Curvature-Based Transfer Functions for Direct Volume Rendering

Jiří Hladůvka
Andreas König
Eduard Gröller

Institute of Computer Graphics
Vienna University of Technology

Abstract

In this paper we present a new concept of transfer functions for direct volume rendering. In contrast to previous work, we attempt to define a transfer function in the domain of principal curvature magnitudes. Such a definition helps the user to suppress or enhance structures of a specific shape class. It also allows to set a smooth color or opacity transition within thick surfaces or even solid objects. From the user's point of view the attractiveness of such transfer functions resides in their easy, (semi)automatic specification.

Keywords: Volume Graphics, Transfer Function, Curvature

1 Introduction

In direct volume rendering, the transfer function is responsible for the classification of a data set. Its task is to assign optical properties to values the data set consists of. During the rendering process, the sampled and/or reconstructed data values are passed through the transfer function to determine their contribution to the final image.

Generally, we can think of a transfer function as a mapping from a cartesian product of scalar fields F to a cartesian product of optical properties O (Fig. 1):

$$\tau : F_1 \times F_2 \times \cdots \times F_n \longrightarrow O_1 \times O_2 \times \cdots \times O_m$$

Due to user interaction problems, the values of n and m are usually small in practice. Typically, a transfer function maps density values ($n = 1$) to opacity and color ($m = 2$), while other optical properties are determined by an illumination model. More sophisticated transfer functions include also the magnitude of the gradient ($n = 2$) in the domain of the transfer function. From a user's point of view, even in this restrictive case ($n, m \leq 2$), it is a problem to specify such a transfer function. There are systems which analyze input data [2, 6] or output images [1] to provide the user with an initial, easy to customize transfer-function setup. Others generate an initial set of transfer functions, pass it to an evolution mechanism [3]

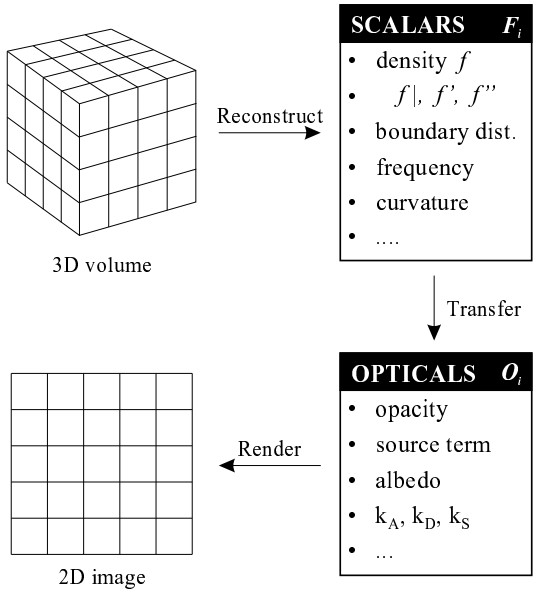

Figure 1: Transfer function: its task, domain and range.

or arrange pre-rendered results to provide the user with an overview of possibilities, hence easier choice of an appropriate transfer function [9]. Such an approach requires much computational time to provide a preview image for each generated transfer function, which made these methods non-interactive for a long time. With upcoming rendering hardware [12], however, the usefulness of these interfaces has increased and the specification problem seems to be solved. This progress made us think of other transfer function types.

According to Lichtenbelt at al [7], the more general ($m > 2$) transfer functions are those that assign opacity, color, and emittance. Other ideas to extend the *range* of transfer functions can be found by studying optical models [10] (see also Fig. 1). Our approach attempts to extend the *domain* of a transfer function. As already said, the typical mapping is defined over densities and gradient magnitudes. The possible, yet not complete list of other choices, can be seen in Fig. 1. To track surfaces, for in-

stance, Kindlmann [6] defines the transfer function with the help of the first and second derivatives in the direction of the gradient. Although not presented as a transfer function approach, Lürig [8] involves frequency information to visualize the thickness of objects.

The domain of transfer functions presented in this paper is defined by the magnitudes of the principal-curvatures.

From differential geometry it is known, that the vicinity of any point on a regular surface can be described by two tangent vectors - *principal directions* and two corresponding real numbers - *principal curvatures*. This description yields an unique, view-independent characterization.

Although originally developed for smooth analytic surfaces, in recent years curvature information is also used in a variety of applications in the field of volume visualization. An obvious application is to use *Gaussian curvature* to distinguish among parabolic, elliptic and hyperbolic parts of surfaces. Interrante [4, 5] has used the *principal directions* to define a flow field over a surface to accentuate its shape. Trucco and Fisher [17] attempted to segment the sampled data with the help of both Gaussian and *mean curvatures*. Recently, Tang and Medioni [15] extended the tensor voting mechanism by *curvature sign* information to get better densification (i.e. reconstruction) of sparse input data.

The rest of this paper is organized as follows. In the next section we will present a new concept of transfer-functions specification. The definition involves principal curvatures, whose estimation will be discussed in section 3. The results section 4 demonstrates the applicability and initial, automatically generated specifications of the proposed transfer function. The conclusions and hints for future work will be given in section 5.

2 Concept of a Curvature-Based Transfer Function

For a specific point P on a regular surface, the principal directions $\vec{s}_1$ and $\vec{s}_2$ give us an idea about where the surface bends the most and the least, respectively. The corresponding quantitative measure (i.e. how much does the normal change in these directions) is expressed by two real numbers known as principal curvatures κ_1 and κ_2.

With the help of principal directions and curvatures, a surface can be locally approximated, up to order two, by a quadratic patch. According to the signs of κ_1, κ_2 we know immediately whether the surface is locally approximated by a

(i) plane (iff $\kappa_1 = \kappa_2 = 0$),

(ii) parabolic cylinder (iff $\kappa_1 > \kappa_2 = 0$ or $0 = \kappa_1 > \kappa_2$)

(iii) paraboloid (iff $\kappa_1.\kappa_2 > 0$) or

(iv) hyperbolic paraboloid (iff $\kappa_1.\kappa_2 < 0$).

The transfer function we are going to design will map pairs of principal curvatures to optical properties, e.g. traditional color and opacity in the RGBα model:

$$\tau : \kappa_1 \times \kappa_2 \longrightarrow R \times G \times B \times \alpha$$

Such a definition will be useful to:

distinguish among shapes. In specific applications, it may be useful to visualize surfaces with respect to their shape. This does not include only different properties of the four different cases introduced above. Our approach allows also to distinguish shapes of the same class employing curvature magnitudes. In engineering for instance, it can be distinguished between planar ($\kappa_1 = \kappa_2 = 0$) and tubular ($\kappa_1 > \kappa_2 = 0$) structures. In addition, within the class of tubular structures, different properties can be specified with respect to the magnitude of κ_1 (i.e with respect to the cylinder's radius).Using the same principle, a medical application is able to suppress registration markers which are typically of tubular (chord) or planar (landmarker) shape. Surgical tools can also be selected, if they have specific shape properties. Last but not least, several human organs, like bones, vessels or colon polyps might be segmented because of their specific shape.

set smooth transitions. We can understand thick surfaces and solid objects as a set of coherent layers. This, for instance, can be defined with the help of isovalues or a distance transform. Within such a surface it might be useful to see how the curvature (hence shape) changes inside. To convey this, Interrante [4] exploits principal directions to define a texture. In her approach, however, a maximum of three or four such layers can be visualized simultaneously without loss of comprehensiveness. Apart of that, this method could be hardly applicable for small structures, e.g. blood vessels. Involving curvature magnitudes, smooth color transition within even small solid objects (say five or six voxels in diameter) can be set, allowing to understand what happens inside. A possible application in medicine is the identification of stenose.

set the transfer function (semi)automatically. To enhance or suppress specific shape class, it is obvious what combination of values κ_1, κ_2 has to be chosen. In order to provide flexibility with respect to shape-by-magnitude distinction and color transition for application specific tasks, however, a simple user interface is necessary. The setup issues will be briefly discussed in section 4.

Being dependent on two real numbers it would be necessary to specify the proposed transfer function in the entire

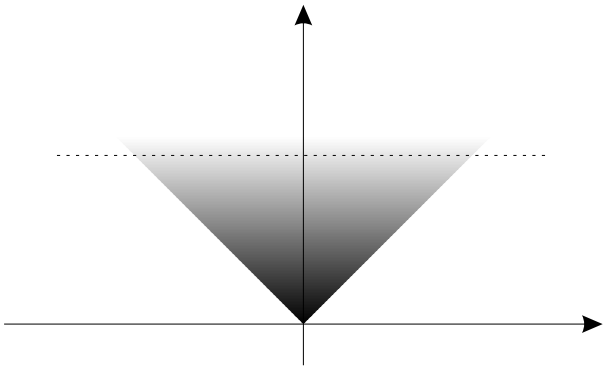

Figure 2: The domain of curvature-based transfer functions

plane $\mathbf{R^2}$. Fortunately, the following facts gradually allow us to restrict its domain:

1. For analytically defined patches, curvatures are by definition given such that $\kappa_1 \geq \kappa_2$ (see also section 3.1). The convexity or concavity of paraboloids and parabolic cylinders is determined by the curvature sign(s).

2. For surfaces defined implicitly (as density volumes are), the normal orientation is (at least locally) ambiguous and so is also the convexity/concavity of paraboloids and parabolic cylinders. It is impossible to judge the cases $\kappa_1 > \kappa_2 \geq 0$ and $0 \geq \kappa_1 > \kappa_2$. The signs of κ_1, κ_2 surely identify just the planar (both are zero) and hyperbolic (they straddle zero) cases. Therefore we can rearrange the principal curvatures such that κ_1 is nonnegative and reflects the faster bending of the surface. This is simply done by a sign change and a swap of curvatures in points where $|\kappa_1| < \kappa_2$. Such a rearrangement ensures, for all possible cases, that $\kappa_1 \geq |\kappa_2|$.

3. The principal curvature, as a curvature of a planar curve, is defined as a reciprocal of radius of its osculating circle. Since in unit–distance cartesian grids we can surely assume a nonpresence of circles with radius smaller than 1/2, the curvature magnitudes will be always less than two.

Due to 2) and 3) the domain of transfer function shrinks from $\mathbf{R^2}$ to

$$|\kappa_2| \leq \kappa_1 \ll 2 \tag{1}$$

Referring to Fig. 2, the origin corresponds to planar points, the positive κ_1 axis to parabolic points, and the areas to the right and the left correspond to elliptic and hyperbolic points, respectively. This layout will serve as a base for specification discussed in section 4.

For analytically defined patches, we could start presenting results here. Curvature estimation (our concept relies on) in digital scenes, however, is a nontrivial task which is discussed in the following section.

3 Curvature Estimation

Despite big efforts in research on recovery of curvature information from sampled data, the results are still at least disputable. Particular success can be seen in estimation of qualitative properties (the principal directions and the sign of the Gaussian curvature) rather than quantitative (the principal curvature magnitudes) [15]. The difficulties of magnitude estimation arise from the properties of digital scenes, mainly noise, anisotropy and related directional dependencies.

Methods which estimate the curvature of a surface can be basically divided into two groups. There are algorithms which estimate derivatives and apply fundamental forms (e.g. [4]). These methods strongly rely on accurate derivative reconstruction, are sensible to noise and require low-pass filtering with a rather large window. The alternative approach is the local fitting of a patch, the curvatures can be analytically computed from. McIvor [11] concludes that fitting of a quadratic patch gives better results than fitting of an arbitrary surface.

In our implementation we have, for several reasons, adopted an algorithm introduced in [16]. Firstly, in this paper the authors come up with a concept which gradually reduces the surface-curvature estimation to finding a set of *planar* curves, estimation of their tangents and curvatures, and fitting of a quadratic curve. Neither derivative estimation in 3D nor patch fitting are therefore necessary. Due to two-dimensionality of all these steps one can expect not only easier implementation but also more reliable results. Secondly, the authors conclude presenting results which are superior to those highlighted in [11], achieved by fitting of a quadratic interpolant. Finally, in contrast to the methods which rely on estimation of the surface normal, the normal is computed from the estimated curve tangents as a side product.

3.1 From Surface to Planar Curves

At a fixed point P of a regular surface S, an arbitrary unit tangent vector $\vec{t}$ together with the surface normal $\vec{n}$ define a plane which in the vicinity of P meets S in the curve of intersection. The curvature $\kappa_n(\vec{t})$ of this curve is referred to as *normal curvature* in point P and direction $\vec{t}$. The normal curvature as a real function of $\vec{t}$ being defined on the compact set of unit tangent vectors reaches a maximum and a minimum. Directions in which this happens are known as principal directions $\vec{s_1}$, $\vec{s_2}$ and the corresponding curvatures $\kappa_1 = \kappa_n(\vec{s_1})$, $\kappa_2 = \kappa_n(\vec{s_2})$ as principal curvatures. Vectors $\vec{s_1}$, $\vec{s_2}$ and $\vec{n}$ make up the *principal frame*.

The principal curvatures κ_1 and κ_2, we need to estimate, can also be found in the definition of the *Dupin indicatrix*. The Dupin indicatrix is either one or a pair of conics in the

tangent plane defined, assuming an arbitrary orthonormal coordinate system in the tangent plane with origin at point P, by the following equation:

$$Lx^2 + 2Mxy + Ny^2 = \pm 1 \tag{2}$$

Changing the coordinate system such that the eigenvectors of the quadratic form (2) become its axes, however, the Dupin indicatrix will be expressed in the more convenient form:

$$\kappa_1 x^2 + \kappa_2 y^2 = \pm 1 \tag{3}$$

Therefore, if we know the Dupin indicatrix we also know the principal curvatures κ_1, κ_2.

In order to reconstruct the Dupin indicatrix, we need to know at least three of its points. To compute them we define the following map:

$$D(\vec{t}) = \vec{t} \,/\, \sqrt{|\kappa_n(\vec{t})|} \tag{4}$$

This map scales each given unit tangent vector $\vec{t}$, in which the normal curvature $\kappa_n(\vec{t})$ is nonzero, to a positional vector of a point on the Dupin indicatrix. This can be proven with the help of the *Euler theorem*, which establishes a relation between principal curvatures and a normal curvature in an arbitrary direction $\vec{t}$:

$$\kappa_n(\vec{t}) = \kappa_1 \cos^2\varphi + \kappa_2 \sin^2\varphi$$

where $\varphi = \varphi(\vec{t})$ is the angle between $\vec{t}$ and $\vec{s_1}$. Taking the principal frame, unit vector $\vec{t}$ becomes $(\cos\varphi, \sin\varphi)$ and for its image $D(\vec{t}) = (D_x, D_y)$ holds:

$$\begin{aligned} \kappa_1 {D_x}^2 + \kappa_2 {D_y}^2 &= \frac{\kappa_1 \cos^2\varphi + \kappa_2 \sin^2\varphi}{|\kappa_n(\vec{t})|} = \frac{\kappa_n(\vec{t})}{|\kappa_n(\vec{t})|} = \\ &= sign\ \kappa_n(\vec{t}) = \pm 1 \end{aligned}$$

which corresponds to definition (3).

Taking k ($k \geq 3$) nonzero normal curvature estimates $\kappa_n(\vec{t_i})$ in k distinct unit tangent directions $\vec{t_i}$ we can therefore reconstruct k points (x_i, y_i) on the Dupin indicatrix and set up a system of k equations

$$L{x_i}^2 + 2Mx_iy_i + N{y_i}^2 = sign\ \kappa_n(\vec{t_i}) \tag{5}$$

The coefficients L, M, N are found as a solution of a linear equations system ($k = 3$) or a least square fitting algorithm ($k > 3$). Principal curvatures κ_1, κ_2 and principal directions $\vec{s_1}, \vec{s_2}$ are eigenvalues and eigenvectors of the quadratic form (2), i.e. of the matrix

$$\begin{bmatrix} L & M \\ M & N \end{bmatrix}$$

The estimation of the normal curvature in a given tangent vector would require firstly a knowledge of the surface normal $\vec{n}$ in P and secondly a reconstruction of a curve in the normal–section. This would of course cause the usual reconstruction problems.

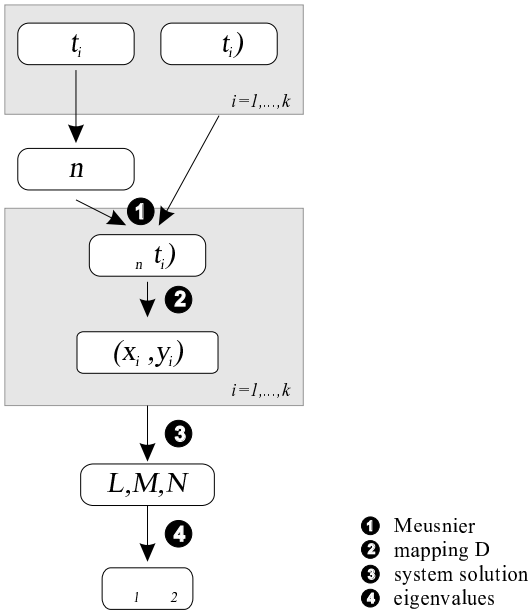

Figure 3: Towards principal curvatures

Instead, due to another result from differential geometry, we just need to reconstruct k arbitrary planar (i.e. not necessarily normal–section) curves γ_i passing through P and estimate their tangent vectors $\vec{t_i}$ and curvatures $\kappa(\vec{t_i})$. Averaging the cross products $\vec{t_i} \times \vec{t_j}$ of tangent vectors allows us to compute the surface normal $\vec{n}$. Normal curvatures $\kappa_n(\vec{t_i})$ can then be enumerated due to the *Meusnier theorem*:

$$\kappa_n(\vec{t_i}) = \kappa(\vec{t_i}) \cos\psi \tag{6}$$

where ψ denotes the angle between the plane of the normal section (given by vectors $\vec{n}, \vec{t_i}$) and the plane of curve γ_i.

In this section we have shown how to reduce the problem of principal–curvature estimation to that of estimating of plane curvatures[1]. The entire procedure is summarized in Fig. 3.

3.2 Plane Curvature Estimation

The computation of curvature from a digitized curve is a non-trivial task which should be considered with care [18]. In research on shape analysis of digital curves, Worring and Smeulders [19] identify five essentially different methods for measuring curvatures of digital curves. These methods are based on three different formulations of curvature: tangent orientation change, second derivative of the curve considered as a path, and osculating circle touching the curve. In their work the authors conclude, that none of the presented methods is robust and applicable for all curve types. They advice, however, which method outperforms the others for a specific application.

[1] *plane curvature* refers to the curvature of a planar curve

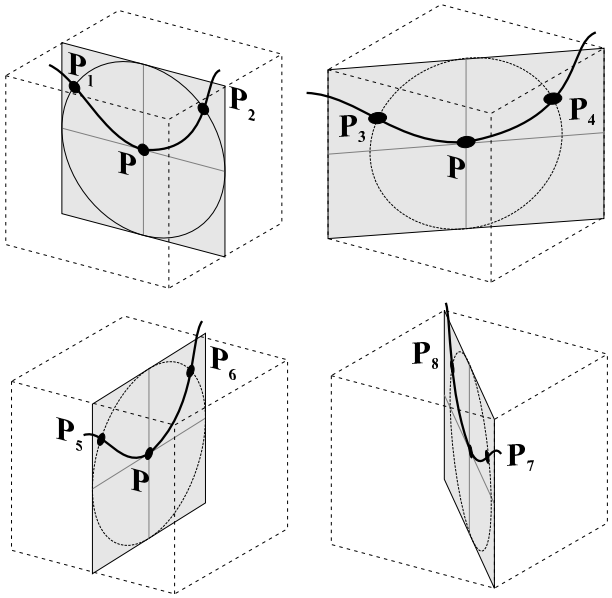

Figure 4: Reconstruction of neighborhood isopoints

The very first aim of this work was to come up with a transfer function for visualization of tubular structures, which feature nearly constant and large radii. For this case, the conclusion of [19] recommends to formulate the curvature with the help of osculating circles. In the following we describe our implementation for grid data sets.

To approximate curvatures κ_i of planar curves γ_i passing through P, we reconstruct points in the neighborhood of P from the isosurface defined by the density value of P. The triplets consisting of P and two isosurface points will approximate the osculating circles. To avoid anisotropy typical for rectilinear data sets, these points should lie on a unit sphere with the center in P. In order to reduce reconstruction errors we advice to reconstruct these points via bilinear interpolation in four planes passing through grid points (Fig. 4). As a result we have eight surface points $P_1,\ldots,P_8$ in a small neighborhood of P. For $u \neq v$, each triplet of points P,P_u,P_v lie on some planar curve passing through P. There are two cases:

A) P,P_u,P_v are colinear and define a tangent vector $\vec{t_i}$ to the surface at P. The corresponding normal curvature $\kappa_n(\vec{t_i})$ is zero and can therefore not be used to compute a point on the Dupin indicatrix according to the map defined in equation (4). This case provides us, however, with a tangent vector and contributes as such to a better estimation of normal $\vec{n}$ which is necessary for the use of equation (6).

B) P,P_u,P_v approximate an osculating circle with the center C. In order to use the Meusnier theorem (6) we need to compute, in addition, a tangent vector $\vec{t_i}$ as a cross product $((P_u-P)\times(P_v-P))\times(C-P)$ and the curvature $\kappa(\vec{t_i})$ as a reciprocal of the circle's radius, i.e. $1/|(C-P)|$.

3.3 Implementation Issues

The definition of map (4) presumes a nonzero normal curvature κ_n. This is not fulfilled, however, for case A mentioned in the previous section. Here, such a triplet of points is of no use for the computation of a point on the Dupin indicatrix, and consequently does not contribute to the total number k of equations in system (5). In the worst case, e.g., for planar points where all the possible triplets are colinear, the total number k of equations can be less than three, which is not sufficient for finding the coefficients L,M,N. To circumvent this difficulty and, at the same time, to handle all the cases uniformly, we reassign the zero curvature κ_n to some small constant ε. In order to avoid numerical problems, this constant should not be too small. On the other hand it should sufficiently reflect the planarity of the neighborhood of P. For low resolution (e.g. 128^3) volumes we have successfully (Fig. 9) set $\varepsilon = 10^{-4}$. This corresponds to curves of sufficiently large radii of 10000 voxels.

The algorithm described in this section is computationally expensive. A possible place to speed–up would be the reconstruction of neighborhood points. As we show, the curvature estimation is very sensitive to how this reconstruction is done, therefore this should be considered with care. Having a 3×3 density matrix with the center in P in the reconstruction plane, a first simplification might be achieved interpolating just from the 4-neighbors. The second one would be to find the isovalues on a diamond ($|x|+|y|=1$) rather than on the unit circle with center in P.

To demonstrate the influence of improper interpolation on curvature estimation, we present a middle slice of the first principal curvatures κ_1 reconstructed from a $361\times361\times3$ volume of concentric cylinders (Fig. 5). In order to see the curvature isolines we depict the intensities of $1/\kappa_1$ *mod* 32. Where concentric circles are expected, the left image exhibits an anisotropy with the maximum in diagonal directions. This is a consequence of interpolating just from four neighbors. The situation improves considerably using all eight neighbors for bilinear interpolation. The remaining artifacts appearing in diagonal direction of the right image have been caused by an approximation of the circle by a diamond.

4 Results

To demonstrate our new concept we refer to the color plate[2] . Fig. 6 depicts an example of a transfer–function specification scheme with respect to the definition (1) of its domain (see also Fig. 2).

Recalling the distinction of the four shape classes introduced in section 2 one would expect an exact segmentation of the transfer–function domain. For practical appli-

[2] Also available via http://www.cg.tuwien.ac.at/research/vis/vismed

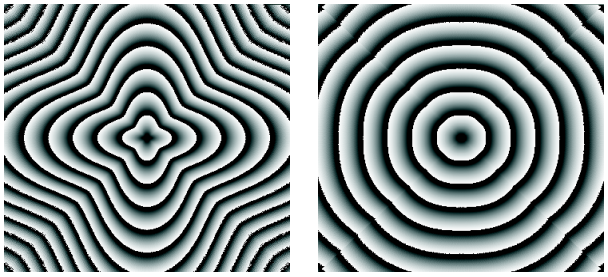

Figure 5: Influence of interpolation on curvature estimation

cations, however, we find it useful to provide the user with a certain degree of tolerance. Consequently, the sharp borders between shape classes change to *transition areas*.

The green area in the vicinity of the origin corresponds to planar points. The blue–yellow transition area specifies the curvature change inside parabolic structures and is intended to reflect the diameter change within solid cylinders presented in the data sets. The red area corresponds to elliptic points.

Slight alterations of this specification are used to render the following density volumes, generated by [13]:

A wire frame cube (Fig. 7). This is a demonstration of the curvature change inside solid cylinders ($\kappa_1 > \kappa_2 \approx 0$) of a $38\times38\times38$ cube. Note, that the diameter of cylinders in the data set is less than six voxels. In order to attract the user's attention, the high values of κ_1 (i.e. small diameters) have been mapped to bright yellow. The smooth transition to blue towards lower values of κ_1 corresponds to diameter increase. As the cylinder axes do not define a surface, they have been excluded from curvature computation and therefore do not affect the final image. The red parts correspond to elliptic points ($\kappa_1 \geq \kappa_2 > 0$).

A wire frame octahedron (Fig. 8). The transfer function has been specified in the same way as for Fig. 7, with more emphasis on smaller cylinders (depicted in yellow). A staircase effect in diagonal directions can be noticed. The resolution of the data set is $59\times59\times59$ voxels.

A facet cube (Fig. 9). A $38\times38\times38$ data set similar to that used in Fig. 7 with attached faces. The transfer function maps the corresponding (i.e. zero) curvatures to transparent green. The joint of faces with cylinders was not smooth and exhibits therefore high curvature depicted in yellow. Similarly as in Fig. 7, the red areas correspond to elliptical points.

Transfer functions used for rendering of figures 10 and 11 additionally require the specification also in the area of hyperbolic points ($\kappa_2 < 0$):

A torus (Fig. 10). The transfer function has been set to distinguish among elliptic (red), parabolic (green) and hyperbolic (blue) points of a $59\times59\times20$ torus. The green points on the outer side are identified as planar due to a volume crop.

The Möbius strip (Fig. 11). Visualization of low (green) and high curvature (red) points of a $50\times52\times16$ thickened Möbius strip.

The reconstruction of curvature using the method described in section 3 involves many steps. Its time complexity depends mainly on how many plane curvatures (i.e., equations of system (5)) are reconstructed, and how the points on the isosurface are reconstructed. In the discussion in section 3.3 we gave reasons why the reconstruction of curve points should be done as accurately as possible. Therefore we do not encourage to save time there. Instead, time should be saved adopting the number k of the equations in the system (5). To demonstrate the timing we have used the extreme values of k. For $k = 3$, the curvature has been reconstructed in approximately 4500 voxels per second while for $k = 28$ the speed was about 2000 voxels per second. The time has been measured on a PC with a 400 MHz PentiumII CPU and 512 MB of RAM.

5 Conclusion and Future Work

We have proposed a new class of transfer functions which assign optical properties to principal curvatures reconstructed from the input data. Such transfer functions allow to set (at least locally) the optical properties to objects with respect to their shape. Moreover, within one shape class, the objects can be distinguished by curvature magnitudes. As opposed to density transfer functions, curvature transfer functions allow to see the structural changes inside solid objects even if the density changes are small. In contrast to the density transfer functions, moreover, both the domain and the significance of its parts are application independent. This yields an automatic initial setup and easy specification by the user.

On the other hand, there are several facts which make the implementation of the presented concept difficult. Firstly, it is an absence of a robust algorithm for estimation of principal curvature magnitudes. The algorithm we have used reduces this problem to curvature estimation of planar curves and is thus only dependent on the accuracy of methods which deal with this two-dimensional sub-problem. These methods, however, are not supposed to be robust and a specific algorithm should be chosen with care for a particular application. Our implementation of the osculating circle method, for instance, tends to exhibit staircase artifacts in areas where the surface's principal directions are not aligned to the grid axis of the input volume. Secondly, the curvature estimation can be time demanding, which can make the concept unsuitable for online rendering. The curvatures can be, however, computed in a preprocessing step and stored in separate volumes.

Future research should primarily concentrate on a better estimation of curvature. For the presented method, for instance, the use of a larger neighborhood or better reconstruction filters for the description of planar curves can be taken into consideration. A quantitative error analysis and a comparative study with other algorithms would help to find out the method usable for visualization of real data.

Acknowledgments

The work presented in this publication has been funded by the V[is]M[ed] project. V[is]M[ed] is supported by *Tiani Medgraph*[3], Vienna, and the *Forschungsförderungsfonds für die gewerbliche Wirtschaft*[4], Austria. Please refer to `http://www.vismed.at` for further information on this project.

The artificial data sets have been voxelized by vxTools [13] using algorithms described in [14].

References

[1] S. Fang, T. Biddlecom, and M. Tuceryan. Image-based transfer function design for data exploration in volume visualization. In *Proceedings of IEEE Visualization*, pages 319–326, 1998.

[2] I. Fujishiro, T. Azuma, and Y. Takeshima. Automating transfer function design for comprehensible volume rendering based on 3D field topology analysis. In *Proceedings of IEEE Visualization*, pages 467–470, 1999.

[3] T. He, L. Hong, A. Kaufman, and H. Pfister. Generation of transfer functions with stochastic search techniques. In *Proceedings of IEEE Visualization*, pages 227–234, 1996.

[4] V. Interrante. Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. In *Proceedings of ACM SIGGRAPH*, pages 109–116, 1997.

[5] V. Interrante, H. Fuchs, and S. Pizer. Illustrating transparent surfaces with curvature-directed strokes. In *Proceedings of IEEE Visualization*, pages 211–218, 1996.

[6] G. Kindlmann and J. W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of IEEE Volume Visualization*, pages 79–86, 1998.

[7] B. Lichtenbelt, R. Crane, and S. Naqvi. *Introduction to Volume Rendering*. Prentice Hall, 1998.

[8] C. Lürig and T. Ertl. Hierarchical volume analysis and visualization based on morphological operators. In *Proceedings of IEEE Visualization*, pages 335–342, 1998.

[9] 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 *Proceedings of ACM SIGGRAPH*, pages 389–400, 1997.

[10] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[11] A. M. McIvor and R. J. Valkenburg. A comparison of local surface geometry estimation methods. *Machine Vision and Applications*, 10(1):17–26, 1997.

[12] H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer, and L. Seiler. The VolumePro real-time ray-casting system. In *Proceedings of ACM SIGGRAPH*, pages 251–260, 1999.

[13] M. Šrámek. vxt: A class library for voxelization of geometric objects. http://www.cs.sunysb.edu/ vislab/vxtools/.

[14] M. Šrámek and A. Kaufman. Alias-free voxelization of geometric objects. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):251–267, 1999.

[15] C.-K. Tang and G. Medioni. Robust estimation of curvature information from noisy 3D data for shape description. In *Proceedings of International Conference on Computer Vision*, pages 426–433, 1999.

[16] P. Todd and R. McLeod. Numerical estimation of the curvature of surfaces. *Computer-Aided Design*, 18(1):33–37, 1986.

[17] E. Trucco and R. B. Fisher. Experiments in curvature-based segmentation of range data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):177–182, 1995.

[18] M. Worring. *Shape Analysis of Digital Curves*. PhD thesis, Department of Computer Science, Faculty of Science, University of Amsterdam, 1993.

[19] M. Worring and A. W. M. Smeulders. Digital curvature estimation. *CVGIP: Image Understanding*, 58(3):366–382, 1993.

[3] http://www.tiani.com

[4] http://www.telecom.at/fff

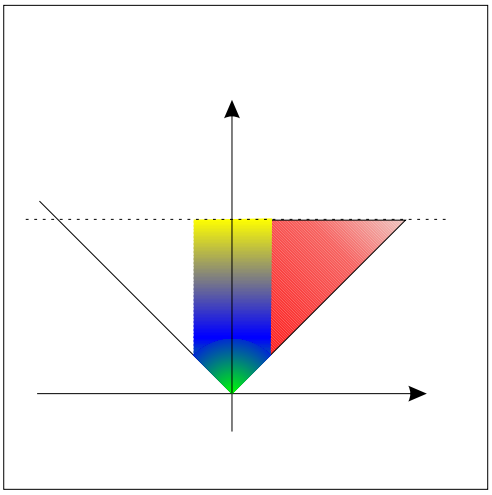

Figure 6: Specification for Figures 7, 8, and 9

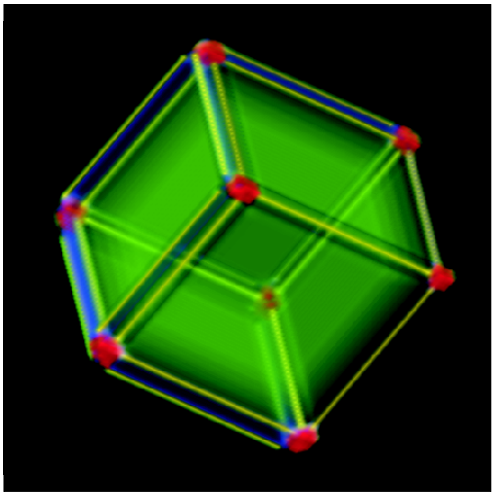

Figure 9: Facet cube

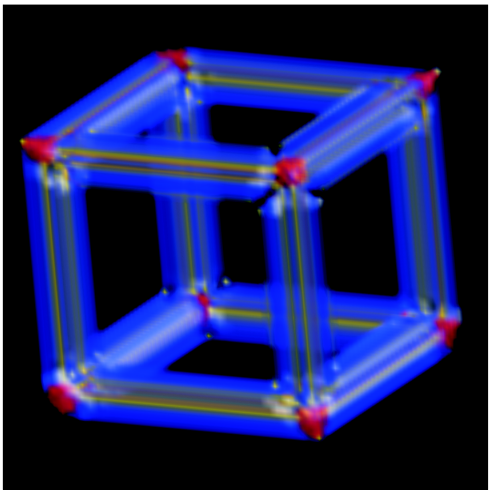

Figure 7: Wire frame cube

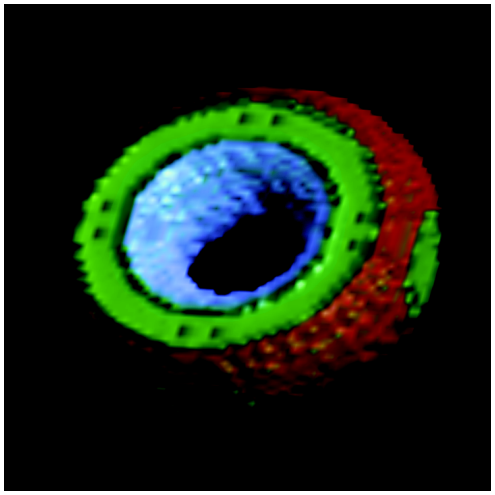

Figure 10: Torus

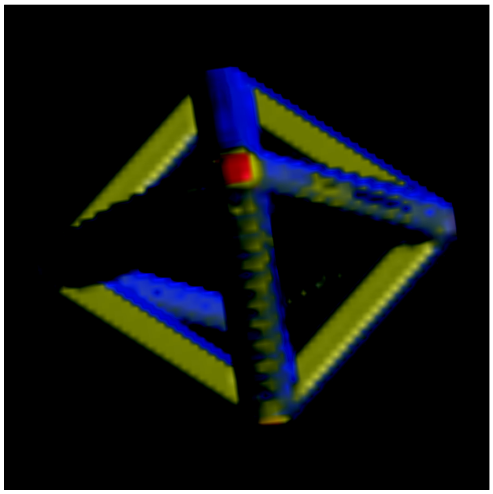

Figure 8: Wire frame octahedron

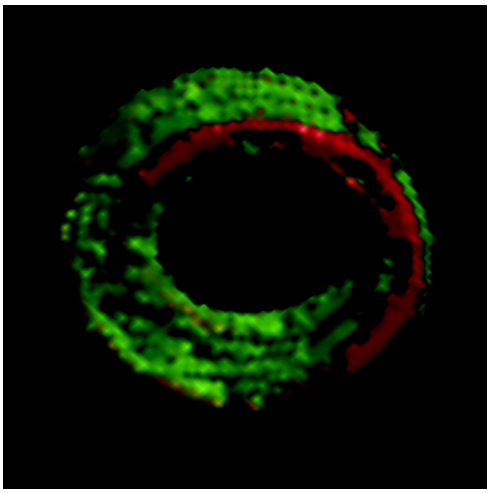

Figure 11: Möbius strip