Extracting Surfaces from Fuzzy 3D-Ultrasound Data

Georgios Sakas, Stefan Walter
Fraunhofer Institute for Computer Graphics[1]

Abstract

Rendering 3D models from 3D-ultrasonic data is a complicated task due to the noisy, fuzzy nature of ultrasound imaging containing a lot of artifacts, speckle etc. In the method presented in this paper we first apply several filtering techniques (low-pass, mathematical morphology, multi-resolution analysis) to separate the areas of low coherency containing mostly noise and speckle from those of useful information. Our novel BLTP filtering can be applied at interactive times on-the-fly under user control & feed-back. Goal of this processing is to create a 'region-of-interest' (ROI) mask, whereas the data itself remains unaltered. Secondly, we examine several alternatives to the original Levoy contouring method. Finally we introduce an improved surface-extraction volume rendering procedure, applied on the original data within the ROI areas for visualizing high quality images within a few seconds on a normal workstation, or even on a PC, thus making the complete system suitable for routine clinical applications.

CR Descriptors: General Terms: **Algorithms**. I.3.3 [**Computer Graphics**]: Picture/image generation; I.3.8 [**Computer Graphics**]: Applications; I.4.3 [**Image Processing**]: Enhancement, Smoothing, Filtering; I.4.6 [**Image Processing**]: Segmentation, Edge and Feature Detection, Pixel Classification; J.3 [**Life and Medical Sciences**]. **Additional Keywords:** 3D ultrasound, multi-resolution analysis, morphology, volume rendering

1 Introduction

3D ultrasound is a very new and most interesting application in the area of 'tomographic' medical imaging, able to become a fast, non-radiative, non-invasive, and inexpensive volumetric data acquisition technique with unique advantages for the localisation of vessels and tumours in soft tissue (spleen, kidneys, liver, breast etc.). In general, tomographic techniques (CT, MR, PET etc.) allow for a high anatomical clarity when inspecting the interior of the human body. In addition, they enable a 3D reconstruction and examination of regions of interest, offering obvious benefits (reviewing from any desired angle, isolation of crucial locations, visualization of internal structures, 'fly-by', accurate measurements of distances, angles, volumes etc.).

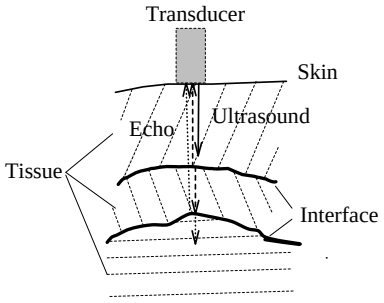


Figure 1: The principal function of ultrasound

The physical principle of ultrasound is as following [11]: sound waves of high frequency (1–15 MHz) emanate from a row of sources that are located on the surface of a transducer which is in direct contact with the skin. The sound waves penetrate the human tissue travelling with a speed of 1450–1580 m/s, depending upon the type of tissue. The sound waves are reflected partially if they hit an interface between two different types of tissue (e.g. muscle and bone). The reflected wavefronts are detected by sensors (microphones) located next to the sources on the transducer. The intensity of reflected energy is proportional to the sound impedance difference of the two corresponding types of tissue and depends on the difference of the sound impendances Z_1 and Z_2:

$$I_r = I_e \cdot \frac{\left(1 - \frac{Z_2}{Z_1}\right)}{\left(1 + \frac{Z_2}{Z_1}\right)} \qquad (1)$$

An image of the interior structure can be reconstructed based upon the total travelling time, the (average) speed, and the energy intensity of the reflected waves. The resulting 3D images essentially represent hidden internal "surfaces". The principle is similar to radar with the difference being that it uses mechanical instead of electromagnetic waves.

In contrast to the common 2D case where a single image slice is acquired, 3D ultrasonic techniques cover a volume within the body with a series of subsequent image slices. The acquisition of these slices can be achieved by means of various scan and tracking techniques and will be not discussed further in this paper; please refer to [12], [19], and [21] for more details.

[1]Fraunhofer Institute for Computer Graphics
Wilhelminenstr. 7, 64283 Darmstadt, Germany
fax: +49/6151/155-199, email: {gsakas,walter}@igd.fhg.de

2 Previous Works & Drawbacks

One of the major reasons for the limited acceptance of 3D ultrasound to date is the complete lack of an appropriate visualisation technique, able to display clear surfaces out of the acquired data. Our very first approach was to use well known techniques, used for MRI and CT data to extract surfaces. Such techniques include binarization [7], iso-surfacing [4], contour connecting [3], marching cubes [9], and volume rendering either as semi-transparent cloud [13], or as fuzzy gradient shading [8]. Manual contouring [19] is too slow and impractical for real-life applications. Unfortunately, ultrasound images posses several features causing all these techniques to fail totally. The most important of these features as reported in [16] are:

1. significant amount of noise and speckle
2. much lower dynamic range as compared to CT or MR
3. high variations in the intensity of neighbouring voxels, even within homogeneous tissue areas
4. boundaries with varying grey level caused by the variation of surface curvature and orientation to the sound source
5. partially or completely shadowed surfaces from objects closer and within the direction of the sound source (e.g. a hand shadows the face)
6. the regions representing boundaries are not sharp but show a width of several pixels
7. poor alignment between subsequent images (parallel–scan devices only)
8. pixels representing varying geometric resolutions depending on the distance from the sound source (fan–scanning devices only)

The general appearance of a volume rendered 3D ultrasound dataset is that of a solid block covered with 'noise snow' (fig. 13 right). A closer analysis proved that noise and speckle contained in the image caused so many obstacles (blobs) around the objects, that rays usually fail to penetrate deep enough to reach the crucial internal surfaces. The low dynamic range makes a straight-forward discrimination between speckle and information (e.g. by means of thresholding) impossible. Even when a surface is reached, methods based on a single threshold value ([4], [7], [9] etc.) fail to detect a continuous surface due to the features 3, 4, & 5 listed above. Lastly, the nature of gradient shading employed, e.g. in [8], is very sensitive to high frequency noise and speckle and therefore it is not suitable to generate a smooth surface because of reasons 3, 6, 7, & 8 (see fig. 2 upper left). However, Levoy's continuous opacity classificator (eq. 2) has been found to give good estimations of the presence of a surface contour (a and r are scaling factors and S a threshold value):

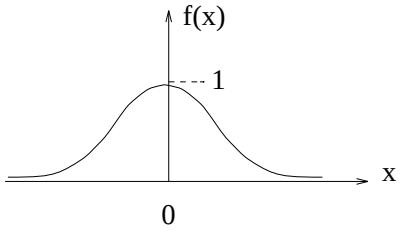
$$opacity = a \cdot \begin{cases} 1 : g(u,v,w) = S \\ 1 - \frac{1}{r} \cdot \frac{|g(u,v,w) - S|}{|\nabla g(u,v,w)|} : |\nabla g(u,v,w)| > 0 \\ 0 : otherwise \end{cases} \quad (2)$$

Some time ago we started to search for ways of discriminating the tissue of interest from the lerge amount of apparently noisy signal. Initially we filtered the original voxels with an approximation of a Gaussian kernel with discrete binomial coefficients due to its separability, normality and symmetry [6]:

$$G(x,\sigma) = \frac{1}{2 \cdot \pi\sigma^2} \cdot e^{\left(\frac{x^2}{2 \cdot \sigma^2}\right)} \quad (3)$$

In a discrete implementation the factors of this kernel are usually calculated by the binomial coefficients:

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} \quad 0 \le k \le n$$



Filtering a function g with a filter F is a convolution [6]:

$$g'(x) = g(x) * F(x) = \int_{-\infty}^{+\infty} g(x) \cdot F(x - x_1) \cdot dx_1 \quad (4)$$

In the discrete case,, a convolution is expressed as a weighted sum of the signal g over the $(2n+1)$ size of the filter kernel F:

$$g'(u) = \sum_{i=-n}^{n} g(u) \cdot F(u - i) \quad (5)$$

An extension to higher dimensions is straightforward by convoluting the filter function by itself. The main effect of such a low-pass filtering is a smoothing of sharp details together with a reduction of noise and speckle. The separability enables a filtering of a 256^3 dataset by a Gaussian kernel of 3^3 within 6.51 seconds, see also tab. 2. On the other hand, the intermediate results must be buffered, which temporally requires a duplication of memory space.

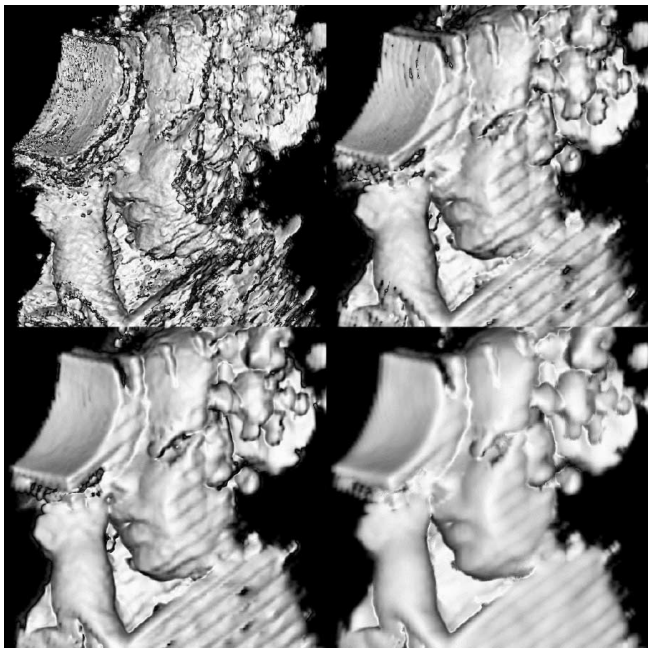


Figure 2: Filtering and volume rendering of an original 256^3 dataset with Gaussian kernels of 3^3, 5^3, and 7^3

Fig. 2 presents a 256^3 dataset after filtering with Gaussian kernels of increasing size and volume rendering using Levoy's method [8]. Although noise and speckle artifacts are reduced, the surfaces gradually appear over-smoothed, non-sharp, unnatural, and do not show much details.

In an earlier study [16] we tested the processing of the original data by the combination of three different filters: Gaussian for noise/speckle reduction, speckle removal for contour smoothing, and median for both noise/speckle reduction and closing of small gaps caused by misalignments and differences in the average luminosity between subsequent images. The filters had been implemented in 3D and applied to the acquired data during a preprocessing stage. Although the results have been very encouraging, such filters require significant computing times and therefore they remain pre-processing stages. Further, the large pre-processing times do not allow interactive control of the filtering level by the user, thus making this approach inflexible, non-interactive and not suitable for clinical applications. Finally filtering also changes the original nature of the data and is therefore not generally accepted in the medical community.

3 Multi-Resolution Surface Identification

The aim of this paper is to develop a fast, intuitive and effective method for rendering clear contours from fuzzy data. All techniques must allow for a user-defined, interactive, on-line processing on average computers in order to be used under routine clinical conditions. This goal has been achieved through a combination of multi-resolution analysis [2], [18], [10], filtering [6], and volume rendering [8], and consists of three subsequent tasks:

1. identify and separate noisy areas carrying effectively no information (segmentation), discussed in section 4
2. process the data within the remaining areas (filtering), discussed in section 5
3. use adequate visualization methods (volume rendering), discussed in section 6.

The first task is used to allow the rays to penetrate up to the internal surfaces of interest. In addition, it accounts for significant speed-up, since the following computer intensive tasks can be performed only within the remaining areas. The second task enables the reconstruction of a well-defined surface and the third task is necessary to create a smooth-looking surface meaningful to the human eye. All three tasks are performed in a multi-resolution framework. After reading the original data, a Gaussian pyramid of levels-of-detail (fig. 3) is initially generated [14]. Such a pyramid requires only 14% additional memory space and is computed within 0.27 additional seconds for a 128^3 and 0.88 additional seconds for a 256^3 field for a kernel of 3^3 on a SUN Sparc20 due to the separability of the Gaussian kernel (see tab. 2).

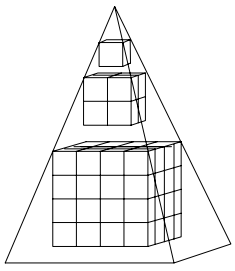


Figure 3: The Gaussian pyramid

4 Segmentation

The main idea for the multi-resolution feature identification and segmentation is that 'real' surfaces posses a higher coherency as compared to speckle or noise. This means that when filtering the data to successive lower levels of detail(corresponding to higher pyramid levels with lower voxel resolution), noise, speckle and minor structures could be expected to disappear faster than useful surfaces due to their lower coherency, whereas 'larger' surfaces should still remain detectable even on lower resolutions. By inverting this idea, after generating a level-of-detail representation of the original data, we first try to identify crucial regions on higher pyramid levels where little or no noise/speckle is expected. Information carrying areas are then propagated (mapped) to higher resolution levels and the process is repeated until the original data level is reached. Regions 'masked out' by this processing are not further examined by any of the subsequent levels. The result of the segmentation processing is a *binary mask defining a region of interest* ROI (since we operate in voxel space, we generate a 3D mask defining a volume of interest VOI). Only within this VOI region are valid contours expected and therefore only these are accessed during volume rendering. The remaining non-valid areas are regarded to be empty.

In a multi-resolution framework, the process theoretically starts with the first node (the root) of the pyramid. In practice, however, only 1 to 3 levels over the highest resolution (pyramid basement) are used. Starting on a certain level n, regions of interest are identified: we tested two different selection methods discussed in the following sections 4.1 and 4.2. Then each valid voxel is projected on the next higher resolution level. Due to the multi-resolution framework, one voxel of level n projects onto 8 voxels of the level $n-1$. Changing the levels, the same procedure is repeated with levels $n-1$ and $n-2$ and so on until the highest resolution level 0 is reached. However, after leaving the starting level n, only voxels falling within the already selected areas are considered, reducing the amount of computing time, a crucial fact with increasing volume resolution.

An implication of each voxel projecting on 8 successors is that the selection procedure is a crucial decision for the success of the method. A voxel misclassified as empty on level n will introduce a gap of 8 voxels on level $n-1$, 64 voxels on level $n-2$ etc. Such gaps introduce serious visible artifacts similar to those shown in fig. 7 left. On the other hand, if the selection criterion is rather relaxed, the effect of noise/speckle reduction will be missed. A possible solution is to start with a rather relaxed criterion and become gradually more restrictive when propagating to higher resolutions.

Instead of filtering the grey values of the voxels $V_n[u,v,w]$, we decided to process the opacity volume $O_n[u,v,w]$ and leave the data itself unchanged. Thus, for every level of the data pyramid we allocate a second opacity field of equal size thereby creating an opacity pyramid and assigning to each element the corresponding opacity value as calculated by Levoy's classificator (eq. 2).

4.1 Mathematical Morphology

Mathematical morphology can be used to implement a wide spectrum of operations on binary images or datasets. The fundamental morphological operations on a binary image P with an element E are *erosion* and *dilation*, as well as their combination *opening*.

$$\begin{aligned} \text{EROSION}(P,E) &:= \{p | E_p \text{ included in } P\} \\ \text{DILATION}(P,E) &:= \{p | E_p \cap P \neq \{\}\} \\ \text{OPENING}(P,E) &:= \text{DILATION}(\text{EROSION}(P,E),E) \end{aligned}$$

The theoretical details are given in [5] and will not be repeated here. In simple words, depending on the shape and size of E, erosion removes structures smaller than a certain size and can be used for removing noise or speckle blobs. Dilation enlarges details larger than a minimum size and can be used to fill up small contour gaps. Opening is an erosion followed by a dilation. All operations can be used in an iterative way, i.e. an already eroded volume can be

eroded again etc. The element for the morphology can have several shapes. The most common candidates are a small cube of 3^3 voxels or a 3D cross containing the central voxel and its 6 face-connected neighbours (see fig. 4).

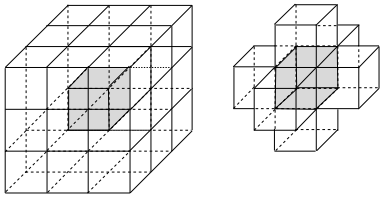


Figure 4: Two possible morphological elements E_{27} and E_7

An additional degree of freedom is to decide how many voxels of E should be occupied in order to select the central voxel as to be valid. For reasons of computational efficiency, we choose to use the cross with 3 occupied voxels on higher and 4 on lower pyramid levels. This means that our selection criterion is more relaxed in the beginning and becomes more restrictive on higher resolution levels.

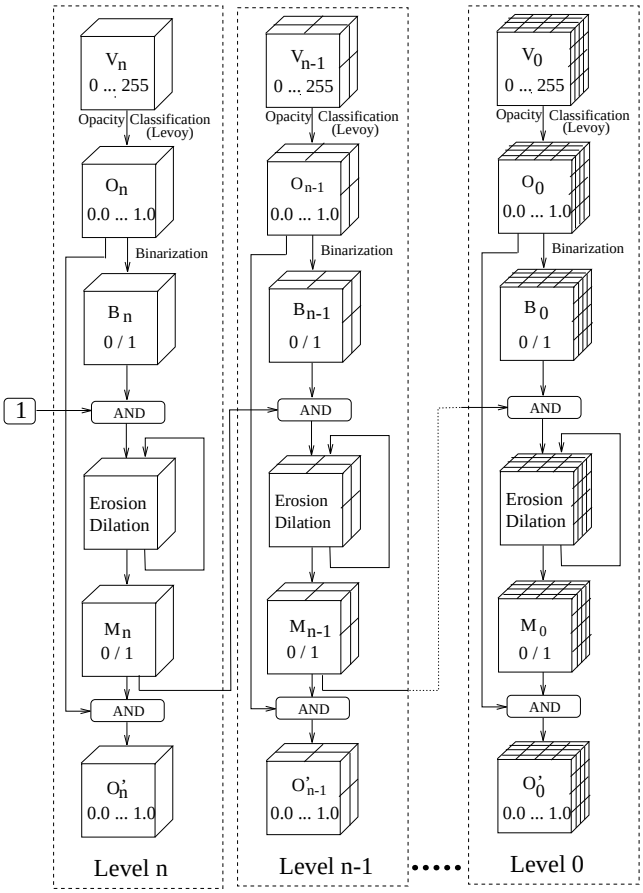


Figure 5: The morphological pipeline

The first action is to calculate for each voxel field $V_n[u, v, w]$ on pyramid level n the corresponding opacity field $O_n[u, v, w]$. For each opacity field we calculate a binary volume $B_n[u, v, w]$ by setting all elements with a non-zero opacity to 1. This binary field is then morphologically processed and the result is the mask $M_n[u, v, w]$, which is then used to mask the local level opacities by means of the AND operation and is propagated to the higher resolutions as shown schematically in fig. 5.

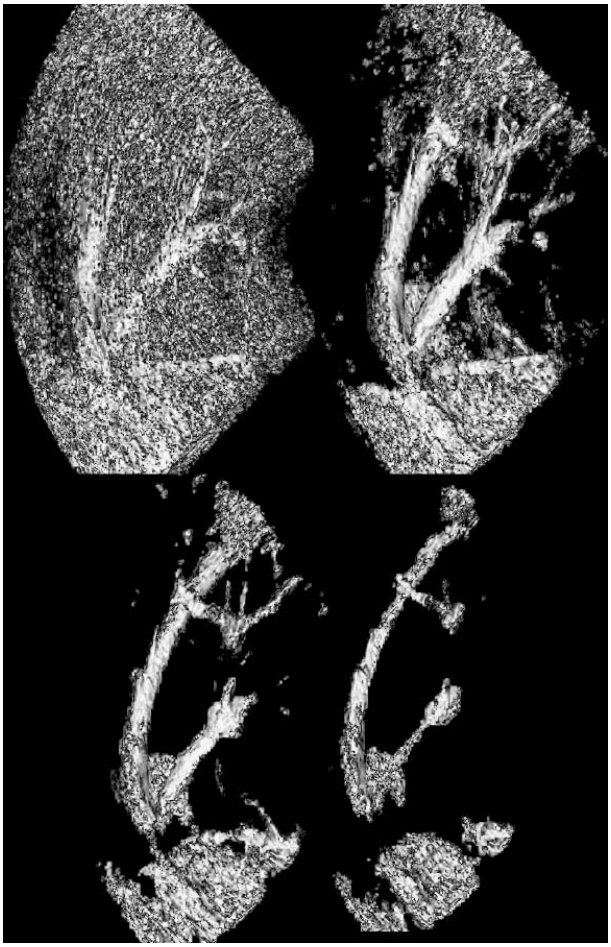


Figure 6: From upper left to lower right: 256^3 dataset showing liver vessels before filtering and processed with erosion after 20, 30 and 40 iterations

Fig. 6 demonstrates the effect of erosion after several iterations The example shows and erosion performed with one level and an element E_6 size of 3. Small and isolated noise/speckle areas have been successfully suppressed but the surface of the objects has also been eroded, producing image artifacts clearly visible on the magnification shown in fig. 7 left. To suppress aliasing of the surfaces we apply the opening operation, i.e. a dilation after the erosion in order to 'fill the surface gaps' created by the erosion. Opening successfully suppresses the surface artifacts, preserving the shape of major structures. A comparison of erosion and opening can be seen in fig. 7. However, more than 20–30 iterations are necessary in order to obtain a good image, resulting in a very time intensive procedure as shown on tab. 2.

To conclude, applying erosion alone is not sufficient, since it removes parts of the useful surface together with speckle and noise, whereas opening gives better results. Both filters require several iterations before providing an acceptable image quality. An additional drawback of using morphological filters lies in the large number of parameters to be adjusted, i.e. Levoy threshold and

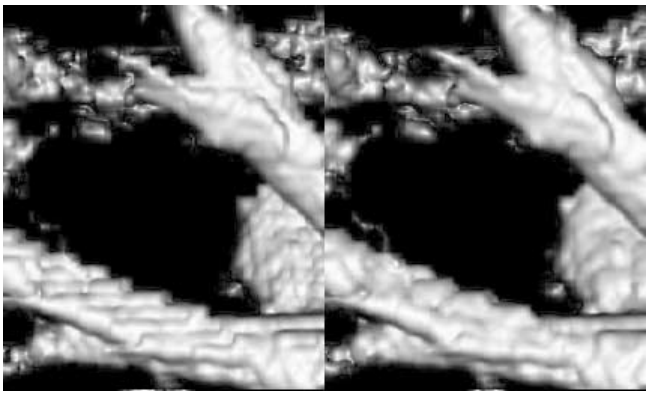


Figure 7: Left a magnification of fig. 6 after 30 erosion iterations showing visible artifacts. Right the same data after 30 opening operations.

tolerance, shape & order of E, and number of iterations for erosion and dilation. This gives a total of over 6 degrees of freedom, making the whole process non-intuitive and impractical for clinical applications.

4.2 BLTP-Method

In order to overcome the drawbacks of the morphological filters in intuiton and computational efficiency, we developed a method we called BLTP meaning *binarize, low-pass, threshold & propagate.* The principle of the filter is better explained in a 1D example shown in fig. 8 and 9, an extension to 3D is straightforward.

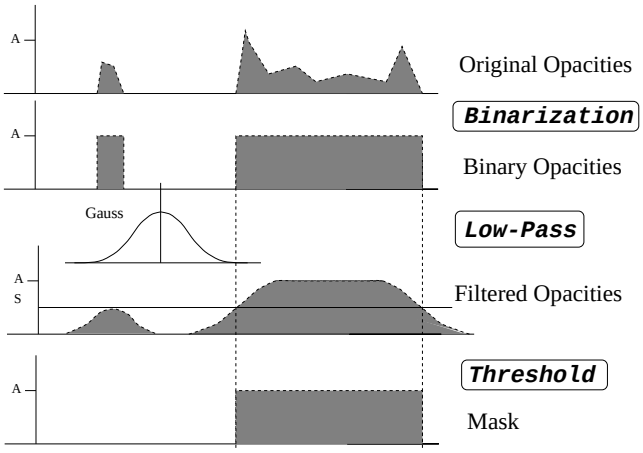


Figure 8: The principle of extracting a VOI mask using BLTP filtering in 1D

Again, the first action is to create for each opacity field $O_n[u,v,w]$ a two-valued *real numbers* volume $D_n[u,v,w]$ by setting all voxels with an non-zero opacity to 1.0, see fig. 10. Small details, speckle and isolated noisy voxels will be converted to small unity steps, whereas large occupied areas will generate more extended structures as demonstrated in fig. 8. Due to the usage of discrete unity steps, the convolution (low-pass filtering) of $D_n[u,v,w]$ with a Gaussian kernel is represented by a discrete

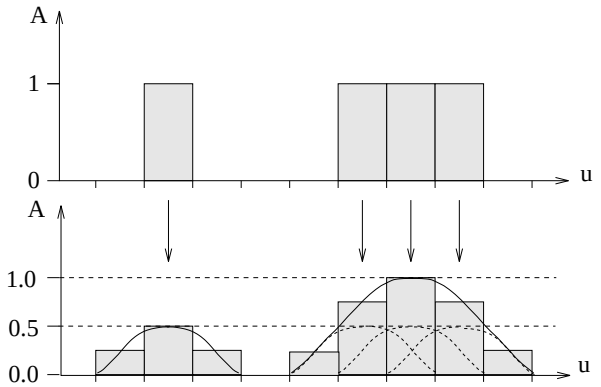


Figure 9: Result of the convolution of unity steps (upper) with a Gaussian kernel of size 3 (lower)

sum of Gaussian kernels, symmetrical to the position of the occupied voxel and accordingly shifted in relation to each other:

$$g'(u) \quad = \quad \sum_{i=-n}^{n} g(u) \cdot \frac{1}{2 \cdot \pi\sigma^2} \cdot e^{\frac{(u-i)^2}{2 \cdot \sigma^2}}$$

where g represents the original grey values, g' the filtered function, and $2n+1$ the size of the Gaussian kernel. The graphical representation of such a convolution applied on structures of different size is shown in fig. 9.

The numerical result of such a filtering for the 1D case can be seen on tab. 1. The structures are symmetrical over the corresponding maximum value; Due to lack of space some 'overflowing' values have been omitted in tab. 1. We call the low-pass filtered field $LP_n[u,v,w]$.

Kernel	3					5				
1 Vox.		.25	**.5**	*.25*		.0625	.25	**.375**	*.25*	*.0625*
2 Vox.		.25	**.75**	*.75*	*.25*	.0625	.3125	**.625**	*.625*	*.3125*
3 Vox.	.25	.75	**1.0**	*.75*	*.25*	.0625	.3125	.6875	**.875**	*.6875*
4 Vox.	.25	.75	**1.0**	*1.0*	*.75*	.0625	.3125	.6875	**.9375**	*.9375*
5 Vox.	.25	.75	**1.0**	*1.0*	*1.0*	.0625	.3125	.6875	.9375	**1.0**

Table 1: 1D low-pass filtering discrete unit jumps of varying size with different Gaussian kernels. The structures are symmetrical over the corresponding maximum value; Due to lack of space some values have been omitted

In simple words, low-pass filtering will change the sharp edges of the unity steps to smoother slopes and change the maximum of small structures to a lower value. The shape of a structure after filtering (inclination of the slope, value of maximum) depends on both the order of the kernel and the size of the initial structure. From tab. 1 one can easily see that filtering a structure of k unit steps width with a kernel of $2n+1$ results in a structure with an extend of $k+2n$ pixels. In addition, if $k \leq 2n$, all values of the result will be below 1.0, and when $k \geq 2n+1$, there will be $k-2n$ coefficients having the value 1.0. This allows filtering of structures up to a given size as explained below.

There are two different possibilities to process the field $LP_n[u,v,w]$. The first arises from the above observation that structures smaller than the kernel size will have all their coefficients smaller than 1.0. Therefore, in order to eliminate structures up to a given structure size k, one has to select a filter with the next greater odd size $2n+1 \geq k$ and threshold the resulting field LP_n, setting all opacities below 1.0 to zero. The binary mask

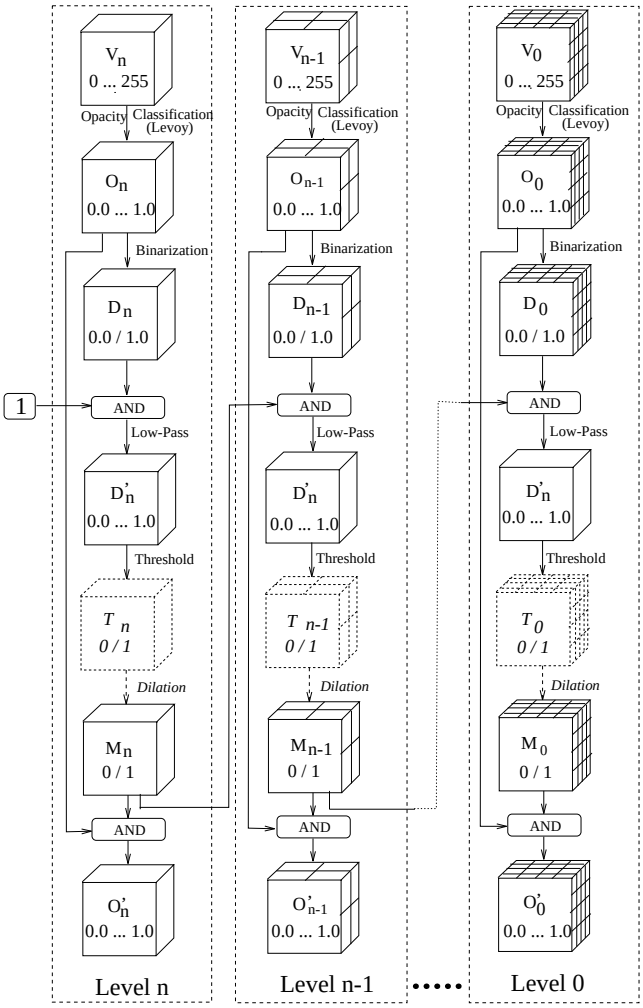


Figure 10: The principle of the BLTP pipeline

$M_n[u, v, w]$ is calculated by dilating the result field after thresholding $T_n[u, v, w]$ by an element E of the size k in order to expand again the remaining 1.0–coefficients to the original extend of the structures[1]. After dilation the resulting mask $M_n[u, v, w]$ is used for masking the opacities of the current level (AND operation) as well as propagating to the next level as shown in fig. 10. The method can effectively and accurately suppress noise and speckle blobs. Due to the fact that thresholding with 1.0 is a trivial operation and dilation is performed only in regions where 1.0–coefficients are presented, this processing is completed quick even with large datasets.

A simplified and faster version of the above algorithm avoids dilation. After choosing the appropriate kernel size and filtering the binary field $D_n[u, v, w]$ in a way similar as above, $LP_n[u, v, w]$ is subsequently filtered to a binary field $T_n[u, v, w]$ by means of a free definable threshold th_s, i.e. values below th_s are mapped to 0 and above th_s to 1. By varying the width k of the kernel and the threshold value of th_s one can easily control the degree of blob suppression. As an example, one can see on tab. 1 that when using a kernel of size 5, a threshold just above 0.375 will eliminate structures of unit size and leave larger ones unchanged, a threshold just above 0.625 will eliminate structures with a size of one or two voxels etc. A graphical representation of this effect can also be seen in fig. 8. The result after thresholding is the binary field $M_n[u, v, w]$ which is used to mask the opacities of level n and propagated to level $n + 1$ as shown schematically[2] in fig. 10. Thus this variant is an approximation of the previous one, using a free adjustable threshold and avoiding the dilation after calculating $T_n[u, v, w]$.

Figure 11: BLTP filtering of a 256^3 liver dataset without dilation with a Gaussian kernel of 5^3 and thresholds increasing from upper left to lower right

The advantage of this variant of the BLTP method as compared with morphological operation lies in its simplicity and computational efficiency. A BLTP re-calculation including thresholding, mask calculation and propagation without low-pass re-calculation requires only 24.8 seconds for a 256^3 pyramid and 8.78 seconds for a 128^3 one (see tab. 2), making the process suitable for interactive purposes. Although the exact size of the eliminated structures is only approximately known, the interactive adjustment of the threshold th_n and the quick feed-back allows a comfortable and intuitive processing of the dataset. The effects of using the BLTP filter can be shown in fig. 11. By varying the threshold th_n, a significant amount of speckle and noise could be reduced whereas the liver vessels remained effectively unchanged. To date, this BLTP variant is

[1] To eliminate structures with size 1 and 2 one has to use an appropriately scaled kernel of size 3

[2] Fig. 10 demonstrates only the principle of the method. For reasons of computational efficiency the C-code implementation includes several differences, e.g. buffering stages are omitted or merged together with filter stages etc.

Volume resolution	Kernel size	Lowpass level 0	Pyramid level 1... n	Opacity field	Dilation 1 iteration	Opening 1 iteration	Threshold, Mask, Project	BLTP pipeline	Morphology 20 iterations
128^3	3^3	1.0	0.27	3.38	3.15	6.54	8.78	10.05	139.53
	5^3	1.2	0.38					10.36	
256^3	3^3	6.51	0.88	20.36	28.21	54.78	24.8	32.19	1099.6
	5^3	12.17	1.24					38.21	

Table 2: CPU times on SUN Sparc 20 for various filtering operations with two different kernel sizes and volume resolutions. Lowpass & pyramid are computed once per dataset, opacity only when changing Levoy's iso-value; these three stages are regaded as offset times since they change very rarily. Morphological operations given for an element $E = 3$. On the right the total processing time for BLTP with variable threshold and opening after 20 iterations.

the one most frequently used.

All three examined methods suppress only blobs of noise or speckle. Other obstacles, such as the ultrasound field near the transducer or the hand of the fetus obscuring the face, show the same characteristics as the 'real surfaces' and therefore must be removed semi-automatically using methods reported in [16].

5 Contour Filtering

After segmenting the regions of interest, our second task was to improve the appearance of the contours within these VOIs. Levoy's opacity formula (eq. 2), being essentially a combination of an iso-surfacing weighted by the local grey level gradient, has been used as starting point for further experiments. We examined two additional edge detectors more closely as an alternative to the gradient proposed by Levoy (see [1] for an excellent survey). In the first case we employed a 3D version of the Sobel-operator instead of the gradient in the denominator of eq. 2. The 2D kernel is:

$$S_x = \frac{1}{8} \cdot \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad S_y = \frac{1}{8} \cdot \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$opacity = a \cdot \begin{cases} 1 : g(u,v,w) = S \\ 1 - \frac{1}{r} \cdot \frac{|g(u,v,w) - S|}{\sqrt{S_x^2 + S_y^2 + S_z^2}} : \sqrt{S_x^2 + S_y^2 + S_z^2} > 0 \\ 0 : otherwise \end{cases}$$

The result can be seen in fig. 12. The differences in the quality as compared to Levoy's original method are rather small, however, the computation time has been almost doubled. In the second case we used a Laplacian-of-Gaussian (LOG) operator. Again, the 2D kernel is:

$$\begin{aligned} L_{u,v} &= \frac{1}{16} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{16} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \\ &= \begin{bmatrix} -1 & -2 & -1 \\ -2 & 12 & -2 \\ -1 & -2 & -1 \end{bmatrix} \end{aligned}$$

$$opacity = a \cdot \begin{cases} 1 : g(u,v,w) = S \\ 1 - \frac{1}{r} \cdot \frac{|g(u,v,w) - S|}{|L_{u,v,w}|} : |L_{u,v,w}| > 0 \\ 0 : otherwise \end{cases}$$

This filter is a combination of a low-pass followed by an edge detector and is expected to give good results even with noisy images. Although the computing time has been doubled again, the result shown on fig. 12 has not been significantly improved as compared to Levoy's original method.

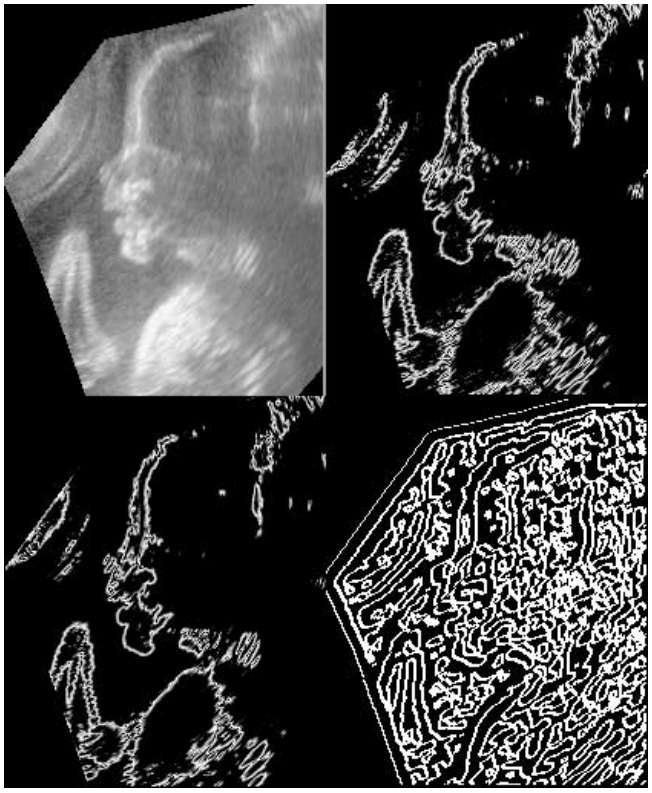


Figure 12: Testing alternative edge detection methods for estimating the opacity. From upper left to lower right: grey image, Levoy, Sobel and LOG edge detection

We implemented several other experiments with the goal to improve the selectivity of Levoy's formula, e.g. by using information for contours and/or grey values from neighbouring resolution levels. All these experiments failed to provide significant improvements and therefore are not reported here. As a result, the original Levoy formula has so far provided the best trade-off between quality and speed.

6 Volume Rendering

In order to volume render the processed datasets we had to apply three modifications to the standard volume rendering pipeline [15]. The first modification is due to the nature of ultrasonic data as

is shown in fig. 13. The ultrasound cone shown on the left is surrounded by empty space, i.e. voxels with zero value. Due to the opacity formula (eq. 2) this causes a very high gradient and thus a high opacity value along the interface to the empty space, which is manifested as a solid 'curtain' obscuring the volume interior. This effect can be totally suppressed by turning every opacity neighbouring an empty voxel to zero: Due to the nature of ultrasonic imaging, voxels with a zero value may occur only in the empty space and not within the data itself. The size of this neighbourhood must be adjusted in accordance with the size of the Gaussian kernel for the higher pyramid levels. For a kernel with a size of $2n+1$ the suppression neighbourhood is:

$$opacity(u,v,w) = \begin{cases} opacity & : \quad g(u+i, v+j, w+k) \neq 0 \\ & \quad \forall \ (i,j,k) \ \in (-n \ldots n) \\ 0 & : \quad otherwise \end{cases}$$

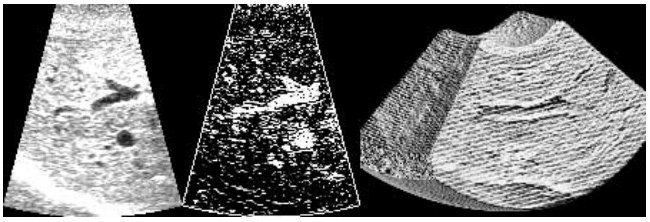


Figure 13: Left a grey image of the liver, middle the corresponding opacity values, right a volume rendered dataset. Note the high opacity values along the interface between data and empty space (middle) causing a solid 'curtain' obscuring the volume interior (right)

The second modification deals with the visual appearance of the reconstructed surface. As mentioned in section 2, Levoy's method used the local gradient as the surface normal, thus being very sensitive to intensity fluctuations typically apparent in ultrasonic data, thereby creating an extremely rough surface. Fig. 14 upper left shows how rough such a surface can be after removing almost 100% of the noise and speckle using BLTP filtering. It is important to remember that all filtering operations only mask out regions not containing coherent edges leaving thereby the original data unchanged. Thus, the small intensity fluctuations remain on the surface and cause large normal vector perturbations resulting in an unrecognisably rough surface. Our solution is to accumulate the opacity values as originally proposed but *to use a normal vector from higher pyramid levels*. These levels contain low-pass filtered data, so that the normal vectors are also less perturbed and change more smoothly. Similarly, one can use normal vectors from a lower level to add 'sharpness' to a smooth surface. By means of a slider the degree of smoothness or sharpness can be changed continuously through (linearly) interpolating the normal vectors from the corresponding pyramid levels.

$$N(u,v,w) = N_n(u,v,w) \cdot (1-|t|) + N_{n+\lceil t \rceil}(u,v,w) \cdot |t| \qquad -1.0 \leq t \leq 1.0 \quad (6)$$

This effect is shown in fig. 14. It is important to note that for all images *exactly the same opacities* have been used; the different visual appearance is caused only by the modification of the normal vector.

The last feature is that we extract both surface as well as volumetric information during a single traversing. Extracted features are depth-sorted and stored together with their individual depth in a 2D structure similar to a ZZ-buffer [17]. After traversing and

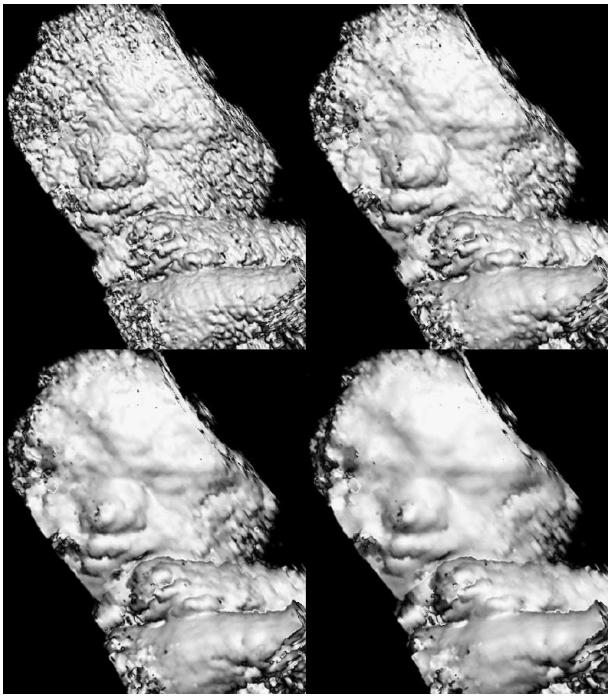


Figure 14: Fetus after removing almost all blobs, before (upper left) and after normal vector smoothing with $t = 0.3, 0.7, 1.0$

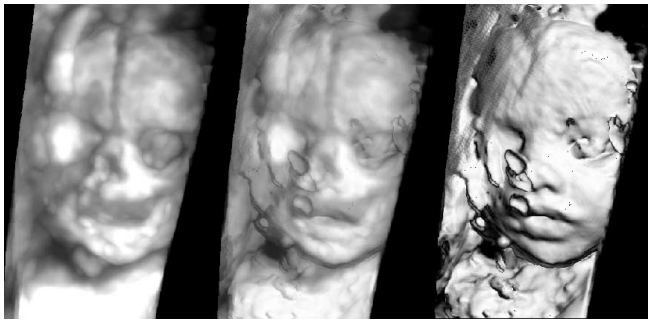


Figure 15: Continuous transition between MIP and surface rendering

during visualisation one can select the type of desired visual representation interactively. Also mixed values (e.g. 40% surface, 60% X-ray) can be displayed as shown in fig. 15. The visualisation evaluates the values of the ZZ-buffer and since it requires no new ray-traversing, it can be performed at rates of several frames per second. The $O(n^2)$ requirements of space allow the pre-calculation and storage of a complete rotation sequence in the main memory. The visual representation can be selected interactively during the play-back of a pre-recorded loop.

7 Results & Further Works

For our tests we used datasets from different devices showing either fetuses around the 25th week of pregnancy (3–4 months before birth, fig. 16) or scans of the liver (fig. 5, 11). More than a dozen

datasets like those presented in fig. 16 have been acquired under routine clinical conditions in the Heidelberg University Hospital, using a 'Kretz Voluson 530' device. The typical resolution in a Cartesian coordinate system is 10 Mvoxels, the maximum resolution 16 Mvoxels (256^3) . Using the methods reported here and in [16], we succeeded in visualizing *all* of the provided datasets. The time required for the complete processing of each set was less than five minutes and included loading from the disc, filtering, segmenting, smoothing, visualization and loop generation. The first results from the clinical routine have been very encouraging indicating clinical evidence at least in the area of the fetal diagnosis [20]. Fig. 17 compares an image reconstructed from data acquired in the 25th pregnancy week with a photo of the baby 24 hours after birth. The time for volume rendering one image with resolution of 512^2 pixels is about 10 seconds on an IBM R6000, a rotation sequence required only 3 minutes. Diagnosis of abnormal cases can be definitely improved. Tab. 3 presents the results for volume rendering a field of 4 MVoxels in a resolution of 300^2 pixels using various hardware platforms.

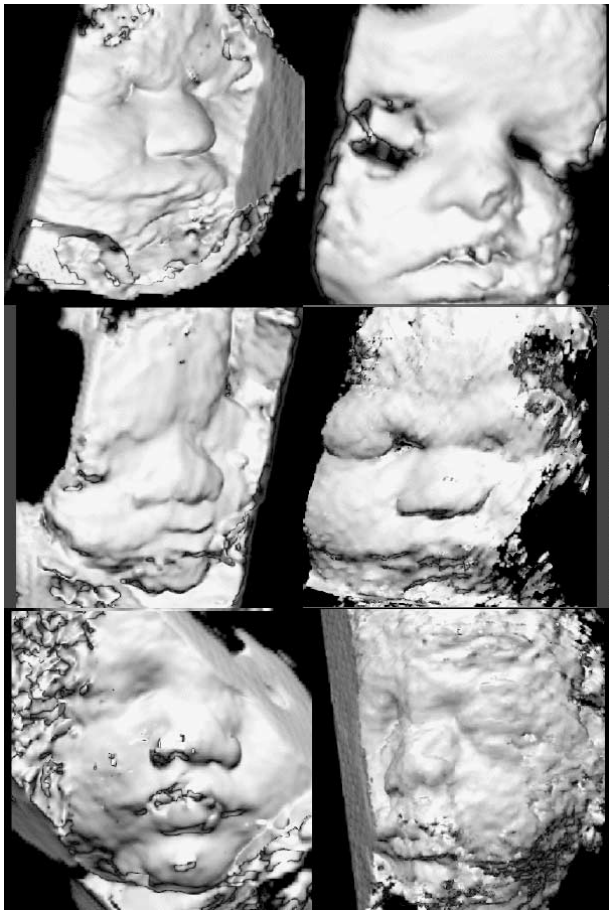


Figure 16: Different fetal faces 3–4 months before birth

The major advantages of our method in visualizing ultrasonic data are:

- Understanding the content of 3D ultrasound images has been significantly improved. Our visualisation methods provide a true 'added value' for diagnosis.
- Noise and speckle included in the 3D ultrasound data has been significantly reduced without altering the original data.

Figure 17: Left data of the 25th pregnancy week, right a photograph 24 hours after birth

Hardware Platform	Surface Vol. Rendering			Surface & MIP Vol. Rendering		
	Standard	High	Highest	Standard	High	Highest
SUN Sparc10	5.1	11.7	13.6	5.3	12.2	22
SUN Sparc20	2.3	7.8	9.6	3.6	6.8	14.2
SGI Indigo	2.5	7.7	8.4	2.9	5.9	11.0
IBM R6000	1.9	5.3	5.5	2.5	4.9	9.1
486 40 MHz	10.9	27.5	44.6	16,5	42.4	68.9
586 90 MHz	1.8	4.4	5.9	3.1	5.4	11.0

Table 3: Runtimes in CPU seconds for volume rendering a dataset of $256^3 \times 64$ voxels creating both surface and transparent (MIP) images at a resolution of 300^2 pixels at three different image quality levels

- The type and degree of noise/speckle reduction, surface smoothing, visual representation, etc., can be interactively selected. The original data remains thereby unchanged and can be referred to at all times. This allows the user (physician) to control the degree of altering the original data.
- The speed of visualisation enables an immediate evaluation of the acquired data during examination of the patient.
- The system runs on common commercial workstations including PCs and needs no special hardware.

In addition, the BLTP filtering is clearly the method of choice because it provides results similar to mathematical morphology but runs at least one order of magnitude faster, is more intuitive and easy-to-use and allows interactive manipulation.

After several years of research we recognise that the work just started. The most important conclusion from this work is that we showed that global filtering provides good results but is unacceptably slow for practical applications. On the other hand, VOI identification and local, possibly adaptive, filtering provides even better results within interactive times. Advanced filtering that adapts to the local data features in a multi-resolution framework is the most important goal for future work.

First of all, the BLTP method should be further extended. Instead of using a Gaussian low-pass filter, we should consider the possibility of employing more suitable band-limiting filters, enab-

ling an elimination of structures of a specific size. A rich amount of literature exists on this topic. As a very promising avenue of research we are currently investigating the possibility of using wavelets as an alternative for BLTP filtering. Wavelets offer very attractive possibilities, because they combine highly desired signal processing features with the multi-resolution framework employed here. Furthermore, one could also investigate the usage of 3D-correlation functions, most likely implemented in the frequency domain, for identification of surfaces. Both these methods require substantial computing power and have therefore not been considered yet for the desired interactive low-cost system. However, processing times can be reduced when working in the frequency domain by using signal processors, which makes the additional hardware costs rather reasonable. Different possibilities employ filters for edge-preserving resolution reduction, possibly combined with local speckle removal. We expect that such filters may provide better results than the Gaussian kernels employed here.

A remarkable piece of work is presented in [18] claiming excellent results in both precision and robustness for extracting edges from noisy images. The author also uses a pyramid of Gaussian filtered 2D images to recognise edges in a multi-resolution framework. In addition he provides a very good review of the theory and mathematics proofing of the applicability and superior performance of his method. The main idea there is to calculate edges on different scales using the LOG operator and then multiply the corresponding magnitudes. Major edges will thereby be amplified whereas minor edges will be attenuated below a given threshold. This shows similarities to our concept of mask propagation along resolution levels. An additional step eliminates 'false positives', i.e. points not surrounded by edges of similar strength along the direction perpendicular to the local gradient. This reflects the idea that edges must show a continuity perpendicular to their surface and shows similarities to morphological operators. Although the principle ideas are similar, the two approaches also show substantial differences. Schunck's method works without resolution reduction, a fact introducing a memory space explosion not tractable in 3D imaging. Secondly, he detects edges whereas we identify areas (VOI). Thirdly, the calculation and multiplication of edge magnitudes at different scales is extremely computation intensive whereas we restrict the calculations within the areas masked from the lower resolution. Nevertheless, a combination of our approach with his advanced and robust edge estimators seems possible and promising.

Acknowledgements Thanks to Dr. Meindl from the DKD, W. Duda and K. Holle from Kretz Ultrasound for providing clinical data, as well as to Prof. Wischnik and Dr. Hiltmann from the Universitäts-Frauenklinik Heidelberg for medical advice and for being the first users of the system.

REFERENCES

[1] Antoniu, E: *Trends in Edge Detection Techniques*, EUROGRAPHICS'91 State of the Art Report, pp. 111-162, Vol. EG 91 STAR, ISSN 1017-4656, 1991

[2] Burt, P.: *The Pyramid as a Structure for Efficient Computation*, A. Rosenfeld (Ed.), Multi-Resolution Image Processing and Analysis, Springer Verlag, New York, 1984

[3] Fuchs, H., Kedem, Z., Uselton, S.: *Optimal Surface reconstruction from Planar Contours*, Comm. of the ACM, Vol. 20, No. 10, pp. 693-702, 1977

[4] Gallagher, R., Nagtegaal, J.: *An Efficient 3D Visualization Technique for Finite Elements*, Computer Graphics, Vol. 23, No. 3, pp. 185-194, July 1989

[5] C.R. Giardina, E.R. Dougherty: *Morphological Methods in Image and Signal Processing*, Prentice Hall, Englewood Cliffs, New Jersey, 1988

[6] Gonzalez, R. C., Wintz, P.: *Digital Image Processing*, Second Edition, Addison Wesley Publishing Company, 1987

[7] Herman, G., Liu, H.: *3D Display of Human Organs from Computer Tomograms*, Computer Graphics and Image Processing, Vol. 9, No. 1, pp. 1-21, January 1991

[8] Levoy, M.: *Display of Surfaces from Volume Data*, IEEE Computer Graphics & Applications, Vol. 8, No. 3, pp. 29-37, May 1988

[9] Lorensen, W., Cline, H.: *Marching Cubes: A High Resolution 3D Surface Construction Algorithm*, ACM Computer Graphics, SIGGRAPH-87, Vol. 21, No. 4, pp. 163-169, July 1987

[10] S. Mallat and S. Zhong, *Characterization of Signals from Multiscale Edges*, IEEE Transactions on Pattern Analysis and Machine Inteligence, Vol. 14, pp. 710-732, 1992

[11] Millner, R.: *Ultraschalltechnik, Grundlagen und Anwendungen*, Physik Verlag, Weinheim, ISBN 3-87664-106-3, 1987

[12] Nelson, T., Elvis, T.: *Visualization of 3D Ultrasound Data* IEEE Computer Graphics & Applications, Vol. 13, No. 6, pp. 50-57, November 1993

[13] Sabella, P.: *A Rendering Algorithm for Visualizing 3D Scalar Fields*, ACM Computer Graphics, Vol. 22, No. 4, pp. 51-58, August 1988

[14] Sakas, G., Gerth, M.: *Sampling and Anti-Aliasing Discrete 3D Volume Density Textures, EUROGRAPHICS'91 Award Paper*, Computer and Graphics, Vol. 16, No. 1, pp. 121-134, Pergamon Press, 1992

[15] Sakas, G.: *Interactive Volume Rendering of Large Fields,* 'The Visual Computer, Vol. 9, No. 8, pp. 425-438, August 1993

[16] Sakas, G., Schreyer, L., Grimm, M.: *Pre-Processing, Segmenting and Volume Rendering 3D Ultrasonic Data,* IEEE CG & A, Special Issue, Vol. 15, No. 4, July 1995. Revised version of *Case Study: Visualization of 3D Ultrasonic Data, "*Outstanding Case Study Award", Proceedings Visualization'94, pp. 369-373

[17] Salesin, D., Stolfi, J.: *The ZZ-Buffer: A Simple and Efficient Rendering Algorithm with Reliable Antialiazing,* Proceedings 'PIXIM'89 Conference', pp. 451-466, Hermes Editions, Paris, September 1989

[18] B. G. Schunk : *Edge detection with Gaussian Filters at Multiple Scales of Resolution*, in: 'Advances in Image Analysis', Y. Mahdavieh and R. C. Gonzales (Eds.), McGraw Hill 1987

[19] Stata, A., Chen, D., Tector, C., Brandl, A., Chen, H., Ohbuchi, R., Bajura, M., Fuchs, A.: *Case Study: Observing a Volume Rendered Fetus within a Pregnant Patient,* Proceedings Visualization'94, pp. 364-368, 1994

[20] Sakas, G., Walter, S., Hiltmann, W., Wischnik, A.: *Foetal Visualization Using 3D Ultrasonic Data*, Proceedings CAR'94, Berlin/Germany, 21-24 June 1995

[21] Sohn, C., Zoller, W. (Eds.): *Proceedings II Symposium 3D Ultraschalldiagnostik*, Munich, 25-26 June 1993, 'Imaging - Application & Results', Vol. 61, No. 2, pp. 81-135, Karger Pub., June 1994