Extraction and Visualization of Dimensions from a Geometric Model on the basis of its Medial Axis

by

Terence Michael Bahlen, B.Sc

A Thesis
Submitted to the School of Graduate Studies
in Partial Fulfillment of the Requirements
for the Degree of

Master of Science

Presented at

Delft University of Technology
Faculty of Electrical Engineering,
Mathematics, and Computer Science,
Computer Graphics and CAD/CAM Group

Summary

This thesis discusses a method for the extraction and visualization of certain dimensions, specifically thicknesses and angles, from a geometric model, based on the medial axis. The points on the medial axis and the corresponding points on the surface, the so-called footpoints, are classified as thickness or angles points, after which the dimensions are extracted. The classification is based on the connectivity of the governors of the medial axis. This works properly for polygonal models, since they contain junctions points that divide the medial axis and hence the boundary, *i.e.*, the governors, into different areas, capturing different dimensions. For arbitrary surfaces, an extra step was needed, since there are many situations in which there are no junctions points. In this case, the classification of footpoints, capturing different dimensions, is extracted using the radius function of the medial axis. Starting at the edges of the model, the radius function has an increasing value, and as long as this continues and the governors remain the same, an angle area is defined.

After the correct classification, the dimensions are extracted and either a thickness or angle dimension is assigned to a footpoint on the surface. At concave edges and at rounded regions, extra steps are needed to get useful dimensions. Footpoints on a concave edge get more than one extracted dimensions assigned to it, depending on the number of governors; this requires an extra step in the visualization process. At rounded regions, the diameter is taken as the thickness.

The next step, given a dimension at each footpoint of the surface, is to visualize these dimensions on the boundary of the geometric model. This is done using the mesh that is used for the computation of the medial axis. All mesh vertices are footpoint of the medial axis and have associated with it a dimensional value. These values are then interpolated across the triangles of the mesh. Depending on the classification, a different color map is used. Between the angle and thickness areas and at concave footpoints, extra visualization steps are needed to get a desirable visualization.

The major bottleneck of the method, is the computation of the medial axis. Since this computation needs to work for any kind of input and format, the computation is based on a mesh. The point cloud, consisting of the mesh vertices of the mesh, is used as a representation of the object, after which each mesh vertex is considered to be a footpoint, and the corresponding maximal sphere of the medial axis is sought. The latter is done by constructing potential maximal spheres, using the other footpoints, from which the best match, *i.e.*, the sphere with the smallest radius value, is taken. This results in a cloud of points on the medial axis, which have a link to the mesh vertices and governors to which these vertices belong. This is needed for the correct classification of the points.

To compute a correct medial axis, a dense mesh is needed, which makes its computation a bottleneck. To overcome this problem, a spatial subdivision is used that divides the space within the bounding box into cells, containing the mesh vertices. This allows a fast computation of a subset of vertices in the neighborhood of a mesh vertex, given a guessed radius value. This value is specified between neighboring footpoints, since footpoints that are close on the object's surface, have radius values that are close.

To further speed-up the computation of the medial axis, the computation was done in parallel. The computation of the medial axis without the spatial subdivision, is easily made parallel and shows exemplary speed-up results, *i.e.*, close to linear. The parallel computation using the spatial subdivision, is not as scalable, but has a good speed-up. This suggests that in the near future, as the number of cores in computers increases, the computation without the spatial subdivision will have the best performance.

The visualization gives intuitive feedback, making a clear distinction between thickness and angle dimensions, using different color maps. With this visualization design guidelines related to injection molding can be verified, parametric design dimensions can be reverse engineered and other properties of the geometric model can be verified.

In conclusion, the extraction and visualization of design dimensions, gives good insight into the geometric information contained within a model, and it works for almost any kind of shape.

Preface

This thesis was developed in collaboration between Delft University of Technology and McMaster University, under the supervision of Dr. W.F. Bronsvoort and Dr. A.D. Spence. Most of the work was developed at the Mechanical Engineering department of McMaster University, in Canada, for the duration of a year.

The thesis discusses a method that extracts and visualizes dimensions on the basis of the medial axis to gain insight into the information enclosed within a geometric model. Preliminary results were presented at the CAD '09 Conference in Reno, Nevada, and a final paper was published at the CAD '10 Conference in Duba, UAE, see [1], which is enclosed as an appendix in this thesis.

Personally, I had a great time in Canada and in the subsequent months in the Netherlands developing and implementing this method, even though it turned out to be something completely different from what I expected, which made it hard at times to see what the final result was going to be. I would like to thank Dr. A. D. Spence and Dr. W. F. Bronsvoort for their support, patience, suggestions and for the opportunity to do my thesis in Canada. Finally, I would like to thank my family, pets and friends for their support and enthusiasm.

Contents

List of Figures

1

Introduction

Dimensions, such as thicknesses and angles, play an important role in many CAD/ CAM applications. Not only in design environments, but also in the analysis and simulation of manufacturing processes. One of the most commonly used manufacturing process is injection molding. It offers the possibility to make simple parts, but also fully-finished complex-shape parts, while producing at high speed and low cost. Products made with injection molding include computer mouses and keyboards, interior car body panels, mobile phone cases, etc. Injection molding is a manufacturing process for the production of plastic parts; in short, melted plastic is forced into a cavity where it takes the shape of the cavity once it cools down. Theoretically, there is no limit to the shapes that can be made, but there are many practical considerations that have to be met, to make a successful part. Molded parts can exhibit surface defects, such as warpage and surface sink marks, see Figure 1.1 for an example of a surface sink mark.

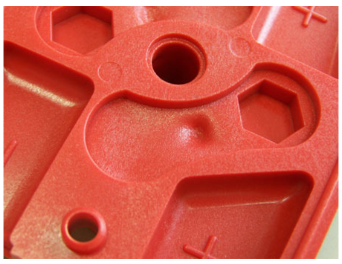


Figure 1.1: An example of a surface sink mark, an indentation of the surface of an object made with injection molding, source [2].

The origin of these two defects is in the cooling stage of the process; during the cooling stage one part of the object may cool down faster than another part, because it is in a thinner section. This causes differential shrinkage of the plastic and this shrinkage can result in sink marks and warpage. Sink marks are indentations on the surface of the molded part caused by the thermal contraction of the melt during the cooling stage, which occurs when there is a significant local change in the part wall thickness. If one area of the part has another level of shrinkage than another area, caused by changes in thickness, the resulting stresses between the two areas can cause the part to warp, which may even lead to cracking of the part's surface. Therefore, one of the main design guidelines when it comes to designing for injection molding is to maintain a uniform wall thickness throughout the part, to reduce the chance of uneven shrinkage, and as a result a reduction in the chance of surface sink marks and warpage. If the thickness variation is unavoidable, the transition between the thicknesses should be gradual, to prevent abrupt changes in the thickness and later on prevent abrupt changes in the temperature during solidification.

Besides warpage and surface sink marks, another important property that is influenced by the thickness and by the size of the angles within a part, is the flow pattern of the material as it flows through the cavities of the mold. A nominal thickness maintained throughout a part can simplify the melt flow. In thin sections of a part and at sharp angles, there is a sudden pressure drop because of the amount of heat loss. In an extreme situation, the plastic could freeze off, impeding the flow; as a consequence thin sections and areas with sharp angles may have voids within them. To avoid sharp angles, it can be chosen to round certain regions.

Many design guidelines related to injection molding have to do with ribs and bosses, which are structures that reinforce the strength of a part. For instance, instead of increasing the wall thickness to achieve a stronger part, so-called ribs can be added, because it increases part strength considerably, as illustrated in Figure 1.2. Improper placement of these ribs and bosses can cause more problems then they solve, since they can increase the occurrence of surface sink marks and warpage. For instance, ribs are most effective in thin areas which must bear a load perpendicular to its place. Some of the design guidelines related to ribs are: ribs should be around 50% to 60% of the walls to which they are attached, the height of ribs should be less than three times the wall thickness, and round the corners at the point of attachment, as illustrated in Figure 1.2(b).

The example design guidelines mentioned above are only a small part of the design guidelines related to injection molding, and they can differ depending on the type of plastic used and a whole range of other parameters that effect the injection molding process. But what does become clear, is that most of these design guidelines are related to thicknesses and angles, and that wrongful placement of, for instance, ribs and bosses can make things worse. Especially, since a designer does not explicitly input all dimensions, but only a limited number, most of the dimensions within the model are implicit and these have to be made explicit to check if the design guidelines are maintained. At every stage of the design process these guidelines have to be checked, but most tools only allow manual inspection and do not give a complete overview of core dimensions

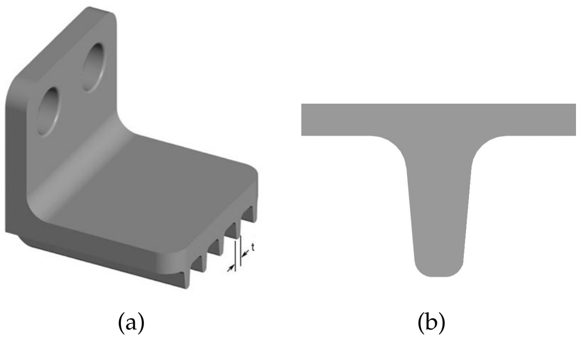


Figure 1.2: Design guideline for adding ribs, (a) showing a design with added ribs and (b) showing a close-up of how to properly add ribs, source [3].

such as thicknesses and angles. Of course, the successful completion of an injection molded part is not only related to design guidelines; for instance, the choice of the type of plastic also plays an important role.

This thesis introduces a method that extracts and visualizes information on dimensions from a geometric model, in order to make design dimensions more explicit, to, for instance, verify design guidelines. The research focuses on thickness and angle dimensions, because these two dimensions are sufficient to get a good overview and a basic understanding of the dimensions within a model. At first, this research focused only on thickness dimensions but later on angles where incorporated, since the extraction and visualization of only the thickness does not provide a good insight into all areas of a CAD model. A previous approach that only visualizes thickness information, see [4], uses the radius function of the medial axis as the value for the thickness and visualizes this on the boundary of the geometric model. This approach therefore always indicates a thickness of value zero in the corners of the model, which is not useful at all. We, on the other hand, display the size of the angle at a corner and also have a better measure of thickness. As will become clear, a good distinction can be made when to visualize angles and when thicknesses. It is based on the medial axis, or skeleton. It uses this representation to extract dimensions, which are then visualized in a consistent way on the boundary of the geometric model.

With this method, dimensions in a model can be verified, *e.g.*, in design for injection molding as mentioned above, but also certain other properties, such as symmetry or the lack thereof can be detected. For example, a model created with reverse engineering, *e.g.*, with laser scanning, can contain multiple instances of the same feature and visualizing the dimensions of each of these features allows the user to see whether they are indeed the same. One of the features may differ from the others, because it endured larger forces and has therefore a larger amount of wear. Another example in which it is useful is when a CAD model is transferred from one system to another system. Often the parametric design dimensions do not transfer and only a geometric model remains, as illustrated in Figure 1.3. If the dimensions are needed in the other system, they have to be recovered from the geometric model.

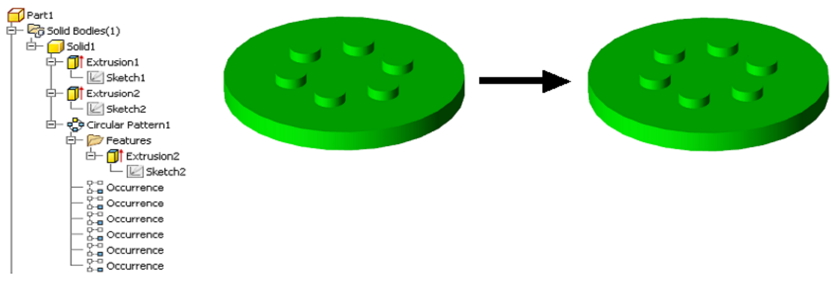


Figure 1.3: When a geometric model is transferred to another system, only the geometric information within the model remains and the link between several occurrences of the same feature is lost.

This thesis is divided into seven chapters, of which Chapter 2 gives background information on the medial axis transform, since this representation is used throughout the entire process and a good understanding of this is needed to understand the rest of the thesis. Chapter 3 discusses how dimensions are extracted and calculated from the medial axis transform, and also briefly discusses the visualization process to make this chapter self-containing. Chapter 4 then discusses the computation of the medial axis, including several implementation details, after which Chapter 5 discusses the visualization process and Chapter 6 the results that can be obtained using the steps from Chapters 3 through 5. Chapter 7 concludes the thesis. In the appendix, the paper that was written as part of this research is included.

2

Medial Axis Transforms

As mentioned in Chapter 1, the extraction of design dimensions is based on the representation of the medial axis, therefore this chapter will give background information on the medial axis transform.

Section 2.1 will give the general definition of the medial axis transform and some basic concepts, after which, in Section 2.2, some important properties are discussed, of which the reconstructability is the most important one, since it is the basis of the extraction of dimensions as explained in Chapter 3. Finally, Section 2.3 discusses some algorithms for both the 2D and 3D medial axis transform.

2.1 Basic Concepts of the Medial Axis Transform

The medial axis transform was first introduced by Blum [5] to describe biological shapes. Since then it has been extensively researched and developed, mostly in areas involving shape analysis. Often the medial axis transform is referred to as the skeleton of an object. The medial axis is used in shape simplification [6], shape matching [7], routing in sensor networks [8], etc.

The definition of the medial axis and the medial axis transform given by Sherbrooke et al [9] is as follows. *Let D be a subset of R^n. The Medial Axis (MA), or skeleton, of a subset D of R^n, denoted M(D), is the locus of points which lie at the centers of all closed spheres which are maximal in D, together with the limit points of this locus. A closed sphere (or circle in a 2D figure) is said to be maximal in D if it is contained in D but is not a proper subset of any other sphere (circle) contained in D. The radius function of the medial axis of D is a continuous, real-valued function defined on M(D) whose value at each point on the medial axis is equal to the radius of the associated maximal sphere. The medial axis transform of D is the medial axis together with its associated radius function.*

Figure 2.1 illustrates an example in 2D and Figure 2.4(b) an example in 3D.

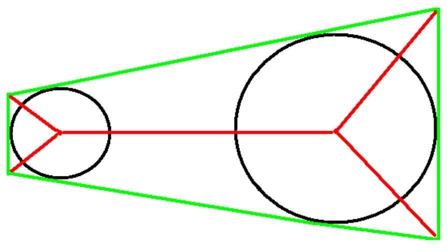


Figure 2.1: An example of a 2D medial axis, where the green lines represent the original object, the red lines the medial axis and the two black circles two of the maximal circles within the object.

Another way to look at the medial axis is the grass fire analogy, which is a more dynamic interpretation. In this analogy, each point on the boundary is considered to be a point of fire, all points burning with the same intensity. The fire spreads perpendicular from the boundary point, at which it started, towards the inside of the object, and burns with a constant rate of one unit distance per unit time. At time t the outer extent of the burned area is the curve parallel to the boundary offset by distance t. The medial axis consists of the closure of the quench points of the fire, *i.e.*, the points where fires, which began at two or more different boundary points, meet and douse one another. The times at which the fires meet is the radius function of the medial axis transform. The grass fire analogy is illustrated in Figure 2.2.

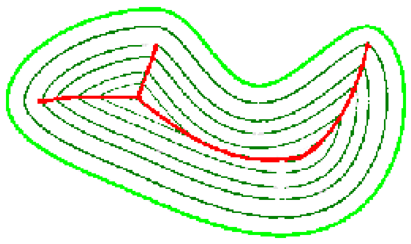


Figure 2.2: The grass fire analogy of the medial axis, where the green lines represent the boundary and the offsets of the boundary and the red lines the medial axis, which consists of the quench points of the offsets of the curves, source [10].

Many algorithms that compute the medial axis transform from a geometric model make use of some basic concepts related to the medial axis transform. These basic concepts involve a classification of the points on the medial axis, which is determined by so-called footpoints. A footpoint is a point of contact with the object boundary of the maximal circle, or in 3D the maximal sphere, of a point on the medial axis. In the footpoint, the circle is tangent to the object boundary. Depending on the shape of the object boundary there can be either a discrete point contact or an area contact with the boundary; in the latter case the radii of curvature of the boundary and the maximal circle are equal. Since the maximal circle is tangent to the object boundary, the lines from the medial axis point to its footpoints are perpendicular to the object boundary.

Together with the notion of a footpoint, comes the notion of a governor, which is the face, edge or vertex in which the footpoint lies, as illustrated in Figure 2.3.

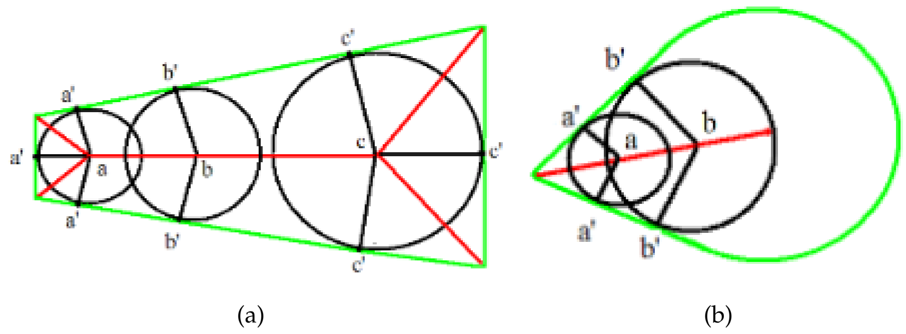


Figure 2.3: Examples of 2D medial axis transforms; the red lines indicating the medial axis, the green lines the boundary of the object and the black circles maximal circles, with their centers at a, b, c and their footpoints at a', b' and c'; (a) having four edges and four vertices as governors, and (b) having one curved edge and one vertex as governors.

Blum and Nagel [11] use the concepts of governors and footpoints to subdivide the medial axis into several types of points, and depending on the type of the points, they consider the medial axis transform to be a set of simplified segments. The types of medial axis points Blum and Nagel distinguish are:

- a normal point: a point whose maximal disc touches the object border in exactly two separate contiguous sets of points,
- a branch point: a point whose maximal disc touches the object border in three or more separate contiguous sets; Figure 2.3(a) has 2 branch points and Figure 2.3(b) none,
- and an end point: a point whose maximal disc touches the object border in exactly one contiguous set; Figures 2.3(a) and 2.3(b) have 4 and 2 end points, respectively.

The sets of simplified segments are the sets of contiguous normal points bounded by either a branch and, or an end point.

Algorithms for the 3D medial axis transform use another classification of points, because more cases can be distinguished. This classification scheme is based on the scheme by Brandt [12] and was used by one of the first algorithms for computation of the medial axis transform for a 3D case; see Sherbrooke et al. [9]. The approach by Sherbrook et al. identifies the following types of points, illustrated in Figure 2.4:

- a seam point: a point that has three or more footpoints on the boundary (a seam is a connected curve of seam points),
- a seam-end point: a point where the seam runs into the boundary (the limit points of the medial axis transform),
- a junction point: a point where seams intersect,
- a sheet point: a point with exactly two footpoints,
- and a sheet: the collection of connected sheet points (a sheet is a surface).

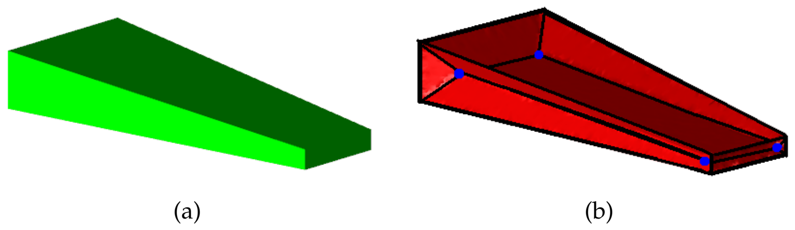


Figure 2.4: Classification of points on the medial axis: (a) the original shape and (b) the medial axis; black lines indicate seams and edges, blue dots junction points and red areas sheets. This medial axis contains 13 sheets, 4 junction points, 8 seam-end points and 12 seams.

2.2 Properties of the Medial Axis Transform

There are some interesting properties of the medial axis which will be discussed in this section: reconstructability, sensitivity to small changes in the boundary, non-generic cases, path connectivity, and the relationship with Voronoi diagrams. Of these properties the reconstructability is, in our case, the most important one, since the extraction and visualization of dimensions relies on this property.

Reconstructability The reconstructability theorem by [13] states that a geometric model can be uniquely determined from a medial axis transform by taking the union of all points on the medial axis and the associated maximal spheres. This one-to-one correspondence between the geometric model and the medial axis transform means that given a geometric model there is a unique medial axis transform and vice versa. The reconstructability theorem relies on the fact that although the medial axes of two geometric models can be the same, the medial axis transforms, which include the radius functions, are different if the two models are different; see the example in Figure 2.5.

Sensitivity to small changes in the boundary One of the drawbacks of the medial axis

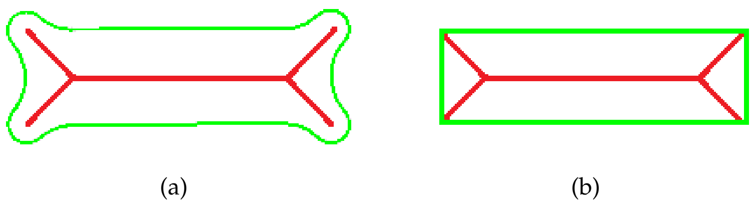


Figure 2.5: Two geometric models with the same medial axis, but different medial axis transforms.

transform is its sensitivity to small changes in the boundary, as illustrated in Figure 2.6, where it can be seen that a small change in the object boundary can lead to a significant change in the medial axis. Some algorithms try to alleviate this problem by classifying offshoots of the medial axes as less significant, based on a substance measure, see [14]. These extraneous axes are then removed and only the major portions of the medial axis are maintained.

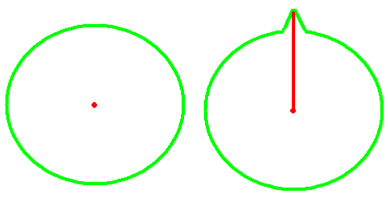


Figure 2.6: The problem of sensitivity of the medial axis transform, illustrating that a small change in the object's boundary can lead to a significant change in the medial axis transform.

Non-generic cases Some shapes exhibit special cases of the medial axis transform. Two common cases are a junction point with more than four governors and a seam with more than three governors. These special forms are incompatible with the generic case algorithms and separate special treatment is needed for them. Two examples of non-generic cases are given in Figure 2.7.

Path Connectivity Another property of the medial axis transform is that it is path connected, *i.e.*, if the original object, from which the medial axis transform is derived, is path connected, then the medial axis transform is also path connected. Path connectivity means that any two points on an object can be connected by a path in the object, or for the case of the medial axis transform, any two points on the medial axis can be connected by a path on the medial axis. Several papers prove this for the medial axis transform, all in a somewhat different way. Wolter [13] proves it by means of the homotopy theory and Choi et al. [15] prove the connectivity of the medial axis by showing the continuity of the maximal spheres or, stated differently, the continuity of the radius function of the medial axis.

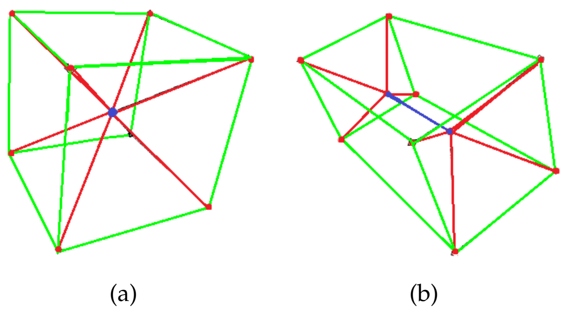


Figure 2.7: Two examples of non-generic cases, where only the seams of the medial axis and the wireframe of the models are visualized; (a) representing a cube where the blue junction point is a non-generic point, since it is associated with six governors, instead of the usual four, and (b) a rectangular cube, where the blue seam has four governors, instead of the usual three, source [9].

Relationship with Voronoi Diagrams Given the properties of a Voronoi diagram, it is clear that it is related to the medial axis, since they are both part of equidistant (point) sets. Lee [16] showed that the Voronoi diagram and the medial axis of any convex polygon are identical, and that the medial axis of an arbitrary polygon is a subset of its Voronoi diagram. This is obvious, because the maximal circles of the medial axis touch the boundary of a 2D object in more than one point (two at least). Therefore a dense subset of points on the medial axis is equidistant from more than one point on the boundary. Fabbri et al. [17] state that for the case of 3D polyhedral regions, the Voronoi diagram is a superset of the medial axis. Given the proof by Lee, Ramamurthy and Farouki [18] mention that for the same reason, the medial axes of piecewise-linear/circular boundaries are subsets of their Voronoi diagrams. However, they also mention that for domains bounded by free-form curve segments, the medial axis is not a subset of a Voronoi diagram.

In order to calculate the medial axis of an arbitrary surface with the help of Voronoi diagrams, usually a scheme is considered where a set of sample points (a sample of the boundary of a shape is a finite set of points (exactly and not just approximately) on that boundary) on the shape is taken. From that the Voronoi diagram is constructed, which is then used to approximate the medial axis of the shape. Brandt [12] states that if the sample density approaches infinity, the Voronoi vertices converge to the medial axis. Figure 2.8 clearly illustrates that the Voronoi vertices converge to the medial axis of the figure; first the boundary is replaced with a sequence of points densely placed on the boundary, and from that the Voronoi diagram is constructed.

The convergence of Voronoi vertices does not hold in the case of 3D. The validity of

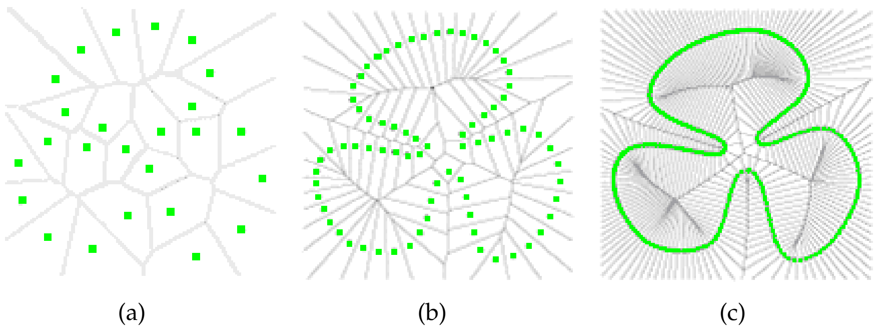


Figure 2.8: (a) to (c) showing an increasing sample point density and the corresponding Voronoi diagrams, illustrating that the Voronoi vertices converge to the medial axis, source [10].

the extension to 3D is spoiled by the existence of slivers. Roughly, a sliver is a tetrahedron whose four vertices are almost co-circular. The location of the Voronoi vertex corresponding to the sliver depends on the four vertices, but is generally unrelated to any feature of the surface and does not necessarily lie near the medial axis (see Attali et al. [10]). However, this does not mean that Voronoi diagrams cannot be used for 3D shapes to approximate the medial axis. Algorithms that use Voronoi diagrams in 3D, eliminate all but a few Voronoi vertices, referred to as poles. These poles are then used to approximate the medial axis.

2.3 Algorithms for the Medial Axis Computation

The computation of the medial axis transform is hard in general, because of numerical instabilities and because the medial axis transform is sensitive to small changes in the boundary. Therefore there are many algorithms that compute the medial axis transform for various types of shapes. This section will discuss several examples of algorithms, starting with some algorithms for the 2D medial axis in Section 2.3.1, after which several 3D algorithms for the computation of the medial axis transform are discussed in Section 2.3.2.

2.3.1 Two-Dimensional Medial Axis Transform Algorithms

Culver et al. [19] make a distinction between two categories of algorithms that calculate the medial axis transform. The first category, continuous algorithms (or sometimes called algebraic methods), are so-called tracing approaches. The second category, discrete algorithms, are based on surface sampling or a spatial subdivision. These sampling approaches involve approaches using Voronoi diagrams based on point data sets on the boundary or some other type of structure, for example a grid on which shock waves are propagated.

To understand tracing algorithms, consider Figure 2.9 (for more details on such algorithms, see [20]). Tracing algorithms execute tracing steps, with a step size. The step size controls the level of approximation of the medial axis transform. In Figure 2.9, the algorithm starts at a vertex and based on the footpoints it chooses a direction to trace in. The step size controls how far in that particular direction the next point is taken. The algorithm then checks whether that point is still a part of the medial axis. This is called tracing. Figures 2.9(b) and 2.9(c) show two other steps of the tracing of the medial axis transform. As can be seen in Figure 2.9(b), the algorithm traced in the wrong direction, because it did not yet consider another footpoint. Figure 2.9(c) shows the correct tracing, in which a branch point is identified.

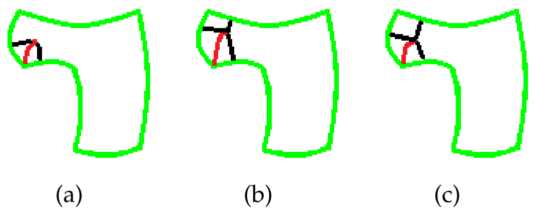


Figure 2.9: (a) to (c) illustrating three steps of a 2D tracing algorithm, in which it can be seen that in the second trace step (b), the tracing algorithm, depending on the step size, stepped too far and therefore makes an adjustment in (c).

As said earlier, discrete methods can use Voronoi diagrams to calculate the medial axis, but since this has already been mentioned in Section 2.2, another kind of algorithm will be illustrated here, one that is more according to the grass fire analogy, based on a grid. The algorithm by Tek and Kimia [21] uses a discrete wave propagation and shock wave propagation technique. The algorithm identifies two kind of waves; normal waves (discrete waves) and shock waves (where normal waves of opposing fronts meet or discontinuities on a discrete wave occur), see Figure 2.10. As can be seen, the shock waves form the medial axis.

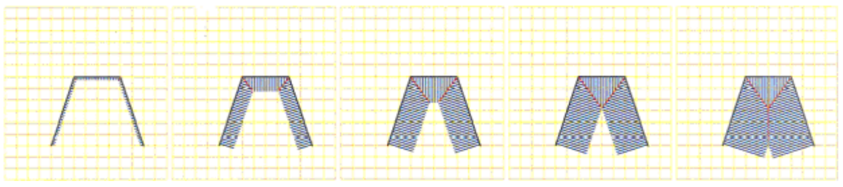


Figure 2.10: Discrete waves (blue lines) in various steps of the algorithm. The red lines indicate the formation of a shock wave, the medial axis. In the final two pictures the formation of a branch point can be seen, because of the intersection of three shock waves, source [21].

2.3.2 Three-Dimensional Medial Axis Transform Algorithms

This section will briefly discuss two kinds of algorithms for the computation of the medial axis transform for 3D shapes. These two algorithms were chosen, because they both use properties of the medial axis transform that are used in subsequent chapters.

The first algorithm, see [9], calculates the medial axis transform for 3D polyhedral shapes based on the relationships that exists between the various types of points, as discussed in Section 2.1. A simple example is that if g_1 is the governor of one sheet and g_2 of another, then the seam connecting the two sheets has as governors g_1 and g_2. The algorithm is a tracing algorithm, which identifies the seams of the medial axis, based on relationships as mentioned before. For instance, once the traversal is at a junction point, it begins by taking all three-element subsets of the governor set of the junction point in order to trace out all adjacent seams. Based on whether the end point is another junction point or a seam-end point, the traversal is continued. As it proceeds with the traversal it builds up an adjacency graph containing the geometry of the medial axis. After the traversal, the algorithm loops through the adjacency graph to identify sheets, by identifying loops in the adjacency graph.

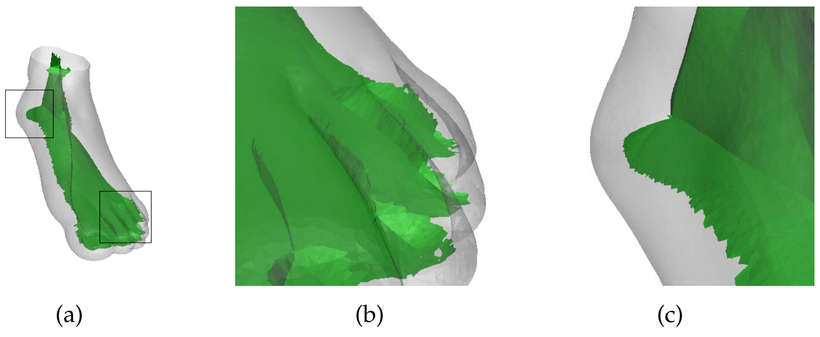


Figure 2.11: An example of a jagged result on the medial axis in (a), which can occur when the point cloud approximating the object's surface is not accurate enough, the two close-ups in (b) and (c) better illustrate the jagged result, source [22].

The second kind of algorithm uses a surface sampling approach, see [22], just like the algorithm discussed in Chapter 4. In such a method the initial object is represented as a dense cloud of sample points presumed to be on the boundary. One of the main issues when applying such an approach is the generation of an appropriate set of point samples on the boundary, to ensure a tight approximation of the medial axis transform. Under-sampling may result in an approximation of the medial axis that is not accurate enough and thus in jagged results, as illustrated in Figure 2.11. Because of this, some papers focus on simplicity and others on improved accuracy, as the paper by Dey and Zhao [22], which uses an angle and ratio condition to improve the resulting medial axis. Another paper [23] uses a numerical optimization step to move the sample points of the initial approximation so that the identified Voronoi vertices are closer to the medial axis. The authors of [22] have a follow up paper [24], in which they solve the problem of jagged results, due to under sampling, by maintaining a link to the original CAD model. Using the original CAD model allows them to use the information from that model to remove the jagged results.

3

The Medial Axis and Dimensions

This chapter will provide a complete overview of how, given a geometric model and its corresponding medial axis, dimensions are extracted, based on the classification of points on the medial axis. To determine both types of dimensions, thickness and angle, we extend the classification of points on the medial axis, as discussed in Chapter 2, in Section 3.1. Once the classification of points is provided, the dimensions can be extracted as discussed in Section 3.2, and these dimensional values are then used in the visualization process, briefly discussed in Section 3.3. Before these steps are discussed, the entire pipeline, from geometric model to the visualization of dimensions, is discussed.

The pipeline starts with a geometric model, which is then meshed. This approach was chosen, because the pipeline then allows any format as input as long as it can be meshed. In addition, a mesh aids in fast computation of the medial axis and allows for a simple visualization of all dimensions. A mesh is not a severe requirement, since there are many meshing tools that can mesh almost any kind of format. Some of the meshing details are discussed in Chapter 4. The next step along the pipeline, is the computation of the medial axis using the mesh as input; more details of this are also discussed in Chapter 4. The next step, given the medial axis, is the classification and extraction of dimensions, which is discussed in this chapter. This step is discussed before the computation of the medial axis to provide a complete overview within this chapter of all steps. The next step is the visualization of the dimensions, shortly discussed in this chapter and in more detail in Chapter 5. The pipeline ends with the complete visualization, of which one example will be given in this chapter and more in Chapter 6. The pipeline is illustrated in Figure 3.1, where the various steps are illustrated per chapter.

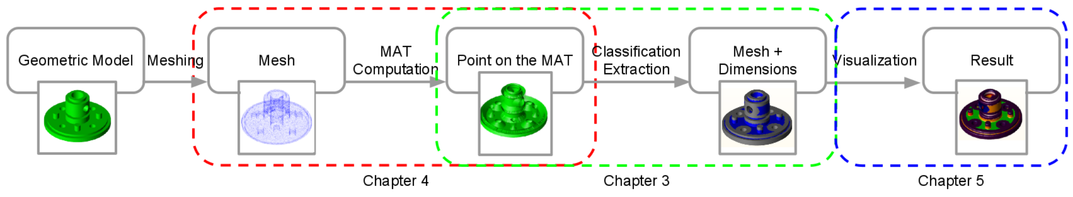


Figure 3.1: The complete pipeline listed per chapter; from geometric model to visualization.

3.1 Classification of Points on the Medial Axis

As discussed in Section 2.1, there is a classification of points on the medial axis, used by many algorithms to explain their working. This section will go from there and further classify points on the medial axis to determine both types of dimensions, thicknesses and angles.

The medial axis is a dimensionally-reduced structure that captures, in a consistent way, a notion of thicknesses and also of angles. As mentioned in Chapter 2, there is a one-to-one correspondence between the medial axis transform and the original model, which reduces to a relation between each medial axis point and its corresponding footpoints and governors. Since angles are defined between faces or edges sharing an edge or a vertex, medial axis points of which the governors are directly connected, *i.e.*, share an edge or vertex, correspond to angle dimensions. This leaves all medial axis points that have not-connected governors for the measurement of thickness. However, the classification of these medial axis points is not due to the fact that these points remain after the classification of medial axis points corresponding to angles. When people are asked what the thickness of a 5x5x5 cube is, most would answer 5. In this case the thickness is measured from a face towards the opposing face, which is intuitive. Considering the rectangular shape in Figure 3.2, a measurement of thickness from edge *c*-*e* towards edge *d*-*f*, for instance from a' on *c*-*e* towards a' on *d*-*f* is intuitive, again because they are opposing edges. Keep in mind that the measurement of thickness from a' towards a' on those edges is not along the surface normals of the points a'. For the edges *c*-*d* and *e*-*f*, a similar analogy for the measurement of thickness can be used, for instance from a' on *c*-*d* towards b' on *e*-*f*. However, this analogy can not be extended throughout the entire edge *e*-*f*; it can for the edge *c*-*d*. In other words, the entire edge *c*-*d* can be linked to the edge *e*-*f*, but not vice versa, without losing intuitiveness. This simple example illustrates that opposing edges give a good notion of thickness, *c*-*e* and *d*-*f*, but not in every instance, *c*-*d* and *e*-*f*. Incorporating the medial axis into this observation, means that the edges *c*-*e* and *d*-*f*, which are not directly connected and share the same medial axis points, give a good indication of thickness. The edges *c*-*d* and *e*-*f*, which are also not directly connected, but they do not share the same medial axis point, do not give a intuitive notion of thickness. Therefore, medial axis points that have governors which are not directly connected give a good measurement of thickness.

In conclusion, medial axis points of which the governors are directly connected, *i.e.*, share an edge or vertex, correspond to angle dimensions and medial axis points of which the governors are not directly connected give a good indication of thickness. These areas are usually bounded by junction points and vertices, but this does not hold for every case as will become clear.

This distinction between thickness and angle medial axis points subdivides the medial axis into areas capturing thicknesses and areas capturing angles, as illustrated for a 2D medial axis in Figure 3.2, where the area *a*-*b* captures thicknesses and all other areas capture angles. The subdivision is based on the junction points of the medial axis. The foot points of these junction points (a' and b') in their turn subdivide the governors and hence the boundary into areas capturing thicknesses (blue areas) and areas capturing angles (grey areas). These areas thus correspond to specific areas of the medial axis.

Any medial axis point on the line from *a* to *b* has two governors, namely the edges *c*-*e* and *d*-*f*, and as can be seen these edge are not directly connected, whereas the areas from *a* to *c* and *d* and from *b* to *e* and *f* correspond to edges that are connected.

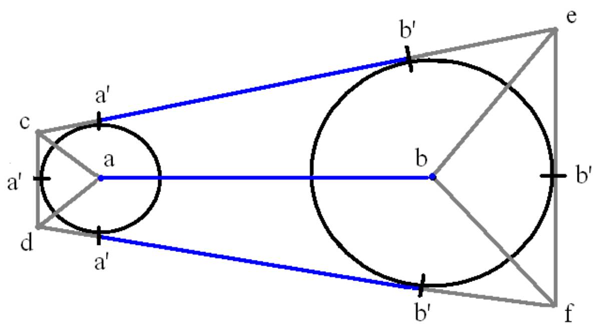


Figure 3.2: The division between thickness and angle areas in 2D, a' and b' indicating footpoints of the junction points, dividing both the medial axis and the boundary into areas capturing thicknesses (blue) and areas capturing angles (grey).

The sheets on a 3D medial axis are bounded by junction points, seam-end points and seams, as can be seen in Figure 3.3. This figure illustrates the two types of sheets capturing thicknesses (blue) and angles (grey), respectively. In a similar way as in 2D, areas on the governors capturing thicknesses and angles are determined.

As indicated before, there are not always junction points, seam-end points and seams bounding the different areas of the medial axis. For the case in Figure 3.4(a), there are no parts of the medial axis that can be directly classified as being a thickness or an an-

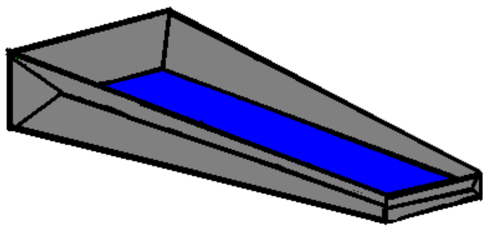


Figure 3.3: An example in 3D of the two types of sheets on the medial axis, again blue capturing thicknesses and grey angles.

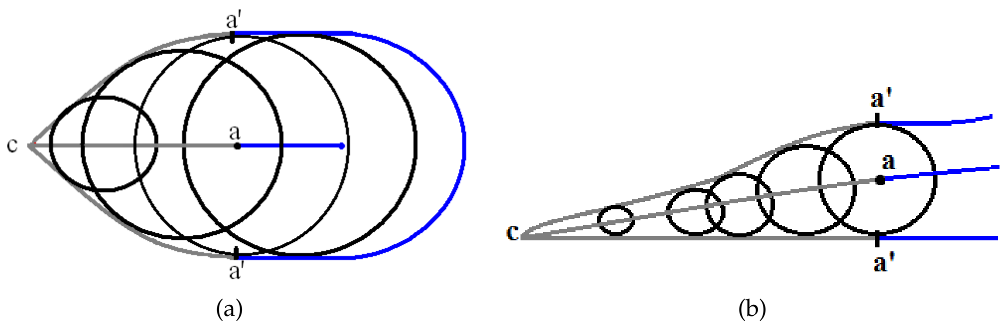
gle area, since there is no junction point. Considering the area at c, which is clearly an angle, the question arises where the angle area should stop and the thickness area should begin. In this case, the medial axis point a, and hence the footpoints a' on the boundary, give a good subdivision, since this is the first point where the radius value of the medial axis no longer increases. In this simple example, there are only a single curved edge and the vertex c as governors, but in more complicated examples an angle area from c to a is also bounded if another governor starts influencing the medial axis, even though the radius value can still increase. Another way to look at it is that this is the first point at which another part of the geometric model begins, since the radius value remains the same or decreases, or another governor starts influencing the medial axis.

As will become clear in the next section, the calculation of the angle value depends on the normals at the footpoints of a medial axis point. This means that the calculation of the angle value from c to a does not always give a constant value, illustrated in Figure 3.4(b), where the edge is curved. In this situation, a so-called "changing angle" is visualized, which will be further discussed in the next section. This approach is loosely based on the identification of such areas in the paper by Blum and Nagel [11]. They name these areas wedges, cups or flares, as illustrated in Figure 3.5. In our case, since we want to identify angle areas, the wedges, cups and flares are connected by an edge, otherwise there is no angle area.

The classification of points on the medial axis, as being either a thickness or angle value, thus works for any kind of shape, whether or not it is a polygonal shape, contains discontinuities in the boundary or is a free-form shape. The next step, the calculation of the actual dimensional values is discussed in the next section.

3.2 Extraction of Dimensions

Given the classification of points as described in the previous section, the actual dimensions still have to be extracted from the medial axis point and its corresponding



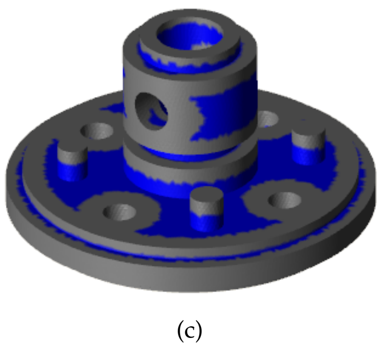


Figure 3.4: When there is no junction point present in the medial axis, a proper division of the medial axis and the corresponding governors can still be constructed, by identifying different parts of the geometric model. In (a) and (b) this is done by identifying the first point from *c* which no longer has an increasing radius value, in these cases the points at *a*. In more complicated examples the angle area can also be bounded because another governor becomes active, *i.e.*, starts influencing the medial axis. An example of the division between angle and thickness areas in 3D is given in (c), where again blue identifies thickness areas and grey angle areas.

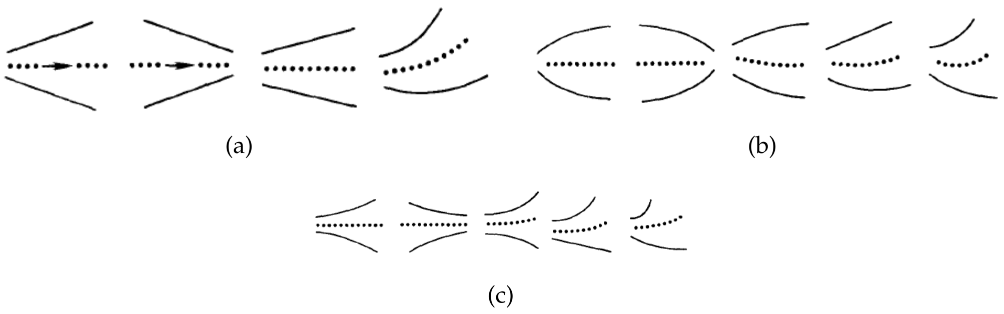


Figure 3.5: The identification of angle areas by Blum and Nagel, were these areas are called elementary descriptors (subsegments) of simplified segments, based on width properties, where width in this case means the radius values; (a) are called wedges, (b) cups and (c) flares, source, [11].

footpoints and governors, which is discussed in this section.

The calculation of the actual thickness value, if two footpoints m' correspond to a point *m* on the medial axis, is illustrated in Figure 3.6. The distance *d* between the footpoints is taken as the value for the thickness.

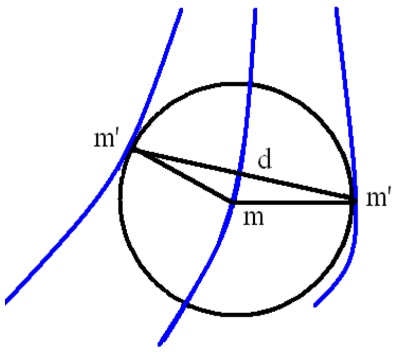


Figure 3.6: The calculation of the thickness value, for which we take the distance *d* between the two footpoints *m*′.

There are many instances in which the object's boundary and the maximal sphere of the medial axis coincide over an interval, *i.e.*, there are more than two footpoints corresponding to a medial axis point, *e.g.*, at round(ed) regions of a geometric model. In such cases there is ambiguity in calculating the thickness value, since it can be calculated between any two footpoints. In round geometric models, think of a sphere, the thickness is twice the radius value, *i.e.*, the diameter. At the limit points of the medial axis at rounded regions, as illustrated in Figure 3.7, the diameter value is taken for the entire region on the object's surface associated with the medial axis point *m*; it can be seen seen as a continuation of the thickness as extracted before the maximal sphere and the object's boundary coincided over an interval. This is chosen to illustrate that there is a rounded region and not a curved region that looks rounded, for instance an oval shaped region.

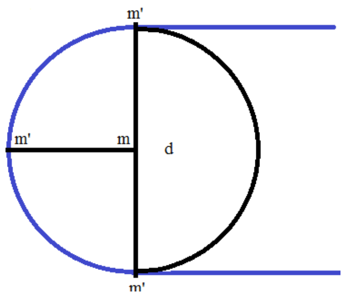


Figure 3.7: To give an intuitive thickness value in rounded regions, the diameter value of the point *m* is taken, since it can be seen as a continuation of the thickness as extracted before the maximal sphere and the object's boundary coincided over an interval.

Another situation in which the calculation of thickness values has ambiguity is at concave edges and points. As illustrated in Figure 3.8, a concave point has a range of medial axis points associated with it, in this case from *a* to *b*. The question becomes which thickness value to assign to the concave point at *a′*, because multiple thickness values can be calculated in the range from *a* to *b*. In these situations, per governor a thickness value is calculated at the limit points in the range from *a* to *b*, the black lines. To properly visualize this, an extra step in the visualization process is needed as discussed in Chapter 5. Any other non-concave footpoints associated with the range from *a* to *b* use the calculation as specified in Figure 3.6. In the case of Figure 3.8, these are the footpoints on the edges from *c* to *d*.

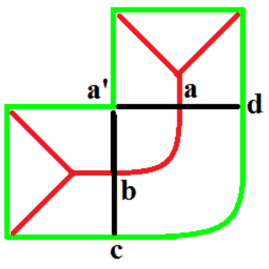


Figure 3.8: At a concave footpoint *a′* there is not one medial axis point, but an entire range of the medial axis associated with it, *i.e.*, the range from *a* to *b*. The two black lines are the thickness values that are assigned to the one footpoint *a′*, which are later used in the visualization process.

The calculation of the value for an angle *a* corresponding to a point on the medial axis, is illustrated in Figure 3.9. We take the angle *b* at the medial axis point *m*, between the surface normals at the corresponding footpoints *m′* and calculate the value at the actual corner as *a = 180 - b*.

Looking at Figure 3.3, it can be said that for some medial axis points, multiple angles can be specified, since they are on a seam of the medial axis and that seam divides the medial axis into different angle areas. At those points one angle value can be picked since they are exactly on the seam. In the visualization process, this can result in a jagged result, as discussed in Chapter 5, but as long as the mesh is dense enough this is not bothersome.

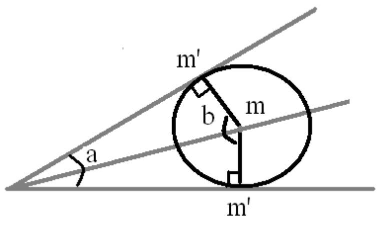


Figure 3.9: The actual calculation of the angle value *a*, for which we take *180 - b*.

Mentioned before was the situation in which there was no junction point dividing the medial axis into areas capturing thicknesses and areas capturing angles. In such situations, the division is based on the increasing radius value associated with the medial axis or another governor starting to influence the medial axis. Here, as illustrated in Figure 3.10, a changing angle is calculated. In Figure 3.10, several maximal spheres of the medial axis are illustrated each with a different angle value, *i.e.*, on the medial axis from *a* to *c* a changing angle is calculated. This results in a changing angle visualization.

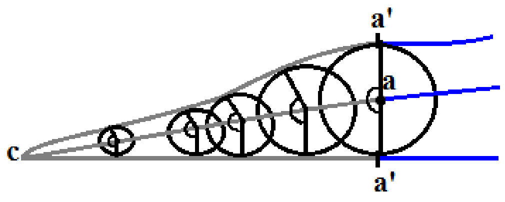


Figure 3.10: Taking the calculation of an angle as specified in Figure 3.9, different angles at different points of the angular area are calculated, which in the end visualizes a changing angle.

3.3 Visualization of Dimensions

After the classification of the points on the medial axis and the extraction of dimensions, using the footpoints and governors, the extracted dimensions are assigned to the corresponding footpoints. As mentioned in the introduction, a mesh is used as input for the pipeline. Of this mesh, all mesh vertices are considered to be footpoints of the medial axis. This means that any mesh vertex has either a thickness or an angle dimension assigned to it. This section will briefly discuss how this is used to visualize the dimensions. The exact details of this are discussed in Chapter 5.

Two distinct color maps are used to visualize both dimensions on the mesh. The thickness values are associated with a color map in the range from red to green, where red indicates the minimum thickness and green the maximum thicknes. This was chosen because small thickness values are usually not desirable, especially in injection molding. Each extracted thickness value is then made relative to the minimum and maximum thickness, resulting in each value being in the range from 0 to 1. The value at each mesh vertex is then assigned to a color, a 0 value to red and 1 to green, and any value in between to its corresponding color according to the color map. This means that every thickness mesh vertex, has a color associated with it in the range from red to green. The next step is to interpolate these colors across the triangular thickness mesh elements. This results in a complete visualization over all triangular mesh elements in the thickness areas.

A similar approach is used for angular values, where the color map ranges from pink to blue and each value is made relative between 0 and 180 degrees. These values are then assigned to colors in the corresponding color map and visualized using the same principle as for the thickness areas. There are some adjustment that have to be made to get a proper visualization, for instance at concave edges; these details are discussed in Chapter 5. Figure 3.11 illustrates the complete visualization, with the interpolation across the triangular mesh elements and in the red box the result of a changing angle visualization. What can also be seen is a jagged result between thickness and angle areas, which is due to the discrete nature of the mesh.

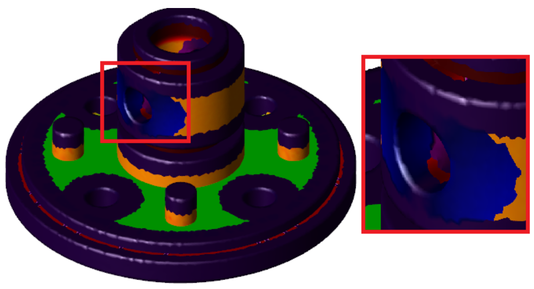


Figure 3.11: An example of the complete visualization of both the thickness and angle dimension; the close-up in the red box illustrating a changing angle visualization.

4

Computation of the Medial Axis

This chapter explains how the medial axis transform is computed, given a geometric model. Since the computation of the medial axis is hard in general and because the extraction of dimensions needs to work for any kind of geometric model, it was decided to use a mesh as input. This is not a unrealistic requirement, because most environments have meshing capabilities, either from a CAD model or from a point cloud. The algorithm expects a fairly uniform and dense (triangular) mesh, not only to get a good approximation of the medial axis, but also to get a good visualization of the dimensions.

The meshing functionality used to create a mesh from the geometric model does not contain any actual implementation work and will therefore only be briefly discussed here. At first STL (stereo-lithography) files were used as the basis for the computation of the medial axis transform. STL files usually have a rough triangulation of the original geometric model, containing a lot of bad-aspect-ratio triangles. Because of this, it was decided to use a better mesh. The ACIS geometric modeller, see [25], was used for this purpose, since it contains meshing functionality capable of generating dense meshes with good-aspect-ratio triangles, which is not only needed to accurately compute the medial axis, but also to be able to better visualize the dimensions.

The mesh, as provided by the ACIS geometric modeller, consists of two arrays; the first array, the vertex array, contains all coordinates of the mesh vertices. The second array, the elements array, indicates how to connect the mesh vertices to each other to create the mesh. For instance, the first four elements of this array can be 3, 0, 1, 2, specifying that 3 elements are part of the same mesh element, *i.e.*, the mesh vertices in the vertex array at offset 0, 1 and 2 form a mesh element, in this case triangular because three vertices are used. Since the mesher returns triangles, every fourth element is a 3. The order of the vertex array indices in the elements array is always counterclockwise; going back to the 3, 0, 1, 2, example, the edges of the triangular mesh element are from vertex 0 to 1, from 1 to 2 and from 2 to 0. This order is important for the computation of the normals, which is provided by the ACIS geometric modelling environment, because it uses the right-hand rule to determine the outward pointing normal. In the computation of the medial axis, the inward pointing normals are considered, because the medial axis is on the inside of the object, by reversing the direction of the given normals. Given the information in the elements array, it is fairly straightforward how neighboring mesh elements can be identified. For instance, the two sequences 3, 0, 1, 2 and 3, 0, 1, 3 are neighboring mesh elements since they point to two the same mesh vertices, namely the mesh vertices at offset 0 and 1 in the vertex array. And then using this information, neighboring mesh vertices can be identified. Using this information, a structure is built in which each mesh element knows which vertices form its triangle and what its neighboring mesh elements are. Each mesh vertex knows of which mesh elements it is part of and what its neighboring mesh vertices are.

Before the extraction of dimensions can start, the governors of the mesh need to be properly defined. Governors are properly defined for a CAD model, since they are the entire face, edge or vertex in which a foot point lies. For a mesh, however, the governors are not properly defined, since faces and edges are subdivided into mesh elements. If there is a link from the mesh elements to the original CAD model, the proper governors can still be identified, but if only a mesh is given, a different approach has to be used. Our approach depends on a so-called significant angle. This angle specifies whether neighboring mesh elements are considered to be smoothly connected and thus part of the same governor. This subdivides the boundary into sets of mesh elements that we consider governors of the medial axis. During the construction of the governors, a link is created between each mesh element and the corresponding governor(s). This is needed for the correct classification of the medial axis points and their foot points, for which a link is needed between a medial axis point, its foot points and the corresponding governors.

After the appropriate governors have been constructed, a connection graph needs to be built that specifies which governors are directly connected, since this is needed to determine if a medial axis point is an angle or thickness point as discussed in Chapter 3. As explained, each mesh element knows its neighboring mesh elements, and if one of these neighboring mesh elements is part of another governor, the two governors are directly connected. By comparing neighboring mesh elements also the mesh vertices between different governors can be identified and these mesh vertices, and the connectivity between them according to the mesh elements array, identify the edges of the governors; these vertices are called edge vertices.

The next step, after the edge vertices have been identified, is the identification of the concave edges and concave mesh vertices. These concave vertices are part of the edge vertices, since the mesh vertices of mesh elements that are not smoothly connected form the edge between different governors and these edges can be concave. To understand the identification of concave edges, the meaning of an edge direction vector is needed. This is the direction of an edge as provided by the elements array, counterclockwise. Going back to the example of the indices in the elements array, 3, 0, 1, 2, the edge direction vectors are the vectors from vertex 0 to vertex 1, from 1 to 2 and from 2 to 0. The test procedure for identifying a concave edge is based on defining the angle between two faces joined by a given edge. The test procedure is as follows: calculate the cross product between the normals of the mesh elements, this direction will either be identical to the edge direction vector, in which case it is a concave edge, or in the opposite direction, in which case it is a convex edge. For the example in Figure 4.1, the direction vector e_1, from vertex 1 to 2, which defines the edge e_1, is computed. Next the cross product of the normals of faces F_1 and F_2, f_1 and f_2, respectively is computed and they have the same direction because the dot product is positive, and hence it is a concave edge.

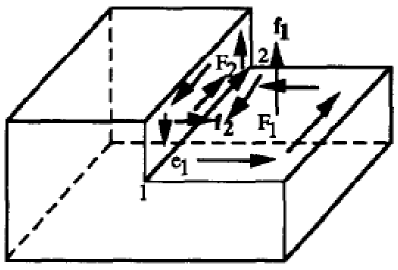


Figure 4.1: The identification of a concave edge, using the edge direction vector e_1 and the normals of the adjacent faces, f_1 and f_2

This chapter assumes that a mesh is given that approximates the geometric model without major errors and captures all significant shape characteristics. What one needs to know to understand the rest of this chapter is the term *target edge length*. The target edge length specifies the optimal size of all mesh elements of the mesh. A square can be meshed using only two triangles, but if the target edge length is set small as compared to the size of the square, more triangles are created to approximate the square. This is needed because the algorithm for computation of the medial axis requires a dense (triangular) mesh.

Section 4.1.1 discusses the general approach for our computation of the medial axis transform. It does this by explaining the principle for a continuous method that identifies the maximal sphere of the medial axis of a given point on the surface. This approach is based on the grass fire analogy of the medial axis transform, as discussed in Chapter 2. This continuous approach is then ”turned” discrete for the mesh of a geometric model, in Section 4.1.2. Since the medial axis transform is computation intensive, several constructs were used to speedup the computation of the medial axis, namely a spatial subdivision and a parallel implementation, discussed in Section 4.1.3 and Section 4.2, respectively.

4.1 Identification of Maximal Spheres of the Medial Axis

To make the computation of the medial axis plausible, first a continuous approach is discussed. Both the continuous and the discrete approach are based on the identification of maximal spheres of the medial axis transform, given a foot point on the surface.

4.1.1 Continuous Approach

The grass fire analogy states that a burning point (a foot point) f_a, on the boundary, from which the fire spreads perpendicular, *i.e.*, along the surface normal at that foot point, travels at a rate of one unit distance per unit time, and at time t the outer extent of the burned area is parallel to the boundary offset by distance t. The medial axis consists of the quench points of the fire. These points are the mid points of the maximal spheres of the medial axis. The approach will be explained here with the help of 2D figures. It identifies the maximal circles by taking a continuous sequence of circles tangent to the boundary at f_a, which all have their midpoint on the surface normal at f_a, with an increasing radius value. The smallest circle c that touches another part of the boundary, as illustrated in Figure 4.2, is the circle of which the center is the corresponding medial axis point m, and the point where the circle touches another part of the boundary another foot point f_b. The cricle that is identified in this way is maximal within the object, since it touches two distinct parts of the boundary and it is not contained within any other circle, since that circle would be larger, and therefore not be contained within the object’s boundary, *i.e.*, it would contain points outside the object in the neighborhood of f_b. The difference with the grass fire analogy is that in the grass fire analogy the fire spreads from the entire boundary until it hits another fire and douses out, whereas in our approach the fire continues until it hits another part of the boundary, as illustrated in Figure 4.2. Using this approach, a maximal sphere, given a foot point on the boundary and the corresponding surface normal, can be determined.

4.1.2 Discrete Approach

In the discrete approach, a mesh is used as input, where all vertices are considered to be foot points. In short, the algorithm traverses over all footpoints and finds the corresponding medial axis point for each footpoint by searching another footpoint of that

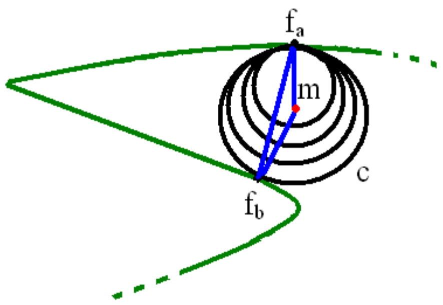


Figure 4.2: A possible continuous approach in which the maximal circle c for a foot point f_a is the smallest circle tangent to the boundary at f_a that touches another part of the boundary f_b

medial axis point. To find another footpoint, given a footpoint, a potential maximal sphere is constructed, using the criteria illustrated in Figure 4.3. For a footpoint f_a, the corresponding medial axis m point is always along the surface normal to the object boundary at f_a. Any other footpoint f_b of m has equal distance m, due to the nature of the medial axis. Using this criterion, the triangle $f_a m f_b$ can be constructed, in which the edges $f_a m$ and $f_b m$ have equal length and the angles a and b are equal. The maximal circle of m has the length of $f_a m$ as its radius, and two of its footpoints are f_a and f_b.

In each iteration, the footpoints f_a and f_b are known, together with the surface normal at f_a, which is in the direction of m, and because f_a and f_b are known, the line and corresponding length from f_a to f$_b$ are also known, and hence the angle at a is known by combining the surface normal at f_a with the line from f_a to f_b. These known values are enough to calculate the triangle as illustrated in Figure 4.3, using simple trigonometry.

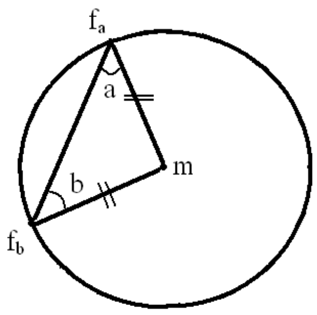


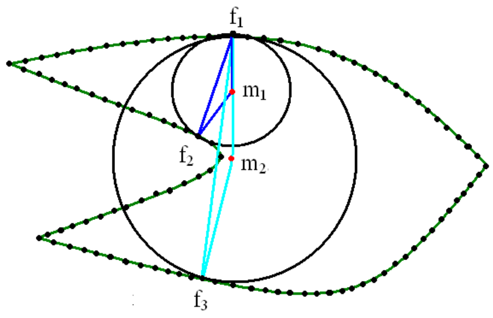
Figure 4.3: The construction of the medial axis point m, given the surface normal at f_a and the line $f_a f_b$.

In the algorithm, given a footpoint f_1, a triangle is constructed for every other footpoint f, *i.e.*, every other vertex of the mesh, by taking the same angle a at f as at f_1. If no triangle can be constructed, because $f_1 m$ and fm do not intersect, f is discarded. If a triangle can be constructed, the corresponding circle could be maximal. Figure 4.4 illustrates

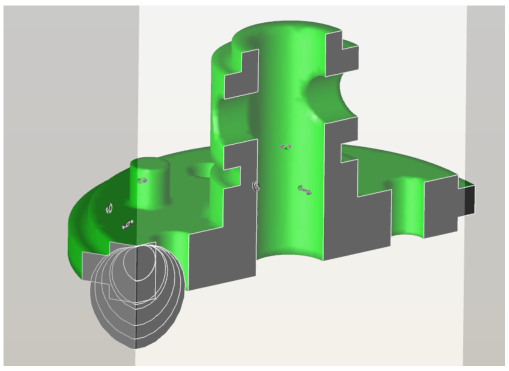
Listing 4.1: Discrete calculation of Medial Axis (simple iteration)

```
for each footpoint $f_i$
        for each footpoint $f_j$ where $j \neq i$
                extract current radius
                if(current_radius < best_match)
                        best_match = $f_j$.radius
        return best_match
```

that this can result in multiple circles. The circle with the smallest radius value is taken, since all other circles contain at least one mesh point and are therefore not a maximal circle. In this way, the algorithm finds for each footpoint another footpoint, the corresponding medial axis point and the corresponding governors, since each mesh vertex knows of which governors it is part of. At the end of the iteration a cloud of points on the medial axis has been computed. This point cloud is used as starting point for the classification and extraction of dimensions as discussed in Chapter 3. The pseudo code is given in Listing 4.1 and the complexity of this pseudo code is obviously O(n^2).



(a)



(b)

Figure 4.4: An actual situation as it can occur within a mesh; in (a) the triangle $f_1m_1f_2$ is chosen and not the triangle $f_1m_2f_3$, because the circle corresponding to $f_1m_1f_2$ has a smaller radius; (b) shows an example of a comparable situation in 3D, the inside made visible with two cut-away views.

Looking at Figure 4.4, it can be observed that for the iterations visualized, not all other footpoints matter, *i.e.*, many of footpoints are not ”within range” of the current part of the medial axis. This means that not every footpoint needs to be checked against every other footpoint, as is done in the pseudocode. The next subsection will discuss this in more detail and how this observation is used to change the pseudocode in Listing 4.1.

4.1.3 Speedup constructs

As mentioned at the end of the previous section, checking each footpoint against each other footpoint is a wasteful approach, because most footpoints do not influence the current iteration. This section will explain how this observation can be used to speedup the computation of the medial axis by discussing two contructs and how a spatial subdivision within the bounding box of the object is used to accomplish this.

Since the geometric model is completely contained within the bounding box, the maximal sphere within the bounding box acts an upper limit for any maximal sphere encountered within the object's boundary. Assigning this preliminary radius to each footpoint allows the iteration of the footpoints to consider only the footpoints within that maximal sphere. This will not always lead to a significant speedup, only if the bounding box is fairly flattened. The radius value of the maximal sphere within the bounding box is determined by taking the minimum difference between the x, y or z coordinates of the bounding box.

If a dense mesh is used for the computation of the medial axis, the radius value of neighboring footpoints of a footpoint f_1, of which the radius value is known, can be guessed within a certain accuracy. We will now consider the case in which the circle for f_1 and f_2 is known and we are going to calculate an upper limit for a neighboring footpoint f_3, as illustrated in Figure 4.5. Suppose the black circle is maximal for footpoints f_1 and f_2, and let us assume that the footpoint f_2 is also the other footpoints of f_3. Using that assumption, the circle $f_2 m f_3$ can be constructed using the criteria from Figure 4.3, resulting in the blue circle. This circle acts as an upper limit for the footpoint f_3, because any larger circle along the surface normal of f_3 will have the footpoint f_2 contained in it, and can therefore not be maximal. This means that any footpoint not within distance $f_3 m$ of point m does not have to be considered for the iteration of f_3 over all foopoints. Only the blue mesh vertices, including f_2, have to be considered, and as it turns out the footpoint f_4, resulting in the red circle, is the actual medial axis point. It can easily be seen that an even denser mesh will be able to better predict a radius value for neighboring footpoints of f_1 and f_2. If a concave footpoint f_c is encountered, its calculated guessed radius value can only be used for neighboring footpoints within the same governor, because the neighboring footpoints that are not part of the same governor are not directly connected to the medial axis point of f_c.

To effectively make use of this guessed radius value, the first step is to assign the radius value of the maximal sphere of the bounding box to each footpoint, to have an upper limit for each footpoint. The second step is to not randomly iterate over all the footpoints, but neighbor after neighbor to make optimal use of the guessed radius values

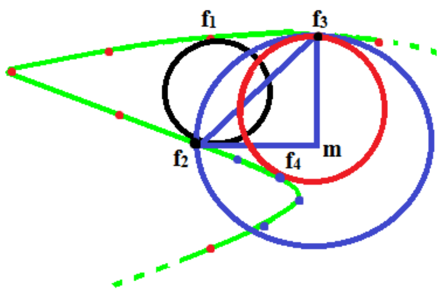


Figure 4.5: The computation of a geussed radius value for the footpoint f_3, given the maximal circle of f_1 and f_2, making sure that in the iteration for f_3 not all other footpoints, as in Listing 4.1, have to be considered.

Listing 4.2: Advanced discrete calculation of Medial Axis (advanced iteration)

```
for each footpoint $f_i$
        $f_i$.guessed_radius = bounding_box.max_radius

// iteration is neighbor after neighbor
for each footpoint $f_i$
        footpoints_range = footpoints within range of $f_i$.guessed_radius
                for each footpoint $f_j$ in footpoints_range where $j \neq i$
                        extract current radius
                        if(current_radius < best_match)
                                best_match = $f_j$.radius
                        for each neighbor $f_k$ of $f_j$
                                set guessed_radius
                return best_match
```

of neighboring footpoints. As explained, at concave vertices it is impossible to specify a guessed radius value for all neighboring footpoints and that is where the maximal radius value of the maximal sphere of the bounding box comes in, because that is already set for each footpoint, as is done in Listing 4.2. This makes the algorithm a surface sampling approach using some sort of tracing, because information from the previous iteration is used in the next iteration. The best starting point of the algorithm would be a convex edge vertex, since the radius value in that footpoint is 0. The corresponding pseudocode is in Listing 4.2.

All that remains is the ability to fastly calculate the subset of points within reach of either the value from the maximal sphere of the bounding box or the guessed value provided by one of the neighbors of a footpoint. To do this, the space within the bounding box is subdivided into rectangular cells of size target edge length, as mentioned in the introduction of this chapter. A footpoint within the bounding box knows in which cell it is, if it knows the distance in x, y and z direction to the minimum vertex of the bounding box, by dividing those distances by the target edge length. For a footpoint, given a guessed radius value, the guessed mid point of the potential maximal sphere is calculated. Of that guessed mid point, the cell in which it is, is computed. From that cell, all cells within the offset of the guessed radius value are taken and since each cell knows which footpoints are within its boundary, the subset of points that matter for the current iteration are known. The advantage of using this approach is that the subset of mesh vertices can be fastly calculated since only simple calculations are needed to know where the current mesh vertex is and what cells are within reach. This process is illustrated in Figure 4.6, where only the blue rectangular cells and the mesh vertices witin them are considered for the iteration of f_3. As can be seen, the subset of blue vertices is larger than in Figure 4.5, because of the discrete nature of the rectangular cells.

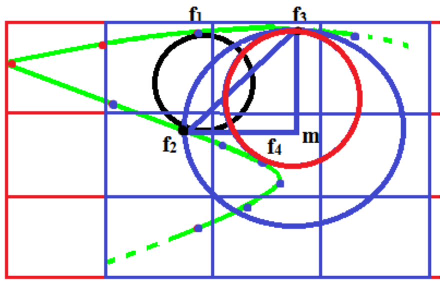


Figure 4.6: The computation of the subset of mesh vertices that matter for the iteration of f_3 using a spatial subdivision of rectangular cells.

4.2 Parallel Implementation

This section will discuss the parallel implementation that was developed using openMP, see [26]. It will first give a short introduction into parallel programming and the use of openMP, after which the parallel computation of the medial axis, without, see Listing 4.1, and with, see Listing 4.2, the spatial subdivision, is discussed.

The main idea behind parallel programming is to improve the application's performance by utilizing all processors and not just one as in sequential programming. The improvements occur when parallelism is achieved in areas of the code that have been modified to support this multi-threading. A good parallel implementation is scalable, *i.e.*, it allows performance to improve as the number of processors increases. The ideal situation is achieved when performance increases linearly as the number of processors increases.

In practice it is almost impossible to achieve this, because of the interactions between the various threads (synchronization phase) and shared resources, for which mutual exclusion and critical sections need to be introduced, which create overhead. According to Amdahl's law [27], the speedup of a program using multiple processors in parallel computing is limited by the time needed for the sequential part of the program. In the case of the complete algorithm as discussed in this thesis, that would include the meshing process at the beginning and the visualization of the dimensions at the end. The meshing cannot be made parallel, since there is no access to the code, only several parameters can be influenced. The visualization cannot be made parallel either, because the underlying architecture of HOOPS does not allow it.

Considering Figure 3.1, the four major steps in the algorithm are meshing, medial axis computation, classification and extraction of dimensions and visualization; the most time is spent in the meshing and medial axis computation. The meshing process can take up to one minute depending on the quality of the mesh required, but as explained, this can not be influenced. The medial axis computation can take several minutes if the simple iteration is used, but this is significantly improved by the speed up construct, discussed in the previous section. The last two steps take fractions of seconds, since for the classification and extraction all information is present from the medial axis computation and the visualization process is optimized within the HOOPS environment, see [28]. Therefore this section will focus on the bottleneck of the implementation, namely the computation of the medial axis.

4.2.1 OpenMP

For the speed-up of the computation of the medial axis, OpenMP was used. This is a shared-memory parallel programming language that allows C++ code to be run in parallel, which will be briefly discussed in this section.

OpenMP allows the programmer to add constructs to the program that tell the compiler which instructions to execute in parallel and how to distribute them among the threads that will run the code. This distribution is the most important part of making a program parallel, because the wrong choice for a specific distribution can lead to a worse performance than the sequential solution. OpenMP realizes a shared-memory (or shared address space) programming model. This model assumes, as its name applies, that programs will be executed on one or more processors that share some or all of the available memory. Shared-memory programs are typically executed by multiple independent threads; the threads share data, but may also have private data. It supports the so-called fork-join programming model, see Figure 4.7. Under this approach, the program starts as single thread of execution, just like a sequential program. Whenever an OpenMP parallel construct is encountered by a thread, while it is executing the program, it creates a team of threads (this is the *fork*), becomes the master of the team and collaborates with the other members of the team to execute the code dynamically enclosed by the construct. At the end of the construct, only the original thread or master of the team continues; all others terminate (this is the *join*). At the end of a parallel region is an implicit barrier synchronization; this means that no thread can progress until all other threads in the team have reached that point in the program.

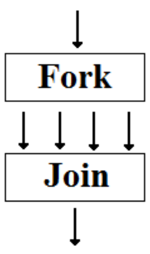


Figure 4.7: The fork-join programming model supported by OpenMP, the program starts as a single thread of execution, the initial thread. A team of threads is forked at the beginning of a parallel region and joined at the end, source [26].

OpenMP offers several tools to achieve the best possible speedup, of which the schedule clause is the most important one. The schedule clause influences the behavior of the threads, *i.e.*, it governs how the work is distributed among the threads. It is used to control the manner in which loop iterations are distributed over the threads, which can have a major impact on the performance of the program. Each thread in the team gets a so-called chunk of size 'chunk size', a contiguous nonempty subset of the iteration space. There are four constructs supported by OpenMP:

- static: iterations are divided into chunks; the chunks are assigned in a round-robin manner, in the order of the thread-number,
- dynamic: the iterations are assigned to threads as the threads request them; the threads execute the chunk of iterations and request another chunk until there are no more chunks to work on,
- guided: similar behavior to dynamic, except the size of each chunk is proportional to the number of unassigned iterations, divided by the number of threads, decreasing to the chunk size over time,
- and runtime: the decision regarding scheduling kind is made at run time.

The most straightforward schedule is static; it has the least overhead. Both the dynamic and the guided schedules are useful for handling poorly balanced and unpredictable workloads. The difference between them is that with the guided schedule, the size of the chunks decreases over time. The idea behind this scheme is that initially larger chunks are desirable because they reduce the overhead. Load balancing is often more an issue toward the end of the computation. The system then uses relatively small chunks to fill the gaps in the schedule. It is not always straightforward to select an appropriate value for the chunk size up front, since it can depend on a specific problem size and number of threads. For this purpose the schedule runtime was introduced, allowing the user to set these parameters at runtime.

The next section discusses how OpenMP is used in both the simple iteration, see Listing 4.1 and the advanced iteration, see Listing 4.2.

4.2.2 Speedup using OpenMP

In this section it is discussed how OpenMP is used to speedup the simple iteration through all footpoints, where each footpoint is compared to each other footpoint, as in Listing 4.1 and how the more advanced iteration using the spatial subdivision is performed, see Listing 4.2.

In the simple iteration, any of the constructs of OpenMP, as mentioned in the previous section, can be used, since there is no dependency between the data, *i.e.*, each iteration computes a medial axis point, using but not changing any data. The construct that is used is the dynamic construct, with a chunk size of 25. The chunk size is set fairly low, because if there are fluctuations in the time it takes to execute an iteration, these have to be divided as much as possible among the various threads, which can be accomplished by setting the chunk size fairly low. The first table lists some of the achieved speed-ups, and as can be seen it is almost linearly, which suggests that the simple iteration is highly scalable.

The simple iteration does not make use of the speedup construct as discussed in Section 4.1.3. A possible situation of this is illustrated in Figure 4.8(a), where an array and four threads are illustrated. The array contains mesh vertices and if mesh vertices have the same color they are neighbors of each other. The 'chunksize' in this example is three, since each thread computes three consecutive medial axis points. As can be seen, the iterations per thread do not make use of the fact that there are neighboring footpoints.

Model	# Vertices	# Triangles	1 Core	2 Core	Speedup	4 Core	Speedup
1	38866	77728	141.5 s	70.6 s	2.0	36.1 s	3.9
2	42247	84502	168.3 s	84.3 s	1.9	43.2 s	3.8
3	52028	104080	216.3 s	131.6 s	1.9	66.2 s	3.9

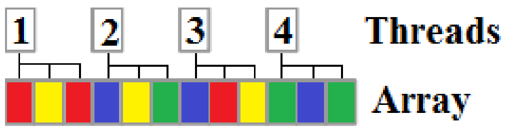
In the advanced iteration over all footpoints, as in Listing 4.2, the iteration is neighbor after neighbor. To make the most use of the parallel construct of OpenMP and the speedup construct, the array needs to be reorganized to make sure that consecutive mesh vertices of size 'chunksize' are neighboring footpoints. This is illustrated in Figure 4.8(b), where each chunk of size three, contains neighboring footpoints. In this way, there is only dependency between the data per chunk and hence per thread, but not among different threads.

Each first iteration of a chunk cannot make use of a guessed radius value, since none is provided, otherwise there would have to be data dependency between different chunks and hence between different threads. The guessed radius value is provided for every subsequent neighboring footpoint within the same chunk. In the sequential advanced iteration, there may only be a couple of footpoints for which no guessed radius value can be specified, for instance at concave mesh vertices. But in the parallel iteration, every first iteration of a chunk has no guessed radius value, other than the radius value of the maximal sphere of the bounding box, this means that the parallel iteration does more work than the sequential one and hence a linear speedup is impossible, as can be seen in the second table.

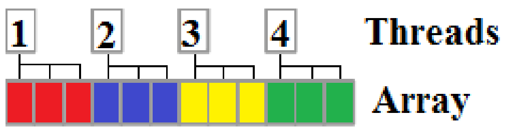
Model	# Vertices	# Triangles	1 Core	2 Core	Speedup	4 Core	Speedup
1	38866	77728	34.5 s	21.6 s	1.6	11.5 s	3.0
2	42247	84502	39.1 s	23.0 s	1.7	12.6 s	3.1
3	52028	104080	41.6 s	29.7 s	1.4	16.6 s	2.5

In conclusion, the parallelization of the simple iteration over all footpoint is highly scalable and, because the number of threads on personal computers keeps increasing, shows promise for the future. The more advanced iteration over all footpoints, using the spatial subdivision, is less scalable and has overhead in creating the work items and at some point in the future the simple iteration will catch up.

Looking only at the sequential solutions, the speedup, between the simple and advanced iteration, is somewhere between 4 and 5 for the test models, but since the advanced iteration is less predictable, depending on how good the prediction of guessed radius values is, this speedup is only an indication of the speedup that can be achieved with the speedup construct using the spatial subdivision.



(a)



(b)

Figure 4.8: The difference in parallel execution between the simple and advanced iteration over all footpoints, where the colors represent neighboring footpoints and as can be seen the array for the advanced iteration is sorted according to neighboring footpoints, making sure the parallel implementation makes the most use of the speedup construct.

5

Visualization of Dimensions

The visualization process is done entirely with the help of the HOOPS Graphics Development Platform. This chapter will focus on the preparation of the mesh and the corresponding dimensions to properly visualize all extracted information. Section 5.1 starts with the basics of the visualization process using HOOPS, after which, in Section 5.2, the visualization of both the thickness and angle dimensions is discussed. Finally, Section 5.3 discusses some of the adjustments that had to be made to the visualization process, in order to properly visualize the dimensions.

5.1 Basic Visualization

HOOPS is a graphics development platform, see [28], which offers many tools to fastly build a scene graph and visualize geometric models in a variety of ways. This section will explain which options of HOOPS are used in the visualization process.

HOOPS offers a rendering option, the color interpolation feature, that allows a color interpolation across faces and edges, when different colors are set on each of the vertices of a face. The effect of a simple color interpolation is illustrated in Figure 5.1(a). In this figure, the red markers illustrate the vertices of a simple mesh, which have colors associated with them, in this case blue and goldish colors, and as can be seen the color interpolation feature interpolates the colors across the mesh elements. In most cases when color interpolation is employed, individual colors have a specific value or range of values associated with them. In our case, this is also true, since the thickness and angle dimensions need to be mapped to certain colors in different color maps. In order to accomplish this, the color index interpolation option is used, as illustrated in Figure 5.1(b). This options interpolates between the blue and goldish colors, according to a color map. This color map specifies that the colors green and purple are in between, with thicker bands than the blue and goldish colors in the color map. To get a blended effect, both options, color interpolation and color index interpolation are used, as illustrated in Figure 5.1(c), source [28].

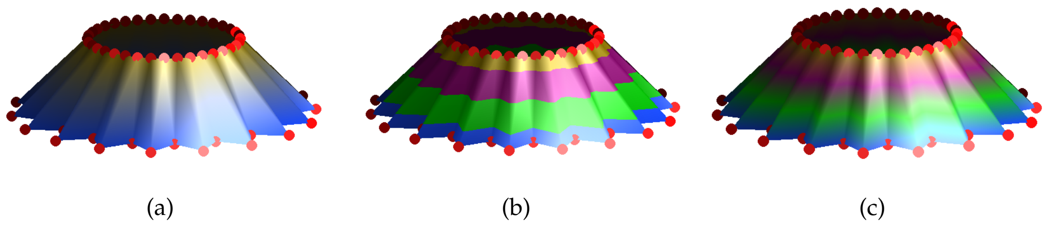
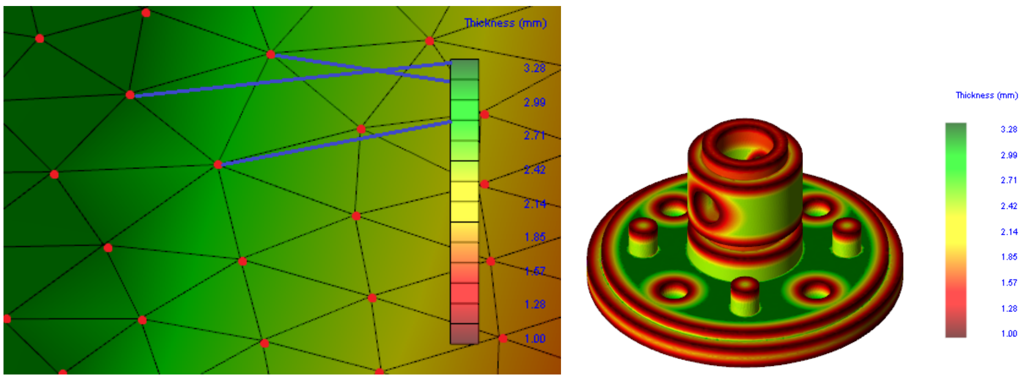


Figure 5.1: The color interpolation functionality provided by HOOPS; (a) simple linear interpolation between two colors, (b) interpolation as provided by a color map, and (c) color interpolation provided by a color map with a blended effect, which is a combination of (a) and (b).

In Chapter 4, it is explained how the medial axis is computed and in Chapter 3 how, given this computation, points on the medial axis are classified as being either a thickness or an angle dimension point. These dimensions are then extracted and assigned to the footpoints. As explained in these chapters, the mesh vertices are the footpoints of the medial axis transform, and only those were used for the computation of points on the medial axis. For the visualization process, the entire mesh is used and the dimensional values at the mesh vertices are interpolated across the mesh elements using the HOOPS color interpolation functionality. As explained in the beginning of this section, this interpolation is done on the basis of a color map, using the color index interpolation functionality of HOOPS. Figure 5.2(a) illustrates how this works for one of the mesh elements; at the mesh vertices, red markers, dimensional values are given, and those values are then mapped to the appropriate colors in a color map, blue lines. These colors are then assigned to the mesh vertices and interpolated across the entire mesh element according to the color map. In this case, the color map ranges from red to green in a linear fashion and hence the interpolation across the mesh elements is also linear. The visualization provided in Figure 5.2(b), illustrates the thickness visualization as defined by [4], where it can be seen that at every corner or discontinuity the thickness measurement is 0.

The next section will illustrate how the HOOPS functionality is used to create a complete visualization of both dimensions, thicknesses and angles, using two different color maps.



(a) (b)

Figure 5.2: The basic visualization process, where (a) illustrates a close-up and how the values are interpolated across the triangles using a color map, and (b) the complete result of this visualization, which illustrates the thickness visualization according to [4].

5.2 Complete Visualization

In the previous section it was mentioned that the dimensional values are mapped to colors in a color map. This section will further elaborate on this and illustrate the complete visualization of both dimensions.

Since the range in which the thickness values lie is unbounded, the thickness values are made relative to the minimum and maximum thickness values, resulting in a value between 0 and 1. These values are then mapped to the indices of a color map. If the resulting value is 1.5, it is not rounded or cut off to 1 or 2, because the HOOPS functionality will compute which color is exactly at 1.5 of the color map, resulting in a mapping that is correct. The color map for the thickness values ranges from red to green, as illustrated in Figure 5.2(b) and Figure 5.3. Red was chosen for the minimum thickness, since thin regions are usually not desirable, especially in injection molding.

The angle values are also made relative between 0 and 1, against a maximum angle of 180. The color map for the angle values ranges from pink to blue. Looking at Figure 5.3, it can be seen that there is not a lot of change in colors from 135 to 180 degrees, since these values rarely happen, especially in meshes considering the significant angle from Chapter 4. The other values are more common and hence have more variation in colors. This shows that the interpolation across the mesh elements associated with angle values is not linear, because it is based on the angle color map, which is not a linear color map. Although, it will become clear that many angle areas have a constant color.

The example in Figure 5.3 illustrates that the distinction between the two color maps, looking at the complete visualization, immediately becomes clear, since the colors in the two color maps do not overlap.

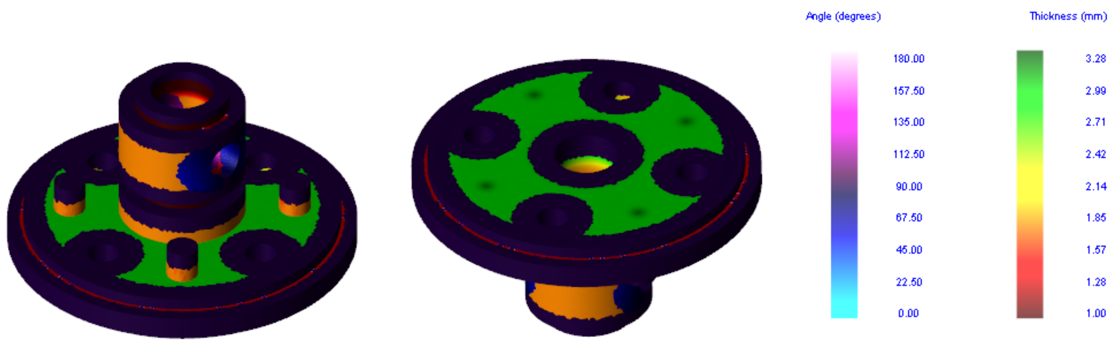


Figure 5.3: The complete visualization of both dimensions, using two different color maps.

5.3 Adjustment of the Visualization

As explained in the previous sections, the dimensions are interpolated across the triangles, and depending on the classification of the footpoints, a different color map is chosen for the visualization. This results, within a thickness or within an angle area, in a smooth result of the dimensions across the areas. This, however, does not give a desirable result between the two areas. HOOPS does not allow two different color maps to be used for one mesh, so a trick was used to make it look like two color maps were used. One color map containing both the colors from the thickness and angle values is used, where the thickness values are mapped to indices of colors related to thickness values and angle values to indices related to angle colors. Since this is one color map, the interpolation across the mesh elements that contain mesh vertices in both areas, results in a undesirable result, as illustrated in Figure 5.4. This results in undesirable, because the resulting interpolation is not defined, *i.e.*, it is impossible to interpolate from an angular value towards a thickness value. The desirable result would be a sharp transition of colors, clearly indicating the end of a thickness area and the beginning of an angle area.

Another situation, in which the visualization process gives an undesirable result, is at concave vertices. As explained in Section 3.2, at concave footpoints more than one value for a dimension is computed, see Figure 3.8; the number of computed dimensions is equal to the number of governors at that concave footpoint. If one dimensional value is calculated, the resulting color interpolation could look like Figure 5.5. Another way to look at it is, that the growing sphere of the medial axis transform would suddenly make a jump with its radius value and associated dimensions, resulting in a interpolation across the triangles that does not make sense.

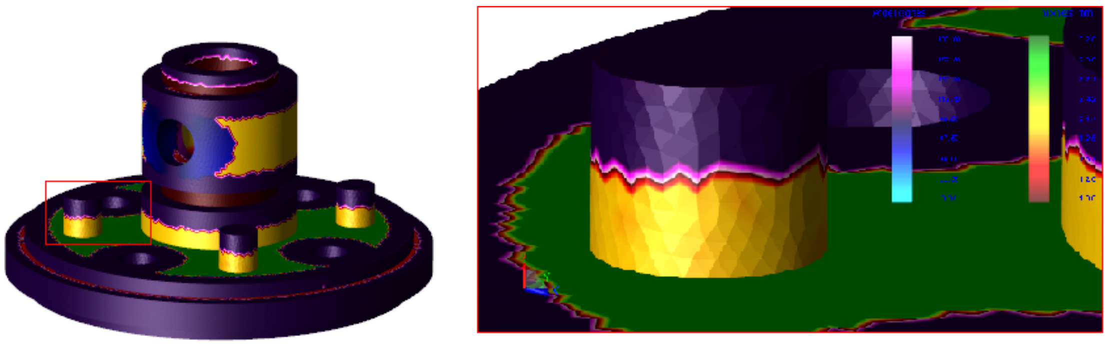


Figure 5.4: Invalid interpolation between a thickness and an angle area.

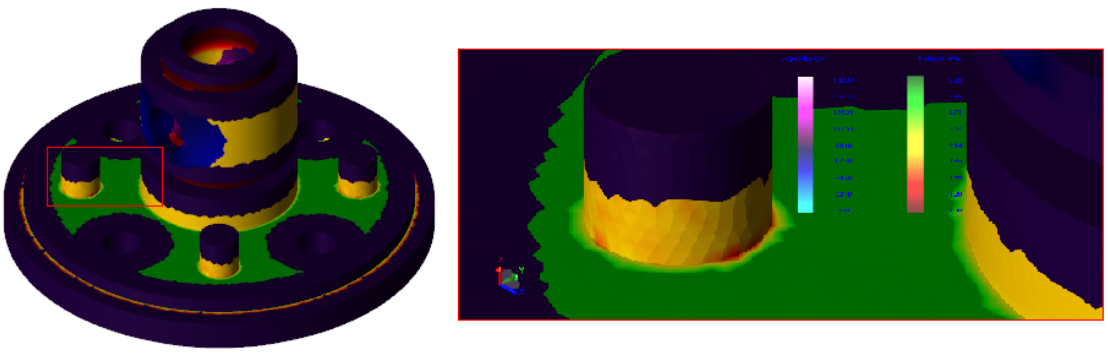


Figure 5.5: Undesirable interpolation at concave vertices, since the resulting visualization is no longer related to neighboring footpoints.

To adjust the visualization, at mesh elements having mesh vertices in both thickness and angle areas, the visualization process is adjusted, as illustrated in Figure 5.6 for a close-up of a mesh. This mesh contains several mesh vertices, indicated by the red dots, of which one of its neighbors is a thickness point and another is a angle point. Considering the point *c*, classified as an angle point and connected to two thickness points, *a* and *b* and two angle points, *d* and *e*, the latter two points will do a correct visualization, but the points *a* and *b* not, since they would interpolate from a thickness value to an angle value. All vertices, indicated by the red dots, have a similar problem, a thickness value interpolates to an angle value or vice versa. These vertices get appointed two dimensional values and associated color, one being a thickness dimension and the other an angle dimension. This allows the neighboring angle mesh vertices to interpolate towards an angle color and neighboring thickness mesh vertices towards a thickness color, resulting in a sharp transition of the colors between those mesh elements. In the case of Figure 5.6, the triangle *abc* interpolates *a* and *b* towards the thickness value at *c* and the triangle *cde* interpolates *d* and *e* towards an angle value at *c*. This also illustrates why there is a jagged result between thickness and angle areas, since the mesh vertices are not exactly on the edge between a thickness and angle area, but deviate a little from it, more on this on the end of this section.

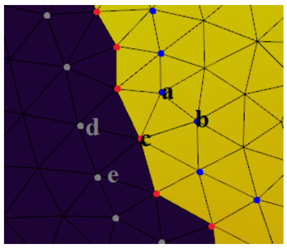


Figure 5.6: Adjustment of the visualization between thickness and angle areas, where the red mesh vertices have more than one dimensional value associated with them, allowing a sharp transition between colors between the two different areas.

The same principle, letting vertices interpolate towards different colors at the same mesh vertex, is used at concave vertices, as illustrated in Figure 5.7. But in this situation the dimensional value can be duplicated more than once, depending on the number of governors of the concave footpoint. For each governor, the dimensional value is calculated and all neighboring non-concave vertices interpolate the color according to the dimensional value calculated for their governor.

This results, as illustrated in Figure 5.7, in a sharp transition between the colors at concave vertices. It illustrates that for the concave vertex at the red marker, there are two governors associated with it, one being the green area and the other the yellow area. The yellow area has two footpoints that interpolate towards the concave vertex for the dimensional value calculated for their governor, and the green area three footpoints. The other two neighboring vertices of the concave footpoint are not considered in the interpolation, since they themselves are concave footpoints.

If the mesh vertices close to an edge between a thickness and an angle are known, as was the case in Figure 5.6 a valid question is whether or not the jagged result can be removed. An attempt was made to accomplish this. In Figure 5.8 the red line indicates the edge between the thickness and angle area and the green dots the intersection of this edge with the edges of the mesh elements. We know the exact radius values at the vertices indicated by the blue dots, but based on that information, we cannot determine where the green dots are, because we do not know the exact radius values at the green dots. This means that we cannot constructs the edge and hence the jagged result cannot be removed. Would the radius value on an edge be constant, the jagged result could be removed, but this is not always the case, as in the figure, where the radius value on the edge differs from 1.3 to 1.6. What can be done to diminish the jagged result is to remesh only the mesh elements which the edge intersects, but this only has added value if the mesh was not dense to begin with.

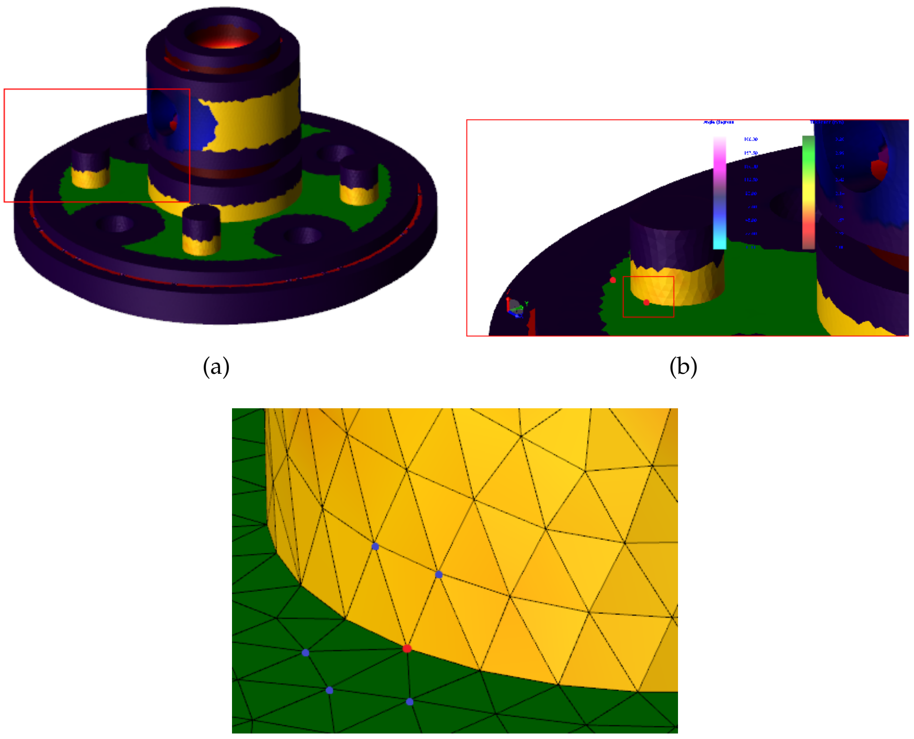


Figure 5.7: The adjustment of the visualization between thickness and angle areas and at concave vertices (a), the close-ups (b) and (c) further illustrating the adjustments.

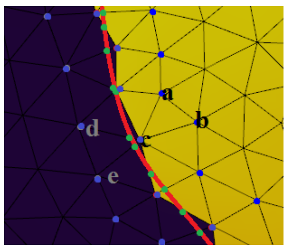


Figure 5.8: The removal of the jagged result is impossible since the exact location of the edge is not known, local remeshing of the mesh elements through which the edge (red line) goes is an option to diminish the jagged result, but this has the same effect as using a denser mesh to begin with.

6

Results

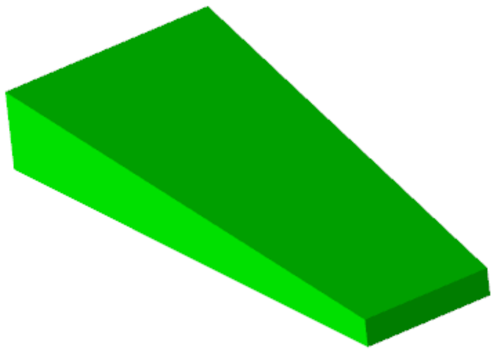
This chapter will discuss various examples of the complete visualization, starting with a simple example, after which three more complicated examples are discussed.

The first example, a simple polygonal model, illustrates the general principle and that the distinction between the two types of areas is clear. Figure 6.1(a) shows the original model and Figure 6.1(b) only the visualization of the thickness dimension. Figures 6.1(c) and 6.1(d) illustrate the general principle of our visualization, where both dimensions, thicknesses and angles, are visualized.

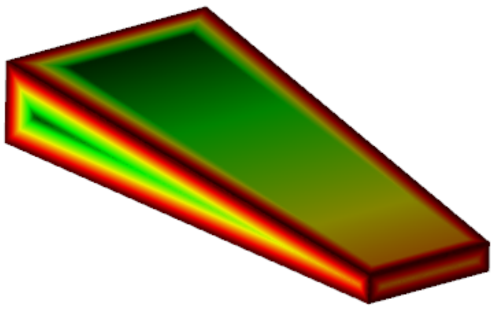
Comparing Figure 6.1(b) to Figures 6.1(c) and 6.1(d), it can be seen that visualization of both dimensions offers better insight into the model, since the differences between various angles within the model are now visualized instead of a decreasing thickness towards the edges in these areas. This model is a good example that shows that the jagged result between thickness and angle areas is not bothersome.

The second example, encountered in almost all previous chapters, illustrates various interesting properties. Looking at Figure 6.2(a) it appears as if all instances of a feature are the same and one would expect the visualization as in Figure 6.2(b). However, looking at Figure 6.2(c), which gives a visualization of the model with the method introduced here, it can be seen that one of the features has more red, which indicates that it is thinner, something easily overseen by just looking at the model without a visualization of dimensions. Figures 6.2(d) and 6.2(e) illustrate the effect of the changing angle, a smooth change in angle across the surface, where Figure 6.2(e) shows a cut-away view of the inside.

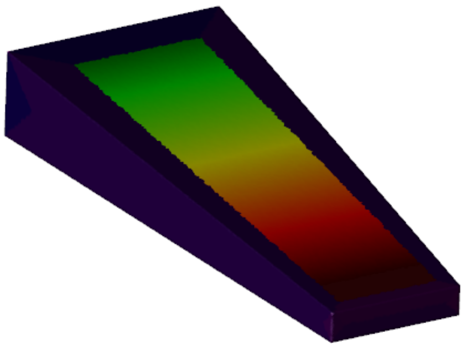
A third example of a model, Figure 6.3, containing almost no angles, except at two areas encountered at the front and the back of the model. What is interesting with this model is that the visualization of the thickness brings out an important property, namely that the inside of the object is not completely smooth, but contains some sort of bump and



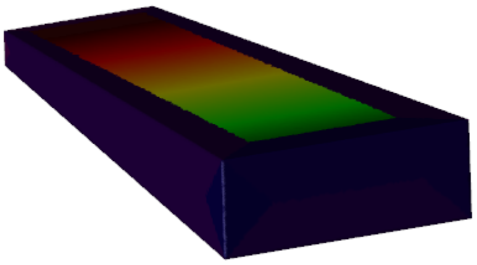
(a) Simple polygonal model, with various different angles



(b) Visualization of only the thickness dimensions



(c) Visualization of both the thickness and angle dimensions, front view

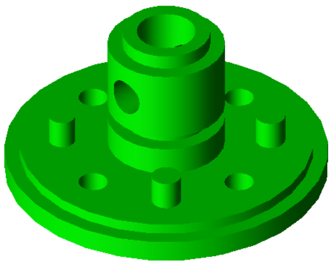


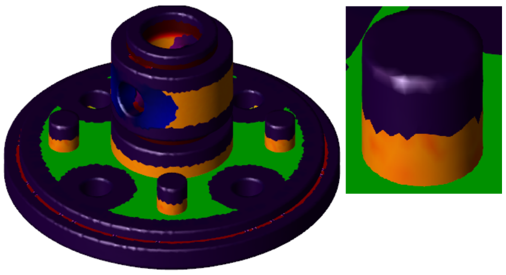
(d) Visualization of both the thickness and angle dimensions, back view

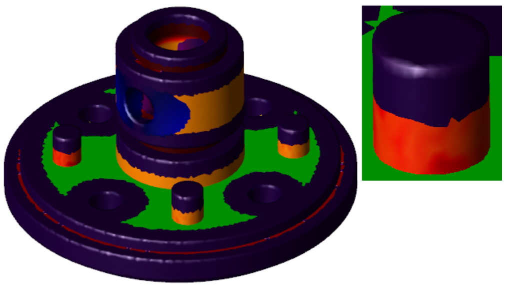
Figure 6.1: Visualization of a simple polygonal model; the original model (a), only the thickness visualization (b) and the complete visualization in (c) and (d).

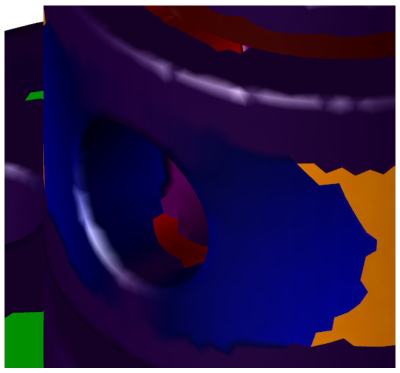
further inspection may be needed to determine the shape at the inside of the model.

The final example, see Figure 6.4, illustrates the difference between non-rounded, see Figure 6.4(a) and rounded regions, see Figure 6.4(b). The difference can easily be seen. As explained in Chapter 3, the diameter is taken in rounded regions and as can be seen this is intuitive.


(a) The original model


(b) Visualization if all features encountered within the model are the same


(c) Visualization if one of the features differs from the others


(d) A close-up of a changing angle area

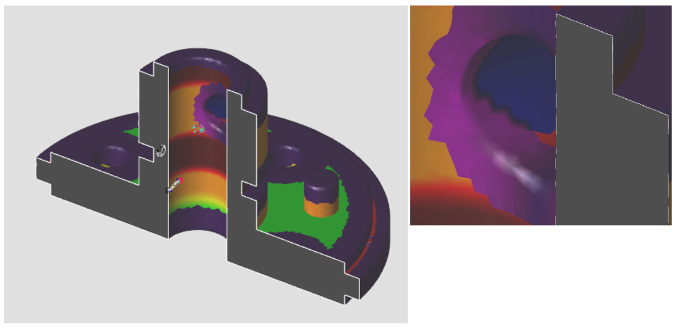
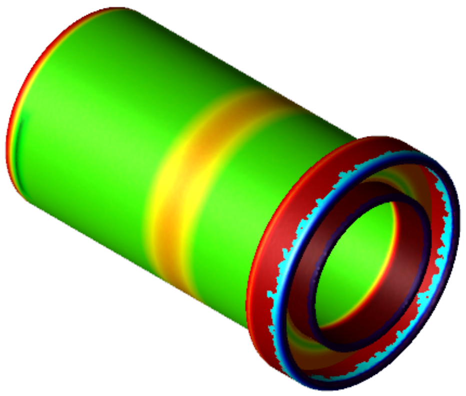

(e) Another close-up of a changing angle area

Figure 6.2: A more complicated example of a complete visualization, illustrating in (a) to (c) the detection of a feature that differs from the others, and (d) and (e) examples of changing angle areas encountered within the object.


(a) Front view of a simple rounded model

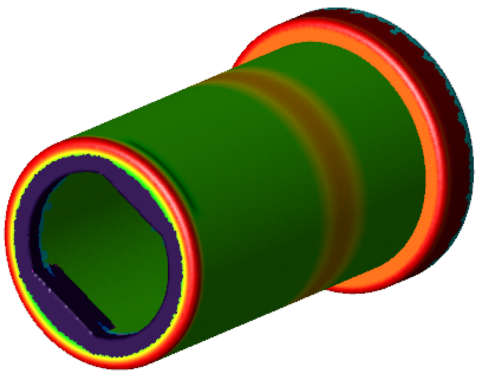
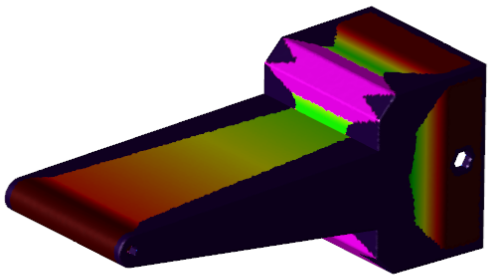

(b) Back view of a simple rounded model

Figure 6.3: A simple rounded model illustrating that the visualization brings out properties of the inside of an object.


(a)

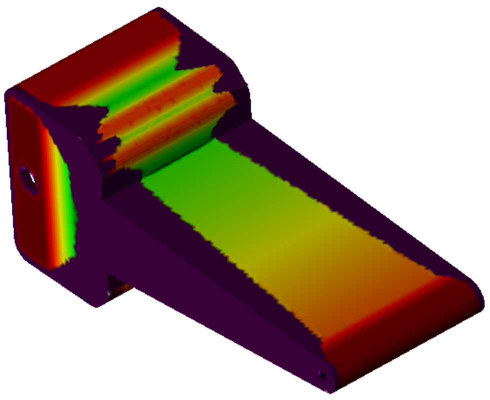

(b)

Figure 6.4: The difference between non-rounded (a) and rounded (b) regions.

7 Conclusions

This thesis discusses a method for the extraction and visualization of certain dimensions, specifically thicknesses and angles, from a geometric model. It does this on the basis of its medial axis, which is a dimensionally reduced structure, unique for each geometric model. Points on the medial axis and corresponding footpoints on the surface of the geometric model are classified as thickness or angle points, after which the corresponding dimensions are extracted and visualized on a mesh of the geometric model. The dimensional values are interpolated across the triangles of the mesh and depending on the classification of the footpoints of the medial axis, *i.e.*, mesh vertices, a different color map is used.

The visualization gives intuitive feedback, making a clear distinction between thickness and angle dimensions, using different color maps. In the introduction, some of the guidelines related to injection molding were discussed. Most of these design guidelines have to do with the thickness within the object; is the thickness of a rib 40% compared to the part wall thickness, is the thickness as uniform as possible and many others. Visualization of only the thickness dimensions would catch many of these design guidelines, but it would not give good insight into the angular regions. In angular regions, taking only the thickness, a decreasing thickness towards the edges of the object would always be visualized. This does not give a good insight into what happens at those regions, because a decreasing thickness value only tells you that there is an angle and not how large the angle is. The visualization of the angle dimensions in such areas offers more insight into geometric models.

Also mentioned in the introduction, is the case in which the parametric design dimensions do not transfer between different packages. An example was shown in Chapter 6, that illustrates that the relationship between various similar features can be reconstructed by the designer, given the insight the visualization offers. Without the visualization, the designer would have to manually check all dimensions to determine whether the features that look similar are indeed similar.

Visualization of both the thickness and angle properties can be used to verify many properties, but of course not all of them. The method discussed in this thesis is an approach capable of visualizing dimensions from any shape and size. An attempt was made to adjust the design given the extracted dimensions to, for instance, make a uniform part wall thickness possible. However, the problem encountered was that there is no intent of the designer within the geometric model, only geometric information, and adjusting the design without further knowledge does not seam feasible.

Future work involves the removal of the jagged results between the thickness and angle areas. An attempt was made to accomplish this, but as explained in Chapter 5 this would not yield a lot of added value, since it would not completely remove the jagged results only making them less. In order to completely remove the jagged result a link to the original geometric model can be used.

Chapter 2 mentioned the non-generic cases of the medial axis. An example of a special case is a cube-like object in which there are no sheet points capturing thickness, but only one seam that contains this information. Since seams are not mapped to the boundary, the visualization only offers information on the angle dimensions and not on thickness dimensions. In such situations, the visualization needs to be adjusted to incorporate some thickness information. The adjustment of such cases will probably not visualize the angles over the entire surface, but only at a certain offset from the boundary, leaving room for the visualization of thickness information. In realistic models, these special cases rarely occur.

In conclusion, the extraction and visualization of the core dimensions, thicknesses and angles, gives a good insight into the geometric information contained within a model, and it works for almost any kind of shape.

References

[1] T.M. Bahlen, W.F. Bronsvoort, and A.D. Spence. Extraction and visualization of dimensions from a geometric model. *Computer-Aided Design and Applications*, 7(4):579–589, 2010.

[2] Rebling Plastics, http://www.reblingplastics.com/quality.htm.

[3] Custom Part Net, http://www.custompartnet.com/wu/InjectionMolding.

[4] J. G. Lambourne, D. Brujic, Z. Djuric, and M. Ristic. Calculation and visualisation of the thickness of 3d cad models. *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications*, pages 340–344, 2005.

[5] H. Blum. A transformation for extracting new descriptors of shape. *Models for the Perception of Speech and Visual Form; proceedings of a symposium*, pages 362–380, 1967.

[6] R. Tam and W. Heidrich. Shape simplification based on the medial axis transform. *VIS '03: Proceedings of the 14th IEEE Visualization*, page 63, 2003.

[7] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, and S.W. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1):13–32, 1999.

[8] J. Bruck, J. Gao, and A. Jiang. Map: medial axis based geometric routing in sensor networks. *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 88–102, 2005.

[9] E.C. Sherbrooke, N.M. Patrikalakis, and E. Brisson. An algorithm for the medial axis transform of 3d polyhedral solids. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):44–61, 1996.

[10] D. Attali, J. Boissonnat, and H. Edelsbrunner. Stability and computation of medial axes a state-of-the-art report. *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, 2004.

[11] H. Blum and R.N. Nagel. Shape description using weighted symmetric axis features. *Pattern Recognition*, 10(3):167–180, 1978.

[12] J.W. Brandt. Describing a solid with the three-dimensional skeleton. *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, 1830:258–269, 1992.

[13] F.E. Wolter. Cut locus and medial axis in global shape interrogation and representation. *MIT Design Laboratory Memorandum*, (92-2), 1993. Revised version.

[14] R.A. Katz and S.M. Pizer. Untangling the blum medial axis transform. *International Journal of Computer Vision*, 55(2-3):139–153, 2003.

[15] H.I. Cho, S.W. Choi, and H.P. Moon. Mathematical theory of medial axis transform. *Pacific Journal of Mathematics*, 181:57–88, 1997.

[16] D.T. Lee. Medial axis transformation of a planar shape. *Pattern Analysis and Machine Intelligence*, 4(4):363–369, 1982.

[17] R. Fabbri, L.F. Estrozi, and L.F. Costa. On voronoi diagrams and medial axes. *Journal of Mathematical Imaging and Vision*, 17:27–40, 2002.

[18] R. Ramamurthy and R. T. Farouki. Voronoi diagram and medial axis algorithm for planar domains with curved boundaries — ii: detailed algorithm description. *Journal of Computational and Applied Mathematics*, 102(2):253–277, 1999.

[19] T. Culver, J. Keyser, and D. Manocha. Accurate computation of the medial axis of a polyhedron. *SMA '99: Proceedings of the fifth ACM symposium on Solid modeling and applications*, pages 179–190, 1999.

[20] M. Ramanathan and B. Gurumoorthy. Constructing medial axis transform of planar domains with curved boundaries. *Computer-Aided Design*, 35(7):619 – 632, 2003.

[21] H. Tek and B.B. Kimia. Symmetry maps of free-form curve segments via wave propagation. *International Journal of Computer Vision*, 54(1-3):35–81, 2003.

[22] T.K. Dey and W. Zhao. Approximate medial axis as a voronoi subcomplex. *SMA '02: Proceedings of the seventh ACM symposium on Solid modeling and applications*, pages 356–366, 2002.

[23] G.M. Turkiyyah, D.W. Storti, M. Ganter, H. Chen, and M. Vimawala. An accelerated triangulation method for computing the skeletons of free-form solid models. *Computer-Aided Design*, 29(1):5–19, 1997.

[24] H. Dey, T.K. Woo and W. Zhao. Approximate medial axis for cad models. *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 280–285, 2003.

[25] ACIS 3D Modeling, http://www.spatial.com/products/3d-acis-modeling.

[26] OpenMP, http://www.openmp.org.

[27] Vipin Kumar. *Introduction to Parallel Computing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[28] HOOPS Graphics Development Platform, http://www.techsoft3d.com.

Appendix

Paper CAD’ 10 Conference [1]

Extraction and Visualization of Dimensions from a Geometric Model

Terence M. Bahlen[1], Willem F. Bronsvoort[2] and Allan D. Spence[3]

[1]Delft University of Technology, t.m.bahlen@student.tudelft.nl
[2] Delft University of Technology, w.f.bronsvoort@tudelft.nl
[3]McMaster University, adspence@mcmaster.ca

ABSTRACT

Many applications require information on dimensions of a geometric model, but these are usually not all explicitly present in the model. A method is introduced that extracts and visualizes information on dimensions from a geometric model. It first computes the medial axis of the model, and then uses this representation to determine certain dimensions in the model, in particular thicknesses and angles, and to visualize these on the boundary of the model. Presented results show that the method can visualize important dimensional information in a geometric model.

Keywords: geometric modeling, dimensions, medial axis transform, visualization.
DOI: 10.3722/cadaps.2010.579-589

1 INTRODUCTION

Dimensions, such as thicknesses and angles, play an important role in many CAD/CAM applications. An example is in the design of products to be manufactured by injection molding. There are many design guidelines related to thicknesses and angles for a proper CAD model, to avoid potential problem areas that cause warpage and surface sink marks during the injection molding. This research focuses on thickness and angle dimensions, because these two dimensions are sufficient to get the required insight into the dimensions of the geometric model for injection molding and many other applications.

A major problem is that usually not all dimensions are explicitly present in a model, and thus have to be made explicit when needed. Three situations where this occurs will be mentioned here. First, in a CAD system only a limited number of dimensions are explicitly input by the designer, and all other dimensions are implicitly defined. Second, when a CAD model is transferred from one system to another system, often the parametric design dimensions do not transfer, and only a geometric model remains. If dimensions are needed in the other system, these have to be recovered from the geometric model. Third, when an object model has been created with reverse engineering, e.g. with laser scanning, again no dimensions are present in the geometric model, and so, if needed, have to be computed.

In this paper, a method is introduced for extracting dimensions from a geometric model in a consistent way. It computes the medial axis, or skeleton, of the model, determines dimensions on the basis of this representation, and visualizes these in a straightforward way on the boundary of the model.

With the method, dimensions in a model can be verified, e.g. in designs for injection molding as mentioned above, but also certain properties, such as symmetry or the lack thereof, can be detected. For example, a model created with reverse engineering can contain multiple instances of the same feature and visualizing the dimensions of each of these features, allows the user to see whether they are indeed the same. One of the features may differ from the others, because it endured larger forces and has therefore a larger amount of wear.

Section 2 gives background information on the definition and possible computation of the medial axis transform of a model. Section 3 explains how we define dimensions on the basis of the medial axis and how they can be calculated. Section 4 describes how the medial axis is computed in our method and how the dimensions are visualized on the boundary of the model. Section 5 shows some results. Section 6 enumerates our conclusions on the method.

2 MEDIAL AXIS TRANSFORM

The medial axis transform was introduced in [1] to describe biological shapes and is sometimes referred to as the skeleton of an object. Since then the medial axis transform has been extensively researched and developed in many areas involving shape analysis. The medial axis is used in shape simplification [9], shape matching [8], routing in sensor networks [2], etc.

The definition of the medial axis transform given in [7] is as follows. *Let D be a subset of R^n. The medial axis is the locus of points which lie at the centers of all closed spheres which are maximal within D, together with the limit points of this locus. A closed sphere is said to be maximal in D if it is contained in D, but not a proper subset of any other sphere contained in D. The radius function of the medial axis of D is a continuous, real-valued function defined on the medial axis, whose value at each point on the medial axis is equal to the radius of the associated maximal sphere. The medial axis transform is the medial axis together with its associated radius function.* Fig. 1 illustrates an example of a medial axis in 3D, and Fig. 2 examples of 2D medial axes.

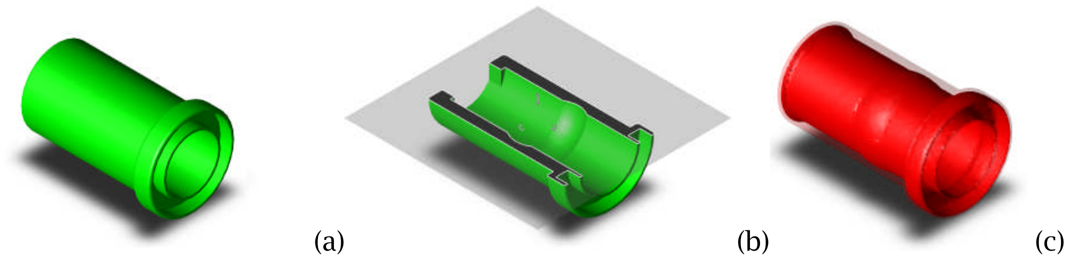


Fig. 1: An example of a 3D medial axis: (a) original model, (b) cut-away view, (c) the medial axis (with the original object boundary transparent), which shows information about relationships between faces, edges and vertices not apparent in the original model (a), without further inspection (b).

Another way to look at the medial axis is the grassfire analogy, which is a more dynamic interpretation of the medial axis. In this analogy, each point on the boundary is considered to be a point of fire, all points burning with the same intensity. The fire spreads perpendicular from the boundary point, at which it starts, towards the inside of the object, and burns with a constant rate of one unit distance per unit time. At time *t* the outer extent of the burned area is the curve parallel to the boundary offset by distance *t*. The medial axis consists of the closure of the quench points of the fires, i.e. the points where the fires meet and douse one another.

There is a one-to-one correspondence between the geometric model and its medial axis transform. To each geometric model belongs a unique medial axis transform, and vice versa. The geometric model can be determined from a medial axis transform by taking the union of all points on the medial axis and the associated maximal spheres. This reconstructability relies on the fact that, although the medial axes of two geometric models can be the same, the medial axis transforms, which include the radius functions, are different if the two models are different.

Many algorithms that compute the medial axis transform from a geometric model make use of some basic concepts related to the medial axis transform. These basic concepts involve a classification of the points on the medial axis, which is determined by so-called foot points. A foot point is a point of contact with the object boundary of the maximal circle, or in 3D the maximal sphere, of a point on the medial axis. In the footpoint, the circle is tangent to the object boundary. Depending on the shape of the object boundary, there can be either a discrete point contact or an area contact. In the latter case, the maximal circle coincides over an area with the boundary, i.e. the radii of curvature of the boundary and the circle are equal. Since the maximal circle is tangent to the object boundary, the lines from the medial axis point to its footpoints are perpendicular to the object boundary. Together with the notion of a foot point, comes the notion of a governor, which is the face, edge or vertex in which the foot point lies, as illustrated in Fig. 2.

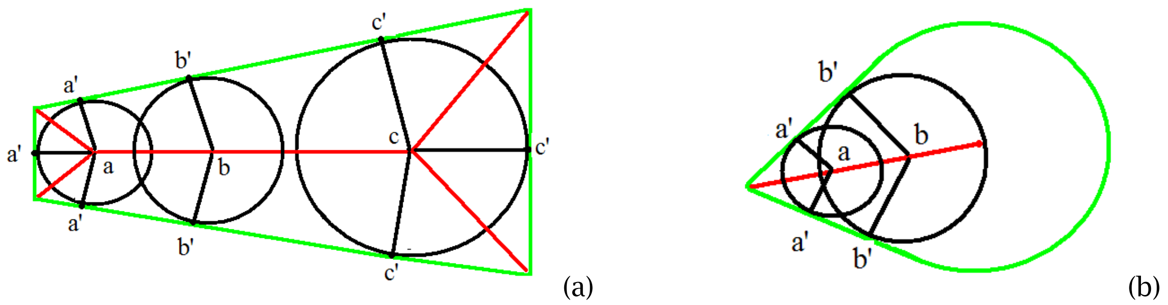


Fig. 2: Examples of 2D medial axis transforms; the red lines indicating the medial axis, the green lines the boundary of the object, and the black circles maximal circles, with their centers at *a*, *b* and *c* and their footpoints at *a'*, *b'* and *c'*; (a) having four edges and four vertices as governors, and (b) having one curved edge and one vertex as governors.

Based on the number of governors of a point on a 3D medial axis, the medial axis can be subdivided into several types of points, see [7]:

- a seam point: a point that has three or more governors (a seam is a connected curve of seam points)
- a seam-end-point: a point where a seam runs into the boundary
- a junction point: a point where three or more seams intersect
- a sheet point: a point with exactly two governors (a sheet is a connected surface of sheet points).

The types of points are illustrated in Fig. 3, where (a) is the original shape and (b) the corresponding medial axis.

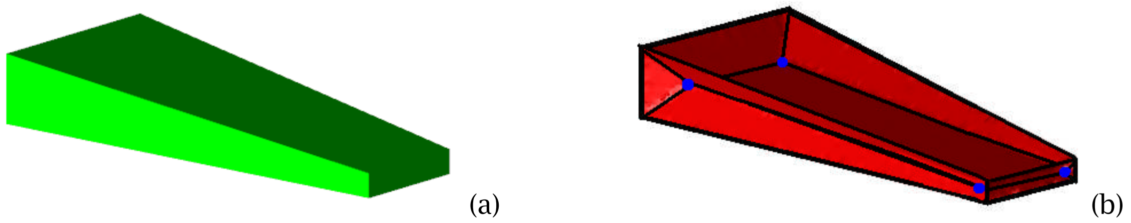


Fig. 3: Classification of points on a medial axis: (a) the original shape and (b) the medial axis; black lines indicate seams and edges, blue dots junction points, and red areas sheets. This medial axis contains 13 sheets, 4 junction points, 8 seam-end points and 12 seams.

One of the drawbacks of the medial axis transform is its sensitivity to small changes in the boundary, as illustrated in Fig. 4, where it can be seen that a small change in the object boundary can lead to a significant change in the medial axis. Some algorithms try to alleviate this problem by classifying offshoots of the medial axis as less significant, based on a substance measure, see [5]. These extraneous axes are then removed and only the major portions of the medial axis are maintained.

Some shapes exhibit special cases. The most common ones are a junction point with more than four governors and a seam with more than three governors. These special cases are incompatible with the generic case algorithms and thus separate, special treatment is needed for them.

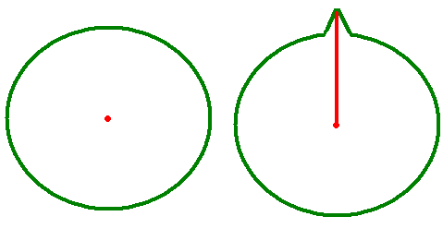


Fig. 4: The problem of sensitivity of the medial axis transform, illustrating that a small change in the object's boundary can lead to a significant change in the medial axis transform.

The computation of the medial axis transform is hard in general, because of numerical instabilities and because the medial axis transform is sensitive to small changes in the boundary. Therefore there are many algorithms that compute the medial axis transform for various types of shapes. These methods can be subdivided into continuous and discrete methods.

Continuous methods rely on the fact that the algebraic form is explicitly known for each sheet. The continuous method [7] calculates the medial axis transform for 3D polyhedra based on the relationships that exists between the various types of points. For example, if G1 is a governor of one sheet and G2 of another, then the seam connecting the sheets has as governors G1 and G2.

Discrete methods usually employ a surface sampling approach [4] or some spatial subdivision. Surface sampling algorithms represent the initial object as a dense cloud of sample points, presumed to be on the boundary. The medial axis is then approximated with a subset of the Voronoi diagram of the point cloud. One of the main issues when applying these discrete methods is the generation of an appropriate set of point samples on the boundary, to ensure a close approximation of the medial axis transform. Under-sampling may result in an approximation of the medial axis that is not accurate enough and thus in jagged results. One method that achieves a good approximation of the medial axis, using surface sampling, keeps a link to the original CAD model [3]. This allows for a better approximation using certain heuristics.

3 THE MEDIAL AXIS AND DIMENSIONS

In this section it will be shown how the medial axis can be used to extract dimensional information, in particular thicknesses and angles, from a geometric model.

The dimension thickness is, in general, not properly defined. For instance, when people are asked what the thickness of a 5 by 5 by 5 cube is, most would answer 5. However, for a cube-like model, which does not have all angles at 90°, there is ambiguity, since the thickness can now be measured in multiple directions.

The medial axis is a dimensionally-reduced structure that captures, in a consistent way, a notion of thicknesses and also of angles. As mentioned in Section 2, there is a one-to-one correspondence between the medial axis transform and the original model, which reduces to a relation between each medial axis point and its corresponding foot points and governors. We make use of this by extracting a dimensional value for each medial axis point, by looking at its foot points and governors, and assigning this value to these foot points.

To determine both types of dimensions, thickness and angle, a further classification of points on the medial axis is needed. Sheet points of which the governors are not directly connected give a good indication of thickness. Since angles are defined between edges or faces sharing a vertex or edge, sheet points of which the governors are directly connected, i.e. share a vertex or an edge, correspond to angle dimensions.

This distinction between thickness and angle sheet points subdivides the medial axis into areas capturing thicknesses and areas capturing angles, as illustrated for a 2D medial axis in Fig. 5(a), where the area *a-b* captures thicknesses and all other areas capture angles. The subdivision is based on the junction points of the medial axis. The foot points of these junction points (*a'* and *b'*), in their turn, subdivide the governors, and hence the boundary, into areas capturing thicknesses (blue areas) and areas capturing angles (grey areas). These areas thus correspond to specific areas of the medial axis. The sheets on a 3D medial axis are bounded by junction points, seam-end points and seams, as can be seen in Fig. 5(b). This figure illustrates the two types of sheets capturing thicknesses (blue) and angles (grey), respectively. In a similar way as in 2D, areas on the governors capturing thicknesses and angles are determined.

For the case in Fig. 5(c), there are no parts of the medial axis that can be directly classified as being a thickness or an angle area, since there is no junction point that divides the medial axis. Considering the area at *c*, which clearly is an angle, the question arises where the angle area should stop and the thickness area should begin. In this case, the medial axis at point *a*, and hence the foot points *a*' on the boundary, give a good subdivision, because the angle of the normals at the foot points *a'* changes compared to the angle at *c*. If the object boundary from *c* to the footpoints *a'* would consists of curves, the angle of the normals would change immediately and only point *c* would remain as the angle area. If this is not desirable, a value can be specified that defines how far along the curved edges the angle area is extended.

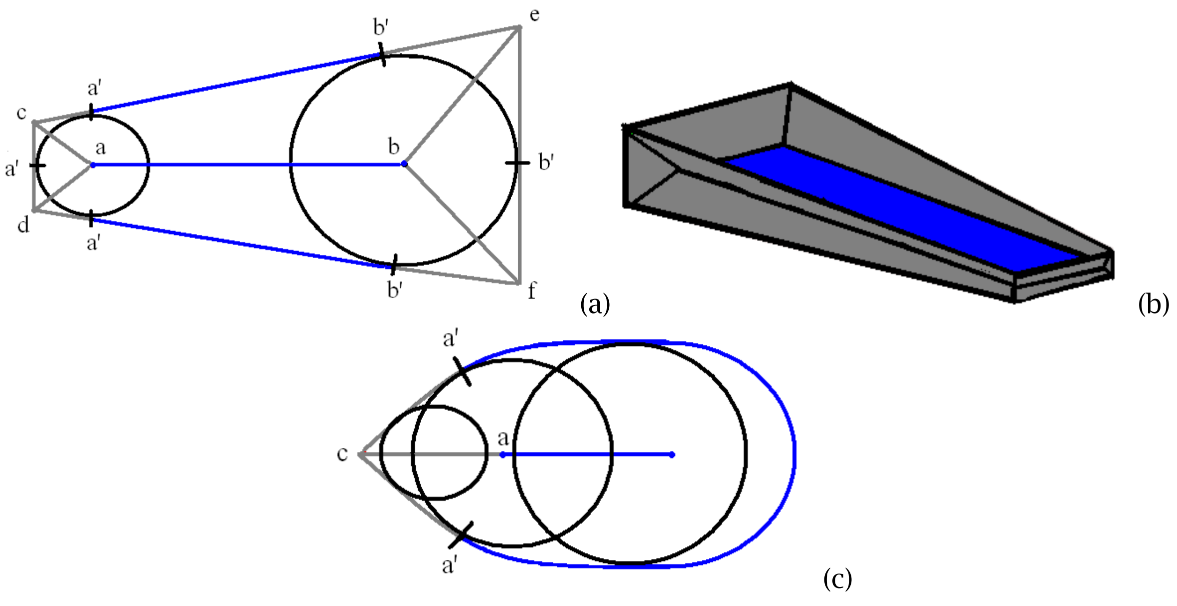


Fig. 5: The division between thickness and angle areas, (a) the 2D example of the medial axis, *a'* and *b'* indicating foot points of the junction points, dividing both the medial axis and the boundary into areas capturing thicknesses (blue) and areas capturing angles (grey), (b) an example in 3D of the two types of sheets on the medial axis, blue again capturing thicknesses and grey angles, and (c) an example of a medial axis without junction points, but still with a proper identification of areas capturing thicknesses (blue) and angles (grey).

The calculation of the actual thickness value if two foot points *m'* correspond to a point *m* on the medial axis, is illustrated in Fig. 6(a). We take the distance *d* between the two foot points. In the case of Fig. 5(c), there is ambiguity when calculating the thickness at the rightmost limit point of the medial axis, because the maximal circle and the object boundary coincide. The thickness can here be calculated between different combinations of two footpoints, which can result in different thickness values. In such cases, the radius value is taken as the thickness value, since this gives a good indication of the thickness. In general, this occurs at rounded regions of an object.

The calculation of the value for an angle *a* corresponding to a point on the medial axis, is illustrated in Fig. 6(b). We take the angle *b,* at medial axis point *m,* between the surface normals at the corresponding foot points *m'* and calculate the value at the actual corner as $a = 180^{\circ} - b$.

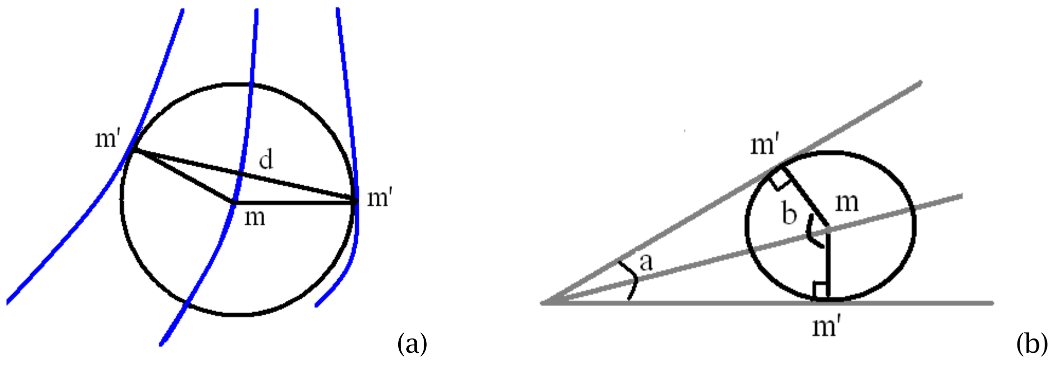


Fig. 6: The actual calculation of the thickness (a) and angle (b) values; for the thickness we take the distance *d* between the foot points *m'*, and for the angle we take 180°– *b*.

As a final step, the thickness and angle values are assigned to the footpoints, and the values are mapped to different color maps, to illustrate the difference between the two areas. Since the interval in which the thickness values can lie is not bounded, the values are taken relative to the maximum thickness within the object, when mapped to the thickness color map. The thickness values are interpolated across the surface, because the thickness changes from point to point on the surface. The angle values, having a limited range, are mapped to fixed color values, and an angle area gets a single color, since the angle value does not change within the area. How these colors are then used to create a complete visualization is discussed in Sections 4 and 5.

A previous approach for extracting and visualizing thickness information [6], visualizes the thickness according to the medial axis transform by taking the value of the radius function at each point of the medial axis, and using this value as an indication for thickness at the corresponding foot points. This approach therefore always indicates a thickness of value zero in the corners of the model, which is not useful at all. We, on the other hand, display the size of the angle at a corner and also have a better measure for thickness (the distance between foot points, instead of the radius of the maximal sphere).

4 COMPUTATION OF THE MEDIAL AXIS AND VISUALIZATION OF DIMENSIONS

This section will explain how the medial axis is computed, and how this representation is used for the visualization of the thickness and angle dimensions.

As mentioned in Section 2, the computation of the medial axis is hard in general. Because of that, and because the extraction of dimensions needs to work for any kind of geometric model, we decided to use a mesh as input. This is not a severe requirement, because most environments have meshing capabilities, either from a CAD model or from a point cloud. The algorithm expects a fairly uniform and dense (triangular) mesh, not only to get a good approximation of the medial axis, but also to get a good visualization of the dimensions. This means that in some situations remeshing may be necessary.

Governors, as mentioned in Section 3, are faces, edges or vertices in which a foot point of the medial axis lies. These governors are properly defined for a CAD model, since it is the entire face, edge or vertex in which a foot point lies. For a mesh, however, the governors are not properly defined, since faces and edges are subdivided into mesh elements. If there is a link from the mesh elements to an original CAD model, the proper governors can still be constructed, but if only a mesh is given, a different approach has to be used. Our approach depends on a so-called significant angle (usually between 0 and 15°) between the normals of two adjacent mesh elements, which specifies whether neighboring mesh elements are considered to be smoothly connected and thus part of the same governor. This subdivides the boundary into sets of mesh elements that we consider governors of the medial axis. During the construction of the governors, a link is created between each mesh element and the corresponding governor(s). This is needed for the correct classification of the medial axis points and their footpoints.

For the computation of the medial axis, all vertices of the mesh are considered to be footpoints. The algorithm traverses all footpoints, and finds the corresponding medial axis point for each footpoint by searching another foot point of that medial axis point. It is explained here for the 2D case; the algorithm for the 3D case is similar, but instead of circles, spheres have to be considered. The criterion for identifying the corresponding medial axis point is illustrated in Fig. 7(a). For a foot point f_a, the corresponding medial axis point m is always along the normal to the object boundary at f_a. Any other footpoint f_b of m has an equal distance to m, because of the nature of the medial axis. Using this criterion, the triangle $f_a m f_b$ can be constructed, in which the edges $f_a m$ and $f_b m$ have equal length and the angles a and b are equal. The maximal circle of m has the length of $f_a m$ as its radius, and two of its foot points are f_a and f_b.

It is worth mentioning that the algorithm in [4], mentioned in Section 2, uses a similar criterion based on the properties of the medial axis, but the focus there is on a convergence guarantee for the medial axis approximation from the Voronoi diagram, using a subset of Voronoi vertices called poles. Our approach is directed towards the identification of maximal spheres.

In the algorithm, given a footpoint f_1, a triangle is constructed for every other footpoint f, i.e. every other vertex of the mesh, by taking the same angle a at f as at f_1. If no triangle can be constructed, because $f_1 m$ and fm do not intersect, f is discarded. If a triangle can be constructed, the corresponding circle could be a maximal circle. Fig. 7(b) illustrates that this can result in multiple circles. The circle with the smallest radius value is taken, since all other circles contain at least one mesh point and are therefore not a maximal circle. In this way, the algorithm finds a medial axis point for each footpoint, and so we get a discrete approximation of the medial axis.

If a footpoint f_1 lies at a concave edge or vertex, there is a possibility of identifying multiple footpoints f with different medial axis points m and hence with different radius values. For such footpoints f_1, therefore no corresponding footpoint f is searched, i.e. they are skipped within the iteration over all footpoints. However, they can be a footpoint of another footpoint that is not at a concave edge or vertex, and will be found when for the latter the corresponding footpoint is searched. As will be discussed later, these concave footpoints are visualized in a different way.

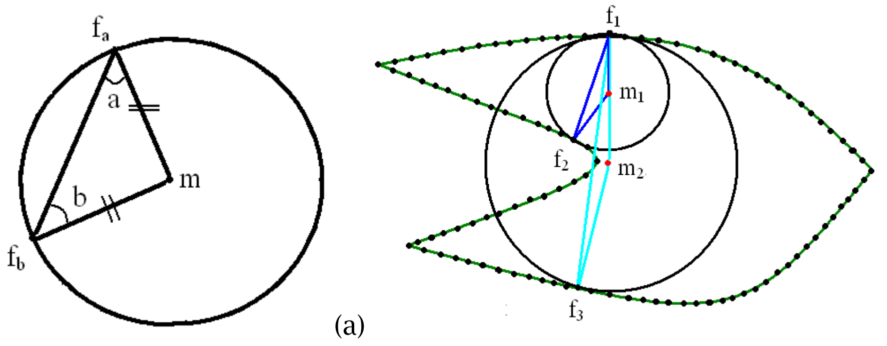


Fig. 7: The identification of a medial axis point, (a) the construction of the medial axis point m, given the surface normal at f_a, and the line $f_a f_b$, (b) an actual situation as it can occur within a mesh; the triangle $f_1 m_1 f_2$ is chosen, and not the triangle $f_1 m_2 f_3$, because the circle corresponding to $f_1 m_1 f_2$ has a smaller radius.

To make this approach plausible, we consider a continuous approach, in which not a discrete model (a mesh) but a continuous model is taken. Such an approach can identify a medial axis point for a footpoint f_a by taking a continuous sequence of circles tangent to the boundary at f_a, which all have their midpoint on the surface normal at f_a, with an increasing radius value. The smallest circle c that touches another part of the boundary, as illustrated in Fig. 8, is the circle of which the center is the corresponding medial axis point m, and the point where the circle touches another part of the boundary is another footpoint f_b. Our approach identifies all possible circles for the discrete model with their center along the surface normal at f_a and touching another part of the boundary. The smallest circle we find corresponds to the circle c of the continuous approach; compare Fig. 8 and Fig. 7(b).

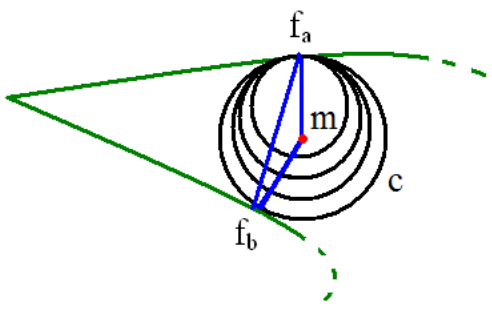


Fig. 8: A possible continuous approach in which the maximal circle *c* for footpoint f_a is the smallest circle tangent to the boundary at f_a that touches another part of the boundary.

For each medial axis point, the algorithm finds the corresponding footpoints and thus the corresponding governors; see begin of this section. All sheet points are classified as a thickness or an angle sheet point, according to the connectedness of the governors as mentioned in Section 3. Depending on the classification of a sheet point, either a thickness or an angle value is calculated, which is then assigned to its footpoints. These values are mapped to their corresponding color maps; for more details see Sections 3 and 5. If a footpoint lies at a concave edge or vertex, as discussed earlier, the value for that point is based on the dimensional values at its neighboring mesh vertices.

The triangular mesh is used to visualize the dimensions, i.e. the color values at the mesh vertices are interpolated across the triangles, resulting in a visualization of the dimensions across the whole boundary. Some mesh elements may have vertices in two different angle areas or in both a thickness and an angle area. The visualization between these and the neighboring mesh elements can give a jagged result, but as long as the density of the mesh is high enough, this is not really distracting.

5 RESULTS

This section will illustrate the general principle, show two examples with more details, and finally discuss some results of a parallel implementation.

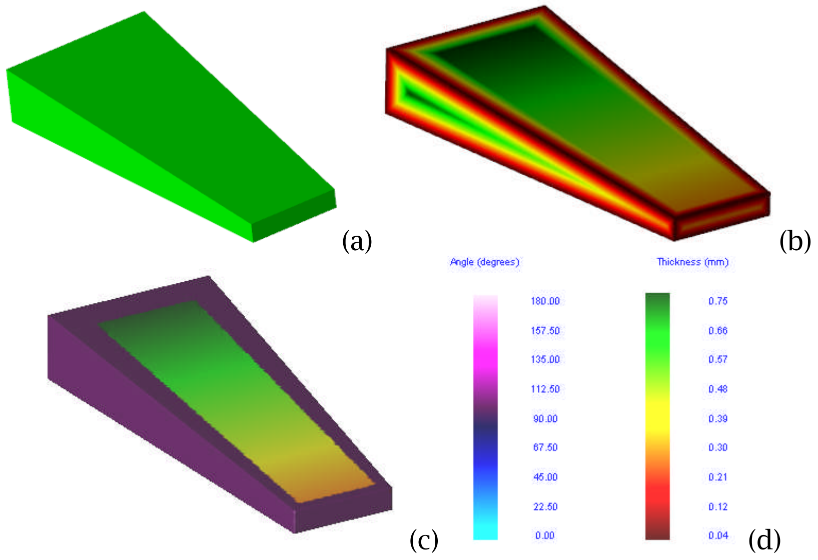


Fig. 9: Model (a), visualization of only thickness according to [6] (b), and visualization of both thickness and angle areas (c) according to different color maps (d).

The general principle of our approach is illustrated in Fig. 9. The model, only thickness visualization according to [6], and our visualization, including both thicknesses and angles, are displayed. The thickness values are mapped to colors ranging from red to green, where red was chosen for the thinner regions and green for the thicker regions. The angular values are mapped to colors ranging from blue to pink, to clearly illustrate the difference with the thickness areas. This simple example illustrates that the distinction between the two types of areas is clear.

A second example, see Fig. 10, illustrates that the differences between two parts can more easily be seen with our visualization. The part in Fig. 10(a) has rounded regions in two locations, whereas the part in Fig. 10(c) has straight edges. Our visualizations of the two parts, given in Fig. 10(b) and Fig. 10(d), are clearly different.

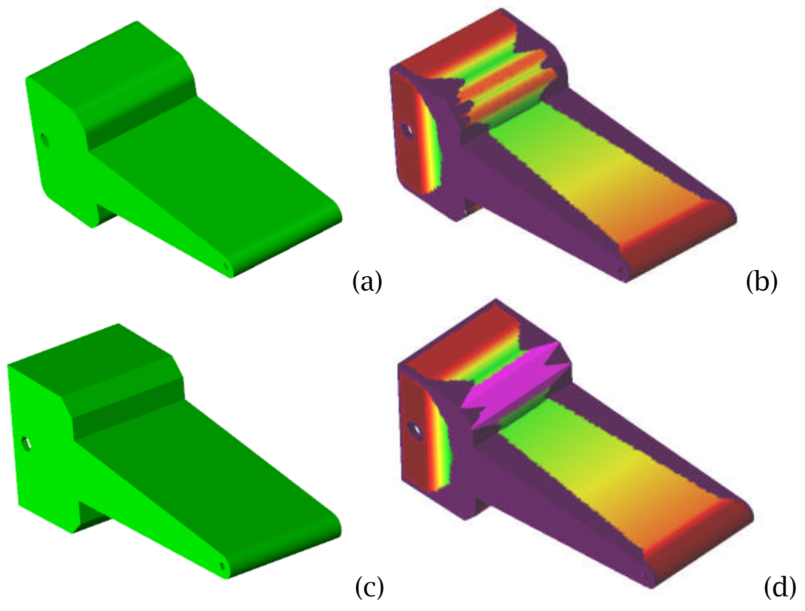


Fig. 10: Two parts: (a) has rounded regions in two locations, whereas (c) has straight edges. Their visualizations show thickness areas (b) and angle areas (d), respectively.

In the final example, see Fig. 11, a part contains four similar instances of the same feature (extruded cylinders). However, one of them is smaller than the others, which cannot be seen in a standard display of the part, see Fig. 11(a). The visualization in Fig. 11(b) shows a somewhat different color for one of the features, more clearly visible in the two close-ups. The features only differ 0.5 mm, hence the difference in visualization is also small, but large enough to be noticed.

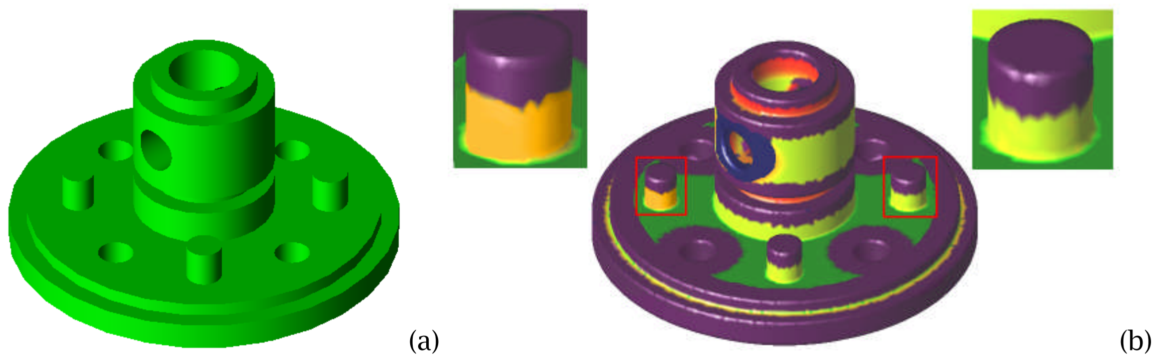


Fig. 11: A part that has several instances of the same feature (a), the extruded small cylinders that appear similar. However, one of them deviates by only 0.5 mm, and this is visible since the smaller feature has a somewhat different color (b).

Because the medial axis transform is computation intensive, especially if one wants a high degree of accuracy, the serial implementation was turned into a parallel one, with the help of openMP [10]. As can be seen in Tab. 1, the achieved speedup is very high, because the iteration over the footpoints can easily be divided among various threads. Implementation of a spatial subdivision scheme could further improve overall speed.

Model	*# Vertices*	*# Triangles*	*1 Core*	*2 Core*	*Speedup*	*4 Core*	*Speedup*
Fig. 9	38866	77728	141.5 s	70.6 s	2.0	**36.1 s**	**3.9**
Fig. 10 (a)	42247	84502	168.3 s	84.3 s	1.9	**43.2 s**	**3.8**
Fig. 11	52028	104080	261.3 s	131.6 s	1.9	**66.2 s**	**3.9**

Tab. 1: Results from the parallel implementation using openMP [10], illustrating that the scalability of the solution is very high, since the achieved speedups are close to what can maximally be expected. The results were tested on an Intel Core 2 Quad @ 2.67 GHz, 6GB.

6 CONCLUSIONS

This paper discussed a method that extracts and visualizes dimensions, specifically thicknesses and angles. The method computes the medial axis and determines the dimensions based on the information provided by this representation. The medial axis gives a consistent way of measuring the dimensions in question, and makes a distinction which dimension gets preference over the other to give a useful visualization. Since the method needs to work for any input and format, the calculation and visualization of the dimensions is based on a mesh. By making a parallel implementation of the method, the dimensions can be extracted and visualized fast, making sure that this does not become a bottleneck of the design process.

The visualization gives intuitive feedback, making a clear distinction between thickness and angle dimensions. It is capable of illustrating symmetry, whether or not surfaces are parallel, what is the inclination of a face compared to the rest of the part, what is the relationship between different parts of an object, e.g. does this part of the object have a thickness of around 40 % of the thickness encountered elsewhere in the object, etc. So, it can be used to verify several properties of the geometry of an object.

Future work involves the removal of the jagged results between the thickness and angle areas and handling special cases, mentioned in Section 2. An example of a special case is a cube-like object in which there are no sheet points capturing thickness, but only one seam that contains this information. Since seams are not mapped to the boundary, the visualization only offers information on the angle dimensions and not the thickness dimensions. In such situations, the visualization needs to be adjusted to incorporate some thickness information. In realistic models, however, these special cases rarely occur, and the visualization is already very useful.

ACKNOWLEDGEMENTS

This work was supported by the Canadian Natural Sciences and Engineering Research Council of Canada (Discovery Grant for A. Spence). Academic access to the Spatial Corp. ACIS 3D Mesh generation software, and the Tech Soft 3D HOOPS graphics software, is gratefully acknowledged too.

REFERENCES

[1] Blum, H.: A transformation for extracting new descriptors of shape, ACM Computing Surveys, 23(3), 1991, 345-405.
[2] Bruck, J.; Gao, J.; Jiang, A.: MAP: medial axis based geometric routing in sensor networks, Wireless Networks, 13(6), 2007, 835-853.

[3] Dey, T. K.; Hyuckje, W.; Zhao, W.: Approximate medial axis for CAD models, Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications, June 16-20, Seattle, WA, USA, ACM, New York, NY, USA, 2003, 280-285.

[4] Dey, T. K.; Zhao, W.: Approximate medial axis as a Voronoi subcomplex, Computer-Aided Design, 36(2), 2004, 195–202.

[5] Katz, R. A.; Pizer, S. M.: Untangling the Blum medial axis transform, International Journal of Computer Vision, 55(2/3), 2003, 139-153.

[6] Lambourne, J. G.; Brujic, D.; Djuric, Z.; Ristic, M.: Calculation and visualization of the thickness of 3D CAD models, Proceedings of the International Conference on Shape Modeling and Applications, June 15-17, Cambridge, MA, USA, IEEE Computer Society, Los Alamitos, CA, USA, 2005, 340-344.

[7] Sherbrooke, E. C.; Patrikalakis, N. M.; Brisson, E.: An algorithm for the medial axis transform of 3D polyhedral solids, IEEE Transactions on Visualization and Computer Graphics, 2(1), 1996, 44-61.

[8] Siddiqi, K.; Shokoufandeh, A.; Dickinson, S.; Zucker, S.: Shock graphs and shape matching, International Journal of Computer Vision, 35(1), 1999, 13-32.

[9] Tam, R.; Heidrich, W.: Shape simplification based on the medial axis transform, Proceedings of the 14th IEEE Visualization Conference, October 22-24, Seattle, WA, USA, IEEE Computer Society, Los Alamitos, CA, USA, 2003, 481-488.

[10] OpenMP, www.openmp.org