IEEE/EG International Symposium on Volume Graphics (2010)
R. Westermann and G. Kindlmann (Editors)

A Survey of Transfer Functions Suitable for Volume Rendering

S. Arens and G. Domik

University of Paderborn, Germany

Abstract

There are many transfer functions (TF) that emphasize or hide special features of volume data. Their potential to cleverly generate a color and opacity value for direct volume rendering is primarily determined by the used metrics besides the input data value. Despite this variety of TFs, for the most part simple one dimensional data value only based TFs are used in practice. This survey will therefore examine the differences in representative TF types (defined by their used metrics) to provide a basis for selecting the right TF type for the boundary conditions that describe an individual field of application and task. Besides fundamental properties like metrics or memory consumption, we will also give an assessment about user interaction and quality of feature emphasis.

Categories and Subject Descriptors (according to ACM CCS): l.3.3 [Computer Graphics]: Picture/Image Generation—l.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—

1. Introduction

In clinical routine volume data is widely used for diagnosis and surgical planning. While radiologists mainly use 2D slices for diagnosis and imagine a 3D representation, surgeons prefer a meaningful 3D visualization, since it matches their view onto the operating field. Transfer functions (TF) are a commonly used technique to achieve a meaningful 3D view in volume visualization. In this process data values are mapped to color and opacity, with the result that features can be emphasized or hidden. Since different organs often correspond in their signal they produce in medical 3D scanners, a simple distinction by measured value is not sufficient in most cases. Thus additional metrics are computed that help to differentiate types of tissue, leading to two- and multi-dimensional TFs.

Often, decisions made by algorithms are not acceptable in medical usage, hence automatic segmentation of a dataset is not applicable in every case (in spite of the fact, that segmentation might be very difficult anyway). TFs are much more harmless and flexible, since they typically do not provide a binary decision. Rather they demand a visual exploration of datasets (which is good [PLB*01]) and provide a sort of original view to the data in different contrast ratios. Nevertheless, higher dimensional TFs are not often used in practice, due to the fear of high amount of user interaction, missing precision or lack of intuition in their use.

For a further clarification which TF fits best for which field of application and task, their risks and properties must be compared. Hence in this paper the differences of the following 6 TF types *1D data-based TF* (1dTF), *gradient-based 2D TF* (gbTF) [Lev88, KD98, KKH02], *curvature based TF* (cbTF) [KWTM03], *size-based TF* (sbTF) [CM08, WK09b], *texture-based TF* (tbTF) [CR08] and *distance-based TF* (dbTF) [TPD06] are compared in a table. It would hardly be possible to compare all published TFs, therefore a representative selection has been chosen, covering the most relevant types. Besides some fundamental properties, like used *metrics*, *real-time capabilities*, *memory consumption* and degree of *automaticity* we also give an assessment about the needed *user interaction*, *constraints* and the quality of *feature emphasis*. Additionally, widgets and methods are listed that work together with the TF types to support the user. The analysis is made with a medical view, but could be easily applied to other fields of application. It can also be of value for students looking on different TFs in their volume rendering investigations.

DOI: 10.2312/VG/VG10/077-083

In literature the use of the term "transfer function" is not consistent. It can be a single specification of a mapping from its domain (that is the used metrics) to its range (in our case always color and opacity), a technique how to specify such a mapping, or a type of such a mapping, defined by its domain and its range. Because they are functions in the strict mathematical sense, we use the third, defining the type of a function by its domain and range, as suggested in [Kin02].

2. Previous work

The development of TFs for volume rendering started with volume rendering itself. Since tissues in medical data often share one signal and a 1D data-only driven TF is not able to distinct between samples with the same signal, a lot of effort was spent to develop two-dimensional or even higher-dimensional TFs. First, the use of the gradient allowed to distinct transitions between tissues [Lev88]. Enhanced with the second derivative even more distinctions in transitions were possible [KD98]. Driven by these capabilities of separation even more second- and higher order metrics were calculated to distinguish between tissues, e.g. [HKG00, KWTM03, CM08, CR08]. When specifying these TFs in the domain of data values (Figure 1 bottom), instant visualization in the spatial domain (Figure 1 top) is quite important. Only direct feedback allows a trial and error approach for specifying a TF.

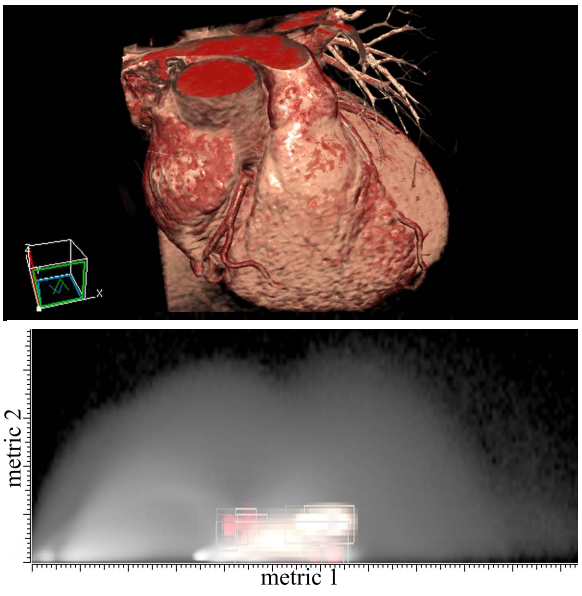

Figure 1: *The spatial domain (top) provides visual feedback of a TF that is specified in the domain of data values (bottom).*

However, the higher the dimension of the TF, the higher the complexity of specifying it. Even two-dimensional TFs require a fair amount of user interaction to find a meaningful specification. To address this issue, intuitiveness of metrics and widgets as well as automated techniques has been a research topic in the past. Concerning the first matter, size seems to be a much more intuitive metric than the gradient [WK09a]. In "the transfer function bake-off" [PLB*01] four possible ways for specifying a gradient based 2D TF are compared, regarding their different user interface approaches. Two of them use histograms in the domain of data values for a first insight into the underlying data, leading to a better identification of features [KD98]. They are not very intuitive, though. Several widgets are presented in [KKH01] to support the complex handling in both domains, leading to a more intuitive so called *dual-domain interaction*. Another intuitive image centric interface approach is presented in [MAB*97], where the user chooses from a couple of renderings in the spatial domain, created by possible TF specification.
Automated techniques are especially needed in higher dimensional TFs [TLM03, CR08]. Nevertheless, 2D-TFs are also supported by neural networks and spatial grouping, leading to semi-automatic interfaces [ZEAD08] and precolorized histograms in the domain of data values [RBS05]. But often the important process of data exploration gets lost, when automated techniques are used.

Regarding past research of TFs, two kinds of publications have to be distinguished. First, the group of TFs that are defined by the set of used *input metrics*. Second, the group that finds new ways of *specifying* such a TF, using automated techniques or widgets. Although often named "TF" the second group describes an extension to the first. In this paper only publications of the first group are considered.

3. The comparison of the transfer functions

The analysis consists of both a short per TF description with illustrations demonstrating their properties (section 3.1 - 3.6), as well as comparison of the TFs in a table (see columns of Table 1). Several properties (rows of Table 1) have been regarded:

- Real-time capability. Since it is difficult to compare this property objectively as a whole, we divide it into *real-time editability*, *pre-processing* and the possibility to compute the TF in shader-programs. Regarding the latter, the number of *texture look-ups* shall give a quantitative statement, as texture look-ups mostly constitute the critical part of rendering time. Due to caching, the expressiveness of this value can be inconsistent. Times quoted are the original measurements in the referenced papers. That is because in our implementation for example of size-based TF we do not use the GPU for preprocessing and thus achieve quite different timings.
- Memory consumption during run-time, i.e. in particular the number of *additional volumes* needed during rendering (this is often a bottleneck, since they need to reside on

	Name	1D data-based TF	gradient-based TF	curvature-based TF	size-based TF	texture-based TF (automatic/semi-automatic)	distance-based TF
	Metrics	*Data value*	*Data value, gradient*	*Data value, curvature*	*Data value, scale*	*Data value, 20 texture metrics*	*Data value, distance to previously segmented structure*
	Literature (e.g.)	-	[Lev88,KKH02]	[KWTM03,PF05]	[CM08]	[CR08]	[TPD06]
Real-time capability	Real-time editing*	*Yes*	*Yes*	*Yes*	*Yes*	*No* **	*Yes*
	# 3D texture look-ups per sample	*1(data)*	*7 (1 data + 6 gradient: needed for illumination anyway)*	*Minimum 27* **	*2 (data + scale)*	*2 (data + label) / 21 (data + metrics)*	*2 (data + distance)*
	#1D + 2D look-ups	*1*	*1*	*1*	*1*	-	*1*
	Time of preprocessing	-	-	-	*~10s for a dataset of 256^3 and 128 scales*	*261s / 233s for a dataset of $256^2 \times 128$*	*20s for a dataset of 256^3***
Memory	Additional volumes during rendering	-	-	-	*1*	*1 / up to 20 (not necessarily full size)*	*1 (not necessarily full size)*
	Other data structures	*1D-look-up-table*	*2D-look-up table*	*2D-look-up table*	*2D-look-up table, 3 volumes in preprocessing*	*20 metric-, 1 GLCM-, 1 run-length-volumes in preprocessing*	*Segmentation, 2D-look-up table*
Emphasis	Emphasized regions	*Monotone areas*	*Surfaces*	*Surfaces,contours*	*Monotone areas*	*Textured areas*	*Neighborhood*
	Emphasis ranking***	●○○	●●○	●●○	●●○	●●●	●●○
	Risk of a critical misinterpetation	*Low*	*Moderate*	*Moderate*	*Moderate*	*High (no exploration process)*	*Low*
Constraints	Constraints to dataset	-	*No additional information in blurry datasets compared to 1dTF*	-	-	*The bigger the dataset, the smaller the precision (memory consumption)*	*Only segmented data, not practicable for time-dependent data*
	Error-prone to noise	*Medium*	*Very*	*Medium*	*Little*	*Very*	*No, only the segmentation*
User interaction	Type of interaction	*Specification on 1D diagram*	*Specification on 2D diagram*	*Specification on 2D diagram (checkbox or similar for contours only)*	*Specification on 2D diagram*	*Specify a number of different structures / setting of weights, mark one/two ROIs*	*Specification on 2D diagram*
	Duration	*<1 Min*	*1 – 2 Min*	*1 – 2 Min*	*1 – 2 Min*	*0 / >2 Min*	*1 – 2 Min*
	Helpful prior knowledge	*Application: values*	*Application: values, technical: gradient*	*Application: values*	*Application: values*	*- / technical: statistics (optional)*	*Application: values*
Automaticity	Type of automaticity	-	*Spatial analysis and neural networks (possible)*	-	-	*Clustering of structures with similar texture properties*	-
	Assisting widgets (possibility)	*1D histogram, data probe*	*2D histogram, data probe*	-	*2D histogram*	*Dictionary with predefined weights for metrics*	*2D histogram*
	Enhances a precise segmentation	*No*	*No*	*Yes, additional information about curvature*	*Yes, e.g. allows to colorize constriction in vessels*	*No*	*Yes, e.g. displays surrounding features*

Table 1: *Comparison of the 6 inspected transfer functions. Read the itemization in section 3 for a more detailed description of the rows.* * *Considering the real-time specification of the mapping: Input metrics → RGBA (If 'Yes', the mapping implicates its computation in shader). There might be some other options, that require an update of the pre-processing.* ** *See sections 3.1 - 3.6 for more detail.* *** *Based on the number of used metrics: ○○○ few features can be distinguished, ●●● many features can be distinguished.*

the graphics card). If worth mentioning, *other data structures* used are listed also.

- The ability to emphasize features. Since the TFs emphasize different kinds of *regions*, it is not possible to compare their quality in a fair way. Therefore we only provide a *ranking* that considers how many features can be distinguished, based on the number of used metrics. This ranking is vague and needs to be considered together with the *risk of a critical misinterpretation*, caused by inadequate display of the data. This risk is influenced by the provided automaticity and the number of used metrics. Automaticity substitutes the trial and error specification that reveals properties of the dataset and is therefore critical. The number of used metrics is influencing, because every additional metric enhances the domain of data values, hence the space where a feature can be overlooked. Once again, the ranking does not mean that the texture-based TF is the best one, but is capable of distinguishing more features.
- *Constraints to the datasets*, especially how much *noise* effects the TFs.
- Type and amount of user interaction as well as useful prior knowledge. The *type of interaction* describes the task that needs the main effort by the user. Specification on a single 2D diagram is preferred over specification on two 1D diagrams. The *duration* is indicated by the time needed to specify a TF with specialist knowledge. Since this value varies heavily depending on the use case and task, it can

rather be seen as a ranking among the TFs. If specialist knowledge is absent, specifying is done by trial and error only and time needed can be much higher. This *helpful knowledge* is subdivided into technical (i.e. understanding of the metric computation) and application (i.e. understanding of the dataset and its respective values) knowledge, hence it is possible to decide, if a certain TF is suitable for an explicit group of users.

- *Automated techniques* and *widgets* known that support the user, to cope with the complexity of the input metrics. If entries are not listed, it does not mean that they do not work with this TF, only that there is no such experience listed in publications. The image-based approach in [MAB*97] provides a number of possible renderings and is therefore usable for all TFs. Hence it is not mentioned explicitly.
- How much does a perfect segmentation benefit from the TFs, i.e. is it reasonable to use this TF additionally, if you already have a precise segmentation e.g. of the coronaries.

Since it is not possible to compare every TF, we compare a representative selection of types as explained earlier. Our argument for choosing the following 6 TFs is as follows. The 1D data-based TF serves as a reference. The 2D gradient-based TF leads to a quite common enhancement to distinguish surfaces. The 2D distance-based TF stands for TFs that are not based on the measured signal, but on additional information (in this case a segmentation). The 2D size-based TF provides a metric more intuitive than gradient, namely size. The texture-based TF is representative for higher-dimensional TFs, that take into account a combination of several metrics, meeting their complexity with some automaticity. The 2D curvature-based TF was chosen as representative of the group of TFs, mostly used for non-photorealistic rendering. Except for the distance-based TF, all TFs have been implemented in our volume visualization software *Volume Studio*, to evaluate their behavior. Wherever possible, measurements are adopted from the original papers, but are enhanced with own evaluations and experiences (e.g. error-proneness to noise, helpful prior knowledge, duration of specifying TF).

To compare the TFs visually, every TF description is supplemented by an explaining 2D Figure (Fig. 2–7). The Figures show four circular features, sharing one grey-level histogram, but are different in size, texture and background. Those areas of the four features, that can be distinguished by the individual TFs, are colorized in red and blue. The content of the figures is rather contrived, and is not supposed to compare the behavior of the TF types with real data. They serve for a simple visual explanation.

3.1. 1D data-based transfer function (1dTF)

The 1dTF maps data values directly to color and opacity. Hence it is not capable of differentiating between samples that share the same data value. Although they appear visually quite different, in Figure 2 the 1dTF can not distinguish the four features.
On the other hand, it is the TF that needs the lowest amount of computation power and memory. Furthermore, it provides a very high safety regarding errors that lead to a misinterpretation and needs little user interaction.

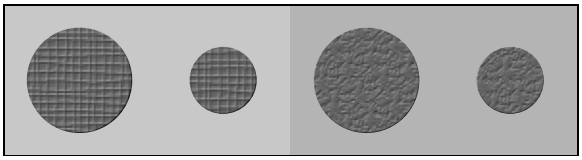

Figure 2: *The 1d data-based TF can not distinguish between the four circular features, since they have the same grey level. If trying to colorize one feature, all four would be colored.*

3.2. 2D gradient-based transfer function (gbTF)

In volumes, the gbTF is especially helpful to display surfaces of features [Lev88, KKH02]. It maps the data value and the gradient magnitude to color and opacity. Hence it is capable of differentiating between samples that share the same data, but not the same neighborhood. Thus, in Figure 3 it is possible to distinguish between the borders/surfaces of the left and the right features, that are placed on a different background. In Figure 8 (a) a human head is exposed with a gbTF. The gradient metric is highly error-prone to noise. On the other hand, gbTFs only marginally provide additional information (compared to the 1dTF) in very blurry volumes, like PET-scans.
The standard way of gradient computation needs 6 texture-look-ups. Since the gradient is needed for illumination anyway, this normally does not lead to a higher computation time. Of course the gradient can be precomputed, reducing the texture lookups to 2. This results in an additional volume needed on the graphics card and reduces the smoothness of the illumination.

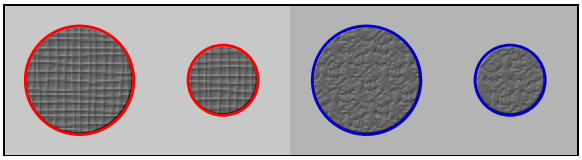

Figure 3: *The left and the right features are placed in a different environment. Since this leads to a different gradient at the feature borders (surfaces in 3D), the gradient-based TF can distinguish the borders of the left and right features.*

3.3. Curvature-based transfer function (cbTF)

In combination with the camera position, cbTFs are mainly used to display a contour around features. More generally, it

is possible to map data and curvature to any color and opacity, leading to ridge and valley visualization or emphasis of smallest bumpiness. In Figure 4 the borders/surfaces of the small and the big features can be distinguished, since they exhibit a different curvature. The effort to specify a cbTF depends on the usage. If it is used for contouring features, it is easy. To pointedly highlight bumpiness, its effort is comparable to the gbTF. A combination of contours and ridge and valley emphasis is shown in Figure 8 (b). Since many feature share the same curvature, it is rather difficult to emphasize a targeted feature only. Instead, some additional unintended emphasis will occur. Noise amplifies this behavior.
The computation of the curvature needs at least a 26-point neighborhood. A real-time GPU variant of the approach presented in [KWTM03] can be found in the GPU Gems 2 [PF05] or in [HSS*05] for the iso-surfaces. Hence the table has entries that correspond to this real-time implementation.

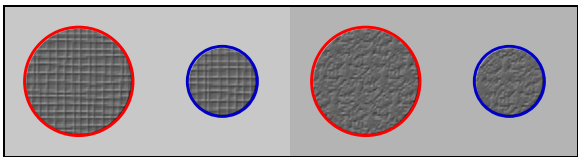

Figure 4: *The curvature-based TF is able to distinguish between the borders of the small and the big features, since they have a different curvature. With additional information about the camera position it can be used to contour the features in 3D space.*

3.4. Size-based transfer function (sbTF)

The sbTF maps data values and size to color and opacity. In Figure 5 the big and the small features can be distinguished. The size of a feature is quite complex to define and has been determined in very different ways. This analysis relies on the approach of [CM08], which uses the scale space for size detection. This approach benefits from a small error-proneness to noise. Furthermore, the quite complex preprocessing can be performed using the GPU, and is therefore very fast. Figure 8 (c) shows a size-based colorization of an aneurism.

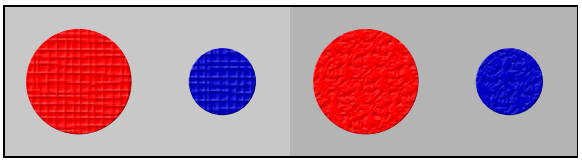

Figure 5: *The size-based TF is able to distinguish between the big and the small features.*

3.5. Texture-based transfer function (tbTF)

The tbTF massively differs from the other TFs in the number of metrics. It uses 20 texture metrics [CR08], six 1st order, seven 2nd order statistics and seven run-length matrices. Since it is too elaborate to map every of the 20 used texture metrics manually, the tbTF provides one full and three semi-automatic ways of specifying it. The fully automatic way uses clustering to recognize structures and only needs the number of structures as input. Since this automated technique is barely controllable, the ability to emphasize a determined feature is poor and the risk of a wrong result, leading to critical misinterpretation, high. That is why three further semi-automatic possibilities of parameterizing this TF are given. One rather manual, where single statistics can be weighted, a second one to specify two structures to be distinguished, and a third option, that tries to separate a single marked structure. The first one needs a very high understanding of the used metrics and is therefore difficult and time-consuming to use. Table 1 holds entries for the "automatic" and the "semi-automatic" options. Since noise makes texture analysis difficult, datasets have to be very clean for this method. However, in Figure 6 the tbTF is able to distinguish the left and the right feature due to their different texture. Figure 8 (d) shows a texture-based accentuation of the cerebellum (red) over the rest of the brain (green).
Computing the metrics and clustering is very expensive leading to high preprocessing times combined with very high memory consumption. Up to 20 volumes are needed, one for each statistical metric. Excepting the full automatic method, the memory amount and number of texture look-ups is the same during rendering. Two additional volumes are needed in pre-processing, one holding the Gray Level Co-occurrence Matrices (GLCM), one for the run-length matrices. The texture analysis not necessarily requires full volume resolution, if less precision is acceptable. That is why this TF is not limited to small datasets, but involves a trade-off between time and precision.

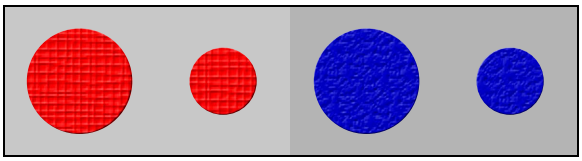

Figure 6: *The texture-based TF distinguishes the left and the right features by means of their different textures.*

3.6. Distance-based transfer function (dbTF)

The dbTF also differs from the other TFs, since it uses a previously segmented structure instead of the measured signals for an additional metric computation. This additional metric is the distance to the surface of the segmented structure. In Figure 7 the dbTF can colorize features in the neighborhood

of the reference structure differently from features further away. Figure 8 (e) shows a colorized neighborhood of a tumor.

In [TPD06] the preprocessing time needed is indicated with 20 seconds for a 256^3 volume, but it is easy to imagine heuristics or the computation on the GPU to cut processing time. For simple geometric reference forms like points and spheres, there is no need for pre-computation at all. The needed distance volume during rendering needs not to be the same resolution as the dataset, since distances are trilinearly interpolated, hence leading only to small inaccuracy in regions near a surface. The need for a segmentation makes this TFs rather unfeasible for time-dependent data.

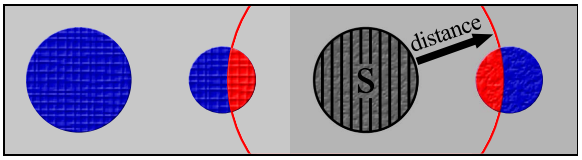

Figure 7: *The distance-based TF uses the distance to a given segmented object S, to provide additional information in the near environment of the feature.*

4. Conclusion

The described TF types differ in behavior and abilities: Some need technical expertise, some application-oriented expertise. Some emphasize areas, some surfaces. So it is not a single TF that is perfect for everything. Rather it depends on the boundary condition, such as the dataset, the user and the available time for a TF to be a perfect candidate for a particular field of application and task. Sometimes a trial and error approach is right, sometimes a more automated technique is needed.

Expertise is required to determine the most suitable TF type. By default the 1dTF is widely used, since it is a robust all-rounder with workable quality. This survey and the appended table give an overview of the most used and practicable TF types and assists in choosing the right TF for a specific goal. Once a TF is chosen, a first hint is given, to what widgets and automating techniques work fine in the specification of this TF, depending on wether a trial and error approach or a more targeted approach is desired. Furthermore, statements are applicable or at least adaptable to other TFs, since we provide a representative but widespread selection. The assessment was made with a medical view, but could be easily adapted to other fields of application. The table could also work as a starting point for a combination of TFs, to benefit from the advantages and eliminate the disadvantages.

Acknowledgements

We appreciate the helpful comments of our seminar group and therefore thank M. Bachmann, N. Bredenbals, M. Haupt, R. Petrlic, D. Riemer, A. Schwabe, F. Steffen and P. Wette. We also appreciate cooperation with Dr. W. Burchert, R. Weise and Drs. E. and H. Fricke of the Heart and Diabetes Center North-Rhine Westphalia, Germany.

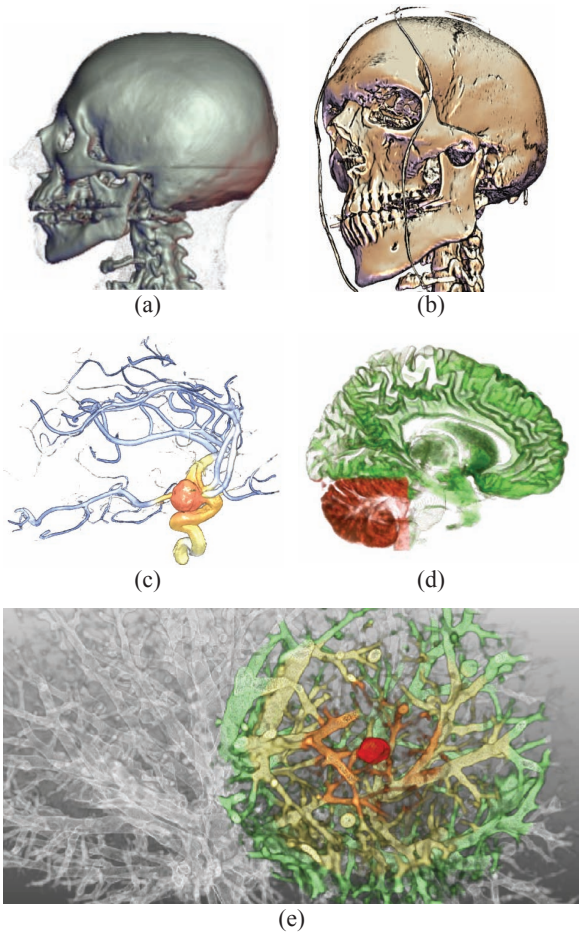

Figure 8: *Renderings made with the single TFs, courtesy of the respective authors. (a) Gradient-based TF [Kin99] showing surfaces of a human head. (b) Curvature-based TF [KWTM03] showing contours and ridge and valley emphasis. (c) Size-based TF [CM08] showing a differently colorized aneurism due to its size. (d) Texture-based TF [CR08] showing an emphasized cerebellum due to its different structure. (e) Distance-based TF [TPD06] showing an emphasized neighborhood.*

References

[CM08] CORREA C. D., MA K.-L.: Size-based transfer functions: A new volume exploration technique. *IEEE Trans. Vis. Comput. Graph. 14*, 6 (2008), 1380–1387.

[CR08] CABAN J. J., RHEINGANS P.: Texture-based transfer functions for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1364–1371.

[HKG00] HLADUVKA J., KÖNIG A., GRÖLLER E.: Curvature-based transfer functions for direct volume rendering. In *In Bianca Falcidieno, editor, Spring Conference on Computer Graphics 2000* (2000), pp. 58–65.

[HSS*05] HADWIGER M., SIGG C., SCHARSACH H., BÜHLER K., GROSS M. H.: Real-time ray-casting and advanced shading of discrete isosurfaces. *Comput. Graph. Forum (Proc. Eurographics) 24*, 3 (2005), 303–312.

[KD98] KINDLMANN G., DURKIN J. W.: Semi-automatic generation of transfer functions for direct volume rendering. In *VVS '98: Proceedings of the 1998 IEEE symposium on Volume visualization* (New York, NY, USA, 1998), ACM, pp. 79–86.

[Kin99] KINDLMANN G.: *Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering*. Master's thesis, Cornell University, January 1999.

[Kin02] KINDLMANN G.: Transfer functions in direct volume rendering: Design, interface, interaction. In *SIGGRAPH 2002 Course Notes* (2002).

[KKH01] KNISS J., KINDLMANN G., HANSEN C.: Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *VIS '01: Proceedings of the conference on Visualization '01* (Washington, DC, USA, 2001), IEEE Computer Society, pp. 255–262.

[KKH02] KNISS J., KINDLMANN G., HANSEN C.: Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (2002), 270–285.

[KWTM03] KINDLMANN G., WHITAKER R., TASDIZEN T., MOLLER T.: Curvature-based transfer functions for direct volume rendering: Methods and applications. In *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)* (Washington, DC, USA, 2003), IEEE Computer Society, p. 67.

[Lev88] LEVOY M.: Display of surfaces from volume data. *IEEE Computer Graphics and Applications 8*, 3 (1988), 29–37.

[MAB*97] MARKS J., ANDALMAN B., BEARDSLEY P. A., FREEMAN W., GIBSON S., HODGINS J., KANG T., MIRTICH B., PFISTER H., RUML W., RYALL K., SEIMS J., SHIEBER S.: Design galleries: a general approach to setting parameters for computer graphics and animation. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), ACM Press/Addison-Wesley Publishing Co., pp. 389–400.

[PF05] PHARR M., FERNANDO R.: *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation (Gpu Gems)*. Addison-Wesley Professional, 2005.

[PLB*01] PFISTER H., LORENSEN B., BAJAJ C., KINDLMANN G., SCHROEDER W., AVILA L., RAGHU K., MACHIRAJU R., LEE J.: The transfer function bake-off. *Computer Graphics and Applications, IEEE 21*, 3 (May/Jun 2001), 16–22.

[RBS05] RÖTTGER S., BAUER M., STAMMINGER M.: Spatialized transfer functions. In *EuroVis* (2005), pp. 271–278.

[TLM03] TZENG F.-Y., LUM E. B., MA K.-L.: A novel interface for higher-dimensional classification of volume data. In *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)* (Washington, DC, USA, 2003), IEEE Computer Society, p. 66.

[TPD06] TAPPENBECK A., PREIM B., DICKEN V.: Distance-based transfer function design: Specification methods and applications. In *SimVis* (2006), pp. 259–274.

[WK09a] WESARG S., KIRSCHNER M.: 3d visualization of medical image data employing 2d histograms. *International Conference in Visualisation 0* (2009), 153–158.

[WK09b] WESARG S., KIRSCHNER M.: Structure size enhanced histogram. In *Bildverarbeitung für die Medizin* (2009), pp. 16–20.

[ZEAD08] ZUKIC D., ELSNER A., AVDAGIC Z., DOMIK G.: Neural networks in 3d medical scan visualization. In *3IA 2008: International Conference on Computer Graphics and Artificial Intelligence* (New York, NY, USA, 2008), ACM Press/Addison-Wesley Publishing Co., pp. 183–190.