ROI-emphasized Volume Visualization guided by Anisotropic Structure Tensor

Wei Xie[1] Fei Hou[1] Shuai Li[1*] Aimin Hao[1] Hong Qin[2]
[1]Beihang University, lishuai@buaa.edu.cn [2]Stony Brook University

Most of Focus+Context visualization methods, such as [4], differentiate the magnification unit only by simply assigning each voxel/cell with an importance value while ignoring the shape content embedded in the volume data. In this paper, we take the volumetric structure information as important cue to facilitate Focus+Context visualization, which can homogeneously or non-homogeneously scale the volume data in a structure-sensitive way.

I. Method

A. Local Feature Specification. Given the volume data and the transfer function, we first filter out the voxels whose opacity are less than 0.1. Then we compute the structure tensor on the remaining voxels, and denote them as $R_i = \{p_i, n_i\}$, where p_i is the position of i-th voxel and n_i is the eigen-vector corresponding to the largest eigen-value of the structure tensor. Inspired by [1], we choose a subset of voxels Γ close to the i-th voxel as the cutting plane voxel set. We minimize $E = min \sum_{p_i \in \Gamma, \|v\|=1} (n_i \cdot v)^2$ to find a vector v that is perpendicular to all the gradient directions in Γ as much as possible. We use v as the normal vector of the cutting plane, and then re-estimate voxels close to the current cutting plane, and alternate between the two steps until the iteration converges.

B. Adaptive Tetrahedron Mesh. We establish a tetrahedral grid over the volumetric data, and formulate the magnification problem as a mesh deformation problem. The importance value of each tetrahedron is determined by summing the interior voxel's opacity, which is further divided by the tetrahedron volume to normalize itself to fit into [0,1]. The weight of a point is assigned as the average weight of its adjacent tetrahedra. Each line's weight is set as the average weight of the two endpoints. Each time, we choose the line with minimal weight and merge the two endpoints into one to collapse the edge and tetrahedron. If the merged result leads to self-intersection, we re-select a line to merge. After decimating the tetrahedral grid to a user-specified resolution, we average the direction of all the inner points as the direction of the tetrahedron.

C. Deformation Energy. We denote the input and output tetrahedron sets as T^I and T^O respectively. Each tetrahedron will be assigned a scaling factor as $S_i = \begin{pmatrix} s_x^i & 0 & 0 \\ 0 & s_y^i & 0 \\ 0 & 0 & s_z^i \end{pmatrix}$. Since the display space is limited, some of the tetrahedra may be rotated or sheared. We denote the expected deformation of the i-th tetrahedron as $R_i S_i$, where R_i is the rotation component.

Each input tetrahedron T_i^I can be deformed to T_i^O by a 4x4 matrix D_i. The linear portion of D_i is a 3x3 Jacobian matrix J_i involving scaling, rotation, and shearing transformations. Because we sample the initial position of control points on the grid, the elements of matrix J_i are linearly dependent on the vertices in T_i^O. Thus, we can define the deformation energy as $E_d = \sum_{i \in T} \omega(i) \parallel J_i - R_i S_i \parallel_F^2$, where $\| \cdot \|_F$ is the Frobenius norm, and $\omega(i)$ is the weight of i-th tetrahedron. Introduced in [2], this energy function can be rewritten in terms of the coordinates of the original grid and the deformed grid vertices [3].

The system initially sets R_i and the context regions as identity matrices, and gets ROIs scaling factor S_i from user specification. After that, we compute J_i, which is then decomposed into a rotation matrix R_i^J and a scaling matrix S_i^J using polar decomposition. We set R_i to R_i^J. For the context regions, we set S_i to the average of the diagonal elements in S_i^J. For ROIs, we do not change the scaling factor S_i. Then we iteratively compute J_i, and use J_i to compute R_i and S_i until E_d is convergent.

D. Smoothness Energy. To increase the smoothness of results, we limit the deformation between adjacent tetrahedra by $E_s = \sum_{i,j \in T} \Psi(i,j) \parallel R_i S_i - R_j S_j \parallel_F^2$. Here i, j are adjacent voxels and $\Psi(i,j) = (\omega(i) + \omega(j))/2$. Thus, the total energy can be defined as the weighted sum of two energies: $E = \lambda E_d + \mu E_s$, and λ and μ are the weights.

E. Constraints. Since the size of volume data is fixed, for a control point P on the left plane of V^I, we enforce that P_x equals to zero, and for a control point Q on the right plane of V^I, we enforce that Q_x equals to the width of the volume data. The y-axis and z-axis boundary constraints are similar. Since tetrahedron flip will lead to self-intersection and produce messy results, we require that $det(R_i^J) > 0$. Otherwise, we reverse the sign of the vector corresponding to the smallest singular value of J_i, and recompute R_i^J.

By optimizing above energy functions, we can get the position of the deformed tetrahedral mesh. Then we use the trilinear interpolation to get the density of inner points of the tetrahedra which can be accelerated on GPU.

II. Experiments

In Fig. 1, we can see the ROI is magnified and other regions are compressed to retain the original volume bound-

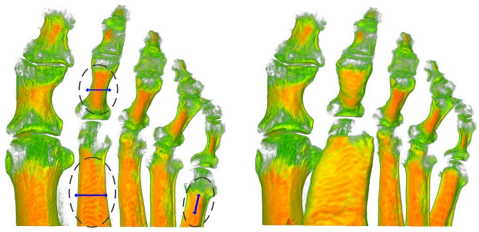
978-1-4799-2576-6/13 $31.00 © 2013 IEEE
DOI 10.1109/CADGraphics.2013.89

Table I
PERFORMANCE STATISTICS (IN SECOND).

dataset	volume size	initial tetrahedra		reduced tetrahedra		pre-computation time	deformation time
		points	tetrahedra	points	tetrahedra		
aneurism	256x256x256	4096	14576	2000	5216	230	0.12
foot	256x256x256	9261	48000	5000	29276	410	0.27
bonsai tree	512x512x189	3969	19200	3969	19200	10	0.24

(a) original grid (b) deformed grid (c) original data (d) deformed data

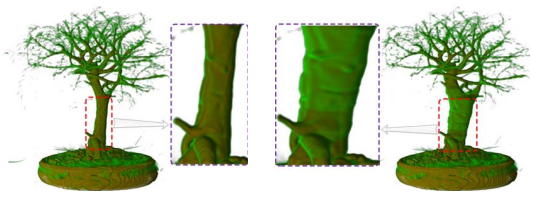
Figure 1. The original grid and the corresponding deformed grid. The grid color from blue to red indicates the weight range from 0.0 to 1.0.



(a) original foot (b) deformed foot

Figure 2. The magnified results corresponding to different scaling directions on foot dataset. The black ellipses indicate the user-specified ROIs, and the blue arrows indicate the scaling direction.

ary. Compared with existing methods, our method can not only magnify the features in ROIs, but also magnify them in an anisotropic way. For the tubular-shaped objects, we can magnify them along or perpendicular to the gradient direction. For volumetric data having explicit local directions, our system can set different scaling factors along different tensor directions. As shown in Fig. 2, with user-specified ROIs, our system can coarsen the toes while keeping their length (or elongate the toes while keeping their diameters). Our method also preserves the structure well and automatically excludes self-intersection when simultaneously magnifying multiple ROIs with large scaling factors. By zooming the magnified



(a) original bonsai dataset (b) magnified result of (a)

Figure 3. The magnified result of user-specified ROIs on bonsai dataset.

regions in Fig. 3, it shows that our method can clearly magnify the local features and well preserve the details. Besides, compared with the cubic grid [4], tetrahedral grids are more flexible when being manipulated. We can clearly see how the local feature changes and avoid edge flip with the help of Jacobian matrix.

To evaluate the efficiency of our method, we document the timing performance in Tab. I. The local structure tensor computation is done in pre-processing stage, whose time consumption mainly depends on volumetric data size, and the scale of sampling points in volume rendering. In our experiments, we always use $600 \times 600 \times 500$ sampled points for ray-casting based volume rendering, however, the time cost in pre-computation is relatively little.

III. CONCLUSION

We have presented a framework for interactive anisotropic magnification of volumetric data. The fundamental idea is the integrated strategy of local anisotropic tensor measurement and global deformation over 3D grids. In this way, we are able to transform a visualization problem of magnification/shrinking to the space deformation problem. Our experimental results demonstrate that the anisotropic magnification is both meaningful and works extremely well.

Acknowledgment: This research 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] G. Li, L. Liu, H. Zheng, N. J. Mitra, Analysis, reconstruction and manipulation using arterial snakes, ACM Trans. On Graph. vol.29, no.5, pp. 1-10, 2010.

[2] U. Pinkall, S. D. Juni, K. Polthier, Computing discrete minimal surfaces and their conjugates, Experimental Mathematics, vol.2, no.1, pp. 15-36, 1993.

[3] L. Liu, L. Zhang, Y. Xu, C. Gotsman, S. J. Gortler, A local/global approach to mesh parameterization, Computer Graphics Forum, vol.27, no.5, pp. 1495-1504, 2008.

[4] Y.-S. Wang, C. Wang, T.-Y. Lee, K.-L. Ma, Feature-preserving volume data reduction and focus+context visualization, IEEE Trans. on Vis. and Comp. Graph. vol.17, no.2, pp. 171-181, 2011.