Universität Konstanz

Why CAD Data Repair Requires Discrete Techniques

Karsten Weihe
Thomas Willhalm

Konstanzer Schriften in Mathematik und Informatik
Nr. 61, Mai 1998

ISSN 1430–3558

Why CAD Data Repair Requires Discrete Algorithmic Techniques

Karsten Weihe[1] Thomas Willhalm[1]

Lehrstuhl für praktische Informatik I
(Algorithmen und Datenstrukturen)

61/1998
Universität Konstanz
Fakultät für Mathematik und Informatik

Abstract

We consider a problem of reconstructing a discrete structure from unstructured numerical data. It arises in the computer-aided design of machines, motor vehicles, and other technical devices. A CAD model consists of a set of surface pieces in the three-dimensional space (the so-called *mesh elements*). The neighbourhoods of these mesh elements, the *topology* of the model, must be reconstructed. The reconstruction is non-trivial because of erroneous gaps between neighboured mesh elements.

However, a look at the real-world data from various applications strongly suggests that the pairs of neighboured mesh elements may be (nearly) correctly identified by some distance measure and some threshold. In fact, to our knowledge, this is the main strategy pursued in practice. In this paper, we make a first attempt to design systematic studies to support a negative claim: We demonstrate empirically that human intuition is misleading here, and that this approach fails miserably even for "innocent-looking" real-world data. In other words: It does not suffice to look at individual pairs of surface pieces separately; incorporating discrete relations between mesh elements is necessary.

1 Introduction

Computer-aided design plays an important role in today's engineering. In this paper, we deal with CAD data models like in Figure 1. Such a model consists of mesh elements and approximates the surface of the workpiece. In general, the mesh elements are not plane, and their edges are not straight lines. For example, in the data available to us, *trimmed parametric surface patches* were used (see [8]).

One of the tasks which are to be done automatically is reconstructing the so-called *topology* of the CAD data model, i.e. the information whether and where two mesh elements are to be regarded as immediately neighboured (Figure 5). Many wide-spread data formats for CAD models do not provide the neighbourhoods.

The topology of a CAD model is important, since almost every further step of the CAD process relies on this information. Section 3 gives examples of such steps.

Pictures such as Figure 1 or Figure 9 suggest that the edges of neighboured mesh elements fall together geometrically. This is generally not the case. There are normally gaps between the mesh elements – gaps that can be as large as in Figures 2 and 3. These gaps are sometimes even larger

[1]Universität Konstanz, Fakultät für Mathematik und Informatik, Postfach 5560/D188, D-78434 Konstanz, {karsten.weihe,thomas.willhalm}@uni-konstanz.de, http://www.informatik.uni-konstanz.de/~{weihe,willhalm}.

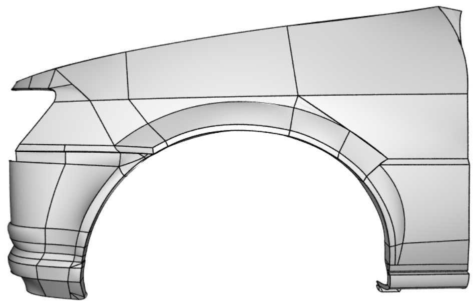

Figure 1: This mudguard is a typical example of our real world instances. It consists of surfaces patches (the so-called mesh elements). The black curves indicate the edges of the mesh elements.

than mesh elements in the same workpiece. They are the reason why the problem is non-trivial, since we have no knowledge whether an edge is neighboured to one, several or no other edges (Figure 6). Consider two edges that are situated next to each other. The gap between them may be intended by the designer of the workpiece to separate them. But the two surface patches might as well be regarded as parts of one closed surface. The automatic tools have to guess the intention of the engineer who designed the workpiece. Hence, the problem is inherently subjective and therefore not mathematical in nature. In particular, the quality of the output is the point of interest; in fact, time and space consume is negligible compared to other steps of the CAD process.

In this paper, we focus on a common approach, which is highly suggested by Figures 1 and 9 and all other pictures that we produced: to define a distance measure of two edges and choose a threshold value. Examples of distance measures are the maximal distance of two points on the respective edges or the sum of the distances of the end points. Pairs of edges whose distance is smaller than the threshold value are considered as neighboured and pairs with a larger distance are not. This may be viewed as estimating the *absolute error* of the placement of the edges.

Of course, this approach can be refined by replacing the absolute with the *relative error*. Properties such as the lengths of the edges, or the area or perimeter of the mesh elements seem reasonable scalings for such a scaled distance measure. To our knowledge, these approaches with scaled or unscaled distance measures are the main strategy in today's industrial practice [7, 8]. Even Figures 2 and 3 suggest that this approach finds a very good approximation of the right neighbourhoods. So the problem does not seem to deserve further research.

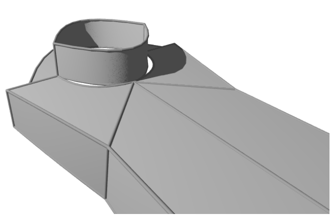

Figure 2: an example of a sloppy design.

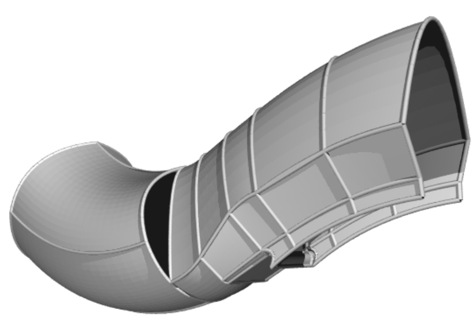

Figure 3: an extreme gap from the interior of Figure 9. The black, semicircular hole does not make any sense from an engineering point of view and might be the result of a format conversion.

The main contribution of this paper is to demonstrate empirically that in general this fails miserably against all expectations – even if we assume that the algorithm automatically finds the best threshold value for each edge (of course, this assumption is by far too optimistic). We have examined a variety of unscaled distance measures and possible extensions to scaled distance measures. Of course, our investigations only cover a subset of all possible distance measures. But we believe that the most natural ones were treated and that they are good representatives for all useful distance measures, because our results are not too sensitive to the choice of the distance measure. This will be further discussed in section 4.

Our results imply that an algorithm that is solely based on some measure for the (relative or absolute) error cannot determine a satisfying approximation of the topology. In other words, unstructured information such as the positions, sizes, and shapes of mesh elements does not suffice to decide whether or not two mesh elements are neighboured. One seems forced to gain a higher point of view and use *discrete techniques*. By that we mean that rules are applied in which geometrical and numerical details are abstracted away: some kind of *logical inference rules*, which incorporate more than two mesh elements in their reasoning. A simple example is that the relation of neighbourhoods has to be transitive: if patches A and B as well as patches B and C are neighboured, then A and C must also be neighboured. Such conclusions might add missing neighbourhoods along branchings like in Figure 6. Furthermore, an algorithm may use meta-information about the structure, the topology, of the workpiece. Special cases where the CAD model is the surface of a (connected) solid object or a plane deformed by a press are examples of strong meta-information: The graph induced by the mesh elements and neighbourhoods is planar.

In [10] we presented an algorithm for the reconstruction of the topology, which relies on structural techniques and which produced acceptable results for our benchmarks. As a by-product, the results presented here in retrospect justify the choice of a discrete approach in [10].

In summary, we were faced with a practical problem that does not seem to be mathematical in nature, since the dirtiness of the problem cannot be abstracted away: It is at the heart of the problem. This paper is a first attempt to design systematic studies on real-world data to support negative claims. It demonstrates that the dirtiness is not limited to artificial and pathological examples. Errors that caused the common approach to fail were found all over the test set. Cleaning the problem up to a (mathematically) nice problem would make a completely different problem of it – actually, the real problem would vanish.

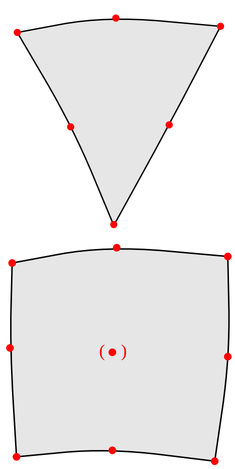

Figure 4: a triangular mesh element with six supporting points and a quadrilateral mesh element with eight supporting points on the boundary and one (optional) point in the interior.

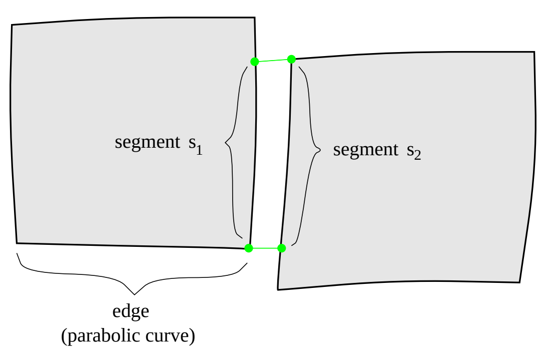

Figure 5: two neighboured mesh elements and the segments, s_1 and s_2, that constitute this neighbourhood.

2 Problem Description and Discussion

Input. To give a concrete, illustrative example, we will describe the format of the data that is available to us. A CAD model consists of a set of parametric surface patches. In our case, a patch can be three or four sided (Figure 4). The edges are parabolic curves, and the whole element is a special case of a *bicubic patch* as it is defined in chapter 5 of [9]. Basically, this means that the restrictions imply that the three sided patch is uniquely determined by six points. The four sided mesh elements exist in two versions, which are defined with eight or nine points, respectively.

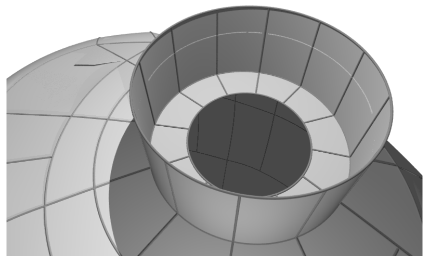

Figure 6: A mesh element can have edges that are not neighboured to any other edge. They may be around a *hole* or along a *rim*. Edges can also have more than one neighbour along so-called *branchings*.

Figure 5 demonstrates that edges need not be neighboured on a whole edge. On the other hand, not only two, but three or even more mesh elements may have an edge in common (Figure 6). Figure 6 also shows a rim and a hole as two exemplary constellations where a mesh element is not neighboured to any other mesh element on an edge.

Gaps. The main problem when finding neighbourhoods is that usually the CAD models contain significant gaps. These gaps have to be distinguished from holes that are intended by the designer of the CAD model: Consider two edges that are situated next to each other. The algorithm has to decide whether they are intended to be neighboured or just two separate edges with a hole in between. Since the gaps can be as large as in Figure 2 and Figure 3, this decision becomes a serious problem. (A few possible reasons why gaps of this size may occur will be discussed in section 3.) The problem is made even more difficult by the occurrence of branchings: If for one of a pair of edges a neighbour has already been found, this does not exclude that the edges are nevertheless neighboured.

Discrete information. We have found gaps that are larger than mesh elements of the same workpiece. This has some unpleasant consequences, which are depicted in Figure 7: On the left side, the upper and lower mesh elements are neighboured, whereas this is not the case on the right side, because of the triangle between them.

REMARK: The detection of this kind of constellation has been applied in [10]. It remarkably reduced the number of errors in the result of our algorithm.

Not only a mesh element between an examined pair of edges, but also the surrounding mesh elements must be incorporated into the interpretation of a local constellation. For example, in Figure 8 two gaps of the same length and width are shown. But in the upper case the gap continues on both sides and in the lower case the gap suddenly closes. This is a strong argument that the first gap is intended as opposed to the second one. Roughly speaking, all of these considerations are examples of what we mean by incorporating discrete information.

Claim. We have seen that in certain situations discrete information must be used. Based on a systematic computational study, we will show that such situations occur too often to be ignored: They are typical, not pathological.

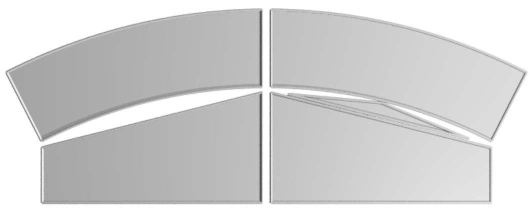

Figure 7: On the left side, the mesh elements are neighboured, but their mirror-symmetric counterparts on the right side are not, due to the triangle between them.

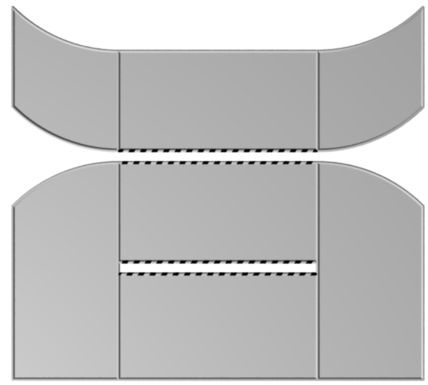

Figure 8: Two gaps of the same width and length. The surrounding mesh elements are a hint that the upper one is probably intended whereas the lower one is an error.

3 Background

Need for topology. The topology of a CAD model is essential for most automated tasks. For example, applying finite element methods only makes sense when forces or heat flow are transmitted correctly from element to element.

The topology is also necessary to refine or coarsen the mesh of a workpiece. Figure 9 shows an example of a CAD data model that was constructed manually, and a refined mesh of the same workpiece that was automatically produced by the algorithm in [5].

Furthermore, the topology is needed for smoothing surfaces or replacing several mesh elements by others of a simpler type. See [7] for a list of further tasks.

Reasons for gaps. As already stated in section 2, the unintented gaps are the main reason why the reconstruction of the topology is difficult. The gaps in Figure 2 and Figure 3 are obviously not due to mere numerical rounding errors. We have found several explanations why these and other gaps exist:

- While creating a workpiece, the designer can position the edges freely in the three dimensional space: Two edges can be neighboured on parts of their full length. The low order of the polygons that model the edge makes gaps unavoidable then.
- Conversions between non-isomorphic data formats sometimes make the approximation of a free form surface by another type necessary. This may also result in gaps like in Figure 3.
- Automatic tools that create CAD models from input devices like computer tomographs or 3d scanners do not have any knowledge about the topology of the workpiece. Errors and an inaccurate precision of measurement may then lead to gaps.

See [3] for a detailed discussion of these issues.

Existing approaches. Some restrictions on the input format as well as much cleaner CAD models sometimes allow the use of a distance measure in combination with a threshold value. Sheng and Meier examined in [8] the case when the surface is the boundary of a solid object. They also faced the problem of gaps that are as large as mesh elements. Their solution was to interact with the user in cases of doubt.

Knowledge about the discrete structure is used in very few existing approaches. In [2] Bøhn and Wozny restrict the problem to workpieces where the topology is already known for big parts (which may be viewed as another kind of discrete pre-knowledge). This is of course less difficult than a total reconstruction of the topology. Apart from this, their approach is heavily based on

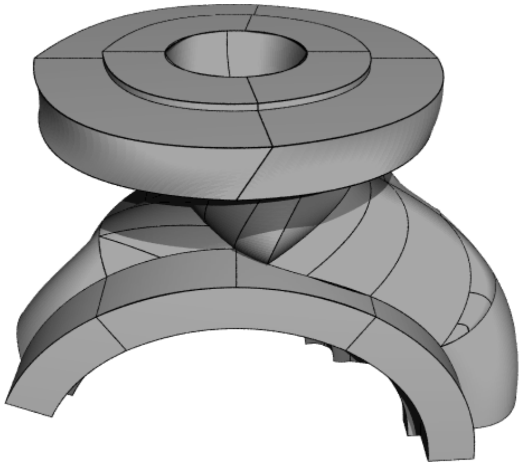

Figure 9: a CAD model of a pump.

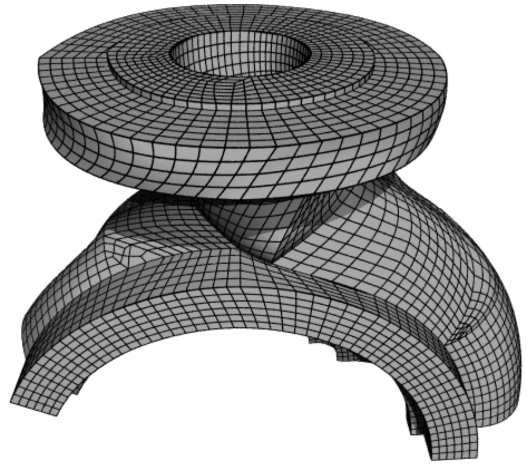

Figure 10: the same workpiece refined by the algorithm in [5].

the assumption that the mesh elements form the surface of a solid object. It incorporates the fact that the closed surface is orientable and uses the information on which side of the surface the solid object is situated. However, their algorithm can only find a special kind of gaps. The same problem was also treated by Barequet and Sharir in [1]. Their approach is not limited to a special kind of gaps. We presented an algorithm in [10] for the problem in the generality we described in section 2. In contrast, in our problem variant, the topology – or any other kind of meta-information – is not even partially available.

4 Methodology

Test cases. The workpieces that we have examined stem from industrial applications. We believe that it is impossible to systematically generate artificial instances that model realistic workpieces: A random set of parametric surface patches might never resemble a workpiece like a pump or a console. If we introduced artificial gaps into closed surfaces to imitate the dirtiness of the data, we would not have examined the nature of common errors, but our interpretation of them.

Examined class of algorithms. In this paper, we consider the class of algorithms that work according to the following pattern: An algorithm decides whether or not a pair of edges is neighboured by comparing the distance of the edges with a threshold value. The algorithms differ only in the used distance measure. Furthermore, we assume that the algorithms always find the best threshold value for each workpiece. In case of scaled distance functions, we even assumed the choice of the best threshold value for each single edge. Both assumptions are by far more than one can expect without human interaction.

Reference neighbours. We use a set of *reference neighbours* for making statistics and assessing the output of different algorithms. Since the problem is so ill-posed, it is impossible to automatically check the output of an algorithm otherwise. The reference neighbourhoods were produced by a commercial CAD tool [4] and our algorithm in [10]. The result was checked manually using the method described in [10]. Where necessary the neighbourhoods were corrected in this project and the project of [5].

Potential neighbours. It does not suffice to count the number of reference neighbourhoods missed by an algorithm; we also have to test for pairs of edges that an algorithm erroneously considers as neighboured. For a fair test, we do not examine all pairs of edges, but only those that are not obviously wrong. For a more precise explanation, recall that each parametric surface patch is bordered by its trimming curves. Each edge is a segment of such a parameterized curve. The segment is defined by a parameter interval of the parameter of the curve. If the projection

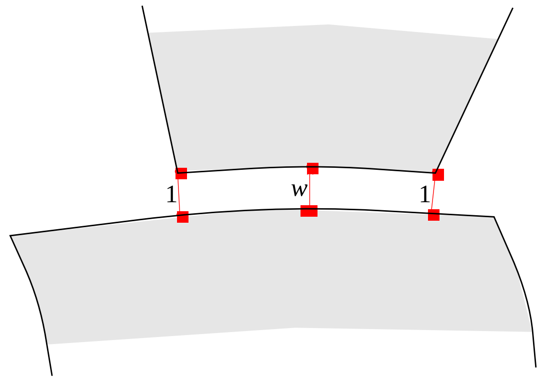

Figure 11: The distances of the end points are added to the distance in the middle of a segment with weight w to calculate the weighted distance.

of another edge onto this curve does not intersect with this interval, we exclude this pair of edges from our investigations.

Distance measures. We have examined two kinds of distance measures. The first one is represented by the distance measure that computes the mean value of the distances at ten pairs of mutually opposite points. These points are equidistant according to the parametrisation of the edge. We will call this distance measure *uniform*.

The distance measures of the second kind are *weighted*. They result from the insight that an engineer sets the end points of an edge manually, whereas the interior of the edge is interpolated automatically. Therefore, the end points might be more precisely positioned than the interior of the edge. We measure the distance of two edges at three pairs of points. Like in Figure 11, the distances at the end points are included with weight 1 for sake of symmetry. We refer to the individual distance measures by the weight of the third distance, which is the only difference between them. We will discuss at the end of this section, why we think that these distance measures are good representatives.

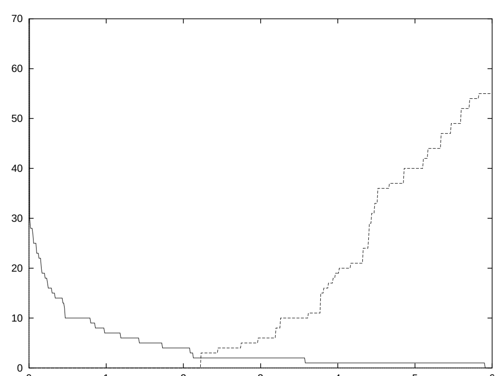

Figure 12: The number of reference neighbours that fail (decreasing line) and the number of pairs of edges that would be considered erroneously as neighboured (increasing line) for workpiece 8 and the uniform distance measure.

Examined minima. For each distance measure and workpiece we produced a diagram like Figure 12. The threshold value varies on the abscissa. The decreasing curve shows the number of reference neighbours that are missed by using the specified threshold value, whereas the increasing line documents the number of non-neighboured pairs of edges that are falsely found.

There are mainly two points of special interest in these diagrams. The first one is the total minimum number of faults – reference neighbours that are missed plus false neighbourhoods. The second interesting number is the minimum of wrong neighbours subject to the condition that all reference neighbourhoods are found. In [10] we discussed why this minimum can be more important than the first one. For each distance measure and each workpiece, we determined both values. This is what we meant above by claiming to implicitly assume an algorithm that always finds the best threshold value for every workpiece.

Relative error. Our experiments have shown that estimating the relative error with respect to geometric attributes does not produce better results than using the absolute error. For a meaningful visualization of the results, the estimation

$$\frac{\text{distance}}{\text{scaling}} \leq \text{threshold}$$

has been replaced by

$$\text{distance} \leq \text{threshold} \cdot \text{scaling}.$$

We have then determined all feasible threshold values for the (unscaled) uniform distance measure for every edge:

- The minimal feasible threshold value is the maximum of the distances to all reference neighbours.
- The maximal feasible threshold value is the minimum of the distances to all other edges.

A scaled threshold value for an edge between these bounds separates the correct neighbours of this edge from its false neighbours. We have examined the following four scalings:

1. the *length* of the edge,
2. the *perimeter* of the mesh element,
3. the *area* of the mesh element, and
4. the extent of the edge above the straight line through the end point of the edge as shown in Figure 13. We call this distance the *curvature* of the edge.

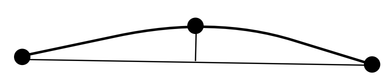

Figure 13: The *curvature* of an edge is measured by the distance of the supporting point in the middle of the edge to the line through the points at the ends of the edge.

We have generated a diagram for every scaling and workpiece. In such a diagram, for every edge a vertical interval according to the lower and upper bound of the threshold occurs. The intervals are positioned on the abscissa according to the value of the scaling for the particular edge. (An example of such a diagram is shown in Figure 14.)

Our goal is to decide whether there is a reasonably simple interrelation between geometric characteristics and feasible threshold values. In our opinion, a suitable visualization produces the most convincing arguments about the complexity of a possible interrelation – a function that passes through all intervals. In particular, we determined the curve with minimal length in the set of continuous functions that pass between the permitted bounds.

This function with minimal length can be constructed automatically: The upper and lower bounds for each appearing value generate a polygon in the two dimensional plane. We seek for

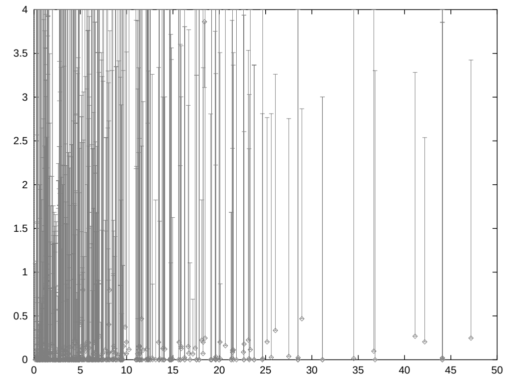

Figure 14: The possible threshold values for workpiece 12. Each interval shows the upper and the lower bound for one edge. Their position on the abscissa is the length of the edge.

the shortest path from the leftmost segment to the rightmost one. According to [6] section 8.2.1, a shortest path in a polygon is a subpath of the visibility graph of the vertices of the polygon. Up to a slightly different handling by replacing start and end points by segments, this solves our problem.

Representative distance measures. For performing our tests we had to choose different distance measures. We think that our distance measures are good representatives and include the most natural ones. It is obvious that the distance measure should rely on a metric of the three dimensional space. The metric has to be invariant under translation and rotation of the workpiece. Since the algorithm should produce the same result for a scaled workpiece, the distance must be scalable, and thereby only those metrices that are derived from a norm are reasonable. Since all of those metrics are equivalent in a sense, we selected the Euclidean distance as a representative. A natural extension of the Euclidean distance between two points to a distance measure of edges is the (discrete approximation of the) area between the edges.

In this paper, we consider some related distance measures: Other scalings than the length are treated, and different weights are used. Nevertheless, our results provide strong arguments that this does not improve the situation.

Of course, one might argue that a significantly more complicated and less natural distance measure could solve the problem. But the conclusions from the empirical results turned out not to be too sensitive to variations of the distance measure. Switching to another distance measure can improve or impair the quality of the output for a single workpiece, but on average the result does not change dramatically.

5 Computational Results

Absolute errors. We will first discuss the quality of the output for distance measures that represent the absolute error estimation. All of these distance measures failed miserably for some workpieces. Furthermore, none of the distance measures turned out to be generally preferable over the others.

In Table 1, the minima according to section 4 are listed for individual distance measures and workpieces. For model no. 6, the number of reference neighbours that would considered as wrong for a given threshold value did not drop to zero in the observed interval $[0; 10]$ for the uniform distance measure.

Number of supporting points. We will first compare the uniform distance measure to a distance measure with only three measured distances, two at the end points and one in the middle of the edge. The results for the latter are listed in the column labelled “weighted 1.0”. By comparing the two mentioned columns, the reader sees that the more fine-grained uniform distance measures

	neighbours		uniform		weighted		weighted		weighted		weighted	
No.	reference	potential			0.0		0.5		1.0		1.5	
1	55	1025	4	7	4	4	4	4	4	4	4	4
2	210	7860	0	0	0	0	0	0	0	0	0	0
3	676	35134	13	56	12	33	13	34	13	40	12	41
4	128	2304	0	0	0	0	0	0	0	0	0	0
5	573	23338	0	0	0	0	0	0	0	0	0	0
6	321	11525	26		18	72	18	72	20	72	22	72
7	823	48689	7	447	7	27	5	35	5	33	5	37
8	222	8669	2	55	0	0	0	0	1	5	2	11
9	1205	129596	0	0	0	0	0	0	0	0	0	0
10	1280	198055	0	0	0	0	0	0	0	0	0	0
11	40	567	0	0	0	0	0	0	0	0	0	0
12	409	20537	12	968	9	749	8	831	9	853	11	889
13	321	28860	3	3	3	3	2	2	2	2	2	3
14	446	29072	4	10	2	13	2	13	2	13	4	13
15	341	19085	0	0	3	3	0	0	0	0	0	0
16	261	17695	4	101	35	293	27	193	7	76	4	268

Table 1: Minimal number of wrong neighbourhoods for different distance measures. The weighted distance measure is determined according to Figure 11. For each distance measure, the first column lists the minimal number of false neighbourhoods – missing reference neighbours plus additional false neighbours. The second column contains the minimal number of errors, when all reference neighbours are found.

does not lead to a better result. This realization explains furthermore why we did not consider more than three supporting points for weighted distance measures.

Weighted distances. We have chosen four different values for the weight in the middle of the edge. The search for a favorable weight appears fruitless: For every distance measure there is at least one data file where this weight does not seem appropriate. In most cases, however, the choice of the weight influences the quality of the output only marginally: The results are almost the same for all four weights.

Relative errors. We will now present the upper and lower bounds for the threshold value and discuss a possible interrelation with scaling values of the mesh elements. In the data that is available to us, we have even found some cases where the upper bound was smaller than the lower bound. To proceed with our investigations, we increased the upper bound to match the lower one in these cases, in which no feasible threshold value exists at all.

The appendix contains diagrams for all four scalings. They are typical in the sense that all of these functions "zigzag" irregularly, if there is no feasible unscaled threshold value for the workpiece. There are not even two functions that reveal any resemblance. These and all other diagrams show that probably there is not any interrelation between a suitable threshold value and the examined scalings.

6 Conclusion and Outlook

Summary. The common approach of using a distance measure and a global threshold value seems to fail for a variety of unscaled distance measures. Beyond this, we have seen that also a relative error estimation does not improve the situation even if the threshold value is individually chosen for each edge. The empirical results suggest that the problem cannot be solved completely without additional discrete information.

Source of discrete structure. Unfortunately, the topology is not available to an algorithm, since it is the output of the algorithm itself. This leads to an obvious contradiction. As presented in

[10], this conflict can be resolved. We first generated a rough approximation of the topology with a conventional approach. However, this source of a discrete structure was still good enough to be used by a discrete algorithm that applied logical inference rules as described above to correct errors in the topology. This dramatically improved our result.

Global properties. In some special cases, there exist global characteristics that can help reconstruct the topology. For example, the knowledge that each edge is neighboured to at most or maybe even exactly one other edge. This knowledge may be provided by the user or implicitly given in the input. For instance, the surface of a solid object cannot contain any intended holes. A program processing such input can therefore safely close each gap since it knows that all gaps are errors.

Final remark. By treating this problem, a difficulty becomes apparent that seems to be inherent to some practical problems. The standard stategy of analysing the problem, lifting it to an abstract, mathematical model, and solving the abstract problem efficiently does not work here. One has to "stay in the mud".

References

[1] Gill Barequet and Micha Sharir. Filling gaps in the boundary of a polyhedron. *Computer Aided Geometric Design*, 12:207-229, 1995.

[2] Jan Helge Bøhn and Michael J. Wozny. A Topology-Based Approach for Shell-Closure. *Geometric Modeling for Product Realization*, IFIP Transactions B-8 p. 297-319, 1993.

[3] Georges M. Fadel and Chuck Kirschman. Accuracy issues in CAD to RP translations. In *Internet Conference on Rapid Product Development.* MCB University Press, 1995. http://www.mcb.co.uk/services/conferen/dec95/rapidpd/fadel/fadel.htm.

[4] Günter Krause. Interactive finite element preprocessing with ISAGEN. *Finite Element News*, 15, 1991.

[5] Rolf H. Möhring, Matthias Müller-Hannemann, and Karsten Weihe. *Mesh refinement via bidirected flows; modelling, complexity, and computational results.* Journal of the ACM 44(3), p. 395-426, 1997.

[6] Joseph O'Rourke. *Computational Geometry in C.* Cambridge University Press, 1994.

[7] Stephen J. Rock and Michael J. Wozny. Generating topological information from a 'bucket of facets'. In H.L. Marcus et al., editors, *Solid Freeform Fabrication Symposium Proceedings*, pages 251-259, 1992.

[8] Xuejun Sheng and Ingo R. Meier. *Generating Topological Structures for Surface Models.* IEEE Computer Graphics and Applications Vol. 15, No. 6. 1995.

[9] Su Bu-qing and Liu Ding-yuan. *Computational Geometry – Curve and Surface Modeling.* Academic Press, Inc., 1989.

[10] Karsten Weihe and Thomas Willhalm. Reconstructing the Topology of a CAD model - a discrete approach. *Proceedings of the 5th European Symposium on Algorithms, ESA 1997*, Springer Lecture Notes in Computer Science 1284, p. 500-513.

Appendix: Diagrams about the Examined Scalings

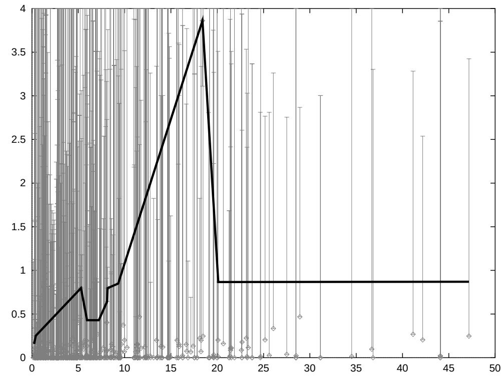

Figure 15: Approximating function for a possible threshold value depending on the length of edges for workpiece 12,

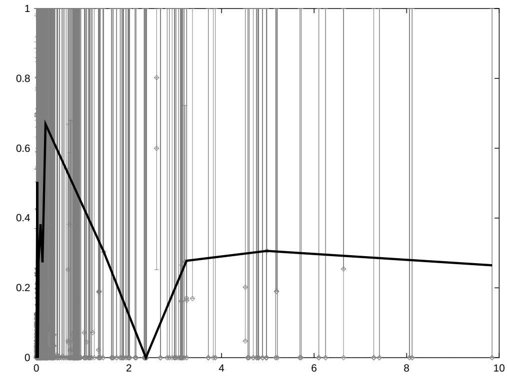

Figure 16: depending on the curvature of edges for workpiece 3,

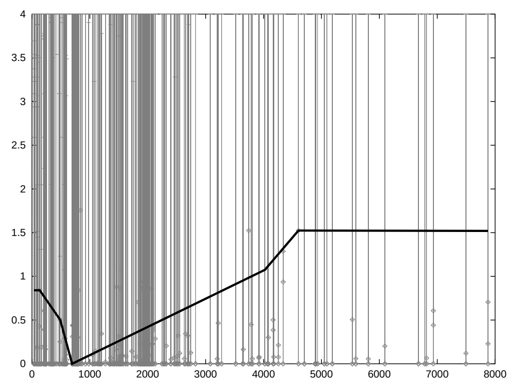

Figure 17: depending on the area of mesh elements for workpiece 14, and

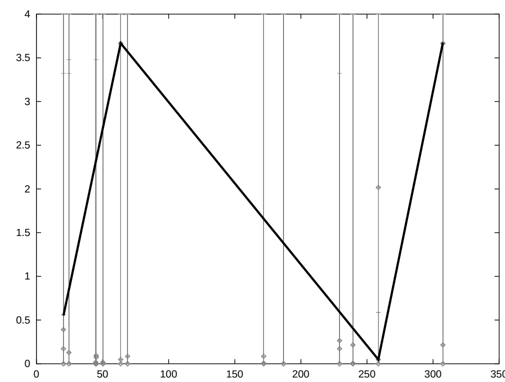

Figure 18: depending on the perimeter of mesh elements for workpiece 1.