3D Surface Reconstruction by Hierarchical Clustering and Incremental Triangulation

Kiattisak Sritrakulchai and Viboon Sangveraphunsiri
Robotic and Advanced Manufacturing Laboratory, Department of Mechanical Engineering
Faculty of Engineering, Chulalongkorn University
Phayathai Rd. Pathum Wan Bangkok 10330
Tel. 66(2)218-6585, 66(2)218-6448 Fax 66(2)218-6583
Email : reverserp@eng.chula.ac.th, viboon.s@eng.chula.ac.th

Abstract

In this paper, we propose a neural network based on our originated two-level adaptive hierarchical clustering algorithm in a preparation process. The preparation process is to manage 3D point clouds by eliminating the effect of the inconsistency and instability of unorganized point clouds, So that the triangular 3D surface models can be correctly obtained from the incremental triangular mesh creation algorithm. We also develop an adaptive self-adjustable connectivity to improve triangular mesh structure. Only one parameter, the edge length of triangle, is needed in the neural network. The proposed two-level algorithm consists of the level for clustering the point clouds that have the same order of limited edge length into the same cluster and the second level for generating a triangular surface model or drape surfaces over the points in the same cluster. The normal vectors for the generated triangular 3D surfaces model can be obtained from the incremental mesh creation algorithm. These normal vectors are used for generating the STL file or stereolithography format. From the experimental result, it can be shown that the proposed method is very effective for clustering unorganized point clouds, for generating triangular mesh of complex surfaces, as well as non-continuous surfaces such as a surface with holes.

Keywords: Unorganized Points / Surface Reconstruction / Hierarchical / Clustering Algorithm

1. Introduction

Global competition throughout nearly all industry makes it increasingly necessary for companies to be able to react to the needs and wants of their customers. The most significant attributes towards achieving this goal include the reduction of product development time. Both reverse engineering and rapid prototyping are emerging technologies that can play a promising role in reducing product development time.

A reverse engineering application is useful is when no CAD-data of the component available. It can be utilized as a design tool for developing a creative product, designed by a stylist. It can also be applied for quality control in manufacturing processes. Typical steps include scanning or digitizing 3-D objects such as clay, wood or any existing physical object by using various types of digitizing tools to obtain point clouds. The surface reconstruction from a point cloud can be used for other downstream applications such as CNC manufacturing and finite element analysis. It can also be used as a referenced surface for creating a more accurate CAD-model in a featured-based CAD system. Currently, a large number of data points can be efficiently measured using optical 3D measurement systems as well as automatic laser scanning systems. The point cloud is typically unorganized.

There are many research works which have proposed techniques to overcome the difficulty involved in the problem of surface reconstruction of unorganized points. They tried to correct the connectivity among the sampled points. The correct connectivity will assure us a reconstructed surface mesh that faithfully represents the shape and topology of the original

object from which the set of sample points were drawn. For an interpolation technique, Edelsbrunner [1] uses α-shape for generating the convex hull of a point set. The α-shape is a polytope surrounding the set of points and controls the maximum curvature of any cavity of the polytope. The triangular mesh is created inside the polytope by using Delaunay triangulation. The Delaunay triangulation is a collection of edges satisfying a circumsphere with the interior empty of sample points. The vertices of the triangle are at the boundary of circumsphere. Therefore, the major drawback of α-shape as applied to surface reconstruction is the correct parameter needed for constructing the circumsphere. This is normally done by trial and crror. Due to variations in the sampling density for any shape of point sets, according to the local feature area, there is no unique or ideal value for α. This technique works well for uniform sampling but often varies over different parts of the surface. Amenta and Bern [2] presented algorithms based on the three-dimensional Voronoi diagram and Delaunay triangulation for creating "crust". A set of triangles is created inside the crust. All vertices of crust and triangles inside the crust are sample points. The crust has been provably corrected if the sampling density distributes well enough over the shape surface. But the variation of sampling density depends on the curvature of the feature areas. If the high curvature is very dense, then holes or edges may appear. Due to undersampling in the high curvature area, local topology may appear as incorrect connectivity. Hoppe [3] proposed an approximation technique. His technique used the input points to define a signed distance function for determining an approximate tangent plane at each sample point by using least squares on k nearest neighbors, and then took the signed distance to the nearest point's tangent plane as the distance function. The output mesh is created by zero-set of the distance function, interpolated and polygonalized by a marching cubes algorithm. This technique needs very accurate estimation of tangential planes so that the correct surface reconstruction is obtained otherwise approximating inherently by some low-pass filtering of the data. This is desirable, if noise is present, but it causes some loss of details for the features of the surface.

More recently, there are many research works on surface reconstruction using neural networks. There are two forms commonly used for surface reconstruction from an unorganized point cloud. The first form is the Kohonen's self organizing map. It needs to predetermine topology or connectivity of a triangular surface, which is the network structure, for a reconstructing surface. Then a neural network learning algorithm is used for adjusting 3D coordinates of each vertex of triangular surfaces so that they are close enough to the input point with a specified constraint. Its disadvantage is the fact that the network structure had to be pre-specified in advance. The details of the procedure can be consulted [4-6]. The choice of an unsuitable network structure, however, can badly degrade network performance. The second form of neural network applied in order to overcome drawbacks of Kohonen's model is the Growing Cell Structures [7-10]. It is similar to the Kohonen's Self Organizing Maps. For the Growing Cell Structure, network structure is a triangle and network nodes are vertices. The network structure starts with a very simple network, the triangle, and grows incrementally by adding one new node or deleting a node to the network which depends on the distribution of the input point set. The major drawback is that it is very difficult to assign the suitable parameters needed for training the learning network to adjust the position of vertices of the triangular mesh close to the original points which is the approximation of vertices of triangular mesh from actual input points. This will cause some loss of information. This is also unsuitable for high accuracy applications such as rapid prototyping, and manufacturing with CNC systems.

In processes of computer graphics, which need to turn the point cloud into a graphics model, there are several steps such as the reconstruction of initial piecewise-linear model, cleanup, smoothing, simplification and fitting with curved surface patches. The researches reviewed above as well as our work focus on the first step. For most of researches mentioned above, the focus is on the way that the techniques generate a computer graphics models, but our work is to develop digital models which are suitable for downstream applications such as product development in manufacturing systems. Our technique also

generates the triangular mesh in a STL file which can directly be used for rapid prototyping.

The main differences of our technique compared with prior researches is that instead of using a sampling points set, we use only one point at each cycle in our algorithms. Every input point will be sorted and ordered by a sorting and ordering process. Our work is a unique technique, this process used a series of slicing planes moving in a user-specified direction, which is normally x-director of the object, to sort and order the point cloud. Then the sorted data points are used in the organizing and clustering process of the two-level adaptive hierarchical clustering algorithm. This algorithm will help to arrange the new input to a suitable group of points so that the more accurate triangular mesh can be created. After that, the polygonizing is followed and normal vectors for the STL file will be created by the incremental triangular mesh creation algorithm. So with our algorithm, there is no effect of undersampling. The other methods used sampling points with interpolation and approximation. This will cause an undersampling effect.

2. Preparation Process

Figure 1 illustrates the data flow process of our whole process. Point clouds will be sorted and ordered to minimize complexity or disordering. Then, in the neural network, the sorted data points are used in two processes as the organizing and clustering process and the deletion and combination process. Finally, the output data from the neural network are ready for generating a triangular mesh with the incremental triangular mesh creation algorithm.

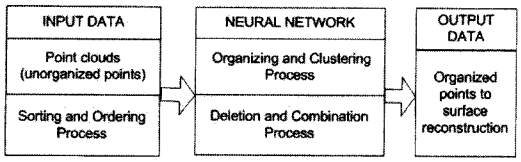


Figure 1 Preparation process for reconstruction surface.

2.1 Sorting and Ordering Process

For the sorting and ordering process, a series of slicing planes, which are normal to the x-direction of the object, are used to the define intervals for surface reconstruction. The xy-plane is the top-view of the object. Then, each input point will be sorted and ordered in the x-direction of the object. The procedure of this process is as follows:

READ Point Cloud File (obtained from a digitizing machine)

SET or define the interval of the slicing plane in x-direction

COMPUTE the sorted and ordered points in ascending order of x-direction.

The sorted and ordered points are grouped into regions by using slicing planes. Figure 2 shows 2 slicing planes with intervals which cover the region of $0 < x \leq 10$ and $10 < x \leq 30$. If we do not need to specify the whole point cloud for grouping, we can just specify only a portion the point cloud in the grouping process.

The points are sorted by considering the distance from the slicing plane along the x-axis. So, point p1, p2 and p3 in Figure 2 will be grouped to the slicing plane of $0 < x \leq 10$ and ordered starting from the nearest x-distance. Similarly, point p4, p5, p6, p7, and all of the rest, are ordered in the same way and grouped to the $10 < x \leq 30$ slicing plane. After this arrangement, the organizing and clustering process will proceed.

The advantage of this technique is that it can be extended for cluster computing processes in case huge surfaces need to be reconstructed. In addition, section views can also be obtained if needed. But in this paper, we are interested only in grouping.

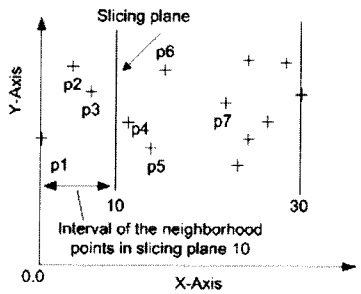


Figure 2 Ordering of slicing planes and points within each plane.

2.2 Organizing and Clustering Process

We need to define some index symbols to help clustering, organizing, and recognizing point data used in the two-level adaptive hierarchical clustering algorithm. The point cloud will be grouped into a hierarchical structure as shown in Figure 3 as follows.

M: the set of all point data, where $P_i \in M$; i=1,2, … , n, is the point data in M. And n is the number of point in the set M.

SM_m: The set of the partition of point data, subdivided from all point data. For the first level clustering, distance similarity will be used for generating these partitions of mesh at index m; m=1,2, … ,a, where a is the number of set partition of the point data.

$NC_{c,m}$: The sub-set of the partition of point data at index c of second level clustering and c=1,2, … , b, where b is the number of sub-set partitions of point data. So, $NC_{c,m}$ is the child at index c of the parent SM_m (partition of mesh at m).

$C_{v,c,m}$: The members of sub-set $NC_{c,m}$ at index v. So, $C_{v,c,m}$ will be called a cell or members of $NC_{c,m}$ at v, where v = 1, 2, …, d, and d is the number of members of sub-set $NC_{c,m}$. So, v indicates the point number in the point cloud. See the example shown in Figure 3.

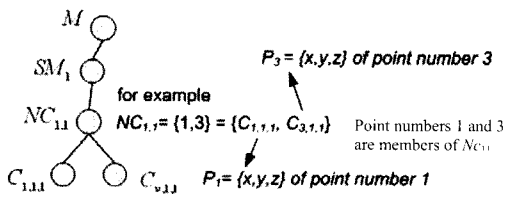


Figure 3 The symbols of two-level hierarchical clustering structure

In addition to the symbols specified above, a parameter, for specifying the similarity of points, will be specified by a user. This parameter is needed to start the searching algorithm. This parameter is called the limited length of similarity or edge length. (ρ). It represents the limited distance among the points used in the clustering process.

After the clustering process, the set of points which have a similar distribution will be clustered into the same partition which will be represented by a mutually disjoint subsets or clusters in the first level, that is $SM_1, SM_2, ..., SM_q, ...SM_a$. Each SM_q is divided into NC clusters or second level, $NC_1, NC_2, ..., NC_h, ...NC_b$. Then, the triangular mesh will be generated from these NC clusters.

In conclusion, the organizing and clustering process is to map sampled points (in M) to the associated two-level hierarchical cluster of similarity of point data as shown in Figure 3.

SM, level-one cluster, is a set partition of M. Then, SM is a collection of nonempty sets, $\{SM_1, SM_2, ..., SM_a\}$, such that, $M = SM_1 \cup SM_2 \cup ... \cup SM_a$, and sets, $SM_1, SM_2, ..., SM_a$ are mutually disjoint, for all $SM_q \cap SM_l = 0$ where q, $l = 1, 2, ..., a$, whenever $q \neq l$. The NC, level-two cluster, is the sub-set partition of SM_q. So, NC is a collection of nonempty sets, $\{NC_1, NC_2, ..., NC_b\}$, such that, $SM_m = NC_1 \cup NC_2 \cup ... \cup NC_b$, and sub-sets, $NC_1, NC_2, ..., NC_b$ are mutually disjoint, for all $NC_h \cap NC_e = 0$ where $h, e = 1, 2, ..., b$, whenever $h \neq e$.

Figure 4 shows that the two-level hierarchical clustering structure is more than one partition, SM_1 and SM_2, and/or the number of sub-partitions is greater than 1, $NC_{1,1}$ and $NC_{2,1}$.

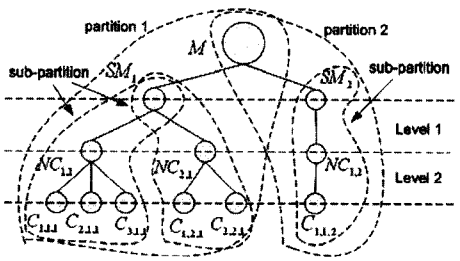


Figure 4 Subdivision of partition in the two-level hierarchical clustering structure

Details of the algorithm is as follows:

CALL Sorting and Ordering Process
SET or define the limited length of similarity or edge length
SET or initialize the first input point into the first two-level hierarchical cluster, for example, $SM_1, NC_{1,1}, C_{1,1,1}$ where P_1= the first input point.
FOR $2 \leq i \leq n$
SET for new input point, P_i, create a new two-level hierarchical cluster as SM_{new}, $NC_{1,new}$, $C_{1,1,new}$ where *new* indicates the new point
FOR every SM_m where $m = 1,2,...,q$
FOR every NC_c of SM_m where $c = 1,2,...,h$
SET or initialize set *Lg*=[]
FOR every point in $C_{v,c,m}$ of NC_c where $v = 1,2,...,t$
COMPUTE $(D_v)_{c,m}$, the distance between the new input point, P_i, and the point P_v in $C_{v,c,m}$ or $(P_v)_{c,m}$ So, $(D_v)_{c,m}$= $\left\|(P_v)_{c,m} - P_i\right\|$
IF D_v < limited length, ρ
ADD D_v to the members of set *Lg*
COMPUTE number of members in set *Lg*
IF number of members in *Lg* = = 1 and number of members in the set $NC_{c,m}$ = =1
COMBINE $SM_m, NC_{c,m}$ to $SM_{new}, NC_{1,new}$
DELETE SM_m, $NC_{c,m}$
ELSE IF the number of members in set *Lg* = = 0 and the number of set $NC_{c,m} > 1$
INSERT the new two-level hierarchical cluster, SM_{new}, $NC_{1,new}$, $C_{1,1,new}$, to *M*.
ELSE IF the number of members of set *Lg* = = 1 and the number of members of set $NC_{c,m} > 1$
COMBINE SM_m to SM_{new}
DELETE SM_m
ELSE IF the number of members in *Lg* > 1 and number of members in set $NC_{c,m} > 1$
COMBINE $SM_m, NC_{c,m}$ to $SM_{new}, NC_{1,new}$
DELETE $SM_m, NC_{c,m}$
CALL the Incremental Triangular Mesh Creation Algorithm
UPDATE the number of partition, SM , NC

2.3. Learning and Inserting Process

Figure 5 illustrates the organizing and clustering process. The network starts from the simple neural network structure, as shown in Figure 5(a), which will consist of 4 nodes as point cloud (M), a partition (SM_1), a sub-partition $(NC_{1,1})$ of SM_1 and a cell $(C_{1,1,1})$ which contains a point.

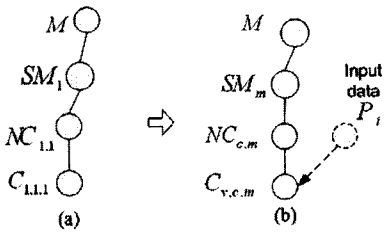


Figure 5 Starting neural network structure.

Then, we apply the distance similarity equation for searching as $(D_v)_{c,m}$ = $\left\|(P_v)_{c,m} - P_i\right\|$, where $\|.\|$ denotes the Euclidean vector norm. So, $(D_v)_{c,m}$ is a set $\{d_1, d_2, \cdots, d_v\}_{c,m}$, the distance between point

data, $(P_v)_{c,m}$, (Point v of sub-set partition index c of set partition index m) and new sampled point data P_i which is the 3D coordinate of input point data as shown in Figure 5(b). The distance in the set $(D_v)_{c,m}$ is used for clustering by comparing to the limited length, ρ. In this clustering process, there are four possible cases or events of learning and insertion for every input point data as indicated in Figure 6. We will define the distance set L_g, $\{d_1, d_2, \cdots, d_g\}$; where the distance ($d_g$) is less than ρ. The number of members in the set $(L_g)_{c,m}$ will indicate the event that will occur.

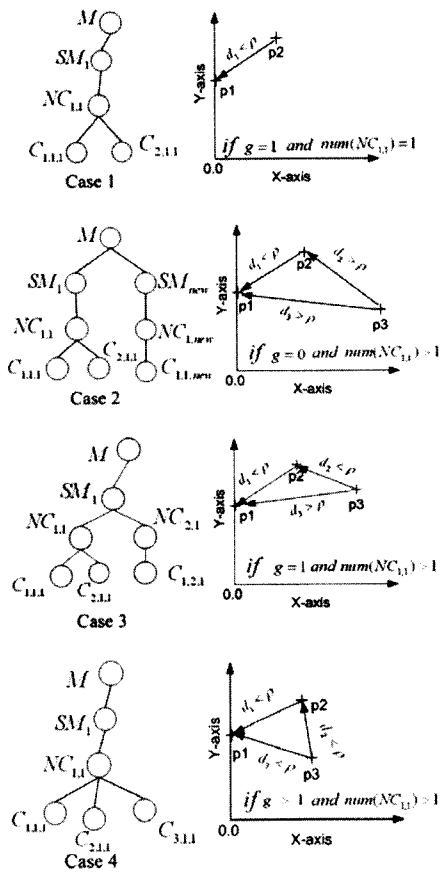


Figure 6 Learning and insertion of the four events for new fed input point data (P_i).

Figure 6 shows the four cases of the learning and insertion of the new fed input point data (P_i) (in this example case is point P_2). Case 1: When $d_1 < \rho$, which means that the input point P_2 can be inserted into the sub-partition $NC_{1,1}$ and total number of points in this set is 2 which is not enough to create a triangular mesh. In other words, the condition of case 1 occurs when the number of members of the set Lg = 1 or g = 1 and the number of members in set $NC_{1,1}$ is equal to 1. Case 2: The new input point is fed (in this example is P_3). If $d_2 > \rho$ and $d_3 > \rho$, this means that the new point P_3 is not close to any point P_1 and P_2 in sub-partition $NC_{1,1}$ or $g = 0$. So, the new partition needs to be created as shown in Figure 6. Case 3: The new input point P_3 is fed. If $d_2 < \rho$ $(g = 1)$ and $d_3 > \rho$ which mean that point P_3 can not be inserted into the sub-partition $NC_{1,1}$. So, we need to create a new sub-partition $NC_{2,1}$ and point P_3 to be inserted into the $NC_{2,1}$. Case 4: If $d_3 < \rho$ (or $g > 1$), then point P_3 will be inserted into sub-partition $NC_{1,1}$. So, the number of points in the sub-partition is enough to create a triangular mesh.

2.4. Deletion and Combination Process

The deletion and combination process will be used when the number of members of set $Lg \geq 1$, that is case 1, case 3 and case 4 occur, in the organizing and clustering process.

In Figure 7, If the new input point P_2 of two-level hierarchical clustering structure, $SM_{new}, NC_{new,1}$, $C_{1,1,new}$, is close to P_1 of $SM_1, NC_{1,1}, C_{1,1,1}$ and $d_1 < \rho$., this means that $SM_1, NC_{1,1}$ and $SM_{new}, NC_{new,1}$ can be combined as shown in Figure 7 for case 1. After the combination, SM_1 and $NC_{1,1}$ set will be deleted. If the new input point P_3 is close to P_2 but further to P_1 of SM_1, $NC_{1,1}$, $C_{1,1,1}$, $C_{2,1,1}$ with $d_2 < \rho$ and $d_3 > \rho$, this means that SM_1 and SM_{new} can be combined as shown in Figure 8 for case 3 and SM_1 should be deleted. If the

new input point P_3 is close to P_1 and P_2 with $d_2 < \rho$ and $d_3 < \rho$, this means that $SM_1, NC_{1,1}$ and $SM_{new}, NC_{new,1}$ can be combined as shown in Figure 9 for case 4 and SM_1 and $NC_{1,1}$ sets should be deleted.

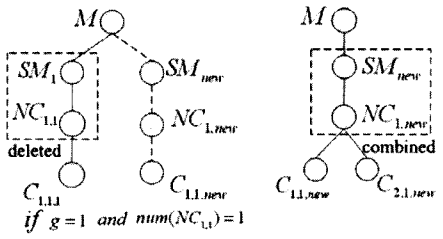


Figure 7 Deletion and combination for case 1.

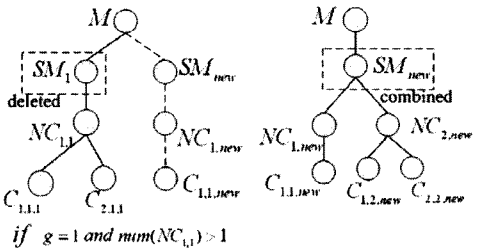


Figure 8 Deletion and combination for case 3.

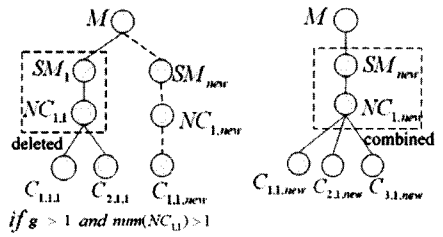


Figure 9 Deletion and combination for case 4.

After the deletion and combination process, we eliminate all the duplicate set of points. Each *NC* will contain the points and neighborhood points that have the same similarity or the distance between these points is less than the specified limited length. This will provide the set of points for creating a triangular mesh more efficiently.

3. The Incremental Triangular Mesh Creation Algorithm

The triangle mesh creation process will proceed, whenever, the number of cells within the sub-set of the partition of point data, $NC_{c,m}$, is greater than or equal to 3, and the number of members in the set $\left(L_g\right)_{c,m}$ is greater than 1 or $g > 1$. This is the condition of case 4 as described in the organizing and clustering process. For ease of explanation, we will demonstrate only a single partition SM_1. So, from this point to the rest of this paper, the member in the sub-partition $C_{1,1,1}$ will be replaced by C_1.

3.1 The First Triangle Creation

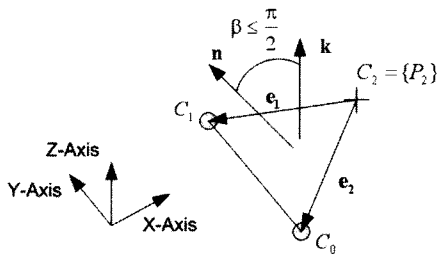


Figure 10 The normal vector direction of the first triangle.

When the number of cells is 3 or the number of members in the set $NC_{c,m}$ equal to 3, a triangle can be created by setting $\beta \le \pi/2$ as shown in Figure 10. The β is the angle between vector $\mathbf{k}$, a vector along Z axis specified by a user for a slicing plane, and $\mathbf{n}$, the normal vector of the triangle T_1, is calculated as shown in Figure 10.

$$\mathbf{n} = \frac{\mathbf{e}_1 \times \mathbf{e}_2}{|\mathbf{e}_1 \times \mathbf{e}_2|}, \quad (1)$$

$$\beta = \cos^{-1}\left[\frac{\mathbf{n} \bullet \mathbf{k}}{nk}\right], \quad (2)$$

For example, $\mathbf{e}_1 = \overline{C_2C_1}$ and $\mathbf{e}_2 = \overline{C_2C_0}$, if $\beta \le \pi/2$, then the triangle can be created along

the ordered cells $\{C_2, C_1, C_0\}$, otherwise, the order of cells is $\{C_2, C_0, C_1\}$. In other words, the triangle composed of three edges is $\{\overline{C_2C_1}, \overline{C_1C_0}, \overline{C_0C_2}\}$ or $\{\overline{C_2C_0}, \overline{C_0C_1}, \overline{C_1C_2}\}$ according to the direction of the normal vector.

Details of the algorithm are shown in the following algorithm:

IF the number of members in set $NC_c = = 3$ and g > 1
  COMPUTE **n** and β
  IF $\beta \le \pi/2$
    COMPUTE order of cell in counter clockwise direction
  ELSE
    COMPUTE order of cell in clockwise direction
  COMPUTE create normal vector

3.2 The General Triangle Creation

The other triangles are computed, after the first triangle was created as shown in Figure 11.

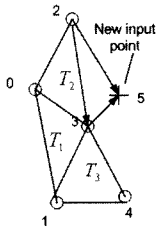


Figure 11 The computation for the other triangles.

We need to search for the best edge of the existing triangular mesh for sharing with the new created triangle of the new input point, for example, point 5 shown in Figure 11. For efficiency and effective searching, our algorithm just searches the best edge within the boundary triangular mesh (BTM). The BTM is composed of the edges of the triangles that do not share any other edge of triangles or the edges at outside of the triangular mesh in 3D space. Using BTM will reduce computation and searching time, especially for much data. For example in Figure 11, it shows the BTM of the new input point 5 is composed of edges $\overline{C_0C_1}$, $\overline{C_3C_2}$, $\overline{C_2C_0}$, $\overline{C_4C_3}$, and $\overline{C_1C_4}$.

Figure 12 shows two possibilities of the nearest distance calculated from the new input point. One possibility is that, the calculated distance is projected outside the edge of boundary triangular mesh as shown in Figure 12(a), $d_{4,5}\cos\beta > d_{3,4}$. So, the nearest distance is $d_{3,5}$. If $d_{4,5}\cos\beta < d_{3,4}$, then the nearest distance is $d_{3',5} = d_{4,5}\sin\beta$ as shown in Figure 12(c). The angle β is calculated from equation (2). The new created triangle will share the best edge with the existing triangular mesh. This best edge will be called the winner edge which is the nearest distance between the BTM and the new input point according to the conditions shown in Figure 12.

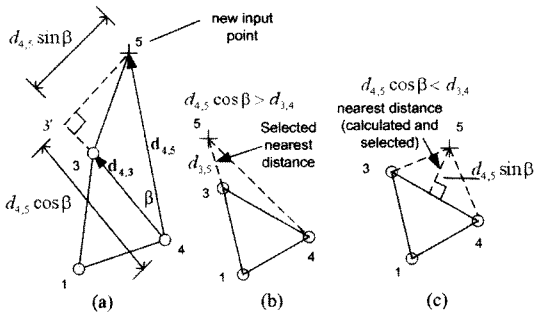


Figure 12 Condition for searching the nearest distance.

For example, if we assume that the winner edge is $\overline{C_3C_4}$, then the winner triangle is the triangle formed by the edges $\overline{C_3C_4}$, $\overline{C_3C_1}$ and $\overline{C_1C_4}$. The ordering of cells of this winner triangle are based on the normal vector of the winner triangle. For example, Figure 13 shows the normal vector, $\mathbf{n}_j$ as calculated from equation (1), of the winner triangle, the cells of the winner triangle will be ordered as $\{C_4, C_3, C_1\}$, in a counter-clockwise direction. The same direction will be assigned to the new triangle of the triangular mesh as $\{C_5, C_3, C_4\}$.

There are some special cases such as surface discontinuity, or surfaces that are very

steep, point error, or noise due to the combination of more than one point cloud. We need to check whether these conditions occur by inspecting the angle between the normal vector of the winner triangle and of the new triangle including the neighborhood triangle of the winner triangle.

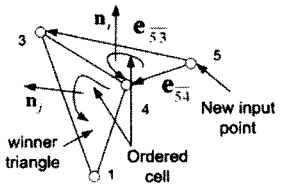


Figure 13 The new created triangle with respect to the winner triangle.

For example in Figure 14, it shows the direction change of angle θ, which is the angle between the new created triangle for a new input point and the winner triangle. If $\theta < \pi/2$, the new triangle is created by sharing its edge with an edge of the winner triangle. But if $\theta > \pi/2$, this means that there is a high steep surface or surface discontinuity. To confirm the case, we need to examine the neighborhood of the winner triangle, whether at least one $\gamma < \pi/2$ is met, So, the new triangle can be created. Otherwise the new input point will be moved to the new extra case, as compared to the cases explained in Section 3 , where the distance of the new input point is within the limited length $(g > 0)$ but the angle $\theta > \pi/2$ as well as $\gamma > \pi/2$.

The γ is the angle between the neighborhood of the winner and the check triangle, the check triangle is created for inspection to tell us about the closeness of the new triangle and the winner triangle. This case means that the new input point (point 4 in Figure 15) is very close to the winner triangle. The closeness is defined by the angle ψ. The angle ψ is the angle between the normal vector of the winner triangle and the normal vector of the check triangle as illustrated in Figure 15. The check triangle is formed by an edge of the neighborhood triangle and the edge of the new triangle. An example of a check triangle is shown in Figure 15.

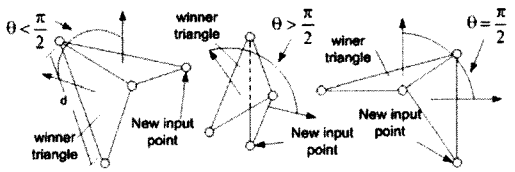


Figure 14 The angle between winner triangle and a new created triangle.

The decision whether the new triangle should be created from the new input point or the new input point will be moved to the new extra case can be concluded as follows:

1) If $g > 1$, $\theta > \pi/2$ and $\gamma < \pi/2$, the new triangle can be created.

2) If $g > 1$, $\theta > \pi/2$ and $\gamma > \pi/2$ the new input point will be considered as an extra case for creating a new partition as described in Section 3. But after dividing into the new partition, the condition $g = 0$ may occur which is the condition of case 2.

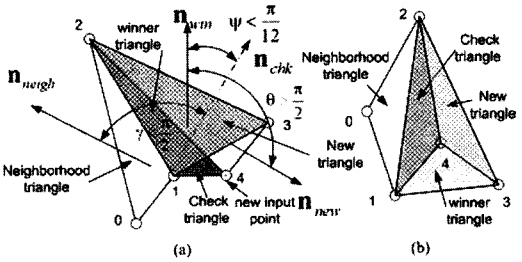


Figure 15 Checking overlapped triangle for subdivision to three triangles.

In addition to condition 1, if $\psi < \pi/12$, as shown in Figure 15, this means that the new input point is very close to the winner triangle. The two triangles, the new triangle and the check triangle, will almost overlap the winner triangle. This will affect the quality of the triangular mesh. So, we need to divide the winner triangle into three triangles so that the normal vectors of the additional three triangles will point to the same direction. To create these three new triangles, we will substitute the new input point, point 4 or C_4, into the set of points, $\{C_3, C_2, C_1\}$, of the winner triangle as follows:

The first triangle will be formed by substituting C_4 for C_3 in the set $\{C_3, C_2, C_1\}$ to

get the triangular set $\{C_4, C_2, C_1\}$. The normal vector of the triangular set $\{C_3, C_2, C_1\}$, and the normal vector of the triangular set $\{C_4, C_2, C_1\}$ point to the same direction. Similarly, the second and the third triangle can be obtained as $\{C_3, C_4, C_1\}$ and $\{C_3, C_2, C_4\}$ respectively. The value $\pi/12$ for the angle ψ is a heuristic value. Alternatively, we can specify this value manually within the range $(0, \pi/2)$ which depends on the complexity of the surface. This range can be re-specified by a user if needed. For a smooth surface, the range of angle θ can be specified smaller to reduce the effect of noise.

Details of the algorithm is as follows:

IF the number of members in $NC_c > 3$ and g > 1
  SET Near_EachEdge=[]
  FOR every vertices that satisfy the BTM condtion
    COMPUTE the distance between the new input point and vertices
    IF $d_{4,5}\cos\beta > d_{3,4}$
      Add.Near_EachEdge= $d_{3,5}$
    ELSE
      Add.Near_EachEdge = $d_{4,5}\sin\beta$
  COMPUTE min(Near_EachEdge)
  COMPUTE create normal vector of the new triangle
  COMPUTE angle θ, γ, ψ
  IF $\theta > \pi/2$
    IF $\gamma < \pi/2$
      IF $\psi < \pi/12$
        COMPUTE split triangle to 3 triangle
      ELSE
        COMPUTE create new triangle create normal vector
    ELSE
      INCREASE create new partition $SM_{new}, NC_{1,new}, C_{1,1,new}$
  ELSE
    COMPUTE create new triangle create normal vector

3.3 Creation of The Neighborhood Triangles of A New Triangle

After creating a new triangle from the shared edge, $\overline{C_3C_5}$, of the winner triangle, we also need to create neighborhood triangles of the new triangle. The neighborhood triangles and the new triangle have to share the same common point. There are 2 common points as C_3 and C_5, which are the vertices of $\overline{C_3C_5}$. Figure 16 illustrates that there are three possibilities of creating new neighborhood triangles. The first possibility is shown in Figure 16(a). There are three neighborhood triangles which need to be considered to form the required neighborhood triangle of the new triangle. It is very easy to show that none of them is not qualified for creating the required neighborhood triangle. The new neighborhood triangles, $\{C_6, C_3, C_0\}$ and $\{C_4, C_5, C_6\}$ created by the three neighborhood triangles, are overlaping with the winner triangle as shown in Figure 16(a). Actually, the $\{C_2, C_5, C_6\}$, $\{C_1, C_5, C_6\}$ and $\{C_6, C_3, C_1\}$ triangles can also be created by the three neighborhood triangles, but we can not consider this case because it is not formed by the BTM. For the second possibility, there are two neighborhood triangles to be considered to form two new neighborhood triangles as $\{C_4, C_3, C_6\}$ and $\{C_2, C_5, C_6\}$ as shown in Figure 16(b). Only triangle $\{C_4, C_3, C_6\}$ does not overlap with the winner triangle. So, $\{C_4, C_3, C_6\}$ is the new neighborhood triangle. For the third possibility, there are three new neighborhood triangles as $\{C_4, C_3, C_6\}$, $\{C_1, C_3, C_6\}$, and $\{C_3, C_0, C_6\}$. In this case, triangle $\{C_4, C_3, C_6\}$ and triangle $\{C_1, C_3, C_6\}$ do not overlap with the winner triangle. This is because $\phi_1 > \pi/2$ and $\phi_2 > \pi/2$. The angle ϕ is the angle between the normal vector of the new triangle and the normal vector of the new neighborhood triangles. Because the two new neighborhood triangles are overlapped with each other as shown in Figure 16(c), they share the same

common point C_3. so, we need to select only one of them by considering the angle ϕ .The small angle of ϕ will be used to select the new neighborhood. From Figure 16(c), it can be shown that the new neighborhood triangle is the triangle $\{C_4, C_3, C_6\}$. Figure 16(d) shows the result of the new neighborhood triangle of the new triangle.

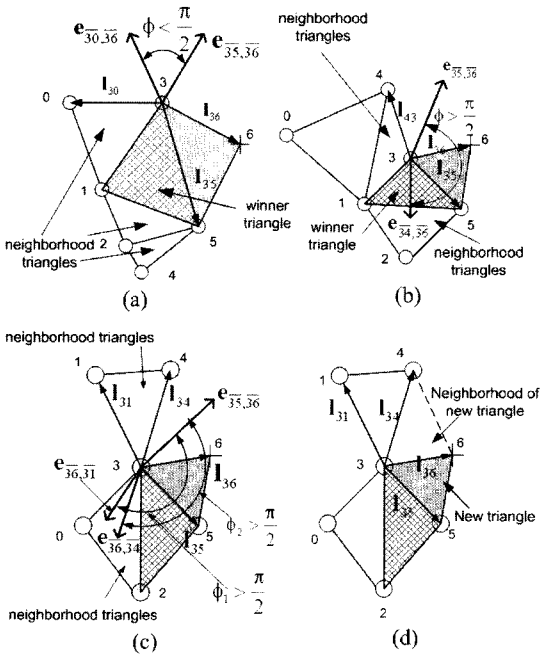


Figure 16 Condition for creating neighborhoods of a new triangle.

Details of the algorithm are shown in the following example:

COMPUTE angle ϕ of the neighborhood triangle of the new triangle which depends on BTM
SET NumNeighCreated=0
COMPUTE NumNeighCreated=the number of ϕ, for$(\phi > \pi/2)$
IF NumNeighCreated > 1
  IF vertices of new neighborhood = = vertices of edge winner
    COMPUTE minimum(angle ϕ) and create neighborhood triangle
  ELSE
    COMPUTE create every neighborhood triangle
ELSE IF NumNeighCreated = =1
  COMPUTE create neighborhood triangle

3.4 Self-Adjustable Connectivity

To improve triangular mesh structure, we present self-adjustable connectivity. For example in Figure 17, the two triangles, shared with the edge l_{12} are triangle $\{C_2, C_1, C_0\}$ and $\{C_3, C_1, C_2\}$. The edge of vertices, between C_1 and C_2, before adaptation is l_{12}. After adaptation, the edge l_{03} will be used to separate the triangular mesh instead of l_{12}. Because the length of l_{03} (variable le_{cross}), is less than the length of l_{12}, (variable le_{share}). So, the adaptation of connectivity of triangular mesh will modify the existing triangular mesh as shown in Figure 17(a) to the new triangular mesh as shown in Figure 17(b). This will improve the quality of the mesh.

More systematic checking whether the self-adjustable connectivity should be applied can be considered from the angle φ as shown in Figure 17. The angle φ is the angle measured between the normal vector of the two triangles formed by the non-share edges of the existing two triangles as illustrated in Figure 17. If $\varphi > \pi/2$, then the adaptation of the two triangles occurs. The adaptation is continued for every sharing edge of the two triangles. This will improve the quality of mesh.

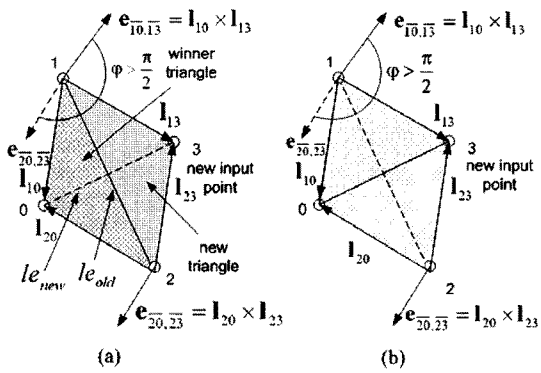


Figure 17 The two triangles for self-adjustable connectivity of triangular mesh structure.

Details of the algorithm are shown in the following algorithm:

COMPUTE le_{share}, le_{cross}, and φ of new triangle and winner triangle
IF ($\varphi > \pi/2$) and ($le_{share} > le_{cross}$)
  COMPUTE modify connectivity
  SET modified_triangle =[]
  WHILE number of triangle sharing edge with other triangle not equal to 0
    COMPUTE le_{share}, le_{cross}, φ
    IF ($\varphi > \pi/2$) and ($le_{share} > le_{cross}$)
      COMPUTE modify connectivity
      INCREASE modified_triangle
  ENDWHILE
ELSE continue

4. Results

To demonstrate our algorithm, we develop application modules using Python. Python is an object oriented script language. The main advantage of using Python is it provides us the ability to interface to other languages. Python can also be implemented in various operating systems. Our tested platform is a Pentium 4, 2 GHz running windows operating system with 256 M of memory. For post processor, we are using the Blender, a freeware, for graphic display of triangular mesh. We also developed our own post-processor for graphic display [11]. The point clouds, used in the experimentation, were obtained by using 3 machines as 1) Our developed automatic laser scanning machine [12], the digitized points used in example 1 and 2. 2) Contact CMM (Brown&Sharpe Microval 343), and the data used in example 3. 3) The optical 3D-measurement systems, (ATOS II 400, Advanced Topometric Sensor from GOM), the data used in example 4- 9; The digitized point clouds are typically unorganized. The output results of the examples using our procedure are in STL format. For triangular mesh creation, the parameters for checking overlap of the triangles and for checking discontinuity surface are $\psi < \pi/12$ and $\theta > \pi/2$ respectively.

Nine examples are presented here to illustrate the efficacy of the algorithm for creating STL models. Example 1 and 2, as shown in the Figure 18(a) and 18(b), are the STL models of a bottle and a mouse, which the point cloud is obtained from the in-house development automatic laser scanning machine. Example 3, as shown in Figure 18(c), is the STL model of a human face which the point cloud is obtained from the contact CMM. And examples 4 – 9, as shown in the Figure 18(d) – 18(h), are the STL models of the other human face, a pan handle, a teeth, a spare part, and a car seat, which the point cloud is obtained from ATOS II 400. As shown in Figure 18(d) and Figure 18(g), the surface has some discontinuities, 2 holes represent eye sockets and 2 holes for bolts, with open surface. Figure 18(e) is the pan handle model. It is a closed surface with combination of the continuous and discontinuous surface (grooves). Figure 18(f) and Figure 18(h) are STL of the teeth model and the car seat. They are very complex surfaces and the surface features have sharp edges. Figure 18(i) is intended to show the capability of our technique for combining multi-portions of surfaces. We divided data in example 4 into 2 portions. We processed each portion individually as shown in Figure 18(i). Then, we combined those 2 portions together as shown in Figure 18(j).

From the examples, it is obvious that our technique is suitable for any type of point cloud, organized or unorganized point cloud. The quality of the surface depends on the quality of the digitizing machine. Our work is concentrated on triangular mesh creation based on an unorganized point cloud obtained directly from a digitizing machine. We do not cover the clean up, smoothing and simplification. Most of the commercial digitizing machines, such as ATOS, have some filter capability which is good enough for our technique as shown in example 3 - 9. In examples 1 and 2, the quality of the surface is not that good due to the quality of the digitizing machine. Figure 19(a) shows the model obtained by using Hoppe [3] (an approximation method). Figure 19(b) is the model obtained by using Amenta [2] (an interpolation method). These methods normally require huge amount of data points with good distribution over the surface. So, if the surface has some discontinuities, such as holes, those methods can not be applied. Figure 19(c) shows the model obtained by using Ivrissimtzis [7] (a neural network procedure). The disadvantage of this method, for manufacturing and reverse engineering applications, is that the vertices of the triangular mesh are the approximation of the

actual points which can not be controlled. So some detail of the surfaces and accuracy will be lost. This may be suitable for computer graphics, but for manufacturing and reverse engineering applications, the model created from the points close to the actual points, is required. Another inconvenience of the Ivrissimtzis is that he needs to define parameters for the starting neural networks structure in the loop. It is very difficult for surfaces with holes and other type of discontinuities such as eye sockets. For complex surfaces and the huge amounts of data points needed, those methods will consume a lot of computer time. For our technique, the computer time depends only on the number of points used in the process, not the complexity of the surface, as shown in Table 1. For the teeth model (very complex surface) and the spare part model (simple surface model), both have nearly the same number of points, and used almost nearly the same computer time. In theory, not like the other methods, a lot of computer time is expected for the interpolation, approximation, and iteration, especially for complex surfaces.

Table 1 The run time and the number of points.

Model	Max Edge Length(ρ)	Time(second)	Num. Pts.
Bottle	8.0	17	886
Mouse	5.0	35	2,055
Human face 1	5.0	64	3,157
Human face 2	3.5	720	17,992
Pan handle	2.8	138	5,271
Teeth	1.6	300	7,886
Spare part	2.5	270	6,642
Car seat	3.2	420	9,680

5. Conclusion

We presented a technique for creating triangular mesh from unorganized point clouds. The technique starts from sorting and ordering the unorganized point clouds. The sorted data points will be fed to the Two-level Adaptive Hierarchical Clustering algorithm to organize the structure of the data points. Then, the organized data points are used in the Incremental Triangular Mesh Creation algorithm to create a triangular mesh. We also develop an adaptive self-adjusted connectivity to improve the triangular mesh structure. This algorithm is the utility for surface reconstruction and creating the STL model. We demonstrated our algorithm with several experiments. There are 3 parameters needed for creating an STL model an from unorganized points. These parameters are ρ, θ, and ψ. If a good quality digitizing machine is used for obtained the unorganized points, then only one parameter, ρ, is needed for the organizing and clustering process. A major advantage of our process is that it can handle not just surface complexity but also surfaces with discontinuities. The other advantage of this technique is that it can be extended for cluster computing processes in the case of huge surfaces for reconstruction as illustrated in example 9. The computer time depends only on the number of points used in the process, not the complexity of the surface, as shown in Table 1.

6. References

[1] Edelsbrunner H, Mücke E.P Three-dimensional Alpha Shapes. ACM Transactions on Graphics 13:43-72, 1994.

[2] Amenta N, Bern M, Kamvysselis M A New Voronoi-Based Surface Reconstruction Algorithm. SIGGRAPH'98 Proceedings Conference: 415-421, 1998.

[3] Hoppe H, Derose T, Duchamp T, McDonald J, Stuetzle W Surface Reconstruction from Unorganized Points. SIGGRAPH'92 Proceedings Conference: 71-78, 1992.

[4] Yu Y Surface Reconstruction from Unorganized Points Using Self-Organizing Neural Networks. IEEE Visualization 99 Proceedings Conference: 61-64, 1999.

[5] Mangiameli P, Chen S, West D A Comparison of SOM Neural Network and Hierarchical Clustering Methods. European J. Operational Research Vol. 93: 402-417, 1996.

[6] Fritzke B Kohonen Feature Maps and Growing Cell Structures–A Performance

Comparison. Giles CL, Hanson SJ, Cowan JD. (eds). Advances in Neural Information Processing Systems-5-(NIPS-92), 1993.

[7] Ivrissimtzis I, Jeong W, Seidel H Using Growing Cell Structures for Surface Reconstruction. Proceedings of the Shape Modeling International: 78-86, 2003.

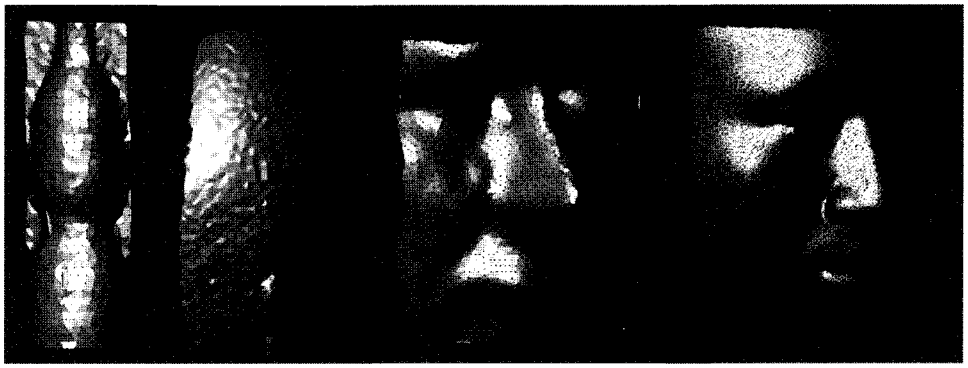
[8] Fritzke B Growing Self-Organizing Networks–why?. ESANN'96: European Symposium on Artificial Neural Networks: 61-72, 1996.

[9] Fritzke B Growing Self-Organizing Network for Unsupervised and Supervised Learning. Technical Report ICSTR-93-026, International Computer Science Institute, Berkeley, 1993.

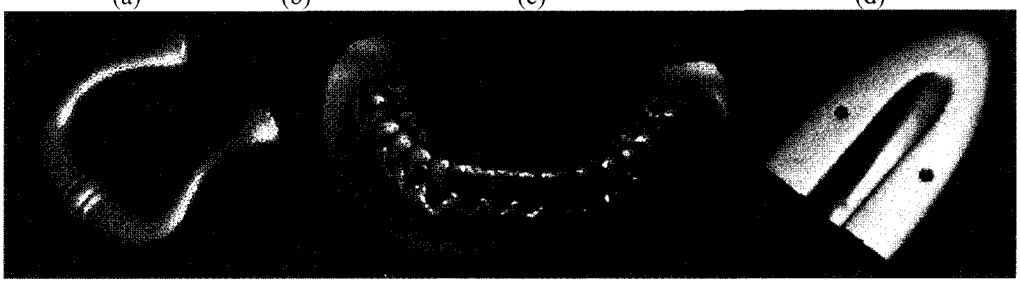
[10] Kawahara S, Saito T On A Novel Adaptive Self-Organizing Network. CNNA'96, Fourth IEEE International Workshop on Cellular Neural Networks and Their Applications: 41-46, 1996.

[11] Sangveraphunsiri V, Uttamang K Development of A 3-D Solid Modeling System Based on The Parasolid Kernel. The 12th International Pacific Conference on Automotive Engineering, IPC-12, Bangkok, Thailand, 2003.

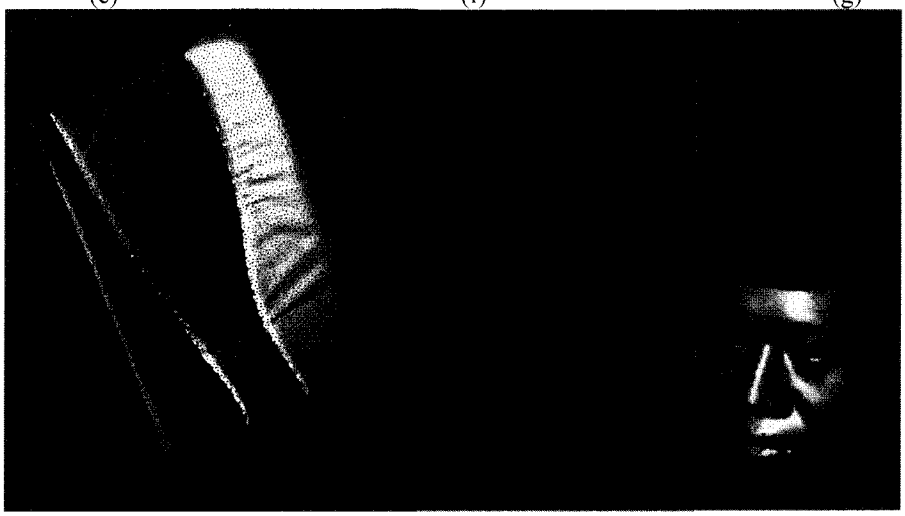
[12] Sangveraphunsiri V Information Technology for Innovation of Manufacturing. Proceedings of Asian Academy Seminar on Advanced Manufacturing System, Hyderabad, India, 2000.



(a) (b) (c) (d)



(e) (f) (g)



(h) (i) (j)

Figure 18 The experimental results

(a) (b) (c)

Figure 19 Works from other techniques