Non-Local Topological Clean-Up

Guy Bunin

Department of Physics, Technion, Haifa 32000, Israel
buning@tx.technion.ac.il

Abstract. A new approach to topological clean-up of 2D meshes is presented. Instead of searching for patterns in a mesh and replacing them as in other methods, the proposed method replaces a region in a mesh only according to the boundary of that region. This simplifies the classification of the different cases, and allows mesh modification over greater regions in the mesh. An algorithm for quadrilateral meshes utilizing this approach is presented in detail, and its effects on example problems are shown.

1. Introduction

Topological clean-up is the name given to methods aimed to improve the quality of a mesh, by changing its connectivity [1,2,3,4,6]. It is usually carried out after a mesh has been generated by some meshing algorithm. The goal of the clean-up is to decrease the number of nodes that are attached to too few or too many cells. In an all-quadrilateral mesh, if there are nodes attached to more or less than 4 cells, some cells will be forced to have inner angles different than 90 degrees. Decreasing the number of irregular nodes could therefore improve the quality of the mesh.

The methods presented in the literature on the subject are usually based on predefined "cases" or "patterns". Cases are specific configurations of the mesh connectivity that, once found in the mesh, are modified according to the case found. Because the number of possible configuration of a mesh increases dramatically with the configuration size, the cases are *local*, spanning a region of a few cells at most. By applying local clean-up operations successively on a mesh, many problematic configurations can be resolved. There are, however, possible mesh improvements that are not covered by these methods.

In this paper a new clean-up method is proposed. In this approach, the configurations are regions of the mesh classified according to the *boundary* of the region. The mesh connectivity is modified by replacing the mesh of the region with a different mesh connectivity, that was created by "topologically meshing" the interior of the boundary of that region. In other words, a new mesh connectivity conforming to the boundary of the replaced region is created, and if this new connectivity has a better topological quality, it can replace the existing mesh of the region. The method can be applied to improve the mesh structure even if irregular nodes are not close to each other, thus allowing the clean-up of cases not specified by previous methods.

The method presented, like other topological clean-up methods, does not take the vertex locations into account. Therefore, if applied without any restrictions, it might reduce the geometric quality of the mesh. This is especially true near mesh boundaries, since boundary vertices cannot be moved, and the amount of possible mesh smoothing is more limited. In such cases it may be better to use techniques that are more geometric in nature. For further discussion see section 2.5, where criteria for allowing clean-up operations are suggested.

The paper is organized as follows. In chapter 2 the algorithm is described. Chapter 3 presents examples of clean-up results. Chapter 4 presents conclusions and possible directions for future research.

2. Algorithm

2.1 Algorithm Overview

In the method suggested, every topological clean-up operation consists of 3 steps, see figure 1. First, a simply-connected part of the mesh is selected (Fig. 1, (1)). Then, the inside of the loop surrounding the chosen part of the mesh is "topologically-meshed", i.e. a new mesh-connectivity is created (Fig. 1, (2)). This "meshing" phase is based on the loop structure, which is determined by the first layer of cells outside the loop. Finally, if the new topological-mesh meets a number of conditions, the principal condition being the improvement of the topological structure, the existing part of the mesh is replaced by the new mesh (Fig. 1, (3)).

After a replacement takes place the new mesh nodes have no location. The geometric stage, of assigning coordinates to the vertices can be done as a fourth step after the replacement, or perhaps at once after all the topo-

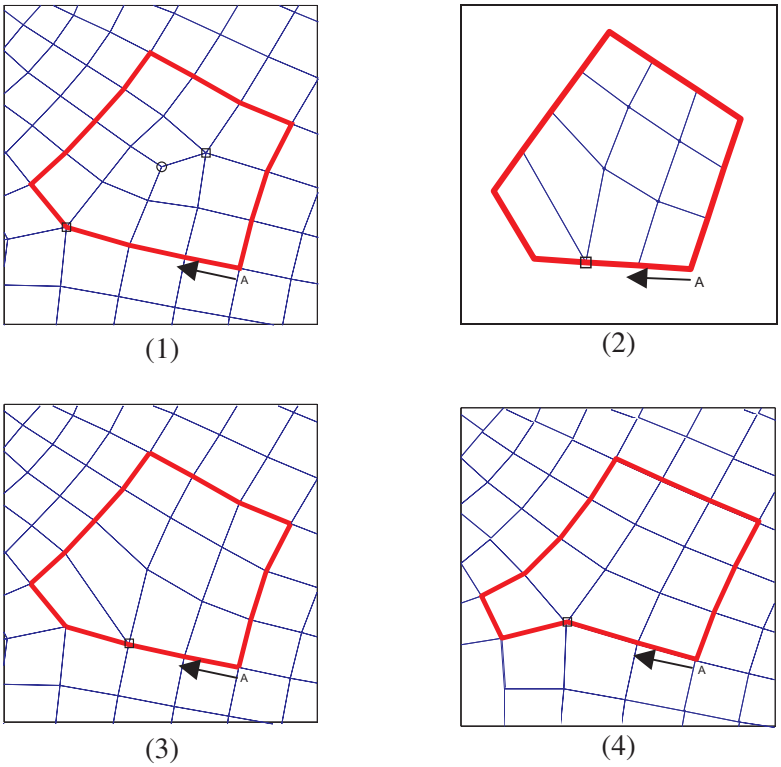

Fig. 1. The steps of a single clean-up operation. (1) A loop surrounding a part of the mesh is chosen. (2) The loop is given a different topological-mesh (connectivity only), based on the topological structure of the loop. (3) The mesh inside the loop is replaced by the new mesh created in step 2. (4) The new nodes are given coordinates, and the whole mesh is smoothed. (This part can also be done after all clean-up operations are over).

logical clean-up process has ended (Fig. 1, (4)). The problem of assigning locations to the new vertices will not be addressed here. For research on the subject see [7] and references therein.

In what follows only the case of quadrilateral meshes is addressed, though the basic principle may be extended to other cases, such as triangular and mixed meshes.

2.2 Definitions

- Node Valence – the number of cells (or edges) incident on the vertex.

- Clean Mesh – a mesh in which all nodes are of valence 4.
- Defect – a node of valence other than 4[1].
- Loop – a closed path traversing edges in a mesh.
- Topological Outer Angle – a closed loop in a mesh divides the mesh into two parts: the cells inside the loop, and the cells outside the loop. A vertex which is on the loop itself will therefore be incident upon cells inside and outside the loop. The outer angle of a vertex on a loop is calculated to be (see Figure 2):

$$< \text{Topological Outer Angle} > = \tag{1}$$

$$<\text{number of outside cells incident}> - 2$$

If the vertex is on the mesh boundary, a factor of $\alpha / 90°$, α being the geometric outside angle of the boundary, is added to the right hand side of equation (1), to account for the lack of cells outside the mesh.

- Convex Loop – a convex loop is a loop whose nodes all have a non-negative topological outer angle.
- Corner of a Convex Loop – a corner of a convex loop is a vertex with a positive outer angle.
- Side of a Convex Loop – the edges of a loop between two given corners.

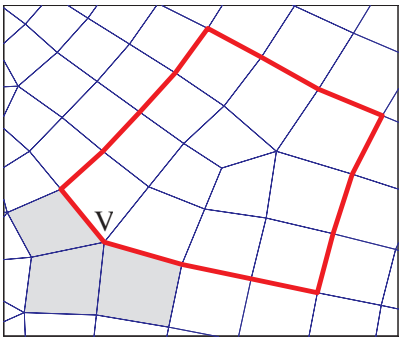

Fig. 2. The vertex V in the figure is incident upon 3 cells that are outside the loop. According to Eq. (1), its topological outer angle is 3 – 2 = +1.

[1] Some authors refer to it as an irregular node.

2.3 Remeshing a Convex Loop

This section describes a way of topologically meshing a convex loop. That is, a new mesh connectivity inside a loop is created. This new connectivity is used in step 2 of the algorithm, after a loop has been chosen.

The input to the algorithm is the loop, and the topological outside angle (TOA) of each vertex on the loop. Since it is a convex loop (see definition above) the TOA is non-negative. The following algorithm has an even more restricted input domain: it only deals with loops that have TOA's of 0 or +1. This restriction actually limits the input to a topological analog of a polygon: there are N sides and N corners (see definition above), each corner has TOA = +1.

One Defect Solution

There is a class of convex loops that can be meshed with only one defect in the mesh. Figure 3 shows such a mesh, for a convex loop of 3 sides. As can be seen in the figure, certain constraints on the lengths of the sides should hold. We will discuss these relations now.

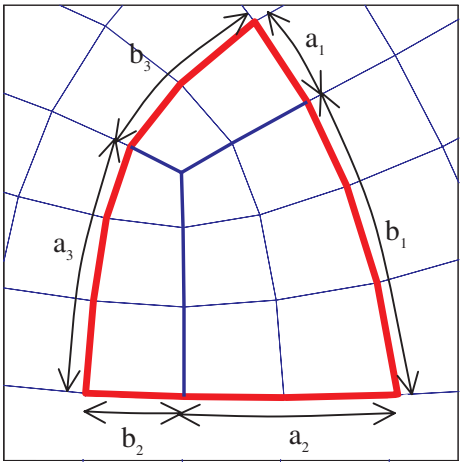

Fig. 3. A single-defect mesh inside a convex loop, for N = 3. The mesh can be viewed as composed of N structured meshes (clean meshed with 4 sides). The seaming of the loop constrains the side lengths. For N = 3: $b_1 = a_3; b_2 = a_1; b_3 = a_2$.

A single defect mesh of N sides can be viewed as N structured meshes seamed together. Every side of the loop is composed of 2 sides of the 2 logical meshes, see figure 3. Let l_i, I = 1...N be the length of the loops sides, and a_i, b_i the lengths of the 2 parts of each l_i. Then $a_i + b_i = l_i$. Moreover, due to the seaming of the sides of the logical meshes, some parts must have equal lengths, for example $b_1 = a_3$, $b_2 = a_4$, etc., or in general: $b_i = a_{(i+1) \bmod N+1}$. For a loop with N sides, the constraints can be summarized in the following matrix equation:

$$M \begin{pmatrix} a_1 \\ \vdots \\ a_n \\ b_1 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ l_1 \\ \vdots \\ l_n \end{pmatrix} \tag{2}$$

Where $M = \begin{pmatrix} -D_2 & I \\ I & I \end{pmatrix}$ is a 2N x 2N matrix, and D_2 is the N x N identity matrix, with columns N-1,N brought to the beginning (becoming columns 1,2). E.g. for N = 3, D_2 is:

$$D_2 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{3}$$

And for N = 5 D_2 is:

$$D_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \tag{4}$$

[2] A structured mesh is a clean mesh with 4 sides.

When N is not a multiple of 4, the matrix M can be inverted. Then the matrix equation (2) can be solved. If the a_i and b_i of the solution vector are all non-negative integers, this solution represents a single defect mesh. If not, the loop cannot be meshed with a single defect.

Many Defects

This section describes an algorithm for meshing loop that cannot be meshed with a single defect. We limit ourselves to loops having 3, 4 or 5 sides. We first define a "penalty factor" for loops. For loop with 3 or 5 sides:

$$P = \min(0, abs(\min_i(a_i, b_i))) \tag{5}$$

where a_i,b_i are the solutions to equation (2) for the loop. A loop with penalty 0 can be meshed with one defect. A loop with positive penalty can only be meshed with more defects[3].

For 4-sided loops we define the penalty as follows. Let (a,b,c,d) be the side lengths of the loop. Then we define the penalty of the loop using the penalty defined for 3-sided loops.

$$P_4 = 1 + \max(P_3(a, c, abs(d-b)), P_3(b, d, abs(c-a))) \tag{6}$$

Here P_3, P_4 are the 3-sided loop and 4-sided loop penalties, respectively. The rationale behind this definition will be clear below, when we describe the meshing algorithm of 4-sided loops.

We now describe the meshing algorithm. The algorithm presented is recursive. For the case of a loop with 3 sides, the penalty of the loop is computed, Eq. (5). If P=0, then the loop can be meshed with one defect, completing the meshing of this loop. Otherwise, one of the sides is "broken" into two, and the resulting 4-sided loop is meshed recursively. This effectively puts a 3 defect into the mesh, see figure 4,A. The location of the "brake" is chosen according to the penalty of the 4-sided loop it would create.

For a 5-sided loop, if the penalty of the loop is 0, the loop is meshed with one defect. If not, the algorithm joins two adjacent sides of the loop

[3] Actually, it can be proved that an odd number of defects with valence 3 or 5 is required.

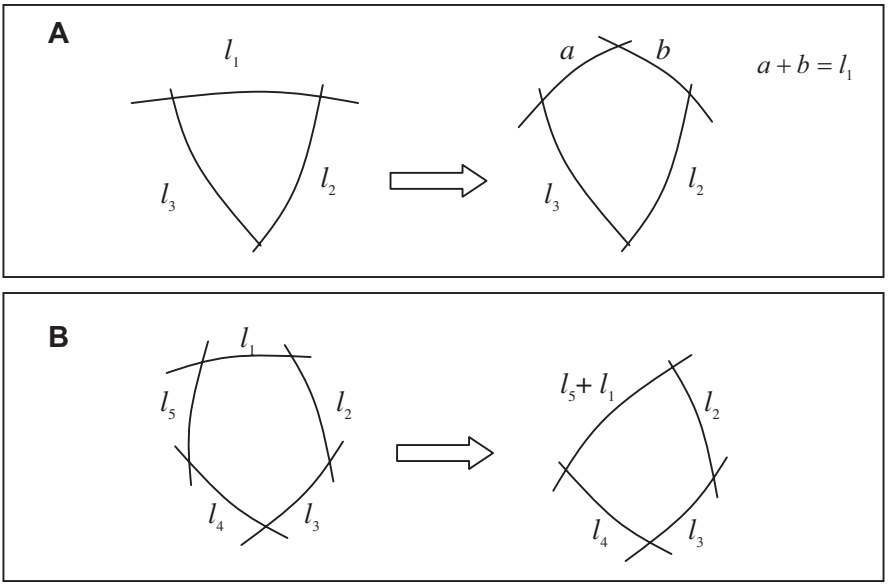

Fig. 4. Meshing of 3-sided (A) and 5-sided (B) loops that cannot be meshed with one defect. The loops are converted to 4-sided loops and meshed as such.

into one, and sends the result to be meshed as a 4-sided loop, see figure 4,B. The joining effectively creates a 5-defect. The sides to be joined are chosen according to the penalty of the 4-sided loop that would be created.

For meshing a 4-sided loop, let a,b,c,d be the side lengths of the loop. If $a = c$ and $b = d$, then the loop can be meshed with a clean mesh. Otherwise, the algorithm adds rows until a triangle is formed, see figure 5. Since the rows stretch between a pair of opposite sides, there are two possible directions for adding the rows, see figure 5A,B. The direction of adding the rows is chosen according to the penalty of the resulting triangle. Note that, in the side that was closed, a 5-defect is formed. This meshing method is the reason for the penalty definition for 4-sided loop: the penalty is just one plus the better of the penalties of the triangles that can be formed by adding rows. If the penalty of a 4-sided loop is 1, it can be meshed with 2 defects. If the penalty is higher, more defects are required.

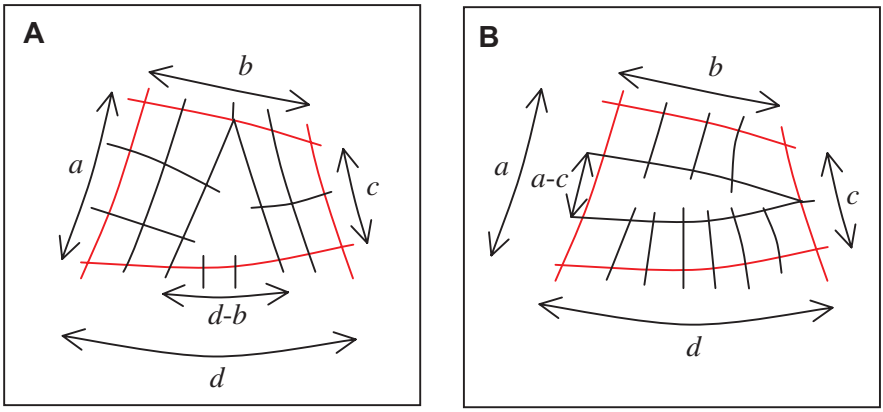

Fig. 5. Meshing of a 4-sided loop. In the case shown $a = 3; b = 4; c = 2; d = 7$. The 2 possible directions for adding rows are presented. The 3-sided loop left in case A takes a penalty of 0. The penalty for the loop in case B is 1.

2.4 Choosing the Loop to Remesh

The first step in a single clean-up operation is choosing a region to remesh. This region should preferably contain defects, so that a clean-up operation may be possible. There are various ways of finding loops around defects. Presented here is a very simple way; there are other, perhaps better ways.

The algorithm starts from a single defect in the mesh, called the "seed defect". A single layer of cells is added by adding all cells that are incident upon this defect. This group of cells will be called a "blob". If the loop surrounding this blob is not a convex loop (i.e. some outer angles are negative, see figure 6), then cells incident on the vertices with negative outer-angle are added to the blob. This process is repeated until the loop is convex. The loop is then sent to the topological meshing algorithm described in section 2.3. If a mesh replacement has been made, a new seed defect is found, and the process is repeated from the beginning. Otherwise, if no mesh replacement has been made, the loop is grown further by adding the cells incident on the blob, and making the loop convex again. This process is repeated until a replacement has been made, or it is certain that no replacements will be made by extending the blob further.

The seed defects are chosen by preferring defects that are furthest away from the boundary. If no replacement was made using the innermost defect as a seed defect, the process is repeated with the second innermost one, and so on. Once a mesh replacement made, the process starts again from the innermost defect.

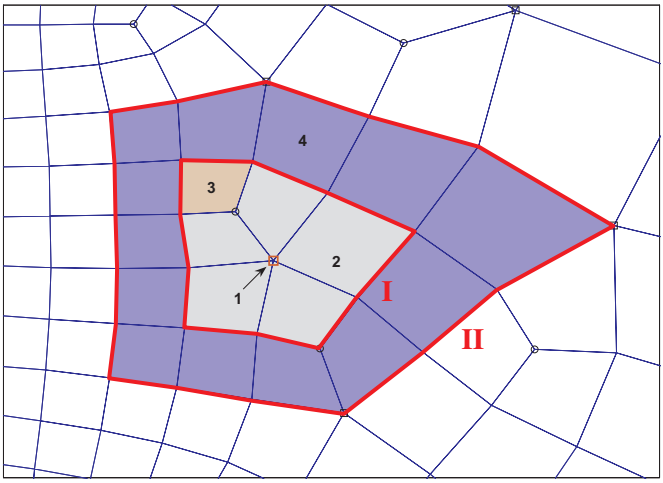

Fig. 6. Choosing a mesh region to be replaced. **1.** The seed defect. **2.** The cells incident on the seed defect. **3.** The cell added to the first layer (2) to make its loop convex. The first loop (designated as I) surrounds cells of (2)+(3). **4.** If no mesh replacement is made for loop I, the blob is extended. Loop II surrounds the larger blob.

2.5 Criteria for Allowing the Replacement

Certain criteria must be met for the mesh replacement to be held. The first, obvious demand is that the replacement will improve the topological quality of the mesh. The topological quality is defined as the sum of how much the valence of the nodes varies from 4. A mesh replacement will only be carried out if this sum is lower after the replacement.

Also, there are convex loops which the topological-meshing algorithm described above cannot mesh, such as a loop containing a node with topological outer angle greater than +1. Obviously, in such a case no replacement is made.

A topological clean-up process does not take into account the geometry of the mesh. This is less problematic when making mesh replacements far from the boundary of the mesh, where a smoothing process can move the mesh vertices to a great extent without creating low quality cells. Close to the mesh boundary, however, the mesh has less flexibility due to the fixed location of the boundary vertices, and the replacement is more risky. The following conditions help avoid creating poor cells.

A mesh replacement is allowed only if:

1. The blob radius (defined by the number of times it was grown, see section 2.4) is smaller than the distance of the closest defect in the blob to the boundary.

2. The new defects (if any) in the replaced mesh are further from the boundary than the defects in the existing blob.

The conditions above are of heuristic nature, but some intuitive reasoning can be given in their favor. The first condition stems from the idea that big modifications of the mesh (modifications over great distances) require more flexibility of the mesh that exists only far from the mesh boundary. The second condition reflects the importance of well aligned rows along the boundaries, as well as a reasoning similar to the one done for the first condition, to say that the flexibility of the mesh far from the boundary can help in reducing the effect of the defects on mesh quality.

3. Examples

In the examples below, we show the results of applying the clean-up algorithm on sample meshes. The input meshes used were created with the MSC/PATRAN software. To allow a cleaner comparison, all meshes were smoothed with Laplacian smoothing.

It is worth while noting that the mesh generator used *already contains* its own topological clean-up stage [2], so the comparison presented here shows mesh improvement on-top of what was achieved using the clean-up described in [2].

In the comparisons shown below 2 different quality measures are used. The shape measure β [4,5] is used to evaluate the quality of a single cell. As in [5], it is defined here as the minimum of the shape metrics of the 4 triangles that can be formed by the vertices of the quadrilateral. The purpose of the second measure is to evaluate cell size transitions. A gradual size transition is an important characteristic of a mesh. To quantify this property of a mesh we define a size transition metric as follows. For every pair of neighboring cells (cells that share an edge) the area of the larger cell is divided by the area of the smaller one. The size transition metric of the mesh is the average of this quotient over all pairs of neighboring cells. It is always larger or equal to 1. A gradual size transition will be characterized by a metric close to 1.

The first example shows a uniform size mesh. Figures 7-8 show the mesh before and after clean-up. Mesh defects are marked by squares or circles, according to node valence. Figure 9 compares the shape quality histograms. As can be seen, there is a great reduction in the number of cells with shape quality $\beta < 0.7$. This is directly connected to the reduction in the number of defects, as is shown in figure 10, in which the mesh

before the clean-up is drawn again, with cells having $\beta < 0.7$ filled in black. As can be seen, amongst the cells surrounding a node with valence 3 there is typically one with $\beta < 0.7$. Table 1 presents a comparison of mesh characteristics for this example.

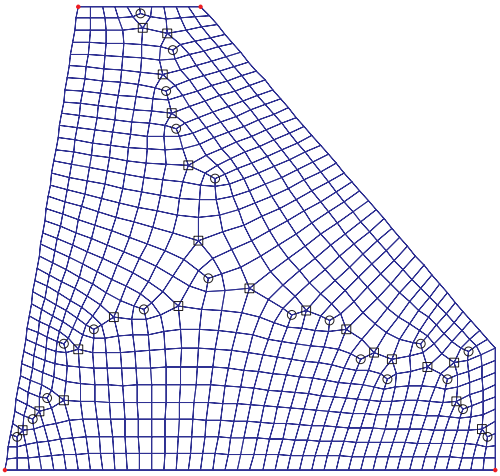

Fig. 7. Mesh of example 1 before clean-up.

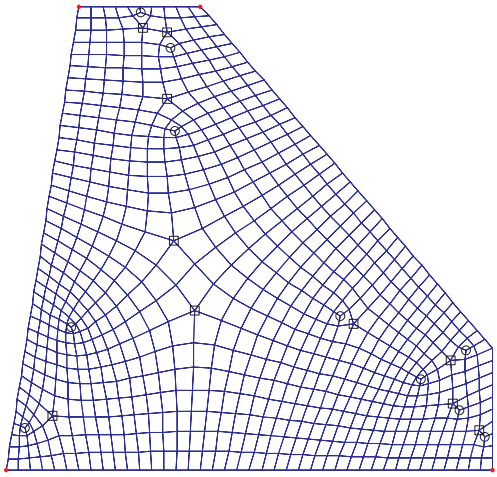

Fig. 8. Mesh of example 1 after clean-up.

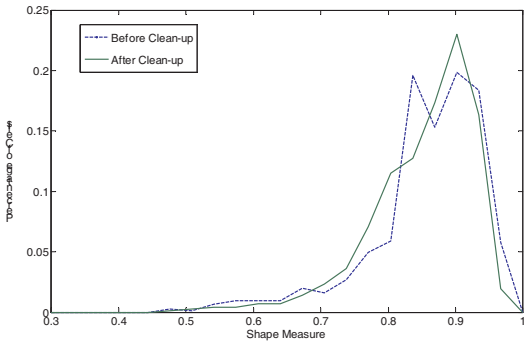

Fig. 9. Comparison of shape quality distributions in example 1 before and after clean-up.

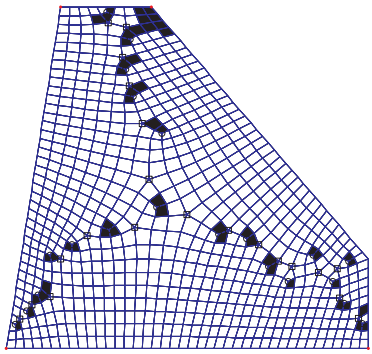

Fig. 10. Mesh of example 1 before clean-up; cells with $\beta < 0.7$ are filled.

Table 1. Mesh characteristics of example 1.

	Before Clean Up	**After Clean Up**
Number of Cells	745	721
Number of Defects	42	20
Average Metric β	0.857	0.852
Minimal Metric β	0.478	0.464
Number of Cells with $\beta < 0.7$	52	34
Size Transition Metric	1.115	1.119

The second example shows a mesh that contains a large change in mesh size, due to variation in boundary edge length, see Figs. 11,12. A new feature seen here is an improvement in the size transition. The size transition metric is reduced from 1.223 to 1.166.

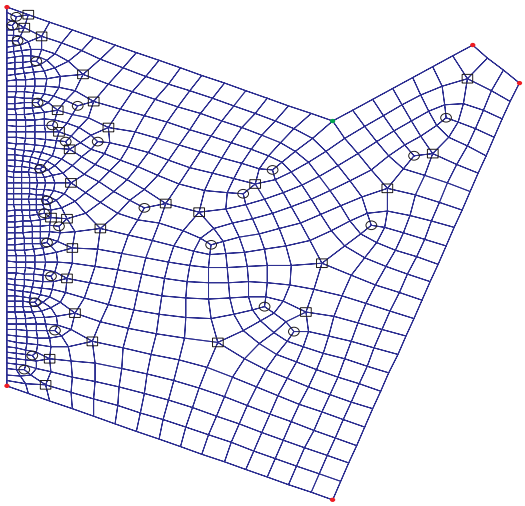

Fig. 11. Mesh of example 2 before clean-up.

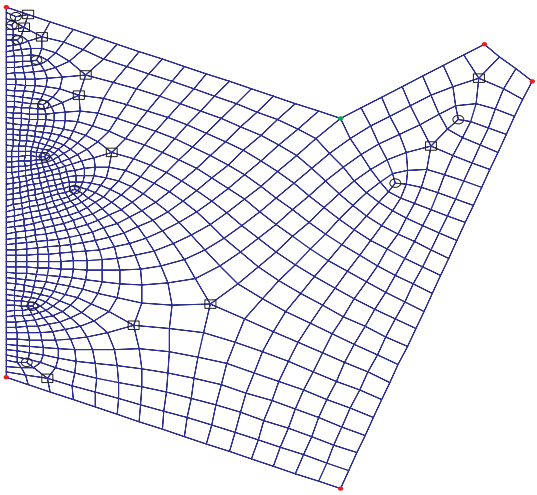

Fig. 12. Mesh of example 2 after clean-up.

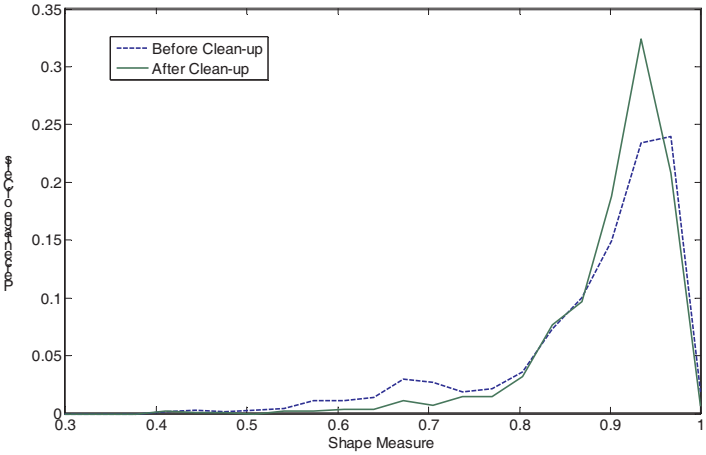

Fig. 13. Comparison of shape quality distributions in example 1 before and after clean-up.

Table 2. Mesh characteristics of example 2.

	Before Clean Up	**After Clean Up**
Number of Cells	696	802
Number of Defects	56	22
Average Metric β	0.881	0.900
Minimal Metric β	0.410	0.408
Number of Cells with $\beta < 0.7$	61	28
Size Transition Metric	1.223	1.166

The third example shows the action of the algorithm on a mesh with concave boundaries, and an interior boundary (hole).

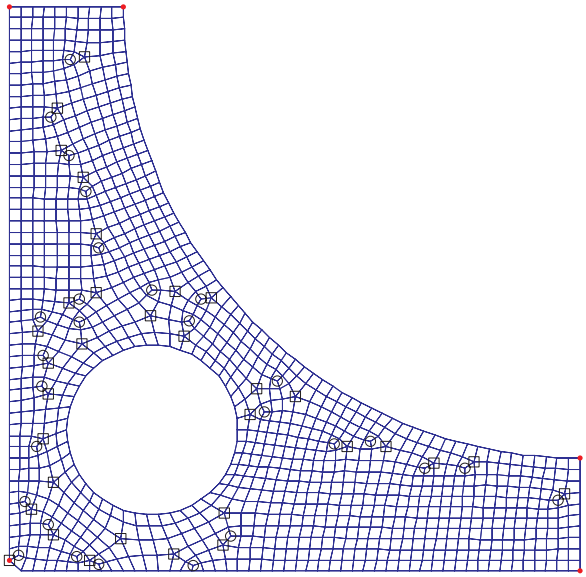

Fig. 14. Mesh of example 3 before clean-up.

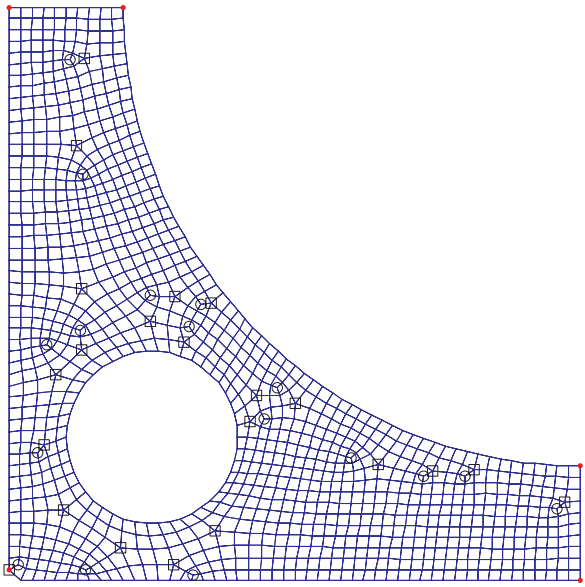

Fig. 15. Mesh of example 3 after clean-up.

Table 3: Mesh characteristics of example 3.

	Before Clean Up	**After Clean Up**
Number of Cells	1047	1041
Number of Defects	61	39
Average Metric β	0.900	0.906
Minimal Metric β	0.466	0.471
Number of Cells with $\beta < 0.7$	58	45
Size Transition Metric	1.107	1.108

4. Conclusions

A new approach to topological clean-up of 2D meshes was proposed. The basic clean-up operation in this method is creating a new connectivity for a region of the mesh, and replacing the existing mesh of the region. The new connectivity is created using information on the structure of the boundary of that region only. If this new connectivity is thought to improve the mesh quality, a replacement of this mesh region is made.

An algorithm utilizing this approach for quadrilateral meshes was presented in detail, and the results of applying it on example meshes were shown. These examples, as well as others, suggest that a large reduction in the number of cells with lower shape quality ($\beta < 0.7$) can be achieved. This fact is closely tied to the reduction in the number of defects (irregular nodes) in the mesh. The effect on the average shape quality is usually small, a few percent at most. Another observation is that the clean-up can help improve the cell size transitions in meshes where there are significant changes in cell size.

The proposed approach seems to be especially beneficial for clean-up far from the boundary. There, replacement operations over larger distances can be carried out, and the relative advantage over local clean-up techniques is more pronounced. This makes the technique a good complement to advancing front mesh generators, which tend to create defects inside the mesh, where the fronts collide in the meshing process.

Further research can include finding improved region selection algorithms, and perhaps improved topological meshing methods. Good algorithms for finding legal locations for the new nodes after replacement are important as well; the speed and robustness of the mesh replacement stage relies on them. The general approach can probably be applied to triangular and mixed quad-tri meshes as well.

Acknowledgements

The author wishes to thank Shlomi Hillel, Dov Levine, Zev Lovinger and Irad Yavne for many helpful discussions.

References

1. Blacker T.D., Stephenson M.B., *Paving: A New Approach to Automated Quadrilateral Mesh Generation*, Int. Journal for Numerical Methods in Eng., Vol. **32**, pp. 811–847, 1991.
2. Canann S.A., Muthukrishnan S. N., Phillips R. K., *Topological Improvement Procedures for Quadrilateral Finite Element Meshes*, Engineering with Computers, Vol. 14, no. 2, pp. 168–177, 1998.
3. Kinney P., *Clean Up: Improving Quadrilateral Finite Element Meshes*, Proc. of the 6th International Meshing Roundtable, pp. 449–461, 1997.
4. Lee C.K., Lo, S.H., *A new Scheme for the Generation of Graded Quadrilateral Meshes*, Computers and Structures, Vol. **52**, pp. 847-857, 1994.
5. Owen S.J., Staten M.L., Canann S.A., Saigal S., *Advancing Front Quadrilateral Meshing Using Triangle Transformations*, Proc. of the 7th International Meshing Roundtable, 1998.
6. Staten M.L., Canann S.A., *Post Refinement Element Shape Improvement for Quadrilateral Meshes*, **AMD-220**, Trends in Unstructured Mesh Generation, ASME, pp. 9–16, 1997.
7. Vachal P., Garimella R. V., Shashkov M. J., *Untangling of 2D meshes in ALE simulations*, J. of Computational Physics, vol. **196**, no. 2, pp. 627–644, 2004.