Two-level Volume Rendering

Helwig Hauser, Lukas Mroz, Gian-Italo Bischi, M. Eduard Gröller

Abstract—**In this paper we present a two-level approach for volume rendering, i.e., two-level volume rendering, which allows for selectively using different rendering techniques for different subsets of a 3D data-set. Different structures within the data-set are rendered locally on an object-by-object basis by either DVR, MIP, surface rendering, value integration (x-ray-like images), or non-photorealistic rendering. All the results of subsequent object renderings are combined globally in a merging step (usually compositing in our case). This allows to selectively choose the most suitable technique for depicting each object within the data, while keeping the amount of information contained in the image at a reasonable level. This is especially useful when inner structures should be visualized together with semi-transparent outer parts, similar to the focus-plus-context approach known from information visualization. We also present an implementation of our approach, which allows to explore volumetric data using two-level rendering at interactive frame rates.**

Keywords—**visualization, volume rendering, dynamical systems, medical applications**

I. Introduction

IRRESPECTIVE of a certain application, the general goal of visualization is to provide insight into the data of interest. This means that visualization facilitates the process of answering specific user-defined questions about the data under investigation. As a consequence, the question of *what* visualization method is well-suited or even optimal in the context of a specific application is not only dependent on the application data itself, but it also significantly depends on the actual questions of the user.

Especially in cases where the simultaneous visualization of all the data is not possible, e.g., in cases of very large data-sets or data of high dimensionality, the question of *what aspect or what subset* of the data to show becomes a very important decision during investigation. In medical applications, for example, when visualizing three-dimensional data-sets, it is in general not possible to concurrently show all the data. Instead, certain selective representations of the data are used for visualization: iso-surfaces which bound certain subsets of the data, voxels of maximal intensity which are displayed using maximum-intensity projection, as well as several others.

Of course, certain properties of the data under investigation themselves can also imply the use of a specific visualization technique. Given a specific subset of interest, for example, which actually has a sharp boundary that delimits it from the rest of the data, the use of an iso-surface might be a well-suited choice for visualization. In contrast, another subset of the data, which is characterized by relatively high data-values, might better be visualized by the use of maximum-intensity projection.

H. Hauser and L. Mroz are with the VRVis Research Center in Vienna, Lothringerstraße 16/4, A-1030 Wien, Austria, URL: `http://www.VRVis.at/vis/`, eMail: {`hauser`,`mroz`}`@vrvis.at`

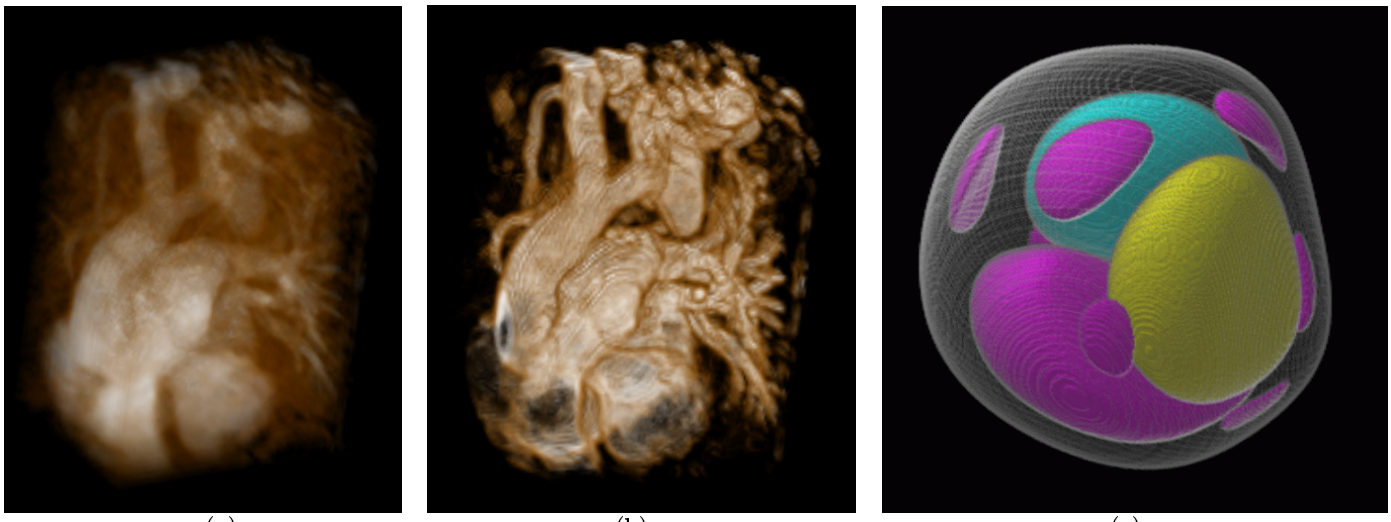
G.-I. Bischi is with the University of Urbino, I-61029 Urbino, Italy, URL: `http://www.uniurb.it/`, eMail: `bischi@econ.uniurb.it`

M. E. Gröller is with the Institute of Computer Graphics and Algorithms, Vienna University of Technology, Karlsplatz 13/186, A-1040 Wien, Austria, URL: `http://www.cg.tuwien.ac.at/home/`, eMail: `groeller@cg.tuwien.ac.at`

Fig. 1. Two-level volume rendering of a medical data-set (parts of a human hand): bones are rendered using DVR, surface rendering is used for vessels, and non-photorealistic rendering is used for the skin.

In this paper we now present our two-level approach for volume rendering, i.e., *two-level volume rendering* (2lVR), which allows to selectively use individual rendering techniques such as direct volume rendering (DVR), maximum-intensity projection (MIP), iso-surfaces, x-ray-like summation, and non-photorealistic rendering (NPR) for different objects within a common data-set. See Fig. 1 for an example, where several different rendering techniques have been used to visualize a medical data-set of a human hand.

An interesting observation about human perception of 3D computer graphics is that viewers seem to inherently separate individual objects from rendered images of 3D scenes. Users tend to see solid objects, which are bounded by opaque or semi-transparent surfaces. Even objects with intrinsic 3D characteristics such as clouds, fire, aso., are



(a) (b) (c)

Fig. 2. Direct volume rendering vs. surface rendering: (a) DVR using a rather smooth TF results in gel-looking objects; (b) DVR using a rather sharp TF simulates surface rendering; (c) surface rendering very well communicates 3D shape.

perceived as individual entities. Additionally, many visualization goals also require such a notion of data-sets as being composed of individual objects. Medical investigation, for example, often is object-oriented: some specific organ of interest is to be examined whereas all the rest is considered to be context. In the following, we will therefore understand any volumetric data-set as being composed of semantically distinguished subsets of the data, which in turn we will call the *objects* within the data-set.

In this paper, we also demonstrate how two-level volume rendering is used as a focus-plus-context (F+C) technique in volume rendering, comparable to similar solutions which are well-known from information visualization. Especially interesting subsets of the data, which are considered to be "in focus", are rendered, for example, using iso-surfacing or DVR, whereas the rest of the data, i.e., the context in this case, is depicted using semi-transparent MIP or non-photorealistic rendering.

II. Experiences with DVR, MIP, etc.

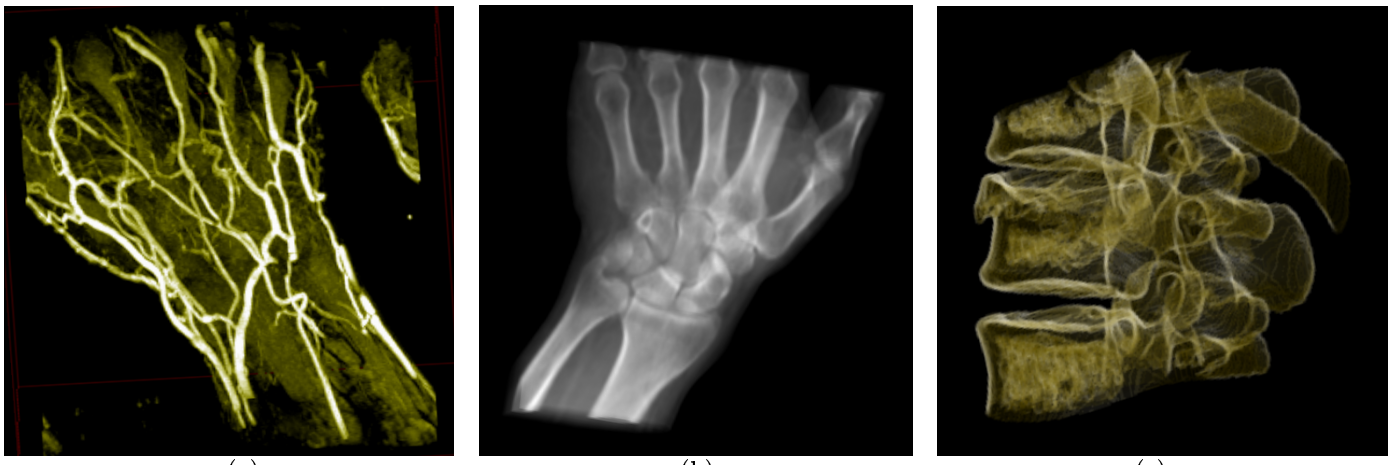
Before we actually describe the new approach of two-level volume rendering, we first give a brief review of some experiences which we gained from previous work with different volume rendering techniques.

One of the standard techniques for displaying volumetric data is direct volume rendering [1], [2], which is based on the sorted composition of visual properties along viewing rays. The most important parameter of DVR probably is the so-called transfer function (TF) which describes the mapping of data-values to optical properties. Depending on the transfer function in use, mainly two typical forms of object appearances can be distinguished: either objects look like semi-transparent gel due to a rather smooth TF, or object boundaries are depicted similar to iso-surfacing as a result of a rather sharp or even binary TF – see Figs. 2(a) and (b) for sample results. In general, we experienced that results from DVR usually feature good impression of 3D shape, especially if photorealistic shading is used. Contrarily, occlusion sometimes becomes a significant problem of DVR, which is mainly because pixel-values do not only depend on the local TF mapping, but also on the number of samples to be composited. The latter is not under control of the TF and can drastically vary among all viewing rays, consequently resulting in sometimes large variations with respect to object transparency.

Due to the fact, that it is often very difficult to intuitively set up a proper transfer function, given a specific visualization goal in mind, the design of transfer functions has become a research topic on its own [3], [4], [5], [6], [7]. Producing useful images with DVR is even more difficult if there is no close correspondence between differences in data-values and the discrimination of objects. MRI data-sets, for example, typically contain large differences in data-values for objects of similar type.

Two solutions have been proposed to help in such a case: one is to use an alternative rendering technique for depicting the features of interest – for solutions different to DVR see the paragraphs below. Another approach is to first apply segmentation [8] to the data-set, which is followed by selective rendering of the separated objects, for example, by the use of different transfer functions [9]. Various techniques have been proposed for automatic and semi-automatic segmentation of volumetric data, such as thresholding, water-shed, etc. In this paper we do not focus on segmentation itself, but very well make use of optional segmentation information, if present as a result of a preprocessing step. As two-level volume rendering inherently depends on the notion of a 3D data-set which is composed of distinguishable objects, our implementation of 2lVR also provides a threshold-based segmentation technique for all cases, where no segmentation information is given a priori.

Maximum-intensity projection [10], [11], [12], which is used to display the most important data-values along viewing rays, features a clearly different object appearance in



(a) (b) (c)

Fig. 3. Three alternative methods for volume rendering: (a) MIP displays structures of maximal importance; (b) value integration results in x-ray-like images; (c) non-photorealistic rendering may enhance contours, for example.

comparison to DVR. MIP images are usually quite sharp as only one data-value is shown per pixel. Also, assuming that importance is directly correlated with data-values, the more important some data-values are, the more likely it is that they are actually visible in the resulting image – the amount of important information which is hidden is minimal. However, due to the fact that neighboring pixels do not necessarily represent spatially coherent data-values, MIP images tend to look rather flat. This undesirable effect of MIP actually is an inherent property of this rendering method, and can only be diminished to a certain extent by variations of MIP such as local MIP [13], depth-shaded and/or animated MIP, etc. For an example see Fig. 3(a).

Surface-based approaches – in contrast to DVR and MIP – are suitable in situations, where sharp object boundaries are present in the data-set and the interior structure of the object is not relevant for visualization. Basically, there are two approaches to extract and render iso-surfaces. One class of techniques deals with the explicit computation of a geometric representation of the iso-surface – the marching cubes algorithm [14] is a well-known example. The second class deals with depicting the surface by means of a transfer function [2]. DVR and surface rendering can also be combined into a hybrid technique depicting both, truly volumetric information, and objects defined by polygonal models [15]. In this paper we integrate the rendering of object surfaces by directly depicting surfaces from the volumetric data [16] – see Fig. 2(c) for an example –, not dealing with any polygonal model at all. This is also in contrast to volumetric ray tracing [17], where actual ray tracing is used to compute high-quality images from 3D scenes which are composed of volumetric objects and polygonal models.

We have also implemented value integration along viewing rays through volumetric data for image synthesis. Images, which have been rendered using this technique – see Fig. 3(b) for an example –, simulate an x-ray-appearance of the data. This is due to the fact, that value integration is nothing else than the inverse operation compared to data reconstruction within a 3D scanning device such as a CT.

Recently, non-photorealistic rendering (NPR) methods have been proposed for volumetric data [18], [19], which, for example, depict object contours in a view-dependent manner. Here, a great variety of object appearances can be achieved, all useful for different visualization goals. In our case, we found the depiction of object contours by the use of NPR to be especially useful, also as an alternative to traditional rendering methods as described above. Fig. 3(c) shows an example of NPR rendering of volumetric data.

Good visualization strongly depends on *what data* is visualized, *what structure* this data consists of, as well as on the *visualization goals* of the user. Depending on these prerequisites several useful approaches exist, and individual decisions (what rendering method to choose) have to be made for specific applications. Two-level volume rendering as described in this paper allows for selectively combining different rendering techniques for different objects within a common data-set.

III. Two-level volume rendering

After the preceeding discussion of pros and cons of several volume rendering techniques, we now present two-level volume rendering as a useful way of combining different rendering methods with respect to their respective advantages.

A. Object discrimination

An important goal for two-level volume rendering not only was to provide the ability for using different rendering techniques for different objects within a 3D scene, but also to come up with visualization results which actually allow to easily discriminate the individual objects from each other. From previous work [20] we know, that displaying multiple semi-transparent surfaces on top of each other, quickly imposes significant problems with the identification and perception of the individual objects. Usually just two or three layers are easily distinguished from rendered images. Therefore, a key feature of two-level volume rendering is that pixel-values are composed of a very limited number of values only.

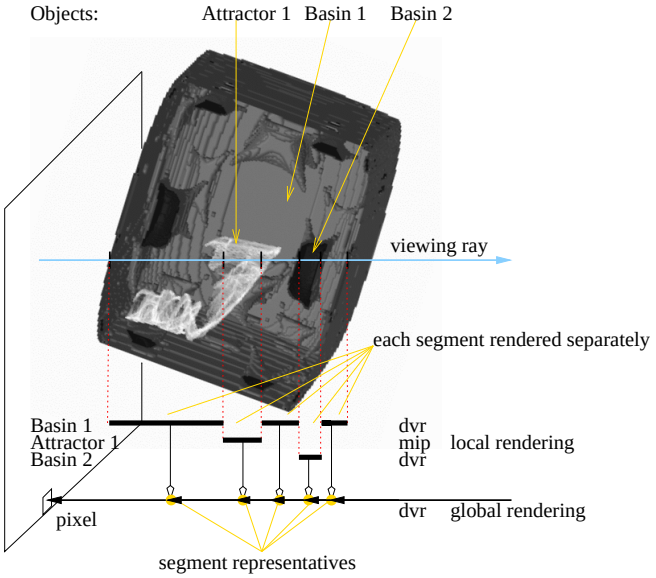


Fig. 4. Object segmentation implicitly yields viewing rays to be partitioned into segments (one per object intersection).

B. Two-level rendering

For achieving our goal of generating images which are composed of a few, but meaningful object representatives per pixel, we decompose the rendering calculations into two levels, i.e., local and global rendering. For every pixel, we theoretically investigate a viewing ray through the data and detect what objects are intersected. For every object intersected, a single, representative value is computed (by the use of local rendering). These object representatives are finally composed to yield a pixel value by combining them, usually by the use of DVR compositing (global rendering). Only a small number of data-values are combined for a pixel (one per object), therefore limiting the number of samples which finally make up one pixel-value.

For more specifically explaining two-level volume rendering, it is useful to utilize the model of ray casting. Of course, an implementation of our technique is not bound to this image-order approach. In this paper, for example, we will later present a fast object-order implementation of two-level volume rendering.

For calculating the value of a pixel within the resulting image, we may assume a viewing ray to be cast right into the volumetric data-set (cf. Fig. 4). A 3D segmentation mask determines for any voxel within the data which object it belongs to. Therefore, also the viewing ray may be interpreted as being decomposed into several segments, depending on what objects are intersected by the ray. We now utilize this ray segmentation as follows:

While traversing a viewing ray – for example, back-to-front – two tracks of rendering are processed in parallel. As long as one segment of the ray is traversed, i.e., as long as the viewing ray traverses one individual object within the data, local rendering is performed to compute an object representative associated to the segment (rendering at the object level). Individual rendering methods can be used for different segments, depending on what objects are traversed. Fig. 5, for example, was rendered with DVR using a sharp TF along segments through bones, surface rendering for vessels, and MIP for the skin.

Fig. 5. Combining DVR, surface rendering, and MIP: whereas the bones have been rendered using DVR (sharp TF), and the vessels have been rendered using surface rendering, the skin was depicted using MIP.

At those points along the ray, where the object classification changes, i.e., at the points where viewing rays cross object boundaries, update steps in the global rendering track are performed. We usually use DVR-compositing on the global level. The only exception, where we could experience that MIP is useful instead, is if all objects in the data-set are rendered by the use of MIP themselves, also. In contrast to standard MIP, this “MIP of MIP” approach allows to easily distinguish between different objects within the scene, as different transfer functions and thus different colors can be assigned to different objects.

C. Focus-plus-context

Due to the ability to selectively use different rendering techniques for different objects within a 3D data-set, two-level volume rendering strongly supports applications which are of F+C style: depending on whether objects are selected to be “in focus” or not, their visual appearance can be different. Whereas objects “in focus” may feature signifi-

	focus	context
DVR	± rather opaque, surface-like	± rather transparent, gel-like
MIP	+ complex focus, high contrast	+ very uniform transparency
surfaces	+ rather opaque	+ semi-transparent
x-ray	−	± inner context
NPR	± only as add-on	+ sparse contours

TABLE I
F+C CONFIGURATIONS FOR TWO-LEVEL VOLUME RENDERING

cant opacity, for example, objects which are considered to be context rather act as semi-transparent reference. See Fig. 5 again for an example, where bones and vessels are considered to be "in focus", whereas the skin just acts as the context in this application. Table I gives an overview about recommended F+C configurations when two-level volume rendering is used. Depending on the internal structure of objects, different rendering techniques seem to be well-suited. Objects "in focus", for example, can very well be rendered using rather opaque surfaces in case of spatially contiguous objects (Fig. 5, for example). Objects of interest, which are characterized by a complex or even fractal inner structure, can very well be depicted by the use of MIP (Fig. 9(c), for example). In case of context visualization, the use of MIP, quite transparent surfaces, and NPR proved to be very useful. MIP is useful, because it features quite homogeneous transparency throughout an entire object (Fig. 5, for example). Shape properties of context objects are well-communicated by the use of semi-transparent surfaces. NPR can be used to sparsely apply shape cues of outer context objects (Fig. 1, for example).

D. Technical details

Since the model of two-level volume rendering is based on ray casting, an implementation as image-order technique would be straight-forward. Nevertheless, the idea of two-level rendering has originally been developed to aid exploration of data from the field of complex dynamical systems. Interactivity is crucial for exploring and investigating data, and especially for finding proper settings for optical properties of objects. Thus we present a fast object-order implementation [21], based on shear-warp rendering [22], which allows for interactive visualization of medium-sized datasets. For performance reasons, no interpolation is done within the volume, each voxel is projected onto exactly one pixel of the base plane. During the warp step, bilinear interpolation is used, also considering the scaling component of the projection matrix.

To allow for efficient skipping of empty space in-between objects, each object within the investigated data is stored separately as an array of all its member voxels. For each voxel, its position and attributes, e.g., data-value and gradient information, are stored. The list is replicated and stored separately for each principal viewing axis (x, y, and z) with voxels sorted by the principal viewing coordinate. For the sake of clarity, the principal viewing direction will be called z from now on.

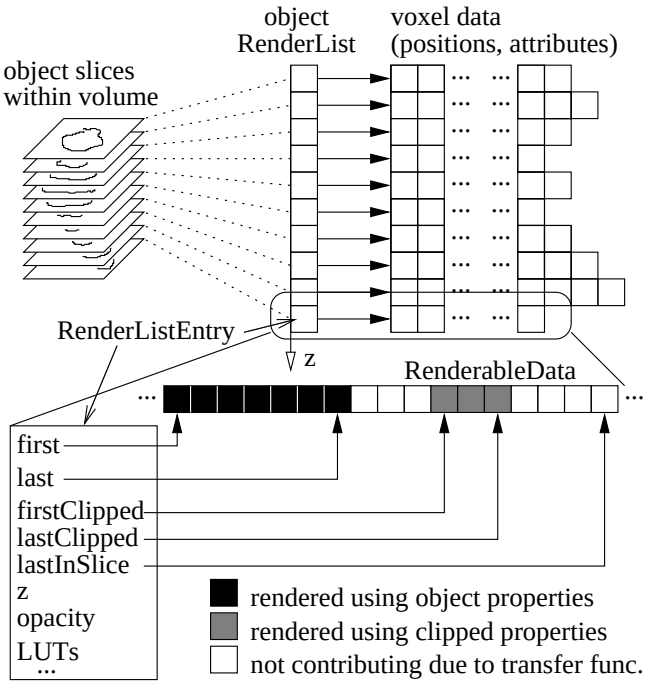


Fig. 6. Storage scheme for two-level volume rendering.

All voxels of an object which share the same z coordinate are grouped within a so-called `RenderListEntry`, the value of z is stored with the `RenderListEntry` instead with all the voxels (Fig. 6). For rendering, the `RenderListEntrys` of all objects are merged into a single `RenderList` sorted by z.

```
PROCEDURE ComputeRenderList:

FOR pvd (principal viewing dir.) BEING x, y, z:
{ FOREACH obj (object in the scene) DO:
    FOREACH val (depth value regarding pvd) DO:
      LET RenderListEntry[pvd,obj,val] = set of
          all voxels of obj. obj with depth val
  MERGE ALL RenderListEntry[pvd,.,.] into a
    RenderList[pvd] (sorted by val)
}
```

Processing this list sequentially during rendering results, for example, in a back-to-front traversal of the scene, even though the voxels within one `RenderListEntry`, i.e., which are at equal depth from the base plane, are projected in an arbitrary order. By storing only voxels which actually belong to objects, empty space in-between is skipped automatically without any effort during rendering.

For implementing two-level volume rendering, two sets of buffers are used for the base plane. One buffer (local object buffer) is used for performing rendering within an object, while a global buffer is used to perform inter-object rendering. In addition to pixel values, each pixel of the object buffer additionally stores the unique ID of the currently front-most object. If a voxel is projected onto the base plane, its ID is compared with the stored ID in the

object buffer. If both IDs match, the value in the object buffer is updated using an operation which corresponds to the local rendering mode of the object. If the ID of the voxel differs from the ID of the pixel in the buffer, the viewing ray though this pixel must have entered a new object. The content of the object buffer pixel is combined with the corresponding global buffer pixel using an operation which depends on the global rendering strategy (usually DVR). Then the object buffer pixel is initialized according to the voxel of the new object and the new local rendering mode.

After all voxels have been projected, the contribution of the front-most segment at each pixel has to be included by performing an additional scan of the buffers and merging the segment values left in the local buffer into the global buffer.

As the application of clipping planes is a widely used method for enhancing volume visualization, support for this technique has also been included in our object storage scheme (see Figs. 7 and 9(c)). The voxel lists used for rendering allow a very efficient implementation of arbitrary cutting planes. As the order in which voxels of the same depth are rendered to the base plane is irrelevant, voxels removed by the application of a cutting plane can be simply moved to the end of the voxel array belonging to a `RenderListEntry`. An end-mark is used to indicate the last visible voxel within a `RenderListEntry` (Fig. 6). Removed voxels which are stored after this mark can be ignored during rendering without slowing down the process. Instead, we also enable the clipped parts of objects to be rendered using another rendering techniques (cf. 10(c)).

```
PROCEDURE TwoLevelRendering:

Choose pvd (according to current viewing dir.)
Init(localBuf,0); Init(globalBuf,0)
FOREACH RenderListEntry IN RenderList[pvd]:
{ object = RenderListEntry.objectID
  FOREACH voxel IN RenderListEntry.activeSet:
  { pixel = projection(voxel,pvd)
    IF localBuf(pixel).object = object THEN:
      Update(localBuf(pixel),voxel)
    ELSE:
    { Update(globalBuf(pixel),localBuf(pixel))
      Init(localBuf(pixel),voxel)
} } }
FOREACH pixel DO:
  Update(globalBuf(pixel),localBuf(pixel))
image = Warp(globalBuf)
```

For lighting of objects the Phong shading model is employed. The gradient vector at each voxel is pre-calculated during the preprocessing step [23], converted to polar coordinates and quantized to 12 Bit for limiting the storage requirements. During rendering, this representation of the gradient is used to access a look-up table with shading information. The shading table includes the influence of ambient, diffuse, and specular components of the lighting model. It is quickly recalculated for each new viewing or light position and also after a change of material properties.

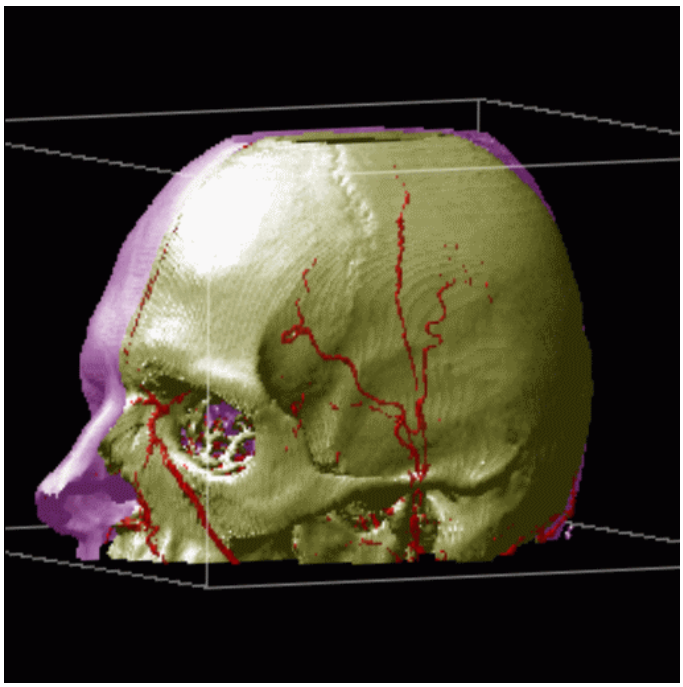


Fig. 7. Selectively applied clipping planes allow further insight.

Opacity values for compositing voxels and objects are derived from two sources: first, the ability to change the opacity of a whole object turned out to be useful when adjusting viewing parameters. Secondly, the object-wide opacity can be modulated by a per-voxel opacity which depends on some property of the voxel, like for example data-value, gradient magnitude, or the distance to some other object (see Sect. IV-B).

IV. Applications and results

Two-level volume rendering, as described above, can be used in various fields of applications. In this paper we describe two applications and show how two-level volume rendering improves the results.

A. Medical data visualization

Visualization of medical data is one of the most important application fields of volume rendering. Several rendering techniques are used to depict different features of the data. Pure surface rendering, for example, is used when actual boundaries of objects within the medical data-set are to be shown, e.g., the surface of bones or the skin (cf. Fig. 7). Similar to surface rendering, DVR is often used to render tissue transitions within medical data-sets (see Fig. 2(b)). Through photorealistic shading, DVR-images as well as surface depictions very well communicate 3D shape. Furthermore, DVR results feature a usually good impression of object inter-relations and depth information. MIP, on the other hand, is very useful for visualizing rather complex structures, like blood vessels, organs, etc. Images usually are less over-loaded, compared to DVR, and focus on the important parts of the data. In Fig. 3(a) we show blood vessels together with lower-valued bones by the use

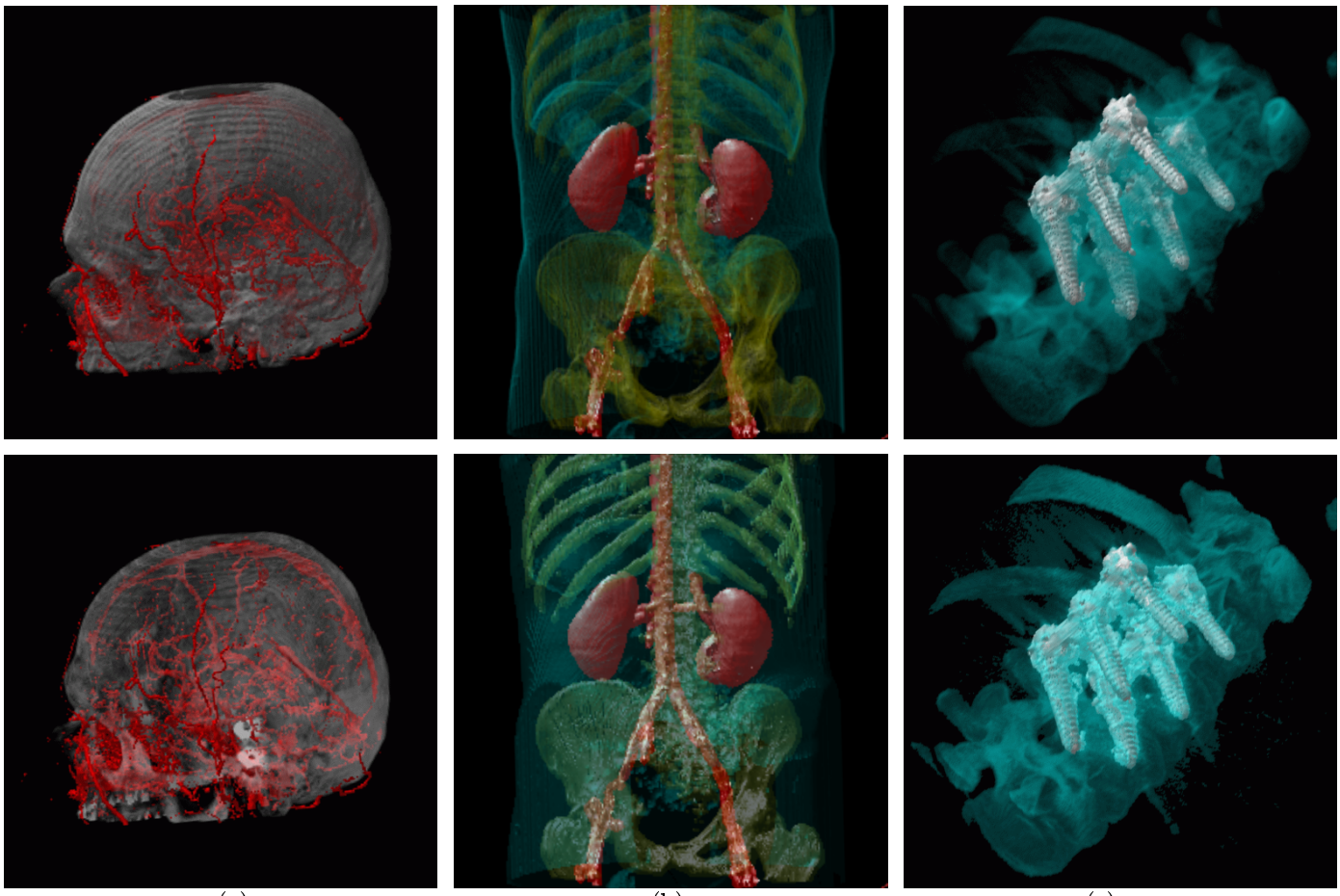
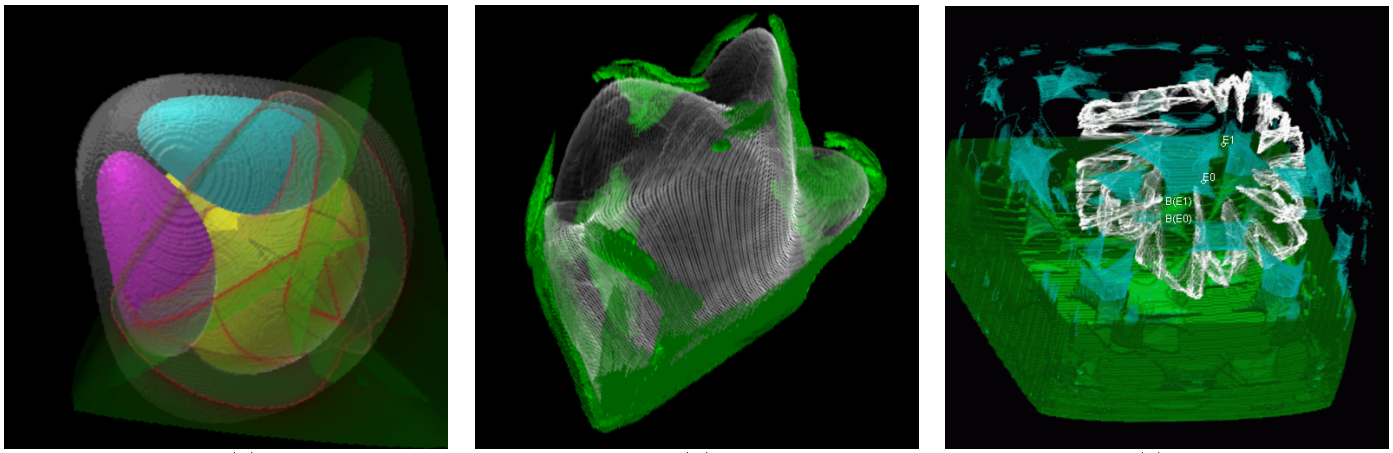


Fig. 8. Comparisons between standard DVR (first row) and two-level volume rendering (second row). (a) Two-level volume rendering allows to show blood vessels within the head, even near the skull. (b) Using MIP for the visualization of context (bones, skin) the object of interest (blood vessels plus stenosis) becomes well visible from all viewing directions. (c) Again MIP proves to work fine for context visualization.

of MIP. In addition to DVR and MIP, simple value integration (x-ray-like results) becomes useful when object thickness should be displayed. Also, x-ray-like object appearance might be familiar to user (see Fig. 10(a) for an example). Finally, non-photorealistic rendering of medical data-sets has been proven to allow for showing shape cues without the consumption of lots of screen space (cf. Figs. 1, 10(b), and 10(c)).

However, also some disadvantages can be experienced when working with the above mentioned methods: DVR images sometimes tend to be over-loaded with content (blurred images). Due to the convolution character of the compositing step, and the high sensitivity of DVR on the transfer function in use, it happens easily that too many data-values are merged into the final image. The use of gradient information as another input to the transfer function [2] partially improves this situation. For objects with a complex interior structure, nevertheless, it is difficult to avoid this accumulation problem. MIP images, on the other hand, usually lack depth information. This is due to the weak inter-pixel correlation of neighboring viewing rays. Data-maxima often are detected at significantly different depth locations along neighboring rays. Consequently, MIP images usually look rather flat, and view-point variations (via animation or user-interaction) are necessary to re-generate the 3D impression. Also, no occlusion is considered, showing objects, which are actually beyond each other, in an arbitrary order. With standard MIP usually just one structure is visualized, i.e., the object of interest lacks context information.

Two-level volume rendering allows to overcome these difficulties to a certain extent. Context information, for example, can be displayed by the use of MIP, and combined with DVR-rendered inner structures. The MIP-representation of the outer hull avoids the over-loading problem of DVR as MIP hulls feature rather uniform transparency. Through DVR-rendering the 3D shape of inner structures becomes more visible. In Fig. 8(a), lower image, the skull is rendered by the use of MIP, producing rather continuous transparency throughout the image. This enables insight to the blood vessels, which are represented by the use of DVR. Two further comparisons are also depicted in Fig. 8. In general, we found it useful to depict the context of DVR-rendered inner structures by the use of MIP [24]. We also experienced that in the case of rather complex structures, MIP is working fine, whereas for objects with limited spatial frequencies usually DVR works fine.



(a) (b) (c)

Fig. 9. Dynamical systems visualized by the use of two-level volume rendering.

B. Dynamical system visualization

We also successfully applied two-level volume rendering to discrete dynamical systems in 3D. We are interested in the long-term evolution of a discrete dynamical system, which is given as a set of three difference equations $\mathbf{x}_{t+1} = \mathbf{f}_{\mathbf{p}}(\mathbf{x}_t)$; $\mathbf{x}_t, \mathbf{x}_{t+1} \in \Omega \subseteq \mathbf{R}^3$, where t denotes the (discrete) time of evolution and $\mathbf{p} \in \Pi \subseteq \mathbf{R}^m$ is a vector of parameters of the dynamical system.

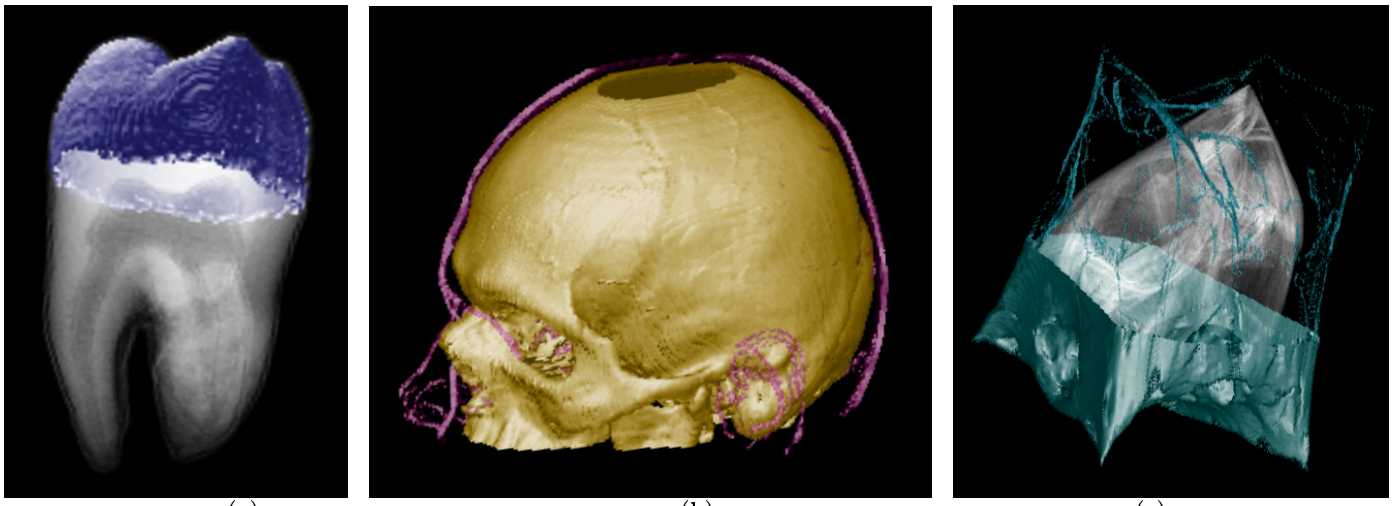
A trajectory (or evolution over time) of the system is defined as the sequence of states $\{\mathbf{x}_t\}_{t\geq 0}$, starting from a given initial condition $\mathbf{x}_0$, by the repeated application (iteration) of the map $\mathbf{f}_{\mathbf{p}}$, i.e., $\mathbf{x}_t = \mathbf{f}^t_{\mathbf{p}}(\mathbf{x}_0)$. Investigating the long-term behavior of such a dynamical system, we are first interested in attracting sets which are present for a specific parameter setting $\mathbf{p}$. Attracting sets might be quite simple, e.g., single points, periodic cycles of limited period, or quasi-periodic trajectories which fill a closed curve, or more complex objects such as chaotic sets, also called strange attractors [25].

Not only the attractor as the limit of evolution, but also its basin of attraction should be shown. A basin is equal to the set of all starting configurations $\mathbf{x}$, which (in the long-term) finally converge to the corresponding attractor. Similar to the attractors themselves, also the basins may have either simple boundaries, just a box, for example, or complex boundaries which may be formed by many disconnected portions and sometimes may even have a fractal structure. A basin is a set which spatially includes the corresponding attractor. When several attractors coexist, it becomes very important to visualize the boundaries which separate the corresponding basins together with the attractors which are nested inside them.

In our case, two-level volume rendering is successfully used in the scientific investigation of two discrete systems of international interest. The first one is a three-dimensional system which models a triopoly, i.e., an economic system where three producers share the market of a given product – for example, a competition among three companies in an international context. The question is which company will finally dominate the market, given a particular initial condition (the initial productions of the three companies) and given a certain set of so called reaction-functions $\mathbf{f}_{\mathbf{p}}$ which represent the optimal choices [26]. In Fig. 9(a) two-level volume rendering was used to visualize four basins of attraction, three enclosed basins using DVR, and one surrounding basin using MIP. Additionally a so-called critical surface is depicted using MIP. The color of the surface corresponds to the distance to the closest basin boundary. Critical surfaces, which are sometimes responsible for global bifurcations, are basically defined to be the 3D locus where the determinant of the Jacobian matrix of $\mathbf{f}_{\mathbf{p}}$ vanishes.

Another system we are investigating is a three-dimensional non-invertible quadratic, i.e., a second degree map, which can be considered as an extension of a broadly used two-dimensional quadratic map [27]. Such two-dimensional maps have often been used to explain global bifurcations. These are qualitative changes occurring in the structure of the attractors and/or of their basins when the parameters are changed, as a consequence of contacts between particular curves (the so-called contact bifurcations [27]). Two-level volume rendering for the visualization of basins and their associated attractors eases the investigation of bifurcation events in three-dimensional space. A fast visual inspection allows to relate bifurcations to contacts between particular surfaces whose analytic equation is generally unknown. Interactive 3D visualization is crucial for the detection of global bifurcation events in 3D. Fig. 9(b) shows, for example, an attractor which is close to a contact with the boundary of its basin. The attractor has been rendered using MIP, the boundary is depicted using DVR. The opacity of the boundary voxels is modulated according to the distance to the closest point of the attractor, clearly showing areas of potential contact. In fact, the use of cutting planes or simple projections does not work in such cases.

In Fig. 9(c) MIP was used to render a chaotic attractor. The inner structure of the fractal object becomes visible quite well. Another attractor, also apparent within this parameter setting, is a two-cycle (two points), represented



(a) (b) (c)

Fig. 10. Further results computed with two-level volume rendering: (a) value integration (x-ray-like depiction) and surface rendering, (b) surface rendering and NPR (c) surface rendering, NPR, and MIP.

by label "E1". Both basins of attraction are rendered by the use of DVR to better show their 3D shape. They subdivide phase space in a complex way: the basin of attraction corresponding to the simple attractor is a non-connected set, whose distribution in space has a fractal structure.

C. User interaction

Two-level volume rendering, as presented in this paper, has been implemented in an interactive Java-based [28] tool. The applet provides the user with a set of interactive tools for exploring and visualizing the data. After loading a data-set, it is decomposed into distinct objects. Segmentation information is always present for data originating from dynamical system computations. Medical data-sets however may not necessarily contain segmentation information. Non-segmented medical data is automatically decomposed using a simple, threshold-based segmentation.

All operations which affect viewing parameters and optical properties of objects are performed interactively. Each object can be either shown or hidden for selective visualization of specific features within the scene. In addition to the global rendering mode, the object-level rendering mode as well as the opacity can be selected separately for each object. Also, the color of objects can be set independently of the range of values of the object's voxels. In our implementation we use color mainly to distinguish between objects within the scene, whereas DVR and MIP mainly work on opacities, color saturation and lightness, as well as shading information. Additional information (like gradient magnitude or distance to another object) can be used to modulate either the color or the opacity of an object's voxels. Cutting planes at certain values of x, y, and z can be applied to subsets of objects and manipulated in real-time.

Using the mouse to directly interact with the scene, further visualization parameters can be influenced: viewing position, zoom factor, position of the light source, material properties, and opacity transfer function. Additionally the windowing metaphor, which is widely employed in medical imaging applications, is used to adjust contrast and emphasize specific data ranges within the color transfer function.

As empty regions within the volume are not stored in our implementation, we are able to handle 4D data (or even data of higher dimensionality) in a memory efficient way. 4D data is obtained from dynamical system research as parameter variation produces several (10–100) distinct volume data-sets.

figure	*volume size*	*rendering time*
3(a)	$256 * 256 * 100$	0.09 s
8(b)	$202 * 152 * 255$	0.18 s
8(c)	$256 * 256 * 241$	0.17 s
9(c)	$256 * 256 * 256$	0.17 s

TABLE II
RENDERING TIMES.

D. Results

Table II shows the rendering times of some of the figures from this paper measured on an AMD Athlon 600 PC. All images are 512x512 in size. Most of the images are single frames from animation sequences which can be found at the web page of this work [29], where also further results are shown. Fig. 10 shows three of them: surface rendering and value integration (x-ray-like depiction) were used for the tooth data-set, surface rendering together with NPR resulted in the image of the human head, whereas surface rendering, MIP, and NPR were used to depict the data from a dynamical system simulation.

V. Summary and conclusions

In this paper we presented the idea of merging several different volume rendering techniques into a two-level volume rendering method. Segmentation information is utilized to apply individual rendering techniques to different ob-

jects, which additionally are merged into a resulting image through a global rendering step.

We applied our new technique in two areas, namely medical data visualization and the visualization of dynamical systems. DVR-rendered objects of interest, for example, which are displayed with MIP-rendered context objects proved to be useful in both cases.

The decision what rendering method to choose strongly depends on what data is to be visualized, what structure this data consists of (complex, rather simple, etc.), and what the user actually wants to see within the data. Often, even one data-set contains objects, which differ significantly with regard to their inner structure or boundary surface. Our approach allows to take the most appropriate volume rendering method on a object-by-object basis, and globally merge the subsequent result to the final image.

We also experienced that certain visualization setups are quite common to various application fields. Focus-plus-context, for example, which is well-known from information visualization, also shows up in visualization goals for medical data, or data of dynamical systems. Users want to peer inside 3D data to investigate some inner structures, but they also would like to keep surrounding objects integrated within the visualization, for spatial reference.

Also, being able to implement this new approach of two-level volume rendering as an interactive tool on PC-hardware, allowed to experience the importance of interactive investigation of volumetric data. Being able to vary viewing parameters as well as visualization attributes interactively significantly helps to generate quite useful results.

VI. Acknowledgments

Acknowledgments go to VisMed, an FFF-funded project (`http://www.vismed.at/`), for the collaboration in the field of medical visualization. We want to thank the radiologists at the Innsbruck University Clinic for providing the medical data-sets shown in this paper. Parts of the work presented in this paper have been carried out as part of the BandViz project (`http://bandviz.cg.tuwien.ac.at/`), which is supported by FWF under project number P 12811. Further parts of this work have been carried out as part of the basic research on visualization at the VRVis Research Center in Vienna, Austria (`http://www.VRVis.at/vis/`), which is funded by an Austrian governmental research program called Kplus.

References

[1] J. T. Kajiya, "Ray tracing volume densities," in *Proc. of ACM SIGGRAPH '84*, pp. 165–174.

[2] M. Levoy, "Display of surfaces from volume data," *IEEE Computer Graphics & Applications*, vol. 8, no. 5, pp. 29–37, 1988.

[3] T. He, L. Hong, A. Kaufman, and H. Pfister, "Generation of transfer functions with stochastic search techniques," in *Proc. of IEEE Visualization '96*, pp. 227–234.

[4] C. L. Bajaj, V. Pascucci, and D. R. Schikore, "The contour spectrum," in *Proc. of IEEE Visualization '97*, pp. 167–174.

[5] J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber, "Design galleries: A general approach to setting parameters for computer graphics and animation," in *Proc. of ACM SIGGRAPH '97*, pp. 389–400.

[6] G. Kindlmann and J. Durkin, "Semi-automatic generation of transfer functions for direct volume rendering," in *Proc. of IEEE Volume Visualization '98*, pp. 79–86.

[7] A. H. König and M. E. Gröller, "Mastering transfer function specification by using volumepro technology," in *Proc. of Spring Conference on Computer Graphics and its Applications 2001*, Budmerice, Slovakia.

[8] K. K. Udupa and G. T. Herman, *3D Imaging in Medicine*, CRC Press, 1999.

[9] U. Tiede, T. Schiemann, and K. H. Höhne, "High quality rendering of attributed volume data," in *Proc. of IEEE Visualization '98*, pp. 255–262.

[10] K. J. Zuiderveld, A. H. J. Koning, and M. A. Viergever, "Techniques for speeding up high-quality perspective maximum intensity projection," *Pattern Recognition Letters*, vol. 15, pp. 507–517, 1994.

[11] G. Sakas, M. Grimm, and A. Savopoulos, "Optimized maximum intensity projection," in *Proc. of 5th EUROGRAPHICS Workshop on Rendering Techniques*, Dublin, Ireland, 1995, pp. 55–63.

[12] L. Mroz, A. König, and M. E. Gröller, "Maximum intensity projection at warp speed," *Computers and Graphics*, vol. 24, no. 3, pp. 343–352, 2000.

[13] Y. Sato, N. Shiraga, S. Nakajima, S. Tamura, and R. Kikinis, "LMIP: Local maximum intensity projection - a new rendering method for vascular visualization," *Journal of Computer Assisted Tomography*, vol. 22, no. 6, 1998.

[14] W. E. Lorensen and H. E. Cline, "Marching cubes: a high resolution 3D surface construction algorithm," in *Proc. of ACM SIGGRAPH '87*, pp. 163–189.

[15] K. Kreeger and A. Kaufmann, "Mixing translucent polygons with volumes," in *Proc. of IEEE Visualization '99*, pp. 191–198.

[16] L. Mroz, R. Wegenkittl, and M. E. Gröller, "Mastering interactive surface rendering for java-based diagnostic applications," in *Proceedings IEEE Visualization 2000*, 2000, pp. 437–440.

[17] L. Sobierajski and A. Kaufman, "Volumetric ray tracing," in *Proc. of IEEE Volume Visualization '94*, Arie Kaufman and Wolfgang Krueger, Eds., pp. 11–18.

[18] D. Ebert and P. Rheingans, "Volume illustration: non-photographic rendering of volume models," in *Proc. of IEEE Visualization 2000*, pp. 195–202.

[19] B. Csebfalvi, L. Mroz, H. Hauser, A. König, and M. E. Gröller, "Fast visualization of object contours by non-photorealistic volume rendering," accepted for publication in the Proc. of EUROGRAPHICS 2001.

[20] V. Interrante, H. Fuchs, and St. Pizer, "Illustrating transparent surfaces with curvature-directed strokes," in *Proc. of IEEE Visualization '96*, R. Yagel and G. Nielson, Eds., Los Alamitos, pp. 211–218.

[21] L. Mroz, *Real-Time Volume Visualization on Low-End Hardware*, Ph.D. thesis, Vienna University of Technology, Austria, 2001.

[22] Ph. Lacroute and M. Levoy, "Fast volume rendering using a shear–warp factorization of the viewing transformation," in *Proc. of SIGGRAPH '94*, A. Glassner, Ed. pp. 451–458, ACM Press.

[23] L. Neumann, B. Csébfalvi, A. König, and M. E. Gröller, "Gradient estimation in volume data using 4D linear regression," *Computer Graphics Forum*, vol. 19, no. 3, pp. C–351–C–357, 2000.

[24] H. Hauser, L. Mroz, G.-I. Bischi, and M. E. Gröller, "Two-level volume rendering - fusing MIP and DVR," in *Proceedings IEEE Visualization 2000*, 2000, pp. 211–218.

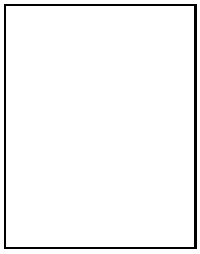
[25] C. Grebogi, E. Ott, and J. A. Jorke, "Chaos, strange attractors, and fractal basin boundaries in nonlinear dynamics," *Science*, vol. 238, pp. 256–261, 1987.

[26] H. N. Agiza, G.-I. Bischi, and M. Kopel, "Multistability in a dynamic cournot game with three oligopolists," *Mathematics and Computers in Simulation*, vol. 51, pp. 63–90, 1999.

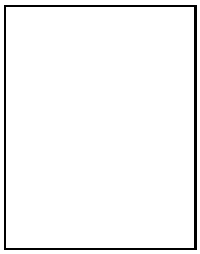
[27] C. Mira, L. Gardini, A. Barugola, and J. C. Cathala, *Chaotic Dynamics in Two-Dimensional Noninvertible Maps*, World Scientific, Singapore, 1996.

[28] Sun Microsystems, "Java," URL: `http://java.sun.com/`.

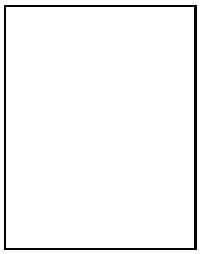
[29] H. Hauser, L. Mroz, G.-I. Bischi, and M. E. Gröller, "Two-level volume rendering," URL: `http://www.VRVis.at/vis/research/two-level/`.



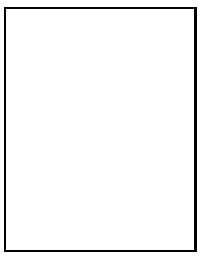
Helwig Hauser graduated in 1995 from the Vienna University of Technology (TU Wien), Austria, as a Dipl.-Ing. (≈ MSc.) after studying computer science with special focus on computer graphics. In 1998, he received the degree of Dr.techn. (PhD) from the same university (TU Wien) after a research project in the field of scientific visualization (visualization of complex dynamical systems). From 1991 until mid-2000, Helwig Hauser was Univ.-Ass. (assistant professor) at the Institute of Computer Graphics (TU Wien, also). There he was a founding member of the visualization group around Prof. M. E. Gröller (`http://www.cg.tuwien.ac.at/research/vis/`). Since then, Helwig Hauser is key researcher at the VRVis Research Center, also located in Vienna, leading the basic research in the field of visualization there (`http://www.VRVis.at/vis/`). His main interests include volume, flow, and information visualization, as well as visualization and computer graphics in general. Helwig Hauser is member of the IEEE Computer Society.



Lukas Mroz graduated in 1998 from the Vienna University of Technology (TU Wien), Austria, as a Dipl.-Ing. (MSc.) in the field of computer graphics. Since then he has been working on projects in the field of Internet-based visualization, medical visualization, and real-time volume rendering. He is just about to receive his PhD from the same university (TU Wien) on the topic of real-time volume rendering. His main interests include volume visualization on low-end hardware, and the visualization of dynamical systems.



Gian Italo Bischi was born in Urbino, Italy, on January 31, 1960. He is married with Nadia and father of Matteo. In 1984 he graduated in Physics from the University of Bologna, Italy, with a thesis on theoretical seismology. His research interests where mainly focused on Mathematical Physics and nonlinear dynamic modelling. From 1985 to 1987 he was researcher and teacher at Italian National Society for Energy (ENI). From 1987 to 1994 he was a teacher of Mathematics and Physics and research collaborator at the Institute of Biomathematics of Urbino University. In this period he published papers on nonlinear modelling of biomedical systems, concerning stability and bifurcation problems in ecology and pharmacokinetics. From 1994 to 2000 he was a researcher in Mathematics for Economic Applications at the Faculty of Economics of Urbino University, and now he is associate professor of Mathematics in the same university. His main research interests concern the global analysis of nonlinear dynamical systems and their applications to the description of time evolution and complexity of Economic, Social, Biological and Physical systems, with special focus on global bifurcations which change the qualitative structure of the attractors and their basins of attraction.



M. Eduard Gröller is an associate professor at the Institute of Computer Graphics and Algorithms, Vienna University of Technology. In 1993 he received his PhD from the same university. His research interests (`http://www.cg.tuwien.ac.at/research/vis/`) include computer graphics, flow visualization, and medical visualization. He is heading the visualization group at the Institute of Computer Graphics and Algorithms. He is member of IEEE Computer Society, ACM (Association of Computing Machinery), GI (Gesellschaft fuer Informatik), and OCG (Austrian Computer Gesellschaft).