Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization (2002)
D. Ebert, P. Brunet, I. Navazo (Editors)

Feature-Preserving Volume Filtering

László Neumann[†], Balázs Csébfalvi[‡], Ivan Viola[§], Matej Mlejnek[¶], and Eduard Gröller[||]

Vienna University of Technology, Austria, http://www.cg.tuwien.ac.at/home/

Abstract

In this paper a feature-preserving volume filtering method is presented. The basic idea is to minimize a three-component global error function penalizing the density and gradient errors and the curvature of the unknown filtered function. The optimization problem leads to a large linear equation system defined by a sparse coefficient matrix. We will show that such an equation system can be efficiently solved in frequency domain using fast Fourier transformation (FFT). For the sake of clarity, first we illustrate our method on a 2D example which is a dedithering problem. Afterwards the 3D extension is discussed in detail since we propose our method mainly for volume filtering. We will show that the 3D version can be efficiently used for elimination of the typical staircase artifacts of direct volume rendering without losing fine details. Unlike local filtering techniques, our novel approach ensures a global smoothing effect. Previous global 3D methods are restricted to binary volumes or segmented iso-surfaces and they are based on area minimization of one single reconstructed surface. In contrast, our method is a general volume-filtering technique, implicitly smoothing all the iso-surfaces at the same time. Although the strength of the presented algorithm is demonstrated on a specific 2D and a specific 3D application, it is considered as a general mathematical tool for processing images and volumes.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Feature-preserving smoothing, derivative and gradient estimation, direct volume rendering, antialiasing, noise filtering.

1. Introduction

There are several research areas, like noise filtering of signals, image processing, or direct volume rendering, where the input data is available as a continuous function sampled at regular or irregular grid points. It is a recurring problem, how to reconstruct the most important features of the original signal from the sampled values. For instance, after discretization the exact derivatives are not available anymore, therefore they have to be estimated from the samples.

The traditional approach is convolution-based filtering. The support of the filter kernels used in image-processing or volume-visualization practice is usually limited to a narrow neighborhood. An ideal but computationally impractical reconstruction however would require convolution filters with infinite support, like the *Sinc* filter for function reconstruction. A consequence of the local support is for example the staircase aliasing in direct volume rendering. Even if some low-pass filtering is used it remedies the problem only locally since the voxels far away from a given voxel do not have an effect on its estimated gradient. The global influence can be ensured using iterative methods but they are rather time-consuming and restricted to binary volumes[11, 12].

Our goal is to develop a global smoothing method, where each sample has an influence on all the others, but samples far away from each other interact only very slightly. We want to obtain smooth homogeneous regions but also achieve a preservation of important features. Sharp edges of an image or well defined iso-surfaces in a volumetric data set have to be retained.

First our method is illustrated on a 2D example which is a *dedithering* problem. Afterwards we discuss the 3D extension and its major application field which is *feature-preserving volume filtering*. In Section 2 we overview the previous work that has been done in 3D feature reconstruc-

† lneumann@cg.tuwien.ac.at
‡ csebfalvi@cg.tuwien.ac.at
§ e9726070@student.tuwien.ac.at
¶ e9726073@stud3.tuwien.ac.at
|| groeller@cg.tuwien.ac.at

tion. Section 3 presents in detail the mathematical background of our new method. Section 4 discusses the 2D case and its application in image processing. Section 5 describes the extension to 3D volumes and the adaptation for direct volume rendering of binary and gray-scale data sets. In Section 6 the contribution of this paper is summarized.

2. Previous work

Function and derivative reconstruction from sampled values is a fundamental problem in computer graphics. One research direction is *interpolation oriented* assuming that accurate samples are available. The *Sinc* and *Cosc* functions are considered as ideal interpolation and derivative filters respectively [8, 9]. For a practical use of the *Sinc* filter windowing is suggested [10]. Möller et al. [1] give a survey of existing digital derivative filters and compare their accuracy analytically. These derivative reconstruction techniques based on windowing are local methods as for practical reasons only a limited number of neighboring samples are taken into account.

Another approach for derivative reconstruction is *approximation oriented*. Here it is assumed that the sampled function is noisy, which is typical when some real physical properties are measured. The basic idea is to estimate the inclination or the normal from a larger neighborhood. In order to reduce staircase aliasing several methods were proposed for normal computation especially in binary volumes [2, 3, 4]. *Contextual shading* techniques try to fit a locally approximating plane [5] or a biquadratic function [6, 7] to the set of points that belong to the same iso-surface. These methods are time-consuming and limited to a certain neighborhood. Bryant and Krumvieda [5] solve a set of linear equations by Gaussian elimination in order to obtain an approximate tangent plane at a given surface point. Webber's technique [6, 7] is similar to [5]. In a 26-neighborhood the surface is approximated by a biquadratic function producing accurate results for objects with C^1 continuous faces. Neumann et al. [16] linearly approximate the density function itself using a 3D regression hyperplane. Their approach leads to a computationally efficient convolution operation. The common drawback of these approximation techniques is the local influence of the neighboring samples. It is rather easy to define cases, where staircase aliasing still appears in the generated image even if some more sophisticated local gradient estimation is applied.

A rather new research direction is based on distance-transformation methods [11, 14]. Sramek [14] proposes a voxelization method, where the generated volume contains densities proportional to the nearest distance from an analytical surface. Smooth distance maps created also from binary volumes [13] can be exploited in gradient estimation. Gibson [11] uses an iterative method based on this idea. Her "elastic surface net" creates a globally smooth surface model from binary segmented volumes. Staircase artifacts can be eliminated also using shape-based interpolation calculating smooth 2D distance maps [15]. The main disadvantage of distance transformation is its limitation to binary segmented volumes or to iso-surface-oriented applications.

Whitaker's [12] approach is similar to Gibson's [11] method but an iteration is performed directly on the volume without generating an intermediate triangular mesh. His technique is also restricted to binary data sets.

In contrast, our method is a general tool for filtering binary and gray scale volumes, making all the iso-surfaces smoother at the same time. Unlike convolution-based filtering, the smoothing effect is global due to a global curvature minimization. Feature preservation is the main characteristic of our novel filtering approach. By globally penalizing large gradient deviations important features and fine details like edges or iso-surfaces are preserved.

3. The new filtering algorithm

The input data is given as a noisy n-dimensional function f sampled at regular grid points. We want to generate a filtered function $\tilde{f}$ with reduced noise and preserved features. The discrete samples are denoted by f_i and $\tilde{f}_i$ respectively. We assume that the original function f contains continuous and smooth features as well as discontinuities. Our method is based on a quadratic penalty function E. Given N samples f_i we are looking for N unknown samples $\tilde{f}_i$ which minimize the penalty function. E is defined so that feature preservation and smoothing is simultaneously possible. For reasons of simplicity we illustrate the method first for the 1D case. Penalty function E consists of the following three components summed over every sample point i:

- difference squared between filtered value $\tilde{f}_i$ and original value f_i. This term ensures that the original values f_i are closely approximated by the filtered values $\tilde{f}_i$.
- difference squared between the gradients of the filtered value $\tilde{f}_i$ and the original value f_i. This term ensures feature preservation which means gradient preservation. In areas of high gradient magnitude the gradient of filtered function $\tilde{f}$ must closely follow the gradient of f. Otherwise a large unwanted contribution to penalty function E results. Filtered function $\tilde{f}$ will be a smooth function therefore central differences $((\tilde{f}_{i+1} - \tilde{f}_{i-1})/2)$ are sufficient to approximate the gradient of $\tilde{f}$. This is not the case for data values f_i which might result from a binary or noisy function f. Using central differences to estimate the gradient of function f, would induce $\tilde{f}$ to follow staircase or noise artifacts in f instead of approximating the true underlying gradient of f. The gradient at f_i is therefore approximated with g_i which is calculated according to a more sophisticated gradient-estimation scheme (Appendix 8.1.)[16]. The derivative function g is based on linear regression and contains only locally-reduced staircase artifacts. Additionally the gradient differences between $\tilde{f}_i$

and f_i are multiplied by a weighting factor G which determines the impact of feature preservation on the penalty function E.

- the squared curvature of the filtered function $\tilde{f}$. To achieve smoothing we are looking for a function $\tilde{f}$ with small overall curvature. The curvature at point i is approximated by the difference between the gradient at point i (approximated by $\tilde{f}_{i+1} - \tilde{f}_i$) and the gradient at point $i-1$ (approximated by $\tilde{f}_i - \tilde{f}_{i-1}$). This term is multiplied by a weighting factor S which determines the importance of smoothing.

In the 1D case the penalty function E therefore has the following structure (the second term is multiplied by 4 in order to simplify the further mathematical derivation):

$$E = \sum_i [(\tilde{f}_i - f_i)^2 + \tag{1}$$

$$4 \cdot G \cdot [(\tilde{f}_{i+1} - \tilde{f}_{i-1})/2 - g_i]^2 +$$

$$S \cdot (\tilde{f}_{i+1} + \tilde{f}_{i-1} - 2\tilde{f}_i)^2].$$

Using the above curvature and gradient estimation schemes it is assumed that the values $\tilde{f}_i$ represent N number of replicated samples in an infinite periodical signal, therefore $\tilde{f}_i = \tilde{f}_{i+N}$. This assumption is necessary, since in our method, as it will be shown in the further discussion, a 3D Fourier transform of the original volume will be exploited. Before we describe how to find filtered values $\tilde{f}_i$ that minimize the global error we shortly discuss several degrees of freedom in penalty function E. The weights S and G determine the relative importance of feature preservation as opposed to smoothing. The linear regression based gradient estimation (function g) takes a local neighborhood into account (Appendix 8.1.). Increasing the size of this neighborhood reduces local staircase artifacts but also increases smoothing. Noise reduction can be achieved by omitting gradients below a certain threshold. In this case function $\tilde{f}$ does not try to follow small noise-related gradient variations in f. Instead of simple thresholding non-linear operations on gradient magnitudes of g are also possible. By, e.g., emphasizing high gradient magnitudes, only the most important features are preserved. The minimization of the penalty function E results in a globally smooth filtered function $\tilde{f}$ preserving the fine details due to the gradient component.

Since we use non-negative G and S parameters the penalty function E is a convex quadratic function having a unique global minimum. At the minimum location the partial derivatives according to all the N unknown values $\tilde{f}_i$ have to be equal to zero, where $i = 0, 1, 2, ..., N-1$:

$$\partial E(\tilde{f}_0, \tilde{f}_1, \tilde{f}_2, ..., \tilde{f}_{N-1}) / \partial \tilde{f}_i = 0. \tag{2}$$

As penalty function E is a quadratic function, the partial derivatives are linear functions of variables $\tilde{f}_i$. Therefore, having all of these partial derivatives evaluated, a large linear equation system is obtained with N unknown variables:

$$A \cdot \tilde{f} = m, \tag{3}$$

where A is a sparse coefficient matrix, and $\tilde{f}$ is the unknown vector containing the N samples of the filtered function. Vector m is derived from the original function samples f_i in the following way:

$$m_i = f_i - 2G \cdot (g_{i+1} - g_{i-1}). \tag{4}$$

Matrix A, derived from the partial derivatives, is defined by a symmetric point spread vector p:

$$p = [p_1, p_2, p_3, p_4, p_5] = [S-G, -4S, 1+6S+2G, -4S, S-G], \tag{5}$$

$$A = \begin{bmatrix} p_3 & p_4 & p_5 & 0 & . & p_1 & p_2 \\ p_2 & p_3 & p_4 & p_5 & . & 0 & p_1 \\ p_1 & p_2 & p_3 & p_4 & . & 0 & 0 \\ 0 & p_1 & p_2 & p_3 & . & p_5 & 0 \\ . & . & . & . & . & p_4 & p_5 \\ p_5 & 0 & 0 & p_1 & p_2 & p_3 & p_4 \\ p_4 & p_5 & 0 & 0 & p_1 & p_2 & p_3 \end{bmatrix}.$$

In 1D, the coefficient matrix A is a symmetric circular matrix. At first sight, it seems that the solution of the large equation system (3) would require tremendous computational time. One possibility of optimization is to exploit the special structure of matrix A. Unfortunately, when the minimization problem is extended to 2D and 3D the coefficient matrix will not have such a simple structure. In this case, the unknown vector $\tilde{f}$ contains all the pixels of the unknown image or all the voxels of the unknown volume. The structure of the coefficient matrix depends on the ordering of these elements. However, matrix A is always a symmetric positive definite matrix, therefore the inverse exists and it is also symmetric and positive definite. A linear equation system defined by such a coefficient matrix can be solved by applying the conjugated gradient method. According to our experience, in case of $S < 50$ and $G < 50$ this matrix was well conditioned ensuring the numerical stability of gradient methods.

As an alternative, we present a fast method for solving the linear equation in frequency domain. In fact, function m is calculated by a convolution of the unknown function $\tilde{f}$ with the kernel p:

$$m_i = \sum_{j=1}^{5} \tilde{f}_{i-j+3} \cdot p_j. \tag{6}$$

Here we assumed that $\tilde{f}_{-1} = \tilde{f}_{N-1}, \tilde{f}_{-2} = \tilde{f}_{N-2}, \tilde{f}_N = \tilde{f}_0$, and $\tilde{f}_{N+1} = \tilde{f}_1$. Because of this assumption, there will be a slight interaction between the opposite border regions of the output signal. This interaction can be reduced by adding an appropriately thick zero extension at the borders.

In the frequency domain the convolution operation is represented by a multiplication: $M(\omega) = \tilde{F}(\omega) \cdot P(\omega)$. Therefore, the Fourier transform $\tilde{F}$ of the unknown vector $\tilde{f}$ can be calculated as $\tilde{F}(\omega) = M(\omega)/P(\omega)$. In the spatial domain the multiplication with matrix A is equivalent with a convolution with kernel p. Similarly, the multiplication with inverse matrix A^{-1} is equivalent with a deconvolution with p, which can be written as a convolution with a kernel p', which is the inverse Fourier transform of $P'(\omega) = 1/P(\omega)$. Such a deconvolution operation in frequency domain is unambiguous (a division by zero problem does not occur) if there is a unique solution of the equivalent linear equation system, which means that matrix A is invertable.

This can be proven in the following way. Matrix A is derived from the partial derivatives of a convex (G and S are non-negative parameters) quadratic penalty function. The penalty function is formally the sum of N number of $(\mathbf{a}_i\mathbf{x} + b_i)^2$ terms, therefore the coefficient matrix A can be written as the sum of diadic products of the corresponding terms $\mathbf{a}_i \cdot \mathbf{a}_i^T$. Since each term is positive semidefinite, matrix A is also positive semidefinite. It is easy to see that in case of non-negative G and S parameters the matrix A is not only positive semidefinite but positive definite, and therefore it is invertable.

Thus, the unknown function $\tilde{f}$ is calculated as an inverse Fourier transform of $\tilde{F}$. Using fast Fourier transformation (FFT) our filtering algorithm consists of the following steps:

1. estimation of gradients g_i using linear regression
2. non-linear operations on the gradient function g
3. calculation of function m using the modified g
4. $M = FFT(m)$ - Fourier transform of function m
5. $P = FFT(p)$ - Fourier transform of function p
6. $\tilde{F} = M/P$ - deconvolution in frequency domain
7. $\tilde{f} = INVFFT(\tilde{F})$ - inverse Fourier transform of $\tilde{F}$

The second step is optional and can be an arbitrary non-linear operation on the gradient function g. The 2D and 3D extension of our method requires 2D and 3D kernel p and m. The derivation is analogous to the 1D case. Results for p and m are given in Appendix 8.2.

4. 2D application - Dedithering

In the image-processing literature there are various local edge-preserving smoothing methods. Global techniques, which are similar to our approach, like the *snake* (active contours) [17, 19, 18] method or the *total variation* [20, 21] method also minimize a global penalty function in order to reduce noise and enhance contours. The first and second order derivative terms used by these methods are fundamentally different from the schemes of our method. The global error is minimized applying different iterative techniques, and usually the numerical stability requires an appropriate preconditioning. In contrast, our method does not rely on an iterative solution and does not use any additional (for example Lagrange) constraints, therefore it is fast and numerically stable.

In order to make the volume smoothing application more understandable, we illustrate our filtering algorithm on a 2D example which is a dedithering problem. A binary dithered image can be considered as a noisy representation of the original image with significant loss of information. The problem is how to reconstruct the original features like sharp edges, smooth homogeneous regions, and fine details from the binary pixels of the dithered image.

In the 2D penalty function the gradients $g(i,j)$ are estimated using linear regression (Appendix 8.1.). In the second term of the penalty function (weighted by G) the gradients of the unknown image are defined by central differences:

$$\tilde{f}_x(i,j) = [\tilde{f}(i+1,j) - \tilde{f}(i-1,j)]/2, \qquad (7)$$

$$\tilde{f}_y(i,j) = [\tilde{f}(i,j+1) - \tilde{f}(i,j-1)]/2.$$

In the third term of the penalty function (weighted by S) we use the following quadratic norm of the Hessian matrix as a measure of the local curvature:

$$\tilde{f}_{xx}^2 + \tilde{f}_{yy}^2 + 2\tilde{f}_{xy}^2. \qquad (8)$$

The 2D algorithm is analogous to the 1D case. The constant term m in Equation 3 and the convolution kernel p (defining the coefficient matrix A) are derived from the Equations 2. The result of the derivation is presented in Appendix 8.2.

We illustrate the 2D version of our filtering method on a binary dithered image. A low resolution detail of this is shown in Figure 1a. For the sake of comparison, first we tried to apply a Gaussian filter to generate a gray-scale image approximating the original one. We used a rather narrow kernel ($\sigma = 1.5$) in order to smooth the homogeneous regions without removing the fine details. Figure 1b shows the result of the Gaussian filtering.

Figure 1c shows an image generated applying our feature-preserving filtering. We used parameter settings $G = 10$ and $S = 24$. Note that, compared to the Gaussian filtering the high frequency details are well preserved. There are also some clearly visible details which are hardly recognizable even by the human eyes when you take a look at the dithered image in Figure 1a.

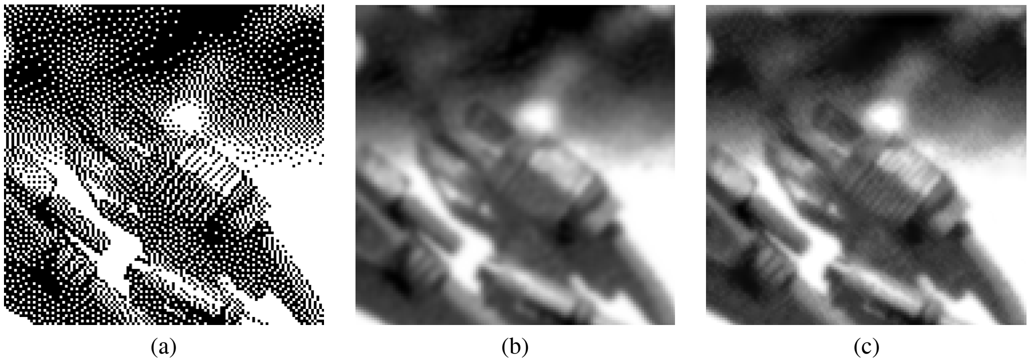


Figure 1: *(a) The original binary dithered image. (b) Gaussian filtering. (c) Feature-preserving filtering.*

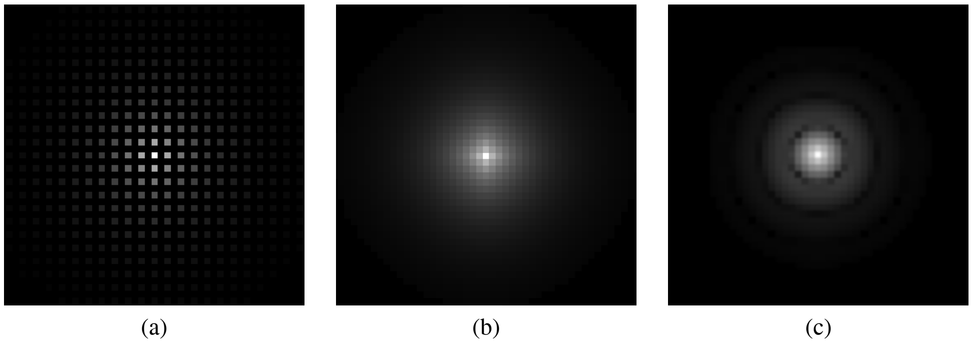


Figure 2: *The middle slice of the equivalent convolution kernel p' using varying parameter settings: (a) S = 0, G = 1, (b) S = 1, G = 1, (c) S = 1, G = 0.*

5. 3D application - Volume filtering

In this section we discuss how to extend our method to the 3D case and how to apply it for feature-preserving volume filtering. Each gradient value $g(i,j,k)$ is estimated using linear regression (Appendix 8.1.).

In the 3D penalty function E the gradients in the filtered volume are approximated by central differences. The gradient component of E (weighted by G) is extended to 3D as follows:

$$G \cdot [(\tilde{f}_x(i,j,k) - g_x(i,j,k))^2 + \qquad (9)$$

$$(\tilde{f}_y(i,j,k) - g_y(i,j,k))^2 +$$

$$(\tilde{f}_z(i,j,k) - g_z(i,j,k))^2].$$

The curvature term of E (weighted by S) can be extended to 3D using finite differences to approximate the following formula:

$$\tilde{f}_{xx}^2 + \tilde{f}_{yy}^2 + \tilde{f}_{zz}^2 + \qquad (10)$$

$$2 \cdot \tilde{f}_{xy}^2 + 2 \cdot \tilde{f}_{xz}^2 + 2 \cdot \tilde{f}_{yz}^2.$$

Note that, the derived linear equation system (3) contains as many unknown variables as the number of voxels which is 256^3 for a typical data set. Fortunately, due to the deconvolution performed in frequency domain, the optimization problem can be efficiently solved. The 3D filtering algorithm is analogous to the 1D case. Appendix 8.2. presents the derived constant term m of Equation 3 and the 3D kernel p used for the deconvolution in Step 6.

In order to illustrate the global behavior of our method we calculated the equivalent convolution kernels which have to be applied for the modified volume m to obtain the same results. The Fourier transform P' of the equivalent kernel p' contains the reciprocals of the coefficients in P. Therefore, kernel p' can be easily calculated as the inverse transform of P'. Figure 2 shows the middle slice of p' evaluated for varying S and G parameters and rendered as gray-scale images. The filter values are non-linearly overemphasized in order to visualize also the low weights. Note that, if weight G is dominant then the kernel is similar to a Gaussian filter while setting a dominant S weight the kernel is getting similar to a *Sinc* function. In fact, the images in Figure 2 depicting convolution kernels in the spatial domain, represent the inverse of the coefficient matrix A in Equation 3. Such an equivalent convolution kernel exists if a non-linear operation in the 2. step of the algorithm is not applied. Since this is a crucial step of the algorithm, in general it is not possible to substitute our method with a simple convolution-based filtering.

Parameter S controls the smoothing of iso-surfaces and parameter G is responsible for preserving the fine details in the volume. Figure 3 shows a very low-resolution ($64 \times 64 \times 32$) artificial binary volume which contains several discretized objects. This binary data set was filtered using varying S and G parameters. The higher the parameter G is, the better the estimated gradients $g(i, j, k)$ are approximated. Since linear regression using a larger voxel neighborhood results in a smooth gradient function an increased G has also a local smoothing effect. This effect is controlled by the radius r of the considered neighborhood and the spherically symmetric weighting function. Here the gradients $g(i, j, k)$ were estimated from a 26-neighborhood weighted by $1/d^2$, where d is the Euclidean distance of a neighboring voxel from the current voxel. In a special case gradient estimation based on linear regression provides the same results as central differences, making the local smoothing effect minimal (Appendix 8.1.). Gradients approximated from such a narrow voxel neighborhood are preferred when also voxel-size details are required to be preserved.

Increasing parameter S the global smoothing effect is getting stronger because of the weighted curvature minimization. In case of $G = 0$ and $S >> 0$ the results approximate only very roughly the original surfaces, and small details are removed. This can be compensated by setting a higher value of G. Figure 3 illustrates how these contradictory conditions fight each other. Generally, the appropriate parameter setting is always a compromise, depending on whether one wants to emphasize small details or rather to enhance smooth characteristic surfaces.

For the sake of comparison, Figure 4 shows the rendering of the original binary volume (a) and the filtered volume (b). In order to make the surfaces smoother and to preserve the small details at the same time, we used parameters $G = 3$ and $S = 3$.

Figure 5 shows a gray-scale data set of a lobster filtered using varying S and G parameters. Figure 6 compares the images of the original and the filtered data generated with the same rendering conditions. For the filtering, we used $S = 3$ and $G = 12$ in order to preserve small details like the feeler of the lobster.

We tested our method also using a more complex transfer function emphasizing two different density ranges at the same time. Figure 7 (see color section) shows the rendering of a human body using the original CT data set (a) and the filtered volume (b). The test data set is rather noisy and it contains contrast material in the blood vessels having nearly the same densities as the bone. Therefore, an appropriate transfer function has to be very fine tuned in order to render the vessels separately from the bone. Figure 7a shows an image rendered using the original data set. Note that, the final image contains point like noise and typical staircase artifacts. In order to remedy these problems we filtered the volume with parameters $G = 1$ and $S = 1$. We also applied a threshold cutting according to the gradient magnitudes for noise reduction. The entire filtering process took 8 minutes for such a $202 \times 152 \times 255$ resolution volume on an 800 MHz Pentium PC with 512M RAM. Figure 7b shows the visualization of the filtered volume using the same rendering parameters as for the original data. Note that the details are well preserved and the iso-surfaces are much smoother. The point like noise is also significantly reduced.

Figure 7 also illustrates that our method is basically a volume-filtering method making all the iso-surfaces smoother inside the volume and it is not restricted to one single iso-surface like the previous global smoothing techniques.

6. Conclusion

In this paper a feature-preserving volume filtering method has been presented. With a three-component penalty function, approximation of the original values, feature preservation and curvature minimization can be controlled efficiently. Images generated by direct volume rendering from the filtered data contain only reduced point like noise and staircase artifacts. Furthermore, the sampled smooth surfaces and fine details can be reconstructed at the same time. Unlike local convolution-based filtering techniques, our method provides a global smoothing effect because of the global curvature minimization. The scalability is ensured by the weighting parameters of the three-component penalty function. Due to the applied FFT method filtering is performed efficiently. Our approach is not restricted to binary data or segmented iso-surfaces, unlike previous techniques based on iterative solution. Although the presented filtering algorithm has been illustrated on a specific 2D and a specific 3D example it can be considered as a general mathematical tool usable for image or volume-processing purposes. Among the possible 2D application fields we mention feature-preserving smoothing or zooming, image restoration, and terrain-modeling. In the 3D case, our technique is applicable to gradient-driven or shape-based interpolation, and smoothing of binary segmented masks.

7. Acknowledgements

The work presented in this publication has been funded by the ADAPT project (FFF-804544). ADAPT is supported by *Tiani Medgraph*, Vienna (http://www.tiani.com), and the *Forschungsförderungsfonds für gewerbliche Wirtschaft*, Austria. See http://www.cg.tuwien.ac.at/research/vis/adapt for further information on this project. Special thanks to Torsten Möller and Attila Neumann for their useful comments.

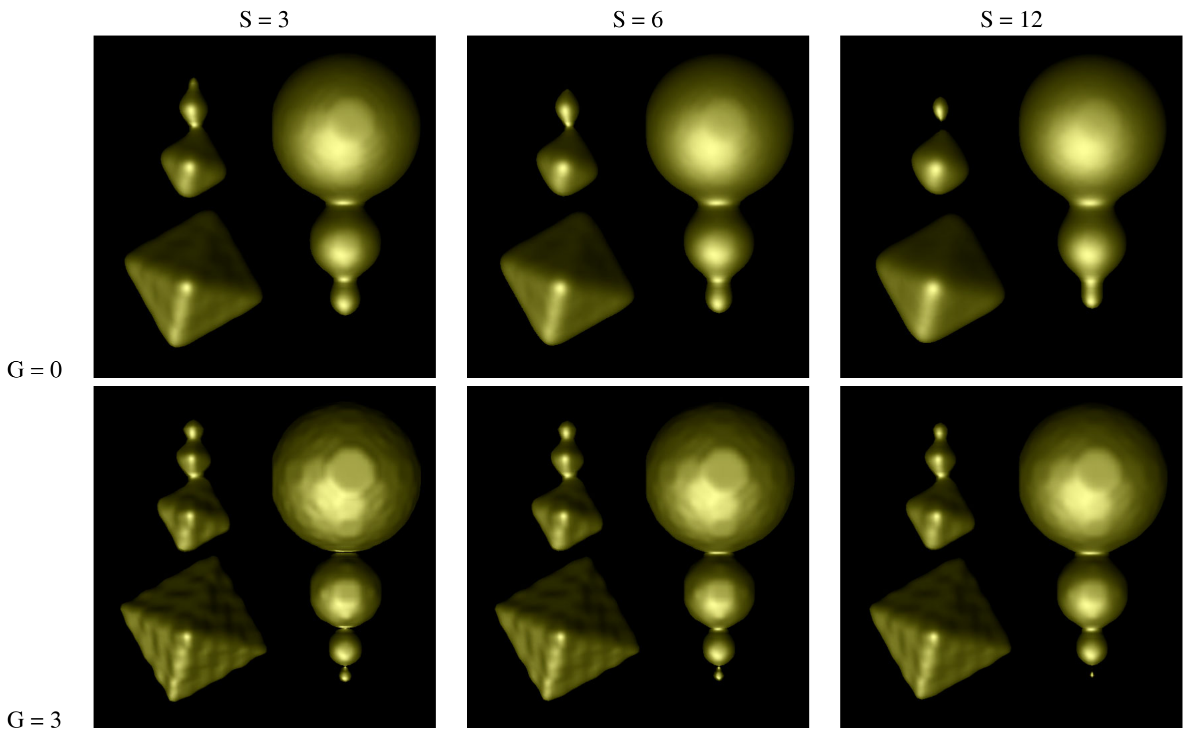


Figure 3: *A binary volume filtered with varying S and G parameters.*

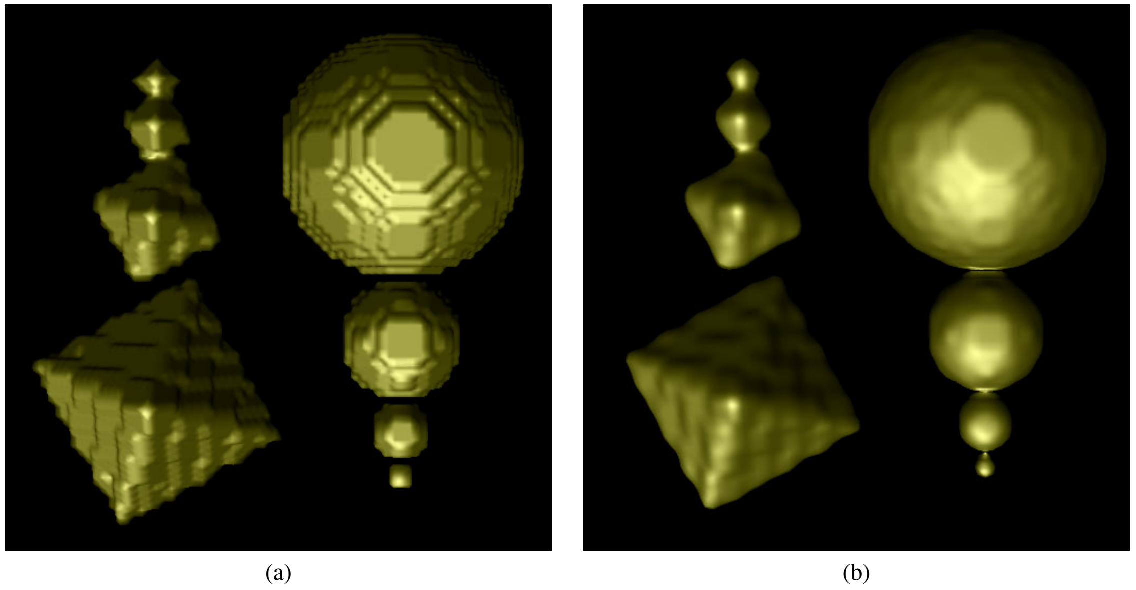


Figure 4: *Rendering of a* $64 \times 64 \times 32$ *binary volume using the original (a) and the filtered (S = 3, G = 3) data (b).*

References

1. T. Möller, R. Machiraju, K. Müller, R. Yagel. A Comparision of Normal Estimation Schemes. *Proceedings of IEEE Visualization '97*, pages 19–26, 1997.

2. G. Thürmer, C. A. Wüthrich. Normal Computation for Discrete Surfaces in 3D Space. *Computer Graphics Forum (Proceedings of EUROGRAPHICS '97)*, pages 15–26, 1997.

3. R. Yagel, D. Cohen, A. Kaufman. Normal Estimation in 3D

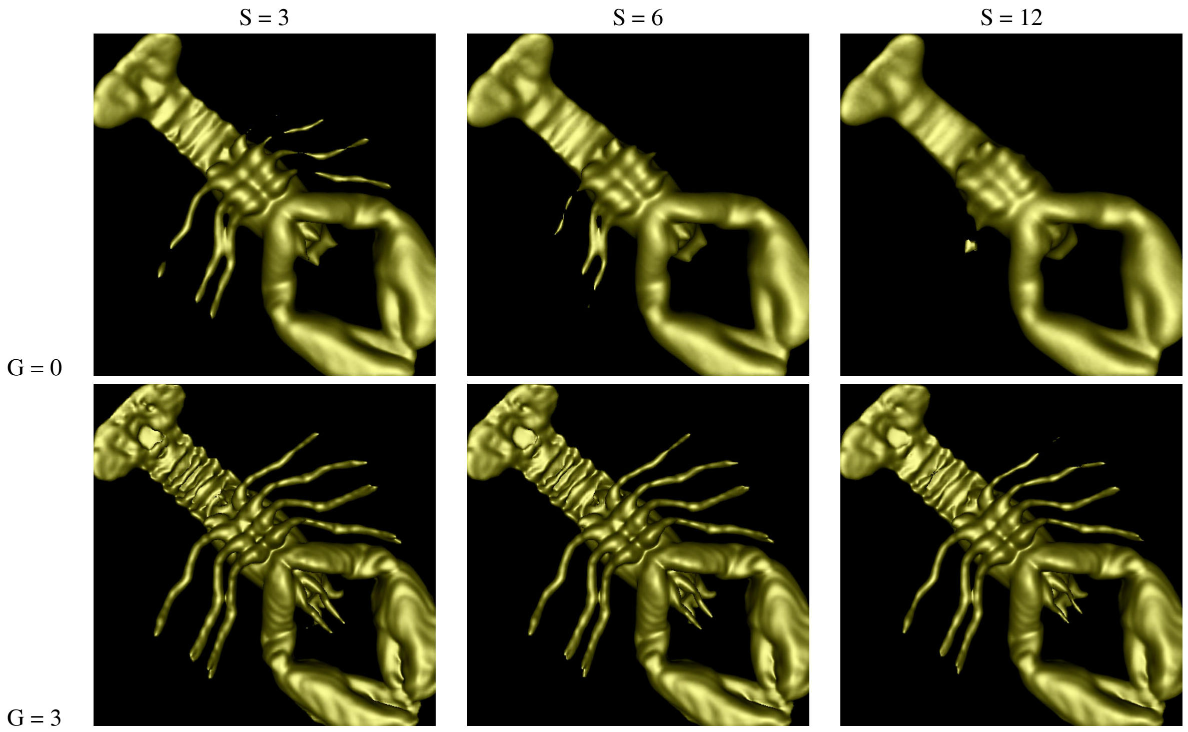


Figure 5: *A CT scan of a lobster filtered with varying S and G parameters.*

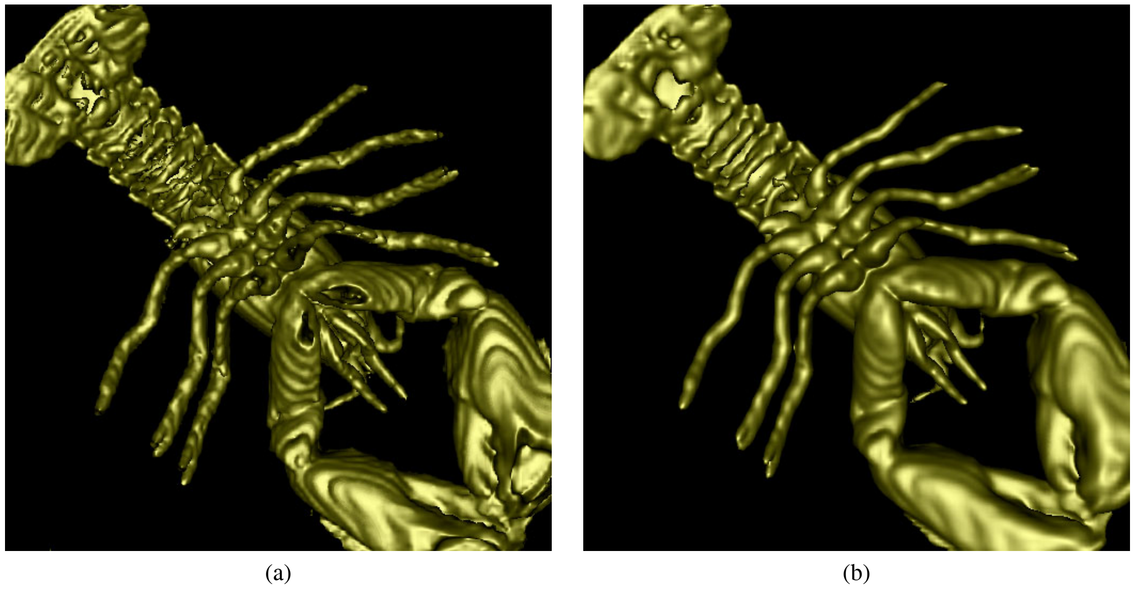


Figure 6: *Rendering of the lobster using the original (a) and the filtered (S = 3, G = 12) data (b).*

Discrete Space. *The Visual Computer*, 8, pages 278–291, 1992.

4. D. Cohen, A. Kaufman, R. Bakalash, S. Bergman. Real Time Discrete Shading. *The Visual Computer*, 6, pages 16–27, 1990.

5. J. Bryant, C. Krumvieda. Display of 3D Binary Objects: I-shading. *Computers and Graphics*, 13(4), pages 441–444, 1989.

6. R. E. Webber. Ray Tracing Voxel Data via Biquadratic Local

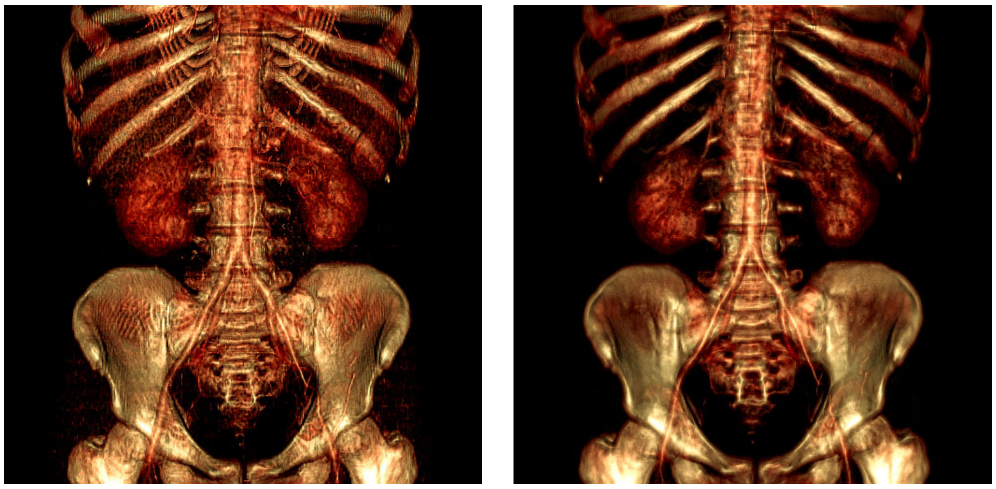

(a) (b)

Figure 7: *Direct volume rendering of a human body using the original (a) and the filtered (S = 1, G = 1) data (b).*

Surface Interpolation. *The Visual Computer*, 6(1), pages 8–15, 1990.

7. R. E. Webber. Shading Voxel Data via Local Curved-surface Interpolation. *Volume Visualization*, (A. Kaufmann, ed.), IEEE Computer Society Press, pages 229–239, 1991.

8. M. J. Bentum, B. B. A. Lichtenbelt, T. Malzbender. Frequency Analysis of Gradient Estimators in Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 2(3), pages 242–254, September, 1996.

9. S. C. Dutta Roy, B. Kumar. Digital Differentiators. in *Handbook of Statistics*, (N. K. Bise and C. R. Rao eds.), vol. 10, pages 159–205, 1993.

10. R. W. Hamming. Digital Filters. in *Prentice Hall Inc.*, Second Edition, Englewoods Cliffs, NJ, 1983.

11. S. F. F. Gibson. Using Distance Maps for Accurate Surface Representation in Sampled Volumes. *IEEE Symposium on Volume Visualization '98*, pages 23–30, 1998.

12. R. T. Whitaker. Reducing Aliasing Artifacts in Iso-Surfaces of Binary Volumes. *IEEE Symposium on Volume Visualization 2000*, pages 23–32, 2000.

13. K. Zuiderveld, A. Koning, Viergever and A. Max. Acceleration of ray casting using 3D distance transformation. *Visualization in Biomedical Computing*, pages 324–335, 1992.

14. M. Sramek, A. Kaufman. Object Voxelization by Filtering. *IEEE Symposium on Volume Visualization '98*, pages 111–118, 1998.

15. G. Turk, J. F. O'Brien. Shape Transformation Using Variational Implicit Functions. *Computer Graphics (Proceedings of SIGGRAPH '99)*, pages 335–342, 1999.

16. L. Neumann, B. Csébfalvi, A. König, E. Gröller. Gradient Estimation in Volume Data using 4D Linear Regression. *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2000)*, pages 351–357, 2000.

17. C. Xu and L. Prince. Generalized gradient vector flow external forces for active contours. *Signal Processing*,(71), pages 131–139, 1998.

18. M. Kass and A. Witkin, D. Terzopoulos. Snakes: Active contour models. *International Journal Computer Vision*, 1(4), pages 321–331, 1987.

19. L. D. Cohen. On active contour models and balloons. *CVGIP: Image Understanding*, (53), pages 211–218, 1987.

20. D. M. Strong and T. F. Chan. Edge-Preserving and Scale-Dependent Properties of Total Variation Regularization. *University of California TR-00-38*, 2000.

21. T. Chan and G. Golub and P. Mulet. A Nonlinear Primal-Dual Method for Total Variation-based Image Restoration. *SIAM J. Scientific Computing*, (20), pages 1964–1977, 1999.

8. Appendix

8.1. Gradient estimation using linear regression

Derivatives of a 1D sampled function $f(x)$ can be estimated using a simple linear regression method [16]. According to this approach a linear function $a \cdot x + b$ can be determined, which approximates locally the original function $f(x)$ with a minimal error. The value of a is considered as an estimated local derivative. Let $w(i)$ be a symmetric weighting function, where $w(i) = w(-i)$ and $w(i+1) < w(i)$ if $i > 0$. Weighting the error contribution of the neighboring samples by function $w(i)$ the error optimization leads to the following gradient estimation formula:

$$f'(x) = \frac{1}{\sum_i w(i) \cdot i^2} \sum_i w(i) \cdot i \cdot f(x+i). \qquad (11)$$

This method can be easily extended to 2D and 3D gradient estimation. In 2D, denoting the sampled function as $f(x,y)$ and the symmetric weighting function as $w(i,j)$ the estimated gradient components are calculated the following way:

$$f_x(x,y) = \frac{1}{\sum_{i,j} w(i,j)\cdot i^2} \sum_{i,j} w(i,j)\cdot i \cdot f(x+i, y+j), \quad (12)$$

$$f_y(x,y) = \frac{1}{\sum_{i,j} w(i,j)\cdot j^2} \sum_{i,j} w(i,j)\cdot j \cdot f(x+i, y+j),$$

where $w(i,j) = w(-i,-j)$ and $w(i,j) < w(k,l)$ if $i^2 + j^2 > k^2 + l^2$.

The 3D gradient estimation formulas are analogous to the 2D case:

$$f_x(x,y,z) = \frac{1}{W\cdot i^2} \sum_{i,j,k} w(i,j,k)\cdot i \cdot f(x+i, y+j, z+k), \quad (13)$$

$$f_y(x,y,z) = \frac{1}{W\cdot j^2} \sum_{i,j,k} w(i,j,k)\cdot j \cdot f(x+i, y+j, z+k),$$

$$f_z(x,y,z) = \frac{1}{W\cdot k^2} \sum_{i,j,k} w(i,j,k)\cdot k \cdot f(x+i, y+j, z+k),$$

where $W = \sum_{i,j,k} w(i,j,k)$. In fact, these gradient estimation schemes define kernels for computationally efficient convolution. The convolution is evaluated only for those neighboring samples, where the weighting function w is greater than zero. The weighting function controls the smoothness of the estimated gradient function. The wider the considered neighborhood is the stronger is the local smoothing effect. For instance, in 3D the classical gradient estimation based on central differences is a special case of the linear regression method using the following weighting function:

$$w(i,j,k) = \begin{cases} 1 \text{ if } i = \{1,-1\}, j = 0, k = 0 \\ \text{ or } i = 0, j = \{1,-1\}, k = 0 \\ \text{ or } i = 0, j = 0, k = \{1,-1\} \\ 0 \text{ otherwise.} \end{cases} \quad (14)$$

In this special case the local smoothing is minimal since just a narrow voxel neighborhood is taken into account.

8.2. The 2D and 3D convolution kernels

In the 2D case, the convolution kernel p derived from Equations 2 is defined as the following 5×5 point spread matrix:

$$p = \begin{bmatrix} \frac{S}{8} & 0 & \frac{3}{4}S - G & 0 & \frac{S}{8} \\ 0 & 0 & -4S & 0 & 0 \\ \frac{3}{4}S - G & -4S & 1 + 4G + 12.5S & -4S & \frac{3}{4}S - G \\ 0 & 0 & -4S & 0 & 0 \\ \frac{S}{8} & 0 & \frac{3}{4}S - G & 0 & \frac{S}{8} \end{bmatrix}.$$

The 2D function $m(i,j)$ in Equation 3 is calculated similarly to the 1D case:

$$m(i,j) = f(i,j) - \quad (15)$$
$$2G \cdot [(g_x(i+1,j) - g_x(i-1,j) +$$
$$g_y(i,j+1) - g_y(i,j-1)].$$

In the 3D case, the derived convolution kernel p is a $5 \times 5 \times 5$ volume. Let us denote the ith slice by p_i. Because of the symmetric kernel $p_1 = p_5$ and $p_2 = p_4$. The slices are defined by the following matrices:

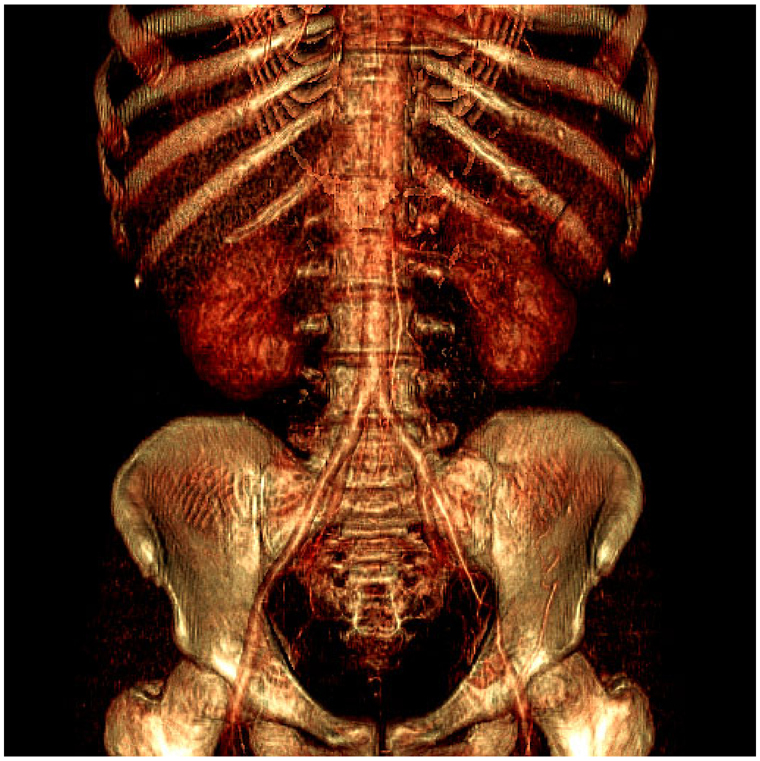
$$p_1 = \begin{bmatrix} 0 & 0 & \frac{S}{8} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{S}{8} & 0 & \frac{S}{2} - G & 0 & \frac{S}{8} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{S}{8} & 0 & 0 \end{bmatrix},$$

$$p_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -4S & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

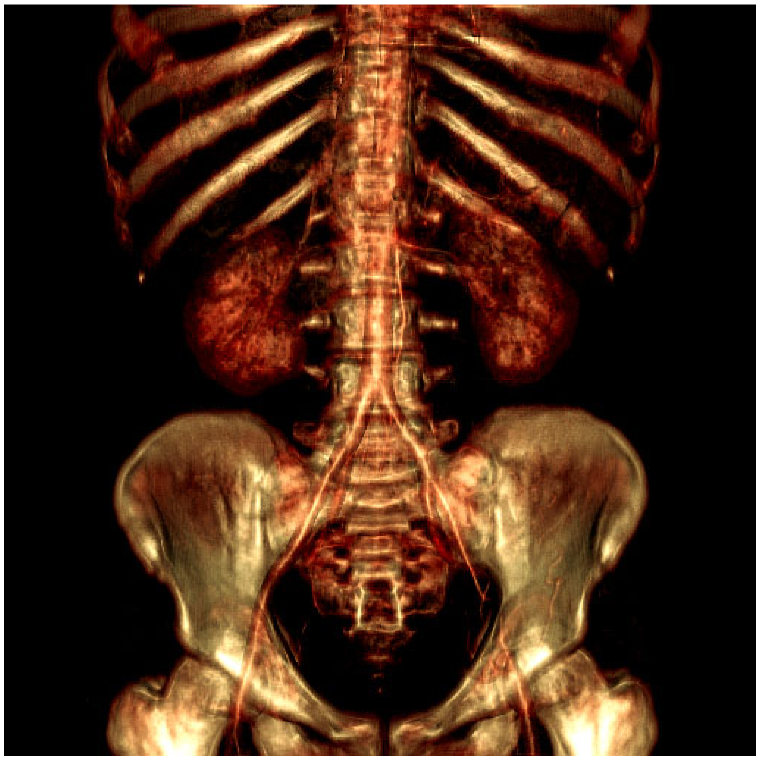
$$p_3 = \begin{bmatrix} \frac{S}{8} & 0 & \frac{S}{2} - G & 0 & \frac{S}{8} \\ 0 & 0 & -4S & 0 & 0 \\ \frac{S}{2} - G & -4S & 1 + 6G + 19.5S & -4S & \frac{S}{2} - G \\ 0 & 0 & -4S & 0 & 0 \\ \frac{S}{8} & 0 & \frac{S}{2} - G & 0 & \frac{S}{8} \end{bmatrix}.$$

The constant term $m(i,j,k)$ in Equation 3 is analogous to the 2D case:

$$m(i,j,k) = f(i,j,k) - \quad (16)$$
$$2G \cdot [(g_x(i+1,j,k) - g_x(i-1,j,k) +$$
$$g_y(i,j+1,k) - g_y(i,j-1,k) +$$
$$g_z(i,j,k+1) - g_z(i,j,k-1)].$$



(a)



(b)

Figure 8: *Direct volume rendering of a human body using the original (a) and the filtered (S = 1, G = 1) data (b).*