ISSN 0965-5425, Computational Mathematics and Mathematical Physics, 2019, Vol. 59, No. 12, pp. 1945–1964. © Pleiades Publishing, Ltd., 2019.
Published in Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki, 2019, Vol. 59, No. 12, pp. 2024–2044.

Hybrid Voronoi Mesh Generation: Algorithms and Unsolved Problems

V. A. Garanzha[a,b,*], L. N. Kudryavtseva[a,b,c], and V. O. Tsvetkova[c]

[a] *Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control," Russian Academy of Sciences, Moscow, 119333 Russia*

[b] *Moscow Institute of Physics and Technology (State University), Dolgoprudnyi, Moscow oblast, 141701 Russia*

[c] *Federal Research Center Keldysh Institute of Applied Mathematics, Russian Academy of Sciences, Moscow, 125047 Russia*

**e-mail: garan@ccas.ru*

Received June 26, 2019; revised June 26, 2019; accepted August 5, 2019

Abstract—We consider problem of constructing Voronoi mesh where the union of Voronoi cells approximates the computational domain with a piecewise smooth boundary. In the 2d case the smooth boundary fragments are approximated by the Voronoi edges and Voronoi vertices are placed near summits of sharp boundary corners. We suggest self-organization meshing algorithm which covers the boundary of domain by an almost-structured band of non-simplicial Delaunay cells. This band consists of quadrangles on the smooth boundary segment and convex polygons around sharp corners. Dual Voronoi mesh is double layered orthogonal structure where central line of the layer approximates the boundary. Overall Voronoi mesh has a hybrid structure and consists of high quality convex polygons in the core of the domain and orthogonal layered structure near boundaries. We introduce refinement schemes for the Voronoi boundary layers, in particular near sharp corners. In the case when the boundary of domain is defined explicitly we suggest Voronoi meshing algorithm based on circle placement on the boundary. We discuss problems related to 3d case generalization of suggested algorithm and illustrate ideas and difficulties on relatively simple 3d test cases.

Keywords: Delaunay–Voronoi meshes, orthogonal Voronoi meshes, polygonal meshes, polyhedral meshes

DOI: 10.1134/S0965542519120078

1. INTRODUCTION

Construction of hybrid polyhedral meshes in complicated 3d domains is interesting and actively developing field of mesh generation. A well-established approach to polyhedral meshing is based on construction of tetrahedral mesh and its approximate dualization [1, 2]. In most cases this technique produces high quality polyhedra. Unfortunately, near boundary it creates a number of cut cells which should be optimized to get acceptable mesh. Optimality criteria in most cases are contradictory hence costly multicriterial optimization is needed with uncertain outcome. One can speculate that good solution is construction of Voronoi polyhedral mesh with full uncut Voronoi cells near boundary. We are not aware about such an algorithms. Hence, the goal of the paper is to try to construct algorithm which solves above problem in 2d, at least in the practical sense, and consider problems related to more complicated 3d case. Presented paper is an extended version of the conference paper [3].

Note that approximation of domains by Voronoi tilings and their generalization has rich history, especially in surface reconstruction problems [4]. Many algorithms for construction and optimizations of Voronoi meshes were suggested, see [5, 7–9]. Unfortunately, these algorithms are not suitable to build Voronoi meshes with regular Voronoi layers near boundaries, which is the topic of the present research.

1.1. Definition of Implicit Multimaterial Domain

Consider bounded domain Ω which is partitioned into N subdomains Ω_i, $i = 0,\ldots,N-1$. Intuitively one can just imagine a body glued from different materials. We assume that boundary of each subdomain is piecewise smooth and Lipschitz continuous. The simplest case of multimaterial domain is based on two

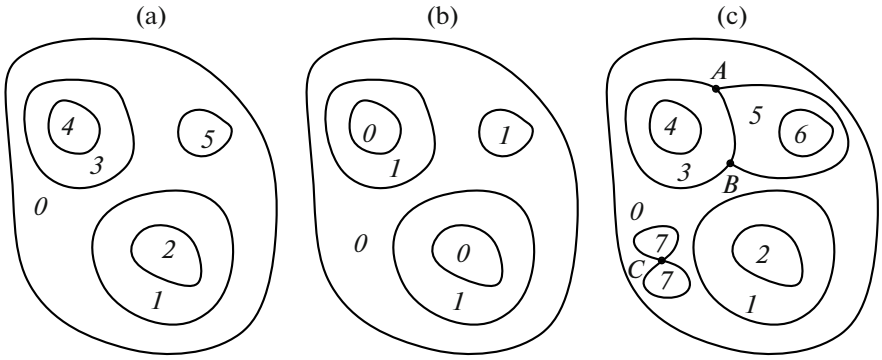

Fig. 1. Examples of multimaterial domains.

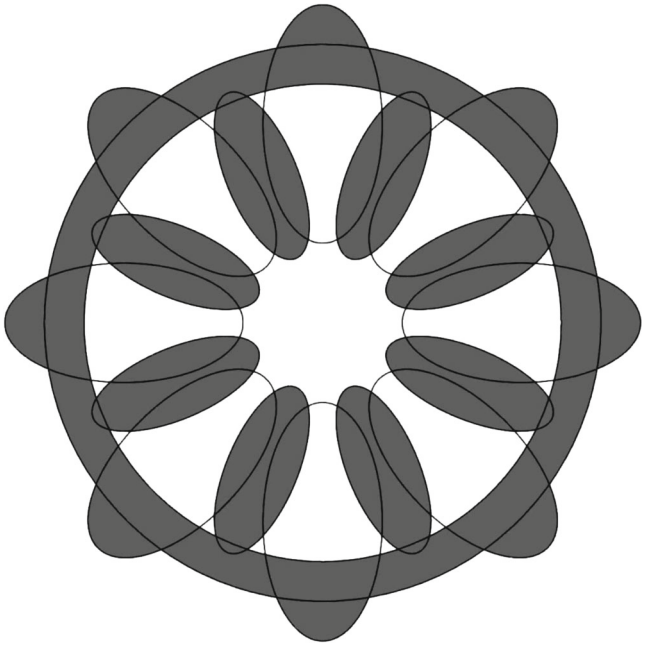

Fig. 2. Model "wheel": construction of implicit domain using Boolean operations.

assumptions: (a) the boundary of each subdomain is a manifold and (b) multimaterial vertices with neighborhoods containing more than two materials are absent. An example of such a domain is shown in Fig. 1a.

Mesh generation problem in this multimaterial domain is equivalent to mesh generation problem in bimaterial domain shown in Fig. 1b. One can model such a domain by a single scalar function $u(x) : \mathbb{R}^d \to \mathbb{R}$, which is negative inside Ω_1, positive inside Ω_0, and zero isosurface of this function is the boundary. Moreover, it is convenient to immerse the computational domain in a domain of simple geometry whose boundary does not need to be approximated. A more complicated case is presented in Fig. 1c. Here, two multimaterial vertices A, B and non-Lipschitz vertex C are present. Meshing algorithm described below potentially can be applied in this case as well, but we did not tested such a configurations yet.

One can use Boolean operations and build quite complicated domains from primitives. Figure 2 shows (in gray) the planar domain used as a test case for the meshing algorithm.

It is assumed that the function $u(x)$ is piecewise smooth, Lipschitz continuous and its derivatives along a certain vector field transversal to the internal boundary Γ are not equal to zero in a finite layer around the boundary. In fact, it is assumed that the behavior of the implicit function resembles that of the signed distance function. In particular, we always assume that the norm of $\nabla u(x)$, when defined, is bounded from below and above in a certain layer around Γ.

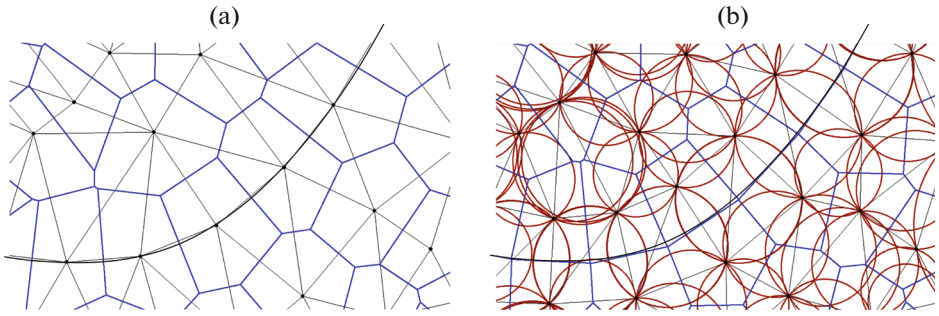

Fig. 3. (a) Boundary of domain is approximated by Delaunay edges, (b) boundary of domain is approximated by Voronoi edges.

1.2. Voronoi Mesh in Implicit Domain

Consider a planar mesh $\mathcal{D}$ consisting of convex polygons D_i inscribed into circles B_i as shown in Fig. 3b. D_i is the convex envelope of all mesh vertices lying on ∂B_i. Each circle is empty in a sense that it does not contain any mesh vertices inside. Such a mesh is called Delaunay mesh (Delaunay partitioning). Considering the convex envelope of all centers c_i of circles B_i passing through Delaunay vertex p_k we get Voronoi cell V_k. The set of Voronoi cells constitutes a partition which is generally called Voronoi diagram. Since in our setting outer boundary is not approximated, we are not interested in infinite Voronoi cells so we just call resulting object Voronoi mesh. The internal boundaries can be approximated using a Delaunay mesh as shown in Fig. 3a or using a Voronoi mesh, see Fig. 3b.

Let us briefly explain the difference. Piecewise smooth boundary Γ is approximated by a system of polylines. It is assumed that with mesh refinement polylines converge to Γ in the following sense: (a) distance from each straight edge of polyline to certain distinct simple arc of Γ should be small; (b) deviation of normal to straight edge from exact normals on the arc should be small; (c) sharp vertices on Γ are approximated by sharp vertices on polyline. For Delaunay mesh this polyline is built from Delaunay edges, while for Voronoi mesh polyline is constructed from Voronoi edges. Delaunay edges, dual to the boundary Voronoi edges, are orthogonal to the boundary. For smooth fragment of boundary Delaunay cells should be quadrilaterals which make up a band covering the boundary. The midline of this band consisting of Voronoi edges, approximates the boundary as shown in Fig. 3b. It is well known that state-of-the-art algorithms generate Delaunay triangulations and not general Delaunay partitions. However, the edges splitting boundary Delaunay cells into triangles have zero dual Voronoi edges and do not influence the Voronoi mesh.

Typical behavior of Delaunay–Voronoi mesh around sharp boundary vertex is shown in Fig. 4. Regular Delaunay bands consisting of quads are glued together through convex polygonal Delaunay cell. The number of sides in this polygon depends on the sharp vertex angle.

2. VORONOI MESHING ALGORITHM BASED ON SELF-ORGANIZATION OF ELASTIC NETWORK

In order to build Voronoi meshes in domains with a nonsmooth boundary, we adapt the algorithm of [11], which was originally developed for Delaunay meshing in 2d and 3d implicit domains with piecewise smooth boundaries. The unknowns in the presented algorithm are Delaunay mesh vertices, which are considered as material points repulsing each other, thus modelling elastic medium. Repulsive forces are applied to each pair of vertices belonging to Delaunay edges, i.e., edges with circumferential open balls not containing any other vertices. Each Delaunay edge is treated as a compressible strut that tries to expand until the prescribed length is reached. At each step a dual Voronoi mesh is constructed and partitioned into two subdomains according to the value of the implicit function at the Delaunay vertices. The Delaunay mesh is split into three subdomains: subdomain 0, subdomain 1, and a set of bands covering the boundary. All Delaunay triangles with circumcenters close to Γ are added to the bands. An approximate Voronoi boundary polyline is constructed. At this moment mesh refinement is applied provided that a local minimum of energy is attained. The idea of mesh refinement is to try to eliminate long Voronoi edges that are not orthogonal to the boundary. It is explained in Fig. 5.

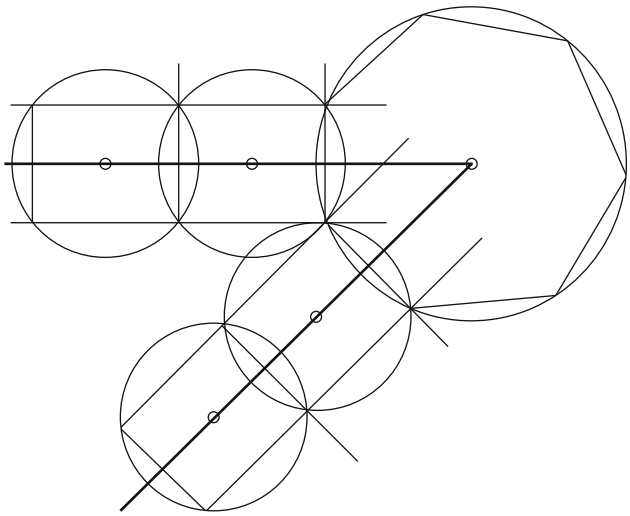

Fig. 4. Band of polygonal Delaunay cells and dual Voronoi edges on the boundary of domain.

With each Voronoi edge we associate "sharpening energy" and "boundary attraction potential." Boundary attraction potential is used as a penalty term for obvious condition that each boundary Voronoi edge is tangential to Γ and touches it in certain "touching point". Sharpening energy is minimized when the Voronoi edge e is orthogonal to the vector ∇u at the "touching point" for e. We use special variant of preconditioned gradient search method to make one step of minimization. It is convenient to call directions vectors in the minimization technique "elastic forces." When due to point displacement under elastic forces edge loses Delaunay property it should be excluded from the list of struts and new Delaunay edges should be created. Hence, Voronoi mesh should be rebuilt as well. These steps are repeated until boundary is approximated with reasonable accuracy and correct topology of the near-boundary layers is recovered.

The outcome of the algorithm is certain "equilibrium" mesh where elastic forces acting on each point sum to zero. As was suggested in [10], we build an equilibrium mesh in the slightly compressed state.

2.1. Elastic Potential

Suppose that a system of points $\mathscr{E} = \{p_1, p_2, \ldots, p_n\}$ in $\mathbb{R}^2$ is prescribed. Let us denote by $\mathscr{T}(\mathscr{E})$ its Delaunay triangulation. We denote by $\mathscr{T}_e$ the set of the edges of triangulation and by $\tilde{\mathscr{E}}_b$ the set of the Delaunay edges crossing Γ. All vertices constitute $2 \times n$ matrix P with ith column equal to p_i. We denote the set of the near-boundary Delaunay vertices by P_Γ. Voronoi mesh dual to $\mathscr{T}$ is denoted by $\mathscr{V}$, and the set of the Voronoi edges detected as a current guess to polyline approximating Γ is denoted by $\mathscr{E}_v$.

With each mesh T we associate the following elastic potential

$$W(P) = \theta_r W_r(P) + \theta_s W_s(P) + \theta_a W_a(P), \tag{1}$$

where $W_r(P)$ is the repulsion potential, $W_s(P)$ is the sharpening potential which serves to align Voronoi boundary edges along isolines of function u, $W_a(P)$ is the sharp edge attraction potential.

The repulsion potential is written as follows

$$W_r(P) = \sum_{e \in \mathscr{T}_e} w_r(e),$$

$$w_r(e) = \begin{cases} L_0^2\left(\dfrac{L}{L_0} - 1 - \log\left(\dfrac{L}{L_0}\right)\right), & L < L_0, \\ 0 \quad L \geq L_0, \end{cases}$$

where

$$L = |p_i - p_j|,$$

is the length of the edge e and $L_0(e)$ is the target length of this edge defined by

$$L_0(e) = Mh\left(\frac{1}{2}(p_i + p_j)\right).$$

Here, $h(\cdot)$ is relative sizing function which is dimensionless, while scalar multiplier M defines the actual length. As suggested in [10], parameter M has the meaning of the average mesh edge length and may slightly change in the process of mesh self-organization.

In practice we use $L_0(e) = M\frac{1}{2}(h(p_i) + h(p_j))$ in order to diminish number of sizing function calls.

The sharpening functional is written as

$$W_s(P) = \sum_{e_v \in \mathcal{E}_v} w_s(e_v),$$

where the contribution from the boundary Voronoi edge e_v with vertices c_1, c_2 looks like

$$w_s(e_v) = \frac{1}{2}|c_1 - c_2|(n^{\mathrm{T}}(c_2 - c_1))^2,$$

where

$$n = \frac{1}{|\nabla u(v^*)|}\nabla u(v^*), \tag{2}$$

and v^* is the current approximation of the touching boundary point for the Voronoi edge e_v. The simplest choice of v^* is projection of the middle point

$$c = \frac{1}{2}(c_1 + c_2)$$

of e_v onto Γ.

The Voronoi edge boundary attraction term is written as

$$W_a(P) = \sum_{e_v \in \mathcal{E}_v} w_a(e_v),$$

where

$$w_a(e_v) = \frac{1}{2}\left(\frac{L_0}{L}\right)^2 u^2(c).$$

Here, L is the length of the Delaunay edge dual to e_v. Hence, the energy assigned to shorter Delaunay edges is higher. Since the unstable Delaunay edges used to triangulate near-boundary approximate Delaunay polygons are, in general, longer than stable edges, they make a smaller contribution to total energy and have a small influence on the positions of the vertices.

2.2. "Elastic Forces" and Practical Iterative Algorithm

It is convenient to introduce the notions of "repulsive forces", "sharpening forces" and "boundary attraction forces" which denote the contribution to the direction vector from the repulsion, sharpening and boundary attractions terms, respectively.

Roughly speaking, these "forces" are introduced as follows

$$\begin{aligned}\delta p_i^k &= -\frac{\theta_r}{d_{ri}^k}\frac{\partial W_r}{\partial p_i}(P^k) - \frac{\theta_s}{d_{si}^k}\frac{\partial W_s}{\partial p_i}(P^k) - \frac{\theta_a}{d_{ai}^k}\frac{\partial W_a}{\partial p_i}(P^k)\\ &= F_r(p_i^k) + F_s(p_i^k) + F_a(p_i^k).\end{aligned} \tag{3}$$

Here, k is the iteration number; p_i is the ith vertex in the Delaunay mesh P^k; and d_{ri}^k, d_{si}^k, d_{ai}^k are the scaling factors.

Since the Newton law is not used to describe the motion of mesh vertices, these "forces" are speculative and are just used to facilitate intuitive understanding of the algorithm.

In order to present precise formulae for computation of forces it is convenient to introduce the following notations. Let $\mathrm{star}_e(p_i)$ denote the set of the mesh edges originating from the vertex p_i, while $\mathrm{star}(p_i)$ will denote the set of vertices of these edges excluding p_i. In all cases we assume that every boundary star is ordered, i.e., its entities are numbered counterclockwise around p_i looking from outside the domain. Below, we omit the upper index k.

For the internal vertex p_i, the repulsive "force" looks like

$$F_r(p_i) = -\frac{\theta_r}{d_i}\sum_{p_j\in \mathrm{star}\ p_i}\phi_r(p_i,p_j)(p_i-p_j),\quad d_{ri} = \sum_{p_j\in \mathrm{star}\ p_i}\phi_r(p_i,p_j),$$

where

$$\varphi_r(p_i,p_j) = \left(\frac{L_0}{L}-1\right)\frac{L_0}{L},\quad L = |p_i-p_j|,\quad L_0 = Mh\left(\frac{1}{2}(p_i+p_j)\right)$$

Sharpening force can be written as follows

$$F_s(p_i) = -\frac{\sum_{e_v:p_i\in \mathrm{dual}e_v}\Pi_r(q|c_1-c_2|n^{\mathrm{T}}(c_1-c_2))}{\sum_{e_v:p_i\in \mathrm{dual}e_v}|c_1-c_2||q|^2},$$

here, c_1 and c_2 are vertices of the edge e_v, $c = \frac{1}{2}(c_1+c_2)$, the vector n is defined in (2), and

$$q = (C_2-C_1)^{\mathrm{T}}n,\quad C_1 = \frac{\partial c_1}{\partial p_i},\quad C_2 = \frac{\partial c_2}{\partial p_i}.$$

In order to write down an expression for the matrix C_1, consider Delaunay triangle T_1 with counterclockwise ordered vertices p_i, p_j, p_k whose circumcenter is c_1. Then

$$C_1^{\mathrm{T}} = (c_1-p_ic_1-p_i)(p_j-p_ip_k-p_i)^{-1}.$$

The formula for C_2 is similar.

Nonlinear operator Π_r is responsible for the interaction between the repulsive force and the sharpening force.

Consider the contribution to $F_s(p_i)$ made by the Voronoi edge e_v. Let $e = (p_j-p_i)/|p_j-p_i|$, where p_i, p_j are vertices of the Delaunay edge dual to e_v. If

$$e^{\mathrm{T}}F_r(p_i)e^{\mathrm{T}}qn^{\mathrm{T}}(c_1-c_2) < 0,$$

then

$$\Pi_r(qn^{\mathrm{T}}(c_1-c_2)) = qn^{\mathrm{T}}(c_1-c_2) - ee^{\mathrm{T}}qn^{\mathrm{T}}(c_1-c_2);$$

otherwise

$$\Pi_r(qn^{\mathrm{T}}(c_1-c_2)) = qn^{\mathrm{T}}(c_1-c_2).$$

After local corrections for the sharpening terms, the assembled sharpening force at the vertex p_i is used to correct the repulsive force F_r:

$$F_r \leftarrow F_r - \frac{1}{2|F_s|^2}F_s\left(F_s^{\mathrm{T}}F_r - \left|F_s^{\mathrm{T}}F_r\right|\right).$$

The attraction force looks like

$$F_a(p_i) = -\sum_{e_v:p_i\in \mathrm{dual}e_v}\frac{1}{2}\left(\frac{L_0}{L}\right)^2 u(c)(C_1+C_2)^{\mathrm{T}}\frac{\nabla u(c)}{|\nabla u(c)|}.$$

The Delaunay vertices are displaced in two steps. The first step is written as

$$\tilde{p}_i^0 = p_i^k + w_r\tau_rF_r + w_s\tau_sF_s,\quad w_r = \frac{1}{20},\quad w_s = \frac{1}{2},$$

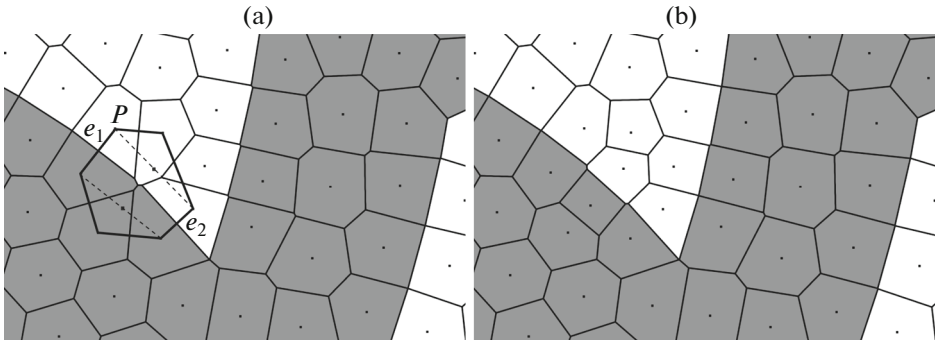

Fig. 5. (a) Fragment of Voronoi mesh with nonorthogonal edges, (b) correct connectivity is attained by adding new Delaunay vertices.

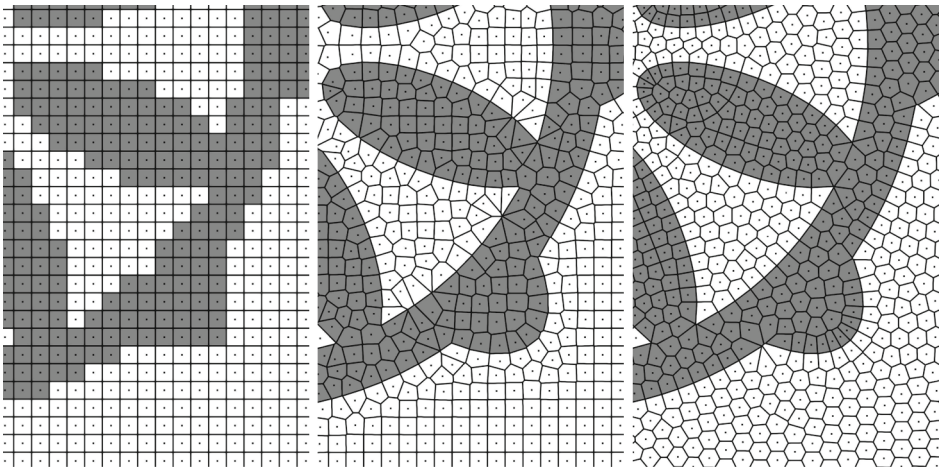

Fig. 6. Fragment of initial Voronoi mesh, result after a few iterations and stabilized Voronoi mesh.

$$\tau_r = \min\left(1, \frac{L_0}{5w_rF_r}\right), \quad \tau_s = \min\left(1, \frac{L_0}{5w_sF_s}\right).$$

After this displacement, we use M iterations with attraction force to project the Voronoi edges onto the boundary

$$\tilde{p}_i^{m+1} = \tilde{p}_i^m + \tau_a F_a(\tilde{p}_m^l), \quad \tau_a = \frac{1}{10}.$$

Finally,

$$p_i^{k+1} = \tilde{p}_i^M.$$

2.3. Numerical Experiments

We ran series of numerical experiments with artificially constructed domains. The complexity of the tests is well represented by the model "wheel" shown in Fig. 2. In this model multiple sharp vertices are present on the boundary.

Numerical evidence suggests that algorithm recovers internal boundaries quite fast. However, this guess contains approximation defects and topological defects when near-boundary Voronoi edges are not orthogonal to boundary. The origin of these errors is simple: Delaunay vertex does not have good mirror across the boundary. Hence, most of the topological errors can be eliminated by reasonable Delaunay vertex insertion, as shown in Fig. 5. We consider the polygon P that is the closest guess to the Delaunay poly-

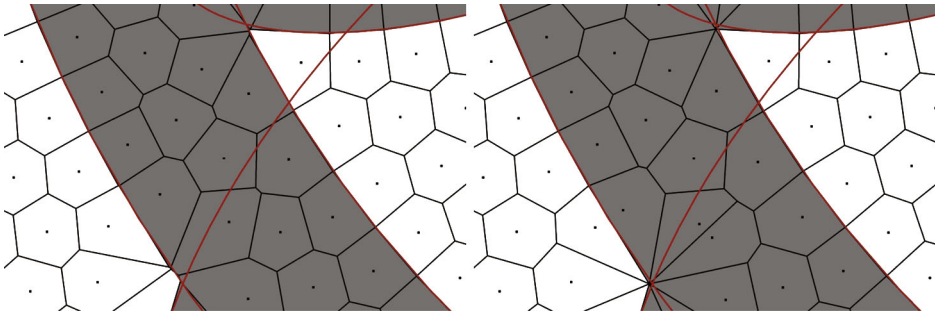

Fig. 7. Elimination of Voronoi faults: enlarged view.

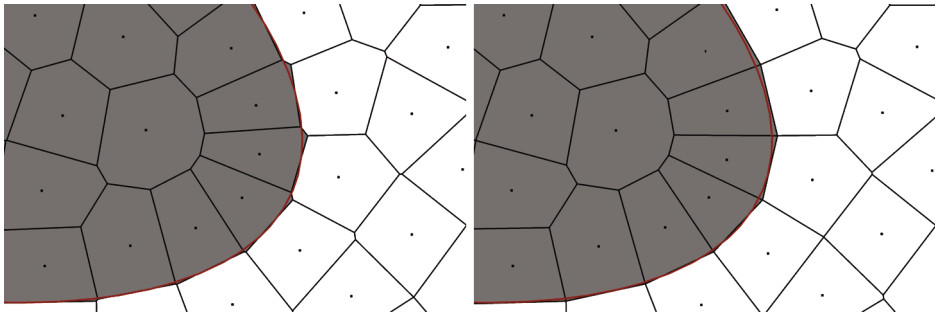

Fig. 8. Elimination of Voronoi faults: enlarged view.

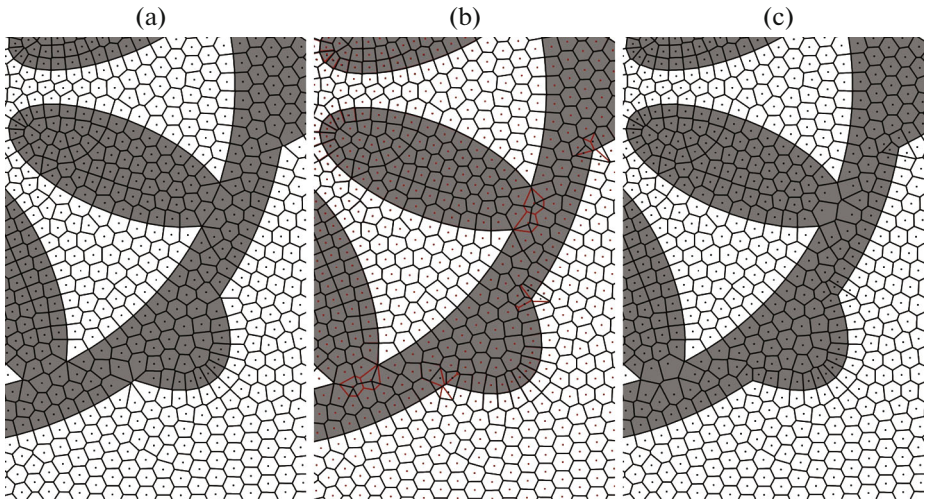

Fig. 9. (a) Voronoi mesh before correction of corner Delaunay polygons, (c) Voronoi mesh after correction of corner Delaunay polygons, (b) overlapped meshes.

gon built upon two stable Delaunay edges e_1 and e_2 crossing the boundary. We build a quadrilateral cell upon these two edges and add new vertices at the middle of virtual opposite edges.

Approximate Delaunay hexagon is resolved by inserting two vertices, while approximate Delaunay pentagon is resolved by adding single vertex. In our test cases there was no need to consider more complex polygons.

Figure 6 illustrates Voronoi mesh evolution for an enlarged fragment of the “wheel” model.

Figures 7 and 8 demonstrate that elimination of short Voronoi edges does not lead to deterioration of boundary approximation quality. To this end we glue together nearby Delaunay circles and find new

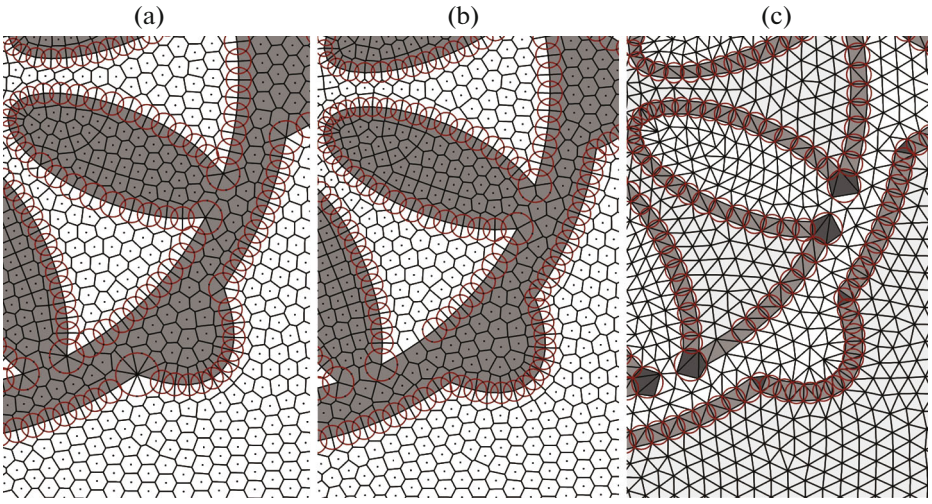

Fig. 10. (a) Mesh after elimination of short Voronoi edges, (b) Voronoi mesh after correction of Delaunay polygons, and (c) Delaunay layer covering boundary.

Delaunay vertices as intersections of corrected circles. We call the short boundary edges "Voronoi faults" by analogy with geology. Elimination of faults creates final mesh where internal boundaries are approximated by Voronoi edges and normals to the boundary are approximated by discrete normals.

Figure 9 illustrates importance of corrections of corner Delaunay polygons. As one can see, after reduction of Delaunay polygons to triangles, quadrilaterals, and pentagons, large Delaunay circles disappear and the deviation from orthogonality for near-boundary Voronoi cells is reduced.

Figure 10b shows a fragment of the final Voronoi mesh with boundary Delaunay circles, while the set of bands of Delaunay cells is shown on the right.

As soon as the basic layered structure of the Delaunay–Voronoi mesh is constructed, one can try to build anisotropic orthogonal Voronoi mesh layers using the following anisotropic refinement algorithm:

- new couple of vertices is added symmetrically on each Delaunay edge crossing the boundary;
- special refinement schemes are applied to the corner Delaunay polygons;
- new Voronoi cells are computed, boundary Voronoi edges are projected onto the boundary;
- this inward-directed splitting procedure is repeated until required mesh compression rate is attained;
- Voronoi fault filtering procedure based on close Delaunay circles gluing is applied starting from the boundary outward to the core of the domain;
- near corners constrained Delaunay circle gluing procedure is applied.

Refinement of regular Delaunay band creates "faults", namely, short Voronoi edges which are misaligned with the direction of the approximated curves. In order to eliminate faults we compute all Delaunay circles for a quasi-regular band of Delaunay cells. Short Voronoi edges correspond to circles which almost coincide. We simply interpolate circle centers and radii and get a regular band of circles. New potential Delaunay vertices are computed as intersections points of two adjacent circles. The circle interpolation procedure should take into account constrained vertices near sharp corners. We repeat this procedure for all layers until outer boundary is reached. This procedure proved to be quite stable, provided that the initial quasi-regular Delaunay mesh is built by structured refinement of coarse regular Delaunay band.

A detailed view of the refined hybrid meshes is shown in Figs. 11–13.

As one can see, algorithm can produce Voronoi layers consisting of almost orthogonal highly anisotropic quadrilaterals.

Figure 14 shows the large fragment of hybrid Voronoi mesh before and after fault elimination.

It should be noted that the refinement schemes for corner Delaunay pentagons shown in Fig. 12, and those for corner Delaunay triangles, see Fig. 13, produce Voronoi bumps around corners. We were not

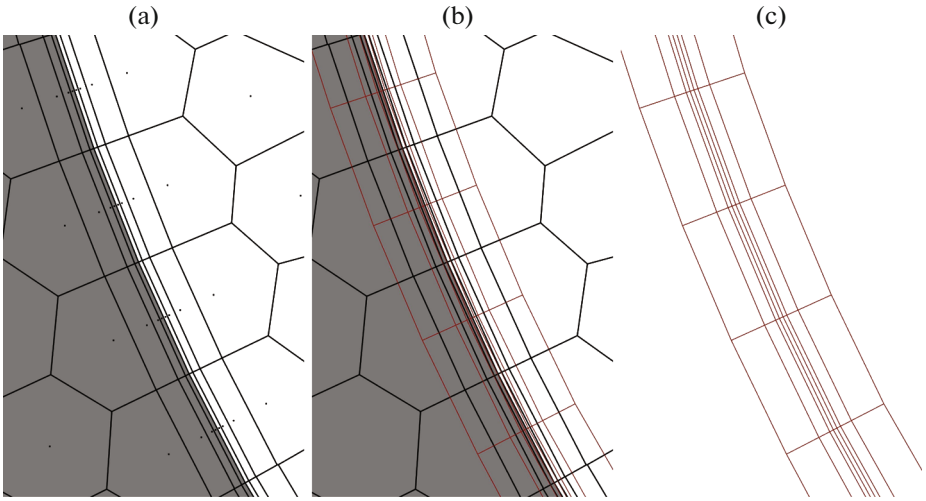

Fig. 11. (a) Anisotropic Voronoi mesh layer, (c) dual anisotropic Delaunay layer, (b) overlapped meshes.

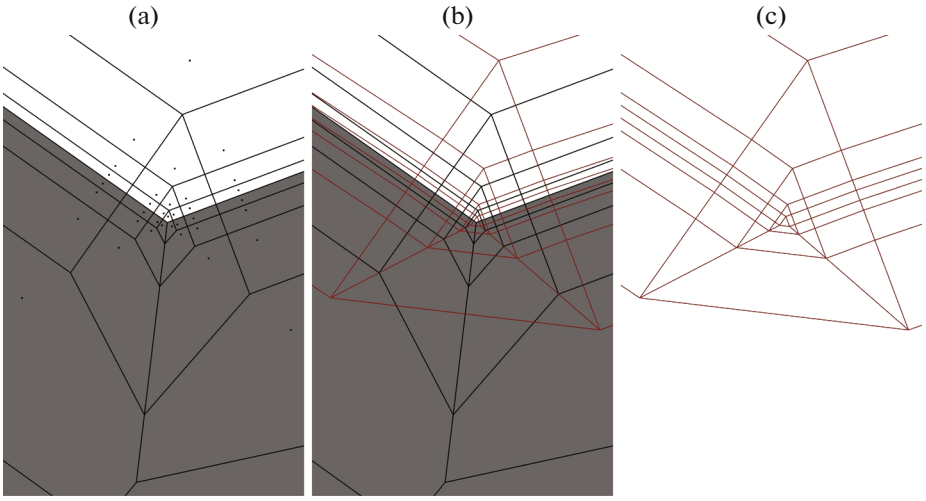

Fig. 12. Refinement scheme for corner Delaunay pentagon: (a) anisotropic Voronoi mesh layer, (c) dual anisotropic Delaunay layer, (b) overlapped meshes.

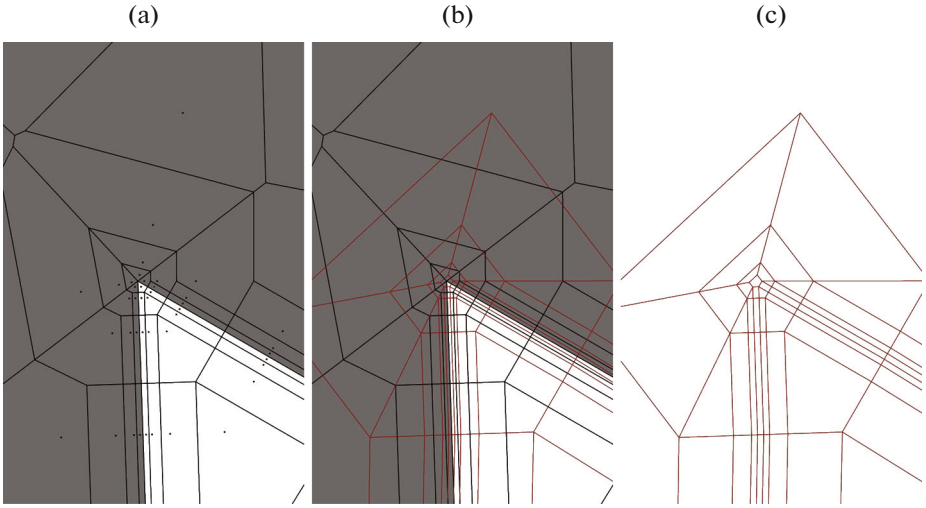

Fig. 13. Refinement scheme for corner Delaunay triangle: (a) anisotropic Voronoi mesh layer, (c) dual anisotropic Delaunay layer, (b) overlapped meshes.

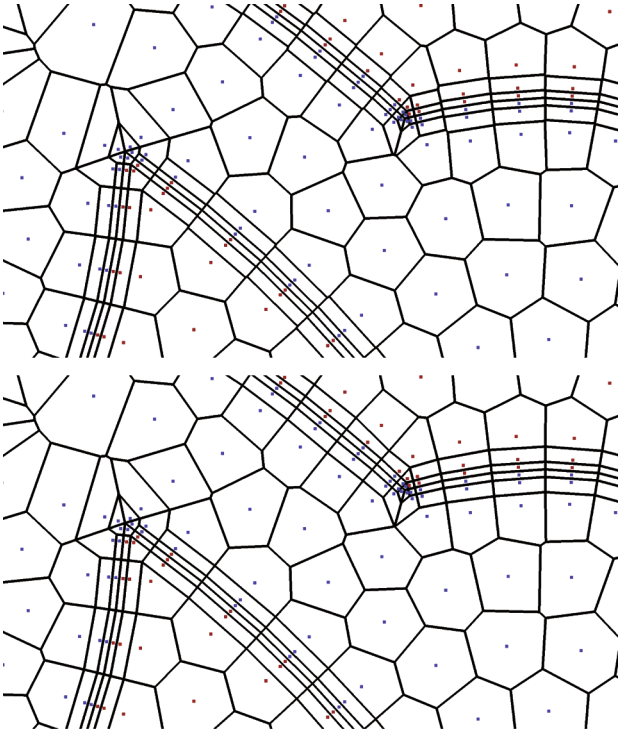

Fig. 14. Voronoi mesh after refinement and after cleaning.

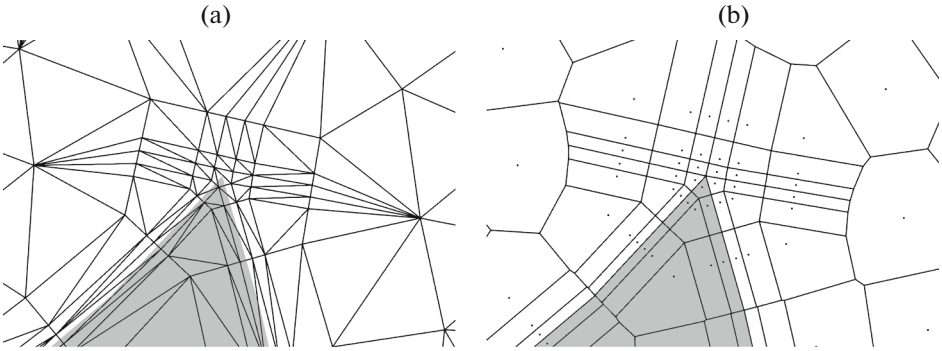

Fig. 15. (a) Delaunay mesh after refinement, Voronoi fault cleaning is not applied, (b) Voronoi mesh.

able to find anisotropic Delaunay/Voronoi refinement schemes which do not create such a bumps. Note that the problem appears only for Voronoi cells, while Delaunay edges round off sharp corner in a nice and smooth manner. Figs. 15, 16 illustrate the behavior of the refinement schemes which are isotropic near corners.

Such a refinement schemes for Delaunay triangle, quadrilateral and pentagon are very close in spirit to the refinement of the block-structured meshes when the blocks are glued together using well-known C-type and H-type meshes. Here, Voronoi faults are not eliminated in order to clarify the amount of deviation from structured mesh introduced by refinement schemes. Note that in order to get final mesh one should add smooth transition from each corner to the core Voronoi mesh.

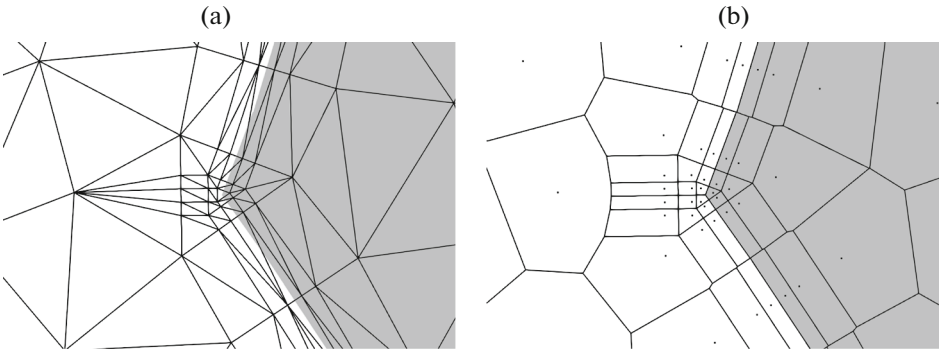

Fig. 16. (a) Delaunay mesh after refinement, Voronoi fault cleaning not applied, (b) Voronoi mesh.

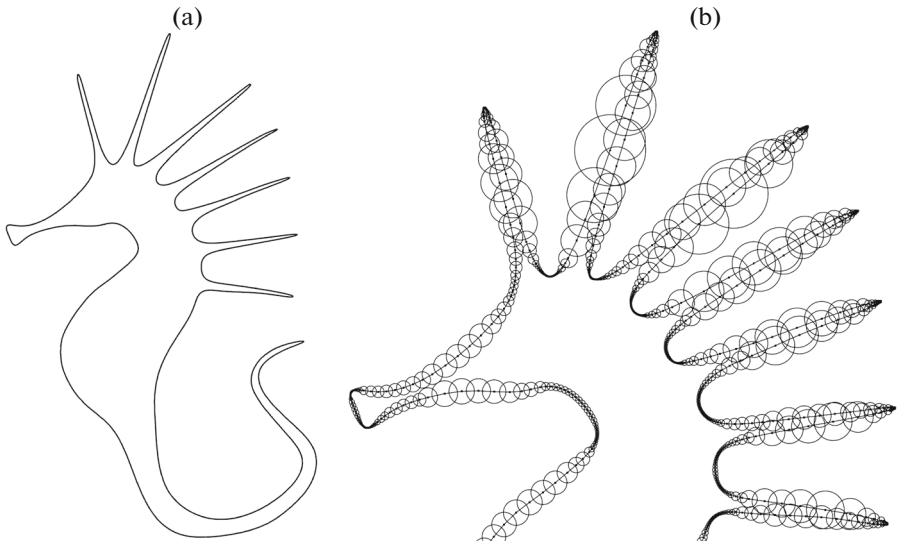

Fig. 17. (a) Seahorse contour, (b) initial set of circles.

3. VORONOI MESH GENERATION VIA CIRCLE PLACEMENT

When boundary of the planar domain is defined explicitly, simple and efficient Voronoi meshing algorithm can be used. Suppose that internal boundary is a closed contour specified by the set of the consecutive vertices p_i. Let us assign to the ith vertex the circle B_i with radius r_i defined by the formula

$$r_i = \frac{1}{2\sqrt{2}}\left(|p_i - p_{i-1}| + |p_i - p_{i+1}|\right). \tag{4}$$

For smooth contour the system of balls is correct when circle B_i intersects only the circles B_{i-1} and B_{i+1}. This set of circles defines the band of quadrilateral Delaunay cells covering the contour. For straight fragment and uniform center distribution, the convex envelope of the set of intersection points on the boundary of each circle is a square.

Figure 17a shows the seahorse contour used in [6] as a test case for Voronoi meshing algorithm while Fig. 17b shows the initial set of circles. As one can see, numerous nonadjacent circle intersections are present.

In order to resolve this problem, the following optimization algorithm for circle placement is applied.

All pairs of intersecting nonadjacent circles are identified. Note that if one circle is contained inside another one then it is guaranteed that this circle has nonadjacent intersection. For each pair of bad circles

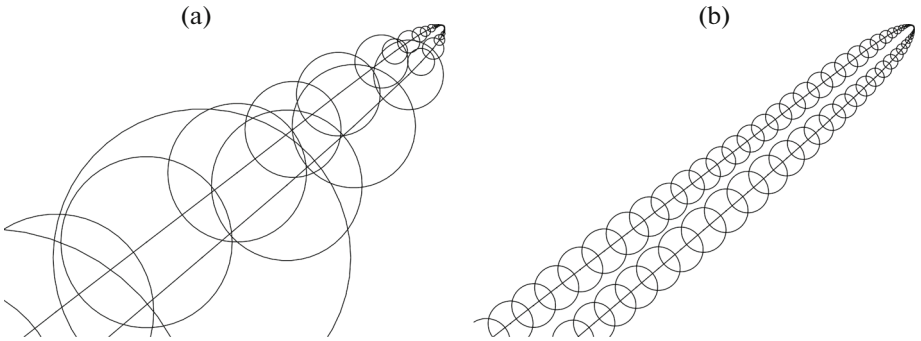

Fig. 18. (a) Initial covering circles, (b) corrected circles.

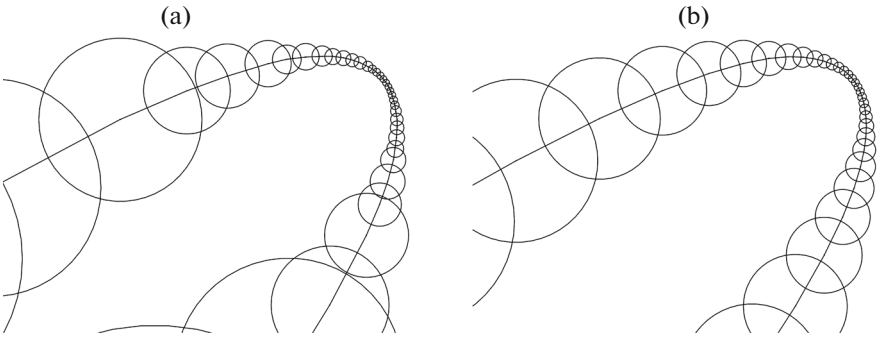

Fig. 19. (a) Fragment of initial set of covering circles, (b) improved covering circles suitable for initial Voronoi meshing.

the largest one is split, meaning that it is replaced by a pair of smaller circles. Denote the circle to be split by B_i. The vertex p_i is eliminated and replaced by two new vertices a and b with coordinates

$$a = \frac{1}{3}p_{i-1} + \frac{2}{3}p_i, \quad a = \frac{1}{3}p_{i+1} + \frac{2}{3}p_i.$$

The radii of new circles B_a and B_b are defined by formula (4). The positions of vertices on the contour are then slightly smoothed by a few iterations of discrete Laplace smoothing along the contour. This refinement/smoothing procedure is stopped when all incorrect circle intersections are eliminated.

The results for this algorithm are illustrated in Fig. 18, and enlarged fragment is shown in Fig. 19. Note that the byproduct of this algorithm is circle adaptation to the curvature of the contour.

In order to build a 2d Voronoi mesh, it is enough to distribute mesh points over the computational domain and eliminate all the points covered by the system of circles B_i. Adding circle intersection points, we get an initial set of vertices. Building a Delaunay mesh upon these vertices, we guarantee that boundary covering Delaunay layer is present and Voronoi edges connect the centers of the prescribed circles. The resulting mesh is shown in Fig. 20a. In this case the background Voronoi mesh is just the Cartesian mesh.

Initial mesh is optimized using the following criteria: (a) the shape of the Delaunay triangles should be good enough, and (b) the Voronoi cells should be close to the centroidal ones. Namely, we require that for each Delaunay triangle the ratio of circumradius to its smallest edge is smaller than $\sqrt{2}$, and the center of mass for each Voronoi cell is close to its generator (the Delaunay vertex). In order to improve the shape of Delaunay triangles, we combine the Delaunay refinement scheme closely resembling the Ruppert algorithm [19] with few intermediate Lloyd iterations [18], which put each Delaunay vertex into the center of mass of its Voronoi cell and serve to improve the shape of Voronoi cells.

According to the logic of the Ruppert algorithm, at each step one creates a set of candidate vertices for insertion. This list contains all circumcenters of poor-shaped triangles, excluding those getting inside the

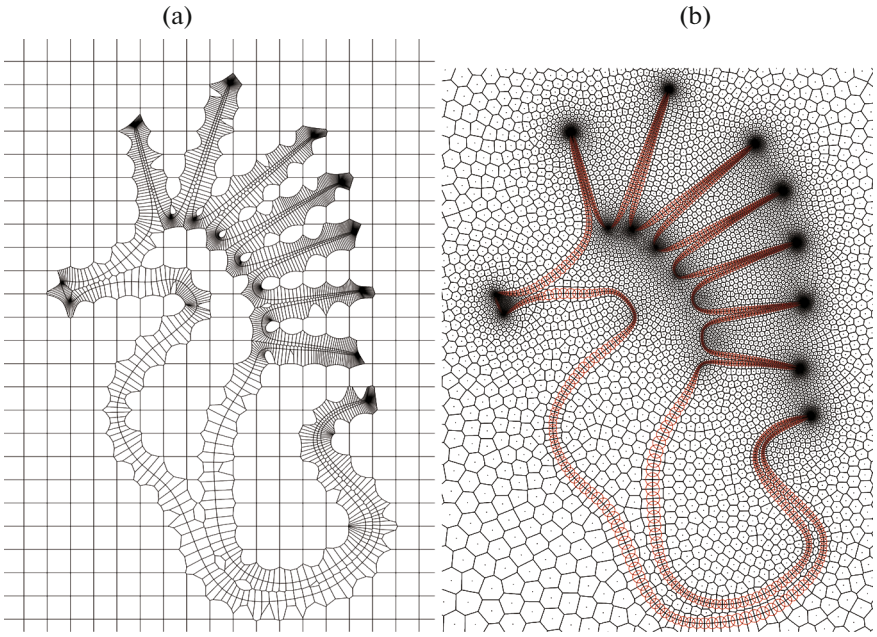

Fig. 20. (a) Initial Voronoi mesh, (b) improved Voronoi mesh.

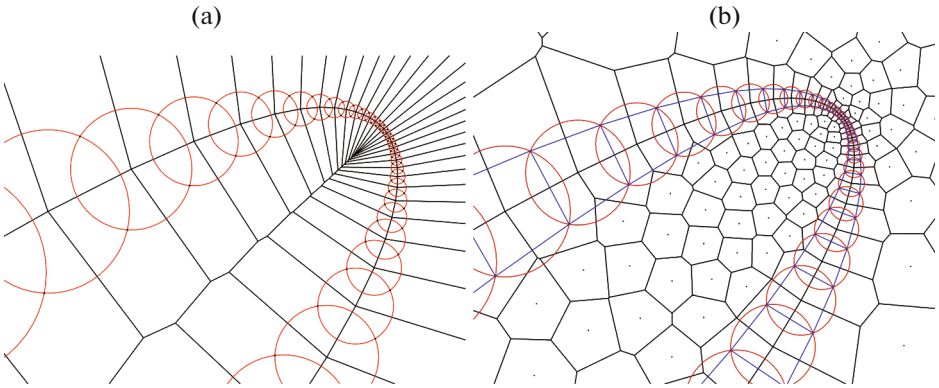

Fig. 21. (a) Fragment of initial Voronoi mesh, (b) fragment of improved Voronoi mesh.

set of boundary protecting circles. If triangle circumcenter appears inside protecting circle B_i, this circle is marked for splitting, and circumcenter is eliminated from the list.

All new vertices are added inside the domain and on the boundary contour and Laplacian smoothing is applied to the system of boundary circles. Voronoi mesh is constructed and few constrained Lloyd iterations are applied to it. Boundary circles serve as constraints. Any Delaunay vertex that gets inside the protected circle is eliminated; hence, the boundary approximation does not deteriorate.

Figure 21 shows that structured Delaunay layer is correctly adapted to the boundary curvature while Fig. 22 shows how the Voronoi meshing algorithm allows one to obtain high quality meshes.

Note that interior Voronoi cells in the resulting mesh are almost centroidal ones, while the near boundary Voronoi cells are not. Anyway, the Voronoi layers near the boundaries can be split into orthogonal sublayers using layer refinement algorithm described above thus creating hybrid curvature-sensitive Voronoi mesh. Note that in the presented numerical example we put Voronoi vertices on the boundary. Another option is to create dual contour approximation when Voronoi edge touches the contour in the intersection point with dual Delaunay edge.

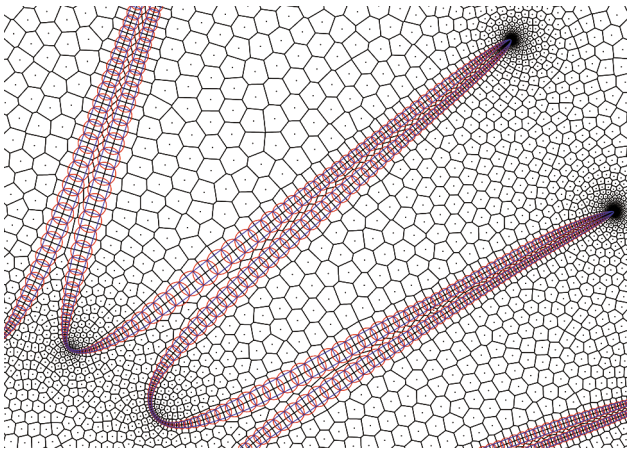

Fig. 22. Fragment of final Voronoi mesh with structured Delaunay layer covering boundary.

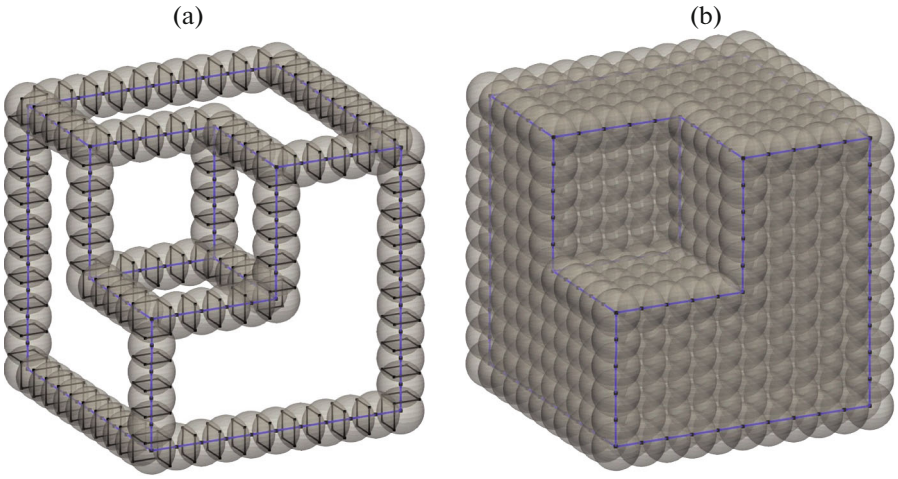

Fig. 23. (a) Delaunay balls and Delaunay faces dual to sharp boundary Voronoi edges, (b) Delaunay balls covering the boundary of domain.

4. DISCUSSION OF 3D VORONOI MESHING ALGORITHM

To our knowledge, we present the first algorithm for construction of hybrid planar Voronoi meshes which demonstrates the ability to build orthogonal layers of Voronoi cells near internal boundaries with correct resolution of sharp vertices. In order to make the suggested algorithm more universal, the following problems should be addressed: nonuniform and curvature-sensitive meshing test cases should be considered - which was taken into account in the previous section, the case of multimaterial vertices and thin material layers should be addressed, and most important, generalization to 3d case has to be investigated.

At first glance, generalization of the Voronoi meshing algorithm described above to 3d is straightforward. Consider bimateral domain with piecewise smooth internal boundary. The treatment of sharp edges in 3d is fairly straightforward: for each edge one can create the set of balls with centers on the edge in such a way that each ball intersects only two adjacent balls and intersection is shallow in a sense that intersection angle is below $\pi/2$. One can create the set of Delaunay prisms around this edge by putting at least 3 vertices on each circle being intersection of two adjacent spheres.

One should apply special treatment and create balls around conical vertices and around transitions points on the sharp edges where the switch from one type of prisms to another one should be applied.

Figure 23a shows the set of protecting balls around sharp edges and conical vertices for simple domain with piecewise planar boundaries.

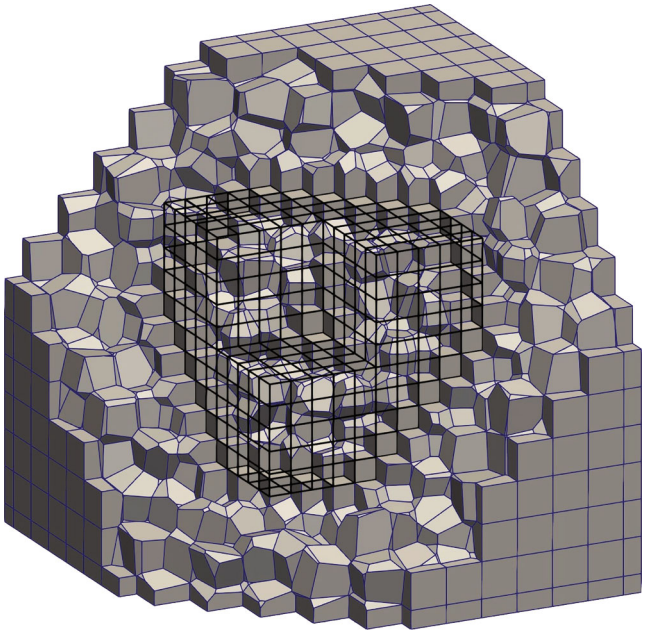

Fig. 24. Voronoi boundary and cut view of Voronoi mesh.

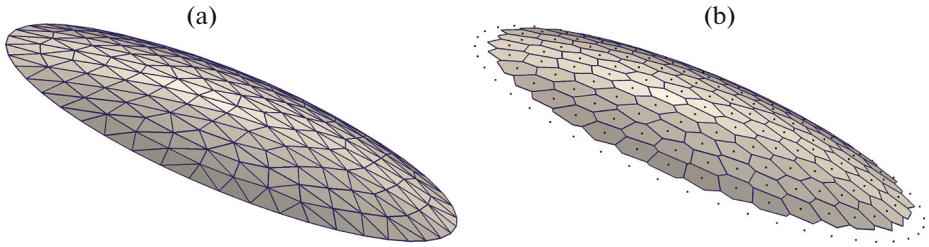

Fig. 25. (a) Primal and (b) dual polyhedral approximations for convex surface.

Figure 23b shows the set of protecting balls for the entire surface. When the dihedral angles between flat faces are not too small, one can generate a protecting ball using a quasi-2d algorithm, when each polygonal face is covered by a set of Delaunay circles that serve as equatorial circles for the spheres. Creating from each vertex on the plane a pair of mirror seeds by shifting a copy of each vertex along the normal to the surface by the same distance in the opposite directions, we get locally the set of vertices lying on the expanded sphere. Intersections of these spheres create transverse Delaunay edges. Any other vertices can be placed in the computational domain outside the set of protecting balls.

The resulting Voronoi boundary and 3d Voronoi mesh are shown in Fig. 24.

Unfortunately, in general case 3d Voronoi mesh generation from the theoretical and practical point of view is an unsolved problem. It is tightly related to the problem of polyhedral approximation of piecewise smooth surfaces. A number of approximation algorithms was suggested which can be referred to as "dual" methods, since they attempt to approximate surface by faces which are dual either to certain vertices or to edges. Dual methods are well suited for approximation of piecewise smooth surfaces. Among those methods one can refer to the class of "dual contouring" algorithms [16] and to the primal-dual surface mesh optimization with sharpening [17]. The dual faces in these algorithms can be nonplanar. The algorithm suggested in [12] can approximate quite general surfaces by polyhedral surfaces with flat faces. These faces are in general nonconvex. In [13] primal and dual polyhedral approximations were combined to obtain discrete counterpart of spherical mapping and its gradient. It was shown that behavior of dual cells for the surface of positive Gaussian curvature ($K > 0$) and negative Gaussian curvature ($K < 0$) is intrinsically different. Consider surface triangulation with the vertices lying on the regular surface with strictly positive

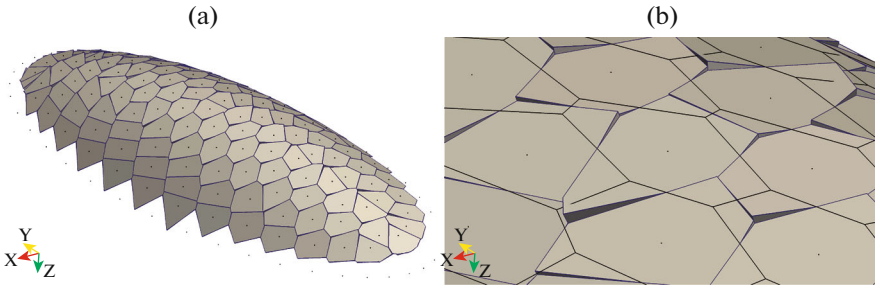

Fig. 26. Voronoi surface approximation and fragment of overlapped dual and Voronoi polyhedral surfaces.

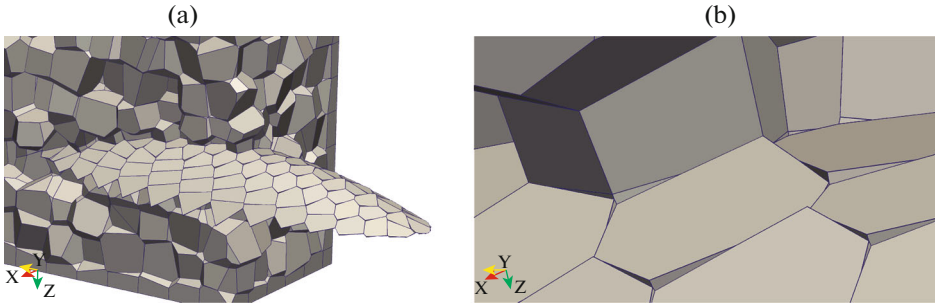

Fig. 27. Voronoi surface approximation coupled with 3d Voronoi mesh and its enlarged fragment.

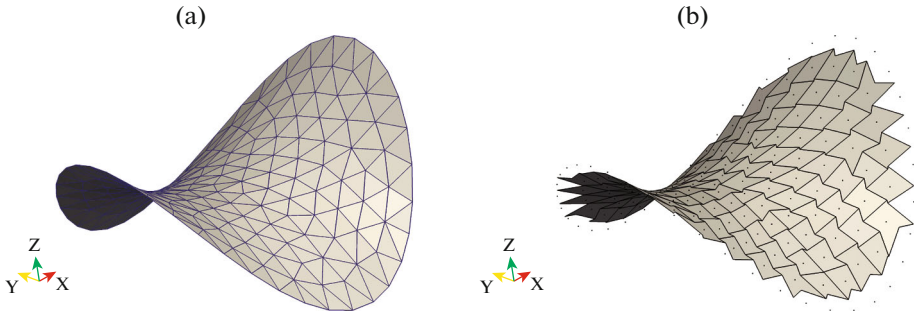

Fig. 28. Primal and dual polyhedral approximations for saddle surface.

curvature. It is assumed that triangulation is regular in a sense that it defines strictly convex polyhedral surface, as shown in Fig. 25a.

Dual surface is defined by tangent planes to the surface at the vertices of triangulation. Each dual face approximates affine image of 2d Voronoi cell for projections of a local subset of triangulations vertices onto the tangent plane [13]. One can try to model 3d Voronoi mesh by replacing each vertex of the surface triangulation by a close pair of mirror vertices (positive and negative seeds), such that middle plane, orthogonal to the Delaunay edge connecting the vertices is exactly the tangent plane. Constructing Voronoi faces dual to the pairs of seeds from different families, one obtains polyhedral Voronoi approximation of the surface. Voronoi surface tends to contain faults, which are more pronounced for surfaces with anisotropic curvatures, as shown in Fig. 26.

Note that dual faces can be quite anisotropic, while Voronoi faces tend to be isotropic and need faults to fit together.

Figure 27 shows how the Voronoi surface is integrated into a 3d Voronoi mesh.

An even more complicated case is related to saddle surfaces ($K < 0$). In [13] it was shown that when saddle surface is approximated by the correct saddle triangulation, any regular dual face in this case is quadrilateral domain with polygonal concave edges as shown in Fig. 28.

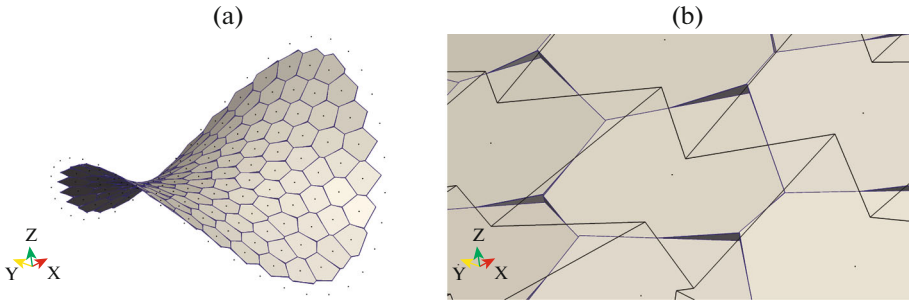

Fig. 29. Voronoi approximation of saddle surface and fragment of overlapped dual and Voronoi polyhedral surfaces.

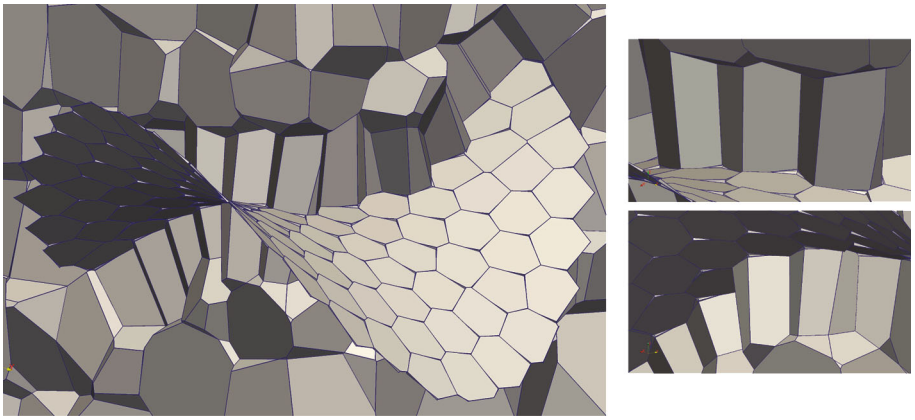

Fig. 30. Voronoi approximation of saddle surface and 3d Voronoi mesh.

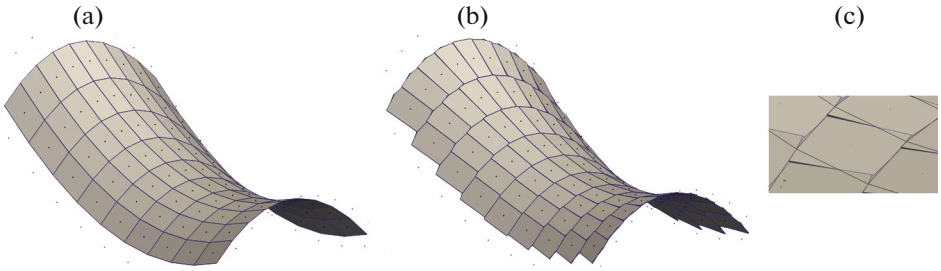

Fig. 31. (a) Planar quadrilateral dual mesh, (b) Voronoi surface, and (c) superimposed view.

The Voronoi faults in this test case are smaller since anisotropy is reduced to make the illustration more clear. Still these faults cannot be eliminated by simply gluing together close Delaunay balls, since the connectivities of Voronoi and dual polyhedral surfaces are inherently different as shown in Fig. 29b.

Figure 30 shows how Voronoi surface is integrated into 3d mesh along with its enlarged fragments.

This test case clearly shows the prismatic-like layer of Voronoi cells near boundaries. Unfortunately, unlike the 2d case, the boundary of this layer has complex facetization due to the presence of Voronoi faults on the surface.

Note that for saddle surfaces approximated by saddle triangulations, the only possibility for the dual faces to become convex is to become quadrilaterals with straight edges, which requires very special vertex arrangements on the surfaces. Figure 31a shows dual mesh which is planar quadrilateral one (pq-mesh, see [15]).

The primal mesh is also pq-mesh, but its quadrilaterals are quite skewed hence Voronoi surface (Fig. 31b) essentially differs from the dual surface and faults cannot be eliminated but by the Delaunay ball

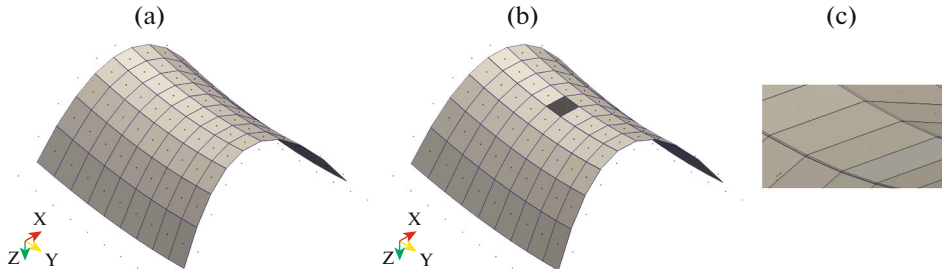

Fig. 32. Curvature line-based dual mesh, Voronoi mesh and its superposition with Voronoi surface.

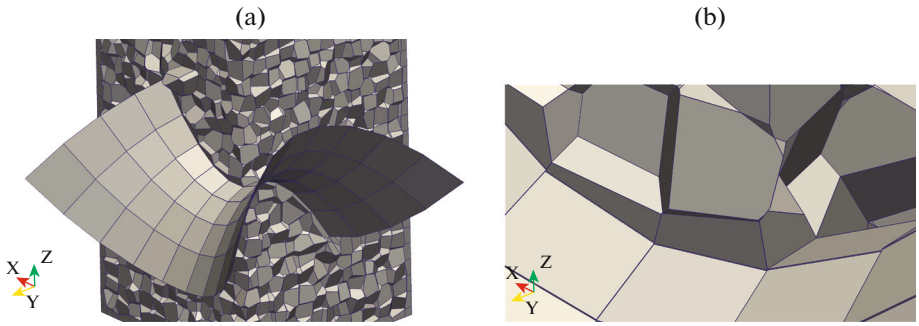

Fig. 33. Voronoi boundary surface integrated into 3d Voronoi mesh and its enlarged fragment.

gluing. Creation of new seed arrangement near boundary is necessary. It seems that one has a chance to build fault-free Voronoi surface when primal approximation is based on the circular pq-cells [15], when each flat primary cell lies on the equatorial section of certain empty sphere. Approximations to circular pq-meshes can be constructed using network of lines of curvature of the surfaces for primal vertices. Figure 32 shows dual surface, which consists of convex quadrilaterals.

When seed vertices follow the curvature lines, the Voronoi faces are almost optimal in a sense that faults are absent or quite small as shown in Fig. 32 and hopefully can be suppressed via gluing close Delaunay balls.

Figure 33 shows the 3d Voronoi cells around almost structured boundary. Small Voronoi faults create complicated facetized boundaries for near-boundary Voronoi cells which is not nice feature for numerical simulation.

It may be assumed that Voronoi surfaces without faults can be constructed if one is able to build a connectivity of dual surface mesh consisting of triangles and quadrilaterals.

CONCLUSIONS

An algorithm for constructing hybrid Voronoi meshes in planar multimaterial domains was suggested. It allows one to build Voronoi meshes with structured orthogonal Voronoi layers near boundaries. Both implicit and explicit definitions of the boundaries are supported. The generalization to the 3d case is discussed and illustrated on simple 3d test cases. We show that the key ingredient of the Voronoi meshing algorithm is the capability of controlling the positions and radii of Delaunay balls.

FUNDING

This work was supported by the Russian Foundation for Basic Research, grant 18-01-00726 A.

REFERENCES

1. R. Garimella, J. Kim, and M. Berndt, "Polyhedral mesh generation and optimization for nonmanifold domains," *Proceedings of the 22nd International Meshing Roundtable* (Springer, Cham, 2014), pp. 313–330.

2. S. Y. Lee, "Polyhedral mesh generation and a treatise on concave geometrical edges," Procedia Eng. **124**, 174–186 (2015).
3. V. A. Garanzha, L. N. Kudryavtseva, and V. O. Tsvetkova, "Structured orthogonal near-boundary Voronoi mesh layers for planar domains," in *Lecture Notes in Computational Science and Engineering*, Vol. 131: *Numerical Geometry, Grid Generation, and Scientific Computing*, Ed. by V. A. Garanzha, L. Kamenski, and H. Si (Springer International, Berlin, 2019).
4. N. Amenta and M. Bern, "Surface reconstruction by Voronoi filtering," Discrete Comput. Geom. **22** (4), 481–504 (1999).
5. Y. Liu, W. Wang, B. Levy, F. Sun, D. M. Yan, L. Lu, and C. Yang, "On centroidal Voronoi tessellation: Energy smoothness and fast computation," ACM Trans. Graphics **28** (4), 101 (2009).
6. D. M. Yan, B. Levy, Y. Liu, F. Sun, and W. Wang, "Isotropic remeshing with fast and exact computation of restricted Voronoi diagram," Comput. Graphics Forum **28** (5), 1445–1454 (2009).
7. M. Budninskiy, B. Liu, F. de Goes, Y. Tong, P. Alliez, and M. Desbrun, "Optimal Voronoi tessellations with Hessian-based anisotropy," ACM Trans. Graphics **35** (6), Article 242 (2016).
8. J. Tournois, P. Alliez, and O. Devillers, "2D centroidal Voronoi tessellations with constraints," Numer. Math. Theory Methods Appl. **3** (2), 212–222 (2010).
9. B. Lévy and Y. Liu, "L_p centroidal Voronoi tessellation and its applications," ACM Trans. Graph. **29** (4), Article 119 (2010).
10. P.-O. Persson and G. Strang, "A simple mesh generator in MATLAB," SIAM Rev. **46** (2), 329–345 (2004).
11. V. A. Garanzha and L. N. Kudryavtseva, "Generation of three-dimensional Delaunay meshes from weakly structured and inconsistent data," Comput. Math. Math. Phys. **52** (3), 427–447 (2012).
12. D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," ACM Trans. Graphics **23** (3), 905–914 (2004).
13. V. A. Garanzha, "Discrete extrinsic curvatures and approximation of surfaces by polar polyhedra," Comput. Math. Math. Phys. **50** (1), 65–92 (2010).
14. F. Gunther, C. Jiang, and H. Pottmann, "Smooth polyhedral surfaces" (2017) arXiv:1703.05318.
15. Y. Liu, H. Pottmann, J. Wallner, Y. L. Yang, and W. Wang, "Geometric modeling with conical meshes and developable surfaces," ACM Trans. Graphics **25** (3), 681–689 (2006).
16. T. Ju, F. Losasso, S. Schaefer, and J. Warren, "Dual contouring of Hermite data," ACM Trans. Graphics **21** (3), 339–346 (2002).
17. Y. Ohtake and A. Belyaev, "Dual/primal mesh optimization for polygonized implicit surfaces," *Proceedings of the 7th ACM Symposium on Solid Modeling and Applications* (*SMA'02*) (ACM, 2002), pp. 171–178.
18. S. P. Lloyd, "Least squares quantization in PCM," IEEE Trans. Inf. Theory **28** (2), 129–137 (1982).
19. J. Ruppert, "A Delaunay refinement algorithm for quality 2-dimensional mesh generation," J. Algorithms **18** (3), 548–585 (1995).