Available online at www.sciencedirect.com

ScienceDirect

Mathematics and Computers in Simulation 147 (2018) 2–26

MATHEMATICS AND COMPUTERS IN SIMULATION

www.elsevier.com/locate/matcom

Original Articles

Delaunay meshing of implicit domains with boundary edge sharpening and sliver elimination

A.I. Belokrys-Fedotov, V.A. Garanzha*, L.N. Kudryavtseva

Dorodnicyn Computing Center RAS, Federal Research Center of Informatics and Control RAS, Moscow, Vavilova 40, Russia
Moscow Institute of Physics and Technology, Doldorpudny, 141700, Russia

Received 2 March 2016; received in revised form 5 July 2017; accepted 20 July 2017
Available online 29 July 2017

Abstract

In 2004 Strang and Persson suggested the Delaunay mesh generation algorithm in implicit domains with smooth boundary based on the self-organization of elastic network, where each Delaunay edge is treated as a strut. Implicit domains in the algorithm are defined by the signed distance functions. Repulsive forces were suggested which allow to distribute mesh vertices according to prescribed size function. The Delaunay property is restored after each elastic relaxation step. This algorithm was generalized in Garanzha, Kudryavtseva, 2012 for the case of domains defined as zero levels of piecewise-smooth functions using edge sharpener suggested by Belyaev, Ohtake, 2002. In this paper we present variational version of self-organization algorithm. Suggested elastic potential is combination of repulsion potential and sharpening potential which acts on boundary edges and serves to minimize deviation of cell boundary normals from direction of gradient of implicit function which allows to approximate sharp edges on the boundary as polylines made from mesh edges without their predefinition. Numerical experiments demonstrated that Belyaev–Ohtake edge sharpener based on alignment of boundary cell normals becomes unstable when implicit function strongly deviates from the signed distance function. Stable version of sharpener is suggested and verified by numerical tests. When surface near sharp edge is highly curved sharpener based on alignment of normals fails to place vertices precisely on the sharp edge. To this end we apply special sharp edge attraction procedure. While theoretical foundations are not available numerical evidence shows that Delaunay meshes can be constructed for domains with very acute and curved sharp boundary edges. As soon as domain boundary is recovered cell shape deformation terms are added to the elastic potential and subsequent iterations allow to eliminate slivers without deterioration of approximation of sharp edges.

Keywords: Delaunay meshing; Elastic network; Edge sharpening; Variational method; Surface reconstruction

* Corresponding author at: Dorodnicyn Computing Center RAS, Federal Research Center of Informatics and Control RAS, Moscow, Vavilova 40, Russia.
E-mail address: garan@ccas.ru (V.A. Garanzha).

http://dx.doi.org/10.1016/j.matcom.2017.07.007
0378-4754/© 2017 International Association for Mathematics and Computers in Simulation (IMACS). Published by Elsevier B.V. All rights reserved.

1. Introduction

The starting point for presented research was the paper by Persson and Strang [16], where simple algorithm for construction of 2d and 3d Delaunay meshes in implicit domains was suggested. This algorithm was implemented as one-page Matlab code. The idea of algorithm [16] is the following: all mesh edges are treated as elastic struts with prescribed size distribution and mesh generation is treated as a problem of attaining equilibrium of elastic network. Target lengths are chosen in such a way that in nearly equilibrium state struts are slightly compressed. Mesh vertices which go outside the domain during relaxation are projected back to its boundary. When due to elastic relaxation the Delaunay property is lost, mesh topology is changed to recover Delaunay mesh. Signed distance function was used in [16] in order to define implicit domain. The remarkable property of the algorithm is topological regularity of resulting meshes even for chaotic initial guesses.

Comparative analysis [3] has shown that in 2d the Persson–Strang algorithm is very efficient mesh self-organization tool in a sense that for uniform meshes it produces relatively small number of irregular vertices with number of neighbors not equal to six and tend to create well balanced nonuniform meshes. It was found that it depends less on initial guess compared to the well-known Lloyd algorithm [13]. Note that in the Lloyd algorithm all mesh vertices are moved into geometrical centers of their Voronoi cells until convergence is reached. Generalizations of Lloyd algorithm for 3d meshing are considered in [1,4,18]. Lloyd-type algorithms tend to create fairly shaped Voronoi cells but eventually may admit flat tetrahedra—so-called slivers.

In [3,8] it was suggested algorithm which can be applied to non-smooth implicit functions and automatically reproduce sharp boundary edges as a subsets of boundary mesh edges without predefinition of sharp features using special sharpening forces suggested in [15]. In this paper we show that this self-organization algorithm can be implemented in the variational setting and present fairly complicated test cases with large variations of gradient of implicit function near boundary when original sharpener [15] tends to fail.

2. Description of self-organization algorithm for meshing implicit domains

Definition of implicit domain. Consider bounded domain $\Omega \subset \mathbb{R}^3$ with piecewise regular Lipschitz boundary. We assume that $\partial\Omega$ is zero level surface of the function $u(x) : \mathbb{R}^3 \to \mathbb{R}$. For interior points $u(x) < 0$, while outside the domain $u(x) > 0$. It is assumed that function $u(x)$ is piecewise smooth, Lipschitz continuous and its derivatives along certain vector field transversal to $\partial\Omega$ are not equal to zero in a finite layer S around $\partial\Omega$. In fact it is assumed that behavior of implicit function resembles that of the signed distance function. We can formalize this condition, say, by assuming the existence of quasi-isometric mapping $y(x) : \mathbb{R}^3 \to \mathbb{R}^3$, such that $y(\Omega) = \Omega_y$, and $u(x) = d_s(y(x))$, where $d_s(y)$ is the signed distance function for the domain Ω_y. Lipschitz constant for signed distance function cannot exceed unity hence range of values of local Lipschitz constants for function $u(x)$ is defined by the quasi-isometry constants of the mapping $y(x)$. The case when local Lipschitz constant has big jump near sharp edge is especially difficult from the numerical point of view which will be explained below.

We do not use this rigorous set of requirements in practice since suggested algorithm is the heuristic one.

Tetrahedral mesh in implicit domain. One should specify what kind of tetrahedral mesh is acceptable solution for meshing of implicit domain. 3d domain Ω defined by inequality $u(x) \le 0$ is approximated by polyhedron Ω_h with boundary $\partial\Omega_h$ made up from flat triangles glued along full edges. Tetrahedral mesh inside Ω_h is a normal tetrahedral partitioning. It is required that with the size of boundary edges tending to zero, distance from $\partial\Omega_h$ to $\partial\Omega$ should tend to zero. Another requirement is that the piecewise constant field of outside unit normals ν_h to $\partial\Omega_h$ should converge to piecewise continuous field of normals ν to $\partial\Omega$. Parameter h serves as a typical mesh size, namely it is assumed that maximal edge lengths of tetrahedral mesh $\mathcal{T}$ does not exceed Ch, where $C > 0$ is a constant.

From theoretical point of view it is assumed that sequence of homeomorphisms $\psi_h : \mathbb{R}^3 \to \mathbb{R}^3$ can be constructed such that $\psi_h(\Omega_h) = \Omega$, $\psi_h(\partial\Omega_h) = \partial\Omega$, and (a) for all $p \in \partial\Omega_h$ it holds $|p - \psi_h(p)| \to 0$ when $h \to 0$, (b) for all $p \in \Omega_h$, not belonging to the set of edges of $\mathcal{T}$, it holds $|\nu_h(p) - \nu(\psi_h(p))| \to 0$ when $h \to 0$. Of course, construction of such homeomorphisms in practical algorithm is not affordable and heuristic closeness measures are used.

Standard approach to tetrahedral meshing is based on the construction of triangular surface grid which approximates the boundary of the domain Ω and subsequent construction of 3d tetrahedral mesh inside resulting polyhedron, see [7]. In suggested algorithm surface and volume meshes are built simultaneously. Note that a number of algorithms for surface reconstruction are essentially based on construction of 3d Delaunay meshes, see for example [14].

Self-organization algorithm. Suppose that mesh vertices are material points which repulse each other thus modeling 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 compressible strut which tries to expand until prescribed length is reached. In case a point goes outside the domain it is projected back to its boundary. This projection procedure serves as a barrier which does not allow points to escape. As an analogy one could imagine construction foam which expands and fills the target volume producing mesh of cells. 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.

The outcome of the algorithm is certain "equilibrium" mesh where elastic forces acting on each point sum to zero. In [16] it was suggested that equilibrium mesh should be in the slightly compressed state. It turned out that this simple rule in 2d allows to obtain triangulations comparable to quality to the ones created by the advancing front technique.

3. Elastic potential

Suppose that system of points $\mathcal{E} = \{p_1, p_2, \dots, p_n\}$ in $\mathbb{R}^3$ is prescribed. Let us denote by $\mathcal{T}(\mathcal{E})$ the tetrahedral Delaunay mesh where tetrahedra violating criteria of being inside the implicit domain Ω are eliminated. The union of all tetrahedra from $\mathcal{T}$ constitutes polyhedron Ω_h which in general can provide quite poor approximation of Ω. We denote by $\mathcal{T}_e$ the set of edges of tetrahedral mesh and by $\mathcal{F}_b$ the set of boundary faces. Notation $\mathcal{E}_b$ is used for sharp boundary edges. All vertices constitute $3 \times n$ matrix P with ith column equal to p_i. We denote the set of boundary vertices by $\Gamma(P)$.

With each mesh $\mathcal{T}$ we associate the following elastic potential

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

where $W_r(P)$ is the repulsion potential, $W_s(P)$ is the sharpening potential which serves to align boundary faces with gradient of implicit function, $W_a(P)$ is the sharp edge attraction potential, and $W_d(P)$ is the deformation functional which measures shape deviation of tetrahedra from the regular ones.

Repulsion potential. The repulsion potential is written as follows

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

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

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

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

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.

Mesh size distribution is defined by function $f_h(x) : \mathbb{R}^3 \to \mathbb{R}$, $f_h(x) > 0$ which can be interpreted as a target edge length at the point x. It was shown in [16] that direct definition of function $f_h(x)$ is not convenient. It was suggested to use representation $f_h(x) = Mh(x)$, where $h(x)$ is the relative sizing function and coefficient M depends on the volume of Ω which is typically unknown at the start of the algorithm and on the characteristic mesh size. The last dependence can be replaced by the dependence on the total number of mesh points n.

Sharpening potential. The sharpening functional is written as follows

$$W_s(P) = \sum_{f \in \mathcal{F}_b} w_s(f),$$

where contribution from the boundary face f looks like

$$w_s(f) = \frac{1}{2} \frac{\text{area}(f)}{|\nabla u(c)| L_0(c)^2} \sum_{i=1}^{3} ((p_i - c)^T \nabla u(c))^2.$$

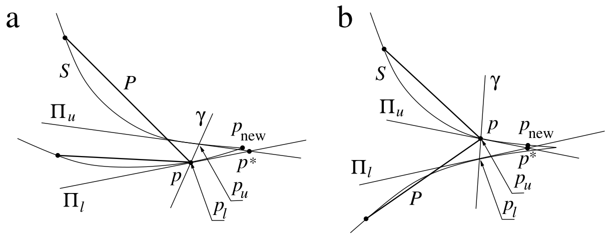

Fig. 1. Deficiency of normal alignment technique and mesh correction via attraction to sharp vertices.

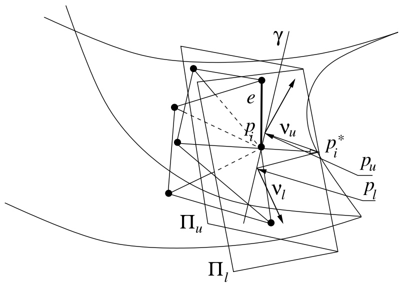

Fig. 2. Construction of 3D sharp edge attraction potential.

Here p_1, p_2, p_3 are vertices of the edge f, $c = \frac{1}{3}(p_1 + p_2 + p_3)$ is the centroid of the face and $L_0(c) = Mh(c)$.
Sharp edge attraction potential. Sharpening potential serves to align normals of boundary faces with the gradient of implicit function. Unfortunately, when boundary surface near sharp edge is highly curved this alignment may prevent precise placement of mesh vertices onto sharp boundary edge, as shown in Fig. 1.

Fig. 1 shows simple 2d examples which illustrate the case when polygon which is constructed using minimization of sharpening functional provides poor approximation to piecewise regular boundary. In the presented examples sharp corner p of polygon P is quite far from exact sharp vertex of curvilinear boundary S. In order to correct this defect the following procedure is applied. Straight line γ which is orthogonal to sharp angle bisector is constructed. This line locally intersects boundary S in two points p_u and p_l. One can find tangent directions at these points defining straight lines Π_u and Π_l, respectively. Intersection of these lines is the point $p^\star$ which can be considered as target point for optimal displacement. Vertex positions after projection onto boundary are denoted by p_{new}. Hence simple way to attract vertices to sharp edges is to add to general functional a term

$$\frac{1}{2}|p - p^*|^2.$$

Construction of the sharp edge attraction potential in the 3d case is illustrated in Fig. 2.

Consider boundary sharp edge e and one of its vertices denoted by p_i. Edge is sharp when angle between normals to adjacent faces is above $\pi/5$. Consider straight line γ, which is orthogonal to the bisector plane of the dihedral angle at e. This line locally intersect exact boundary $\partial\Omega$ in two points p_l and p_u. Gradient of implicit function at these points define unit vector ν_l and ν_u, respectively, which in turn define two planes $\Pi_l = \left\{x : \nu_l^T(x - p_l) = 0\right\}$ and $\Pi_u = \left\{x : \nu_u^T(x - p_u) = 0\right\}$. Denote by p_i^* the orthogonal projection of the point p_i onto intersection line of these two planes. Hence vertex p_i adds the term $\frac{1}{2}\theta_e|p_i - p_i^*|^2$ to the global edge attraction functional.

The resulting expression for edge attraction functional can be written as follows

$$W_a(P) = \sum_{e\in\mathcal{E}_b} w_a(e), \tag{2}$$

$$w_a(e) = \frac{1}{2}(|p_i - p_i^*|^2 + |p_j - p_j^*|^2),$$

where p_i, p_j are vertices of the edge e. In this formula the direction of line γ is the same for both vertices since it is defined by the bisector of the dihedral angle, but the pairs of tangent planes are different.

Deformation potential. The deformation potential is defined as follows

$$W_d(P) = \sum_{T \in \mathcal{T}(\mathcal{E})} w_d(T),$$

where contribution from tetrahedron T with vertices p_i is given by

$$w_d(T) = \frac{L_0^2}{6 \cdot 2^{1/3}} \frac{\sum_{i \neq j} |p_i - p_j|^2}{((p_4 - p_1)^T (p_3 - p_1) \times (p_2 - p_1))^{\frac{2}{3}}}$$

where $L_0(c) = Mh(c)$ and $c = \frac{1}{4}(p_1 + p_2 + p_3 + p_4)$. Coefficient L_0^2 serves to balance the dimension of the terms of functional. Shape deformation potential is just a measure of deformation which can be expressed as the ratio of arithmetic mean to harmonic mean of the singular values of the affine map from unit tetrahedron to current tetrahedron. It is used for mesh optimization for many years [12] and was used to check the quality of tetrahedra in [10]. For visualization purposes we use inverse value as a shape quality measure

$$q_d(T) = 6 \cdot 2^{1/3} \frac{((p_4 - p_1)^T (p_3 - p_1) \times (p_2 - p_1))^{\frac{2}{3}}}{\sum_{i \neq j} |p_i - p_j|^2}. \tag{3}$$

Variational formulation. Elastic network is in the equilibrium state when current position matrix P provides local minimum of elastic energy $W(P)$ provided that the vertices of faces from $\mathcal{F}_b$ belong to the surface $u(x) = 0$. All vertices which go outside the domain should be projected back to its boundary.

Hence equilibrium state of elastic network is the solution $\tilde{P}$ of the variational problem

$$\min W(P)$$

under the constraint

$$u(p_i) \leq 0, \quad i = 1, \ldots, n.$$

The necessary condition for minimum are the well-known Karush Kuhn Tucker (KKT) conditions: there exists vector $\Lambda = (\lambda_1 \ldots \lambda_n)^T \in R^n$ such that

$$L_W(\tilde{P}) = \min L_W(P),$$

where

$$L_W(P) = W(P) + \sum_{i=1}^{n} \lambda_i u(p_i)$$

is the Lagrange function, and $\lambda_i \geq 0$ are the Lagrange multipliers which satisfy the complementarity slackness condition

$$\lambda_i u(\tilde{p}_i) = 0$$

and dual feasibility condition

$$\lambda_i \geq 0.$$

Solution $\tilde{P}$ satisfies also primal feasibility condition

$$u(\tilde{p}_i) \leq 0.$$

In the case when functions W and u are smooth the solution satisfies the set of differential equations

$$\frac{\partial W}{\partial p_i} + \lambda_i \frac{\partial u}{\partial p_i} = 0.$$

For internal vertices $u(p_i) < 0$ and $\lambda_i = 0$, hence standard stationarity conditions hold

$$\frac{\partial W}{\partial p_i} = 0.$$

For boundary vertex p_i equality

$$u(p_i) = 0$$

holds and Lagrange multipliers satisfy inequality $\lambda_i > 0$.

When elastic energy and constraint function $u(x)$ are not smooth one should consider sub-gradient version of KKT conditions. Note however that from practical point of view computation of tangent cones for non-regular functions in our case is too expensive hence finite difference method is used to compute approximation of gradient vector.

Coordinates of boundary points can be found using projected gradient search technique: points are displaced along anti-gradient of W and in the case when p_i goes outside domain, it is projected back to its boundary.
Theoretical self-organization algorithm. Algorithm of mesh self-organization is formulated as follows. For $k = 0, 1, \dots$ do

(a) For a matrix of positions P^k construct the Delaunay tetrahedral mesh or rebuild existing one if necessary;
(b) Tetrahedra which do not belong to the domain Ω are eliminated using special set of criteria. The resulting mesh is denoted by $\mathcal{T}^k$ and the set of its boundary faces is denoted by $\mathcal{F}_b^k$;
(c) One step of conditional minimization of the elastic energy is performed

$$\tau^k = \operatorname{argmin}_\tau(W(V(P^k + \tau\delta P^k)))$$

$$P^{k+1} = V(P^k + \tau^k\delta P^k).$$

Here matrix δP^k is the direction of minimization, and V is the operator of projection for points from $\Gamma(P)$ onto $\partial\Omega$ along the integral curves of the function $u(x)$. In order to compute direction matrix we use variant of projected gradient method with preconditioning.

Operator V projects mesh boundary vertex p with $u(p) > 0$ onto surface $u(x) = 0$ using simple iterative algorithm

$$p^{m+1} = p^m - \tau_1 \frac{u(p^m)\nabla u(p^m)}{|\nabla u(p^m)|^2}. \tag{4}$$

Here m is the local iteration count. Formula (4) defines orthogonal projection onto current guess for tangent plane and parameter $\tau_1 \leq 1$ serves as a relaxation parameter which stabilizes iterative process for sharply varying function $u(x)$. Iterations are repeated until deviation of p^m from $\partial\Omega$ is below the threshold ϵ_b. The deviation measure looks as follows

$$|u(p^m)| \leq \epsilon_b|\nabla u(p^m)|. \tag{5}$$

Parameter ϵ_b has a meaning of geometry definition error. In [16] it was suggested to define it as $\epsilon_b = h/10$. Note that when possible exact projection is used. Say for triangulated surface the signed distance function is used which provides the projection point during evaluation. For approximate projection we use threshold value equal to 10^{-8} times the diameter of the domain.

The steps of the Delaunay re-meshing, tetrahedra elimination and elastic relaxation are repeated until convergence is reached. One should note however that sometimes oscillating process is obtained when due to topological mesh changes steady state is not reached. In this case stopping criteria is based on the mesh quality and on the criteria of approximation of $\partial\Omega$ by $\partial\Omega_h$.

The procedure of minimization of elastic energy looks more or less standard but it turned out that computation of search directions required quite elaborate modifications near boundary in order to allow stable recovery of sharp edges.

In order to compute the gradient of elastic potential one should sum contributions from the mesh tetrahedra, edges, boundary faces and sharp edges

$$\frac{\partial W}{\partial p_i} = \sum_{e\in\mathcal{T}_e, p_i\in\mathcal{V}(e)} \theta_r \frac{\partial w_r}{\partial p_i} + \sum_{f\in\mathcal{F}_b, p_i\in\mathcal{V}(f)} \theta_s \frac{\partial w_s(f)}{\partial p_i} + \sum_{e\in\mathcal{E}_b, p_i\in\mathcal{V}(e)} \theta_a \frac{\partial w_a(e)}{\partial p_i} + \sum_{T\in\mathcal{T}, p_i\in\mathcal{V}(T)} \theta_d \frac{\partial w_d}{\partial p_i}.$$

Here notation $\mathcal{V}(Q)$ denotes the set of vertices of the geometric object Q.

Let us write down derivative of local potential for boundary face:

$$\frac{\partial w_s}{\partial p_i} = \frac{\text{area}(f)}{(L_0(c))^2} g(p_i - c)^T g + \frac{\sum_{j=1}^3((p_j - c)^T g)^2}{2(L_0(c))^2} \frac{\partial \text{area}(f)}{\partial p_i}$$

where the dependence of gradient direction $g(c) = \nabla u(c)/|\nabla u(c)|$ on p_i is neglected. Moreover, in practical algorithm derivative of face area is neglected as well. The derivative of local potential which attracts point to sharp edges is just

$$\frac{\partial w_a}{\partial p_i} = (p_i - p_i^*)$$

since we neglect the dependence of projection p_i^* on p_i.

Derivative of logarithmic potential for the edge with vertices p_l, p_m looks like

$$\frac{\partial w_r}{\partial p_l} = \begin{cases} \left(1 - \frac{L_0}{L}\right)\frac{L_0}{L}(p_l - p_m) & \text{when } L < L_0 \\ 0 \quad \text{when } L \geq L_0. \end{cases}$$

The derivative of deformation term can be written as

$$w_d(T) = C\frac{\varphi}{J^{2/3}}, \quad C = \frac{L_0^2}{6 \cdot 2^{1/3}}$$
$$\varphi = \sum_{k \neq m} |p_k - p_m|^2, \qquad J = (p_4 - p_1)^{\mathbb{T}}(p_2 - p_1) \times (p_3 - p_1)$$
$$\frac{\partial w_d(T)}{\partial p_i} = C\left(\frac{1}{J^{2/3}}\frac{\partial \varphi}{\partial p_i} - \frac{2}{3}\frac{\varphi}{J^{5/3}}\frac{\partial J}{\partial p_i}\right).$$

4. "Elastic forces" and practical iterative algorithm

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

Roughly speaking, these "forces" are introduced as follows

$$\begin{aligned} \delta p_i^k &= -\frac{\theta_r}{d_i^k}\frac{\partial W_r}{\partial p_i}(P^k) - \frac{\theta_d}{d_i^k}\frac{\partial W_d}{\partial p_i}(P^k) - \frac{\theta_s}{d_i^k}\frac{\partial W_s}{\partial p_i}(P^k) - \frac{\theta_a}{d_i^k}\frac{\partial W_a}{\partial p_i}(P^k) \\ &= F_e(p_i^k) + F_d(p_i^k) + F_s(p_i^k) + F_a(p_i^k), \end{aligned} \tag{6}$$

where

$$d_i^k = d_{r_i}^k + d_{d_i}^k + d_{s_i}^k + d_{a_i}^k;$$

are the scaling factors which are computed using simplified expressions for second derivatives of expansion, deformation, sharpening and sharp edge attraction potentials, respectively, which is explained below.

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

In order to present precise formulae for computation of forces it is convenient to introduce the following notations. Let $\text{star}_e(p_i)$ denote the set of the mesh edges originating from the vertex p_i, while $\text{star}(p_i)$ will denote the set of vertices of these edges excluding p_i. It is convenient also to use surface star of the boundary vertex: $\text{star}_f^s(p_i)$ is the set of boundary faces adjacent to boundary vertex p_i, $\text{star}_e^s(p_i)$ is the set of boundary edges and $\text{star}^s(p_i)$ is the set of boundary vertices of the star. 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 upper index k.

Repulsive "force". For internal vertex p_i

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

where

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

On the boundary this formula is modified

$$F_r(p_i) = -\frac{\theta_r}{d_i}\sum_{p_j \in \text{star} p_i} \phi_r(p_i, p_j)\Pi_{ij}(p_i - p_j),$$

where operator Π_{ij} is defined as follows

$$\Pi_{ij}(p_i - p_j) = \begin{cases} p_i - p_j & \text{when } p_j \notin \text{star}^s(p_i) \\ (I - \nu_i \nu_i^T)(p_i - p_j) + \max(0, \nu_i^T(p_i - p_j))\nu_i, \end{cases}$$

and I is the notation for the unity matrix.

This nonlinear operator eliminates the component of expanding force directed inside the domain. Here vector ν_i is the discrete outer unit normal to the surface mesh in the vertex p_i. It is computed using angle-averaging of normal vectors to the triangles adjacent to p_i:

$$\nu_i = \frac{\sum_{f \in \text{star}^s_f(p_i)} \alpha_i(f)\nu(f)}{\left|\sum_{f \in \text{star}^s_f(p_i)} \alpha_i(f)\nu(f)\right|}. \tag{7}$$

In this formula $\nu(f)$ is the outer unit normal to the triangular face f while $\alpha_i(f)$ if the angle of f at the vertex p_i. This modification eliminates the possibility of repulsive force to be directed inside the domain which may happen on concave or saddle surfaces when repulsion of boundary vertices from each other may result in forces pointed inside the domain.

Another modification is applied to repulsive force when the Delaunay edge becomes a compressed strut between two opposite boundaries near sharp corner. Sharp corners are handled using special affine stretching which opens up the corner. Consider internal Delaunay edge e of the internal Delaunay tetrahedron T and assume that both vertices p_1 and p_2 of this edge lay on the boundary of domain. Suppose that gradients of u at these vertices are opposite in the following sense. For vertices p_1, p_2 define the set of gradient directions G_1, G_2 which include $\nabla u(p_i)/|\nabla u(p_i)|$ and $\nabla u(c(f))/|\nabla u(c(f))|$, $f \in \text{star}^s_f(p_i)$, i.e. the set of gradient directions in the centroids of triangles adjacent to p_i.

Now compute the directions g_1^* and g_2^* as the angle maximizers

$$\pi - \alpha(T) = \max \angle(g_1, g_2) \quad \text{for } g_1 \in G_1, g_2 \in G_2. \tag{8}$$

If maximal angle is above $\frac{3}{4}\pi$ special anisotropic scaling coefficient k is computed otherwise it is just set to $k = 1$. Consider two planes $g_i^{*T}(x - p_i) = 0, i = 1, 2$. These planes intersect through certain straight line which is orthogonal to g_1^* and g_2^*. Creating circular cylinder centered around this axis and passing through centroid denoted by $c(T)$ we get two parameters—radius R of this cylinder and width $D = R\alpha$ of the cylinder part laying inside the dihedral angle between two planes. If inequality

$$D < Mh(c(T))$$

holds, then

$$k = \frac{D}{Mh(c(T))}. \tag{9}$$

Otherwise k is set to one. Geometrical meaning of parameter R, α and D is illustrated on Fig. 3.

In the case $k < 1$ the length L of the edge e is recomputed in the transformed coordinates

$$x' = ((I - \nu\nu^T) + \frac{1}{k}\nu\nu^T)x, \quad \nu = \frac{g_1^* - g_2^*}{g_1^* - g_2^*} \tag{10}$$

which means that this edge is enlarged in the direction ν and value L in the expression for ϕ_e is replaced by the L':

$$L' = \left(L^2 + \left(\frac{1}{k^2} - 1\right)(\nu^T(p_1 - p_2))^2\right)^{\frac{1}{2}}.$$

This correction serves to diminish the amount of repulsive force acting on the boundary vertices near sharp angle and tend to create internal edges which are less inclined with respect to direction ν. When k is too small it is replaced by certain positive threshold. Edges marked for enlargement are shown in Fig. 4.

Shape deformation force. Shape deformation "force" is computed as follows

$$F_d(p_i) = -\frac{\theta_d}{d_i} \sum_{T, p_i \in T} \chi(p_i, T), \qquad d_{di} = -\sum_{T, p_i \in T} \zeta(p_i, T)$$

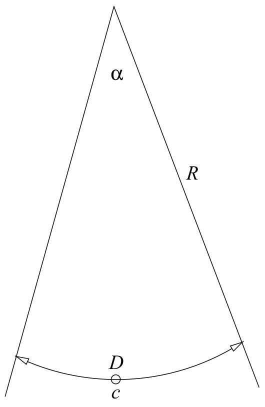

Fig. 3. Anisotropic scale change inside acute corner.

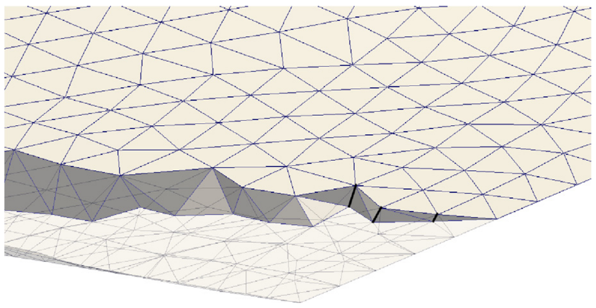

Fig. 4. Bold edges are marked for affine mapping (10).

where

$$\chi(p_i,T)=\frac{\partial w_d}{\partial p_i}=C\left(\frac{1}{J^{2/3}}\frac{\partial\varphi}{\partial p_i}-\frac{2}{3}\frac{\varphi}{J^{5/3}}\frac{\partial J}{\partial p_i}\right),$$

and

$$\begin{aligned}\zeta(p_i,T)&=\operatorname{tr}\operatorname{diag}\frac{\partial^2 W_f(p)}{\partial p_i\partial p_i^T}\\&=\operatorname{tr}\operatorname{diag}C\left(\frac{1}{J^{2/3}}\frac{\partial^2\varphi}{\partial p_i\partial p_i^T}-\frac{2}{3}\frac{1}{J^{5/3}}\left(\frac{\partial J}{\partial p_i}\frac{\partial\varphi}{\partial p_i^T}+\frac{\partial\varphi}{\partial p_i}\frac{\partial J}{\partial p_i^T}\right)+\frac{10}{9}\frac{\varphi}{J^{8/3}}\left(\frac{\partial J}{\partial p_i}\frac{\partial J}{\partial p_i^T}\right)\right).\end{aligned}$$

For convenience we introduce local numbering in tetrahedron T and associate vertex p_i with local vertex p_1. Then

$$\varphi=\sum_{m\neq j}|p_m-p_j|^2,\qquad J=(p_4-p_1)^T(p_3-p_1)\times(p_2-p_1)$$

and

$$\begin{aligned}&\frac{\partial\varphi}{\partial p_1}=2\sum_{j=2}^{4}(p_1-p_j)\\&\frac{\partial J}{\partial p_1}=-(a^{23}+a^{34}+a^{42}),\qquad a^{jm}=(p_m-p_1)\times(p_j-p_1).\end{aligned}$$

If tetrahedron is situated inside sharp corner of domain then additional affine mapping (10) should be applied and shape distortion measure is computed in uniform coordinate frame.

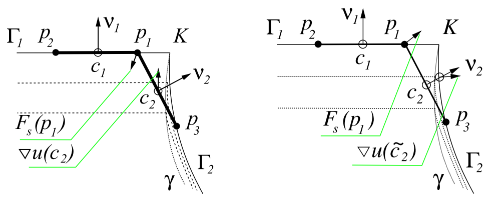

Fig. 5. (left) Unstable case for sharpening force, (right) stabilized sharpening force.

For boundary vertices deformation force is modified as follows

$$F_d(p_i) = -\frac{\theta_d}{d_i} R_i \left(\sum_{T, p_i \in T} \chi(p_i, T) \right)$$

where nonlinear operator R_i is defined as

$$R_i(a) = (I - \nu_i \nu_i^T)a + \max(0, \nu_i^T a)\nu_i.$$

This modification means that the component of the deformation force which is directed along interior discrete normal to the domain boundary is eliminated in order to prevent competition of repulsion and deformation forces.
Sharpening force. Sharpening force can be written as follows [2]

$$F_s(p_i) = -\frac{1}{d_i} \sum_{f \in \mathrm{star}_f^s p_i} \psi(p_i, f) g(p_i, f), \qquad d_{si} = \sum_{f \in \mathrm{star}_f^s p_i} \Psi(p_i, f), \quad g(p_i, f) = \frac{\nabla u(\tilde{c}(f))}{|\nabla u(\tilde{c}(f))|}$$

$$\Psi(p_i, f) = \frac{\mathrm{area}(f)}{(Mf(\tilde{c}(f)))^2}, \qquad \psi(p_i, f) = \Psi(p_i, f)(g - p_i)^T g.$$

Here $\tilde{c}(f)$ is the intersection point of the surface

$$u(x) = -\min_i |u(p_i(f))|,$$

where p_i are the vertices of the face f, with the straight line

$$x(t) = c + \nu(f)t$$

which is orthogonal to the plane of f and passes through its centroid. This sharpening force is not the same as in [15] due to the additional projection procedure. This modification plays critical role when implicit function $u(x)$ is far from the distance function. In particular, when gradients of the implicit function near different banks of the sharp edge differ, say by the order of magnitude. In this case original sharpening force can result in numerical instability as shown in Fig. 5, left. Here dashed lines are implicit function isolines and dotted line γ marks the boundary between domains with highly different Lipschitz constants.

All boundary mesh vertices are grouped into “smooth” and “non-smooth” ones according to the absolute value of the discrete Laplace–Beltrami operator of the surface representation at the vertices. Mean value Laplacian discretization due to Floater [5] is used which has a meaning of the Laplace–Beltrami operator of the surface elevation in the local tangent plane integrated over certain contour around the mesh vertex.

$$Bx(p_i) = \sum_{f \in \mathrm{star}_f^s(p_i)} \tan\left(\frac{1}{2}\alpha_{j+\frac{1}{2}}\right) \left(\frac{\nu_i^T p_j - \nu_i^T p_i}{|p_j - p_i|} + \frac{\nu_i^T p_{j+1} - \nu_i^T p_i}{|p_{j+1} - p_i|} \right).$$

Here ν_i is the discrete unit normal at the vertex p_i defined in (7), p_i, p_j, p_{j+1} are vertices of triangle f and $\alpha_{j+\frac{1}{2}}$ is the angle of f at p_i. We use simple criterion

$$|Bx(p_i)| > \frac{7}{10} \tag{11}$$

in order to mark non-smooth vertices. Note that expression for $Bx(p_i)$ is dimensionless and serves as a measure of angle sharpness at the vertex p_i. Evidently for coarse meshes this measure tends to over-predict angle sharpness.
Sharp edge attraction force. Sharp edge attraction force can be written as follows

$$F_a(p_i) = -\frac{1}{d_i} \sum_{e \in s_e(p_i)} (p_i - p_i^*)$$

where $s_e(p_i)$ is the set of boundary sharp edges adjacent to boundary vertex p_i. Here d_{ai} is just the number of such an edges for a point p_i.
Interaction of the "forces" on the boundary. First consider the case of "non-smooth" boundary vertex. We use notation F_e for volumetric "elastic" surface

$$F_e = F_r + F_d$$

and notation F_{sa} for the sum of sharpening and sharp edge attraction forces

$$F_{sa} = F_s + F_a.$$

Consider orthogonal decomposition of computed forces with respect to gradient direction ∇u:

$$F_e = F_{en} + F_{e\tau}, \quad F_{e\tau} = \frac{1}{|\nabla u|} \nabla u^T F_e, \; F_{en} = F_e - F_{e\tau}$$

$$F_{sa} = F_{san} + F_{sa\tau}, \quad F_{sa\tau} = \frac{1}{|\nabla u|} \nabla u^T F_{sa}, \; F_{san} = F_{sa} - F_{sa\tau}.$$

Tangent component of elastic force is orthogonalized with respect to the tangent component of the sharpening force:

$$F'_{e\tau} = F_{e\tau} - \frac{w}{|F_{sa\tau}|} F_{e\tau}^T F_{sa\tau}$$

and final formula for elastic force on the boundary looks like

$$F_e = F_{en} + F'_{e\tau}.$$

Parameter w depends on the angle between vectors ∇u and F_{sa}:

$$w = \min\left(\frac{\sin \angle(\nabla u, F_{sa})}{\sin \frac{\pi}{9}}, 1 \right).$$

Parameter w is introduced to account for the fact that tangential component of the sharpening force could be zero.

This splitting helps mesh vertices to travel to their positions on sharp edges under sharpening force while expanding and deformation forces tend to distribute vertices along sharp edges.

For smooth boundary vertices the tangent component of the sharpening force is just set to zero while elastic force is used as is:

$$F = F_e + F_{san}.$$

This decomposition helps to smooth out deviations from the iso-surface while shape of triangles on the smooth boundary tends to be quite regular.

5. Practical boundary approximation error measures

Consider practical error measures for approximation of the domain boundary. Consider boundary mesh face f. Let us denote by $\tilde{c}(f)$ is the intersection point of the surface

$$u(x) = 0,$$

with the straight line

$$x(t) = \frac{1}{3}(p_1(f) + p_2(f) + p_3(f)) + \nu(f)t$$

which is orthogonal to the plane of f and passes through its centroid. Here $p_i(f)$ are the vertices of the face f. Consider tetrahedron $T_b(f)$ with vertices $\tilde{c}(f)$, $p_1(f)$, $p_2(f)$, $p_3(f)$. The height of this tetrahedron over plane of f is $H(f)$. We can use the height of these auxiliary tetrahedra as a maximal measure of deviation of $\partial\Omega_h$ from $\partial\Omega$ as

$$E_d^\infty = \max_{f\in\partial\Omega_h} H(f).$$

Integral deviation is simply the total volume of auxiliary tetrahedra

$$E_d^1 = \max_{f\in\partial\Omega_h} \mathrm{vol}(T(f)).$$

The ratio $E_d^1/\mathrm{vol}(\Omega_h)$ can be used as a relative integral error measure. To check the quality of the sharp edge reconstruction one should use maximal boundary shape approximation error

$$E_s^\infty = \max_{f\in\partial\Omega_h} \frac{H(f)}{L_{\max}(f)} \tag{12}$$

where $L_{\max}(f)$ is the maximal edge length of the boundary face f. Integrating shape error over boundary one can obtain average shape approximation error E_s^1. The above error measures are still heuristic but in practice provide very good closeness criteria. Note that ratio

$$\frac{H(f)}{L_{\max}(f)} \tag{13}$$

can be visualized for each surface triangle.

Unfortunately, these error measures are almost insensitive to mesh deviations shown in Figs. 1 and 2. In these cases auxiliary tetrahedra have very small height. The sharp edge attraction potential by itself is a reasonable error measure. One can make shape error estimate more robust adding the following formula to the above set

$$E_a^\infty = \max_{e\in\partial\Omega_h} \max_{p\in e} \frac{|p-p^*|}{L(e)}. \tag{14}$$

Here p^* is the orthogonal projection of vertex p onto predicted sharp edge. This estimate can be computed for all boundary edges where angle between normals to adjacent faces is above certain threshold, in our case it is equal to $\pi/5$.

So iterations can be stopped when the closeness condition is below given threshold and mesh size is close to predefined distribution.

Note that the declared goal of the algorithm—construction of equilibrium mesh, in many cases is not attained. Geometrical oscillations may arise due to the appearance of the almost circumferential sets of vertices which create the Delaunay tetrahedra being unstable with respect to small displacement of vertices. However computational domain is not disrupted by topological switches.

6. Elimination of tetrahedra

Basic Delaunay tetrahedrization algorithm creates convex envelope of the set of the input points. After creation of tetrahedra special shelling procedures should be applied in order to eliminate tetrahedra not belonging to domain.

We use the following set of criteria for shelling:

1. Elimination of tetrahedra with centroids not belonging to domain, i.e.

$$u(c) > 10^{-10}.$$

Start the cycle of shelling

2. Check the admissibility of tetrahedron for elimination. If tetrahedron has two or more boundary faces it is admissible. If it has only one boundary face then orthogonal projection of internal vertex onto boundary face should have barycentric coordinates μ_1, μ_2, μ_3 satisfying inequality $\mu_i > -\frac{1}{10}$.

For admissible tetrahedra do the checks 3, 4, 5 below

3. Eliminate tetrahedra T which are slightly inside domain

$$u(c(T)) > -k\varepsilon_b|\nabla u(c(T))|.$$

Here $k = k(T)$ is the local stretching parameter for the sharp corner defined in (10)

4. Eliminate tetrahedra with bad shape.

Denote by $l_{\max}(T)$ the solution of the max-problem

$$l_{\max} = \max_{e \in \mathcal{E}(T)} \frac{|e|}{Mh(c(e))}.$$

Tetrahedron T with bad shape

$$H_{\min}(T) < \frac{1}{20} Mh(c(T)) l_{\max}$$

is marked for elimination. Here $H_{\min}(T)$ is minimal height of tetrahedron, $\mathcal{E}(T)$ is the set of edges of T. If T is marked for elimination and $k < 1$ one can apply to T affine mapping (10). If after stretching the tetrahedron T is still badly shaped it is eliminated.

5. Elimination using dihedral angle threshold

Define dihedral angle threshold as $\beta = \frac{1}{18}\pi$. In the case when tetrahedron T lies inside sharp corner we use reduced threshold

$$\beta = \frac{1}{2} \max \left(\frac{1}{36}\pi, \alpha_c(T) \right)$$

where by $\alpha_c(T)$ the angle of the local sharp corner defined in (8). Tetrahedron is eliminated when

$$\alpha_{\min}(T) < \beta, \quad \text{or} \quad \alpha_{\max}(T) > \pi - \beta$$

where $\alpha_{\min}(T)$ and $\alpha_{\max}(T)$ are minimal and maximal dihedral angles of tetrahedron T.

if during steps 3, 4, 5 some tetrahedra are eliminated process is repeated.

Sliver elimination. We have found experimentally that adding shape deformation potential was sufficient in order to eliminate slivers. It is important to note that shape optimization prevents recovery of sharp edges on the boundary and should applied only after domain boundary reconstruction. During cell shape optimization vertices move along the boundary and along sharp edges. From time to time slivers reappear due to topological changes restoring Delaunay property. Bur we have found that at certain iterations mesh is sliver free and approximation of boundary is not deteriorated.

7. Numerical experiments

In all test cases below uniform initial mesh is defined by dense hexagonal lattice.

In most test cases we use the following set of weights in representation of functional (1) and in the final representation of "forces" (6). For the boundary shape recovery stage we use the weights $\theta_d = \theta_a = 0$, $\theta_r = \theta_s = \frac{1}{2}$ since cell shape optimization prevents boundary recovery. At the mesh optimization stage the set of weights $\theta_r = \frac{3}{10}, \theta_s = \frac{6}{10}, \theta_a = 0, \theta_d = \frac{1}{10}$ is used. Number of iterations for domain boundary recovery can be quite large, while for mesh optimization $10 - 15$ iterations were found to be enough.

Sharp edge attraction term is tested in the last and most complicated test case which is explained below.

Example 1. First results for model "cube with extracted ball" from [15] are presented. This model is defined by the following implicit function

$$u(x) = \max \left(\max\left(|x_1|, |x_2|, |x_3|\right) - 12, -\sqrt{x_1^2 + x_2^2 + x_3^2} + \frac{31}{2} \right).$$

Due to errors in the definition of initial domain mesh is not covering exact domain which is shown in Fig. 6(a). We have decided to retain this erroneous initial guess to underline the ability of point repulsion technique in handling bad initial meshes.

After few iterations mesh fills the domain (Fig. 6(b)). Eventually all sharp edges are recovered (Fig. 6(c)) and after cell shape optimization boundary approximation is not deteriorated (Fig. 6(d)).

Fig. 7 illustrates efficiency of mesh optimization stage. Considerable number of tetrahedra with dihedral angles below 15° are present in the mesh after domain recovery stage (Fig. 7(a)). All these almost flat tetrahedra are eliminated in the process of mesh optimization, as shown in Fig. 7(b).

Fig. 8 shows histogram of dihedral angle distribution before (a) and after (b) optimization, while Fig. 9 illustrates behavior of cell shape quality measure (3) before and after optimization. Limiting values are shown on x-axis.

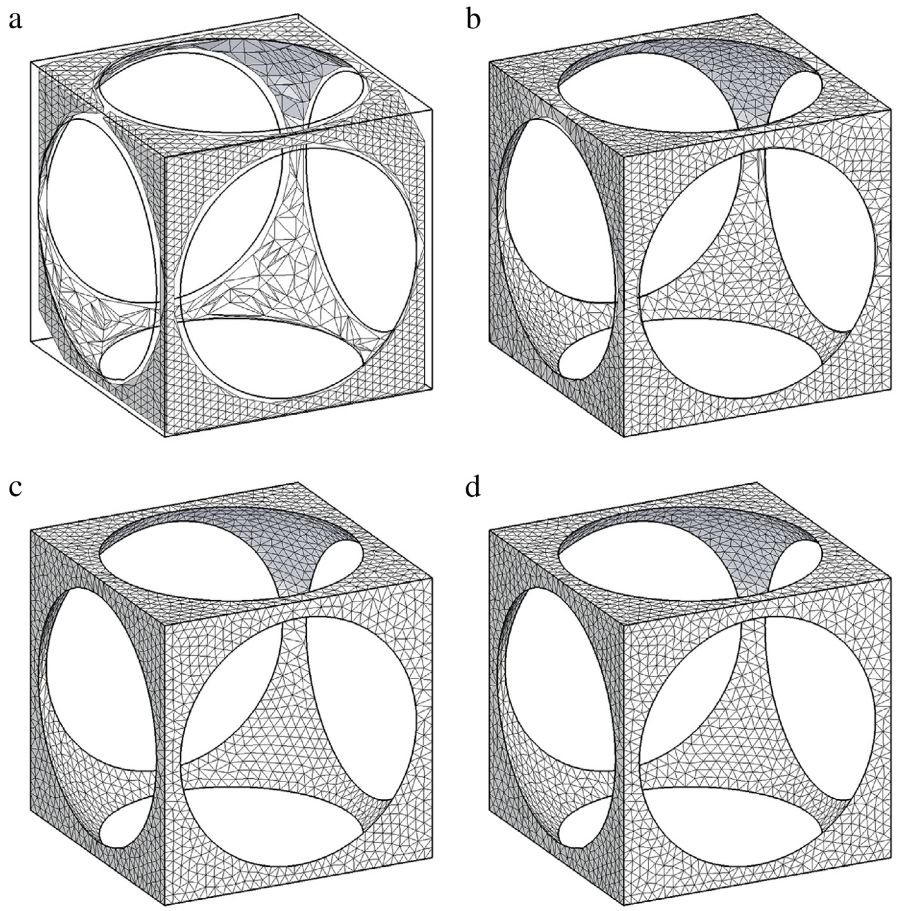

Fig. 6. Delaunay mesh for “cube minus ball” model from [15]: (a) initial guess, (b) intermediate mesh, (c) mesh after domain recovery, (d) mesh after optimization.

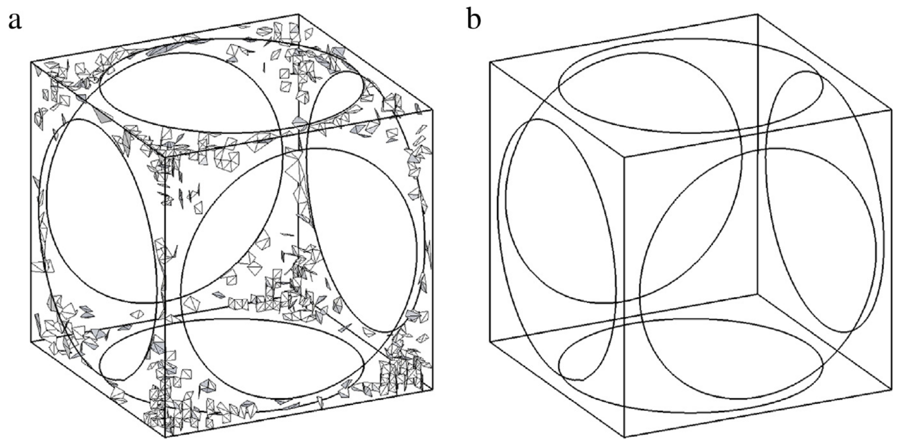

Fig. 7. (a) Tetrahedra with dihedral angle less than 15° after domain recovery, and (b) after shape optimization.

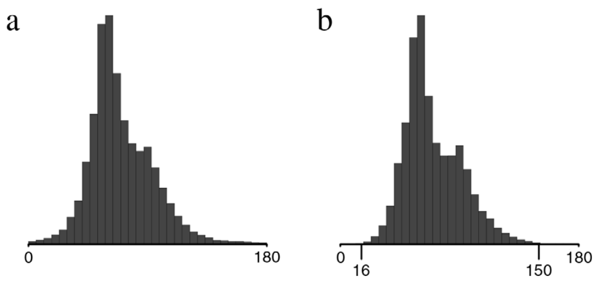

Fig. 8. (a) Dihedral angle distribution before cell shape optimization and (b) after shape optimization.

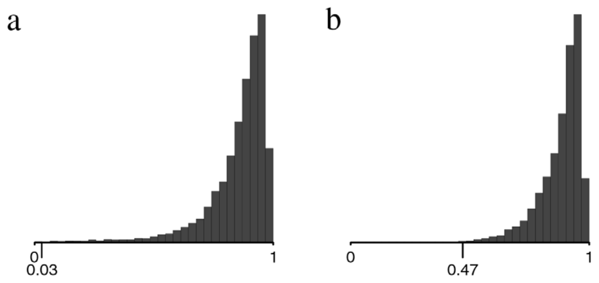

Fig. 9. (a) Tetrahedral shape measure distribution before cell shape optimization and (b) after shape optimization.

Chordal error for boundary faces and shape error for boundary faces are presented in Fig. 10. Here mean mesh edge length is about 1 and radius of extracted circle is equal to 15.5. Hence computed values of error are in very good accordance with these value and imply precise recovery of sharp edges. Note also that shape optimization of Delaunay mesh does not lead to deterioration of domain boundary approximation as shown in Fig. 10(c), (d).

Example 2. Next test case suggested in [15] is the curved icosahedron model. We were not able to reproduce this model exactly and constructed the body of interest by subtracting twenty balls with radius $53/50$ from the ball with radius $7/5$. Centers of smaller balls coincide with vertices of regular dodecahedron inscribed in larger sphere.

Fig. 11 shows initial mesh after elimination of exterior tetrahedra (a), during the domain boundary recovery stage (b), precisely after recovery of boundary (c) and after mesh optimization (d).

Fig. 12 shows boundary vertices which are marked as "sharp" using discrete Laplacian threshold (11) for intermediate solution (a) and for final mesh (b). Fig. 13 illustrates how almost flat tetrahedra are eliminated from the mesh during mesh optimization without deterioration of approximation quality for sharp edges.

Dihedral angle histograms on Fig. 14 do not contain small angles of the tetrahedra which are adjacent to the summits of the curved peaks of icosahedron. Thin tetrahedra near curved peak summits are not slivers (see Fig. 15).

Example 3. Next model from [15] is the "twisted prism". Since precise model description was not available we created reconstruction which resembles model from [15].

Our twisted prism model is constructed by sweeping planar rectangle defined on the plane $x_3 = 0$. During sweep the size of rectangle is changed and it is rotated around vertical axis. Rotation is defined by the spatial transformation $y(x)$

$$y_1 = x_1 \cos\phi - x_2 \sin\phi, \qquad y_2 = x_1 \sin\phi + x_2 \cos\phi, \qquad y_3 = x_3$$

where

$$\phi = 2\pi(x_3 + 1).$$

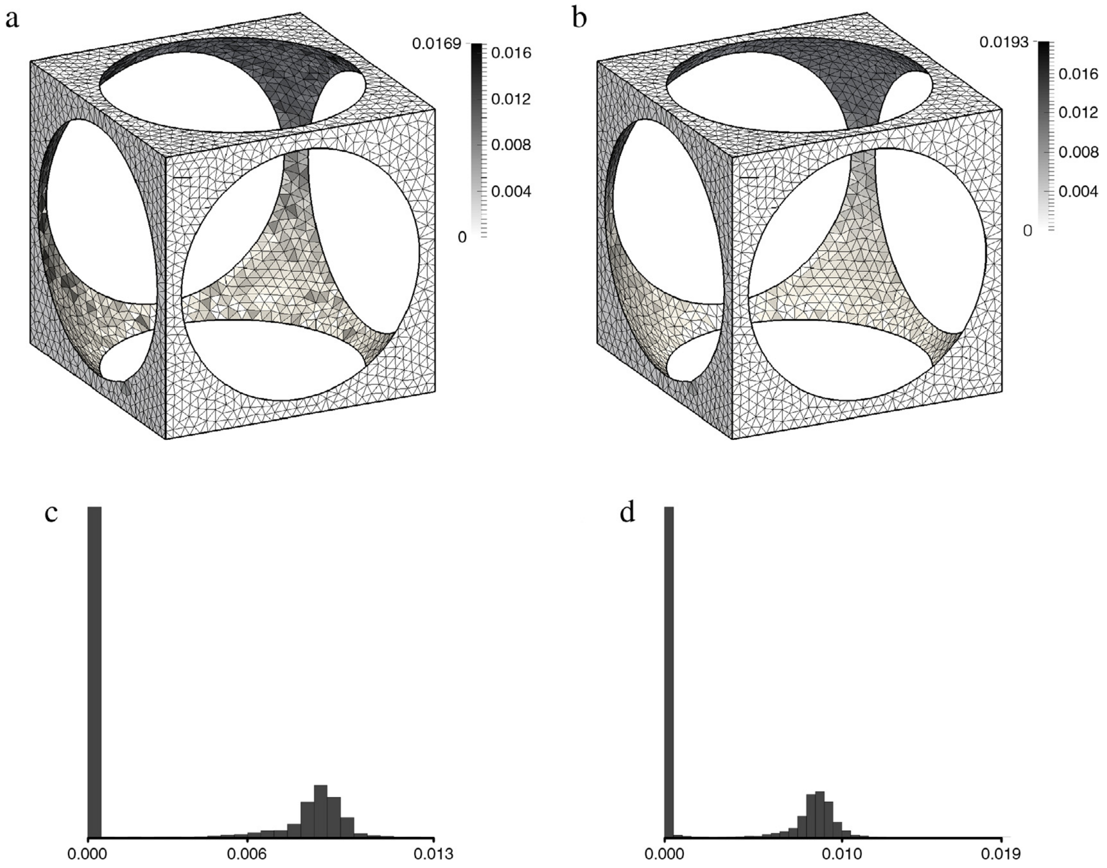

Fig. 10. (a) Chordal error distribution, (b) boundary shape error distribution, (c) boundary shape error histogram before optimization and after optimization.

Below we define auxiliary implicit functions for swept figure

$$u_1(y) = \frac{3}{10}\left(\max\left(5|y_1|, \frac{5}{2}|y_2|\right) - \frac{4}{5}(y_3 - 1)^2\right),$$

for bounding cuboid

$$u_2(y) = \max\left(\frac{|y_1|}{2}, \frac{|y_2|}{2}, |y_3|\right) - 1$$

and for ball-like fragment to be extracted from the body

$$u_3(y) = \sqrt{y_1^2 + y_2^2 + \left(y_3 + \frac{3}{10}\right)^2} - \frac{7}{20}.$$

The final representation of implicit function is constructed as

$$u(x) = \max(\min(u_1(y(x)), u_2(y(x)), -u_3(y(x)))).$$

Results for this model are shown in Fig. 16.

Fig. 16 shows initial mesh after elimination of exterior tetrahedra (a), during the domain boundary recovery stage (b), precisely after recovery of boundary (c) and after mesh optimization (d).

Fig. 17 illustrates how mesh optimization allows to eliminate almost flat tetrahedra. Boundary of this model contain sharp edges with internal dihedral angle close to 5°. Hence a few badly shaped tetrahedra are inevitably left in the mesh near thin sharp edges and in the vicinity of the thin summit.

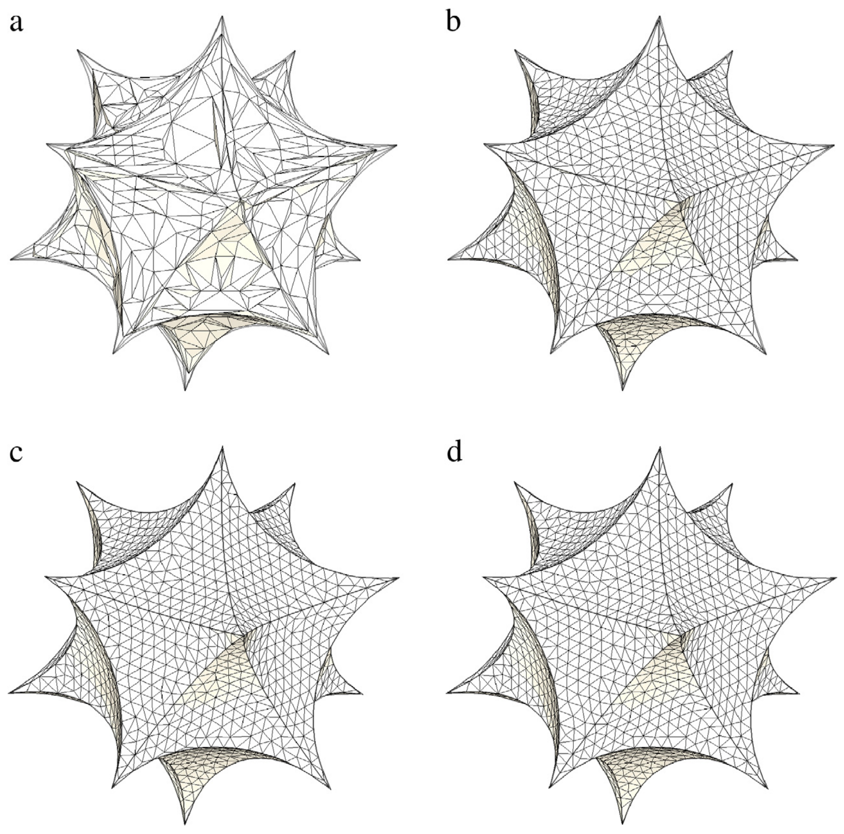

Fig. 11. Delaunay mesh for curved icosahedron: (a) initial guess, (b) intermediate mesh, (c) mesh after domain recovery, (d) mesh after optimization.

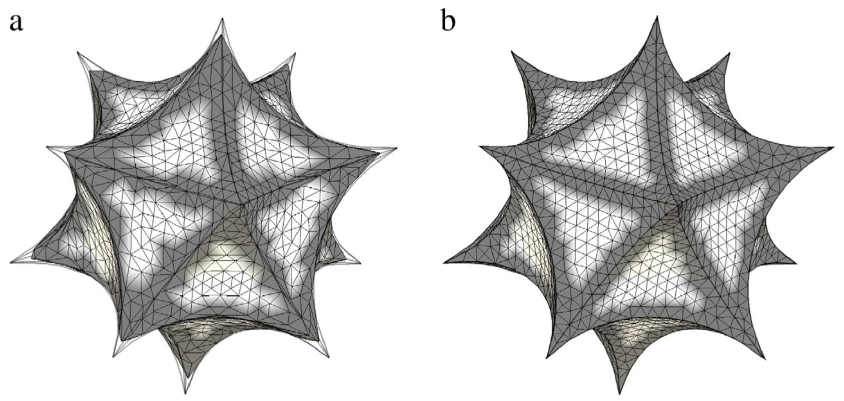

Fig. 12. (a) Vertices marked as "sharp" for intermediate mesh, (b) "sharp" vertices for final mesh.

Mesh quality histograms in Figs. 19 and 20 do not take into account thin tetrahedra adjacent to the peak summit and sharp edges. These tetrahedra are shown separately on Fig. 18.

Fig. 21 illustrates limitations of the suggested algorithm. One can see that numerical summit on the right, Fig. 21(b) is a bit lower compared to the one on the coarse mesh. The boundary fitting procedure allows to place vertices onto sharp edges and cone summits with very high accuracy, but the exact total angle around summit is so small that with

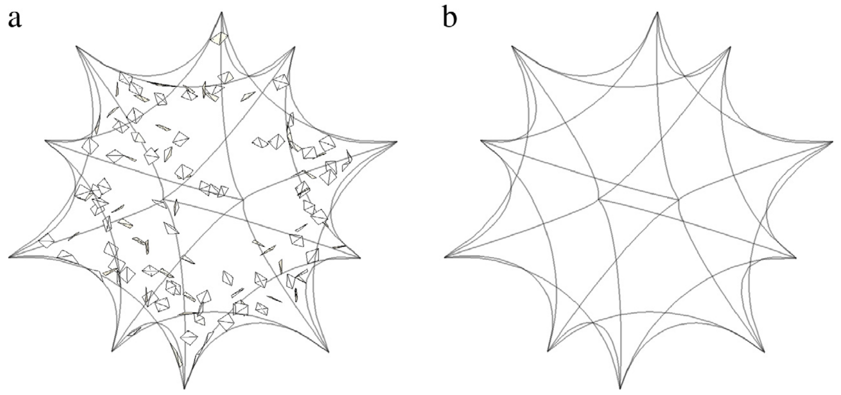

Fig. 13. (a) Tetrahedra with dihedral angle less than 15° after domain recovery, (b) after shape optimization flat tetrahedra disappear.

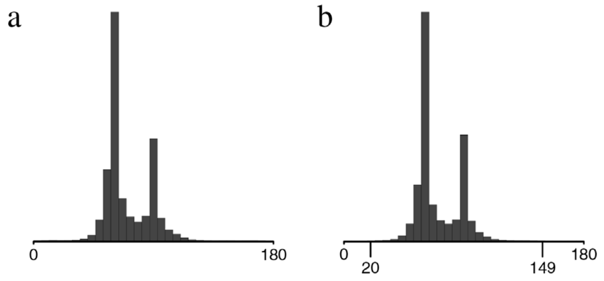

Fig. 14. (a) Dihedral angle distribution before cell shape optimization and (b) after shape optimization.

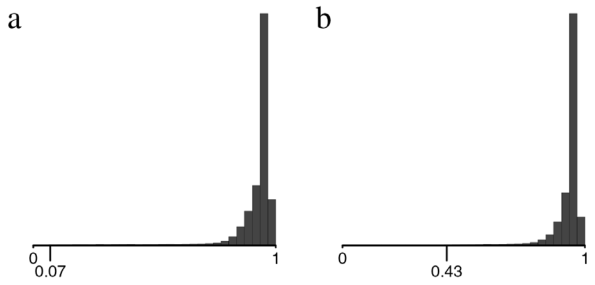

Fig. 15. Histogram of cell shape quality measure (3) before (a) and after (b) shape optimization.

mesh refinement tetrahedral shelling algorithm fails to distinguish between true tetrahedra and the ones that should be shelled away. Otherwise quality of approximation for this test case is quite satisfactory.

Example 4. Next model from [15] is the "spiral". Precise description for this model was not found hence we created our own analytical representation. Implicit function for spiral is created by sweeping planar rectangle along vertical axis. Auxiliary spatial transformation $y(x)$ is defined as

$$y_1 = \left(x_1 - \frac{R}{4}\right)\cos\phi - x_2 \sin\phi + \frac{R}{4}, \qquad y_2 = \left(x_1 - \frac{R}{4}\right)\sin\phi + x_2\cos\phi, \qquad y_3 = x_3$$

where

$$\phi = 5\pi\frac{x_3+1}{2}, \qquad R = \frac{4}{5}x_3^2 + \frac{1}{5} \quad \text{for } -1 \le x_3 \le 1,$$

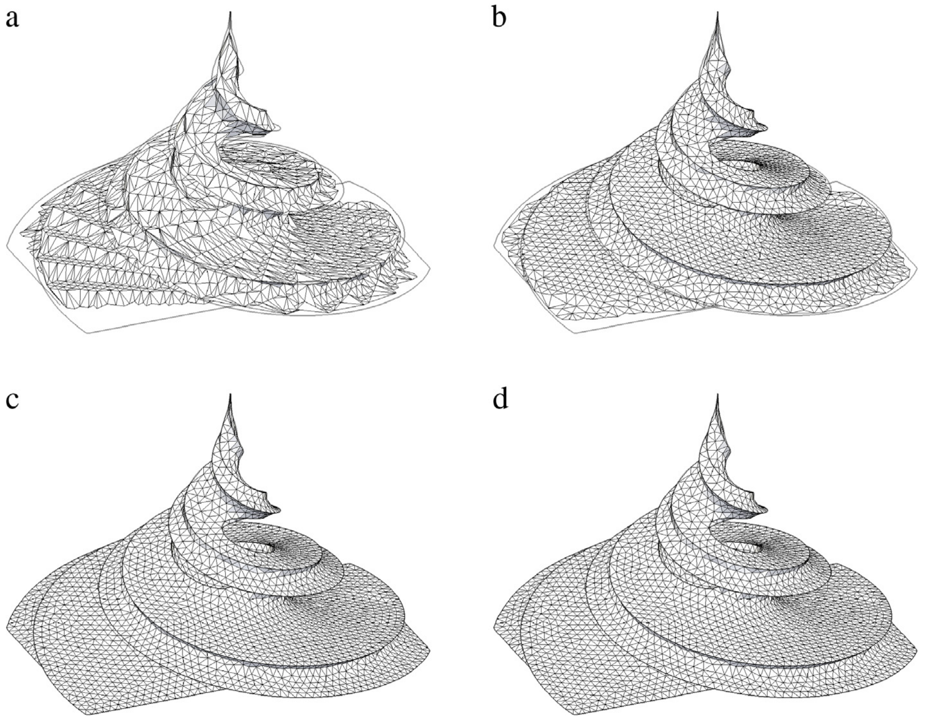

Fig. 16. Delaunay mesh for twisted prism: (a) initial guess, (b) intermediate mesh, (c) mesh after domain recovery, (d) mesh after optimization.

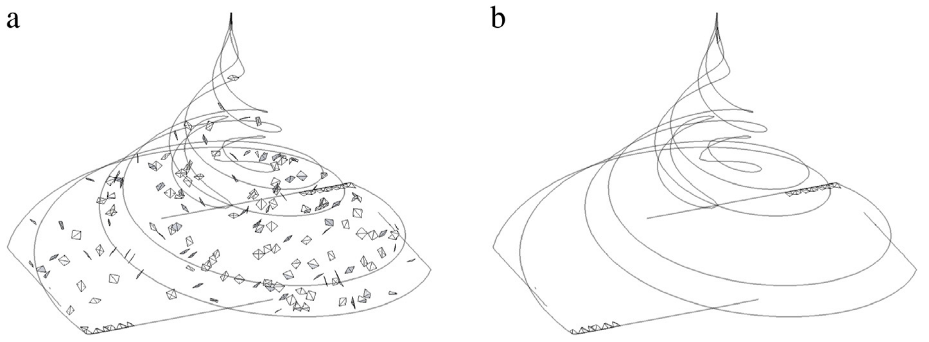

Fig. 17. Tetrahedra with dihedral angle less than 9° before optimization (a) and after optimization (b).

$$\phi = 0, \qquad R = 1 \quad \text{for } x_3 > 1,$$

$$\phi = 5\pi, \qquad R = 1 \quad \text{for } x_3 < -1.$$

Auxiliary implicit functions for sweeping rectangle and for bounding cuboid are defined by the following expressions:

$$u_1(y) = \frac{1}{2}\left(\max\left(5|y_1|, \frac{5|y_2|}{4}\right) - R(y_3)\right)$$

$$u_2(y) = \max\left(\frac{|y_1|}{2}, \frac{|y_2|}{2}, |y_3|\right) - 1.$$

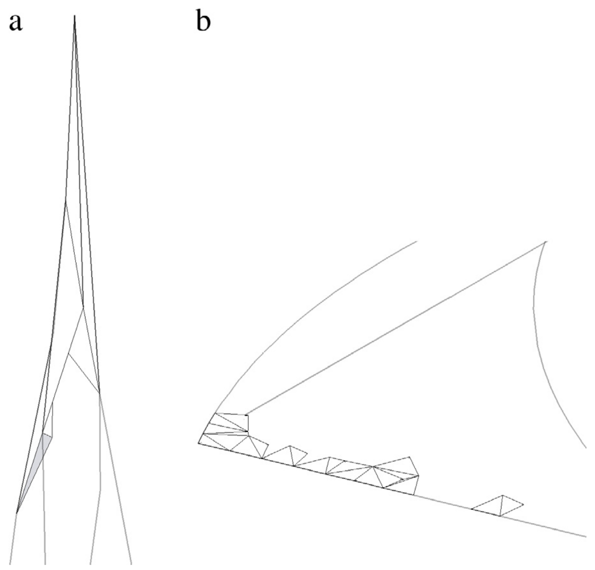

Fig. 18. Tetrahedra which are not accounted for in histogram: (a) near curved peak summit, (b) near sharp edges.

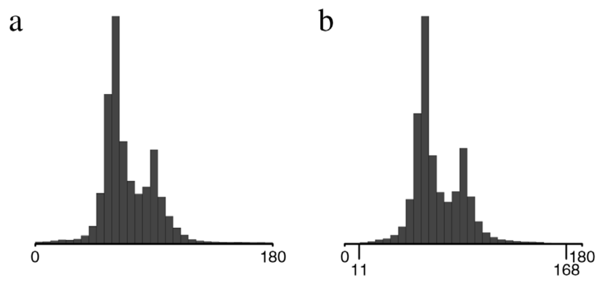

Fig. 19. Dihedral angle distribution before (a) and after (b) optimization.

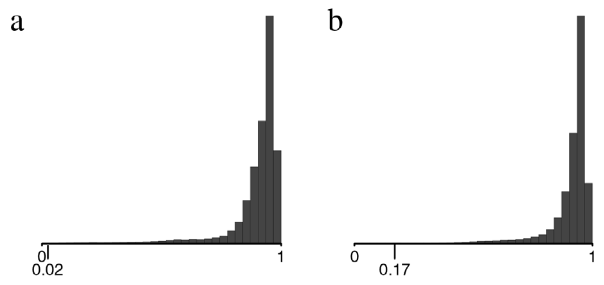

Fig. 20. Cell shape quality measure (3) distribution before (a) and after (b) optimization.

Final representation of implicit function is constructed as

$$u(x) = \min(u_1(y(x)), u_2(y(x))).$$

Results for this model are shown in Fig. 22 (see Figs. 27 and 28).

Fig. 23 illustrates mesh optimization stage. Most of the almost flat tetrahedra are eliminated and sharp edge approximation quality is retained.

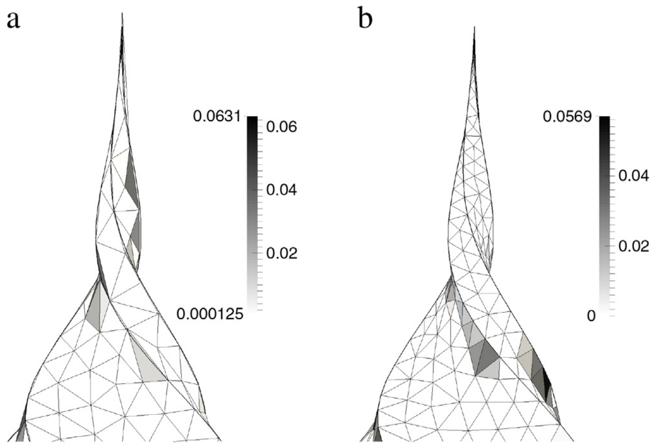

Fig. 21. (a) Boundary shape error for coarse mesh with sharp edges from the refined mesh shown as bold lines, (b) boundary shape error for mesh refined near summit.

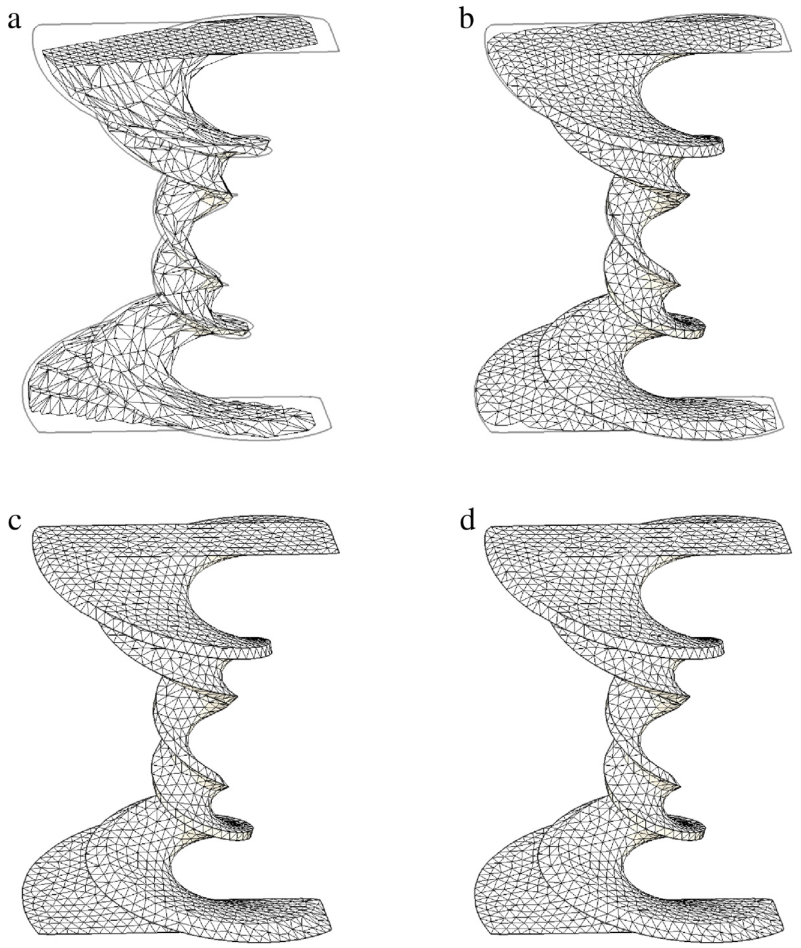

Fig. 22. Delaunay mesh for spiral: (a) initial guess, (b) intermediate mesh, (c) mesh after domain recovery, (d) mesh after optimization.

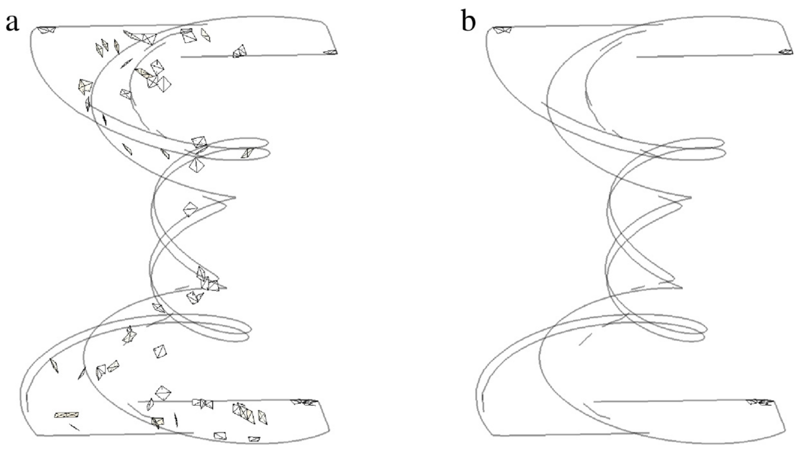

Fig. 23. Tetrahedra with dihedral angle less than 9° before optimization (a) and after optimization (b).

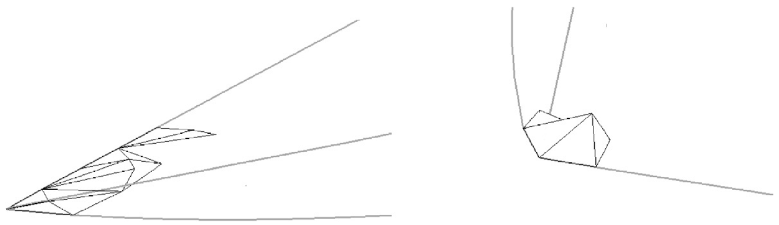

Fig. 24. Tetrahedra adjacent to sharp edges.

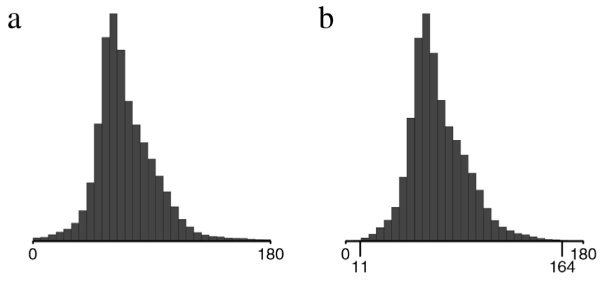

Fig. 25. Dihedral angle distribution before (a) and after (b) optimization.

Quality histograms shown in Figs. 25 and 26 do not take into account thin tetrahedra adjacent to very sharp edges which are shown separately on Fig. 24.

Fig. 29 illustrates effect of using sharp edges attraction potential $W_a(P)$ defined in (2). In all numerical experiments described above we used only sharpening technique based on boundary normal alignment. In most cases this technique is quite efficient in a sense that it allow to place mesh vertices onto exact sharp edges with quite small deviations. However for the models like "spiral" the alignment technique may create considerable deviation of computed sharp edges from exact ones which is explained in Fig. 1 and illustrated in Fig. 29(a). In this test case solution is obtained with the standard set of weights $\theta_d = \theta_a = 0, \theta_r = \theta_s = \frac{1}{2}$ for shape recovery and $\theta_r = \frac{3}{10}, \theta_s = \frac{6}{10}, \theta_a = 0, \theta_d = \frac{1}{10}$ for mesh optimization.

To test the ability for precise recovery of highly curved sharp edges on highly curved surfaces we have used the following set of parameters $\theta_d = 0, \theta_r = \theta_s = \frac{1}{10}, \theta_a = \frac{4}{5}$ for shape recovery, and $\theta_d = \theta_r = \theta_s = \frac{1}{10}, \theta_a = \frac{7}{10}$ for shape optimization. Effect of sharp edge attraction is illustrated in 29(b).

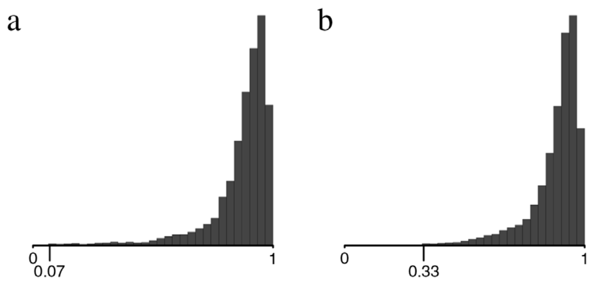

Fig. 26. Tetrahedral shape quality measure (3) distribution before (a) and after (b) optimization.

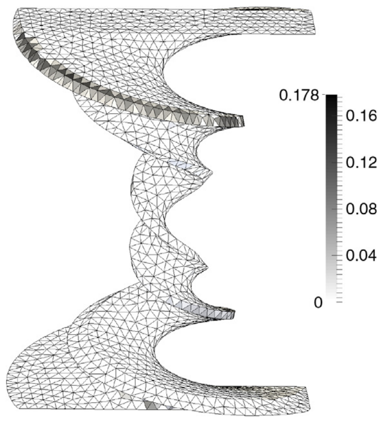

Fig. 27. Local boundary shape error (13) distribution.

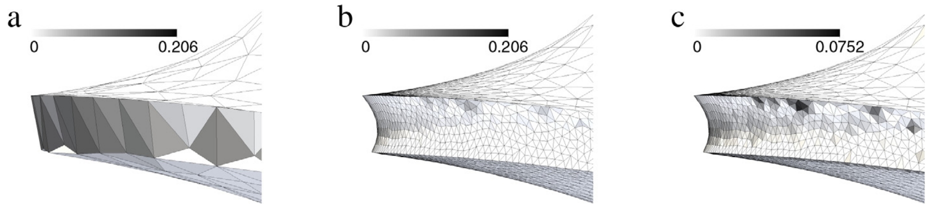

Fig. 28. (a) Boundary shape error (13) on coarse mesh, (b) boundary shape error on finer mesh, (c) rescaled boundary shape error on finer mesh.

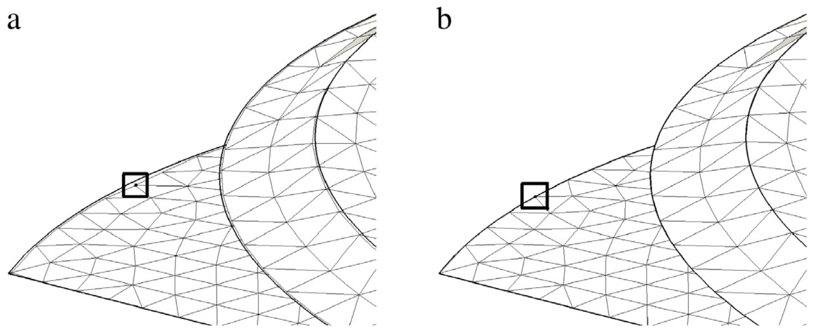

Fig. 29. Sharp edge recovery without sharp edge attraction (a) and with sharp edge attraction (b).

Table 1
Local effect of sharp edge attraction.

	(a)	(b)
L_0	7.40e−2	7.48e−2
$\lvert p - p^* \rvert$	5.36e−3	2.83e−4
$\frac{\lvert p-p^* \rvert}{L_0}$	7.24e−2	3.78e−3
$\lvert p - p_b \rvert$	8.36e−3	8.85e−4
$\frac{\lvert p-p_b \rvert}{L_0}$	1.13e−1	1.18e−2

Visually it is clear that sharp edge recovery error is sharply reduced. We show in Table 1 numerical data for the mesh vertex p marked on Fig. 29 by a square. We denote by p^* the vertex position predicted by the sharp edge attraction method and by p_b the point on the exact sharp edge with smallest distance from p. Column (a) corresponds to standard algorithm, while column (b) presents data when sharp edge attraction is applied.

As one can see from Table 1, shape error (14) is reduced locally by the order of magnitude.

8. Discussions and directions of future research

Analysis of behavior of self-organization algorithm on the set of the test problems led to the following conclusions. Repulsive forces lead to redistribution of mesh points over the domain Ω according to the sizing function $f_h(x)$. The boundary points of the mesh $\mathcal{T}^k$ are distributed near the boundary of Ω also in agreement with $f_h(x)$. Boundary faces tend to approximate regular fragments of $\partial\Omega$ and sharp edges on the boundary are gradually manifested.

It should be noted that mechanism of repulsion admits degeneration of tetrahedra. This property is important since construction of optimal mesh is both continuous and discrete optimization problem. Continuous optimization is related to the search of best positions of mesh vertices. Discrete optimization is related to the problem of constructing optimal mesh connectivity. Note that we do not use special topological optimization methods like those from [6,10]. It fact potential degeneration of tetrahedra and topological changes according to the Delaunay criteria serve as a ways to get out of a local minima of certain mesh quality function. During shape optimization stage topological changes are also admitted and we have found that they are not detrimental to the quality of approximation of sharp boundary edges.

Current implementation of the self-organization algorithm is not efficient since it follows in general structure original algorithm by Persson and Strang [16]: at each step full Delaunay mesh is reconstructed, implicit function and its gradient is computed everywhere, shelling of excessive tetrahedra is applied on each iteration, initial mesh is not sensitive to geometry, etc. Hence resulting algorithm is much slower compared to algorithms from [9]. We tested several open-source codes for Delaunay meshing of the set of points and eventually used TetGen [17]. In order to mimick the behavior of the Matlab code [16] we used only classical Delaunay mesher option from TetGen which creates Delaunay tetrahedra for convex envelope of the input point set. As a result we have found that the Delaunay algorithm itself takes only small fraction of the computational time. For example, for model "spiral" with 3550 vertices and 12 908 tets the average time for iteration of mesh relaxation is about 0.49 seconds on a single core of Intel i7-3630QM 2.40 GHz CPU, while the Delaunay algorithm time is just about 5% of total time. This fraction can be reduced even more since Delaunay remeshing can be done in one of ten iterations. Total time for domain reconstruction is quite large since for test case "'spiral" automatic stopping criteria fails hence about 5000 iterations are made to recover the boundary with sharp edges using lattice mesh as initial guess. Note that using sharp edge attraction reduces this iteration count to about one thousand. Cell shape optimization iteration cost is about 0.72 s, but it took just 5 iterations in order to eradicate slivers.

Thus algorithm that we describe here is not competitive as a meshing algorithm. It should be considered as a simple computational platform for testing variants of meshing algorithms near sharp curved edges.

It is necessary to point out the evident drawbacks of the suggested algorithm. The most important issue of the current version is that constructing Delaunay mesh inside thin dihedral wedge is much simpler compared to the exterior mesh construction. The problem with exterior mesh construction is due to the fact that in the process of mesh expansion around thin wedge the center of the Delaunay ball of fictitious thin exterior tetrahedron can move into computational domain and destroy the integrity of the boundary. One has to apply special measures in order to prevent crossing of domain boundary by such circumcenters. Another drawback is related to reconstruction of ravines on the surface. One can imagine thin sector between sharp feature lines where surface is concave or saddle like. These sharp feature lines can by quite close to each other, say near the summit of the cone. When the local distance between feature lines is comparable to the edge length one can encounter numerical instability of the surface reconstruction process. Our observation is that boundary face alignment technique tends to be quite stable but produces sharp lines which essentially deviate from the correct ones. The use of sharp edge attraction technique sharply reduces approximation error for the edges but it turns out that the geometrical centers of some internal tetrahedra go outside the domain thus destroying boundary integrity. The procedure for correct shelling in this case does not look obvious at all.

It is clear how this algorithm can be accelerated. Dynamic Delaunay remesher should be used which recomputes mesh only around the non-Delaunay faces and thus avoids most of the costly operations described above. Delaunay meshing algorithm or general tetrahedral meshing algorithm [11] for domains with smooth boundary can provide good initial guess for meshing, while initial reconstruction of sharp edges and conical vertices can be obtained using variant of the dual marching cubes algorithm. Interesting generalization of the above algorithm is related to approximation of the boundary by the Voronoi faces while dual Delaunay edges should intersect the boundary almost orthogonally creating structure similar to prismatic layer. This is the topic of current research.

Acknowledgments

Partial funding of this work by the NSH-8860.2016.1 project of the governmental program for supporting leading scientific schools is gratefully acknowledged. Research on relaxation algorithms for alignment of tetrahedral meshes with implicitly defined computational domain boundaries is supported by the Russian science foundation 16-11-10350 project.

References

[1] P. Alliez, D. Cohen-Steiner, M. Yvinec, M. Desbrun, Variational tetrahedral meshing, TOG 24 (3) (2005) 617–625.

[2] A.I. Belokrys-Fedotov, V.A. Garanzha, L. Kudryavtseva, Delaunay meshing of implicit domains with boundary edge sharpening, in: Optimization and Application, Issue IV, Computing Center RAS, 2015, pp. 191–219.

[3] L. Belousova, V.A. Garanzha, Construction of Delaunay meshes in implicit domains with non-smooth boundary, in: Proc. 51th Scientific Conference. Modern Problems of Fundamental and Applied Sciences. Part VII. Control and Applied Mathematics, Vol. 2, Melentiev Energy Systems Institute Siberian Branch of the Russian Academy of Sciences, ESI SB RAS, 2008, pp. 98–101, ISBN: 978-5-93908-052-1.

[4] Q. Du, D. Wang, Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations, Internat. J. Numer. Methods Engrg. 56 (9) (2003) 1355–1373.

[5] M.S. Floater, One-to-one piecewise linear mappings over triangulations, Math. Comp. 72 (242) (2003) 685–696.

[6] L.A. Freitag, C.F. Ollivier-Gooch, A comparison of tetrahedral mesh improvement techniques, in: 6th International Meshing Roundtable, Office of Scientific and Technical Information, OSTI, 1996, pp. 87–100. http://doi.org/10.2172/414383.

[7] P.J. Frey, P.-L. George, Mesh Generation: Application to Finite Elements, Hermes Science, Paris, 2000, ISBN: 1903398002.

[8] V.A. Garanzha, L.N. Kudryavtseva, Generation of three-dimensional Delaunay meshes from weakly structured and inconsistent data, Comput. Math. Math. Phys. 52 (3) (2012) 427–447.

[9] C. Jamin, P. Alliez, M. Yvinec, J.-D. Boissonnat, CGALmesh: A generic framework for Delaunay mesh generation, ACM Trans. Math. Softw. 41 (4) (2015) 1–24.

[10] B. Joe, Construction of three-dimensional improved-quality triangulations using local transformations, SIAM J. Sci. Comput. 16 (6) (1995) 1292–1307.

[11] F. Labelle, J.R. Shewchuk, Isosurface stuffing: Fast tetrahedral meshes with good dihedral angles, ACM Trans. Graph. TOG 26 (3) (2007) 57:1–57:10 Special issue on Proceedings of SIGGRAPH 2007.

[12] V.D. Liseikin, Grid Generation Methods, Springer, New York, 2010.

[13] S. Lloyd, Least squares quantization in PCM, IEEE Trans. Inform. Theory 28 (2) (1982) 129–137.

[14] M. Mandad, D. Cohen-Steiner, P. Alliez, Isotopic approximation within a tolerance volume, ACM Trans. Graph. TOG 34 (4) (2015) 64:1–64:12.

[15] Y. Ohtake, A. Belyaev, Dual/primal mesh optimization for polygonized implicit surfaces, in: N.M.P. Kunwoo Lee (Ed.), Proceedings of the 7th ACM Symposium on Solid Modeling and Applications, (SMA0́2), ACM, New York, 2002, pp. 171–178.

[16] P.-O. Persson, G. Strang, A simple mesh generator in MATLAB, SIAM Rev. 46 (2) (2004) 329–345.

[17] H. Si, TetGen, a Delaunay-based quality tetrahedral mesh generator, ACM Trans. Math. Softw. (TOMS) 41 (2) (2015) 11:1–11:36 Special issue on Proceedings of SIGGRAPH 2007.

[18] J. Tournois, C. Wormser, P. Alliez, M. Desbrun, Interleaving Delaunay refinement and optimization for practical isotropic tetrahedron mesh generation, TOG 28 (3) (2009) 71.1–75.9.