Abstract

Skeleton is a lower dimensional shape description of an object. The requirements of a skeleton differ with applications. For example, object recognition requires skeletons with primitive shape features to make similarity comparison. On the other hand, surface reconstruction needs skeletons which contain detailed geometry information to reduce the approximation error in the reconstruction process. Whereas many previous works are concerned about skeleton extraction, most of these methods are sensitive to noise, time consuming, or restricted to specific 3D models.

A practical approach for extracting skeletons from general 3D models using radial basis functions (RBFs) is proposed. Skeleton generated with this approach conforms more to the human perception. Given a 3D polygonal model, the vertices are regarded as centers for RBF level set construction. Next, a gradient descent algorithm is applied to each vertex to locate the local maxima in the RBF; the gradient is calculated directly from the partial derivatives of the RBF. Finally, with the inherited connectivity from the original model, local maximum pairs are connected with links driven by the active contour model. The skeletonization process is completed when the potential energy of these links is minimized.

1 Introduction

Surface reconstruction and object recognition are all concerned about the shape description. Intuitively, the skeleton of an object is regarded as a good description to be applied to the above problems. Hence, how to extract skeletons, especially from arbitrary 3D objects, becomes an important subject in recent years. Medial Axis Transform (MAT), which is introduced by Blum [5] and has been widely adopted in the computer vision area, is a way to extract reliable skeletons from given shapes. The medial axis, as the locus of the inscribed circle centers inside a 2D object, is a kind of reduced representation and provides region-based shape features.

![Figure 1. Two arbitrary shapes and their skeletons](image)

Nevertheless, different applications may require different skeleton descriptions. To give a circumstantial account of above statement, we take two arbitrary shapes in Figure 1 for example. These two leaves have high similarities if we compare their skeletons with each other. However, if we use the left skeleton to reconstruct its own object, the error may not be acceptable since the left skeleton loses lots of the boundary information.

We take MAT as another example. A MAT skeleton performs well in surface reconstruction because it preserves all the boundary information of the given shape. However, if we try to apply a MAT skeleton to object recognition, sometimes it may contain too many branches of minor importance and using it as a comparison key may have undesired recognition results. In this paper, we try to focus on the skeleton which is able to represent the object both morphologically and geometrically. A 3D object can be transformed into an implicit surface, which has the following form:

\[ f(x) = h \]

where \( x \) is a point in \( \mathbb{R}^3 \) and \( h \) is a level value of a designated surface, usually we use \( h = 0 \) to indicate the object surface. The function can be regarded as a distance function [4]; it returns a higher value as the input point \( x \) gets deeper.
away from the object surface. Given an implicit surface of a 3D object, we can extract the skeleton from it by using techniques such as gradient descent and active contour model.

2 Previous Works

A good skeleton should be able to fulfill the following requirements:

1. Morphology: The skeleton must retain the morphology of the original shape, such as branching and termination.
2. Geometry: Force the skeleton to be in the middle of an object to preserve the region-based geometry features (e.g., surface bends).
3. Affine-invariant: The skeleton should be invariant under affine transformations, especially translation and rotation.

Many previous works focus on 2D shape skeletons. Traditionally, there are two kinds of approaches to construct the skeleton of an object. One is to compute the skeleton directly from the object surface points, such as using Voronoi diagram. The other is to extract the skeleton by constructing a distance field of an object.

Direct computation method: Voronoi diagram is widely adopted to construct 2D skeletons [21, 24], and this idea is straightforward to be extended into 3D space. The medial axis in 3D, which is the locus of the inscribed sphere centers inside a 3D object by definition, has already been discussed in many studies [23, 31]. With this concept, Edelsbrunner and Mücke [13] introduce \( \alpha \) shape to formalize an intuitive shape representation for the scattered data points. The \( \alpha \)-shape is a generalization of convex hull and a subgraph of the Delaunay triangulation. The real number \( \alpha \), which represents the radius of a sphere with \( 0 < \alpha < \infty \), controls the desired level of detail of the shapes.

Following the idea of \( \alpha \)-shape, Amenta et al. [2] propose the power crust algorithm for MAT approximation and 3D surface reconstruction. The algorithm computes the Voronoi diagram of the scattered data points first and then retrieves a set of polar balls by selecting candidates from the Voronoi balls that have maximal distance to the sampled surface. After labeling these polar balls with power diagram, the object described by the data points is now transformed into the polar ball representation. Connecting these polar ball centers forms a good approximation to the MAT.

A comparatively up-to-date direct computation method is to use the Reeb graph. The definition of the Reeb graph is introduced by Reeb [25]. The Reeb graph is regarded as a one-dimensional skeleton. Lazarus and Verroust [18] use the idea of Reeb graph to construct the skeleton called the level set diagram (LSD). Unfortunately, LSD is dependent on its source points, so different LSDs may be produced from the same model. Mortara and Patané [22] choose the source points in high curvature regions to guarantee the result is affine-invariant.

Distance field method: Building a 3D Voronoi diagram needs at least \( O(n^2) \) in time complexity where \( n \) is the input point number. Due to this reason, direct computing method is not suitable for the condition of large number of input points. Another approach is to use the distance field of an object to extract the skeleton. The basic methodology is as follows:

1. Construct the distance field of an arbitrary object.
2. Find the local maxima of the distance field.
3. Connect these local maxima to generate the skeleton.

Several distance field methods are proposed to produce skeletons. Leymarie and Levine [19] implement a 2D MAT (also called grassfire transformation) by minimizing the distance field energy of an active contour model. Zhou and Toga [33] propose a voxel-coding method which is based on recursive voxel propagation. The algorithm starts from a set of seed voxels, and then uses a coding schema to construct connectivity relationship and distance fields. Bitter et al. [3] propose a penalized-distance algorithm to extract skeleton from volumetric data. The algorithm first computes distance and gradient vector fields for locating the initial skeleton. Then, the skeleton is refined iteratively by discarding its redundant voxels. Chuang et al. [9] use each face in a polygonal object as a force field source, then several designated points, which are regarded as skeleton candidates, move to balanced positions in the force field. By tracing the locus of the candidates and connecting them together, a MAT skeleton can be constructed. Recently, Wade and Parent [30] use a discrete distance field to generate 3D skeletons for animation use.

3 Problem Analysis

Let \( S \) be a surface. A skeleton of surface \( S \) is a set of points denoted as \( M(S) \). For each point \( q \) on surface \( S \) we can find a point \( p \) which belongs to skeleton \( M(S) \), and we denote it as \( p = M(q) \). For each \( p \) on skeleton, we can also find an inverse of the skeleton point denoted as \( M^{-1}(p) \), which is a set of points on the surface. Then, if \( M(S) \) is a well-defined skeleton of surface \( S \), we propose that \( M(S) \) should satisfy the following hypotheses:

1. Neighborhood property: if \( p_1 \) and \( p_2 \) are neighbors in \( M(S) \), \( M^{-1}(p_1) \) and \( M^{-1}(p_2) \) are neighbors in \( S \).
2. Uniformity property: Let \( q \in S \), the distance between \( q \) and \( M(q) \) is denoted as \( \rho(q) \). Let \( p \in M(S) \), for all
3. Compactness property: The size of \( M(S) \) is minimal if the difference between \( M^{-1}(M(S)) \) and \( S \) is under a given error tolerance.

Algorithms based on property 2 produce symmetrical skeleton preferentially. Leyton [20] has inferred the Symmetry-Curvature Duality. The theorem indicates that each curvature extremum can be assigned a unique symmetry axis leading to the extremum. In another word, if the shape has more curvature variation, it may generate more skeleton branches. It explains that property 2 conflicts with property 3. MAT prefers property 2 so that it is suitable for surface reconstruction. However, we prefer property 3 since we want to find major skeleton points to preserve the primary shape structure. Due to this reason, we modify the property 2 by not limiting \( \rho(q_i) \) be the same value but trying to minimize the variant of \( \rho(q_i) \).

\[
\min \left( \int_{q \in M^{-1}(p)} \|\rho(q) - \text{Average}(\rho(q))\| dA \right)
\]

One benefit from the modified uniformity property is that a skeleton point now can embed an arbitrary shape instead of regular spheres (or circles) in MAT to represent part of the object.

Let \( \Omega_0 \) be a connected bounded domain in \( \mathbb{R}^3 \). A point \( p \) on the \( S \) can be denoted as \( p \rightarrow S \). If there exists a distance function \( D_{\text{g}}(x) \) that calculates the distance between point \( x \) and surface \( S \), a gradient operator \( \nabla D_{\text{g}}(x) \) can be applied and for \( p \rightarrow S \) we can apply the gradient descent \( G(p) = p^0 = p_n + \nabla D_{\text{g}}(p^n) \). If \( D_{\text{g}}(p^n) = 0 \), \( p^{n+1} \) is a local maximum and \( G(p) = p^{n+1} \). We then denote \( M \) as the set of local maxima. In our definition, the local maximum \( m_i \in M \) is a representative of the set \( P(m_i) = \{ p_j \mid G(p_j) = m_i \} \). We can complete the skeleton graph by linking local maximum pair \( m_i \) and \( m_j \) if \( P(m_i) \) and \( P(m_j) \) are adjacent. The linking process conforms to the neighborhood property.

4 Implicit Surfaces with Radial Basis Functions

Many previous studies have been done for implicit surfaces and RBFs [6, 12, 27, 28, 29]. Given a set of constraint points which define the shape of a 3D object, an implicit surface can be created by using 3D scattered data interpolation. Turk and O’Brien construct implicit surfaces with RBFs by solving a linear system [29]. This method is widely adopted for height interpolation or deformable models. In this paper, we invoke radial basis functions as height (distance) interpolation for building a distance field.

\[
f(x) = \sum_{j=1}^{n} d_j w(x - c_j) + p(x)
\]

where \( c_j \) are the surface constraints, \( d_j \) are weight coefficients, \( w(x) \) is a basis function and \( p(x) \) is a degree one polynomial for the linear and constant terms of \( f(x) \). To solve the weights and polynomial parameters we need to assign interpolation constraints as follows:

\[
h_i = f(c_i)
\]

then

\[
h_i = \sum_{j=1}^{n} d_j w(c_i - c_j) + p(c_i)
\]
Figure 4. Cross section pictures from a mouse model. (a) Exterior constraint $\lambda (\lambda_{ex}) = 0.0$, surface constraint $\lambda (\lambda_{surf}) = 0.0$. (b) $\lambda_{ex} = 1.0$, $\lambda_{surf} = 1.0$, (c) $\lambda_{ex} = 1.0$, $\lambda_{surf} = 0.1$ and (d) $\lambda_{ex} = 1.0$, $\lambda_{surf}$ = 0.01. The darker the color, the higher the RBF value is. The RBF level set seems to have better distribution when proper $\lambda$ values are applied. The control radius of all the above RBF level set is 0.05.

Assume that there are $n$ interpolation constraints, the equation above can be formulated as a linear system:

$$\begin{bmatrix} W & C \end{bmatrix} \begin{bmatrix} D \\ C^T 0 \end{bmatrix} = \begin{bmatrix} H \\ 0 \end{bmatrix}$$

where

$C_i = (1, c_i^x, c_i^y, c_i^z) \quad i = 1, ..., n$

$W_{ij} = w(c_i - c_j) \quad i, j = 1, ..., n$

$D = (d_1, ..., d_n)^T$

$P = (p_0, p_1, p_2, p_3)^T$

$H = (h_1, ..., h_n)^T$

The system is symmetric and positive semi-definite, so there is always a unique solution for $D$ and $P$. We may solve this linear system to get $D$ and $P$ using LU factorization [10] or singular value decomposition (SVD) [11].

As for the interpolation constraints, we apply the normal constrains method described in [29]. There are two kinds of constraints assigned to the linear system in this method:

1. Surface constraints: We use the vertex set $V$ of a arbitrary 3D model as surface constraints. For each vertex $v_i \in V$, we set the value $h_i = 0$.

2. Normal constraints: For each vertex $v_i$, we create the paired positive-valued normal constraint $r_i = v_i + k \times (-\mathbf{n})$, where $\mathbf{n}$ is the normal at vertex $v_i$. For each $r_i$, we set the value $h_i$ a positive value, usually $h_i = 1$. The value of $k$ depends on the size and resolution of an object. In our case, we set $k = 0.01$ for the 3D objects which are scaled within a $2 \times 2 \times 2$ cube.

We can choose many basis functions for the interpolation, usually the appropriate basis function to use is $w(r) = |r|^3$.

Dinh et al. [12] proposed a multiple order basis function that can control the smoothness of the implicit surface by changing basis function parameters. The basis function we invoke here is $w(r) = (|r|^2 + c^2)^{-0.5}$, where $c$ is the control radius. This function has a simple form and similar behavior to the multiple order one. Notice that the basis function must be continuous.

4.2 Properties of Radial Basis Functions

In this section, we will discuss about how the RBF parameters, such as interpolation constraints, radius and regularization parameters influence the RBF level set.

4.2.1 Interpolation Constraints

The RBF level set behavior is concerned with constraint distribution. The RBF of a sphere with uniform constraint distribution is prefect. All the level set surfaces are concentric spheres, which are shown in Figure 2 (a). As the reduced sphere models in Figure 2 (b) and (c), distortion happens in the RBF level set although the reconstructed surface boundary still seems the same. The RBF level set relaxes at the locations without the constraints, shown in the top region of the sphere in Figure 2 (c).

4.2.2 RBF Radius

Figure 3 shows the effect of the RBF control radius when applied to a reduced sphere model. As shown in Figure 3 (a), RBF surface with small control radius $c$ sometimes may become out of control. If we increase the control radius, the result will be better and smoother (Figure 3 (b) and (c)). Using large radius to construct a RBF surface, however, may lose some geometry details due to the global smoothing effect.
Figure 5. Illustrations of the skeletonization algorithm, where color representation is the same as in Figure 4. (a) The object and its RBF level set. (b) A gradient vector field (the small arrows) is constructed with the partial derivatives of RBF. Surface points (the black points) shrink to the local maxima (the white points) enforced by the gradient vector field. (c) After the local maxima being gathered, we link two local maxima with connectivity relationship together with a Snake (active contour model, the black dash line). The arrows indicate the external force that pushes the Snake toward the direction that decreases the potential energy. The Snake itself deforms continuously until the potential energy is minimized (the black solid line), and a skeleton is found.

4.2.3 Regularization Parameter

We apply the method proposed by Dinh et al. [12] for the RBF surface smoothing. Several λ values can be added to the diagonal elements in the RBF system matrix, which becomes:

\[
\begin{bmatrix}
w(r_{11}) + \lambda_1 & \cdots & w(r_{1n}) & 1 & c_1 \\
\vdots & \ddots & \vdots & 1 & \vdots \\
w(r_{n1}) & \cdots & w(r_{nn}) + \lambda_n & 1 & c_n \\
1 & 1 & 1 & 0 & 0 \\
c_1 & \cdots & c_n & 0 & 0
\end{bmatrix}
\]

The smoothness of the surface is proportional to the λ values. As the λ values become larger, the RBF surface becomes smoother. From the viewpoint of the RBF level set behavior, the level set seems to distribute better if we apply this technique, which is shown in Figure 4. The level set distribution behavior is important because it affects the RBF gradient vector field, which directs the convergence of the surface points.

5 Skeletonization Algorithm

Our skeletonization algorithm can be stated as follows:

1. Compute the RBF level set from a given 3D model.
2. For each point on the model, apply the gradient descent to it until a local maximum position is reached.
3. Cluster the local maxima in a given range.
4. Link any two maxima which have connectivity relationship together with the Snake (active contour model) method [17] and minimize the Snake energy.
5. Output the final positions of the Snakes as skeleton.
Figure 6. The skeleton deforming process, where it deforms successively until an energy-minimized position is found (From left to right).

Figure 7. Featured details of the selected skeleton. (a) The connection joint of the head and the eight tentacles of an octopus model. (b) The leg of a mouse model. (c) The head of a mouse model.

the final position of \( v \) is recorded as a local maximum. The whole process stops when all of the points remain in their corresponding local maxima. Figure 5 (b) illustrates the shrinking process.

5.2 Clustering

In practice, the final positions of the surface points may be close to, but not exactly the same as where the local maxima are. The range between the final position and the local maximum depends on the precision of gradient descent step. Due to this reason, we merge the final positions of these points in a given range to become a single local maximum point.

5.3 Connecting

As we use the vertices of the 3D model to be the gradient descent points, it is intuitive that the edges of a model become the connectivity relationship. For two shrinking points \( v_i \) and \( v_j \) and their corresponding local maxima position \( m_i \) and \( m_j \), the connectivity relationship is defined as if \( v_i \) and \( v_j \) are adjacent in the original 3D model, then \( m_i \) and \( m_j \) have connectivity relationship (either \( m_i = m_j \) or \( m_i \) and \( m_j \) should have a link between them). We employ a Snake (active contour model) method to link two local maxima which have connectivity relationship. An active contour model is generated by adjusting a contour to minimize an energy function \( E_{\text{Snake}} \):

\[
E_{\text{Snake}} = E_{\text{Internal}} + E_{\text{External}}
\]

where \( E_{\text{Internal}} \) and \( E_{\text{External}} \) represent the internal and external energy respectively. The internal energy depends on the intrinsic properties of the snake itself, such as its length or curvature. The external energy depends on the extrinsic factors, such as distance field in this case. Initially, we connect two local maxima as a straight line. We sample several points on the straight line as Snake candidates. The position of each candidate is adjusted on the constrained plane which is perpendicular to the original straight line (described in Appendix A). When the energy of the active contour model is minimized, we regard the Snake itself as a part of the skeleton (Figure 5 (c)). Figure 6 shows the deforming process of a Snake.
Model & Constraint & RBF & RBF construction & Shrinking & Skeleton construction
Mouse & 1,912 & 0.1 & 51.05 & 44.80 & 182.45
Bull & 1,982 & 0.2 & 55.93 & 79.96 & 210.33
Octopus & 2,004 & 0.1 & 57.44 & 41.25 & 344.62
Cobra & 1,752 & 0.2 & 39.17 & 62.13 & 110.04
Frog & 1,890 & 0.06 & 48.14 & 21.39 & 312.27
Pig & 2,170 & 0.1 & 74.83 & 74.28 & 294.30

Table 1. Execution time statistics on a desktop computer with Intel Pentium-4 processor running at 1.5 GHz and with 256 Mbytes RAMBUS memory. We apply MathWork MATLAB C++ Library for the RBF construction matrix computation.

6 Results

Detailed views of several models are shown in Figure 7. Six results from different models are shown in Figure 8 in colors. The execution time statistics for each model in Figure 8 is shown in Table 1. The error evaluation algorithm is described in Appendix B, and the skeleton we generated as compared to the ideal medial axis has a deviation from 2.9% to 8.5%.

Some detailed features, such as the hands of the mouse and the frog are not obviously because we apply larger RBF radius to make the level set smoother and the global smoothing effect has been applied to the fine features of the model. However, significant shape features are preserved well in the skeleton representation, and these are important to be used in object retrieval algorithms.

7 Conclusions and Future Work

Creating implicit surfaces with RBFs provides us a simple and robust method to generate arbitrary 3D object distance field, which is needed for building gradient vector field and then, surface point gradient descent. Our skeletonization method extracts both morphological and geometrical information. Significant geometry details, such as arms and legs in the model, are well preserved. There are many works need to be done in the future. From the viewpoint of research, we will keep working on the complicated relationship between the implicit surface, constraint distribution, control radius and regularization parameters. As for the basis functions, currently we only use the basis function \( w(r) = (|r|^2 + c^2)^{-0.5} \) for all the RBF level set generation. In the future, we may continue to explore the RBF level set property with different basis functions. Computation efficiency is also an important task to be optimized. Meanwhile, since how to build a good RBF for our purpose still needs more consideration, recently we also try to use a different method which uses visibility and a repulsive force model to extract 3D skeletons. [32].

Besides, from the viewpoint of applications, there are several projects related to this method. For example, a 3D object retrieval system is going to take the skeletons as main comparison keys. We still need to evaluate the results compared with the method proposed by Hilaga et al. [16], Funkhouser et al. [14] and Chen et al. [7, 8]. We also want to apply the skeleton to the original model for animation use. We may extract a skeleton from a 3D model and select several points on the skeleton as joints. The pose of the 3D model can be changed by adjusting the skeleton. With this idea, we want to develop a system for artists to simply edit the skeleton movements and accurately animate the models with skin mesh method. This may become a useful tool for key frame animation.

Acknowledgements

We would like to thank the reviewers for their valuable comments. We also thank 3DCAFE [1] to provide all 3D models used in this paper. This work was partially supported by grants from National Science Council, Taiwan, R.O.C. under NSC91-2213-E-002-066 and Silicon Integrated Systems (SiS) Education Foundation.

Appendices

A Coordinate Transformation

To adjust a point which moves in a plane, we can define a point as \( r = (p_x, p_y, 0) \) in plane coordinate. The normal of the plane is as \( \mathbf{n} = (n_x, n_y, n_z) \) that is also the straight line between two local maxima. A rotation axis is \( \hat{\mathbf{n}} = \mathbf{z} \times \mathbf{n} = (-n_y, n_x, 0) \). A rotation angle is \( \Phi = \cos^{-1}(\mathbf{z} \cdot \mathbf{n}) \). As Figure 9 shows, we can find the relation between before and after rotation[15].

\[
r' = \mathbf{O}N + \mathbf{N}V + \mathbf{V}Q
\]
B Error Evaluation

Let \( \hat{n} = (x, y, z) \), then

\[
\mathbf{r} = \mathbf{r} + \hat{n}(\hat{n} \cdot \mathbf{r})(1 - \cos \Phi) - (\mathbf{r} \times \hat{n}) \sin \Phi
\]

where \( \mathbf{M} \) equals to

\[
\mathbf{M} = \begin{bmatrix}
  x^2 + \cos \Phi(1 - x^2) & x y (1 - \cos \Phi) - z \sin \Phi & x z (1 - \cos \Phi) + y \sin \Phi \\
 x y (1 - \cos \Phi) + z \sin \Phi & y^2 + \cos \Phi(1 - y^2) & y z (1 - \cos \Phi) - x \sin \Phi \\
 x z (1 - \cos \Phi) - y \sin \Phi & y z (1 - \cos \Phi) + x \sin \Phi & z^2 + \cos \Phi(1 - z^2)
\end{bmatrix}
\]

Based on this relation, we can transform a point in plane coordinate into world coordinate \( \mathbf{p}_w \) as follows:

\[
\mathbf{p}_w = \mathbf{M} \cdot \mathbf{r} + \mathbf{T}
\]

where \( \mathbf{T} \) is the position of a shrinking candidate.

B Error Evaluation

Here we describe the error evaluation method used in this paper:

1. The surface of the input model is densely sampled using \( n \) points (In our evaluation, the models are sampled by more than 20,000 points).
2. For each surface point \( v \), find the nearest point \( k \) on the skeleton then define \( v \in M^{-1}(k) \) and the distance between \( v \) and \( k \) is \( \rho(v) \).
3. For each \( k \), sort each \( v \in M^{-1}(k) \) and find the smallest \( \rho(v) \) and define it as \( r(k) \), which is the zero error position. For each surface point \( v \) we calculate the error \( e(v) = \frac{\rho(v)}{r(k)} \).
4. The total MSE equals to \( \sqrt{\frac{\sum e^2(v)}{n}} \)

References


