Quad-Dominant Mesh Adaptation Using Specialized Simplicial Optimization

Ko-Foa Tchon and Ricardo Camarero

Department of Mechanical Engineering, École Polytechnique de Montréal,
C.P. 6079, Succ. Centre-ville, Montreal (QC) H3C 3A7, Canada
[ko-foa.tchon|ricardo.camarero]@polymtl.ca

Summary. The proposed quad-dominant mesh adaptation algorithm is based on simplicial optimization. It is driven by an anisotropic Riemannian metric and uses specialized local operators formulated in terms of an L_∞ instead of the usual L_2 distance. Furthermore, the physically-based vertex relocation operator includes an alignment force to explicitly minimize the angular deviation of selected edges from the local eigenvectors of the target metric. Sets of contiguous edges can then be effectively interpreted as active tensor lines. Those lines are not only packed but also simultaneous networked together to form a layered rectangular simplicial mesh that requires little postprocessing to form a cubical-dominant one. Almost all-cubical meshes are possible if the target metric is compatible with such a decomposition and, although presently only two-dimensional tests were performed, a three-dimensional extension is feasible.

Key words: Quad-dominant, mesh adaptation, anisotropic Riemannian metric.

1 Introduction

Elementary modification operators are essential to optimize the computational meshes used by finite element and finite volume solvers. They have been successfully employed to automatically adapt simplicial meshes of triangles, in two dimensions, and tetrahedra, in three dimensions [1]. However, this level of automation has not yet been duplicated for cubical meshes of quadrilaterals, in two dimensions, and hexahedra, in three dimensions. Generation, let alone adaptation, of such meshes is still a challenge. There is, nevertheless, a strong demand for quality cubical meshes either due to the intrinsic properties of such elements or simply for compatibility with existing solvers.

Although fairly robust two-dimensional methods have been developed, current conformal all-hexahedral algorithms are usually limited in scope and cannot automatically process arbitrary shaped domains [2]. Acknowledging this difficulty, cubical-dominant algorithms allow a small percentage of non-cubical elements in order to achieve an increased level of automation. The present work proposes such an algorithm that combines cubical particle packing [3, 4]

with tensor line networking [5] and recasts the whole process as a specialized simplicial optimization. More precisely, physically-based attraction-repulsion forces are used to distribute the vertices of a simplicial mesh according to the local density prescribed by an anisotropic Riemannian control metric. Coupled with appropriate particle or vertex population control, this results in an approximate centroidal Voronoi-Delaunay triangulation. Furthermore, the proximity-based particle interaction force is modified to promote cubical Voronoi regions by using an L_∞ norm instead of the usual L_2 norm to compute metric distance. Particle population control is also reformulated in terms of simplicial refinement and coarsening operations using the same chessboard distance. These modifications provide, however, only local alignment and an additional constraint is needed to recover the globally layered structure of an ideal cubical mesh. An angle-based torsional spring-like force is used for this purpose and aligns selected mesh edges with the local metric eigenvectors. A physical interpretation of this optimization process can best be described as line networking. Set of contiguous mesh edges indeed effectively form tensor lines that are not only packed but also simultaneously interconnected together to form a layered rectangular simplicial mesh that requires little processing to form a cubical-dominant one. Almost all-cubical meshes are also possible if the target metric is compatible with such a decomposition. Finally, although presently only two-dimensional cases were considered, a three-dimensional extension is feasible and should be computationally competitive with classical simplicial optimization.

Following a brief summary of existing adaptation methods, the present paper describes the proposed specialized simplicial optimization algorithm and uses both academic and practical test cases to illustrate its capabilities.

2 Simplicial Versus Cubical Mesh Adaptation

2.1 Riemannian Metrics and Distance

Mesh adaptation algorithms are typically controlled by a target map specifying the desired element size according to its location in the domain. For anisotropic maps, size also varies according to element orientation. Metric-based algorithms cast such a target map as a tensor representing a deformation of space that modifies how the length, area and volume of mesh entities are measured [6, 7]. Such a Riemannian metric tensor is a symmetric positive definite matrix $\mathbf{M}$ and can be factored as the product of a rotation matrix $\mathbf{R}$ and a diagonal matrix $\mathbf{\Lambda}$ as in this two-dimensional formula

$$\mathbf{M} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{-1} = \begin{pmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} \mathbf{e}_1^t \\ \mathbf{e}_2^t \end{pmatrix}. \tag{1}$$

The columns of $\mathbf{R}$ are the eigenvectors of $\mathbf{M}$ and correspond to two prescribed directions $\mathbf{e}_1$ and $\mathbf{e}_2$. The diagonal terms λ_1 and λ_2 are the strictly positive eigenvalues of $\mathbf{M}$. The target sizes h_1 and h_2 along $\mathbf{e}_1$ and $\mathbf{e}_2$ are given by the inverse square root of those eigenvalues, i.e., $h_i = 1/\sqrt{\lambda_i}$.

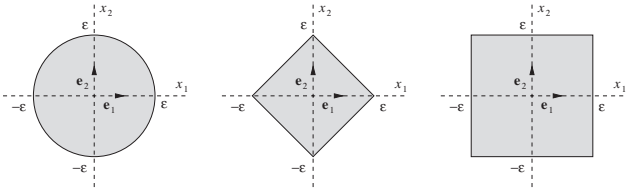

Fig. 1. Two-dimensional ϵ-balls for a metric distance based on an L_2 (left), an L_1 (middle) and an L_∞ norm (right).

Using such a target map, adapting a simplicial mesh is equivalent to requiring that all its edges have a unit metric length, as explained in Sect. 2.2. The notion of metric distance between two vertices is thus essential. For a locally constant metric $\mathbf{M}$, it is defined as follows

$$l_{ab} = \sqrt{(\mathbf{p}_b - \mathbf{p}_a)^t \, \mathbf{M} \, (\mathbf{p}_b - \mathbf{p}_a)} \tag{2}$$

where a and b are any two vertices and $\mathbf{p}$ is a position vector. For non-constant metrics, l_{ab} can be approximated by integrating this formula along segment ab or by using an averaged metric. Following Minkowski's formula in Euclidean space, a generalized distance can also be defined in metric space as an L_p norm

$$l_{ab} = \left(\sum_{i=1}^{d} |x_i|^p \right)^{1/p} \tag{3}$$

where $x_i = \sqrt{\lambda_i} \, \mathbf{e}_i . (\mathbf{p}_b - \mathbf{p}_a)$ and d is the considered dimension. If $p = 2$ then distance is measured using the classical L_2 norm given in Eq. 2. On the other hand, $p = 1$ corresponds to an L_1 norm also called taxicab or Manhattan distance while $p = \infty$ corresponds to the infinity norm also called Chebyshev or chessboard distance

$$l_{ab} = \max_{1 \le i \le d} |x_i| \, . \tag{4}$$

The differences induced by those norms in the partition of space are illustrated in Fig. 1. The chessboard distance, i.e., the L_∞ norm, is of particular interest for cubical adaptation as explained in Sect. 3.1.

2.2 Simplicial Adaptation

The ideal simplex is considered to be a *regular* one, i.e., an equilateral triangle in two dimensions (Fig. 2) or an equilateral tetrahedron in three dimensions. A quality regular simplicial mesh should be composed of such elements. However, equilarity implies constant or almost-constant element sizes. To facilitate the generation of variable density simplicial meshes, a Riemannian metric tensor can be introduced. A quality mesh should then be composed of regular simplices in metric space, i.e., the edges of its elements should all have the same metric length or, more precisely, a unit metric length. A perfectly adapted

mesh is therefore called a unit mesh. This compact and elegant framework to measure shape quality and size conformity has been extensively used to generate adapted simplicial meshes. See for example [1] and the references cited therein. Furthermore, adapting a mesh to a solution is an iterative process that can be done by global mesh regeneration or local mesh modifications. In an iterative adaptation context where modifications to a previous mesh are expected to be minimal, the latter approach may be advantageous. The elementary simplicial mesh modification primitives or operators optimize the vertex density and element shape quality to get as close as possible to a unit mesh in metric space. They can be classified as follows:

- refinement and coarsening improve local vertex density by inserting and deleting vertices;
- reconnection improves element shape by flipping faces and edges;
- relocation improves both element shape and local vertex density by repositioning individual vertices at optimal locations.

Those primitives can also be used after global mesh generation as postprocessing operations. Global remeshing can then be viewed as a way to produce good initial meshes for a local optimization process.

2.3 Cubical Adaptation

As for simplices, the ideal cubical element is a *regular* one, i.e., a square in two dimensions (Fig. 2) or a cube in three dimensions. This, however, implies edges not only equal in length but also joined at right angles. In practice, this orthogonality is very important and, in an anisotropic metric-based adaptation framework, can be preserved in physical space only if the edges

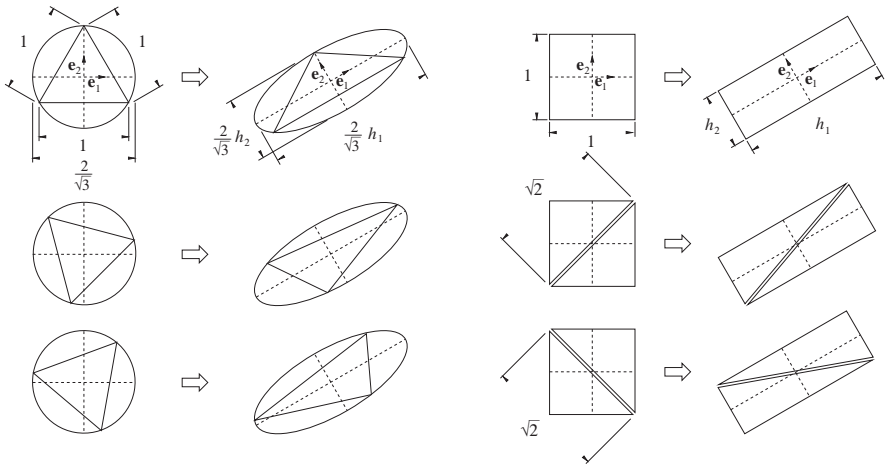

Fig. 2. Ideal triangle (left) versus ideal quadrilateral and its triangular decompositions (right) in metric space and in physical space. The metric tensor is considered locally constant.

of the cubical element are aligned with the local eigenvectors. This explicit directional constraint contrasts with the weak and indirect alignment of an adapted simplex. A perfect simplicial element can indeed have any orientation as long as its edges have a unit metric length and its loose alignment with the minor eigenvector of the metric, corresponding to the smallest eigenvalue and thus the biggest target size, is only due to the stretching introduced by the spacial transformation (Fig. 2). A perfectly adapted cubical mesh is thus understandably more difficult to obtain than a simplicial one. In addition to this directional constraint, cubical connectivity is also more complex to manipulate. It indeed has a layered structured that practically precludes strictly local modifications. Vertex relocation or smoothing is the most widely used method to adapt cubical meshes, particularly structured ones [8]. However, although it can improve both vertex density and element shape, smoothing is limited by a fixed mesh topology. Mesh refinement and coarsening, on the other hand, involve connectivity modifications [9, 10, 11, 12]. The major issue with conformal refinement-coarsening is, however, that element shape quality cannot be maintained without a powerful local reconnection method and the coveted cubical flip operator is still only theoretical in three dimensions [13].

Simplicial-to-cubical conversion methods avoid this problem by optimizing simplicial meshes and generating cubical elements only in a postprocessing step. Local mesh modifications thus only involve simplex manipulation and are greatly facilitated. Once the target vertex density is achieved, adjacent simplicial elements are merged to form cubical ones. Different merging strategies are possible including advancing fronts [14, 15, 16], graph theory [17] and quality constraints [18, 3, 4]. Although, some non-cubical elements may remain, particularly in three dimensions, such cubical-dominant meshes are acceptable in many applications. Furthermore, while Riemannian metrics have been used with direct cubical adaptation, the lack of a proper local reconnection operator limits their appeal. This indirect approach, on the other hand, can use the full complement of local optimization operators available for simplicial meshes. However, the ideal vertex distribution for a unit simplicial mesh is very different from the one required by perfectly adapted cubical elements. Not all edges should indeed have a metric length of one. For example, in two dimensions, edges that correspond to diagonals in the quadrilateral mesh should have a metric length of $\sqrt{2}$. Additionally, right triangles with sides aligned with the local metric eigenvectors are better suited than equilateral ones for simplicial-to-cubical conversion (Fig. 2).

Better vertex distributions can be obtained using physically-based packing algorithms that approximate centroidal Voronoi-Delaunay triangulations. An appropriate modification of the proximity-based force used to distribute the mesh vertices indeed enables the partition of the domain into cubical Voronoi regions [3, 4]. This improves the local alignment with the metric eigenvectors of the resulting simplices. The merged cubical elements thus have better shape and require much less postprocessing. The globally layered structure of the ideal cubical mesh is, however, difficult to recover using a strictly local method. Networks of tensor lines everywhere tangent to the local metric eigenvectors can be used for this purpose [5]. Due to the orthogonal nature

of this network, the resulting quadrilateral elements are very well shaped and triangular elements are only inserted when the tensor lines are fragmented to ensure the conformity of the final mesh. The practical implementation of such a method requires, however, an assortment of techniques that are not as computationally efficient as simplicial adaptation and may be difficult to extend to three dimensions.

The next section proposes an alternative approach that combines cubical particle packing with tensor line networking and recasts the whole process as a specialized simplicial optimization.

3 Specialized Simplicial Optimization

3.1 Specialized Simplicial Operators

For high quality simplicial-to-cubical conversion, edges corresponding to diagonals should have an L_2 metric length of $\sqrt{2}$, in two dimensions, or $\sqrt{3}$, in three dimensions, and not 1. The unit meshes generated by classical simplicial operators based on such an L_2 distance are thus not best suited. This discrepancy can be avoided by using an L_∞ norm to compute metric distance because all edges should then have a unit length including diagonals. This chessboard distance has thus been used to modify the specialized two-dimensional operators presented here.

Vertex relocation – A physically-based approach was chosen in the present work [19, 20, 3, 4]. In this paradigm, mesh vertices are particles and the following potential is used to derive the interaction forces between those particles

$$\phi(x) = \frac{1}{4}\, e^{-x^4} - \frac{3}{16}\, \Gamma\left(\frac{1}{4}, x^4\right) + C \tag{5}$$

where $\Gamma(a,z) = \int_z^\infty t^{a-1}e^{-t}dt$ is the incomplete gamma function and C is an arbitrary constant. The first derivative of this potential corresponds to the function given by Bossen and Heckbert [20]

$$\phi'(x) = \frac{d\phi}{dx} = \left(1 - x^4\right)\, e^{-x^4}. \tag{6}$$

If an L_2 norm is used to measure distance then the following attraction-repulsion force is exerted on vertex or particle i by particle j

$$\mathbf{F}_{ij} = -\frac{\phi'(l_{ij})}{l_{ij}}\, (\mathbf{p}_j - \mathbf{p}_i) = -\phi'(l_{ij})\, h_{ij}\mathbf{u}_{ij} \tag{7}$$

where l_{ij} is the metric distance between i and j as defined in Eq. 2, h_{ij} is the target size along edge ij and $\mathbf{u}_{ij}$ is the unit vector $(\mathbf{p}_j - \mathbf{p}_i)/\|\mathbf{p}_j - \mathbf{p}_i\|$. When coupled with appropriate population control, such a force will distribute the particles according to the density prescribed by the target metric. At equilibrium, the empty region maintained by this force around each particle

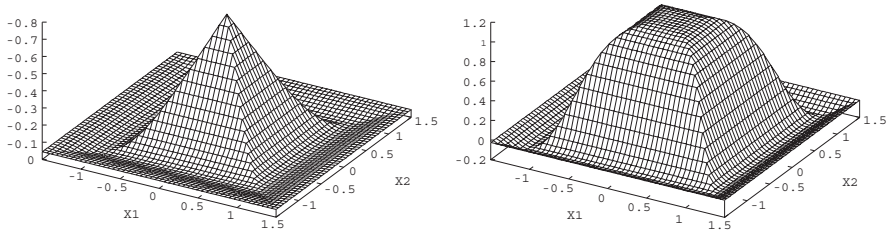

Fig. 3. Two-dimensional attraction-repulsion potential field (left) and the modulus of the corresponding force (right) for an L_∞ distance and an Euclidean metric, i.e., the identity matrix. The constant C in Eq. 5 is chosen so that the potential is equal to zero at a unit distance.

is analogous to an approximate Voronoi cell. Furthermore, since the particle is at the center of this cell, the resulting partition of space is an approximate centroidal Voronoi tessellation. The Voronoi cells have the shape of the ϵ-ball associated with the type of distance used to define the attraction-repulsion potential (Fig. 1). When an L_2 distance is used, those Voronoi cells have an elliptic or ellipsoidal shape. However, if an L_∞ or chessboard distance is used instead then those cells will have the desired cubical shape. Let m be the index of the eigenvector for which $|\sqrt{\lambda_m}\,\mathbf{e}_m.(\mathbf{p}_j - \mathbf{p}_i)|$ is maximum. The chessboard attraction-repulsion force is then given by

$$\mathbf{F}_{ij} = -\alpha\,\phi'(l_{ij})\,h_m\mathbf{e}_m \tag{8}$$

where l_{ij} is computed with Eq. 4, α is the sign of $\mathbf{e}_m.(\mathbf{p}_j - \mathbf{p}_i)$ and $h_m = 1/\sqrt{\lambda_m}$ is the target size associated with the local metric eigenvector $\mathbf{e}_m$ (Fig. 3). Using a first order equation of motion [20], the position of a particle or vertex i is updated at each iteration n as follows

$$\mathbf{p}_i^{n+1} = \mathbf{p}_i^n + \omega \sum_{j \in \mathcal{N}_i^v} \mathbf{F}_{ij}^n \tag{9}$$

where ω is a constant set to 0.2 and $\mathcal{N}_i^v$ is the set of vertices sharing an edge with i. Of course boundary vertices have to be reprojected after each update.

Refinement and coarsening – Mesh refinement and coarsening correspond to particle population control in the physical paradigm. When the local density is too low, a particle is created and, when it is too high, a particle is destroyed. The normalized density around a particle i can be estimated by the inverse of the averaged metric area of its neighbors

$$\rho_i = \frac{\beta\,|\mathcal{N}_i^e|}{\sum\limits_{j \in \mathcal{N}_i^e} \sqrt{\det(\mathbf{M}_j)}\;A(T_j)} \tag{10}$$

where $A(T_j)$ is area of triangle T_j, $\mathbf{M}_j$ is the averaged metric within T_j, $\mathcal{N}_i^e$ is the set of triangles sharing i and $|\mathcal{N}_i^e|$ is the number of triangles in this set.

The constant β is set to $\sqrt{3}/4$ for ellipse packing and 0.5 for square packing. The associated error is computed as follows

$$\delta(\rho) = \begin{cases} \rho - 1 & \text{if } \rho \geq 1, \\ \frac{1}{\rho} - 1 & \text{otherwise.} \end{cases} \tag{11}$$

The edge splitting and collapsing primitives from simplicial optimization can be used to insert and delete vertices and control this density. More precisely, if an edge has a metric length greater than a given threshold $l_{\max}$ then it is split in two by introducing a new vertex in the middle. Similarly, if an edge has a metric length lower than a given threshold $l_{\min}$ then it is collapsed by merging its two end vertices. Depending on the optimization strategy used, those operators may, however, tend to oscillate for rapidly varying metrics: they delete vertices that they just inserted. To stabilize the process, an estimation of the normalized density after each operation can be used as a safeguard [20]. Let ρ_i be the local density around a vertex i as defined in Eq. 10. When one of the interior edges connected to i is split then the local density becomes

$$\rho_i^+ = \rho_i \left(|\mathcal{N}_i^e| + 2\right) / |\mathcal{N}_i^e| \tag{12}$$

because two neighboring triangles must also be split in two. Similarly, when one interior edge connected to i is collapsed then the local density becomes

$$\rho_i^- = \rho_i \left(|\mathcal{N}_i^e| - 2\right) / |\mathcal{N}_i^e| \tag{13}$$

because two neighboring triangles must also be collapsed. Only the length based criterion is used for boundary edges but interior edges are split or collapsed only if $\delta(\rho_i^+) \leq \delta(\rho_i)$ or $\delta(\rho_i^-) \leq \delta(\rho_i)$ respectively. Again, to make the packing cubical, an L_∞ norm is used to compute the metric length of the edges. Furthermore, care must be taken when dealings with edges on curved boundaries to avoid degenerate configurations. See for example [21] for more details on how it can be done.

Local reconnection – In simplicial-to-cubical conversion methods, what is important is the rectangular distribution of the vertices more than their connectivity, as long as it is reasonable, i.e., coherent with the local target mesh density. This simplicial connectivity is essentially used for fast neighbor searching during relocation and population control operations. Once the vertex distribution is rectangular their simplicial connections will naturally be made of right angle simplices. That is why the local reconnection used in the present work is the classical operator from simplicial optimization. Furthermore, as only two-dimensional cases have been considered for now, only an edge swap or flip has been implemented. This operator replaces the edge shared by two triangles with a new edge linking their opposite vertices. This flip is only performed if the worst shape of the resulting triangles is better than the shape of the initial ones. The following measure of shape quality was used

$$\mathcal{Q}(T) = 4\sqrt{3}\, A(T) \min_{1 \leq i \leq 3} \frac{\sqrt{\det(\mathbf{M}_i)}}{\sum\limits_{1 \leq j < k \leq 3} (\mathbf{p}_k - \mathbf{p}_j)^T \mathbf{M}_i (\mathbf{p}_k - \mathbf{p}_j)} \tag{14}$$

where i, j and k refer to the vertices of triangle T.

3.2 Tensor Line Alignment

Although an improvement over classical simplicial operators, the modifications presented in the previous section provide only local alignment with the eigenvectors of the target metric. The resulting Voronoi regions will be cubical but there is no guaranty that those regions will be aligned to form continuous layers that are so important to maximize the proportion of cubical elements in the final mesh. The staggered configuration in Fig. 4 is as valid as the aligned one. Only adequate boundary conditions can favor the latter one. A stronger and more explicit global alignment force is needed. The approach proposed here is reminiscent of the method introduced in [5]. The difference is that here the global metric topology embodied by the tensor line network is no longer traced beforehand but is recovered through essentially local modifications of a simplicial mesh.

For this purpose a torsion spring-like force explicitly minimizes the angular deviation of selected mesh edges with the local metric eigenvectors. Associated with lineal springs, such angle-based forces are already used in mesh smoothing [22]. Their use here is, however, more analogous to active polylines in computer graphics. More precisely, those active polylines are tensor lines formed by links that correspond to selected mesh edges. These edges will become the sides of the final cubical elements. They are identified by computing their angular deviation from the local eigendirections. Taking into account the deformation introduced by a metric $\mathbf{M}$, this angular deviation for an edge ij is given by

$$\theta_{ij} = \arccos \frac{\sqrt{\lambda_m}\, \mathbf{e}_m.(\mathbf{p}_j - \mathbf{p}_i)}{\sqrt{(\mathbf{p}_j - \mathbf{p}_i)^t \mathbf{M} (\mathbf{p}_j - \mathbf{p}_i)}} \tag{15}$$

where m has the same definition as in Eq. 8. If $|\theta_{ij}| \leq \theta_{\max}/2$ then ij is a candidate tensor line link for the eigenvector field m (Fig. 5). If more than one edge is admissible for a given combination of eigenvector and orientation along this eigenvector then the one with the smallest deviation is chosen. The corresponding alignment force acting on vertex i is computed as follows

$$\mathbf{G}_{ij^*} = C_\theta \, (\mathbf{p}_j - \mathbf{p}_{j^*}) \tag{16}$$

where C_θ is a constant and $\mathbf{p}_{j^*} = \mathbf{p}_i + [(\mathbf{p}_j - \mathbf{p}_i).\mathbf{e}_m]\, \mathbf{e}_m$ is the projection of $\mathbf{p}_j$ along $\mathbf{e}_m$ (Fig. 5). This alignment constraint can be introduced as a modification to the relocation force and the vertex position update is then written as follows

$$\mathbf{p}_i^{n+1} = \mathbf{p}_i^n + \omega \left(\sum_{j \in \mathcal{N}_i^v} \mathbf{F}_{ij}^n + \sum_{j^* \in \mathcal{N}_i^{v*}} \mathbf{G}_{ij^*}^n \right) \tag{17}$$

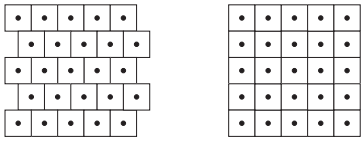

Fig. 4. Staggered (left) and aligned (right) square packing.

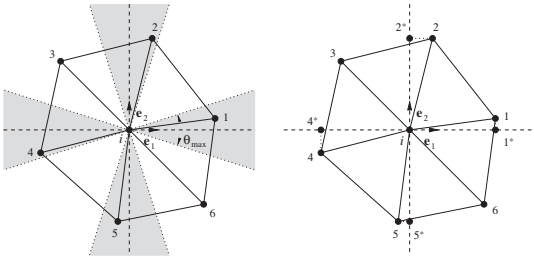

Fig. 5. Edge alignment.

where $\mathcal{N}_i^{v^*}$ is the set of links associated with i. Note that the constant C_θ must be chosen carefully. The greater C_θ, the better the alignment but also the stiffer the optimization problem will be. After some numerical experimentations, C_θ was set to 2 and $\theta_{\max}$ to $\pi/4$.

Reevaluated each time a vertex position is updated, those links play the role of a specialized reconnection operator. More than simplicial connectivity, those links indeed reflect the topology of the final cubical mesh. Because of the continuity of the metric, the end-to-end collection of links also forms tensor lines. Those lines are continuous as long as the metric is continuous and the prescribed element size is reasonable compared to the metric variation or tensor line curvature. Those active lines can also be considered as directional agglomerations of particles and their thickness is equal to the local target size prescribed by the metric. When links are set or broken then those lines are effectively fused or split. When particles are created, moved or destroyed so are those lines. The packing of particles also means that lines are packed and form layers upon layers. Furthermore, since particles are linked along both eigenvectors, those lines are also interconnected. The process can thus be interpreted as physically-based line networking or weaving.

3.3 Optimization Algorithm

Algorithm 1 sums up the main steps performed during the optimization process. A coarse initial triangulation is required as an input. The boundary of the domain must be represented by this triangulation and, when refining boundary edges, those constraints must be preserved. An empty constrained Delaunay triangulation is a good example of initial mesh. The algorithm is vertex-wise. It loops through all the vertices, moves them and enforces population control until convergence or a given number of iterations is completed. The threshold $l_{\min}$ is set to 0.75 while $l_{\max}$ is set to 1.33. The movement of a vertex due the attraction-repulsion as well as alignment forces is limited to the visibility zone of its immediate neighbor vertices, i.e., no inverted triangle should be created by the computed displacement. Any displacement that does not meet this requirement is discarded. Furthermore, after each successful relocation or refinement-coarsening operation, the local reconnection operator is called recursively to maintain the quality of the neighboring simplices. Note

Algorithm 1 Specialized simplicial optimization

Require: constrained empty triangulation
  repeat
    for all vertex i **do**
      update position
      if moved **then**
        reconnect neighborhood
      end if
      find is and il, the shortest and longest edges connected to i
      if $l_{is} < l_{\min}$ and (is on boundary or $\delta(\rho_i^-) \leq \delta(\rho_i)$) **then**
        collapse is
      else if $l_{il} > l_{\max}$ and (il on boundary or $\delta(\rho_i^+) \leq \delta(\rho_i)$) **then**
        split il
      end if
      if an edge has been collapsed or split **then**
        reconnect neighborhood
      end if
    end for
  until convergence

also that the metric is stored at each vertex of the triangulation. For new vertices or when vertices are moved, the metric is computed using an analytical function or by interpolation in a background mesh. In all edge-based computations, the metric is considered constant and the average of the values stored at the two end vertices is used.

Although similar in purpose, the present cubical packing force field was established by a very different reasoning and does not have the same shape than the one proposed in [3, 4]. An alignment force is also used to further promote the formation of continuous layers and ultimately increase the percentage of cubical elements in the final mesh. Additionally, like the pliant method proposed by Bossen and Heckbert [20], this process is recast as a simplicial mesh optimization and should eventually have the same computational efficiency. The problem has, however, more constraints and should thus take more time to converge. One potential difficulty is, as with other optimization methods, to get stuck in a local minimum. A possible approach to avoid such local minima is to start with a wider permissible edge length range and to slowly tighten it as the mesh is adapted. This results in a gradual and globally consistent evolution of the mesh that is more likely to reach a global minimum. An additional strategy is to first adapt the mesh using only attraction-repulsion forces combined with refinement-coarsening operations. The result is then used as an initial mesh for the stiffer problem with alignment forces. Both of these strategies are used for the test examples presented in the next section.

Once optimized, very little additional processing is needed to produce the final high quality quad-dominant mesh. Available edges for simplicial-to-cubical conversion are first identified. Forbidden edges are tensor line links and constrained boundary edges. For each of the available edges, the quality of the candidate quadrilateral element Q formed by the adjacent triangles is

computed as follows

$$\mathcal{Q}(Q) = \frac{2}{\sqrt{3}} \min_{1 \leq i \leq 4} \mathcal{Q}(T_i) \tag{18}$$

where T_i is the corner triangle associated with vertex i of Q. Edges with $\mathcal{Q} < 0.3$ are discarded. The remaining edges are listed in decreasing order according to this quality, the corresponding quadrilaterals are constructed and their edges removed from the list until it is empty.

4 Numerical Results

4.1 Academic Test Examples

The first example is defined on a $[0,\ 7] \times [0,\ 9]$ rectangular domain. The prescribed target sizes for this example are

$$h_1 = \begin{cases} 1 - \frac{19}{40} y & \text{if } y \in [0, 2]\,, \\ 20^{(2y-9)/5} & \text{if } y \in \left]2, \frac{9}{2}\right]\,, \\ 5^{(9-2y)/5} & \text{if } y \in \left]\frac{9}{2}, 7\right]\,, \\ \frac{1}{5} + \frac{1}{20}(y-7)^4 & \text{if } y \in]7, 9] \end{cases} \tag{19}$$

and $h_2 = 1.01\, h_1$ with $\mathbf{e}_1$ and $\mathbf{e}_2$ corresponding to the Cartesian axes. This is a quasi-isotropic version of the analytic metric from [1]. Purely isotropic metrics are indeed considered degenerate for tensorline-based cubical meshing because $\mathbf{e}_1$ and $\mathbf{e}_2$ would then be indefinite [5]. The resulting adapted triangular and quad-dominant meshes are presented in Fig. 6. The size variation prescribed by this metric does not allow an all-quadrilateral mesh but the algorithm is still able to directionally partition the domain and maximize the number of such elements, i.e., there is 84.8 percent of quadrilaterals. Table 1 gives some

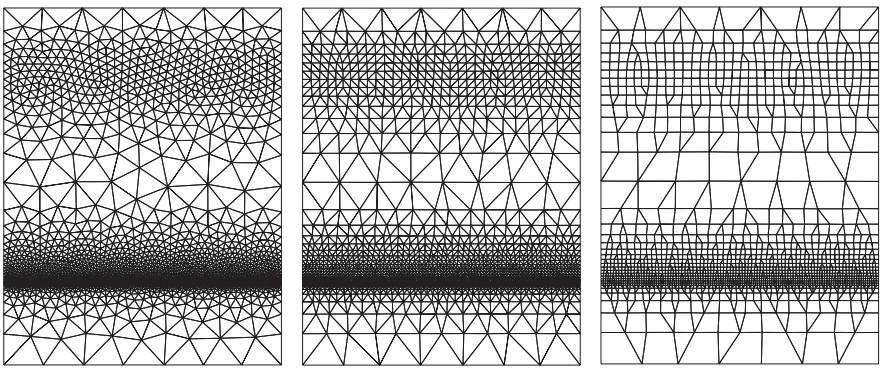

Fig. 6. Quasi-isotropic metric (Eq. 19) – Unit simplicial mesh obtained using classical L_2-based (left) and specialized L_∞-based operators (middle) as well as final quad-dominant mesh (right).

Table 1. Some statistics for the meshes presented in Figs. 6 to 10.

	Fig. 6	Fig. 7	Fig. 8	Fig. 9	Fig. 10
Nb edges	5268	2853	8895	11815	39917
Min. length	0.652	0.983	0.487	0.485	0.499
Ave. length	1.006	0.999	0.986	0.997	0.987
Max. length	1.364	1.026	1.890	1.686	1.778
Std dev.	0.071	0.005	0.096	0.070	0.094
Nb triangles	292	0	606	446	2100
Min. quality	0.703	-	0.501	0.507	0.454
Ave. quality	0.854	-	0.885	0.845	0.864
Max. quality	0.982	-	1.000	0.989	1.000
Std dev.	0.044	-	0.065	0.084	0.070
Nb quads.	1624	972	2671	3874	12307
Min. quality	0.530	0.667	0.310	0.353	0.304
Ave. quality	0.894	0.943	0.873	0.876	0.855
Max. quality	0.996	1.000	0.996	0.990	0.995
Std dev.	0.119	0.057	0.122	0.068	0.101
Quad-dom.	84.8%	100%	81.5%	89.7%	85.4%

statistics for this mesh as well as the other meshes presented in this section. These statistics include the number of edges and their metric length as well as the number of triangular and quadrilateral elements and their shape quality as computed with Eqs. 14 and 18. Note that, for comparison, a mesh obtained with classical operators is also presented in Fig. 6.

The next example is anisotropic and is defined on the same rectangular domain. Again the prescribed directions correspond to the Cartesian axes while the target sizes are computed as follows

$$h_1 = \begin{cases} 1 - \frac{19}{40}x & \text{if } x \in [0,2]\,, \\ 20^{(2x-7)/3} & \text{if } x \in \left]2, \frac{7}{2}\right]\,, \\ 5^{(7-2x)/3} & \text{if } x \in \left]\frac{7}{2}, 5\right]\,, \\ \frac{1}{5} + \frac{1}{20}(x-5)^4 & \text{if } x \in]5,7]\,, \end{cases} \quad h_2 = \begin{cases} 1 - \frac{19}{40}y & \text{if } y \in [0,2]\,, \\ 20^{(2y-9)/5} & \text{if } y \in \left]2, \frac{9}{2}\right]\,, \\ 5^{(9-2y)/5} & \text{if } y \in \left]\frac{9}{2}, 7\right]\,, \\ \frac{1}{5} + \frac{1}{20}(y-7)^4 & \text{if } y \in]7,9]\,. \end{cases} \tag{20}$$

The resulting mesh is presented in Fig. 7. Contrary to the first test case, a completely quadrilateral mesh is possible and the present method is able to generate it.

The final analytic test case is another quasi-isotropic metric, dubbed *banana*, that uses a size distribution taken from Lewis et al. [23] and is defined as follows

$$h_1 = \frac{1}{100}\left[1 + 30\,(y - x^2)^2 + (1-x)^2\right] \tag{21}$$

and $h_2 = 1.01\,h_1$ with $(x, y) \in [-1.25; 1.25] \times [-0.5; 1.25]$. The normalized gradient of the size distribution, i.e., $\nabla h_1 / \|\nabla h_1\|$, is taken as $\mathbf{e}_1$ while $\mathbf{e}_2$ is simply equal to $\mathbf{e}_1$ rotated by an angle of 90 degrees. The tensor line network

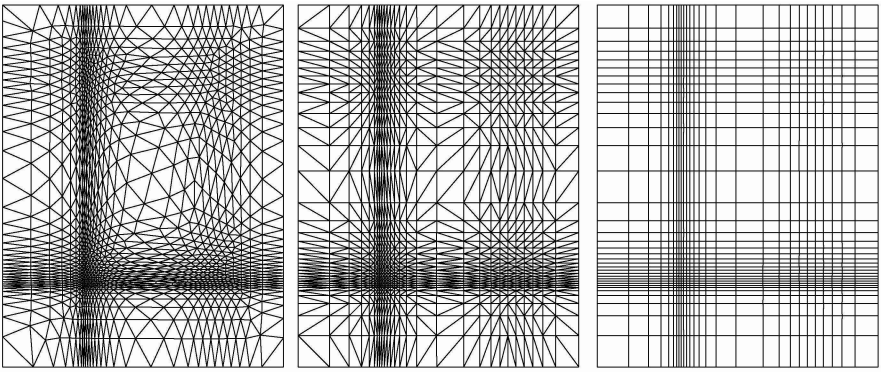

Fig. 7. Anisotropic metric (Eq. 20) – Unit simplicial mesh obtained using classical L_2-based (left) and specialized L_∞-based operators (middle) as well as final quadrilateral mesh (right).

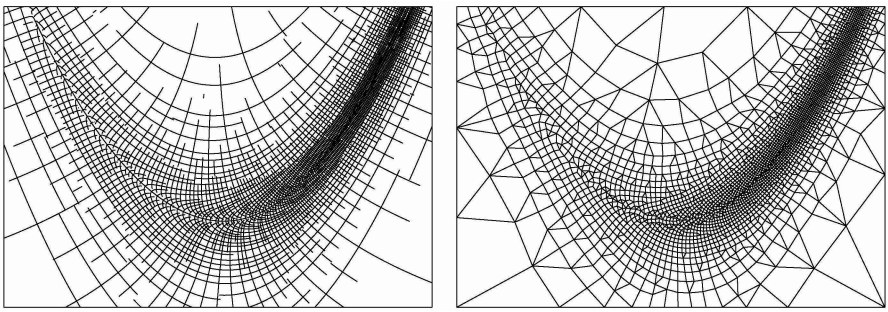

Fig. 8. Banana metric (Eq. 21) – Tensor line network (top) and corresponding adapted quad-dominant mesh (bottom).

for this metric is not aligned with the Cartesian axes as in the previous examples and is presented in Fig. 8 along with the final adapted quad-dominant mesh. The quality of the quadrilaterals in this mesh is substantially superior to the quality of the merged elements obtained by Borouchaki and Frey [18], i.e., an average of 0.87 versus 0.63, although the percentage of triangular elements is higher, i.e., 18.5 versus 3 percent (Table 1). Element shape quality is intentionally favored by the present method.

4.2 Practical Application Examples

The first practical application example is a mesh generated for a geometry-based metric. It shows how to use level set information encapsulated in a metric to generate high quality meshes. The considered domain is a heat exchanger and the metric is constructed using a variation of the ideas proposed in [24]. The vector $\mathbf{e}_1$ is set to the normalized gradient of ϕ, the distance-to-the-closest-boundary function, while $\mathbf{e}_2$ is again equal to $\mathbf{e}_1$ rotated by an angle of

90 degrees. The target size h_1 along $\mathbf{e}_1$ is limited by the local thickness of the domain. The function ϕ can be used to compute the medial axis of the domain and deduce this local thickness [24]. Furthermore, a user defined clustering has been added to the strictly geometric information extracted from ϕ. More precisely, the target size h_1, which is normal to the boundaries of the domain, is computed as follows

$$h_1 = \min\left(h_1^w + (\gamma - 1)\phi, h_\tau\right) \tag{22}$$

where h_1^w is the user prescribed size of the elements at the closest boundary, γ is the associated geometric growth ratio and h_τ is a limiting size computed from the local thickness τ of the domain. For the considered example, h_τ was set to 0.25τ. Along $\mathbf{e}_2$, the target size h_2 is computed as follows

$$h_2 = h_2^w(1 + |\kappa|\phi) \tag{23}$$

where κ is the curvature of the closest boundary and $h_2^w = 2\pi/N|\kappa|$. The user prescribed constant N corresponds to the number of elements needed to discretize a perfectly circular boundary. Furthermore, using Eq. 23 with adjacent circular and completely flat boundaries introduces very difficult size transitions and explicit gradation has to be used to ensure high quality meshes [25]. The maximum target size growth between adjacent elements, γ_0, was set to 1.2. Figure 9 shows the tensor line network for $N = 32$, $\gamma = 1.2$ and $h_1^w = 0.1r$ with r being the radius of the internal circular boundaries. The resulting adapted mesh is also presented in Fig. 9 and counts almost 90 percent of quadrilaterals with an average quality of 0.876 (Table 1).

The final example uses a solution-based metric extracted from the computed flow around a NACA 0012 airfoil using an Hessian-based error estimator

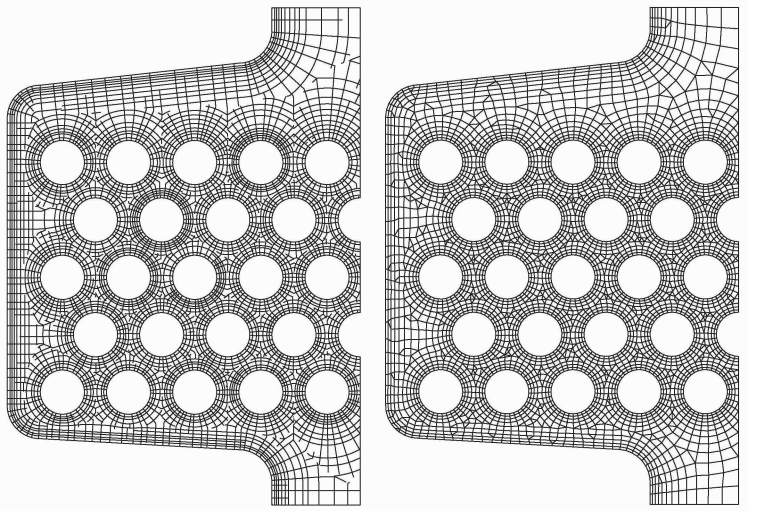

Fig. 9. Heat exchanger – Tensor line network (left) and corresponding quad-dominant meshes (right).

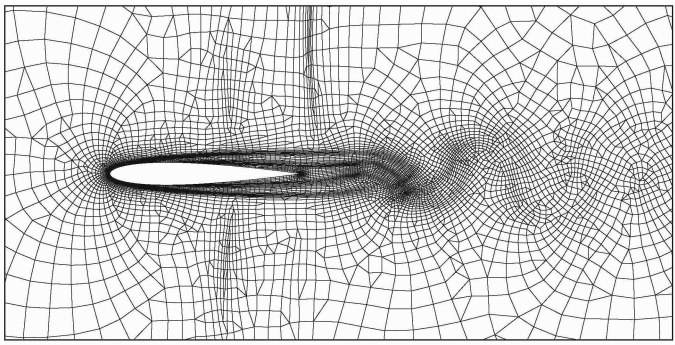

Fig. 10. Naca 0012 airfoil – Quad-dominant mesh adapted to an unsteady flow at a Mach number of 0.85 and a Reynolds number of 5000.

[26]. An unsteady laminar transonic flow with a Reynolds number of 5000 and a Mach number of 0.85 was considered. Again, gradation with $\gamma_0 = 1.2$ was used. The resulting mesh is presented in Fig. 10 and counts about 85 percent of quadrilaterals with an average quality of 0.855 (Table 1).

4.3 Future Developments

The results presented above show that the proposed algorithm is able to generate high quality quad-dominant meshes. However, to obtain high percentages of quadrilateral elements, the metric must prescribe smoothly varying target sizes. Just as for triangles, there is a physical limit on how fast target size can vary to get well shaped quadrilaterals. A specialized gradation algorithm must be developed as already suggested in [5]. All-cubical meshes are also possible if this gradation algorithm can provide compatible metrics. The problem of obtaining such meshes is thus displaced from the actual generation task to the metric construction task. This is hopefully a simpler problem.

The more compatible the metric, the easier it is to obtain a global minimum. However, even for a completely compatible metric such as the one defined by Eq. 20, local minima are possible and this implies the presence of unnecessary triangles in the final mesh. To avoid such a local minimum, the following two-step strategy was used. The mesh is first adapted without alignment forces. Three passes are used: 200 iterations with $l_{\min} = 0.75$ and $l_{\max} = 2.66$ then 200 iterations with $l_{\max}$ reduced to 2.00 and finally 200 iterations with $l_{\max} = 1.33$. The second step uses the result as an initial mesh and performs an additional 200 iterations with alignment forces, $l_{\min} = 0.75$ and $l_{\max} = 1.33$. This fixed number of iterations is probably overkill but the goal of the present work is to prove the soundness of the proposed specialized operators and not yet their efficiency. Given those considerations, CPU timings are not very meaningful but, to give an order of magnitude, the quadrilateral mesh presented in Fig. 7 took 101.62 seconds on an Intel Pentium M running at 1.3GHz. There is, furthermore, little overhead per iteration per vertex compared to classical operators. The real increase in computer time is due to

the number of iterations needed to converge to a more difficult problem. An optimized iterative process will also be the goal of future developments.

Finally a three-dimensional extension is very possible and more practical than tensor surface networks. Simplicial-to-cubical conversion templates indeed already exist [4, 17] and classical simplicial reconnection operators can be used in a future three-dimensional optimizer. The generalization of all the other operators presented in Sect. 3 is mostly trivial.

5 Conclusion

The proposed specialized simplicial optimization is capable of automatically generating high quality quad-dominant meshes with a layered structure aligned along the local eigenvectors of an anisotropic Riemannian control metric. This confirms the soundness of the L_∞-based simplicial operators and the angle-based alignment force. The computational efficiency is not yet established but is expected to be similar to classical simplicial optimization. Only two-dimensional cases were considered in the present paper but a three-dimensional extension is feasible. Besides cubical meshes, the present method can also provide simplicial mesh postprocessing when right angle simplices are required. All-cubical meshes are also possible if the metric is compatible. Automatic cubical mesh generation can thus be achieved using a two-prong approach: the construction of a compatible metric and the generation of the corresponding mesh. The present paper deals with the second part. Future developments should take care of the first part.

References

1. George, P.-L. and Borouchaki, H. (1998) Delaunay Triangulation and Meshing. Applications to Finite Elements. Hermès, Paris.
2. Blacker, T. (2001) Automated conformal hexahedral meshing constraints, challenges and opportunities. Engineering with Computers, 17:201–210.
3. Shimada, K., Liao, J.-H., and Itoh, T. (1998) Quadrilateral meshing with directionality control through the packing of square cells. Seventh International Meshing Roundtable, Dearborn, MI, Oct., pp. 61–76, Sandia National Laboratories.
4. Yamakawa, S. and Shimada, K. (2003) Fully-automated hex-dominant mesh generation with directionality control via packing rectangular solid cells. Int. J. Numer. Meth. Engng, 57:2099–2129.
5. Tchon, K.-F., Guibault, F., Dompierre, J., and Camararo, R. (2005) Adaptive hybrid meshing using metric tensor line networks. 17th AIAA Computational Fluid Dynamics Conference, Toronto, ON, Canada, Jun., no. AIAA-2005-5332.
6. Vallet, M.-G. (1992) Génération de maillages éléments finis anisotropes et adaptatifs. Ph.D. thesis, Université Pierre et Marie Curie, Paris VI, France.
7. Simpson, R. B. (1994) Anisotropic mesh transformations and optimal error control. Applied Numerical Mathematics, 14:183–198.
8. Thompson, J. F., Warsi, Z. U. A., and Mastin, C. W. (1985) Numerical grid generation: Foundations and applications. North-Holland.

9. Schneiders, R. (2000) Octree-based hexahedral mesh generation. Int. J. of Comp. Geom. & Applications, 10:383–398.
10. Maréchal, L. (2001) A new approach to octree-based hexahedral meshing. Tenth International Meshing Roundtable, Newport Beach, CA, Oct., pp. 209–221, Sandia National Laboratories.
11. Tchon, K.-F., Dompierre, J., and Camarero, R. (2004) Automated refinement of conformal quadrilateral and hexahedral meshes. Int. J. Numer. Meth. Engng, 59:1539–1562.
12. Benzley, S. E., Harris, N. J., Scott, M., Borden, M., and Owen, S. J. (2005) Conformal refinement and coarsening of unstructured hexahedral meshes. Journal of Computing and Information Science in Engineering, 5:330–337.
13. Bern, M., Eppstein, D., and Erickson, J. (2002) Flipping cubical meshes. Engineering with Computers, 18:173–187.
14. Owen, S. J., Staten, M. L., Canann, S. A., and Saigal, S. (1999) Q-morph: An indirect approach to advancing front quad meshing. Int. J. Numer. Meth. Engng, 44:1317–1340.
15. Owen, S. J. and Saigal, S. (2000) H-morph: An indirect approach to advancing front hex meshing. Int. J. Numer. Meth. Engng, 49:289–312.
16. Lee, Y. K. and Lee, C. K. (2003) A new indirect anisotropic quadrilateral mesh generation scheme with enhanced local mesh smoothing procedures. Int. J. Numer. Meth. Engng, 58:277–300.
17. Meshkat, S. and Talmor, D. (2000) Generating a mixed mesh of hexahedra, pentahedra and tetrahedra from an underlying tetrahedral mesh. Int. J. Numer. Meth. Engng, 49:17–30.
18. Borouchaki, H. and Frey, P. J. (1998) Adaptive triangular-quadrilateral mesh generation. Int. J. Numer. Meth. Engng, 41:915–934.
19. Shimada, K. and Gossard, D. (1995) Bubble mesh: Automated triangular meshing of non-manifold geometry by sphere packing. ACM Third Symposium on Solid Modeling and Applications, Salt Lake City, UT, May, pp. 409–419.
20. Bossen, F. J. and Heckbert, P. S. (1996) A pliant method for anisotropic mesh generation. Fifth International Meshing Roundtable, Pittsburgh, PA, Oct., pp. 63–76, Sandia National Laboratories.
21. Li, X., Shephard, M. S., and Beall, M. W. (2003) Accounting for curved domains in mesh adaptation. Int. J. Numer. Meth. Engng, 58:247–276.
22. Farhat, C., Degand, C., Koobus, B., and Lesoinne, M. (1998) Torsional springs for two-dimensional dynamic unstructured fluid meshes. Computer Methods in Applied Mechanics and Engineering, 163:231–245.
23. Lewis, R. W., Zheng, Y., and Usmani, A. S. (1995) Aspects of adaptive mesh generation based on domain decomposition and Delaunay triangulation. Finite Elements Anal. Des., 20:47–70.
24. Tchon, K.-F., Khachan, M., Guibault, F., and Camarero, R. (2005) Three-dimensional anisotropic geometric metrics based on local domain curvature and thickness. Comp.-Aided Design, 37:173–187.
25. Li, X., Remacle, J.-F., Chevaugeon, N., and Shephard, M. S. (2004) Anisotropic mesh gradation control. Thirteenth International Meshing Roundtable, Williamsburg, VA, Sep., pp. 401–412, Sandia National Laboratories.
26. Habashi, W. G., Dompierre, J., Bourgault, Y., Aït-Ali-Yahia, D., Fortin, M., and Vallet, M.-G. (2000) Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD. Part I: General principles. Int. J. Numer. Meth. Fluids, 32:725–744.