Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volume Data

James V. Miller David E. Breen William E. Lorensen†
Robert M. O'Bara Michael J. Wozny

Rensselaer Design Research Center
Rensselaer Polytechnic Institute

†General Electric Company
Corporate Research and Development

ABSTRACT

We propose a new approach to the problem of generating a simple topologically-closed geometric model from a point-sampled volume data set. We call such a model a Geometrically Deformed Model or GDM. A GDM is created by placing a 'seed' model in the volume data set. The model is then deformed by a relaxation process that minimizes a set of constraints that provides a measure of how well the model fits the features in the data. Constraints are associated with each vertex in the model that control local deformation, interaction between the model and the data set, and the shape and topology of the model. Once generated, a GDM can be used for visualization, shape recognition, geometric measurements, or subjected to a series of geometric operations. This technique is of special importance because of the advent of nondestructive sensing equipment (CT, MRI) that generates point samples of true three-dimensional objects.

CR Categories: I.3.3 [**Computer Graphics**]: Picture/Image Generation — *display algorithms* — *viewing algorithms*; I.3.5 [**Computer Graphics**]: — Computational Geometry and Object Modelling — *curve, surface, solid, and object representations*;

Additional Keywords and Phrases: Deformable Models, Geometric Modelling, Volume Visualization, Volume Modelling, Constraint Minimization

1 INTRODUCTION

The development of remote sensing and scanning technology permits the nondestructive examination of an object's internal structure. This ability has proven to be essential in numerous engineering and medical fields. It allows for the inspection of mechanical parts without destroying the product and the examination of internal organs without operating on the patient. The technology generates a discrete three-dimensional scalar field where each value is a measure of some physical property, for example density. Since this data is produced via point sampling, it inherits all the properties and problems of sampled data. These include sampling artifacts, spatial aliasing, and noise. The scalar field can be composed of a series of two-dimensional slices, that when stacked, form the three-dimensional volume. Traditionally, each 2D slice was viewed separately, requiring a specialist to deduce the true 3D structure represented in the data. There are two alternative methods of displaying and analyzing the raw scalar field. One treats the volume data in its original form, as in both morphology and volume rendering [1, 2, 3]. The other transforms the data into something that is more readily displayed, such as a surface [4, 5, 6, 7, 8].

However, a more powerful approach generates geometric models of the scanned objects using the volume data as a measure of the object configuration [9, 10, 11]. This differs from the second method in that it approximates rather than interpolates the data. The major motivation behind this approach is that a geometric model provides the greatest number of options for analyzing and visualizing the original object. Once created, such a model can be used for inspection, visualization, or subjected to a series of geometric measurements and operations. Generating a model has the effect of removing the "noise" from scanned data making object identification easier. Defects in an object will result in a model that is malformed, thus emphasizing the defect. A geometric measurement, such as volume, may be easily performed on a geometric model. CSG operations may be applied to both the model and other geometry in order to convey further information about the extents and interrelationships of the structures.

In this paper, we present a methodology for extracting a topologically closed geometric model from a volume data set. The technique, called Geometrically Deformed Models (GDM's), starts with a simple model that is already topologically closed, and deforms the model based on a set of constraints, so that the model grows (or shrinks) to fit the object within the volume while maintaining its closed and locally simple (non-self-intersecting) nature. The initial model is a non-self-intersecting polyhedron that is either embedded in the object or surrounds the object in the volume data representation. A function is associated with every vertex of the polyhedron that associates costs with local deformation, adherence to properties of simple polyhedra, and the relationship between noise and feature. By minimizing these constraints, one achieves an effect similar to inflating a balloon within a container or collapsing a piece of shrink wrap around an object.

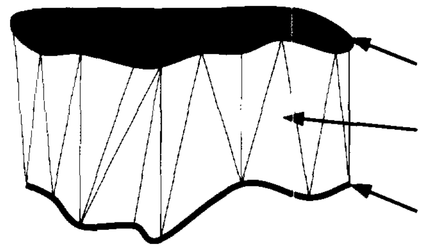

Figure 1 A surface can be created by stitching together the 2D contours extracted from adjacent slices in a volume data set.

2 PREVIOUS WORK

Previous techniques for extracting three-dimensional geometries from volume data fall into three categories: contour stitching, surface construction, and deformable models. Fuchs, Kedem and Uselton developed a means of stitching a series of two-dimensional contours together by fitting a triangular strip between adjacent contours [6] (Figure 1). Lin, Chen and Chen connected two-dimensional contours using spline theory, quadratic variation-based surface interpolation, and dynamic elastic contour interpolation [5]. In either method, every contour that composes the object needs to be identified for every slice in the data set. Plus the complexity of the algorithms increases when adjacent slices have a different number of contours, referred to as the branching problem. GDM's have neither of these complications because they treat the data set as a complete volume as opposed to a series of slices. This allows the branching problem to be handled implictly by treating concavities in the direction normal to the slice plane (branches) in the same manner as any other concavity in the data set. The problem of identifying all the contours that compose the object is removed because a GDM naturally probes through the entire object following all of its branches.

Herman, Frieder, and Artzy tracked the surface of an object using the voxel data as a graph [4]. A voxel containing the surface of the object is identified and the algorithm traverses the neighboring voxels generating a topology that is guaranteed to be closed. This work does not suffer from the branching problem discussed with contour stitching because the algorithm follows the surface as it travels through the volume. Lorensen and Cline developed marching cubes [7] to simply extract a list of polygons from volume data with no connectivity information. In their algorithm, a cube is bounded by eight pixels located on two adjacent slices. Each vertex is coded as either inside or outside the object relative to the surface defining threshold. Based on the configuration of vertices that lie inside and outside the object; the cube is triangulated, each triangle indicating a portion of the surface. Marching cubes was extended into dividing cubes [8] by Cline et al. Dividing cubes resamples the voxels to the desired display resolution in order to generate points with normals instead of triangles. Marching and Dividing cubes do not suffer from the branching problem because they extract the entire surface located in the volume. The problem with this group of surface construction algorithms is that they are restricted to generating models where each element in the model is at most the size of a voxel, hence they cannot approximate the data. Also, these algorithms are not applicable to the task of generating a closed model of an object that is not necessarily closed, for instance the interior of an opened wire bottle. In this case, they will either extract a model with little resemblance to the desired object, or they will extract multiple objects. GDM's on the other hand can produce models of varying resolution. This provides a data reduction and aides in the GDM's ability to "bridge" over the holes in the boundary of an object.

Kass, Witkin, and Terzopolous have developed snakes [9] which model the contours of an image by minimizing the energy associated with a spline. The energy of a snake configuration is based upon the image and its first and second derivative, the curvature of the edge components in the image, and the first and second derivative of the spline. Terzopoulus, Witkin, and Kass extended the concept of snakes into symmetry-seeking models [10], that derive a three-dimensional shape from a two-dimensional image by modelling an axisymmetric elastic skin spread over a flexible spine. These approaches provide a compact representation of an object or feature and should be tolerant of noise, but they are currently limited to 2D data and at most 2.5D models (for symmetry-seking models). Snakes and symmetry–seeking models can take advantage of *a priori* information about the configuration and orientation of the object being modelled, but they do not provide a multi-resolution approach to probing the data. Finally, since the internal energy of the spline is a global operation, it would appear to be difficult to parallelize the algorithm. GDM's are very similar to snakes except they can probe volume data, thus generating 3D models; they can probe the data with a low resolution model then substitute a higher resolution model; and a GDM is controlled through local geometric operations on a discrete model, hence it is easily parallelized.

In another deformable matching technique, Bajcsy and Kovačič used a multiresolution approach to elastically deform a known brain atlas to match a scanned brain[12]. This approach decreases the resolution of the data set then deforms the brain atlas so the outer edge and ventricles matches the data. The resolution is then increased and the deformation process repeated. This approach motivated GDM's to operate on the slice data as a true volume and to vary the resolution of the model during its deformation. The drawback to deforming an atlas to fit an object is that an atlas is required for every object to be modelled.

The proposed solution of deforming a model to fit an object is based upon Witkin et al.'s [13] work on energy constraints and Breen's work on goal-oriented motion for computer animation [14], and reflects a simpler approach to the problems presented by Kass et al. [9], Terzopoulos et al. [15, 10], and Bajcsy et al. [12]. These other approaches model the elastic nature of a curve or surface to control the model's deformation. Such models are based on the differential equations of elastic materials. GDM's, on the other hand, do not try to model an elastic substance; in contrast they model a simpler discrete deforming structure influenced by local geometric constraints.

3 CONSTRAINT MODELLING AND MINIMIZATION

GDM's may be envisioned as a semi-permeable balloon located inside the scanned object. The balloon expands until its surface reaches the boundary of the scanned object. The balloon is actually a collection of discrete polygons. The volume data is sampled only at the vertices of these polygons. Permeability is achieved because elements of noise and insignificant features pass through the faces of the polyhedron, thus allowing the vertices of the polygonal mesh to miss or work around these elements. By placing a cost function at each vertex in the mesh, the relevant characteristics of the balloon can be modelled. By minimizing these cost functions, the balloon is expanded while maintaining its topology.

3.1 CONSTRAINT MODELLING

GDM's are created using a top down algorithm specification. First the behavior and characteristics of the model are defined. Then constraints are selected to achieve the desired behavior. Finally, functions are developed that model the constraints. Three orthogonal behaviors must be specified. The first is a mechanism for generating gross deformations. In the balloon analogy, this mechanism expands the balloon. Second, a mechanism is needed that will interact with the data set and identify voxels possibly containing the object boundary. This function restricts the balloon from expanding through the boundary of the object being modelled. Finally, since all operations are performed locally and the boundary of the object may be incomplete, the third function maintains the local topology of the model. This keeps the balloon from intersecting itself locally.

Each of these behaviors can be modelled by a term in a local cost function associated with each vertex in the model (cost functions are also referred to as potential functions). As each cost function is minimized, the model deforms while searching for the boundary of the object and maintaining its topology ([11] provides cost functions suitable for a 2D GDM). At each time step, every model point has the opportunity to move to a position of lower potential. Each constraint function, therefore, must produce a lower cost as the model moves towards satisfying that constraint. The cost for the current position of the vertex is a linear combination of the individual cost functions, which allows for one term to dominate the deformation. Each cost term must therefore have the ability to assert itself and dominate the overall cost function when its constraint is being violated, as well as seem insignificant when its constraint is being satisfied.

The cost function associated with the current location of a model point is the weighted sum

$$C_i(x,y,z) = a_0 D(x,y,z) + a_1 I(x,y,z) + a_2 T_i \qquad (1)$$

where:

$C_i(x,y,z)$ is the cost associated with this position of the current model point,
$D(x,y,z)$ is the potential field that drives the model point towards the boundary,
$I(x,y,z)$ is the image term that identifies feature events,
T_i is a measure of how the local configuration of polygonal faces satisfies the topology of the model,
a_0, a_1, a_2 are the individual weighting coefficients that allow the magnitudes of the various parameters to be scaled,
$C_i(x,y,z), D(x,y,z), I(x,y,z), T_i, a_0, a_1, a_2 \geq 0$.

3.1.1 DEFORMATION POTENTIAL — $D(x,y,z)$

The deformation potential defines a scalar field where each position in space is assigned a value based on a frame of reference. In this case, a frame of reference can be any configuration of image or model parameters. The frame of reference may be a point inside the feature to be modelled, or it may be a set of vertex points in their previous configuration. The deformation potential must monotonically decrease (or increase) from the frame of reference and will repel (or attract) the current model point away from (or towards) its frame of reference.

Normal Tracking: Simple concave models can be created using a localized deformation potential. Each vertex is attracted to a point located in the direction of the polyhedron surface normal at that vertex. During each deformation cycle, each vertex moves in the general direction of the local surface normal. As a concavity is encountered, the topology and image event constraints influence the deformation. The surface normal rotates around the concavity, allowing the model to continue its deformation inside of the region that was previously hidden from view (Figure 2).

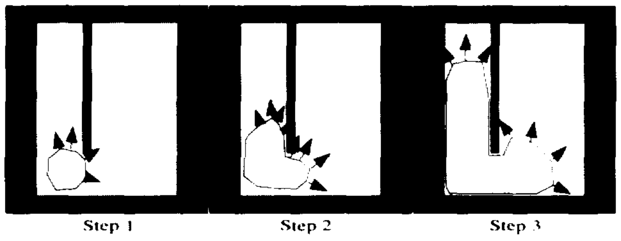

Figure 2 Surface normals directing a model to bend around a concavity.

3.1.2 IMAGE EVENTS — $I(x,y,z)$

This class of constraint counterbalances the deformation potential. It is used to restrict, direct, and oppose the general progression of the deformation. Basically this constraint informs the vertex that it may be in contact with a voxel containing the original object (feature voxel). This constraint need not be able to distinguish noise and object since at the resolution of 1 voxel the two are indistinguishable, but must be able to identify the transition from a region of the data set that could be a feature to a region of the data set that is definitely not a feature. The important aspect of this constraint is that it introduces a local minimum at boundary events. Operations that identify boundary events include digital gradients [16], the Canny operator [17], and morphological operations [2]. Although any of these operators would suffice, GDM's can operate with a much simpler event detector.

A shifted threshold operator

$$I(x,y,z) = \begin{cases} 0 & Image(x,y,z) < T \\ Image(x,y,z) - T & Image(x,y,z) \geq T \end{cases} \qquad (2)$$

where:

$Image(x,y,z)$ is the grey-level intensity of the voxel at (x,y,z),
T is a threshold value that identifies the object;

is shown in Figure 3. Recall that the image event detector identifies the transitions from regions that are definitely not-object to regions of the image that could be object. The threshold, T, categorizes each voxel as either not-object or possibly object. Here a voxel that is not part of the object returns a value of zero, while a voxel that is part of the object returns the amount it exceeds the object identifying threshold. The image event operator in conjunction with the minimization process and the trilinear interpolation of voxel values allows for the true object edge to be located. When a model point steps over the edge of an object, $I(x,y,z)$ returns a value that should increase the overall cost of the system. The minimization process is forced, therefore, to either move the vertex by a smaller amount or to not move the vertex at all. Hence the vertex will approach the edge without crossing over it (unless its neighbors pull it over the edge).

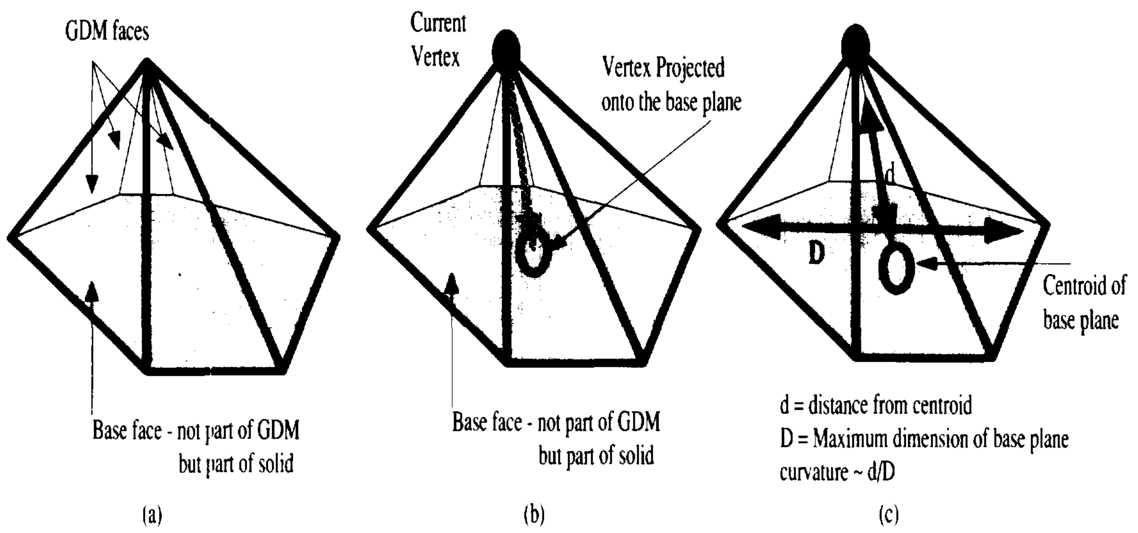
GDM faces
Base face - not part of GDM but part of solid
(a)
Current Vertex
Vertex Projected onto the base plane
Base face - not part of GDM but part of solid
(b)
d
D
Centroid of base plane
d = distance from centroid
D = Maximum dimension of base plane
curvature ~ d/D
(c)

Figure 4 (a) A local solid model can be cut out of a GDM, by assuming that adjacent neighbors are connected in a fashion that will close the local solid model. (b) A vertex is contained by its base if the projection of the vertex onto the base is a point interior to the base. (c) The ratio of the distance between a model point and the centroid of its neighbors to the maximum dimension of the base plane gives an estimate of the curvature.

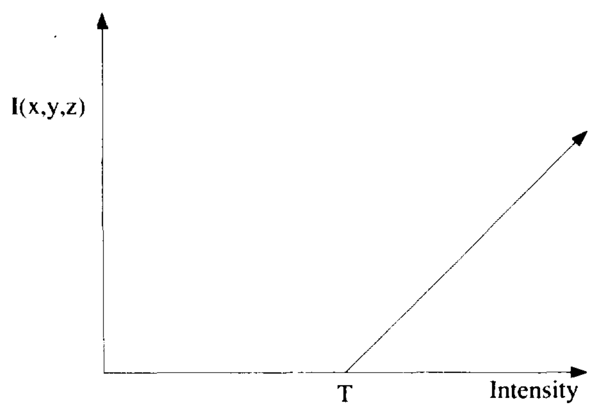

Figure 3 The image event detector used for GDM's is a simple shifted threshold $(X - T)^+$. Here the cost function returns zero if the voxel value is below the edge threshold, otherwise it returns a value that indicates how much the voxel value exceeds the edge threshold.

3.1.3 MAINTAINING TOPOLOGY — T_i

The final constraint maintains the topological integrity of the model and controls the spatial frequency of the model. The first two constraints cause the model to deform until all the vertices reach the boundary of the object. These two constraints are not sufficient to extract a geometric model from real data. For instance, the boundary of the object may be incomplete, consisting of gaps and holes. This may allow the vertices of the model to leak out of the object and travel without restriction towards the boundary of the data set. Alternatively, the data set may have elements of noise that could cause the image event detector to incorrectly categorize the noise as the boundary of the object. These two situations, coupled with the expansion or lack of expansion of the remainder of the model, may result in a geometric model that has little resemblance to the original object.

It is therefore necessary to have the geometry of the model influence a portion of the deformation. Since a topologically simple geometric model is desired, a constraint is added to the system that will maintain the locally simple nature of the initial model. The topological constraint is also referred to as a maintenance constraint. This term also controls the spatial frequency of the model by keeping vertices from leaking out of the holes in the boundary of object, as well as preventing vertices from being caught on an element of noise. These two behaviors are essentially duals. In the case of a vertex leaking out of a hole, there is a single vertex that is continuing its deformation while its neighbors have reached the boundary of the object. In the case of noise confrontation, a single vertex believes that it has found the boundary of the object while its neighbors continue their deformation. In either case, the faces associated with this vertex will become much larger than the faces in the immediate vicinity.

It is desirable for a vertex not to stray far from its neighbors or have its neighbors stray far from it. It is also desirable that the topology be maintained. Therefore, a vertex should be contained by its neighbors. A solid can be formed by the current model point and its neighbors. Imagine that the current model point and its neighbors are cut out of the GDM. By connecting the adjacent neighbors, a solid is created. (Figure 4(a)). Any face of this solid that contains the current model point is also a face of the GDM. Any face strictly composed of the current model point's neighbors is not a face in the GDM, but will be referred to as the "base" in the new solid. For a planar "base", the current model point is contained by its neighbors if, when it is projected onto the base plane, the projected point is in the interior of the polygon defined by the base (Figure 4(b)). If the "base" is not planar, this concept can still be applied to a polygon that is a planar approximation to the base.

Curvature Estimation: Keeping a model point contained by its neighbors while keeping the model point from straying too far from its neighbors suggests that the local curvature of the model should be constrained. The ratio of the distance from the current model point to the centroid of its neighbors and the maximum distance between the neighbors of the current model point gives an indication of the curvature (Figure 4(c)). This ratio defines the topological constraint

$$T_i = \frac{\left\| (x, y, z) - \frac{1}{n} \sum_{1}^{n} (x_j, y_j, z_j) \right\|}{\max_{j,k} \left(\left\| (x_j, y_j, z_j) - (x_k, y_k, z_k) \right\| \right)} \qquad (3)$$

where:

(x, y, z) is the current model point,
n is the number of neighbors to the current model point,
$(x_j, y_j, z_j), (x_k, y_k, z_k)$ are the neighbors of the current model point, $1 \leq j, k \leq n$.

This function directs the vertex towards the centroid of the base, which in turn, attempts to make all the faces incident to the current model point coplanar. Since all the vertices are simultaneously trying to move onto the plane of their neighbors, the entire model defaults to being spherical (in the absence of the other constraints). Dividing the distance to the centroid by the maximum base point separation maintains scale invariance.

3.2 OPTIMIZATION METHOD

The cost function minimization technique utilizes an adaptive algorithm to move a vertex of the model in the direction of steepest descent along the cost surface. This direction is opposite to the gradient of the cost function C_i, and is estimated by numerically approximating the differentials $\left(\frac{\partial C_i}{\partial x}, \frac{\partial C_i}{\partial y}, \frac{\partial C_i}{\partial z}\right)$. The amount that a point moves is adjusted based upon the current configuration of the cost space. The stepsize can be reduced three times if movement by the current stepsize results in an increase in the cost function. If a step cannot be completed that will reduce the cost of the vertex point, then the vertex point is not moved. For the purposes of geometrically deformed models, the stepsizes are maintained in the range of [1/4, 1] voxels. This allows for rapid changes in the dimensions of the model when it is in a void, as well as fine adjustments in the model when it encounters an element of noise or the boundary of the cavity.

This technique will find local minima. No global minimization techniques such as simulated annealing [18] are performed; so global minima are not always found; however, a gradient descent has proven sufficient for the data sets tested. The algorithm actually exploits the fact that only local minima are found, by defining its cost functions to introduce local minima whenever a critical point is crossed. A critical point occurs whenever a maintenance constraint (T_i) is violated or when a possible feature voxel is encountered.

4 3D MODELS

The initial model chosen for 3D GDM's is an icosahedron. An icosahedron is a 20 sided approximation to a sphere. The methodology does not require an icosahedron for its initial model; it was simply chosen for the property that when resampled, forming a geodesic, the connectivity remains relatively uniform. Triangular faces ensure that the faces of the model are planar and allow the model the greatest degree of flexibility when fitting the scanned object. Each vertex in an icosahedron is connected to five other vertices. If the entire icosahedron is resampled, then each new vertex is six connected while the original vertices remain five connected. All vertices added through subsequent global resamplings of the geodesic result in new vertices that are six connected while maintaining the 12 original vertices as five-connected. Figure 5 illustrates how the triangular faces in an icosahedron can be resampled to form the faces of a geodesic [19]. Bisection of each edge produces three vertices that are connected to form four faces from the original face.

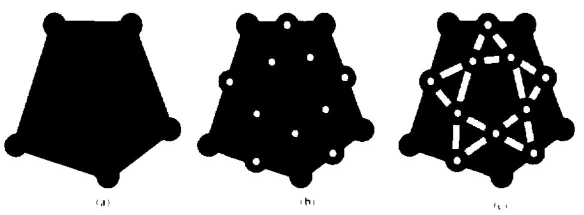

Figure 5 An icosahedron's triangular face is divided into four faces by connecting the midpoints of each edge. The connectivity of the new vertices is six while the connectivity of the original vertices remains five.

A global resampling of a GDM follows the same steps as a global resampling of a geodesic. It is irrelevant that the faces of the GDM may not be as regular as the faces of a geodesic. Any triangular face can be divided into four faces by connecting the midpoints of the edges. A global resampling of a GDM refines the entire model on command. This allows a low resolution model to probe the data initially, while a higher resolution GDM can be substituted in order to capture finer detail. Using an initial low resolution GDM greatly reduces the computation time in extracting models. Note that the number of vertices increases with the number of edges (each edge is subdivided to form a new vertex). The number of vertices in the globally resampled model is roughly four times the number of vertices in the original model.

Although a global resampling of a GDM refines the model, the increase in the number of model points limits its appeal. After each refinement, the amount of work to deform the model increases by a factor of four. Thus a global resampling must be used judiciously. Alternatively the resampling may be localized, increasing the complexity of the model only in those regions where it is necessary.

A local resampling can be performed in two operations. The first operation identifies the regions of the model that need to be resampled. The second step subdivides the associated faces while maintaining the topological database and keeping all faces triangular. The simplest way to identify the regions of the model that need to be resampled is based upon the desired level of detail. The level of detail in the extracted model is essentially the number of voxels approximated by a single face. Thus in order to maintain the level of detail, subdivide the faces that exceed the threshold set by the level of detail. Identifying these faces can be accomplished by a simple area calculation. Miller [20] discusses the details of face resampling.

5 CREATING GDM'S

Before a GDM can probe a data set, several parameters and facets of the algorithm must be specified: the object (Object) and not-object ($\overline{\mathrm{Object}}$) classification, the deformation mode (grow, shrink), and the GDM parameters (a_0, a_1, a_2, T_i). Fortunately this process benefits from the high degree of duality inherent to a GDM. Figure 6 and Table 1 summarize the duality between both

Duals			
Mode	Image Event	Mode	Image Event
shrink	$\overline{\text{Object}} \rightarrow$ Object	grow	Object $\rightarrow \overline{\text{Object}}$
shrink	Object $\rightarrow \overline{\text{Object}}$	grow	$\overline{\text{Object}} \rightarrow$ Object

Table 1 Duality relationships

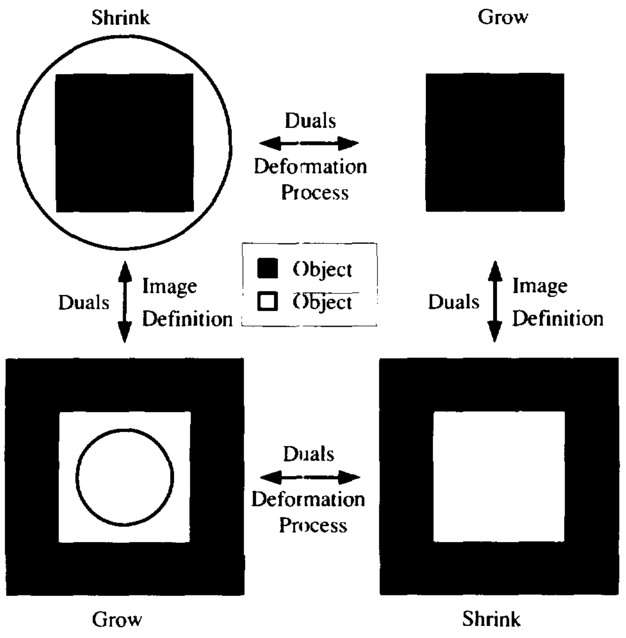

Figure 6 Duality relationships - the rows indicate duals formed by changing the deformation mode while the columns indicate duals formed by changing the object definition.

the deformation mode and object classification. Growing a GDM with one classification of Object and $\overline{\text{Object}}$ results in the same model as shrinking a GDM with the roles of Object and $\overline{\text{Object}}$ reversed. The two models differ only in deformation time.

Due to the resampling process embedded in the GDM topological database, it is possible for a vertex to be added to the model on the wrong side of the Object $\rightarrow \overline{\text{Object}}$ boundary. Recall that the minimization technique does not allow a vertex to move to a position of higher potential. Therefore all of the model points will approach the boundary of the object from the same side. A model point that tries to move to the opposite side of the boundary will have its image event detector active and thus will have a higher potential. But the resampling algorithm may place new vertices on the opposite side of this boundary. Therefore, in order to move these model points to the other side of the boundary and hence increase the accuracy and quality of the model, the surface normal used in the deformation potential is defined to point in the opposite direction. This locally flips the deformation mode (i.e. from growing to shrinking). The effect is that a model point will migrate towards the true boundary of the object regardless of whether the model point is located in Object or $\overline{\text{Object}}$.

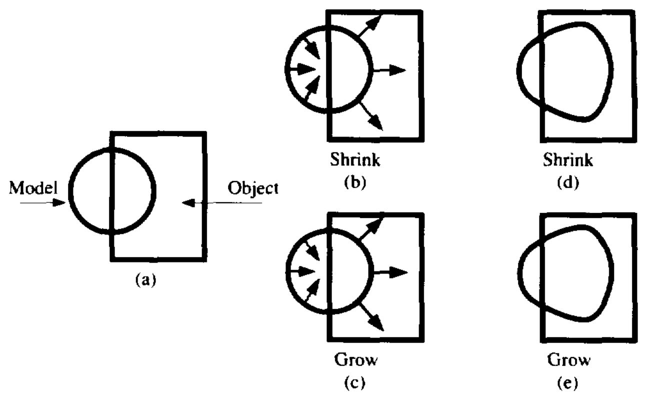

Figure 7 A GDM intersects the boundary of the Object $\rightarrow \overline{\text{Object}}$. The two deformation modes are equivalent in this case because the normals locally flip (dotted arrows indicate a change in the deformation sense).

A beneficial side effect of a GDM locally reversing the sense of deformation is related to the placement of the initial model. As long as the initial model intersects the Object $\rightarrow \overline{\text{Object}}$ boundary (i.e. some of the model points are inside object, the remainder are inside not-object), the model tends to seek out the true boundary of the object regardless of the deformation mode. This is a direct consequence of the duality and locality of deformation modes. Note that the model extracted from growing will differ from the model extracted by shrinking only by the dimensions of the boundary. Figure 7(a) shows a model intersecting an Object $\rightarrow \overline{\text{Object}}$ boundary. The deformation direction is shown for shrinking (Figure 7(b)) and growing (Figure 7(c)) a model. The solid deformation arrows indicate the deformation direction agrees with the primary deformation mode while the dotted deformation arrows indicate that the deformation mode has locally flipped sense. Figures 7(d) and (e) show the models at a later time step. Note that the two models approach the same boundary but they approach the boundary from different sides.

The duality and locality of deformation modes works in favor of a GDM if the GDM intersects both object and not-object and if the GDM does not intersect multiple objects. If a GDM does not intersect both regions of the data, then the deformation mode must agree with the placement of the initial model (in object or not-object). If the deformation mode is indeed correct, the GDM process extracts a geometric model of the boundary of the object. Otherwise, the GDM either collapses upon itself or extends out to infinity. Note that from the duality and locality of deformation modes, either case is feasible with either deformation mode.

Our experiments show that the GDM parameter values (a_0, a_1, a_2) are relatively data independent. This is due to a_0, a_1, a_2 governing the GDM process, not the sampled data. Table 2 summarizes the suggested parameter values. Note that the object classifying threshold, T, is data dependent.

6 RESULTS

The 3D GDM figures all present a GDM expanding within an object. The figures contain four frames. The first frame shows the initial model (upper-left). The second frame (upper-right) shows the model after several iterations. The third frame (lower-left) presents the GDM after an initial convergence to the shape of the

3D Parameters		
Deformation Gain	a_0	1
Image Event Gain	a_1	1
Topological Gain	a_2	5
Resampling Threshold	face area	10

Table 2 3D Suggested Parameters

object. The final frame (lower-right) presents the final model. The final model was created by performing a global resampling on the GDM after the initial convergence to the shape of the object. The GDM was then allowed to converge to the shape of the object a second time.

6.1 CUBE

The first 3D example is an artificially generated cube with one of its corners removed (Figure 8). The voxels inside the cube were assigned one intensity while the voxels outside the cube were assigned a different intensity (creating a 64x64x64 volume). The initial model consisted of 20 triangles and the resampling algorithm added roughly 1000 triangles. A global resampling was then performed to increase the model quality. The final model contains 4080 triangles. The marching cubes model consists of 11528 triangles. Therefore, a substantial data reduction was achieved (1000 triangles vs. 11528 triangles) before the final global resampling was applied (1000 face model is in the lower-left of Figure 8). A moderate data reduction is achieved if a final global resampling is applied to the model after the initial convergence (4080 triangles vs 11528 triangles). The lower-right frame of Figure 8 shows the 4080 triangle GDM. The entire deformation required 50 iterations (approximately 15 minutes on an HP9000 835).

6.2 TURBINE BLADE

The next 3D GDM example is a cooling chamber of a turbine blade. The source of the data is 96 industrial CT slices (256 by 256). The data is very clean, and the resulting GDM is shown in Figure 9. This model consists of 6560 faces and required 100 deformation cycles (approximately 30 minutes on an HP9000 835). The marching cubes model for this object is composed of 19000 triangles.

6.3 TOOTH

The final 3D GDM is a model of the nerve in a tooth (Figure 10). The tooth was scanned using industrial CT (161 slices at 256 by 256 pixels). The initial model was placed in one of the roots of the nerve. This GDM illustrates that highly concave models can be created. The final model has remarkable detail and is composed of only 7392 faces while the marching cubes model for this object is composed of 20944 triangles.. This model required 200 deformation cycles (approximately 1.25 hours on an HP9000 835).

6.4 VERTEX GENERATION

Figure 11 shows a GDM where each vertex is assigned a scalar value based upon the generation (deformation cycle) of its creation. The red vertices were created early in the deformation,

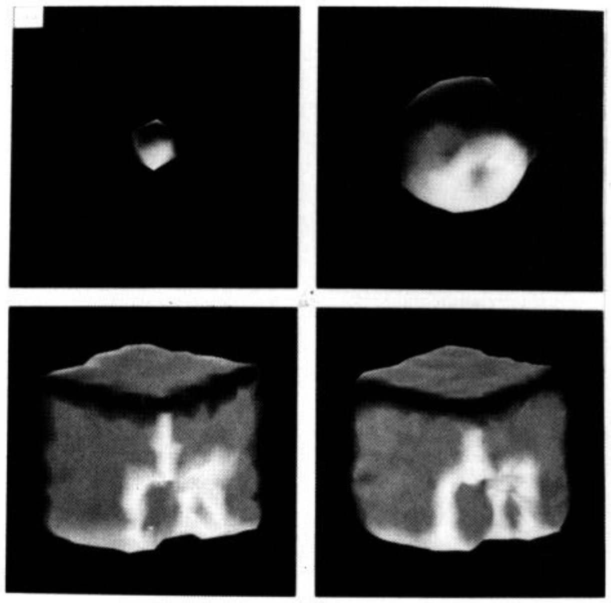

Figure 8 A 3D GDM expanding within a cube with a corner missing. The final model is composed of 4080 faces. Note that the corners and edges of the cube are rounded. The topology constraint lowered the spatial bandwidth of the model below the spatial bandwidth of the original data.

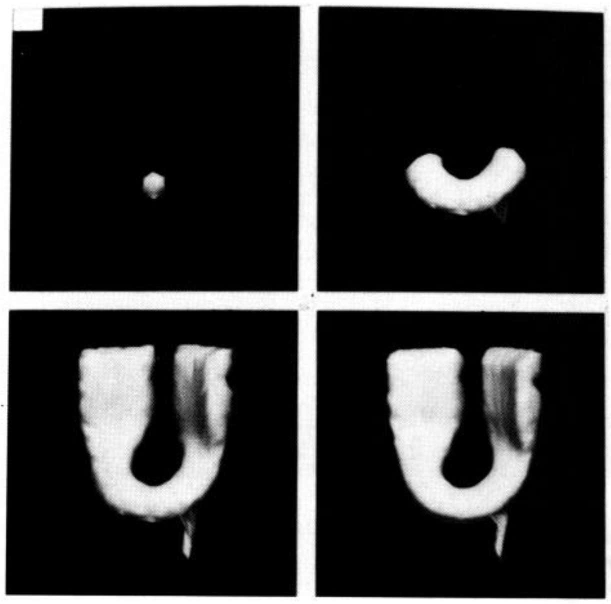

Figure 9 A 3D GDM of the cooling chamber in a turbine blade. The normal tracking deformation potential allows for the concave model to be extracted.

hence they are the oldest vertices. The yellow and green vertices were added later in the deformation process, hence they are progressively younger. Finally the blue vertices were added to the model as it approached its final orientation, hence they are the youngest of all. This illustrates how the vertices are added only in

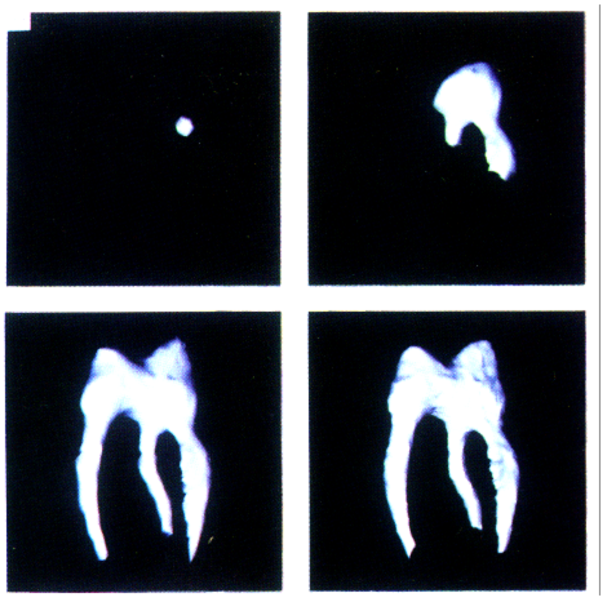

Figure 10 The nerve in a too:h is modelled through a highly concave GDM.

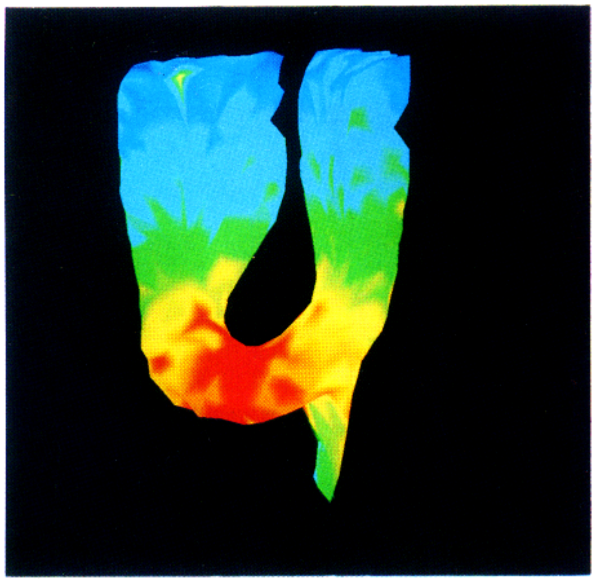

Figure 11 The GDM has its vertices colored based upon the generation which the vertex was created. This illustrates that model points are only added in the regions of the model still expanding.

the regions of the model that are still deforming. This emphasizes the importance of the proper selection of the initial location for the GDM. The GDM should be placed such that the model can deform across the maximum number of fronts. This figure also illustrates the locality of the resampling algorithm. Since vertices are only added in the region still deforming, the colored model has a series of "bands".

7 DISCUSSION

A GDM's performance can be measured by the quality of the final model and the amount of time required to generate the model. Several factors influence both of these measures. These include *a priori* information, the weights associated with the GDM constraints, global crossings, locality of measures, and precision.

A priori information about an object's geometry can be presented to a GDM in the form of its initial configuration. This information can influence the initial size and shape of the model. Our experiments show that the initial shape of the model does not affect the quality of the final GDM model. A simple convex model such as an icosahedron can be deformed to fit a topologically simple object with the same level of success as a more complicated initial model. The initial size of the model also has little effect on the quality of the final GDM. Both the size and shape of the model, however, may influence the time required to extract the final model. The closer the initial model is to the size and shape of the final model, the less time that is required to extract the GDM since the vertices of the GDM have less distance to travel in order to find the boundary of the object.

The parameters presented in Table 2 are for the most part data independent; however, a GDM's performance is greatly affected by their values. The weights were devised to compensate for any scale invariance in the constraint functions and to scale each constraint function to the same order of magnitude. If these weights are altered, one constraint function may dominate the deformation and result in a malformed GDM.

Since GDM's are currently ruled by local constraints, there is a possibility that the model will self-intersect. The GDM has no means of knowing that the model has self-intersected globally and will continue to deform. This can result in a model that continuously intersects itself. The probability for self-intersection can be minimized by presenting GDM's only with topologically simple objects to model. Unfortunately, even using topologically simple objects, self-intersection is possible. There exist topologically simple data sets that cause a GDM to fold into itself. Objects with components of high spatial frequency can cause such an occurrence.

GDM's are governed by local geometric operations. The size of the neighborhood used to calculate the geometric measures influences the GDM process. If the neighborhood is too large, the measure is not really local. If it is too small, the measure may not be accurate. The surface normal defines the deformation potential's influence on the deformation direction. Ideally, the surface normal can be calculated as the shared normal of all the incoming faces to that vertex; however, through experimentation it has been found that the GDM process is more stable when a larger neighborhood of faces is used to calculate the surface normal (a GDM is stable if global crossings do not occur). For instance, using the incoming faces and the faces adjacent to them provides the GDM with added stability; however, if this neighborhood is made too large, then the surface normal in not accurate and the added stability of a large neighborhood is lost.

The presentation thus far has hinted that a lack of precision may be the cause of a number of GDM idiosyncracies. The sensitivity of weights, global crossings in simple objects, and the surface normal stability are a function of precision. After all, the geometric measures are composed of a series of floating point operations. Each operation reduces the numerical precision. This can easily explain the global crossings in a simple object. An edge of a concavity can be smaller than the faces in the GDM. If new vertices are added in this region, imprecision may place them on different sides of the edge or perhaps within the other faces of the model. The surface normal can also be affected by a lack of precision. If the faces in the model are rather small, small errors in arithmetic can produce a large errors in the surface normal. Therefore the larger neighborhood is essentially performing a low pass filter on the surface normals in order to compensate for precision problems. Precision also limits the choice of initial models. For example, a 640 point geodesic approximation to a sphere of radius 1 created a very nonuniform model until the size of the model increased to a sphere of reasonable radius.

8 FUTURE WORK

The framework for GDM's is complete; however, there are a few ideas and concepts that could not be completely investigated. These concepts are secondary in nature, in that they are not essential to the theory or operation of a GDM, but they may improve performance or create additional applications. The current implementation of GDM's was established so that various ideas and geometrical relationships could be tested with relative ease. As such, the implementation is far from optimal. An alternative data structure that stores semi-permanent relations (information that is constant through a deformation iteration) could reduce the deformation time by an order of magnitude.

Future research efforts should concentrate on two basic areas: model quality and alternative data sets. Model quality may be improved by an alternative resampling algorithm and by preventing global crossings (model self-intersection). GDM's should also be extended to automatically handle data sets with multiple objects and to handle higher dimensional spaces, for instance time varying volumes.

9 CONCLUSION

A GDM extracts a closed topologically simple (non-self-intersecting) geometric model of an object located in a discrete or continuous data set. An initially closed model is embedded in the data set and deformed to fit the object through the minimization of a set of constraints. These constraints are local operations that quantify the deformation, the properties of simple polyhedra, and the relationship between object and not-object. The final model remains closed, because the initial model is closed and the constraints used to deform the model maintain the closed and locally simple nature.

The major benefit of GDM's is that they aggregate sampled data by placing geometrical relationships on the model, as opposed to interpreting and analyzing the sampled data directly. This allows the model to interact favorably with artifacts of noise that either remove portions of the boundary or insert false boundaries. GDM's are highly adaptive allowing for a generic initial convex model to be transformed into a highly concave object. Alternative initial models can be used that reduce the deformation time. GDM's explicitly handle the branching problem, multiple contours in one slice of a volume data set mapping into one contour in an adjacent slice, by treating a collection of 2D slices as a true 3D data set. Therefore, concavities in the direction normal to the slice plane are treated with the same mechanism as any other concavity in the data set. GDM's can use a local resampling algorithm to minimize the amount of work required to deform a model and to increase the model's quality. The level of detail can be set by the user, so quick estimates of an object can be generated and later refined for higher quality. GDM's provide a considerable data reduction in comparison to traditional techniques.

A GDM can be used for visualization, object recognition, geometric measurements, or subjected to a series of geometric operations. Applications abound in such fields as medicine, where GDM's could be used to generate models of internal organs; engineering, where GDM's could be used to model scanned mechanical parts or their faults; and science, where GDM's could be used to model higher dimensional spaces not accessible using traditional algorithms.

The computation to extract a model is proportional to the size and complexity of the object, not the size of the original data. GDM's are controlled through local geometric operations rather than the physical modelling of an elastic or plastic structure, hence the computations are much simpler. Finally, the dual GDM problem may be simpler to solve, resulting in a model of the same quality as the primal problem but with a much shorter deformation time.

10 ACKNOWLEDGMENTS

We would like to thank Prof. W. Randolph Franklin for his suggestions on the geodesic data structure and Dr. Donald House for his comments on this manuscript. This research was partially funded by General Electric Corporate Research and Development through the Rensselaer Design Research Center's Industrial Associates Program. The research was conducted at both GE CR&D and the RDRC.

REFERENCES

[1] J. Serra. *Image Analysis and Mathematical Morphology Volume 1*. Academic Press, 1982.

[2] S.R. Sternberg. Grayscale morphology. *Computer Vision, Graphics, and Image Processing*, (35):333–355, 1986.

[3] R.A Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. *Computer Graphics*, 22(4):65–74, 1988.

[4] E. Artzy, G. Frieder, and G. Herman. The theory, design, implementation, and evaluation of a three-dimensional surface detection algorithm. *Computer Graphics and Image Processing*, 15:1–24, 1980.

[5] W.C. Lin, S.Y. Chen, and C.T. Chen. A new surface interpolation technique for reconstructing 3d objects from serial cross-sections. *Computer Vision, Graphics, and Image Processing*, 48:124–143, 1989.

[6] H. Fuchs, Kedem Z.M., and Uselton S.P. Optimal surface reconstruction from planar contours. *Comm. ACM*, 20(10):693–702, 1977.

[7] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[8] H.E. Cline, W.E. Lorensen, S. Ludke, Crawford C.R., and B.C. Teeter. Two algorithms for the three-dimensional reconstruction of tomograms. *Medical Physics*, 15(3):320–327, 1988.

[9] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, pages 321–331, 1988.

[10] D. Terzopoulus, A. Witkin, and M. Kass. Symmetry-seeking models and 3d object reconstruction. *International Journal of Computer Vision*, 1(3):211–221, October 1987.

[11] J.V. Miller, D.E. Breen, and M.J. Wozny. Extracting geometric models through constraint minimization. *Visualization '90 Proceedings*, pages 74–82, 1990.

[12] R. Bajcsy and S. Kovačič. Multiresolution elastic matching. *Computer Vision, Graphics and Image Processing*, (46):1–21, 1989.

[13] A. Witkin, K. Fleischer, and A. Barr. Energy constraints on parameterized models. *Computer Graphics*, 21(4):225–232, July 1987.

[14] D.E. Breen. Choreographing goal-oriented motion using cost functions. *State–of–the–Art in Computer Animation (Computer Animation '89 Conference Proceedings)*, pages 141–151. eds N. Magnenat-Thalmann and D. Thalmann (Springer-Verlag, Tokyo, June 1989).

[15] D. Terzopoulos and K. Fleischer. Deformable models. *The Visual Computer*, 4:306–311, 1988.

[16] R.C. Gonzalez and P. Wintz. *Digital Image Processing*. Addison-Wesley, 1987.

[17] J.F. Canny. Finding edges and lines in images. Masters thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, June 1983.

[18] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[19] M.J. Wennington. *Spherical Models*. Cambridge Univerity Press.

[20] J.V. Miller. On gdm's: Geometrically deformed models for the extraction of closed shapes from volume data. Masters thesis, Rensselaer Polytechnic Institute, Troy, New York, December 1990.