Appendix B – Volumetric Segmentation

12.1 Introduction

The availability and clinical requirements of medical imaging as a source of 3D data set has generated a significant interest in the processing and segmentation of volumetric data. The problems of understanding 3D structure from a discretely sampled volume have shown the benefit of visualisation techniques. Surface approximations, such as isosurfacing, allow surfaces to be extracted that when, rendered and shaded, provide an invaluable insight into a volume’s internal structure.

The reconstruction of multi-modal data sets from different sources of volumetric data is greatly simplified by the successful segmentation of surface topology. Surfaces that directly correspond to a volume can be matched far more simply than the original volumes [Moshfeghi 94].

In addition to structural insight, surface approximations are invaluable in reducing the processing time needed for traditional image processing techniques, as processing can be localised to a contour boundary. Furthermore, these surfaces can provide a mathematical representation of shape which can then be used statistically to model and classify shape and deformation [Cootes 95] [Bowden 96].

If a statistical model is to be constructed which represents 3D surfaces or features extracted from medical or other volumetric datasets, a method of extracting surfaces from these datasets is required in order to produce the training examples necessary for statistical analysis.

A common technique for surface extraction is isosurfacing. Isosurfaces are structures that represent surfaces of equal value, normally made out of graphical primitives such as triangles connected together and rendered using standard graphical techniques.

There are five basic algorithms for Isosurfacing:

1. Opaque cubes or the Cuberille algorithm [Herman 79]
2. Contour connecting [Barequet 96][Fuchs 77][Keppel 75]
3. Marching Cubes [Mullick 95][Cline 88][Lorenson 91]
4. Dividing Cubes [Cline 88]
5. Marching Tetrahedra [Shirley 90].

The Marching Cubes algorithm is by far the most popularly implemented algorithm for iso-intensity surface extraction, efficiently generating isosurfaces with low memory requirements. The Contour Connecting method requires localisation of the contour in each slice of the data and, like the Cuberille algorithm, is prone to artefacts when handling small features and branches in the data. Though the Marching Tetrahedra approach reduces ambiguous topological connections, it generates many more graphical primitives than the Marching Cubes algorithm. Finally, the Dividing Cubes algorithm creates points and corresponding normals requiring special purpose hardware/software for visualisation, making it inappropriate for many applications [Mullick 95].

Barequet *et al* [Barequet 96] propose a technique for piecewise-linear surface reconstruction from a series of parallel polygonal cross sections. As well as the applications of such algorithms in visualisation (isosurfacing) it is an important problem in medical imaging, where contours are often detected in single layers of the volume. By reducing the problem to the piecewise linear interpolation between each pair of successive slices, they use a partial curve matching technique for matching parts of the contours. The major advantage with this over such a scheme as marching cubes is that the size of the resulting polygons compared those produced by marching cudes, where each voxel can produce multiple polygons.

Since the original formulation of Active Contour Models (Snakes) [Kass 88] a significant interest has been shown in extending the technique to dynamic 3D models. Snakes have been shown to be useful in contour reconstruction, but require large amounts of user intervention to successfully segment complex objects. As has been shown, Point Distribution Models [Cootes 95] can simplify the problem of object recognition and segmentation by statistically constraining the shape of the model within suitable bounds, through the analysis of a training set of shapes. However, in 3D, where models become too large to manufacture by hand, another means of generating training sets for statistical analysis must be found.

Terzopolous and Vasilescu [Terzopoulos 91] extended the snake model to include an inflation force that helps remove the need for initial contour placement and thus avoid convergence on local minima. The inflation force drives the snake model outwards towards the object boundary like an inflating balloon. Terzopolous and Vasilescu formulated the model as a finite element mesh and later extended the model to a thin plate spline, demonstrating successful results in the reconstruction of range data and volumetric CT data surface representations [McInery 93].

This chapter presents an iterative, dynamic mesh model which uses simulated physical forces to segment desired surface approximations from volumetric datasets. The work is based on the work of Chen and Medioni [Chen 95] which is itself a continuation of the work on dynamic balloon models by Terzopoulos and Vasilescu [Terzopoulos 91]. Chen and Medioni applied the work to the constrained problem of reconstruction from pre-registered range images.

It will be shown how simplifications can be made to the model which increases the iterative speed of converging on segmented features. It is also shown how balloon models can be reformulated to remove explicit data attraction forces to image features. The process hence behaves like a region growing technique which locates isointensity boundaries within the image. This removes the need for parameter selection which must be balanced against the internal parameters for standard snake [Kass 88] and balloon [Terzopoulos 91] models and further reduces suseptibility to initial placement and image noise.

The remainder of this chapter is organised as follows, Section 12.2 provides an overview of the dynamic mesh balloon model. Section 12.3 discusses mesh structure and connectivity while Section 12.4 covers dataset scaling and interpolation issues. Section 12.5 then formulates the dynamic mesh structure and subdivision mechanisms. Section 12.6 shows the resulting model applied to sample volumetric data sets. Finally conclusions and further work are discussed.

12.2 Overview of the Dynamic Mesh Model

The mesh structure consists of a triangular mesh which can vary in size, shape and connectivity. Each vertex is connected to other vertices in the model by the edges of the polygonal facets. These interconnections are used to simulate springs that connect the mesh mathematically. The force of these springs gives a resulting surface tension to the model which attempts to keep the surface as smooth as possible. An inflation force is used at each vertex to inflate the overall model, while surface tension attempts to keep the mesh spherical. A simple local feature detection scheme is used at each vertex to remove the inflation force as nodes reach the boundaries of desired structures. A dynamic mesh subdivision scheme is used to subdivide polygons locally if they exceed set size or curvature criteria. This allows the mesh to inflate and grow until a boundary is located. Once the mesh has converged on a solution, a good local edge detection scheme can be used to lock vertex points to the boundary. The process starts with a small polygon object which is inflated from within a volumetric image with the inflation force driving the surface towards the object boundary. The mesh grows in size and complexity to fill the object like an inflating balloon until the mesh vertices lie close to the true object boundary (See Figure 12.2.1). This technique requires no user intervention after the initial placement and provides a simple, fast method for object segmentation, which produces surfaces with a low polygon count.

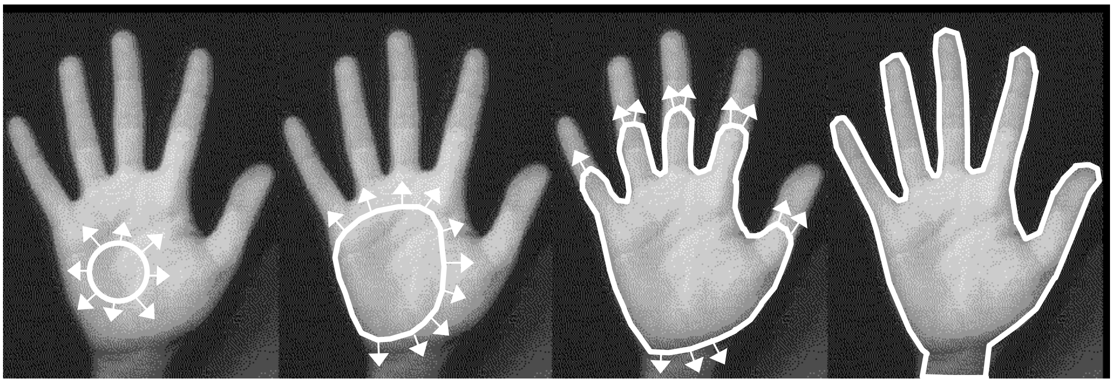


Figure 12.2.1 - Simple 2D Contour Inflating Towards the Object Boundary

12.3 Mesh Structure

To provide the successful extension of the balloon model into 3D, the mesh structure must fulfil a number of criteria:

1. It should allow the dynamic manipulation of a surface and its local properties.
2. It should be structured to ensure render times and processing times are kept to a minimum
3. It should have the ability to represent features accurately by ensuring planar facets and hence reducing mathematical inaccuracies.
4. It must maintain knowledge of its connectivity, to provide a simulated physical model like snakes [Kass 88].
5. It must provide a faithful render of the volume providing accurate visualisation of complex features within a given dataset.

The addition of this final constraint also ensures that the surface will look continuous when rendered with a suitable shading routine such as Gouraud/Phong shading. Perhaps the simplest of mesh structures is that of the simplex mesh, proposed by [Delingette 94]. A simplex mesh is an interconnected set of nodes, where each node is connected to exactly three other nodes (Figure 12.3.1a).

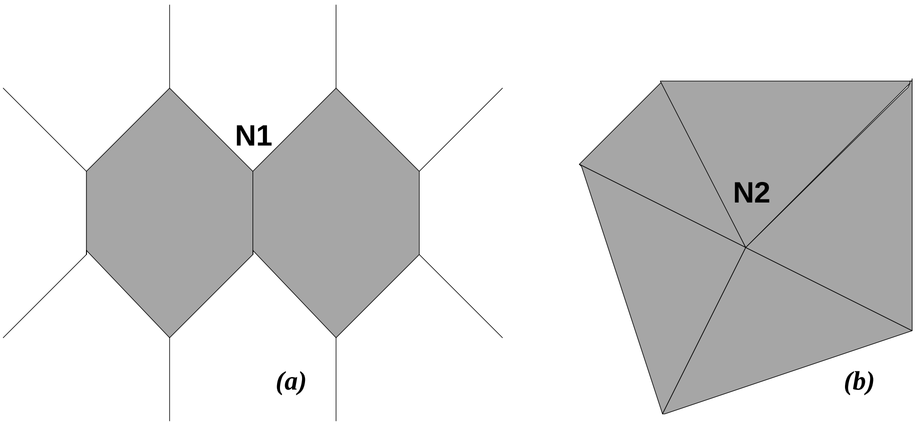


Figure 12.3.1 - Mesh structures

(a) The Simplex Mesh Structure (b) A Planar Mesh Structure

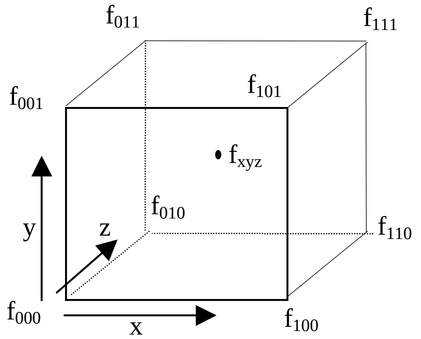
Each node (**N1**) always connects to exactly 3 other nodes producing a simple interconnected surface model from which mathematical simulations of physical properties can easily be implemented. However the polygons bounded by these nodes are non-planar. The calculation of a vertex normal is produced by averaging the normals from the connecting polygons surrounding that vertex. Since non-planar polygons produce inaccurate normal calculations, this mesh formulation will produce inaccuracies in rendering or physical simulation calculations. Inaccuracies in normals will result in non-uniform shading as lighting equations depend upon normals and planar polygons. Surface features will also suffer from the use of non-planar polygons.

A better solution is to use a mesh that has planar facets. Three points always ensure a unique plane and it is simple to subdivide a triangle into multiple triangles. This does, however introduces problems with the connectivity, as any node must be able to connect to any other number of nodes to ensure a complete and evenly spaced surface. In Figure 12.3.1b the node (**N2**) connects to five other nodes.

This provides a mechanism that represents how each vertex connects to other vertices allowing simple physical properties to be represented, i.e. elasticity can be manifested as the force that each of the connected vertices applies on a specific vertex by the direction and length of the connections. However other operations such as rendering and normal calculations require polygons to be expressed as the connection of vertices that constitute a surface facet and it is therefore necessary to retain a dual representation of the surface.

12.4 Volume scaling and Interpolation

Volumetric data is commonly stored as a 3D array of discrete values for each voxel (Volumetric Element) of a volume. The resolution of these volumes tends to be far lower than standard images due to the size and memory requirements. A typical 256x256 grey scale image would occupy 64KB of memory, however a 256x256x256 volume using 256 grey levels would occupy 16MB of memory. Due to the low resolution of volumes and non-cubic voxels it is necessary to smoothly interpolate intensities and attempt to estimate missing information. Tri-linear interpolation is used to reconstruct missing data from the discrete data set and allows a value to be estimated for any position within the volume. Higher order interpolation schemes can be used but introduce additional computational complexity for little gain.



$$\begin{aligned} f_{xyz} = {} & f_{000}(1-x)(1-y)(1-z) + \\ & f_{100}x(1-y)(1-z) + \\ & f_{010}(1-x)y(1-z) + \\ & f_{110}xy(1-z) + \\ & f_{101}x(1-y)z + \\ & f_{011}(1-x)yz + \\ & f_{111}xyz \end{aligned}$$

Figure 12.4.1 - Tri-linear Interpolation

Figure 12.4.1 demonstrates the principal behind tri-linear interpolation. The normalised point within the unit cube is first converted to the discrete volume and its eight discrete corner values determined along with the normalised position within this new sub-unit cube. Placing these values within the equation f_{xyz} gives a linearly-interpolated value for the required point. The equation, although not complex, can quickly become a computational overhead where a large number of interpolated values are required. This technique does not therefore lend itself well to normal image processing techniques where many samples are required for each iteration of an algorithm. However in the case of meshes/3D-surfaces where the presence of the surface greatly reduces the number of interpolations per iteration, the technique enables the dataset to be treated as a continuous volume, smoothing edges and noise.

Higher order interpolation schemes can be used (e.g. tri-cubic interpolation) however, the additional computational cost involved with such schemes outweighs the benefits gained. It should be pointed out that no matter which interpolation scheme is used it is never possible to reconstruct missing data, the values are merely estimated from the available information.

Volumetric data from the medical imaging field tends to have non-cubic voxels where the in-slice resolution is much smaller than that of the depth resolution, and for this reason the volume should be scaleable. This artefact of acquisition can be overcome by translating and rescaling the volume to a cube of 2 unit size. A scaling in x, y and z can then be applied to rescale the volume and associated voxels in to a cuberville (a volume with cubic voxels). Tri-linear interpolation will then attempt to fill-in this missing inter-slice resolution.

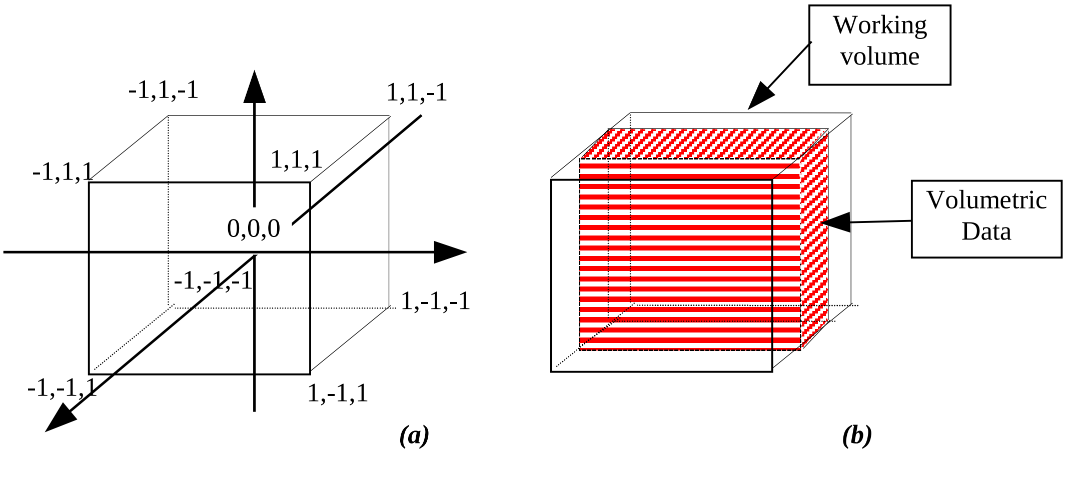


Figure 12.4.2 - The working volume of the 3Dinterpolator

Figure 12.4.2 shows this cube centred about the object, this enables meshes to be built that are of the same scale. For a given dataset the scale is set such that the largest dimension of the volume occupies the full size of the unit cube centred about the origin. As the same scale applies to all dimensions a non cubic volume (eg 200x200,100 voxels) would produce a scaling demonstrated in Figure 12.4.2b, any attempt to access part of the volume outside the volumetric data as outside the cube results in a value of zero. This allows the dataset to easily be rescaled to suit applications.

12.5 The Balloon Model

The balloon model consists of a mesh of triangular facets or patches. The initial triangulated surface can be any shape or size allowing the re-application of a segmented surface to a new data set. Each node (vertex) has two forces acting upon it. The spring force derived from the sum of the vectors of the interconnections of the mesh, and the inflation force, derived from the weighted normal direction of the surface at each node.

The operation of the inflating balloon model can be encapsulated by the following algorithm.

Algorithm 12.1.

```
for a given closed form polygonal model do,
    build a connected mesh of vertices
    while number of polygons is not constant do
        compute the normal at each node
        for each node do,
            compute the elastic force using Equation 12.5-4 (See Section 12.5.2),
            test node position in dataset using feature detection scheme,
            if feature not found calculate the inflation force using Equation
                12.5-5 (See Section 12.5.3) and add to the elastic force
            compute the new node position $v_i^{t+\Delta t}$ using Equation 12.5-3 (Section
                12.5.1) and update node
            perform dynamic subdivision using Algorithm  (See Section 12.5.4)
```

12.5.1 A Simple Dynamic Model.

The motion of any element *i* on a finite element mesh model can be described by the set of coupled second order differential equations [Terzopoulos 91]

Equation 12.5-1
$$m_i \frac{d^2 x_i}{dt^2} + \gamma_i \frac{dx_i}{dt} + g_i = f_i \qquad i = 1, \ldots, n.$$

Here, x is the location of the element *i*, m is its mass, g is the surface tension, generated by the interconnections of the elastic mesh, f is the inflation force and γ is the velocity-dependent damping coefficient that controls the rate of dissipation of kinetic energy. Giving the mesh these simulated physical properties provide a robust model that performs well but at a computational cost. The main rationale for the momentum term $\left(m_i \frac{d^2 x_i}{dt^2} \right)$ is its ability to reduce the mesh's susceptibility to noise. Due to the momentum of nodes the damping term γ is necessary to bring the model to rest. The mesh reaches an equilibrium state when $\frac{d^2 x_i}{dt^2} + \frac{dx_i}{dt} = 0$ which can take some time [Chen 95]. Chen and Medioni simplify this model by making *m=0* and $\gamma = 1$ for all *i* reducing
Equation *12.5-1* to

Equation 12.5-2
$$\frac{dx_i}{dt} = f_i - g_i \qquad i = 1, \ldots, n.$$

Due to this simplification the equation (2) has a very simple explicit integration [Chen 95]

Equation 12.5-3
$$x^{t+\Delta t} = (f_i^t - g_i^t)\Delta t + x^t$$

Unlike the work of Terzopoulos, the approach described here does not use an explicit data force that attracts the balloon surface to image features. Instead the inflation force is used to inflate the surface until the desired feature is located. In order to overcome the noise inherent in medical imaging datasets, the surface is not anchored to positive data features. When a feature is detected at a node position, the inflation force is removed for that node. The surface is then free to oscillate around features until it converges on a solution.

12.5.2 Simplified Spring Force

The spring force exerted on node *i* by the spring linking node *i* and *j* of natural length l_{ij} can be expressed as [Terzopoulos 91],

$$s_{ij} = \frac{c_{ij} e_{ij}}{\left\| r_{ij} \right\|} r_{ij}$$

where c_{ij} is the stiffness, $r_{ij} = x_j - x_i$ the vector separation of the nodes, $\left\| r_{ij} \right\|$ is the length of the spring and $e_{ij} = \left\| r_{ij} \right\| - l_{ij}$ is the deformation of the spring.

In order to generate a generic technique for the segmentation of objects, and due to the large nature of 3D objects it is not feasible to assign values to c_{ij} and l_{ij} for each node. Further simplifications can therefore be made by setting all stiffness coefficients to a constant value with a minimum spring length of zero, $c_{ij} = c$ and $l_{ij} = 0$.

The total elastic force on a node *i* is therefore,

Equation 12.5-4

$$g_i = \frac{c}{n} \sum_{j=0}^{n} r_{ij}$$

12.5.3 Inflation Force

The inflation force applied to each node *i* is

Equation 12.5-5 $$f = k\hat{n}_i$$

where $\hat{n}_i$ is the normal at node *i* and k is the amplitude of the inflation force. The value of *k* can be selected to be a constant for a specific data set or dynamically generated as $k = \frac{5}{4}\|g_i\|$, which ensures that the inflation force for each node always exceeds the surface tension of the model. Although this removes the parameter selection of *k*, it produces a slower convergence on solutions as non optimum parameter selection results.

Node normals are calculated as the average normal of the surrounding polygons sharing the node *i*, gained from the cross product of polygonal edges between vertices. Other, more complicated schemes as used by Chen and Medioni [Chen 95], give little benefit as errors in this normal estimation technique are reduced by the surface smoothing properties of the surface tension (elastic force). This also gives a significant performance increase as normals must be recalculated at least once every iteration of the algorithm.

12.5.4 Dynamic Subdivision

As the inflation force increases the surface area of the mesh, individual polygons grow in size. Since the elastic force is directly proportional to the size of polygons, there comes a point where the elastic force will not allow the mesh to increase in size further, unless the inflation amplitude is increased accordingly. Dynamic subdivision can be used to subdivide polygons which exceed set size criteria and keep polygons within a suitable limit. Each edge of the mesh is checked in turn at each iteration to see if it exceeds the subdivision threshold. Figure 12.5.1 demonstrates how the process works.

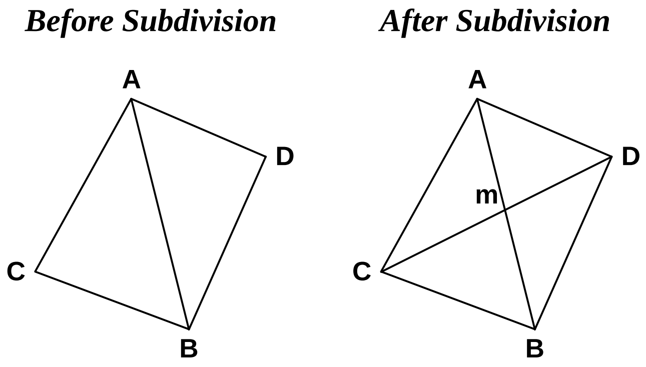


Figure 12.5.1 - Dynamic Subdivision

When the length of a node connection **AB** exceeds set criteria, distance or curvature, the two triangles that contain this edge are located (**ABC, ADB)** and removed from the polygon list. The midpoint **m** of **AB** is calculated and four new triangles constructed **AMC, CMB, ADM, and MDB.** The internal connectivity of the mesh is also altered to reflect this new local structure. Long thin triangles are undesirable, as they do not model local surface properties well. This technique ensures that they never occur, as any edge that exceeds a distance threshold is immediately subdivided. This procedure allows the mesh to grow asymmetrically to fit any feature located within the data set.

The dynamic subdivision procedure can be encapsulated by the following algorithm.

Algorithm 12.2.

```
• for each node (V1) do
    • for each connection to another node (V2) do
        • if the connection (V1V2) matches the subdivision criteria do
            • remove connection (V1V2)
            • remove the two polygons that share this edge
            • find the mid point m of V1V2
            • construct four polygons using m as a common node
            • update the connections of the mesh
• recalculate the normal at each node
```

12.5.5 Subdivision Criteria

Using a distance threshold for subdivision produces an evenly spaced mesh which can alter its structure locally to fit any dataset. It is also possible to use other criteria to provide a more flexible approach. As the normal at each node is known for use with the inflation force, the dot product of two adjacent vertices' normals represents local surface curvature. This can be used to further subdivide the mesh if the dot product drops below a certain threshold value, i.e. the area has a high degree of curvature, allowing more vertices to be placed in these areas of high curvature. This is useful where long narrow features are present in the dataset.

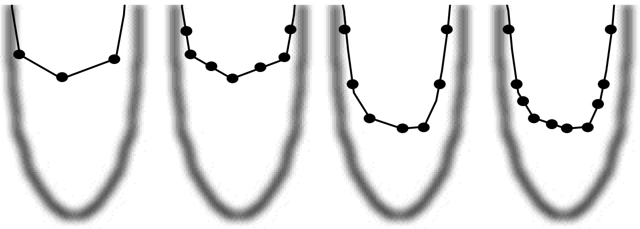


Figure 12.5.2- Curvature Based Subdivision

Figure 12.5.2 demonstrates an image boundary and an inflating balloon front. The boundary shown has found an equilibrium state in the narrow feature. By subdividing the mesh on a curvature basis, in addition to distance, extra vertices are added to the front of the model providing the inflation force needed to successfully segment the long narrow feature.

Both subdivision criteria can be used in conjunction to minimise the polygon count of a mesh, removing the need for post-processing techniques such as Delaunay Triangulation [Soucy 96]. An edge is subdivided only if it exceeds both a distance and a curvature threshold. Polygons on parts of the surface with low curvature grow beyond the threshold keeping polygon counts to a minimum. Therefore, areas of high curvature have larger numbers of small polygons that better model the surface features.

12.5.6 Feature Detection

Edge features within an image are typically identified as a change in intensity from one range to another via an isointensity which depicts the boundary of these two regions.

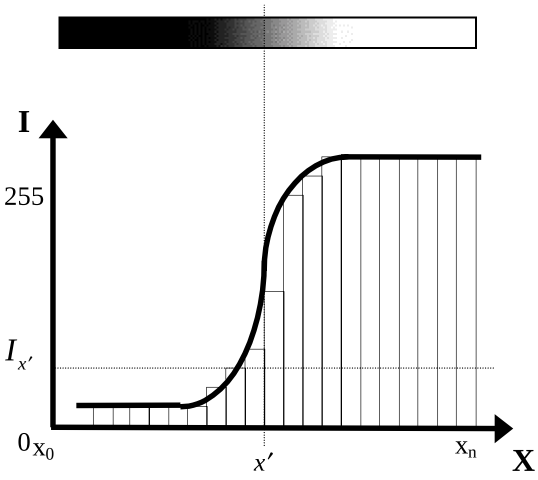


Figure 12.5.3 - The Boundary between Light and Dark

Figure 12.5.3 shows a cross section through an image depicting a sharp boundary between light and dark. The intensity x_i depicts the threshold that would generate an isointensity boundary for this feature within an image. Providing scanning starts within the model boundary, it can be said the boundary (x_i) has been passed when either

$$I_{x'} < I_{x_i} \quad or \quad I_{x'} > I_{x_i} \quad where \quad i = 0,1,\ldots,n$$

depending on the direction of the intensity gradient along the isosurface boundary normal.

This simple thresholding mechanism can be used to detect when the balloon has just passed through a possible isosurface boundary, at which point the inflation force can be removed for that node. Due to the simplicity of this mechanism, many false boundary points are detected and hence results in a noisy segmentation. However, elasticity is a constant force and as such provides the function of a simple momentum term which pulls the nodes away from false boundary points.

Where complicated internal structures are required this approach may not provide adequate results. In this situation, other more sophisticated feature detection schemes can be employed. However as the feature of primary concerned is the external boundary of the model, where a distinct boundary is present, this approach provides an efficient and simple solution.

12.5.7 Robustness to Noise

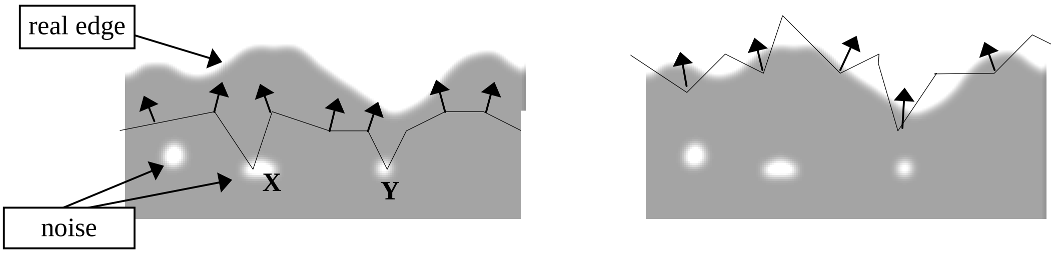


Figure 12.5.4 - Balloon Boundary,

(a) Contour is pulled away from noise (b) Contour oscillates at real edge

Figure 12.5.4 demonstrates this invulnerability to noise spikes. In Figure 12.5.4(a) the boundary moves towards the true boundary through the influence of the inflationary force. Points X and Y are located on noisy areas of the image. Where these false edges are located the inflation force is removed. However, as the remainder of the contour progresses forward under the inflation force the elasticity pulls these points away from the noise. Once a sufficient distance from the noise has been reached the edge detection criteria no longer apply and the inflation force is reapplied. Elasticity then helps smooth these features as the process iterates. Figure 12.5.4b demonstrates what happens when the contour approaches the true boundary. As points are inflated beyond the boundary their inflation force is removed and elasticity pulls the point back within the model, where the inflation force is then re-applied. This causes the contour to oscillate around the true edge. As points oscillate back and forth chaotically their overall movement is at a minimum and therefore mesh subdivision approaches zero. At this point a local edge detection scheme can be used to clamp nodes onto their closest edge. This creates an evenly spaced mesh that is a good surface approximation to the desired object.

12.6 Results

12.6.1 Synthetic Dataset

A synthetic data set of a 3D-horseshoe shape was constructed. The volume consisted of 20x20x6 cubic voxels where each 20x20 slice is identical throughout the volume. Figure 12.6.1 shows one slice from this volume. An initial diamond-shaped seed balloon consisting of 8 vertices is placed inside the object and the model grown to fill the volume. The resulting surface segmentation is shown in Figure 12.6.2. As the model expands to fill the volume, vertices that reach the outer boundary oscillate as their inflation force is turned on and off. The resulting segmentation has almost a circular cross section although the original data had very distinct straight edges. This is due to the tri-linear interpolation which smoothes the data, and is very apparent due to the low number of constituent voxels within the volume. The ends of the model continue to grow under the inflation force and as the distances between vertices increases the dynamic subdivision introduces addition polygons allowing the model to locally deform to fit the dataset.

x →

y ↓

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	255	255	255	255	255	255	0	0	0	0	0	0	0
0	0	0	0	0	0	255	255	255	255	255	255	255	255	0	0	0	0	0	0
0	0	0	0	0	255	255	255	255	0	0	255	255	255	255	0	0	0	0	0
0	0	0	0	255	255	255	0	0	0	0	0	0	255	255	255	0	0	0	0
0	0	0	255	255	255	0	0	0	0	0	0	0	0	255	255	255	0	0	0
0	0	255	255	255	0	0	0	0	0	0	0	0	0	0	255	255	255	0	0
0	0	255	255	0	0	0	0	0	0	0	0	0	0	0	0	255	255	0	0
0	255	255	255	0	0	0	0	0	0	0	0	0	0	0	0	255	255	255	0
0	255	255	0	0	0	0	0	0	0	0	0	0	0	0	0	0	255	255	0
0	255	255	0	0	0	0	0	0	0	0	0	0	0	0	0	0	255	255	0
0	255	255	255	0	0	0	0	0	0	0	0	0	0	0	0	255	255	255	0
0	0	255	255	0	0	0	0	0	0	0	0	0	0	0	0	255	255	0	0
0	0	255	255	255	0	0	0	0	0	0	0	0	0	0	255	255	255	0	0
0	0	0	255	255	255	0	0	0	0	0	0	0	0	255	255	255	0	0	0
0	0	0	0	255	255	255	255	0	0	0	0	255	255	255	255	0	0	0	0
0	0	0	0	0	255	255	255	0	0	0	0	255	255	255	0	0	0	0	0
0	0	0	0	0	0	255	255	0	0	0	0	255	255	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 12.6.1 - Single slice of Synthetic Dataset

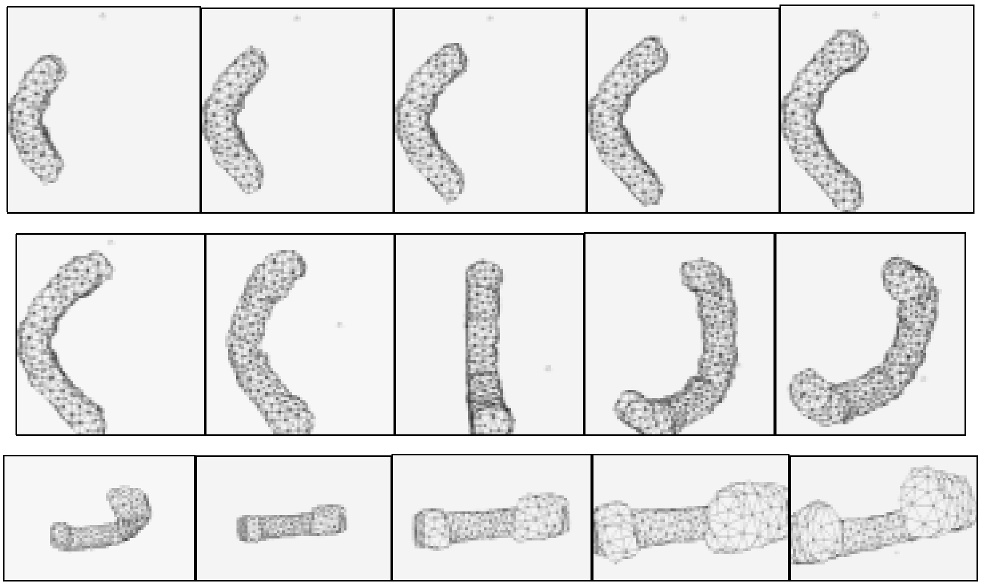


Figure 12.6.2 - Balloon Growing to fill Synthetic Dataset

12.6.2 MRI Dataset

To demonstrate the ability of the balloon model segment a real volumetric dataset the model was applied to a raw MRI scan of a human hand[11]. This is also compared to the results of segmentation gained from a standard isosurface and an 3D elastic mesh model (3D snake). The volume is 256x256x20 voxels in size. This is rescaled by 1x1x2 to reconstruct a cuberville and tri-linear interpolation used to estimate values within the volume. Figure 12.6.3 shows an insosurface generated from the dataset. Although it clearly shows the shape of the hand within the volume the surface is discontinuous and noisy. The background noise in the image is perhaps the most prominent feature and is the cause of the speckled effect of the surface. Another disadvantage of the technique (as mentioned earlier) is that for each voxel, a number of polygons are produced. The isosurface shown in Figure 12.6.3 was generated from a super-sampled volume of 128x128x20 to allow the resulting model to be rendered as a surface generated from the original volume would result in some 235,000 polygons.

[11] MRI data of the hand model was provided by the Centre for Medical Imaging Research (CoMIR) at the University of Leeds

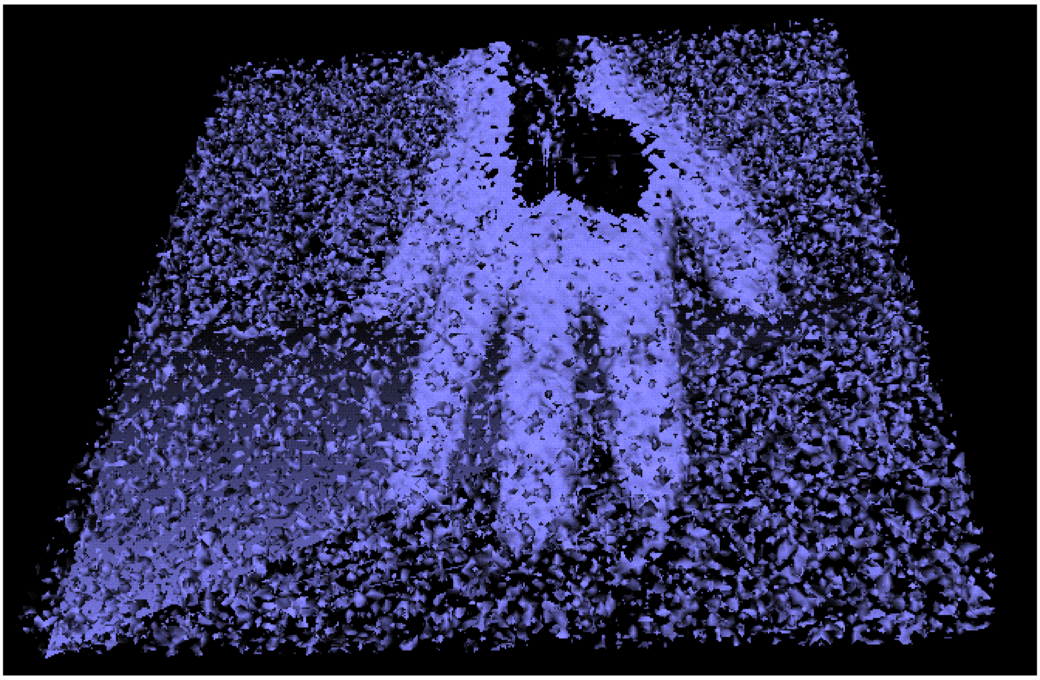


Figure 12.6.3 - Isosurface of MRI Hand Dataset

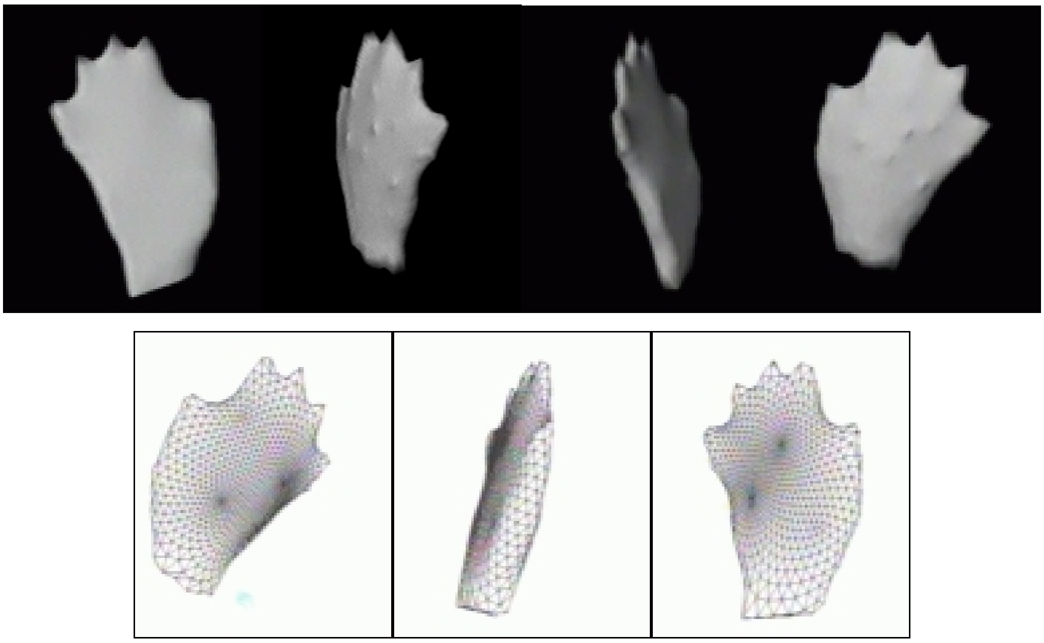


Figure 12.6.4 - 3D Surface Snake Applied to MRI Hand Dataset

Figure 12.6.4 shows the results of applying a 3D elastic surface to the dataset. This produces a poor segmentation for two reasons

1. The large amount of background noise in the volume means that the snake easily gets stuck as it shrinks to fit around the hand.

2. The long narrow features of the fingers make it difficult for the surface to succesfully segment their structure.

By increasing the data attraction force of the snake, the ability to locate and segment the fingers is increased. However, if this attraction force is increased the suseptibility to background noise is also increased and segmentation fails.

Figure 12.6.5 shows the development of the balloon mesh when applied to this dataset. Initially, a seed balloon is placed within the volumetric dataset.

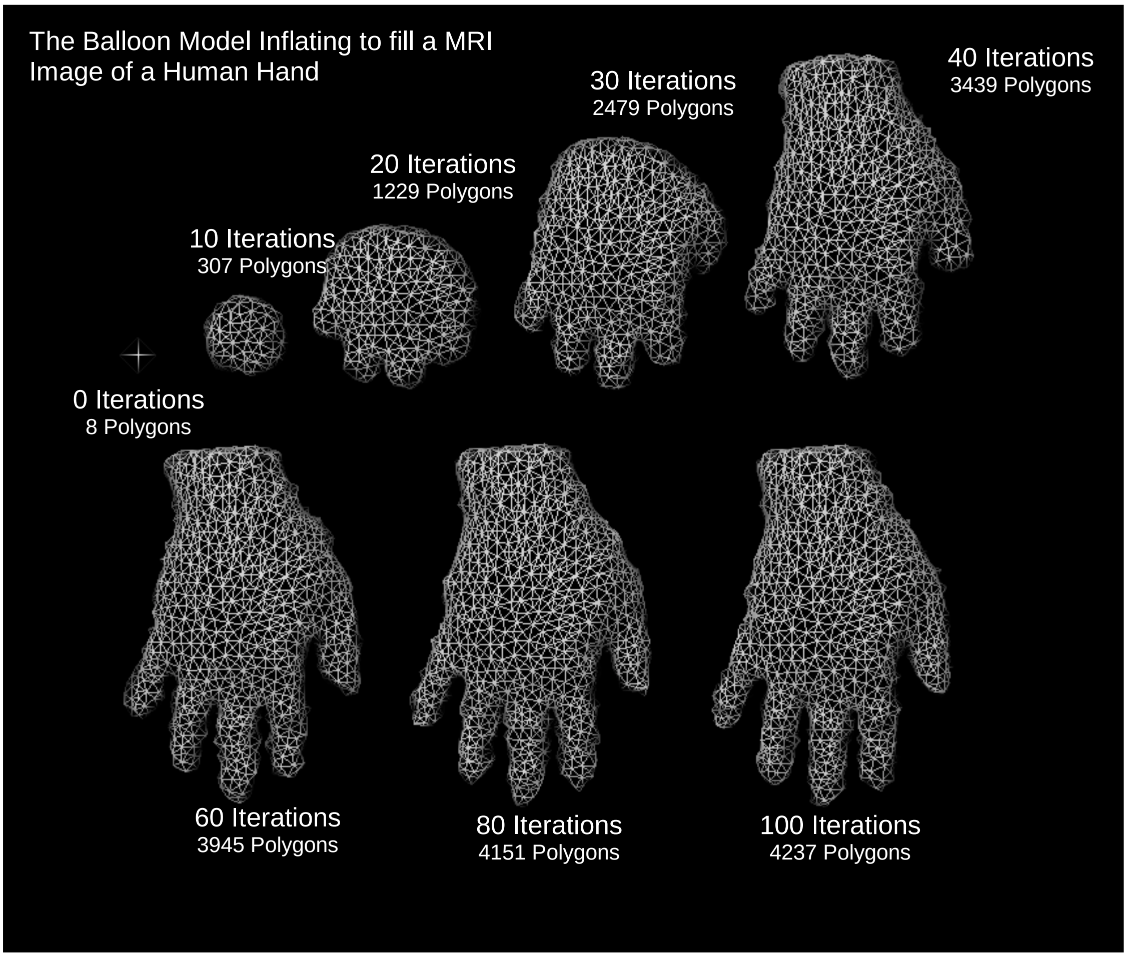


Figure 12.6.5 - Segmentation of an MRI dataset of the Human Hand

The seed consists of a simple diamond shape with 8 polygons and 6 vertices. Forces are applied to the model and after 10 iterations it has grown to 307 polygons. The almost spherical shape is due to the surface tension of the model. Its non-spherical symmetry shows that positive features have been detected early on in the process and thus the inflation force has not been applied evenly. This demonstrates the algorithm's robustness to false boundaries and noise.

As the process iterates further the final shape very quickly starts to take form. Although mesh subdivision continues we can see that it is starting to decrease in rate considerably after 40 iterations. Figure 12.6.6 shows the rate of growth of the mesh.

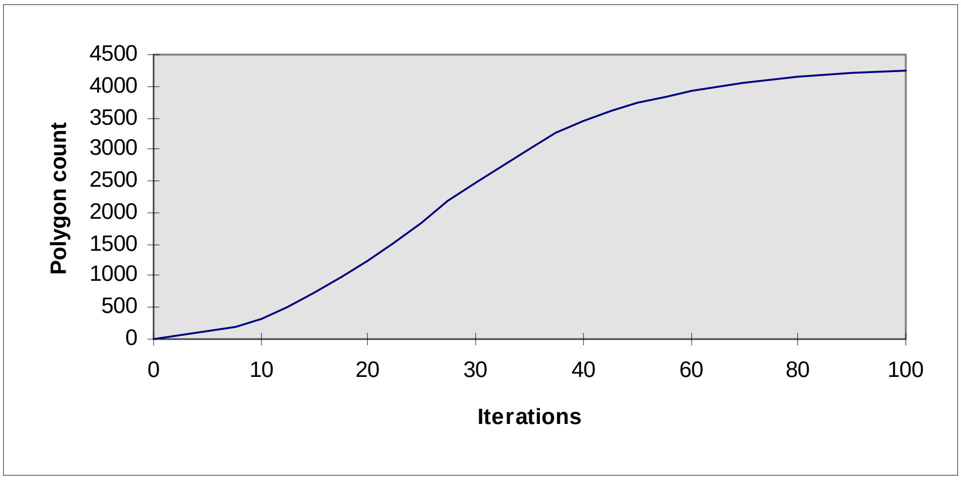


Figure 12.6.6 - Graph Showing the Rate of Polygonal Increase.

Although the model will finally converge on a stable solution, it is sufficiently complete at around 70 iterations which takes approximately 35 seconds on a single MIPS R4400 200MHz processor, including render time. This is significantly faster than previous researchers' techniques, the most comparable being the work of Chen and Medioni [Chen 95], where a comparable complexity model takes approximately 30 mins to iterate on a SUN Sparc-10 machine. This can also be compared with a standard isosurface of the external hand boundary that generated a surface of 235000 polygons as compared to the balloon model of 4000 polygons.

The hand dataset is a good example of the effectiveness of the technique, demonstrating its ability to work with complex noisy images which contain an object with convex, concave and long narrow features.

12.7 Conclusions

This chapter has presented a surface segmentation method which uses a simulated inflating balloon model to estimate structure from volumetric data using a triangular mesh. The model uses simulated surface tension and an inflationary force to grow from within an object and find its boundary. Mechanisms have been described that allow either evenly spaced or minimal polygonal count surfaces to be generated. Unlike previous work by researchers, the technique uses no explicit attraction to data features and as such is less dependent on the initialisation of parameters and local minima. Instead, the model grows under its own forces, never anchored to boundaries but constrained to remain inside the desired object. Results have been presented that demonstrate the technique's ability and speed at the segmentation of a complex, concave object with narrow features, while keeping model complexity within acceptable limits.

12.8 Future Work

This work is ongoing, the primary rationale being the ability to produce low level polygonal surface approximations to allow 3D Point Distribution Models to be built for automatic recognition, segmentation and analysis of volumetric data. Work has also been done in the area of mesh self-intersection. A set of criteria have been developed which allow the detection of mesh self-intersection. Future work includes allowing this criteria to be used to detect intersections, and re-join the mesh at these points to allow more complex torus like shapes to be successfully extracted.