Initialization of Deformable Models from 3D Data

Abstract

The robustness of shape recovery based on deformable models depends in general, on the relative difference of position and topology of the model with respect to the data : a close initialization with correct topology guaranties a proper recovery of the object. Furthermore, the closeness of the initial model greatly influences the time of computation needed for the recovery. In this paper, we propose a method for initializing deformable models from range data or volumetric images. The proposed method solves two distinct problems. First, we use the topological segmentation of volumetric images in order to recover the approximate topology of the object. Second, we use an efficient mesh sampling algorithm to control the number of vertices of the initial model. The method takes into account missing data and outliers.

Key Words: Shape Recovery, Deformable Models, Range Imaging, Medical Imaging.

1 Introduction

In the field of surface reconstruction from tridimensional data, many systems based on deformable models have been developed in the past few years. Those systems are essentially used when the tridimensional data is not perfect. The 3D data consists either of range data or volumetric images. Three types of defects can appear in the data. First, the 3D data may be *noisy* due to the limitation of the accuracy of the range sensor. Second, the data may be *incomplete* because of occlusions in the scene. Finally, there may be *outliers* due to the failure of the acquisition algorithm.

Deformable models are well suited for removing the noise and extrapolating missing data, thanks to the regularizing constraint applied on the model. However, there are very sensitive to the presence of outliers.

Another limitation of deformable models is the *local minimum* problem. This problem originates from the ill-posed nature of surface recovery. To improve the robustness of the recovery, two strategies have been often described in the research community. The first consists in initializing the models close from the object to recover. The second consists in varying the scale of deformation from global to local.

In this paper, we address the first problem by proposing an algorithm for initializing deformable models. We consider that a proper initial model should meet three criteria :

1. **Proper Topology** The model must have the same topology as the object to recover.
2. **Closeness to the data** The model must be close to the data.
3. **Mesh Sparsity** The model must have as few vertices as possible in order to improve the computational cost of the deformation.

Through the analysis of the surface topology, our initialization algorithm identifies missing data as holes in the surface. Furthermore, because we are extracting connected components in volumetric images, we can eliminate most of the outliers. However, pre-segmentation of the dataset may still be required before applying the initialization algorithm.

1.1 Previous Work

Most surface recovery systems based on deformable models, consider as an initial model, a surface primitive such as an ellipsoid or a plane. The model is, in general, initialized manually at the center of the scene. However, several researchers have developed deformable models that can adapt their geometry to the geometry of the object. Chen and Medioni [CM94] use a spring model to recover complex objects from multiple range images. Mc Inerney and Terzopoulos [McI93] as well as Koh and Metaxas [KMB94] uses adaptive finite elements models to refine the model at parts of high curvature.

In addition to refining the deformable model, several deformable models are able to adapt their topology to the topology of the data. Delingette [Del94b] adapts the mesh topology by creating holes or increasing the genus of the surface. Mc Inerney *et al* proposed topologically adaptable snakes [MT95] and surfaces[MT97]. Malladi *et al* [MSV95] proposed a level set approach to representing defomable models. However, the robustness of the topological adaptation of deformable models is dependent on the pre-segmentation of the dataset. We believe that creating an appropriate initial model, is often more adequate than relying on the topological adaptation of deformable models.

Finally, even if deformable models can be geometrically and topologically adapted, the goal of our initialization stage is to provide a considerable speed-up of the recovery, because the cost of initializing the models is much smaller than the cost of deformation from a far position with inedequate topology. Furthermore, even if the data has been pre-segmented, the convergence of a deformable model is not guaranteed if the initialization is not close enough.

Few researchers have fully addressed the initialization problem. Szelisky *et al.* [ST92] initializes a set of oriented particles at each data point in order to recover a smoothed and extrapolated surface. Algorri and Schmitt [AS95] initializes a spring-based deformable model from a volumetric image from a surface tracking algorithm [GU89]. Eck and Hoppe [EH96] initialize a set of B-Spline patches from dense range data of general topology. Curless and Levoy [CL96] create a volumetric image and extract the object of interest as an isosurface mesh.

1.2 Contributions

In this paper, we describe a method for initializing a deformable model of general topology from 3D data. The method applies to deformable models whose surface representation is based on a triangulation or simplex meshes. Our method guaranties the creation of meshes of general topologies with a control over the number of vertices.

The algorithm presented below solves two main problems. The first problem consists in finding a surface topology given some unstructured 3D data. Our original approach has been to embed the surface has a digital surface (volumetric binary image) and to perform the topological segmentation of that surface. This technique is described in section 2. We have implemented a new voxel tracking algorithm that creates 2-manifold and that runs on grey level images.

The second problem consists in creating a mesh with a limited number of vertices from a mesh. We have created a mesh sampling algorithm that efficiently builds a mesh of the same topology as the original mesh, with a prescribed number of vertices. The algorithm is described in section 5. In section 6, we use the meshes created by our method, as initial deformable models for object reconstruction.

Finally, the different algorithms presented in this paper are computationally efficient since our goal is to provide a significant speed-up for shape recovery.

2 Overview of the algorithm

2.1 Deformable Models

Deformable models can have many different surface representations. We only consider here those that are well-suited for representing all types of topologies. Surfaces represented with triangular finite elements for instance have been used by Mc Inerney and Terzopoulos[McI93] and Metaxas[KMB94] to recover surfaces from volumetric images. Vasilescu et al[VT92] have used a set of springs and masses forming a triangular mesh, to extract shapes from range data. Simplex meshes [Del94a] (see also section 4) that dual of triangulations are naturally good candidate for general deformable models.

Other general surface representation includes four-sided patches with varying adjacency at vertices. In [BF96], Borouchaki shows how to build those meshes from triangulations. Hoppe *et al.* [EH96] have used four-sided B-Spline patches to build G^1 continuous parametric surfaces.

2.2 Initialization Scheme

The principle of our initialization scheme is shown in figure 1. The initialization can be decomposed into two stages. The first stage consists in producing a mesh having a topology as close as possible from the desired object to recover. The second stage consists in creating a triangulaion or simplex mesh having as few vertices as desired.

We consider three different types of datasets. Structured range data are stored as parametric surfaces of planar, cylindrical or toroidal topology. Therefore, there is not need to search for their topology. However, we need to isolate the object of interest from the background based on distance or curvature distribution.

On the contrary, for unstructured range data and volumetric images, we need to detect the surface topology after eventually performing some pre-segmentation.

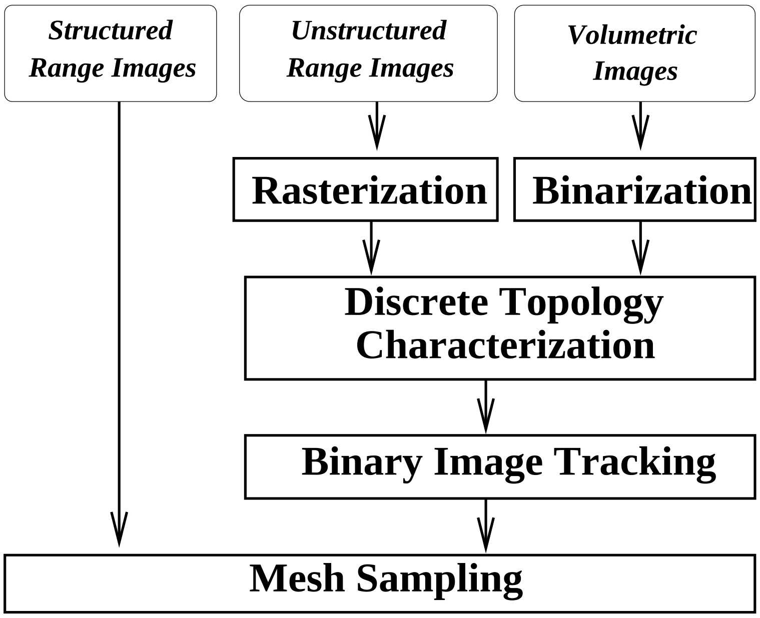


Figure 1: The initialization scheme

3 First Stage : Topology Segmentation

3.1 Notion of 2-Manifold

We define here the notion of manifold that will be used along the paper. Our goal is to create a deformable model corresponding to either a triangulated mesh suited for finite elements computation or a simplex mesh. In both cases, the mesh resulting from the first stage must be a 2-manifold.

We first define a mesh with a set of vertices $\mathcal{V} = \{V_i\}$,$i = 1 \dots n_v$, a set of edges $\mathcal{E} = \{(V_{p(i)}, V_{q(i)})\}$, $i = 1 \dots n_e$ and a set of faces $\mathcal{F} = \{(V_{p[i][0]}, V_{p[i][1]}, \dots, V_{p[i][n-1]})\}$, $i = 1 \dots n_f$. A mesh is a 2-manifold, if it obeys the following 3 rules :

1. Each face is homeomorphic to a disc.

2. Two faces intersect along at most one edge.

3. The neighborhood of each vertex must be homeomorphic to a disc or a half-disc.

To those three rules, we add a fourth rule :

4. Each face has at least 3 vertices.

Given a mesh, we can generate a conformal triangulation (suited for finite elements computation) if and only it is a 2-manifold.

3.2 Initialization from Structured Range Data

In general, range finders delivers range images, where at each pixel $P(i,j)$ is stored the X,Y and Z coordinate in sensor coordinate frame. The image forms a natural parameterization of the surface that can be described in a parametric manner : $X(i,j)$, $Y(i,j)$ and $Z(i,j)$. However, in general, there are many pixels where no tridimensional information is available, because of the failure of the acquisition algorithm.

From this structured data, we do not need to solve the surface embedding problem. However, it may be required to isolate the object of interest from the background. We propose a two stages method :

1. **Computation of jump discontinuities**

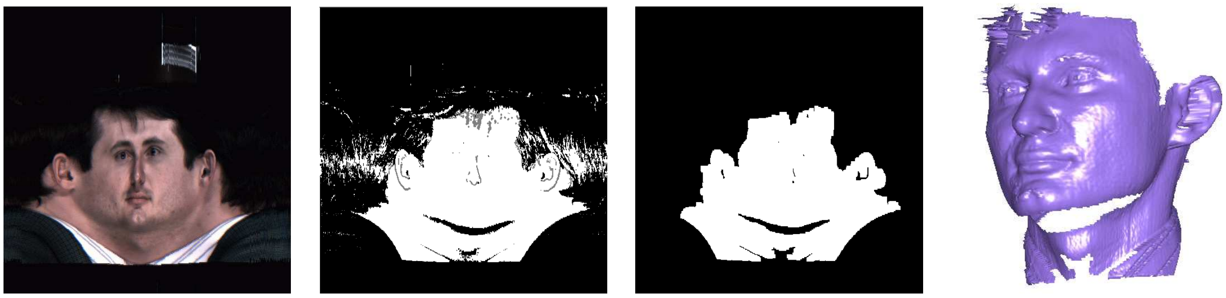
2. **Extraction of the main connected component**

The first stage consists in finding pixels corresponding to jump discontinuities. Those pixels often correspond to boundaries between two neighboring objects or between an object and the background. Those pixels can be detected by computing the maximum distance between a pixel and its 4 neighbors. With a simple thresholding technique, we can segment the jump discontinuity voxels.

We then extract the main connected component of pixels that are not labeled as jump discontinuities. It is often desirable to performed several mathematical morphology operation such as openings to clean the mask image.

Other methods of range image segmentation could have been used. A survey of range image segmentation is available in [TF].

Figure 2 shows the different stages on a Cyberware range image. The resulting mesh has 65077 vertices and 63768 square faces. Those operations naturally eliminates outliers.



(a) (b) (c) (d)

Figure 2: (a) The texture associated with the range image; (b) The occupancy grid where a 3D range data is available; (c) The grid after connected component computation, jump discontinuities removal and after several openings; (d) The corresponding mesh surface.

3.3 Initialization from 3D Unstructured Data : Rasterization

In an unstructured dataset, no information about the connectivity between tridimensional points is known. This unstructured data may originate from a cloud of points or the merging of several range images into the same world-centered coordinate frame. The natural parameterization of the image cannot be used anymore to represent the object.

In this case, it is problematic to recover the embedding of the surface, ie its topology. We propose to transform those 3D points into a tridimensional image and perform topological segmentation of that image.

The process of creating a volumetric image from 3D data has been studied especially for merging several range images in the same coordinate frame. Curless [CL96] uses the signed distance to the tangent plane in order to create a volumetric image with sub-voxel accuracy. Because our aim is not to reconstruct directly the surface, but to initialize a deformable model for the reconstruction, we use a simpler method.

When merging several range images, we compute the projection of each datapoint in the image. If we are dealing with several range images, we draw a 3D line between adjacent pixels based on the 3D Bresenham line method. Otherwise, depending on the resolution of the dataset, we first create a grey-value image by drawing, for each 3D point, a box of voxels centered around the projected voxel. Each voxel inside that box, has a value decreasing with its distance to the 3D point. The binary image is created by thresholding the grey value image.

All volumetric images are created with isotropic voxels (same size in the three direction). The three orthogonal directions of the volumetric image are computed based on its moments of inertia. The voxel size is chosen as a function of the resolution of the dataset. In practise, we found that a voxel size equal to 10 times the dataset resolution gives good results. Furthermore, it seems that the choice of the voxel size is not critical because there is a large range of voxel size values for which the results are satisfactory. The volumetric images are in general of small size (less than $50 \times 50 \times 50$) which ensures efficient computation.

In figure 3, we show a volumetric image created from a set of four range images. Those range images [1] were acquired with a $90°$ increment around the object. Large portion of the object is missing. We choose a voxel size of 0.004988 and the size of resulting image is $29 \times 38 \times 33$.

3.4 Initialization from Volumetric Images : Binarization

Given a grey-level volumetric image such as a CT-scan image or a MRI image, we need to create a binary image corresponding to the object of interest. There are several means to create a binary image

[1] Those images correspond to the files bun000.ply bun090.ply bun180.ply and bun270.ply available at http://www-graphics.stanford.edu/data/3Dscanrep/

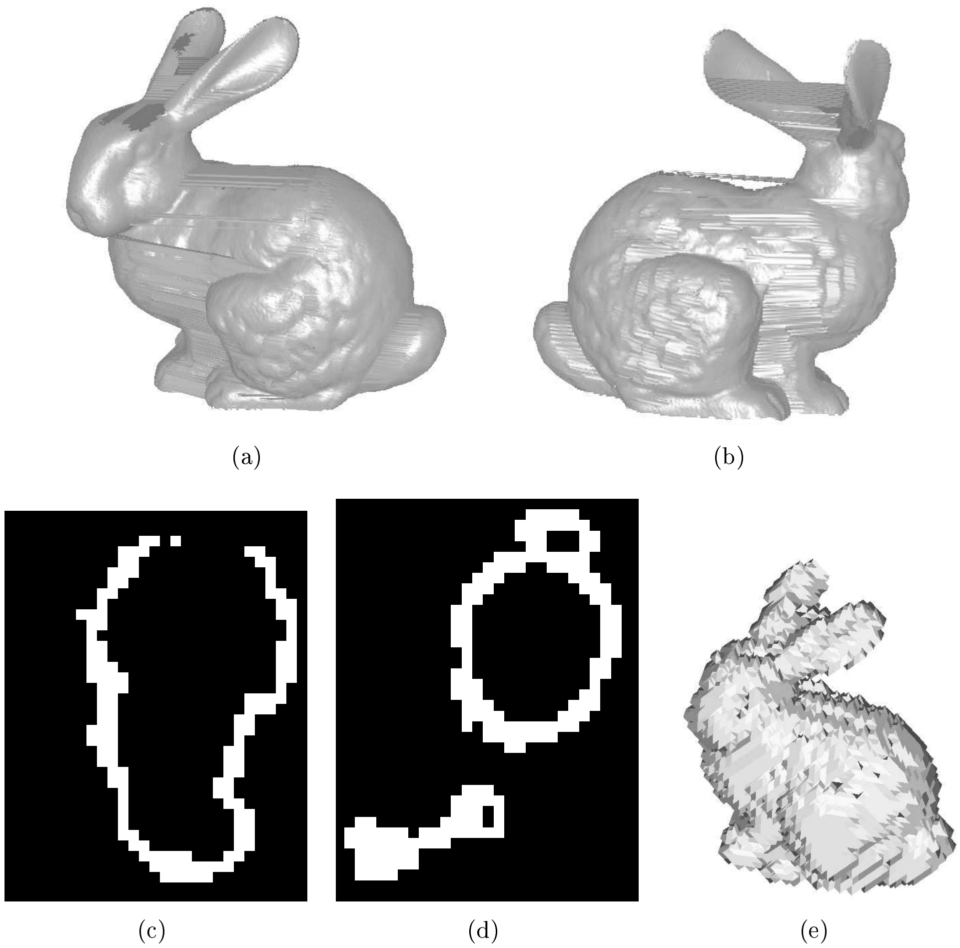


Figure 3: (a) and (b) View of the four ranges images : the description of the object is uncomplete; (c) and (d) Slice of the volumetric image; (e) View of the volumetric image.

approximating an object. Thresholding and contour extraction are the two techniques mainly used in volumetric image processing (see figure 4). Thresholding creates a binary image representing the *volume* of the object whereas contour extraction produces binary images representing the *surface boundary* of the object. Those techniques can be combined with mathematical morphology operators to produce a rough estimate of shape of the object.

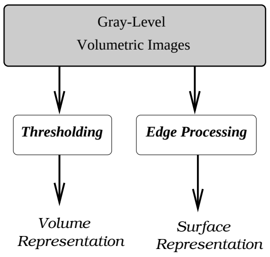


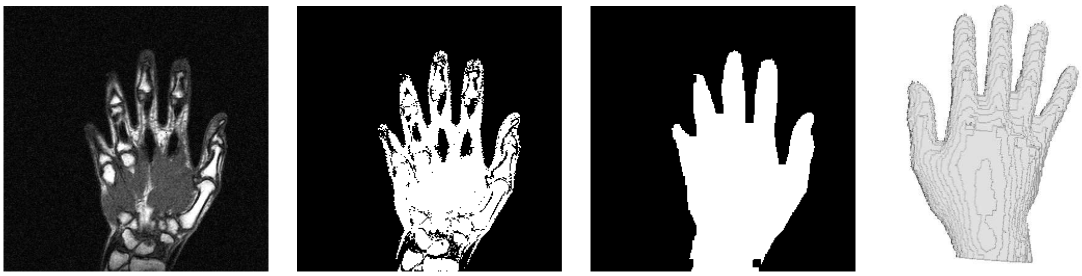
Figure 4: The different scheme for creating a binary image of the region of interest from a volumetric image.

When the binary image represents a surface boundary, we operate in exactly the same way as a binary image created from unstructured range data. When the binary image represents the whole volume of the object, it means that the boundary is necessarily close and therefore we do not need to detect border voxels. We then simply apply the connectivity correction algorithm followed by the tracking algorithm.

In figure 5, we show an example of thresholding combined with several 2D closures of the binary image. The hand model cannot be extracted with a simple isosurface technique because of the bias existing in the MR image.

3.5 Discrete Topology Characterization of Digital Surfaces

The binary volumetric image extracted from unstructured range data or volumetric images represents either a digital volume as in figure 5 (c) or a digital surface as in figure 3 (e). With digital volumes, there is no need for detecting topological characteristics. We can directly apply the voxel tracking algorithm to find the mesh enclosing the volume.



(a) (b) (c) (d)

Figure 5: (a) A 512×512×20 MRI image of a human hand; (b) Binary image after thresholding; (c) Binary image after performing several closures; (d) The mesh extracted from the tracking algorithm.

With digital surfaces however, we need to find the border of the surface. Otherwise, we would create an initial model with both sides of the surface.

3.5.1 Thinning the Digital Surface

From the volumetric binary image, we first extract the main 18-connected component. We then thin the image using the algorithm of Bertrand and Malandain[Ber95]. Thinning is necessary to make the detection of boundary voxels more robust. The thinning algorithm is actually based on the topological segmentation described in section 3.5.2. Basically, the algorithm operates by iteratively removing *volume voxels* until only *surface, border and junction* voxels can be found. The thinning algorithm usually removes only a few voxels because the image was created as the binary image of a digital surface. The processing takes in general less than 5 seconds on a DEC Alphastation 200/233.

3.5.2 Topological Segmentation of the Digital Surface

We use the topological segmentation of Malandain and Bertrand[MBA93] to label voxels according to their topology. Each voxel is labeled as a *surface voxel*, a *border voxel* or a *junction voxel* (see figure 6). The labelling of a voxel is relative to a choice of connectivity and is based on two numbers : the number of connected components of the background $\hat{C}$ and the number of connected components when the voxel is removed $C^\star$. For instance *surface voxels* are characterized by $\hat{C} = 2$ and $C^\star = 1$.

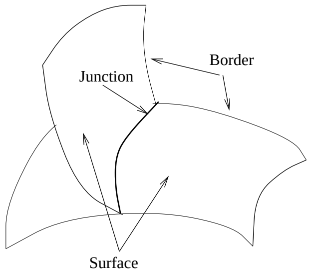
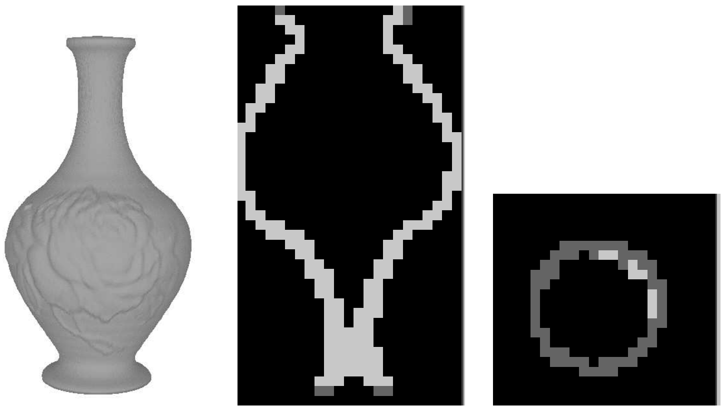


Figure 6: The 3 types of topological characteristics

The output of this topological segmentation is a volumetric image with three grey levels for each foreground voxel. In figure 7 we show the resulting image on a digital surface representing a vase. The two extremities of the vase are correcly labels as border voxels.



(a) (b) (c)

Figure 7: (a) Range data of a vase; (b) and (c) are two orthogal slices after the topological segmentation. Dark grey level indicates border voxel while light grey indicate surface voxels.

3.6 Binary Image Tracking

Surface tracking from volumetric binary images have been developed essentially for display and quantitative measurements on segmented data. The surface tracking outputs a set of rectangular faces corresponding to the "outside" facing voxels. Gordon and Udupa[GU89] have devised a fast surface tracking algorithm that is well-suited for tracking complex topological surfaces.

We do not use the surface tracking of Gordon *et al* for three reasons. First, we do not apply the tracking algorithm on a binary image but on a labeled image. The tracking algorithm must track only the surface voxels and stop when it meets border or junction voxels. Second, the surface sampling algorithm described in section 5 needs a data structure that is more complete that the voxel indexing use by Gordon. Our algorithm builds a data structure where the 4 neighboring are stored during the tracking. Finally, we need to create a mesh that is a 2-manifold following rules of section 3.1.

The tracking takes place by first selecting a surface voxel adjacent to the background. We then propagate the selection to neighboring faces in a recursive manner the by considering only 4 faces adjacent to the current face. If the face does not belong to a surface voxel, we stop the propagation.

The set of faces thus extracted is a mesh that is not necessarily a manifold. In figure 8, we show different configurations where the rules of section 3.1 do not apply.

We transform a non-manifold mesh into a manifold mesh by first processing the vertices whose neighborhood is homeomorphic to two discs (for instance 8 (a) and (b)). At those vertices, we create a new vertex that will be connected to one of the neighborhood of the original vertex. The position of the new vertex is the same than the original vertex.

After removing all pathological vertices, we have to remove pathological edges where four faces are adjacent. It can be demonstrated that only three configurations are possible. They are shown in figure 8 (c). To create a 2-manifold, we create an additional edge by duplicating the pathological edge.

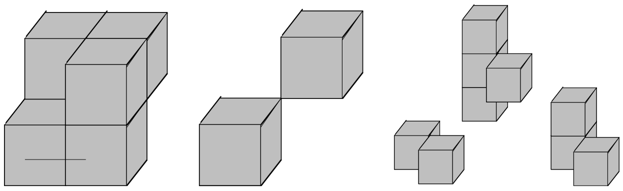
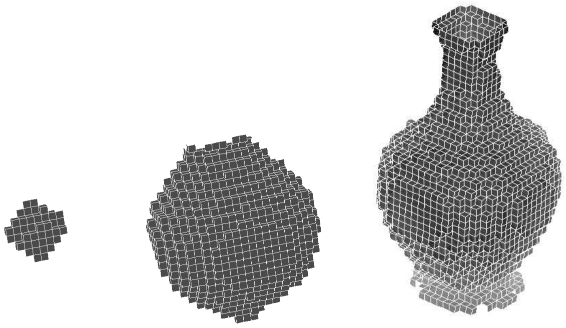


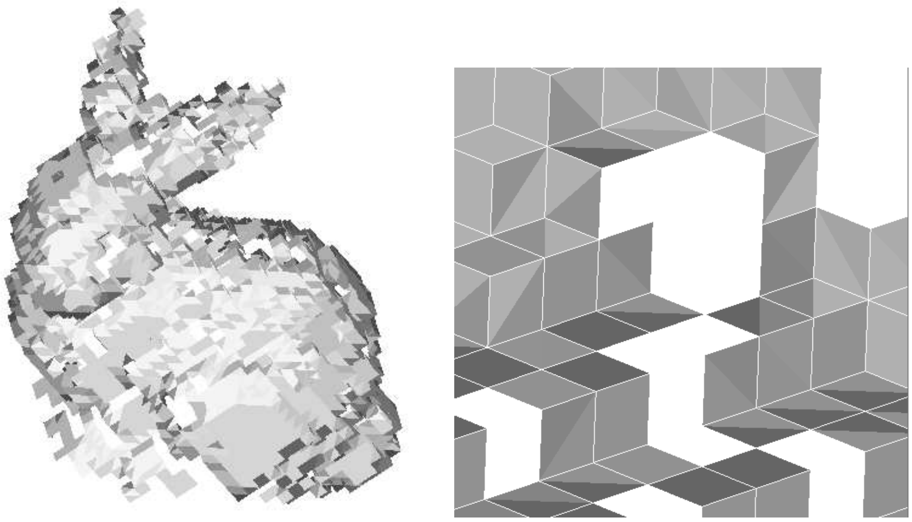
Figure 8: The 3 configurations where the mesh is not a manifold : in (a) and (b) the neighborhhod of a vertex is homeomorphic to two discs. In (b) 4 edges are adjacent to the same edge

In figure 9, we show the surface extracted with the tracking algorithm at different iterations. Without the proper topological segmentation, we would have obtained the inside and the outside of the surface. In figure 10, several holes have been created due to the missing data in the original image.



(a) (b) (c)

Figure 9: (a) Surface tracking of the vase image after 4 iterations; (b) surface tracking after 20 iterations; (c) Final mesh.



(a) (b)

Figure 10: (a) The bunny mesh after surface tracking (b) Details of holes in the surface mesh.

4 Simplex Meshes

In this section, we briefly introduce the notion of simplex mesh (see [Del94a] for more details). Those meshes are handled at two different levels in this paper. On one hand, in the mesh sampling algorithm described further, simplex meshes are used as surface representation dual of a triangulation. On the other hand, simplex meshes are used as deformable models in the section 6.

A k-simplex mesh is defined as a union of p-cells ($p < k$). The notion of cells are described in figure11(a). In this paper, we will only use 2-simplex meshes as the representation of 2-manifolds. Simplex meshes have two important properties. First, a simplex mesh is simply connected : the vertices of a 2-simplex mesh have 3 neighbors. Second, a simplex mesh has a dual structure with triangulations. To create a simplex mesh from a triangulation, we associate a vertex of a simplex mesh with a triangle and a face with a vertex of the triangulation (see figure 11(b)).

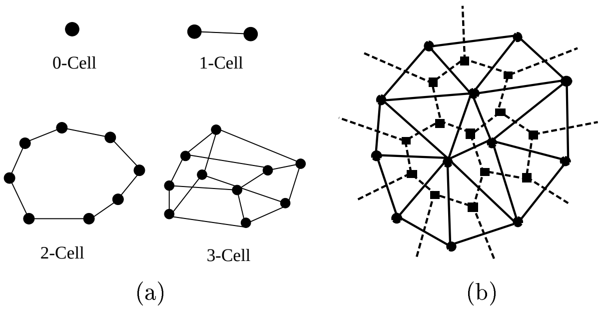


Figure 11: (a) p-cells of a simplex mesh; (b) The duality principle between a triangulation and a simplex mesh.

Simplex meshes are surface representation well-suited for representing deformable surfaces. Unlike most deformable surface models, simplex meshes are not parametric models. They can handle all types of topologies and in particular, they include the notion of deformable contours which is useful to represent the boundary of surfaces.

A vertex P_i of a deformable simplex mesh is moved according to the law of motion :

$$m\frac{d^2 P_i}{dt^2} = -\gamma\frac{dP_i}{dt} + \mathbf{F}_{int} + \mathbf{F}_{ext} \tag{1}$$

$\mathbf{F}_{int}$ is the internal force enforcing the smoothness of the mesh. Several orders of smoothness constraints can be applied : position continuity, surface orientation continuity, simplex angle continuity, or shape memory constraint. $\mathbf{F}_{ext}$ on the other hand, is proportional to the distance of P_i to the 3D data, and directed along the normal of the mesh at P_i.

5 Second Stage : Mesh Sampling

5.1 Sampling Scheme

The mesh sampling algorithm aims at creating a mesh with the same topology than a given mesh but with fewer vertices. If the original mesh is a 2-manifold then the sampled mesh is guaranteed to be a 2-manifold. The resulting mesh may be a simplex-mesh or a triangulation.

We believe that our mesh sampling scheme is better suited than existing decimation algorithm. Decimation algorithm usually involves backtracking and they are based on the computation of curvature or distance to the data. In terms of computational efficiency, the computation time is proportional to the number of vertices to be removed.

The complexity of the mesh sampling algorithm is solely dependent on the final number of vertices, independently of the initial number of vertices. This is clearly advantageous when considering decimation rate of 95% in average. Furthermore, the mesh sampling is based only on the mesh topology not on the mesh geometry. It implies that we do not compute any geometric quantity such as distance or curvature. The algorithm proceeds by directly creating vertices, edges and finally faces.

There exists other direct methods that build a mesh from a polyhedron. The most closely related work is the Voronoï diagram built on a polyhedron[Mou85]. Eck *et al.*[EDDH95] uses an approximation of the geodesic distance to build a Voronoï diagram on a polyhedron.

Our method does not use any notion of geometric distance, but only the topological distance of adjacency. We found that relying on solely topology was sufficient to produce initial models. The introduction of geometric constraints (for instance distance to the original mesh) would entail additional processing and therefore would not suit our purpose.

The principle of the sampling algorithm is the following : given a mesh representing a 2-manifold, we randomly select a number n of faces. We then propagate the labels of those site faces until we realize a pavement of the mesh into n regions. If all regions follow the 4 rules of section 3.1, then we extract a simplex mesh based on those regions. Otherwise, we modify the site faces, and iterates until a mesh is created.

More precisely, the sketch of the algorithm is the following:

Initialize-Site-Faces()

repeat

    Grow-Regions()

    Extract-Vertices()

    if Test-Vertices() = TRUE **then**

        Extract-Edges()

        if Test-Edges() = TRUE **then**

            Mesh-is-Correct = TRUE

        else

            Modify-Site-Faces-From-Edges()

Mesh-is-Correct = FALSE

end if

else

Modify-Site-Faces-From-Vertices()

Mesh-is-Correct = FALSE

end if

until Mesh-is-Correct = TRUE

Build-Simplex-Mesh()

The algorithm presented above can be applied to any type of mesh, independently of the number of vertices per face. The only requirement is that the mesh represents a 2-manifold. Furthermore, the data structure requires the knowledge of face adjacency and that all faces are oriented consistently.

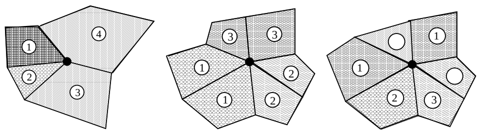
5.2 Creation of Region Vertices

Given a mesh, we randomly select n site faces. We attach to those site faces a label, and then iteratively propagate those labels until all faces are labeled. We call *region* the set of faces having the same label (see figure 13).

Vertices of the regions correspond to vertices of the original mesh where three or more regions meet. For each face, we test the presence of the 2 cases shown on figure12 (a) and (b). In the first case, a face is adjacent to two faces belonging to two different regions along two consecutive edges. In the second case, we need to test two or three adjacent faces to detect the vertex. In general, a vertex is surrounded with three regions. In practise, we have found that less than 10% of the vertices are surrounded with four regions, and almost none with more than four regions.

After creating those vertices, we test if their neighborhood is homeomorphic to a disc or a half disc.

If it is not the case, as in figure 12 (c), we create a new site face around the problematic vertex.



(a) (b) (c)

Figure 12: (a) and (b) two configurations for the creation of a region vertex; (c) Example of a vertex whose neighborhood is not homeomorphic to a disc or a half disc

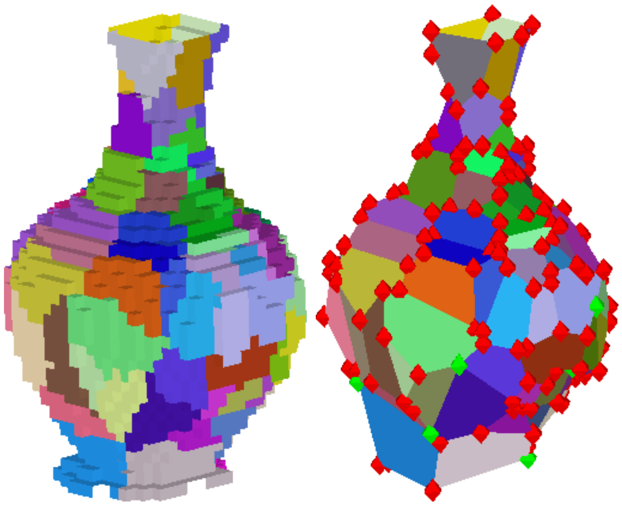
5.3 Creation of Region Edges

An edge between two regions is a set of adjacent edges of the initial mesh separating two regions. Topologically, an edge can be closed or open.

We start by building the open edges. We scan all region vertices, and track all region edges starting from each region vertex. During the tracking, we label the edges between two faces that have been processed (see figure 13(b)). We then build the closed region edges by detecting unprocessed edges between two faces. We track those edges until we complete the loop around the region edge.

After creating the edges, we test if the sampled mesh is a 2-manifold. We first check the existence of closed edges, and add 3 site faces along close edges. We then compute for each region, the number of closed loops bordering the region to check if it is homeomorphic to a disc. When more than one loop is detected, we add one site face per closed loop. Finally, we test if two regions intersect along more than one edge and then create an additional face site per edge.

If we cannot build a mesh representing a 2-manifold, we recreate the regions based on the new set of site faces, and iterate. The algorithm is guaranteed to succeed because in the worst case, it will create as many site faces as the number of faces in the original mesh. In general, no more than 10 iterations are needed.



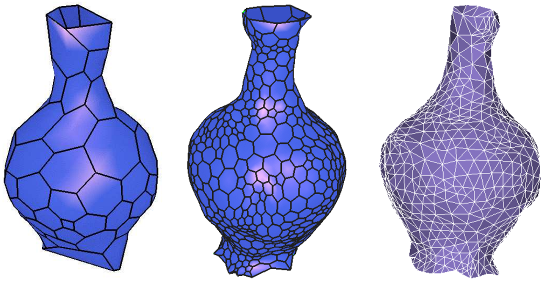
(a) (b)

Figure 13: (a) The regions grown from the site faces; (b) The sampled mesh after the creation of region edges

5.4 Creation of a Simplex Mesh

The sampled mesh representing a 2-manifold has vertices adjacent to 3 or more regions. To produce a simplex mesh and subsequently a triangulation, we create a new region with p vertices at vertices adjacent to p regions, $p > 3$. The new vertices are positioned at the middle of the adjacent region edges.

After the creation of the mesh, we perform two operations, edge swapping and smoothing, aimed at improving the mesh quality in terms of geometry (smoothness) and topology. Edge swapping consists in iteratively swapping edges adjacent to large faces. It ensures that all faces have a number of vertices per face close to 6. Smoothing is performed locally by considering the mesh as a deformable model under the only influence of internal forces. The simplex mesh can then be triangulated using the duality transformation (see figure 14 (c)). In figure 14, we show two sampled meshes from the same original mesh. The decimation rate corresponds respectively to 96% and 80% of the original number of faces.



(a) (b) (c)

Figure 14: (a) The simplex mesh after edge swapping and smoothing; (b) Same as (a) with more vertices; (c) Triangulation extracted from (b)

6 Results

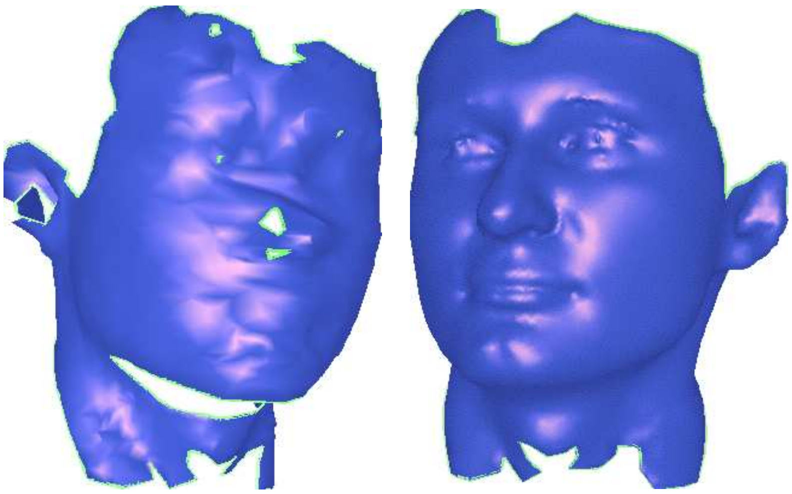
6.1 Initialization from Structured Data

In figure 15 (a), we show the initial model created from the mesh of figure 2(d). The original mesh extracted from a Cyberware range data, had 63768 facets and 65077 vertices. The sampled mesh of figure 15 (a) has 696 faces and 1388 vertices, corresponding to a decimate rate of 97.9%. The computation time needed for sampling the mesh was 6.29 seconds on a Dec Alphastation 200 4/233 with 3 iterations. The initial mesh has 9 holes created at parts where the range data was not complete (see figure 2(c)).

Starting from this initial mesh, we have built a compact, complete, smooth and accurate mesh representing the original data. We have chosen to extrapolate the missing parts by removing 8 of the 9 holes based on their area. We have then automatically refined the simplex mesh based on its curvature. Details about the refinement of simplex meshes are provided in [Del94a]. The refined and extrapolated mesh is shown in figure 15 (b). The mesh has 1962 faces and 3920 vertices.

6.2 Initialization from Unstructured Data

In figure 16 (a) we show the result of the mesh sampling on the mesh of figure 10. We chose a small decimation rate of 40% in order to get enough data points at the level of the ears. The mesh of figure



(a) (b)

Figure 15: (a) The initial simplex mesh created with 9 holes; (b) The deformable model after refinement and extrapolation of 8 holes.

figure 16 (a) is a simplex mesh with 5982 vertices. This mesh has 26 holes and we keep the hole with the largest number of vertices to get a mesh with the proper topology as seen in figure 16 (b) and (c). Finally, the simplex mesh has been refined at the ears based on the distance to the data. Further refinement could have been applied to increase the accuracy of reconstruction. The mesh of figure 16 (d) has 9038 vertices. The numerous holes of the dataset have been interpolated by the deformable model. The total processing time for generating an initial model (including rasterization, topological segmentation, and mesh sampling) was less than 1mn 30s.

6.3 Initialization from Volumetric Images

From the original mesh corresponding to figure 5 (d), we have extracted a sampled simplex mesh. The number of faces of the original mesh where 57668 and the sampling algorithm has needed 5 iterations to build the new mesh with 1316 faces. This initial simplex mesh have been deformed under the influence of external forces computed from the gradient. The deformed mesh is shown in figure 17 with the trace of the mesh on an image slice.

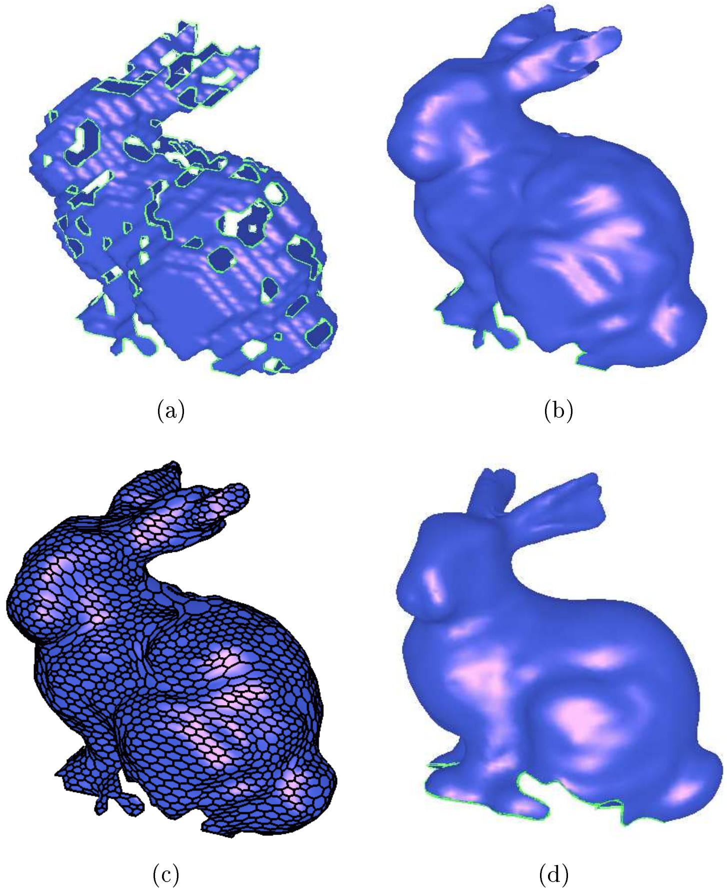
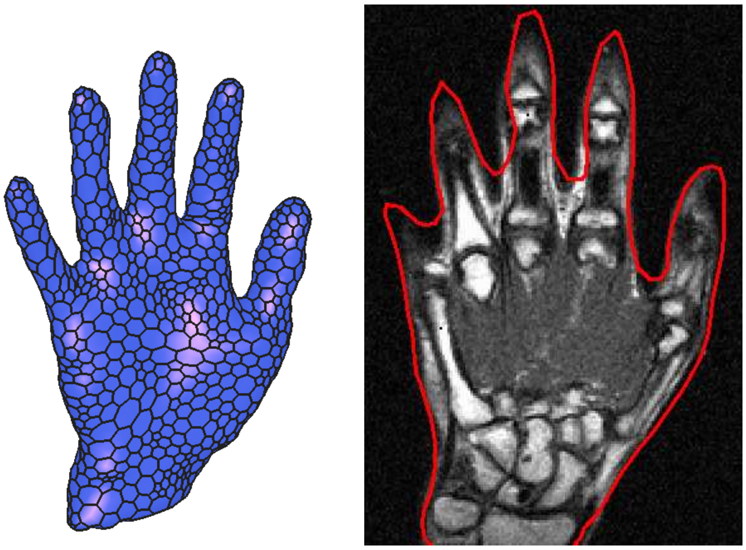


Figure 16: (a) sampled mesh at 40%; (b) and (c) Simplex mesh after removal of small holes; (d) Final reconstructed model



(a) (b)

Figure 17: (a) The mesh after it was fit on the image; (b) A slice of the image with the reconstructed hand.

6.4 Initialization from Serial Slices

We have applied our initialization scheme to the problem of reconstruction from serial slices. In this case, the original data is a volumetric image consisting of a set of closed contours, manually extracted from a medical image. Because of the inter-slice distance is usually very important compared with the pixel size, a successful approach has consisted in building a mesh from planar Delaunay triangulations[BG93]. However, the reconstruction is not fully tridimensional since all vertices of the triangulation are located on planar slices. We have successfully applied our method on this dataset and we have built real tridimensional meshes that approximate smoothly the set of planar contours (see figure 18). The resulting mesh has 6452 vertices and 3194 faces.

7 Conclusion

The combination of topological segmentation, digital surface tracking and mesh sampling provides a powerful tool for building surface models without a priori knowledge. We believe that a volumetric

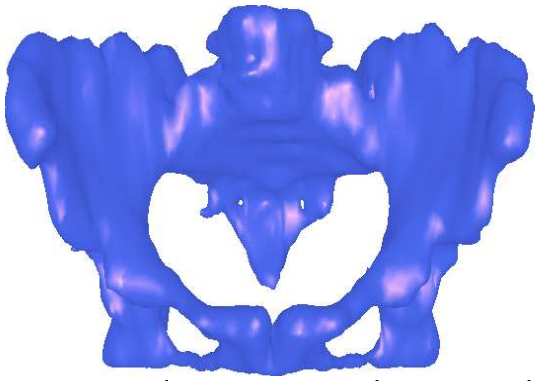


Figure 18: Reconstruction of a pelvis model from a set of planar contours.

image is an effective data structure for the analysis of complex scenes. We would like to extend the analysis of those scenes by including some notion of geometry such as curvature.

Acknowledgments

We would like to thank Tony Heap (Leeds University) for providing the MRI hand image, and Bernhard Geiger (Siemens) for providing the slice contours.

References

[AS95] M.-E. Algorri and F. Schmitt. Deformable models for reconstructing unstructured 3d data. In *First International Conference on Computer Vision, Virtual Reality and Robotics in Medicine*, pages 420–426, Nice, April 1995.

[Ber95] G. Bertrand. A parallel thinning algorithm for medial surfaces. *Pattern Recognition Letters*, 16:979–986, 1995.

[BF96] H. Borouchaki and P.J. Frey. *Adaptive Triangular-Quadrilateral Mesh Generation*. Technical Report 2960, INRIA, Sophia-Antipolis, France, August 1996.

[BG93] J.D. Boissonnat and B. Geiger. Three dimensional reconstruction of complex shapes based on the delaunay triangulation. In R.S. Acharya and D.B. Goldgof, editors, *SPIE Conference on Biomedical Image Processing and Biomedical Visualization*, San Jose, CA, February 1993.

[CL96] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Computer Graphics (SIGGRAPH'96)*, 1996.

[CM94] Y. Chen and G. Medioni. Surface description of complex objects from multiple range images. In *Proc. of Int. Conf. on Computer Vision and Pattern Recognition (CVPR'94)*, pages 153–158, Seattle, USA, June 1994.

[Del94a] H. Delingette. *Simplex Meshes: a General Representation for 3D Shape Reconstruction.* Technical Report 2214, INRIA, March 1994.

[Del94b] H. Delingette. Simplex meshes: a general representation for 3d shape reconstruction. In *Proc. of Int. Conf. on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, USA, June 1994.

[EDDH95] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In *Computer Graphics (SIGGRAPH'95)*, pages 173–182, 1995.

[EH96] M. Eck and Hugues Hoppe. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In *Computer Graphics (SIGGRAPH'96)*, 1996.

[GU89] D. Gordon and J.K. Udupa. Fast surface tracking in three-dimensional binary images. *Computer Vision, Graphics, and Image Processing*, 45:196–214, 1989.

[KMB94] E. Koh, D. Metaxas, and N. Badler. Hierarchical shape representation using locally adaptive finite elements. In Jan-Olof Eklundh, editor, *3rd European Conference on Computer Vision (ECCV'94)*, pages 441–446, Stockholm, 1994.

[MBA93] G. Malandain, G. Bertrand, and N. Ayache. Topological segmentation of discrete surfaces. *International Journal of Computer Vision*, 10(2):183–197, 1993.

[McI93] D. McInerney, T. Terzopoulos. A finite element model for 3d shape reconstruction and nonrigid motion tracking. In *Proc. of the Fourth Int. Conf. on Computer Vision (ICCV'93)*, pages 518–523, 1993.

[Mou85] D.M. Mount. *Voronoi Diagrams on the Surface of a Polyhedron.* Technical Report CS-TR-1496, University of Maryland, May 1985.

[MSV95] R. Malladi, J. Sethia, and B. Vemuri. Shape modeling with front propagation: a level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, 1995.

[MT95] T. McInerney and D. Terzopoulos. Topologically adaptable snakes. In *Proc. of the Fifth Int. Conf. on Computer Vision (ICCV'95)*, pages 840–845, IEEE Computer Society Press, Cambridge, MA, June 1995.

[MT97] T. McInerney and D. Terzopoulos. Medical image segmentation using topologically adaptable surfaces. In J. Troccaz E. Grimson R. Mosges, editor, *First Joint Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed-MRCAS'97)*, pages 23–32, Greboble, France, March 1997.

[ST92] R. Szeliski and D. Tonnesen. Surface modeling with oriented particle systems. In *Computer Graphics (SIGGRAPH'92)*, pages 185–194, July 1992.

[TF] E. Trucco and R.B. Fisher. Experiments in curvature-based segmentation of range data.

[VT92] M. Vasilescu and D. Terzopoulos. Adaptative meshes and shells. In *Proc. of Int. Conf. on Computer Vision and Pattern Recognition (CVPR'92)*, pages 829–832, 1992.