An angular approach to untangling high-order curvilinear triangular meshes

Mike Stees and Suzanne M. Shontz

Abstract To achieve the full potential of high-order numerical methods for solving partial differential equations, the generation of a high-order mesh is required. One particular challenge in the generation of high-order meshes is avoiding invalid (tangled) elements that can occur as a result of moving the nodes from the low-order mesh that lie along the boundary to conform to the true curved boundary. In this paper, we propose a heuristic for correcting tangled second- and third-order meshes. For each interior edge, our method minimizes an objective function based on the unsigned angles of the pair of triangles that share the edge. 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.

1 Introduction

The appeal of high-order methods for solving partial differential equations lies in their ability to achieve higher accuracy at a lower cost than low-order methods. One challenge in the adoption of these high-order methods for problems with curved geometries is the lack of robust high-order mesh generation software [19]. More specifically, to fully leverage the accuracy of high-order methods in the presence of curved geometries, such methods need to be paired with a high-order mesh that correctly reflects the curvature of the geometry, as demonstrated in [1, 8].

Mike Stees
Department of Electrical Engineering and Computer Science, Information and Telecommunication Technology Center, University of Kansas, 1520 W. 15th Street, Lawrence, KS, 66045, USA, e-mail: mstees@ku.edu

Suzanne M. Shontz
Department of Electrical Engineering and Computer Science, Bioengineering Graduate Program, Information and Telecommunication Technology Center, University of Kansas, 1520 W. 15th Street, Lawrence, KS, 66045, USA, e-mail: shontz@ku.edu

The most common approach for high-order mesh generation methods is to transform a coarse linear mesh [2], [3–5], [9, 10], [6, 12–14],and [16–18, 20]. The main challenge of the transformation is obtaining a valid high-order mesh. In general, these methods involve three steps: (1) adding additional nodes to the linear mesh; (2) moving the newly added boundary nodes to conform with the curved geometry, and (3) moving the interior nodes. There are two categories of methods which are especially popular for transforming the initial mesh. The first category involves transforming the mesh based on optimization of an objective function [2], [4, 5], [13, 14], and [15–17]. Several of the objective functions proposed in this category include a measure of element validity, which allows them to untangle invalid elements [2,4,5,13,14,17]. While they do not guarantee successful untangling, many of them are robust. The second category of methods transform the mesh based on the solution to a partial differential equation [3,9,12,20].

In this paper, we describe an optimization-based approach for untangling invalid second- and third-order meshes. The primary goal of this work is to untangle invalid meshes that result from deforming the newly added boundary nodes to conform with the true boundary. Toward that end, we demonstrate our method on several meshes composed of second- and third-order elements that became invalid following the projection of the boundary nodes onto the true boundary. We also explore the untangling of meshes that became invalid as a result of small deformations. The remainder of this paper is organized as follows. In Section 2, we present our new method for high-order mesh untangling. In Section 3, we illustrate the performance of our method on several examples. Finally, in Section 4, we offer concluding remarks and discuss some possibilities for future work.

2 Untangling high-order curvilinear meshes

In this section, we propose a local edge-based optimization method for untangling high-order curvilinear meshes based on the unsigned angles of curvilinear triangles. For each interior mesh edge, we identify the two triangles that share the edge and compute the distortion of each of the two triangles. For each pair of triangles with a minimum distortion measure less than 0, we solve the following unconstrained optimization problem:

$$x^* = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{i=1}^{4} \alpha_{\mathbf{i}}(\mathbf{x}), \tag{1}$$

where

$\alpha_{\mathbf{i}}$ = the i^{th} entry of the vector of the four unsigned angles,

$\mathbf{x}$ = the nodal positions of the high-order nodes that lie on the edge

In Fig. 1 we give an example which shows an interior edge in red and the pair of triangles that share that edge in green. We also label the four unsigned angles that are calculated, the nodes that are allowed to move during the optimization (black diamonds), and the nodes that are fixed (black dots).

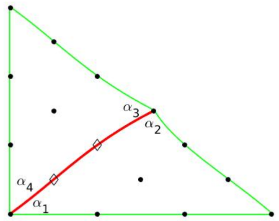
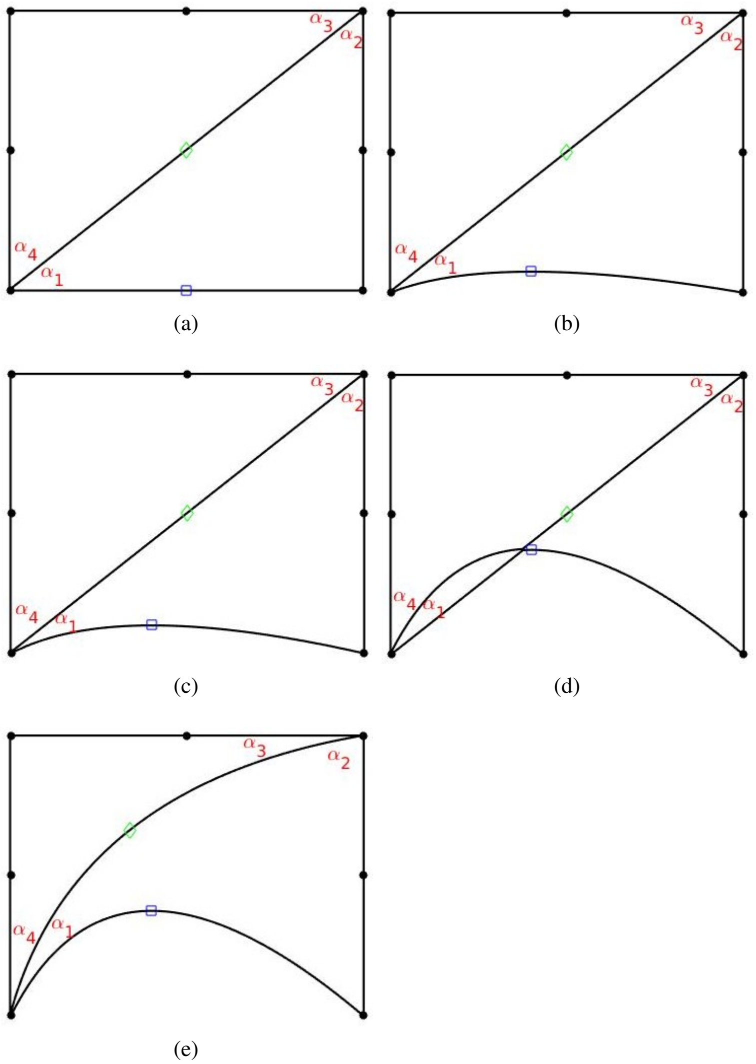


Fig. 1: A pair of triangles showing the interior edge (red), the free nodes (black diamonds), the fixed nodes (black dots), and the four angles α_i

To better understand the behavior of the objective function, consider the five examples shown in Fig. 2. In Fig. 2(a) for $\alpha_1 + \alpha_4$ and $\alpha_2 + \alpha_3$, moving the free node (green diamond) will shift the proportion of each term, while leaving the overall sum fixed. In other words, increasing α_1 will cause a corresponding decrease in α_4 while the quantity $\alpha_1 + \alpha_4$ remains the same. Similarly, increasing α_2 will cause a corresponding decrease in α_3 while the quantity $\alpha_2 + \alpha_3$ remains the same. This behavior means that the sum of all four angles cannot be further decreased by moving the free node. Furthermore, this behavior is desirable because patches with no distortion will not be modified since the optimization will not move the free node (as there is no step that will lead to a decrease in the objective function). Fortunately, this behavior holds true as we add minor distortion as well. In Fig. 2(b), we moved the bottom node (denoted by a blue square) to increase the distortion of the bottom element. In Fig. 2(c), we moved the node slightly further to increase distortion. In both cases, we can see that the overall sum cannot be further decreased by moving the free node. Finally, in Fig. 2(d), we move the node to the point that it causes tangling. Now that α_1 is an angle between tangled edges, this angle can be decreased by moving the free node. By decreasing the value of α_1, we decrease the value of $\alpha_1 + \alpha_4$, and thus decrease the overall sum of the four angles. In other words, minimizing our objective function attempts to decrease the value of angles that occur between tangled edges by moving the free node away. In Fig. 2(e), we show results of moving the free node to minimize our objective function.

To measure distortion, we use the scaled Jacobian [2]. To solve our unconstrained optimization problem, we use the Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton method decribed in Chapter 6 of [11]. In place of the analytical gradient, we use a 6^{th} order centered finite difference with a step size of 10^{-6}. As our initial



(a) (b)

(c) (d)

(e)

Fig. 2: A simple patch showing the angles (α_i), the free node (green diamond), and the node that is moved to increase distortion (blue square). In (a), the patch with no distortion is shown. In (b-d), the amount of distortion is gradually increased. In (e), the mesh after applying our method to minimize the sum of the angles is shown.

Hessian approximation, we use a scaled version of the identity matrix. In Alg. 1 and Alg. 2, we give pseudocode descriptions of our untangling method and optimization method, respectively. Our implementation of the BFGS quasi-Newton method uses a backtracking line search. This backtracking approach based on the Wolfe conditions ensures that the step results in a sufficient decrease in our objective function. In the next section, we discuss how the angles $\alpha_i(x)$ of the curved elements are calculated.

Algorithm 1 Pseudocode for our edge-based mesh untangling method

```
while there are tangled elements or passes $<$ count do
  for each interior edge $e$ do
    Find the two triangles $t_1$ and $t_2$ with $e$ as a common edge
    Compute the element distortions $ed_1$ and $ed_2$ for $t_1$ and $t_2$, respectively
    if $\min(ed_1, ed_2) < 0$ then
      Solve Eq. 1 for $x^*$ using Alg. 2
      Update nodal positions of the free nodes on $e$ to $x^*$
    end if
  end for
  passes = passes + 1
end while
```

Algorithm 2 Pseudocode for our BFGS quasi-Newton method

```
Given an initial value $x_0$, an initial value for the Hessian $B_0$, and a tolerance $tol$;
while $\|\nabla f(x_k) > tol\|$ do
  Compute Cholesky factorization $B_k = LL^T$
  Compute the direction vector $d_k$ by solving $LL^T d_k = -\nabla f(x_k)$.
  $\rho_k = 1.0$
  while $f(x_k + \rho_k d_k) > f(x_k) + 10^{-4} \rho_k \nabla f(x_k)^T d_k$ do
    $\rho_k = 0.5\rho_k$
  end while
  $x_{k+1} = x_k + \rho_k d_k$
  $s_k = x_{k+1} - x_k$
  $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$
  $B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$
  $k = k + 1$
end while
```

2.1 Measuring the angles of curvilinear triangles

In order to compute the angle between two curves at a given point, we compute the derivatives of the curves, evaluate the derivatives at the given point, and then compute the angle between the resulting tangent vectors. Following this approach, we will compute the angles between each pair of edges of curvilinear triangles. For our derivation, we use the third-order Lagrange elements. Derivation for the second-order Lagrange elements is similar.

Consider the third-order Lagrange triangle shown in Fig.3 with shape functions defined as follows:

$$\begin{aligned}
s_1 &= \frac{9}{2}(1-\xi-\eta)\left(\frac{1}{3}-\xi-\eta\right)\left(\frac{2}{3}-\xi-\eta\right)\\
s_2 &= \frac{9}{2}\xi\left(\xi-\frac{1}{3}\right)\left(\xi-\frac{2}{3}\right)\\
s_3 &= \frac{9}{2}\eta\left(\eta-\frac{1}{3}\right)\left(\eta-\frac{2}{3}\right)\\
s_4 &= \frac{27}{2}(1-\xi-\eta)\xi\left(\frac{2}{3}-\xi-\eta\right)\\
s_5 &= \frac{27}{2}(1-\xi-\eta)\xi\left(\xi-\frac{1}{3}\right)\\
s_6 &= \frac{27}{2}\xi\eta\left(\xi-\frac{1}{3}\right)\\
s_7 &= \frac{27}{2}\xi\eta\left(\eta-\frac{1}{3}\right)\\
s_8 &= \frac{27}{2}(1-\xi-\eta)\eta\left(\eta-\frac{1}{3}\right)\\
s_9 &= \frac{27}{2}(1-\xi-\eta)\eta\left(\frac{2}{3}-\xi-\eta\right)\\
s_{10} &= 27\xi\eta(1-\xi-\eta).
\end{aligned}$$

The mapping $\phi(\xi,\eta)$ from the reference unit element in Fig.1 onto the physical element is then given by:

$$\phi(\xi,\eta) = \sum_{i=1}^{10} \mathbf{x_i}\, s_i(\xi,\eta), \tag{2}$$

where $\mathbf{x_i}$ are the nodal positions, and (ξ,η) is a point in the reference element. Since we are concerned with the angles between each pair of edges, we need to define mappings from each point on the edges of the reference element to the corresponding point on the edges of the physical element. The edges correspond to third-order Lagrange elements in 1D. The shape functions associated with these elements are

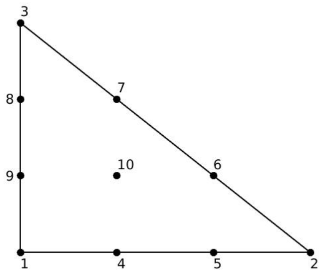


Fig. 3: Third-order Lagrange reference unit triangle

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:

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

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

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

The notation f_{ij} denotes the edge between nodes i and j in Fig. 1. In their expanded forms, each $\mathbf{f_{ij}}(\mathbf{t})$ is a cubic polynomial in the variable t. Next, we need to compute the derivatives of our functions. Straightforward differentiation 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. 1. To calculate the angle in radians, we use the following formula:

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

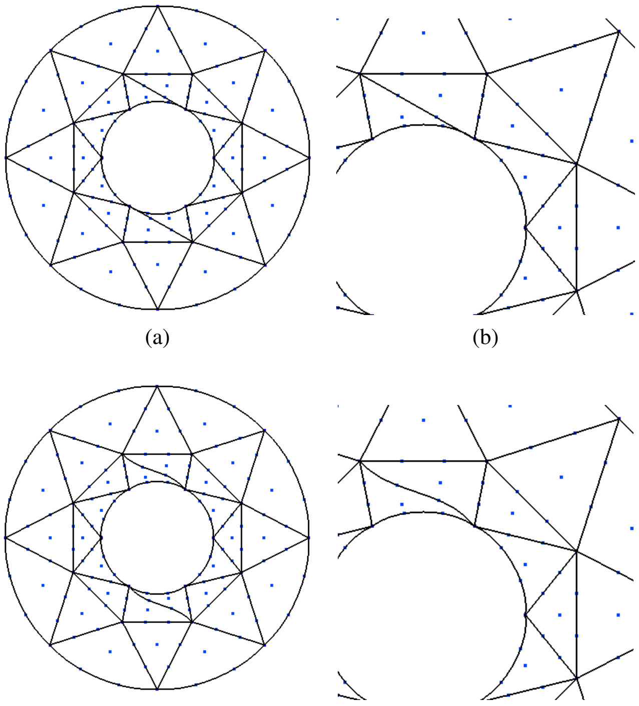
Returning to the calculation of $\alpha_{\mathbf{i}}(\mathbf{x})$ in Eq. 1, we loop over each triangle in the patch and calculate the two angles of each triangle formed by the edges incident to the shared edge between the triangles as described above.

3 Numerical experiments

In this section, we show the results from performing several numerical experiments to untangle invalid second- and third-order meshes. For each example, we show the initial meshes; the meshes which result after untangling them with our method; the minimum distortion, maximum distortion, average distortion computed over all elements (referred to as Avg1 in figures), and average distortion computed over curved elements (referred to as Avg2 in figures), and the run time needed for our method to untangle the mesh. For each mesh, we show the nodes associated with the element of the given order. We do not show the location of the quadrature points. The code was run using Matlab R2017b, and the execution times were measured on a machine with 8GB of RAM and an Intel Xeon(R) W3520 CPU. All mesh visualizations and distortion calculations were done using Gmsh [7].

Our first example is a third-order annulus composed of 30 elements. During the process of curving the boundary, tangled elements were created near the top and bottom of the inner ring. Figure 4(a,c) show the initial invalid mesh and the final mesh resulting from our method, respectively. Figure 4(b,d), show detailed views of the inner ring from Fig. 4(a,c), respectively. In Fig. 4(e) we give the mesh element distortion.

Our second example is the leading edge of a third-order NACA0012 airfoil. In Fig. 5(a), we can see that curving the inner boundary resulted in two tangled elements near the leading edge of the airfoil. In Fig. 5(b), we show the final mesh resulting from our method. Finally, Fig. 5(c) gives the mesh element distortion.



(a)

(b)

(c)

(d)

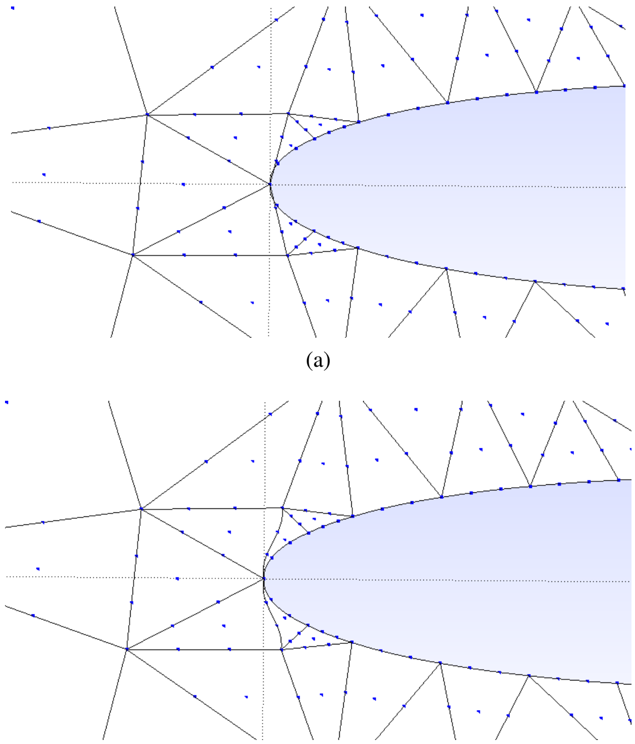
	Distortion			
Example	**Min**	**Max**	**Avg1**	**Avg2**
original mesh	-0.223	1.0000	0.615	0.279
resulting mesh	0.039	1.0000	0.595	0.325

(e)

Fig. 4: Annulus example: (a) the tangled third-order mesh; (b) a detailed view of one tangled element along the top of the inner boundary; (c) the mesh resulting from our method; (d) a detailed view of the untangled element from (c), and (e) the mesh quality as measured by the element distortion metric.

Our third example is a second-order mesh of a mechanical part with several holes. Figure 6(a-c) shows the initial invalid mesh, the final mesh resulting from our method, and the mesh quality as measured by the distortion metric. In Fig. 6(a), we can see that curving the boundaries resulted in tangled elements near the top and bottom holes.

Our fourth and fifth examples are valid meshes of a square plate with a circular hole. To induce mesh tangling in the fourth example, we applied a rotation of 10 degrees counterclockwise to the inner ring followed by a horizontal shear with a shear factor of 0.5. In Fig. 7(a,b,d), we show the initial valid mesh, the tangled mesh



(a)

(b)

Example	Distortion			
	Min	**Max**	**Avg1**	**Avg2**
original mesh	-0.566	1.000	0.968	0.967
resulting mesh	0.050	1.000	0.970	0.970

(c)

Fig. 5: Airfoil example: (a) the tangled third-order mesh; (b) the mesh resulting from our method, and (c) the mesh quality as measured by the element distortion metric.

resulting from rotation and shearing, and the final untangled mesh resulting from our method. In Fig. 7(c,e), we show detailed views of the inner ring. In Fig. 7(f), we give the element distortion for the initial, tangled, and final meshes, respectively. In the fifth example, we applied a rotation of 10 degrees counterclockwise to the inner ring followed by a stretching of the bottom half of the plate. In Fig. 8(a,b,c), we show the initial valid mesh, the tangled mesh resulting from rotation and stretching, and the final untangled mesh resulting from our method. In Fig. 8(d), we give the element distortion for each mesh. In our final example, we show a valid mesh of a two-dimensional beam. To create mesh tangling, we treated the beam as a simply supported beam and applied a center load. After applying the load, we translated the left and right sides of the beam. In Fig. 9(a,b,c), we show the initial valid mesh, the tangled mesh resulting from our transformations, and the final untangled mesh resulting from our method. In Fig. 9(d,e), we show detailed views of the left side of the beam from Fig. 9(b,c). Finally in Fig. 9(f), we give the mesh element distortion.

Example	Number of Elements	Wall Clock Time(s)
annulus	30	2.56
airfoil	282	16.63
mechanical part	182	2.44
plate 1	597	7.40
plate 2	597	7.48
beam	542	6.73

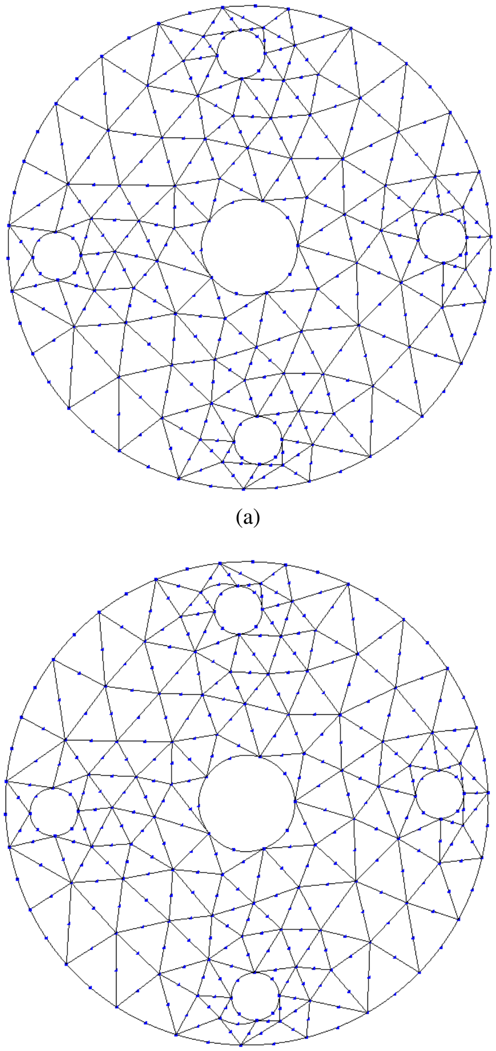
Table 1: The number of elements and the wall clock time for each mesh

While the test cases are relatively straightforward, our goal was to explore the types of tangling that occur as a result of moving the new boundary nodes onto the curved boundary during the typical high-order mesh generation process. We were also interested in tangling that might result from small deformations to a valid mesh. With these points in mind, the examples demonstrate that our method is able to handle the small deformations that might result in tangling for second- and third-order meshes. Additionally, our method only required a single pass for each of the test cases. We demonstrate the runtime performance of our method in Tab. 1. We list the number of elements and wall clock time for each of our numerical examples in Tab. 1. While these times are reasonable, for large meshes, faster run times will be required. Fortunately, there is high potential for improved performance using parallel computing, as our method can be applied to non-adjacent patches simultaneously.

4 Conclusions

We have presented a new optimization-based method for untangling the edges of second- and third-order meshes. The two-dimensional examples have shown that our proposed method based on the unsigned angles of curvilinear triangles is able to successfully untangle several invalid second- and third-order meshes.

We note that presently our method has a few limitations. The first limitation is that it only allows movement of the high-order nodes that lie on the interior edge (e.g. the free nodes show in Fig. 1). That is, it does not allow movement of the endpoints. The second limitation is that non-edge nodes (e.g. like node 10 in Fig. 3) are not moved at all. The final limitation is that our objective function does not measure element validity. Due to these limitations, our method does not guarantee that it will successfully untangle a given tangled patch. With these limitations in



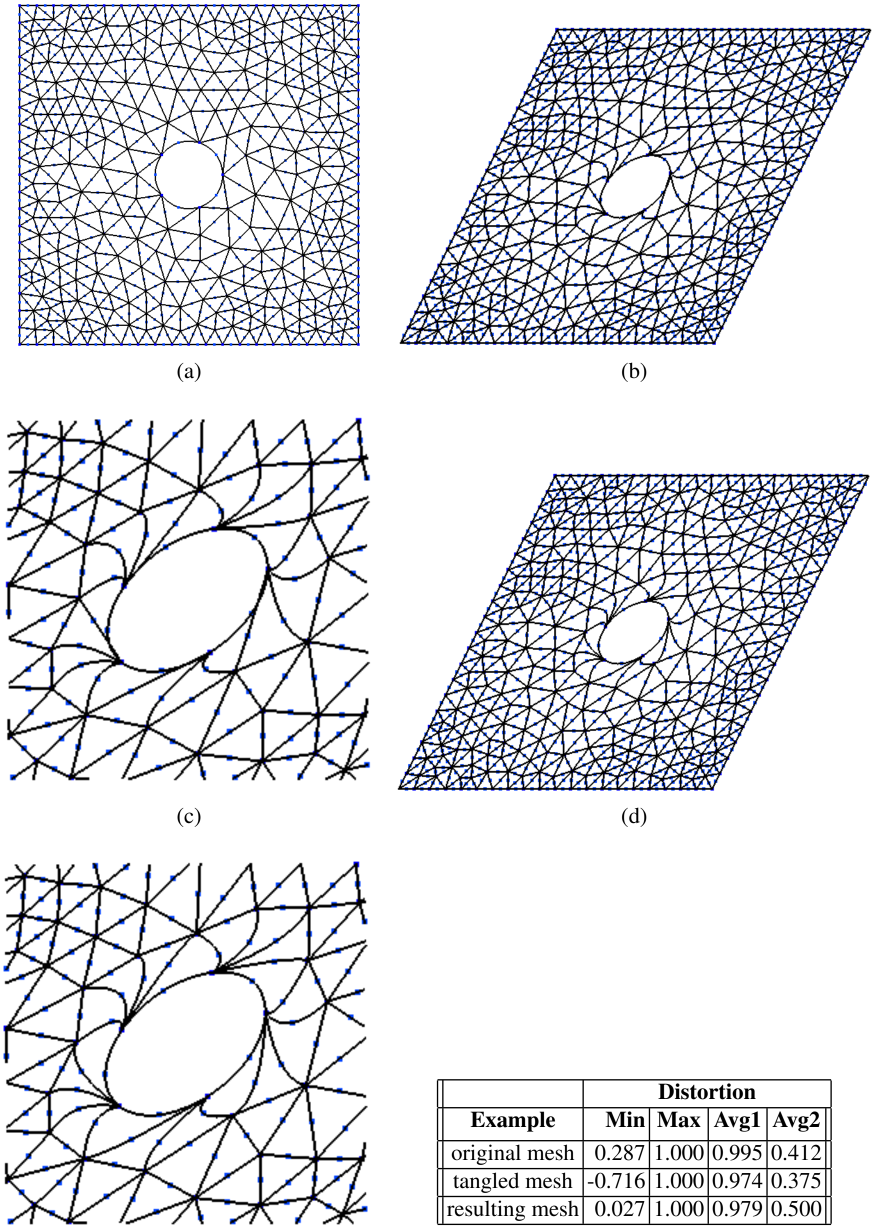
(a)

(b)

	Distortion			
Example	**Min**	**Max**	**Avg1**	**Avg2**
original mesh	-0.049	1.000	0.904	0.601
resulting mesh	0.008	1.000	0.905	0.625

(c)

Fig. 6: Mechanical part example: (a) the tangled second-order mesh; (b) the mesh resulting from our method, and (c) the mesh quality as measured by the element distortion metric.



(a)

(b)

(c)

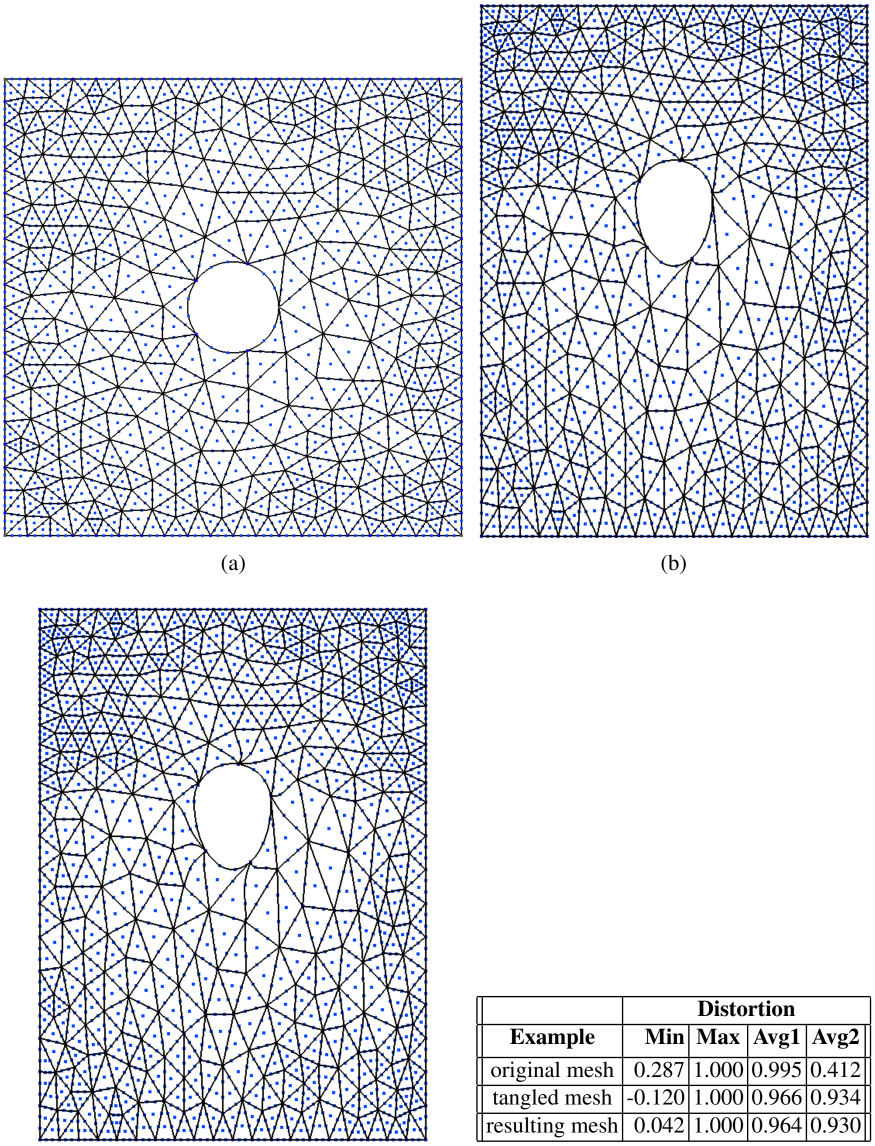
(d)

(e)

	Distortion			
Example	**Min**	**Max**	**Avg1**	**Avg2**
original mesh	0.287	1.000	0.995	0.412
tangled mesh	-0.716	1.000	0.974	0.375
resulting mesh	0.027	1.000	0.979	0.500

(f)

Fig. 7: Square plate example: (a) the initial second-order mesh; (b) the mesh resulting from rotating the inner ring 10 degrees counterclockwise and applying a horizontal shear with a shear factor of 0.5; (c) a detailed view of the elements along the inner ring; (d) the mesh resulting from applying our method, (e) a detailed view of the elements along the inner ring; and (f) the mesh quality as measured by the element distortion metric.



(a)

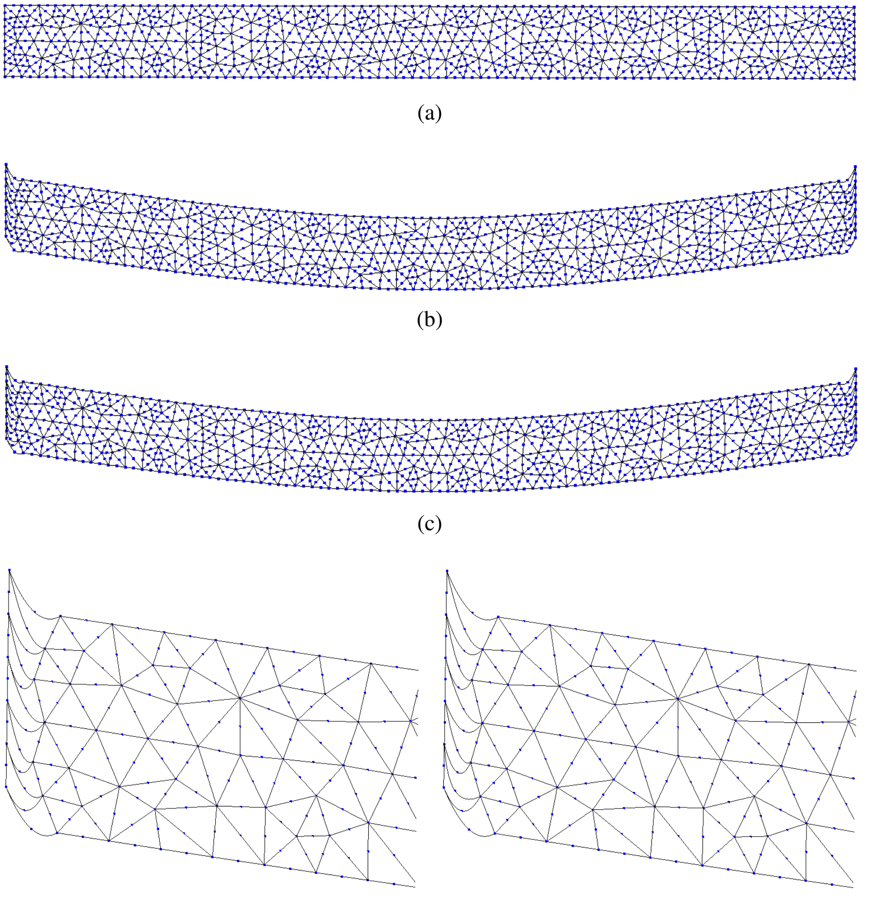
(b)

(c)

	Distortion			
Example	**Min**	**Max**	**Avg1**	**Avg2**
original mesh	0.287	1.000	0.995	0.412
tangled mesh	-0.120	1.000	0.966	0.934
resulting mesh	0.042	1.000	0.964	0.930

(d)

Fig. 8: Square plate example: (a) the initial third-order mesh; (b) the mesh resulting from rotating the inner ring 10 degrees counterclockwise and stretching the bottom half of the plate; (c) the mesh resulting from applying our method, (d) the mesh quality as measured by the element distortion metric.



(a)

(b)

(c)

(d)

(e)

	Distortion			
Example	**Min**	**Max**	**Avg1**	**Avg2**
original mesh	1.000	1.000	1.000	1.000
tangled mesh	-0.044	1.000	0.982	0.975
resulting mesh	0.032	1.000	0.980	0.973

(f)

Fig. 9: Beam example: (a) the initial second-order mesh; (b) the mesh after treating it as a simply supported beam with a center load and translating the left and right ends; (c) the mesh resulting from applying our method; (d) a detailed view of the left edge of (b); (e) a detailed view of the left edge of (c), and (f) the mesh quality as measured by the element distortion metric.

mind, our future work will include extending the capabilities of our method to include moving non-edge nodes, as well as allowing the endpoints of edges to move. We will also explore the use of signed angles, where a negative angle indicates that tangling is present. This would allow us to directly check element validity, but would likely require modification of the objective function to achieve the desired untangling behavior. Other future improvements include extending our approach to three dimensions by using the dihedral angles between curved faces of high-order tetrahedral elements, and extending our implementation to allow for elements with $p > 3$.

5 Acknowledgments

The work of the first author was funded in part by the Madison and Lila Self Graduate Fellowship and NSF CCF grant 1717894. The work of the second author was funded in part by NSF CCF grant 1717894.

References

1. F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2D euler equations. *Journal of Computational Physics*, 138(2):251 – 285, 1997.
2. S. Dey and M. S. Shephard. Curvilinear mesh generation in 3D. In *Proceedings of the 8th International Meshing Roundtable*, 1999.
3. M. Fortunato and P.-O. Persson. High-order unstructured curved mesh generation using the Winslow equations. *J. Comput. Phys.*, 307(2016):1–14, Feb. 2016.
4. A. Gargallo-Peiró, X. Roca, J. Peraire, and J. Sarrate. Distortion and quality measures for validating and generating high-order tetrahedral meshes. *Engineering with Computers*, 31(3):423–437, 2015.
5. A. Gargallo-Peiró, X. Roca, J. Peraire, and J. Sarrate. Optimization of a regularized distortion measure to generate curved high-order unstructured tetrahedral meshes. *International Journal for Numerical Methods in Engineering*, 103(5):342–363, 2015.
6. P. L. George and H. Borouchaki. Construction of tetrahedral meshes of degree two. *International Journal for Numerical Methods in Engineering*, 90(9):1156–1182, 2012.
7. C. Geuzaine and J.-F. Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities. *International journal for numerical methods in engineering*, 79(11):1309–1331, 2009.
8. X. Luo, M. S. Shephard, and J.-F. Remacle. The influence of geometric approximation on the accuracy of high order methods. *Rensselaer SCOREC report*, 1, 2001.
9. D. Moxey, D. Ekelschot, Ü. Keskin, S. J. Sherwin, and J. Peiró. High-order curvilinear meshing using a thermo-elastic analogy. *Computer-Aided Design*, 72:130–139, 2016.
10. D. Moxey, M. Green, S. Sherwin, and J. Peiró. An isoparametric approach to high-order curvilinear boundary-layer meshing. *Computer Methods in Applied Mechanics and Engineering*, 283:636 – 650, 2015.
11. J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2nd edition, 2006.
12. P.-O. Persson and J. Peraire. Curved mesh generation and mesh refinement using Lagrangian solid mechanics. In *Proceedings of the 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, page 949, 2009.

13. X. Roca, A. Gargallo-Peiró, and J. Sarrate. Defining quality measures for high-order planar triangles and curved mesh generation. In *Proc. of the 20th International Meshing Roundtable*, pages 365–383. Springer Berlin Heidelberg, 2012.
14. E. Ruiz-Gironés, J. Sarrate, and X. Roca. Generation of curved high-order meshes with optimal quality and geometric accuracy. In *Proc. of the 25th International Meshing Roundtable*, volume 163, pages 315–327. Procedia Engineering, 2016.
15. S. J. Sherwin and J. Peiró. Mesh generation in curvilinear domains using high-order elements. *International Journal for Numerical Methods in Engineering*, 53(1):207–223, 2001.
16. M. Stees and S. M. Shontz. A high-order log barrier-based mesh generation and warping method. In *Proc. of the 26th International Meshing Roundtable*, volume 203, pages 180 – 192. Procedia Engineering, 2017.
17. T. Toulorge, C. Geuzaine, J.-F. Remacle, and J. Lambrechts. Robust untangling of curvilinear meshes. *Journal of Computational Physics*, 254:8 – 26, 2013.
18. M. Turner, D. Moxey, J. Peir, M. Gammon, C. R. Pollard, and H. Bucklow. A framework for the generation of high-order curvilinear hybrid meshes for CFD simulations. In *Proc. of the 26th International Meshing Roundtable*, volume 203, pages 206 – 218. Procedia Engineering, 2017.
19. Z. J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. T. Huynh, et al. High-order CFD methods: Current status and perspective. *International Journal for Numerical Methods in Fluids*, 72(8):811–845, 2013.
20. Z. Q. Xie, R. Sevilla, O. Hassan, and K. Morgan. The generation of arbitrary order curved meshes for 3D finite element analysis. *Computational Mechanics*, 51(3):361–374, 2013.