

---

**Conformal Refinement of All-Quadrilateral and All-Hexahedral  
Meshes According to an Anisotropic Metric**

Ko-Foa Tchou   Julien Dompierre   Ricardo Camarero

Centre de recherche en calcul appliqué (CERCA)  
5160, boul. Décarie, bureau 400, Montréal, Québec, H3X 2H9, Canada

[tchou|julien|ricardo]@cerca.umontreal.ca

Rapport CERCA R2002-??

1er août 2002

Présenté à la

*11th International Meshing Roundtable,*

Ithaca, New York, É.-U.

15–18 septembre 2002.



# CONFORMAL REFINEMENT OF ALL-QUADRILATERAL AND ALL-HEXAHEDRAL MESHES ACCORDING TO AN ANISOTROPIC METRIC

Ko-Foa Tchou   Julien Dompierre   Ricardo Camarero

*Centre de recherche en calcul appliqué (CERCA)  
5160, boul. Décarie, bureau 400, Montréal, QC, H3X 2H9, Canada.  
[tchon|julien|ricardo]@cerca.umontreal.ca*

## ABSTRACT

Conformal refinement using a shrink and connect strategy known as pillowing or buffer insertion contracts contiguous elements of an all-quadrilateral or an all-hexahedral mesh and reconnects them to locally increase vertex density. Using layers as shrink sets, the present method anisotropically refines an initial mesh to match a prescribed size map expressed as a metric field. An anisotropic smoother further enhances vertex clustering to capture the features of the metric. Both two and three-dimensional test cases confirm the feasibility of such an approach. The present refinement method is not yet complete, however, because it lacks local vertex removal and reconnection capabilities. It is, nevertheless, a step towards an automated tool for conformal adaptation of all-quadrilateral and all-hexahedral meshes.

**Keywords:** unstructured mesh, quadrilateral, hexahedral, conformal refinement, anisotropic metric.

## 1. INTRODUCTION

*M*ESH adaptation is used to accurately compute the numerical solution to discretized partial differential equations. Adaptation methods for unstructured meshes of simplicial elements, triangles in two dimensions and tetrahedra in three dimensions, have received extensive attention over the years. These methods are based on mesh density control by vertex insertion and removal, i.e., mesh refinement and coarsening, and mesh quality control by vertex relocation and reconnection. See [1], [2] or [3] and the references cited therein for example. The Object Oriented Remeshing Toolkit (*OORT*) developed at CERCA implements such methods [4].

However, for meshes of non-simplicial elements, such as quadrilaterals in two dimensions and hexahedra in three dimensions, adaptation methods are usually limited to vertex relocation to preserve their implicit grid structure. With such a limitation, the full power of mesh optimization cannot be unlocked [5]. The advent of tools such as paving [6], plastering [7], whisker weaving [8], medial axis and surface [9], embedded Voronoi graph [10] or grid-based methods [11–14] for unstructured quadrilateral and hexahedral mesh generation brought the need for a new crop of adaptation methods. Recombination of quadrilaterals from adapted triangles [15, 16] is particularly attractive but an extension in three dimensions has not yet been developed. On the other

hand, iterative local refinement methods using quadtrees in two dimensions can be extended in three dimensions using octrees but result in non-conformal elements with hanging nodes that cannot be treated by some solvers. See [12] for example. Although all-quadrilateral transition templates to remove those hanging nodes are readily available for quadtrees [17], it is only recently that all-hexahedral transition patterns have been developed for octrees [13, 14]. The directional refinement method of Schneiders uses an octree to insert conformal buffer layers in alternating  $x$ ,  $y$ , and  $z$  directions [13]. The resulting mesh is all-hexahedral but the initial octree must be Cartesian. The clever octree conforming method described by Maréchal reduces the set of transition templates to only two but results in hybrid meshes mixing hexahedral and prismatic elements [14]. This method does not require the initial octree to be Cartesian but, to obtain an all-hexahedral mesh, a final uniform refinement step dicing each hexahedron in eight and each prism in six is needed.

The present paper introduces an alternative method for conformal anisotropic all-quadrilateral and all-hexahedral mesh refinement. This method builds on the directional refinement idea of Schneiders. However, instead of using a quadtree or an octree to drive the process, the present method searches for layers of elements to refine according to a prescribed Riemannian metric, i.e., a size specification map. Contracting these layers and reconnecting them to the mesh results in local anisotropic refinements. This shrink and connect strat-

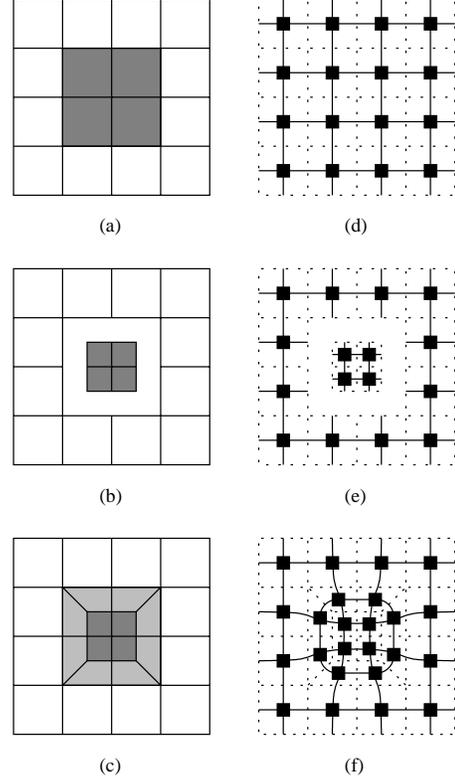
egy known as pillowing is iterated until the mesh satisfies the prescribed size map. The resulting algorithm is similar to the directional buffer insertion strategy of Schneiders but, since no Cartesian quadtree or octree is needed, it can be used on initially unstructured meshes. It can also be combined with vertex relocation methods between each refinement iteration resulting in a faster convergence and a better conformity to the desired mesh density, stretching and orientation.

This refinement algorithm has been implemented in **OORT** and takes full advantage of the adaptation utilities already available in this package. Before describing the details of the algorithm, the next two sections present the conformal refinement primitive, based on a shrink and connect strategy, and the anisotropic control metric. Experiments with two and three-dimensional test cases then show the capabilities of the method as well as its current shortcomings to be addressed in future work.

## 2. CONFORMAL REFINEMENT PRIMITIVE

The present refinement primitive is based on a shrink and connect strategy and has been extensively used, although under several names, because it is one of the few local operations that preserves the conformity and the all-quadrilateral or all-hexahedral characteristic of a mesh. Mitchell and Tautges [18] dubbed it pillowing and used it to remove degenerated configurations resulting from plastering [7] and whisker weaving [8]. The cleave and fill tool of Borden et al. [19] uses pillowing to locally refine all-hexahedral swept meshes. The directional buffer insertion of Schneiders [13] is also a shrink and connect method and, in fact, inspired the present algorithm. Boundary mesh extrusions used to remove degenerated configurations resulting from the body fitting step of grid-based or octree-based hexahedral mesh generators [11–14] can also be interpreted as shrink and connect methods. Figure 1 illustrates this refinement primitive in two dimensions. First, a set of contiguous elements is selected (Fig. 1(a)), those elements are then shrunk (Fig. 1(b)) and, finally, the void thus created is filled by a buffer connecting the shrunk elements to the remainder of the mesh (Fig. 1(c)). Note that all the vertices located on the hull of the selected elements are duplicated. The shrink and connect strategy is, therefore, a vertex insertion or refinement primitive. Note also that the buffer is in fact an extrusion of this hull and is exclusively composed of quadrilateral elements. This primitive is thus conforming and generates only quadrilaterals in two dimensions. Most interestingly, it keeps those properties when extended in three dimensions. The hull of the selected elements is then a quadrilateral mesh that, when extruded, generates only hexahedral elements.

As noted by Mitchell and Tautges [18] and by Schneiders [13], shrink and connect strategies, such as pillowing or buffer insertion, can be viewed as spatial twist continuum (STC) operations. STC is an interpretation of the dual of a quadrilateral or hexahedral mesh [20]. This dual has a vertex for every mesh element and an edge for every mesh edge in two dimensions or mesh face in three dimensions. The present refinement primitive corresponds to the insertion of dual cycles around the shrunk elements. These cycles, loops in two dimensions and shells in three dimensions, are the



**Figure 1: Conformal refinement primitive (a–c) and its dual interpretation (d–f).**

dual of the buffer layers connecting the shrunk elements to the remainder of the mesh. Figures 1(d)–1(f) are the dual interpretations of Figs. 1(a)–1(c) respectively.

## 3. ANISOTROPIC CONTROL METRIC

To determine where to apply the local refinement primitive, an anisotropic control map must not only prescribe the size but also the stretching and orientation of the mesh elements to be built. These specifications are given here as the metric of the transformation that maps a perfect mesh element, a quadrilateral in two dimensions or an hexahedron in three dimensions, into a unit square or a unit cube. This Riemannian metric is defined at every point of the domain by a symmetric positive-definite matrix  $\mathcal{M}$ ,  $2 \times 2$  in two dimensions and  $3 \times 3$  in three dimensions. The length  $l_{AB}$  of an edge between point  $A$  and point  $B$  in this metric is given by

$$l_{AB} = \int_0^1 \sqrt{\vec{A}\vec{B}^T \mathcal{M}(\vec{P}) \vec{A}\vec{B}} dt \quad (1)$$

where  $\vec{P} = \vec{A} + t(\vec{B} - \vec{A})$ . Such a size specification map can be given analytically or constructed from a *posteriori* error analysis, from the geometrical properties of the domain or from any other user defined inputs. An isotropic size specification map reduces to an identity matrix multiplied by  $h^{-2}$  where  $h$  is the desired element size. Whatever it's origin, the control metric contains information on the pre-

scribed size, stretching and orientation of the mesh to be built as an anisotropic metric field. See [21] as well as [1] and [2] for a more complete discussion on metrics.

To identify mesh elements to be refined, we thus compute the length of their edges in the control metric using Relation (1). If an edge is shorter than unity in our control space then the mesh is already locally too dense and no refinement is needed. If an edge is longer than unity then the mesh is locally too coarse and the connected elements are processed by the anisotropic refinement algorithm.

#### 4. ANISOTROPIC REFINEMENT ALGORITHM

The control metric locates regions to refine and the refinement primitive gives a method to insert additional vertices. However, to achieve the desired anisotropic effect, a strategy is needed to group elements in sets that, once shrunk, will directionally enrich the mesh. Following Schneiders [13], the present method uses layers of contiguous elements as shrink sets. Figure 2 illustrates the successive steps of the algorithm in two dimensions:

1. Compute the metric length of every edge of the mesh. If it is significantly higher than the unit target, the edge is tagged for refinement (Fig. 2(a)). The tagging threshold is typically chosen to be around 2.0. This is higher than the 1.4 value used in **OORT** for simplex adaptation because the present refinement operator has essentially a 1-to-3 refinement ratio while the edge splitting operator used with simplices has a 1-to-2 ratio. Note that tagged edges are plotted thicker in Fig. 2.
2. Form a layer by agglomerating contiguous elements connected by opposite edges tagged for refinement (Fig. 2(b)). In three dimensions, the edges of a hexahedron can be grouped in three sets, along the three local axes of the element. Edges in each of these sets are considered opposite to each other. The three-dimensional layer is propagated perpendicularly to those edges (Fig. 3). Note that, as shown in Fig. 2(b), the layers are terminated by elements having only one of their edges tagged in the propagation direction.
3. Shrink and connect this layer (Fig. 2(c)). If the layer extends up to the boundary of the domain, the refinement primitive considers the boundary edges to be included in the shrink set. The refinement tags are removed for all the edges used to form the layer. On the other hand, the edges located on the hull of the shrink set are duplicated during the refinement process and each newly created edge inherits the tag of its parent, if any, except on corners. Corner edges are defined in two dimensions as contiguous hull edges connected to the same layer element. A similar definition can be given in three dimensions. Figure 4 presents additional examples of shrink sets and their treatment by the refinement algorithm for less obvious cases such as loops and self intersecting layers.
4. Repeat steps 2 and 3 until there is no more edges tagged for refinement (Figs. 2(d)–2(f)). Note that

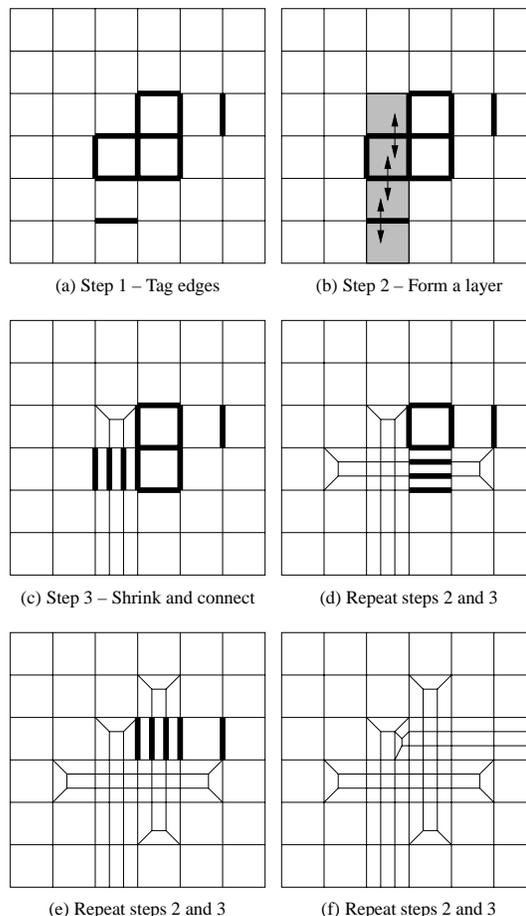


Figure 2: Refinement algorithm steps.

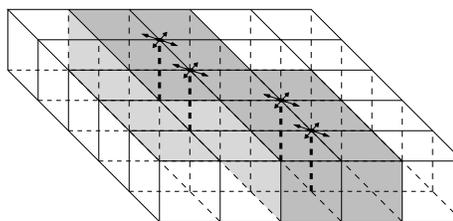
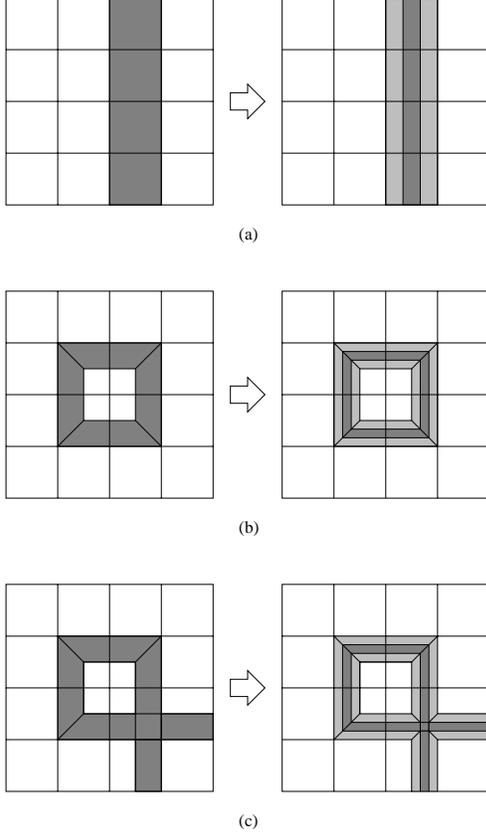


Figure 3: Layer agglomeration in three dimensions.

the corner configurations located at the extremities of the layers degrade mesh quality. This phenomenon is even more acute when several layers terminate at the same location (Fig. 2(f)). See below for strategies used to minimize those configurations.

5. Repeat steps 1 through 4 until all the edges of the mesh have a metric length below the chosen threshold.

The resulting algorithm was implemented in **OORT** and uses the many adaptation utilities provided by this package such as all the metric related operations. Note that a major



**Figure 4: Examples of shrink layers: (a) whole row; (b) loop; (c) self-intersecting layer.**

concern during the development of the present algorithm has been maintaining an acceptable mesh quality. Internal vertices in a perfect quadrilateral mesh have a valence of four, i.e., each vertex is connected to four other vertices. In a perfect hexahedral mesh, those vertices have a valence of six. Starting from such a perfect mesh, the present refinement primitive will modify this valence and tend to degrade mesh quality, particularly at the corners of the shrunk layers. That is why, to minimize this side effect, edge tags are propagated in step 1 using a strategy similar to quadtree and octree balancing. If a quadrilateral element has more than two edges tagged then all its edges are tagged. Similar rules are applied for hexahedra. Furthermore, layers are augmented in the direction perpendicular to the refinement by their immediate neighbors in order to smooth out abrupt size variations.

Additional quality improvement methods are still being explored. Mesh adaptation by vertex relocation, i.e., anisotropic smoothing, can be performed after step 4 of the algorithm to improve mesh quality and to align element rows with the features of the metric to be captured. The combination of anisotropic refinement and smoothing should have a synergic effect on the convergence of the adaptation process. The thickness of the layers can also be increased. The layers shrunk by Schneiders have a thickness of two elements [13] which is twice what is used in the present algorithm. Alternatively, the edges of the mesh can be adapted to  $n$  times the

actually required length and then each of its  $d$ -dimensional elements can be diced in  $n^d$  pieces. The rationale behind this strategy is again to decrease the percentage of corner configurations resulting from refinement but also to smooth out element size variations. Results using each of these methods are presented in the next section.

## 5. RESULTS AND DISCUSSION

To illustrate the capabilities of the present method, only academic test cases with analytical metrics are presented hereafter. The first goal is, indeed, to assess how close to the prescribed metric the present algorithm can adapt a quadrilateral or hexahedral mesh. A second goal is to determine which additional mesh quality improvement method is effective. We hope to tackle more realistic test cases in the next development stage of this method.

### 5.1. Two-Dimensional Test Cases

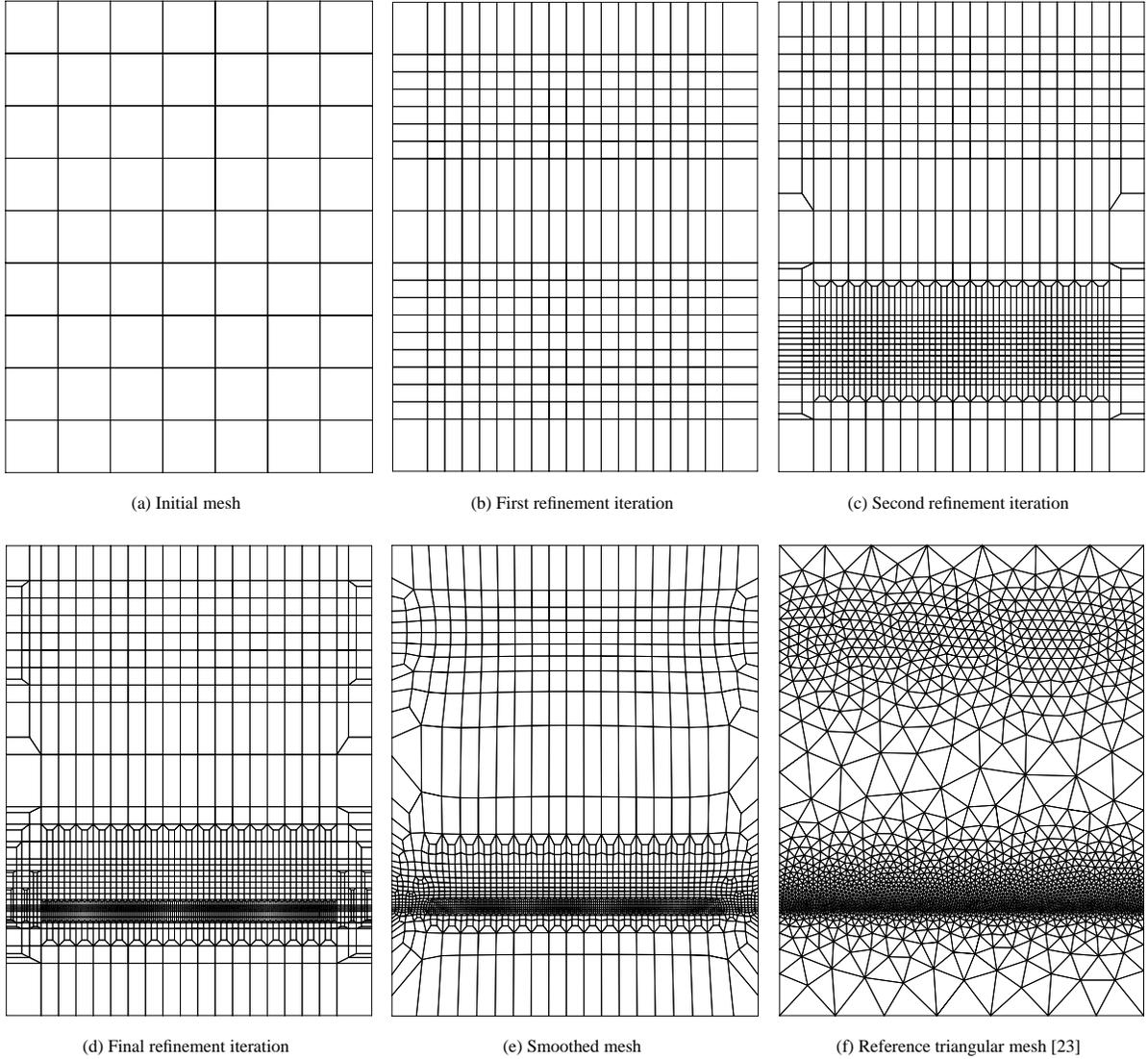
The first two cases are typically used to test triangular mesh adaptation [1]. The first one has an isotropic metric defined on a  $[0, 7] \times [0, 9]$  rectangular domain by

$$\mathcal{M} = \begin{pmatrix} h_1^{-2} & 0 \\ 0 & h_2^{-2} \end{pmatrix} \quad (2)$$

where  $h_1 = h_2 = h(x, y)$  is given by

$$h(x, y) = \begin{cases} 1 - 19y/40 & \text{if } y \in [0, 2], \\ 20^{(2y-9)/5} & \text{if } y \in ]2, 4.5], \\ 5^{(9-2y)/5} & \text{if } y \in ]4.5, 7], \\ 1/5 + (y-7)^4/20 & \text{if } y \in ]7, 9]. \end{cases} \quad (3)$$

A uniform initial mesh with  $7 \times 9$  square elements was used for convenience. Figures 5(a) to 5(d) present the evolution of the refinement process. Note that, although we start from a uniform Cartesian mesh, it becomes unstructured just after a few iterations and the algorithm is, therefore, effectively applied on irregular meshes. Figure 5(e) presents the final mesh after adaptation by the vertex relocation algorithm implemented in **OO $\mathcal{R}\mathcal{T}$** . This algorithm basically smoothes the mesh in the Riemannian metric space [22]. For comparison, Figure 5(f) presents a triangular mesh adapted to the same metric by **OO $\mathcal{R}\mathcal{T}$**  [23] and gives a visual clue of the features of the metric. As can be seen in those figures, the refined mesh is clustered around  $y = 2$  and  $y = 7$  both horizontally and vertically. In contrast, adaptation by vertex relocation only of a structured quadrilateral grid could achieve some clustering along the  $y$  axis but not along the  $x$  axis: it would always have either too many or too few vertices somewhere in the grid. Table 1 compares the minimum, the average  $\mu$ , the maximum and the standard deviation  $\sigma$  of the metric edge length  $l$  and the metric element shape measure  $\eta$  associated with the mesh before and after refinement. The metric length  $l$  is computed using Relation (1) for each edge and the corresponding entries  $Nb$  in Table 1 indicate the total number of edges in the mesh. The measure  $\eta$  is computed as the worst metric shape of the corner simplices of each element. The corresponding entries  $Nb$  in Table 1



**Figure 5: Mesh obtained using the isotropic metric given by Eqs. (2) and (3).**

thus indicate the total number of elements in the mesh. The metric shape of a  $d$ -dimensional corner simplex  $K$  is given by

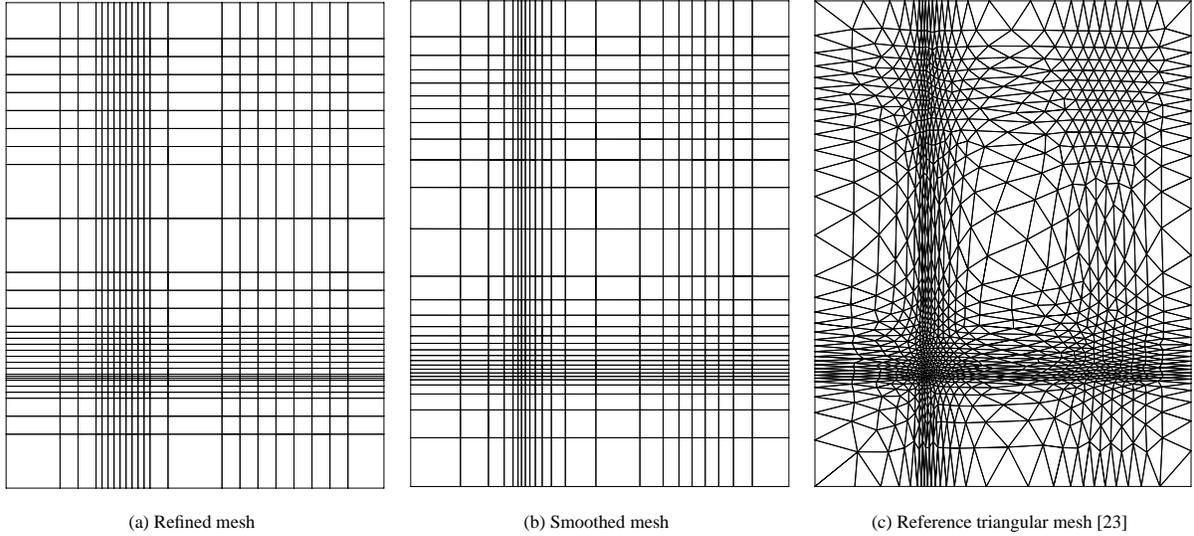
$$\eta = CV_K^{2/d} \left/ \sum_{i=1}^{n_e} l_i^2 \right. \quad (4)$$

where  $C$  is a normalization factor such as  $\eta$  ranges from 0 for a degenerated simplex to 1 for a perfectly right-angle unit corner,  $n_e$  is the number of edges of  $K$ ,  $l_i$  is the metric length of edge  $i$  and the area or volume of a simplex  $K$  in a metric  $\mathcal{M}$  is given by  $V_K = \int_K \sqrt{\det(\mathcal{M})} dV$ . Note that  $\eta$  was chosen because it measures the shape in a Riemannian metric and thus better reflects how well the final mesh conforms to the prescribed metric size map than the usual shape measures computed in Euclidean space. See [24] for further information on element shape measures evaluated with a metric.

In a mesh perfectly adapted to a metric, all the edges have a unit metric length and all the elements have a unit metric shape. Although the refined mesh in Fig. 5(e) does not perfectly match the prescribed metric, Table 1 shows that the average edge length is very close to unity instead of the ini-

**Table 1: Quality indicators evaluated in the isotropic metric given by Eqs. (2) and (3) for the meshes of Fig. 5.**

Indicator	Nb	min	$\mu$	max	$\sigma$
$l_{\text{ini}}$ (Fig. 5(a))	142	1.000	4.205	20.0	1.042
$\eta_{\text{ini}}$	63	0.179	0.642	1.04	0.398
$l_{\text{fin}}$ (Fig. 5(e))	4427	0.321	0.919	1.88	0.303
$\eta_{\text{fin}}$	2182	0.271	0.782	0.98	0.247
$l_{\text{ref}}$ (Fig. 5(f))	6326	0.789	0.998	1.31	0.076
$\eta_{\text{ref}}$	4189	0.880	0.982	1.00	0.018



**Figure 6: Mesh obtained using the anisotropic metric given by Eqs. (2) and (5).**

tial value of 4.205 and the standard deviation was divided by three. A simplicial adaptor gives an even better average edge length and element shape and, most importantly, a much smaller standard deviation, almost one order of magnitude smaller as can be seen in Table 1. This was to be expected since the present algorithm is mainly for refinement and cannot coarsen nor reconnect the mesh contrary to simplicial algorithms. It has, therefore, less flexibility in controlling vertex density and cannot achieve the same level of performance except for special cases.

The second two-dimensional test is such a special case. Its anisotropic metric is defined by Eq. (2) where  $h_1$  and  $h_2$  are given by

$$\begin{aligned}
 h_1 &= \begin{cases} 1 - 19x/40 & \text{if } x \in [0, 2], \\ 20^{(2x-7)/3} & \text{if } x \in ]2, 3.5], \\ 5^{(7-2x)/3} & \text{if } x \in ]3.5, 5], \\ 1/5 + (x-5)^4/20 & \text{if } x \in ]5, 7], \end{cases} \\
 h_2 &= \begin{cases} 1 - 19y/40 & \text{if } y \in [0, 2], \\ 20^{(2y-9)/5} & \text{if } y \in ]2, 4.5], \\ 5^{(9-2y)/5} & \text{if } y \in ]4.5, 7], \\ 1/5 + (y-7)^4/20 & \text{if } y \in ]7, 9]. \end{cases}
 \end{aligned} \tag{5}$$

**Table 2: Quality indicators evaluated in the anisotropic metric given by Eqs. (2) and (5) for the meshes of Fig. 6.**

Indicator	Nb	min	$\mu$	max	$\sigma$
$l_{\text{ini}}$	142	1.275	3.875	11.7	0.703
$\eta_{\text{ini}}$	63	0.229	0.730	1.00	0.280
$l_{\text{fin}}$ (Fig. 6(b))	1372	1.181	1.197	1.21	0.012
$\eta_{\text{fin}}$	660	0.953	0.993	1.00	0.007
$l_{\text{ref}}$ (Fig. 6(c))	3238	0.768	1.022	1.35	0.071
$\eta_{\text{ref}}$	2119	0.806	0.985	1.00	0.020

The same domain and initial mesh as in the isotropic case are used. Figures 6(a) and 6(b) present the refined mesh before and after smoothing respectively. For comparison, Figure 6(c) presents a triangular mesh adapted to the same metric by **OO** $\mathcal{RT}$  [23]. The corresponding quality indicators are given in Table 2. These indicators are much closer to unity than in the first case with a very small standard deviation indicating a high degree of adaptation bettering what can be achieved with simplices. This is, however, due to the alignment of the features to be captured with the rows of the initial mesh.

To see how the algorithm fairs on non-mesh-aligned features, the third test case has the following metric defined on a  $[-1, 1] \times [-1, 1]$  square domain

$$\mathcal{M} = \mathcal{R}_\theta \begin{pmatrix} h_1^{-2} & 0 \\ 0 & h_2^{-2} \end{pmatrix} \mathcal{R}_\theta^{-1} \tag{6}$$

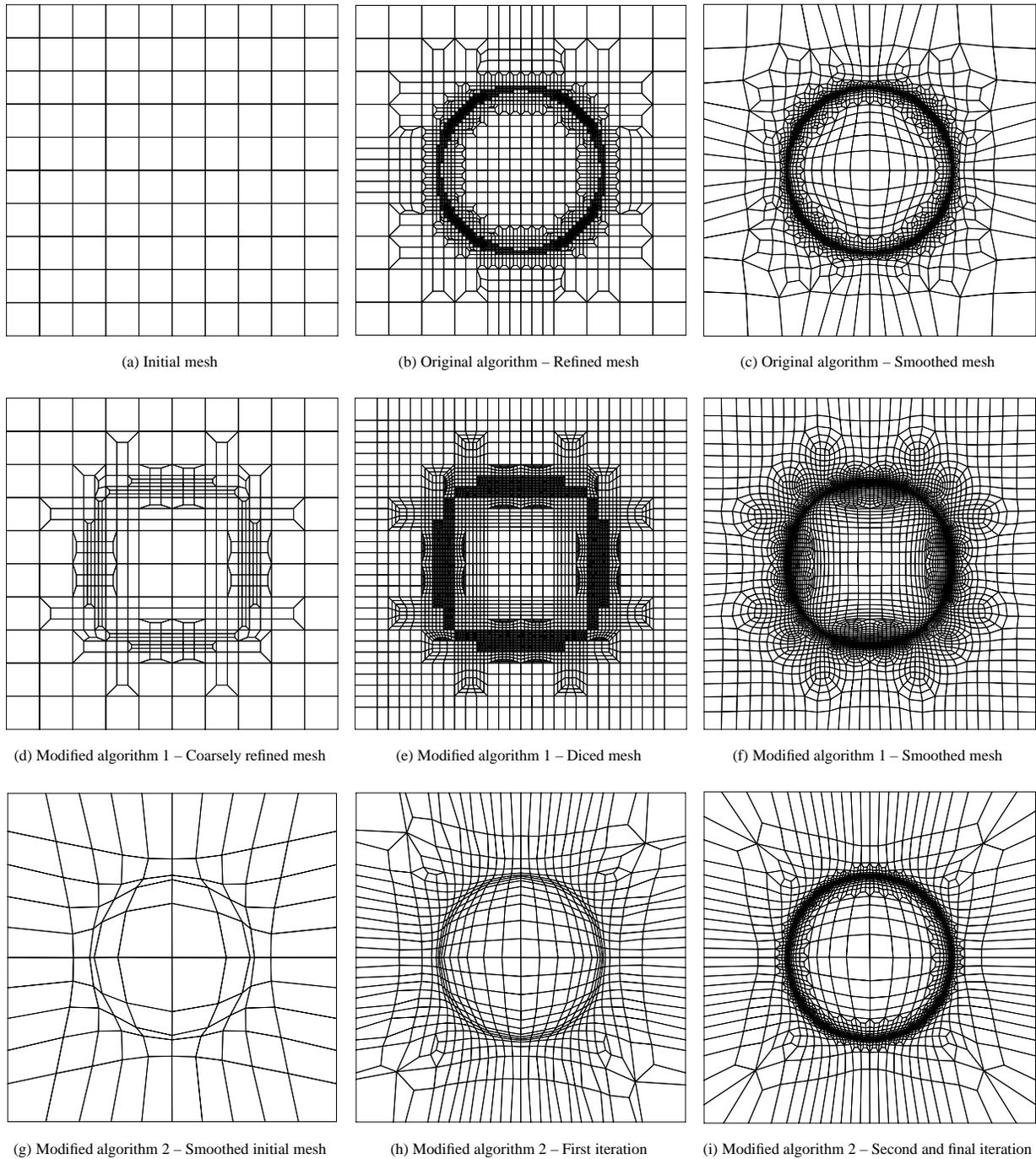
where  $\mathcal{R}_\theta$  is a rotation matrix

$$\mathcal{R}_\theta = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \tag{7}$$

$$\text{and } \begin{cases} \theta = \arctan(y/x), \\ h_1 = r/120 + |1-r|/2, \\ h_2 = r/30 + |1-r|/2. \end{cases} \tag{8}$$

with  $r = \sqrt{x^2 + y^2}$ . This metric was inspired by a test case used in [25] and prescribes stretched quadrilaterals aligned along a 0.5 radius circle centered at the origin. The initial mesh is presented in Fig. 7(a). The refined mesh before smoothing is plotted in Fig. 7(b) and the final smoothed mesh is plotted in Fig. 7(c).

To test the effectiveness of the quality improvement strategies suggested in section 4, this mesh was regenerated twice using modified algorithms. The first modification implements a coarse refinement followed by a uniform dicing



**Figure 7: Meshes obtained for the metric given by Eqs. (6)–(8) using the original algorithm and two modified versions.**

strategy. Figure 7(d) presents the refined mesh obtained for a coarse metric where the prescribed edge sizes have been multiplied by three. Figure 7(e) presents the same mesh after all interior quadrilateral elements have been diced in nine and Fig. 7(f) plots the final result after smoothing. The second modification implements a combined refinement and smoothing strategy by the addition of interme-

diated smoothing after step 4 of the algorithm. Figure 7(g) plots the smoothed initial mesh. Figure 7(h) presents the mesh after the first combined refinement and smoothing iteration and Figure 7(i) shows the final mesh obtained after the second iteration. Note that without intermediate smoothing, four refinement iterations are needed to generate the mesh in Fig. 7(c). Figure 8 magnifies the upper right quad-

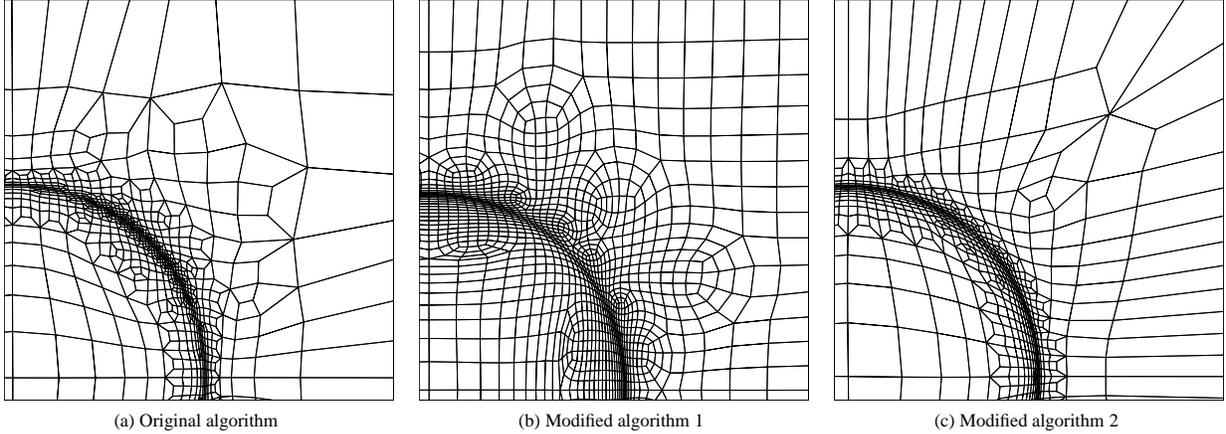


Figure 8: Detail of the meshes obtained using the metric given by Eqs. (6)–(8).

rant of the three meshes showing a region where the initial rows of quadrilateral elements were completely misaligned with the specified metric. First note that the original algorithm doesn't realign the elements when the rows of the initial mesh do not follow the features of the metric. That is why, as can be seen in Fig. 8(a), the refinement is effectively isotropic in those regions although the metric prescribes stretched elements. **OOT** anisotropic smoother can counteract this effect but the particular mesh topology resulting from the present refinement strategy is very stiff and precludes significant improvement. However, dicing a coarse mesh produces a more regular topology and the smoother can do its work (Fig. 8(b)). Combining refinement and anisotropic smoothing also improves the shape of the elements in this region (Fig. 8(c)). The smoother tends to align the element rows with the features of the metric and, indeed, changes the layers the refinement primitive will choose to contract. Table 3 compares the corresponding quality indicators. Dicing a coarse mesh improves the shape of the elements from an average of 0.645 to 0.743 but increases their overall number from 3316 to 4672. This method produces nicer transitions but inserts too many vertices. Dicing could also complicate coupling with solvers because a coarsely adapted mesh won't be available. A better way to achieve the same effect would be to shrink thicker layers. On the other hand, combining refinement with anisotropic smoothing not only improves the shape of the elements from an average of 0.645 to 0.672 but also decreases their final number elements from 3316 to 2460 and improves convergence, i.e.,

Table 3: Quality indicators evaluated in the metric given by Eqs. (6)–(8) for the meshes of Fig. 7.

Indicator	Nb	min	$\mu$	max	$\sigma$
$l_{\text{ini}}$ (Fig. 7(a))	220	0.447	2.126	10.3	1.246
$\eta_{\text{ini}}$	100	0.103	0.689	0.94	0.375
$l_{\text{fin}}$ (Fig. 7(c))	6660	0.295	0.663	1.39	0.292
$\eta_{\text{fin}}$	3316	0.215	0.645	0.97	0.269
$l_{\text{fin}}$ (Fig. 7(f))	9400	0.139	0.529	1.32	0.474
$\eta_{\text{fin}}$	4672	0.205	0.743	0.98	0.156
$l_{\text{fin}}$ (Fig. 7(i))	4968	0.199	0.775	2.05	0.307
$\eta_{\text{fin}}$	2460	0.139	0.672	0.98	0.260

two iterations instead of four. Intermediate smoothing thus improves the overall efficiency of the algorithm. However, at present, the smoothing process has to be controlled manually. Too few iterations and the effect is lost. Too many iterations and the mesh is badly distorted. A better approach would be to let the smoother fully converge to a modified metric. This metric should selectively smooth out brutal variations to preserve intermediate mesh quality.

One final observation valid for all three meshes is that the present algorithm tends to overly refine them: the final average metric edge length ranges from 0.529 to 0.775. The first explanation is that we tried to bias the algorithm towards shape quality over size conformity. The layers are augmented by their immediate neighbors to have better size transition for example. The second explanation is more fundamental: the algorithm can only refine and there is no operation to coarsen the mesh. If initial edges are already too small then the algorithm cannot do anything. All it can do is split edges that are too long and drive the maximum metric edge length under the chosen threshold value. The present method is not competing with simplicial adaptation algorithms but rather addresses the automated conformal refinement problem for quadrilateral and hexahedral meshes. To truly adapt such meshes, local coarsening and reconnection tools have to be developed to complement the present refinement and vertex relocation tools.

## 5.2. Three-Dimensional Test Cases

Some very preliminary results were also generated for three-dimensional test cases using an extension of the original two-dimensional algorithm. The first test case, from [26], is a refined mesh for a unit cube with a non-uniform target edge length of  $0.05 + 0.15z$ , and is plotted in Figs. 9(a) and 9(b). For comparison, Figure 9(c) presents a tetrahedral mesh adapted to the same metric by **OOT** [26]. The initial mesh was uniform and had  $4 \times 4 \times 4$  cubic elements. The corresponding statistical data is given in Table 4. The final metric edge length has an average of 1.072 and a maximum value just around the refinement threshold of 2.0. The standard deviation of the metric edge length was divided by two compared to the initial mesh. We have again, how-

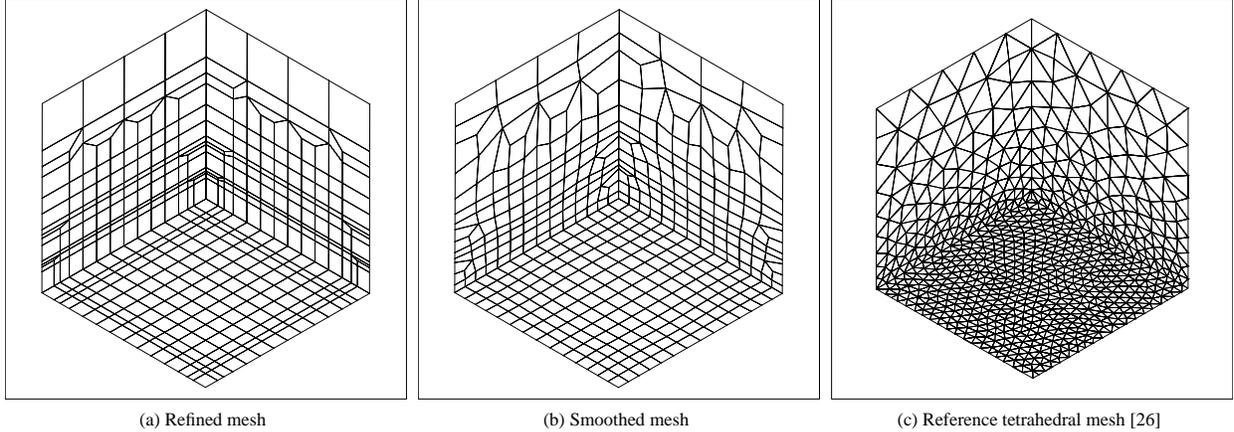


Figure 9: Mesh obtained for a unit cube with a target edge length of  $0.05 + 0.15z$ .

ever, a non-negligible reduction of the average shape indicator from 0.895 to 0.770. The present algorithm trades off shape quality for size conformity.

The second three-dimensional test case is also a unit cube but this time the metric is given by

$$\mathcal{M} = \mathcal{R}_\phi \mathcal{R}_\theta \begin{pmatrix} h_1^{-2} & 0 & 0 \\ 0 & h_2^{-2} & 0 \\ 0 & 0 & h_3^{-2} \end{pmatrix} \mathcal{R}_\theta^{-1} \mathcal{R}_\phi^{-1} \quad (9)$$

where  $\mathcal{R}_\theta$  and  $\mathcal{R}_\phi$  are rotation matrices

$$\mathcal{R}_\theta = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (10)$$

$$\mathcal{R}_\phi = \begin{pmatrix} \cos \phi & 0 & \sin \phi \\ 0 & 1 & 0 \\ -\sin \phi & 0 & \cos \phi \end{pmatrix} \quad (11)$$

$$\text{with } \begin{cases} \theta = \arctan(y/x), \\ \phi = \arccos(z/r) - \pi/2, \\ h_1 = r/40 + 3|1-r|/2, \\ h_2 = h_3 = r/10 + 3|1-r|/2. \end{cases} \quad (12)$$

with  $r = \sqrt{x^2 + y^2 + z^2}$ . This metric field prescribes stretched elements along a 0.5 radius sphere centered at the origin. The mesh plotted in Fig. 10 was obtained starting from a uniform mesh of  $5 \times 5 \times 5$  cubic elements and the corresponding statistics are presented in Table 5. Note the clustering of the mesh vertices along the sphere and the reduction in the maximum edge length. However, the average edge length was smaller than 1.0 in the initial mesh and could only be reduced in the refined mesh since the present algorithm cannot remove vertices. Furthermore, the shape of the smoothed elements is not as good as expected. Improvements using a new vertex relocation scheme are, however, possible [22].

## 6. FUTURE WORK

As noted in the previous section, refinement iterations should be combined with a vertex relocation adaptor, i.e., a smoother.

The metric for these intermediate smoothing iterations should, however, be modified to preserve mesh quality by avoiding brutal size transitions during the early stages the refinement process.

Another way to improve mesh quality is to modify the layer agglomeration method. The thickness of the layers can be increased. Path optimization can be added. Balancing inspired from quadtree and octree methods can also help.

Although the present **OOT** anisotropic smoother works well with simplices, it can be improved for quadrilateral and hexahedral elements. Modifications to the present algorithm are being explored and some results are presented in [22].

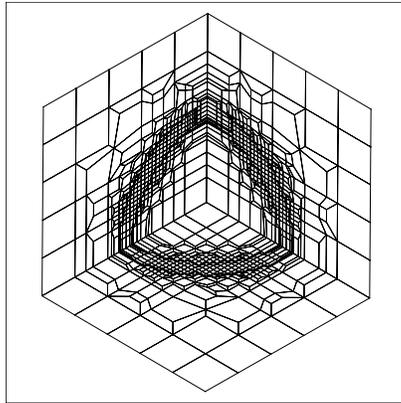
The present refinement method, even when combined with vertex relocation, is not a full adaptation method. It lacks vertex removal and reconnection capabilities. Reconnection methods to improve shape quality have already been developed for quadrilateral meshes [27, 28]. Figure 11 presents a quadrilateral mesh adapted by the combination of the present algorithm with a limited implementation of such reconnection.

Table 4: Quality indicators evaluated for a unit cube with a target edge length of  $0.05 + 0.15z$  plotted in Fig. 9.

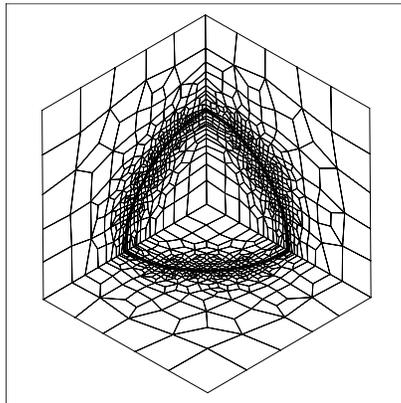
Indicator	Nb	min	$\mu$	max	$\sigma$
$l_{\text{ini}}$	300	1.250	2.456	5.00	0.497
$\eta_{\text{ini}}$	64	0.815	0.895	0.94	0.056
$l_{\text{fin}}$ (Fig. 9(b))	4168	0.272	1.072	2.12	0.251
$\eta_{\text{fin}}$	1144	0.244	0.770	0.95	0.179
$l_{\text{ref}}$ (Fig. 9(c))	14219	0.726	1.034	1.38	0.124
$\eta_{\text{ref}}$	10899	0.582	0.914	1.00	0.053

Table 5: Quality indicators evaluated in the metric given by Eqs. (9)–(12) for the mesh of Fig. 10.

Indicator	Nb	min	$\mu$	max	$\sigma$
$l_{\text{ini}}$	540	0.105	0.464	4.69	1.406
$\eta_{\text{ini}}$	125	0.099	0.784	0.95	0.260
$l_{\text{fin}}$	18385	0.020	0.339	1.48	0.459
$\eta_{\text{fin}}$	5638	0.002	0.566	0.93	0.394



(a) Refined mesh



(b) Smoothed mesh

**Figure 10: Mesh refinement using the metric given by Eqs. (9)–(12).**

tion tools. The target metric map for this mesh was extracted from a portrait of German mathematician Bernhard Riemann (1826-1866). For comparison, a triangular mesh adapted to the same metric by **OORT** is also presented in Fig. 11. Similar topologic operators have been explored in three dimensions [29, 30]. Those operators should be tested on meshes generated by the present algorithm to improve the quality of their hexahedral elements.

## 7. CONCLUSION

An all-quadrilateral and all-hexahedral refinement method has been presented. Using a shrink and connect strategy known as pillowing or buffer insertion, layers of elements are contracted and reconnected to achieve directional mesh enrichment according to a size specification map defined as a Riemannian metric. An anisotropic smoother that optimizes the location of the vertices of the refined mesh further enhances clustering. Both two and three-dimensional test cases confirm the feasibility of such an approach to refine an initial mesh to an arbitrary size distribution.

Compared to other non-simplicial conformal refinement approaches, the present method should prove more flexible. It

works both in two and three dimensions, it can start from unstructured meshes and it does not require a final uniform dicing step. Furthermore, the combination of the present refinement operator with an anisotropic smoother and a metric control map is innovative. Element quality can, nevertheless, still be improved. Future work should include implementation of reconnection primitives, stronger coupling between refinement and smoothing, thicker shrink layers and an improved vertex relocation scheme.

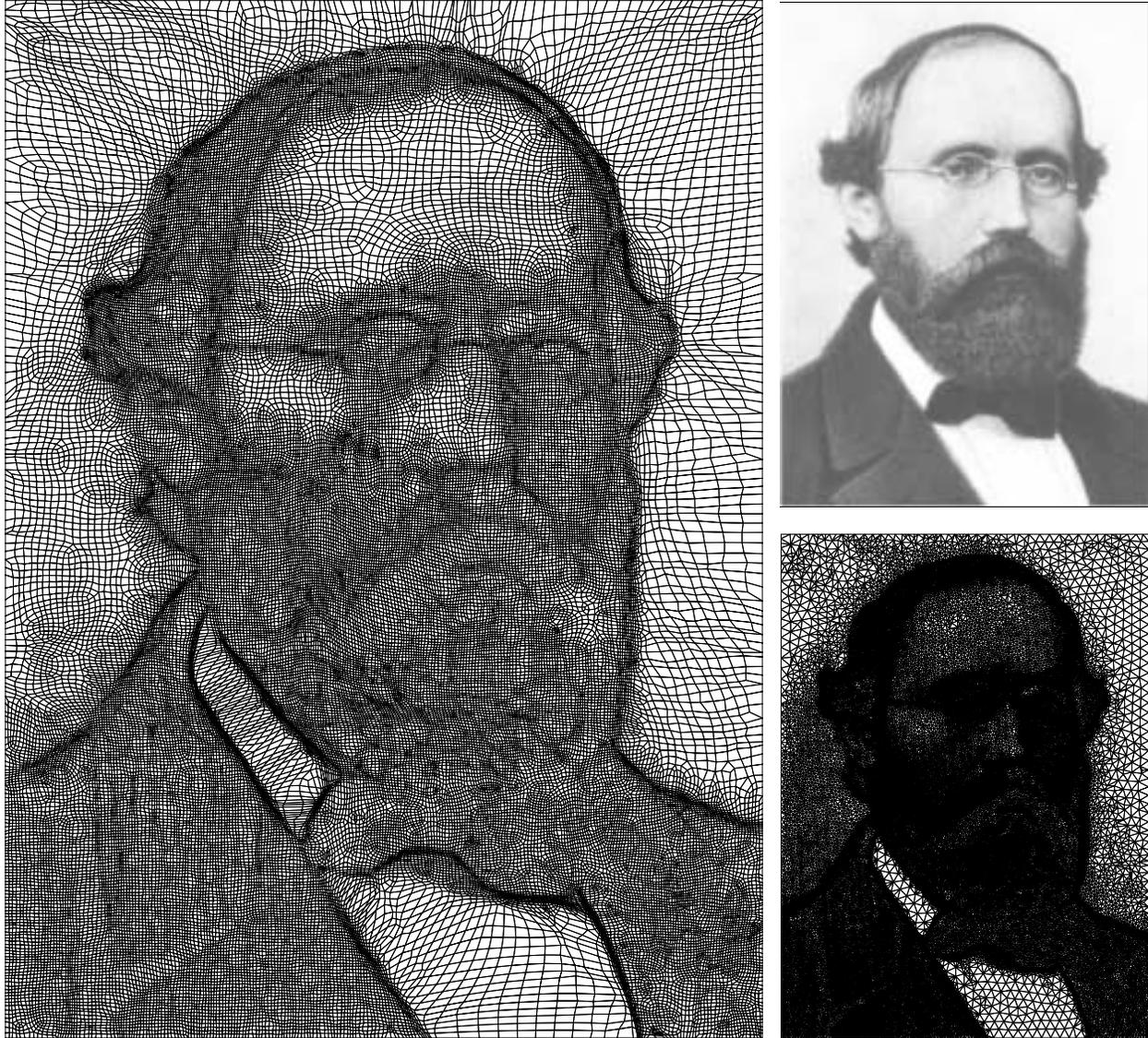
Coupling with solvers is possible but the present method is not yet a full adaptation scheme because it lacks local coarsening and topologic improvement capabilities. Applications to problems such as steady flow computations in CFD should however be possible in the next development stage of the present project. Another possible application area is initial refinement for grid-based hexahedral mesh generation methods.

## 8. ACKNOWLEDGEMENTS

The authors would like to thank NSERC for its financial support. Many thanks also to Paul Labbé and François Guibault who co-developed **OORT** with the second author and are the developers of its framework, i.e., **PIRATE** (<http://www.cerca.umontreal.ca/pirate>).

## REFERENCES

- [1] P.-L. George and H. Borouchaki, *Delaunay Triangulation and Meshing. Applications to Finite Elements*. Paris: Hermès, 1998.
- [2] P. J. Frey and P.-L. George, *Mesh Generation. Application to Finite Elements*. Paris: Hermès, 2000.
- [3] D. Eppstein, “Global optimization of mesh quality.” Tutorial presented at the Tenth International Meshing Roundtable, Oct. 2001. <http://www.ics.uci.edu/~eppstein/pubs/geom-tri.html>.
- [4] J. Dompierre, P. Labbé, and F. Guibault, “OORT (Object-Oriented Remeshing Toolkit).” <http://www.cerca.umontreal.ca/oort>.
- [5] D. Aït-Ali-Yahia, G. Baruzzi, W. G. Habashi, M. Fortin, J. Dompierre, and M.-G. Vallet, “Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD. Part II: Structured grids,” *Submitted to Int. J. Numer. Meth. Fluids*, 2001.
- [6] T. D. Blacker and M. B. Stephenson, “Paving: A new approach to automated quadrilateral mesh generation,” *Int. J. Numer. Meth. Engng*, vol. 32, pp. 811–847, 1991.
- [7] T. D. Blacker and R. J. Meyers, “Seams and wedges in plastering: A 3D hexahedral mesh generation algorithm,” *Engineering with Computers*, vol. 2, no. 9, pp. 83–93, 1993.
- [8] T. J. Tautges, T. Blacker, and S. A. Mitchell, “The whisker weaving algorithm: A connectivity-based



**Figure 11: Quadrilateral (left) and triangular (lower right) meshes adapted to a metric extracted from a portrait (upper right) of Bernhard Riemann (1826-1866).**

method for constructing all-hexahedral finite element meshes," *Int. J. Numer. Meth. Engng*, vol. 39, pp. 3327–3349, 1996.

- [9] C. Armstrong, D. Robinson, R. McKeag, T. Li, S. Bridgett, R. Donaghy, and C. McGleenan, "Medials for meshing and more," in *Fourth International Meshing Roundtable*, pp. 277–288, 1995.
- [10] A. Sheffer, M. Etzion, A. Rappoport, and M. Bercovier, "Hexahedral mesh generation using the embedded Voronoi graph," in *Seventh International Meshing Roundtable*, (Dearborn, Michigan), pp. 347–364, Sandia National Laboratories, Oct. 1998.
- [11] R. Schneiders, "A grid-based algorithm for the generation of hexahedral element meshes," *Engineering with Computers*, vol. 12, pp. 168–177, 1996.
- [12] K.-F. Tchou, C. Hirsch, and R. Schneiders, "Octree-based hexahedral mesh generator for viscous flow simulations," in *13th AIAA Computational Fluid Dynamics Conference*, no. AIAA-97-1980, (Snowmass, CO), AIAA, June 1997.
- [13] R. Schneiders, "Octree-based hexahedral mesh generation," *Int. J. of Comp. Geom. & Applications*, vol. 10, no. 4, pp. 383–398, 2000.
- [14] L. Maréchal, "A new approach to octree-based hexahedral meshing," in *Tenth International Meshing*

- Roundtable*, (Newport Beach, CA), pp. 209–221, Sandia National Laboratories, Oct. 2001.
- [15] H. Borouchaki and P. J. Frey, “Adaptive triangular-quadrilateral mesh generation,” *Int. J. Numer. Meth. Engng*, vol. 41, pp. 915–934, 1998.
- [16] S. J. Owen, M. L. Staten, S. A. Canann, and S. Saigal, “Advancing front quadrilateral meshing using triangle transformations,” in *Seventh International Meshing Roundtable*, (Dearborn, Michigan), pp. 409–428, Sandia National Laboratories, Oct. 1998.
- [17] R. Schneiders, “Refining quadrilateral and hexahedral element meshes,” in *5th International Conference on Numerical Grid Generation in Computational Field Simulations*, (Mississippi State University), pp. 679–688, Apr. 1996.
- [18] S. A. Mitchell and T. J. Tautges, “Pillowing doublets: Refining a mesh to ensure that faces share at most one edge,” in *Fourth International Meshing Roundtable*, pp. 231–240, 1995.
- [19] M. Borden, S. Benzley, S. A. Mitchell, D. R. White, and R. Meyers, “The cleave and fill tool: An all-hexahedral refinement algorithm for swept meshes,” in *Ninth International Meshing Roundtable*, (New Orleans, Louisiana), pp. 69–76, Sandia National Laboratories, Oct. 2000.
- [20] P. Murdoch and S. Benzley, “The spatial twist continuum,” in *Fourth International Meshing Roundtable*, pp. 243–252, 1995.
- [21] M.-G. Vallet, *Génération de maillages éléments finis anisotropes et adaptatifs*. PhD thesis, Université Pierre et Marie Curie, Paris VI, France, 1992.
- [22] Y. Sirois, J. Dompierre, M.-G. Vallet, P. Labbé, and F. Guibault, “Progress on vertex relocation schemes for structured grids in a metric space,” in *8th International Conference on Numerical Grid Generation*, (Honolulu, USA), pp. 389–398, June 2002.
- [23] P. Labbé, J. Dompierre, M.-G. Vallet, F. Guibault, and J.-Y. Trépanier, “A measure of the conformity of a mesh to an anisotropic metric,” in *Tenth International Meshing Roundtable*, (Newport Beach, CA), pp. 319–326, Sandia National Laboratories, Oct. 2001.
- [24] J. Dompierre, P. Labbé, M.-G. Vallet, F. Guibault, and R. Camarero, “Critères de qualité pour les maillages simpliciaux,” in *Maillage et adaptation* (P.-L. George, ed.), pp. 311–348, Paris: Hermès, Oct. 2001.
- [25] H. Borouchaki, F. Hecht, and P. J. Frey, “H-correction,” Tech. Rep. 3199, Institut National de Recherche en Informatique et en Automatique, France, June 1997.
- [26] J. Dompierre, P. Labbé, F. Guibault, and R. Camarero, “Proposal of benchmarks for 3D unstructured tetrahedral mesh optimization,” in *Seventh International Meshing Roundtable*, (Dearborn, MI), pp. 459–478, Sandia National Laboratories, Oct. 1998.
- [27] M. L. Staten and S. A. Canann, “Post refinement element shape improvement for quadrilateral meshes,” in *AMD-Vol. 220 Trends in Unstructured Mesh Generation*, pp. 9–16, ASME, July 1997.
- [28] P. Kinney, “Cleanup: Improving quadrilateral finite element meshes,” in *Sixth International Meshing Roundtable*, (Park City, Utah), pp. 437–447, Sandia National Laboratories, Oct. 1997.
- [29] P. Knupp and S. A. Mitchell, “Integration of mesh optimization with all-hex mesh generation, LDRD subcase 3504340000, final report,” Sandia Report SAND99-2852, Sandia National Laboratories, 1999.
- [30] M. Bern and D. Eppstein, “Flipping cubical meshes,” in *Tenth International Meshing Roundtable*, (Newport Beach, CA), pp. 19–29, Sandia National Laboratories, Oct. 2001.