3D ultrasound examination of large organs

G.M. Treece, R.W. Prager,
A.H. Gee and L. Berman

CUED/F-INFENG/TR 367

December 1999

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England

E-mail: gmt11,rwp,ahg@eng.cam.ac.uk, lb@radiol.cam.ac.uk

Abstract

Freehand 3D ultrasound is particularly appropriate for the measurement of organ volumes. For small organs, which can be fully examined with a single sweep of the ultrasound probe, the results are known to be much more accurate than those using conventional 2D ultrasound. However, large or complex shaped organs are difficult to quantify in this manner because multiple sweeps are required to cover the entire organ. Typically, there are significant registration errors between the various sweeps, which generate artifacts in an interpolated voxel array, making segmentation of the organ very difficult. This report describes how *sequential* freehand 3D ultrasound can be used to measure the volume of large organs without the need for an interpolated voxel array. The method is robust to registration errors and sweep orientation, as demonstrated in simulation and also using *in vivo* scans of a human liver, where a volume measurement precision of $\pm5\%$ is achieved.

Contents

1 Introduction

1.1 Motivation

Accurate measurement of organ volume is an application where three-dimensional (3D) ultrasound can provide a real benefit. Volume measurement is important in several anatomical areas, for instance the heart [12], foetus [7], placenta [14], kidney [11], prostate [1], bladder and eye [6]. Traditionally, volumes have been estimated with 2D ultrasound, but it is generally accepted that 3D ultrasound can provide much greater accuracy, although it is also acknowledged that more needs to be done to make such methods clinically acceptable [26]. Greater measurement accuracy can enable the tracking of smaller volume changes, particularly important in assessing the progress of tumours under treatment. Volume measurement by 3D ultrasound is approaching the accuracy only previously attainable through Computed Tomography (CT) or Magnetic Resonance Imaging (MRI). This is of practical benefit, since ultrasound is less expensive, more widely available and, in comparison to CT, less damaging to the patient. However, it has generally not been possible to use 3D ultrasound to examine anatomy which is larger then the width of a single B-scan, since this would require multiple sweeps to cover the entire organ.

There are several cases where the anatomy is too large, or the shape is too awkward, or the direction of view is too restricted, to scan the entire volume in a single sweep. For instance the foetus beyond mid-term is often too large or awkwardly positioned to scan in this way. Tumours, too, can grow to be larger than will fit into a single B-scan, or if they are located beneath the lower ribs (e.g. in the liver) can be impossible to scan in one motion of the probe. The liver itself is also in this category, though there is less clinical justification for being able to measure the volume of the liver *accurately*[1]. The liver does, however, provide an excellent test case for using ultrasound in this context, since it is large, has a relatively complex shape, and its position beneath the lower ribs constrains the direction of insonification.

There have been several attempts in the past to use a primitive form of 3D ultrasound — where the position of each B-scan is recorded manually — to measure liver volume [4, 5, 9, 20]. The earliest of these used a freehand scanning technique [20], although the accuracy of the volume measurements was limited by the large number of hand calculations and inaccurate B-scan localisation. The remainder used parallel transverse slices, with which it is difficult to examine the entire liver. In all cases, it was not possible to include the entire liver cross-section in all of the B-scans — approximations to the shape of missing sections were required. It is only recently that freehand scanning techniques have again been applied to liver volume measurement [13] and foetal liver volumes [25].

In addition to providing accurate measurements, to be clinically useful, it is crucial that any volume measurement system is also:

- relatively fast — ideally the assessment should be performed with the patient still present, as this enables repeat scanning should this be found to be necessary after processing the data.
- easy to understand and use.
- transparent — it is vital in a clinical setting that any artifacts in the data due to the

[1] Assessment by palpation of the *approximate* liver volume, by contrast, is a standard part of an abdominal examination.

acquisition and display alone are clearly identifiable as such and do not detract from, nor are confused with, the 'real' data.

1.2 Existing techniques

Systems which allow the construction and visualisation of 3D data from ultrasound have been widely investigated in the last two decades [8, 17, 30]. Such systems can be categorised into *freehand*, where the clinician is free to move the probe in any manner, and *mechanical* systems. Mechanical systems in general consist of a probe assembly which moves the probe along a pre-defined path, such that a volume is scanned, the whole assembly being held fixed during acquisition [17]. The size of volume is therefore restricted by what can physically be held in good acoustic contact with the skin surface. In order to acquire larger volumes, it would be necessary to attach a position sensor to the whole assembly, and acquire in several positions — to our knowledge, no such systems are currently available.

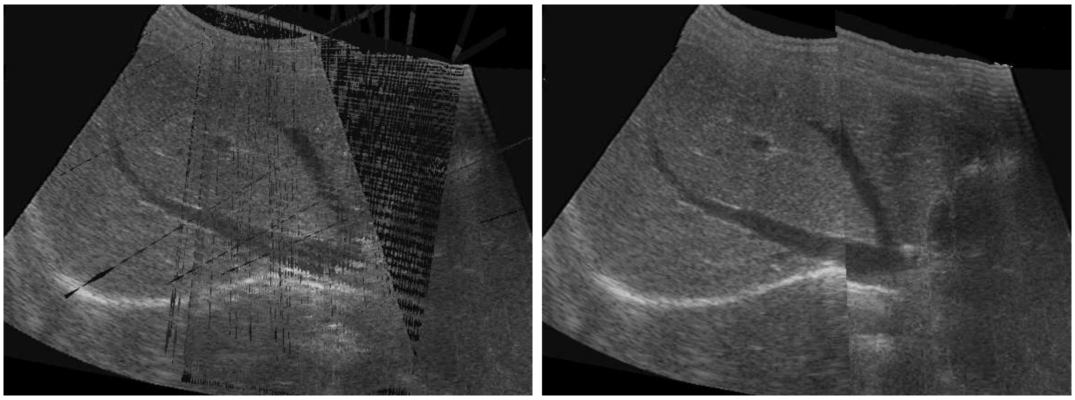
The restriction in acquired volume also applies to 2D transducer arrays [16], for much the same reason as for mechanically located probe assemblies.

1.3 Freehand 3D ultrasound

Freehand 3D ultrasound preserves one of the inherent advantages of ultrasound over CT and MRI; namely, the ability for the clinician to move the probe in an unrestricted manner — the probe position is sensed remotely, often by use of a magnetic position sensor [18]. Hence there is no inherent limit on the size of the volume that can be scanned. Both this and the flexibility of insonification direction make freehand 3D ultrasound an ideal technique for scanning large organs such as the liver. Since the data generated from such scans is irregular, it is generally first interpolated to a regular voxel array, in order to reduce the complexity of further processing. There are several techniques which can successfully generate this voxel array from single sweeps [24]. However, most of these techniques are very simple (in order to process the large amount of data in an acceptable time), and if applied unaltered to data which is from multiple sweeps of the probe, can produce undesirable results. Figure 1(a), for instance, shows two sweeps from a liver examination which have been interpolated using the *voxel nearest neighbour* [24] interpolation. The poor image quality is due to a combination of treating the black regions around each B-scan as 'real' data, and misregistration of the data itself.

Both these problems become much more apparent when using multiple sweeps — in particular, misregistration is much worse between data sweeps than within them. This is primarily a result of movement of the organ under examination with respect to the position reference frame, due to varying probe pressure from one sweep to the next. This misregistration has been studied in the context of spatial compounding (a technique to reduce the noise in ultrasound images) [3, 22, 23]. The data from each sweep must be registered before it can be compounded, which is a time consuming and often poorly constrained problem.

Combining data from multiple ultrasound sweeps has also been successfully achieved for use in an ultrasound simulator [2]. Creating a good data set for simulation is, however, a somewhat different problem to that of examining a patient. The techniques of image warping and blending used to register the data are both time consuming and inappropriate for use in an unsupervised clinical context, since the warping algorithm can make changes to the data which are not justified by the physics of the acquisition. The simplest, and possibly most



(a) Nearest neighbour on multiple sweeps (b) Also using dividing planes

Figure 1: **Interpolation from multiple sweeps.** Performing simple nearest neighbour interpolation on multiple sweeps generates results as in (a). (b) is the result of using additional planes to separate the sweeps.

appropriate, way to combine data from multiple sweeps in a clinical context is to use only one sweep in any one part of the interpolated volume, as in Figure 1(b). The data from each sweep can now be clearly seen; however, it is still very difficult to segment from these interpolated images, due to the registration artifacts — for instance at the posterior edge of the liver.

1.4 Sequential freehand 3D ultrasound

Recently, we have proposed an alternative approach to freehand 3D ultrasound, which bypasses the voxel array stage. In *sequential* freehand 3D ultrasound, the data is visualised and analysed directly from the original B-scans and positions — see Figure 2. Any-plane slicing, non-planar slicing, panoramic imaging, volume estimation and surface rendering can be performed without the use of an intermediate voxel representation [10, 18, 19, 28]. The sequential approach offers several advantages:

- When reslicing, the data is resampled only once, from the B-scan pixels to the slice pixels. The conventional approach requires two resampling stages, from the B-scan pixels to the voxel array, then from the voxel array to the slice pixels. Since resampling usually involves data approximation, more accurate visualisation is possible by avoiding one resampling process.
- Reslicing can be performed at the full resolution of the B-scans, without the significant memory overhead of a high resolution voxel array.
- Visualisation and data analysis can be performed in real time, as the data is being acquired, since the sequential visualisation and volume measurement algorithms reference each B-scan only once, in the order in which they are acquired.

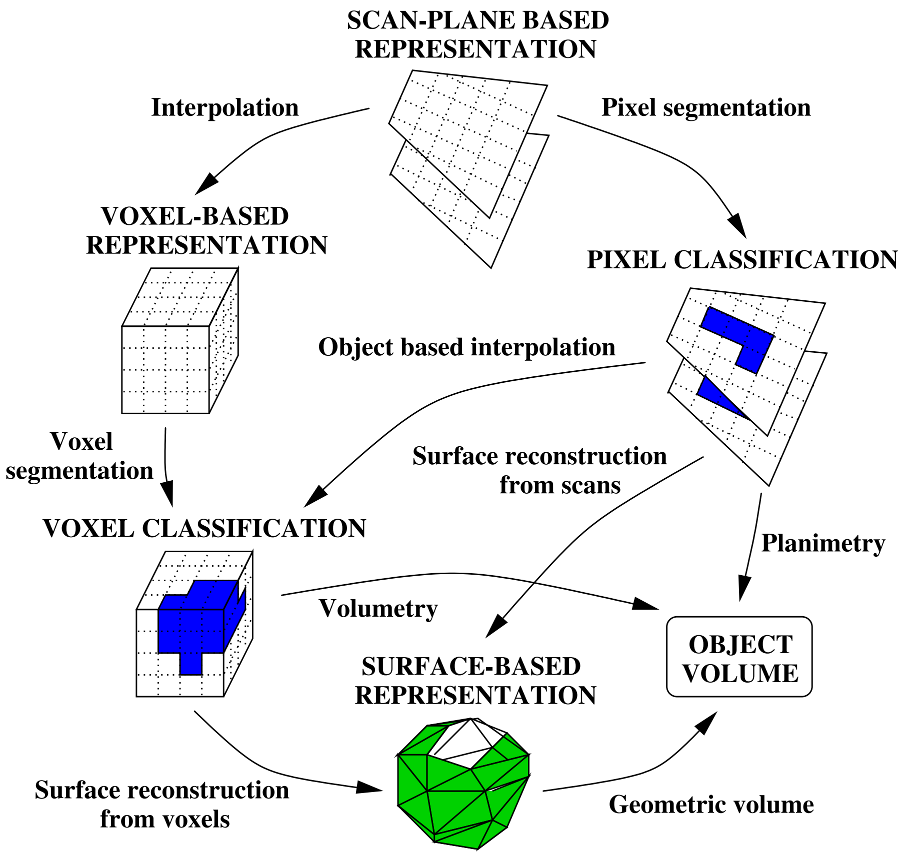


Figure 2: **Volume and surface estimation in sequential freehand 3D ultrasound.** Conventional freehand 3D ultrasound systems adopt the left hand route, through the voxel-based representation. The *sequential* approach avoids this by performing pixel classification in the original B-scans, from which volumes and surfaces can be estimated directly.

- Segmentation (for volume measurement and surface rendering) is performed on the B-scans themselves, instead of parallel slices through the voxel array. The B-scans are high resolution and exhibit no reconstruction artifacts, making them relatively easy to interpret for manual or assisted segmentation. The same cannot be said of slices through the voxel array.

1.5 Original contribution

In this paper, we present an extension of the sequential paradigm for freehand ultrasound data acquired from multiple sweeps. We adopt a simple framework which does not require registration of the data, yet is capable of faithful reslices and accurate volume measurements. Segmentation of the data, which is required for measuring volume accurately, is performed in the original B-scans — this is much simpler for the clinician than segmenting interpolated data, especially in the presence of interpolation artifacts like those in Figure 1(a), or registration artifacts like those in Figure 1(b). Only a handful of segmentations are required to give an accurate volume estimate. The accuracy of the method is verified mathematically and by simulation, and tested *in vivo* by repeated observations of two subjects' livers. The method is, however, generic and can be used for any large or awkward-to-scan object.

So far as we are aware, only one other freehand 3D ultrasound system exists which uses data from multiple sweeps in order to calculate organ volume, without constructing a voxel array [13]. In this system, points at the edge of the organ are selected from the original B-scans, and a piecewise smooth subdivision surface is fitted to these points, subject to some smoothness criterion. The volume can then be estimated from this surface. Our approach has several advantages over this system:

- The method of combining data from multiple sweeps allows reslicing of the data in addition to volume measurement and surface extraction.
- The volume estimation method can calculate volume directly from the cross-sections (there is no need to extract the organ surface), fast enough to be updated immediately as the cross-sections are drawn.
- Our method has also been proven to give accurate results with only a handful of segmented cross-sections [28], a result which is confirmed for multiple sweep data here.
- The surface reconstruction method is a *sequential interpolation* technique which is guaranteed to pass through the actual cross-sections, rather than approximating to them, and places no constraint on the topology of the surface. This has the advantage of remaining faithful to the data, but the disadvantage of looking less pretty, since there is no smoothness constraint on the surface.

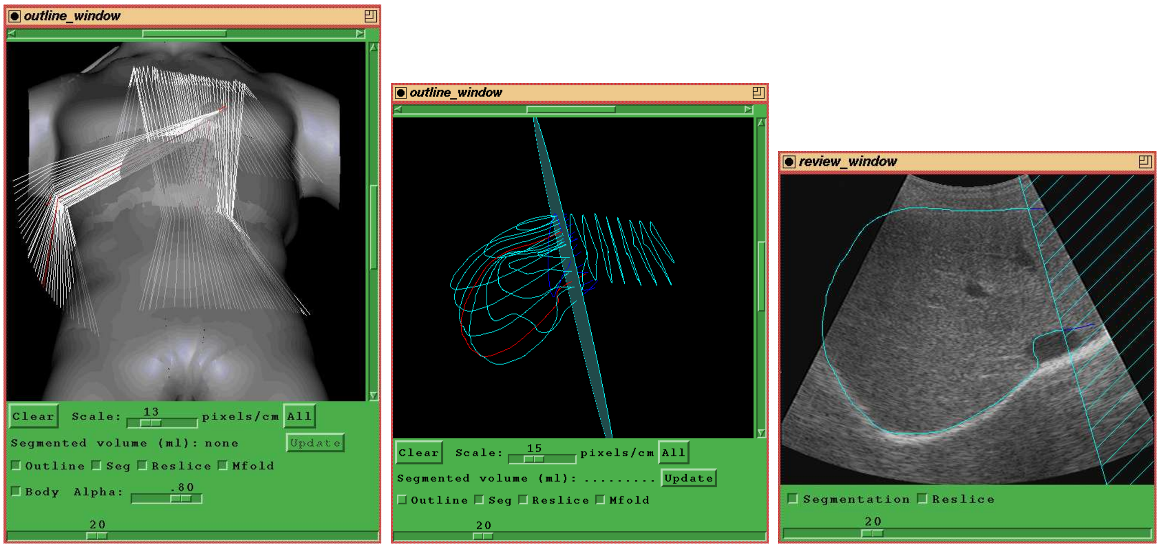
2 Method

In-vivo ultrasound data was recorded using a Toshiba Powervision 7000[2] with a 3.75MHz convex curvilinear array probe. A Polhemus FASTRAK[3] magnetic field position sensor was mounted on this probe. The position signal from this, in addition to the video output of the ultrasound machine, were then fed to a Silicon Graphics Indy workstation[4] (roughly equivalent in processing power to a 200Mhz Intel Pentium II) with a video acquisition card. Acquisition of the ultrasound images and position readings, calibration of the system, and all segmentation and processing of the data were performed using our own software, Stradx, v5.4 of which is freely available on the internet[5]. A future release will include all the functionality described in this paper. The calibration process gave accuracies of typically ±1mm in all directions.

To assess the *in vivo* precision of the system, ten examinations were performed on the livers of each of two healthy subjects. For each subject, five of these examinations involved two ultrasound sweeps, as in Figure 3(a), and the other five involved three sweeps as in Figure 4(a). The actual orientation of these sweeps varied somewhat between examinations. Each complete examination was performed in a single breath hold (roughly 20 seconds).

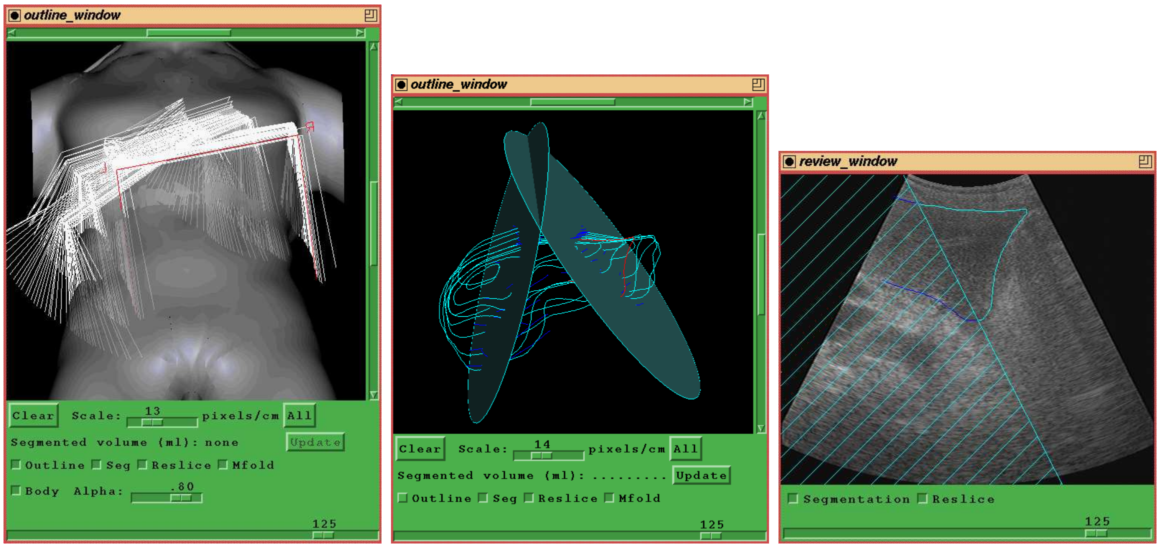
Simulated data was generated by 'scanning' mathematically defined shapes for which the volume was known precisely, then processed in the same way as the *in vivo* data. These scans

[2] Toshiba America Medical Systems, Tustin, California
[3] Polhemus Incorporated, Colchester, Vermont
[4] Silicon Graphics Incorporated, Mountain View, California
[5] `http://svr-www.eng.cam.ac.uk/~rwp/stradx/`



(a) Orientation of B-scans (b) Segmentation (c) Typical B-scan

Figure 3: **Liver examination using two sweeps.** The viewing direction in (b) is the same as in (a). The position of the dividing plane shown in (b) is based on the sweep orientation. Cross-hatching in the B-scan, as in (c), indicates areas which are better covered by a B-scan from another sweep. One longitudinal and one horizontal sweep is used.



(a) Orientation of B-scans (b) Segmentation (c) Typical B-scan

Figure 4: **Liver examination using three sweeps.** The figures and orientation are as in Figure 3. In this case, the liver is covered by three overlapping longitudinal sweeps.

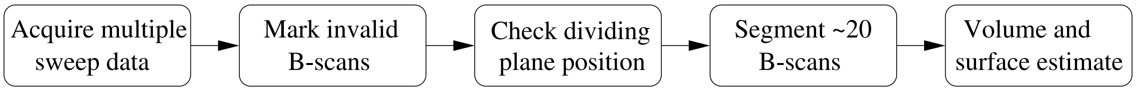


Figure 5: **Volume measurement from multiple sweeps.** The multiple sweep data is acquired and invalid B-scans (i.e. those between sweeps) marked. Dividing planes (which separate the sweep data) are calculated automatically, but can be manually adjusted if necessary. Cross-sections are then outlined in the original B-scans — only a handful are required to give an accurate volume estimate.

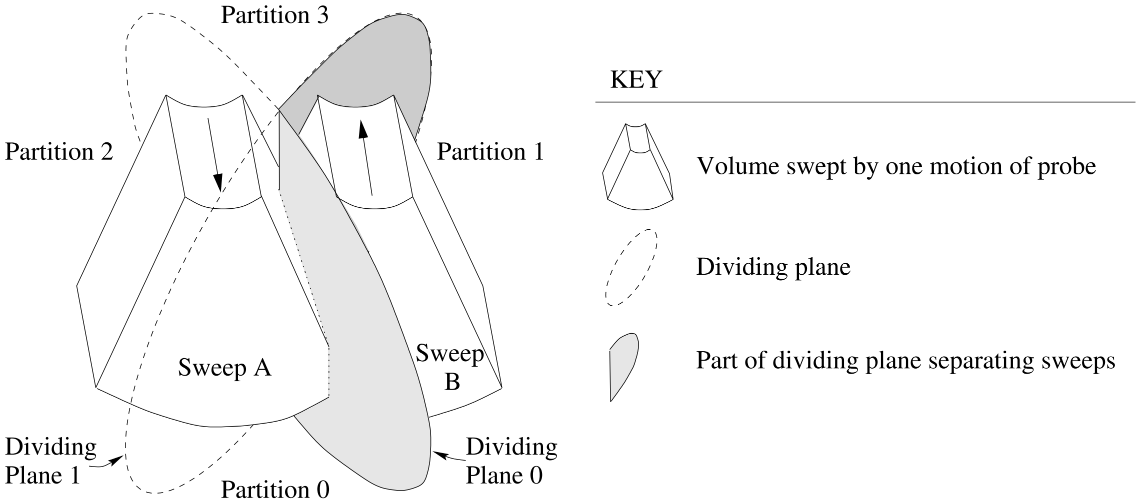


Figure 6: **Definition of terms for multiple sweeps.** Sweep A and B are separated by two *dividing planes* 0 and 1. These planes form four *partitions* of space $0 \ldots 3$. In this case sweep A is used in partitions 0, 2 and 3, sweep B is used in partition 1.

could be generated with various sweep patterns and orientations, and both hand unsteadiness and registration errors could also be modelled. Registration and segmentation are the most dominant source of error in all freehand 3D ultrasound systems — the ability to eliminate them from experiments allows an assessment of the accuracy inherent to the volume measurement algorithm itself.

The entire process of volume measurement is outlined in Figure 5. Once the ultrasound (or simulated) data had been acquired, the orientation of the B-scans could be displayed in an 'outline' window, as in Figures 3(a) and 4(a). These scans have been registered to a computer generated manikin. The clinician can then review the scans in a 'review' window, as in Figures 3(c) and 4(c). B-scans which are in between the valid sweeps, for instance where the probe has been lifted from the skin surface temporarily, are then marked. Each sweep of data is thus defined as a contiguous set of B-scans separated by scans which have been marked as invalid.

Once the sweeps have been defined, a set of 'dividing planes' is automatically generated to partition space such that only one sweep of data is used in each partition — see Figure 6 for a definition of these terms. Dividing planes are displayed as discs in the 'outline' window, Figures 3(b) and 4(b), and can also be edited manually if required.

The dividing planes having been defined, the clinician can then segment the area of interest in the 'review' window. Any region which is not in a partition of space for which its sweep is being used, is cross-hatched. The result of this is that the clinician can clearly see which data is relevant, and which data is better covered by another B-scan. Segmentation can then be performed manually using a mouse. Segments must either be closed or end in cross-hatched regions, which is easily ensured by automatically closing a contour if either the start or end point is outside a cross-hatched region. Typical B-scans with cross-hatching and open contours are shown in Figures 3(c) and 4(c), together with the result of a complete segmentation (roughly 20 cross-sections) in (b).

Volume measurement is performed from these cross-sections *in each partition of space*, where if necessary incomplete cross-sections are closed by the dividing planes defining the partition. The total volume is the sum of the absolute volumes in each partition. Surface reconstruction is also performed in each partition of space, and each surface is clipped to the dividing planes which make up that partition. The total surface (which is not guaranteed to be closed) is the union of these surfaces. Volume can also be estimated from each surface, provided that it is only clipped by a maximum of two dividing planes (which is usually the case in practice).

Each step in this process is outlined in more detail in Section 3. It is fundamental to this system that the apparent complexity is hidden from the clinician behind a carefully designed user interface. The important features of this interface are outlined in Section 4.

3 System details

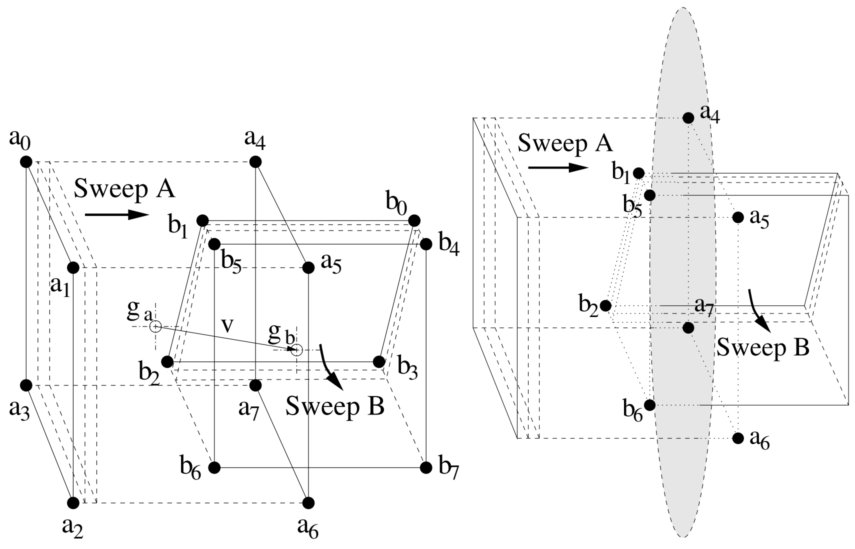
3.1 Definition of sweeps

The only restriction on the scanning pattern for each of the sweeps is that the sweep must pass into and out of the object under investigation, and the side of the scan plane which first enters the object must also be the first side to leave it. This means that the freehand nature of the acquisition process during each sweep is preserved, in that the B-scans can have any orientation and spacing, and can even be overlapping. Currently, multiple sweeps are recorded in one sequence, and the number of sweeps determined after recording by marking B-scans in-between each valid sweep. This has the advantage of enabling recording during a single breath hold. However, the sweeps could equally be separated by pausing the acquisition process between each sweep, which would then make the marking of invalid B-scans unnecessary, although it is likely that the examination could not then be completed in a single breath hold.

3.2 Positioning of dividing planes

Initially, one dividing plane is placed between each pair of sweeps (redundant planes are removed later), up to a maximum of 32. This limit allows each partition of space to be labelled with an integer, where bit i defines which side of dividing plane i that partition exists in[6]. In most practical situations only one to three planes are required to separate the data — the worst case processing time is exponential in the number of planes.

[6]There is an inherent redundancy in this representation, since for more than three planes, it is not possible to partition space into 2^n partitions, where n is the number of planes. However, this simple representation improves the calculation speed for the usual case of only a few planes.



(a) Labelling of sweep corner points (b) Position of dividing plane

Figure 7: **Calculation of dividing plane position.** Sweep A is a linear sweep, and B is a fan sweep. (a) The corner points of the extreme scans and the average centre point for each sweep are calculated. (b) The dividing plane is positioned at the average location of the two sweep sides which are most nearly orientated perpendicular to the vector $\vec{v}$ joining the sweep centres.

The position of each dividing plane is based on the corners of the extreme B-scans in each sweep and the centre of the sweep: for sweep A in Figure 7(a), these are the points $a_0 \ldots a_7$ and g_a respectively. The vector from the centre of one sweep to the other, $\vec{v}$, is compared with the orientation of each of the average planes through $\{a_0, a_1, a_2, a_3\}$, $\{a_0, a_4, a_5, a_1\}$, $\{a_1, a_5, a_6, a_2\}$, $\{a_5, a_4, a_7, a_6\}$, $\{a_3, a_2, a_6, a_7\}$ and $\{a_4, a_0, a_3, a_7\}$. That plane for which the dot product of its normal with $\vec{v}$ is greatest is selected for defining the new dividing plane. In the case of Figure 7(b), the planes defined by $\{b_1, b_5, b_6, b_2\}$ and $\{a_4, a_5, a_6, a_7\}$ are used. The dividing plane is then defined by $\vec{p} \cdot \hat{n} - o = 0$, where $\vec{p}$ is any point on the plane, and o and $\hat{n}$ are defined by equation (1):

$$
\begin{aligned}
\hat{n_a} &= \operatorname{norm}\left((\vec{a_4} - \vec{a_5} - \vec{a_6} + \vec{a_7}) \times (\vec{a_4} + \vec{a_5} - \vec{a_6} - \vec{a_7})\right) \\
\hat{n_b} &= \operatorname{norm}\left(\left(\vec{b_1} - \vec{b_5} - \vec{b_6} + \vec{b_2}\right) \times \left(\vec{b_1} + \vec{b_5} - \vec{b_6} - \vec{b_2}\right)\right) \\
\hat{n} &= \operatorname{norm}\left(\hat{n_a} + \hat{n_b}\right) \\
o &= \frac{1}{8}\left(\vec{a_4} + \vec{a_5} + \vec{a_6} + \vec{a_7} + \vec{b_1} + \vec{b_5} + \vec{b_6} + \vec{b_2}\right) \cdot \hat{n}
\end{aligned} \tag{1}
$$

where norm indicates vector normalisation.

Once all the planes have been defined, the next step is to decide which sweep should be used in which partition of space. This is done by considering for each partition (of which there are up to 2^n, where n is the number of planes), which sweep has its centre farthest inside (or least outside) that partition. This can be done by calculating $l_{s,p}$:

$$l_{s,p} = \frac{1}{N_s} \sum_{i=1}^{N_s} \min_{1 \le j \le D} (d_{ij}) \tag{2}$$

where s is the sweep, p is the partition, N_s is the number of B-scans in sweep s and D is the number of dividing planes. d_{ij} is the perpendicular distance of the centre of B-scan i to dividing plane j, where the sign is positive if the centre is the same side of that plane as the partition. The sweep with the largest value of $l_{s,p}$ is the one used for partition p.

Redundant planes can be found by considering, for each partition, whether the same sweep is used in that partition as in the one which is the opposite side of the plane. If this is the case for *all* partitions, then the plane does not separate any sweeps and can be removed.

After the number and position of the dividing planes has been established, they can be manually adjusted, if necessary, with feedback in both the 3D 'outline' window and the 'review' window. This is explained in Section 4.

3.3 Reslicing

Once dividing planes have been defined, they can be used to reslice the data (i.e. view it on arbitrary planes). The reslice algorithm is as explained in [18], save that for each point in the reslice, its partition is calculated, and only B-scans from the sweep used in that partition are considered for interpolation to the reslice plane. An example of the result of using one dividing plane has already been demonstrated in Figure 1(b) — the misregistration of the data can still be seen, but no longer detracts from the useful information in the B-scans.

3.4 Volume measurement from cross-sections

Volume measurement from parallel cross-sections (i.e. multiplying area by slice thickness), in this context sometimes called step-section planimetry, is in widespread use. Less widely used is a more general version of this same formula for measuring volume from non-parallel sequential cross-sections [29]:

$$v = \left| \sum_{i=2}^{N} \frac{1}{2} (\vec{s}_i + \vec{s}_{i-1}) \cdot (\vec{\omega}_i - \vec{\omega}_{i-1}) \right| \tag{3}$$

where v is the volume obtained from N cross-sections which have vector areas $\vec{s_1} \dots \vec{s_N}$ and centroids $\vec{\omega_1} \dots \vec{\omega_N}$.

This is a very flexible equation which can be applied to highly non-parallel, even overlapping B-scans and hence it is appropriate for freehand 3D ultrasound. It is also straightforward to extend the trapezoidal assumption in equation (3) to cubic interpolation, which gives a flexible and accurate volume estimate from a handful of cross-sections: *cubic planimetry* is studied in much more detail in [28].

In this case, we need to calculate the volume in each partition of space, then sum these to give the total volume. It is possible to examine what effect this will have on the volume estimate by considering a set of cross-sections cut by a single dividing plane, as in Figure 8. We assume for the moment that the dividing plane passes through *all* of the cross-sections. Note that the vector areas $\vec{s}$ and centroids $\vec{\omega}$ in each partition are related to the true values

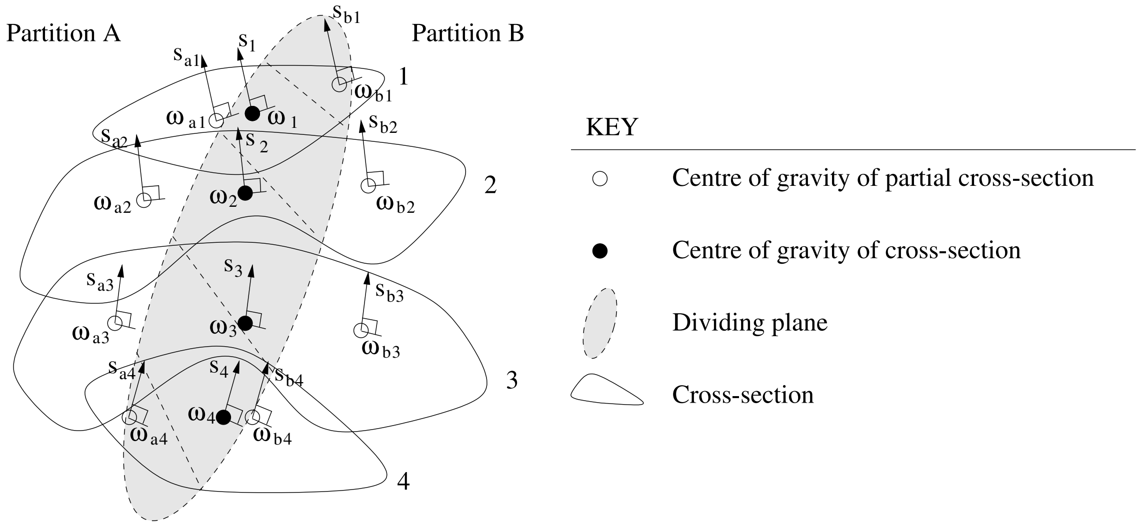


Figure 8: **Calculation of volume in each partition.** The dividing plane splits the cross-sections $1 \ldots 4$ into partitions A and B. The object volume can be calculated from equation (3) with $\vec{\omega_1} \ldots \vec{\omega_4}$ and $\vec{s_1} \ldots \vec{s_4}$, or from the sum of the similar values with subscript a and b, in partitions A and B respectively.

as follows:

$$\vec{s_i} = \vec{s_{ai}} + \vec{s_{bi}}, \quad \hat{s_i} = \hat{s_{ai}} = \hat{s_{bi}} \tag{4}$$

$$|\vec{s_i}|\,\vec{\omega_i} = |\vec{s_{ai}}|\,\vec{\omega_{ai}} + |\vec{s_{bi}}|\,\vec{\omega_{bi}} \tag{5}$$

The total volume of the object, calculated from the volume in each partition using equation (3), is given by[7]:

$$v = \frac{1}{2}\sum_{i=2}^{4}\left\{(\vec{s_{ai}} + \vec{s_{ai-1}}) \cdot (\vec{\omega_{ai}} - \vec{\omega_{ai-1}}) + (\vec{s_{bi}} + \vec{s_{bi-1}}) \cdot (\vec{\omega_{bi}} - \vec{\omega_{bi-1}})\right\} \tag{6}$$

If we define l_{ai} as the ratio of partition A to the whole cross-sectional area, $|\vec{s_{ai}}| \,/\, |\vec{s_i}|$, and $\vec{\alpha_{ai}}$ as the vector distance between the centroid of partition A and the whole cross-section centroid, $\vec{\omega_i} - \vec{\omega_{ai}}$, for any cross-section i, this leads to the substitutions:

$$\begin{aligned}
\vec{s_{ai}} &= l_{ai}\vec{s_i} \\
\vec{s_{bi}} &= (1 - l_{ai})\,\vec{s_i} \\
\vec{\omega_{ai}} &= \vec{\omega_i} - \vec{\alpha_{ai}} \\
\vec{\omega_{bi}} &= \vec{\omega_i} + \frac{l_{ai}}{1 - l_{ai}}\vec{\alpha_{ai}}
\end{aligned} \tag{7}$$

Substituting equations (7) into equation (6) and rearranging leads to:

$$v = \frac{1}{2}\sum_{i=2}^{4}\left\{(\vec{s}_i + \vec{s}_{i-1}) \cdot (\vec{\omega}_i - \vec{\omega}_{i-1}) + (l_{ai} - l_{ai-1})\left[\frac{\vec{s_i} \cdot \vec{\alpha_{ai-1}}}{1 - l_{ai-1}} + \frac{\vec{s_{i-1}} \cdot \vec{\alpha_{ai}}}{1 - l_{ai}}\right]\right\} \tag{8}$$

[7] In fact, we use the cubic planimetry version of this formula [28], which gives more accurate results — the linear version is examined here in order to make the maths more accessible.

The first half of equation (8) represents the volume calculated from equation (3) applied to the whole cross-sections, without using the dividing plane. The second part therefore represents the error in the volume estimate introduced by splitting the cross-sections into two parts and performing the volume calculation on each part. There are two interesting points to note from this expression.

Firstly, the error is dependent on terms like $\vec{s_i} \cdot \alpha_{a\vec{i}-1}$. Since $\vec{s_i}$ is by definition a vector normal to the plane i, and $\alpha_{a\vec{i}-1}$ is by definition a vector lying within the plane $i-1$, if i and $i-1$ are parallel, the error will be zero. This is no surprise, since equation (3) reduces to *area* $\times$ *slice thickness* for the case of parallel B-scans; and we would not expect this formula to be affected by summing volumes over several sections of the object in this way.

Secondly, the second term in equation (8) scales with $(l_{ai} - l_{ai-1})$, which can be interpreted as the difference in the area ratio into which the cross-sections on sequential B-scans are cut by the dividing plane. Hence, if the dividing plane cuts all the cross-sections with the same area ratio, the error will once again be zero. This scenario will tend to happen for dividing planes which are orthogonal to the B-scans.

The result of this is that equation (6) gives volumes close to that of equation (3) in all situations except where the B-scans are very non-parallel *and* the dividing plane passes through these B-scans at an acute angle — which happens rarely in practice. This observation is consistent with the results of the simulated scanning experiments in Section 5.

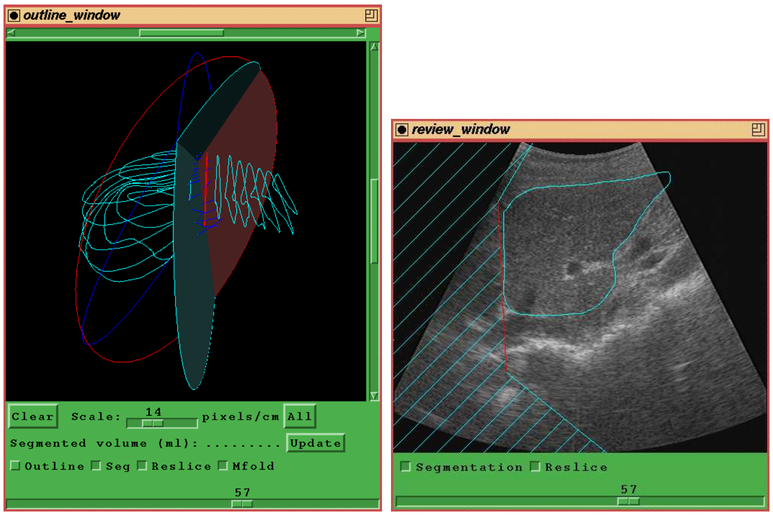
In practice, the main error introduced by using dividing planes is that of *partial voluming*, i.e. where a part of the object is missed from the volume calculation entirely. In Figure 8, where the dividing plane cuts through all the cross-sections, this is not a problem. However, this is often not the case in practice, and the inclusion of dividing planes, especially at acute angles to the B-scans, will tend to increase slightly the effect of partial voluming, since the volume calculation for a partition cannot proceed past the last cross-section which intersects that partition.

3.5 Surface reconstruction from cross-sections

While surface reconstruction of the segmented organ gives a less accurate volume than the method described in the previous section, it is useful for verification of the segmentation and hence also the volume estimate. The method used is a variant of shape based interpolation [21], as described in [28], followed by triangulation by Regularised Marching Tetrahedra [27]. The result is a sequential surface estimator which is robust to varying topologies and can handle non-parallel cross-sections. The triangulation method is designed to generate triangles of good aspect ratio, which reduces the number of triangles and improves the display of the surface.

This method needs some adjustment for multi-sweep scans, since the cross-sections are not generally closed in this case, but a closed shape is required in order to calculate the signed distance field used in shape based interpolation. Cross-sections which are not closed are joined at their end-points, which lie entirely outside of the partition in which the data is used. Then, after the surface is interpolated and triangulated, triangles lying outside the relevant partition are removed, and those which intersect the partition are clipped to the dividing planes surrounding the partition. The clipping requires some care, since it is possible for triangles to intersect more than one dividing plane, in which case they must be clipped to all such planes — this is done by a recursive algorithm described in Appendix A.2.

The surface is extracted for each partition, using in each case the sweep most appropriate



(a) Segmentation and planes (b) B-scan review window

Figure 9: **Editing dividing planes.** The data is from the same examination as in Figure 3, although a more complex arrangement of dividing planes has been used in this case. Planes being edited are marked in red in both 'outline' and 'review' windows, and both windows are updated as the plane is moved.

to that partition, as defined by equation (2). This results in a set of surfaces, which together make up the entire object. Although the surfaces are not in general closed, edges *are* guaranteed to lie on dividing planes. Examples of such surfaces are included in Figure 10 for simulated scans and Figure 13 for scans of a human liver. As with the reslice, errors due to misregistration are clearly apparent, but do not detract from the rest of the data.

It is also possible to calculate a secondary estimate of volume from this set of surfaces, providing each surface intersects no more than two dividing planes. This is based on Gauss' theorem [15], save that the direction of calculation is constrained to be parallel to both intersected dividing planes, as explained in Appendix A.3.

4 System interface

4.1 Selecting B-scans

A specific B-scan can be selected at any time, either by using the slider in the 'review' or 'outline' window, or by clicking on the B-scan outline (or segmentation if there is one) in the 'outline' window. This selection changes the current B-scan in *both* windows. Thus it is easy to mark those B-scans which are not valid (as described in Section 3.1), by first selecting the B-scan, then toggling the 'valid' state with a key press.

4.2 Editing dividing planes

The number and position of the dividing planes is calculated automatically from the number and position of sweeps, as described in Section 3.2. However, they can also be edited in the 'outline' window. Figure 9 shows an example of a more complex dividing plane arrangement from the same data as in Figure 3. There are two points of interest in the display of these planes.

Firstly, dividing planes are inherently infinite — since it is not possible to show the orientation of an infinite plane, all planes are clipped to a sphere, centred at the mid-point of the data and large enough to contain all of it. Clipping all the planes to the same sphere also guarantees that the displayed discs will meet at their edges, as in Figure 9(a). Secondly, also as in this figure, dividing planes are only rendered opaque *where they separate data from two different sweeps*: in other areas only the outer edge of the disc representing that plane is rendered. This results in the number of displayed surfaces being dependent only on the number of sweeps, and not on the number of dividing planes — in this figure there is only one surface, since there are only two sweeps. Rendering is achieved by clipping the disc representing each plane to each partition, resulting in a set of convex polygons, using the algorithm in Appendix A.1.

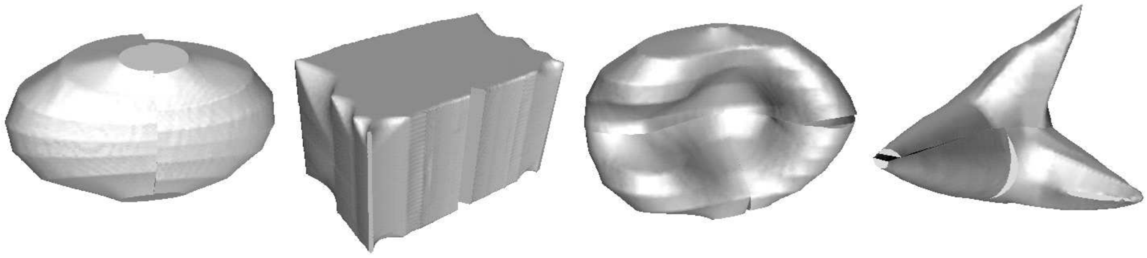
A particular plane can be edited by selecting it with a right mouse click (whereupon it is drawn in red), then moved by clicking and dragging (one button rotates, the other translates). Planes can also be inserted and deleted. As the plane is being moved, the sweep used in any given partition is recalculated using equation (2), and the opaque shading of the planes is also updated. This means that as the plane is moved it is immediately obvious which part is being used to separate data (i.e. the opaque part of the plane), and whether the plane is being used at all — if not it will be shown as a red circle with no opaque part.

In addition, as the plane is being moved, the position of its intersection with the current B-scan in the 'review' window is also updated, i.e. the red line and cross-hatching in Figure 9(b). This is useful for showing whether the plane intersects the actual B-scan data, rather than just the rectangular recorded video window.

4.3 Drawing cross-sections

Cross-sections are drawn in the 'review' window. Any part of the review window which is in a partition for which data from another sweep is being used is cross-hatched. This means that it is clear which part of the B-scan is better covered by data from another sweep, as in Figure 9(b). The regions to cross-hatch can be determined by the same algorithm used to calculate the opaque regions of the dividing plane discs (see Appendix A.1), initialised with a rectangle representing the B-scan rather than a disc.

Contours can be drawn in one of two modes: either by holding the mouse button down and dragging, or by marking points with the left button, the final point being marked with the right button. Once the last point has been marked (or the mouse button has been released, if using the former mode), the contour is joined, *unless* both the start and end points lie within the cross-hatched region, in which case the contour is left open. This ensures that contours are either closed, or closed by the dividing planes — which fulfils a requirement of the volume measurement and surface reconstruction algorithms in Section 3. In both the 'outline' and 'review' windows, cross-sections are coloured cyan where they lie in partitions for which the sweep is being used, and dark blue otherwise. Once again, this makes it clear which part of



(a) Ellipsoid (b) Box (c) Concave object (d) Branching object

Figure 10: **Simulated objects.** Objects (a) to (d) are shown as surface reconstructions from multiple sweeps as in Figure 11(b), (c), (g) and (h) respectively. The box (b) violates the usual assumption of smoothness. (c) and (d) exhibit features typically found in human organs.

the cross-section is going to be used in the volume and surface calculations.

5 Results

5.1 Simulated examinations of geometric objects

In order to verify the volume measurement and surface reconstruction techniques, without introducing errors common to all systems due to registration and segmentation, several objects were "scanned" in simulation, using the tool described in Section 2. These objects are shown (reconstructed from multiple sweep scans) in Figure 10.

Each object was precisely segmented by thresholding the simulated B-scans, using a range of cross-sections between 4 and 23 *per sweep*, in each of the eight sweep patterns described in Figure 11. These sweep patterns where chosen to be representative of actual clinical situations; for instance, scanning between ribs.

The pixel size in all cases was 0.012cm, and the average volume of the objects was 5.4cm^3, leading to inherent volume inaccuracy due to the sampling resolution of approximately $\pm 0.7\%$. As discussed in Section 3.4, the position of the first and last cross-sections can cause a partial voluming effect — in order to minimise this effect, the first and last cross-sections were chosen at all times to be close to the edge of the object being scanned.

Volume measurements were made from the cross-sections by cubic planimetry (see Section 3.4), and also by linear planimetry, and by calculating the volume from the interpolated surface. The latter two methods are in much more common usage than the first, and serve as a comparison with the cubic planimetry technique.

Table 1 contains the results for each object and for each sweep pattern. In each case, the total number of cross-sections required for the cubic planimetry estimate to be within $\pm 2\%$ of the real volume was calculated. This is shown in the top four lines of the table — where *all* the experiments on an object gave volumes within $\pm 2\%$, the lowest number of cross-sections investigated is shown. The remaining rows of Table 1 show the accuracy of the other two volume measurement methods for the number of cross-sections detailed in the top four rows.

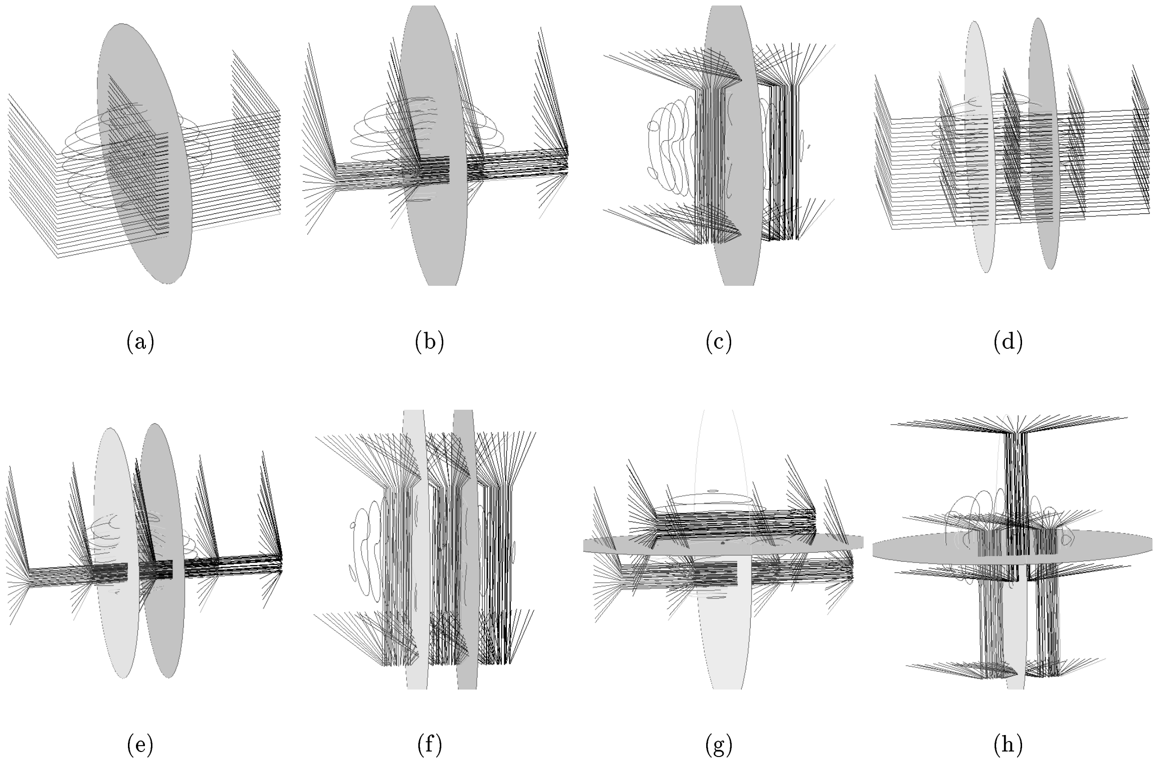


Figure 11: **Simulated ultrasound sweeps.** (a) to (h) are the sweep patterns used to test the volume measurement accuracy on simulated objects. Each B-scan is drawn as a 'goal post', where the 'crossbar' is at the top of the B-scan. Here all are shown with a segmentation from the 'glove' object. Dividing planes are calculated automatically from the sweep position and orientation.

	Sweep:	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
No. of scans for	Ellipsoid	<12	13	24	<16	17	21	22	<17
±2% accuracy	Box	<12	16	15	<18	19	<18	19	<19
	Concave	<12	<12	18	17	17	18	16	18
	Branching	16	13	16	17	22	22	18	19
Accuracy of linear	Ellipsoid	>4.8	5.9	3.8	>5.8	7.3	6.9	5.4	>7.2
planimetry volume	Box	>1.5	3.5	7.7	>1.5	5.6	>8.8	4.4	>5.7
at this point, ±%	Concave	>5.7	>7.7	4.1	6.4	7.9	6.7	7.4	6.8
	Branching	1.0	5.2	5.2	3.0	6.5	5.3	5.3	5.9
Accuracy of volume	Ellipsoid	>7.0	7.7	1.5	>9.3	9.3	1.8	4.0	>9.0
from surface at	Box	>1.6	4.0	2.0	>1.5	4.4	>0.6	0.5	>4.0
this point, ±%	Concave	>12.0	>11.3	2.5	9.0	10.7	4.7	6.2	5.8
	Branching	7.8	12.6	4.7	10.8	10.5	2.5	7.0	7.7

Table 1: **Simulation results.** The table shows the total number of cross-sections, using sweep configurations (a) to (h), required for the cubic planimetry volume to be within ±2% of the actual volume. The accuracy of the volume as calculated from linear planimetry and the surface interpolated from these cross-sections is also shown for comparison.

2% was selected as a test point since it is far enough from the resolution limit (0.7%) not to be overly affected by it, but still more accurate than the entire *in vivo* system, as demonstrated in Section 5.2.

It is clear from this table that volume measurements to an accuracy of within ±2% are possible for *all* the objects described in Figure 10 and sweep patterns described in Figure 11. Whilst there is *some* variation in the number of cross-sections required to obtain this accuracy, typically only 7 or 8 cross-sections are required *per sweep* (sweep patterns (a) to (c) contain two sweeps, the remainder contain three). The only exception to this, where 12 cross-sections are required, is for sweep pattern (c). This is a good example of the case described in Section 3.4, where the cross-sections are non-parallel, and the dividing plane cuts through them at a highly acute angle, leading to larger errors in the volume estimate.

In all cases, with the single exception of the branching object with sweep (a), the linear planimetry accuracy is worse than that of cubic planimetry — typically greater than ±5%. This is similar to our previous observation with single sweep data [28], and indicates that those results can be carried through to the multiple sweep case presented here.

The accuracy of the volume calculated from the surface representation is generally slightly poorer than that of linear planimetry, with a greater variability across shape and sweep pattern. However, the similarity of the volume measurements to that of cubic planimetry suggests that there were no gross errors in either the surface interpolation or the volume calculation algorithms. This is backed up by the surfaces themselves, a sample of which are shown in Figure 10.

5.2 *In vivo* examination of human liver

The *in vivo* precision of this volume measurement method can be estimated by considering multiple examinations of the same organ. It was not possible to verify actual *in vivo* accuracy, since we did not have access to a secondary measurement method (e.g. CT or MRI). We have

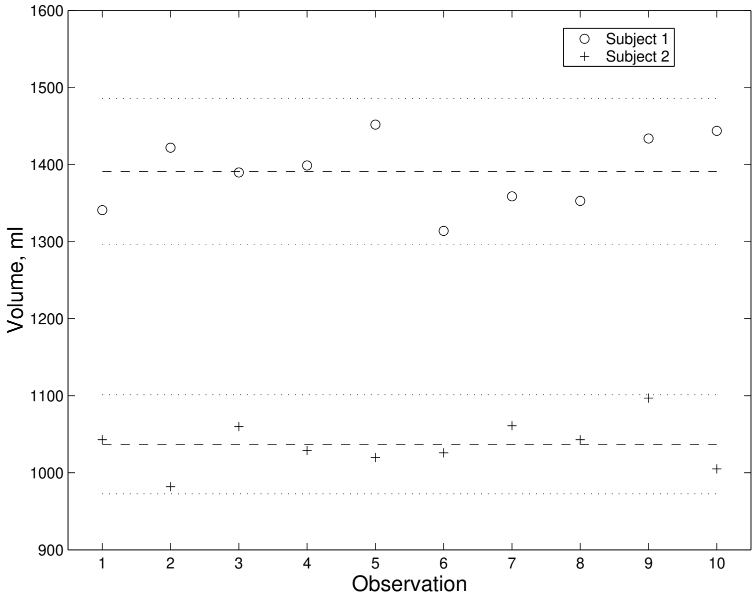


Figure 12: **Liver volume for two subjects.** The first five observations used two sweeps, and the second five used three sweeps. Volumes were calculated using cubic planimetry. Dashed lines show the mean, and dotted lines the 95% confidence intervals.

previously demonstrated that the same system is accurate to within $\pm 7\%$ when applied to *in vivo* data from single sweeps [28].

The equipment and method used is described in Section 2. Typically, 20 cross-sections were outlined in each case, with visual feedback from the 'outline' window and the interpolated surface to assist in cases where it was not obvious where the cross-section should be. This whole process (from scanning to volume measurement) took approximately 30 minutes per data set, the vast majority of time being spent on manual segmentation. For the purpose of this experiment, the actual volume measurement was hidden during segmentation, so there could be no chance of increasing or decreasing the size of the cross-sections in an unconscious attempt to make the volumes similar across the same subject.

Figure 12 shows the results for both subjects, and the mean and 95% confidence interval in each case (assuming a normal distribution). The volume of the liver of subject 1 was 1391 ± 90ml (6.5%) and that of subject 2 was 1037 ± 61ml (5.9%). The first five observations were from examinations using two sweeps, and the second using three sweeps. Considered separately, the volume for the two sweep examination of subject 1 was 1401 ± 74ml (5.3%) and for subject 2 was 1027 ± 52ml (5.1%), and that for the three sweep examination for subject 1 was 1381 ± 100ml (7.2%) and for subject 2 was 1046 ± 62ml (5.9%). These results indicate that the overall precision of the system is approximately $\pm 7\%$, and that using two sweeps was in this case better than three, giving a precision of approximately $\pm 5\%$. It is not clear whether this improvement was due to the use of fewer sweeps (and hence fewer dividing planes) or the improved definition of the liver boundary in the sagittal (as in the right hand sweep of Figure 3(a)), rather than horizontal (as in all other sweeps in Figures 3 and 4), scanning planes.

Surfaces reconstructed from each of the examinations of Figure 12 are shown in Figure 13,

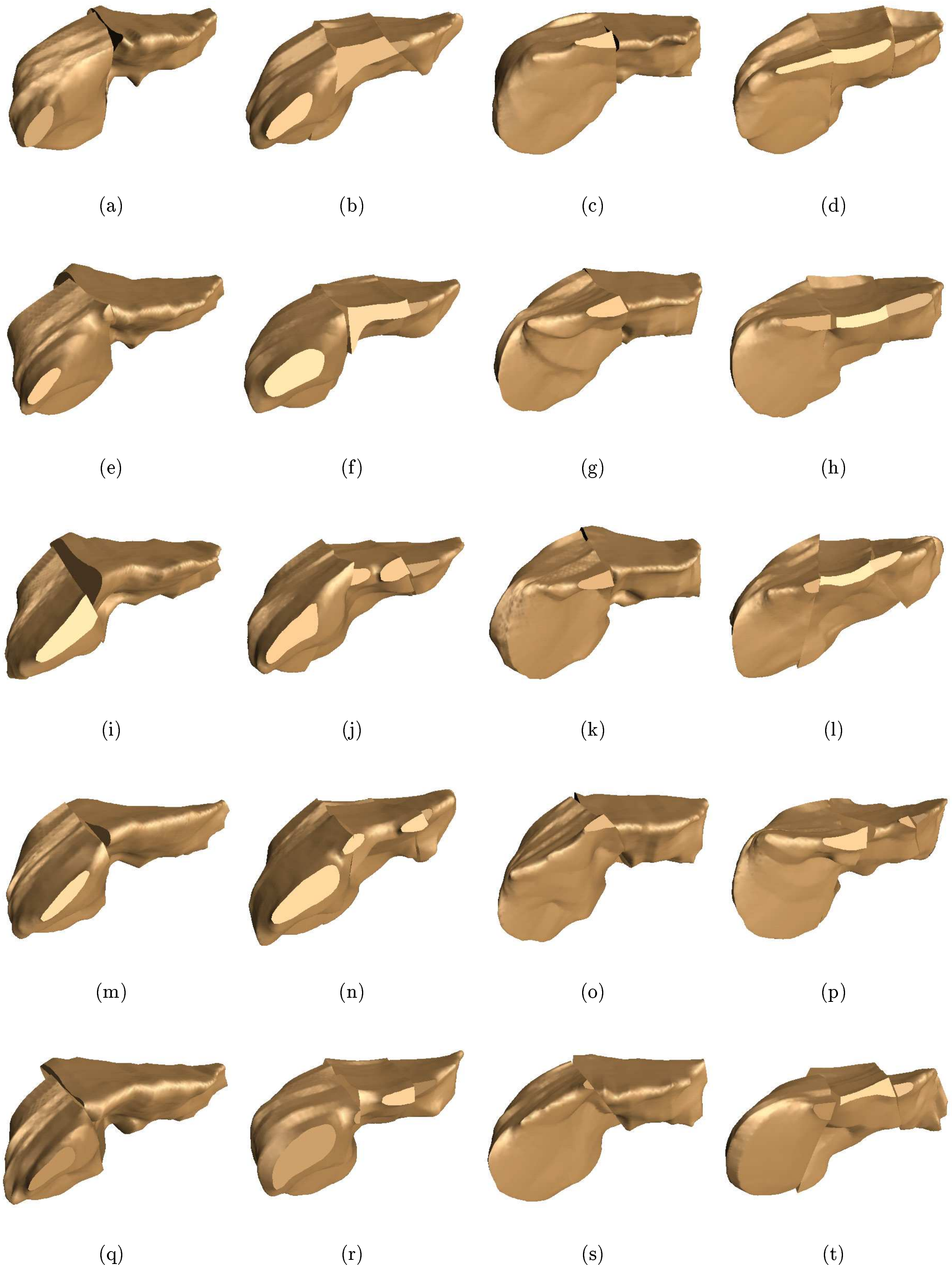


Figure 13: **Reconstructed surfaces of the human liver.** The left hand two columns are from subject 1, and the right from subject 2. The left hand column of each of these is constructed using two sweeps, and the right using three.

looking along the longitudinal axis to the inferior side of the liver, such that the scan-head was at the top of the surfaces, i.e. the same orientation as in Figures 3 and 4. Several things are apparent from these surfaces.

Firstly, although there is considerable variation due to both the scanning pattern and segmentation, all the livers can be clearly categorised as being from subject 1 or 2. Variation in the surfaces is caused by probe pressure at the top (the curve of the probe can be clearly seen in all the surfaces, for instance Figures 13(b) and (d)), and also difficulty in segmenting the liver from the gall bladder and inferior caval and portal veins, especially given the sparsity of the cross-sections. The fanning action used in most of the sweeps also tended to result in oblique incidence at the edges of the liver, particularly in the right lobe. This made it difficult to segment the first and last cross-sections of that sweep, aggravating the partial voluming effect, and in many cases parts of the right lobe were missing from the volume calculation entirely. In addition, some B-scans, for instance the one in Figure 3(c), missed part of the right lobe entirely, and in this case the segmentation had to be estimated from the remaining data.

6 Conclusions

We have presented a novel method for measuring the volume of large organs using 3D ultrasound. That it is an inherently accurate method is demonstrated by simulation across several very different objects and sweep patterns — accuracies better than $\pm 2\%$ where achieved in all cases, using only 7 or 8 cross-sections per sweep. *In vivo* precision has been demonstrated on examinations of the human liver to be as good as $\pm 5\%$ in the two sweep case. Since none of the algorithms make topological assumptions about the organ under investigation, the technique is equally appropriate in other areas, e.g. obstetrics.

Most of the residual error is due to misregistration of the data and inaccurate manual segmentation. In allowing the cross-sections to be drawn on the original B-scans, this method reduces the complexity of segmentation. In addition, the method is robust to organ movement along the dividing planes caused by probe pressure — further work is required to determine how significant this type of misregistration is.

The use of dividing planes allows reslices through multiple sweep data in only 0.2 seconds or so, and the estimation of surfaces from cross-sections in 10 seconds or so. In both these situations, features due to misregistration are clearly visible to the clinician, without detracting from the 'real' information in the data. Although this leads to reslices and surfaces which appear less smooth than those generated by warping [2] or smooth surface approximation [13], the displayed information is more faithful to the original data, with no unsupervised suppression of inevitable artifacts.

Acknowledgements

Graham Treece is supported by an EPSRC studentship, and a Newton Trust award from the University of Cambridge Department of Engineering.

A Appendix

A.1 Intersection of a convex planar polygon with a partition

This function is required in order to calculate which part of a dividing plane to shade in the 'outline' window, and which area to cross-hatch in the 'review' window. In the former case, it is initialised with a polygon approximating to a disc, and in the latter with a rectangle marking the edge of the current B-scan. The 'distance_to_plane(x,y,z,i)' function returns the distance of the point (x, y, z) to dividing plane i.

```
/* Perform intersection of polygon with one partition, defined by dividing planes */
/* Initial polygon has p1 vertices, at locations x1[], y1[], z1[] */
for (i=0; i<dividing_planes; i++) {                           /* For all dividing planes */
  d1 = distance_to_plane( x1[p1-1], y1[p1-1], z1[p1-1], i );  /* Distance for last vertex */
  if ( !(partition & (1<<i)) ) d1 = -d1;                      /* check sign against partition */
  p2 = 0;                                                     /* Initialise new polygon p2, x2[], y2[] */
  for (v=0; v<p1; v++) {                                      /* and loop through vertices */
    d1 = distance_to_plane( x1[v], y1[v], z1[v], i );         /* Distance for this vertex */
    if ( !(partition & (1<<i)) ) d2 = -d2;                    /* check sign against partition */
    if ( d1 > 0 ) {              /* Check for an intersection with line from (p1-1) to (v) */
      if ( d2 < 0 ) {                            /* Just gone outside partition - need new vertex */
        x2[p2] = (d1 * x1[v] - d2 * x1[(v-1+p1)%p1]) / (d1 - d2) + 0.5;
        y2[p2] = (d1 * y1[v] - d2 * y1[(v-1+p1)%p1]) / (d1 - d2) + 0.5;
        z2[p2] = (d1 * z1[v] - d2 * z1[(v-1+p1)%p1]) / (d1 - d2) + 0.5;
        p2++;
      } else {                                   /* Still inside partition - keep this vertex */
        x2[p2] = x1[v]; y2[p2] = y1[v]; z2[p2] = z1[v];
        p2++;
      }
    } else if ( d2 > 0 ) {        /* Just gone inside partition - need this and new vertex */
      x2[p2] = (d1 * x1[v] - d2 * x1[(v-1+p1)%p1]) / (d1 - d2) + 0.5;
      y2[p2] = (d1 * y1[v] - d2 * y1[(v-1+p1)%p1]) / (d1 - d2) + 0.5;
      z2[p2] = (d1 * z1[v] - d2 * z1[(v-1+p1)%p1]) / (d1 - d2) + 0.5;
      p2++;
      x2[p2] = x1[v]; y2[p2] = y1[v]; z2[p2] = z1[v];
      p2++;
    }
    d1 = d2;                                     /* Keep record of last distance to plane */
  }
  copy_polygon( p2, x2, y2, z2, &p1, &x1, &y1, &z1 );         /* Update original polygon */
}
```

A.2 Clipping a surface to a partition

A triangulated surface can be clipped to a given partition by clipping each of the triangles in turn to each dividing plane in turn. The function must be recursive, since triangles may intersect more than one dividing plane. The 'distance_to_plane(x,y,z,i)' function is as in the previous section.

```
/* Intersect triangle t, whose vertices are t.a, t.b and t.c, with this partition */
/* List of new_tris new triangles returned in new_tri[] */
new_tris = 1; new_tri[0] = t;                              /* Initialise list of triangles */
for (i=0; i<dividing_planes; i++) {                          /* Loop through all planes */
  add_tris = 0;
  for (j=0; j<new_tris; j++) {                 /* Loop through all triangles in current list */
    t = new_tri[j];                                                  /* Get new triangle */
    da = distance_to_plane( t.a.x, t.a.y, t.a.z, i );              /* and distance to plane */
    db = distance_to_plane( t.b.x, t.b.y, t.b.z, i );                  /* for each vertex, */
    dc = distance_to_plane( t.c.x, t.c.y, t.c.z, i );                    /* checking sign */
    if ( !(partition & (1≪i)) ) { da = -da; db = -db; dc = -dc; }
    abi = 1; aci = 1; bci = 1;     /* Find intersections with each edge ab, ac and bc */
    if ( da > 0 ∧ db > 0 ) {                                           /* Check edge ab */
      ab.x = (da * t.b.x - db * t.a.x) / (da - db);
      ab.y = (da * t.b.y - db * t.a.y) / (da - db);
      ab.z = (da * t.b.z - db * t.a.z) / (da - db);
    } else abi = 0;
    if ( da > 0 ∧ dc > 0 ) {                                           /* Check edge ac */
      ac.x = (da * t.c.x - dc * t.a.x) / (da - dc);
      ac.y = (da * t.c.y - dc * t.a.y) / (da - dc);
      ac.z = (da * t.c.z - dc * t.a.z) / (da - dc);
    } else aci = 0;
    if ( db > 0 ∧ dc > 0 ) {                                           /* Check edge bc */
      bc.x = (db * t.c.x - dc * t.b.x) / (db - dc);
      bc.y = (db * t.c.y - dc * t.b.y) / (db - dc);
      bc.z = (db * t.c.z - dc * t.b.z) / (db - dc);
    } else bci = 0;
    if ( da < 0 ) index = (abi≪2)|(aci≪1)|(bci);             /* Form index from type of */
    else index = ∼((abi≪2)|(aci≪1)|(bci));                     /* intersection, and sign */
    switch ( index ) {
    case 0:                   /* No intersections - triangle entirely outside partition */
      for (k=j+1; k<(new_tris+add_tris); k++) new_tri[k-1] = new_tri[k];
      new_tris -= 1; j -= 1;
      break;
    case 1:                               /* ab and ac intersection - one new triangle */
      new_tri[j] = form_triangle( new_tri[j].a, ab, ac );
      break;
    case 2:                              /* ab and bc intersection - two new triangles */
      new_tri[j] = form_triangle( new_tri[j].a, ab, new_tri[j].c );
      new_tri[new_tris+add_tris] = form_triangle( new_tri[j].c, ab, bc );
      add_tris++;
      break;
    case 3:                               /* ac and bc intersection - one new triangle */
      new_tri[j] = form_triangle( new_tri[j].c, ac, bc );
      break;
    case 4:                              /* ac and bc intersection - two new triangles */
      new_tri[j] = form_triangle( new_tri[j].a, new_tri[j].b, ac);
```

```
      new_tri[new_tris+add_tris] = form_triangle( new_tri[j].b, bc, ac );
      add_tris++;
      break;
    case 5:                           /* ab and bc intersection - one new triangle */
      new_tri[j] = form_triangle( new_tri[j].b, bc, ab );
      break;
    case 6:                           /* ab and ac intersection - two new triangles */
      new_tri[j] = form_triangle( new_tri[j].b, new_tri[j].c, ab );
      new_tri[new_tris+add_tris] = form_triangle( new_tri[j].b, ac, ab );
      add_tris++;
      break;
    case 7:                     /* No intersections - triangle entirely inside partition */
      break;
    }
  }
  new_tris += add_tris;                                  /* Update number of triangles */
}
```

A.3 Volume estimation from a clipped surface

The volume of the surface can be calculated from the triangulation provided that the surface is everywhere closed, except on the dividing plane, and only bordered by at most two dividing planes. This is a variant of Gauss' theorem [15]. The 'cross_product(a,b)', 'dot_product(a,b)', 'normalise(a,b)', 'add_points(a,b)' and 'subtract_points(a,b)' all perform vector operations on the two vectors a and b.

```
/* Calculate volume within partition */
volume = 0;                                      /* Initialise calculated volume to zero */
if ( data_in_partition( partition ) ) {          /* Check for any data in this partition */
  important_planes = 0;                          /* Calculate number of bordering planes */
  for (i=0; i<dividing_planes; i++) {
    if ( data_in_partition( partition^(1<<i) ) ) {  /* Plane is bordering if there is data */
      normal[important_planes++] = plane_normal( i );  /* on other side of the plane */
    }
  }
  switch ( important_planes ) {                  /* Calculate projection direction */
  case 0:                                        /* No bordering planes - direction arbitrary */
    vector.x = 0; vector.y = 0; vector.z = 1;
    break;
  case 1:                                        /* One bordering plane - direction within this */
    vector.x = normal[0].z; vector.y = normal[0].x; vector.z = normal[0].y;
    vector = cross_product( normal[0], vector );
    vector = normalise( vector );
    break;
  case 2:                                        /* Two bordering planes - direction parallel to both */
```

```
    vector = cross_product( normal[0], normal[1] );
    vector = normalise( vector );
    break;
  default:
    return -1;          /* More than two bordering planes - calculation not possible */
  }
  volume = 0.0;                                         /* Initialise partition volume */
  for (i=0; i<triangles; i++) {                          /* Loop through all triangles */
    t = triangle[i];                              /* triangle vertices are t.a, t.b and t.c */
    ac = subtract_points( t.c, t.a );             /* Calculate area of triangle in plane */
    ab = subtract_points( t.b, t.a );                                /* normal to vector */
    p = cross_product( ab, ac );
    v = dot_product( p, vector )/2;
    p = add_points( t.a, add_points( t.b, t.c ) )/3;                      /* multiply by */
    volume += v * dot_product( p, vector );          /* projected distance along vector */
  }
  return volume;
} else {
  return 0;                                        /* There is no data in this partition */
}
```

References

[1] R. G. Aarnink, J. J. M. C. H. de la Rosette, F. M. J. Debruyne, and H. Wijkstra. Reproducibility of prostate volume measurements from transrectal ultrasonography by an automated and a manual technique. *British Journal of Urology*, 78:219–223, 1996.

[2] D. Aiger and D. Cohen-Or. Real-time ultrasound imaging simulation. *Real-Time Imaging*, 4:263–274, 1998.

[3] C. P. Allott, C. D. Barry, R. Pickford, and J. C. Waterton. Volumetric assessment of carotid artery bifurcation using freehand-acquired, compound 3D ultrasound. *The British Journal of Radiology*, 72:289–292, March 1999.

[4] H. Baddeley, M. Benson, G. Liefman, T. Singcharoen, V. Siskind, K. Soon, and J. Williams. Measurement of liver volume using water delay ultrasonography. *Diagnostic Imaging in Clinical Medicine*, 55:330–336, 1986.

[5] D. Carr and J. G. Duncan. Liver volume determination by ultrasound: a feasibility study. *British Journal of Radiology*, 49:776–778, September 1976.

[6] A. Cusumano, D. J. Coleman, R. H. Silverman, D. Z. Reinstein, M. J. Rondeau, R. Ursea, S. M. Daly, and H. O. Lloyd. Three-dimensional ultrasound imaging — clinical applications. *Ophthalmology*, 105(2):300–306, February 1998.

[7] J. Deng, J. E. Gardener, C. H. Rodeck, and W. R. Lees. Fetal echocardiography in three and four dimensions. *Ultrasound in Medicine and Biology*, 22(8):979–986, 1996.

[8] A. Fenster and D. B. Downey. 3-D ultrasound imaging: A review. *IEEE Engineering in Medicine and Biology*, 15(6):41–51, November 1996.

[9] P. Fritschy, G. Robotti, G. Schneekloth, and P. Vock. Measurement of liver volume by ultrasound and computed tomography. *Journal of Clinical Ultrasound*, 11:299–303, August 1983.

[10] A. H. Gee, R. W. Prager, and L. Berman. Non-planar reslicing for freehand 3D ultrasound. In *Proceedings of Medical Image Computing and Computer-Assisted Intervention — MICCAI'99*, LNCS 1679, pages 716–725, Cambridge, UK, September 1999. Springer.

[11] O. H. Gilja, A. I. Smievoll, N. Thune, K. Matre, T. Hausken, S. Ødegaard, and A. Berstad. *In vivo* comparison of 3D ultrasonography and magnetic resonance imaging in volume estimation of human kidneys. *Ultrasound in Medicine and Biology*, 21(1):25–32, 1995.

[12] A. S. Gopal, M. J. Schnellbaecher, Z. Shen, O. O. Akinboboye, P. M. Sapin, and D. L. King. Freehand three-dimensional echocardiography for measurement of left ventricular mass: *In Vivo* anatomic validation using explanted human hearts. *Journal of the American College of Cardiology*, 30(3):802–810, September 1997.

[13] T. Hausken, D. F. Leotta, S. Helton, K. V. Kowdley, B. Goldman, S. Vaezy, E. L. Bolson, F. H. Sheehan, and R. W. Martin. Estimation of the human liver volume and configuration using three-dimensional ultrasonography: effect of a high-calorific liquid meal. *Ultrasound in Medicine and Biology*, 24(9):1357–1367, 1998.

[14] D. Howe, T. Wheeler, and S. Perring. Measurement of placental volume with real-time ultrasound in mid-pregnancy. *Journal Clinical Ultrasound*, 22:77–83, 1994.

[15] S. W. Hughes, T. J. D'Arcy, D. J. Maxwell, J. E. Saunders, C. F. Ruff, W. S. C. Chiu, and R. J. Sheppard. Application of a new discrete form of Gauss' theorem for measuring volume. *Physics in Medicine and Biology*, 41:1809–1821, 1996.

[16] E. D. Light, R. E. Davidsen, J. O. Fiering, T. A. Hruschka, and S. W. Smith. Progress in 2-D arrays for real time volumetric imaging. *Ultrasonic Imaging*, 20:235–250, 1998.

[17] T. R. Nelson, D. B. Downey, D. H. Pretorius, and A. Fenster. *Three-Dimensional Ultrasound.* Lippincott Williams & Wilkins, 1999.

[18] R. W. Prager, A. H. Gee, and L. Berman. Stradx: real-time acquisition and visualisation of freehand 3D ultrasound. *Medical Image Analysis*, 3(2):129–140, 1999.

[19] R. W. Prager, R. N. Rohling, A. H. Gee, and L. Berman. Rapid calibration for 3-D free-hand ultrasound. *Ultrasound in Medicine and Biology*, 24(6):855–869, 1998.

[20] S. N. Rasmussen. Liver volume determination by ultrasonic scanning. *British Journal of Radiology*, 45:579–585, August 1972.

[21] S. P. Raya and J. K. Udupa. Shape-based interpolation of multidimensional objects. *IEEE Transactions on Medical Imaging*, 9(1):32–42, 1990.

[22] R. N. Rohling, A. H. Gee, and L. Berman. Three-dimensional spatial compounding of ultrasound images. *Medical Image Analysis*, 1(3):177–193, April 1997.

[23] R. N. Rohling, A. H. Gee, and L. Berman. Automatic registration of 3-D ultrasound images. *Ultrasound in Medicine and Biology*, 24(6):841–854, July 1998.

[24] R. N. Rohling, A. H. Gee, and L. Berman. A comparison of freehand three-dimensional ultrasound reconstruction techniques. *Medical Image Analysis*, 3(4):339–359, 1999.

[25] C. F. Ruff, S. W. Hughes, and D. J. Hawkes. Volume estimation from sparse planar images using deformable models. *Image and Vision Computing*, 17:559–565, 1999.

[26] G. Schwartz. Opinion — three-dimensional volume measurement. *Ultrasound in Obstetrics and Gynecology*, 11:4–5, 1998.

[27] G. M. Treece, R. W. Prager, and A. H. Gee. Regularised marching tetrahedra: improved iso-surface extraction. *Computers and Graphics*, 23(4):583–598, 1999.

[28] G. M. Treece, R. W. Prager, A. H. Gee, and L. Berman. Fast surface and volume estimation from non-parallel cross-sections, for freehand 3-D ultrasound. *Medical Image Analysis*, 3(2):141–173, 1999.

[29] Y. Watanabe. A method for volume estimation by using vector areas and centroids of serial cross sections. *IEEE Transactions on Biomedical Engineering*, 29(3):202–205, 1982.

[30] T. A. Whittingham. New and future developments in ultrasound imaging. *The British Journal of Radiology*, 70:S119–S132, November 1997.