Abstract: - Designing surface models for products used in biomedical applications requires geometric softwares which can design good convex surfaces. Therefore the geometric modellers need to contain modules which can convexify a given grid of points, which in turn can be interpolated by surfaces using known algorithms to obtain almost convex surface. In this paper we explain our algorithm to design a smooth convex surface model for cranial implant fitting the cranial CT data in the least square sense.

Key-Words: - CAD, CAGD, Cranial implant, Least square fitting, Orientation of fitting.

1 Introduction

Computation of convex surfaces plays a vital role in engineering applications such as designing envelope of a car body [6, Bézier], ship hull, body of a soap or any other product. Apart from these convex surfaces are also usefull in designing cranial implants for skulls, helmets for various purposes and other biomedical applications. Therefore geometric modellers requires software modules which can help to get convexification of input grid points (as interpolating any given grid of points by a convex surface may not be possible) and also to search a convex surface taking a non-convex surface as a initial surface. In this paper we present our algorithm to solve the above problems. The plan of our paper is as follows. In section 2, we give problem formulation and motivation for convexification of a given grid of points. In section 3, we explain our algorithm to compute a surface model for cranial implant for grid of cranial CT data points approximating it in the least square sense. The least square fitting condition is imposed on the convexification process so that the shape of the output grid of points is approximately of same shape as its corresponding input grid of points. In section 4, we discuss salient features of our surface model for cranial implant.

2 Problem of Convexification and Modeling of Cranial Implant

In this section we explain the problem formulation of convexification of structured grid of points with the help of a case study from biomedical application. Figure 1 shows a male with a large frontal bone defect [7, Haex et al.] caused due to trauma happened some years ago, which needed to be repaired due to protective and cosmetic reasons. The patient was cured using cranioplastic surgery. Cranioplasty is the procedure of restoring ideal shape of a part of an organ with defective bone shape. In this procedure cranial CT data of the defective part is obtained, and cranial implant, fitting the data, is manufactured which is then fixated on the bone of the defective organ. In order to manufacture a cranial implant we need to get 3D model of it so that it fits cranial CT data. A good fit of cranial implant is necessary to minimize the need of alteration at the time of fixation during surgery. In [8, Carr et al.] designing surface model of cranial implant using radial basis function is explained. Though there are numerous papers on the
designing surface model of cranial implant there is no standard process for it. The problem is treated according to case based specifications. Note that in most of the cases, except in complicated cases like canio-facial deformities, the carnial implant should be modeled as a smooth surface with convex shape fitting the grid of points obtained from cranial CT scan. In the subsequent section we explain our algorithm to get convex grid of points, which best fits the given grid of points for which we get a smooth interpolating surface using BezierPkg[5, Patkar et al.].

Definition 2.1 The grid of point which forms convex polyhedra having all faces of triangular shape is called convex grid of points.

Some times during designing process certain data points is considered to be undesirable for getting a best fit smooth convex surface, giving rise to the problem of data free region. Moreover in cases explained in [8, Carr et al.] cranial implant has to be designed for skull with a hole, for which we have a large data free region. We provide the solution to the problem of data free region in the next section.

3 Designing Surface Model for the Cranial Implant

From Figure 1 we observe that the frontal surface of the skull has deformations and the ideal shape as is evident from Figure 2 should be a smooth convex surface. Thus the structured grid of points obtained from the CT scan will be nonconvex for which one needs to find a smooth convex surface fitting the points. Our strategy is to first find a convex grid of points fitting the given grid of points and then interpolating the convexified grid of points by smooth surface. In order to convexify a given grid of points, what strikes at the first in one’s mind is to shift the co-ordinates of all the points in appropriate directions so that now all the points lie on a convex hull and the grid becomes smooth. We follow the same strategy, as mentioned here, to tackle with this problem of convexification.

Definition 3.1 The grid of point which forms convex polyhedra having all faces of triangular shape is called convex grid of points.
3.1 Convex hull

The smallest convex polyhedra having all faces of triangular shape containing the given grid of points is called its convex hull. We use point insertion algorithm to compute the convex hull of given rectangular array of grid points. The implementation of the algorithm does the following:

First it takes 2 different vertices. Then it finds third vertex such that the 3 vertices are not collinear, or in other words they form a triangle. Again it finds next fourth vertex such that the 4 vertices are not coplanar, i.e. they surely form a tetrahedron. If we don’t get any such 4 vertices it means the given vertices are coplanar and the convex hull for that cannot be found. Once we get 4 such vertices we divide the given vertices into 2 sets. First set, say F, consists the 4 faces of the tetrahedron, and hence the 4 vertices also. Second set, say V, consists of the rest of the given vertices which are not in F.

Now each time it takes 1 vertex from V, say v, and finds out all the visible faces from v. Next all the visible faces are deleted and new faces are created using the 3 edges of each visible face and the vertex v. Out of all these faces each distinct face is then added to set F. This procedure continues until V is empty and finally F consists of all the faces of the convex hull. Thus, we get the convex hull of the given points.

3.2 Parallel sandwiching planes of convex hull

Once we get the convex hull our next aim is to find the direction in which the given set of points is thinnest. That is we have to find the closest pair of parallel planes sandwiching the convex hull.

A plane can touch a convex hull (consisting of triangular faces, linear edges and vertices) of a set of points at a vertex, or on an edge, or on a face. Let us denote the plane touching the convex hull

- at a vertex by $\Pi_v$
- on an edge by $\Pi_e$
- on a face by $\Pi_f$

A pair of parallel planes $\Pi_1$ and $\Pi_2$ sandwiching the convex hull can be of the following types

(P1) one is of type $\Pi_v$ and other of the type $\Pi_v$
(P2) one is of type $\Pi_v$ and other of the type $\Pi_e$
(P3) one is of type $\Pi_v$ and other of the type $\Pi_f$
(P4) one is of type $\Pi_e$ and other of the type $\Pi_e$
(P5) one is of type $\Pi_e$ and other of the type $\Pi_f$
(P6) one is of type $\Pi_f$ and other of the type $\Pi_f$

Note that

- we consider a pair of planes of type P6 only if the pair of faces taken are parallel. Thus pair of planes of type P6 is a special case of the type P5 in which one of the face containing the edge is parallel to a face opposite to the edge.
- we have to consider only pair of skew (parallel) edges for the pair of planes of type P4.

Proposition 3.1 For a pair of vertex and edge the pair of planes of type P2 with minimum distance is of type P3 or P4 or P5.

Proof: To find out the pair of planes of type P2 with minimum distance for a pair of vertex $v$ and an edge $e$ we go through the following process. We take a pair of parallel planes one containing $v$ and the other $e$ we go through the following process. We take a pair of parallel planes one containing $v$ and the other $e$. Then we rotate them parallel to each other fixed to $v$ and and $e$ respectively. In one of the two directions of rotation the distance between the planes keeps monotonically decreasing. The rotation until either one of the planes falls on a face or the plane containing $v$ coincides with an edge skew (parallel) to $e$. Hence the proof.

Proposition 3.2 For a pair of vertex and vertex the pair of planes of type P1 with minimum distance is of type P3 or P4 or P5

From the above analysis we reach to the conclusion which we state as the following theorem.

Theorem 3.1 Let

\[ D_1 = \min \{ \text{distance between the pair of planes of type P1} \} \]
\[ D_2 = \min \{ \text{distance between the pair of planes of type P2} \} \]
\[ D_3 = \min \{ \text{distance between the pair of planes of type P3} \} \]
\[ D_4 = \min \{ \text{distance between the pair of planes of type P4} \} \]
\[ D_5 = \min \{ \text{distance between the pair of planes of type P5} \} \]
type $P4$},
$D5 = \min \{ \text{distance between the pair of planes of type } P5 \}$,
$D6 = \min \{ \text{distance between the pair of planes of type } P6 \}$.

Then we have $\min\{D1, D2, D3, D4, D5, D6\} = \min\{D3, D4, D5\}$.

Remark 3.1 After having formulated the above theorem when we searched the net for the purpose of literature survey in the related topics we found the same result has been stated in a paper [9, Thompson and Crawford] concerning a field very much unrelated to our area of investigation. Theorem in the paper is stated as follows.

Theorem 3.2 Let two parallel planes, $\Pi_1$ and $\Pi_2$ be as stated above, then either

1 One of the planes ($\Pi_1$ or $\Pi_2$) must be coincident with a face of the convex hull (as opposed to vertex or edge) when the distance between the planes is at a minimum,

or

2 each of the planes $\Pi_1$ and $\Pi_2$ must contain exactly one edge of the convex hull when the distance between the planes is at minimum, and the edges in $\Pi_1$ and $\Pi_2$ may not be codirectional.

3.3 Projection of points on a sphere

We first make the given mesh perpendicular and almost symmetrical about original $z$-axis using an affine transformation. Then we project the points on one of the parallel planes and using the projected points we find a sphere on which we compute a new set of $m \times n$ rectangular array of points which will be convex.

3.3.1 Finding the appropriate affine transformation

Let $\Pi_1$ and $\Pi_2$ be 2 parallel planes sandwiching the convex hull.

Now we find the affine transformation $T$ which can translate the center of mass of the given points to the origin and make normal to the planes $\Pi_1$ and $\Pi_2$ parallel with $z$ axis. This affine transformation is then applied to the position vectors on each of the given points. This makes the given mesh perpendicular and almost symmetrical about original $z$-axis.

3.3.2 Projection of the mesh on $XY$ plane

After the mesh is made horizontal and approximately symmetrical about $z$-axis, here we make the $z$-coordinates of all the points zero, which in turn gives us the projected mesh or set of points lying on the $XY$ plane.

3.3.3 Finding least-square-fitting lines for each row and column of the projected mesh.

As mentioned in the introduction our aim is to find a convex patchwork of Bezier patches approximately interpolating the given grid of points. As we observed it is not possible to find such a patchwork for all grid of points we are in the process of finding a new grid of points approximating the grid of points that we have got by affine transformation of the given grid of points in some sense. We will use inverse affine transformation on this grid of points to get the grid of points approximating the given grid for which we compute convex pathwork.

The necessary condition for the patchwork to be convex is that each row and column of grid points should form a convex polygon (in order of their indices). There exist many methods to find a grid with such property. We present our method, which is computationally efficient, to find such a grid. The idea is to raise a planar mesh of points whose each row and column of points are collinear onto an appropriate spherical cap in the directional perpendicular to the plane. In this subsubsection we compute the planar mesh whose each row and column of points are collinear and in the next section we will find an appropriate spherical cap onto which we raise the planar mesh.

For each row of points of the planar grid of points $\{(x_{i,j}, y_{i,j})\}_{i=0}^{n},_{j=0}^{m}$ we find a line which best fits them using least square method. Similarly we find for each column of points a best fitting line. Thus we have $n$ lines along the row and $m$ lines along the columns. The $n$ lines along the row intersects with $m$ lines along the column to form $n \times m$ array of
points \( \{(\tilde{x}_{i,j}, \tilde{y}_{i,j})\}_{i=0,j=0}^{n,m} \) on \( XY \) plane whose each row and column of points are collinear.

3.3.4 Lifting the above obtained planar mesh to an appropriate spherical cap, whose center is on \( z \) axis

We now compute an appropriate spherical cap such that upon raising the computed planar mesh on it approximates the affinely transformed grid in the sense that the sum of the squares of the difference between the heights of new point and its corresponding previous one is minimum.

In order to achieve our goal first of all, we consider a sphere \( S = 0 \) with its center at \((0, 0, \delta)\) and radius \( r \) are not known. Substituting the value for each \((\tilde{x}_{i,j}, \tilde{y}_{i,j})\) in the equation of the sphere, defined above, we get the value for \( \tilde{z}_{i,j} \) in terms of \( \delta \) and \( r \). Next we find the appropriate sphere by finding \( \delta \) and \( r \), which minimizes the following function,

\[
\sum_{i} \sum_{j} (z_{i,j} - \tilde{z}_{i,j})^2
\]

subject to

\[
\tilde{x}_{i,j} = x_{i,j}, \tilde{y}_{i,j} = y_{i,j}
\]

The problem of fitting a curve or surface to data plays important role in many applications like computer graphics, engineering, statistics, metrology, astronomy, reflectometry, etc. A common technique for fitting the surface to the data is orthogonal distance regression. But this technique is not appropriate for the applications where the fact that the probe of a coordinate measuring machine makes contact with the part moves in a certain direction is to be taken into consideration [1, Watson], [2, Watson], [3, Watson], [4, Hulting]. We now explain our analysis for the problem of computing least square fit sphere along \( Z \)-direction.

We deduce one dimensional minimization problem for the two dimensional minimization problem (1). From the constraint equation (1) we get

\[
\tilde{z}_{i,j} = \delta + \sqrt{r^2 - x_{i,j}^2 - y_{i,j}^2} \quad \forall i, j
\]

We get the cost function as a function of \( \delta \) and \( r \), that is

\[
\text{cost}(\delta, r) = \sum_{i} \sum_{j} (\delta + \sqrt{r^2 - x_{i,j}^2 - y_{i,j}^2} - z_{i,j})^2
\]

Next we find the appropriate sphere by finding \( \delta \) and setting it to 0 we get,

\[
0 = \frac{\partial \text{cost}(\delta, r)}{\partial \delta} = \sum_{i} \sum_{j} 2(\delta + \sqrt{r^2 - x_{i,j}^2 - y_{i,j}^2} - z_{i,j})
\]

\[
\Rightarrow \delta = \frac{\sum_{i} \sum_{j} z_{i,j} - \sqrt{r^2 - x_{i,j}^2 - y_{i,j}^2}}{\sum_{i} \sum_{j} 1} \frac{1}{1}
\]

Substituting the value of \( \delta \) we get the cost function as

\[
\text{cost}(r) = \sum_{i} \sum_{j} \left( \frac{\sum_{i} \sum_{j} z_{i,j} - \sqrt{r^2 - x_{i,j}^2 - y_{i,j}^2}}{\sum_{i} \sum_{j} 1} + \sqrt{r^2 - x_{i,j}^2 - y_{i,j}^2} \right)^2
\]

\[
= \sum_{i} \sum_{j} \left( \frac{r^2 - x_{i,j}^2 - y_{i,j}^2}{(n+1)(m+1)} \right)^2
\]

Clearly \( r \) should be such that \( r \geq \sqrt{x_{i,j}^2 + y_{i,j}^2} \) \( \forall i, j \). We can use any of the known methods viz. gradient search methods, pattern search methods to minimize (4). We have implemented Hookes and Jeeves pattern search algorithm for the purpose and found it to be fairly fast with initial guess as \( r_0 = \max_{i,j} \{ \sqrt{x_{i,j}^2 + y_{i,j}^2} \} \). We finally get convexified grid of points \( \{(x_{i,j}, y_{i,j}, z_{i,j})\}_{i,j} \) for affinely transformed grid of points by projecting \( \{(x_{i,j}, y_{i,j})\}_{i,j} \) on the sphere \( S = 0 \) along \( Z \)-direction.

Now we apply the inverse of the affine transformation \( T \) to the points \( (\tilde{x}_{i,j}, \tilde{y}_{i,j}, \tilde{z}_{i,j}) \) to get the convexified grid of points for the input grid of points. This grid of points is convex and each row and column are convex and coplanar.

In this section we explain our strategy to deal with the problem of data free region that arise during the designing of surface model of cranial implant.

3.4 Solution to the problem of data free region

In this section we explain our strategy to deal with the problem of data free region that arise during the designing of surface model of cranial implant. Suppose we do not have points \( \{P_{i,k} : i \in \{k_1, ..., k_n\} \} \)
Figure 3: Interpolation of given non-convex grid of points done by BezierPkg

in the $k^{th}$ row. In the module which computes planar mesh, explained in section 3.3.3, for $k^{th}$ row we find best fit line for $\{(x_{i,k}, y_{i,k}) : i \notin \{k_1, \ldots, k_n\}\}$. Similarly we best fit line for all the columns. Thus we get planar mesh $\{(\tilde{x}_{i,j}, \tilde{y}_{i,j})\}$ according to the procedure explained section 3.3.3. Now for computation of best fit spherical cap we omit $(x_{i,j}, y_{i,j}, z_{i,j})$ for the points corresponding to the data free region. Once the spherical cap is computed we project all $(\tilde{x}_{i,j}, \tilde{y}_{i,j})$ onto it to get a convexified grid of points which doesn’t have any data free region.

3.5 Interpolation of the convexified grid of points

A smooth surface for the interpolation of convexified grid of points is computed using BezierPkg [5, Patkar et al.]. The computed surface is a smooth patchwork of bicubic Bézier surfaces. The surface of this form is most suitable for CNC milling. We discuss the smoothness and convexity preserving property of the surface in the next section.

4 Salient Features of the 3D Model of Cranial Implant

Our model for cranial implant has three salient features.

4.1 Minimum height

The surface computed as a 3D model for cranial implant lies inside the slab of minimum thickness. Module to find parallel sandwiching planes (section 3.2) find the parallel planes of minimum distance containing the convex hull of the given grid of points. Subsequently, in section 3.3.4, computation of least square fit spherical cap with direction of fit taken along the direction normal to the parallel sandwiching planes (instead of radial direction) ensures that the cap lies between the parallel planes. Then the convexified grid of points obtained by projecting the planar mesh on the spherical cap also lies between the parallel planes. Hence the surface interpolating the convexified grid of points, computed by BezierPkg also lies between the pair of parallel planes of minimum height. From Figure 1 and ref-skill2 importance of this feature is evident. If the cranial implant creates a bulge, the contour of the skull may not get appropriately re-established.

4.2 Orientation of the fitting

The direction of fitting the surface is computed taking the shape of given grid of points into consideration as against the procedure in which the surfaces are computed as functional surface fixing the z-direction apriori. Grooves created on the cranial
implant as well as drilled on the bone, along this direction, for the Titanium screws for fixation of cranial implant. Thus computation of the orientation for fitting the surface is plays important role for designing of cranial implant.

4.3 Smoothness and convexity preserving interpolation surface

We now discuss convexity preserving property of smooth surface generated by our algorithm explained in the [5, Patkar et al.] for the convexified grid of points. In the algorithm the surface is computed as a patchwork(uniformly parametrized in $u$ and $v$ directions) of bicubic Bézier patches. Boundary control points of each patch is computed using cubic Hermite interpolation of each row and column of points with two points each neighbouring to the end points chosen heuristically. Thus for each row and column of points we compute patchwork of Bézier curves. Now since each row and column of convexified grid of points are coplanar and convex, the two control points between each pair of adjacent grid points(computed by Hermite interpolation) lies on the plane of their corresponding rows or column. Control polygons of each Bézier curve is convex for almost all the case except when corresponding row or column is highly unevenly spaced. Also inner control points computed for each Bézier patch using parallelogram law do not cross unless four adjacent grid points form a quadrilateral whose

- one pair of opposite sides is such that one side considerably longer than the other and
- length of the other pair of opposite sides is considerably smaller.

Now since the patchwork computed is $G^1$ continuous, across the patch convexity condition for adjacent patches is automatically maintained. Thus the interpolation surface computed byBezierPkg will be convex for almost all practical purposes. Analysis of smoothness property of interpolating surface is discussed in [5, Patkar et al.] Thus we observe that algorithm for convexification of grid of points is such that the convexified grid of points can also be conveniently used by other known softwares to create a suitable model for cranial implants.

References


