
Anisotropic Mesh Adaptation for Evolving Triangulated Surfaces

Xiangmin Jiao¹, Andrew Colombi², Xinlai Ni², and John C. Hart²

¹College of Computing, Georgia Institute of Technology, Atlanta, GA 30332

²Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801

Summary. Dynamic surfaces arise in many applications, such as free surfaces in multiphase flows and moving interfaces in fluid-solid interactions. In many applications, an explicit surface triangulation is used to track the dynamic surfaces, posing significant challenges in adapting their meshes, especially if large curvatures and sharp features may dynamically appear or vanish as the surfaces evolve. In this paper, we present an anisotropic mesh adaptation technique to meet these challenges. Our technique strives for optimal aspect ratios of the triangulation to reduce interpolation errors and to capture geometric features based on a novel extension of the quadric-based surface analysis. Our adaptation algorithm combines the operations of vertex redistribution, edge flipping, edge contraction, and edge splitting. Experimental results demonstrate the effectiveness of our anisotropic adaptation techniques for static and dynamic surfaces.

Key words: Mesh adaptation; anisotropic meshes; dynamic surfaces; feature preservation

1 Introduction

Many computational applications involve triangulation of complex surface geometry, and an increasing number of them involve dynamically changing surfaces, such as free surfaces in multiphase flows [39] and moving boundaries in fluid-solid interactions [26]. In these simulations, the geometry is not known *a priori* and is part of the solution of a numerical simulation. As a surface evolves, the surface may undergo severe expansion or contraction in different regions or along different directions, leading to large curvatures, sharp features, and even topological changes. It is therefore often necessary to adapt the meshes for these complex dynamic surfaces to maintain a valid surface representation with minimal geometric errors.

Mesh adaptation has been an active research subject in numerical simulations for nearly two decades [32, 35]. In recent years anisotropic generation

and adaptation of 2-D triangular and 3-D tetrahedral meshes have attracted significant attention to reduce or minimize errors [3, 7, 8, 11, 12, 13, 15, 19, 25, 28, 29, 37, 38]. In general, these methods strive to equi-distribute errors by adapting the mesh density based on a metric tensor field, typically estimated based on the Hessian of a solution field. Anisotropic meshes can also significantly enhance the accuracy of a surface representation [1, 6, 10, 20]. Very few methods have been developed on anisotropic meshing and remeshing of a surface mesh [1, 10]. The method of Alliez et al. [1] generates quad-dominant meshes, and the method of Cheng et al. [10] considers only smooth, implicit surfaces, which may not be suitable for triangulated surfaces with sharp features that occur in many simulations. In addition, moving surfaces introduce significant additional complexities and constraints to mesh adaptation. A robust dynamic triangulation algorithm was developed by Cheng et al. [9], which is specialized for skin surfaces and it is unclear how to generalize that algorithm to other surfaces. Therefore, anisotropic mesh adaptation for static or dynamic surfaces remains a significant challenge.

In this paper, we investigate the problem of adapting a dynamic surface mesh within a numerical simulation to reduce geometric errors. We propose an extension of the quadric-based surface analysis by Heckbert and Garland [20] and relate it to the interpolation error of a surface. Based on this analysis, we define a Riemannian metric tensor to adapt the surface mesh anisotropically using a combination of vertex redistribution, edge flipping, edge contraction, and edge splitting. These operations improve not only mesh quality but also the accuracy of the geometric representation. The interplay between different operations is potentially very complicated. To keep the algorithm simple, we optimize the mesh with vertex redistribution and edge flipping under geometric constraints, and use edge splitting and edge contraction to resolve pathological situations due to constraints. We compare the numerical solutions using anisotropic adaptation and isotropic adaptation, and demonstrate the significant advantages of our anisotropic adaptation. For simplicity this paper assumes the surface does not change topology during evolution.

The remainder of the paper is organized as follows. Section 2 presents some background information on anisotropic meshes and quadric-based surface analysis. Section 3 proposes a novel anisotropic transformation for surface meshes and applies it to mesh optimization using vertex redistribution and edge flipping. Section 4 describes the resolution of extreme angles and adaptation of mesh density using edge splitting and edge contraction. Section 5 presents numerical results and comparisons with isotropic adaptation for static and dynamic surfaces. Finally, Section 6 concludes the paper with a discussion.

2 Background

In this section, we present some background information on anisotropic meshes and quadric-based surface analysis.

2.1 Anisotropic Meshes

It is well known that a mesh with long and skinny triangles can interpolate a function with large curvature more accurately than a mesh with equilateral triangles [11, 13, 19, 36, 38]. A general formulation of anisotropic mesh generation or adaptation in 2-D or 3-D is to define a $d \times d$ *metric tensor* $\mathbf{M}(\mathbf{x})$ (or simply \mathbf{M}) at each point \mathbf{x} in \mathbb{R}^d , where d is 2 or 3, respectively. \mathbf{M} is referred to as the *Riemannian metric tensor* or *fundamental tensor* [24]. By definition, \mathbf{M} is symmetric positive-definite and has an eigen decomposition

$$\mathbf{M} = \mathbf{E}\mathbf{\Lambda}\mathbf{E}^T, \quad (1)$$

where $\mathbf{\Lambda}$ is the diagonal matrix of the eigenvalues of \mathbf{M} , which are all real and positive, and \mathbf{E} is the matrix of the eigenvectors of \mathbf{M} . Geometrically, \mathbf{E} corresponds to a rotation matrix and $\mathbf{\Lambda}$ corresponds to scaling factors. Any point $\mathbf{y} = \mathbf{x} + \epsilon$ in an infinitesimal neighborhood of \mathbf{x} is mapped to

$$\tilde{\mathbf{y}} = \mathbf{y} + \sqrt{\mathbf{\Lambda}}\mathbf{E}^T\epsilon = \mathbf{y} + \sum_{i=1}^d \sqrt{\lambda_i}\mathbf{e}_i^T\epsilon\mathbf{e}_i, \quad (2)$$

which maps a circle (or sphere in 3-D) in the physical space into an ellipse (or ellipsoid) in a parametric space with the semiaxes proportional to $\sqrt{\lambda_{ii}}$ for $i \in [1, d]$. A curve $\mathbf{r}(s) : [a, b] \rightarrow \mathbb{R}^2$ is mapped to a curve with length

$$l = \int_a^b \sqrt{\dot{\mathbf{r}}^T\mathbf{M}\dot{\mathbf{r}}}ds, \quad (3)$$

which can be approximated using the midpoint rule as

$$l \approx \sqrt{(\mathbf{r}_b - \mathbf{r}_a)^T\mathbf{M}_{(a+b)/2}(\mathbf{r}_b - \mathbf{r}_a)}. \quad (4)$$

In numerical computations, \mathbf{M} is typically chosen to minimize the interpolation error of a function f in a certain norm. Using Taylor series expansion one can show that the error in interpolating a function f with linear elements is approximately $\epsilon^T\mathbf{H}\epsilon$ as ϵ tends to zero, where \mathbf{H} denotes the Hessian of f . The Hessian is symmetric and therefore has an eigen decomposition $\mathbf{H} = \mathbf{R}\mathbf{D}\mathbf{R}^T$, where \mathbf{D} is a diagonal matrix of the eigenvalues of \mathbf{H} (i.e., the curvatures of f) and \mathbf{R} is composed of the eigenvectors of \mathbf{H} . The eigenvalues of \mathbf{H} may be negative, and the metric tensor \mathbf{M} can be obtained by setting $\mathbf{\Lambda} = |\mathbf{D}|$ and $\mathbf{E} = \mathbf{R}$ in (1), i.e., $\mathbf{M} = \mathbf{R}|\mathbf{D}|\mathbf{R}^T$, which minimizes the interpolation error by equi-distributing the error.

To generalize the above formulation to surface meshes, a 2-D metric tensor may be defined on a local or global parametric space [1, 10]. For solution-based adaptation the metric tensor may be assumed to be given as an input to the meshing algorithm, but in problems such as dynamic surfaces, which themselves are the solutions, the adaptation requires a careful analysis of the evolving surface.

2.2 Quadric-Based Surface Analysis

Surface analysis is a fundamental subject in differential and numerical geometry. Mathematical analysis has traditionally focused on smooth surfaces [24]. Its generalization to discrete surfaces has attracted significant attention in recent years, but the focus has been limited to the asymptotic behavior of discrete representations of smooth surfaces [20, 30]. If a surface discretization is relatively coarse or has singularities, then most asymptotic analyses break down. The quadric-based analysis proposed by Heckbert and Garland [20] seems to generalize well for their connections with approximation theory [31] and singularity analysis, as we will show shortly.

Given a triangulated surface mesh let each vertex v be the origin of a local coordinate frame, and m be the number of the faces incident on v . Let \mathbf{N} be a $m \times 3$ matrix whose i th row vector is the unit outward normal to the i th incident face of v , and \mathbf{W} be an $m \times m$ diagonal matrix with W_{ii} equal to the weight associated with the i th face. We typically use the the face area of the i th incident face of v for the weight W_{ii} . Let \mathbf{A} denote the 3×3 matrix $\mathbf{N}^T \mathbf{W} \mathbf{N}$, which we refer to as the *quadric metric tensor*. Suppose \mathbf{G} is a diagonal matrix containing the eigenvalues of \mathbf{A} . Let λ_i denote \mathbf{G}_{ii} with $\lambda_1 \geq \lambda_2 \geq \lambda_3$, and \mathbf{V} be the matrix of the eigenvectors of \mathbf{A} , so $\mathbf{A} = \mathbf{V} \mathbf{G} \mathbf{V}^T$. We refer to the vector space spanned by the eigenvectors of \mathbf{A} corresponding to the relatively large eigenvalues of \mathbf{A} as its *primary space* and the complementary space as its *null space*. As we will show later, for smooth surfaces the null space of \mathbf{A} is closely related to the Riemannian metric tensor for error minimization, and the threshold between the eigenvalues for the primary and null spaces can be chosen geometrically.

Suppose the triangles incident on v discretize a rectangular neighborhood with dimensions ϵ_1 and ϵ_2 along the maximum and minimum curvature directions, and κ_1 and κ_2 are the maximum and minimum curvatures, respectively. Heckbert and Garland [20] have shown that as ϵ tends to 0 the eigenvalues of \mathbf{A} are

$$\lambda_1 \approx 4\epsilon_1\epsilon_2 - \frac{1}{2}(\lambda_2 + \lambda_3) \quad (5)$$

$$\lambda_2 \approx \frac{4}{3}\epsilon_1^3\epsilon_2\kappa_1^2 \quad (6)$$

$$\lambda_3 \approx \frac{4}{3}\epsilon_1\epsilon_2^3\kappa_2^2 \quad (7)$$

For meshes approximating a smooth surface, λ_2/λ_1 and λ_3/λ_1 approach 0 in the rates of $O(\epsilon_1^2)$ and $O(\epsilon_2^2)$, respectively.

This asymptotic analysis, however, is not applicable near singularities. We generalize it to treat singularities as follows. Consider folding a smooth surface to form a ridge along the minimum curvature direction at a point, and let θ denote the dihedral angle between the two sides of the ridge. It is easy to show that after the folding, the three eigenvalues become

$$\tilde{\lambda}_1 \approx \lambda_1 \cos^2 \frac{\theta}{2} + \lambda_2 \sin^2 \frac{\theta}{2} \quad (8)$$

$$\tilde{\lambda}_2 \approx \lambda_1 \sin^2 \frac{\theta}{2} + \lambda_2 \cos^2 \frac{\theta}{2} \quad (9)$$

$$\tilde{\lambda}_3 \approx \lambda_3 \quad (10)$$

Since $\lambda_2/\lambda_1 = O(\epsilon_1^2)$ before folding, near singularities $\tilde{\lambda}_2/\tilde{\lambda}_1 \approx \tan^2(\theta/2)$ after folding. At sharp corners, the eigenvalues become more complicated, but in general all three eigenvalues are $O(\epsilon_1\epsilon_2)$. This singularity analysis provides a geometrically meaningful way to select the thresholds for the null space (χ_c for λ_2/λ_1 and χ_r for λ_3/λ_1). It is the foundation of the feature detection methods of Jiao [22], Jiao and Alexander [23], which detect ridges and corners by determining the null space along with some safeguards.

The asymptotic interpretation $\lambda_2/\lambda_1 \approx \epsilon_1^2\kappa_1^2/3$ and the singularity interpretation $\tilde{\lambda}_2/\tilde{\lambda}_1 \approx \tan^2(\theta/2)$ are consistent within a constant factor, in that $\tan^2(\theta/2)$ approaches $\epsilon_1^2\kappa_1^2$ as θ tends to zero. This dual asymptotic and singularity analysis is very useful, as it provides a new tool to handle the ambiguous cases that arise frequently in discrete surfaces. In particular, a relatively coarse mesh may behave in an ambiguous manner like neither singular nor smooth surfaces, but λ_2/λ_1 would be approximately $\tan^2(\theta/2)$ and \mathbf{e}_3 would be along the minimum curvature direction. This dual analysis is particularly useful for dynamic surfaces, in which some areas may have increasingly large curvatures before sharp features emerge, and the quadrics may be used to capture such a transition consistently without resorting to two drastically different treatments for smooth regions and sharp features.

The quadrics are also closely related to optimal triangles in an asymptotic sense. We define the aspect ratio ρ of a triangle to be that of its minimum containing ellipse, as advocated by Heckbert and Garland [20]. A dual definition, where the optimal triangle is defined as the triangle with maximum area contained in an ellipse with a given aspect ratio, can also be used [13]. In both cases, the optimal aspect ratio of a triangle is $\sqrt{\kappa_1/\kappa_2}$, where κ_1 and κ_2 are the eigenvalues of the Hessian of the surface with respect to its local parameterization. It has been shown that the quadric-based surface simplification produces optimal anisotropic triangles in an asymptotic sense for smooth surfaces [20], but there seems to be no published result on how to apply this analysis to mesh adaptation. In the following, we propose a quadric-based procedure for anisotropic mesh adaptation.

3 Anisotropic Mesh Optimization

It is challenging to define a suitable tensor field for surface meshes while taking into account the potential singularities. In this section we propose a tensor field and apply it to anisotropic vertex redistribution and edge flipping.

3.1 Anisotropic Transformation

For triangulations of smooth surfaces one can construct the Riemannian metric tensor for anisotropic transformation by computing the curvature tensors at each vertex to approximate the Hessian of a surface [1]. Although this approach is well founded for sufficiently fine discretization of smooth surfaces, the estimated curvatures are not meaningful near singularities and may be sensitive to perturbation for coarse meshes.

To avoid these numerical problems, we propose to construct the Riemannian tensor $\mathbf{M}_{2 \times 2}$ based on the quadrics as

$$\mathbf{M} = \mathbf{R} \begin{bmatrix} \lambda_2 & 0 \\ 0 & \lambda_3 \end{bmatrix} \mathbf{R}^T, \quad (11)$$

where λ_i are the eigenvalues of the quadric metric tensor \mathbf{A} , and \mathbf{R} is a 2×2 rotation matrix from the principal directions to the axes of the coordinate frame. From our previous analysis, for surface meshes conforming to the tensor field \mathbf{M} , $\rho = \sqrt{\lambda_2/\lambda_3} \approx \sqrt{|\kappa_1/\kappa_2|}$. To see this, observe that

$$\rho = \sqrt{\lambda_2/\lambda_3} \approx \frac{|\epsilon_1 \kappa_1|}{|\epsilon_2 \kappa_2|} = \frac{|\kappa_1/\kappa_2|}{\epsilon_2/\epsilon_1}.$$

By definition $\rho = \epsilon_2/\epsilon_1$, and therefore $\rho^2 \approx |\kappa_1/\kappa_2|$. Note that we can multiply \mathbf{M} by any factor without changing the aspect ratio. The computation of \mathbf{M} does not require estimating the curvatures explicitly, so it is simple and efficient. In addition, \mathbf{A} and in turn \mathbf{M} are computed in an integral form, so they are not sensitive to perturbation, and it is justified to construct a larger geometric support by summing up \mathbf{A} (or \mathbf{M}) at neighboring vertices.

It is important to note that the preceding analysis is asymptotic in nature, and may be invalid near singularities and degenerate cases. In particular, if $\kappa_1 \approx \kappa_2 \approx 0$, the aspect ratio is very sensitive to perturbation. If $\lambda_2 \gg \lambda_3$, which may occur near singularities (where $\lambda_3/\lambda_2 = O(\epsilon^2)$ along a ridge) or on a cylindrical patch (where $\lambda_3/\lambda_2 = 0$), the aspect ratio may become arbitrarily large. Too large aspect ratios may lead to too small time steps or larger condition numbers in a numerical simulation [36], and in turn severely decrease the efficiency and defeat the purpose of anisotropic adaptation. To resolve these issues, we impose an upper bound on the aspect ratio by imposing lower and upper bounds on the eigenvalues (the entries of the diagonal matrix in (11)) of \mathbf{M} , i.e.,

$$\widetilde{\mathbf{M}} = \mathbf{R} \begin{bmatrix} \min\{\max\{\lambda_2, \psi_l\}, \psi_u\} & 0 \\ 0 & \min\{\max\{\lambda_3, \psi_l\}, \psi_u\} \end{bmatrix} \mathbf{R}^T. \quad (12)$$

Based on the singularity analysis of the quadrics, ψ_l and ψ_u should be in the form of $\lambda_1 \tan^2(\theta/2)$ for some angle θ . Compared with the thresholds χ_r and χ_c for feature detection, in general $\psi_l/\lambda_1 \leq \chi_r \leq \psi_u/\lambda_1 \leq \chi_c$ (e.g., 8° , 20° , 30° , and 45° after converting ψ_l/λ_1 , χ_r , ψ_u/λ_1 , and χ_c into angles by the mapping $f(\chi) = 2 \arctan \sqrt{\chi}$). These angles are tunable parameters that can be selected based on their geometric meanings and experiments. With this modified tensor, the surface would be strongly anisotropic only near sharp features or large-curvature areas, where anisotropy is desired to avoid large interpolation errors.

3.2 Anisotropic Mesh Smoothing

Mesh smoothing is a popular method for mesh enhancement by redistributing vertices without changing mesh connectivity. It is also often the most difficult step in mesh adaptation. This problem is particularly challenging for smoothing surfaces because the vertex motion must preserve the geometry typically without a CAD model. A high-order surface approximation may be constructed [1, 6, 18], but large errors may still accumulate after repeated smoothing or adaptation of a dynamic surface. A simple solution was proposed by Jiao [22] through a weighted-residual formulation of local volume conservation. Here we use a simpler form of that method to redistribute vertices tangentially within the null space of \mathbf{A} , and focus on the issue of anisotropy in mesh smoothing.

A typical smoothing procedure moves a point toward a weighted average of its neighbor vertices. For anisotropic smoothing we must take into account the Riemannian tensors when computing the average. In the previous subsection we discussed the construction of the tensor at each vertex, which uses its own local coordinate frame. To compute a weighted average at a vertex p , the tensors at p and its adjacent vertices must be transformed into the same coordinate frame.

Given two adjacent vertices q and p , we compute the rotation matrix \mathbf{R} as follows. If the first eigenvectors at the two vertices are the same, then the rotation matrix is simply $\mathbf{T}_q^T \mathbf{T}_p$, where \mathbf{T}_p is a 3×2 matrix whose column vectors are the second and third eigenvectors of \mathbf{A} at p , and similarly for \mathbf{T}_q . If the normals differ, then $\mathbf{T}_q^T \mathbf{T}_p$ is no longer a rotation matrix. Let \mathbf{S} denote $\mathbf{T}_q^T \mathbf{T}_p$. We compute \mathbf{R} based on the projection of the third eigenvector of \mathbf{A} at p onto \mathbf{T}_q , i.e.,

$$\mathbf{R}_2 = \frac{\mathbf{S}_2}{\|\mathbf{S}_2\|}, \text{ and } \mathbf{R}_1 = \frac{\mathbf{S}_1 - \mathbf{R}_2 \mathbf{S}_1 \mathbf{R}_2}{\|\mathbf{S}_1 - \mathbf{R}_2 \mathbf{S}_1 \mathbf{R}_2\|},$$

where \mathbf{R}_i and \mathbf{S}_i denote the i th column vectors of \mathbf{R} and \mathbf{S} , respectively. By plugging \mathbf{R} corresponding to each adjacent vertex q into (12), we trans-

form the tensor at q to the local frame at p . For each edge pq , its midpoint \mathbf{x} is then transformed to $\sqrt{\mathbf{D}_{\frac{1}{2}} \mathbf{R}_{\frac{1}{2}}^T \mathbf{T}_p^T (\mathbf{x} - \mathbf{p})}$, where $\mathbf{R}_{\frac{1}{2}} \mathbf{D}_{\frac{1}{2}} \mathbf{R}_{\frac{1}{2}}^T = \mathbf{M}_{\frac{1}{2}} = \frac{1}{2}(\mathbf{M}_p + \mathbf{M}_q)$, and the length of the edge is estimated based on (4) as $\sqrt{(\mathbf{p} - \mathbf{q})^T \mathbf{T}_p \mathbf{M}_{\frac{1}{2}} \mathbf{T}_p^T (\mathbf{p} - \mathbf{q})}$. Using these transformations we obtain the vertex's smoothed location $\tilde{\mathbf{p}}$ within the tangent plane at p and then project $\tilde{\mathbf{p}}$ to the null space at p (which is the same as the tangent plane at smooth vertices but is the tangent line at ridges) when mapping back to \mathbb{R}^3 .

In our overall smoothing procedure, we first compute the metric tensors at all vertices and then compute the new position for all vertices using a Jacobi-style iteration, which is easy to parallelize in numerical simulations and also avoids recomputing the metric tensors after moving each vertex. Correction steps may be performed to add correction terms to preserve the shape [22]. Note that in principle any averaging scheme may be plugged into our smoothing procedure to compute the new position within the tangent space, but some schemes may exhibit instability when using a Jacobi-style iteration so may not be suitable in our setting.

3.3 Anisotropic Edge Flipping

Edge flipping is a commonly used operation in meshing algorithms such as Delaunay triangulation. For each edge uv with opposite vertices p and q , we flip uv if the Delaunay flipping criterion (i.e., $\angle upv + \angle uqv > \pi$) is satisfied in its parametric space. A key question is how to define this parametric space. Unlike vertex redistribution, where a vertex is at the center of the geometric support of the computation, it is unnatural to choose any vertex as reference for edge flipping. Because of the integral nature of quadric metric tensors, we obtain a reference frame by summing up the quadric metric tensors at the four vertices, and then obtain the reference frame and Riemannian tensor from its null space. The origin of the reference frame is positioned at the average of the four points. This approach avoids biasing toward any vertex. After obtaining the tensor one can compute the angles in the parametric space or use the modified Delaunay criterion [7]. Our algorithm repeatedly flips the edges using this modified Delaunay criterion until convergence.

If the metric tensor is a constant, then this procedure would converge to the Delaunay triangulation within the parametric space independent of the flipping sequence. For general surfaces this process may not converge to a Delaunay mesh and may even run into an infinite loop due to the potential inconsistencies caused by discretization errors. We use a greedy strategy to flip edges in decreasing order of maximum opposite angle, avoiding infinite loops by flipping edges only once. For meshes with sharp features we prohibit flipping any edge that is marked as a ridge edge (which are identified also using the quadric metric tensor [22]) because such a flipping introduces large errors.

4 Anisotropy-Aware Mesh Repair

Vertex redistribution and edge flipping optimize a mesh with a given number of vertices and under other geometric constraints (such as boundary and features). If a dynamic surface undergoes severe expansion or contraction, the number of vertices may need to be increased or decreased to maintain the overall density. Furthermore, there may be some small or large angles that can restrict time steps or lead to arbitrarily large errors in the normal directions [36]. These bad angles may not be resolved by mesh optimization due to geometric constraints. We address these issues using a process referred as *mesh repair*. Mesh repair does not attempt to optimize any quality measures but focuses on safeguarding and resolving pathological cases. Our overall adaptation strategy is to iterate between mesh optimization and mesh repair. Mesh repair is critical for generality and robustness but is also difficult to analyze. To keep it simple, we choose the two simplest operations: edge splitting and edge contraction. These repair operations must be anisotropy-aware so that they do not undo the effect of anisotropic mesh optimization.

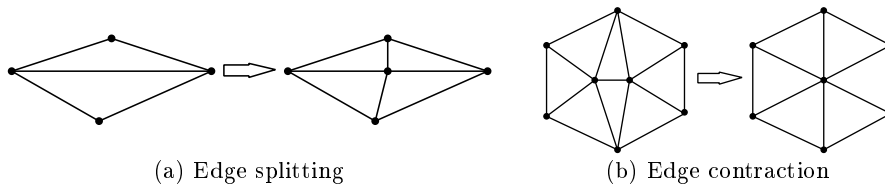


Fig. 1. Operations used in anisotropy-aware mesh repair

4.1 Edge Splitting

Edge splitting inserts a new vertex to an edge, as illustrated in Fig. 1(a). It helps to refine the area where the mesh may be too coarse, and to eliminate large angles with a long opposite edge. We choose the edges to split based on the following two criteria:

Absolute longness: the edge is the longest in its incident triangles and is longer than a given threshold L , or

Relative longness: the edge is longer than a desired edge length $l < L$, one of its opposite angles is close to π (greater than a threshold θ_l), and the shortest edge in its incident triangles is no shorter than s (where $s < l$).

The four parameters above must be chosen in a way consistent with edge contraction, which we consider in Sect. 4.3. In the second criterion the constraint on s is necessary to prevent over-refinement caused by splitting too small triangles. Note that this process leaves out some large angles, which will

be addressed by edge contraction. We split the edges in decreasing order of edge lengths. After splitting an edge the new vertex is first positioned at the edge center, and then projected along the normal direction to the point that minimizes a quadric error metric along the edge to preserve smoothness.

4.2 Edge Contraction

Edge contraction merges two adjacent vertices into a single vertex, and it helps to coarsen the mesh and to eliminate some very small angles (and also some large angles). Edge contraction is more difficult than edge splitting because it can potentially smear features and even cause mesh folding. We determine whether an edge is desirable to contract using the following criteria:

- Absolute small angle: its opposite angle in an incident triangle is smaller than a threshold θ_s , and the longest edge of the triangle is shorter than the desired edge length l , or
- Relative shortness: it is shorter than a small fraction r of the longest edge in its incident triangles, or
- Absolute small triangle: the longest edge in its incident faces is shorter than a given threshold S , and it is the shortest edge in its incident triangles, or
- Relative small triangle: the longest edge in its incident faces is shorter than the desired edge length l , and it is shorter than a fraction R of the longest edges in both physical and normalized space.

The first two criteria address poorly-shaped triangles with a very small angle or a short edge, and as a side product eliminate some triangles with very large angles. The last two criteria address well-shaped but too small triangles to decrease vertex density for mesh coarsening.

We contract edges in increasing order of edge lengths. To preserve features during contraction, if two vertices have different feature ranks (a smooth, ridge, and corner vertex has a feature rank of 0, 1, or 2, respectively) we place the merged vertex at the original vertex with the higher rank. If two vertices have the same rank, to preserve smoothness we obtain a weighted average of the original vertices and then project it along the normal direction to minimize the quadric error metric. To prevent mesh folding we reject contractions that would lead to topological changes or normal inversion of any face. We observe that such violations rarely occur as we contract shortest edges first.

4.3 Parameter Selection

Mesh repair requires a number of parameters as it deals with pathological cases by nature. The interplay among different parameters are quite complex. Overall, we have three different types of parameters: edge lengths (l , L , s , S), edge-length ratios (r , R), and angles (θ_l and θ_s).

For edge-length parameters, in general $s < S < l < L$, since S and L specify desirable lengths for edge contracting and splitting, and s/l is closely

related to the maximum aspect ratio allowed by anisotropic transformation (i.e., $\sqrt{\psi_u/\psi_l}$). For the edge-length ratio r and R , in general $r < R$ as they correspond to poorly-shaped and well-shaped triangles, respectively. The ratio r is related to the ratio of s/l in the relative-longness criterion, and in general $r \ll s/l \approx \sqrt{\psi_l/\psi_u}$. The threshold R is related to S , and we choose $R \approx S/l$.

For the angle thresholds, $\theta_s \ll \theta_l$, and it is desirable that $2\theta_s + \theta_l > 180^\circ$, so that large angles incident on a relatively short edge would be eliminated by edge contraction. The angle θ_s is also related to the thresholds ψ_l and ψ_u . The maximum of the minimum angle of a triangle contained in an ellipse with aspect ratio $\rho = \sqrt{\psi_u/\psi_l}$ is $2 \arctan \rho$, and therefore we choose $\theta_s \approx 2 \arctan \rho$.

Based on these consideration and extensive experiments, we choose the following default values for the parameters:

- $\psi_l = \lambda_1 \tan^2(4^\circ)$ and $\psi_u = \lambda_1 \tan^2(15^\circ)$;
- $r = 0.1$, $R = 0.5$;
- $s = l\sqrt{\psi_l/\psi_u}$, $S = Rl$, $L = 1.5l$;
- $\theta_s = 2 \arctan \sqrt{\psi_l/\psi_u} \approx 15^\circ$ and $\theta_l = 160^\circ$.

In general the desired edge length l may vary in space, but it typically suffices to have a uniform value as the desired average edge length. In the following we present some experimental results, all of which used the above default values.

5 Experimental Results

We present some preliminary results of anisotropic adaptation for static and dynamic surfaces to demonstrate its effectiveness, and compare it with isotropic remeshing for dynamic surfaces.

5.1 Remeshing Static Surfaces

When applied to a static surface our anisotropic adaptation algorithm essentially becomes a remeshing tool. We show two simple examples of static remeshing to demonstrate the effect of anisotropic adaptation. In the first example we remesh a small cube. As shown in Fig. 2, a layer of small triangles was formed around sharp features.

In the second example we remesh a surface mesh with corrupted sharp features, as shown in Fig. 3. We adapt the mesh while adding a normal motion to denoise the surface. The detail of this denoising procedure is beyond the scope of this paper, but its basic idea is to identify the noisy vertices similar to feature detection using the eigenvalues of the quadric metric tensor, and then add a normal motion in a volume-preserving fashion similar to the normal-diffusion approach of Ohtake et al. [33]. As obvious from the figures, the final mesh is anisotropic near high-curvature regions and both the mesh quality and surface geometry were improved substantially.

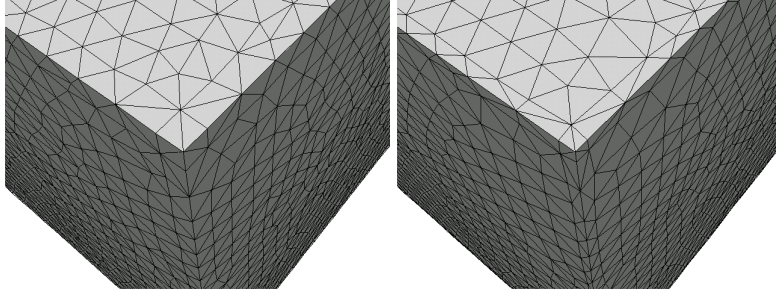


Fig. 2. Remesh cube with anisotropic mesh optimization

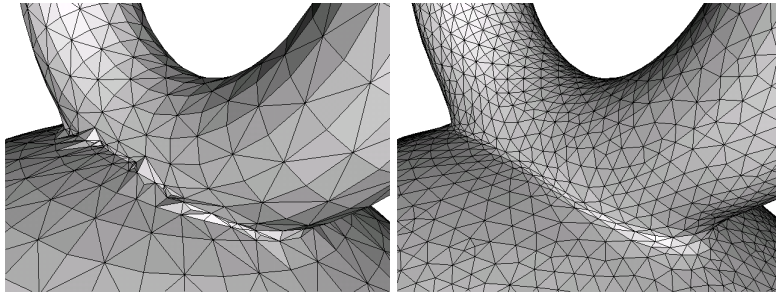


Fig. 3. Adapting a surface with corrupted features using anisotropic mesh adaptation

5.2 Adaptation of Dynamic Surfaces

As discussed earlier, the advantages for anisotropic surface adaptation are most prominent in adapting dynamically moving surfaces. We use a series of dynamic surfaces with an increasing level of difficulty to verify our assertion. In particular, we adapt a surface that is advected in two challenging flow fields for a time period T as detailed later. The larger T is the more severe the deformation becomes. We modulate time by the cosinusoidal function $\cos(\pi t/T)$ to make the flow periodic, so that in principle the shape at time $t = 0$ and $t = T$ should be identical. Such a test has been widely used to test dynamic surfaces or moving interfaces [4, 14, 16, 27]. In these tests, we propagate each vertex using the fourth-order Runge-Kutta integration scheme and then adapt the surface anisotropically. The time step was controlled using the approach of Jiao [21] to prevent mesh folding. We perform anisotropic mesh smoothing at every time step and invoke the full-fledged anisotropic adaptation every few iterations or when the time step becomes too small.

5.2.1 Modest Deformation

Dynamic surfaces pose significant challenges to mesh adaptation, and most traditional adaptation techniques lead to large errors or abrupt failure at a very early stage. In order to compare our method with existing adaptation techniques, we first consider the modest deformation of a sphere of radius 0.15 centered at $(0.5, 0.75, 0.5)$ in a vortex flow with a time period of $T = 2$. The spatial velocity of this deformation field is given by

$$u(x, y, z) = \sin^2(\pi x)(\sin(2\pi z) - \sin(2\pi y)), \quad (13)$$

$$v(x, y, z) = \sin^2(\pi y)(\sin(2\pi x) - \sin(2\pi z)), \quad (14)$$

$$w(x, y, z) = \sin^2(\pi z)(\sin(2\pi y) - \sin(2\pi x)). \quad (15)$$

This flow is a challenge as large curvatures develop at the maximum deformation. In this test we compare the results of using anisotropic mesh optimization with the isotropic remeshing, in particular our own implementation of the method of Alliez et al. [2]. In both cases we used a time step of 0.01. Figure 4 shows the meshes using anisotropic adaptation at times $t = 0, 1$, and 2 using a relative coarse initial mesh with 10,784 vertices and 21,564 triangles. Figure 5 shows the results using isotropic remeshing with uniform spacing using the same initial mesh. The anisotropic results are obviously far superior to the isotropic ones. Quantitatively, the volume loss for anisotropic adaptation was less than 0.1% compared to about 24% for isotropic remeshing. Note that the isotropic remeshing algorithm of Alliez et al. [2] can adapt vertex density based on curvatures, but we observed some numerical instability and worse results when adapting vertex density based on Gaussian or mean curvatures, probably because the curvatures are inherently sensitive to perturbation.

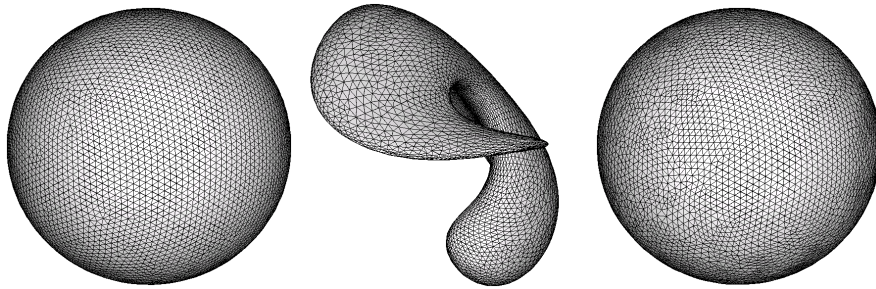


Fig. 4. Results of vortex flow using our anisotropic adaptation at $t = 0, 1$, and 2

In terms of efficiency, anisotropic adaptation took 9 minutes to complete the whole simulation on a PC with 3.2 GHz Pentium D processor. In comparison, the remeshing algorithm by Alliez et al. [2] is very expensive and takes several minutes even for a single remeshing step; therefore, we adopted

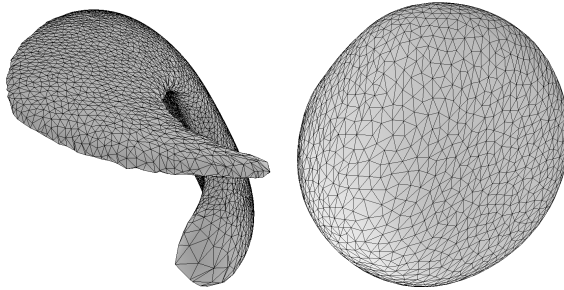


Fig. 5. Results using isotropic remeshing at times at $t = 1$, and 2 for the same flow and initial mesh as in Fig. 4

an approximation solver proposed by Ostromoukhov et al. [34] to make their algorithm more competitive. Even after this speed-up, isotropic remeshing took 30 seconds per remeshing step, which amounts to about 10 times slower than our technique. This relatively simple example shows the effectiveness and efficiency of anisotropic adaptation and demonstrates the challenges in remeshing dynamic surfaces.

5.2.2 Large Deformation

In the literature, another widely used test has a velocity field

$$u(x, y, z) = 2 \sin^2(\pi x) \sin(2\pi y) \sin(2\pi z), \quad (16)$$

$$v(x, y, z) = -\sin(2\pi x) \sin^2(\pi y) \sin(2\pi z), \quad (17)$$

$$w(x, y, z) = -\sin^2(2\pi x) \sin^2(2\pi y) \sin(\pi z), \quad (18)$$

which advects a sphere centered at $(0.35, 0.35, 0.35)$ for a time period of $T = 3$. This problem is sometimes referred to as the Enright test and has been solved using implicit surfaces, such as the level set method and its variations [16, 17] and hybrid front tracking methods [5, 14]. Note that if one simply propagates the vertices of a surface mesh independently, some triangles would become inverted very soon, so mesh adaptation is necessary.

We used an initial mesh with 23,238 vertices and 46,472 triangles with a time step of 0.015, so the whole computation took 200 iterations. Due to distortions introduced by the flow, anisotropic optimization alone cannot meet this challenge, so we used the full-fledged anisotropic mesh adaptation. We invoke anisotropic adaptation every 4 time steps. This flow is mildly unstable during the second half period, so a smoothing term similar to that in Fig. 3 was added to denoise the surface.

Figure 6 shows the surfaces with anisotropic mesh adaptation after 50, 100, 150, and 200 time steps. At the maximum deformation (i.e., $t = 1.5$) the surface area increased by a factor of 4.12, and the numbers of vertices and

triangles also roughly quadrupled (increased to 98,638 and 197,292, respectively). At time $t = 3$ the surface returned back to a nearly perfect sphere, and the errors in both the volume and surface area were less than 0.1%. Note that the popular level-set method lost 80% of volume for this test, while the much-improved particle level-set method of Enright et al. [16] lost 2.6% when using one million grid points. Because of the lower time complexity of our surface mesh-based scheme, the computation time of our method is expected to be orders of magnitude smaller than that of Enright et al. [16].

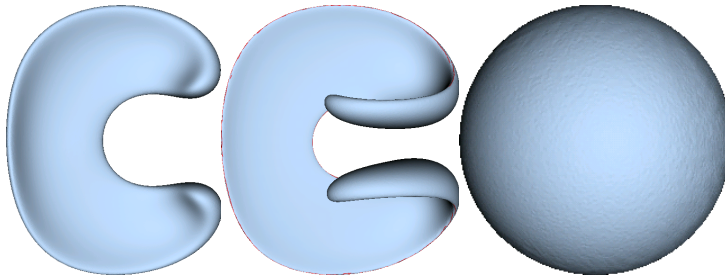


Fig. 6. Solution of Enright test with anisotropic mesh adaptation

5.2.3 Very Large Motion

The motion in the previous test is large, but it is still relatively simple because the surface remained smooth and the time period was relatively short. We test our method using a more challenging problem with the same flow as in Sect. 5.2.1 but a longer period of $T = 6$. Under this flow the sphere swirls for three cycles at the maximum deformation, forming cusps and extremely thin filaments, posing significant challenges to represent the surface accurately. To the best of our knowledge no solution to this problem has been reported previously in the literature, except for a result of $T = 4$ using a hybrid surface-marker and volume-of-fluid method [4]. Our simulation used a time step of 0.015 for 400 iterations using an initial mesh same as that in the previous test. Figure 7 shows the surface after 100, 200, and 400 time steps, respectively. At the maximum deformation the area increased by a factor of 5.6, and the numbers of vertices and triangles increased by a factor of 5.4. At time $t = 6$, the volume error was about 0.3%.

6 Conclusion

In this paper we present an effective approach for anisotropic adaptation of triangulated surfaces, with a focus on adapting dynamic surfaces that are the

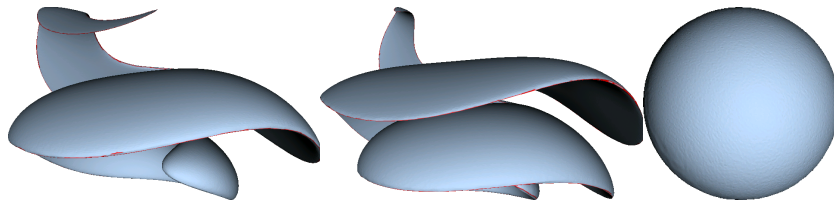


Fig. 7. Solution of vortex flow with anisotropic adaptation for period $T = 6$

solutions of numerical simulations. This setting poses significant challenges in accuracy and efficiency. We present an extension of the quadric-based surface analysis to treat singularities and in turn deliver a unified framework for resolving smooth surfaces, sharp features, and the ambiguities between them. We propose a simple and efficient transformation for anisotropic mesh adaptation with built-in safeguards for degeneracies, and use this transformation to optimize a mesh anisotropically. We also develop a mesh repair strategy to address pathological cases. The effectiveness of anisotropic adaptation was demonstrated using a number of examples, and orders of magnitude of improvements were achieved in accuracy and efficiency for dynamic surfaces compared to adapting the surface meshes isotropically or representing and propagating the surfaces using Eulerian methods. A number of research issues remain open for dynamic surfaces, including accurate resolution of topological changes of surface meshes and volume conservation in full-fledged mesh adaptation, which we are investigating and plan to report in the future.

Acknowledgement. This work was supported by the U.S. Department of Energy through the University of California under subcontract B523819 with the University of Illinois at Urbana-Champaign, and by NSF and DARPA under CARGO grant #0310446. We thank the anonymous reviewers for their helpful comments on improving the exposition of this work.

References

- [1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. *ACM Trans. Graph.*, 22:485–493, July 2003.
- [2] P. Alliez, E. Colin de Verdière, O. Devillers, and M. Isenburg. Isotropic surface remeshing. In *Proc. Shape Modeling Intl.*, 2003.
- [3] R. Almeida, P. Feijoo, A. Galeao, C. Padra, and R. Silva. Adaptive finite element computational fluid dynamics using an anisotropic error estimator. *Comput. Methods Appl. Mech. Engrg.*, 182:379–400, 2000.
- [4] E. Aulisa, S. Manservigi, and R. Scardovelli. A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking. *J. Comput. Phys.*, 197:555–584, 2004.
- [5] A. Bargteil, T. Goktekin, J. O’Brien, and J. Strain. A semi-lagrangian contouring method for fluid simulation. *ACM Trans. Graph.*, 25, 2006.

- [6] H. Borouchaki and P. Frey. Adaptive triangular-quadrilateral mesh generation. *Int. J. Numer. Meth. Engr.*, 41:915–934, 1998.
- [7] F. J. Bossen and P. S. Heckbert. A pliant method for anisotropic mesh generation. In *Proc. 5th Int. Meshing Roundtable*, pages 63–74, Oct. 1996.
- [8] G. Buscaglia and E. Dari. Anisotropic mesh optimization and its application in adaptivity. *Int. J. Numer. Meth. Engr.*, 40:4119–4136, 1997.
- [9] H. Cheng, T. Dey, H. Edelsbrunner, and J. Sullivan. Dynamic skin triangulation. *Disc. Comput. Geom.*, 25:525–568, 2001.
- [10] S.-W. Cheng, T. K. Dey, E. A. Ramos, and R. Wenger. Anisotropic surface meshing. In *Proc. 17th ACM-SIAM Sympos. Discrete Algorithms*, pages 202–211, 2006.
- [11] E. D’Azevedo and R. Simpson. On optimal interpolation triangle incidences. *SIAM J. Sci. Stat. Comput.*, 10:1063–1075, 1989.
- [12] E. D’Azevedo and R. Simpson. On optimal triangular meshes for minimizing the gradient error. *Numer. Math.*, 59:321–348, 1991.
- [13] V. Dolejsi. Anisotropic mesh adaptation for finite volume and finite element methods on triangular meshes. *Comput. Vis. Sci.*, 1:165–178, 1998.
- [14] J. Du, B. Fix, J. Glimm, X. Jia, X. Li, Y. Li, and L. Wu. A simple package for front tracking. *J. Comput. Phys.*, 213:613–628, 2006.
- [15] Q. Du and D. Wang. Anisotropic centroidal Voronoi tessellations and their applications. *SIAM J. Sci. Comput.*, 26:737–761, 2005.
- [16] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell. A hybrid particle level set method for improved interface capturing. *J. Comput. Phys.*, 183:83–116, 2002.
- [17] D. Enright, F. Losasso, and R. Fedkiw. A fast and accurate semi-Lagrangian particle level set method. *Comput. Struc.*, 83:479–490, 2005.
- [18] P. Frey. About surface remeshing. In *Proc. 9th Int. Meshing Roundtable*, pages 123–136, Oct. 2000.
- [19] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M.-G. Vallet. Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD. Part I: General principles. *Int. J. Numer. Meth. Fluids*, 32:725–744, 2000.
- [20] P. S. Heckbert and M. Garland. Optimal triangulation and quadric-based surface simplification. *Comput. Geom.*, pages 49–65, 1999.
- [21] X. Jiao. Face offsetting: A unified approach for explicit moving interfaces. *J. Comput. Phys.*, 2006. To appear.
- [22] X. Jiao. Volume and feature preservation in surface mesh optimization. In *Proc. 15th Int. Meshing Roundtable*, 2006.
- [23] X. Jiao and P. Alexander. Parallel feature-preserving mesh smoothing. In *Proc. Int. Conf. Comput. Sci. Appl.*, pages 1180–1189, 2005.
- [24] E. Kreyszig. *Introduction to Differential Geometry and Riemannian Geometry*, volume 16 of *Mathematical Expositions*. University of Toronto Press, 1968.
- [25] F. Labelle and J. R. Shewchuk. Anisotropic Voronoi diagrams and guaranteed-quality anisotropic mesh generation. In *Proc. 19th Ann. ACM Sympos. Comput. Geom.*, pages 191–200. ACM, 2003.
- [26] M. Lesoinne and C. Farhat. Geometric conservation laws for flow problems with moving boundaries and deformable meshes and their impact on aeroelastic computations. *Comput. Methods Appl. Mech. Engrg.*, 134:71–90, 1996.
- [27] R. Leveque. High-resolution conservative algorithms for advection in incompressible flow. *SIAM J. Numer. Anal.*, 33:627–665, 1996.

- [28] X. Li, M. S. Shephard, and M. W. Beall. 3d anisotropic mesh adaptation by mesh modification. *Comput. Methods Appl. Mech. Engrg.*, 194:4915–4950, 2005.
- [29] S. Lo. 3-d anisotropic mesh refinement in compliance with a general metric specification. *Finite Elements Anal. Design*, 38:3–19, 2001.
- [30] M. Meyer, M. Desbrun, P. Schroder, and A. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In H.-C. Hege and K. Polthier, editors, *Visualization and Mathematics III*, pages 35–58, 2003.
- [31] E. Nadler. Piecewise linear best L_2 approximation on triangulations. In C. K. C. et al., editor, *Approximation Theory V*, pages 499–502. Academic Press, 1986.
- [32] J. Oden, T. Strouboulis, and P. Devloo. Adaptive finite element methods for high-speed compressible flows. *Int. J. Numer. Meth. Fluids*, 7:1211–1228, 1987.
- [33] Y. Ohtake, A. G. Belyaev, and H.-P. Seidel. Mesh smoothing by adaptive and anisotropic Gaussian filter. In *Vision, Modeling and Visualization*, pages 203–210, 2002.
- [34] V. Ostromoukhov, C. Donohue, and P.-M. Jodoin. Fast hierarchical importance sampling with blue noise properties. *ACM Trans. Graph.*, 23(3):488–495, 2004.
- [35] J. Peraire, M. Vahdati, K. Morgan, and O. Zienkiewicz. Adaptive remeshing for compressible flow computations. *J. Comput. Phys.*, 72:449–466, 1987.
- [36] J. R. Shewchuk. What is a good linear element? interpolation, conditioning, and quality measures. In *Proc. 11th Int. Meshing Roundtable*, pages 115–126, 2002.
- [37] K. Shimada, A. Yamada, and T. Itoh. Anisotropic triangular meshing of parametric surfaces via close packing of ellipsoidal bubbles. In *Proc. 6th Int. Meshing Roundtable*, pages 375–390, 1997.
- [38] R. B. Simpson. Anisotropic mesh transformations and optimal error control. *Applied Numer. Math.*, 14:183–198, 1994.
- [39] G. Tryggvason, B. Bunner, A. Esmaceli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. A front-tracking method for the computations of multiphase flow. *J. Comput. Phys.*, 169:708–759, 2001.