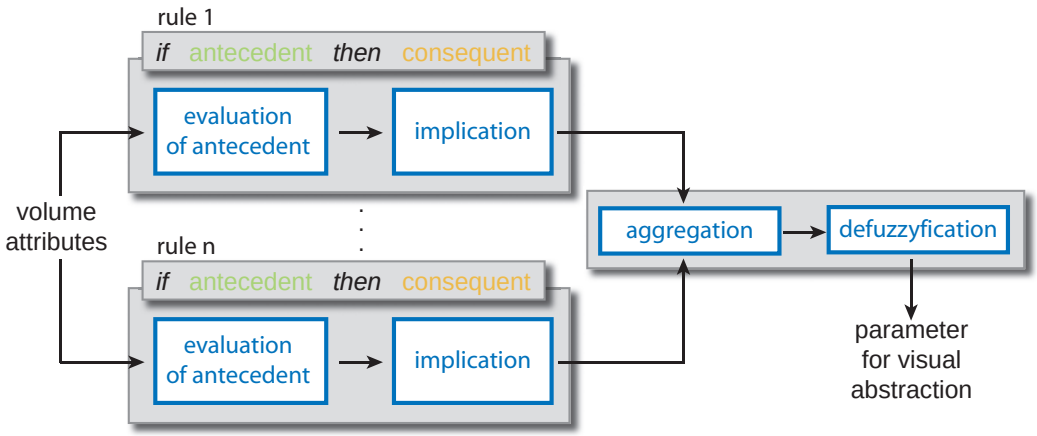


Figure 3.8: *Overview of the fuzzy logic system: The steps of the evaluation of one visual abstraction parameter are shown. The n visualization rules all influence the same visual abstraction.*

lower in priority. To achieve meaningful illustrations it is important to choose the styles and their priority carefully.

Figure 3.7 shows the selective application of visual styles. The leftmost image shows two iso-surfaces of the cube dataset. All styles are fully applied. In the middle image of Figure 3.7 the visualization mapping rule *if density is high then glossyness is high* is applied. The style *glossyness* is affected by this rule and is applied only to regions of high density. The right image of Figure 3.7 demonstrates the selective application of a style following a curvature based rule. The contours are drawn in red in regions with the first principal curvature not close to zero. The rules that lead to the selective application of the visual styles can be read in the opposite direction to resolve the image and to map back the styles to the original semantic parameters. The image in the middle of Figure 3.7 can be interpreted with the sentence *if glossyness is high then density is high* and the image on the right can be interpreted with the sentence *if contours are red then curvature is not close-to-zero*. The augmentation of the final result with the inverted rules ensures the interpretability of the achieved visualization and can be compared to a legend used for traditional visualization mapping methods.

In this Section we introduced the concept of semantic visualization mapping for illustrative volume rendering. Semantic layers are an implementation of the semantic visualization mapping concept that uses layered visual styles as visual abstractions.

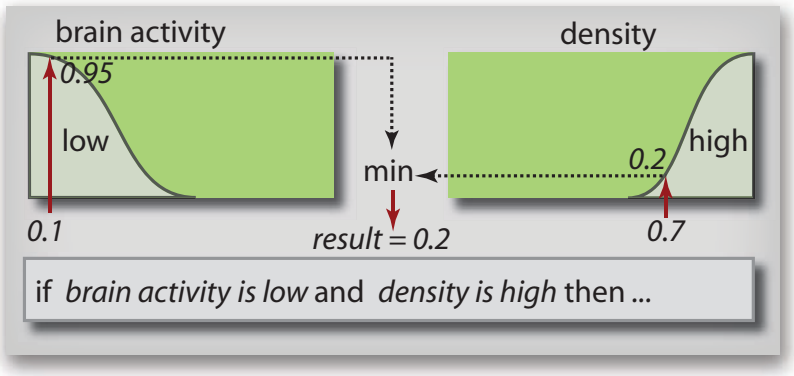


Figure 3.9: *Evaluation of Antecedents: The membership functions are evaluated and fuzzy logic operations are performed. In this example the logical and operation is evaluated.*

3.4 Fuzzy Logic

In this Section we describe the implementation of the fuzzy logic used to evaluate the visualization rules. We chose to implement a fuzzy logic system to allow the definition of semantic values for volume attributes as well as for the description of visual abstractions. Each semantic value is specified by a membership function. The actual mapping of the values of volume attributes to values of visual styles is specified using fuzzy visualization rules. The evaluation of the rule base is done using fuzzy logic operations. We will describe the fuzzy logic system used in our framework in detail. However, a more elaborate discussion on fuzzy logic can be found in the literature [81, 70]. In Figure 3.8 an overview of the fuzzy logic inference process for one parameter of a visual abstraction is shown. The inference process evaluates at each sample position the visualization mapping rules. In Figure 3.8 rules are shown that affect the same visual abstraction. The antecedents are evaluated according to the volumetric attributes. The implication of the antecedents on the consequents is calculated. All rules affecting the same parameter are aggregated and finally defuzzified to provide a crisp value for the parameter of the visual abstraction. The individual steps outlined in Figure 3.8 are evaluation of antecedent, implication, aggregation, and defuzzification and will be described in detail in the following:

3.4.1 Evaluation of Antecedents

The base for the semantic description of parameters are fuzzy sets. Fuzzy sets are specified via membership functions. For example a CT scan registered with a PET scan of a specimen's head represents tissue density and brain activity. In

Figure 3.9 the definition of simple membership functions for *low brain activity* and *high density* are shown, representing the corresponding semantic values.

Rules are specified using these semantic values. The number of attributes and the number of semantic values for each attribute are conceptually not restricted. For the sake of simplicity only two attributes with one semantic value each are used in the example of Figure 3.9. In the antecedent part, i.e., the *if* part of a rule, a logical combination of these semantic values is stated. Figure 3.9 shows the simple rule *if brain activity is low and density is high then* .... In the consequent part, i.e., the *then* part of a rule, semantic values of visual attributes are listed that will be influenced by the respective visualization rule.

In Figure 3.9 hypothetic input values of 0.1 for the brain activity and 0.7 for the density are chosen to illustrate the evaluation of the antecedent. First the membership functions *low brain activity* and *high density* are evaluated. The resulting membership is 0.95 for the fuzzy set *low brain activity* and 0.2 for the fuzzy set *high density*.

Logical operations are applied according to the specified rules. The unary negation operator (i.e., *not*) results in the calculation of one minus the membership. Using one of the binary operators *and* and *or* results in the min respectively the max of the operands. In the example of Figure 3.9 the *and* operator is applied to the fuzzy sets *high density* and *low brain activity*. The result of the evaluation of the antecedent is therefore the minimum of the two memberships which is 0.2 in this example.

3.4.2 Implication

Implication is the operation that modifies the membership functions of all semantic values in the consequent of the rules. In Figure 3.10 two rules are shown that both have an implication on the *contour style*. In Figure 3.10 the two semantic values *middle* and *thick* for the contour style are shown. In this example the evaluation of the antecedent for the first rule gave a value of 0.2, and for the second rule a value of 0.5. Implication is the minimum of the membership function and the value of the antecedent. It results therefore in truncated membership functions. Let $m(x)$ denote a membership function for a visual abstraction parameter, then the result $m'(x)$ of the implication is

$$m'(x) = min(m(x), a_i) \tag{3.1}$$

where a_i is the result of the evaluation of the antecedent of rule i.

In the example of Figure 3.10 the two membership functions for the *contour style* are shown that are truncated at a height of 0.2 and 0.5 respectively. The most common implication methods are scaling and truncation. We found that the truncation of the membership functions yields good results.

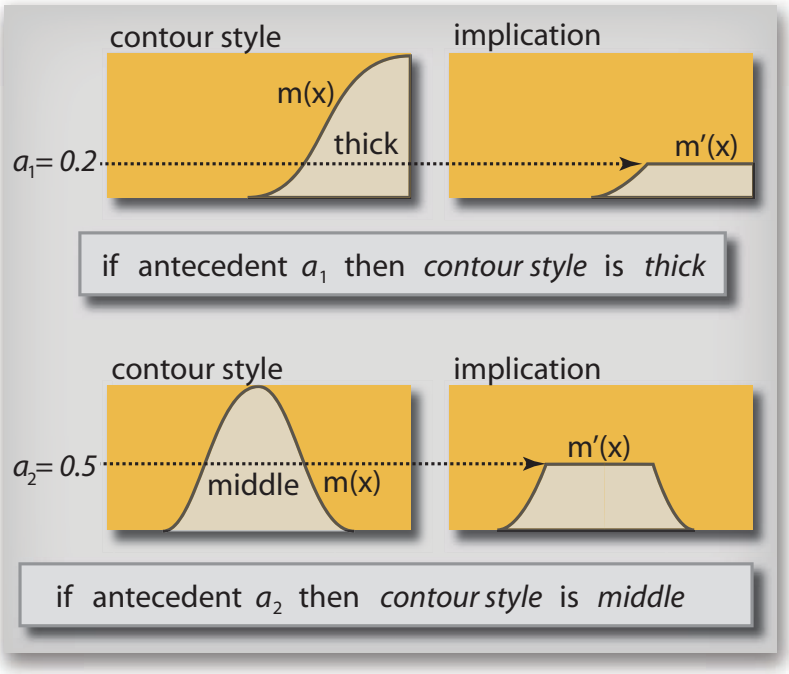


Figure 3.10: *Implication: Two examples for implication are shown that have consequences on the contour style. The membership functions of the consequents are truncated at the height of the evaluated antecedent.*

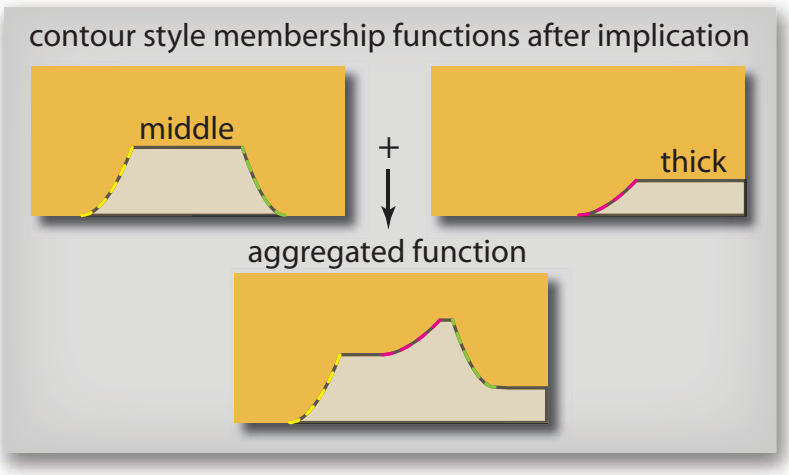


Figure 3.11: *Aggregation: Two membership functions after implication are shown that influence the same style. The sum of all truncated functions influencing the same style is the aggregated function.*

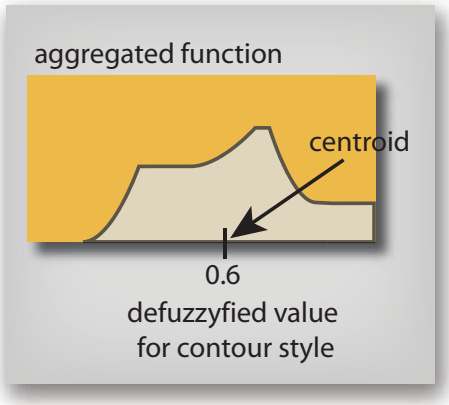


Figure 3.12: *Defuzzification: The aggregated function is defuzzified to get a crisp value for the visual abstraction parameter. In this example defuzzification is done using the centroid method resulting in a value of* 0.6 *for the contour style.*

3.4.3 Aggregation

Aggregation is the operation that combines all results from implication for one specific visual abstraction parameter. In our example the two truncated functions for the *contour style* are combined. Aggregation is commonly done summing up all functions after implication. The result of aggregation for the functions from the example of Figure 3.10 is shown in Figure 3.11. For illustration purpose the increasing and decreasing slopes of the implicated membership functions are dashed and colored. The colors are used again for the corresponding slopes in the resulting aggregated function. From this potentially complicated function a crisp value has to be determined that is used as parameter for the visual abstraction.

3.4.4 Defuzzification

Defuzzification is done to find one crisp value from the aggregated function. Many different defuzzification methods (e.g., smallest-, middle-, and largest of maximum) exist that produce different results. We chose the centroid method that evaluates the centroid of the aggregated function. A desirable property of this particular defuzzification method is that the output varies continuously if the input varies continuously. Other methods that do not have this property might introduce discontinuities in the visual representation. In Figure 3.12 an example of defuzzification can be seen. The result is a value for the parameter of a visual abstraction (i.e., in this example the *contour style*).

3.5 Rendering

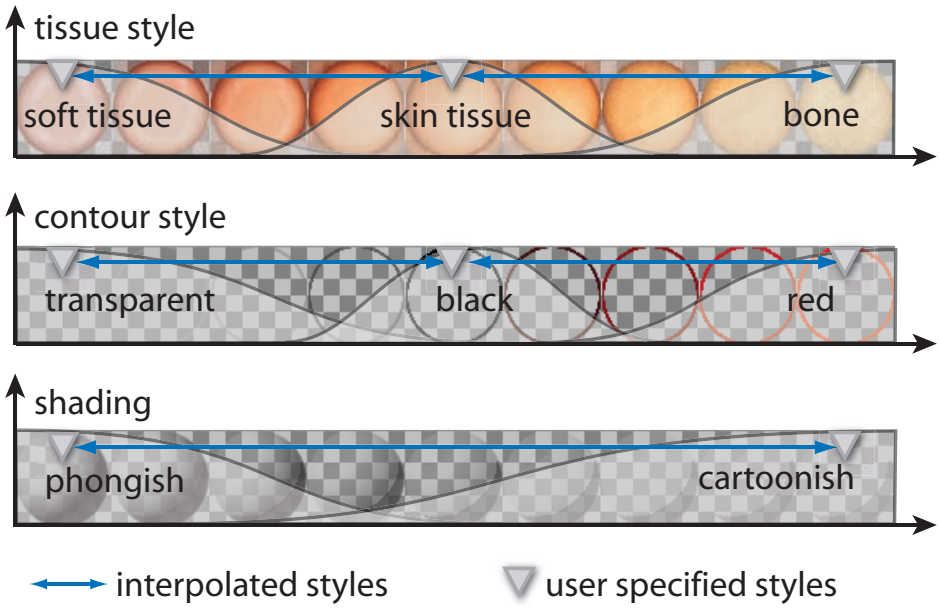


Figure 3.13: *Examples of style descriptors: A few styles are explicitly specified by the user resulting in parameterized styles. Each row shows a few examples of interpolated styles of one style descriptor.*

The visualization rules ensure that visual abstractions (i.e., the illustrative styles in our implementation) are applied gradually and selectively to different regions. The fuzzy logic inference results in a style volume for each specified style. We use a direct volume rendering approach for the visualization of volumetric data. Viewing rays are cast through the volume and sampled at equidistant sample positions. For each sample an opacity transfer function is evaluated determining the visibility of the current sample. Samples with opacity greater than zero are colored according to the style volumes. The styles are described using *style descriptors*. In Figure 3.13 three examples of style descriptors can be seen. Each style is described by a few images of spheres. Each sphere might be manually shaded by an artist simulating the desired artistic style. The small gray triangles in Figure 3.13 indicate the user specified pre-shaded spheres. Each row shows interpolated examples of the style. The top row shows an artistic tissue style often found in medical illustration. The tissue style shows from left to right examples for soft tissue, skin tissue, and bone. The middle row describes different contour styles ranging from transparent, over black to red contour. The last row in Figure 3.13 shows examples for shading styles ranging from phongish shading to cartoonish shading. The styles are applied like layers on top of each other. Additionally the user can specify a value for the

strength of application for each style that is used to blend the layers on top of each other.

Style descriptors provide a uniform framework for the parametrization of styles. The user defines a few spheres that are used to parameterize the styles. Further, the user adds membership functions to assign illustration semantics to the styles. In Figure 3.13 examples of membership functions for the different styles are shown.

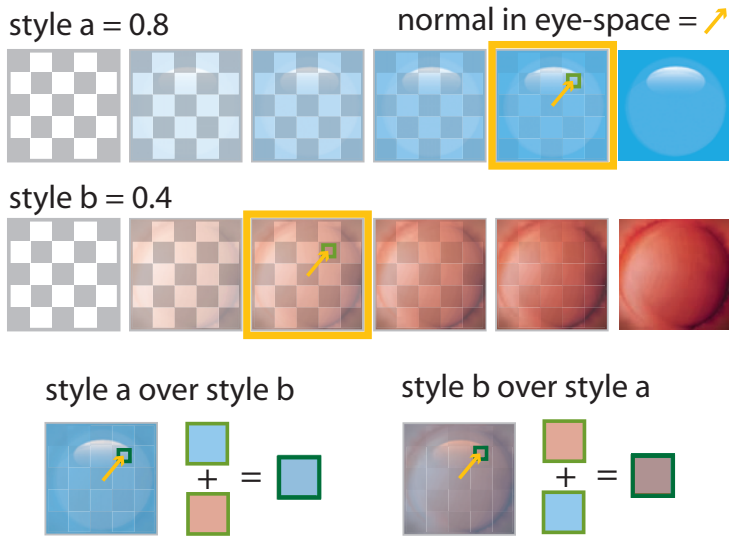


Figure 3.14: *Example of compositing two styles: The exemplary sample has a value of* 0.8 *for style a and a value of* 0.4 *for style b. The corresponding spheres are outlined in yellow.*

For the implementation of style descriptors we use an adaption of style transfer functions [8]. In Figure 3.14 the evaluation of the layered style transfer functions for two styles can be seen. Note that for simplicity in the example both styles vary from transparent to opaque but this is not necessarily the case.

Figure 3.14 shows exemplary values of the style volumes outlined in yellow. In this example the defuzzification for *style a* resulted in 0.8 and for *style b* in 0.4. Following the style transfer function approach, the resulting color for each style depends on the normal of the current sample in eye-space. The yellow arrows in Figure 3.14 indicate an exemplary normal in eye-space. The normal in eye-space is used to index the image of the spheres. In Figure 3.14 the resulting colors for the styles are outlined in light green. The final color of the sample is composited from all used styles. The styles are prioritized and composited from the bottom to the top style, using the over operator. This is also used in image manipulation programs such as Adobe Photoshop or GIMP for layer compositing. In Figure 3.14 two possibilities for the resulting color are shown. The result depends on the priority of the styles. If the priority of *style a* is higher then the priority of *style b*, i.e., style

a over style b, then the resulting style is a blueish sphere and the final color of the sample is blue (outlined in dark green in Figure 3.14). If the priority of *style a* is lower then the priority of *style b*,i.e., style b over style a, then the resulting style is a violet sphere. The final color of the sample is also outlined in dark green in Figure 3.14.

3.6 Results

One of the main contributions of our approach is the separation of the visualization and the domain semantics. Domain scientists model the data semantics, while illustrators and visualization experts model the illustration semantics respectively the visualization mapping. We demonstrate the flexibility of our approach and show exemplary results. We expect our approach to be useful in a wide variety of applications. Therefore we do not concentrate on one specific application, but rather show three different examples. Our approach is capable to produce renderings that are achieved with common transfer function based approaches. Further it provides an alternative interface to describe more complex visualization mappings. Fuzzy logic is known to be robust and also to be able to cope with partially contradicting rules which might exist in several areas. The results show the capability of the system to deal with very different volume attributes and many different styles. In Section 3.6.1 we demonstrate three visualization mappings for different levels of data abstraction. In Section 3.6.2 we briefly report performance measurements for the shown examples.

3.6.1 Levels of Abstraction

We show semantic visualization mappings from three different levels of data abstraction. The first example maps raw data of multiple modalities to visual styles. The second example shows a visualization mapping from the markup *curvature*. The third example demonstrate the semantic visualization mapping from an object-level to visual abstractions. All examples are achieved in short interactive sessions. The modifications of styles is done interactively allowing for a great flexibility in the generation of illustrative renderings.

Multi-Modal Raw Data Visualization Mapping

In Figure 3.15 three images of the monkey atlas dataset are shown. The monkey atlas dataset contains a registered CT and PET scan of a monkey head. The CT data was used for the rendering. The PET data was used for the application of the rule *if brain activity is high contours are red*. In the images the red contours can be

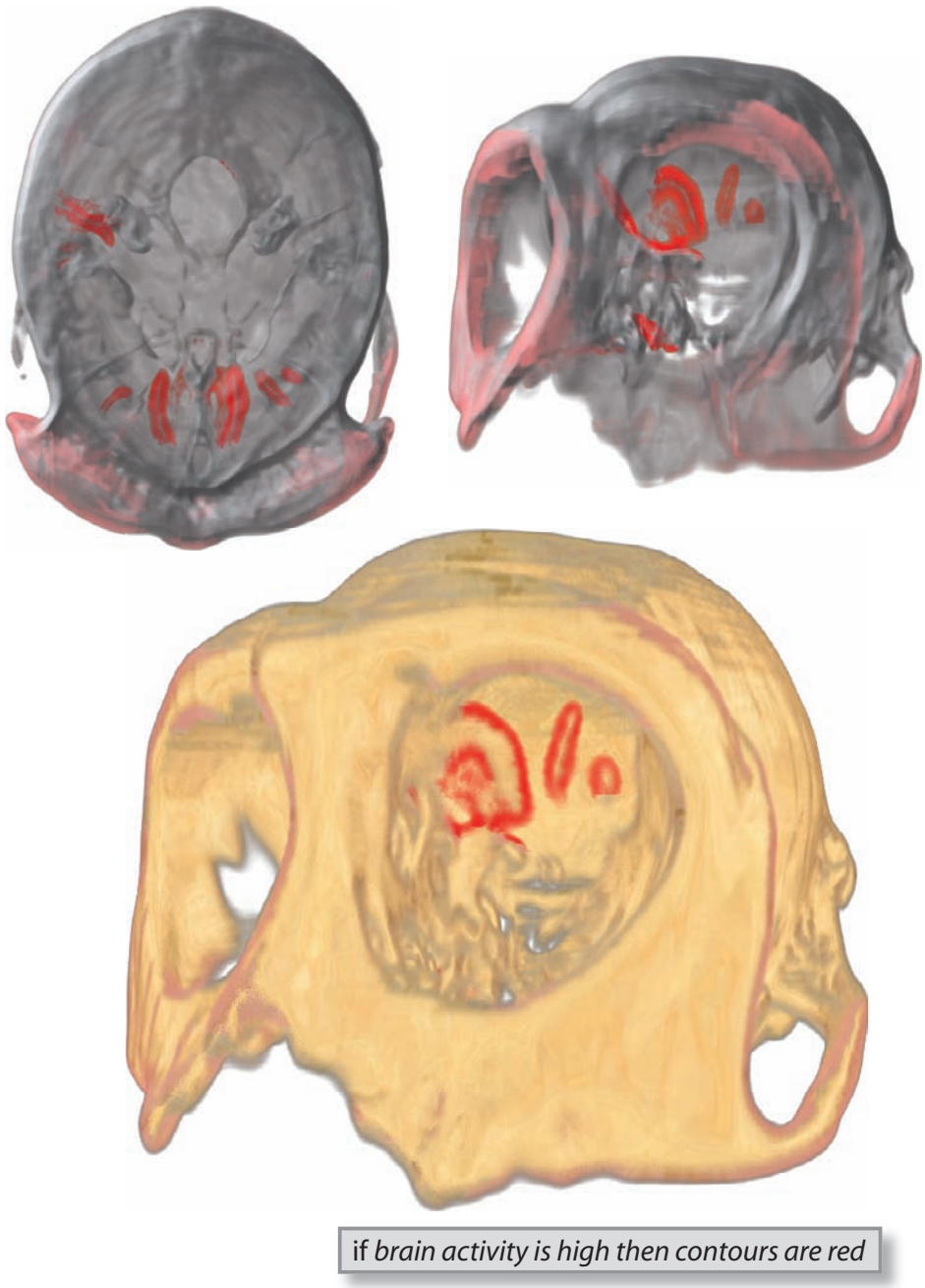


Figure 3.15: *Illustration of the multi-modal monkey atlas dataset. The top images show two different views of the dataset using a transparent opacity transfer function. The lower image shows an illustrative rendering from the same view as the top right image. Red contours are drawn in all three images in region of high brain activity.*

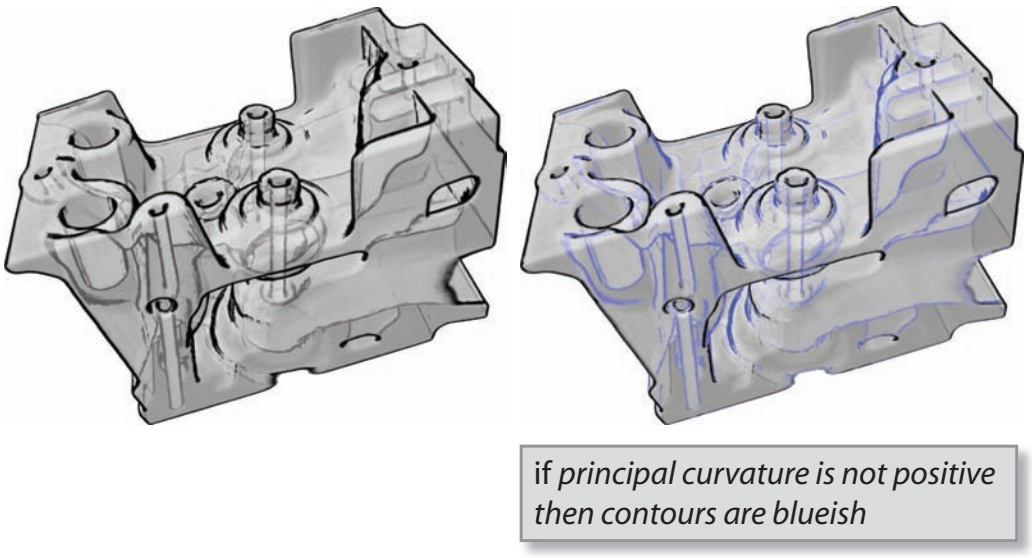


Figure 3.16: *Selective application of styles on the engine block dataset. The left image shows a cartoonish shading effect using contours to enhance the edges. The right image selectively colors the contours. Convex regions are shown in black, whereas concave regions are shown with blueish contours.*

seen. In the top row of Figure 3.15 two images are shown using a semi-transparent opacity transfer function to unveil all regions of high brain activity. The lower image in Figure 3.15 uses a more opaque transfer function and a more illustrative style. The very simple specification of the semantic values and the rule lead to an illustration highlighting regions of interest.

Data Markups Visualization Mapping

In Figure 3.16 the result of a curvature based approach is shown. Two renderings of the engine block dataset are shown. The chosen opacity transfer function and the chosen viewpoint allow the view on many details of the engine block dataset. A cartoonish shading style is globally used and contours are applied to accentuate the individual parts of the engine block. In the right image of Figure 3.16 a rule is applied to selectively apply the contour style. The rule *if curvature is not positive then contours are blueish* is used to distinguish between convex and concave contours in the image.

Object-Level Visualization Mapping

In Figure 3.17 an illustrative rendering of a CTA scan of a human head is shown. The left image of Figure 3.17 uses two different tissue styles for bone and skin. In

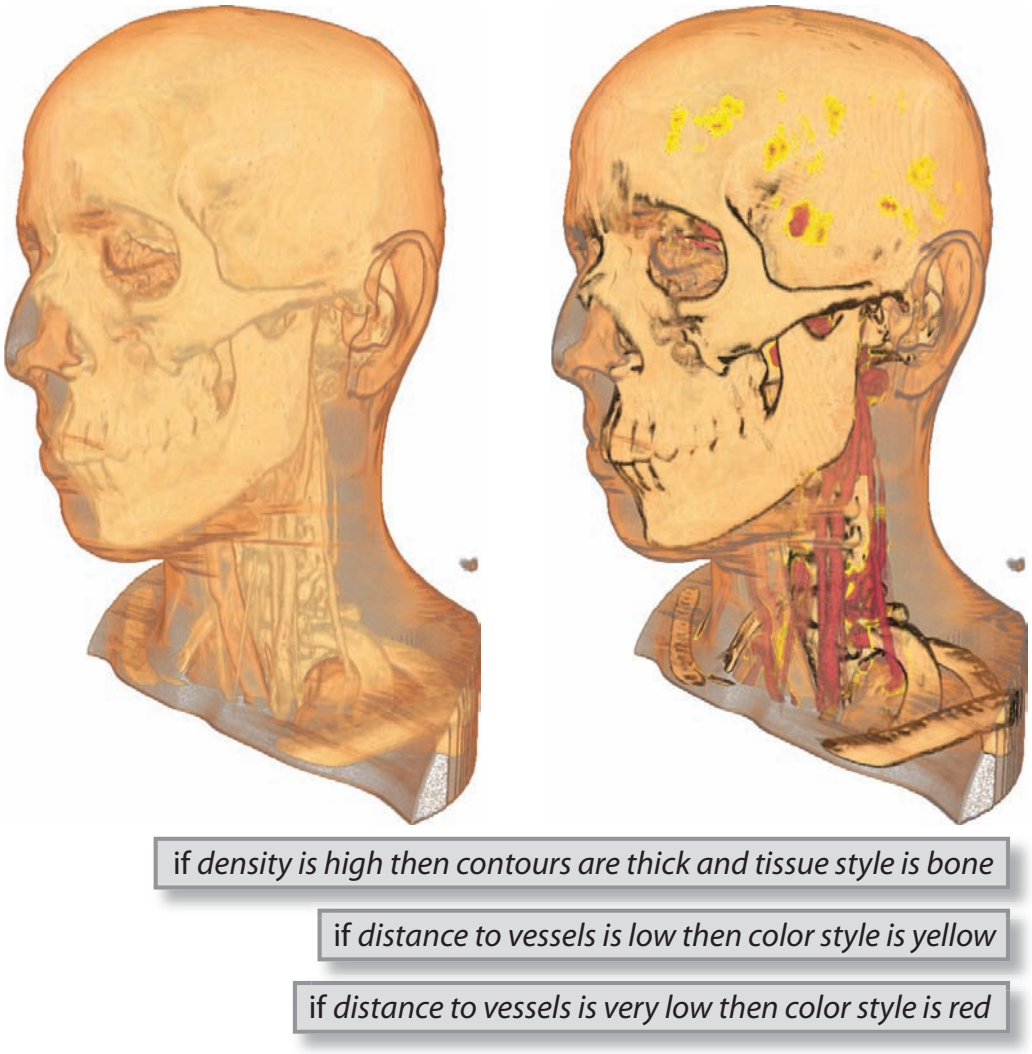


Figure 3.17: *Illustrative rendering of a human head. The left image shows the use of two different tissue styles. The right image shows the selective application of contours to regions of high density. Further a distance based color style was overlayed. Yellow encodes low distance to vessels and red encodes very low distance to vessels.*

the right image of Figure 3.17 black contours are selectively applied on the bone using the simple rule *if density is high then contours are thick*. The major vessels of this dataset are segmented. We used a distance transform of the segmented data to apply a distance based rule. The rules *if distance to vessels is very low then color is red* and *if distance to vessels is low then color is yellow* were used. We applied the style *color* on top of the other styles. Regions of low distance to the vessels can be seen in yellow and red in Figure 3.17. The distance based rendering

was achieved by simply specifying the semantic values *low* and *very low* for the attribute *distance to vessels*, as well as the semantic values *red* and *yellow* for the style *color* and the simple distance based rules.

3.6.2 Performance

Our rendering approach is a GPU based ray-casting implementation. We use one color channel per style. For the evaluation of each style during rendering we do one texture lookup per sample. The styles are blended from the background style to the top most layer. Our approach runs with interactive frame rates on a GeForce 8800 GTX graphics card. We measured the average frame rates for the datasets shown in the result images with a sample distance of 0.5 and a viewport size of 800×600. For the monkey atlas dataset of size $256 \times 256 \times 62$ shown in Figure 3.15 average frame rates from 9 fps to 16 fps were achieved depending on the used opacity transfer function. For the engine block dataset of size $256 \times 256 \times 256$ shown in Figure 3.16 we achieved an average of 7 fps. However using a less transparent opacity transfer function for the same settings resulted in an average frame rate of 20 fps. The average frame rate for the dataset of size $256 \times 256 \times 166$ shown in Figure 3.17 was 9 fps.

3.7 Summary

In this Chapter a novel approach for the specification of the visualization mapping, i.e., the mapping from volume attributes to visual attributes, was presented. Unlike prior work on visualization mapping we introduce the use of semantics. With our approach a linguistic description for the specification of the desired visualization by using semantic values for multiple volume attributes and for visual abstractions is enabled. The novel methodology uses visualization rules to describes *how* different features in the data are rendered.

This approach separates the specification of domain semantics from the visualization of the data. Hence domain experts can specify data semantics independently from, and without prior knowledge about, the visualization algorithm. The rendering algorithm on the other hand is also separated from the data semantics and implements a specific illustration technique. Visualization rules are used to establish expressive visualization mappings between the different semantics. Using this approach we envision systems that offer a great flexibility for the specification of visualization mappings. This opens possibilities to use a wide variety of visualization algorithms that resemble illustration techniques in a uniform framework. We demonstrated that our approach is able to map from various levels of data abstraction to many low-level visual abstractions. The visualization mappings

that are possible to reproduce with this approach include *multi-dimensional transfer functions*, *curvature based transfer functions*, as well as high-level abstraction approaches such as *distance based transfer functions*.

A valuable extension of this approach would be the investigation of automatically (or semi-automatically) derived membership functions. We experienced that the membership function specification for different applications usually follows a similar strategy. The membership functions are typically specified over the whole range of values that occur within the data. Simple heuristics could be used to automatically provide an initial specification of membership functions. The automatic generation of membership functions could be triggered by the keyword *relatively* in a rule. For example a rule stating *if curvature is relatively low …* will be translated into a membership function having a peak at the minimum value of the curvature and is decreasing to a certain value to include a pre-specified percentile of the data.

If you only do what you know you can do - you never do very much.

Tom Krause

Introduction

Caricaturistic Visual Abstraction

Semantic Visualization Mapping

Interactive Illustration

Summary and Conclusions

In traditional illustration the choice of appropriate styles and rendering techniques is guided by the communicative intent. Illustrative volume visualization imitates traditional illustration but has the advantage to be fully interactive and to offer direct control over the illustration technique. The semantic visualization mapping concept establishes a mapping from different levels of data abstraction to the visual abstractions of illustration techniques using visualization rules. However, the specification of the interactive behavior of the illustration is even more challenging. In this Chapter an approach is shown that evaluates the user defined illustration rules directly on the GPU. This drastic increase in performance allows the use of interaction dependent semantics that define the behavior of the illustration. The work described in this Chapter was published in *Computer Graphics Forum* [50].

Chapter 4

Interactive Illustration

4.1 Introduction

Medical doctors use simple illustrations for the purpose of patient briefing. The illustrations describe a specific diagnosis, the future treatment of diseases, or a planned surgical intervention. In the optimal case patients are shown illustrations that are consistent with their anatomy and the special instance of their disease. However, hand drawn (traditional) illustrations are elaborate pieces of art usually done in a very time consuming way. Therefore it is infeasible to create high quality hand-drawn illustrations for each patient.

One goal of illustrative medical visualization is to produce patient specific illustrations derived from measured data. CT-scans or MRI-scans provide measurements of the patients anatomy and are used to automatically generate illustrations. The illustrations are dependent on the intent and therefore constrained by the diagnosis, the treatment, or the possible surgical intervention they should convey. These interactive illustrations commonly provide a user interface to manipulate basic settings like the viewpoint. More advanced interactive illustrations show a special behavior like fading in and out effect of regions of interest that is triggered by the user. These interaction methods offer a more flexible approach for laypersons like patients to explore the illustrated subject. However it is more challenging to implement such interactive illustrations and commonly each interaction metaphor is implemented in a separate algorithm.

In this Chapter we discuss the concept of interaction-dependent semantics for illustrative rendering of volume data. Our approach uses the semantic visualization mapping concept for the specification of the interactive behavior of the illustration. Interaction-dependent visualization rules are introduced that enable the use of many interaction metaphors. In Figure 4.1 an outline of the interactive illustration system is shown. The central component in our system is the semantic visualization mapping approach. The visualization rules use different types of semantics. Data semantics depend on the available data. CT-scans, for example, provide information on tissue densities. Different ranges of densities correspond to semantically

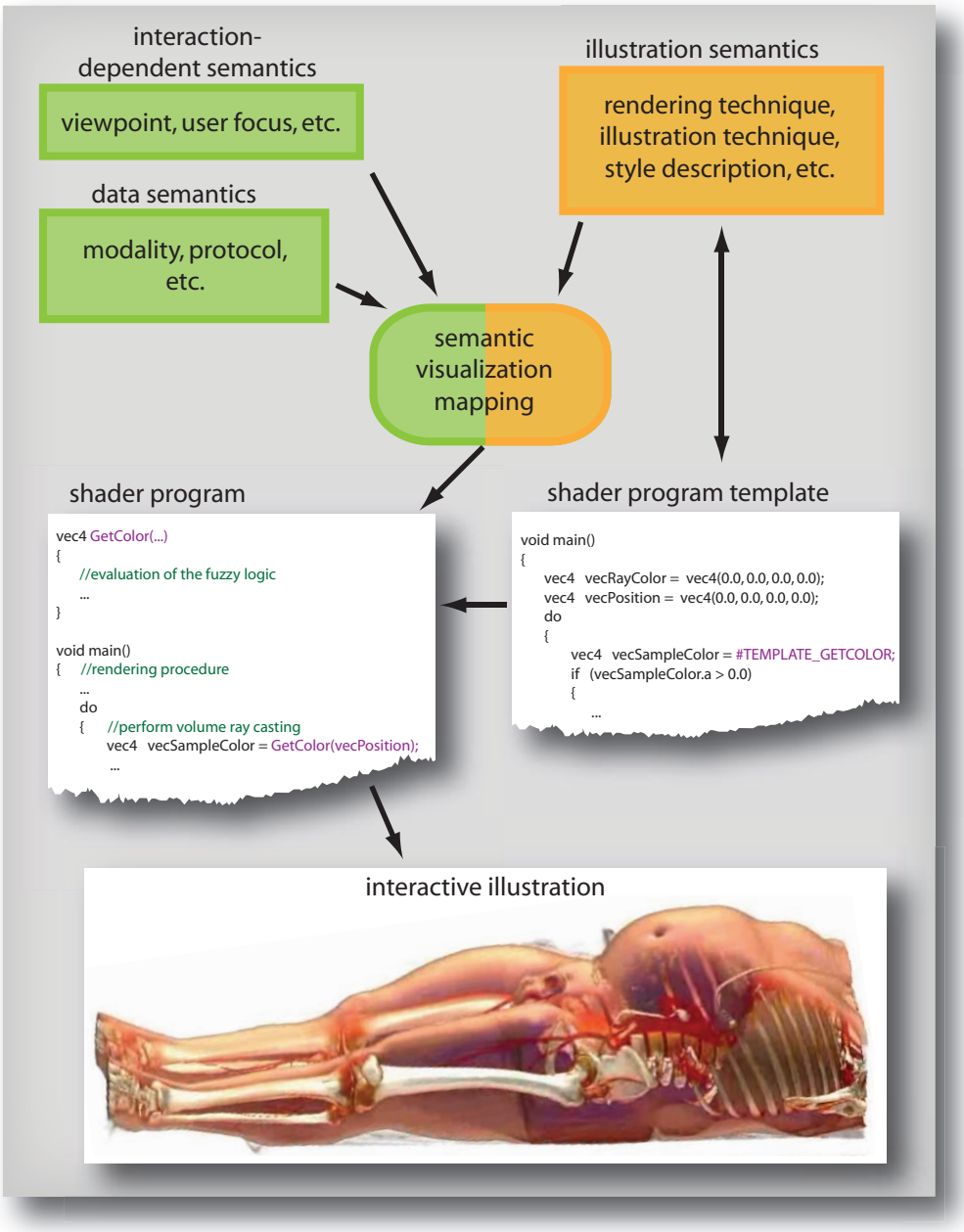


Figure 4.1: *Overview of the interaction-dependent semantics-driven rendering framework: The different types of semantics guide the generation of the interactive illustration and determine its behavior during user interaction.*

meaningful entities (like air, soft tissue, bone, metal, etc.). Interaction-dependent semantics originate from the interactive illustration itself. Examples are the di-

rection the depicted object is viewed from, the distance between features and the image plane, and the region of user focus (e.g., the position of the mouse cursor). These interaction-dependent parameters are used in the visualization rules to alter the illustration behavior during user interaction. This concept allows a textual specification of the interactive illustration.

As shown in Figure 4.1 different types of semantics are used for the visualization rules. The *if* part of rules state constraints using data semantics and interaction-dependent semantics. The *then* part of rules describes the consequences for the illustration using illustration semantics. Illustration semantics originate from the illustrative rendering method that resembles a traditional illustration technique and is implemented in a shader program template. Illustrators use specific terms for the description of styles and of rendering techniques. Examples include: the way regions of interest are emphasized and the remaining features are drawn to provide context, the description of rendering styles, and the way spatial relations are emphasized or to the contrary ignored to depict occluded structures. As shown in Figure 4.1 a shader program template is implemented that offers specific illustration semantics. These semantics are used in the semantic visualization mapping. A compiler translates the visualization rules to executable shader code that is inserted in the shader program template. The use of shader program templates allows to implement a wide variety of illustration techniques that are directly controlled by the visualization rules. The evaluation of the visualization rules is done using fuzzy logic. The translation of fuzzy logic into executable shader code makes it possible to evaluate the rules directly on the GPU.

In this Chapter first related work is reviewed in Section 4.2. In Section 4.3 we introduce interaction-dependent semantics that are evaluated for every frame. Adjustable slicing planes, the position of the mouse cursor, and view-dependent parameters are manipulated by the user. Semantics like *distance to the mouse cursor*, *distance to the slicing plane*, *distance to the image plane*, etc. are used in the antecedent of rules to alter the behavior of the interactive illustration.

We describe the GPU based evaluation of the fuzzy logic pipeline in Section 4.4. All steps in the fuzzy logic reasoning are evaluated directly on the GPU. The shader program is automatically generated according to the fuzzy logic rule base and adapted every time the rule base changes. The GPU based implementation allows interactive evaluation of the fuzzy logic, enabling interaction-dependent semantics.

To show the strength of interaction-dependent illustration behavior we introduce a novel rendering mode in Section 4.5. Illustrators often ignore spatial relations and draw layers with more important features on top of other layers. This results in a flat depiction of the important features on top of more contextual regions. We show the possibility to influence arbitrary rendering parameters with interaction-dependent rules and demonstrate the capability to influence illustration

techniques like the *flat rendering mode* with semantic visualization mapping rules.

4.2 Related Work

Our approach is a general rendering concept that uses visualization rules to translate illustration techniques into images. Because of the wide variety of diverse illustration techniques that can be achieved with this framework extensive related work exists. However, many of the related works have already been discussed in Chapter 3.

Our flat rendering mode is similar to the importance-driven volume rendering approach by Viola et al. [74]. Krüger et al. [36] show a technique for the visualization of hot spots. Our system allows similar results with the introduction of interaction-dependent semantics. Similar in appearance is the context-preserving rendering technique by Bruckner et al. [5], that suppresses regions of low curvature in viewing direction. Inner structures are visible but high curvature regions of outer structures are preserved. This technique is also an example of interaction-dependent volume visualization since the curvature measure depends on the viewing direction.

4.3 Interaction-Dependent Semantics

Semantics-driven rendering makes use of the semantics that accompany the process from acquiring data to drawing an illustration. We use the semantics in a fuzzy rule base. Rules employ data semantics such as *if density is high then…*, *if diffusion is low then…*, or *if curvature is high then…*. Interaction-dependent semantics are represented in the rule base by rules that contain attributes that have to be evaluated during user interaction. Examples are *if distance to slicing plane is low then…*, or *if user focus is close then…*. Here the attributes *distance to slicing plane*, and *user focus* change their values for every frame due to user interaction. Rules using these interaction-dependent semantics determine the behavior of the illustration. The rules can further use any logical combination of the above mentioned semantics such as *if distance to slicing plane is low and density is high then…*. In fuzzy logic the antecedent of a rule is not simply true or false but can have any transitional value in between. The consequent describes the consequences for illustrative styles if the antecedent of a rule is not false. The consequent in our system describes the resulting rendering attributes, like *…then bone style is transparent* or *…then contours are thick*.

As described in Chapter 3 styles are parameterized in our rendering framework. Each parameter is evaluated separately using all fuzzy logic rules that have conse-

quences for the parameter. The antecedents of the rules are evaluated describing to which degree a rule is *true*. Implication, aggregation, and defuzzification are the remaining steps that are performed to derive a value in the interval $0..1$ for each style.

The interaction-dependent semantics that are used in the antecedents potentially change for every frame and make it necessary to evaluate the rules per frame. Implementing a volume rendering system that evaluates all fuzzy rules per sample for every frame is challenging. Modern CPUs are not capable of evaluating fuzzy logic rules for a 3D volume several times per second. On the other hand modern GPUs do not offer the flexibility to fully implement a fuzzy logic system. Our implementation makes use of the flexibility of CPUs and the processing capabilities of modern GPUs.

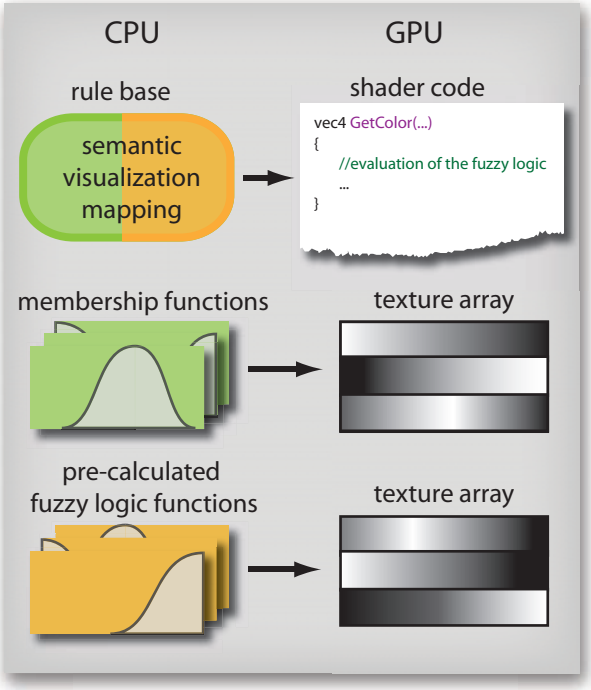


Figure 4.2: *The fuzzy logic rule base is parsed and translated into shader code on the CPU. Membership functions and pre-calculated fuzzy logic functions are encoded in 1D texture arrays. The GPU makes use of the generated shader program and the texture arrays to perform interactive semantics-driven illustrative rendering.*

Figure 4.2 shows the components used on the CPU and the corresponding components on the GPU. The rule base is translated into shader code on the CPU. The shader code is used to generate a shader program for volume rendering performed on the GPU. The membership functions as well as pre-calculated fuzzy

logic functions are stored in 1D texture arrays that are used on the GPU to efficiently evaluate these functions. Figure 4.2 shows three membership functions and three pre-calculated fuzzy logic functions that are translated into two 1D texture arrays with three entries each. Each line in the texture array is a discrete representation of the functions with values from zero to one (black representing zero and white representing one).

The shader program is adapted automatically for every change of the rule base. The update of the shader program is the most costly operation in our system. However it is only done when rules change and takes less than one second on a modern desktop PC. Changes in membership functions result in an interactive update of the corresponding 1D texture arrays.

4.4 Fuzzy Logic on the GPU

Our framework parses the fuzzy logic rules and generates appropriate shader code for the fuzzyfication, fuzzy logic operations, aggregation, implication and defuzzification. The entire fuzzy logic is carried out on the graphics hardware allowing for interactive semantics-driven volume rendering. In the following we describe the fuzzy logic used in our framework.

4.4.1 Evaluation of Antecedents

The evaluation of the antecedents is done using fuzzy logic operations. The antecedent of each rule consists of a logical combination of semantic values of attributes (e.g., *distance to cursor* has semantic values like *low*, *middle*, etc.) The membership functions are evaluated for each attribute that occurs in the antecedent of the rule and combined with the fuzzy logic operations *and* (resulting in the minimum of the operands), and *or* (resulting in the maximum of the operands). Further the unary *not* operation can be used and is evaluated as one minus the operand.

In our implementation the rules are parsed and translated into shader code. We build a fuzzy logic expression tree containing the operations *and*, *or*, and *not*. The nodes of the expression tree are substituted with the corresponding operations *min*, *max*, and $1.0-\ldots$. The leafs of the tree are the operands of the fuzzy logic expression, i.e., the membership functions. We store the membership functions in 1D texture arrays. We substitute the leaf nodes of the expression tree with texture lookups in the 1D texture array and expand the expression tree to generate valid shader code. Figure 4.3 shows an example of a simple rule, the constructed expression tree, and the translation into shader code. First, the rule is split into the antecedent and the consequent part searching for the keywords *if* and *then*. The antecedent is then parsed and split into fuzzy logic operations (i.e., *and, or, not*)

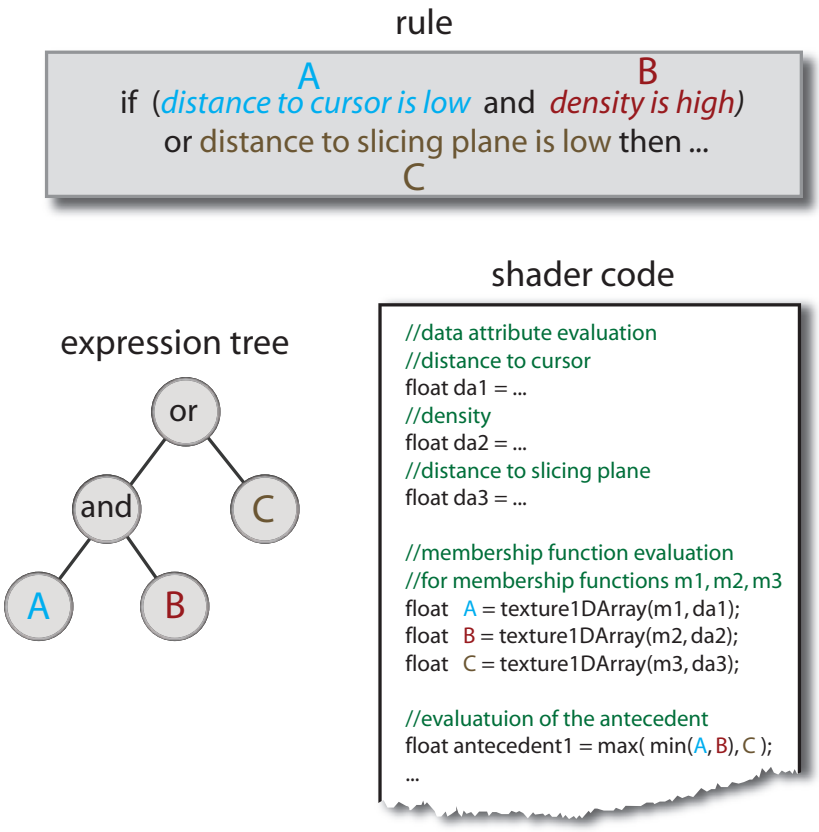


Figure 4.3: *Shader code generation for the evaluation of antecedents. An expression tree and the corresponding shader code are generated from a simple rule.*

and the operands (i.e., the used semantic values - A, B, and C in the example of Figure 4.3). A hierarchy, i.e., the expression tree, is build according to the used brackets and operands. In Figure 4.3 the rule has the structure *if ((A and B) or C) then* .... Finally the expression tree is translated into shader code. The shader code for the evaluation of the used attributes *distance to cursor, density, etc.* is generated. The used membership functions are evaluated in the shader with texture lookups. Shader code for the evaluation of the antecedent is generated translating the fuzzy logic operations to the corresponding mathematical operations, i.e., *min, max,* $1.0 - \ldots$. The resulting shader code is used for the interaction-dependent evaluation of the antecedents.

4.4.2 Implication, Aggregation and Defuzzification

The evaluated antecedent has an implication on the consequent of a rule. Consequents consist of a list of semantic values for visual abstractions that are affected by the antecedent. Semantic values for visual abstractions are also represented by membership functions. Let the value of an antecedent of rule i be $a_i \in [0, 1]$ and the membership function of a semantic value for a given style be $m(x)$ then the implication on this membership function is given by:

$$m'(x) = min(m(x), a_i) \tag{4.1}$$

This results in a truncation of the membership function at the height of a_i. Aggregation is the process of building the sum of all membership functions after implication. Aggregation results in one function for each visualization parameter. Defuzzification is done to derive a crisp value for each parameter. We used the centroid method for defuzzification. The centroid of the aggregated function is the value that is used as rendering parameter.

Implication, i.e., the truncation of the membership function, aggregation, i.e., the sum of the implicated functions, and defuzzification, i.e., the evaluation of the centroid of the aggregated function, are operations that are not straightforward to implement on the GPU. However, we show that the computationally most expensive tasks can be precomputed and stored in 1D texture arrays.

For defuzzification we want to find the centroid of the aggregated function. Let $f(x)$ be the result from the aggregation, then its centroid c_f is given by the equation:

$$c_f = \frac{\int x f(x) dx}{\int f(x) dx} \tag{4.2}$$

Let the semantic values respectively the membership functions of one visual abstraction parameter be $m_j(x)$ where $j \in \{1 \dots n\}$ and n be the number of membership functions for one parameter. The membership function for the semantic value affected by rule i after the implication is then given by the equation

$$m_i{}'(x, a_i) = \min(a_i, m_i(x)) \tag{4.3}$$

where a_i is the antecedent value of the rule i. The aggregated membership function $f(x)$ is then given by

$$f(x) = \sum_{i \in I} m_i{}'(x, a_i) \tag{4.4}$$

where I is the set of indices of rules that affect the given style. The centroid of the aggregated function can then be calculated by substituting Equation 4.4 in Equation 4.2:

$$c_f = \frac{\int x \sum_{i \in I} m_i{}'(x, a_i) dx}{\int \sum_{i \in I} m_i{}'(x, a_i) dx} \tag{4.5}$$

We can rewrite Equation 4.5 as follows:

$$c_f = \frac{\sum_{i \in I} \int x m_i{}'(x, a_i) dx}{\sum_{i \in I} \int m_i{}'(x, a_i) dx} \tag{4.6}$$

In Equation 4.6 it can be seen, that the integrals (i.e., the summands in the nominator as well as in the denominator) do solely depend on the a_i. Equation 4.6 thus can be simplified to:

$$c_f = \frac{\sum_{i \in I} n_i(a_i)}{\sum_{i \in I} d_i(a_i)} \tag{4.7}$$

where n_i are the summands of the nominator and d_i are the summands of the denominator, that do not depend on x. This allows us to pre-compute the summands n_i as well as d_i and to store them in a lookup table.

Figure 4.4 shows an example of shader code generation for two rules affecting the same style. The results of antecedent evaluation are used as indices for the lookup tables to evaluate the pre-computed nominators and denominators. Shader code is generated for the lookups and for the evaluation of Equation 4.7. The Equation 4.7 has to be evaluated for each visual abstraction. For one visual abstraction parameter a total of $2m$ texture lookups (for the precomputed nominators and denominators), $2(m-1)$ summations and one division, where m is the number of rules that affect the same attribute, have to be evaluated.

4.5 Flat Rendering

The flexibility of our framework is achieved using shader program templates. A shader program template implements a rendering technique. It specifies rendering attributes, that can be used in the consequent of rules. Placeholders are put in the shader program template at the fuzzy logic specific parts of the rendering procedure. The parts containing the fuzzy logic evaluation of the rendering attributes are generated automatically and fill in the missing parts of the shader program template.

We implemented the flat rendering technique to demonstrate the possibilities of an interactively changing illustration. The shader template of the flat rendering mode extends the style transfer function approach. The flat rendering mode resembles an illustration technique that ignores spatial relations of features of interest. Each style is assigned an additional importance by the user. Regions of higher priority are always rendered in front of regions of lower importance in order to show hidden structures. Regions in the volume that use styles of higher priority overdraw regions with lower priority. This is conceptually similar to the work of Viola et al. [74]. However, we use this method to apply it gradually and selectively, driven by the visualization rules.

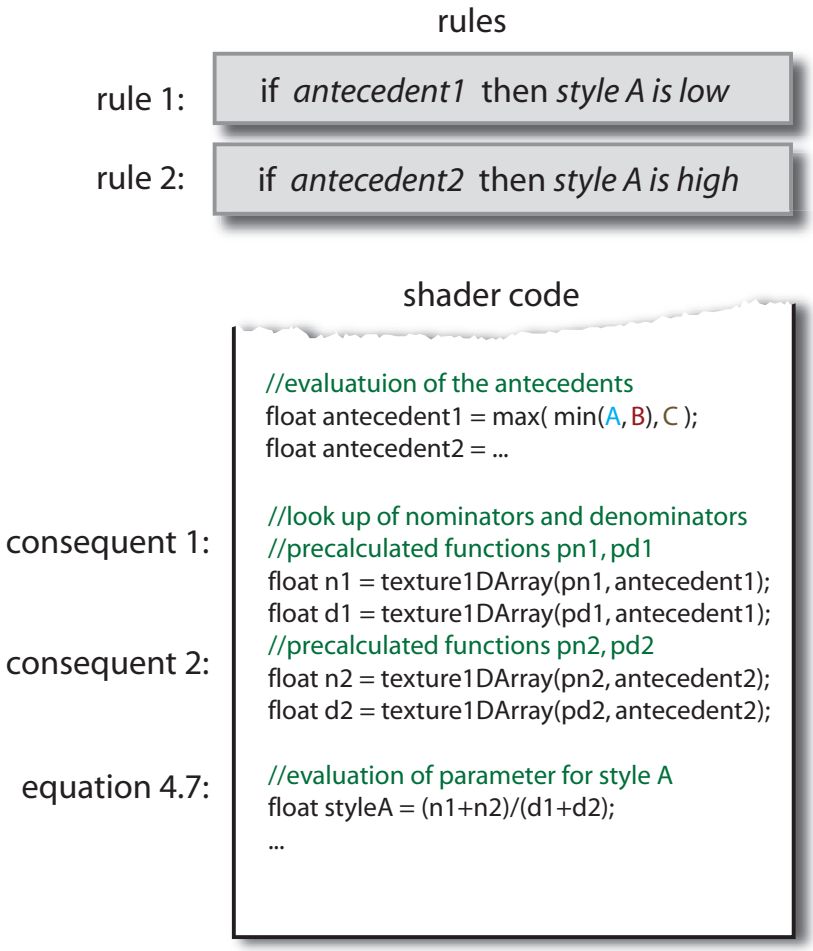


Figure 4.4: *Shader code generation for the implication, aggregation and defuzzification for two rules affecting the same style.*

Figure 4.5 depicts the rendering process using the flat rendering mode. The dashed yellow line shows a viewing ray. The blue and red boxes denote two regions that use different styles according to specific rules. Samples along the viewing ray are evaluated and composited. If the ray reaches a region of higher priority the ray color is influenced according to the flatness parameter. A flatness parameter of 0 results in common volume rendering. A flatness parameter of 1 always shows the regions with highest priority. At each position the ray enters a region of higher priority the ray color $c_r(x_i)$ is set to:

$$c_r(x_i) = c_r(x_{i-1})\,(1 - p_f) \tag{4.8}$$

where x_i is the current sample position, x_{i-1} is the position of the last sample and p_f is the flatness parameter.

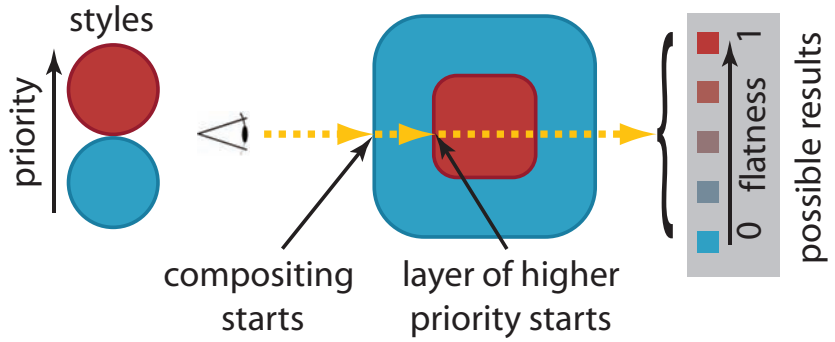


Figure 4.5: *The flat rendering mode favors samples of higher priority during ray casting. The yellow line depicts a viewing ray. Along the ray common compositing is done until a region of higher priority is reached. The composited color is deemphasized according to the flatness parameter.*

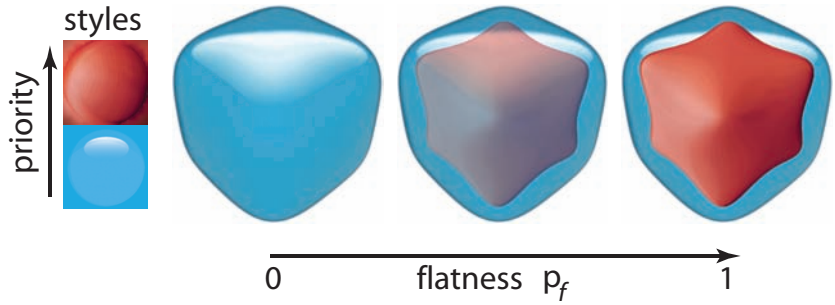


Figure 4.6: *Example renderings using different values of the flatness parameter. The inner cube has higher priority and is therefore shown for a flatness parameter greater than zero.*

Figure 4.6 shows a volume rendering of a cube dataset where densities increase towards the cube center. A simple rule states that the reddish style is high for regions of high density. The leftmost rendering of Figure 4.6 shows just the outer surface of the cube. The region with the style of higher priority remains hidden. The rendering in the middle of Figure 4.6 uses a flatness parameter of 0.5 and the rightmost rendering a flatness parameter of 1.0.

In all three examples of Figure 4.6 the flatness parameter is set globally. However, the *flatness* is a semantic parameter that describes the trade-off between showing spatial relationships and showing important regions. The *flatness* as any other rendering attribute offered by shader program templates can be used in the consequents of fuzzy rules and is dynamically evaluated per sample. This results in a local semantics-driven application of the *flatness* parameter.

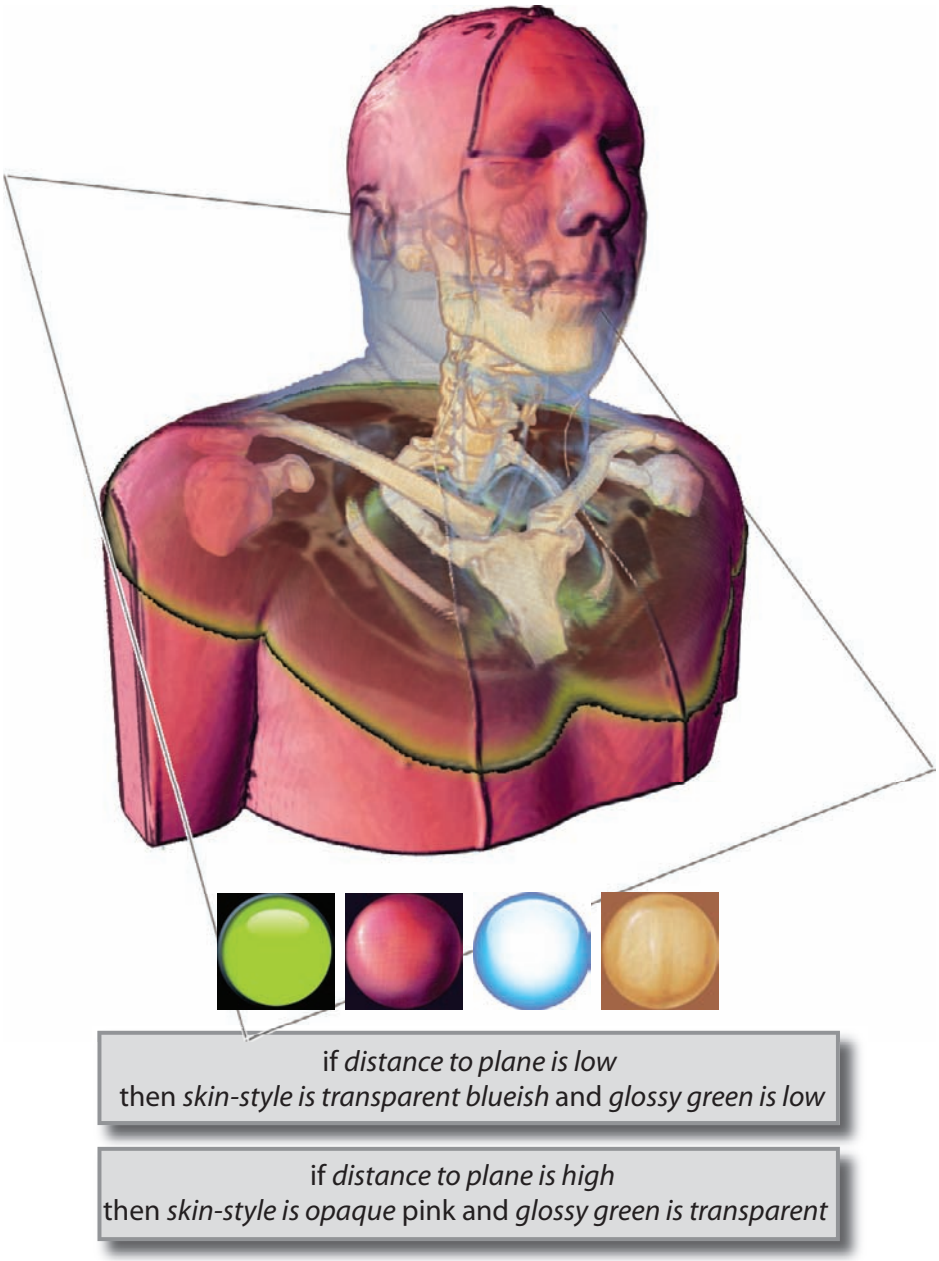


Figure 4.7: *Rendering of the visible human dataset. A slice plane of the histological data is shown. The CT-data is used for the volume rendering providing the context for the slice plane.*

4.6 Results

The evaluation of the fuzzy logic on the CPU takes a few seconds, making it impractical to render interaction-dependent illustrations. The presented GPU based implementation enables the use of interaction-dependent rules designing the behavior of the illustration. Interaction-dependent semantics are capable to produce renderings that put emphasis on a specific focus region and deal with all kinds of view-dependent semantics. Examples for interaction-dependent semantics include *the distance to the image plane* (that allows techniques like depth cueing, depth of field, etc.), *the viewing angle* (e.g., the volume is rendered in a *blue-print* style if it is viewed from the top and in a more tangible style when viewed from the side), etc. Time-dependent semantics are also a subclass of interaction-dependent semantics that can be used to alter the rendering of specific regions over time. We show examples of interactive illustrations that can be achieved with our system and demonstrate the possibilities of interaction-dependent semantics and the view-dependent evaluation of the fuzzy rules. The rules used for the generation of the results are shown in the respective Figures. All results were achieved in interactive sessions with a GeForce 8800 GTX graphics card. No pre-segmentation of the data was used for the shown examples. Renderings of a view port size of 512^2 and a sample distance of 1.0 are achieved at an average of 23 fps for Figure 4.7, of 20 fps for Figure 4.8, and of 6 fps for Figures 4.9 and 4.10 with the respective rules and styles applied. The images shown in the Figures are rendered with higher resolution, a higher quality gradient estimation scheme, and a lower sample distance. However an interaction mode is implemented that automatically adjusts the rendering quality ensuring responsiveness during user interaction.

In Figure 4.7 an illustration of the upper part of the visible human dataset is shown. A slicing plane is used to specify a focus region. The slicing plane additionally shows the histological cut data of the visible human dataset. The spheres used to define the styles are shown at the bottom of Figure 4.7. The left most style is applied in regions very close to the slicing plane. The second and third styles are used to render the skin. Rules that depend on the distance to the slicing plane are specified to modulate the style used for the skin. The right most style is used for regions of high density, i.e., bones. The dataset has a size of 256^3.

In Figure 4.8 an interactive illustration of a human body is shown. The user focus is defined at the position of the mouse. Rules using the distance to the mouse define the appearance of the skin. The skin is shown completely transparent close to the mouse, in unshaded white at a farther distance and in pink for high distances. The spheres used for the styles are shown on the left of Figure 4.8. The two styles on the left are used for the skin color. The two styles on the right are used for the bones and the vessels and are applied according to the rules that solely depend on the density. The dataset shown in Figure 4.8 has a size of $256^2 \times 415$.

In Figure 4.9 a rendering of a CT-scan of a human leg is depicted. A slicing plane is used that shows the CT-data. Rules dependent on the distance to the slicing plane are specified to influence the transparency of the skin and the soft tissue. Skin and soft tissue close to the slicing plane are made transparent. The spheres used to create the styles for skin regions, soft-tissue regions, bone regions, and the metallic implants of the patient are shown in Figure 4.9. The flatness is fully applied.

For the renderings shown in Figure 4.10 rules were specified that influence the *flatness* of the illustration in dependence on the distance to the slicing plane. The flat rendering mode is used gradually only in regions of middle distance to the slicing plane. This results in illustrations, that preserve the spatial relations close to and far away from the slicing plane, but ignore spatial relations in between. The dataset shown in Figures 4.9 and 4.10 has a size of $147 \times 162 \times 429$.

These simple examples show the power and flexibility of our approach. The system is easily extensible for other interactive illustration scenarios due to the use of shader program templates.

4.7 Summary

We present a rendering concept for interactive illustrations that is based on fuzzy logic rules evaluated on the GPU. The rules linguistically define a mapping from data attributes and interaction-dependent parameters to visual styles and rendering techniques. Our framework handles a great variety of rendering techniques in a uniform way. We showed the interactive semantics-driven specification of rendering attributes such as the flatness parameter of the flat rendering mode.

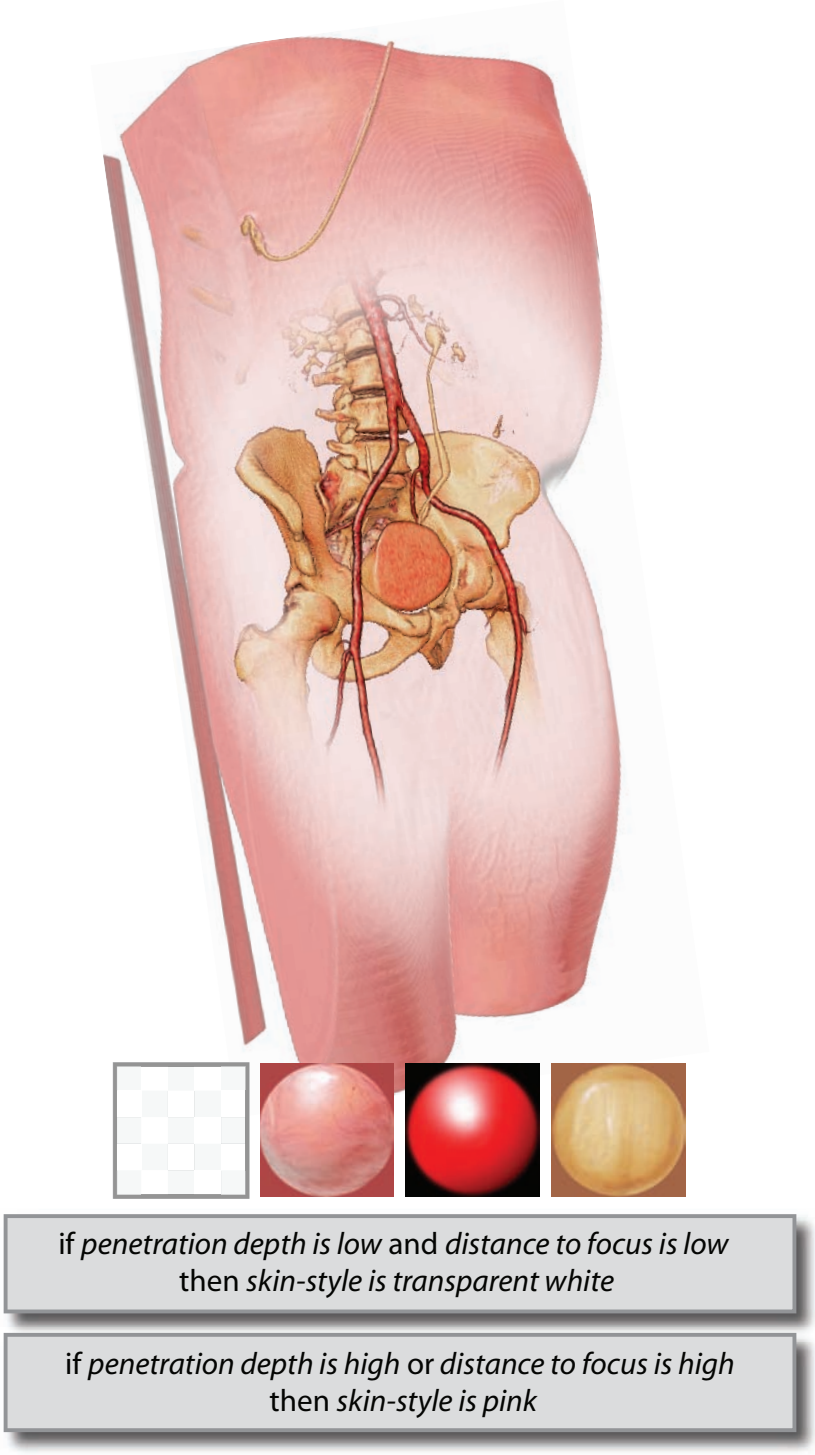


Figure 4.8: *The mouse cursor defines the user focus. Depending on the user focus the illustrative rendering is altered.*

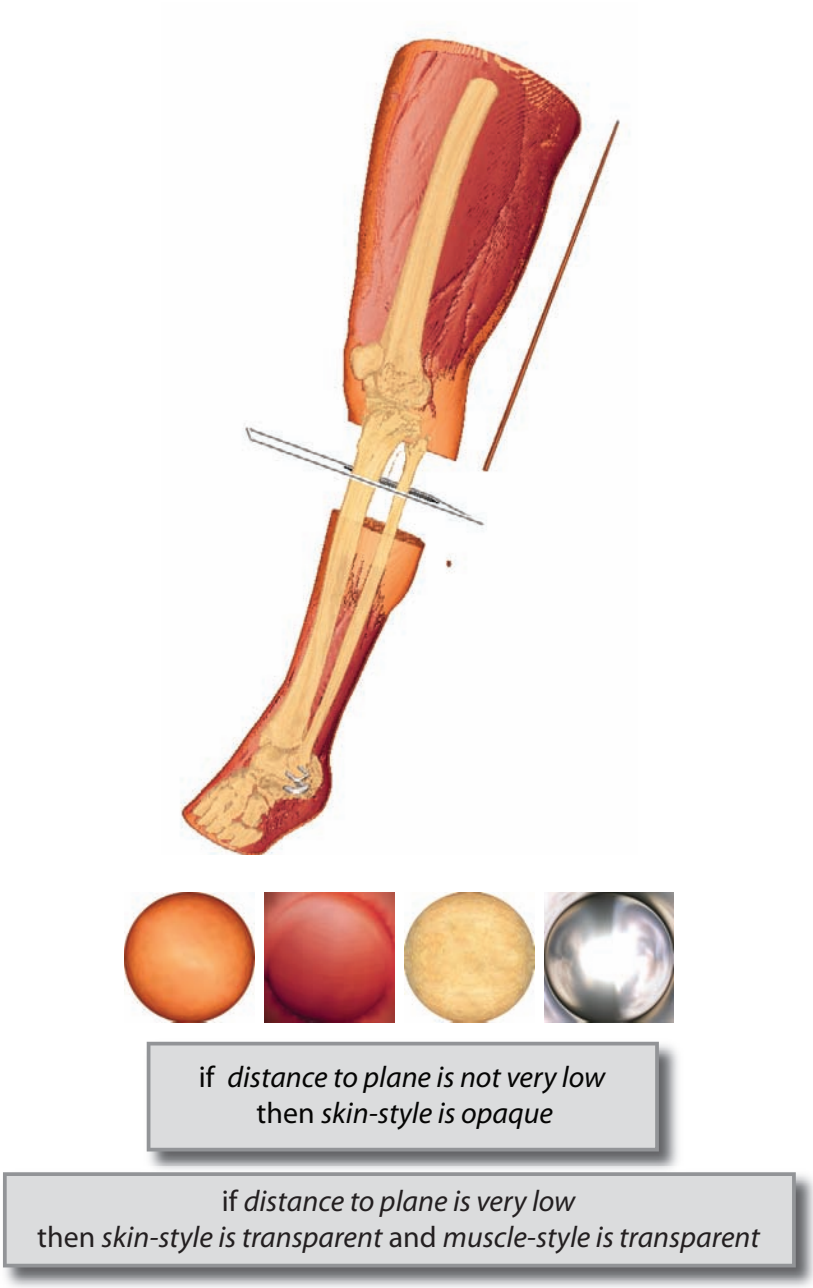


Figure 4.9: *Rendering of a CT-scan of a human leg. The flat rendering mode is fully applied ignoring the spatial relations.*

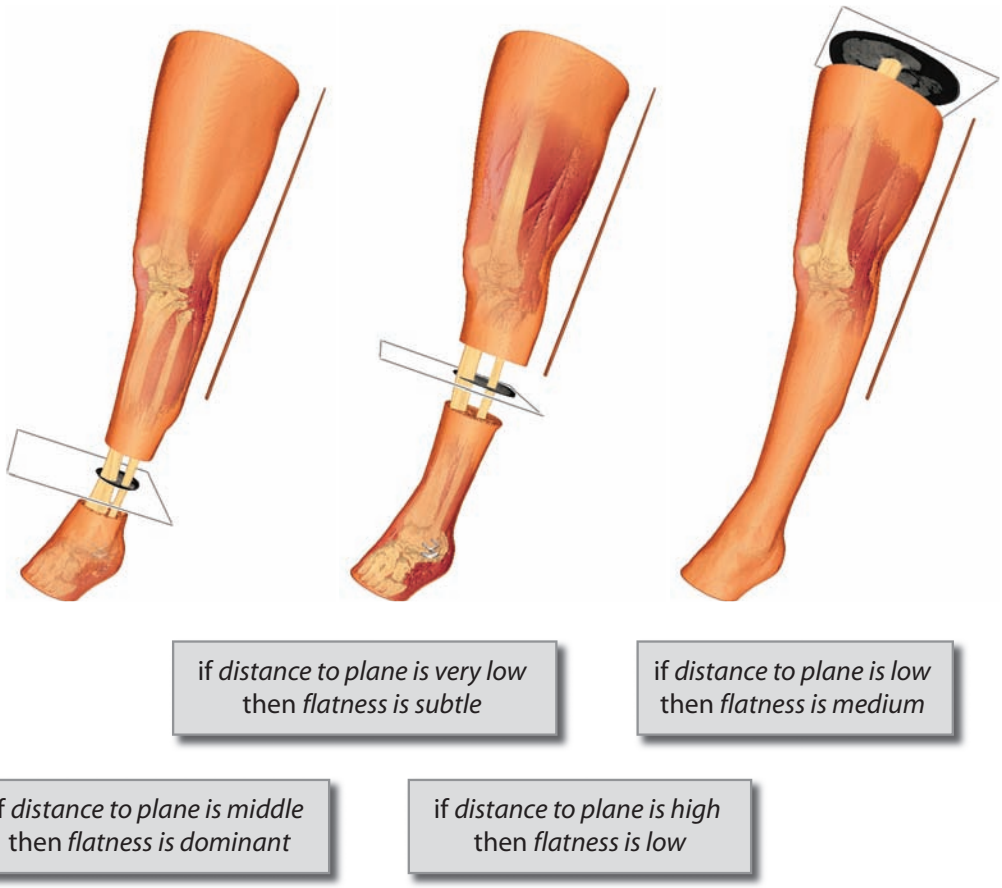


Figure 4.10: *The distance to the slicing plane influences the rendering styles and the flatness parameter.*

There's no hurry any more when all is said and done.

Björn Ulvaeus, ABBA

Introduction
Caricaturistic Visual Abstraction
Semantic Visualization Mapping
Interactive Illustration

Summary and Conclusions

The previous Chapters were dealing with the history as well as novel methods of visual abstractions and visualization mapping in the context of scientific visualization. In this Chapter the thesis is briefly summarized and conclusions are drawn.

Chapter 5

Summary and Conclusions

The ever increasing complexity of data, forces scientists to use advanced visualization methods for the exploration of raw data, the analysis of data abstractions, and the presentation of facts and findings. *Visualizing data* effectively requires adequate visual abstractions and a mapping from the data domain to these visual abstractions that convey the communicative intent. Early examples of visualizations such as scientific illustrations are hand crafted pieces of art. In the course of visualization research much attention was devoted to the development of novel visual abstractions and the imitation of artistic techniques. Recently illustrative visualization methods were researched that mimic the effective information-communication techniques, elaborately developed by scientific illustrators. Less research was done dealing with the specification of expressive visualization mappings.

In this theses a visual abstraction technique was presented that is inspired by traditional caricature. Caricatures are pieces of art, with many properties advantageous also to visualization. Objects are depicted exaggerating (thus emphasizing) the salient details. The differences from the "normal" (i.e., the reference) object are in many cases the features of interest. While concentrating on the essence of objects, caricatures are also sparsely sketching the context. The goals of traditional caricatures and *caricaturistic visualization* are however different. The main goal of caricatures is a humorous depiction of the object of interest. In contrast caricaturistic visualization uses the principle of exaggeration for the purpose of an accentuated depiction of outstanding features.

The visualization mapping that specifies *what* is shown and *how* it is shown is essential to make effective use of visual abstractions. Previously this mapping was designed and specified during algorithm development. Only the parameters of the algorithm, that are often meaningless to the user, are exposed. In this thesis a visualization mapping method was presented that makes explicit use of data as well as visualization semantics. Visualization rules specify a mapping between the meaningful semantics of the data domain and the visualization domain. To evaluate the *semantic visualization mapping* and apply the visual abstractions accordingly,

fuzzy logic is used. A hybrid approach was investigated that models the fuzzy logic rule base in software and evaluates the visualization rules on the graphics hardware. The user interacts with the fuzzy logic rules that specify the visualization mapping. Prior to visualization the rules are translated into valid shader code. This code is interactively executed on the graphics hardware. The hybrid approach provides the flexibility of a software based fuzzy logic system and interactive performance due to the power of modern graphics hardware. *Interaction-dependent* rules are introduced to control the behavior of the interactive volume illustrations.

Bibliography

[1] E. Akleman, J. D. Palmer, and R. Logan. Making extreme caricatures with a new interactive 2D deformation technique with simplicial complexes. In *Proceedings of Visual 2000*, pages 165–170, 2000.

[2] P. J. Benson and D. I. Perret. Perception and recognition of photographic quality facial caricatures: Implications for the recognition of natural images. *European Journal of Cognitive Psychology*, 3(1):105–135, 1991.

[3] P. J. Benson and D. I. Perret. Visual processing of facial distinctiveness. *Journal of Perception*, 23(1):75–93, 1994.

[4] S. E. Brennan. Caricature generator: The dynamic exaggeration of faces by computer. *Leonardo*, 18(3):170–178, 1985.

[5] S. Bruckner, S. Grimm, A. Kanitsar, and M. E. Gröller. Illustrative context-preserving volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1559–1569, 2006.

[6] S. Bruckner and M. E. Gröller. VolumeShop: An interactive system for direct volume illustration. In *Proceedings of IEEE Visualization 2005*, pages 671–678, 2005.

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

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

[9] M. S. T. Carpendale, D. J. Cowperthwaite, and F. D. Fracchia. Distortion viewing techniques for 3-dimensional data. In *Proceedings of IEEE Symposium on Information Visualization 1996*, pages 46–53, 1996.

[10] C. Correa, D. Silver, and M. Chen. Feature aligned volume manipulation for illustration and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1069–1076, 2006.

[11] B. Coyne and R. Sproat. Wordseye: an automatic text-to-scene conversion system. In *Proceedings of ACM SIGGRAPH 2001*, pages 487–496, 2001.

[12] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. In *Proceedings of ACM SIGGRAPH 2003*, pages 848–855, 2003.

[13] H. Doleisch, M. Gasser, and H. Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Proceedings of VisSym, Joint Eurographics - IEEE TCVG Symposium on Visualization 2003*, pages 239–248, 2003.

[14] R. A. Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. In *Proceedings of ACM SIGGRAPH 1988*, pages 65–74, 1988.

[15] F. M. Jr. Dwyer. Adapting visual illustrations for effective learning. *Harvard Educational Review*, 37(2):250–263, 1967.

[16] W. Förstner. A framework for low level feature extraction. In *ECCV '94: Proceedings of the Third European Conference on Computer Vision (Vol. II)*, pages 383–394, 1994.

[17] A. Gooch, B. Gooch, P. Shirley, and E. Cohen. A non-photorealistic lighting model for automatic technical illustration. In *Proceedings of ACM SIGGRAPH 1998*, pages 447–452, 1998.

[18] B. Gooch, E. Reinhard, and A. Gooch. Human facial illustrations: Creation and psychophysical evaluation. *ACM Transactions on Graphics*, 23(1):27–44, 2004.

[19] M. Hadwiger, C. Berger, and H. Hauser. High-quality two-level volume rendering of segmented data sets on consumer graphics hardware. In *Proceedings of IEEE Visualization 2003*, pages 301–308, 2003.

[20] M. A. Hagen and D. Perkins. A refutation of the hypothesis of the superfidelity of caricatures relative to photographs. *Journal of Perception*, 12(1):55–61, 1983.

[21] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[22] H. Hauser. Generalizing focus+context visualization. In G.-P. Bonneau, T. Ertl, and G.M. Nielson, editors, *Scientific visualization: The visual extraction of knowledge from data*, pages 305–327. Springer, 2005.

[23] H. Hauser and M. Mlejnek. Interactive volume visualization of complex flow semantics. In *Proceedings of VMV 2003*, pages 191–198, 2003.

[24] H. Hauser, L. Mroz, G.-I. Bischi, and M. E. Gröller. Two-level volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):242–252, 2001.

[25] J. Hladůvka, A. König, and M. E. Gröller. Curvature-based transfer functions for direct volume rendering. In *Proceedings of the Spring Conference on Computer Graphics 2000*, pages 58–65, 2000.

[26] E. R. S. Hodges, editor. *The Guild Handbook of Scientific Illustration*, chapter 6, pages 134–137. Wiley, second edition, 2003.

[27] A. Joshi and P. Rheingans. Illustration-inspired techniques for visualizing time-varying data. In *Proceedings of IEEE Visualization 2005*, pages 679–686, 2005.

[28] T. Judd, F. Durand, and E. Adelson. Apparent ridges for line drawing. In *Proceedings of ACM SIGGRAPH 2007: 19*, 2007.

[29] A. Kanitsar, R. Wegenkittl, P. Felkel, D. Fleischmann, D. Sandner, and M. E. Gröller. Peripheral vessel investigation for routine clinical use. In *Proceedings of IEEE Visualization 2001*, pages 91–98, 2001.

[30] T. Keahey and E. Robertson. Techniques for non-linear magnification transformations. In *Proceedings of IEEE Symposium on Information Visualization*, pages 38–45, 1996.

[31] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Möller. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *Proceedings of IEEE Visualization 2003*, pages 513–520, 2003.

[32] J. Kniss, G. Kindlmann, and C. Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proceedings of IEEE Visualization 2001*, pages 255–262, 2001.

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

[34] J. Kniss, R. Van Uitert, A. Stephens, G.-S. Li, T. Tasdizen, and C. Hansen. Statistically quantitative volume visualization. In *Proceedings IEEE Visualization 2005*, pages 287–294, 2005.

[35] R. Kosara, S. Miksch, and H. Hauser. Semantic depth of field. In *Proceedings of IEEE Symposium on Information Visualization 2001*, pages 97–104, 2001.

[36] J. Krüger, J. Schneider, and R. Westermann. Clearview: An interactive context preserving hotspot visualization technique. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):941–948, 2006.

[37] A. Lake, C. Marshall, M. Harris, and M. Blackstein. Stylized rendering techniques for scalable real-time 3d animation. In *Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, pages 13–20, 2000.

[38] A. Lerios, C. D. Garfinkle, and M. Levoy. Feature-based volume metamorphosis. In *Proceedings of ACM SIGGRAPH 1995*, pages 449–456, 1995.

[39] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8:29–37, 1988.

[40] C. Liu, A. Torralba, W. T. Freeman, F. Durand, and E. H. Adelson. Motion magnification. *ACM Transactions on Graphics*, 24(3):519–526, 2005.

[41] A. Lu, C. J. Morris, J. Taylor, D. S. Ebert, C. Hansen, P. Rheingans, and M. Hartner. Illustrative interactive stipple rendering. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):127–138, 2003.

[42] E. B. Lum and K.-L. Ma. Lighting transfer functions using gradient aligned sampling. In *Proceedings of IEEE Visualization 2004*, pages 289–296, 2004.

[43] R. Maciejewski, T. Isenberg, W. M. Andrews, D. S. Ebert, M. Costa Sousa, and W. Chen. Measuring stipple aesthetics in hand-drawn and computer-generated images. *IEEE Computer Graphics and Applications*, 28(2):62–74, 2008.

[44] B. H. McCormick, T. A. DeFanti, and M. D. Brown. Visualization in scientific computing. *Computer Graphics*, 21(6), 1987.

[45] P. S. McCormick, J. Inman, J. P. Ahrens, C. Hansen, and G. Roth. Scout: a hardware-accelerated system for quantitatively driven visualization and analysis. In *Proceedings of IEEE Visualization 2004*, pages 171–178, 2004.

[46] J. M. Ottino. Is a picture worth 1,000 words? *Nature*, 421(6922):474–476, 2003.

[47] D. N. Perkins and M. A. Hagen. *The Perception of Pictures II*. Academic Press, 1981.

[48] J. A. Perkins. Netterimages: http://www.netterimages.com/artist/perkins.htm/. December 2008.

[49] P. Rautek, S. Bruckner, and M. E. Gröller. Semantic layers for illustrative volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, pages 1336–1343, 2007.

[50] P. Rautek, S. Bruckner, and M. E. Gröller. Interaction-dependent semantics for illustrative volume rendering. *Computer Graphics Forum*, 23(1):847–854, Proceedings EuroVis 2008, 2008.

[51] P. Rautek, S. Bruckner, I. Viola, and M. E. Gröller. Illustrative visualization - new technology or useless tautology? *ACM SIGGRAPH Computer Graphics Quarterly*, 42(3), 2008.

[52] P. Rautek and I. Viola. Visual abstractions and interaction metaphors for knowledge assisted volume visualization. *Workshop on Knowledge-assisted Visualization*, 2008.

[53] P. Rautek, I. Viola, and M. E. Gröller. Caricaturistic visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1085–1092, 2006.

[54] L. Redman. *How To Draw Caricatures*. In Aspects of Face Processing, Contemporary Books, 1984.

[55] C. Rezk-Salama, M. Keller, and P. Kohlmann. High-level user interfaces for transfer function design with semantics. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1021–1028, 2006.

[56] P. Rheingans and D. Ebert. Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):253–264, 2001.

[57] G. Rhodes, S. Brennan, and S. Carey. Identification and ratings of caricatures: Implications for mental representations of faces. *Cognitive Psychology*, 19(4):473–497, 1987.

[58] G. Rhodes and T. Tremewan. Understanding face recognition: caricature effects, inversion, and the homogeneity problem. *Visual Cognition*, 1:275–311, 1994.

[59] G. Rhodes and T. Tremewan. Averageness, exaggeration, and facial attractiveness. *Psychological Science*, 7(2):105–110, 1996.

[60] T. A. Ryan and C. Schwartz. Speed of perception as a function of mode of presentation. *American Journal of Psychology*, 69(1):60–69, 1956.

[61] Y. Sato, C.-F. Westin, A. Bhalerao, S. Nakajima, N. Shiraga, S. Tamura, and R. Kikinis. Tissue classification based on 3d local intensity structures for volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):160–180, 2000.

[62] D. Seligmann and S. Feiner. Automated generation of intent-based 3D illustrations. In *Proceedings of ACM SIGGRAPH 1991*, pages 123–132, 1991.

[63] D. D. Seligmann. *Interactive Intent-Based Illustration: A Visual Language for 3D Worlds*. PhD thesis, Columbia University, New York, USA, 1993.

[64] P.-P. Sloan, W. Martin, A. Gooch, and B. Gooch. The lit sphere: A model for capturing NPR shading from art. In *Proceedings of Graphics Interface 2001*, pages 143–150, 2001.

[65] S. V. Stevenage. Can caricatures really produce distinctiveness effects? *British Journal of Psychology*, 86:127–146, 1995.

[66] K. Stockinger, J. Shalf, W. Bethel, and K. Wu. Query-driven visualization of large data sets. In *Proceedings of IEEE Visualization 2005*, pages 167–174, 2005.

[67] N. Svakhine, D. S. Ebert, and D. Stredney. Illustration motifs for effective medical volume illustration. *IEEE Computer Graphics and Applications*, 25(3):31–39, 2005.

[68] T. Taerum, M. Costa Sousa, F. Samavati, S. Chan, and J. R. Mitchell. Realtime super resolution contextual close-up of clinical volumetric data. In *Proceedings of EuroVis, Joint Eurographics - IEEE VGTC Symposium on Visualization 2006*, pages 347–354, 2006.

[69] A. Tappenbeck, B. Preim, and V. Dicken. Distance-based transfer function design: Specification methods and applications. In *SimVis*, pages 259–274, 2006.

[70] L. H. Tsoukalas and R. E. Uhrig. *Fuzzy and Neural Approaches in Engineering*. Wiley & Sons, 1997.

[71] E. R. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narrative*, chapter 2, pages 27–54. Graphics Press, second edition, 1997.

[72] F.-Y. Tzeng, E. B. Lum, and K.-L. Ma. An intelligent system approach to higher-dimensional classification of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):273–284, 2005.

[73] I. Viola and M. E. Gröller. Smart visibility in visualization. In *Proceedings of EG Workshop on Computational Aesthetics, Computational Aesthetics in Graphics, Visualization and Imaging*, pages 209–216, 2005.

[74] I. Viola, A. Kanitsar, and M. E. Gröller. Importance-driven volume rendering. In *Proceedings of IEEE Visualization 2004*, pages 139–145, 2004.

[75] I. Viola, A. Kanitsar, and M. E. Gröller. Importance-driven feature enhancement in volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):408–418, 2005.

[76] L. Wang, Y. Zhao, K. Mueller, and A. Kaufman. The magic volume lens: An interactive focus+context technique for volume rendering. In *Proceedings of IEEE Visualization 2005*, pages 47–54, 2005.

[77] C. Weigle and R. M. Taylor. Visualizing intersecting surfaces with nested-surface techniques. In *Proceedings of IEEE Visualization 2005*, pages 503–510, 2005.

[78] M. Wohlfart and H. Hauser. Story telling for presentation in volume visualization. In *Proceedings of EuroVis, Joint Eurographics - IEEE VGTC Symposium on Visualization 2007*, pages 91–98, 2007.

[79] J. Woodring and H.-W. Shen. Multi-variate, time varying, and comparative visualization with contextual cues. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):909–916, 2006.

[80] M. Wynblatt and D. Benson. Web page caricatures: multimedia summaries for www documents. In *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pages 194–199, 1998.

[81] R. R. Yager and L. A. Zadeh, editors. *An Introduction to Fuzzy Logic Applications in Intelligent Systems*, volume 165 of *International Series in Engineering and C.S.* Springer, 1992.

[82] X. Yuan and B. Chen. Illustrating surfaces in volume. In *Proceedings of VisSym, Joint Eurographics - IEEE TCVG Symposium on Visualization 2004*, pages 9–16, 2004.

[83] J. Zhou, A. Döring, and K. D. Tönnies. Distance based enhancement for focal region based volume rendering. In *Proceedings of Bildverarbeitung für die Medizin 2004*, pages 199–203, 2004.

Curriculum Vitae

Personal

Name	Peter Rautek
Date of Birth	November 5th, 1979
Place of Birth	Vienna, Austria
Gender	Male
Languages	German (native), English (fluent)

Contact

Address	Zehenthofgasse 4/37, 1190 Vienna, Austria
Phone	+43 650 3335332
E-Mail	rautek@cg.tuwien.ac.at

Education

11/2005 -	Ph.D. student in Computer Science Vienna University of Technology, Austria
10/2003 - 05/2005	Study of Computer Science, MSc (*Computergraphik und Digitale Bildverarbeitung*) Vienna University of Technology, Austria
10/1999 - 09/2003	Study of Computer Science, BSc (*Medieninformatik*) Vienna University of Technology, Austria
09/1990 - 06/1998	Secondary School: BRG XIX Krottenbachstrasse, Vienna, Austria
09/1986 - 06/1990	Elementary School: Flotowgasse, Vienna, Austria

Work Experience

11/2005 -	Research Associate Vienna University of Technology, Austria
03/2004 - 01/2005	Computer Science Teacher and Custos *Fachschule für Sozialberufe*: Seegasse, Vienna, Austria
03/2003 - 01/2004	Teaching Assistant Vienna University of Technology, Austria
10/1998 - 09/1999	Social service at hospital *Göttlicher Heiland*

Scientific Activities

Reviewing

Journals	IEEE Transactions on Visualization and Computer Graphics, IEEE Computer Graphics and Applications, The Visual Computer, Computer Graphics Forum
Conferences	IEEE Visualization Conference, Eurographics, IEEE/EG International Symposium on Volume Graphics, International Conference on Computer Graphics Theory and Applications, International Workshop on Ontology Alignment and Visualization, International Conference on 3D Web Technology, Central European Seminar on Computer Graphics

Teaching

Organization	LU Visualisierung (*visualization lab*), SE Grundlagen methodischen Arbeitens (*seminar on basics of methodic working*)
Exams	Algorithmen und Datenstrukturen *algorithms and data structures*, VO Visualisierung (*visualization lecture*), LU Computergraphik 1 (*computer graphics 1 lab*)
Supervision	master theses, computer science projects, and seminars

Publications

Journal and Reviewed Conference Publications:

- Rautek, P., Csebfalvi, B., Grimm, S., Bruckner, St., Gröller, E.: D^2VR: High Quality Volume Rendering of Projection-based Volumetric Data. *Data Visualization 2006 (Proceedings of EuroVis 2006)*, pp. 211-218, 2006.

- Rautek, P., Viola, I., Gröller, E.: Caricaturistic Visualization. *IEEE Transactions on Visualization and Computer Graphics (Proc. Visualization 2006)*, 12(5):1085-1092, 2006.

- Rautek, P., Bruckner, St., Gröller, E.: Semantic Layers for Illustrative Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics (Proc. Visualization 2007)*, 13(6):1336-1343, 2007.

- Rautek, P., Bruckner, St., Gröller, E.: Interaction-Dependent Semantics for Illustrative Volume Rendering. *Computer Graphics Forum*, 27(3):847-854, Proceedings EuroVis 2008, 2008.

Other Publications:

- Rautek, P., Reiterer, A., Gröller, E.: Caricaturistic Visualization of Deformation Data based on High Density Point Clouds. Poster at the *8th Conference on Optical 3-D Measurement Techniques*, 2007.

- Rautek, P., Bruckner, St., Gröller, E., Viola, I.: Illustrative Visualization - New Technology or Useless Tautology? *ACM SIGGRAPH Computer Graphics Quarterly*, Volume 42, Number 3, 2008.

- Rautek, P., Viola, I.: Visual Abstractions and Interaction Metaphors for Knowledge Assisted Volume Visualization (Abstract). *Workshop Knowledge-assisted Visualization*, Oct 19, 2008, (KAV 08).