International Journal of CAD/CAM Vol. 6, No. 1, pp. 105~115 (2006)

INTERNATIONAL JOURNAL OF CAD/CAM
www.ijcc.org

Voxel-Based Thickness Analysis of Intricate Objects

K. Subburaj, Sandeep Patil and B. Ravi*
Department of Mechanical Engineering, Indian Institute of Technology, Bombay, India

Abstract – Thickness is a commonly used parameter in product design and manufacture. Its intuitive definition as the smallest dimension of a cross-section or the minimum distance between two opposite surfaces is ambiguous for intricate solids, and there is very little reported work in automatic computation of thickness. We present three generic definitions of thickness: interior thickness of points inside an object, exterior thickness for points on the object surface, and radiographic thickness along a view direction. Methods for computing and displaying the respective thickness values are also presented. The internal thickness distribution is obtained by peeling or successive skin removal, eventually revealing the object skeleton (similar to medial axis transformation). Another method involves radiographic scanning along a viewing direction, with minimum, maximum and total thickness options, displayed on the surface of the object. The algorithms have been implemented using an efficient voxel based representation that can handle up to one billion voxels (1000 per axis), coupled with a near-real time display scheme that uses a look-up table based on voxel neighborhood configurations. Three different types of intricate objects: industrial (press cylinder casting), sculpture (Ganesha idol), and medical (pelvic bone) were used for successfully testing the algorithms. The results are found to be useful for early evaluation of manufacturability and other lifecycle considerations.

Keywords: **CAD/CAM, Medial Axis Transformation, Thickness, Radiography, Voxels**

1. Introduction

'Thickness' is a commonly used term in product design and manufacture. It is often mentioned qualitatively as thin or thick wall/section. In dictionaries, thickness is defined as the minimum distance between two (opposite) surfaces of an object. This applies well to a rectangular block, implying its smallest dimension, but a tapered object may have different thickness values at the two ends. The definition yields ambiguous results for intricate shapes with corners, junctions, fillets, and freeform surfaces.

The interpretation and application of thickness value also depends on the context. While dealing with an entire object, thickness usually refers to the smallest or the largest overall dimension. This is required for applications such as process planning, materials requirements planning and packaging, which are driven by the overall dimensions of the object. For functionality analysis, thickness of individual features (wall, rib, boss, etc.) is important and useful for calculating stresses and strains. For manufacturability analysis (for example, in design for casting and molding), thickness at various points within the part are needed, since it influences mold filling and solidification. Thin sections are difficult to fill, leading to defects such as cold shuts and weld lines. Isolated thick sections are difficult to feed (compensation of solidification shrinkage) leading to shrinkage porosity and sink marks [19].

In most design for manufacture (DFM) applications, the user has to manually enter thickness values. A few CAD programs provide simple tools for thickness measurement and visualization in terms of cross section and point-to-point distance. Some programs enable local thickness measurement using surface normal and inscribed sphere. There is a need for one or more appropriate definitions and corresponding methods for the determination of thickness for various requirements, independent of the solid model representation scheme and its shape complexity. This has been taken up in the present work.

The next section briefly reviews related work by other researchers. Then we present our definition(s) of thickness, and methods for its computation. Voxel-based representation and display scheme used in our approach are described next. This is followed by a description of the implementation of the thickness computation algorithms and its testing on a few intricate parts. The paper concludes with the highlights of the work and future scope.

2. Related Work

Several direct and indirect methods have been proposed for thickness determination, applicable to 2D and 3D shapes. We restrict our discussion to 3D shapes. Direct methods include normal ray tracing, in which a ray is passed normal to the surface of interest, and its intersection with the opposite surface is computed to obtain the thickness value. Indirect methods such as

*Corresponding author:
Tel: +91-22-2576-7510
Fax: +91-22-2572-6875
E-mail: bravi@me.iitb.ac.in

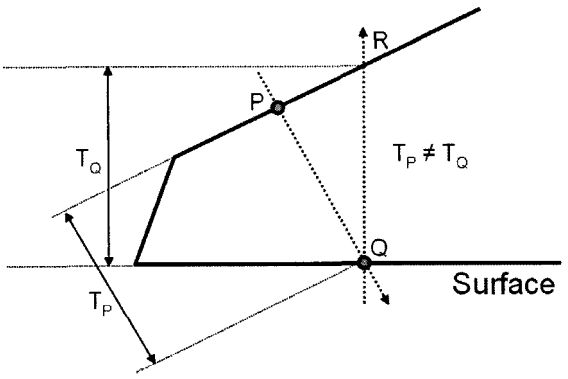


Fig. 1. Thickness determination by normal ray tracing.

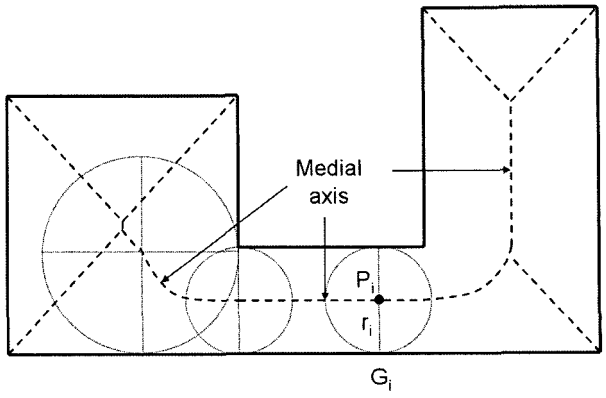


Fig. 2. Medial axis transformation.

Medial Axis Transformation (inscribed sphere) and distance transformation (object thinning) have received relatively more attention. All these methods are briefly described here.

Normal ray tracing is available in a few CAD/CAM programs, and is used to determine the thickness value at any point on the surface of an object. A ray is shot from a given point P on the surface in a direction opposite to the local outward surface normal, to intersect the immediately opposite surface of the object, assuming it is 'water tight'. The Euclidean (straight line) distance between the intersections is considered as the thickness at the given point (Fig. 1). However, if the two surfaces containing the intersection points P and Q are not parallel, then the thickness value computed at the two points will be different depending on the starting point (or ray direction), leading to conflicting results for the same pair of points.

Medial Axis Transformation (MAT) generates thickness information in terms of inscribed spheres. The medial axis for a 3D enclosed shape is defined as the locus of centers of maximal inscribed spheres [15]. It is a function of radius ri, which is given by the distance from a point P_i on the medial axis to the governing point G_i on the object surface where the corresponding sphere touches the surface. The medial axis can comprise surfaces, curves and points depending on the order of contact of the maximal spheres with the object surface [5]. It effectively represents the shape of the object, and has been used in many applications like finite element mesh generation [17], model simplification [7], solidification simulation [16], and feature recognition [11]. The thickness is directly available in terms of the distance of medial axis from the object boundary. However, the sub-branches (leading to the main axis) that are generated during the computation are unnecessary for thickness analysis and need to be removed.

The medial axis can be computed using Voronoi diagram [1] or its dual, Delaunay triangulation [20]. In convex solids, Voronoi diagram is equivalent to the medial surface of a polyhedral object [2]. The Voronoi diagram can also be used to compute the medial axis from a set of sampled points representing the object surface [3] and volumetric data [4]. The medial axis is obtained by removing unnecessary edges from Voronoi diagram which do not belong to medial axis and adding certain edges that do belong to medial axis but are absent from the Voronoi diagram, especially at concave corners [18].

Distance Transformation is a method of generating the skeleton of an object which stores the minimum distance from the object surface. Huang et al. described the abstraction of an object skeleton using distance field representation based on true Euclidean distance [6]. In another work also, distance field transformation was used to simplify the complex polyhedral model into an abstract representation [14]. Lu et al. used distance transformation for thickness evaluation and identification of mass concentration regions for die castability evaluation [12]. For this purpose, they used a voxel model with a resolution of 128 voxels per axis represented as a bit array. Jones et al. explored space filled distance fields using volumetric data of object, and compared the computational time and accuracy of various distance transformation methods [8]. They also used a low resolution voxel model (60 voxels per axis).

Thinning was originally used for object abstraction and recognition from images in computer vision applications, such as locating defective parts during automated manufacturing and inspection. This can be extended to 3D to produce an object skeleton similar to MAT. The difference is that the sub-branches at the ends are absent, and the thinned skeleton is complete even at concave corners (Fig. 3). Thus thinning appears to be a better approach for thickness analysis compared to other methods, including MAT. Object thinning is achieved by successive elimination of the outermost surface or skin of the object, while preserving the topology, until the skeleton is left [10]. The resulting skeleton usually lies along the medial surface of an object and has been used for geometric feature recognition, such as junctions. Another application reported in literature is for automatic suggestion of feeder location for casting [13].

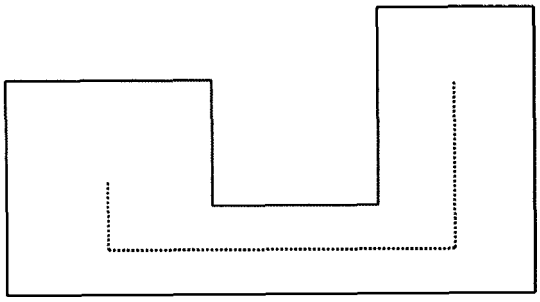


Fig. 3. Object skeleton generated by thinning operation.

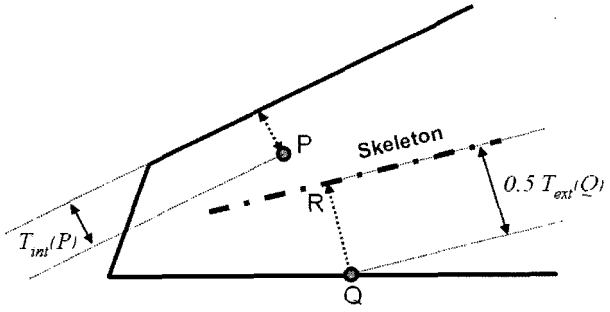


Fig. 4. Interior and exterior thickness.

In summary, a few methods are available for direct or indirect computation of thickness. The medial axis representing a set of maximal inscribed spheres can be computed using Voronoi diagram or Delaunay triangulation. It is however, difficult to compute for intricate shapes, produces unnecessary sub-branches at the ends, and does not work well near concave corners. The distance transformation and 3D thinning appear to be better approaches, but have been implemented using low-resolution voxel models, leading to large errors in thickness values. Voxel based approaches however, promise robust computational algorithms that can be used for objects with any shape and complexity [9]. There is a need for high resolution voxel modelling (more than 500 per axis) for better approximation of object shape and more accurate results. There is also a need to explore alternative yet generic definitions of thickness, computation methods and post-processing techniques for meaningful display of thickness information. These have been taken up in the present work and are described next.

3. Thickness Definition and Computation

Three different definitions of thickness are presented here: one for points inside the object, second for points on the object surface, and a third one for radiographic thickness along a view direction. The methodology for computing various thickness values are also described.

3.1 Thickness definition

Interior thickness at a point P_i inside an object is defined as the minimum distance of P_i from the nearest surface S_j of the object (Fig. 4). Its value can be obtained by growing a sphere or by firing rays from the point along all directions towards the object surface. The shortest ray length (distance between point and surface) gives the thickness at that point.

$$\textit{Interior thickness } T_{int}(P_i) = min\ (dist(P_i, S_j)) \qquad (1)$$

The locus of points with the highest local internal thickness values defines the object skeleton.

Exterior thickness at a point Q_i on the object surface S_j is defined as twice the distance to the nearest point R_k on the object skeleton.

$$\textit{Exterior thickness } T_{ext}(Q_i) = 2\ max\ (T_{int}(R_k)) \qquad (2)$$

As an interior point P_i moves towards the object skeleton, its internal thickness value increases and attains the highest value at reaching the skeleton. On the other hand, if the point moves toward the object surface, its interior thickness value gradually reduces to zero at the surface, but its exterior thickness value (which is defined only on the surface) will have a non-zero value.

Radiographic thickness at a point Q_i on the object surface S_j along a view direction XY is defined as the distance from Q_i to the intersection point of the ray with the opposite surface. If there are no other intersections of the ray with the object along this direction, then the above distance represents the ***total*** radiographic thickness. The total thickness for a set of multiple sections along a view direction will be the same as that for a single section with equivalent total thickness, yielding the same result as seen from a view direction. This limits the usefulness of radiography to interpret internal features and their thickness, which can be overcome by defining two other types of radiographic thickness: ***minimum*** and ***maximum*** (Fig. 5). In general, let there be $N_{i,int}$ intersections ($N_{i,int} > 2$), and the points of intersections be given by $R_{i,u}\{u = 1\ to\ N_{i,int}\}$. Then the three types of radiographic thickness are defined as follows.

$$\textit{Minimum radiographic thickness } T_{rmin}(Q_i) = min(dist(R_{i,u}, R_{i,u+1})) \qquad (3)$$

$$\textit{Maximum radiographic thickness } T_{rmin}(Q_i) = max(dist(R_{i,u}, R_{i,u+1})) \qquad (4)$$

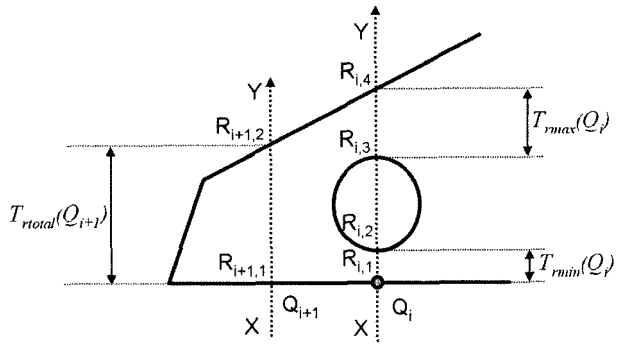


Fig. 5. Radiographic thickness.

Total radiographic thickness $T_{total}(Qi) = \sum dist(R_{i,u}, R_{i,u+1})$ (5)

3.2 Thickness computation

Different approaches are proposed for computing the interior, exterior and radiographic thickness values. The interior thickness distribution of an object is computed using successive skin removal method, and displayed on a desired cross-section of the object. To compute the exterior thickness at a point on the object surface, the nearest point on the object skeleton (obtained from skin peeling) is identified and its thickness value assigned to the point on the surface for display. The radiographic thickness values (minimum, maximum and total) are also computed for points of the surface of the object, by passing rays parallel to the view direction, and computing their intersections with the object.

For all approaches, the object is represented as a voxel model. The accuracy of thickness values is affected by the voxel resolution. For example, a typical engine block of size 600 mm, and wall thickness 8 mm, can have an error of 25-50% in thickness computation if a voxel resolution of 100 per axis (voxel size = 6 mm) is employed. On the other hand, a very high voxel resolution will require large memory sizes, and will be computationally inefficient. Our goal is to achieve reasonable accuracies (less than 10% error) for typical engineering parts, within a few seconds to minutes, on a standard desktop computer (Pentium 4 CPU, 1 GB RAM). The thickness computation algorithms are described here.

Peeling or successive skin removal is analogous to several physical phenomena like fire burning, decay by sublimation, and progressive solidification of pure metals. Each skin removal will expose a new surface inside the object, which is at a uniform distance from the original surface of the object. The meeting point, line or surface of the fire, decay or solidification fronts from opposite directions define the innermost core or *skeleton* of the object.

Let O be the object in Z^3. Let $B_S(P)$ denote the skin of object centered at $M_A(P)$ of plane P. We define the ordered set D (O) by

$$D(O) = \{B_S(P) | B_s(P) \subset O\}$$

Here D(O) is ordered by the set inclusion; that is, D(O) is the set of all skins contained in O. The I(O) of domain O is the set of all maximal elements in D(O), that is,

$$\{I(O) = B_r(P) \in D(O) | B_s(P) \in D(O) \text{ and } B_r(P) \subset B_s(P) \text{ implies } B_r(P) = B_s(P)\}$$

where, $B_r(P)$ is a *maximal skin.* It implies maximum distance from boundary, if $B_r(P) \in I(O)$.

Medial axis or surface of O is the set of the ordered skins of P in I(O); that is,

$$M_A(P) = \{P \in O | B_S(P) \in I(O)\} \quad M_S(O) = \{M_A(P) | M_A(P) \in O\}$$

The fire fronts can be modeled as offset surfaces, and the peeling can be simulated using Constructive Solid Geometry (CSG) operations on a Boundary Represented (B-Rep) model. In this, every facet/face is offset through a small distance toward the interior and new surfaces of the object are calculated. This is repeated for several steps until the skeletal line or surface of the object. This can give accurate results for simple polyhedral objects, if the offset distance is small, but is extremely difficult and inefficient for complex objects. Voxel-based object models allow more robust and faster computation.

For a voxel-based model, the surface voxels represent the object skin. A surface voxel is one that has at least one exposed face (missing neighbour) among the six faces. These voxels are reset as empty voxels during an iteration to remove the skin. This is repeated until the innermost voxels are left, representing the object skeleton. The successive skin removal for a sphere is shown in Fig. 6. The thickness of the peeled skin in a diagonal direction is slightly different (could be either more or less) from that along the coordinate axes along which the voxels are aligned.

The peeling process gives valuable information about the interior of the object in terms of the distance of every interior point from the closest surface of the object. This interior thickness distribution can be obtained during the peeling process by marking the voxels in each iteration with the corresponding thickness value. Thus, after removing the N^{th} skin, the newly exposed

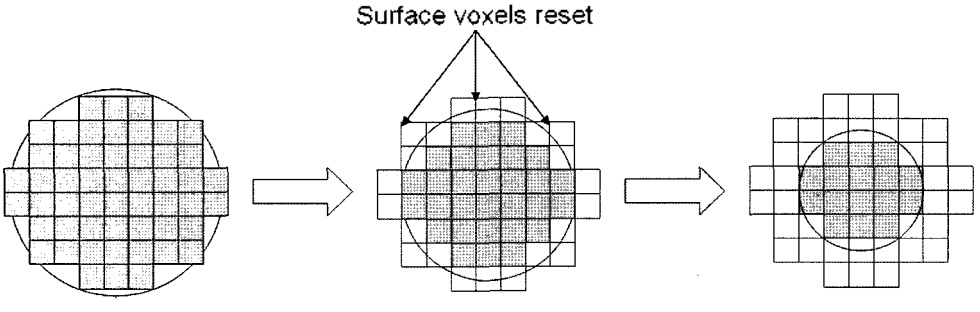


Fig. 6. Successive skin removal for a sphere.

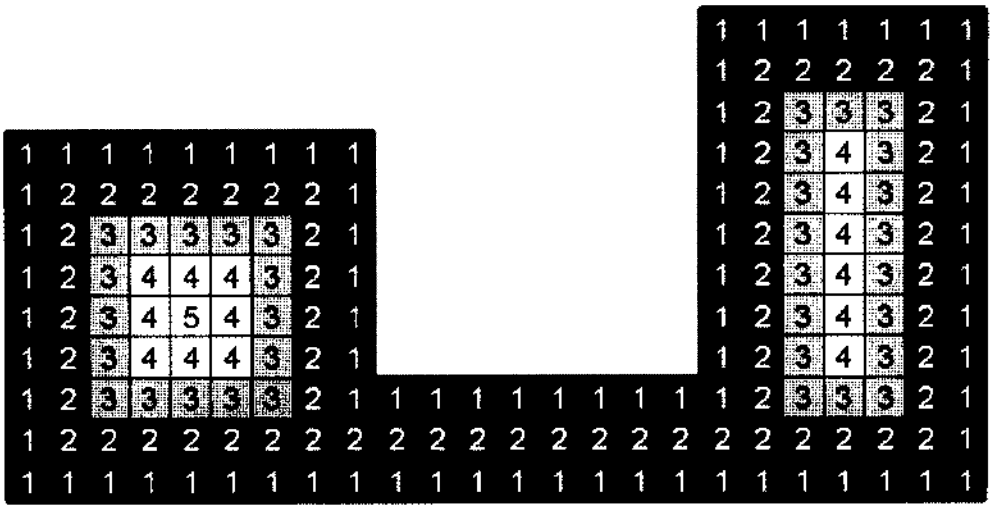


Fig. 7. Internal thickness distribution of a simple voxel model.

surface voxels on skin (N + 1), will be at a distance from the object surface given by $N \times C$, where C is the cell size. The internal thickness distribution in terms of the distance of each voxel from the object surface, for a simple shape is shown in Fig. 7.

The object skeleton represents the set of internal points with the highest local thickness value. To determine this set of points, each voxel v_i with internal thickness $T_{int,i}$ in the object model is tested as follows. The neighbors of v_i in 26 directions are identified $\{v_j \mid j = 1 \text{ to } 26\}$, and their internal thickness value $T_{int,j}$ compared with $T_{int,i}$. The following conditions arise:

1. if $T_{int,j} > T_{int,j}$ *then* $v_i = v_j$
2. if $T_{int,j} = T_{int,j}$ *then* $v_i \in M_S$ and $V_i = v_j$
3. if $T_{int,j} > T_{int,j}$ *then* $v_i \in M_S$

In other words, if the thickness at the test point is the highest among the neighbors (condition 3), then the test point is added to the skeleton. If the thickness at a neighbor is either higher (condition 1) or the same (condition 2) as that of the test point, then the test is repeated at the neighbor point. In the case that multiple neighbors satisfy the first or the second condition, then the first such neighbor is selected for repeating the test. This does not affect the final result, and yet has the benefit of reducing the computation time.

Radiography gives the total interior thickness of a 3D object along a view direction captured in a gray scale image. The gray scale intensities (low values are dark, higher values are lighter) in the image are proportional to the thickness of the object encountered along the view direction, normalized with respect to the voxel resolution (maximum possible thickness). As defined earlier, three different radiographic thickness values are computed: minimum, maximum and total, for all points on the surface along a view direction, yielding three different types of images. The methodology is described here.

The rays are passed through the entire object along one of the preferred view directions (usually one of the coordinate axes), and their intersections with the object are computed and stored in memory. If two consecutive (odd to even numbered) intersection points are R_1 at (x_1, y_1, z_1) and R_2 at (x_2, y_2, z_2), where $\{R_1, R_2 \in O | O \in Z^3\}$, then the Euclidean distance between R_1 and R_2 given by $\sqrt{((x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2)}$ indicates the thickness between the intersection points. If there are only two points of intersection along a ray, then the above distance represents the total radiographic distance. If there are more than two intersections, then the minimum and maximum radiographic thickness are also computed.

The radiographic thickness values can be easily computed for a voxel model. If the model has a resolution of R voxels/axis, then R^2 rays are passed through the model along the view direction, and the number of object voxels between each set of odd-to-even intersection are counted and stored. The thickness of each section of the object (between the intersections) is given by the product of the number of voxels and the voxel size. The radiographic thickness mode (minimum, maximum or total) decides the type of section thickness value considered for generating the image (Fig. 8). For the minimum radiographic thickness mode, the intensity of a surface voxel is proportional to the thickness of the smallest section encountered along the ray passing through the surface voxel. Similarly, the intensity of the surface voxel for the maximum radiographic thickness mode is proportional to the thickness of the longest section. If there are only two intersections along a ray, then the intensity of the surface voxel is the same for all three modes of radiographic thickness.

The percentage error in thickness computation using a voxel-base model is given by:

$$T_{error} = 100 \times (T_{actual} - T_S) / T_{actual}$$

where

T_{actual} = *Actual thickness of object between two successive intersections (odd-to-even)*

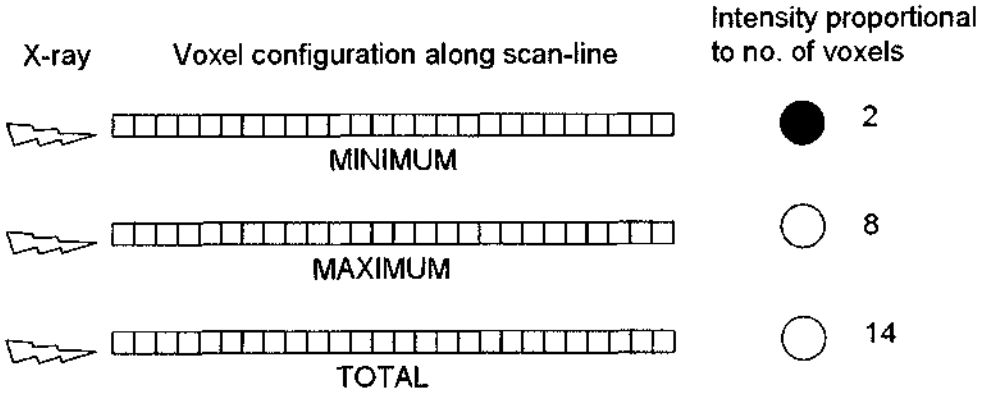


Fig. 8. Modes of Radiographic scan.

T_S = Thickness of section in terms of voxels = $N_{VS} \times C$
N_{VS} = Number of voxels belonging to the section
C = Voxel size

4. Implementation

While B-Rep models are still preferred for engineering parts, the use of voxel-based modeling is gaining ground to handle complex, free-form objects such as art and medical models. The voxel models are also useful for developing geometric reasoning functions, since they contain direct information about the interiors of objects. An additional advantage of voxel-based models is that they are better capable of representing objects with multiple materials, functional gradients, and manufacturing defects (such as voids). This however requires more work in developing efficient algorithms for representing, displaying and analyzing voxel-based models, and the present investigation is a step in this direction.

In the light of the above, the thickness analysis functions have been implemented for voxel-based models. For this purpose, tessellated Boundary-represented models (in STL format) are first converted into voxel models using a voxellisation algorithm, and rendered using a special display algorithm developed in this work. A layer based bit array representation scheme has been employed to store the voxel models. This has been tested for resolutions up to 1000 voxels per axis (maximum 1 billion voxels). The source of the STL files is described first, followed by the algorithms for voxellisation, voxel model storage, and display.

The STL files, which essentially comprise a number of contiguous triangular facets (stored in terms of their vertex coordinates) representing the object surface, are obtained in different ways depending on the type of object. Industrial parts (such as the press cylinder casting used as one of the test objects) can be obtained by exporting from a CAD program (Autodesk Inventor, CATIA, Pro-Engineer, SolidWorks, Unigraphics NX and others). Intricate art and sculptured objects (such as the Ganesha used as another test object) can be modeled using a haptic-based system like Freeform (Sensable Technologies). Models of existing parts can be captured by reverse engineering techniques including 3D laser scanning. Medical models, including the pelvic bone used as the third test object, can be reconstructed from a set of consecutive CT/MRI scan images, using appropriate programs such as MIMICS (Materialise) and 3DDoctor (Able Software).

4.1 Voxelization

The STL model of the object is converted into a voxel model using a ray tracing algorithm. The largest dimension of the bounding box of the object is used to decide the voxel size.

Voxel size = largest dimension of the bounding box / Resolution

The resolution R is decided by the user before voxellisation, and sufficient computer memory is allocated at run time. A set of R^2 rays is passed through the object body along one of the coordinate axes. The rays are aligned with the centers of voxels in the grid plane normal to the chosen axis (Fig. 9). The intersections of each ray with the object facets are computed. An odd number of intersections along any ray indicate an error in the STL file or a very thin section (one voxel thick).

One of the best ways to reduce voxellisation errors is to voxellise from all three coordinate axis directions,

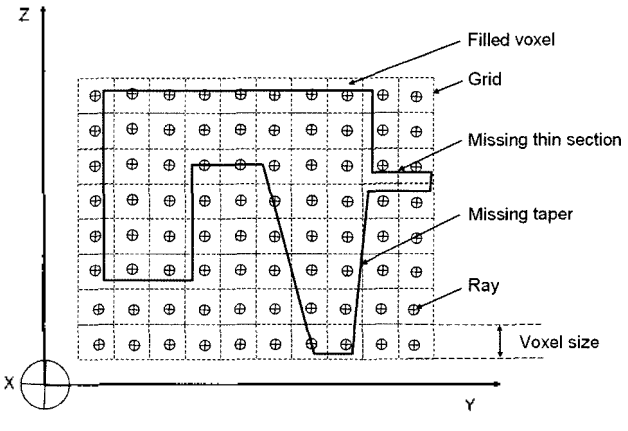


Fig. 9. Voxellisation scheme and some potential errors.

followed by polling for each voxel. Ideally, all three directions should give the same result for each specific voxel (filled or empty). The discretisation error is given by the maximum distance between object surface and the surface of the nearest voxel, which will be less than the voxel size. A higher resolution will obviously reduce the discretisation error. To eliminate the possibility of thin sections disappearing during voxellisaton, the voxel size must be less than half of the minimum section thickness.

In this context, it must be noted that the STL file itself is an approximation of the original object surface created in a CAD system. This is also true for STL files obtained from reverse engineering. The STL file of a medical model obtained after reconstruction from CT/MRI images will also differ from the original tissue depending on the resolution of the scanner (pixels per image), and distance between the scans. Further discussion or handling of such errors is beyond the scope of this work.

4.2 Voxel model storage

The voxel model of an object is a collection of uniformly arranged, axis aligned cubical binary voxels. The voxel volume is split into a number of layers and each layer is stored as an array of bits. There is a direct mapping between the location of a voxel in a given plane and its index in the corresponding array.

The object space O is used to represent a geometry stored as a collection of points in space Z^3. $Z^3 = \{v = (i,j,k) \mid (i,j,k) < (I_{max}, J_{max}, K_{max})\}$, in which every voxel v is represented by its centroid (i,j,k), and has the value of either 0 (empty) or 1 (filled). An object of interest in Z^3 is represented by a non-empty subset O consisting of all the points with value 1. The component of O,O consists of all the points with the value of 0. A voxel $v = (i,j,k)$ in O is called a surface voxel if at least one of its 6-neighbors is in $\overline{O}$. To eliminate any interpretation errors and improve computation speed, all boundary voxels of Z^3 (*i.e.*, $\{(i,j,k) \mid i = 0$ or $I_{max}, j = 0$ or $J_{max}, k = 0$ or $K_{max}\}$) are assumed to belong to $\overline{O}$ (Fig. 10).

The total memory required (M_{Total}) for a voxel cube of resolution R is given by:

$$M_{Total} = (M_{POINTER} + M_{PLANEARRAY}) \times N_P = (4bytes + [(R+2)^2 / 8]bytes) \times (R+2) \quad (8)$$

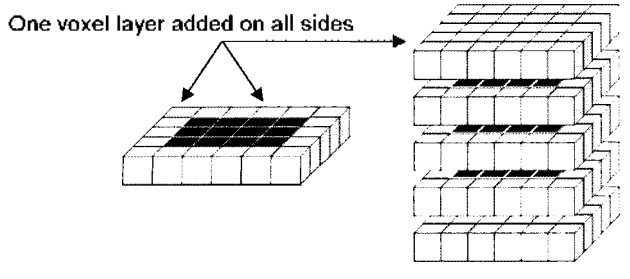


Fig.10. Voxel model storage at layers of bit-arrays.

The memory required for voxel storage is minimized by three approaches. One is by using the smallest memory unit for representing single voxel, that is, a bit for each (binary) voxel. The second is by dynamic memory allocation, balancing pointer overhead and memory reduction. The third is by generating and storing the bit planes only within the Z-extent of the model. The memory allocated for a model depends upon its size and distribution of object along coordinate axes; efficient storage implies orienting the object to minimize the number of voxel planes required. Each voxel layer is an array of $(\text{Resolution} + 2)^2$ voxels, and thus a square plane in the total voxel space irrespective of the object shape. Setting, resetting, accessing and editing the voxel information in the bit array is achieved by appropriate functions written for the purpose.

4.3 Voxel model display

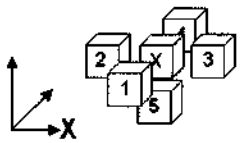
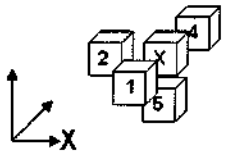
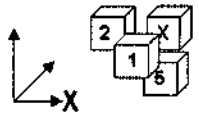
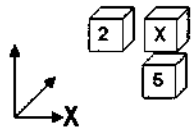
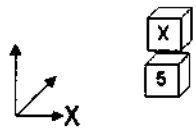
The voxel model does not contain information about the orientation of the original surface. For a realistic rendering of the voxel model, the original surface is required to be generated (assuming it is not available). This is however, a computationally intensive task, and was not found to be practical for high resolution voxel models being attempted in the present work. A direct method of displaying voxels as points was needed. The problem is that all voxels are aligned along the orthographic axes, and a display using their surface normals would produce a blocky appearance, which would be unacceptable.

The above problem was overcome by using a predefined lookup table of grayscale intensity values corresponding to the neighbor configuration of different voxels. Although each voxel can have up to 26 neighbors, only face connectivity between voxels contributes to the visibility of the voxel. Therefore only face neighbor configurations were considered for rendering the voxels. Only surface voxels were considered for display, to further reduce the computation time. Table 1 shows various neighbor configurations of surface voxels and the codes used for classifying them. The intensity values for all possible combinations are stored in a look-up table for minimizing computation time for voxel model display.

5. Results and Discussion

A framework to generate, store and display voxel models with resolutions up to 1000 per axis (total 1 billion voxels) has been implemented in this work. As mentioned earlier, the representation scheme is based on binary cubic voxels aligned with the coordinate axes, and the voxels are stored using a stack of bit-arrays. The voxel-based display method described earlier has also been implemented, using a look-up table containing intensity values corresponding to various voxel configurations. This yielded a much more realistic display compared to

Table 1. Intensity lookup table based on surface voxels visibility configuration.

No. of free faces	Sample configuration	Bit string						Integer returned	No. of similar configurations
		1	2	3	4	5	6		
1		1	1	1	1	1	0	62	6
2		1	1	0	1	1	0	54	15
3		1	1	0	0	1	0	50	20
4		0	1	0	0	1	0	18	15
5		0	0	0	0	1	0	2	6

simple display of cube surfaces. The thickness algorithms and post-processing in terms of internal thickness distribution in a cross-section, successive skin removal, and radiographic images of minimum, maximum and total thickness have been developed. All algorithms have been implemented using Visual C++ language in Windows environment and tested on a Pentium 4, 2.4 GHZ processor and 2 GB RAM computer.

The test objects included several engineering as well as non-engineering models. Here we present the results for three test objects: (1) industrial casting of a press cylinder with 45K facets, modeled using an engineering CAD program, (2) Ganesha idol model with 19K facets, modeled using a haptic-based CAD system, and (3) a pelvic bone model with 502K facets, reconstructed from CT scans using a medical modeling program. Table 2 shows the voxellisation time and memory required for different resolutions. While the Ganesha model required just 8.5 MB RAM and 1 sec at a resolution of 200 voxels per axis, the pelvic model required 1126 MB RAM and 146 sec at a resolution of 800. The Ganesha model at different resolutions is shown in Fig. 11. The display time was less than 2 seconds for resolutions up to 200 voxels per axis, and less than a minute for 1000 voxels per axis resolution

The results of successive skin removal algorithm for

Table 2. Voxellisation memory and time for different models.

Object (STL file)	File size (KB)	Facets	Resolution									
			200		400		600		800		1000	
			M	T	M	T	M	T	M	T	M	T
Ganesha	4,140	19,298	8.5	1.1	17.4	3.6	23.8	5.5	55.5	14.1	90.6	22.6
Press cyl.	7,415	45,152	20.5	3.9	110.3	14.9	342.9	33.4	791.8	62.6	1521.3	95.7
Pelvis	102,289	502,388	115.6	31.2	249.0	47.2	556.5	90.9	1126.3	145.9	-	-

M = Memory required for voxelization (MB), T = Time required for voxelization (sec)

Fig. 11. Ganesha surface model and voxel models. (R = 200, 400, 600, 800, and 1000)

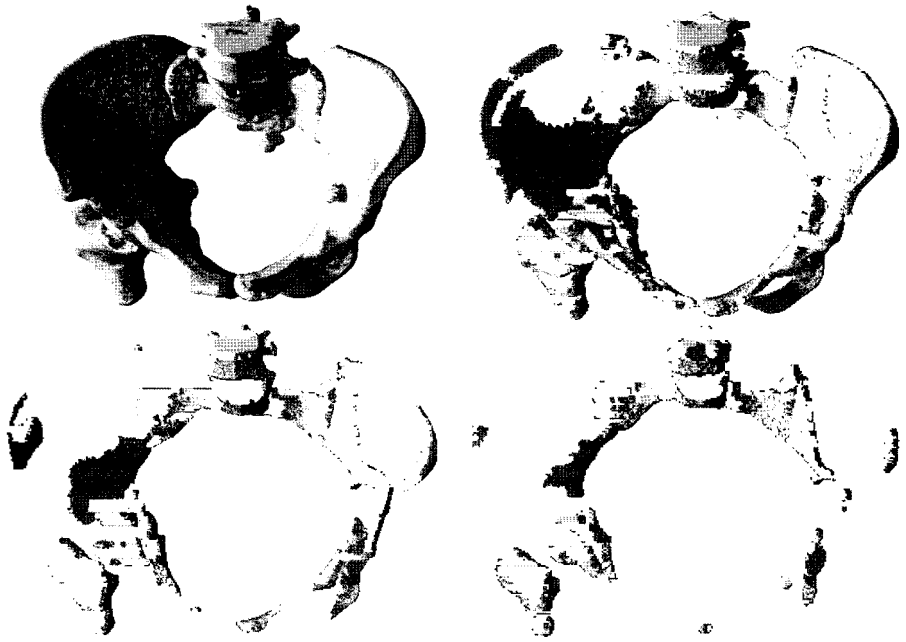


Fig. 12. Successive skin removal after 16, 32 and 48 iterations.(R = 800)

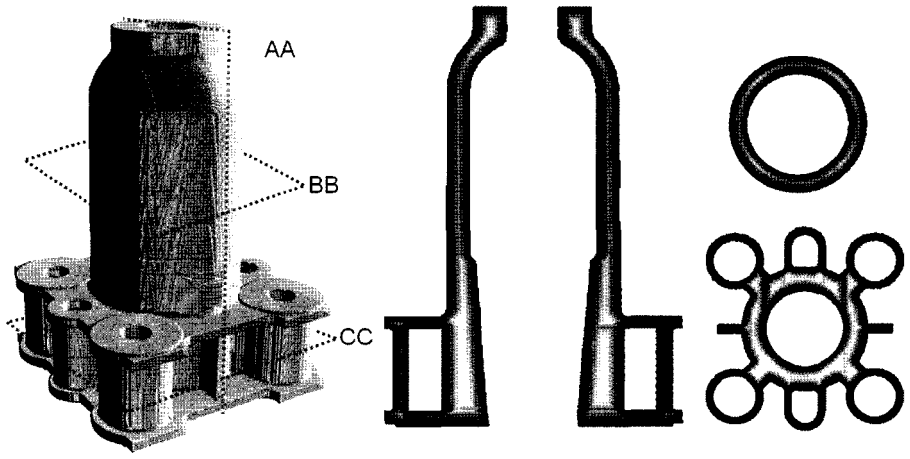


Fig. 13. Interior thickness maps of press-cylinder at AA, BB and CC sections. (R = 800)

computing the internal thickness distribution for the pelvic bone model are shown in Fig. 12. This information is used for displaying the internal thickness distribution on any desired cross-section of the object. These results are shown for the press-cylinder casting in Fig. 13. The skeleton of the object is given by the locus

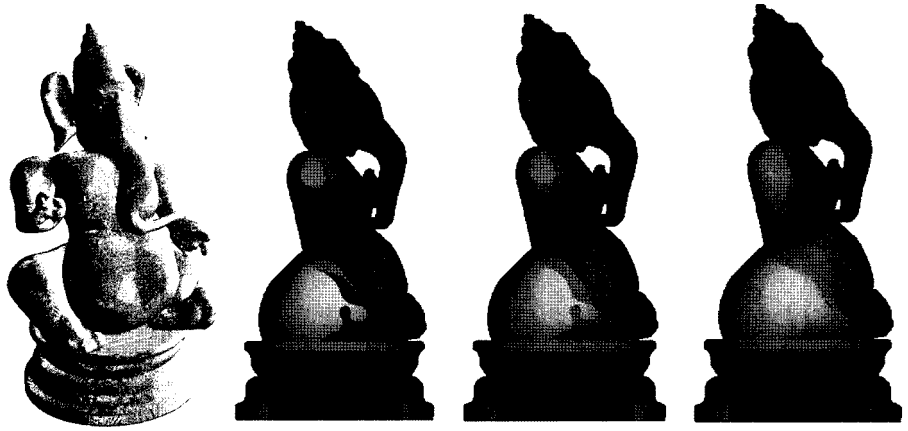


Fig. 14. Radiography images in minimum, maximum and total thickness modes (R=1000)

of the innermost (or thickest) points. Finally, the radiographic thickness with all three options: minimum, maximum and total thickness is shown for the Ganesha model in Fig. 14.

6. Conclusion

Three generic definitions of thickness, and methods for computing them, have been presented in this paper. They are useful for visualizing the internal thickness distribution (including the object skeleton) as well as thickness related features (thin and thick sections in radiography images). The voxel-based representation enabled robust algorithms for various thickness computations. Efficient algorithms for voxel model storage facilitated high resolution models, thereby minimizing the error in thickness computation. The voxel model display scheme, using a look-up table related to voxel neighborhood configurations, enabled realistic visualization of voxels models in near-real time. The successful testing of the algorithms on intricate objects (including a pelvic bone with over 500,000 facets converted into a voxel model with a resolution of 800 voxels per axis) using a standard Windows computer has clearly demonstrated the promise of voxel-based modeling. This work is part of on-going research work on voxel-based automated geometric reasoning for applications in engineering, art and medical fields. Further work involves using the thickness information for identification and analysis of junctions, evolving other geometric reasoning algorithms, and exploring a hybrid scheme that combines voxel and surface models to improve the efficiency of algorithms.

References

[1] Aurenhammer, F. (1991), "Voronoi Diagrams-A Survey of a Fundamental Geometric Data Structure," *ACM Computing Surveys*, 23(3), 345-405.

[2] Boada, I., Coll, N., Madern, N. and Sellares, J.A. (2005), "Approximations of 3D Generalized Voronoi Diagrams," *Proc. of 21st European Workshop on Computational Geometry*, 163-166.

[3] Dey, T.K. and Zhao, W. (2002), "Approximate Medial Axis as a Voronoi Subcomplex," *Proc. 7th ACM Symposium on Solid Modeling Applications*, 356-366.

[4] Etzion, M. and Rappoport, A. (2002), "Computing Voronoi Skeletons of A 3-D Polyhedron by Space Subdivision," *Computational Geometry: Theory and Applications*, **21**(3), 87-120.

[5] Giblin, P.J. and Kimia, B.B. (2004), "A Formal Classification of 3D Medial Axis Points and their Local Geometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. **26**(2), 238-251.

[6] Huang, J., Li, Y., Crawfis, R., Lu, S.C. and Liou, S.Y. (2001), "A Complete Distance Field Representation," *Proc. of IEEE Visualization Conference*, San Diego, 247-561.

[7] Iyer, N., Kalyanaraman, Y., Lou, K., Jayanti, S. and Ramani, K. (2003), "A Reconfigurable 3D Engineering Shape Search System Part-I: Shape Representation," *Proc. of ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Illinois.

[8] Jones, M.W. and Satherley, R.A. (2001), "Shape Representation using Space Filled Sub-Voxel Distance Fields," *International Conference on Shape Modeling and Applications*, 316-325.

[9] Kaufman, A.E. (2000), "State of the art in Volume Graphics," in *Volume Graphics*, Springer-Verlag, 3-28.

[10] Lee, T.C., Kashyap, R.L. and Chu, C.N. (1994), "Building Skeleton Models via 3-D Medial Surface Axis Thinning Algorithms," *CVGIP: Graphical Models and Image Processing*, **56**(6), 462-478.

[11] Lockett, H.L. and Guenov, M.D. (2005), "Graph-based feature recognition for injection moulding based on a mid-surface approach," *Computer-Aided Design*, **37**(2), 251-262.

[12] Lu, S.C., Rebello, A.B., Miller, R.A., Kinzel, G.L. and Yagel, R.(1997), "A Simple Visualization Tool to Support Concurrent Engineering Design," *Computer-Aided Design*, **29**(10), 727-735.

[13] Nagasaka, Y., Nakamura, M. and Murakami, T.(2001), "Extracting and Learning Geometric Features based on a Voxel Mapping Method for Manufacturing Design," *Proc. of 3rd International Conference Intelligent Processing and Manufacturing of Materials*, British Columbia, 1-10.

[14] Nooruddin, F. S. and Turk, G. (2003), "Simplification and Repair of Polygonal Models using Volumetric Techniques",

IEEE Transactions on Visualization and Computer Graphics, **9**(2), 191-205.

[15] O' Rourke, J. (2001), "*Computational geometry in C*," Cambridge University press, 2nd Edition.

[16] Pao, W.K.S., Ransing, R.S., Lewis, R.W. and Lin, C. (2004), "A Medial Axes-based Interpolation Method for Solidification Simulation," *Finite Elements in Analysis and Design*, **40**, 577-593.

[17] Quadros, W. R., Gurumoorthy, B., Ramaswami, K. and Prinz, F.B. (2001), "Skeletons for Representation and Reasoning in Engineering Applications," *Engineering with Computers*, **17**, 186-198.

[18] Ramamurthy, R. and Farouki, R.T. (1999), "Voronoi Diagram and Medial Axis Algorithm for Planar Domains with Curved Boundaries I. Theoretical Foundations," *Journal of Computational and Applied Mathematics*, **102**, 119-141.

[19] Ravi, B., Creese, R.C. and Ramesh, D. (1999), "Design for Casting-A New Paradigm to Prevent Potential Problems," *Transactions of the AFS*, 107.

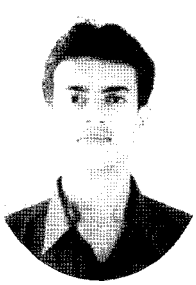
[20] Reddy, J. M. and Turkiyyah, G.M. (1995), "Computation of 3D Skeletons using a Generalized Delaunay Triangulation Technique," *Computer-Aided Design*, **27**(9), 677-694.

K. Subburaj received his B.E. from National Engineering College, Kovilpatti, Tamilnadu in 2003, followed by M.E. in Production Engineering from M.S. University of Baroda, Gujarat in 2005. He is currently pursing his Ph.D. in the Department of Mechanical Engineering, at Indian Institute of Technology, Bombay. His research interests include geometric reasoning, computer graphics, and Rapid Prototyping & Tooling.

B. Ravi completed his BE from NIT Rourkela in 1986, followed by ME (1988) and PhD (2002) from Indian Institute of Science, Bangalore. He is currently an Associate Professor of Mechanical Engineering at Indian Institute of Technology, Bombay. His research interests include intelligent CAD, rapid manufacturing and product lifecycle engineering for metal casting and medical applications. He has published about 120 technical papers in various journal and conference proceedings, and a book on *Metal Casting: Computer-aided design and analysis*. He is a Fellow of the Institution of Engineers (India).

Sandeep Patil received his Bachelor' degree in Production Engineering from Shivaji University, Kolhapur, Maharashtra in 2002, followed by an M.Tech. in Mechanical Engineering from Indian Institute of Technology, Bombay, specializing in Manufacturing Technology, in 2005. At present, he is working as a software developer in a leading CFD company. His areas of interest include geometric modeling, representations and processing, product development, DFX, rapid prototyping and rapid tooling.

K. Subburaj



Sandeep Patil

B. Ravi