3D Skeletonization using an Enhanced Voxel Tree[1]

Xing-Dong Yang and Irene Cheng, Department of CS, UNIV. OF ALBERTA, EDMONTON, CANADA
CONTACT: LIN@CS.UALBERTA.CA, XINGDONG@CS.UALBERTA.CA

I. INTRODUCTION

Although mesh mapped with texture is observed by viewers, skeleton is often used as a compact and simplified form to represent a complex 3D object for backend processing, *e.g.* animation, similarity match from database, shape comparison and perceptual evaluation [ZY96][CGC*02][HK00], because the original object is too computationally expensive to analyze given time constraints and limited resources. Many skeletonization algorithms have been discussed in the literature including the medial axis and thinning approaches [BO04][PSB01]. However, these algorithms may not generate a stable skeleton. These techniques either focus on a volume, *e.g.* medical DICOM data, or mesh data. Our skeletonization method using voxels is stable and can be used for both mesh and volume data. Figure 1 shows (a) the original 3D horse model, (b) model covered with voxels on the surface after three iterations, and (c) the tightly bound voxels after running six iterations of our algorithm, giving a good approximation of the horse.

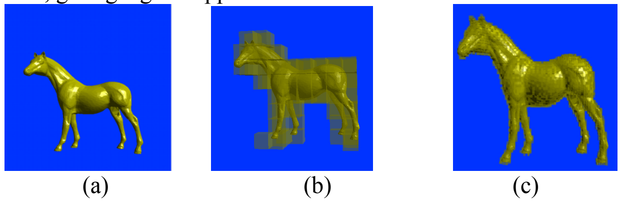

Figure 1: A horse model: (a) original, (b) covered with voxels on the surface after three iterations and (c) after six iterations.

II. ENHANCED VOXEL TREE CONSTRUCTION

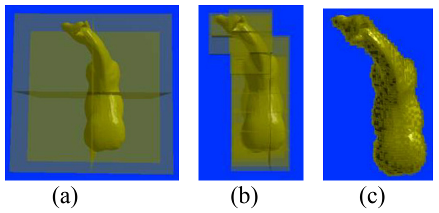

Figure 2: Top view of the horse model: (a) Dividing into eight sub-cubes (1st iteration). (b) 3rd iteration, and (c) 6th iteration.

Our approach is similar to sphere tree construction [BO02], but we replace the spheres with cubes (voxels) and enhance the original algorithm by generating more tightly bound voxels around the surface of the mesh. An octree structure is used in the current implementation. The idea is to generate cubes sufficiently small to cover the mesh surface obtaining a good approximation of the shape. The motivation is that using this approximation of the surface, we can locate the medial axis of the 3D model based on the regular geometry and placement of the cubes, and use the cube centers to compute a skeleton. Our algorithm consists of the following steps:

1) Construct the smallest bounding cube that contains the object. The bounding cube is aligned with the x, y and z axes and becomes the root node of the voxel tree structure.
2) Replace the bounding cube with eight sub-cubes by dividing each dimension by half (Figure 2 (a)). Empty sub-cubes that do not contain the surface of the object are then discarded. The remaining sub-cubes are added to the tree as children.
3) Perform the operation recursively on each child until a certain depth is reached.
4) Push the cubes from outside towards the object boundary. The goal is to achieve as much of the cube volume inside the object as possible, generating a good representation of the shape.

Note that the cubes resulting from step (3) are too regularly stacked and do not follow the silhouettes of the object properly. Thus, Step (4) is introduced to improve the approximation.

III. SKELETON GENERATION

Although the shape approximated by the cubes is satisfactory, the cube centers do not reside inside the object. Connecting their centers generates silhouettes but not a skeleton. This problem can be solved by separating the object into geometric parts and taking a slice from each part.

For the horse model, we perform six iterations to generate a good approximation of the silhouettes *S* composed of cubes. Since these cubes are generated based on the x, y and z axes directions, it is easy to align *S* with the x-y, y-z and x-z planes. By intersecting *S* in the middle with a plane parallel to the x-y coordinates, we get a slice of the horse. The following steps are used to generate the skeleton:

1) To ensure a good match between the skeleton and the shape of the object, we segment the horse into parts, *e.g.* body, legs and head, so that each part is aligned with a plane separately.
2) Once a slice is extracted for each part, we apply a 2D version of the Voronoi algorithm [BKO*00] to extract the medial axis.
3) The medial axes from different parts are then integrated to form the final skeleton. The medial axes can be connected or disconnected depending on the application.

For the horse model with 10,000 faces, the execution time is two seconds and the result looks promising (Figure 3). To extend our approach to volume data, e.g. DICOM, we discard the voxels inside the object and generate a skeleton based on the surface voxels.

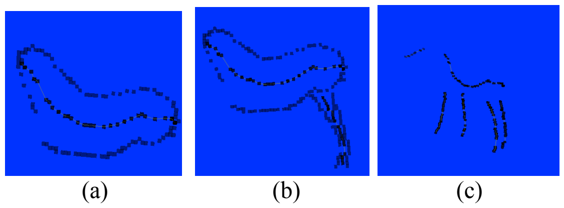

Figure 3: (a) Medial axis of the horse body; (b) Medial axes of the horse body and a leg; (c) Skeleton of the horse.

IV. LIMITATION AND FUTURE WORK

The current implementation is semi-automatic because it relies on known orientation of the model. If there is an angle between a part and the coordinate axes, the part is rotated to align properly with the coordinate planes. In the future, we will incorporate some algorithms to detect the orientation of each part and apply quadric planes instead of rectangular planes to improve the approximation.

REFERENCES

[BKO*00] M. Berg, M. Kreveld, M. Overmars and O. Schwarzkopf, "Computational Geometry," *Springer*, 2nd edition, pp 147-163.
[BO02] G. Bradshaw and C. O'Sullivan, "Sphere-Tree Construction using Medial-Axis Approximation," In *Proc. of SIGGRAPH* 2002.
[BO04] G. Bradshaw and C. O'Sullivan, "Adaptive Medial-axis Approximation for Sphere-Tree Construction," *ACM Transaction on Graphics*, 23(1) Jan 2004.
[CGC*02] S. Capell, S. Green, B. Curless, T. Duchamp and Z. Popovic, "Interactive Skeleton-Driven Dynamic Deformations," In *Proc. of SIGGRAPH* 2002.
[HK00] C. Holleman and L. Kavraki, "A Framework for using the Workspace Medial Axis in prm Planners," In *Proc. of the 2000 IEEE Int. Conf. on Robotics and Automation*, Vol.2, pp1408-1412, Apr 2000.
[PSB*01] K. Palagyi, E. Sorantin, E. Balogh, et al., "A Sequential 3D Thinning Algorithm and Its Medical Applications, " *Springer-Verlag Berlin Heidelberg* IPMI 2001, pp.409-415 .
[ZY96] S. Zhu and A. Yuille, "Form: A Flexible Object Recognition and Modeling System," *Int Journal of Computer Vision*, 20(3):187-212, 1996.

[1] The support of ASRA, iCORE and NSERC for Irene Cheng is gratefully acknowledged.