2013 13th International Conference on Computer-Aided Design and Computer Graphics

Direct Extraction of Feature Curves from Volume Image for Illustration and Vectorization based on 2D/3D Curve Mapping

Liping Wang*, Lili Wang*, Fei Hou*[1], Aimin Hao*, Hong Qin†
*Beihang University, †Stony Brook University (SUNY Stony Brook)

{wanglp, lily_wang, houfei}@vrlab.buaa.edu.cn, ham_buaa@163.com, qin@cs.sunysb.edu

I. Introduction and Motivation

This paper proposes a parallel and direct semantic feature curve extraction method from 3D volume image for vectorization and illustration. Our approach is motivated by reconstructing 3D geometric information from multiple rendered images under multi-view in computer vision [1]. The 2D rendered images are rich in the visual sense by color and opacity that convey the structure of volume data, so it is significant for the user to understand the structure of 3D volume data better if we can recover feature curves from those 2D images. Compared with conventional line extraction methods, which mainly focus on extracting feature curves from iso-surfaces in object space, we extract feature curves directly from volume images. Most of the computation can be computed in parallel on GPU with CUDA acceleration.

II. Method and Algorithm

2D Feature Curve Extraction. We apply GPU raycasting techniques to render volume data with six cameras. We consider that the rendered images express much visual perception that helps the user get the general information about the shape and structure of volume data. Then we use the Canny operator [3] to obtain 2D feature curves.

Mapping to 3D Feature Curves. We map these 2D features to 3D features via an energy minimization formulation and use dynamic programming to solve it on GPU. A 2D curve l_s consists of n 2D points s_i, $1 \le i \le n$. We set k 3D candidate feature points d_{i_j}, $1 \le j \le k$ with high opacity for s_i. We define the following energy:

$$E = \sum_{i=1}^{n} f(d_i) + \sum_{i=2}^{n} g(d_{i-1}, d_i), \quad d_i \in d_{i_j}, 1 \le j \le k. \tag{1}$$

The 3D feature point with high opacity α should be considered preferentially

$$f(d_i) = 1 - \alpha_{d_i}, \qquad 0 \le \alpha \le 1. \tag{2}$$

[1]corresponding author

And the difference between adjacent points with properties of position $\mathbf{p}$, and color $\mathbf{c}$ should be as small as possible.

$$g(d_i, d_j) = k_1 ||\mathbf{c}_{d_i} - \mathbf{c}_{d_j}|| + k_2 ||\mathbf{p}_{d_i} - \mathbf{p}_{d_j}||, \tag{3}$$

where k_1 and k_2 are the weights. We consider that E is the cost or energy of a 3D feature curve, hence the back projection is formulated as an energy minimization problem.

Clustering and Fitting Candidate Curves. We construct a scalar field called *curve density field*, which is implicitly defined by candidate curves, and a potential function representing the contribution of each 3D point to the scalar field. This idea is inspired by convolution surface [4]. The field is shown in Fig. 1, from which we observe that the inner part of the field is more often with high value, because more candidate curves make contribution to the inner region, especially at places where many curves join together.

We cluster the points into groups and trace the groups according to the estimated direction to obtain the geometric representation of volume data, our formulation is devised based on the least squares sense and can be solved iteratively. We are inspired by Li's work [2] who introduced *Cross Section Planes* for computation. We then trace the connectivity of the groups to get curve sets in the field domain according to the position $\mathbf{t}_i$ based on their spatial proximity. Furthermore, for the sake of better connectivity and smoothness, we use Bézier spline to fit them. The results are shown in Fig. 2.

III. Results

Shown in Fig. 3, some results are produced from several data sets. We can see that the vectorization results with Bézier splines are scalable, smooth and compact.

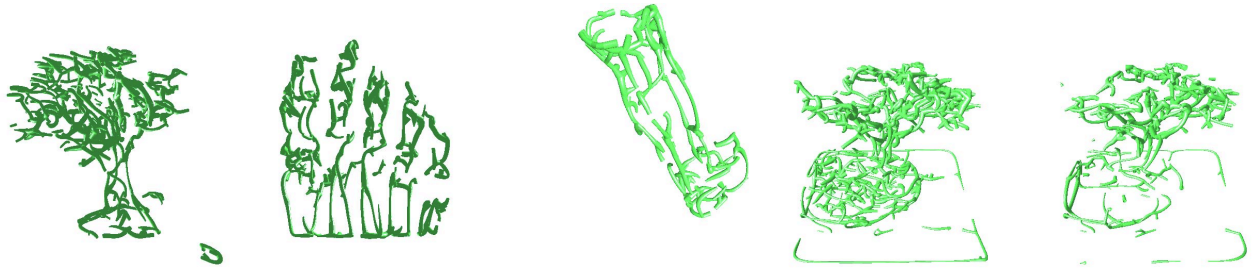
The user also can control different levels of details for volume illustration manually. We can set different parameters to control the number of 2D feature curves, and it is related to the number of extracted 3D feature curves. But even though with very few features, our algorithm can work well because the point/curve features indicate stronger response to the structure information of volume data.

All feature curves are computed offline. But the main parts of the algorithm run on GPU with CUDA acceleration, so most steps take very short time. Tab. I shows timings for the

978-1-4799-2576-6/13 $31.00 © 2013 IEEE
DOI 10.1109/CADGraphics.2013.67

Model	Size	2D Curve Extraction	3D Curve Extraction	Field Construction	Cluster	Vectorization	Number of Curves
Vertebra	$512 \times 512 \times 512$	0.607	0.331	0.902	0.924	1.172	373
Bonsai	$256 \times 256 \times 128$	0.817	0.739	2.729	1.661	7.627	848
Foot	$256 \times 256 \times 256$	1.195	0.905	5.762	2.771	3.511	1144
Leg	$341 \times 341 \times 93$	0.553	1.018	1.696	1.349	3.493	271

Table I: Running time for different models (in second).



(a) Bonsai 256*256*128 (b) Foot 512*512*512 (c) Leg 314*314*93 (d) Bonsai 256*256*256 (e) Bonsai 256*256*256

Figure 3: Results from several data sets in (a-c), and multi-level vectorization results in (d-e).

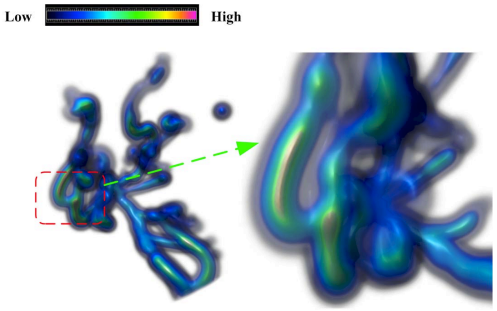


Figure 1: Curve density field. We map the scalar value of field to color for visualization. The inner part of data is with high scalar value (indicated by red color) especially at junctions of branches.

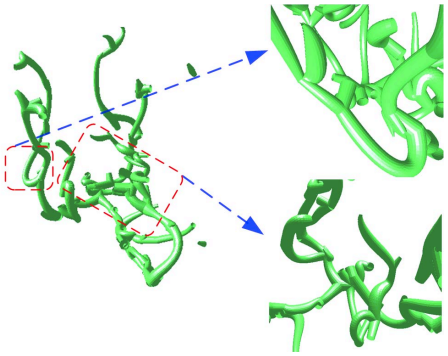


Figure 2: Feature curve fitting and vectorization base on the clustering results.

intermediate steps of several models. The table documents most of the operations in our algorithm. Our method directly extracts feature curves from volume data, so the timing is mainly dictated by the complexity of structure of the data. In particular, it depends on the numbers of both feature curves extracted in 2D image and candidate 3D feature curves.

IV. Conclusion

We have developed a novel feature curve extraction and vectorization algorithm for volume data illustration. In contrast to most of traditional methods that function over iso-surfaces extracted from datasets, we directly extract feature curves from data. We make the best use of highly-efficient GPU and run most parts of our new algorithm on GPU in parallel with CUDA acceleration.

Acknowledgment. This work is supported by National Natural Science Foundation of China (No.61190120, 61190121, 61190125, 61300067, and 61300068) and National Science Foundation of USA (No.IIS-0949467, IIS-1047715, and IIS-1049448).

References

[1] F. Calakli, A. O. Ulusoy, M. I. Restrepo, G. Taubin, and J. L. Mudy, *High Resolution Surface Reconstruction from Multi-view Aerial Imagery*. in Proceedings of 3DIMPVT'12, IEEE Computer Society, pp. 25-32, 2012.

[2] G. Li, L. Liu, H. Zheng, and N. J. Mitra, *Analysis, reconstruction and manipulation using arterial snakes*. ACM Transactions on Graphics, vol. 29, no. 6, article 152, 2010.

[3] J. Canny, *A Computational Approach to Edge Detection*. IEEE Transactions on PAMI, vol. 8, no. 6, pp. 679-698, 1986.

[4] J. Bloomenthal, and K. Shoemake, *Convolution Surfaces*. in Proceedings of SIGGRAPH'91, pp. 251-256, 1991.