Efficient 3D Curve Skeleton Extraction from Large Objects[*]

László Szilágyi, Sándor Miklós Szilágyi, David Iclănzan, and Lehel Szabó

Sapientia - Hungarian Science University of Transylvania,
Faculty of Technical and Human Science, Tîrgu-Mureş, Romania
lalo@ms.sapientia.ro

Abstract. Curve skeletons are used for linear representation of 3D objects in a wide variety of engineering and medical applications. The outstandingly robust and flexible curve skeleton extraction algorithm, based on generalized potential fields, suffers from seriously heavy computational burden. In this paper we propose and evaluate a hierarchical formulation of the algorithm, which reduces the space where the skeleton is searched, by excluding areas that are unlikely to contain relevant skeleton branches. The algorithm was evaluated using dozens of object volumes. Tests revealed that the computational load of the skeleton extraction can be reduced up to 100 times, while the accuracy doesn't suffer relevant damage.

Keywords: 3D curve skeleton, potential fields, hierarchical algorithm, parallel computation, graphical processing units.

1 Introduction

In the two dimensional case, the medial axis of an object is the collection of points that have at least two closest points to the boundary of the object. On the other hand, the skeleton is defined as the locus of centers of maximal circles inscribed within the object. A circle C is considered maximal, if there is no other circle inscribed in the object that entirely contains C. In two dimensions, the medial axis and the skeleton are practically the same.

In three dimensions, the medial surface is the term that corresponds to 2D medial axis. The medial surface also contains linear curves in places where there are at least three surfaces at the same distance. Figure 1(left) shows the medial surface of an object. There are several 3D computer graphics applications, where 3D objects are desired to be represented as a collection of linear curves. For example, the inverse kinematics, which is very practical and thus quite popular in animations, demands such a representation. However, the 3D skeleton of an object does not have a mathematical definition. We could say that the 3D curve skeleton (3DCS) is a subset of the medial surface, a collection of curves which is centered within the object, but that is not a rigorous definition. Figure 1(right) the desired shape of the 3DCS of an object.

[*] This work was supported by the Sapientia Institute for Research Programs.

There are several 3DCS extraction algorithms, which approximate this curve collection based on different physical approaches. All of them give an approximation of the 3DCS. As we will see later, the segments of the skeleton shown in Fig. 1(right) will be slightly bent in the proximity of bifurcation points.

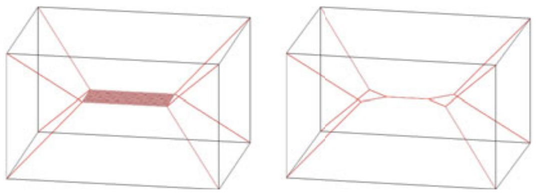

Fig. 1. Medial surface (left) and 3D curve skeleton (right) of a deformed cube

The most important properties the 3DCSs share are:

1. Homotopic: the curve skeleton is required to be topologically equivalent to the original object, that is, the number of connected components, tunnels and cavities should be the same [11].
2. Invariant to isometric transformations: this is important in applications where the skeleton is used as shape descriptor.
3. Thin: curve skeletons are one dimensional. In discrete applications they can be either single voxel wide lines, or zero width lines described by points having real valued coordinates.
4. Centered: the curve skeleton lays within the medial surface of the objects, and is centered within the surface patches it belongs to [12].
5. Curve skeletons are reliable if all surface points are visible from some place on the skeleton [9].
6. A curve skeleton extraction method is robust, if the resulting skeleton is not sensitive to small variations of the boundary.

In the discrete case, 2D skeleton extraction is traditionally performed by thinning via hit-or-miss transform, by applying the L mask family from Golay's alphabet. Most of the algorithms developed for 3DCS extraction also work in 2D. They will be enumerated in the followings. The literature of 3D curve skeleton extraction consists of several various approaches:

1. Thinning and boundary propagation methods iteratively remove so-called *simple points* from the objects, which by definition, do not influence the topology of the object [3]. Methods vary according to the criteria they apply to find simple points.
2. Distance field based methods apply the distance transform with a selected chamfer metric, for each internal point of the object, and extract the skeleton from the distance field data [4].
3. Geometric methods are generally applied to objects described as triangular or polygonal meshes. The most popular algorithmic scheme in this family is mesh contraction [2].

4. General field based methods were first introduced in 2D formulation [1,8]. 3D solutions use potential fields [5,6], repulsive force fields [13,14], or radial basis functions [10] to create a general field. The skeleton is extracted afterwards based on some special points (complete extinguishment and saddle points) within the field.

2 Generalized Potential Field

According to the theory of electrostatics, every electrical charge generates a potential field. This means that whenever another charged object approaches the initial one, it will be attracted or repelled according to the sign of both charges. The magnitude of the force is computed as: $F = kq_1q_2/d^2$, where k is a constant, d is the distance between the two charged objects, while q_A an q_B are the two charges. In case of the generalized potential field applied in computer graphics, the constant k is neglected, the charges are considered unitary and of same polarity, while the distance is treated in a generalized way, in the sense that the magnitude of the force is equal to the $-\alpha$'th power of the distance: $F = d^{-\alpha}$. In other words, the force vector that applies to charged object B, because of the presence of charged object A, is: $\boldsymbol{F} = \boldsymbol{d}_{AB}|\boldsymbol{d}_{AB}|^{-(\alpha+1)}$, where $\boldsymbol{d}_{AB}$ represents the distance vector between the objects. Expressed in 3D Euclidean coordinates, the components of the above force vectors are:

$$F_x = \frac{x_B - x_A}{|\boldsymbol{d}_{AB}|^{\alpha+1}} \qquad F_y = \frac{y_B - y_A}{|\boldsymbol{d}_{AB}|^{\alpha+1}} \qquad F_z = \frac{z_B - z_A}{|\boldsymbol{d}_{AB}|^{\alpha+1}} \ , \tag{1}$$

where A and B represent the points where the charges are placed, and the length of the vector $\boldsymbol{d}_{AB}$ is $|\boldsymbol{d}_{AB}| = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2 + (z_B - z_A)^2}$.

The generalized potential field is applied as follows. At first, electrical charge is placed and uniformly distributed upon the outer surface of the object. In the following step, the generalized potential field is computed in every internal grid point of the object. Or in other words, a small object with unit charge virtually marches over every internal grid point, and the electrostatic force vector that applies to the object is computed for each position.

The formula of the GPF is deduced by summing all its components. All external point charges may have their effect to the electrostatic field in any internal point, so we need to sum up all vectors:

$$\mathcal{F}(I) = \sum_{P \in \Omega} F_x(I,P) \times \boldsymbol{i} + \sum_{P \in \Omega} F_y(I,P) \times \boldsymbol{j} + \sum_{P \in \Omega} F_z(I,P) \times \boldsymbol{k} \ , \tag{2}$$

where $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ represent the unit vectors in the three main axial directions, and Ω is the set of surface points that hold the external charges. The above formula is a good approximation of the GPF in internal point I, but is not exact. As it was already pointed out (but not applied) in [7], only those external points should be included in Ω, from which the internal point I is visible. Visibility can be described mathematically with the following expression:

$$I \text{ is visible from } P \Leftrightarrow \lambda I + (1-\lambda)P \text{ is an internal point } \forall 0 < \lambda \leq 1 \ . \tag{3}$$

The identification of the curve skeleton from the computed GPF is produced in several steps:

1. Find the critical points, where GPF is a zero vector. These will generally be points where two or more branches of the skeleton meet, or in other words, points where the skeleton bifurcates. They usually will not fall in exact grid points, but we can easily locate the unit sized cube volume where they are situated. We need to look for those cubes where all three vector components $\mathcal{F}_x$, $\mathcal{F}_y$, and $\mathcal{F}_z$ change their sign.
2. At the time when critical points are located, we can also check, which are the cubes where not all three, but at least one of the three components change their sign. These cubes will contain so-called saddle points, which will also contribute to the curve skeleton.
3. Compute the exact coordinates of critical points using trilinear interpolation.
4. Locate the segments of the first order curve skeleton. Each such segment must have critical points at both ends, and all such segment should cross only such cubes where saddle points are located. The segments are identified using a backtracking algorithm. The small number of possible ways assures the quick performance of the backtracking algorithm.
5. The exact coordinates of saddle points participating in the skeleton are established along the following two considerations. Those components that change their sign within the saddle point's cube, precisely define one or two exact coordinates via linear or bilinear interpolation. The other coordinates can be established such a way, that saddle points are uniformly distributed along the skeleton, and the curve of the skeleton remains smooth.
6. The second order curve skeleton will additionally contain branches, which connect critical points with high curvature surface points of the object. In this order, a set of high curvature surface points is searched for. Curvature is easily represented by the number of inner neighbors of the surface point. The less inner neighbors a surface point has, the higher its curvature value is. Further on, only those high curvature points are kept within the set, which have a locally maximal curvature value.
7. Secondary branches of the curve skeleton are located again using backtracking, crossing only cubes that contain saddle points.
8. The GPF based curve skeleton extraction algorithm also gives the possibility to neglect some of the irrelevant secondary branches. This is performed via ordering the branches according their divergence, and keeping only those which have a lower value than a predefined threshold, or keeping a predefined percentage that have low divergence values.

3 The Proposed Hierarchical Approach

The GPF based curve skeleton extraction algorithm given in [6] reports a very long execution time in case of objects containing over $N = 10^5$ voxels. The approximate length of such an object is of $n = \sqrt[3]{N}$ units.

The most time consuming part of the algorithm is the computation of the GPF in each grid point, as each couple formed by an internal point and a surface point has to be taken into consideration. As the number of internal points has the order of $\mathcal{O}(n^3)$, and the surface points count up to the order of $\mathcal{O}(n^2)$, the complexity of GPF computation will be of order $\mathcal{O}(n^5)$. This is an enormous load in case of large objects.

Cornea tried to reduce the duration of GPF computation by suppressing the set of external points considered for each internal point. Those external points were neglected, which according to their z coordinate, were guaranteed to be too far from the currently processed internal point. This modification reduces some of the computational complexity, but also may cause deformations of the skeleton. In this paper we propose a hierarchical approach for 3DCS extraction of large objects. The general idea is to reduce the number of internal points where the GPF is computed. This is achieved along the following terms:

Let us resize the object, reduce its size μ times in every direction. For the case of simplicity, let us consider now $\mu = 2$. This practically reduces the number of internal voxels μ^3 times and the number of surface voxels μ^2 times. If we extract the skeleton of this reduced object, it will have approximately the same shape as the skeleton of the original, large object, just it will be reduced in size μ times.

Now let us turn back to the original large object. We magnify the skeleton of the small object μ times and place it into the large object. Only those internal voxels, which are situated closer to the magnified skeleton, than a predefined threshold distance δ will be considered for the time consuming process of GPF computation. Such a way, the $\mathcal{O}(n^5)$-complexity operation is executed with a μ times smaller n, while in the big object, GPF computation will have the complexity of $\mathcal{O}(n^3)$.

This hierarchical size reduction can be performed in more than one steps, too, achieving thus μ^2 or μ^3 times size reduction, and an even more convenient computational load.

The reduction of the object can be performed as long as it does not influence the topology of the object. Further on, it is limited by the fact, that the reduced object must be at least two pixels wide everywhere. The recommended threshold value is: $\delta = (1.5 - 2.0)\mu$ voxels.

4 Efficient Implementation Using GPU

Modern GPU's can efficiently handle large matrices and can perform quick computations on large amounts of data. That is why, if we wish to accelerate the computation of GPF using a GPU, we need to organize our data into matrices. Let us write the coordinates of surface points into a matrix denoted by $\mathcal{S}$:

$$\mathcal{S} = \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \cdots & \cdots & \cdots \\ x_\omega & y_\omega & z_\omega \end{pmatrix},$$

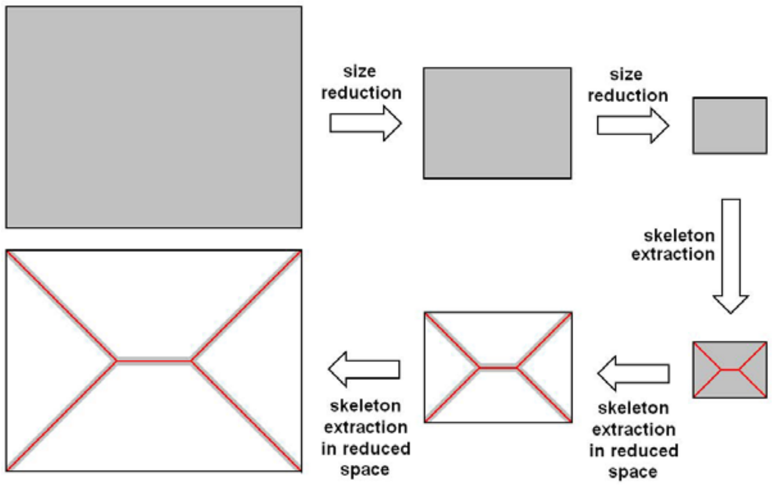
size reduction
size reduction
skeleton extraction
skeleton extraction in reduced space
skeleton extraction in reduced space

Fig. 2. Hierarchical size reduction exhibited in 2D, in two steps. Red lines indicate the extracted skeleton, while gray areas are the regions where the skeleton is looked for. White areas are excluded from the computations as they are not likely to contain any part of the skeleton.

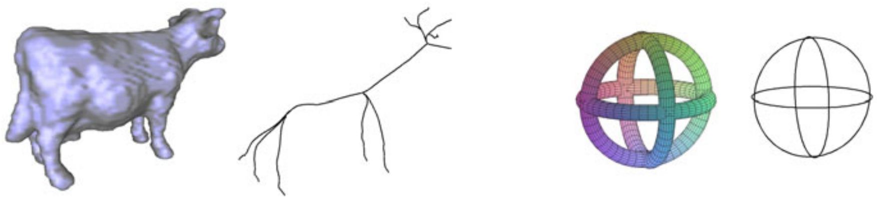

Fig. 3. Some examples of extracted skeletons

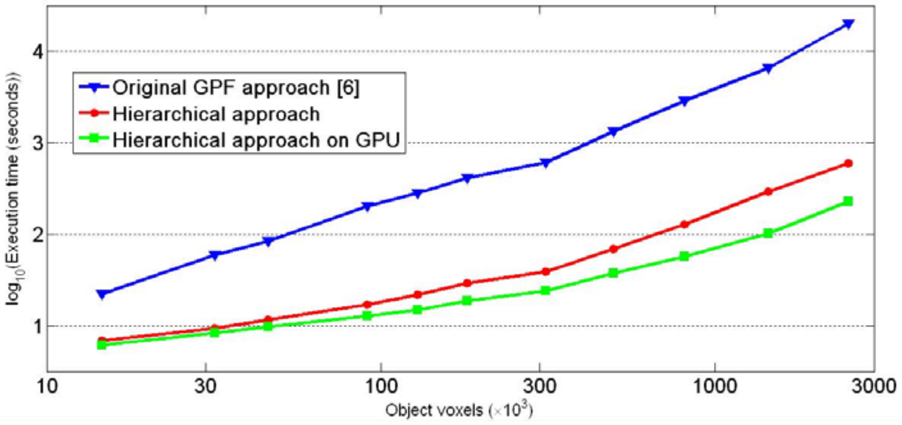

Fig. 4. Execution times for various object volumes, obtained by the original GPF-based skeleton extraction method, and the proposed hierarchical approaches

where ω represents the cardinality of the set of surface points Ω. We keep the coordinates of the current internal point in a vector $I = [x_I, y_I, z_I]$. We need to perform the following steps. First, we subtract I from each row of $\mathcal{S}$, and compute the square of each element in the matrix. Compute the sum of the three columns of the matrix, and denote this single column by D. Raise each element of D to the power $(1 + \alpha)/2$. Use an odd integer for α to make the exponent an integer. Build an ω-row 3-column matrix M by placing ω identical rows $[x_I, y_I, z_I]$ under each other. Subtract the original $\mathcal{S}$ from this matrix M, and denote the result by Q. Divide each row of Q with the corresponding element in D, and finally sum up all columns of the matrix. The resulting 3-element vector will contain the three directional components of the GPF in the point I, namely $[\mathcal{F}_x(I), \mathcal{F}_y(I), \mathcal{F}_z(I)]$. In a GPU, the above computations can be organized to perform the above operations on several internal points in parallel.

5 Results and Discussion

The algorithm has been implemented in c++ programming language, using the JAMA and TNT packages for matrix operations. GPU implementation relies on AMD's FireStream SDK, and an ATI HD 5750 video card.

Several tests have been performed using artificially created object volumes of different sizes. The main evaluation criteria were: accuracy – how much the hierarchical formulation influences the correctness of the extracted skeleton, and efficiency – how many times quicker the proposed approach extracts the skeleton, compared to the speed of the original formulation.

The most simple skeleton extracted by the GPF based algorithms is the core skeleton, which is formed of critical points and first order branches only. The presence of secondary branches is controlled by the divergence threshold. These secondary branches are rarely useful in computer graphics applications, so the divergence threshold has to be kept at a low value. Another reason for this low threshold value could be the possible sensitivity to the noise present on the boundary of the object.

Parameter α controls the variation of the potential field's strength with distance. The optimal value of this parameter we found $\alpha = 5$: this assures good stability of the algorithm, while the computational load is not raised too high.

The hierarchical size reduction of the object has a damaging effect upon accuracy only if the size reduction changes the object's topology. The preservation of topology should limit the maximum applied size reduction factor.

From the point of view of efficiency, we found that the proposed approach can reduce the computational load of the algorithm $5 - 50$ times, depending on the chosen size reduction ratio. Figure 4 shows some execution times obtained on various object volumes. At the skeleton extraction of larger objects, it is more likely to obtain a higher speedup ratios. Involving a GPU for GPF computation can produce a further speedup factor of 2 to 5.

6 Conclusions and Future Work

In this paper we have proposed a hierarchical formulation of the general potential field based 3D curve skeleton extraction algorithm. The proposed method proved accurate as long as the executed size reduction did not change the topology of the object. The computational efficiency of the proposed method is outstanding, a 5-50 times speedup ratio is achievable, depending on the size of the object. Implementing the proposed hierarchical formulation of GPU's can rise the speedup factor well above 100.

As a future work, we would like to implement a different computation method of the GPF field, which will take into consideration the visibility problem, thus improving the accuracy as well.

References

1. Ahuja, N., Chuang, J.: Shape representation using a generalized potential field model. IEEE Trans. Patt. Anal. Mach. Intell. 19, 169–176 (1997)
2. Au, O.K.C., Tai, C.L., Chu, H.K., Cohen-Or, D., Lee, T.Y.: Skeleton extraction by mesh contraction. ACM Trans. Graph. 27(3), article 44, 1–10 (2008)
3. Bertrand, G., Malandain, G.: A new charactrization of three-dimensional simple points. Patt. Recogn. Lett. 15, 169–175 (1994)
4. Bitter, I., Kaufman, A.E., Sato, M.: Penalized-distance volumetric skeleton algorithm. IEEE Trans. Vis. Comp. Graph. 7, 195–206 (2001)
5. Chuang, J., Tsai, C., Ko, M.C.: Skeletonization of three- dimensional objects using generalized potential field. IEEE Trans. Patt. Anal. Mach. Intell. 22, 1241–1251 (2000)
6. Cornea, N.D., Silver, D., Yuan, X., Balasubramanian, R.: Computing hierarchical curve-skeletons of 3D objects. The Visual Computer 21, 945–955 (2005)
7. Cornea, N.D., Silver, D., Min, P.: Curve-skeleton properties, applications, and algorithms. IEEE Trans. Vis. Comp. Graph. 13, 530–548 (2007)
8. Grigorishin, T., Yang, Y.H.: Skeletonization: an electrostatic field-based approach. Patt. Anal. Appl. 1, 163–177 (1998)
9. He, T., Hong, L., Chen, D., Liang, Z.: Reliable path for virtual endoscopy: ensuring complete examination of human organs. IEEE Trans. Vis. Comp. Graph. 7, 333–342 (2001)
10. Ma, W.C., Wu, F.C., Ouhyoung, M.: Skeleton extraction of 3D objects with radial basis functions. In: Proc. IEEE Int'l Conf. Shape Modeling and Applications, pp. 1–10 (2003)
11. Saha, P.K., Chaudhuri, B.B.: 3D digital topology under binary transformation with applications. Comp. Vis. Image Understand. 63, 418–429 (1996)
12. Sanniti di Baja, G., Svensson, S.: A new shape descriptor for surfaces in 3D images. Patt. Recogn. Lett. 23, 703–711 (2002)
13. Wang, Y.S., Lee, T.Y.: Curve-skeleton extraction using iterative least squares optimization. IEEE Trans. Vis. Comp. Graph. 14, 926–936 (2008)
14. Wu, F.C., Ma, W.C., Liou, P., Liang, R.H., Ouhyoung, M.: Skeleton extraction of 3D objects with visible repulsive force. In: Proc. Eurographics Symp. Geometry Processing, pp. 1–7 (2003)