The Visual Computer

Exploiting the Hessian matrix for content-based retrieval of volume-data features

J. Hladůvka,
E. Gröller

Institute of Computer Graphics and Algorithms,
Vienna University of Technology,
Favoritenstraße 9–11/E186, 1040 Vienna, Austria
E-mail: {hladuvka,groeller}@cg.tuwien.ac.at

Published online: 14 May 2002

We propose an algorithm for content-based retrieval of representative subsets of volume data. Our technique is based on thresholding of the eigenvalues of the Hessian matrix. We compare our approach to feature detection based on the gradient magnitude and observe that our method allows the representation of volumes by a smaller amount of voxels. Practical applications of our method include fast volume display due to object-space oriented techniques, generation of preview data sets for web-based repositories, and the related progressive visualization over the network. For these applications, the size of the representative subset can be estimated automatically with respect to the bottleneck of the visualization system or a network bandwidth.

Key words: Volume visualization – Sparse data – Gradient – Hessian matrix – Eigensystem

Correspondence to: J. Hladůvka

The Visual Computer (2002) 18:207–217
Digital Object Identifier (DOI) 10.1007/s003710100141

1 Introduction

With the VolumePro board [1] one might have got an impression that *the technology* of volume graphics has already matured to become *a tool*. “The volume graphics today is where the surface graphics was fifteen years ago,” said David Nadeau at WSCG 2001 [2], however, and he emphasized two reasons: the lack of authoring tools which would help spread volume graphics and the limited display capabilities. The VolumePro board is able to display at 30 fps, just volumes of rather small resolution, i.e., 256^3. Without introducing blur, rendering such volumes can fill only moderately sized output images, i.e., up to 256×256 pixels. Constrained by a 21-inch monitor with a resolution of 1280×1024 pixels, a projection of a 256^3 volume is comparable to as if we were legally blind and a projection of a 1024^3 volume to as if we were visually impaired. In order to approach the “normal” vision determined by the density of cones at the fovea, we will need to render volumes of much higher resolution. Use of a 21-inch screen would require a projection of a 2400^3 volume. Considering a 180° visual field in virtual/augmented reality applications, the volume resolutions necessary to satisfy the human eye increase to $21\,600^3$ voxels.

The current technology is far from displaying such resolutions at interactive frame rates. The main advantage of volume graphics over surface graphics, i.e., the display of *all* volume elements in a data set, is also its main disadvantage – the size of data to be visualized limits the practical use. In his talk [2], Nadeau surveyed how to cope with this phenomenon and concluded a part of the keynote with the question “How about changing our data?”

In this work we give one of the possible answers to this question. We report on a newly developed technique aiming at a content-based retrieval of the most important regions from volume data. Our approach utilizes convolution, computation of eigensystems, and thresholding. It enables representation of volumes by the crucial features contained in much smaller subsets.

2 Motivation and related work

A representation of volume data by just a small subset of *content-carrying* voxels is desirable for and addressed by many applications.

Appropriately reorganized sparse volumes can be rendered using object-space display techniques at interactive frame rates [3, 4]. Saito [5] introduced nonrealistic previewing. Each voxel from a sparse subset of the volume is represented by a simple 3D entity, such as a point, a line, or a polar cross. A list of these entities is passed to a conventional rendering pipeline achieving real-time results. Splatting, introduced by Westover [6] and enhanced over the years [7], usually yields high-quality display results. Researchers further optimize [8] this technique to achieve interactive frame rates for volumes of moderate resolution. Recently, point-based visualization of large data sets became popular due to the work of Rusinkiewicz et al. [9], because of both simplicity and speed. The discussions on quality aspects of this approach were triggered by Pfister et al. [10].

Interactive volume visualization over the internet based on a client/server architecture profits from elaborated strategies for progressive data transmission. Here it is desirable for the content of a volume to be visually interpretable in the early stages of transmission to and visualization by a client. To achieve this, the server may start transmitting salient features earlier than the rest of the data.

Non-distributed visualization may benefit from storing a small, representative subset of the data to disk. Such a representation can be reused later for a quick preview.

There are many techniques aiming at identifying important subsets of a volume data set. Isosurface extraction algorithms belong to the most commonly used techniques for indirect volume rendering. The decisions on the iso-value(s) can be made interactively [11] or, after a previous analysis, semi-automatically [12] or automatically [13]. Saito [5] employed non-uniform stochastic Poisson sampling. The algorithm introduced by Mroz et al. [14] reduces the size of the data with respect to the applied visualization technique, i.e., maximum intensity projection (MIP). Gradient magnitude of the intensity can also be considered as a priority function. The voxels which mostly attract the human attention usually belong to the "strong" boundaries which exhibit high values of the gradient magnitude.

Similarly, our concept is based on a filtering technique. The quantity being filtered is the second-order derivative of the intensity profile. It is extracted from the eigensystem of the Hessian matrix.

3 Our proposal

In this section we introduce a framework for using eigenvalues of the Hessian matrix to identify a useful subset of a given volume. Firstly we recall the necessary definitions of the Hessian matrix and its eigensystem. Then we show how the eigenvalues are related to the Laplacian operator and demonstrate a separability which makes the eigenvalues suitable for thresholding of features. Based on this knowledge we propose a technique resulting in a sparse volume [15]. Finally we report on an extension [16] which assumes that the objects in the volume exhibit higher intensities than the background.

3.1 The Hessian matrix

A common approach to analyze the behaviour of intensity I in the neighborhood of point x_0 of a 2D/3D image is to consider the initial terms of the Taylor series expansion:

$$I(x_0+\Delta x) \approx I(x_0) + \Delta x^T \nabla I(x_0) + \frac{1}{2}\Delta x^{\mathrm{T}} \mathbf{H}(x_0)\Delta x, \tag{1}$$

where ∇I is the gradient vector and $\mathbf{H}$ denotes the Hessian matrix. The Hessian matrix comprises second partial derivatives of I and for a 3D image (volume) is defined as follows:

$$\mathbf{H} = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix}, \quad \text{where } I_{ab} = \frac{\partial^2 I}{\partial a \partial b}. \tag{2}$$

Expansion (1) plays a crucial role in filter design [17, 18] and, up to the second order, locally approximates *the structure of the image* [19]. Its components, both the gradient vector and the Hessian matrix, can also be used separately. The gradient vector is widely used as a normal to an implicitly defined isosurface. Its magnitude provides a tool for edge/boundary detection and in volume visualization can be used as the opacity modulation factor [20].

The Hessian matrix is used whenever a second derivative at a grid point x_0 and a direction Δx is requested. The last term of (1), $\Delta x^{\mathrm{T}}\mathbf{H}(x_0)\Delta x$, represents a quadratic form and yields the desired second derivative.

Applications of second-order derivative information in 3D imaging include boundary detection due to the Laplacian operator (see also Sect. 3.3), semi-automatic transfer-function setup [12], and adding visual cues to isosurfaces due to their curvature properties [21–23].

3.2 The eigenvalues of the Hessian matrix

If there is a nonzero vector **0***neqe* $\in E^3$ such that

$$\mathbf{H}\boldsymbol{e} = \lambda\, \boldsymbol{e} \tag{3}$$

for some scalar λ, then λ is the *eigenvalue* of **H** with the corresponding *eigenvector* $\vec{e}$. As **H** is a real-valued and *symmetric* matrix, its eigenvalues are real-valued, too. The number of nonzero eigenvalues is equal to the rank of **H**, i.e., at most three nonzero eigenvalues are available. In this work we assume the following arrangement of eigenvalues: $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Sato et al. [24] and Frangi et al. [19] independently employed the eigenvalues to design a filter for vessel enhancement in 3D medical digital images. Two years later Sato et al. [25] generalized the previously introduced concept to enhance tubular-, blob-, and sheet-like structures. The task addressed in their work was tissue classification. To achieve this they adopted a concept of a multi-dimensional transfer function which requires the user to set parameters in 5D space (I, ∇I, and three functions of the eigenvalues) or even, including segmentation information, in 6D space.

The technique presented in this paper requires much less user interaction. In the following we explain the theoretical background leading to a two-fold thresholding algorithm where the only task of the user is to specify two thresholds.

3.3 Relation of the eigenvalues to the Laplacian operator

Finding the eigenvalues and eigenvectors of the Hessian matrix is closely related to its decomposition

$$\mathbf{H} = \mathbf{PDP}^{-1}\ , \tag{4}$$

where **P** is a matrix the columns of which are **H**'s eigenvectors and **D** is a diagonal matrix having **H**'s eigenvalues on the diagonal. The right-hand side of (4) is a similarity transformation under which the *trace* is an invariant, i.e., $\mathrm{Tr}(\mathbf{PDP}^{-1}) = \mathrm{Tr}(\mathbf{D})$. Putting these facts together for a 3D image we find the following:

$$\begin{aligned}\lambda_1 + \lambda_2 + \lambda_3 =& \mathrm{Tr}(\mathbf{D}) = \mathrm{Tr}(\mathbf{PDP}^{-1})\\ =& \mathrm{Tr}(\mathbf{H}) = I_{xx} + I_{yy} + I_{zz}\\ =& L\ .\end{aligned} \tag{5}$$

The rightmost term denotes the Laplacian operator and (5) puts it into a relationship with the eigenvalues of the Hessian matrix.

The Laplacian operator and its variant, the Laplacian of Gaussian (LoG), is widely used as an isotropic edge detection filter [26]. The importance of edge information for *machine vision* is usually motivated from the observation that a discontinuity in image brightness can be assumed to correspond to a discontinuity in either depth, surface orientation, reflectance or illumination [27]. The importance of the Laplacian and LoG for *bioperception* has been emphasized by the work of Marr [28].

To use the Laplacian filter for boundary detection, the neighborhood of the inspected voxel must be checked for zero-crossings, i.e., whether or not the second derivatives change the sign. Obviously, this test is more complicated than first-order detection, which only requires thresholding of the gradient magnitude. On the other hand, there are two reasons which make second-order detection attractive. Firstly, only linear operators are involved in the computation. In contrast, the gradient magnitude is only obtained after squaring and adding first-order derivative operators, followed by a square root. Secondly, the result of the search for zero crossings at boundaries of the objects always results in closed contours. This is different to the first-order detection since thresholding of the gradient magnitude can brake contours if the boundaries exhibit varying lengths of gradient.

In the following we introduce a mechanism which utilizes both the advantages of second-order detection and the simplicity of thresholding.

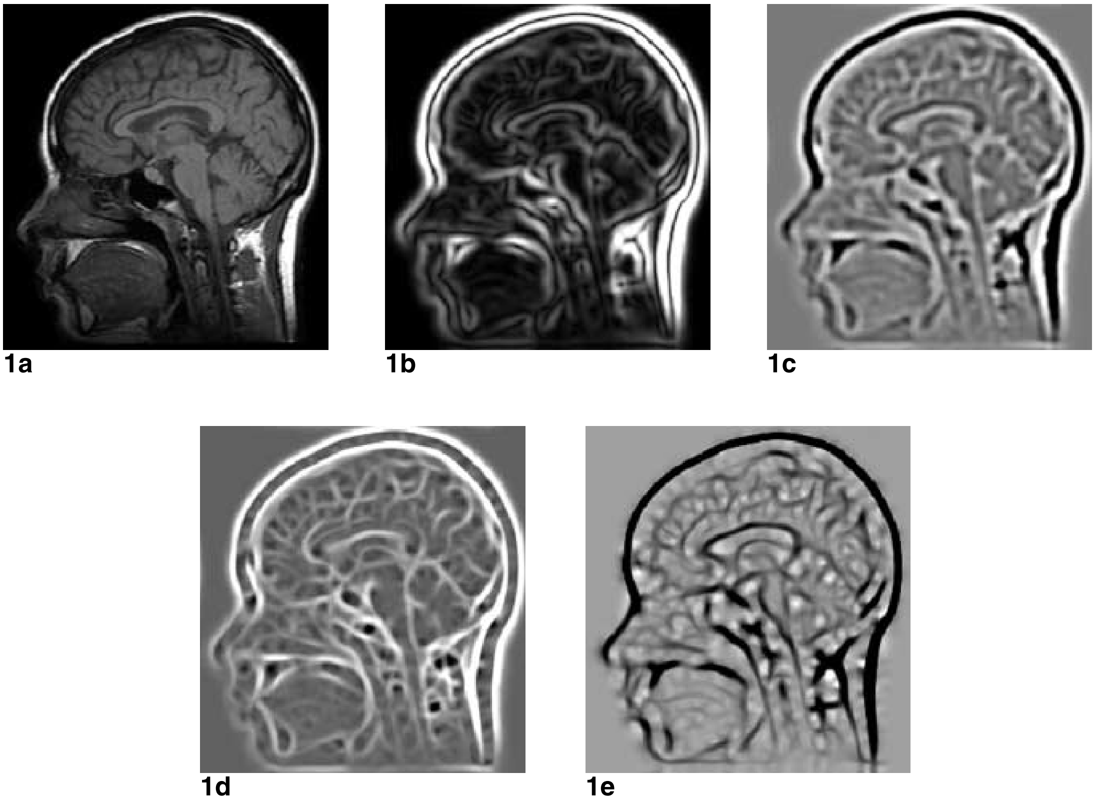
3.4 Thresholding of Hessian's eigenvalues

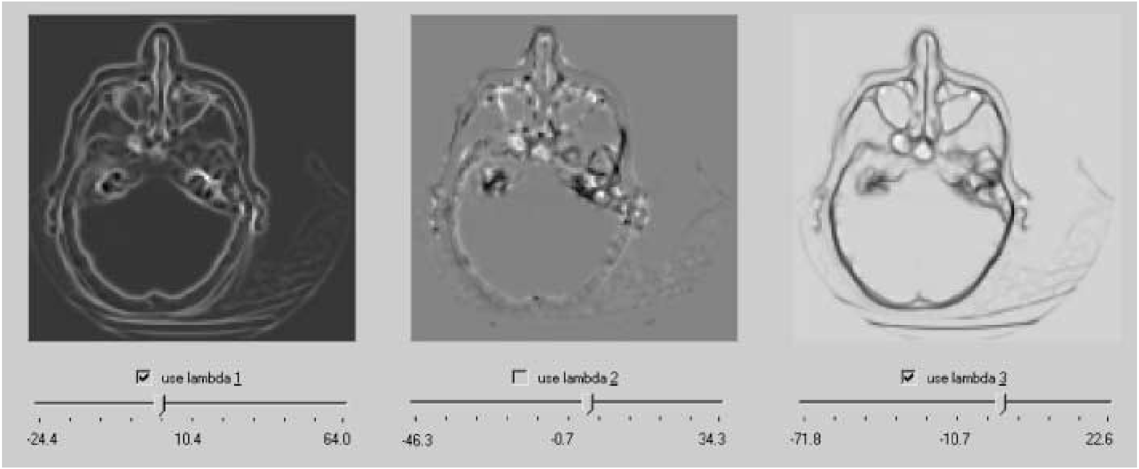
Given the eigenvalues of the Hessian matrix, (5) shows that it would be possible to combine them into the Laplacian operator. By experimenting with eigenvalue images, however, we have found that treating the eigenvalues separately rather than adding them is suitable for thresholding. Figure 1 demonstrates this property on a 2D example.

For an extraction of features from a 2D image we propose a two-fold thresholding as follows:

$$I_{\mathrm{new}}[p] = \begin{cases} I[p] & \text{if } \lambda_1[p] \geq T_1 \vee \lambda_2[p] \leq T_2\ ,\\ 0 & \text{otherwise}\ ,\end{cases} \tag{6}$$

where $p = [x, y]$ denotes the coordinates of a pixel. For a 3D image there are three eigenvalues, $\lambda_1 \geq \lambda_2 \geq \lambda_3$, for each voxel. Figure 2 shows axial slices of eigenvalue volumes computed from a CT head





☑ use lambda 1

-24.4 10.4 64.0

□ use lambda 2

-46.3 -0.7 34.3

☑ use lambda 3

-71.8 -10.7 22.6

2

Fig. 1a–e. An example of an MRI image. **a** original image I; **b** gradient magnitude $\|\nabla I\|$; **c** response to the Laplacian operator; **d** λ_1 image; and **e** λ_2 image. The Laplacian image corresponds to the sum of eigenvalue images, which can be thresholded for feature detection

Fig. 2. Part of an interface and demonstration of suitability for thresholding of the eigenvalues λ_1 and λ_3. The image corresponding to eigenvalue λ_2 lacks contrast information and is therefore excluded from thresholding

data set. Since, in our experience, images corresponding to λ_2 lack good contrast, we propose two-fold (instead of three-fold) thresholding, taking into account only the outer eigenvalues, λ_1 and λ_3:

$$V_{\text{new}}[v] = \begin{cases} V[v] & \text{if } \lambda_1[v] \geq T_1 \vee \lambda_3[v] \leq T_3 \ , \\ 0 & \text{otherwise} \ , \end{cases} \tag{7}$$

where $v = [x, y, z]$ denotes the coordinates of a voxel. The only task of the user is to specify the two thresholds T_1 and T_3 (Fig. 2).

3.5 Experimental results

Table 1 and Fig. 3 indicate that (7) allows volumes to be represented, for visualization purposes, by approximately 10% of the voxels. The thresholds T_1 and T_3 allow the user to interactively control the trade-off between the quality of the display and the amount of the displayed information.

Equation (7) defines a variant of a second-order boundary detector which is important for bioperception (see Sect. 3.3). This is mostly noticeable by comparing the rendered images of the engine block data sets (top row of Fig. 3). In the sparse volume, areas corresponding to boundaries are emphasized and provide the observer with better topological information on the data set.

All tested volumes have been quantized to 256 gray levels. To compare the visual appearance, the volumes have been displayed by a direct volume rendering (DVR) algorithm implemented in the VolumePro architecture [1].

3.6 Further reduction of the subsets

The previously introduced concept can be further extended [16] by assuming that the objects are of a higher intensity than the background. In this case a second-order operator responds with negative and positive values at the inner and outer sides of a boundary, respectively. This is in contrast with gradient-based edge detection, which yields an equal response on both sides of a boundary. We exploited this fact to represent the objects' boundaries only by their internal side. Compared to the gradient method, such a selection requires a smaller amount of voxels for boundary representation.

With a gradient-based approach, the magnitude of the gradient, i.e., the maximal first derivative at the inspected point, is taken as a measurement of the boundary strength (importance) and is passed to the thresholding step.

For the same purpose, we propose the use of the minimal second-order derivative at the inspected point. From the definition of the eigenvalues, $\mathbf{H}\boldsymbol{e}_i = \lambda_i \boldsymbol{e}_i$, it follows that the eigenvalues λ_i give the second derivatives in the direction of the eigenvectors $\boldsymbol{e}_i$: $\lambda_i = \boldsymbol{e}_i^{\mathrm{T}} \mathbf{H} \boldsymbol{e}_i$. Since $\mathbf{H}$ in this context represents a quadratic form, computing the smallest eigenvalue directly yields the minimal second-order derivative at the point x_0. This gives a measurement of the boundary importance – the lower the λ_3, the more important the boundary.

To compare volume representation by subsets obtained by thresholding the quantities λ_3 and $\|\nabla\|$, we generated sparse volumes where the intensity of voxels not present in the subset is set to zero. These volumes were rendered by the OpenSplat package [29]. In the following we refer to the sparse volumes comprising $p\%$ of the original voxels due to thresholding of λ_3 as $\Lambda(p\%)$ and due to thresholding of $\|\nabla\|$ as $\Gamma(p\%)$. We compare the representations consisting of 1, 3, 5, and 7% of the original voxels:

Lobster (Fig. 4): Representation by $\Lambda(1\%)$ subset provides a better idea about the data set than the

Table 1. Results for some typical volume data sets. Columns from left to right: name of the data set, its resolution, number of voxels in KB, threshold values from (7) used in the generation of a sparse volume, number of nonzero voxels in KB in the sparse volume, relative size to original volume, and a reference to Fig. 3

	Original volume		Sparse volume				Fig. 3
Data set	Resolution	KB	T_1	T_3	KB	%	row
Engine block	256 × 256 × 110	7040	21.953	−23.381	658	9.36	1
CT head	128 × 128 × 113	1808	16.981	−19.182	183	10.12	2
MRI head	256 × 256 × 109	6976	7.822	−6.364	871	12.48	3

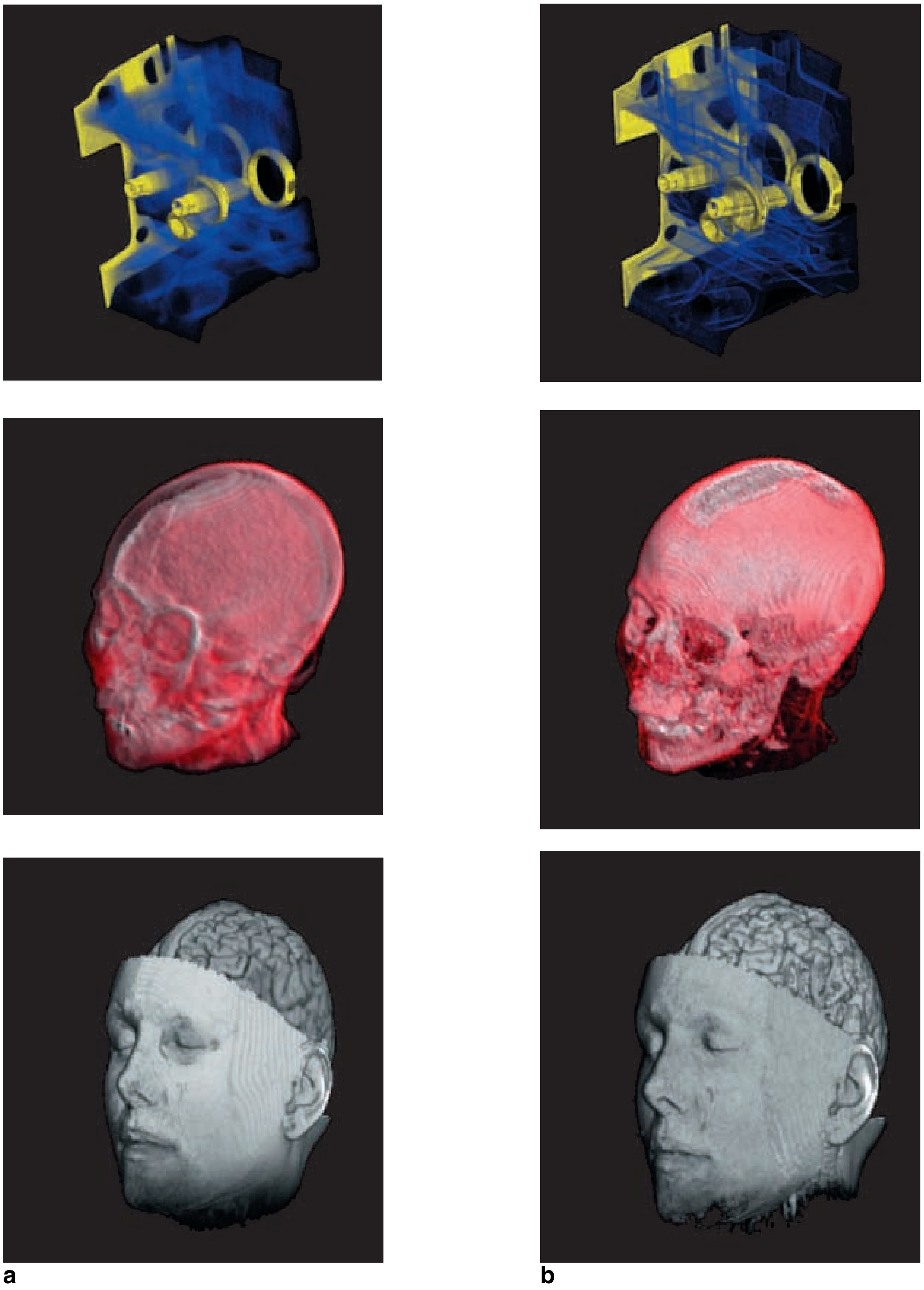


Fig. 3a,b. Raycasting [1] of **a** the engine block, CT head, and MRI head data sets and **b** the corresponding sparse volumes due to (7) with threshold parameters set as in Table 1. The engine block subset consists of 9.36% voxels of the original data set, the CT head subset of 10.12% voxels, and the MRI head subset of 12.48% voxels

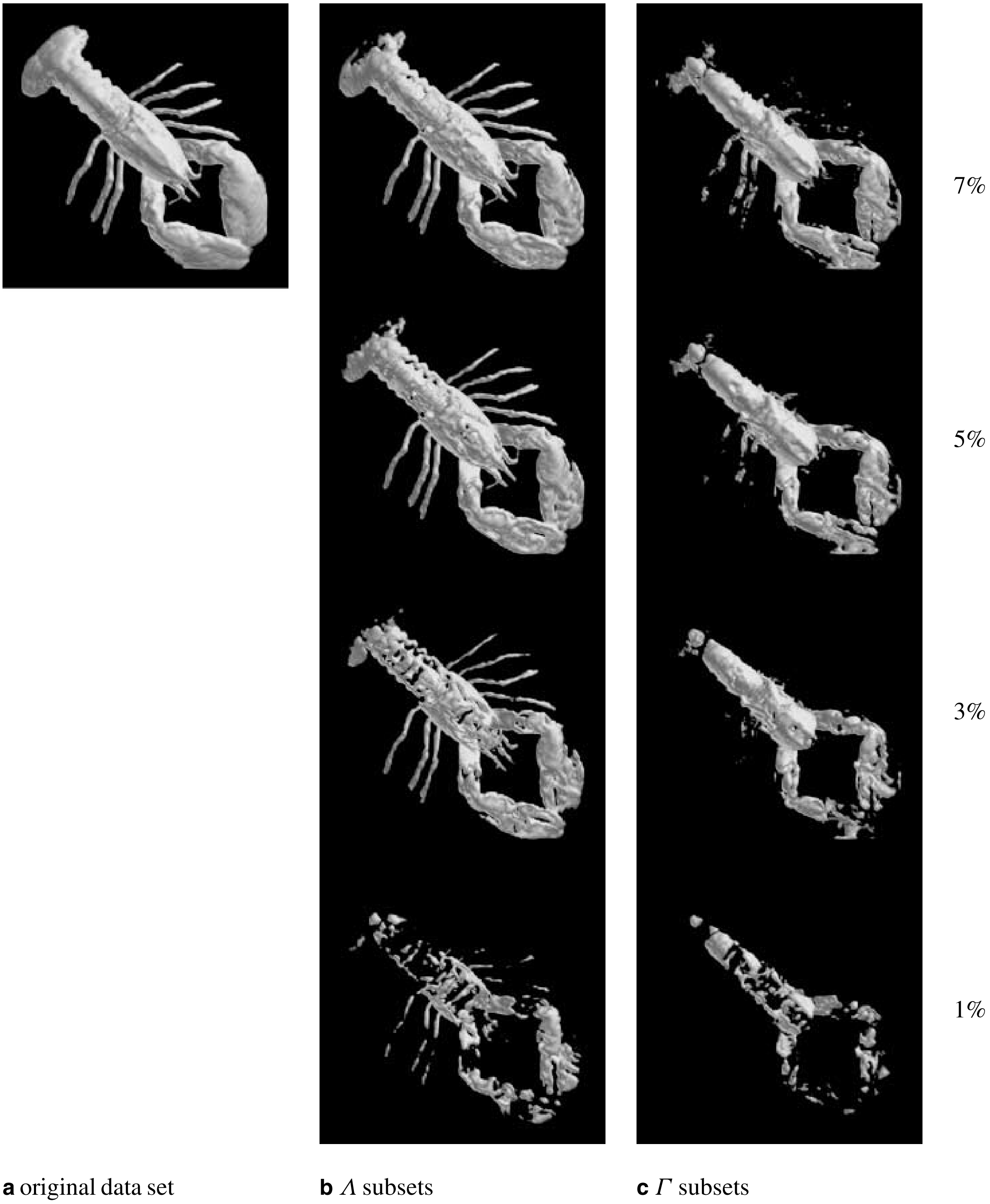
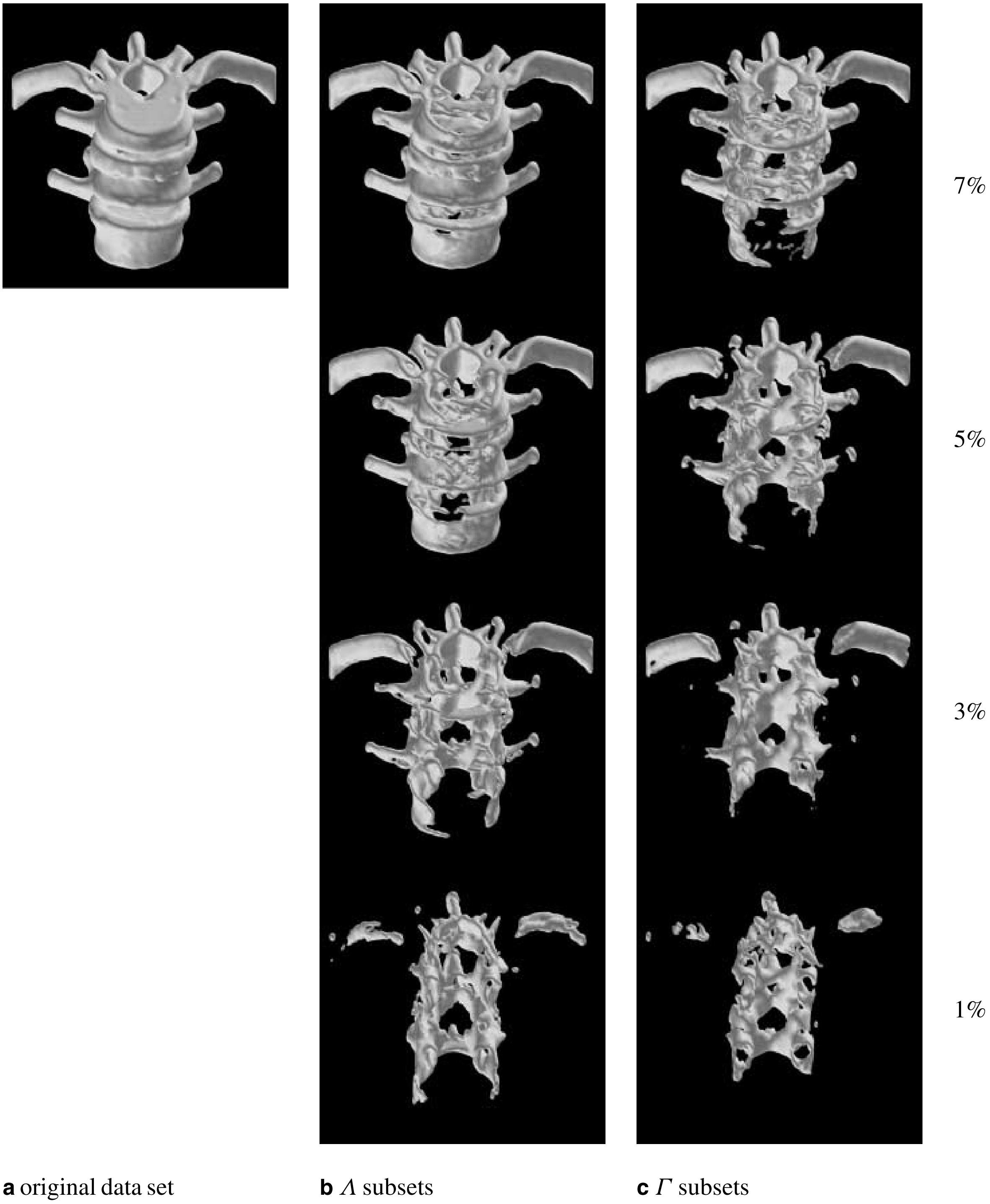


Fig. 4a–c. Splatting [29] of the lobster data set **a** and its representations due to the thresholding of **b** λ_3 and **c** $\|\nabla\|$. From top to bottom the subsets comprise 7%, 5%, 3%, and 1% of the voxels of the original data set



a original data set **b** Λ subsets **c** Γ subsets

Fig. 5a–c. Splatting [29] of the vertebra data set **a** and its representations due to the thresholding of **b** λ_3 and **c** $\|\nabla\|$. From top to bottom the subsets comprise 7%, 5%, 3%, and 1% of the voxels of the original data set

same amount of voxels in the representation by $\Gamma(1\%)$. The legs of the lobster are visible and easily recognizable in the $\Lambda(3\%)$ subset, but they just start to appear in the $\Gamma(7\%)$ subset. Moreover, the $\Lambda(7\%)$ is already close to describing the entire topology of the data set.

Vertebra (Fig. 5): Neither of the 1% representations provide enough information, though there is more content visible in the $\Lambda(1\%)$ subset. At 3, 5, and 7% we observe that the contours in the $\Lambda(\cdot)$ subsets close much faster than in the corresponding $\Gamma(\cdot)$ subsets. Moreover, we estimate that the $\Lambda(1\%)$, $\Lambda(3\%)$, and $\Lambda(5\%)$ subsets provide approximately the same level of information as the $\Gamma(3\%)$, $\Gamma(5\%)$, and $\Gamma(7\%)$ subsets, respectively.

4 Implementation and complexity

4.1 Hessian matrix versus gradient vector

Computation of both the gradient vector and the Hessian matrix at grid points involves an approximation of the first and the second partial derivatives, respectively. For this task, convolution of the data with kernels designed for a particular derivative in a specific direction is employed.

For the first derivatives, kernels of size up to three are usually found in textbooks: Roberts, Prewitt and Sobel filters are feasible for fast computation.

The Hessian matrix requires an estimation of second-order derivatives which is, especially for small kernels, much more sensitive to noise. The usual practice is to pre-smooth the input data with a Gaussian filter. Due to the associativity of convolution, the smoothing step and the differentiation can be combined, resulting in a convolution of the data with a derivative of the Gaussian filter of a bigger size. To remain consistent in the comparison of both the quality of results and the computational costs, we used filters of the same size for both first and second derivatives. Using the Gaussian filter requires that its size k is proportional to the standard deviation, so the kernels usually involved are 5, 7, or 9 voxels wide. Convolution with even moderately sized kernels is usually a computationally expensive process. To speed it up, hardware features can be used on specific platforms [30, 31]. For software implementation, the separability of the Gaussian derivative kernels can be exploited:

$$\left(\overbrace{\frac{\partial^o}{\partial x^a \partial y^b \partial z^c} G_\sigma(x, y, z)}^{k\times k\times k}\right) \otimes I$$
$$= \overbrace{\frac{\mathrm{d}^a}{\mathrm{d}x^a} G_\sigma(x)}^{k\times 1\times 1} \otimes \left(\overbrace{\frac{\mathrm{d}^b}{\mathrm{d}y^b} G_\sigma(y)}^{1\times k\times 1} \otimes \left(\overbrace{\frac{\mathrm{d}^c}{\mathrm{d}z^c} G_\sigma(z)}^{1\times 1\times k} \otimes I \right)\right), \tag{8}$$

where non-negative integers $a+b+c = o \in \{1, 2\}$ determine the order of differentiation and σ is the standard deviation of the Gaussian filter $G_\sigma(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right) / \sqrt{2\pi}\sigma$. Decomposition (8) reduces the time complexity $O(k^3)$ required by convolution with one 3D kernel to $O(3k)$ required by convolutions with three 1D kernels.

4.2 Eigenvalues of the Hessian versus gradient magnitude

While computing the gradient magnitude by the Euclidean norm requires three multiplications, two additions and one square root, the computation of eigenvalues of the Hessian matrix is more complex. The explicit formula would require solving cubic polynomials. In our implementation we used a numerical technique – the fast converging Jacobi's method as recommended by Press et al. [32] for real-valued, symmetric matrices.

5 Concluding remarks

We propose an easy-to-use framework for exploiting eigenvalues of the Hessian matrix to represent volume data by small subsets. We recall the relation of eigenvalues to the Laplacian operator, show the suitability of thresholding eigenvalue volumes, and define a two-fold threshold operation to generate sparse data sets.

For data where it can be assumed that objects exhibit higher intensities than background, we modify the framework taking into account only the smallest eigenvalue. This results in a further reduction of the representative subsets by selecting just voxels at the interior side of object boundaries [16].

We evaluate our method with several data sets from different modalities. We compare our approach to feature-detection based on thresholding of the gradient magnitude. We find that for the same level of perception our method allows data sets to be represented by reasonably smaller subsets. The presented method allows the volume data to be represented by approximately 6% subsets.

The possible applications of such a compact representation are, e.g., fast rendering due to object-space display techniques, progressive transmission over the Internet and the generation of preview data sets. The drawback of our method is a higher computational cost. Computation of the Hessian's eigenvalues is approximately 1.87 times more expensive than the computation of the gradient magnitude [33, 34].

Acknowledgements. The work presented in this publication has been funded by the ADAPT project is supported by Tiani Medgraph (`http://www.tiani.com/`), Vienna, and the Forschungsförderungsfonds für die gewerbliche Wirtschaft (FFF), Austria. A part of the work has also been supported by the BandViz project (`http://bandviz.cg.tuwien.ac.at/`), which is financed by Fonds zur Förderung der wissenschaftlichen Forschung (FWF) under project number 12 811.

References

1. Pfister H, Hardenbergh J, Knittel J, Lauer H, Seiler L (1999) The VolumePro real-time ray-casting system. SIGGRAPH '99 Comput Graph Proc 33:251–260
2. Nadeau DR (2001) Volume visualization: technology to tools. Keynote speech at 9th International Conference in Central Europe on Computer Graphics, Visualization, and Computer Vision (WSCG 2001)
3. Mroz L (2001) Real-Time Volume Visualization on Low-End Hardware. PhD thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology
4. Csébfalvi B (2001) Interactive Volume-Rendering Techniques for Medical Data Visualization. PhD thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology
5. Saito T (1994) Real-time previewing for volume visualization. Symp Vol Visual pp 79–106
6. Westover L (1990) Footprint evaluation for volume rendering. SIGGRAPH '90 Comput Graph Proc 24:367–376
7. Mueller K, Möller T, Crawfis R (1999) Splatting without the blur. Proc IEEE Visual 10:363–370
8. Kilthau S, Möller T (2001) Splatting optimizations. Technical Report TR 2001-02, School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
9. Rusinkiewicz S, Levoy M (2000) QSplat: A multiresolution point rendering system for large meshes. SIGGRAPH 2000 Comput Graph Proc 34:343–352
10. Pfister H, Zwicker M, van Baar J, Gross M (2000) Surfels: Surface elements as rendering primitives. SIGGRAPH 2000 Comput Graph Proc 34:335–342
11. Lorensen WE, Cline HE (1987) Marching cubes: A high resolution 3D surface construction algorithm. SIGGRAPH '87 Comput Graph Proc 21:163–169
12. Kindlmann G, Durkin JW (1998) Semi-automatic generation of transfer functions for direct volume rendering. In: Proc. IEEE Visualization 6:79–86
13. Gerstner T (2001) Fast multiresolution extraction of multiple transparent isosurfaces. In: Ebert D, Favre JM, Peikert R (eds) Data Visualization 2001, Proceedings of the Joint Eurographics – IEEE TCVG Symposium on Visualization, Springer pp 35–44, 336
14. Mroz L, Hauser H, Gröller E (2000) Interactive high-quality maximum intensity projection. Comput Graph Forum 19(3):341–350
15. Hladůvka J, König A, Gröller E (2001) Exploiting eigenvalues of the Hessian matrix for volume decimation. In: 9th International Conference in Central Europe on Computer Graphics, Visualization, and Computer Vision (WSCG 2001), Skala V (ed) pp 124–129
16. Hladůvka J, König A, Gröller E (2001) Salient representation of volume data. In: Data Visualization 2001, Proceedings of the Joint Eurographics – IEEE TCVG Symposium on Visualization, Ebert D, Favre JM, Peikert R (eds) pp 203–211, 351
17. Möller T, Machiraju R, Müller K, Yagel R (1997) Evaluation and Design of Filters Using a Taylor Series Expansion. IEEE Trans Visual Comput Graph 3(2):184–199
18. Theußl T, Hauser H, Gröller E (2000) Mastering windows: Improving reconstruction. Proc. IEEE Vol Visual 7:101–108
19. Frangi AF, Niessen WJ, Vincken KL, Viergever MA (1998) Multiscale vessel enhancement filtering. Lect. Notes Comput Sci 1496:130–137
20. Levoy M (1988) Display of surfaces from volume data. IEEE Comput Graph Appl 8(3):29–37
21. Interrante V, Fuchs H, Pizer S (1996) Illustrating transparent surfaces with curvature-directed strokes. Proc. IEEE Visualization 7:211–218
22. Interrante V (1997) Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. SIGGRAPH '97 Comput Graph Proc 31:109–116
23. Hladůvka J, König A, Gröller E (2000) Curvature-based transfer functions for direct volume rendering. In: Spring Conference on Computer Graphics 2000 (SCCG 2000), Vol. 16, Falcidieno B (ed) pp 58–65
24. Sato Y, Nakajima S, Shiraga N, Atsumi H, Yoshida S, Koller T, Gerig G, Kikinis R (1998) 3D multi–scale line filter for segmentation and visualization of curvilinear structures in medical images. Med Image Anal 2(2):143–168
25. Sato Y, Westin C-F, Bhalerao A, Nakajima S, Shiraga N, Tamura S, Kikinis R (2000) Tissue classification based on 3D local intensity structures for volume rendering. IEEE Trans Visual Comput Graph 6(2):160–180
26. Jähne B (1997) Digital Image Processing, Chap, 10.10: Laplace–Based Edge Detection, 4th edn. Springer, Berlin Heidelberg, pp 336–339
27. Lindeberg T (1996) Edge detection and ridge detection with automatic scale selection. Proc. IEEE Comput Vision Patt Recog. IEEE Comput Soc Press, pp 465–470
28. Marr D (1982) Vision. Freeman, New York
29. Huang J, Mueller K, Shareef N, Crawfis R (2000) OpenSplat: Software package for optimized splatting on rectilinear grids, ver. 2.0.2. http://www.cis.ohio-state.edu/graphics/research/FastSplats/

30. Hopf M, Ertl T (1999) Accelerating 3D convolution using graphics hardware. Proc IEEE Visual 10:471–474
31. Intel Corporation (2000) IPL – Intel Image Processing Library, v 2.5
32. Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical Recipes in C, Chap. 11: Eigensystems, 2nd edn. Cambridge University Press, Cambridge, pp 456–469
33. Hladůvka J (2001) Derivatives and Eigensystems for volume-data analysis and visualization. Institute of Computer Graphics and Algorithms, Vienna University of Technology, December 2001
http://www.cg.tuwien.ac.at/research/theses/hladuvka/
34. Hladůvka J, Gröller E (2002) Smallest 2nd-order derivatives for efficient volume-data representation. Comput Graph 26(2)

JIŘÍ HLADŮVKA received his MS in mathematics from Comenius University, Bratislava, in 1996. Since 1998, he has been a computer science doctoral student at Vienna University of Technology. In 2001 he finished his Ph.D. thesis on volume analysis [33]. His research interests include computer graphics, volume visualization and image processing.

EDUARD GRÖLLER is an associate professor at the Institute of Computer Graphics and Algorithms, Vienna University of Technology. In 1993 he received his Ph.D. from the same university. His research interests include computer graphics, flow visualization, and medical visualization. He is heading the visualization group at the Institute of Computer Graphics and Algorithms. He is a member of the IEEE Computer Society, the ACM (Association of Computing Machinery), the GI (Gesellschaft für Informatik), and the OCG (Austrian Computer Society).