MATEC Web of Conferences – General and efficient algorithms for handling intricate situations in trimmed surface rendering

Luo Bo¹, Gang Zhao¹, Wei Wang^{1,a} and Xiaoxiao Du¹

¹School of Mechanical Engineering and Automation, Beihang University, Beijing, China

Abstract. Trimmed surfaces have been widely used in complicated product design in mechanical engineering especially in complex free form surface models of CAD/CAM/CAE. Focusing on the deficiencies of the algorithms available in published literatures when handling some intricate situations, this paper proposes a more general and efficient algorithm of trimmed surface rendering (ATSR). The ATSR is a general algorithm that can deal with most intricate situations, which frequently occur in practical engineering that no special attention has been paid to in references at hand, as well as common situations. Furthermore, this paper presents a new algorithm for the nodes of polygon connecting into elements (ANCE). The ANCE can handle most tricky polygons appearing in the process of rendering and achieve successful triangulation for the to-be-rendered surface. Preferable results are obtained in the case studies, which indicate that the algorithms (ATSR and ANCE) are efficient and effective in handling complicated as well as common trimmed surface models in real engineering industry.

1 Introduction

Free form curves and surfaces are widely used in mechanical engineering and Computer Aided Design (CAD). The product models are usually conveyed and stored in neutral file formats like IGES or other file standards for the communications among different CAX systems. The widely used IGES file usually contains large amounts of trimmed surface data and the parsing of such data is a main task in data exchange. Whence the IGES file has been translated into mechanical engineering software, the trimmed surface rendering becomes the prerequisite for further work. In this paper, we put forward a general and efficient algorithm of trimmed surface rendering capable of handling these intricate situations that are frequently encountered in real engineering industry.

2 Related work

Sheng and Hirsch put forward an approach for trimmed surface triangulation^[1]. This approach approximates the trimmed regions of the surface into 2D polygonal regions. Then, the polygonal regions are triangulated by a restricted Delaunay triangulation algorithm,^[2-4] after which, the triangles generated are subdivided according to the specified tolerance of the surface. In addition, this approach requires a half-edge data structure to represent the mapping polygons. Therefore, a large amount of preprocessing work and extra data structures in terms of the polygons to be triangulated are required.

Rochwood and Heaton came up with a method of real-time trimmed surface rendering.^[5] This method

changes all trimmed surfaces into Bézier patches and subdivides the trimming region of the patches into uv -monotone regions. Besides, the trimmed boundary curves are divided into uv -monotone sections as well. Then, a coving and tiling algorithm is used to triangulate the trimmed region. This method needs a lot of redundant preprocessing steps and there are some intricate situations which this method does not take into consideration. For example, the situation where multiple loops of the boundary curves contained in a single rectangular patch and the situation where the intersections of different boundary curves contained in a single rectangular patch are not considered. These situations often occur in the self-adaptive subdivision of trimmed surfaces and may lead to wrong rendering results.

Cripps and Parwana proposed an efficient and robust parametric trimmed surface triangulation method.^[6] This method subdivides the parameter domain into uniform rectangular elements and subdivides the trimmed boundary curves into piecewise linear segments. However, there are some special cases which this method does not take into consideration, for instance, the situation where the curvature of trimmed curves contained in a single rectangular element has the value of both plus sign and minus sign and the situation where the number of intersection points between boundary curve and any one edge of rectangular element is more than one. These situations often occur in the self-adaptive subdivision of trimmed surfaces^[7] and may give rise to wrong rendering results.

Aiming at the deficiencies of the above algorithms in handling intricate situations of trimmed surface models ,

^aCorresponding author: jrrt@buaa.edu.cn

which may give rise to wrong rendering results, a more general and efficient algorithm of trimmed surface rendering (ATSR) is proposed in our work. The ATSR is a general algorithm that can deal with arbitrarily intricate situations that may occur in real mechanical industry to help achieve the high-quality rendering of complex engineering trimmed surface models. Moreover, this paper presents a new algorithm for the nodes of the polygon connecting into element (ANCE). The ANCE can handle complex polygons and improve the function of triangulation and rendering. Preferable result is obtained through case studies, which indicates that the algorithms (ATSR and ANCE) are efficient and effective in rendering complicated trimmed surface models in CAD and mechanical industry.

3 ATSR

3.1 General Description of ATSR

The main contribution of this algorithm i.e. ATSR is a general procedure that can handle most intricate situations of trimmed surface models, as well as common situations, and by combination with a new algorithm ANCE to realize the functions of trimmed surface triangulation and rendering. The main processes of the ATSR can be listed in sequence as: the self-adaptive subdivision of trimmed curves, the inverse computation of the 3D piecewise linear segments of the trimmed curves, self-adaptively subdividing the 3D trimmed surfaces into rectangular elements in 2D parameter space, traversing every single rectangular element and sorting the nodes of the polygon which needs to be triangulated, quadtree subdivision of rectangular element to handle intricate situations, the ANCE algorithm triangulating the polygons and rendering.

3.2 Self-adaptive subdivision of trimmed curves and inverse computations

The first step of trimmed surface rendering is self-adaptive subdivision of trimmed curves^[8]. The algorithm in reference^[9] which self-adaptively subdivides trimmed curves into piecewise linear segments to approximate the original curves according to their curvature is utilized in this paper. If the 2D parametric curve's information of 3D trimmed curve is not contained in the models(which often happens in standard IGES files), inverse computation is unavoidable, which needs to self-adaptively subdivide the 3D trimmed curves into piecewise linear segments and convert the nodes of piecewise linear segments into nodes of 2D parametric linear segments.^[10] In this way, the 2D parameter segment can be obtained for triangulation and rendering in the subsequent steps.

3.3 Self-adaptive subdivision of trimmed surface

Uniform self-adaptive subdivision algorithm is used to divide the parameter domain of trimmed surface into rectangular elements^[9]. The rectangular element should be subdivided until every edge of the rectangular element is less than a specific minimum size for purpose of guaranteeing the approximation quality of the 3D trimmed surfaces(Figure 1). Then, each rectangular element will be processed individually.

3.4 Situation classification and subdivision

In the process of dealing with each rectangular element, different situations of trimmed surface models will be encountered. This paper divides all the situations into intricate situations and common situations. The criterion is the ANCE algorithm. If the situation can be handled through the ANCE algorithm (discussed in the next section), it is classified as common situations. The other situations are classified into intricate situations (Figure 2), which are hard for direct triangulation and may give rise to the wrong rendering result.

Figure 1. Subdivision of parameter domain into rectangular elements.

Figure 2. Intricate situations(slash for untrimmed region, blank for trimmed region): (a)The number of intersection points between segments and any one edge of rectangular element is more than one. (b)The rectangular element has multiple loops. (c)The rectangular element has intersection of segments. (d)The curvature of trimmed curves contained in a single rectangular element has the value of both plus sign and minus sign. (e)Two diagonal vertices are located within the trimmed region and the other two diagonal vertices are located outside trimmed region.

As for intricate situations, they need to be subdivided into common situations that can be handled smoothly through the ANCE algorithm, using quadtree subdivision algorithm. This paper summarizes all the intricate situations into these five templates(Figure 2). These five situations are the primary types of intricate situations needing special treatment and almost all the other intricate situations are basically the same as them or derived from them. Therefore we just need to handle these five situations.

In our algorithm ATSR, quadtree subdivision is used to subdivide the rectangular element into 4 sub-rectangular elements. Then, each sub-rectangular

element can be turned into the common type and processed individually. On some occasions, quadtree subdivision once may not be enough and several times of subdivision are needed but finally they all can be turned into common situations that we can process smoothly through ANCE.

3.5 The algorithm of sorting the nodes according to different situations

From the above steps, all the situations are changed into common situations that can be processed using ANCE. In the common situations, we obtain the vertices of the rectangular element, piecewise linear segments, intersections of the element and the piecewise linear segments. Then, they are labelled to convert the trimmed region into to-be-triangulated polygons. The purpose of this section is to reorder the nodes of the to-be-triangulated polygons and ensure they are in the direction of anti-clockwise, which is a prerequisite for the following ANCE algorithm to triangulate the polygon for rendering.

Judge whether each vertex (defined as A, B, C, D in Figure 1) of the rectangular element is located both within the outer boundary segment and outside the inner boundary segment (if any). Then, define the number of such vertex as *vertexnum*. This paper defines the nodes of the polygon which needs to be triangulated as *temppoints*, including the vertices of the rectangular element, the nodes of the piecewise linear segments and the intersections between the segments and the edge of the rectangle. In addition, the nodes of the piecewise linear segments and the intersections between the segments and the edge of the rectangle (*temppoints* except the vertices of the rectangular element) are defined as *the nodes contained in the element*. For convenience, the nodes in *temppoints* are sorted as anti-clockwise in terms of the value of *vertexnum*. Besides, we require that in *temppoints* the vertices of the rectangular element are from the first point to the (*vertexnum*)th point and that *the nodes contained in the element* are from the (*vertexnum*+1)th point to the last point in *temppoints*, which will be necessary for the following ANCE algorithm.

If the value of *vertexnum* is 4, which means the rectangular element is located completely within the outer boundary segments and outside the inner boundary segments (if any), namely in the untrimmed region, the 4 points in 3D space corresponding to the 4 vertices of the rectangle in 2D parametric space form a quadrilateral element for rendering.

If the value of *vertexnum* is 0, which means the rectangular element is located completely outside the outer boundary segments or inside the inner boundary segments (if any), namely outside the untrimmed region, this element should not be displayed for rendering.

If the value of *vertexnum* is 3 or 2 or 1, then, the order of the points in *temppoints* is sorted as the vertex of the rectangle, the nodes contained in the element, which are in anti-clockwise direction.

Through the above process, this paper ensures that the vertices of the rectangular element are from the first point to the (*vertexnum*)th point in *temppoints* and that *the nodes contained in the element* are from the (*vertexnum*+1)th point to the last point in *temppoints*. In addition, all the points in *temppoints* are in the direction of anti-clockwise. Then, we can use the following ANCE algorithm to triangulate the points in *temppoints* for rendering.

3.6 ANCE

This paper proposes a new algorithm, i.e. ANCE, for triangulation rendering of the points stored in *temppoints*. The ANCE can deal with complex polygons and improve the function of triangulation and rendering. The overall process of the algorithm is shown in Figure 3.

Figure 3. The overall process of the ANCE.

On some occasions there is a regulation in the step of finding the point in the vertex of the rectangle as the second division point *N* of *temppoints*: there must be no intersection between the line segment *MN* and the line segment of the *nodes contained in the element* (Figure 4). If all the vertices of the rectangle do not satisfy this regulation, we can adaptively add the point in the edge of the rectangular element as the second division point *N* of *temppoints*. As a result, the line segment *MN* divides the *temppoints* into two parts. Then, each part of the *temppoints* forms a new *temppoints* recursively. Meanwhile, in the new *temppoints*, the order of the points and the value of *vertexnum* is reset to ensure that the vertices of the rectangular element are from the first point to the (*vertexnum*)th point in the new *temppoints* and that *the nodes contained in the element* are from the (*vertexnum*+1)th point to the last point in the new *temppoints*. In addition, all the points in the new *temppoints* are in the direction of anti-clockwise. Use this ANCE algorithm for the new *temppoints* until all points in the original *temppoints* form triangular/quadrilateral elements for rendering.

Figure 4. (a)The wrong N point. (b)The right N point. (c)The final elements result of ANCE..

4 Implementation and results

This section illustrates 8 case studies to demonstrate performances of the ATSR and the ANCE algorithm. These case studies are all IGES models from different mechanical industrial sectors. The results of triangulation rendering of each case and the analysis are presented.

The hardware conditions contain an Intel Core i5 CPU which has a standard Pentium 2.80GHz processor and 4GB memory. The system of trimmed surface rendering using the ATSR and the ANCE has been integrated to the CAD/CAE software ARCHYTAS, which is developed by our laboratory. The trimmed surface models shown here are some real product's surface models from sheet metal workshops of either aircrafts or automobiles. In all these models, different kinds of intricate situations are encountered but they are all basically the same as the five template situations in Figure 2 or derived from them. The rendering result of the cases using the ATSR and the ANCE are shown in Figure 5. The rendering results of ATSR in the five template intricate situations encountered in these cases are presented in Figure 6. Figure 7 presents the comparison between the right rendering results using ATSR and the wrong rendering results of the algorithms without considering intricate situations. The analysis and the comparison of the results and algorithms are shown in Table 1.

Figure 5. (a)Model one: stamping shell. (b)Model two: automobile door. (c)Model three: box. (d)Model four: automobile pillar. (e)Model five: automobile engine protection cover. (f)Model six: automobile roof panel. (g)Model seven: aircraft. (h)Model eight: automobile inner panel.

In the Table1, the statistics in the second column is the basic geometry information contained in the original IGES file. The statistics in the third column and the fourth column is the rendering information using ATSR. Rendering time comparison is presented in the last column.

Figure 6. The rendering results of ATSR in intricate situations: (a)Intricate situations included in model five: The number of intersection points between segments and any one edge of rectangular element is more than one; The curvature of trimmed curves contained in a element has the value of both plus sign and minus sign. (b)Intricate situation included in model four: The rectangular element has multiple loops. (c)Intricate situation included in model four: The rectangular element has intersection of segments. (d)Intricate situation in Model eight: Two diagonal vertices are located within the trimmed region and the other two diagonal vertices are located outside the trimmed region.

Table 1.Analysis and comparison of the rendering results

Cases	Trimmed Surfaces Number	Rendering Triangles Number	Rendering Time of this Paper	Rendering Time of CATIA
Model one	20	2961	0.57s	0.55s
Model two	14	2496	0.46s	0.49s
Model three	17	2424	0.33s	0.31s
Model four	186	8199	2.96s	3.12s
Model five	278	27986	5.82s	5.04s
Model six	796	45776	7.66s	6.98s
Model seven	363	34422	7.06s	6.63s
Model eight	634	69750	9.93s	9.08s

From the above Table1 and Figure 5-7, the ATSR and the ANCE show that they are dependable algorithms in dealing with complex trimmed surface models. Besides, the rendering results in Figure 5-7 indicate that the ATSR is a general algorithm that can handle not only most intricate situations but also common situations, as these trimmed surface models contain all the intricate (Figure 6) and common situations. What's more, the efficiency of the ATSR in rendering is satisfying since the rendering time column in the table gives approving result. This rendering time of CATIA is a rough

estimation since in commercial software a timer for a special process can't be realized if no such API is exposed, but this result approximately reflects that ATSR with ANCE is about on the same level with commercial software as to the time consumption. Certainly, the rendering function in commercial CAD software of trimmed surfaces can achieve high efficiency and precision. But their detailed algorithms are highly confidential and beyond access, which is a practical problem for software development. Thus, in our paper, ATSR is proposed to predigest the arduous efforts in rendering complex trimmed models, as the rendering results in Figure 5-7 and time in Table1 approve the case.

Figure 7. (a)(b)The right rendering result using ATSR.
(c)(d)The wrong rendering result of algorithms without considering intricate situations.

In addition, in the existing references, there are usually no special concerns taken for the listed intricate situations, which will give rise to the wrong rendering results. And the rendering result comparisons between ATSR and the algorithms without considering intricate situations are presented in Fig 7. These figures illustrate that there may be wrong rendering area in the locals of trimmed region if the intricate situations are not taken into consideration. So the aim of ATSR is to provide a general and efficient algorithm which can handle these intricate situations frequently encountered in the real engineering practice.

5 Conclusion

This paper proposes a general and efficient algorithm of trimmed surface rendering(ATSR), aiming at eliminating possible deficiencies of the published algorithms in handling some special intricate situations of trimmed surface models, which are not scarce in practices. Furthermore, this paper presents a new algorithm of the nodes of the polygon connecting into elements (ANCE). The ANCE can handle tricky polygons and achieve the feats of accurate triangulation and rendering. The preferable results are obtained in the case studies, which indicate that the algorithms (ATSR and ANCE) are efficient in manipulating complicated trimmed surface models. Usually there are large amount of complex trimmed surface models in mechanical industrial and engineering field, so the above algorithms have a promising application prospect. Nevertheless, in face of small and simple trimmed surface models, there are some redundant judgements and excessive performing in the above algorithms, such as judging whether the above intricate situations really exist and ANCE recursively rendering algorithm. Thus on such occasion, the above algorithms may not be the best choice, and there may still be some probabilities for improving them in the future.

Acknowledgements

This paper is financially sponsored by the Natural Science and Technology Major Project(2014ZX04001-081-07).

References

1. X Sheng, B E Hirsch. Triangulation of trimmed surfaces in parametric space[J]. CAD, (1992), **24**(8): 437-444.
2. D T Lee, A K Lin. Generalized Delaunay triangulation for planar graphs[J]. DCG, (1986), **1**(3): 201-217.
3. H Sadoyan, A Zakarian, V Avagyan, et al. Robust uniform triangulation algorithm for computer aided design[J]. CAD, (2006), **38**(10): 1134-1144.
4. P Schneider, D H Eberly. *Geometric tools for computer graphics*[M]. Morgan Kaufmann, (2002).
5. A Rockwood, K Heaton, T Davis. Real-time rendering of trimmed surfaces[C]//ACM SIGGRAPH Computer Graphics. ACM, (1989), **23**(3): 107-116.
6. R J Cripps, S S Parwana. A robust efficient tracing scheme for triangulating trimmed parametric surfaces[J]. CAD, (2011), **43**(1): 12-20.
7. R Seidel. A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons[J]. CG, (1991), **1**(1): 51-64.
8. M Ristic, D Brujic, S Handayani. Triangulation of unordered data using trimmed NURBS computer aided design models[J]. JEM, (2000), **214**(6): 509-514
9. S Kumar, D Manocha. Efficient rendering of trimmed NURBS surfaces[J]. CAD, (1995), **27**(7): 509-521.
10. L Piegl, W Tiller. *The NURBS book*[M]. Springer Science & Business Media, (2012).