Frame Field Singularity Correction for Automatic Hexahedralization

Tengfei Jiang[1], Jin Huang[1], Yuanzhen Wang[2], Yiying Tong[2], Hujun Bao[1]

[1]State Key Lab of CAD&CG Zhejiang University, [2]Michigan State University

Abstract—We present an automatic hexahedralization tool, based on a systematic treatment that removes some of the singularities that would lead to degenerate volumetric parameterization. Such singularities could be abundant in automatically generated frame fields guiding the interior and boundary layouts of the hexahedra in an all hexahedral mesh. We first give the mathematical definitions of the inadmissible singularities prevalent in frame fields, including newly introduced surface singularity types. We then give a practical framework for adjusting singularity graphs by automatically modifying the rotational transition of frames between charts (cells of a tetrahedral mesh for the volume) to resolve the issues detected in the internal and boundary singularity graph. After applying an additional re-smoothing of the frame field with the modified transition conditions, we cut the volume into a topologically trivial domain, with the original topology encoded by the self-intersections of the boundary of the domain, and solve a mixed integer problem on this domain for a global parameterization. Finally, a properly connected hexahedral mesh is constructed from the integer isosurfaces of (u, v, w) in the parameterization. We demonstrate the applicability of the method on complex shapes, and discuss its limitations.

Index Terms—automatic hexahedral meshing, frame field, field singularity, volumetric parameterization

1 Introduction

Automatic high quality hexahedral mesh generation has sometimes been dubbed the "Holy Grail" in the meshing community, as such meshes benefit finite element methods due to their tensor product nature, leading to improvement in both speed and accuracy. Recent development in quadrangulation in computer graphics and geometric modeling has stirred new research effort in this direction based on meshing methods guided by the cross-frame fields, fields of equivalence classes of local frames under the chiral octahedral symmetry group (the set of 24 rotations that keep a cube centered at origin invariant without changing the right-handed frame to a left-handed one).

Such methods first create a parameterization of the volume for 3D charts intersecting at common interfaces, followed by extracting the vertices of the hex mesh from the integral points in the parameter domain. The edges of the hexahedra in the mesh would then follow the gradient lines of the parameterization. For computational purposes, the boundary of the resulting mesh must conform to the original boundary, create patches sharing the same integer parameter value over smooth regions, and introduce sharp edges and corners near the original features of the mesh. Thus, feature alignment and angle distortion reduction are both linked to a high-quality frame field with one of the axes aligned to the boundary normal on all surface points.

In theory, CubeCover method [1], extending QuadCover method [2], formulated *necessary conditions* for the volumetric parameterization, and laid the foundation for automatic hexahedralization methods based on such parameterizations. They formulated the basic requirements of rotational and translational transitions on each interface between different charts or different parts of the same chart serving as the domains for parameterizations. The nontrivial transitions (cuts) of the domain are necessary to provide flexibility of creating a mesh with high-quality element shape. Otherwise, a polycube-like topology would be enforced, potentially leading to large distortion [3], undesirable for engineering or scientific computing purposes.

Combining the global volumetric parameterization with compatible rotational and translational transitions as specified in the CubeCover method with

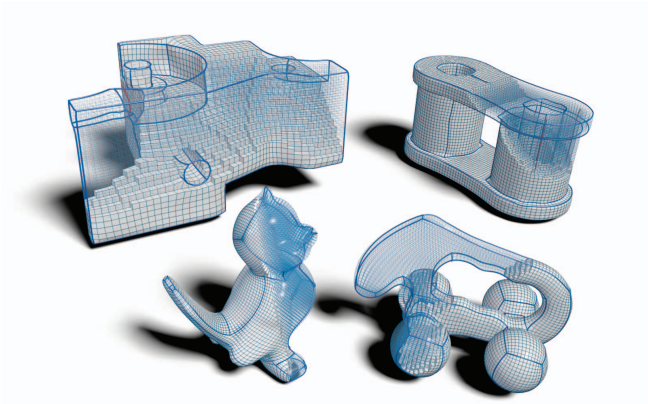


Fig. 1: Automatically generated all hexahedral meshes. Our method is able to handle complex models with various topologies.

Digital Object Indentifier 10.1109/TVCG.2013.250

automatic generation of guidance fields, one would imagine that an automatic hexahedral meshing tool would be straightforward to construct. However, unless manually constructed or adjusted, frame fields do not usually satisfy these compatibility conditions for non-degenerate clean volumetric parameterization. In fact, there can be a multitude of different types of problems if one computes rotational transitions from automatically generated frame fields that are continuous up to a rotation in the chiral octahedral symmetry group. The simplest example of the difficulty in such volume parameterization would probably be the case that two edges of a triangle are both along a same singularity line. Thus all the points in the triangle will share the same set of two variables, rendering the image of the triangle in the parameter domain degenerate (collinear). Some other examples include internal line singularities of degenerate types, boundary edges that require the pairs of triangles adjacent to them to fold onto one another, and nearby singularities forcing the parameters to jump from one integer to another across a short distance.

In this paper, we aim at formulating the common problematic cases in a complete singularity graph mathematically, providing a framework to automatically detect and treat them in a consistent way with the guarantee of convergence, and finally generating an all hexahedral mesh by solving a mixed integer programming problem after reducing the number of integer variables. This paper does not provide a sufficient condition for fixing all possible degeneracies, which remains an open problem. However we provide more comprehensive formulation and fixing strategy for the degeneracy problem in hexahedral remeshing than the state-of-the-art [3]. In summary, our main contributions include:

- We give a definition of surface singularities, and necessary conditions for admissible singularity graph of both internal and surface singularities for hexahedral meshing.
- We provide a procedure with convergence guarantee to fix the above defects in the singularity graph, which would lead to degenerate parameterization if left alone.
- We demonstrate a practical solution for hexahedral mesh generation guided by automatically generated guidance cross-frame input fields.

2 Related Work

As mesh generation has been an active research topic for a long period of time, a huge number of methods have been proposed to generate high quality triangular or tetrahedral meshes automatically [4], [5]. Most of them rely on Delaunay tessellation (dual of Voronoi diagrams), which ensures proper connectivity and element quality. However, quadrilateral or hexahedral remeshing is substantially more difficult because there is no such dual structure for non-simplicial elements. In 2D manifold quadrangulation, some recent works [6], [7], [8] use Morse-Smale Complexes (MSC) [9] to guarantee a pure quadrangulation based on a scalar function with periodic property. However, 3D MSCs do not necessarily lead to hexahedral structures. As a consequence, many hexahedral remeshing methods, such as sweeping [10] and paving [11], rely heavily on manual input to define the proper topological structure by decomposing the volume into simple parts. Such challenges make grid-based methods [12] or hex-dominant mesh-based methods [13], [14] often the practical choice for automatic remeshing in spite of their inferior results.

Recent progress shows that global parameterization has achieved great success in surface quadrangulation [2], [15] with the help of a smooth frame field defined on a manifold surface with compatible transition functions between charts. From such a non-degenerate parameterization, a pure quadrilateral mesh can be extracted by tracing the iso-lines. To extend such a scheme to hexahedral remeshing, [1] has proposed a global parameterization method for hexahedral remeshing, and [16] gives a method to automatically construct a desired frame field. These two works are most relevant to our method. Roughly speaking, we attempt to apply the parameterization method in CubeCover to the automatically generated frame field. However, the task is extremely challenging as the automatically generated frame field does not take into account the conditions on admissible singularity types (both inside the volume and on the boundary). In addition, without a coarse meta-mesh manually specified or generated from a manually specified frame field, a topologically sound partitioning of the volume for global parameterization is not straightforward to generate as one might expect. [17] also proposed a method for all hexahedral remeshing with the topological restriction of no internal singularity, thus leaving few degrees of freedom to optimize the shape of hexahedra. As noted in [1], it is often necessary to move the right angle transition line in the parameter domain on smooth regions of the surface into singular edges inside the volume to have better element quality, as the former turns the dihedral angle between a hexahedron's two faces into nearly 180°, while the latter only turns the sum of three such dihedral angles around the internal edge to 360°.

Singularity plays a critical role in detecting and remedying this issue. [18] lists many topological restrictions that arise in hexahedral meshing, and uses them in a frame-field-independent remeshing algorithm. The topological constraints in this work apply to the primal elements of a hexahedral mesh, such as node, edge and face etc. Some methods have been proposed to analyze the global topological structure of a quadrangular mesh or a 2D symmetry vector field by singularity graphs [19], [20]. In 2D quadran-

gulation, noisy singularity points lead to problematic parameterization results. Thus, some remedies have been proposed for denoising or adjusting the singularities in the frame field [2]. For hexahedral remeshing, [1] provides a definition on internal singularity types and proved some of its properties. Such singularity structures are also important to hexahedral mesh coarsening, e.g., [21] and [22]. However, there is no existing work on automatic surface singularity and internal singularity adjustment for frame fields used in hexahedral meshing, except a concurrent work on hexahedral meshing by Li et al. [3]. Their method does not address the potential issues involving surface singularities. Although such inadmissible singularities may be amended in their method by manually designed "guiding boxes" (all frames inside a given box are set to be aligned to the three edge directions of the box), they often appear in automatically generated frame fields and should be addressed for the sake of robustness.

3 Overview

Algorithm 1: Hexahedralization process overview.

Data: tetrahedral mesh Ω with one frame F per tet computed by, e.g., [16]
Result: hexahedral mesh with edges guided by F
Compute the singularity graph (Sec. 4);
while *there is a zigzag (Sec. 5.2)* **do**
  Remove zigzag by "straightening"
while *a compound edge (Sec. 5.3) adjacent to two admissible edges exists* **do**
  Split the compound edge:
  turn it into an admissible type;
  create a separate singularity path through vertices newly introduced in a local refinement;
Adjust the frame field (Sec. 5.4);
Reduce the number of integer variables (used in boundary and transitions, Sec. 6.1);
while *an untreated integer variable (Sec. 6.2) exists* **do**
  minimize the parameterization energy;
  round the integer variable closest to an integer;
  turn the variable into a constant;
Minimize the parameterization energy;
Construct hexahedral meshes with integer grid points;
Post-processing if necessary (Sec 7);

Our paper focuses on generating all hexahedral meshes from input cross-frame fields generated automatically by [16]. Given a tetrahedral mesh Ω, each tetrahedron is associated with an orthonormal frame $F = (U, V, W)$, with column vectors U, V, and W as the basis vectors. The frame is considered as a representative of a cross-frame (an equivalent class of 24 orthonormal frames with axes chosen from $\{U, -U, V, -V, W, -W\}$). To generate an all hexahedral mesh with edges following such an input, we compute a parameterization with a method similar to the one proposed in [1]. We loosely follow their notations below.

We denote the parameterization $f : \Omega \to \mathbb{R}^3$, which can be expressed in each chart (tetrahedron) as a linear function with three components $(u, v, w)^T$. In tetrahedron t, we denote its expression by f_t.

If we use integral lines of the gradients of the parameterization for edges, they must connect to each other across the tetrahedron boundaries. Thus, for two adjacent tetrahedra s and t, the transition from f_s to f_t for a parameterization whose integer grid points can be used in hexahedralization must satisfy

$$f_t = \Pi_{st} f_s + g_{st}, \tag{1}$$

where $g_{st} \in \mathbb{Z}^3$ and $\Pi_{st} \in \mathbb{O}$, the set of one of the 24 matrices for the cross frame containing the standard identity frame, a.k.a. the chiral octahedral symmetry group. Comparing with the transition from t to s, we have

$$\Pi_{st} = \Pi_{ts}^{-1}, \quad g_{st} = -\Pi_{st} g_{ts}. \tag{2}$$

We call any face with a non-identity rotational transition or non-zero translational transition a jump-face.

For a surface triangle a associated with tetrahedron t, to avoid cutting the corresponding hexahedron by the boundary surface, one of parameter must be an integer, with its gradient aligned to the surface normal. Thus, there is a transition Π_{ta} that makes the parameterization coordinates of a surface point $p \in a$ satisfy

$$(\Pi_{ta} f_t(p)) \cdot (1, 0, 0)^T = g_{ta}, \tag{3}$$

where $\Pi_{ta} \in \mathbb{O}$ aligns the U-axis of the frame to the normal of a, and $g_{ta} \in \mathbb{Z}$.

The translational part (the gap) g_{st} can be resolved during the parameterization process if the hex edge size can be adjusted, but the rotational part Π_{st} must be properly prescribed to avoid fold-overs and degeneracy in parameterization while following the given frame field.

For the parameterization with its gradients following the frame field to be smooth, a natural requirement for the rotational transitions is that they make the angles between the corresponding axes in F_t and $F_s \Pi_{st}$ respectively as small as possible. If one of these angles becomes greater than $\pi/2$, it can lead to large angle distortion in the parameterization. In addition, more constraints are required to avoid the defects caused by singularities as shown in the next section.

4 Rotational Transition and Singularity

A straightforward method to evaluate the rotational transition is to find the best transition matrices that minimize the following "alignment error":

$$\begin{aligned} &\|F_t - F_s\Pi_{st}\| \\ &\|(F_t\Pi_{ta})(1,0,0)^T - n_a\|, \end{aligned} \tag{4}$$

where n_a stands for the normal of the surface triangle a in t, and the matrix norm is the Frobenius norm, which is used throughout the paper. For a tetrahedron t containing more than one boundary faces, say a and b, $\Pi_{ta} \neq \Pi_{tb}$ or t should be split into two. When the rotation Π_{ta} minimizing the above error is non-unique, we choose the one fixing one of the axes. Such a setup is necessary to make the gradient fields of the parameterization smooth while remaining consistent with the guidance frame fields [1]. However, it does not guarantee degeneracy-free parameterizations in the presence of certain types of singularities as discussed below.

4.1 Internal Singularities

If the valence (number of adjacent hexahedral cell) of an internal edge in the final hexahedral mesh is not 4, it is an internal singularity. Such singularities can be detected by checking the rotational transitions. As formulated in [1], for an oriented tetrahedral mesh edge e, surrounded by a small counter-clockwise oriented dual edge loop passing through tetrahedra $(t_0, t_1, \cdots, t_k, t_0)$, we can define the type of the edge with respect to starting tetrahedron t_0 as

$$\text{type}(e, t_0) = \Pi_{t_k t_0}\Pi_{t_{k-1}t_k}\cdots\Pi_{t_0 t_1}. \tag{5}$$

If $\text{type}(e, t_0) \neq I$, we call it an *internal singularity edge*.

An internal singularity can lead to a non-valence-4 segment in the final hexahedral mesh, only if $\text{type}(e, t_0)$ is a rotation around one of the axes in the frame. Other types of singularities are said to be *compound*, and forced to be mapped to a point in any parameterization producing a hexahedral mesh consistent with the transitions, leading to degeneracy. It can be shown by contradiction. Suppose that the image of e in the parameterization is not a point. If we choose a sufficiently fine hexahedral mesh, the image is either parallel to hexahedral faces, or intersecting with a hexahedral face. For a compound singularity edge e, U, V, or W is not an eigenvector of $\text{type}(e, t_0)$, so its image cannot be parallel to one of the gradient lines of the parameters. On the other hand, the hexahedral face that it intersects with contains edges along U, V, or W, which cannot form a loop consistent with accumulated rotational transition $\text{type}(e, t_0)$, leading to a contradiction.

The additional requirement on the inner singularity vertex (Thm 2.2 in [1]) through counting valences would be satisfied automatically when the rotational transitions form the minimizer of the frame field transitions. Although valence 3 and 7 (or 4 and 8) cannot be distinguished by edge type, it is possible to detect high valences, e.g. rotation around Z with an angle $> 2\pi$, through first locally aligning the Z-axes of frames to the edge and then accumulating the rotation angle. In practice though, we have never seen any high valence singularity edges in automatically generated frame fields.

4.2 Surface Singularities

As shown in Section 5, internal singularities may lead to degeneracy in parameterization because of the constraints that they impose. Some surface edges produce similar constraints, and may cause degeneracy as well. Such constraints force two components of the parameterization coordinates along the edge to be both constant integers, i.e. the edge must be along some edges of the hexahedral mesh. We call such surface edges *surface singularities*, which are formulated mathematically below, with a definition consistent with the internal singularities.

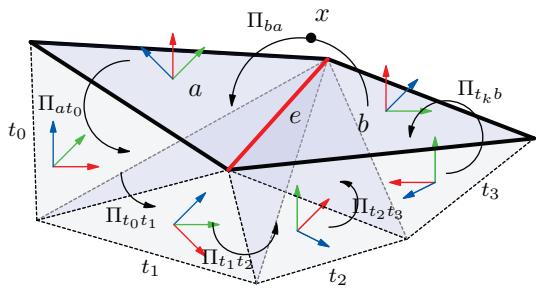


Fig. 2: Surface singularity example.

Similar to the internal case, for an oriented surface edge e, we also create a small counter-clockwise oriented loop around it, which passes through $(a, t_0, \cdots, t_k, b, x, a)$ (e.g., Figure 2), where the faces sharing e, a and b, are the triangles adjacent to tetrahedra t_0 and t_k, respectively, and x stands for a point outside of the tetrahedral mesh. We then define the type of the edge as

$$\text{type}(e, a) = \Pi_{ba}\Pi_{t_k b}\Pi_{t_{k-1}t_k}\cdots\Pi_{t_0 t_1}\Pi_{at_0}, \tag{6}$$

where Π_{ba} is a rotation around axis U aligning the V, W-axes on triangles a and b when U-axis on each triangle is aligned to the normal. More precisely, it minimizes

$$\|R_e(n_a, n_b)F_{t_0}\Pi_{t_0 a}\Pi_{ab} - F_{t_k}\Pi_{t_k b}\|,$$

where $R_e(n_a, n_b)$ is the rotation around e that aligns n_a (normal of a) to n_b (normal of b), which flattens the hinge formed by the two boundary triangles. If $\text{type}(e, a)$ is a rotation around an axis by $\pm\pi/2$, it creates a sharp edge on the surface in the parameter

domain, and denotes an admissible singularity. In other words, the valence of such an edge in the hexahedral mesh would be different from 2. If it is not a rotation around one of the frame axes, we have a *compound boundary singularity*.

The transition between surface faces Π_{ab} describes a rotation on V, W-axes in the parameter domain. The product of Π_{ab}'s along a loop around a surface vertex can be used to detect singularity vertices on the surface, which forms a sharp corner in the parameter domain with a discrete Gaussian curvature (angle defect) of multiples of $\pi/2$. Only $\pm\pi/2$ would lead to reasonable sharp corners. If the angle defect becomes π, the parameterization would wrap two surface squares around a surface vertex, leading to degeneracy. However, such a degeneracy never occurred in all our experiments based on automatically generated guidance fields.

5 Inadmissible Singularities

The whole *singularity graph* G is composed of the internal and surface singularity edges, and the vertices incident to these edges. The following two issues in such a graph will lead to degeneracy in the parameterization, and are thus necessary to eradicate:

- Compound singularity (a rotation not around any axis in the frame): It maps to a single point in the parameter domain, thus inducing degeneracy. In addition, a surface rotation around a frame axis by π is also treated as a compound singularity, because it indicates zero or 4 neighboring hexahedra, leading to great distortion except for the case with a valence-4 boundary edge whose adjacent boundary faces have nearly opposite normals.
- Zigzag (two same type consecutive singularity edges in one tetrahedron): Denote the edges by e_0 and e_1, and the tetrahedron by t, type$(e_0, t) =$ type(e_1, t). Such edges map to a straight line in the parameter domain, thus making the image of the tetrahedron degenerate.

These issues can be viewed as resulting from improper discretization of a smooth frame field with only admissible singularity types. A compound singularity can be viewed as several close-by singularity lines merged onto one tetrahedral edge. A zigzag can be explained as misalignment of the edges of tetrahedral mesh with the singularity line in the smooth frame field. Unless a manually constructed or adjusted frame field is used, these issues almost always exist and must be addressed.

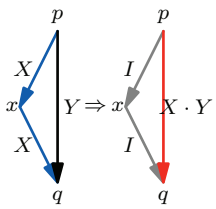
Based on the above observation, we propose a two-step method to schematically adjust the initial rotational transition set derived from Equation 4 to address the above issues. In the first step, we remove all the zigzag issues by straightening the singularities. In the second step, we split the compound singularities into admissible ones.

Additional care may be necessary to avoid extreme angle distortion associated with high hexahedral edge valences. However, for internal singularity edges, the valences are in most cases less than 6 when the singularity types are derived from the automatically generated guidance frame field. For boundary singularity edges, one may want to make sure that the dihedral angles do not deviate far from the dihedral angles specified by the singularity type in the parameter domain. Similarly, the boundary singularity vertex should have compatible Gaussian curvature in the parameter domain and on the mesh to avoid extreme angle distortion at the sharp corners.

5.1 Atomic Operation to Adjust the Singularities

For an internal face shared by tetrahedra s and t, modifying the rotational transition Π_{st} into $\Pi_{st}\Pi$ by a rotation matrix Π on the face will affect the types of its three edges. The type of one of these three edge type(e, s) that is counterclockwise oriented around the direction of $s \to t$ will change to type$(e, s)\Pi$ after the modification. Thus, the types of all these edges change by a same rotation. It can be likewise applied to surface singularities. We use this simultaneous change as the basic operation to remedy the issues in the singularity graph.

5.2 Zigzag Removal



As shown in the right inset, the triangle between tetrahedra s and t contains a zigzag $\overline{pxq}$ with type X. Changing the rotational transition Π_{st} into $\Pi_{st}X^{-1}$ turns the edges $\overline{px}, \overline{xq}$ into regular edges. The type of the third edge $\overline{pq}$ in this triangle with original type Y will become XY. Such an operation can be viewed as straightening the singularities $\overline{px}, \overline{xq}$ into $\overline{pq}$, and superpose their type onto the original type of $\overline{pq}$. If the original type of edge $\overline{pq}$ is not I, its type may become compound singularity after removing the zigzag, which will be split into admissible ones in the second step of our method.

Removing a zigzag may introduce new zigzags, but the procedure always terminates when we repeatedly find a zigzag to remove until all zigzags are eliminated. The number of zigzag removal operations to be performed cannot exceed the total number of singularity edges in the graph, since that number decreases by either one or two in each operation. The monotonic decreasing of singularity edge number provides a proof of convergence.

5.3 Compound Singularity Split

After removing all the zigzags in the graph, the remaining inadmissible singularities are compound singularities. To remove a compound singularity, we

split it into multiple admissible singularities inside its one-ring neighborhood without affecting other singularities or introducing zigzags (Figure 3). During this procedure, tetrahedra may be split into smaller ones through local refinement. In the refinement steps, we keep the rotational transition on split faces, and set the rotation transition to identity for the newly introduced faces. Thus, the split edges share the same type as the original ones, and the newly introduced edges are all regular edges.

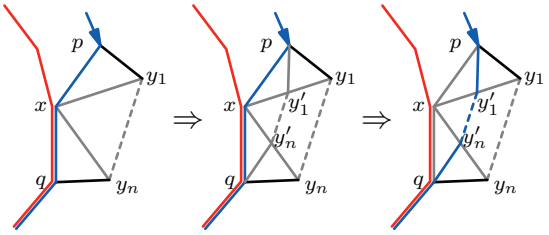


Fig. 3: Splitting a compound singularity edge $\overline{xq}$. The gray edges are regular, and the black edges can be singular.

We first find an open end node x of a compound singularity polyline, i.e. a node connected to only one compound singularity edge. In its one-ring neighborhood Ω_x, we pick one node p on one adjacent admissible singularity, and another node q on the compound singularity. Then, we subdivide each triangle on the boundary of Ω_x and its associated tetrahedron into four by inserting one new node at the middle of each boundary edge. As the newly introduced nodes y_i's are connected, and each original node is adjacent to some of them, we can always find a path of the form $\overline{py_1y_2\cdots y_nq}$. In particular, we can pick the shortest one. These edges and x form a fan of triangles (in the same orientation), with each edge $\overline{xy_i}$ being regular.

To avoid re-introducing zigzag edges when splitting the compound singularity, we further subdivide the triangle fan $x, \overline{py_1y_2\cdots y_nq}$ as follows. Each edge $\overline{xy_i}$, along with each of its incident tetrahedra in the one-ring, is split into two by inserting a new node y'_i at the middle. Then we get a new path $\overline{py'_1y'_2\cdots y'_nq}$ completely inside the original one-ring neighborhood, with each of its edges initialized to the identity type. We modify the rotational transition on all the triangles in the fan composed of x and $\overline{py'_1y'_2\cdots y'_nq}$ by multiplying the inverse matrix of the edge type for $\overline{px}$, turning $\overline{px}$ into a non-singular edge. Then the polyline $\overline{py'_1y'_2\cdots y'_nq}$ becomes an admissible singularity with the same type of $\overline{px}$. The vertex x is now incident to only two singularity edges, with one of them the original incident admissible singularity. According to Proposition 5.1 proved in the appendix, the other edge $\overline{xq}$ must also be an admissible singularity.

Proposition 5.1: If a vertex is incident to only two singularity edges, the types of the two edges must match in a certain local alignment.

The above operation can be viewed as splitting and shifting the singular segment $\overline{pxq}$ to the new paths. It fails when there exist loops of compound singularity edges, which contain no end node. However, it never occurred in all frame fields automatically generated during our experiments. The monotonic decrease in the number of compound singularity edges within each iteration guarantees the termination of the procedure.

In Figure 4, we demonstrate several typical cases in singularity adjustment on real data. To help visualize the details, we show close-ups on parts of the singularities enclosed by colored squares in the insets enclosed by ones with the same colors respectively. The purple box contains a zigzag on $\overline{pxq}$, and the blue box contains a compound singularity $\overline{pq}$.

5.4 Frame Field Adjustment

The above singularity adjustment removes the zigzag and compound inadmissible edges defined in this paper, but it may increase the alignment error, leading to possible difficulties in the subsequent parameterization step. By minimizing the alignment error with respect to F, we can update the frame field according to the new rotational transitions. For robustness, we adopt a scheme that incorporates the non-linear constraints gradually. With the rotational transitions fixed to the adjusted matrices, we first obtain an initial guess by turning the problem into a linear system, minimizing Eq. 4, without the constraints $F \in SO(3)$, i.e. treating each F as an arbitrary 3 by 3 matrix. We then incorporate the orthonormality constraints on F as soft constraints, by including a non-linear penalty term $w\|F^TF - I\|^2$ (w is set to 100 in all our results), and solve the optimization problem again with Gauss-Newton method starting from the initial guess. The solution will be converted to the closest ZYZ Euler angle representation, and serve as the initial value to minimize the alignment error in such a representation. We then transform the solution back to the 3 by 3 matrix representation as the updated frame field satisfying the hard constraints $F \in SO(3)$. In most cases, the largest angle between the corresponding axes of two adjacent tetrahedra is about 30° when measured with the initial rotational transition, but increases to about 120° after adjusting the rotational transition to resolve inadmissible singularities. After the above adjustment, it finally decreases to about 75°.

In some cases, the resulting singularities may be close to each other, resulting in high distortion in the parameterization. However, a simple post-processing of local relaxation as mentioned in Section 7 reduces the distortion of the parameterization in the hexahedral elements between these singularities, producing high quality hexahedral meshes.

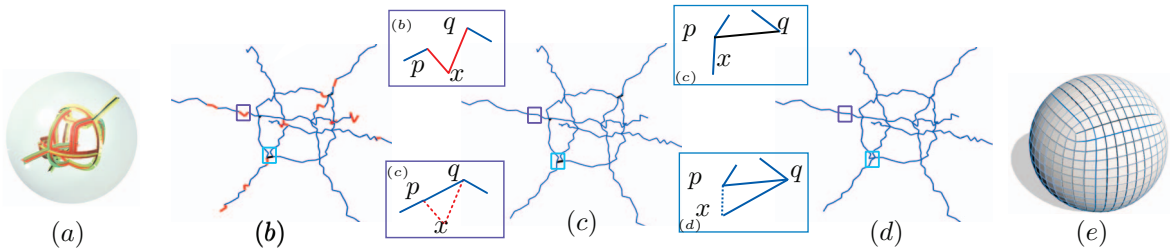


Fig. 4: Example of removing inadmissible singularities. (a) shows the input frame field visualized by streamlines. (b) The initial singularity graph that it induced contains many inadmissible singularities. The zigzag issues and compound singularities are shown in red and black respectively. The zigzag issues are resolved in (c), and the compound singularities treated in (d). After adjusting the singularities, an all hexahedral mesh (e) can be extracted from the parameterization result.

6 Parameterization

We follow essentially the same idea in the CubeCover method for parameterization, i.e. finding the minimizer of the following energy

$$\int_V \|df - F\|^2 \, d\,Vol,$$

which means that the gradient of each component of the parameterization should follow the corresponding axis of the given frame field F as much as possible. The main difference is that we are able to greatly reduce the number of integer variables through a simple preprocessing procedure, making the subsequent steps more efficient and robust.

6.1 Topological Simplification

We merge all the charts to get a global parameterization domain. To improve efficiency, instead of just creating a minimal spanning tree (MST) to remove a minimum set of transition variables across faces between tetrahedra adjacent in the tree as in [1], we additionally create a 3D domain with trivial internal topology to fill the volume, removing as many transition variables associated with faces as possible. Note that the original (possibly nontrivial) topology is encoded by the self-intersection of the new domain boundary. This can be achieved by slowly growing the domain to eventually occupy the whole volume, while maintaining a ball-like topology for the interior of the domain throughout the process. In Figure 5, we show a simple example of the topological simplification process for torus.

After using MST to globally align the frames while keeping the singularity structure intact, we start from a seed tetrahedron, gradually remove faces with trivial transitions to merge tetrahedra, and expand the domain until all tetrahedra are merged to it. Any non-rotational-jump-face is a candidate for removal. If both adjacent tetrahedra of such a face are already merged to the domain, it can still be removed when it contains a non-singularity edge adjacent only to this unremoved face. Since all the faces corresponding to the MST edge can be removed, the entire volume can always to be merged into a single domain. However, an arbitrary order of traversing such face candidates can leave more faces unremoved than necessary, leading to more integer variables. We follow a simple rule when expanding the domain, under which all candidate faces around an edge on the boundary of the growing domain are simultaneously removed. This ordering will leave few rotational transition face patches, which, together with at most P translational transition patches, cut the genus-P volume into a ball-like parameterization domain.

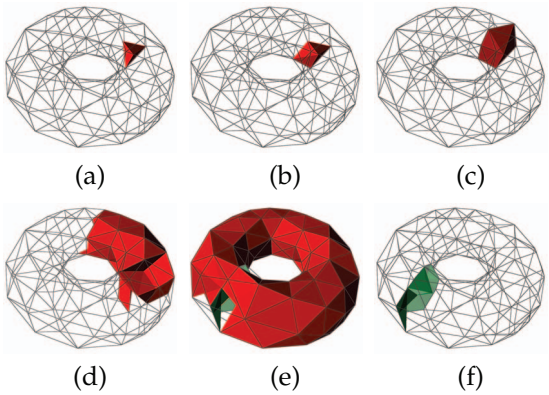


Fig. 5: An example of topological simplification. (a) the arbitrary tetrahedron chosen as the starting cell. (b) (c) the first several steps of expanding the domain by merging neighbouring tetrahedra. (d) (e) the intermediate stages of the merging process. In (e), we render the faces (in green) only for illustrative purposes. They are not detected until the merging process is completed. (f) the final remained faces that cannot be removed in order to maintain the ball-like topology.

6.2 Parameterization as a Mixed Integer Programming Problem

We can easily cluster the remaining faces into patches with same transition types. Each internal patch uses

only three integer jumps, and each boundary patch has one integer parameter. It is equivalent in theory to associate variables with each individual face, and then use Gaussian elimination to reduce the number of variables, or simply leave these variables in the final linear system with an increased number of equations. However, the Gaussian elimination process is significantly slower with a much larger number of constraints. Leaving the constraints as equations in the linear system would turn them into soft constraints when we solve the over-constrained system using least squares method. With the patches clearly identified, we can actually leave a small number of integer constraints only on the internal singularity lines and the boundary patches, and use three floating point values per internal patch. This approach can greatly reduce both the integer degrees of freedom of the system and the number of constraints, and thus expedite the process of the Gaussian elimination when we enforce the hard constraints.

The final system normally contains less than few tens of integer variables for a given input frame field. We successively snap one integer variable from the floating point value obtained in the previous solve in a greedy fashion. The linear system is assembled through first removing redundant hard constraints, and then eliminating them from the variables by substitution. This can be done on the gradient operator. After that, the reduced gradient operator and its transpose can be assembled as the reduced Laplacian operator for the optimization step. The construction process may take long if the model contains a large number of vertices, but the subsequent linear systems turn out to take little time for each solve when we call the sparse linear solver of Matlab.

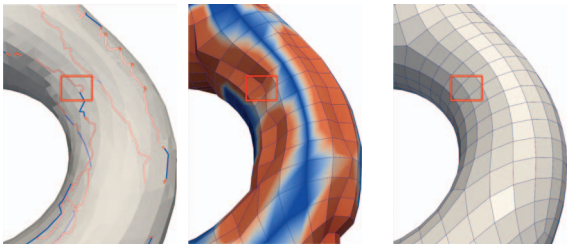
With a singularity graph containing only admissible singularities, the parameterization may still contain degenerate or flipped tetrahedra initially, due to geometric distortion and numerics. In experiments, we use five iterations of local stiffening [15] to alleviate the situation. However, this method can not guarantee flip-free parameterization. Thus we adopt the same strategy in [3] to leave this problem into hexmesh extraction. Besides, there are some degeneracies associated with internal singularities which are very close to the surface. This issue is handled by a post-processing procedure as detailed in the next section.

We use a simple approach to produce the final mesh. First, we generate the integer points $(u, v, w) \in \mathbb{Z}^3$ and half-integer points $(u + 1/2, v + 1/2, w + 1/2)$ within each tetrahedron, and snap nearby points of the same type together through a spatial indexing structure to avoid potential duplicates near the faces. Then, starting from each half-integer point which serves as the center of a hexahedron, we follow the U,V,W directions to find the 8 corresponding corner vertices from the set of integer points and produce the connectivity information for the hexahedral mesh. During this process, we keep track of how far it needs to go in the parameterization domain, and the change of directions to handle any jump-faces encountered along the path.

7 Results and Discussion

The singularity correction method mentioned in Section 5 is able to resolve all the zigzag and compound singularity issues in our experiments. However, it remains an open problem to give a *sufficient condition* on singularity graphs compatible to non-degenerate hexahedral meshing.

In our experiments, some singularities close to the surface lead to degeneracy as well. As shown in Figure 6, some singularities sink right below the boundary surface of the tetrahedra mesh, and lead to degeneracies in regions between them and the boundary. As a result, the boundary of output hexahedral mesh at these singularities edges is not aligned with the input boundary.



(a) (b) (c)

Fig. 6: The red singularity edges in (a) lead to degeneracy and the sunk part (in red) in (b), but the defects can be easily fixed by surface snapping and post-smoothing (c).

Similar to zigzag and compound singularity, discretization of a singularity in a smooth cross-frame field which is close to, but not exactly on the surface (possibly due to the numerics of the frame field generation process) may lead to such singularities. This issue is inevitable when few assumptions can be made on the automatically generated cross-frame field. The difficulty of solving it lies in that such a degeneracy cannot be locally detected from the singularity graph. Fortunately, such sunk parts are all narrow and shallow (often less than two layer of tetrahedra) as they resulted from near-misses, and more importantly, the output hexahedral mesh still shares the same topology to the input tetrahedral mesh. Thus, we simply first snap the sunk nodes in the hexahedral mesh onto the tetrahedral mesh boundary by projecting them along the normal, and then improve the quality of the hexahedral mesh by using Mesquite software [23].

We tested the proposed method on several models. The statistics on the models are given in Table 1 and Table 2. The timing is measured on a PC with an I7-940 CPU at 2.8 GHz and 12 GB RAM. To measure

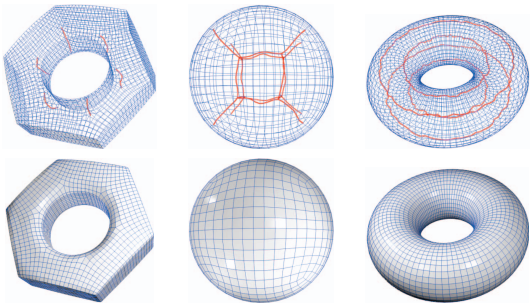


Fig. 7: Topological structures of the singularities.

the dihedral angles of an edge in a hexahedron, the normal of each adjacent quadrilateral face is evaluated by averaging the four triangle normals in two different tessellations of the quadrilateral face. As shown in the table below, nearly all the elements are with decent shape quality measures.

Model	Tet	Comp	Zz-inner	Zz-surf	T(s)
Torus	30k	0	100	15	0.04
Nut	65k	0	0	5	0.10
Sphere	80k	0	26	10	0.05
Pretzel	300k	5	204	35	6.27
Sculpture	105k	0	8	1	0.11
Fandisk	301k	0	148	16	0.22
Elk	123k	12	318	58	5.04
CAD1	68k	0	36	2	0.07
CAD2	130k	0	56	1	0.12
CAD3	530k	0	60	132	0.54

TABLE 1: Statistics of the results. The number of input tetrahedra, inadmissible singularity edges (Compound, Zigzag-inner and Zigzag-surface), and time spent (in seconds) for fixing them are in columns Tet, Comp, Zz-inner, ZZ-surf and T, respectively.

Model	T	Hex	Angle	Length	Scaled_jac	Haus.
Torus	0.5m	12768	90.0/85.7	1.53/1.08	0.974/0.827	0.448
Nut	1m	6640	90.0/88.8	1.28/1.04	0.966/0.389	0.286
Sphere	1.5m	7776	90.1/83.4	1.50/1.05	0.976/0.757	0.745
Pretzel	1.17m	3024	90.0/89.4	1.62/1.13	0.822/0.0203	0.605
Sculpture	1.3m	6348	89.9/88.4	1.42/1.07	0.973/0.557	0.295
Fandisk	5m	24001	90.0/82.1	1.70/1.02	0.943/0.01	0.341
Elk	1.1m	21462	90.0/79.4	1.58/1.01	0.936/0.282	1.33
CAD1	0.5m	21530	89.9/82.4	1.37/1.09	0.958/0.01	0.709
CAD2	2m	13788	90.0/87.3	1.24/1.01	0.981/0.300	3.55
CAD3	81.5m	17784	90.0/88.7	1.18/1.01	0.974/0.208	1.09

TABLE 2: Statistics of the results (average/minimal). The time used in parametrization is listed in the second column. The following columns list mean/minimal of dihedral angles, edge lengths and scaled Jacobian [24], and the Hausdorff distance ($10^{-3}\times$ bounding box diagonal) for measuring the quality of the output hexahedral mesh.

As shown in Figure 7, our method preserves the symmetries captured by the automatically generated frame fields. Even for high genus models, our method is able to automatically find a proper topological structure.

In the following figures, to demonstrate the parameterization results, we show the adjusted singularity graphs, resulting hexahedral meshes, and their cutaway views from left to right. In Figure 8, without any manual input, the models Pretzel, Elk and Fandisk are remeshed into a polycube-like topology, as indicated by their frame fields. In Figure 9, several internal singularities are automatically introduced on the models CAD1, Sculpture and CAD2, producing excellent element quality.

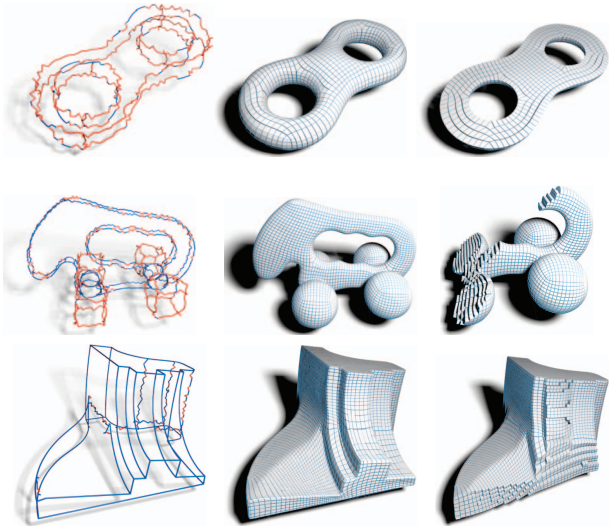


Fig. 8: Results with polycube-like structure. The red lines are near-misses, exposed on the boundary after post-processing.

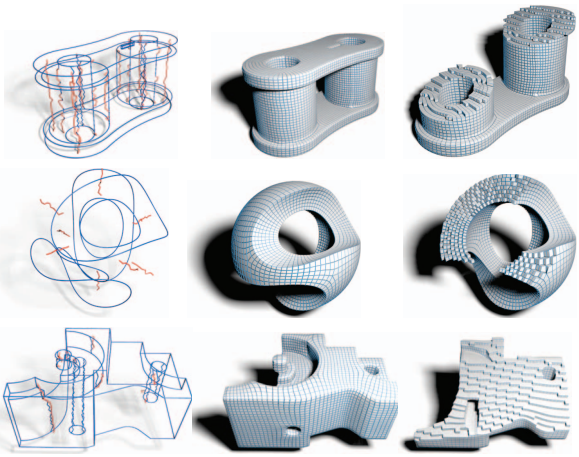


Fig. 9: Results on CAD1, sculpture and CAD2 contain internal singularities.

In Figure 10, we show the result of our method applied on the complex model CAD3. Manually constructing a meta-mesh for such models would have been extremely time-consuming. The models have some small features, which even cause problems in some surface quadrangulation techniques. By choosing a relatively small element size, our method is able

to generate an all hexahedral mesh while preserving important features at the same time. The initial singularity graph extracted from the input frame field has roughly depicted the final connectivity structure. However, the original singularities are on or near the surface, such near-misses can lead to distorted and even degenerate parameterization. After automatic adjustment, all singularities are snapped to the surface for the model shown, resulting in a polycube-like topology.

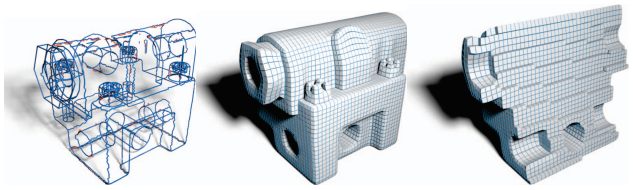


Fig. 10: Our method can automatically remesh the complex model CAD3 while preserving its important features without time-consuming manual meta-mesh construction.

7.1 Comparisons

Automatic generation of high quality hexahedral meshes is still an open problem. Many methods have been proposed under various assumptions. [17] assumes no internal singularities inside of the object. [1] requires a valid singularity graph in the input. [14] allows non-hexahedral elements in the resulting mesh. The most closely-related work [3] also takes a frame field in the input, and tries to reduce the inadmissible singularities. Although our method shares some of the same limitations, such as lack of guarantee in eliminating all inadmissible singularities (e.g. in presence of looped compound singularities), our method has the following advantages:

- In addition to all the inadmissible cases mentioned in [3], our method can detect and fix surface inadmissible singularities. As shown in Figure 12, the inadmissible surface singularities (top left) forces the associated elements to be degenerate, while our method is more robust by taking such inadmissible singularities into account.
- Our method guarantees convergence for both zigzag and compound edge removal steps. The compound edge collapsing strategy [3] may introduce new zigzags (Figure 11), and is thus without convergence guarantee.
- Near miss cases are often inevitable in automatically generated frame fields. We are the first to recognize and address such issues.
- There are known specific local degenerate cases that our method can handle but the method in [3] cannot. For instance, a face with three singularity edges in types R_u, R_v, and R_v cannot be properly handled either in their "matching adjustment" phase, "improper singular edge collapse" phase, or their ad hoc split method (Figure 11). Our pipeline can trivially handle such cases by first removing zigzag, and then splitting the compound singularity.
- Unlike our subdivision strategy, the collapse strategy used in [3] leads to possible numerical issues in parameterization because of the numerous poorly shaped tetrahedral elements introduced in fixing long compound singularities (Figure 13).

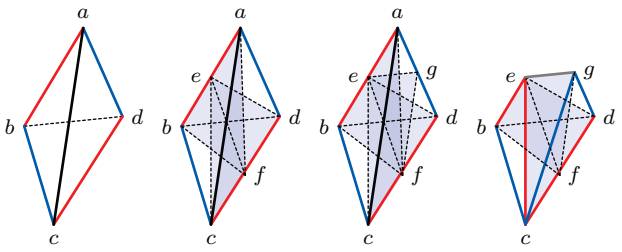


Fig. 11: Left: A tet with R_u (blue), R_v (red) singularities and a compound singularity (black). Following the steps in [3], the tet will be split at edges ab and dc to make each face around ac contain only one admissible singularity (middle left), and then ad is split to make ac collapsible (middle right). Zigzags occur after the collapse (right), which cannot be removed by the "matching adjustment".

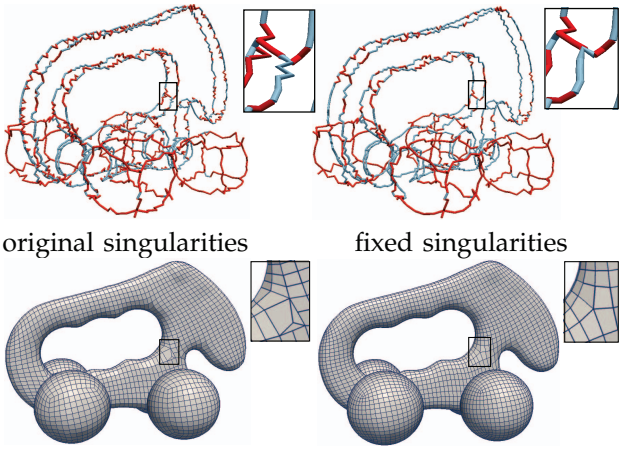


Fig. 12: Compared with the method in [3], which lacks the ability of detecting and fixing inadmissible surface singularities, our method is more robust in generating high quality hexahedral meshes with their presence.

8 Conclusion

We present a global volumetric parameterization-based tool to automatically generate an all hexahedral mesh based on a 3D frame field. Although a manually constructed or adjusted frame field may

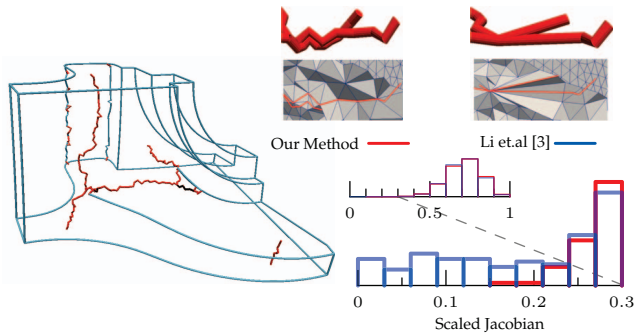
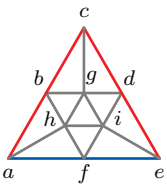


Fig. 13: Unlike our method, the collapse strategy [3] may introduce numerous poorly shaped tetrahedral elements. The histograms show the scaled Jacobian of the tetrahedral meshes. Minimal scaled jacobian of our result is 0.176, and Li et.at [3] is 0.003. Their results contains 55 cells with scaled jacobian < 0.176.

lead to a singularity graph compatible to a non-degenerate parameterization, an automatically generated one usually will not. To eliminate degeneracy in parameterization caused by conflicting rotational transitions in an automatically generated frame field, the definitions and some analysis on commonly seen inadmissible internal and surface singularity types are provided in this paper. We also devised a framework to adjust these problematic singularities by applying a sequence of local operations with guaranteed convergence.

The major limitation of this work is that it cannot detect and fix all the conflicting geometric and topological by using the definition of inadmissible singularity, and thus the method does *not* provide a sufficient condition to guarantee a complete solution for automatic all hexahedral remeshing. There are degenerate cases that cannot be fixed by current methods. Indeed, we find that even a singularity graph containing no zigzag or compound singularities can still lead to degeneracy, for example, near-misses.



We conjecture that it may be related to near misses or certain global inadmissible structure of the singularity graph, which forces certain singularity lines to be mapped to a single point in the parameter domain. For example, as shown in the inset (where red and blue lines represent different types of singularities, and gray lines are non-singularities) these constraints force a and e to have the same parameterization values.

We mainly focused on the singularity structure. Thus, to get a valid parameterization, we often use a relatively small element size, which leads to an excessively large number of hexahedra. As shown in [8], sizing is important for a more controllable tessellation, and proper sizing can lead to coarser hexahedral mesh. Extension on some recent works [25], [26] can potentially be employed to coarsen the results into better meshes.

REFERENCES

[1] M. Nieser, U. Reitebuch, and K. Polthier, "CubeCover - parameterization of 3d volumes," *Computer Graphics Forum (SGP)*, vol. 30, no. 5, pp. 1397–1406, 2011.

[2] F. Kälberer, M. Nieser, and K. Polthier, "Quadcover - surface parameterization using branched coverings," *Computer Graphics Forum*, vol. 26, no. 3, pp. 375–384, 2007.

[3] Y. Li, Y. Liu, W. Xu, W. Wang, and B. Guo, "All-hex meshing using singularity-restricted field," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 177:1–177:11, 2012.

[4] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, and C. Yang, "On centroidal voronoi tessellation-energy smoothness and fast computation," *ACM Trans. Graph.*, vol. 28, no. 4, pp. 101:1–101:17, 2009.

[5] P. Mullen, P. Memari, F. de Goes, and M. Desbrun, "Hot: Hodge-optimized triangulations," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 103:1–103:12, 2011.

[6] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart, "Spectral surface quadrangulation," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1057–1066, 2006.

[7] J. Huang, M. Zhang, J. Ma, X. Liu, L. Kobbelt, and H. Bao, "Spectral quadrangulation with orientation and alignment control," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 147:1–147:9, 2008.

[8] M. Zhang, J. Huang, X. Liu, and H. Bao, "A wave-based anisotropic quadrangulation method," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 118:1–118:8, 2010.

[9] H. Edelsbrunner, J. Harer, and A. Zomorodian, "Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds," *Discrete and Computational Geometry (SoCG)*, vol. 30, no. 1, pp. 87–107, 2001.

[10] B.-Y. Shih and H. Sakurai, "Automated hexahedral mesh generation by swept volume decomposition and recomposition," in *5th International Meshing Roundtable*, 1996, pp. 273–280.

[11] M. L. Staten, R. A. Kerr, S. J. Owen, and T. D. Blacker, "Unconstrained paving and plastering: Progress update," in *In Proceedings, 15th International Meshing Roundtable*, 2006, pp. 469–486.

[12] Y. Su, K. Lee, and A. S. Kumar, "Automatic hexahedral mesh generation for multi-domain composite models using a hybrid projective grid-based method," *Computer-Aided Design*, vol. 36, no. 3, pp. 203 – 215, 2004.

[13] S. Yamakawa and K. Shimada, "Hex-dominant mesh generation with directionality control via packing rectangular solid cells," in *Proceedings of Geometric Modeling and Processing 2002*, 2003, pp. 2099–2129.

[14] B. Lévy and Y. Liu, "Lp centroidal Voronoi tessellation and its applications," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 119:1–119:11, 2010.

[15] D. Bommes, H. Zimmer, and L. Kobbelt, "Mixed-integer quadrangulation," *ACM Trans. Graph.*, vol. 28, 3, no. 3, pp. 77:1–77:10, 2009.

[16] J. Huang, Y. Tong, H. Wei, and H. Bao, "Boundary aligned smooth 3d cross-frame field," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 143:1–143:8, 2011.

[17] J. Gregson, A. Sheffer, and E. Zhang, "All-hex mesh generation via volumetric polycube deformation," *Computer Graphics Forum (SGP)*, vol. 30, no. 5, pp. 1407–1416, 2011.

[18] J. Shepherd, "Topologic and geometric constraint-based hexahedral mesh generation," Ph.D. dissertation, University of Utah, 2007.

[19] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun, "Designing quadrangulations with discrete harmonic forms," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, ser. SGP '06, 2006, pp. 201–210.

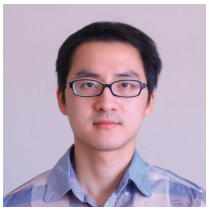
[20] J. Palacios and E. Zhang, "Rotational symmetry field design on surfaces," *ACM Trans. Graph.*, vol. 26, no. 3, 2007.

[21] A. Woodbury, J. Shepherd, M. Staten, and S. Benzley, "Localized coarsening of conforming all-hexahedral meshes," *Engineering with Computers*, vol. 27, no. 1, pp. 95–104, 2011.

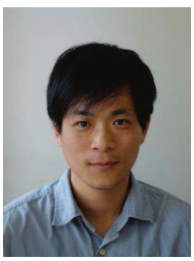
[22] S. J. O. Tim I. Miller, Steven E. Benzley, "Using edge valence prediction to drive localized all-hex coarsening," in *20th International Meshing Roundtable*. Springer-Verlag, 2011, pp. 23–26.
[23] M. L. Brewer, L. F. Diachin, P. M. Knupp, T. Leurent, and D. J. Melander, "The mesquite mesh quality improvement toolkit." in *12th International Meshing Roundtable*, 2003.
[24] J. F. Shepherd and C. J. Tuttle, "Quality improvement and feature capture in hexahedral meshes," The University of Utah, Tech. Rep. UUSCI-2006-029, 2006.
[25] C.-H. Peng, E. Zhang, Y. Kobayashi, and P. Wonka, "Connectivity editing for quadrilateral meshes," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 141:1–141:12, 2011.
[26] M. Tarini, E. Puppo, D. Panozzo, N. Pietroni, and P. Cignoni, "Simple quad domains for field aligned mesh parametrization," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 142:1–142:12, 2011.

Tengfei Jiang is a Ph.D. student in the State Key Laborate of CAD&CG, Zhejiang University, Hangzhou, China. He received his BS degree in Computer and Science from Nanjing University of Posts and Telecommunications in 2010. His research intersects include geometry processing and remeshing, especially for hexahedral remeshing.

Jin Huang is an associated professor in the State Key Lab of CAD&CG of Zhejiang University, P.R.China. He received his PhD. degree in Computer Science Department from Zhejiang University in 2007 with Excellent Doctoral Dissertation Award of China Computer Federation. His research interests include geometry processing and physically-based simulation. He has served as reviewer for ACM SIGGRAPH, EuroGraphics, Pacific Graphics, TVCG etc.



Yuanzhen Wang received the BE degree from Zhejiang Univeristy, Hangzhou, China, in 2009, and the PhD degree from Michigan State University in 2013. His research interests include geometry processing and discrete differential geometry. He is currently a software development engineer in Microsoft.



Yiying Tong is an assistant professor at Michigan State University. Prior to joining MSU, He worked as a postdoctoral scholar at Caltech. He received his Ph.D. degree from University of Southern California in 2004. His research interests include discrete geometric modeling, physically-based simulation/animation, and discrete differential geometry. He received the U.S. National Science Foundation (NSF) Career Award in 2010.

Hujun Bao is a Cheung Kong professor in the school of computer science and technology in Zhejiang University, and the director of State Key Laboratory of CAD & CG. He received the BS and PhD degrees in applied mathematics from Zhejiang University in 1987 and 1993, respectively. His research interests include geometry computing, vision computing, real-time rendering and virtual reality.