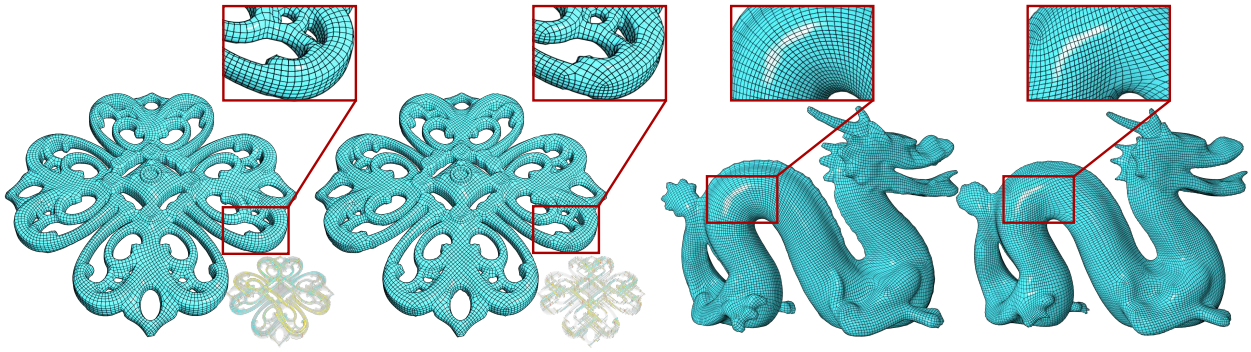
All-Hex Meshing using Closed-Form Induced Polycube

Xianzhong Fang Weiwei Xu Hujun Bao Jin Huang*
State Key Lab of CAD&CG, Zhejiang University



(a) 647, 0.24/0.895 (b) 793, 0.24/0.874 (c) 310, 0.162/0.919 (d) 315, 0.150/0.857

Figure 1: *Our method generates smooth frame fields without internal singularities to guide the polycube construction, and results in high-quality hexahedral meshes. Compared with results from the ℓ_1-based polycube method ((b) and (d)), our results ((a) and (c)) have fewer unnecessary stairs, corners, and better element shapes at the same time. The number of corners and minimal/mean scaled Jacobians are listed below each result.*

Abstract

The polycube-based hexahedralization methods are robust to generate all-hex meshes without internal singularities. They avoid the difficulty to control the global singularity structure for a valid hexahedralization in frame-field based methods. To thoroughly utilize this advantage, we propose to use a frame field without internal singularities to guide the polycube construction. Theoretically, our method extends the vector fields associated with the polycube from exact forms to closed forms, which are curl free everywhere but may be not globally integrable. The closed forms give additional degrees of freedom to deal with the topological structure of high-genus models, and also provide better initial axis alignment for subsequent polycube generation. We demonstrate the advantages of our method on various models, ranging from genus-zero models to high-genus ones, and from single-boundary models to multiple-boundary ones.

Keywords: Polycube map, frame field, volumetric parametrization, automatic hexahedral meshing

Concepts: •**Theory of computation** → **Computational geometry;** •**Computing methodologies** → **Mesh geometry models; Parametric curve and surface models; Volumetric models;**

*Corresponding author: hj@cad.zju.edu.cn

SIGGRAPH '16 Technical Paper, July 24-28, 2016, Anaheim, CA
ISBN: 978-1-4503-4279-7/16/07
DOI: http://dx.doi.org/10.1145/2897824.2925957

1 Introduction

High-quality, boundary-conforming all-hex mesh generation is one of the most challenging meshing problems due to its computational complexity. Recently, much research efforts have been devoted to extending frame field guided quadrangulation algorithms to volumetric hexahedral mesh generation [Nieser et al. 2011], resulting in significant progresses in smooth 3D frame field optimization algorithms [Huang et al. 2011; Li et al. 2012]. However, such optimization algorithms are sensitive to the quality of initial frame fields and volumetric tetrahedral meshes. They cannot guarantee to get a valid singularity structure for a non-degenerate volumetric parametrization. Although existing algorithms to correct local singularity structures of the designed 3D frame fields [Li et al. 2012; Jiang et al. 2014], it is still difficult to design a 3D frame field that possesses a global singularity structure suitable to all-hex mesh generation.

Another line of hexahedralization methods is based on polycube. These methods focus on a special type of hexahedral mesh that has simpler singularity structure, i.e. no internal singularity [Gregson et al. 2011; Livesu et al. 2013; Huang et al. 2014]. They deform the boundary surface of a volumetric model into an axis-aligned polycube, and tessellate the polycube to extract an all-hex mesh. Without the complexity to control the internal singularity structure, such methods are robust for all-hex mesh generation. However, the state-of-the-art polycube methods parametrize an input model with a single volumetric chart, resulting in high-distortion hexahedra. Clearly, the polycube parametrization is only a subset of all valid parametrization for all-hex meshing. Indeed, the polycube methods are equivalent to the parametrization induced by a boundary aligned and internal singularity free 3D frame field, which can be further decomposed into three conservative vector fields, i.e. *exact forms*.

In this paper, we introduce a novel subset of hexahedral meshes. They have no internal singularity as well, but may be not directly parametrized as a single chart. Our method generates such hexahedral meshes by relaxing the restriction of previous polycube methods using the superset of exact forms, i.e. *closed forms*. We cut a 3D model $\mathcal{M}$ bounded by $\partial\mathcal{M}$ to break all its tunnel loops, and

then turn the closed forms on $\mathcal{M}$ into the exact forms on the cut mesh $\mathcal{M}_c$. Each cut imposes a continuous transition condition for hexahedral meshing. The number of cuts is the sum of genus of all boundary surfaces, which is exactly the first Betti number of $\mathcal{M}$ [Dey and Guha 1998]. Thus, our method is able to obtain all the closed forms, or all the topological structures of internal singularity free hexahedral meshes. As a consequence, our method inherits the robustness of polycube methods, and it is able to find high-quality results in a much larger space. Considering a torus as an example, the polycube methods generate no less than 16 corners on the boundary, and produce highly distorted elements around the corners. Such serious issues can be easily addressed in our method, and a more complex result is shown in Figure 1.

Besides the highly restricted topological structure, there is another issue in the existing polycube methods. The final parametrization is almost prescribed by the initial axis alignment of surface normals. Although the ℓ_1-based method [Huang et al. 2014] is able to automatically rotate the model into an optimal orientation, it cannot generate high-quality results for the case shown in the left of the inset. Our closed-form based method initializes the alignment according to a smooth frame field instead of a globally uniform one, and produces a high-quality result, as shown in the right of the inset. Therefore, except for the obvious advantage over previous hexahedral meshing methods on handling high-genus models, our method also can generate hexahedral meshes with much lower distortion on many genus-zero models.

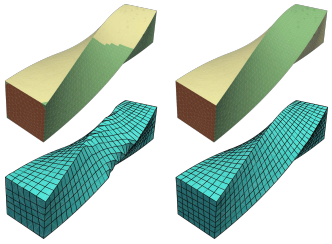


Figure 2: *Our method (right) provides better axis alignment.*

The main contributions of this paper include:

- Extending the polycube method to handle all internal singularity free structures.
- An effective way to construct a boundary aligned smooth frame field without internal singularities, i.e. closed form for hexahedralization.
- Providing better initial values for normal alignment in polycube optimization.

1.1 Related work

All-hex meshing employs hexahedra as basic geometric elements to approximate a volumetric object. Thorough surveys on hexahedral meshing are available in [Shimada 2006; Shepherd and Johnson 2008]. Semi-automatic all-hex meshing methods, such as multiple sweeping [Shepherd et al. 2000] and paving and plastering [Staten et al. 2005], have been well studied and used in a prestigious hexahedral meshing software CUBIT. In addition, Sheffer et al. [1999] investigated how to decompose the volume using Voronoi graph for hexahedral meshing for CAD objects, and Carbonera et al. [2006] presented Geode-Template to constrain hexahedral meshing.

Recent trends in automatic all-hex meshing rely on proper volumetric parametrization to control the singularities and distortion of resulting hexahedral meshes. Inspired by the success of frame-field guided surface parametrization in automatic surface quadrangulation [Bommes et al. 2009], frame-field guided volumetric parametrization for all-hex meshing has been studied in [Nieser et al. 2011; Huang et al. 2011; Li et al. 2012]. The theoretical treatment on how the frame transitions determine the singularity type of an edge in hexahedral meshes is thoroughly studied by Nieser et al. [2011], which gives a local definition for internal improper singularity edges. To avoid the degeneracy in hexahedralization, Li et al. [2012] proposed to locally fix the internal improper singularities via matching adjustment and edge collapse. Jiang et al. [2014] investigated the local degeneracy problem on both internal and surface singularities, and locally fixed all the improper singularities with proof via subdivision. A recent approach [Kowalski et al. 2014] generated block structures for hexahedral meshing guided by a 3D frame field defined on vertices. However, if there are non-local conflicts in the whole singularity structure as mentioned in [Li et al. 2012; Jiang et al. 2014], all such methods fail to generate a hexahedral mesh without degeneracy.

Polycube parametrization maps the volumetric mesh into joined and axis-aligned cubes. It was first proposed by Tarini et al. [2004] for texture parametrization and quad-meshing, and its automatic generation is explored in [Lin et al. 2008]. Gregson et al. [2011] developed a volumetric deformation to deform a complex shape into a polycube shape. A sketch-based polycube mapping method was developed in [García et al. 2013]. The frequently occurred small stairs in polycube parametrization, which will lead to unnecessary singularities on the boundary surface of a resulting hexahedral mesh, can be efficiently reduced through Polycut [Livesu et al. 2013] and ℓ_1-based polycube parametrization [Huang et al. 2014]. The topology of singularity graph for an all-hex mesh can also be improved through base complex optimization [Gao et al. 2015] and edge-cone rectification [Livesu et al. 2015]. By virtue of internal singularity free, such methods enjoy great robustness. However, because of the limited topological structure and strong dependence on initial normal alignment, such methods cannot produce high-quality results for complex models.

Our work can be viewed as a combination of frame-field guided hexahedralization and polycube parametrization. We guide the polycube parametrization through an optimized smooth frame field instead of a globally uniform one. It reduces the number of singularities and the distortion of the resulting all-hex mesh. The internal singularity free constraint is still maintained in our method, which provides great help on getting across the challenge of resolving non-local singularity conflicts.

1.2 Overview

The input to our algorithm is a tetrahedral mesh $\mathcal{M} = \{\mathcal{X}, \mathcal{F}, \mathcal{T}\}$, where $\mathcal{X}$ is a set of vertices $\{x_1, ..., x_n\}$, $\mathcal{F}$ the set of triangles $\{f_1, ..., f_m\}$, and $\mathcal{T}$ the set of tetrahedra $\{t_1, ..., t_q\}$. The output is the vertex coordinates after closed-form induced polycube parametrization, denoted by $\overline{\mathcal{X}}$. As illustrated in Figure 3, the algorithm will cut the mesh along some faces $\mathcal{C} \subset \mathcal{F}$ into $\mathcal{M}_c$. Then starting from an initial frame field computed by [Huang et al. 2011], we optimize an internal singularity free and boundary aligned smooth cross-frame field $\mathbf{R}$ by allowing frame transitions at the cuts. Guided by the resulting frame field and transitions on cuts, the cut mesh $\mathcal{M}_c$ is deformed into a shape close to a polycube, which implies the desired closed forms for polycube parametrization. Using the deformed mesh as the initial value, we apply the extended ℓ_1-based polycube method which takes the transition functions into account to generate the final result and extract the hexahedral mesh. In the following, we will elaborate each step in detail.

2 Cut Generation

According to the proof in [Dey and Guha 1998], the first Betti number β_1 of $\mathcal{M}$ is equal to the sum of genus on all its boundary meshes $\partial\mathcal{M}$. Thus, for each handle loop (i.e. the latitudinal generator) on $\partial\mathcal{M}$, we cut $\mathcal{M}$ by a connected, open 2d-manifold inside $\mathcal{M}$, where the handle is required to be on its boundary, and then introduce *only one* possibly non-trivial transition function on the cut.

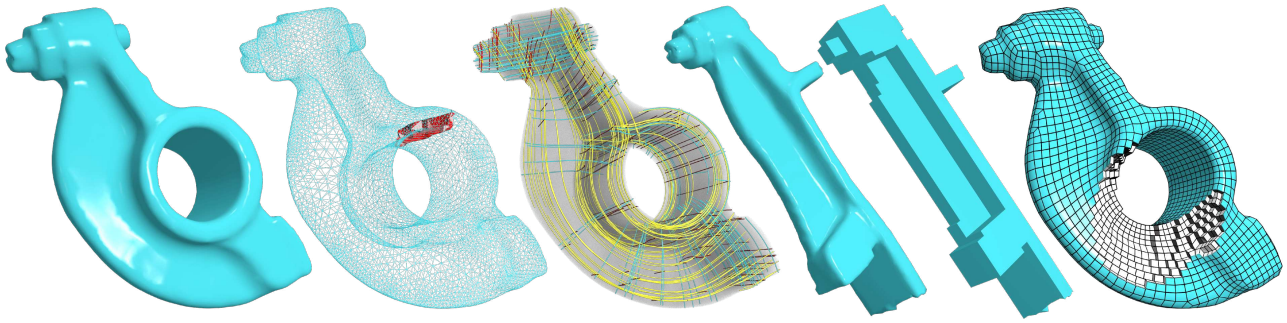


Figure 3: *Pipeline of our algorithm. From the left to the right are input mesh, cut faces, frame field, deformed cut mesh, polycube parametrization and final hexahedral mesh.*

With the additional transition functions, our method can represent all the closed 1-forms. To make sure that one cut is associated with a single transition function, we *recursively* build the cuts, and each cut reduces the first Betti number of $\mathcal{M}$ just by one. Specifically, starting from the input mesh $\mathcal{M}_{c,0} = \mathcal{M}$, at step $k = 1, 2, \cdots, \beta_1$, we find a handle $\mathcal{H}_{k-1}$ on $\mathcal{M}_{c,k-1}$, and then construct a cut c_k in $\mathcal{M}_{c,k-1}$ which does not contain any triangle and any isolated vertex of $\partial\mathcal{M}_{c,k-1}$ according to the handle, and finally cut $\mathcal{M}_{c,k-1}$ into $\mathcal{M}_{c,k}$. $\mathcal{M}_{c,\beta_1}$ is the final cut mesh $\mathcal{M}_c$.

Cuts containing a small number of triangles are preferred for the efficiency in the subsequent computation. Therefore, we use the method introduced in [Dey et al. 2013] to compute surface handle loops. To get the cut for a handle loop $\mathcal{H}_k$, we first construct a small connected submanifold $\mathcal{B} \subset \mathcal{M}_{c,k}$ with the following properties: (a) $\partial\mathcal{B}$ contains $\mathcal{H}_k$; (b) $\partial\mathcal{B} - \partial M_{c,k}$ are two disconnected patches. More specifically, we initialize $\mathcal{B}$ as all the tetrahedra immediately adjacent to vertices in $\mathcal{H}_k$ for (a), and then iteratively add an adjacent tetrahedron to it until $\partial\mathcal{B} - \partial M_{c,k}$ is split into two components for (b), which can be checked by the floodfill algorithm with a seed in $\partial\mathcal{B} - \partial M_{c,k}$. In each iteration, if $\partial\mathcal{B}$ becomes a non-manifold, all the tetrahedra adjacent to the non-manifold vertices or edges are added to correct $\mathcal{B}$ into a manifold one. Then, under the constraints of (a) and (b), we remove tetrahedra in $\mathcal{B}$ containing triangles or isolated vertices of $\partial\mathcal{M}_{c,k}$. In this process tetrahedral subdivision may be required. Then we reduce the number of tetrahedra in $\mathcal{B}$ by removing boundary tetrahedra of $\mathcal{B}$ iteratively. If one of the constraints is violated when reducing, the next candidate tetrahedron is chosen. The process stops with no removable tetrahedron. At last, the patch with fewer triangles in $\partial\mathcal{B} - \partial\mathcal{M}_{c,k}$ is chosen as the cut.

The order of adding and removing tetrahedra is heuristically determined to form a small cut. Giving a fitting plane to $\mathcal{H}_k$, the tetrahedron with a smaller distance between its barycenter to the plane has a high priority for adding and a low priority for removing. Because the handle generated from the method [Dey et al. 2013] is usually small and planar, such a heuristic method works well in our experiments.

3 Frame Field Generation

The goal of this step is to generate a frame field constant in each tetrahedron to guide the volumetric parametrization. In our method, there are three desirable properties of the generated frame field:

- It should be as smooth as possible to obtain a low distortion parametrization. This property is the main optimization objective in previous frame field optimization algorithms [Huang et al. 2011; Li et al. 2012].
- The frame at each boundary tetrahedron should transform the normal of its triangles on boundary surface into one coordinate axis of a selected canonical coordinate system. This is corresponding to boundary alignment conditions.
- It should have no internal singularity, but may have non-trivial transitions on the cuts.

In this section, we will first introduce the rotation-matrix based algorithm, and then detail how to extract the transitions from the optimized frame field.

3.1 Rotation-matrix based frame field generation

We require the frame at each tetrahedron to be orthogonal. Thus it can be conveniently encoded into a rotation matrix $\mathbf{R}$. In our setting, each row of $\mathbf{R}$ corresponds to an axis vector in the frame defined in a tetrahedron. Moreover, such rotation-matrix based representation is also convenient for us to formulate constraints on the transformation of boundary triangle normal n.

Formulation. The generation of the smooth frame field can be posed as an optimization problem with the following objective function:

$$\begin{aligned}\min_{\mathbf{R}} \quad & w_f \int_{\mathcal{F}\setminus\mathcal{C}} \|\nabla\mathbf{R}\|^2 + w_c \int_{\mathcal{C}} \widetilde{\nabla}\mathbf{R} \\ & + w_a \int_{\partial\mathcal{M}} \|\mathbf{R}n\|_1 + w_d \int_{\partial\mathcal{M}\setminus\partial\mathcal{C}} \|\nabla(\mathbf{R}n)\|^2 \\ & + w_R \int_{\mathcal{M}} \|\mathbf{R}^T\mathbf{R} - I\|^2.\end{aligned} \tag{1}$$

The first two terms are used to measure the smoothness of a frame field at tetrahedra since the frame field is piece-wise constant. The ∇ operator indicates that we need to measure the rotation matrix difference there. However, at the cutting boundaries, we need to take the frame transitions into consideration. The operator ∇ thus is changed to $\widetilde{\nabla}$, and its formula will be made clear soon. The third term encodes the constraint that the transformed normal should be axis-aligned. The fourth term penalizes the normal difference under local frame $\mathbf{R}$, which helps to reduce the switching of axis alignment. The last one is used for the orthogonal constraint on $\mathbf{R}$.

Smoothness Terms. The first term, used to measure the smoothness of a frame field, can be discretized into the following formula:

$$\sum_{t_i \cap t_j \in \mathcal{F}\setminus\mathcal{C}} \frac{V_{t_i} + V_{t_j}}{V_F d_{t_i,t_j}^2} \|\mathbf{R}_{t_i} - \mathbf{R}_{t_j}\|^2. \tag{2}$$

Therefore, the smoothness is measured via the Frobenius norm at every two tetrahedra sharing a common triangle. In Equation (2), the volume of a tetrahedron t_i is denoted by V_{t_i}, and d_{t_i,t_j} is the distance between the barycenters of t_i and t_j. The weights of

the difference between the two rotation matrices are normalized by $V_F = \sum_{t_i \cap t_j \in \mathcal{F} \setminus \mathcal{C}} (V_{t_i} + V_{t_j})/(d^2_{t_i,t_j})$.

The second term, used to measure the smoothness of a frame field at the cutting boundaries, is formulated as follows:

$$\sum_{t_i \cap t_j \in \mathcal{C}} \frac{V_{t_i} + V_{t_j}}{V_C d^2_{t_i,t_j}} \sum_{k=1}^{3} h\left((\mathbf{R}_{t_i}^T \mathbf{R}_{t_j})_k\right). \tag{3}$$

The $\widetilde{\nabla}$ is represented by the function $h(r) = r_x^2 r_y^2 + r_y^2 r_z^2 + r_z^2 r_x^2$ introduced in [Huang et al. 2011; Li et al. 2012], which takes k-th column of $\mathbf{R}_{t_i}^T \mathbf{R}_{t_j}$ as the input here. When $\mathbf{R}_{t_i}^T \mathbf{R}_{t_j}$ is one of elements in the cubical symmetry group [Nieser et al. 2011], the value $\widetilde{\nabla}$ is exactly zero. Here, $V_C = \sum_{t_i \cap t_j \in \mathcal{C}} (V_{t_i} + V_{t_j})/(d^2_{t_i,t_j})$.

Normal alignment term. The term is summed at each boundary triangle f that belongs to tetrahedron t:

$$\sum_{f \in \partial \mathcal{M}} \frac{A_f}{A_{\partial \mathcal{M}}} \|\mathbf{R}_t n_f\|_1, \tag{4}$$

where A_f indicates the area of a boundary triangle f, and $A_{\partial \mathcal{M}}$ the total area of the boundary surface. Similar notations will be used below. Similar to [Huang et al. 2014], the ℓ_1 norm of a vector u is approximated by $\|u\|_1 \approx \sum_i \sqrt{u_i^2 + \varepsilon}$, where ε is a very small positive real number.

Normal difference. This term is similar to the shape complexity control in [Huang et al. 2014], and is discretized as:

$$\sum_{f_i \cap f_j = e \in \partial \mathcal{M} \setminus \partial \mathcal{C}} \frac{A_{f_i} + A_{f_j}}{3A_{\partial \mathcal{M}}} \|\mathbf{R}_{t_i} n_{f_i} - \mathbf{R}_{t_j} n_{f_j}\|^2, \tag{5}$$

where $f_i = t_i \cap \partial \mathcal{M}$, $f_j = t_j \cap \partial \mathcal{M}$.

Orthogonality term. We use penalty terms to constrain the matrix $\mathbf{R}$ in each tetrahedron into an orthogonal one:

$$\sum_{t \in \mathcal{M}} \frac{V_t}{V_{\mathcal{M}}} \|\mathbf{R}_t^T \mathbf{R}_t - I\|^2.$$

In practice, we let the energy term weight $w_f = \frac{V_F}{V_F + V_C}$, and $w_c = \frac{V_C}{V_F + V_C}$. The choice of these two weights balances the contribution of the two terms in the objective function well, and we do not meet scaling problems in all the experiments. The other parameters are set as $w_a = 0.1, w_d = 0.01, w_R = 1, \varepsilon = 10^{-3}$. The L-BFGS method implemented in ALGLIB [Bochkanov n.d.] is used to solve the problem. With the stopping criterion that the magnitude of gradient is less than 10^{-5}, it usually converges in less than 300 iterations. After that, we use polar decomposition to extract the rotation part of the result as the final frame field $\mathbf{R}$.

3.2 Initial frame field generation

Since the optimization problem defined in Equation (1) is nonlinear, we need a properly initialized frame field at the beginning of the iteration to reach a local optimal solution.

In this paper, we use the spherical harmonics based method [Huang et al. 2011] to compute the initial frame field in $\mathcal{M}$. However, other techniques, e.g. [Li et al. 2012], can also be used. With the given frame field, we compute a globally aligned frame field by only allowing transitions on the cuts from it. Specifically, we build a minimum spanning tree (MST) in the dual graph of the cut mesh $\mathcal{M}_c$ with minimum alignment error as the edge weight. Then starting from an arbitrary tetrahedron and fixing its frame, we progressively align the neighboring frame field along the MST. Figure 4 shows the initial frame field and the one optimized according to Equation (1).

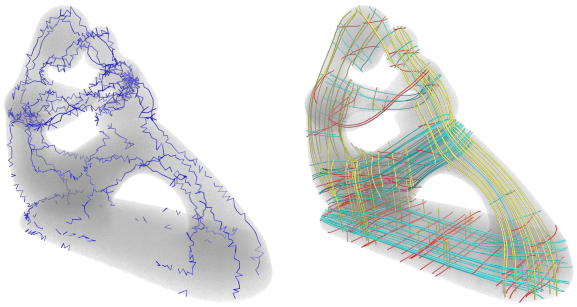


Figure 4: *Given an initial frame field, the internal singularities (blue lines in the left) are removed after optimization (right).*

It should be noticed that the optimization in Equation (1) cannot guarantee that there is no internal singularity in $\mathbf{R}$ when the transitions between adjacent tetrahedra are determined by the criteria of best alignment. In our experiments, a small number of internal singularities may exist, and in most cases, they are near the boundary of the model. However, it does not prohibit us from explicitly assigning the transitions to view it as an internal singularity free frame field, or differential 1-forms without internal singularities, which will be turned into closed ones in the step of Poisson reconstruction.

3.3 Extraction of transitions on cuts

After the optimization of the problem defined in Equation (1), we are now ready to determine the frame transitions at the cuts.

For each cut $c_k \subset \mathcal{C}$, we extract a single transition Π_k in the chiral cubical symmetry group [Nieser et al. 2011] which best aligns the frames on its both sides:

$$\underset{\Pi_k}{\operatorname{argmin}} \sum_{t_i \cap t_j \in c_k} A_{t_i \cap t_j} \|\Pi_k \mathbf{R}_{t_i} - \mathbf{R}_{t_j}\|^2. \tag{6}$$

The optimization can be easily solved by testing 24 choices of Π_k and choosing the one with the minimal error.

4 Frame-field Guided Polycube Generation

The goal of frame-field guided polycube generation is to overcome the limitation of axis alignment of boundary normals in previous polycube methods [Gregson et al. 2011; Livesu et al. 2013; Huang et al. 2014]. It consists of three steps: First, we perform a Poisson reconstruction step to compute an initial parametrization so that its deformation gradient field aligns with the smooth frame field got from Section 3.1; Second, the initial parametrization is further optimized to achieve an axis-aligned polycube using an ℓ_1-based formulation similar to [Huang et al. 2014]; Finally, the integer constraints are enforced upon the axis-aligned polycube result for the final all-hex mesh extraction. The transition constraints are active in all the steps so that the extracted all-hex mesh can be seamlessly stitched at cutting boundaries.

4.1 Poisson reconstruction

Given a frame field $\mathbf{R}$ without internal singularities but with transition conditions at cut faces, an initial parametrization without satisfying axis alignment of boundary normals can be obtained through the following minimization:

$$\min_{\overline{\mathcal{X}}} \quad \int_{\mathcal{M}} \|\nabla \overline{\mathcal{X}} - \mathbf{R}\|^2 dV. \tag{7}$$

For mesh $\mathcal{M}$ with non-zero genus, transition conditions extracted from the optimized frame field are considered as:

$$\Pi_{a,b}\overline{\mathcal{X}}_a(e) = \overline{\mathcal{X}}_b(e), \quad \forall e \in t_a \cap t_b \in \mathcal{C}, \tag{8}$$

where $\Pi_{a,b}$ is the extracted transition about the cut face $t_a \cap t_b$. t_a and t_b are two adjacent tetrahedra, and their edge vectors after Poisson reconstruction are denoted by $\overline{\mathcal{X}}_a(e)$ and $\overline{\mathcal{X}}_b(e)$.

In Poisson reconstruction, the energy function in Equation (7) is a linear least-square problem, and the transition condition constraints are linear. Therefore, the minimization problem can be easily solved by the package Ipopt [Wächter and Biegler 2005]. Figure 5 illustrates the results for genus-zero models, of the Chinese-lion and the Asian-dragon. A genus-one example can be found in Figure 3. It is worthwhile to note that the boundary normal alignment conditions have been approximated through the smooth frame field alignment. More details are given in Section 5 and Figure 2. After this step, the deformed shape implies the desired closed forms for the subsequent polycube generation.

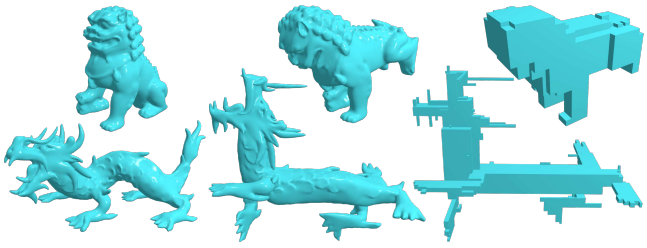


Figure 5: *Via Poisson reconstruction, the step of polycube generation gets good initial values through the closed forms implied by the deformed shapes (middle). The righmost are the corresponding polycube results.*

4.2 ℓ_1-based polycube generation

We follow the ℓ_1-based formulation of boundary surface normal alignment constraints in [Huang et al. 2014] for polycube generation, while enforcing transition conditions at cut faces to facilitate the extraction of a seamless all-hex mesh. The objective energy is defined similar to [Huang et al. 2014] as follows:

$$\begin{aligned} \min_{\overline{\mathcal{X}}} \quad & E_{arap} + w_{align}E_{align} + w_{diff}E_{diff} \\ \text{s.t.} \quad & \overline{A}(\overline{\mathcal{X}}(\mathcal{M})) = A_{\partial\mathcal{M}} \\ & \Pi_{a,b}\overline{\mathcal{X}}_a(e) = \overline{\mathcal{X}}_b(e), \quad \forall e \in t_a \cap t_b \in \mathcal{C}. \end{aligned} \tag{9}$$

The term E_{align} is the normal alignment term, and it is defined using the ℓ_1 norm:

$$E_{align} = \sum_{f\in\partial\mathcal{M}} \frac{1}{2A_{\partial\mathcal{M}}} \left\| (\overline{x}_j^f - \overline{x}_i^f) \times (\overline{x}_k^f - \overline{x}_i^f) \right\|_1, \tag{10}$$

where $(\overline{x}_i, \overline{x}_j, \overline{x}_k)$ are three vertices of a triangle in the boundary.

We also use as-rigid-as-possible energy term E_{arap} to control the non-rigid distortion of tetrahedra:

$$E_{arap} = \sum_{t\in\mathcal{M}} \frac{V_t}{V_{\mathcal{M}}} \left\| \nabla\overline{\mathcal{X}}_t - polar(\nabla\overline{\mathcal{X}}_t) \right\|^2, \tag{11}$$

where $polar(\nabla\overline{\mathcal{X}}_t)$ is the rotation part of deformation gradient. It is computed at the beginning of each iteration using the current parametric vertices $\overline{\mathcal{X}}$. The normal difference term E_{diff} is used to control the details of the generated polycube:

$$\begin{aligned} E_{diff} = & \sum_{f_i\cap f_j = e\in\partial\mathcal{M}\setminus\partial\mathcal{C}} \frac{A_{f_i} + A_{f_j}}{3A_{\partial\mathcal{M}}} \|\overline{n}_i - \overline{n}_j\|^2 \\ & + \sum_{f_i\cap f_j = e\in\partial\mathcal{C}} \frac{A_{f_i} + A_{f_j}}{3A_{\partial\mathcal{M}}} \|\Pi_{ij}\overline{n}_i - \overline{n}_j\|^2, \end{aligned} \tag{12}$$

where $\overline{n}_i, \overline{n}_j$ are mapped normals calculated by three parametric vertices:

$$\overline{n}_i = \frac{(\overline{x}_j^{f_i} - \overline{x}_i^{f_i}) \times (\overline{x}_k^{f_i} - \overline{x}_i^{f_i})}{\|(\overline{x}_j^{f_i} - \overline{x}_i^{f_i}) \times (\overline{x}_k^{f_i} - \overline{x}_i^{f_i})\|_2}. \tag{13}$$

At the boundary of the cuts, the normals are also transformed by the transition Π_{ij}, which is the transition from f_i to f_j.

In order to prevent degeneration, boundary area preserving should be added, i.e. $\overline{A}(\overline{\mathcal{X}}(\mathcal{M})) = A_{\partial\mathcal{M}}$. It is used together with E_{align} so that mapped boundary normals are axis-aligned, which has been verified in ℓ_1-based polycube parametrization algorithm [Huang et al. 2014]. The transition conditions at cut faces are formulated as hard constraints in our optimization problem, which is critical to the extraction of a seamless all-hex mesh.

Our algorithm uses the result from Poisson reconstruction as the initial solution. After using Gaussian elimination to remove the linear transition condition constraints, we apply the numerical method introduced by [Huang et al. 2014] to minimize the energy function. The step of polycube cleanup is applied to fix the possible small topology errors.

4.3 Integer constraints for all-hex meshing

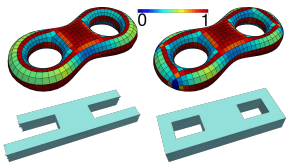
For all-hex meshing, integer constraints at boundary and cuts should be considered. Besides the coordinates of the corners, the translational transitions at cuts should be integers as well. Thus, for each $c_k \subset \mathcal{C}$ we add another three integer variables to represent the gap. After adding all such integer constraints into Equation (9), a mixed-integer programming [Bommes et al. 2009] is used to solve the problem again to get the final parametrization for hexahedral mesh extraction.

5 Results and Discussions

Our method is robust to handle various complex models. Figure 12 shows a gallery of our results, and the statistics about the results are listed in Table 1.

5.1 Comparisons

We compare our results mainly with polycube-based methods, ℓ_1-based Polycube [Huang et al. 2014] and Polycut [Livesu et al. 2013], since they are robust for all-hex mesh generation. Results show that our method can achieve lower distortion hexahedral meshes compared with these state-of-the-art methods.



ℓ_1-based Polycube. Indeed, ℓ_1-based polycube method can be viewed as a special case of our method by using a globally uniform frame field as an initial closed form to start the optimization. As a consequence, it has strong restriction on the topological structure (see Figure 1 and the right result of the inset). For genus-zero models, our method also can produce better results than the ℓ_1-based polycube method, which usually leads to more stairs because of lacking good enough initial axis alignment, and cannot even align the feature lines well (see Figure 2). More comparisons can be found in Figure 6, and their detailed hexahedral quality statistics,

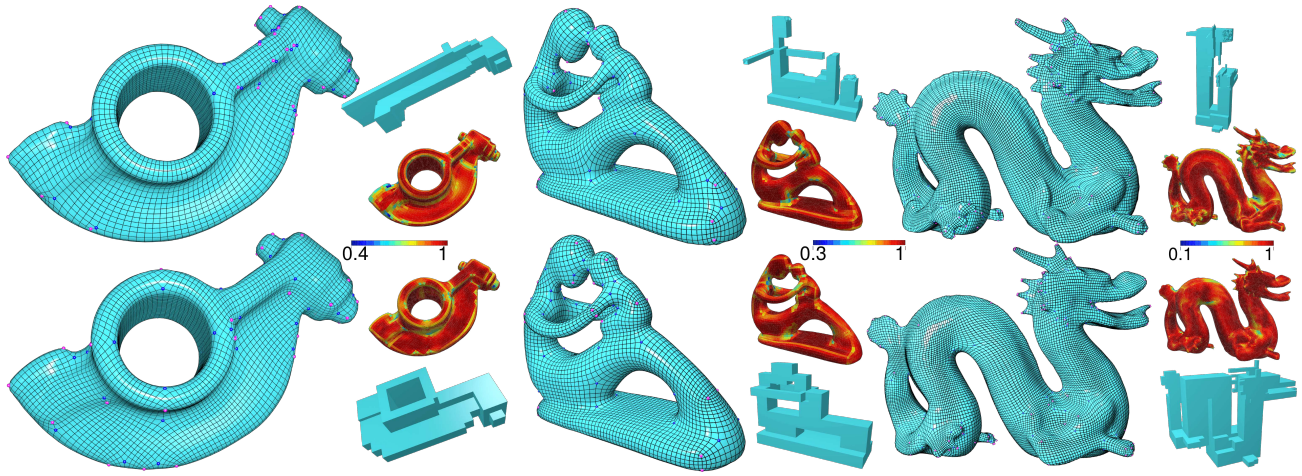


Figure 6: *Comparing to the results from the ℓ_1-based polycube method [Huang et al. 2014] (bottom), our results (top) have fewer corners (see Table 1) and better scaled Jacobian distributions (see Figure 7).*

measured in scaled Jacobian, are shown in Figure 7. Our method generates a larger number of high-quality elements and a smaller number of distorted elements. For the fertility model, the result of our method has 75.3% hexahedra with scaled Jacobian in $(0.9, 1.0]$, and it is only 66.1% for the ℓ_1-based polycube method. Our method also reduces the number of corners (see Table 1).

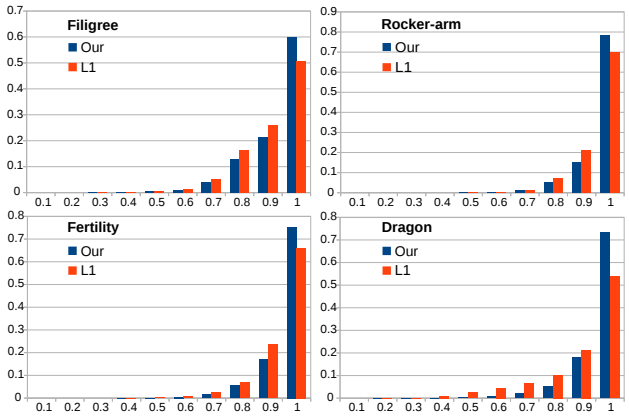


Figure 7: *Comparisons on scaled Jacobian distribution between our method and ℓ_1-based polycube method. The horizontal axis indicates scaled Jacobian, and the vertical one is the proportion of hexahedra in the corresponding scaled Jacobian interval.*

Polycut. In Figure 8, we compare our result with [Livesu et al. 2013] on the carter model. It is obvious that our result has a smaller number of corners (84 vs. 128), and the surface quadrilateral mesh follows the curvature tensor better.

The two features, frame-field based axis alignment and topology cuts, make our method robust to handle the complex spiral and knot models shown in Figure 14. The two high-quality hexahedral meshes without any corner are automatically generated using our method, which is impossible for the previous polycube methods.

5.2 Effect of initial frame field

The quality of the all-hex meshing result is influenced by the initial frame field in solving Equation (1). In the left of Figure 9, we first use a smaller weight value, 100, on the boundary normal alignment during the initial frame field generation. Thus the resulting

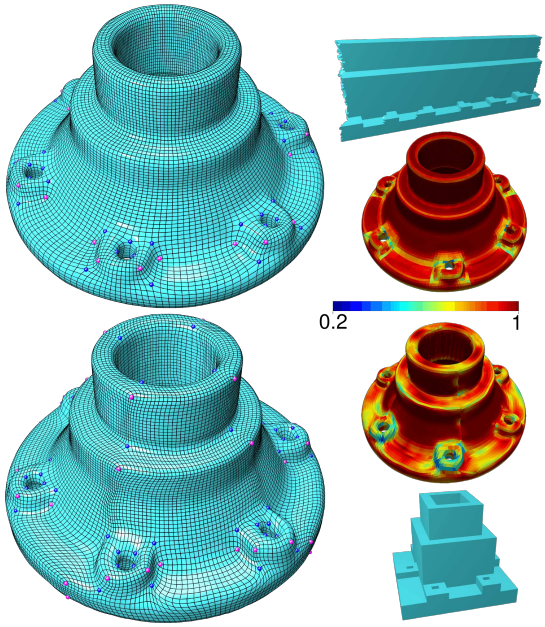


Figure 8: *Top row: Hexahedral meshing results using our method, Bottom row: Results using Polycut [Livesu et al. 2013].*

vector field is more straight and closer to a globally uniform one. As a consequence, the final hexahedral mesh is more like the one generated by the ℓ_1-based polycube method [Huang et al. 2014]. In the right of Figure 9, a larger weight value, 1000, is applied to make the initial frame field tightly aligned with the boundary normal. Comparing to the left one, the result is better in the sense of fewer unnecessary corners, which also demonstrates the advantage of using a non-uniform closed form to guide the all-hex meshing.

5.3 Model with multiple boundaries

In Figure 10, we demonstrate a model with a complex topological structure that has two disconnected boundaries, i.e. there is a void inside. Our method is able to reliably find the cuts, and generate a high-quality hexahedral mesh with a small number of corners. For the shell-like model, meshing one of its boundary surface into quadrilateral mesh and then extruding it towards another one will

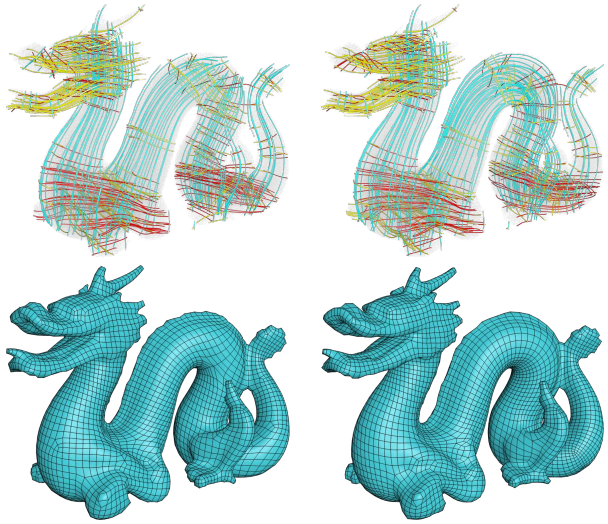


Figure 9: *Results from different initial frame fields.*

generate a hexahedral mesh with internal singularities if there are singularities in the quadrilateral mesh.

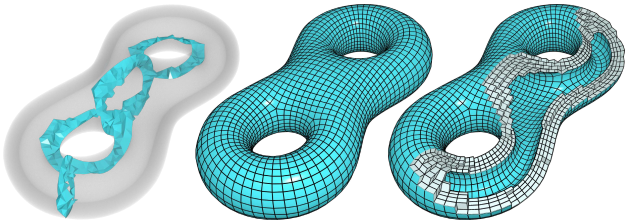


Figure 10: *The hexahedral mesh is generated on a model with two boundaries. The leftmost is the cuts.*

Since our method iteratively cuts the tetrahedral mesh, the handle loop generated by [Dey et al. 2013] may pass previous cuts through interior edges of the original mesh $\mathcal{M}$ as shown in the left of Figure 11. If such edges do not match each other on both sides of the cuts (the intersections of yellow and cyan regions in the middle top of Figure 11), the multiplication of transition matrices around those edges may be not identity since each cut has only one transition function. As a consequence, such edges may become internal singularities. To fix this, we find a contractible loop containing the mis-matched edges on the previous cut and adjust the handle according to the loop to get a new homotopic one without the mismatch problem (see the right of Figure 11).

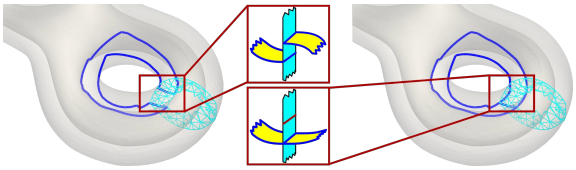


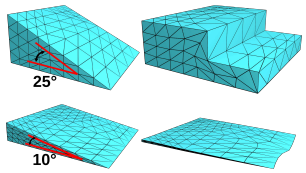
Figure 11: *The two boundary surfaces of the hollow-eight model $\mathcal{M}$ are connected after the first cut (indicated by the cyan triangular mesh). The second handle generated by [Dey et al. 2013] is shown as the blue lines in the left image, which contains mismatched interior edges in $\mathcal{M}$. After handle adjustment (right), the interior edges match each other, and lead to a valid cut.*

6 Conclusion

We propose a method to generate high-quality hexahedral meshes without internal singularities. The method inherits the robustness of polycube-based methods by getting rid of internal singularities. However, sharply different from the existing techniques, it significantly extends the topological structure of the 3D frame fields implied by hexahedral meshes from exact forms to closed forms, which increases the degrees of freedom by the first Betti number of a 3d-manifold. Therefore, our method is able to obtain all the topological structures of hexahedral meshes without internal singularities, and find high-quality results in a much larger space. Besides, the closed forms progressively constructed in our method provide better initial axis alignment for the polycube construction, which reduces unnecessary stairs, corners, and preserves the mesh features more accurately.

One of the major limitations of our method is the high computational cost in the extended ℓ_1-based polycube generation because the current numerical method is not specially designed. For such tasks, using Gaussian elimination to handle the constraints increases the density of Hessian matrix significantly, and introduces large costs. Improving the method of cut generation should be very helpful to reduce the number of constraints.

In our experiments, if the initial frame field contains singularities far from the boundaries, such singularities are difficult to be removed by our optimization, and introduce large distortion in Poisson reconstruction (see the left of Figure 13). Another failure case with inappropriate initial frame field is shown in the right of Figure 13. Under such circumstances, we choose smaller boundary alignment weight during initial frame field generation [Huang et al. 2011] to make the singularities close to the boundaries, which leads to similar results to the ℓ_1-based polycube method. In addition, our method inherits one of the limitation of ℓ_1-based polycube method, which may produce degenerate parametrization for slim wedges. As shown in the inset, our method can handle a wedge with an angle of 25 degrees, but will fail for a slim one with an angle of 10 degrees.



Although our cut generation method works well in our experiments, it cannot theoretically guarantee to obtain cuts. There exist extreme cases that the property (b), defined in Section 2, cannot be satisfied because $\partial\mathcal{B} - \partial\mathcal{M}_{c,k}$ might be split into more than two disconnected patches.

Although our method is able to generate high-quality hexahedral meshes for many complex models, there are plenty of interesting future works. One obvious work is to take the feature alignment, density and orientation control into account [Panozzo et al. 2014; Jiang et al. 2015]. Simplifying the topological structure of the closed-form induced polycube to get coarse meta meshes [Nieser et al. 2011] is also a valuable direction to explore.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments and suggestions. This work is partially supported by NSFC (No. 61522209, No. 61210007 and No. 61322204) and the Fundamental Research Funds for the Central Universities (No. 2015XZZX004-19 and No. 2016FZA5013). Thanks a lot for Tengfei Jiang on the discussion about cut generation and Jiong Chen for providing a code about hexahedral mesh extraction.

Dragon, Asian-dragon are courtesy of Stanford Scanning Repository. Nut, Fancy-ring and KPDloekr are provided by INRIA Gamma dataset. Hollow-sphere, Hollow-eight and Spiral-box are our own design. All the other 3D models are from the AIM@SHAPE shape repository.

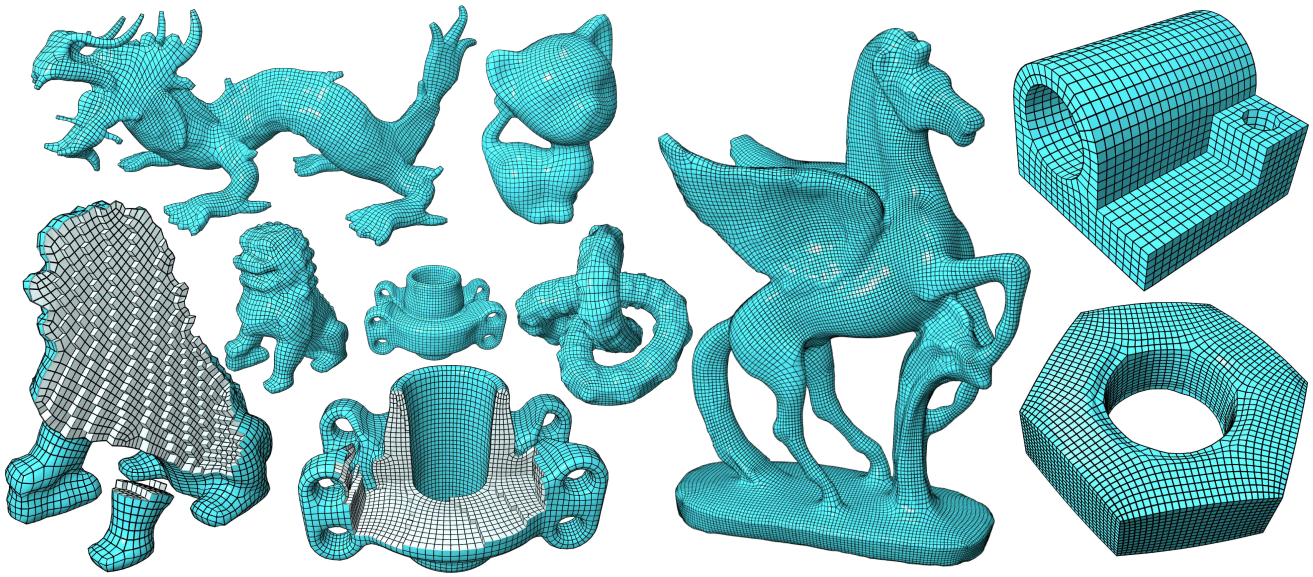


Figure 12: *Gallery of the results automatically generated by our method.*

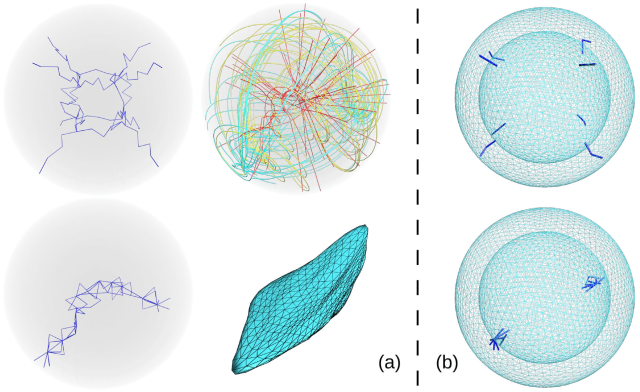


Figure 13: ***(a) Sphere.*** *Since the singularity lines of the given initial frame field (top left) are deeply inside the volume, our method cannot remove them and the singularities of the optimized frame field (bottom left) are far from the boundary. Top middle: The trace lines of the optimized frame field. Bottom Middle: Its highly distorted Poisson reconstruction.* ***(b) Hollow-sphere.*** *Given an initial frame field with* 8 *singularity chains (top), the frame field has some internal singularities after optimization (bottom).*

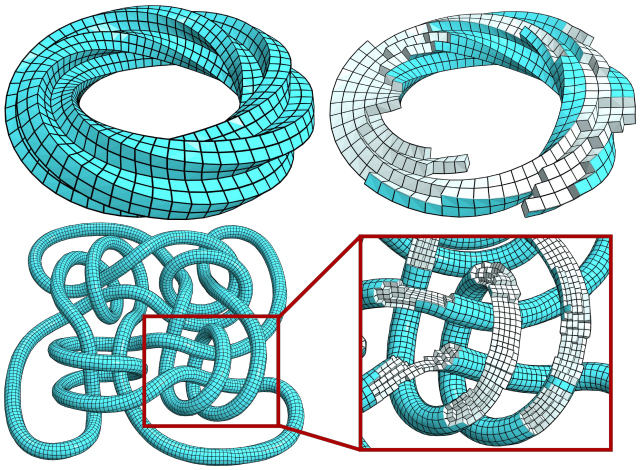


Figure 14: *For above two complex rings, our method produces good hexahedral meshes with high quality.*

References

ASPERT, N., CRUZ, D. S., AND EBRAHIMI, T. 2002. MESH: measuring errors between surfaces using the hausdorff distance. In *Proceedings of the 2002 IEEE International Conference on Multimedia and Expo*, 705–708.

BOCHKANOV, S. n.d. Alglib (www.alglib.net).

BOMMES, D., ZIMMER, H., AND KOBBELT, L. 2009. Mixed-integer quadrangulation. *ACM Transactions on Graphics 28*, 3 (July), 77:1–77:10.

CARBONERA, C., SHEPHERD, J., AND SHEPHERD, J. 2006. A constructive approach to constrained hexahedral mesh generation. In *Proceedings of the 15th International Meshing Roundtable*. 435–452.

DEY, T. K., AND GUHA, S. 1998. Computing homology groups of simplicial complexes in R^3. *J. ACM 45*, 2 (Mar.), 266–287.

DEY, T. K., FAN, F., AND WANG, Y. 2013. An efficient computation of handle and tunnel loops via reeb graphs. *ACM Transactions on Graphics 32*, 4 (July), 32:1–32:10.

GAO, X., DENG, Z., AND CHEN, G. 2015. Hexahedral mesh re-parameterization from aligned base-complex. *ACM Transactions on Graphics 34*, 4 (July), 142:1–142:10.

GARCÍA, I., XIA, J., HE, Y., XIN, S., AND PATOW, G. 2013. Interactive applications for sketch-based editable polycube map. *IEEE Transactions on Visualization and Computer Graphics 19*, 7, 1158–1171.

GREGSON, J., SHEFFER, A., AND ZHANG, E. 2011. All-hex mesh generation via volumetric polycube deformation. *Computer Graphics Forum 30*, 5, 1407–1416.

Model	#Tet	#Hex	Frame field Opt. (s)	Polycube Opt. (m)	#Corner	Scaled Jacobian (min/mean)	Mean/Hausdorff dist.
kitten†	11k	11941	18	2	28	0.435/0.922	0.652/12.1
spiral-box (Figure 2)	15k	5000	21	1	8	0.962/0.989	0.067/0.82
joint	40k	3785	66	4	24	0.778/0.984	0.254/8.08
knot†	40k	23616	72	11	20	0.359/0.921	1.08/11.7
KPDloekr† (Figure 14 bottom)	38k	23499	65	8	0	0.807/0.911	0.692/1.29
fancy-ring (Figure 14 top)	51k	1220	56	12	0	0.886/0.963	0.541/2.63
hollow-eight† (Figure 10)	47k	4357	110	31	16	0.346/0.847	0.660/4.40
rocker-arm† (Figure 3)	66k	7475	111	58	82	0.46/0.913	0.668/14.1
nut	67k	12636	139	11	24	0.728/0.977	0.451/10.3
Asian-dragon†	113k	31156	167	59	628	0.192/0.877	0.537/18.8
pegasus†	147k	120344	294	303	291	0.231/0.944	0.42/10.3
grayloc†	156k	24344	240	314	186	0.354/0.914	0.702/16.2
Chinese-lion†	168k	14508	256	334	248	0.289/0.910	1.444/29.2
dragon† (Figure 9 left)	177k	17320	291	110	323	0.161/0.887	0.852/24.2
dragon† (Figure 9 right)	177k	18824	356	112	300	0.214/0.885	0.765/24.4
filigree† (Figure 1)	195k	33454	298	87	647	0.24/0.895	0.934/10.0
filigree†* [Huang et al. 2014]	195k	33047	-	-	793	0.24/0.874	0.932/9.46
dragon† (Figure 6)	177k	114178	356	120	310	0.162/0.919	0.278/12.6
dragon†* [Huang et al. 2014]	-	117725	-	-	315	0.150/0.857	0.915/23.1
rocker-arm† (Figure 6)	66k	17594	111	60	83	0.464/0.936	0.421/8.06
rocker-arm†* [Huang et al. 2014]	-	24346	-	-	88	0.439/0.920	1.81/8.45
fertility† (Figure 6)	118k	16247	225	180	82	0.396/0.930	0.452/13.6
fertility†* [Huang et al. 2014]	-	17910	-	-	96	0.312/0.911	0.982/12.2
carter† (Figure 8)	123k	64911	236	149	84	0.224/0.923	0.153/4.64
carter†* [Livesu et al. 2013]	-	65214	-	-	128	0.250/0.894	1.44/9.99

Table 1: *Statistics of our results. The unit of mean and Hausdorff distance is 10^{-3} of the bounding-box diagonal of the input model, and the distance is measured with MESH tool [Aspert et al. 2002]. The timing is measured on a server equipped with 2 Xeon 2.60GHz CPUs. Padding layer is used for some models (denoted by †) to improve the quality. The rows denoted by * are the data of other methods.*

HUANG, J., TONG, Y., WEI, H., AND BAO, H. 2011. Boundary aligned smooth 3d cross-frame field. *ACM Transactions on Graphics 30*, 6 (Dec.), 143:1–143:8.

HUANG, J., JIANG, T., SHI, Z., TONG, Y., BAO, H., AND DESBRUN, M. 2014. ℓ_1-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics 33*, 3 (June), 25:1–25:11.

JIANG, T., HUANG, J., WANG, Y., TONG, Y., AND BAO, H. 2014. Frame field singularity correction for automatic hexahedralization. *IEEE Transactions on Visualization and Computer Graphics 20*, 8, 1189–1199.

JIANG, T., FANG, X., HUANG, J., BAO, H., TONG, Y., AND DESBRUN, M. 2015. Frame field generation through metric customization. *ACM Transactions on Graphics 34*, 4 (July), 40:1–40:11.

KOWALSKI, N., LEDOUX, F., AND FREY, P. 2014. Block-structured hexahedral meshes for cad models using 3d frame fields. *Procedia Engineering 82*, 59–71.

LI, Y., LIU, Y., XU, W., WANG, W., AND GUO, B. 2012. All-hex meshing using singularity-restricted field. *ACM Transactions on Graphics 31*, 6 (Nov.), 177:1–177:11.

LIN, J., JIN, X., FAN, Z., AND WANG, C. C. L. 2008. Automatic polycube-maps. In *Proceedings of the 5th International Conference on Advances in Geometric Modeling and Processing*, 3–16.

LIVESU, M., VINING, N., SHEFFER, A., GREGSON, J., AND SCATENI, R. 2013. Polycut: Monotone graph-cuts for polycube base-complex construction. *ACM Transactions on Graphics 32*, 6 (Nov.), 171:1–171:12.

LIVESU, M., SHEFFER, A., VINING, N., AND TARINI, M. 2015. Practical hex-mesh optimization via edge-cone rectification. *ACM Transactions on Graphics 34*, 4 (July), 141:1–141:11.

NIESER, M., REITEBUCH, U., AND POLTHIER, K. 2011. Cube-cover - parameterization of 3d volumes. *Computer Graphics Forum 30*, 5, 1397–1406.

PANOZZO, D., PUPPO, E., TARINI, M., AND SORKINE-HORNUNG, O. 2014. Frame fields: Anisotropic and non-orthogonal cross fields. *ACM Transactions on Graphics 33*, 4 (July), 134:1–134:11.

SHEFFER, A., ETZION, M., RAPPOPORT, A., AND BERCOVIER, M. 1999. Hexahedral mesh generation using the embedded voronoi graph. *Engineering with Computers 15*, 3, 248–262.

SHEPHERD, J. F., AND JOHNSON, C. R. 2008. Hexahedral mesh generation constraints. *Engineering With Computers 24*, 3, 195–213.

SHEPHERD, J. F., MITCHELL, S. A., KNUPP, P. M., AND WHITE, D. R. 2000. Methods for multisweep automation. In *Proceedings of the 9th International Meshing Roundtable*, 77–87.

SHIMADA, K. 2006. Current trends and issues in automatic mesh generation. *Computer-Aided Design & Applications 3*, 741–750.

STATEN, M., OWEN, S., AND BLACKER, T. 2005. Unconstrained paving and plastering: A new idea for all hexahedral mesh generation. In *Proceedings of the 14th International Meshing Roundtable*. 399–416.

TARINI, M., HORMANN, K., CIGNONI, P., AND MONTANI, C. 2004. Polycube-maps. *ACM Transactions on Graphics 23*, 3 (Aug.), 853–860.

WÄCHTER, A., AND BIEGLER, T. L. 2005. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming 106*, 1, 25–57.