UNTANGLING HIGH-ORDER MESHES BASED ON SIGNED ANGLES

Mike Stees[1] Myra Dotzel[2] Suzanne M. Shontz[3]

[1]*Department of Electrical Engineering and Computer Science, Information and Telecommunication Technology Center, University of Kansas, 1520 W. 15th Street, Lawrence, KS, 66045, USA, mstees@ku.edu*

[2]*Department of Electrical Engineering and Computer Science, Department of Mathematics, Information and Telecommunication Technology Center, University of Kansas, 1520 W. 15th Street, Lawrence, KS, 66045, USA, myradotzel@ku.edu*

[3]*Department of Electrical Engineering and Computer Science, Bioengineering Program, Information and Telecommunication Technology Center, University of Kansas, 1520 W. 15th Street, Lawrence, KS, 66045, USA, shontz@ku.edu*

ABSTRACT

One challenge in the generation of high-order meshes is that mesh tangling can occur as a consequence of moving the new boundary nodes to the true curved boundary. In this paper, we propose a new optimization-based method that uses signed angles to untangle invalid second- and third-order triangular meshes. Our proposed method consists of two passes. In the first pass, we loop over each high-order interior edge node and minimize an objective function based on the signed angles of the pair of triangles that share the node. In the second pass, we loop over face nodes and move them to the mean of the high-order nodes of the triangle to which the face node belongs. We present several numerical examples in two dimensions with second- and third-order elements that demonstrate the capabilities of our method for untangling invalid meshes.

Keywords: high-order mesh untangling, optimization, curvilinear triangular meshes

1. INTRODUCTION

One appealing aspect of high-order methods for solving partial differential equations is their ability to obtain more accurate solutions with a lower computational overhead than the corresponding low-order methods. One barrier to the adoption of these methods in the presence of curved domains is the lack of software capable of robustly generating high-order meshes [1]. In particular, to achieve the full potential of high-order methods in the presence of curved domains, these methods need to be paired with a high-order mesh that conforms to the curved domain [2, 3].

The typical approach used by high-order mesh generation methods is to apply a transformation to a coarse low-order mesh [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]. The main difficulty in applying the transformation is obtaining a valid high-order mesh as the result. In general, these methods consist of the following three steps: (1) additional nodes are added to the low-order mesh; (2) the newly-added boundary nodes are projected onto the curved domain, and (3) the interior nodes are moved as a result of the boundary deformation. There are generally two approaches which are especially popular for transforming the low-order mesh. The first approach transforms the mesh based on optimization of an objective function [4, 6, 7, 11, 12, 14, 15, 16, 17, 20, 21]. Many of the proposed objective functions include a measure of element validity, which allows the methods to address

invalid elements. While not all of the methods guarantee successful untangling, many of them are robust [4, 6, 7, 11, 12, 17]. The second approach transforms the mesh based on the solution of a partial differential equation [5, 8, 10, 19]. More specifically, Xie et al. [19] employed a linear elasticity approach, while Persson and Peraire [10] considered a nonlinear elasticity approach. Moxey et al. [8] used a thermoelastic model, and Fortunato and Persson [5] expressed the problem in terms of the Winslow equations.

In this paper, we describe a new two-pass method for untangling invalid second- and third-order triangular meshes. The first pass is an optimization-based approach that minimizes an objective function based on signed angles for each high-order interior edge node. The second pass is a smoothing step for the face nodes. The main focus of this work is to untangle invalid meshes that result from the boundary curving step of a typical high-order mesh generation method. Toward that end, we apply our method to several second- and third-order meshes that have invalid elements following the boundary curving process. The remainder of this paper is organized as follows. In Section 2, we present our new two-pass method for high-order mesh untangling. In Section 3, we demonstrate the performance of our method on several two dimensional examples. Finally, in Section 4, we offer concluding remarks and discuss some directions for our future work.

2. UNTANGLING HIGH-ORDER CURVILINEAR MESHES

In this section, we propose a two-pass local node-based method for untangling high-order curvilinear triangular meshes. The first pass is based on the signed angles of curvilinear triangles, where a negative angle indicates tangling. For each iteration of the problem, we consider a high-order interior edge node. Then, we identify the two triangles that share the node and examine the four angles made by the tangent vectors adjacent to that edge. Our algorithm then moves the high-order edge node with the goal of making these angles positive. In our first pass, we solve the following unconstrained optimization problem:

$$\begin{aligned} f(x) &= (1-\beta)||\mathbf{x}-\mathbf{x_I}||_2 + \beta\sum_{i=1}^{4}\mathbf{e}^{-\mathbf{10}*\alpha_\mathbf{i}(\mathbf{x})}, \\ x^* &= \underset{\mathbf{x}}{\operatorname{argmin}} f(x). \end{aligned} \tag{1}$$

where $\alpha_\mathbf{i}$ is the i^{th} entry of the vector of the four signed angles adjacent to a given interior edge; $\mathbf{x}$ is the nodal position of the high-order edge node to be moved; $\mathbf{x_I}$ is the initial position of the node at the start of the optimization, and β is a user-defined weighting parameter. By changing the value of β, more emphasis can be applied to the angles or the displacement of the node from its initial position. Note, if too much emphasis is placed on the displacement of the node, then the norm will dominate the objective function values, and the mesh will not be untangled.

To better understand the behavior of the objective function, consider the examples shown in Fig. 1 and the corresponding values shown in Fig. 2(a). The β value in this example was 0.35. In Fig. 1(a), we show the initial tangled mesh. At this point, the first term of $f(x)$ is zero because the interior node has not been moved. The second term will thus dominate the value of $f(x)$. In Fig. 1(b), we show the mesh after applying two iterations of the optimization method. As we see in the first two rows of Fig. 2(a), in both examples, the exponential term is the primary contributor to $f(x)$ because of α_3, the negative angle. In Fig. 1(c,d), we see that the impact of the exponential term decreases as the values of the angles increase (e.g., from negative to positive) after four and nine iterations, respectively. In other words, the second term in $f(x)$ acts as a penalty function to enforce positive angles (i.e., an untangled mesh). Once the angles become sufficiently positive, then the first term in $f(x)$ becomes a larger contributor to the overall value of $f(x)$. The goal of this term is to reduce the amount of displacement for a given node by minimizing the node's distance from its initial location.

To find a local minimum of our unconstrained optimization problem, we use a derivative-free method. We do so because of the complexity of evaluating $f(x)$, specifically, the signed angle calculations. In particular, to solve our optimization problem, we use the Nelder-Mead simplex method [22]. For the motivational example in Fig. 1, a relaxed convergence tolerance of 0.01 was used for the Nelder-Mead simplex method. For all of our examples in the next section, the tolerance and maximum number of iterations for Nelder-Mead were 0.0001 and 400, respectively. Convergence is reached when the change in function values and the step size both satisfy the tolerance.

As described in [22], the Nelder-Mead simplex method is a direct search method that maintains a simplex at each step of the method. This simplex is defined by $n+1$ vertices and the corresponding function values, where n is the dimension of the problem space. Before moving forward, let us introduce the following notation for the description of the 2D method. Let the vertices of the current simplex be represented as v_1, v_2, and v_3. In addition, denote their corresponding function values $f(v_1)$, $f(v_2)$, and $f(v_3)$. Given these definitions, each iteration of a typical Nelder-Mead method consists of the following steps. First, the vertices are ordered from the lowest function value, say $f(v_1)$, to

their highest function value, say $f(v_3)$. Second, the midpoint m of the best side of the simplex is computed, i.e., the side opposite v_3. Third, a new simplex is computed from the current one using reflection, expansion, or contraction steps. In Fig. 3, we show examples of the reflection, expansion, and contraction steps, denoted by r, e, and c_o/c_i, respectively. We also illustrate the current simplex with a solid black line and the simplices computed via the operations in dashed black lines. To compute the new simplex, an attempt is made to replace v_3 by reflecting the vertex about the best side. If the reflected vertex r leads to a decrease in the objective function, then an attempt is made at further reduction by computing an expansion vertex e. If $f(e) < f(r)$, then v_3 is replaced with e. Otherwise, v_3 is replaced with r. If the reflected vertex does not lead to a decrease in the objective function, then r is contracted back to c_o, and the function values are compared again. If this step fails to decrease the function, then c_o is reflected about the best edge to get c_i. If all of these steps are unsuccessful, then the simplex is shrunk toward vertex v_1, and a new simplex is formed with v_1, m, and the midpoint between v_1 and v_3

After minimizing the objective function for every high-order interior edge node (i.e., completing the first pass of our untangling algorithm), we perform a second pass to move the non-edge nodes. In this pass, for each non-edge node, we move the node to the mean of the high-order nodes of the triangle to which it belongs. To better motivate the need for two passes, we have included an example in Fig. 4. In Fig. 4(a), we show the initial tangled mesh. In Fig. 4(b), we show the mesh after completing the first pass of our method. Since the objective function applied in the first pass is formulated in terms of angles between edges, which do not apply to face nodes, the first pass neglects to improve the quality of these elements with respect to their face nodes. Thus, a face node's close proximity to the edge of its element could result in an invalid element. To address this kind of situation, we have included the second pass as shown in Fig. 4(c). This pass moves the face nodes toward the interior of the elements to which they belong. These two passes are performed until a tolerance is satisfied. In Alg. 1, we provide a pseudocode description of our untangling method. In the next section, we discuss how the signed angles $\alpha_i(x)$ of the curved elements are calculated.

2.1 Computing the signed angles of curvilinear triangles

To compute the angle between two curves at a given node, we compute the derivatives of the curves, evaluate the derivatives at the given node, and then compute the angle between the resulting tangent vectors.

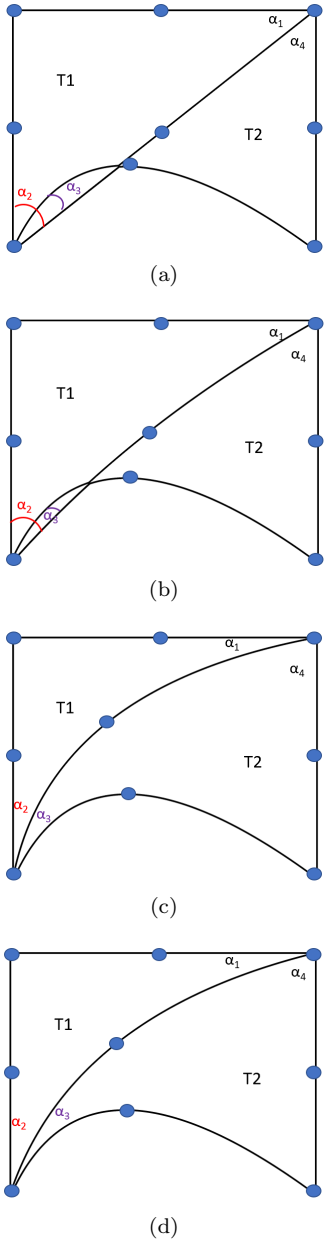
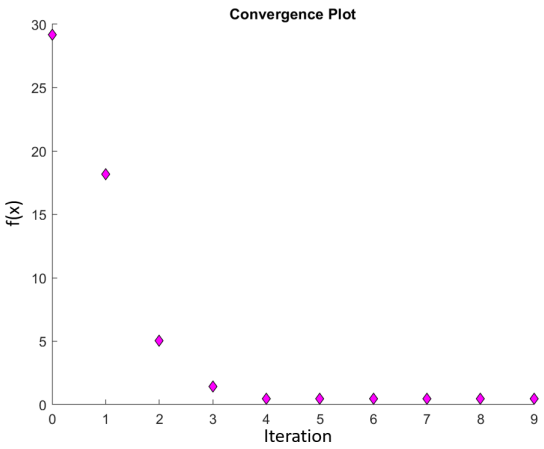


Figure 1: Motivating example: (a) the initial tangled mesh, (b) the mesh resulting from our method after two iterations, (c) the mesh resulting from our method after four iterations, and (d) the final mesh resulting from our converged method.

Iteration Number	Term 1	Term 2	$f(x)$
0	0.0000	29.1931	29.1931
2	0.0812	4.9445	5.0257
4	0.3096	0.1620	0.4716
9	0.2538	0.1963	0.4501

(a)



(b)

Figure 2: Figure showing (a) the contributions of each term in $f(x)$ during different iterations of the optimization method, and (b) a convergence plot of our method applied to the example in Fig. 1.

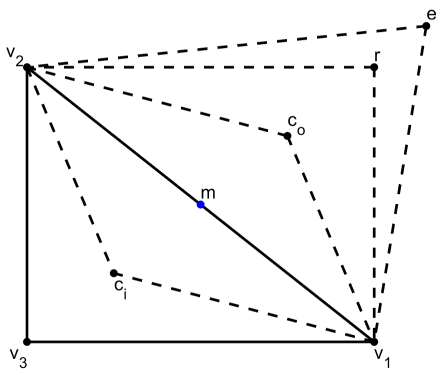
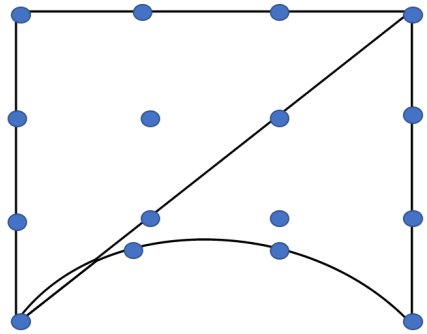
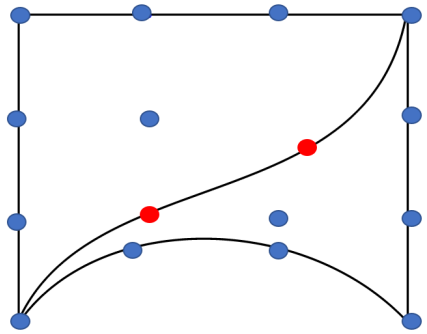


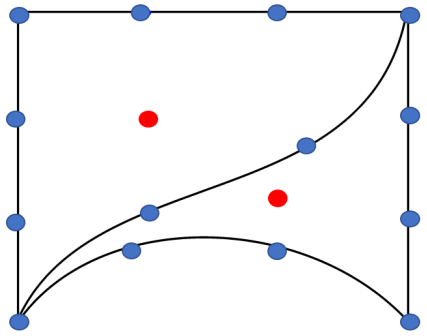
Figure 3: The current simplex marked by a solid line, and the simplices computed using the reflection, expansion, and contraction inside/outside operations during a single iteration of a typical Nelder-Mead method.



(a)



(b)



(c)

Figure 4: Motivating example for two pass approach: (a) the initial tangled mesh, (b) the mesh after completing the first pass with impacted nodes shown in red, and (c) the mesh after completing the second pass with influenced nodes shown in red.

Algorithm 1 Pseudocode for our node-based mesh untangling method

X^0 = the zero matrix
X^1 = the matrix of node positions at iteration 1
while $\frac{||X^k - X^{k-1}||_F}{||X^k||_F} > 10^{-4}$ **do**
  First Pass:
  for each high-order interior edge node i **do**
    1. Find the two triangles t_1 and t_2 which share node i
    2. Solve (1) for x^* using Nelder-Mead simplex method [22]
    3. Update nodal position i to x^*
  end for
  Second Pass:
  for each high-order interior face node i **do**
    1. Find the triangle t_1 which contains node i
    2. Update nodal position i to the mean of t_1's high-order edge nodes
  end for
  X^{k+1} = the matrix with updated node positions
end while

Using this approach, we compute the angles between each pair of edges of curvilinear triangles. For the following derivation, we use the third-order Lagrange element. The derivation for other orders is similar.

Consider the third-order Lagrange triangle shown in Fig. 5. To compute the angles between each pair of edges, we need to define mappings from each node on the edges of the reference element to the corresponding node on the edges of the physical element. Each edge corresponds to a third-order Lagrange element in one dimension. The shape functions associated with these elements are defined as:

$$\begin{aligned} n_1(t) &= \frac{9}{2}(1-t)\left(\frac{2}{3}-t\right)\left(\frac{1}{3}-t\right), \\ n_2(t) &= \frac{27}{2}(1-t)\left(\frac{2}{3}-t\right)(t), \\ n_3(t) &= \frac{27}{2}(1-t)\left(\frac{1}{3}-t\right)(-t), \\ n_4(t) &= \frac{9}{2}\left(\frac{2}{3}-t\right)\left(\frac{1}{3}-t\right)(t). \end{aligned}$$

The derivatives of these shape functions with respect to t are given by:

$$\begin{aligned} n_1'(t) &= \frac{1}{2}\left(-11+36t-27t^2\right), \\ n_2'(t) &= \frac{1}{2}\left(18-90t+81t^2\right), \\ n_3'(t) &= \frac{1}{2}\left(-9+72t-81t^2\right), \\ n_4'(t) &= \frac{1}{2}\left(2-18t+27t^2\right). \end{aligned}$$

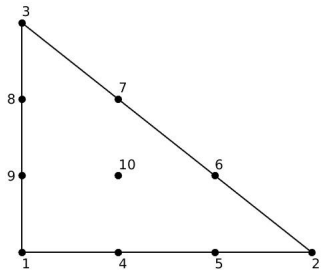


Figure 5: Third-order Lagrange reference unit triangle

Using these shape functions, we can define the mappings from each edge in the reference element to each edge in the physical element as:

$$\begin{aligned} \mathbf{f_{12}(t)} &= \mathbf{x_1}n_1(t) + \mathbf{x_4}n_2(t) + \mathbf{x_5}n_3(t) + \mathbf{x_2}n_4(t), \\ \mathbf{f_{23}(t)} &= \mathbf{x_2}n_1(t) + \mathbf{x_6}n_2(t) + \mathbf{x_7}n_3(t) + \mathbf{x_3}n_4(t), \\ \mathbf{f_{31}(t)} &= \mathbf{x_3}n_1(t) + \mathbf{x_8}n_2(t) + \mathbf{x_9}n_3(t) + \mathbf{x_1}n_4(t). \end{aligned}$$

The notation f_{ij} denotes the edge between nodes i and j in Fig. 5. Now that we have the mappings, we need to compute the derivatives of our functions. Taking the derivative with respect to t results in the following:

$$\begin{aligned} \mathbf{f_{12}}'\mathbf{(t)} &= \mathbf{x_1}n_1'(t) + \mathbf{x_4}n_2'(t) + \mathbf{x_5}n_3'(t) + \mathbf{x_2}n_4'(t), \\ \mathbf{f_{23}}'\mathbf{(t)} &= \mathbf{x_2}n_1'(t) + \mathbf{x_6}n_2'(t) + \mathbf{x_7}n_3'(t) + \mathbf{x_3}n_4'(t), \\ \mathbf{f_{31}}'\mathbf{(t)} &= \mathbf{x_3}n_1'(t) + \mathbf{x_8}n_2'(t) + \mathbf{x_9}n_3'(t) + \mathbf{x_1}n_4'(t). \end{aligned}$$

Given these derivatives, we can return to the problem of calculating the angles between edges. As an example, suppose that we want to calculate the angle between edge e_{12} and edge e_{31} in Fig. 5. To calculate the unsigned angle in radians, we could use the following formula:

$$\theta = \arccos\left(\frac{-f_{12}'(0)\cdot f_{31}'(1)}{||f_{12}'(0)||\ ||-f_{31}'(1)||}\right) = \frac{\pi}{2}.$$

In order to calculate the signed angle in radians, we need to modify our calculations. First, we need to include an orientation unit vector $\mathbf{n}$. Then we need to modify our tangent vectors by adding a third component with a value of zero so that the cross product is defined, as well as normalize them. With these modifications, we can compute the signed angle using the following formula:

$$\text{signed angle} = \text{sgn}(\mathbf{n}\cdot(v1\times v2))\cdot\arccos(v1\cdot v2)$$

where

$$\begin{aligned} \mathbf{v1} &= \frac{[f_{12}'(0),0]}{||\,[f_{12}'(0),0]\,||_2}, \\ \mathbf{v2} &= \frac{[-f_{31}'(1),0]}{||[-f_{31}'(1),0]||_2}, \\ \mathbf{n} &= [0,0,1]. \end{aligned}$$

3. NUMERICAL RESULTS

In this section, we demonstrate the results from applying our method to untangle several high-order meshes. In each example, the nodes are processed in the order in which they occur in the original mesh. While we have explored other node orderings and found that the order does impact the number of outer iterations required for convergence, we note that this ordering does not influence the final resulting mesh. For each example, we provide a description of the mesh, the initial mesh (with tangled elements shown in red), the mesh which results from applying our untangling method, and the mesh element distortion as measured by the scaled Jacobian:

$$\text{scaled Jacobian} = \frac{\min J(\xi)}{\max J(\xi)},$$

where $J(\xi)$ is the Jacobian determinant. When reporting the mesh distortion, we list the minimum distortion and maximum distortion values. We also list the execution times for our untangling method (excluding I/O) in Table 1. The method was implemented in C++, and the wall-clock execution times were measured on a machine with 16GB of RAM and an AMD Ryzen 7 1700 CPU. All mesh visualizations and distortion evaluations were done using Gmsh [23, 24, 25].

In our first example, we use a simple annulus geometry consisting of 30 elements to show the impact of different values of β on the result. In Fig. 6(a), we show the initial mesh with two tangled elements. In Fig. 6(b-d), we show the meshes resulting from β values of 0.1, 0.5, and 0.9, respectively. In Fig. 6(e), we show the min and max element distortions and execution times for each of the three values of β. As expected, higher values of β place more emphasis on the angular component of the objective function which tends to result in larger displacements of the edge nodes. Initially, increasing the value of β from 0.1 to 0.5 led to better elements with respect to distortion. Beyond 0.5, additional emphasis on the angles resulted in increased element distortion. For the remaining examples in this section, we report the value of β that resulted in the mesh with the least distortion. We also plot histograms of the mesh element distortion in addition to reporting the maximum and minimum values.

In the second example, we applied our method to a simple 2D mechanical part consisting of 295 second-order elements. Curving the boundaries resulted in two tangled elements near the innermost boundary. The initial tangled mesh and resulting untangled mesh are shown in Fig. 7(a,b). The minimum and maximum distortion values for these meshes are shown in Fig. 7(c). Finally, we plot histograms for these distortion values in Fig. 7(d,e). In this case, our solution raised the minimum distortion value of the mesh from -0.178 to 0.228.

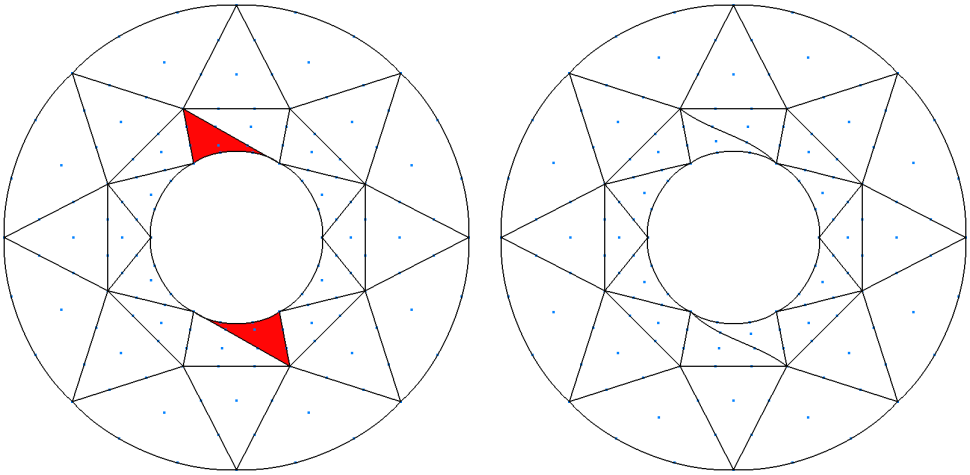
In our third example, we use a simplified bike gear with 672 second-order elements. In contrast with our previous examples, this mesh has several stretched elements near the boundaries which increase the potential for tangled elements after curving the boundaries. The initial tangled mesh and untangled mesh are shown in Fig. 8(a,b). Close-up views of the top third of the mesh are depicted in Fig. 8(c,d). The minimum and maximum distortion values for this mesh are recorded in Fig. 8(e). Lastly, histograms of the distortion values are plotted in Fig. 8(f,g). In this case, our method increased the minimum distortion value from -1.730 to 0.211, thus untangling the initial mesh.

As our last simplified example, we use a pressure plate consisting of 529 second-order elements. After curving the boundaries, six tangled elements were created along the holes in the top and bottom of the geometry. Fig. 9(a,b) show the original tangled mesh and the untangled mesh resulting from our method. We show detailed views of the center of (a,b) in Fig. 9(c,d), respectively. In Fig. 9(e) we give the minimum and maximum mesh element distortion values. Finally in Fig. 9(f,g) we plot histograms of the distortion values. For this example, our method increased the minimum distortion value from -0.178 to 0.345.

For our remaining examples, we progress to more realistic meshes with a larger number of elements. The first example is a third-order gear composed of 1340 elements, eight of which are tangled. In Fig. 10(a,b) we show the initial tangled mesh and the final mesh produced by our untangling algorithm. In Fig. 10(c,d) we show detailed views of the center holes in (a,b), respectively. The minimum and maximum mesh element distortion values are listed in Fig. 10(e). Finally in Fig. 10(f,g) we plot histograms of the distortion values. After applying our method, the minimum distortion value increased from -0.122 to 0.092.

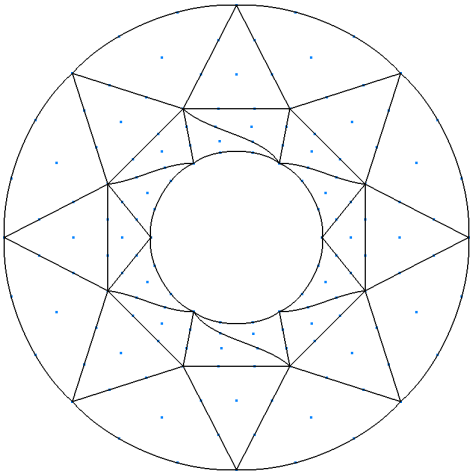
Our next example is a brake rotor composted of 7015 second-order elements, thirty-four of which are tangled. In Fig. 11(a,b) we show the initial tangled mesh and the final mesh produced by our untangling algorithm. In Fig. 11(c,d) we show detailed views of the center holes in (a,b), respectively. The minimum and maximum mesh element distortion values are listed in Fig. 11(e). Finally in Fig. 11(f,g) we plot histograms of the distortion values. After applying our method, the minimum distortion value increased from -0.156 to 0.346.

Finally, our last example is an anisotropic boundary layer mesh of an airfoil. This example is a modified version of an example taken from the 2D benchmarks

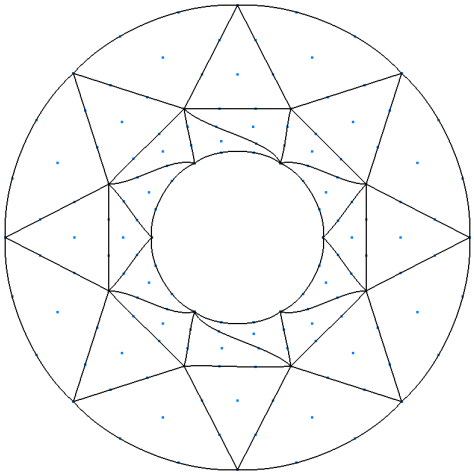


(a)

(b)



(c)

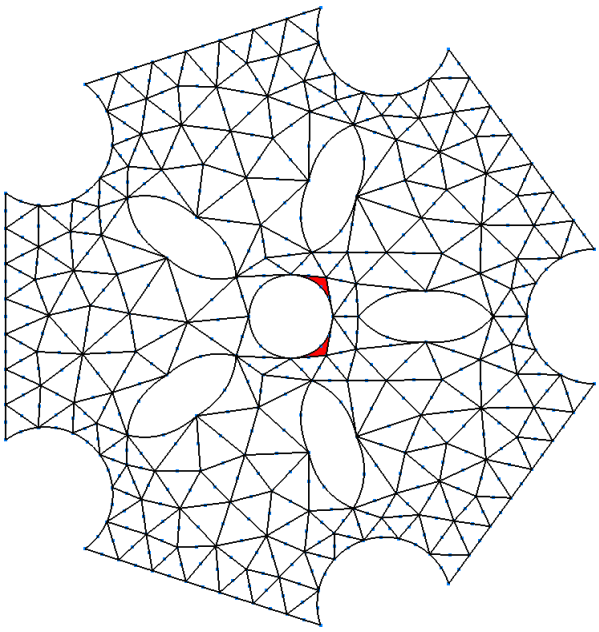


(d)

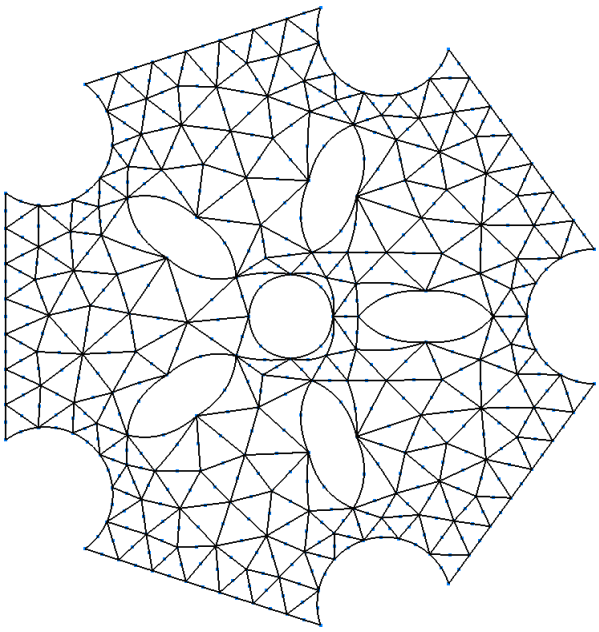
	Distortion		
Beta	**Min**	**Max**	**Runtime (s)**
0.1	0.208	1.000	0.004
0.5	0.472	1.000	0.005
0.9	0.348	1.000	0.014

(e)

Figure 6: Annulus example with three different β values: (a) the initial mesh with two tangled elements; (b) to (d) untangled meshes for β values of 0.1, 0.5, and 0.9, respectively, and (e), shows the minimum and maximum element distortions and runtimes for each value of β.



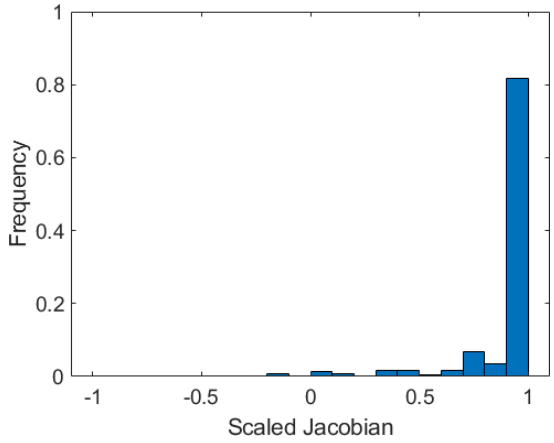
(a)



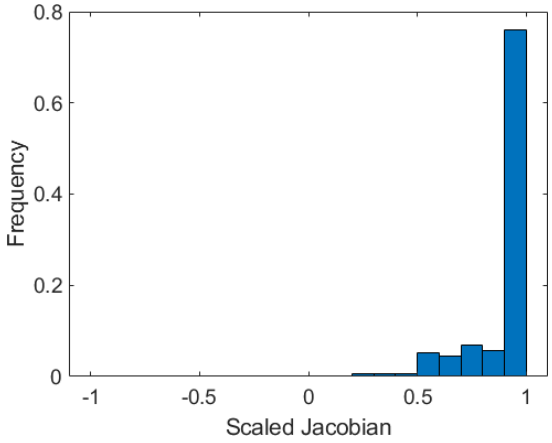
(b)

	Distortion	
Example	**Min**	**Max**
original mesh	-0.178	1.000
resulting mesh	0.228	1.000

(c)

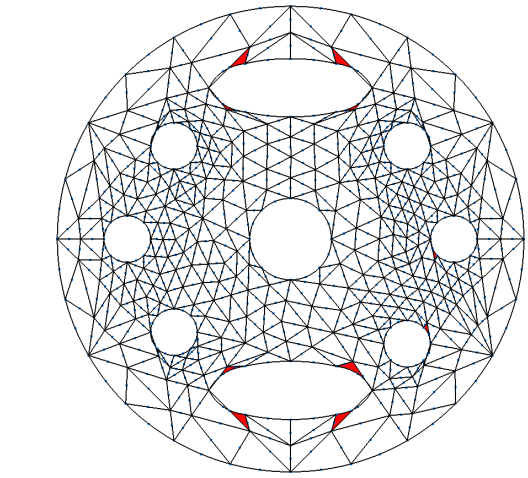


(d)

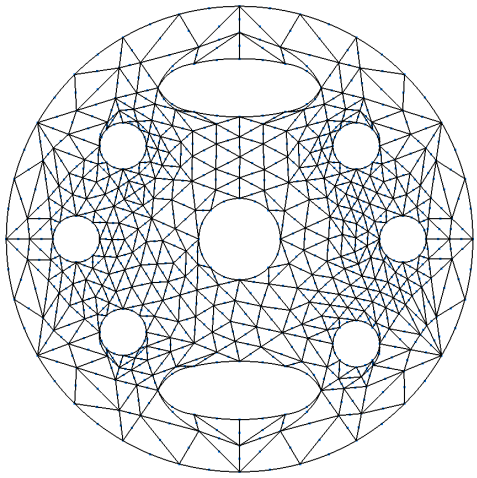


(e)

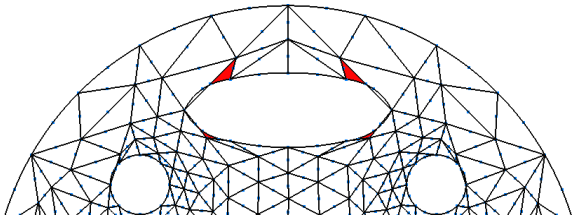
Figure 7: Mechanical part example: (a) the initial second-order mesh with two tangled elements; (b) the untangled mesh resulting from our method; (c) the minimum and maximum element distortion, and (d,e) histogram plots of the distortion metric for each mesh.



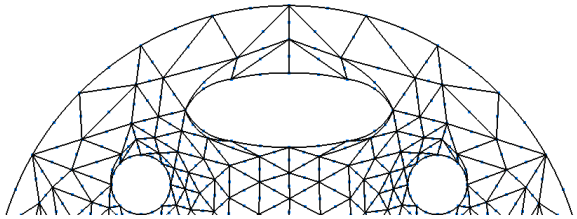
(a)



(b)



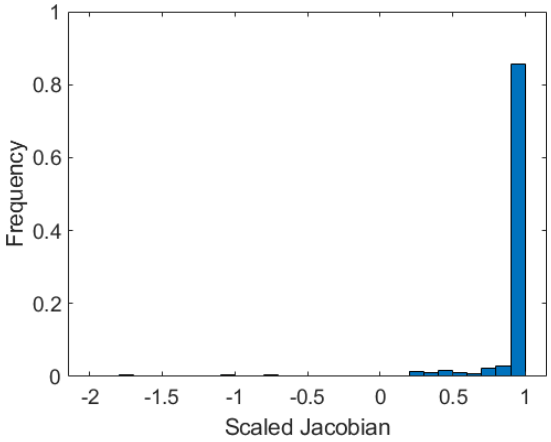
(c)



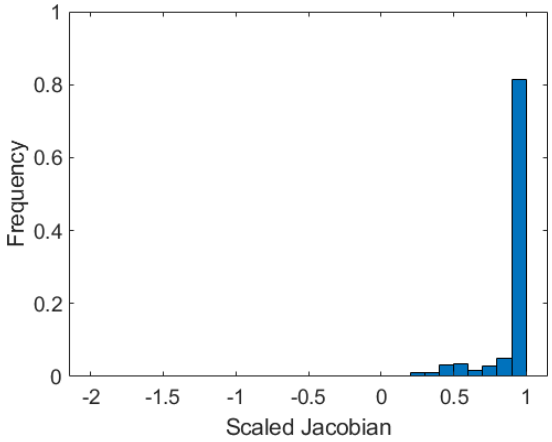
(d)

	Distortion	
Example	**Min**	**Max**
original mesh	-1.730	1.000
resulting mesh	0.211	1.000

(e)

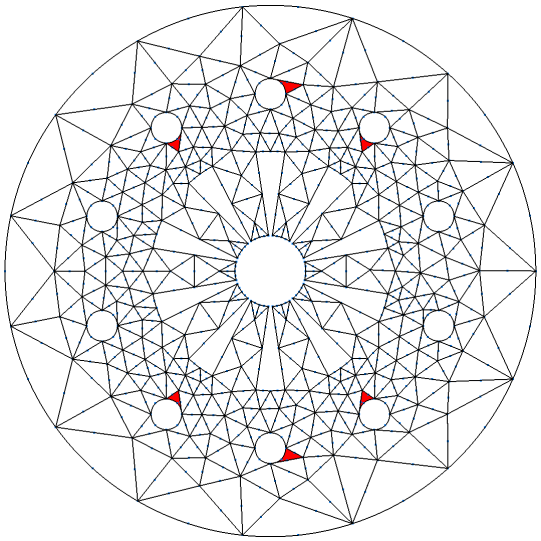


(f)

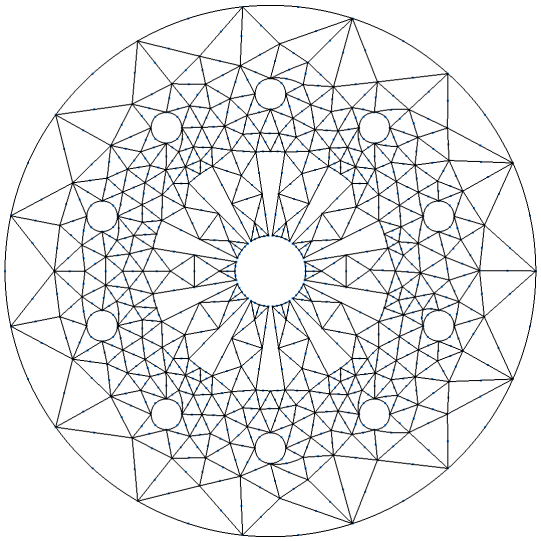


(g)

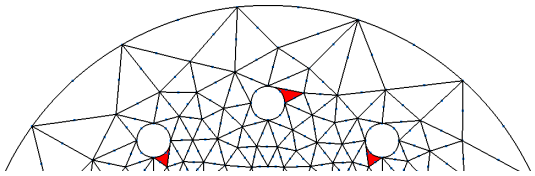
Figure 8: Bike gear example: (a) the tangled second-order mesh with fourteen tangled elements; (b) the mesh resulting from our method; (c,d) detailed views of (a,b), respectively; (e) the minimum and maximum element distortion, and (f,g) histogram plots of the distortion metric for (a,b), respectively.



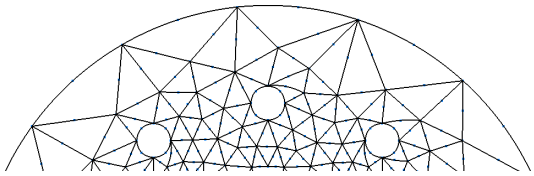
(a)



(b)



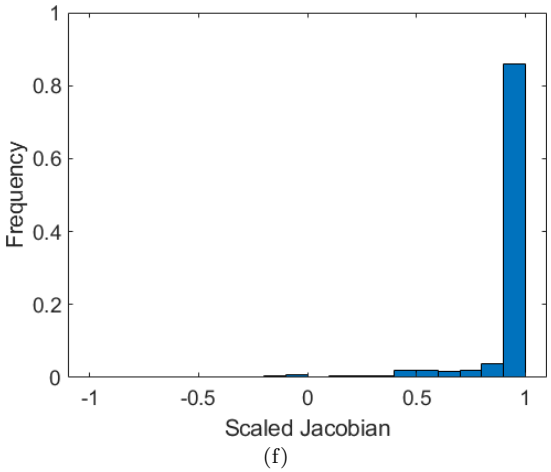
(c)



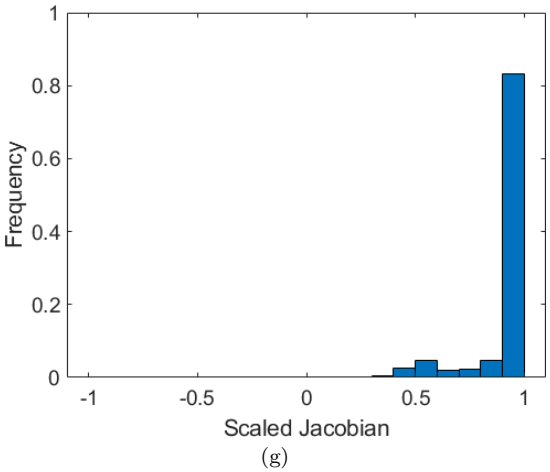
(d)

	Distortion	
Example	**Min**	**Max**
original mesh	-0.178	1.000
resulting mesh	0.345	1.000

(e)

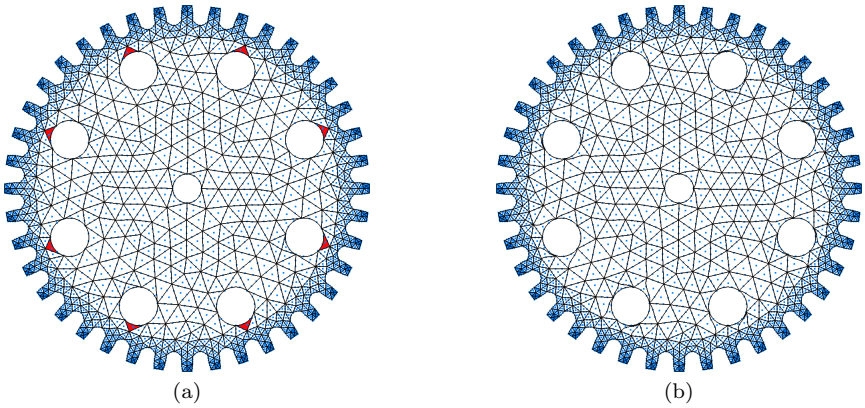


(f)

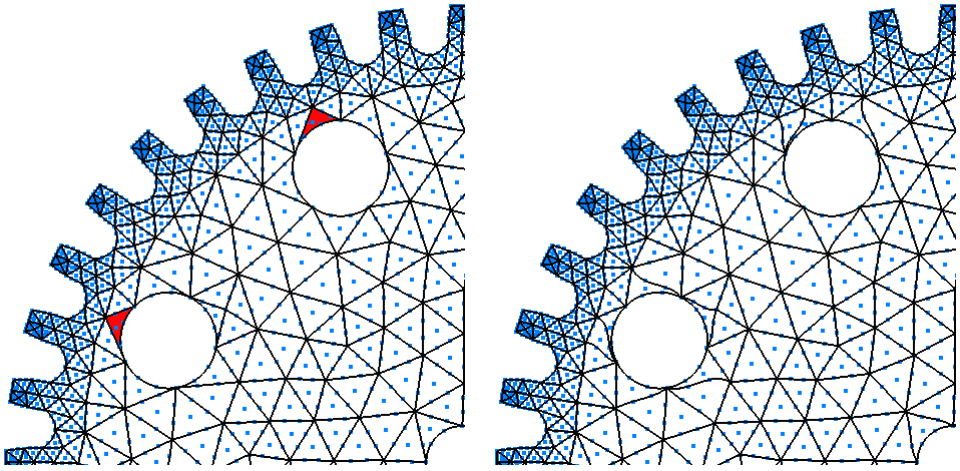


(g)

Figure 9: Pressure plate example: (a) the tangled second-order mesh; (b) the mesh resulting from our method; (c,d) detailed views of (a,b), respectively; (e) the minimum and maximum element distortion, and (f,g) histogram plots of the distortion metric for each mesh.



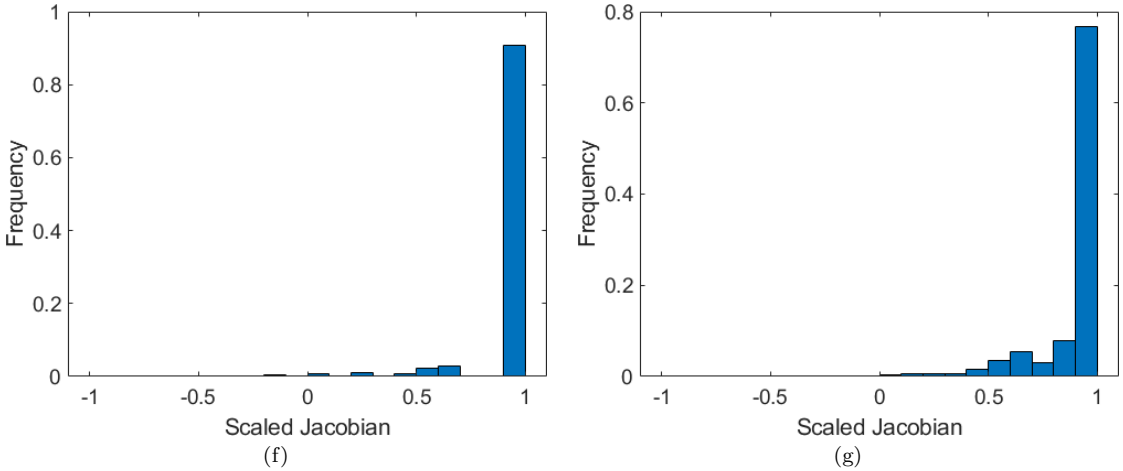
(a) (b)



(c) (d)

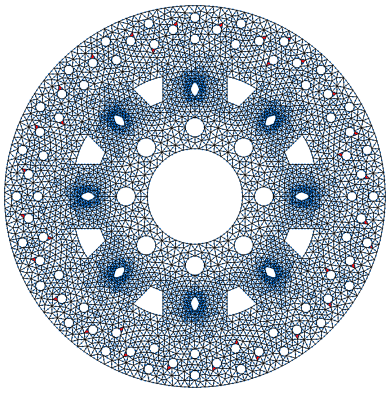
	Distortion	
Example	**Min**	**Max**
original mesh	-0.122	1.000
resulting mesh	0.092	1.000

(e)

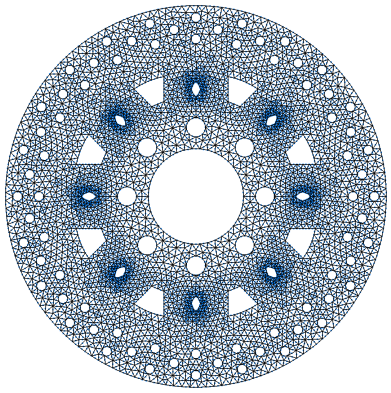


(f) (g)

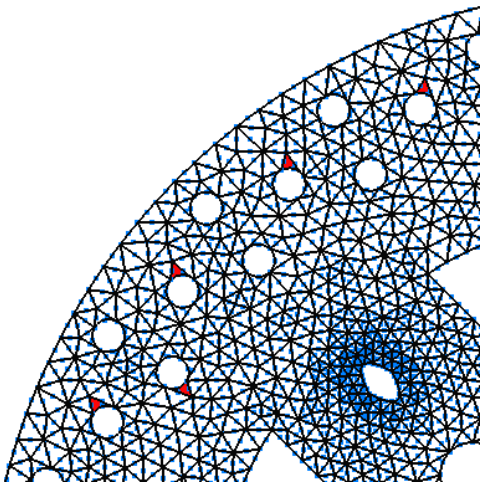
Figure 10: Gear example: (a) the initial third-order mesh with eight tangled elements; (b) the mesh resulting from our method; (c,d) detailed views of (a,b), respectively; (e) the minimum and maximum element distortion, and (f,g) histogram plots of the distortion metric for (a,b), respectively.



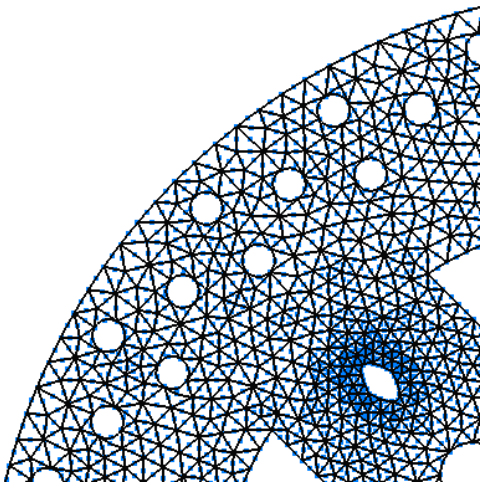
(a)



(b)



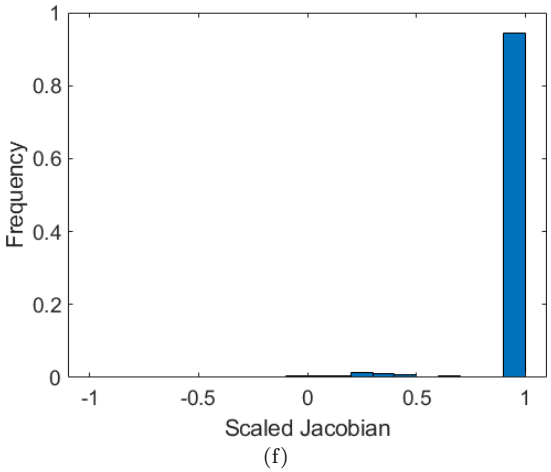
(c)



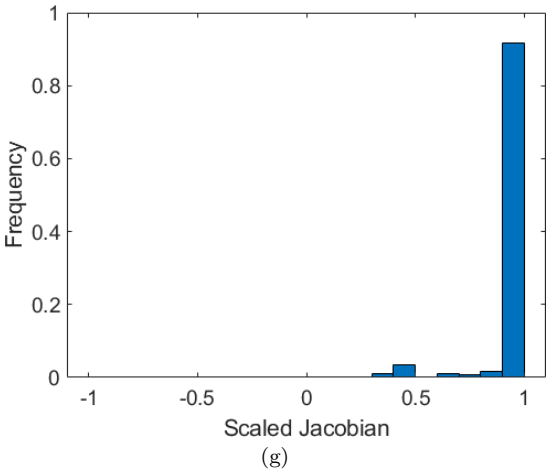
(d)

	Distortion	
Example	**Min**	**Max**
original mesh	-0.156	1.000
resulting mesh	0.346	1.000

(e)

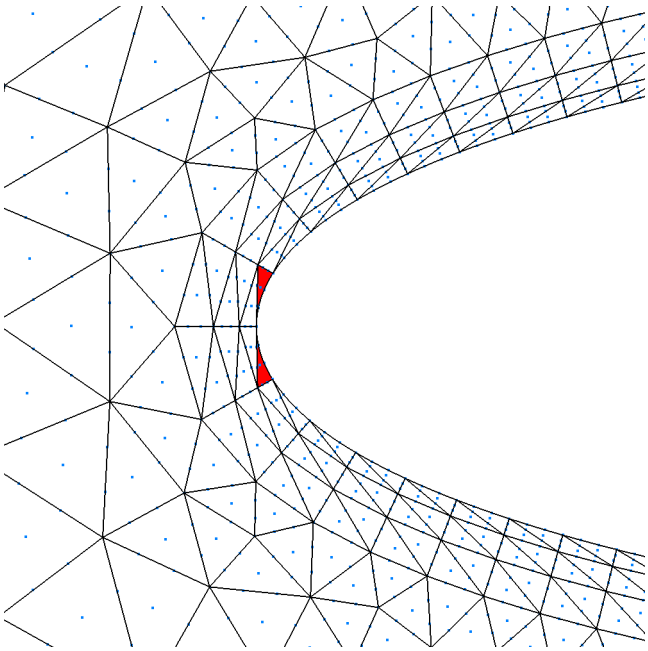


(f)

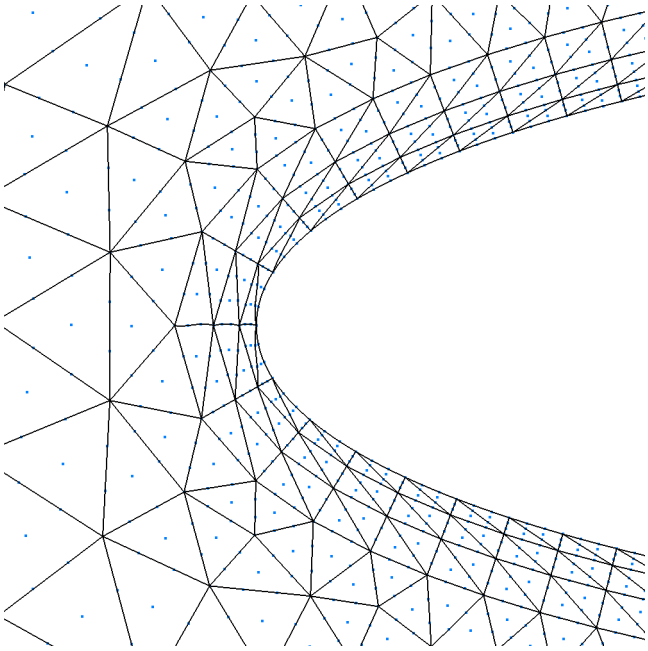


(g)

Figure 11: Brake rotor example: (a) the initial second-order mesh with thirty-four tangled elements; (b) the mesh resulting from our method; (c,d) detailed views of (a,b), respectively; (e) the minimum and maximum element distortion, and (f,g) histogram plots of the element distortion metric for (a,b), respectively.



(a)



(b)

	Distortion	
Example	**Min**	**Max**
original mesh	-0.109	1.000
resulting mesh	0.053	1.000

(c)

Figure 12: Airfoil example: (a) the initial third-order mesh with two tangled elements near the leading edge; (b) the mesh resulting from our method, and (c) the minimum and maximum element distortion.

					Runtime (s)	
Example	**Number of Elements**	**Mesh Order**	β	**Number of Iterations**	**First Pass**	**Second Pass**
annulus	30	3	0.500	2	0.005	0.000
mechanical part	295	2	0.500	2	0.041	—
bike gear	672	2	0.035	5	0.321	—
pressure plate	529	2	0.350	5	0.249	—
gear	1340	3	0.950	8	11.323	0.005
brake rotor	7015	2	0.850	2	19.643	—
airfoil	5328	3	0.001	2	24.039	0.005

Table 1: The number of elements, mesh order, beta value, number of outer iterations, and the wall clock times for each pass our untangling method (excluding I/O) for each example. Since the second-order meshes do not utilize the second pass, the columns are marked with a dash.

in the Gmsh repository. The mesh contains 5328 elements, with two tangled elements along the leading edge. In Fig. 12(a,b) we show the initial tangled mesh and the final mesh produced by our untangling algorithm. The minimum and maximum mesh element distortion values are listed in Fig. 12(c). The histogram plots for this example were omitted because there was minimal distinction between the two plots given the small percentage of tangled elements. After applying our method, the minimum distortion value increased from -0.109 to 0.053.

As we illustrated in Fig. 6, there are usually several choices for the parameter β that will result in an untangled mesh. As shown in Table 1, the trend we have observed thus far is that smaller values of β perform better for meshes with stretched elements near the curved features like our examples in Figs. 8 and 12. In particular, a smaller value of β was critical to maintaining the boundary layers in Fig. 12. Further experiments are necessary to determine what other factors influence the optimal value for β. The goal of these test cases was to explore the types of tangling that occur as a result of small deformations (e.g., moving the new boundary nodes onto the curved boundary during the typical high-order mesh generation process). With that in mind, our examples demonstrate that our method successfully addresses the typical types of tangling seen in this scenario. In addition to untangling the invalid patches, our method tends to reduce the amount of element distortion in all of our examples. In addition, there is potential to improve the performance of our method using parallel computing, as our local method can be applied to non-adjacent patches simultaneously.

4. CONCLUDING REMARKS AND FUTURE WORK

We have presented a new optimization-based method for untangling second- and third-order triangular meshes. The two-dimensional examples have shown that our proposed method based on signed angles is able to successfully untangle a variety of invalid second- and third-order meshes. Furthermore, our method tends to dramatically decrease the amount of element distortion present in the mesh. As our final example in Section 3 showed, the addition of the weighting parameter gives the user increased flexibility in defining the behavior of the objective function. One limitation of our method is that it does not move the low-order nodes. To address this, we plan to combine our untangling algorithm with a weight-based scheme like the one proposed in [14]. By combining these two approaches, we could use the weight-based scheme to move the low-order nodes, and the method we have proposed in this paper to move the high-order nodes.

Our future work will include exploring techniques for determining the ideal value of the weighting parameter β. We will also extend our implementation to untangle meshes composed of elements with $p > 3$. In addition, we plan to extend the capabilities of our method to three dimensions by using signed solid angles between curved faces of high-order tetrahedral elements. We also plan to add support for additional element types (e.g., quadrilaterals, etc). Finally, we plan to explore examples with larger deformations that result in more complicated mesh tangling.

5. ACKNOWLEDGMENTS

The work of the first author was funded by NSF CCF grant 1717894. The work of the second author was funded by an REU supplement to NSF CCF grant 1717894. The work of the third author was supported in part by NSF grants CCF 1717894 and OAC 1808553.

References

[1] Wang Z.J., Fidkowski K., Abgrall R., Bassi F., Caraeni D., Cary A., Deconinck H., Hartmann R., Hillewaert K., Huynh H.T., et al. "High-order CFD methods: Current status and perspective." *International Journal for Numerical Methods in Fluids*, vol. 72, no. 8, 811–845, 2013

[2] Bassi F., Rebay S. "High-order accurate discontinuous finite element solution of the 2D Euler equations." *Journal of Computational Physics*, vol. 138, no. 2, 251 – 285, 1997

[3] Luo X., Shephard M.S., Remacle J.F. "The influence of geometric approximation on the accuracy of high order methods." *Rensselaer SCOREC report*, vol. 1, 2001

[4] Dey S., Shephard M.S. "Curvilinear mesh generation in 3D." *Proceedings of the 8th International Meshing Roundtable*. 1999

[5] Fortunato M., Persson P.O. "High-order unstructured curved mesh generation using the Winslow equations." *Journal of Computational Physics*, vol. 307, no. 2016, 1–14, Feb. 2016

[6] Gargallo-Peiró A., Roca X., Peraire J., Sarrate J. "Distortion and quality measures for validating and generating high-order tetrahedral meshes." *Engineering with Computers*, vol. 31, no. 3, 423–437, 2015

[7] Gargallo-Peiró A., Roca X., Peraire J., Sarrate J. "Optimization of a regularized distortion measure to generate curved high-order unstructured tetrahedral meshes." *International Journal for Numerical Methods in Engineering*, vol. 103, no. 5, 342–363, 2015

[8] Moxey D., Ekelschot D., Keskin Ü., Sherwin S.J., Peiró J. "High-order curvilinear meshing using a thermo-elastic analogy." *Computer-Aided Design*, vol. 72, 130–139, 2016

[9] Moxey D., Green M., Sherwin S., Peiró J. "An isoparametric approach to high-order curvilinear boundary-layer meshing." *Computer Methods in Applied Mechanics and Engineering*, vol. 283, 636 – 650, 2015

[10] Persson P.O., Peraire J. "Curved mesh generation and mesh refinement using Lagrangian solid mechanics." *Proceedings of the 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, p. 949. 2009

[11] Roca X., Gargallo-Peiró A., Sarrate J. "Defining quality measures for high-order planar triangles and curved mesh generation." *Proceedings of the 20th International Meshing Roundtable*, pp. 365–383. Springer Berlin Heidelberg, 2012

[12] Ruiz-Gironés E., Sarrate J., Roca X. "Generation of curved high-order meshes with optimal quality and geometric accuracy." *Proceedings of the 25th International Meshing Roundtable*, vol. 163, pp. 315–327. Procedia Engineering, 2016

[13] George P.L., Borouchaki H. "Construction of tetrahedral meshes of degree two." *International Journal for Numerical Methods in Engineering*, vol. 90, no. 9, 1156–1182, 2012

[14] Stees M., Shontz S.M. "A high-order log barrier-based mesh generation and warping method." *Proceedings of the 26th International Meshing Roundtable*, vol. 203, pp. 180 – 192. Procedia Engineering, 2017

[15] Stees M., Shontz S.M. "An angular approach to untangling high-order curvilinear triangular meshes." *Proceedings of the 27th International Meshing Roundtable*, pp. 327–342. Springer, 2019

[16] Karman S.L., Erwin J.T., Glasby R.S., Stefanski D. "High-order mesh curving using WCN mesh optimization." *46th AIAA Fluid Dynamics Conference*, p. 3178. 2016

[17] Toulorge T., Geuzaine C., Remacle J.F., Lambrechts J. "Robust untangling of curvilinear meshes." *Journal of Computational Physics*, vol. 254, 8 – 26, 2013

[18] Turner M., Moxey D., Peiró J., Gammon M., Pollard C.R., Bucklow H. "A framework for the generation of high-order curvilinear hybrid meshes for CFD simulations." *Proceedings of the 26th International Meshing Roundtable*, vol. 203, pp. 206 – 218. Procedia Engineering, 2017

[19] Xie Z.Q., Sevilla R., Hassan O., Morgan K. "The generation of arbitrary order curved meshes for 3D finite element analysis." *Computational Mechanics*, vol. 51, no. 3, 361–374, 2013

[20] Sherwin S.J., Peiró J. "Mesh generation in curvilinear domains using high-order elements." *International Journal for Numerical Methods in Engineering*, vol. 53, no. 1, 207–223, 2001

[21] Dobrev V., Knupp P., Kolev T., Mittal K., Tomov V. "The Target-Matrix Optimization Paradigm for high-order meshes." *SIAM Journal on Scientific Computing*, vol. 41, no. 1, B50–B68, 2019

[22] Lagarias J.C., Reeds J.A., Wright M.H., Wright P.E. "Convergence properties of the Nelder–Mead simplex method in low dimensions." *SIAM Journal on Optimization*, vol. 9, no. 1, 112–147, 1998

[23] Geuzaine C., Remacle J.F. "Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities." *International Journal for Numerical Methods in Engineering*, vol. 79, no. 11, 1309–1331, 2009

[24] Johnen A., Remacle J.F., Geuzaine C. "Geometrical validity of high-order triangular finite elements." *Engineering with Computers*, vol. 30, no. 3, 375–382, 2014

[25] Remacle J.F., Chevaugeon N., Marchandise E., Geuzaine C. "Efficient visualization of high-order finite elements." *International Journal for Numerical Methods in Engineering*, vol. 69, no. 4, 750–771, 2007