Curve-skeleton Extraction from Visual Hull

Andrey Zimovnov and Leonid Mestetskiy
Faculty of Computational Mathematics and Cybernetics, Moscow State University, Moscow, Russia

Keywords: Curve-skeleton, Visual Hull, Medial Axis, Shape Descriptor, Mean-shift.

Abstract: We present a new algorithm of curve-skeleton extraction from a wide variety of objects. The algorithm uses visuall hull object approximation, which gives us an ability to work with the model in its silhouettes domain. We propose an efficient algorithm for 3D distance transform computation for the inner voxels of visual hull. Using that 3D distance transform we backproject continuous medial axes of visual hull silhouettes that form a first approximation for a curve-skeleton. Then we use a set of filtering techniques to denoise that point cloud to form a thinner approximation. We believe that a resulting approximation is usefull in its own. The described method shows a great improvement in computational time comparing to existing ones. The method shows good extraction results for models with complex geometry and topology. Resulting curve-skeletons conform with most requirements to universal curve-skeletons.

1 INTRODUCTION

Curve-skeleton extraction is a popular topic nowadays. Curve-skeleton is essentially a graph that depicts simplified versions of object's geometry and topology. An inscribed sphere is associated with every node of curve-skeleton. The envelope of all curve-skeleton spheres aims to approximate the shape of the object.

Curve-skeletons find many applications in problems where object's shape analysis is needed, such as object recognition, shape classification, object skeletal animation, object segmentation, finding visually similar objects in databases and others (Cornea et al., 2005).

Most existing methods are based on either voxel thinning techniques or polygonal contraction procedures. The former algorithms are time consuming, though show good results with a lot of desirable properties (Sobiecki et al., 2014). The latter are faster, but at the same time are very sophisticated and difficult to implement with their own disadvantages (Sobiecki et al., 2013). The contraction based methods are considered to be a state of the art nowadays.

Recent research (Livesu et al., 2012; Kustra et al., 2013) introduces a new promising approach to curve-skeleton extraction based on visual hull (Laurentini, 1994). In paper (Mestetskiy and Tsiskaridze, 2009) authors utilize the observation that on a projection of object without occlusions silhouette's medial axis is the projection of curve-skeleton's bones. Although object's occlusions lead to spurious curves in resulting skeleton, experiments (Livesu et al., 2012) show that even with occlusions projections store enough information for curve-skeleton extraction.

In this paper a new approach to curve-skeleton extraction using visual hull is proposed. Our method uses continuous medial axes extraction from object's silhouettes (Mestetskiy and Semenov, 2008) to organize an efficient iterative contraction process similar to (Au et al., 2008).

2 PROPOSED METHOD

The proposed method is an iterative process. One iteration can be divided into the following steps:

- Curve-skeleton approximation with point cloud reconstructed from silhouettes' medial axes.
- Object contraction based on inscribed spheres radii reduction associated with every point of the approximating cloud.

2.1 Curve-skeleton Approximation with Point Cloud

Proposed method is based on the observation that on a projection of object without occlusions silhouette's medial axis is the projection of curve-skeleton's

Zimovnov A. and Mestetskiy L..
Curve-skeleton Extraction from Visual Hull.
DOI: 10.5220/0005359806660671
In *Proceedings of the 10th International Conference on Computer Vision Theory and Applications* (VISAPP-2015), pages 666-671
ISBN: 978-989-758-089-5

bones. Assuming there are no occlusions on a projection, curve-skeleton can be reconstructed by medial axis back-projecting. To eliminate the influence of object's occlusions we propose to use a set of projections. The main idea is to back-project object's parts that are visible without occlusions and filter out spurious occluded parts. In this way we are effectively extracting a curve-skeleton from visual hull approximation of an object, which is an intersection of prisms with object's silhouettes in their bases. A strict definition of Visual Hull (VH) is as follows:

$$VH = \{x \in \mathbb{R}^3 \colon \forall i \ \mathrm{Pr}_i(x) \in S_i\},$$

where $\{S_i\}$ — set of object's silhouettes, $\mathrm{Pr}_i(x)$ — orthogonal projector of point $x \in \mathbb{R}^3$ on the plane of silhouette S_i. Orthogonal projections defining a visual hull are captured from uniformly distributed cameras on a hemisphere around the object (Fig. 1). Uniformly distributed points on a hemisphere are generated using hexahedron subdivision. We need only cameras on one hemisphere since silhouettes from another one will not change the resulting visual hull due to symmetry. Filtered parts acquired from different projections are combined into one graph and are aligned with mean-shift algorithm, which is a kind of denoising technique.

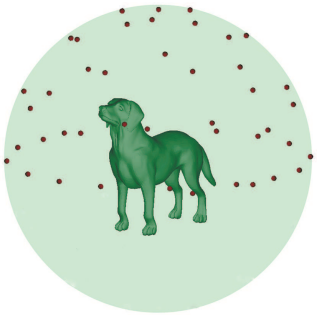

Figure 1: Uniformly distributed cameras on a hemisphere defining a visual hull.

To sum up, the approximation process can be described with the following steps:

- Back-projection of silhouettes' medial axes
- Alignment of skeleton nodes and bones from different projections.

2.1.1 Back-projection of Silhouettes' Medial Axes

For each silhouette, forming a VH, we extract a medial axis which is a continuous graph (Mestetskiy and Semenov, 2008) with nodes and bones (Fig. 2). To back-project the node of medial axis we need to assign a depth value to it along the ray, which is casted

Figure 2: Object's projection, silhouette and medial axis.

from the node towards a camera view. For that purpose we first extract a 3D distance transform of the inner voxels of visual hull. The 3D distance transform $DT_3(v)$ is defined for each voxel $v \in \mathbb{R}^3$ of object O and its surface ∂O as follows:

$$DT_3(v) = \min_u \{\rho(v,u) | u \in \partial O\}.$$

We propose a novel algorithm for the 3D distance transform extraction for the visual hull. It turns out that having a 2D distance transforms $DT_{2,i}(p)$ for pixels of visual hull silhouettes the 3D distance transform $DT_3(v)$ can be calculated as follows:

$$DT_3(v) = \min_i \{DT_{2,i}(\mathrm{Pr}_i(v))\}$$

Fig. 3 shows voxels with different DT_3 values for the elephant polygonal model. We can observe that DT_3 extraction is valid.

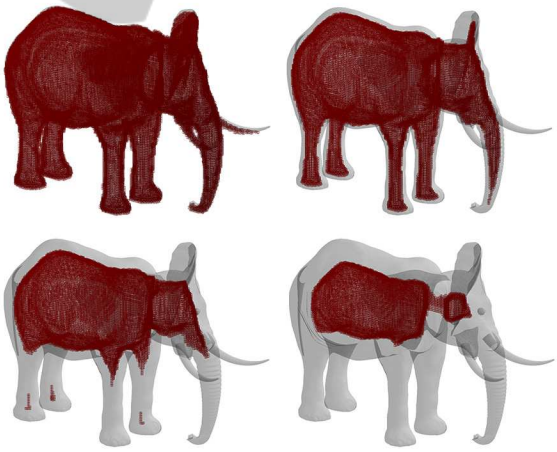

Figure 3: From left to right, top to bottom: voxels with DT_3 values of 0, 3, 6 and 12 accordingly.

Having DT_3 values for all inner voxels of visual hull, we can now find the optimal depth values for medial axes' nodes. For every medial axis node of each silhouette we cast a ray in the direction of that silhouette camera view and find all intersections with visual hull inner voxels. Intersections along the ray are tested with a discrete step equal to one voxel size in object coordinates. All the voxels with maximum DT_3 values on this ray are stored in a temporary list.

The middle element of this list is taken as a 3D back-projection of that particular 2D medial axis node and the radius for the corresponding inscribed sphere is assigned to that maximum DT_3 value. Such procedure guarantees that all the estimated spheres of curve-skeleton approximation will be strictly inside the visual hull.

There's no need to calculate DT_3 values for all inner voxels of visual hull. The DT_3 value for a particular voxel is calculated on-the-fly and cached for future use. It turns out that in average almost 50% of voxels are never visited, thus reducing the computation time.

Fig. 4 shows an example of curve-skeleton approximation point cloud for the same elephant model and a set of inscribed spheres.

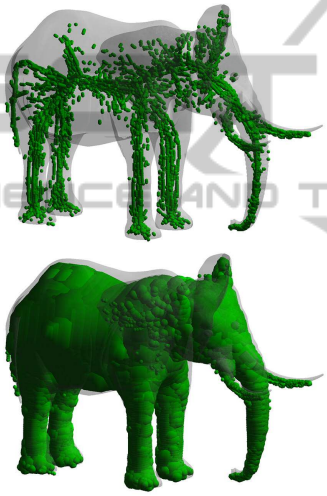

Figure 4: Top to bottom: curve-skeleton approximation point cloud and corresponding set of inscribed spheres.

Occlusions lead to spurious bones in medial axes of silhouettes (Fig. 5). These bones don't depict object's geometry and are not centered inside of the model, and can even fall outside of the model after back-projection. We suggest to delete bones that have a large depth difference between connected nodes compared to the bone's length on the 2D medial axis, i.e. we delete bones that have small angle ($\leq \phi_0$) with corresponding camera view direction. This approach might delete relevant curves, but due to usage of the set of projections this curve will likely be captured from another projection. The example of filtering result is shown in Fig. 6.

2.1.2 Alignment of Skeleton Nodes and Bones from Different Projections

Even without occlusions same part of the object on different projections can be back-projected into centered, but quite different curves (Fig. 7). We suggest

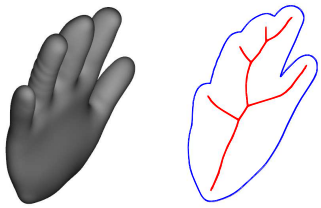

Figure 5: Projection with occlusions and its medial axis. Medial axis doesn't represent object geometry well due to occlusions.

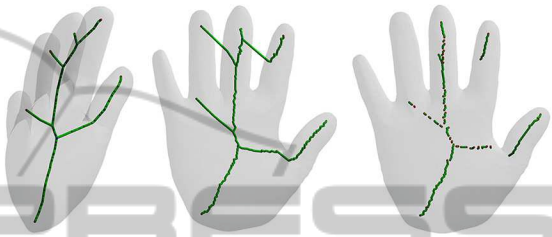

Figure 6: Silhouette medial axis, rotated back-projected medial axis, back-projected medial axis with filtered bones.

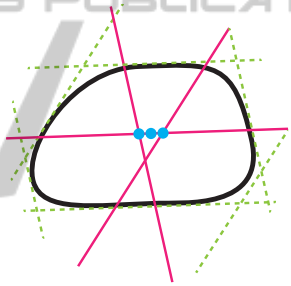

Figure 7: Inconsistent back-projecting throughout different projections due to elliptic shape.

to organize an alignment process to join nodes and bones from different projections and to remove such noise.

At this stage nodes and bones are centered in the model and look like a very noisy curve-skeleton. The basic idea behind alignment is noise canceling by finding resulting nodes with the highest density of neighbors. For this purpose Mean-Shift algorithm (Huang et al., 2013) is used , that iteratively moves every node from starting position to the average position of its neighbors. When the node stops to significantly change its position, the node is replaced with the final average. The algorithm doesn't take into account the skeleton bones, that are automatically shifted with adjacent nodes.

The following algorithm describes this approach:

Require: X — input nodes, ε — minimum shift threshold, α — minimum density of neighbors threshold, $K(x)$ — kernel function;

Ensure: Y — shifted nodes;
1: $Y := \varnothing$;
2: **for** $x_i \in X$ **do**
3: $m := x_i$; {start Mean-Shift from x_i}
4: **repeat**
5: $m_{old} := m$;
6: $m := \frac{\sum_{x_j \in X} K(x_j - m_{old}) x_j}{\sum_{x_j \in X} K(x_j - m_{old})}$;
7: **until** $\|m - m_{old}\| \geq \varepsilon$
8: **if** $\sum_{x_j \in X} K(x_j - m) > \alpha$ **then**
9: $Y := Y \cup \{m\}$;
10: **end if**
11: **end for**

The alignment algorithm is illustrated in case of points on plane in Fig. 8. There's an obvious drawback of such approach: the ending points of the line travel too far. We suggest to use a heuristic that forbids ending points of medial axis (which is a graph) to travel too far.

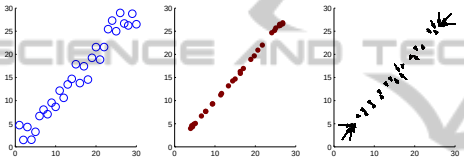

Figure 8: The alignment algorithm for points on plane. From left to right: set of points, aligned points, travel paths.

The result of such alignment is shown in Fig. 9. The result is much cleaner and looks like a good curve-skeleton approximation.

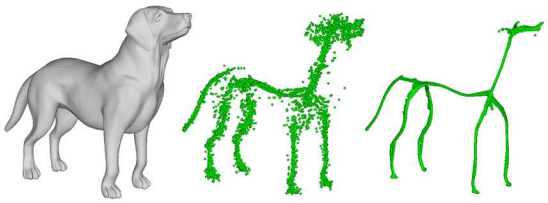

Figure 9: An object, back-projecting and filtering result, aligned nodes and bones.

2.2 Iterative Contraction

We suggest to implement object contraction through spheres of curve-skeleton approximation radii reduction. Point cloud from section 2.1 approximates curve-skeleton and thus is centered and describes shape quite well, hence spheres radii reduction leads to visually correct object contraction (Fig. 10).

Contraction makes the object thinner, which helps to get rid of occlusions, thus making point cloud approximation better through iterations. Fig. 11 illustrates 4 iterations of object contraction. We can see

Figure 10: Object contraction through spheres radii reduction.

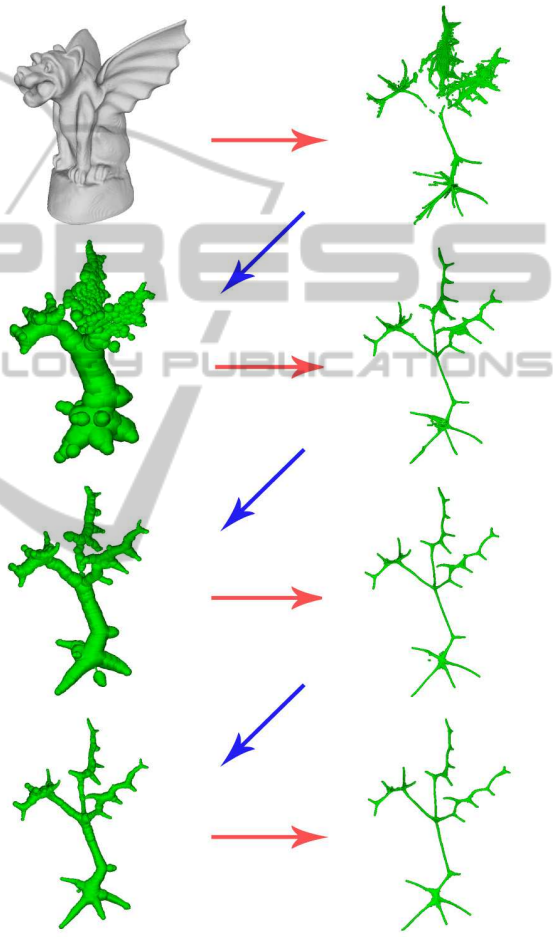

Figure 11: 4 iterations of object contraction. Red arrows depict curve-skeleton approximation with point cloud, blue arrows depict object contraction based on approximation.

that point cloud approximation of curve-skeleton is sufficient for object contraction.

3 EXPERIMENTS

Experiments were conducted on the computer with Core i7 2.2 GHz CPU, 16 GB RAM, Intel HD Graphics 5200 GPU. For continuous skeleton extraction li-

brary (Mestetskiy and Semenov, 2008) was used. 2D distance transform is done in linear time with the method described in (Felzenszwalb and Huttenlocher, 2004).

Method used 33 projections with resolution 300×300. Bones filter was used with $\phi_0 = 45°$. Alignment of nodes and bones was used with ε equal to the height of one pixel in model coordinates, α equal to 5%–10% percentile, kernel function $K(x) = \exp(-C\|x\|^2)$ with $C = 100$.

Table 1 shows timings in milliseconds for different stages of algorithm for different models. Due to the fact that method doesn't deal with polygons in space, model complexity (number of polygons) has little effect on extraction time.

Table 1: Timings in milliseconds for different stages of algorithm.

Model	Faces in model	Medial axes	Back-project	Mean-Shift	Total
Fertility	50,000	44	1,229	496	1,769
Memento	52,550	62	1,426	1,056	2,544
Elk	48,026	61	1,754	1,130	2,954

Fig. 12 shows curve-skeletons for different models. The resulting curve-skeletons depict models' geometry and topology well.

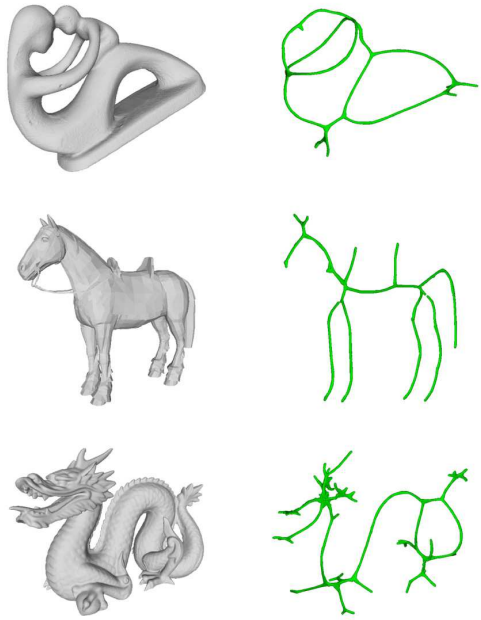

Figure 12: Examples of curve-skeletons extracted with described method for Fertility, Horse and Dragon models.

4 METHOD ANALYSIS

Resulting curve-skeletons conform with most requirements to universal curve-skeletons based on research (Cornea et al., 2005). This is achieved using filters, mean-shift and inherited properties of medial axes like centeredness and isometric invariance.

Compared to (Au et al., 2008) method shows 6 times increase in extraction speed (in average 3 sec. vs 19 sec. for different models), though it's hard to compare the quality of resulting skeletons due to poor formalization of the problem (Dey and Sun, 2006). Visual comparison of resulting skeletons to the ones extracted by (Au et al., 2008) shows that the proposed method yields similar skeletons to the state-of-the-art method (Fig. 13).

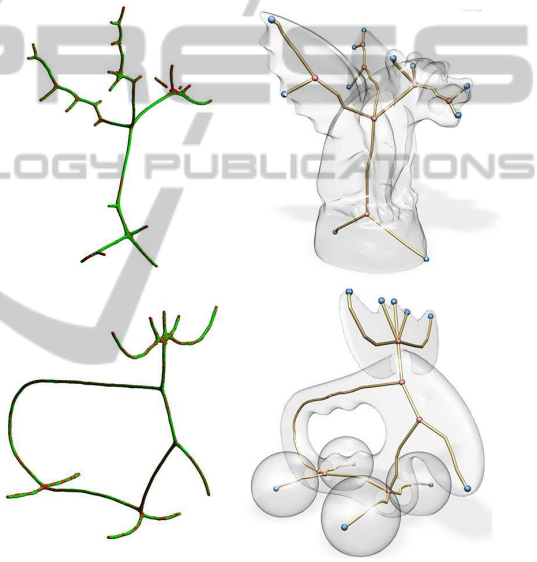

Figure 13: Visual comparison of resulting skeletons (left) for Gargoyle and Elk models to the ones extracted by (Au et al., 2008) (right).

Method looks ideal for parallel implementation as most stages can be done on GPU, which promises a huge decrease in computational time and possibly new real-time applications for curve-skeletons.

Method fails to extract curve-skeletons for heavily occluded objects with parts not visible without occlusions (Fig. 14). Fortunately, most applications of curve-skeletons assume that we have a character-like object with significant visual branches.

5 CONCLUSION

A new method of curve-skeleton extraction is proposed in this paper. It uses a visual hull approximation

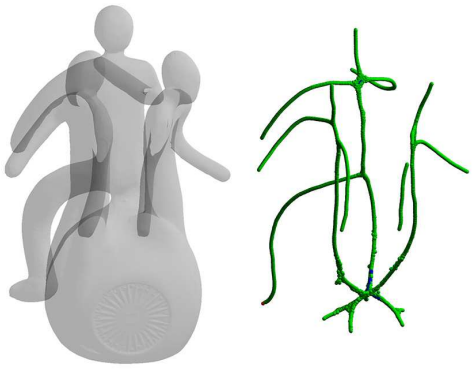

Figure 14: The method breaks topology for Memento model.

and inherently can operate on any kind of model representation which allows rasterization. The usage of continuous medial axes reduces computational time and gives an ability to analyze medial axis as a graph. Described method shows good extraction results for models with complex geometry and topology.

In the future we'll try to eliminate the iterative nature of the algorithm as stopping criteria is not clear and time consumption can be drastically reduced. We'll explore applications for the method to find real world limitations of visual hull approximation. We want to design procedures providing some theoretical guarantees like homotopy of the resulting skeletons. We see a great potential in more sophisticated analysis of graph structure of silhouettes' medial axes, in the current method it's mostly neglected.

ACKNOWLEDGEMENTS

Authors thank the Russian Foundation for Basic Research for the support on this study, research projects 14-01-00716, 14-07-00965 and 12-07-92695-IND.

REFERENCES

Au, O. K.-C., Tai, C.-L., Chu, H.-K., Cohen-Or, D., and Lee, T.-Y. (2008). Skeleton extraction by mesh contraction. In *ACM SIGGRAPH 2008 papers*, SIGGRAPH '08, pages 44:1–44:10, New York, NY, USA. ACM.

Cornea, N. D., Silver, D., and Min, P. (2005). Curve-skeleton applications. In *IEEE Visualization*, page 13. IEEE Computer Society.

Dey, T. K. and Sun, J. (2006). Defining and computing curve-skeletons with medial geodesic function. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 143–152, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science.

Huang, H., Wu, S., Cohen-Or, D., Gong, M., Zhang, H., Li, G., and Chen, B. (2013). L1-medial skeleton of point cloud. *ACM Trans. Graph.*, 32(4):65:1–65:8.

Kustra, J., Jalba, A., and Telea, A. (2013). Probabilistic view-based 3d curve skeleton computation on the gpu. In *VISAPP (2)'13*, pages 237–246.

Laurentini, A. (1994). The visual hull concept for silhouette-based image understanding. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(2):150–162.

Livesu, M., Guggeri, F., and Scateni, R. (2012). Reconstructing the curve-skeletons of 3d shapes using the visual hull. *IEEE Trans. Vis. Comput. Graph.*, 18(11):1891–1901.

Mestetskiy, L. and Semenov, A. (2008). Binary image skeleton - continuous approach. In Ranchordas, A. and Araújo, H., editors, *VISAPP (1)*, pages 251–258. INSTICC - Institute for Systems and Technologies of Information, Control and Communication.

Mestetskiy, L. and Tsiskaridze, A. (2009). Spatial reconstruction of locally symmetric objects based on stereo mate images. In Ranchordas, A. and Araújo, H., editors, *VISAPP (1)*, pages 443–448. INSTICC Press.

Sobiecki, A., Jalba, A., and Telea, A. (2014). Comparison of curve and surface skeletonization methods for voxel shapes. *Pattern Recogn. Lett.*, 47:147–156.

Sobiecki, A., Yasan, H. C., Jalba, A. C., and Telea, A. (2013). Qualitative comparison of contraction-based curve skeletonization methods. In *ISMM*, pages 425–439.