어파인 불변성 사면체 분할법의 가시화
(절편 법을 이용한 사면체 구조의 가시화)

이 건†

요 약

Dirichlet tessellation 과 쌍대관계에 있는 Delaunay triangulation은 어파인 불변성을 가지지 못한다. 즉, 삼각형 분할을 이루는데 있어서 각 꼭지점들을 나타내는 좌표축의 선택에 영향을 받는다. 같은 이유로 Delaunay tetrahedrization(사면체 분할법) 도 어파인 불변성을 가지지 못한다. 본 논문에서는 공간상 점들로 사면체 분할하는데 있어서 변환, 확대 축소, 일그러뜨림, 회전에도 영향을 받지 않는 새로운 유형의 사면체 분할 방법을 제시하였다. 어파인 사면체 분할을 논의 할 때 기존의 어파인 불변성 평면적 삼각형 분할을 삼차원적 사면체 분할로 연장시키는 방법을 사용하였다. 삼차원 공간상의 두 점간의 거리를 새롭게 정의하였다. 사면체 구조의 가시화를 통하여 Delaunay 사면체 분할과 어파인 불변성 사면체 분할 결과를 구별시 킬 수 있었다.

Visualization of Affine Invariant Tetrahedrization
(Slice-Based Method for Visualizing the Structure of Tetrahedrization)

Kun Lee†

ABSTRACT

Delauuany triangulation which is the dual of Dirichlet tessellation is not affine invariant. In other words, the triangulation is dependent upon the choice of the coordinate axes used to represent the vertices. In the same reason, Delaunay tetrahedrization does not have an affine ivecariant transformation property. In this paper, we present a new type of tetrahedrization of spacial points sets which is unaffected by translations, scalings, shearings and rotations. An affine invariant tetrahedrization is discussed as a means of affine invariant 2-D triangulation extended to three-dimensional tetrahedrization. A new associate norm between two points in 3-D space is defined. The visualization of the structure of tetrahedrization can discriminate between Delaunay tetrahedrization and affine invariant tetrahedrization.

1. Introduction

Tetrahedrizations have many applications related to multivariate approximations and scientific computing; including the finite element method and scattered data interpolation. In this paper, we emphasis on a certain type of optimal tetrahedrization of the convex hull of a set of data points which is invariant under affine transformations in 3-D space. It means that the way of tetrahedrization is not changed by scaling, rotation, shearing, and translation. The above property not enjoyed by other tetrahedrization. In many applications, the choice of the location of the coordi-

† 정 회 원:전북산업대학교 컴퓨터공학과
논문접수:1996년 4월 2일, 심사완료:1996년 8월 16일

nate axis or the choice of the units for measured data is rather arbitrary. These choices should not affect the final results. Unfortunately, a change of scale can affect matters (i.e. in case of Delaunay triangulation). We describe a criterion for tetrahedrization that is unaffected by this arbitrary choice. An affine invariant tetrahedrization is discussed as means of affine invariant two-dimensional triangulation extended to three-dimensional tetrahedrization. The problem of tetrahedrizing spacial data sets requires the partitioning of the convex hull into a collection of tetrahedra with vertices from the data set. we assume that the vertices are distinct and that they do not all locate on a planar.

Ω is the convex hull of tetrahedra which do not intersect each other. This definition is stated in detail (see Figure 1).

Let $\Omega \supset V$ be a volume with a tetrahedral boundary $\partial\Omega$ in which all vertices are in V. TH is a set of tetrahedra. In other words, a set $TH=\{TH_i\}_1^t$ of non-degenerate, open tetrahedra is a tetrahedrization of Ω the following conditions hold:

a) V is the set of all vertices of tetrahedra in TH,

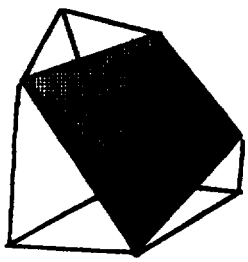
b) T is the set of all triangles of tetrahedra in Th,

c) every edge of a tetrahedron in TH contains only two points from V,

d) every face of a tetrahedron in TH contains only three points from V,

e) $\overline{\Omega}=\cup_{i=1}^{t}\overline{TH_i}$, and

f) $TH_i \cap TH_j=\phi (i \neq j)$.



(Fig. 1) Tetrahedrization

2. Two-Dimensional Affine Invariant Triangulation

G. Nielson[1][2] defines the associate norm between two points as an invariant under affine transformation as:

$$\|V\|^2 = (x, y)\frac{1}{\det}\begin{pmatrix} \Sigma_y^2 & -\Sigma xy \\ -\Sigma xy & \Sigma_x^2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}, \tag{2.1}$$

where $\det=\Sigma_x^2\ \Sigma_y^2\ -\left(\Sigma xy\right)^2$,

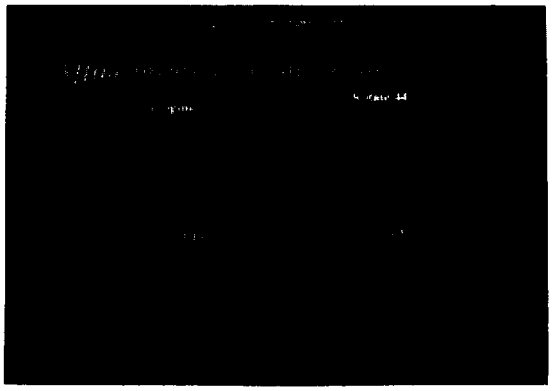
$$\Sigma_x^2=\frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{n},$$

$$\Sigma_y^2=\frac{\sum_{i=1}^{n}(y_i-\bar{y})^2}{n}, \text{ and}$$

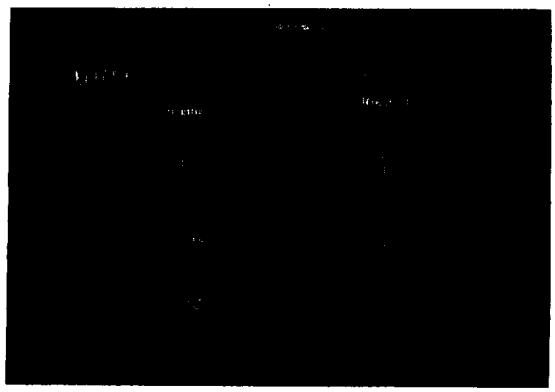
$$\Sigma\bar{x}=\frac{\sum_{i=1}^{n}x_i}{n},\quad \Sigma\bar{y}=\frac{\sum_{i=1}^{n}y_i}{n},$$

$$\text{and } \Sigma xy=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{n}.$$

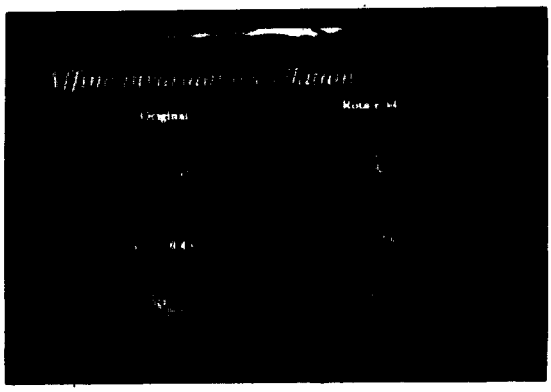
In other words, the distance between two points is independent of scaling, rotation, shearing, and translation. Therefore, the choice of units used to represent the data does not affect the triangulation. Figure 2.1 show 32 representative data points. Original data points are rotated 44° clockwise, scaled by 2 in x direction, and scaled by 0.4 in y direction. In Figure 2.2, tessellation based on affine invariant metric is displayed. We show the triangulation from the tessellations of Figure 2.2 in Figure 2.3. Figure 2.4 shows triangulations based on a standard Euclidean distance. In Figure 2.3, the method of triangulation is not changed according to an affine invariant transformation. In Figure 2.4, the method of Delaunay triangulation is changed according to an affine transformation.



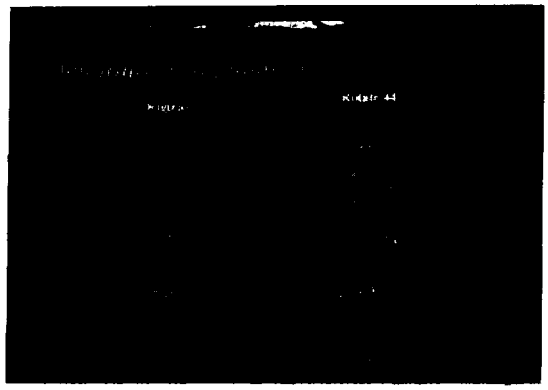
(Fig. 2.1) Affine invariant norm



(Fig. 2.3) Triangulations from the tessellations of figure 2.2



(Fig. 2.2) Tessellations based on affine invariant metric



(Fig. 2.4) Triangulations based on standard Euclidean distance

3. Affine Invariant Tetrahedrization

After Delaunay [3] provided an algorithm for triangulation of a two-dimensional plane, several researchers generalized the triangulation to n-dimensional space. Watson presents the d-dimensional Delaunay tessellation with application to Voronoi polytopes [4]. He gives an algorithm for computing the structure of the Delaunay n-dimensional triangulation for any stochastic array of data points. There are three main computational parts. The first part checks all old circumspheres against a new point. The second part calculates the new sphere; it solves d linear equations in d unknowns. In the third part, all old points are checked against a new circumsphere.

Hazlewood describes a divide and conquer approach to d-dime-nsional triangulations. She also discusses the constrained tetrahedrization problems in d-dimensionalspace [5]. She also describes an algorithm to construct a tetrahedrization that incorporates specified convex, planar, and polygonal regions, including triangles, as unions of facets of tetrahedra [6].

Edelshrunner presents an algorithm that constructs a tetrahedrization of a set P of n points by using stepwise refinement. The algorithm has two phases. First, it constructs the convex hull of P and a tetrahedrization of the hull points of P. Second, it iteratively inserts the interior points by subdividing

the tetrahedron that contains such a point into four tetrahedra [7].

Joe presents an algorithm that can construct the three-dimensional Delaunay triangulation using local transformations by starting with a special triangulation[8][9]. In Joe's paper, the sphere criterion is used for three-dimensional Delaunay triangulation. In other words, no more than four points lie inside the circumsphere in a tetrahedron. Delaunay triangulation does not have an affine invariant transformation property. In other words, the choice of units used to represent the data affects the tetrahedrization. To avoid this problem, one needs to define a new way of measuring distance that is an invariant under affine transformation (i.e. associated norm):

$$(v, v)_{A(\bar{v})} =$$

$$\|v\|^2_{A(\bar{v})} = (x, y, z) \times \left(v \cdot v^T\right)^{-1} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \qquad (3.1)$$

where,

$$v = \begin{bmatrix} x1-\bar{x} & x2-\bar{x} & x3-\bar{x} & \cdots & xn-\bar{x} \\ y1-\bar{y} & y2-\bar{y} & y3-\bar{y} & \cdots & yn-\bar{y} \\ z1-\bar{z} & z2-\bar{z} & z3-\bar{z} & \cdots & zn-\bar{z} \end{bmatrix},$$

$$\text{and } v^T = \begin{bmatrix} x_1-\bar{x} & y_1-\bar{y} & z_1-\bar{z} \\ x_2-\bar{x} & y_2-\bar{y} & z_2-\bar{z} \\ x_3-\bar{x} & y_3-\bar{y} & z_3-\bar{z} \\ \vdots & \vdots & \vdots \\ x_n-\bar{x} & y_n-\bar{y} & z_n-\bar{z} \end{bmatrix}$$

Let $M = (v \cdot v^T)^{-1}$. Then,

$$M = \frac{1}{\det} \begin{bmatrix} \left(\sum_y^2 \sum_z^2 - (\sum yz)^2\right) & -\left(\sum xy \sum_z^2 - \sum xz \sum yz\right) & \left(\sum xy \sum yz - \sum xz \sum_y^2\right) \\ -\left(\sum xy \sum_z^2 - \sum xz \sum yz\right) & \left(\sum_x^2 \sum_z^2 - (\sum xz)^2\right) & -\left(\sum_x^2 \sum yz - \sum xz \sum xy\right) \\ \left(\sum yz \sum xy - \sum_y^2 \sum xz\right) & -\left(\sum_x^2 \sum yz - \sum xy \sum xz\right) & \left(\sum_x^2 \sum_y^2 - (\sum xy)^2\right) \end{bmatrix},$$

where

$$\sum_x^2 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n},$$

$$\sum_y^2 = \frac{\sum_{i=1}^{n} (y_i - \bar{y})^2}{n},$$

$$\sum_z^2 = \frac{\sum_{i=1}^{n} (z_i - \bar{z})^2}{n}, \quad \sum \bar{x} = \frac{\sum_{i=1}^{n} x_i}{n},$$

$$\sum \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}, \quad \sum \bar{z} = \frac{\sum_{i=1}^{n} z_i}{n},$$

$$\sum xy = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{n},$$

$$\sum yz = \frac{\sum_{i=1}^{n} (y_i - \bar{y})(z_i - \bar{z})}{n},$$

$$\sum xz = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(z_i - \bar{z})}{n},$$

and

$$det = \sum_x^2 \sum_y^2 \sum_z^2 + 2(\sum xy \sum yz \sum xz) - \sum_x^2 (\sum yz)^2 - \sum_y^2 (\sum xz)^2 - \sum_z^2 (\sum xy)^2$$

Let $A = L \cdot L^T$. Then, $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$

and $L = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix}$.

where,

$$l_{11} = (a_{11})^{\frac{1}{2}},$$

$$l_{22} = \left(a_{22} - l_{21}^2\right)^{\frac{1}{2}},$$

$$l_{33} = \left(a_{33} - \left(l_{31}^2 + l_{32}^2\right)\right)^{\frac{1}{2}},$$

$$l_{21}=\frac{a_{11}}{l_{21}}\ ,\ l_{31}=\frac{a_{13}}{l_{11}}\ ,$$
$$and\ l_{32}=\frac{(a_{23}-l_{21}\cdot l_{31})}{l_{22}}\ .$$

This associated norm has the property of invariance under an affine transformation, as follows:

$$\|p-q\|_{A(\bar{v})}=\|T(p)-T(q)\|_{A(T(\bar{v}))}, \qquad (3.2)$$

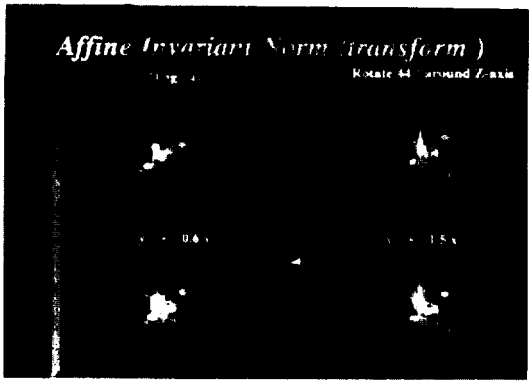
$$(v,\ v)_{A(\bar{V})}=$$

where,

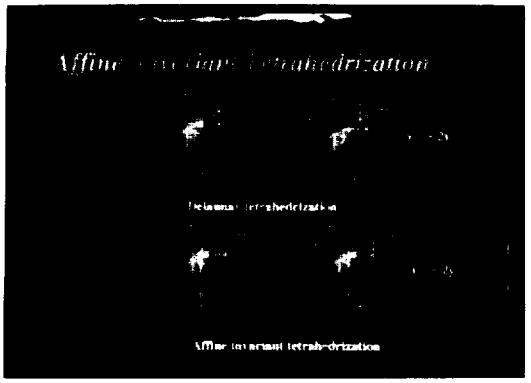
$$\|v\|^2_{A(\bar{v})}=(x,\ y,\ z)\times\left(v\cdot v^T\right)^{-1}\times\begin{bmatrix}x\\y\\z\end{bmatrix}.$$

In Figure 3.1, 13 representative data opints are displayed in 3-D space. Original data are rotated 44° in z-axis, scaled by 1.5 in x-direction, and scaled by 0.6 in y direction. Figure 3.2 shows that the transformed data are not changed when an affine transformation is performed. We choose 13 data points in 3-D space randomly. The two ellipsoids in each figure represent points which are 0.5 and 1 unit (s) from their center point while the radius of data point is 0.15 units. Figure 3.3 and Figure 3.4 present that the method of tetrahedrization based on an affine invariant norm is not changed by an affine transformation, and Delaunay tetrahedrization is changed. Specially, bottom two pictures of figure 3.3

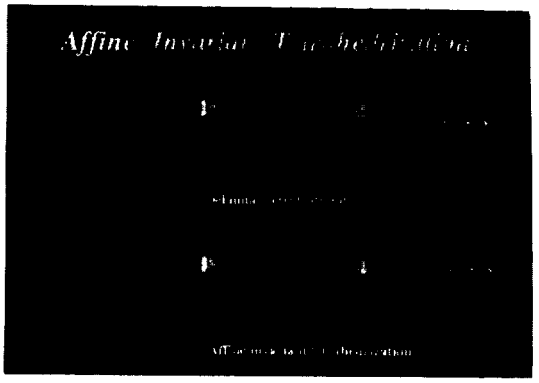
(Fig. 3.1) Representative data points



(Fig. 3.2) Transformed data points



(Fig. 3.3) Comparison between affine invariant tetrahedrization and Delaunay(case 1)



(Fig. 3.4) Comparison between affine invariant tetrahedrization and Delaunay(case 2)

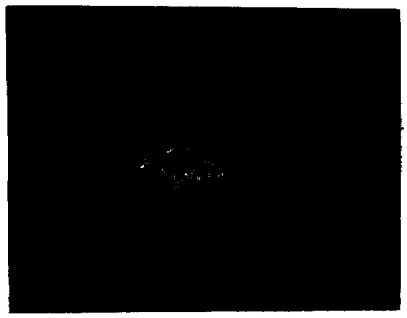
show the same number of tetrahedra through sliced section, while top two pictures show the different number of tetrahedra according to affine transformation (i.e. original data make 18 tetrahedra and affine transformed data make 13 tetrahedra).

4. Visualization of Tetrahedral Domain

Computer modeling and visualization enable us to create a mathematical model of a phenomenon that can be displayed using dynamic computer graphics. The resulting visualization yields new insights for us, and these new ways of looking at the phenomenon permit us to find trends hidden in the original data. Many applications of visualizationcan be found in Nielson and Schriver's paper[10].

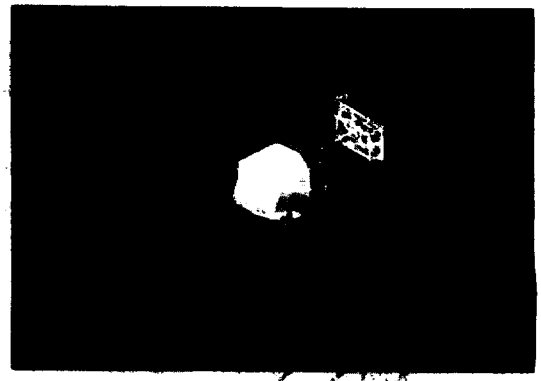
Figure 4.1 shows 125 scattered data points in 3-D space. In Figure 4.2, we show how difficult it is to understand the method of terahedrization through the wire-frame view only. In Figure 4.3, the tetrahedral domain is rendered using a visibility ordering algorithm.

(Fig. 4.1) Scattered data in 3-D space

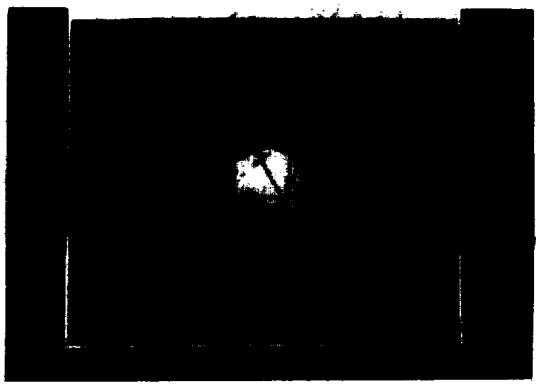


(Fig. 4.2) Wire-frame view

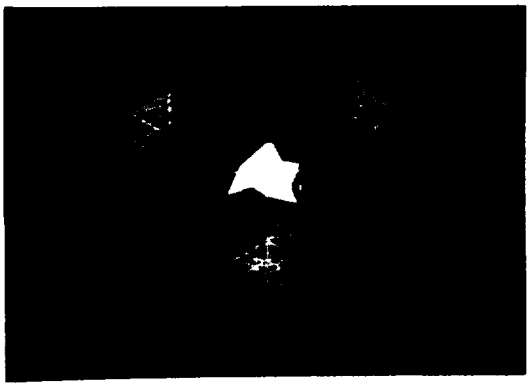
(Fig. 4.3) Rendering tetrahedra by using visibility ordering algorithm



(Fig. 4.4) Sliced view of a tetrahedral domain at z = 0.7



(Fig. 4.5) Sliced view of a tetrahedral domain at z = 0.2

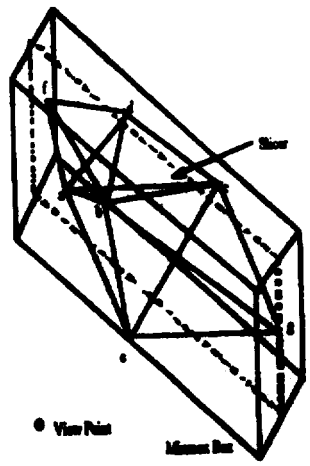


(Fig. 4.6) Sliced view of a tetrahedral domain at x = 0.2, y = 0.5, and z = 0.2

In Figure 4.4 4.5, and 4.6, we show the method of visualizing the internal tetrahedral domain by moving a slicer interactively. In this chapter, we explain the visualization technique for the sliced section of domain and visibility ordering algorithms.

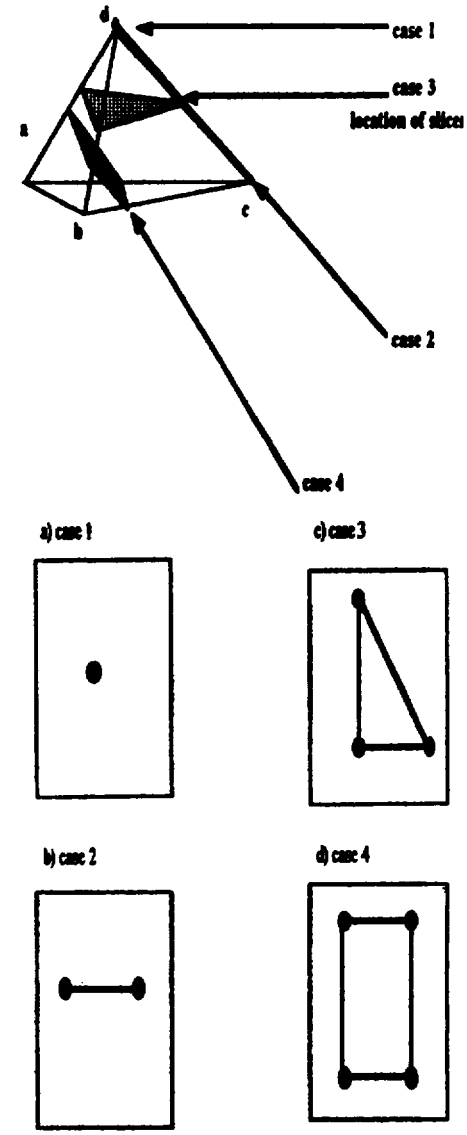
4.1 Sliced Section

One may slice and cut some tetrahedra to show the internal structure. We draw the intersection between slicer and tetrahedra. In our context, slicer means a plane which can cut a tetrahedral domain. We remove tetrahedra that are passed entirely through the moving slicer in order to visualize the inside of the intersected tetrahedral domain (see Figure 4.7). This technique has an analogous to culling in computer graphics hidden surface removal.



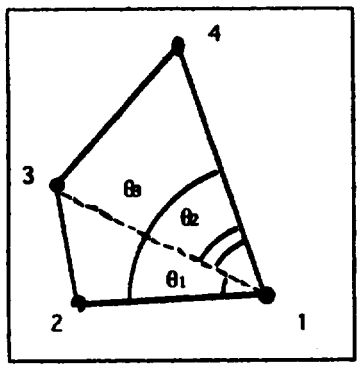
(Fig. 4.7) Intersection plane between the slicer and the tetrahedra (a, b, c, e), (b, c, e, g), (a, b, c, d), and (a, b, d, f)

The intersection between a plane and a tetrahedron can be a point, a line, a triangle or a quadrilateral (see Figure 4.8).



(Fig. 4.8) Four cases of intersection between nlicer and tetrahedra

In the case of the quadrilateral, one has to determine the proper ordering of four points to avoid an incorrect drawing of the intersection between a plane and a tetrahedron (see Figure 4.9).



$\theta_1 < \pi$, $\theta_2 < \pi$, and $\theta_3 < \pi$.

(Fig. 4.9) Three cases of 4 points' ordering

a) If θ_1 is the largest angle; ordering of four points is 1, 2, 4, 3 or 1, 3, 4, 2.

b) If θ_2 is the largest angle: ordering of four points is 1, 3, 2, 4 or 1, 4, 2, 3.

c) If θ_3 is the largest angle: ordering of four points is 1, 2, 3, 4 or 1, 4, 3, 2.

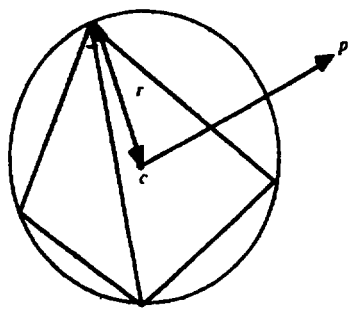
4.2 Visibility Ordering Algorithm

One needs to sort all the generated tetrahedra to apply volume rendering. Several methods are used to sort tetrahedra. Karasick's arrangement is based on power function order [11].

Williams introduces Meshed Polyhedra Visibility Ordering(MPVO) algorithm [12]. MPVO algorithm orders the cells of any acyclic convex set of meshed convex polyhedra based on half-space testing. MPVO algorithm can be divided into three parts. The first part is constructing adjacency graph. The second part is determining a direction to each edge in the adjacency graph. The last part is performing topological sort of graph. MPVO algorithm can take care of nonconvex mesh by converting nonconvex into convex mesh. But, it does not perform correctly in case of cyclic mesh. One can make parallel a topological sorting based algorithm to effectively speed up computations.

4.2.1 Power Function Based Ordering

The tetrahedron closest to the viewpoint has the smallest power function value. The power, $\prod(p, S)$, of a point p concerning a sphere S centered at c with radius r is $(p-c)^2-r^2$ (see Figure 4.10). An iso-surface rendering can be fast, because only the tetrahedra intersecting the iso-surface need to be considered when the viewpoint is changed. Power function based ordering is helpful when animating iso-surfaces; however, it only works in the case of Delaunay triangulation.



(Fig. 4.10) Visibility ordering based on power function

4.2.2 Half-Space Testing Based Ordering

One represents the ordering relation by an arrow through the interior faces of tetrahedra. One determines the direction of the arrow by using half-space testing. There are three steps required to implement this idea.

a) Construct Adjacency Graph

This data structure provides the following information (see Figure 4.11, Figure 4.12, and Figure 4.13) : the identification (ID) of the tetrahedron that shares the common face, directional information for each *behind relation* (i.e. inbound, outbound and none), and coefficients of the plane equations containing 3

vertices of each common face (i.e. $pe(x, y, z) = ax + by + cz + d$).

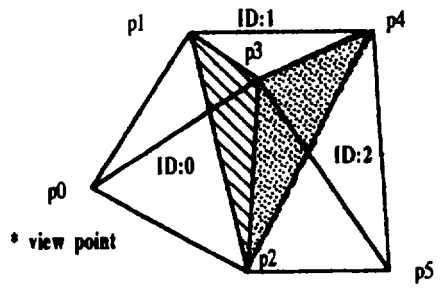
b) Convert Adjacency Graph to Directed Acyclic Graph

One connects each face with the *behind relation*. (see Figure 4.14). The common face is extended into a plane, which separates two half-spaces. Each half-space contains one of the tetrahedral objects. If we represent the *behind relation* by an arrow through the shared face, then the direction of the arrow is towards the tetrahedral object contained in the viewpoint half space.

c) Perform topological sorting

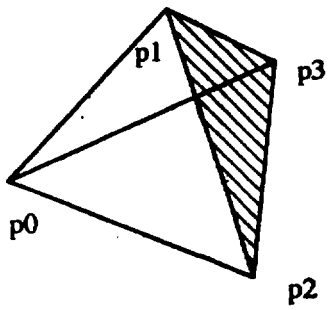
One can use Depth-First-Search (DFS) or Breadth-First-Search (BFS) for sorting tetrahedral objects.

i) DFS: Find all sink nodes, that have no outbound arrows. Place these nodes on a list called the sink cell list. DFS yields a reversed topological sort of the nodes. It will generate all tetrahedra regardless of the existence of cycles.



(Fig. 4.11) Three adjacent tetrahedra

tetrahedron ID : 0

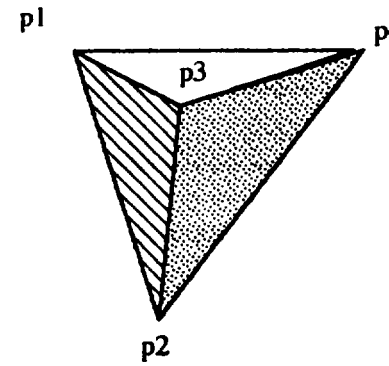


data structure of tetrahedron ID :0

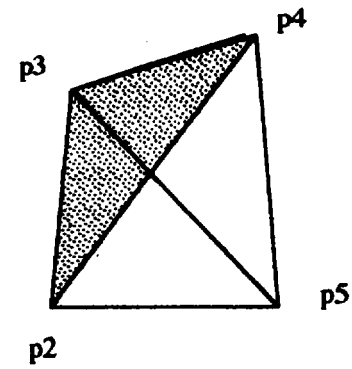
tetra[0][0] = p0
tetra[0][1] = p1
tetra[0][2] = p2
tetra[0][3] = p3
tetra[0][4] = 1 adjacent tetra (P1,p2,p3) ID
tetra[0][5] = -1 adjacent tetra (p0,p2,p3) ID
tetra[0][6] = -1 adjacent tetra (p0,p1,p3) ID
tetra[0][7] = -1 adjacent tetra (p0,p1,p2) ID
tetra[0][8] = 1 (inbound relationship)
tetra[0][9] = 0 (no exterior face)
tetra[0][10] = 0 (no exterior face)
tetra[0][11] = 0 (no exterior face)
tetra[0][12] = 0 (no imaginary face)
tetra[0][13] = 0 (no visited)
tetra[0][14] = 0 (no visited on this descent)
tetra[0][15] = 1 (number of inbound to cell)

(Fig. 4.12) Tetrahedron ID : 0 and its data structure

tetrahedron ID : 1

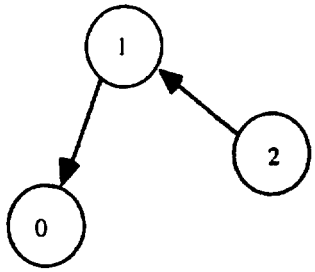


tetrahedron ID : 2



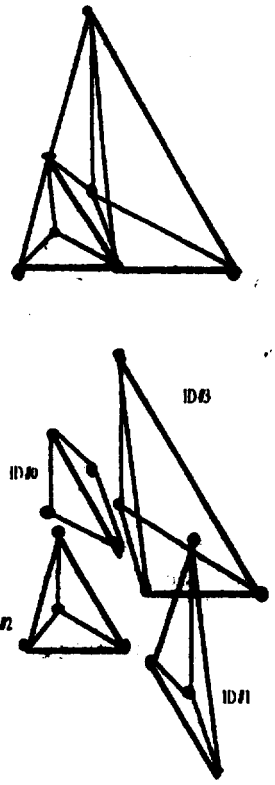
(Fig. 4.13) Tetrahedron ID : 1 and ID : 2

ii) **BFS**: Count the number of inbound arrows for each cell and find all the source nodes, those nodes which have no inbound arrows. Put them in a queue, called the source cell queue. A parallel algorithm may increase the computational speed.

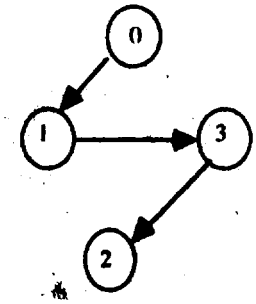


source list : {2}
sink list : {0}

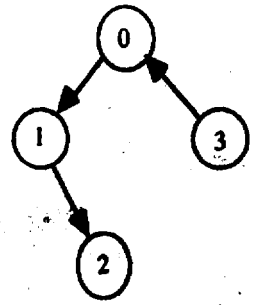
(Fig. 4.14) Directed acyclic graph of three tetrahedra



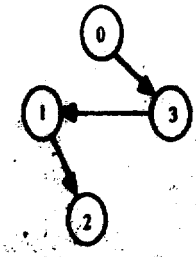
(Fig. 4.15) Four tetrahedra for comparison of visibility ordering methods



(Fig. 4.16) Visibility ordering graph of power function based method

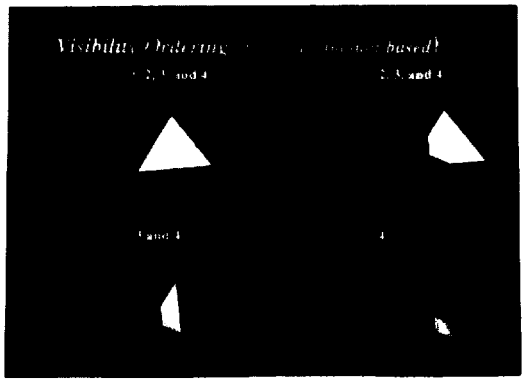


(Fig. 4.17) Visibility ordering graph of DFS based method

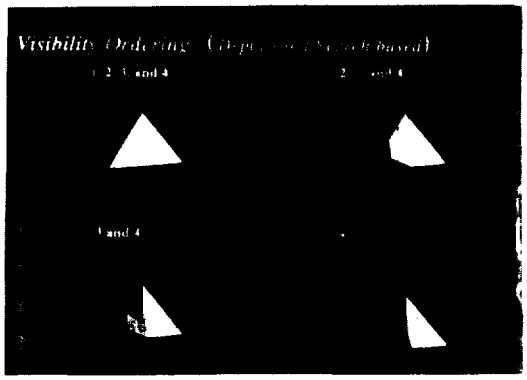


(Fig. 4.18) Visibility ordering graph of BFS based method

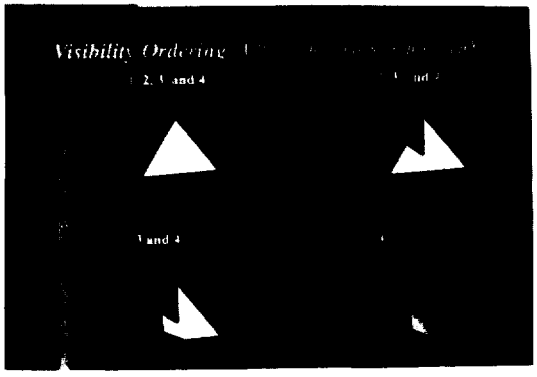
We compare various visibility ordering methods through examples (see Figure 4.15). We show each visibility ordering graph of power function, Depth-First-Search, and Breadth-First-Search, corresponding to Figure 4.16, Figure 4.17, and Figure 4.18, respectively.



(Fig. 4.19) Power function based visibility ordering



(Fig. 4.20) Depth-First-Search based visibility ordering



(Fig. 4.21) Breath-First-Search based visibility ordering

Figure 4.19 shows a rendering problem with power function based visibility ordering. However, Figures 4.20 and 4.21 show that other visibility ordering methods have no such difficulty with rendering. Visibility ordering means that the objects are sorted in terms of distance from a view point. Foe example, a visibility ordering of a set of objects from a viewpoint an ordering such if object A obstructs object B, then B proceeds A in the order.

5. Conclusions

The affine invariant tetrahedrization method is studied as a means of an affine invariant triangulation extended to three-dimension. Affine invariant tetrahedrization has an affine invariant transformation property. It means that the way of tetrahedrization does not changed by scaling. rotation, shearing, and translation.

In many applications, the choice of the location of coordinate axis or the choice of the units for measured data is rather arbitrary. We describe a criterion for tetrahedrization that is unaffected by this arbitrary choice. A new associate norm between two points in 3-D space is defined and used in a criterion. Affine invariant tetrahedrization can be used in case of scaling conversion. For example, some data whose units of measurements are inches and seconds need to be changed to feed and minutes. The result of affine invariant tetrahedrization remains as same while other tetrahedrization does not.

The visualization of tetrahedrization (i.e. tetrahedral domain, representative data points and transformed data points) can be helpful to understand the difference between Delaunay tetrahedrization and affine invariant tetrahedrization.

REFERENCES

[1] Nielson, G. M., and Foley, T. A., "A Survey of Applications of An Affine Invariant Norm," ASU Computer Science Department technical Report TR88-025, 1988.

[2] Nielson, "G. M., "A Characterization of an Affine Invariant Triangulation," Computing Süppl. 8, pp. 191-210, 1993.

[3] Delaunay, B. "Sur la Sphere Vide," Bull. Acad. Sci. USSR(VII), Classe Sci. Mat. Nat., pp. 793-800, 1934.

[4] Watson, D. F., "Computing the n-dimensional Delaunay Tessellation with Applications to Voronoi Polytopes," The Computer Journal 24(2), pp. 167-172, 1981.

[5] Hazlewood, C. T., "A Divide-and-Conquer Approach to D-dimensional Triangulations," Ph. D. Thesis, The University of Texas at Austin, 1988.

[6] Hazlewood, "C. T., "Approximating Constrained Tetrahedrizations," Computer Aided Geometric Design 10, pp. 67-87, 1993.

[7] Edelshrunner, H., Preparata, F. P. and Wast, D. B., "Tetrahedrizing point sets in Three Dimensons," J. Symbolic Computation Vol. 10, pp. 335-347, 1990.

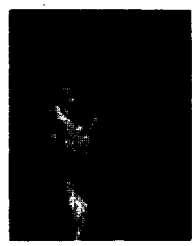
[8] Joe, B., "Three-Dimensional Triangulations from Local Transformatons," SIAM J. Sci. Stat. Comput. 10, pp. 718-741, 1989.

[9] Joe, B., "Construction of Three-dimensional Delaunay Triangulations using Local Transformations," Computer Aided Geometric Design 8, pp. 123-142, 1991.

[10] Nielson, G. M. and Schriver, B. "Visualization in Scientific Computing," IEEE Computer Society Press, 1990.

[11] Karasick, M. S., Lieber, D., Nackman, L. R., and Rajan, V. T. "Fast visualization of three-dimensional Delaunay," submitted., 1992.

[12] Williams, P. L. "Visibility Ordering Meshed Polyhedra," ACM Transactions on Graphics, vol. 11, no. 2, pp. 103-126, 1992.



이 건

1978년 고려대학교 전자공학과 졸업(학사)

1983년 고려대학교 대학원 전자공학과(공학석사)

1985년 아리조나 주립대학교 대학원 전기공학과(공학석사)

1995년 아리조나 주립대학교 대학원 전산과학과(공학박사)

1995년~현재 전북산업대학교 컴퓨터공학과 전임강사

관심분야:컴퓨터그래픽스(scientific visualization), 멀티미디어