A Deformable Modeling Paradigm for 3D Shape Recovery from Visual Inputs

Ye Duan
Department of Computer Science
University of Missouri at Columbia
duanye@missouri.edu

Hong Qin
Department of Computer Science
State University of New York at Stony Brook
qin@cs.sunysb.edu

Abstract

In this paper, we propose a deformable modeling paradigm for 3D shape recovery from visual inputs such as volumetric data and 3D point clouds. The new model is capable of automatically evolving its shape to capture the geometric boundary of the data and simultaneously discover its underlying topological structure. The deformation behavior of the model is governed by partial differential equations (e.g. the weighted minimal surface flow) that are derived by the principle of variational analysis. Unlike the level-set approach, our model always has an explicit representation of geometry and topology. The regularity of the model and the stability of the numerical integration process are ensured by three mesh optimization techniques and the Laplacian tangential smoothing operator. By allowing local adaptive refinement of the mesh, the model can accurately represent fine details. The results of our extensive experiments on synthetic and real data demonstrate robustness, high reconstruction accuracy and visual quality.

1. Introduction

During the past decade, PDE-driven surface evolution has become very popular in the computer vision community for shape recovery and object detection. Most of the existing work is based on the Eulerian approach, i.e., the geometry and topology of the shape is implicitly defined as the level-set solution of time-varying implicit functions over the entire 3D space [14], which can be computationally very expensive. In this paper, we propose a new PDE-based deformable model that, in contrast, takes the Lagrangian approach, i.e., the geometry and topology of the deformable surface are always explicitly represented throughout the simulation process. The elegance of our approach lies in the fact that we can use the same PDE-based model for different types of data. The only thing that is data-dependant is the control function, which describes the interaction with the data. This is an important property that will allow easy application of our methodology to other types of data, such as points, surfels, images and to incorporate other visual cues such as shading and optical flow.

Starting with [10], deformable models have achieved great success in the areas of computer vision and pattern recognition. In general, deformable models can be divided into two categories: explicit models and implicit models. Explicit models include parametric representations [17] and discrete representations [18]. Implicit models [14, 3, 8] handle topology changes, based on the modeling of propagating fronts, which are the level set of some scalar function. The desirable shape must be explicitly evaluated using marching-cube-like techniques [13] in an additional post-processing stage. The narrow band algorithm [8] can reduce the computational cost related to the higher dimension. Recently, topologically adaptive explicit models have been proposed [16, 12], reviewed in detail in [15]. The aforementioned deformable models were proposed mainly for the purpose of shape reconstruction from volumetric data and for medical image segmentation. For shape reconstruction from point clouds, existing work is mostly on static methods. They are either explicit methods [7, 1, 5], implicit methods [9] or based on radial basis functions [2, 6].

Compared with level-set based methods, our new model is simpler, more intuitive, and makes it easier to incorporate user-control during the deformation process. To ensure the regularity of the model and the stability of the numerical integration process, powerful Laplacian tangential smoothing, along with commonly used mesh optimization techniques, is employed throughout the geometric deformation and topological variation process. The new model can either grow from the inside or shrink from the outside, and it can automatically split to multiple objects whenever necessary during the deformation process. More importantly, our model supports level-of-details control through global subdivision and local/adaptive subdivision. Based on our experiments, the new model can generate a good, high-quality polygonal mesh that can capture underlying topological structure simultaneously from various datasets such as volumetric data and 3D unorganized point clouds.

2. PDE-Based Deformable Surface

The deformation behavior of our new deformable surface is governed by an evolutionary system of nonlinear initial-value partial differential equations (PDE) with the following general form:

$$\frac{\partial S(p)}{\partial t} = F(t, k, k', f \cdots)U(p, t) \qquad (1)$$

where F is speed function, t is the time parameter, k and k' are the surface curvature and its derivative at the point p, and f is the external force. $S(p, 0) = S_0(p)$ is the initial surface. U is the unit direction vector and often it represents the surface normal vector. Eq. 1 can be either directly provided by the user, or more generally, obtained as a gradient descent flow by the Euler-Lagrange equation of some underlying energy functionals based on the calculus of variations.

2.1. Model Refinement

Once an initial shape of the object is recovered, the model can be further refined several times to improve the fitting accuracy. In this paper, we have implemented two kinds of model refinement: global refinement and local/adaptive refinement. Global refinement is conducted by Loop's subdivision scheme [4].

Adaptive refinement is guided by fitting accuracy as measured by the variance of the distance from the triangle to the boundary of the object [19]. If the variance of the distance samples for a given triangle is bigger than a user defined threshold, then this triangle will be refined. The variance of a discrete set of distances is computed in the standard way: $V_T[d] = E[d^2] - E[d]^2$, where E denotes the mean of its argument. To calculate the variance of the distance samples for a given triangle, we temporarily quadrisect the triangle T into four smaller triangles and for each smaller triangle, calculate the distance at its barycentric center. At each level of adaptive refinement, all the triangles with fitting accuracy below the user-specified threshold will be quadrisected. The deformation of the model will resume only among those newly refined regions. In Fig. 3, we show different levels of refinements.

2.2. Mesh Regularity

To ensure that the numerical simulation of the deformation process proceeds smoothly, we must maintain mesh regularity so that the mesh's nodes have a good distribution, a proper node density, and a good aspect ratio of the triangles. This is achieved by the incorporation of a tangential Laplacian operator, and three mesh operations: edge split, edge collapse, and edge swap.

The tangential Laplacian operator is used to maintain good node distribution. The Laplacian operator, in its simplest form, moves repeatedly each mesh vertex by a displacement equal to a positive scale factor times the average of the neighboring vertices.

When edge lengths fall outside a predefined range, edge splitting and edge collapsing are used to keep an appropriate node density. Edge swapping is used to ensure a good aspect ratio of the triangles. This can be achieved by forcing the average valence to be as close to 6 as possible [11]. An edge is swapped if and only if the quantity $\sum_{p \in A} (valence(p) - 6)^2$ is minimized after the swapping.

2.3. Topology Modification

In order to recover a shape of arbitrary, unknown topology, the model must be able to modify its topology properly whenever necessary. In general, there are two kinds of topology operations: (1) Topology Merging, and (2) Topology Splitting.

Topology Merging We propose a novel method called "lazy merging" to handle topology merging. The basic idea is that whenever two non-neighboring vertices are too close to each other, they will be deactivated. Topology merging will happen only after the deformation of the model stops and all the vertices become non-active. There are three steps in the topology merging operation: (1) *Collision Detection*, (2) *Merging-vertices Clustering*, and (3) *Multiple-contours Stitching*.

Collision Detection: Collision detection is done hierarchically in two different levels: coarser-level and finer-level. Coarser-level collision detection is mainly for the purpose of collision exclusion. For each active vertex V, we will calculate its distance to all other non-neighboring active vertices. Vertices whose distance to the current vertex V is small enough so that no collision might happen between, will be passed to the finer-level collision detection. For each face f with three corner points (u, v, w) that is adjacent to one of the vertices being passed into the finer level of collision detection, we will calculate the distance between a number of sample points $\alpha u + \beta v + \gamma w$ of the face f with barycentric coordinates $\alpha + \beta + \gamma = 1$ and the current vertex V. If at least one of these distances is smaller than the collision threshold, the two corresponding vertices will be marked as merging vertices and will be deactivated.

Merging-Vertices Clustering: After all the merging vertices have been deactivated, we need to divide them into several connected clusters. We randomly pick any merging vertex and find all of its connected merging vertices by a breadth-first search. We continue recursively, until all the merging vertices belong to appropriate merging vertex clusters. Then for each cluster of merging vertices, all the interior vertices will be removed and the remaining vertices will be put into a linked list. This is based on the following observation: *when two regions are merging together, only the boundary regions will remain, all the interior regions will be burned out (i.e. removed).*

Multiple-Contours Stitching: After the merging vertex linked lists have been created, we stitch them together in three separate steps:

1. For each vertex in the linked lists, find its closest vertex in other linked lists.

2. Based on the proximity information obtained from the previous step, find a pair of vertices A and B such that they are adjacent to each other in the linked list L (Fig. 1 (a)), and their closest merging vertices A' and B' are also adjacent to each other in the corresponding linked list L', in addition, the closest merging vertices of A' and B' are A and B, respectively. Starting from this pair of vertices A and B, iteratively go through the linked lists and if possible, connect each pair of adjacent vertices in one linked list to a corresponding vertex in another linked list and create a new triangle.

3. If there are more than two linked lists to be stitched together, then after stitching all the corresponding vertices, there may be some in-between gaps that need to be filled in. For example, in (Fig. 1 (a)), there is a gap between the linked lists L, L' and L" that consists of vertices B, C, C", C' and B'. We filled in the gap by creating a new vertex E at the center and connecting the new vertex E with all the other vertices in the loop (Fig. 1 (b)).

Topology Splitting Topology splitting occurs when a part of the surface tends to shrink to a single point. In this scenario, the surface has to split up into two parts precisely at that location. We use a method similar to [12]. In particular, a split-operation is triggered if there exists three neighboring vertices which are interconnected to each other, but the face formed by these three vertices does not belong to the model (i.e., a virtual face), and if the length of any of the three edges of the virtual face is smaller than the minimum edge length threshold and thus needs to be collapsed. For example, in Fig. 1(c), face ABC represents a virtual face that needs to be split. We divide the surface exactly at this location by cutting it into two open sub-surfaces. Then we close the two split-in-two surfaces using two faces $A_1B_1C_1$ and $A_2C_2B_2$ whose orientations are opposite to each other. Finally, we reorganize the neighborhood.

3. Surface Reconstruction

We have applied our PDE-based deformable surface to shape reconstruction from volumetric data, unorganized 3D point clouds and multi-view 2D images. The PDE we used is the general weighted minimal surface flow [3]:

$$\frac{\partial S}{\partial t} = (g(v + H) - \nabla g \cdot \vec{N})\vec{N}, \quad S(0) = S_0 \qquad (2)$$

where $S = S(t)$ is the 3D deformable surface, t is the time parameter, and S_0 is the initial shape of the surface. Note that, H is the mean curvature of the surface, and N is the unit normal of the surface. v is a constant velocity that will enable the convex initial shape to capture non-convex, arbitrary complicated shapes. It is also useful to avoid allowing the model to get stuck into local minima during the evolution process. g is a monotonic non-increasing, non-negative function that is used for interaction of the model with the datasets, and will stop the deformation of the model when it reaches the boundary of the object. In essence, Eq. 2 controls how each point in the deformable surface should move in order to minimize the weighted surface area. Hence, the detected object is given by the steady-state solution of the equation: $S_t = 0$, i.e. when the velocity F is zero. In order to apply the model to different types of data, we simply need to provide the definition of g that is appropriate for the dataset.

3.1. Volumetric Images

For volumetric data sets, the stopping function g is defined as:

$$g(S) = \frac{1}{1 + |\nabla(G_\sigma * I(S))|^2} \qquad (3)$$

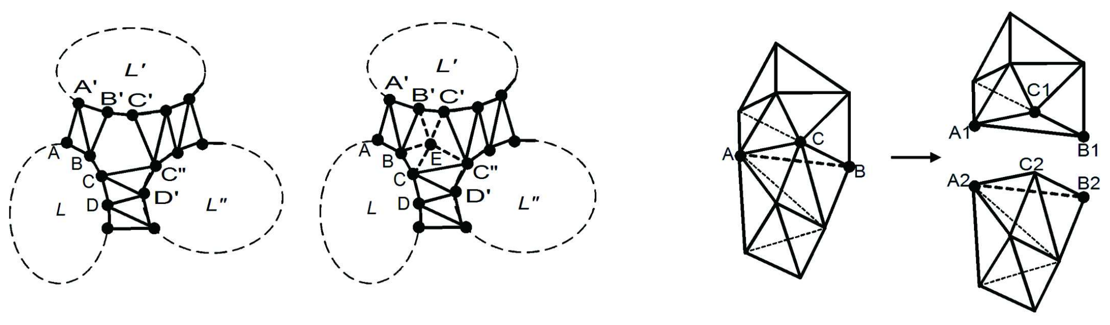
where I is the volumetric density function, and $G_\sigma * I$ is the smoothed density function by convoluting with a Gaussian filter with variance σ.

3.2. Point Clouds

For surface reconstruction from 3D unorganized point clouds, we use a simplified version of Eq. 2 with part $\nabla g \cdot \vec{N}$ removed. The stopping function g is:

$$g(p) = \begin{cases} 1, & \text{if } D(p) < T_D \\ 0, & \text{otherwise} \end{cases} \qquad (4)$$

where $D(p)$ is the distance between current position p and its closest data points, T_D is the threshold distance that is decided by the sampling rate of the point clouds datasets. In order to efficiently find the closest data points of any given position p, we preprocess the point clouds by putting them



(a) (b) (c)

Figure 1. Multiple contours stitching. (a) New triangles are created between corresponding vertices, and a gap is created by vertices $B, C, C", C', B'$. (b) The gap is filled in by creating a new vertex E in the center and connecting it with all the other vertices in the gap. (c) Topology split by splitting the virtual face ABC into two faces whose orientations are opposite to each other.

into a uniform regular grid and connecting all the points inside one grid cell by a linked list. The above distance threshold T_D will stop the movement of the model before it arrives at the "real" boundary of the object. To reduce the distance from the model to the boundary of the object, after the model stops its deformation, we will project each vertex point to the local tangent plane of its k-nearest neighbors.

The local tangent plane can be estimated using principle component analysis (PCA) [9]: For any point p, its local tangent plane can be represented by a center point c and a unit normal vector n. The center point c is the centroid of the one neighborhood of point p, which is denoted as $Nbhd(p)$. The normal vector n is computed by doing eigenanalysis of the covariance matrix C of $Nbhd(p)$, which is a symmetric 3 x 3 positive semi-definite matrix:

$$C = \sum_{p_i \in Nbhd(p)} (p_i - c) \otimes (p_i - c) \tag{5}$$

Here, $\otimes$ denotes the outer product vector operator, and the normal vector n is the eigenvector associated with the smallest eigenvalue of the covariance matrix C. In our experiments, k is set to five.

4. Experimental Results

In this section, we will show experimental results on both real and synthetic datasets. In all the following figures, grey regions represent parts of the model that are still active and deforming, black regions represent deactivated parts of the model that have already reached the boundary of the object. In order to illustrate the good mesh quality generated by our new model, we will show at least one wireframe view of the model in all the following figures. The input volumetric dataset of Fig. 2 is obtained from CT scanning of a phantom of the vertebra. The data size is $128 \times 120 \times 52$ voxels. Fig. 3 and Fig. 4 illustrate the surface reconstruction process from 3D unorganized point clouds. The input dataset of Fig. 3 is the mannequin head with 7440 data points. The original dataset has a big hole in the bottom of the head. We have manually filled in the hole since our model currently can only handle closed shapes. The input dataset of Fig. 4 is obtained by sampling a subdivision surface with 6140 data points.

5. Conclusion

In this paper, we proposed a deformable modeling based shape recovery algorithm that is capable of automatically evolving its shape to capture geometric boundaries and simultaneously discover their underlying topological structure. The deformation behavior of the model is governed by partial differential equations, that are derived by the principle of variational analysis. The model ensures regularity and stability, and it can accurately represent sharp features. We have applied our model to shape reconstruction from volumetric data and unorganized point clouds. Our mathematical formulation allows us to use the same model for different types of data, simply by using the appropriate data interface function. We plan to further exploit this property in future work to apply the model to heterogeneous data such as points, surfels, images and to incorporate other visual cues such as shading and optical flow.

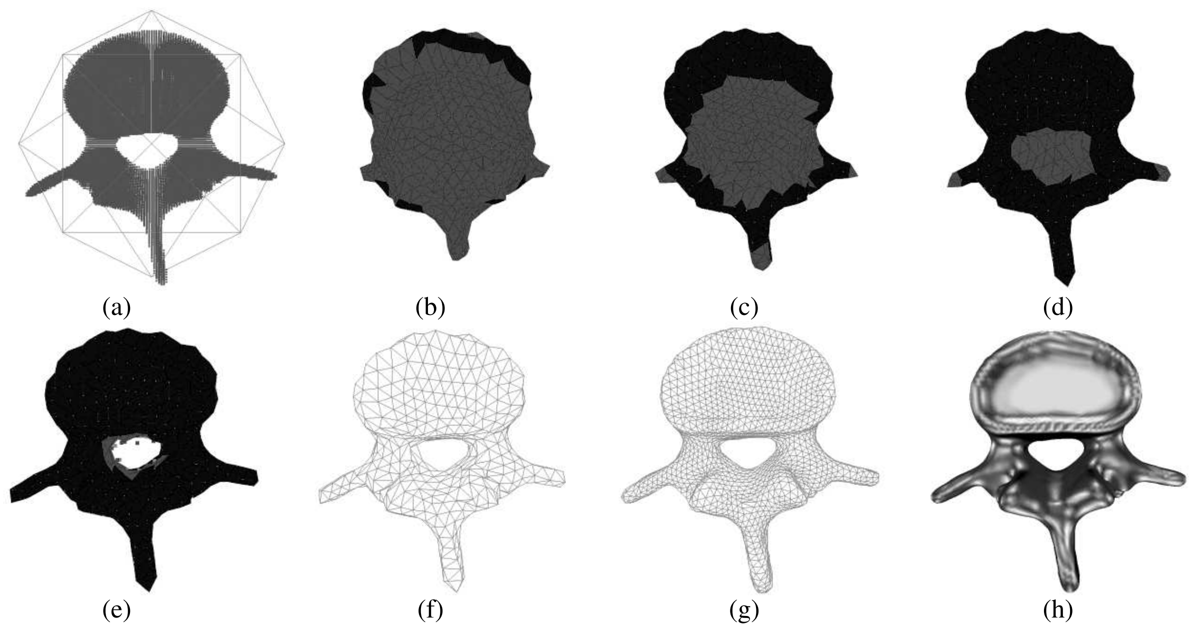


Figure 2. Segmentation of a CT volumetric dataset of a human vertebra. (a) Initial model and input data; (b)–(e) model evolving; (f) mesh model; (g) optimized mesh model; (h) shaded model.

References

[1] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. *SIGGRAPH*, pages 415–421, 1998.

[2] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. *SIGGRAPH*, 2001.

[3] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert. Minimal surfaces based object segmentation. *PAMI*, 19, 1997.

[4] Charles. Smooth subdivision surfaces based on triangles. Master's thesis, Mathematics, Univ. of Utah, August 1987.

[5] D. DeCarlo and D.N. Metaxas. Blended deformable models. *PAMI*, 1996.

[6] H. Q. Dinh, G. Slabaugh, and G. Turk. Reconstructing surfaces using anisotropic basis functions. *ICCV*, 2001.

[7] H. Edelsbrunner and E.P. Mucke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13:43–72, 1994.

[8] H.K.Zhao, S. Osher, and R. Fedkiw. Fast surface reconstruction using the level set method. *VLSM Workshop*, July 2001.

[9] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *SIGGRAPH*, 1992.

[10] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, pages 321–331, 1988.

[11] L. Kobbelt, T. Bareuther, and H.-P. Seidel. Multiresolution shape deformations for meshes with dynamic vertex connectivity. *Eurographics*, pages 249–260, 2000.

[12] J.-O. Lachaud and A. Montanvert. Deformable meshes with automated topology changes for coarse-to-fine 3d surface extraction. *Medical Image Analysis*, 3(2):187–207, 1999.

[13] W. Lorensen and H. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 1997.

[14] R. Malladi, J. Sethian, and B. Vemuri. Shape modeling with front propagation: A level set approach. *PAMI*, 17(2), 1995.

[15] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2), 1996.

[16] T. McInerney and D. Terzopoulos. Topology adaptive deformable surfaces for medical image volume segmentation. *TMI*, pages 840–850, 1999.

[17] D. Metaxas and D. Terzopoulos. Shape and nonrigid motion estimation through physics-based synthesis. *PAMI*, 15(6), 1993.

[18] J.V. Miller, D.E. Breen, W.E. Lorensen, R.M. O'Bara, and M.J. Wozny. Geometric deformed models: a method for extracting closed geometric models from volume data. *SIGGRAPH*, pages 217–226, 1991.

[19] Z. Wood, M. Desbrun, P. Schroder, and D. Breen. Semi-regular mesh extraction from volumes. In *Proceedings of IEEE Visualization*, pages 275–282, 2000.

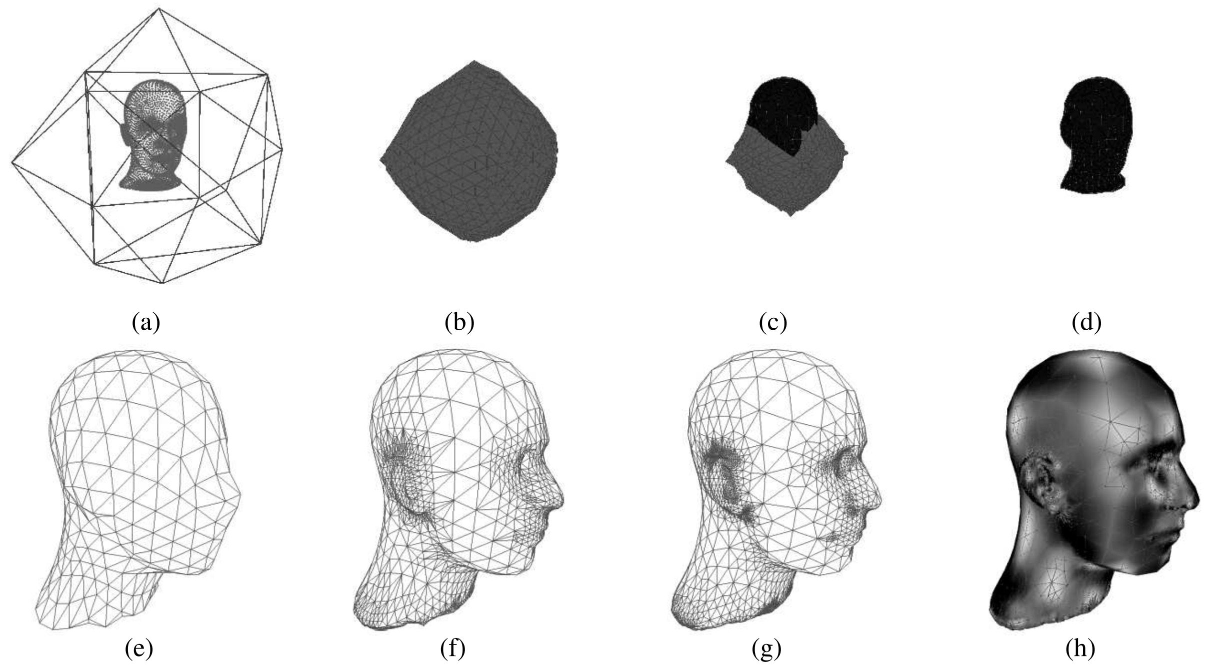


Figure 3. Reconstruction from point-clouds data of mannequin head. (a) model initialization; (b)–(c) during deformation; (d)–(e) final shape of the model; (f)–(g) are two different levels of adaptive refinement; (h) is shaded result

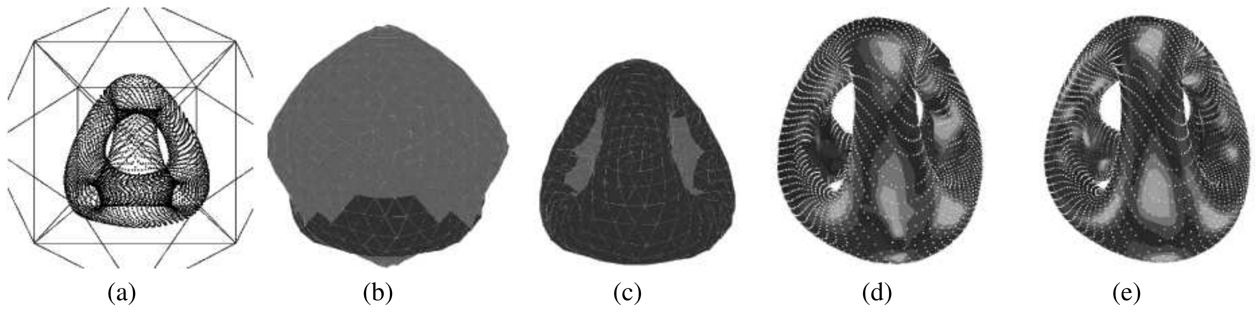


Figure 4. (a) model initialization; (b)–(c) during deformation, dark area stands for non-active, while grey is active; (d) final shape; (e) one level of refinement