Volume Rendering with Least Squares Spline Reconstruction

Željka Mihajlović, Leo Budin and Zoran Kalafatić

Department of Electronics, Microelectronics, Computer and Intelligent Systems,
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Hrvatska
e-mail: zeljka.mihajlovic@fer.hr, leo.budin@fer.hr
phone: +385 1 6129 944; fax: +385 1 6129 653

ABSTRACT

This paper presents a volume rendering procedure with embedded least square spline reconstruction. The first part of this paper describes a technique for the continuous representation of discrete signals by using B-splines. Volume element space is an example of the discrete signal which can be continuously represented in terms of the B-spline. Volume rendering is very time-consuming procedure and therefore we down-sample the initial size of the volume. Before the down-sampling some low-pass prefiltering is required. Our design of the prefilter is based on considering the combined operation of the prefiltering and the reconstruction postfiltering.

The second part of this paper deals with embedding of the least square spline filter into the volume rendering procedure. The position of the reconstruction point is arbitrary in the volume element space. Reconstruction of the normal vector required for shading could be easily derived from the continuous reconstruction function.

1. INTRODUCTION

Visualization techniques are a very powerful tool for investigation and understanding of various types of objects. They are often used for non-invasive inspection of object's interior. Scanning methods such as CT, MR and ultrasound produce large number of slices through the object. One of the techniques that is often used to visualize that kind of data, is volume rendering [1, 2].

In the volume rendering, conversions between continuous and discrete representation are required at the several steps of the procedure. From the viewpoint, rays are cast through the projection plane in the volume element space. In the volume element space an object surface is implicitly defined by the given set of scanned samples. To find the point of intersection of a ray and the surface of an object, the reconstruction of the sampled function is required along the ray. Along the ray the reconstruction is performed only at discrete points. Final image is produced according to the rays that are cast in the volume element space. In all of these steps, alias artifacts are accumulated.

Reconstruction that is applied in the volume rendering procedure plays an important role on the quality of the object rendering. Simple reconstructions, in three-dimensional space, are frequently used because they involve a small number of samples. Nearest neighbor reconstruction involves only one sample, the one nearest to the ray. Alias artifacts due to the nearest neighbor reconstruction are very strong. Trilinear interpolation involves eight samples. Aliasing artifacts still significantly influence the smoothness of the reconstructed surface. Therefore, a better interpolation with a wider reconstruction kernel is required to improve the quality of reconstruction.

Cubic filters make trade-off between computational cost and the suppression of alias artifacts. T. Moller et al. [3] found the Catmull-Rom spline filter and its derivative the most accurate reconstruction and derivative filters, respectively, among the class of BC-splines. On the other hand, the interpolation B-spline, if it is correctly used, could produce even better results. We expand the basic concepts of the interpolation B-spline to make a test-bed for any other filter, such as Catmull-Rom. We also incorporate the reconstruction kernel directly in the volume rendering procedure. The interpolation B-spline requires high-pass prefiltering to improve overall impulse response of the prefilter and reconstruction kernel. Prefiltering could be done in frequency domain or as a combination of two small causal and anticausal filters in spatial domain.

B-splines have been used for a long time in computer graphics for approximation and interpolation of curves and surfaces. Unser et al. in their excellent work [4, 5] propose mechanisms for efficient design and use of B-spline filters in terms of indirect and direct B-spline transforms. Trough the B-spline transform, all the important and interesting features

This work has been carried out within project founded by Ministry for Science, Technology and Informatics of the Republic of Croatia

of B-splines become available for various applications. The original definition of B-spline is unsuitable for implementation of the multi-dimensional approximation and especially for interpolation of discrete sample fields. Indirect and direct B-spline transforms enable the calculation of the B-spline transform coefficients and the reconstruction of the interpolation function through the given set of samples. R. Panda and N. B. Chatterji [6] use a family of generalized B-spline filters in image processing and show that generalized least square cubic B-spline filters could be used to improve the performance of image compression techniques.

2. B-SPLINE INTERPOLATION

An often mistake in the implementation of B-spline reconstruction filter is its implementation as the approximation B-spline instead of the interpolation B-spline. Hence, we have to point out the difference. Interpolation spline passes *through* the given set of points while the approximation spline is only *close to* the given set of points. The B-spline transforms [4] correctly implement the interpolation B-spline. The transforms of B-spline are given with the following procedure. In the formula:

$$\Phi_n(k) = f(k) = \sum_{l=-\infty}^{\infty} c_n(l)\beta_n(k-l)\,, \qquad (1)$$

the given set of points is $f(k)$. By the definition of interpolation, the values of the reconstructed function $\Phi_n(x)$ at the knot points $x = k$ must be equal to the values of the given points $f(k)$. The sequence $c_n(l)$ denotes B-spline coefficients, which have to be calculated. $\beta_n(x)$ is the n-th order B-spline weight function.

2.1. Indirect B-spline Transform

Calculation of coefficients $c_n(l)$ in equation (1) determines the indirect B-spline transform. $\beta_n(x)$ is n-th order B-spline weight function determined by:

$$\beta_n(x) = \frac{1}{n!}\sum_{i=0}^{n+1}(-1)^i \binom{n+1}{i} \left(x - i + \frac{n+1}{2}\right)^n \cdot H\left(x - i + \frac{n+1}{2}\right) \qquad (2)$$

where $H(x)$ is the Heaviside step-function. At this point, it must be stressed that this is only a special case of B-spline, with uniformly distributed knots. For the boundary conditions, periodic extension is assumed. Under these circumstances, B-spline weight functions are periodic and they are significantly easier to calculate than in general case. Moreover, if we define $\beta_0(x)$ as:

$$\beta_0(x) = \begin{cases} 1, & -\frac{1}{2} \le x \le \frac{1}{2} \\ 0, & \text{otherwise} \end{cases}\,, \qquad (3)$$

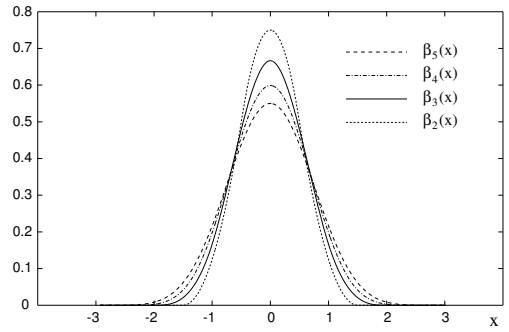


Fig. 1. B-spline weight functions of order 2 to 5.

then B-spline weight functions of any order can be defined by recursive relation:

$$\beta_n(x) = (\beta_{n-1} * \beta_0)(x)\,, \qquad (4)$$

where operator * denotes convolution. This is a very useful property of B-spline weight functions. The Fourier transform of $\beta_0(x)$ can be found easily:

$$B_0(\omega) = \frac{\sin(\frac{\omega}{2})}{\frac{\omega}{2}} = \mathrm{sinc}(\frac{\omega}{2\pi})\,, \qquad (5)$$

where the usual definition of *sinc* function is used:

$$\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}\,.$$

The frequency response of B-spline function of any order n can be derived from its definition (4), using the convolution theorem for Fourier transform. Hence, $B_n(\omega)$ is given by

$$B_n(\omega) = B_0^{n+1}(\omega) = \mathrm{sinc}^{n+1}(\frac{\omega}{2\pi})\,. \qquad (6)$$

Several B-spline weight functions are shown in Fig. 1. This enables further analysis of B-splines by using Fourier transform. Transformation of equation(1) gives:

$$F(\omega) = C_n(\omega)B_n(\omega)\,, \qquad (7)$$

where $F(\omega)$, $C_n(\omega)$ and $B_n(\omega)$ are Fourier domain representations of $\Phi(x)$, $c_n(x)$ and $\beta_n(x)$ respectively. Fourier transform of $c_n(x)$ can be derived easily from (7) in the form:

$$C_n(\omega) = \frac{1}{B_n(\omega)}F(\omega)\,. \qquad (8)$$

This equation is valid only if $B_n(\omega)$ has no zeroes on the frequency axis ω. This condition is satisfied for all B-splines.

2.2. Direct B-spline Transform

After the determination of the coefficients $c_n(l)$, direct B-spline transform is simple. The condition in the starting formula (1) on the coefficients $c_n(l)$ was that the given set of points $f(k)$ must be equal to

the values of reconstructed function $\Phi_n(x)$ at the positions k of the input points. Now, the coefficients $c_n(l)$ are known and continuous interpolation B-spline function $\Phi_n(x)$ function is given by convolution:

$$\Phi_n(x) = \sum_{l=-\infty}^{\infty} c_n(l)\beta_n(x-l)\,. \qquad (9)$$

This is the direct B-spline transform. In computer graphics the indirect B-spline transform is performed by LU decomposition or Gauss elimination of the corresponding matrix equation. If the B-spline is periodic, Toeplitz matrix appears and convolution could be used.

2.3. Least Squares Splines

In order to reduce the memory requirements of the reconstruction procedure, down-sampling of the signal is needed. To suppress aliasing, low-pass prefiltering is required. The choice of low-pass prefilter is often not argued, but the Gaussian prefilter is used instead. The idea here is to design a prefilter by considering the whole procedure. The procedure consists of the following steps: low-pass prefiltering, down-sampling, B-spline indirect transform (high-pass prefilter) and reconstruction with direct B-spline transform.

In the equation (1) down-sampling of the input sequence $[f(k)]_{\downarrow M}$ is required, by factor M. This is analogous to up-sampling of the coefficient sequence by the same factor $[c_n(l)]_{\uparrow M}$, where the sequence $[c_n(l)]$ is M times shorter than $[f(k)]$. To derive a prefilter we will consider the sum of squared differences between the original and the reconstructed sequence ϵ^2:

$$\epsilon^2 = \sum_k \left(f(k) - [c(k)]_{\uparrow \mathrm{M}} * b(k)\right)^2 .$$

Generally, ϵ is a function of $c(p)$, for all possible values of p. At the point of minimum, all its partial derivatives with respect to $c(p)$ must vanish. Defining the reversed (mirrored) reconstruction function as $b_r(i) = b(-i)$, after some manipulation, the above expression may be written as:

$$f(\mathrm{M}p) * b_r(\mathrm{M}p) = [c(\mathrm{M}p)]_{\uparrow \mathrm{M}} * b(\mathrm{M}p) * b_r(\mathrm{M}p).$$

Here, equation holds *only* at points (Mp). So, we down-sample the previous condition by factor M and obtain:

$$[f(p) * b_r(p)]_{\downarrow \mathrm{M}} = c(p) * [b(p) * b_r(p)]_{\downarrow \mathrm{M}}.$$

One of the conditions on $b(k)$ is that $[b(p) * b_r(p)]_{\downarrow \mathrm{M}}$ has the inverse, so finally we have:

$$c(p) = \left([b(p) * b_r(p)]_{\downarrow \mathrm{M}}\right)^{-1} * [f(p) * b_r(p)]_{\downarrow \mathrm{M}}. \qquad (10)$$

This is an expression that holds generally. We use the same formula (10) to derive the prefilters for the Catmull-Rom spline and B-spline. These results suggest a procedure for determination of the least squares prefilters, as follows. First, the input sequence $f(p)$ is prefiltered with the chosen reversed reconstruction kernel $b_r(p)$. Then, the result is down-sampled. The second step is to prefilter the obtained result with the prefilter $\left([b\,(p) * b_r\,(p)]_{\downarrow M}\right)^{-1}$. After that step, the sequence is ready for reconstruction, resampling or calculation of its derivative. We expand this prefiltering procedure to the three-dimensional volumetric and embed the reconstruction kernel into the volume rendering procedure.

3. VOLUME RENDERING

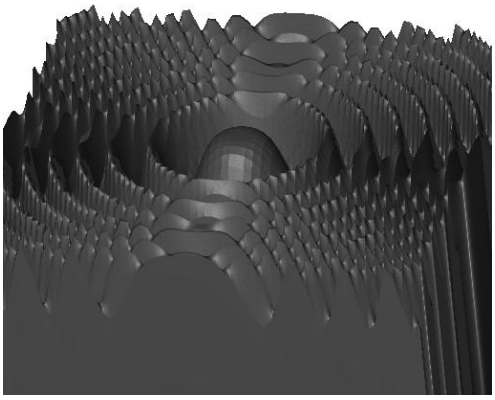
The memory requirements for large volume element spaces is a great problem. Simple down-sampling could cause the irrecoverable loss of information. Prefiltering and down-sampling according to the equation (10) ensures the quality of reconstruction.

In the volume rendering, the reconstruction is can be done in two ways. The first approach is to enlarge the whole data-set in all three dimensions by using a reconstruction kernel. In that case reconstruction in the rendering procedure is still required. The second approach is to embed the reconstruction kernel directly in the rendering procedure.

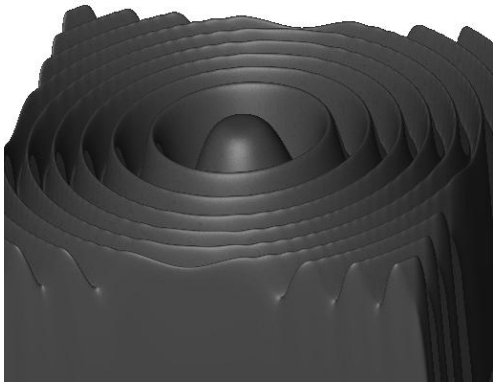
In our implementation we embed reconstruction kernels of Catmull-Rom spline and cubic B-spline directly in the rendering procedure. The volume rendering also requires derivative for shading procedure. It is important that the same prefiltering procedure is valid and only derivative kernel is applied.

Figure 2 shows the results of four different reconstructions applied in the rendering procedure. The first one is simple trilinear interpolation. The data size is 64^3. Aliasing artifacts are very strong for simple interpolation such as trilinear (Fig. 2.a). The second reconstruction is based on approximation B-spline. The reconstruction kernel is same as for the least square B-spline, but prefiltering is not previously applied. Approximation B-spline cause strong blurring and the waves in the resulting surfaces are very shallow (Fig. 2.b).

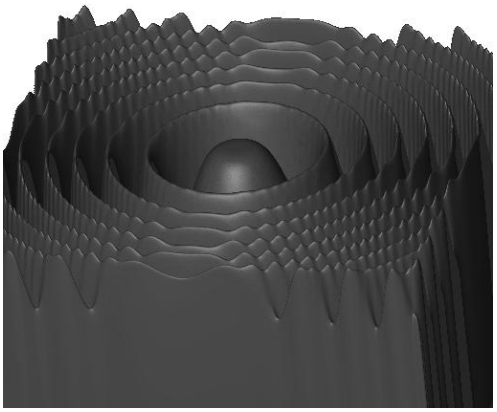
For the Catmull-Rom reconstruction, equation (10) was used with Catmull-Rom reconstruction kernel $b(k)$. The original size 128^3 of the data was down-sampled two times in each direction and the appropriate high-pass prefiltering was applied. The alias artifacts appear on the crest of the waves (Fig. 2.c). For the least squares B-spline reconstruction, we used the same data and procedure as for the Catmull-Rom spline. The prefilters were generated with cubic B-spline reconstruction kernel. The results obtained by the least squares B-splines show the smallest alias artifacts (Fig. 2.d).



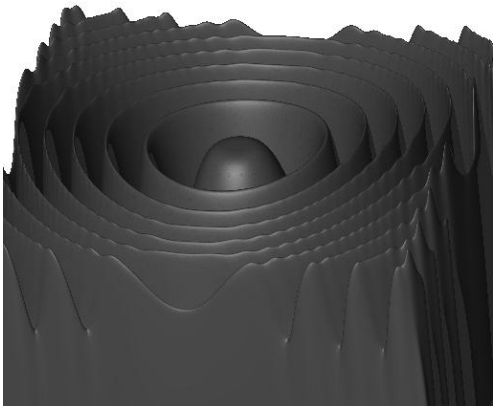
(a) Trilinear interpolation.



(b) Cubic B-spline approximation.



(c) LS Catmull-Rom spline.



(d) LS Cubic B-spline interpolation.

Fig. 2. Four reconstruction schemes. Least square calculation was performed for the images (c) and (d).

4. CONCLUSIONS

In this paper, we have presented a volume rendering procedure with least square splines used for reconstruction. We have shown that the reconstruction in the volume rendering is a very important and delicate step. We consider the whole procedure of prefiltering, down-sampling and reconstruction to generate prefilters and we embed the reconstruction and derivative kernels in the volume rendering. We tested our procedure on the Catmull-Rom and cubic B-spline reconstruction kernels, but the other reconstruction kernels could also be used.

5. REFERENCES

[1] M. Levoy, "Display of surfaces from volume data," *Proc. IEEE Computer Graphics and Applications*, vol. 8, no. 3, pp. 29–37, May 1988.

[2] P. Lacroute and M. Levoy, "Fast volume rendering using a shear-warp factorization of the viewing transformation," *Computer Graphics*, vol. 28, no. 4, pp. 451–458, July 1994.

[3] T. Moller, R. Machiraju, K. Mueller, and R. Yagel, "Evaluation and design of filters using a Taylor series expansion," *IEEE Transactions on Visualization and Computer Graphics*, vol. 3, no. 2, pp. 184–199, June 1997.

[4] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part I-theory," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 821–833, February 1993.

[5] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part II-efficient design and applications," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834–848, February 1993.

[6] R. Panda and B.N. Chatterji, "Least squares generalized B-spline signal and image processing," *Signal Processing*, vol. 81, no. 10, pp. 2005–2017, October 2001.