Polycube Shape Space

Hui Zhao 1,2 † Xuany Li 4 ‡ Wencheng Wang 1,2 § Xiaoling Wang 3 Shaodong Wang 1,2 Na Lei 5 ¶ Xiangfeng Gu 4

1 State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences
2 University of Chinese Academy of Sciences 3 University of Science and Technology Beijing, China
4 State University of New York at Stony Brook, USA 5 Dalian University of Technology, China

Figure 1: (a) the “kitten” mesh of genus one, (b) the polycube graph, (c) the homology loops, (d) the arc colorization, (e) the initial directions, (f) the polycube topology.

Abstract

There are many methods proposed for generating polycube polyhedrons, but it lacks the study about the possibility of generating polycube polyhedrons. In this paper, we prove a theorem for characterizing the necessary condition for the skeleton graph of a polycube polyhedron, by which Steinitz’s theorem for convex polyhedra and Eppstein’s theorem for simple orthogonal polyhedra are generalized to polycube polyhedra of any genus and with non-simply connected faces. Based on our theorem, we present a faster linear algorithm to determine the dimensions of the polycube shape space for a valid graph, for all its possible polycube polyhedrons. We also propose a quadratic optimization method to generate embedding polycube polyhedrons with interactive assistance. Finally, we provide a graph-based framework for polycube mesh generation, quadrangulation, and all-hex meshing to demonstrate the utility and applicability of our approach.

CCS Concepts

- Mathematics of computing → Graphs and surfaces;
- Computing methodologies → Mesh models; Mesh geometry models;

1. Introduction

The relationship between a graph and a polyhedron is a crucial and challenging problem. However, there are few of attempts on it. Steinitz’s theorem [Zie12; GKKZ67] establishes a bi-directional relationship between a graph and a convex polyhedron in \( \mathbb{R}^3 \) by the statements that “a graph is the skeleton graph of a convex polyhedron if and only if it is 3-connected and planar.” Branko Grünbaum call it “the most important and deepest known result on 3-polytopes” [GKKZ67].

A \( k \)-connected graph remains connected when few than \( k \) vertices are removed. A planar graph can be embedded in the plane. If each vertex has \( k \) neighbors, it is called a \( k \)-regular graph. A bipartite graph is a graph whose vertices can be divided into two disjoint and independent groups, a bipartite graph does not have odd-length cycles.

Steinitz’s theorem works only for convex polyhedra. Eppstein et al. [EM10] propose an analogous theorem for a special kind of non-convex polyhedra named simple orthogonal polyhedra, which satisfies three requirements: 1) the sphere topology, 2) simply connected faces, 3) exactly three mutually-perpendicular axis-parallel edges meeting at every vertex. Eppstein’s theorem states “a graph is the skeleton graph of a simple orthogonal polyhedron if and only if it is bipartite, planar, 3-regular and removal of any 2 of its vertices disconnects it into at most 2 components.” Furthermore, based on

† alanzhao@163.com
‡ xuanli2@cs.stonybrook.edu
§ whn@ios.ac.cn, corresponding author
¶ nalei@dlut.edu.cn, corresponding author
these characterizations of simple orthogonal polyhedra, Eppstein et al. [EM10] present a linear algorithm to construct a simple orthogonal polyhedron from a valid graph.

Simple orthogonal polyhedra is a special kind of well-known polycube polyhedron in the graphics community, whose every face is perpendicular to one of the coordinate axes. In this paper, we prove a theorem that generalizes Steinitz’s and Eppstein’s theorems for a broader class of polycube polyhedra with any genus, non-simply connected faces, but we also require that three edges meeting at every vertex (Note that there are polycubes whose vertices have more than three edges). Our theorem states that “a polycube graph (3.1) is a skeleton graph of a polycube polyhedron (3.2) if the patch number of the graph is bigger than 3.” Based on our framework, we further design a linear system to build a space of polycube polyhedra from the corresponding graph. Our approach is flexible to adjust the geometry of a polycube polyhedron. In summary, given a graph, our theorem addresses the following problem: 1) the existence of the polycube polyhedron; 2) the dimension of polycube shape space; 3) the method to generate the polycube polyhedron.

Polycube mesh (or polycube) plays an important role in graphics research. Polycube is firstly proposed in [THCM04] to extend cube mapping to general shapes. Polycube has a highly regular structure and a special global parametric domain. It is a special geometric shape that all face normals are aligned with one axis of a specific orthonormal coordinate frame. Polycube removes the detail of the original meshes and captures its global features. It has been used in many kinds of graphics applications, such as: surface texturing [THCM04], volumetric texturing [CL*10], parameterization [GZH*13], reconstruction [WJH*08], shape morphing [FJSF05] and T-mesh construction [LZLW15]. Many algorithms of hexahedral remeshing [GSZ11; HXH10; LVS*13; HJS*14; FBL16] also rely on polycube mesh heavily. Usually, they generate a corresponding polycube mesh from the original shape by deformation, the difficult hexahedral meshing operation is transferred onto the polycube mesh.

Currently, there are two major frameworks in quadrangulation and all-hex meshing: deformation based [FBL16; HJS*14; LVS*13; GSZ11] and frame-field based [NRP11; HTWB11; LLX*12; JHW*14; LBK16]. By the relationship between a graph and a polycube polyhedron, we provide a graph-based framework for the quadrangulation and all-hex meshing applications. Our framework works as follows: given an input graph embedded in a mesh, first, we build a polycube polyhedron, then we map the original mesh onto it to achieve a polycube mesh, which can be used in quadrangulation and all-hex meshing.

**Contribution.** Our major contribution is proving a theorem which characterizes the relationship between a graph and a polycube polyhedron, our theorem provides a necessary condition for the skeleton graph of a polycube polyhedron. Our second contribution is to design a linear system to realize a set of adjustable polycube polyhedra for a valid graph. Our third contribution is presenting a graph-based framework for the applications of polycube mesh generation, quadrangulation, and all-hex meshing, as demonstrated in Fig. 2.b.

The organization of the paper is: Section 2 summarizes the relevant works; the concepts are introduced in Section 3; polycube shape space is explained in Section 4; the embedding is detailed in Section 5; finally the applications of quadrangulation and all-hex meshing are illustrated in Section 6.

2. Related Works

The most related work to ours is Steinitz’s theorem [Zie12; GKKZ67] and Eppstein’s theorem [EM10]. Our polycube shape space is also motivated by the applications of polycube generation, mesh parameterization, quadrangulation, all-hex meshing. Because of the abundance of literature, we will focus only on the ones which are directly relevant to ours.

In two dimensions, a graph is a skeleton graph of a polytope if and only if it is a cycle. In three dimensions, Steinitz’s theorem addresses what kind of graph can be the skeleton graph of a convex polyhedron. The sufficient and necessary condition between a graph and a general non-convex polyhedron is still unknown. For convex polyhedra with all vertices on a sphere, the graph-theoretic characterizations are discussed in [DS96; HRS92; Riv96]. For non-convex polyhedra with star-shaped faces, and all but one face are visible from a common viewpoint, the characterizations are addressed in [Hon08].

Two special cases of simple orthogonal polyhedron : corner polyhedron and xyz polyhedron are investigated in [EM10] (see [EM10, Fig. 1]). If all but three faces of a simple orthogonal polyhedron are oriented towards the vector, it is called corner polyhedron. If each axis-parallel line of a simple orthogonal polyhedron has at most two vertices, it is called xyz polyhedron. A corner polyhedron is always an xyz polyhedron. Eppstein et al. [EM10] prove the following theorems:

**Theorem 2.1 (EM10)** A graph is the skeleton graph of a corner polyhedron if and only if it is planar, bipartite, 3-regular and its dual is 4-connected.

**Theorem 2.2 (EM10)** A graph is the skeleton graph of an xyz polyhedron if and only if it is planar, bipartite, 3-regular and 3-connected.
A simple orthogonal polyhedron can be divided into xyz polyhedrons, which furthermore is separated into corner polyhedrons. Their induction proof also gives an efficient linear time algorithm to construct the simple orthogonal polyhedron for a given graph. While Steinitz’s and Eppstein’s statements are remarkable, it is limited to a small class of polyhedra, our approach can address and construct more polyhedra.

In the graphics community, polycubes are constructed from not graphs, but meshes. The methods in [THCM04; YL08] construct polycube manually, then another extracting step is applied to achieve the cross-surface mapping between the mesh and its polycube shape [WJH*08]. The method in [LIFW08] uses segmentation and box-primitives to approximate original models, but it fails on complicated meshes. A divide-and-conquer algorithm is applied in [HWFQ09], however, it generates over-refined polycubes. Both of [HWFQ09; LIFW08] can not build the cross-surface mapping automatically.

The deformation based approaches in [GSZ11; HJS*14; FBL16] obtain polycube meshes by minimizing an energy. Given a surface and a valid polycube topology, Zhao et al. [ZLL*18] propose a novel method to deform a mesh into its polycube mesh by rotating face normals. The algorithm in [GSZ11] uses a rotation-driven method to align each surface normal to one direction of \((\pm X, \pm Y, \pm Z)\) gradually; then a position-driven deformation is used to enforce the planarity and straightness. Based on the observation that every face normal of a polycube is aligned with one axis and the value of the \(\ell_2\)-norm of each unit face normal is equal to 1, Huang et al. [HJS*14] propose a \(\ell_2\)-norm energy weighted by triangle areas and a variational method to generate polycube meshes. However, their system is nonlinear and the results are affected by the orientation of the mesh. They need another energy to find an optimal global orientation and a post-processing cleanup step to fix spurious topological degeneracy.

The graph-cut based approach [LVS*13] balances the parameterization distortion and the number of singularities of polycube vertex. Fu et al. [FBL16] propose another face normal-rotation method to deform a mesh into a polycube mesh. A method proposed in [CLS16] simplifies the complicated polycube topology. A special method that extends polycubes and parameterizes a mesh with more than one chart is proposed in [FXBHI16]. However, this method is designed for all-hex meshing directly, and can not produce a polycube mesh.

Frame-field based hexahedral mesh generation is applied in [HTWB11; JHW*14; LKB16], which has internal singularities and can not guarantee a valid singularity structure. Polycube based hexahedralization can produce a special kind of hexahedral mesh which has no internal singularity [GSZ11; HJS*14; LVS*13; FFXBHI16]. These mesh-based methods normally generate one single polycube mesh, instead, our graph-based one can obtain a set of results by embedding a graph on the mesh. Analyzing and optimizing the polycube structure by a graph is also studied in [CXW*19].

3. Background

3.1. Basic Concepts

In this paper, we distinguish the concepts of polycube polyhedron and polycube mesh. A polycube polyhedron is a polyhedron with planar orthogonal polygonal faces (Fig. 2e). A polycube mesh has the same overall shape with its corresponding polycube polyhedron, but every planar polygonal face is triangulated (Fig. 2f). We call a layout graph on a mesh as a polycube graph if it can lead to a valid polycube topology (Fig. 2b). If all patches of a mesh separated by a layout graph can be assigned a valid set of coordinate labels (as shown in Fig. 2c and 2e, the red, green and blue colors represent the labels of X, Y, Z axis respectively), we call this assignment a valid polycube topology.

Definition 3.1 A layout graph is called a polycube graph \(G = (N,A,P)\) with the nodes \(N\), the arcs \(A\) and the patches \(P\), if it satisfies two conditions: 1) each node \(n_i \in N\) is of valence 3, i.e. there are 3 arcs incident to it; 2) each patch \(p_i \in P\) has an even valence, i.e. there are even number of nodes along the border of each patch.

A polycube graph is 3-regular. A patch corresponds to a graph cycle. In this paper, we only discuss a special subset of all valid polycubes in general sense. For example, the red node of the polycube in Fig. 3a has valance 4, therefore this layout graph does not satisfy our requirements. However, we can always modify this kind of graph to be valid, then generate a similar polycube (Fig.3b). Note that if each patch is of valence 4, then it is called quad layout graph [CK14].

![Figure 3: (a) A polycube that violates our requirements; (b) modified valid version.](image)

Definition 3.2 A polycube polyhedron is a three-dimensional polyhedron (not necessarily convex) which has exactly three mutually-perpendicular axis-parallel edges meeting at every vertex.

The vertices, edges, faces of a polycube polyhedron correspond to the nodes, arcs, patches of a polycube graph. The surface normals of a polycube polyhedron are axis-aligned and each face is perpendicular to one coordinate axis. Note that a valid polycube graph may lead to an invalid polycube polyhedron. For example, in Fig. 4, the polycube graph is valid, however, the heights of the two red edges in the vertical direction are 0, the polyhedron is not valid.

Definition 3.3 If a polycube graph has an embedding in \(\mathbb{R}^3\) to form a polycube polyhedron, such an embedding is called a polycube embedding of the polycube graph.

When a polycube graph \(G\) is embedded in a mesh \(M\), each node \(n_i\), each arc \(a_i\) of \(G\) are mapped to a vertex \(v_i\) and a chain of edges \(c_i = \{e_1, e_2, \ldots, e_n\}\) of mesh \(M\) respectively (Fig. 5). A mesh can have many different embedded polycube graphs (Fig. 6).
3.2. The Characteristics of Polycube Polyhedra

A polycube polyhedron is also called an orthogonal polyhedron [LVS13; EM10]. It can have any kind of genus. The faces on a polycube polyhedron can be simply or non-simply connected. A simply connected face is of disk-topology, while a non-simply connected face has some holes inside it (Fig. 7).

For a polycube polyhedron of genus zero, the difference between the numbers of vertices with Gaussian curvature $+\frac{\pi}{2}$ and $-\frac{\pi}{2}$ is eight; for a polycube polyhedron of genus one, the two groups have the same number of vertices.

For a non-simply connected face of a polycube polyhedron, there are more than one arc loops in the corresponding polycube graph. If an inner arc loop consists of four arcs, then the Gaussian curvatures of the vertices on this inner arc loop are all $-\frac{\pi}{2}$; if an outer arc loop consists of four arcs, then the Gaussian curvatures of the vertices are all $+\frac{\pi}{2}$. For example, the white vertices ($-\frac{\pi}{2}$) and the black vertices ($+\frac{\pi}{2}$) in Fig. 10. In fact, all Gaussian curvatures of the vertices can be pre-determined in Fig. 10. However, a polycube graph corresponds to many different polycube polyhedra, its combinational structure cannot determine all Gaussian curvatures.

4. Polycube Shape Space

In this section, we present an algorithm to construct a vector space of polycube polyhedron from a polycube graph and prove the necessary condition for the skeleton graph of a polycube polyhedron.
4.1. Arc Colorization

We observe that a polycube polyhedron has two crucial properties: 1) the directions of all edges on one face \( f_i \) must switch between two axes of \( d_1, d_2 \); 2) the normal of the face \( f_i \) must point to the third axis of \( d_3 \). Based on these observations, we design an arc colorization algorithm to assign axis labels to all arcs of a polycube graph, such that we can determine the coordinate axes for the edges and the faces of its corresponding polycube polyhedron accordingly. Note that the color label on an arc indicates its axis, it can not distinguish the positive or the negative directions of the axis.

Let red, blue, green colors represent the \( X, Y, Z \) coordinates respectively. First, we start from a node of the polycube graph \( G \) and assign three colors to its three neighboring arcs; then our algorithm travels the whole polycube graph patch by patch with a breadth-first method and colorizes the arcs of every patch by switching between two corresponding colors. Some colorized polycube graphs are exhibited in Fig. 11. The pseudocode is shown in Alg. 1.

Three colors have six permutations, which leads to six different colorizations for a polycube graph. However, all of them are equivalent under rotation and reflection transformations. Any permutation can be our input for the next step.

4.2. Differential One-Form

A polycube embedding induces a natural vector valued differential one-form and differential 0-form on a polycube graph. First, we define an edge length function \( L \) on all arcs \( A \) of a polycube graph as: \( L : A \to \mathbb{R} \). Second we define a positive orientation on \( A; \ d : A \to \{e_1, e_2, e_3\} \), where \( e_1, e_2, e_3 \) is the standard basis of \( \mathbb{R}^3 \). Then we propose a vector-valued differential one-form on the polycube graph \( G \) as: \( \omega = L \cdot d \), where \( \cdot \) is the scalar product. The exterior derivative of \( \omega \) is a 2-form on the patch set \( P \): \( d\omega(p_i) = \sum_{\gamma \in \partial\gamma} \omega(\gamma) \) here \( \gamma \) is the boundary arcs of the patch \( p_i \). The integration of \( \omega \) on a directed arc chain \( \gamma \) is defined as: \( \int_\gamma \omega = \sum_{\gamma \in \partial\gamma} \omega(\gamma) \).

For a polycube polyhedron embedded in \( \mathbb{R}^3 \), the total sums of the edge length vectors on every loop of the polycube polyhedron are 0. It means the differential one-form \( \omega \) is closed and exact. A closed differential one-form \( \omega \) satisfies:

\[
\int_\gamma \omega = 0.
\]

The integration of an exact differential one-form on any closed loop is 0, it is equivalent to that the integration is 0 on every homology basis of the homology group \( H(G) \) of a polycube graph \( G \):

\[
\int_\gamma \omega = 0, \quad \forall \gamma \in H(G),
\]

Accordingly, we can compute a vector valued differential 0-form \( V_\omega \) on the nodes \( N \) of the polycube graph \( G \) by integrating this differential one-form, such that the differential 0-form \( V_\omega \) determines the vertex positions of the corresponding polycube polyhedron:

\[
V_\omega : N \to \mathbb{R}^3.
\]

We pick up any node \( n_0 \) as the root and set its position to be \((0,0,0)\), then the position of any other node \( n_i \) can be integrated as:

\[
V_\omega(n_i) = \int_{\gamma(n_0,n_i)} \omega,
\]

where \( \gamma(n_0,n_i) \) is an arc chain from \( n_0 \) to \( n_i \).

4.3. Linear System

The differential one-form \( \omega \) consists of two parts: \( L \) and \( d \), the axis of \( d \) of each arc is already computed by the colorization. We can
randomly choose a positive direction for \( d \) (Fig. 12). The variable \( L_t \) may be negative, which means that the corresponding edge has the opposite direction to this initial assignment.

\[
\text{Algorithm 2: Non-simply connected patches}
\]

\textbf{Input:} A polycube layout \( G = (V, A, P) \) with some non-simply connected faces

\textbf{Output:} New polycube layout \( G' = (V', A', P') \) without non-simply connected patches

\textbf{for} \( p_i \) in the set of non-simply connected faces \textbf{do}

1. Assume \( p_i \) is colorized by color \( c_1 \) and \( c_2 \). Find two arcs \( a_1, a_2 \) colored by \( c_1 \) in outer boundary of \( p \).
2. Get all inner holes of \( p_i \): \( h_1, h_2, \ldots, h_n \).

\textbf{for} \( i \leftarrow 1 \) to \( n \) \textbf{do}

3. Find an arc in \( h_i \) colorized by \( c_2 \). Assume its end vertices are \( v_1, v_2 \).
4. Add a vertex \( v_1 \) to \( a_1 \), \( v_2 \) to \( a_2 \).
5. Add an arc between \( v_1 \) and \( v_1 \), an arc between \( v_2 \) and \( v_2 \).

\textbf{end}

\textbf{return} New polycube graph \( G' \).

\[
\text{Graph} \quad (a) \quad \text{Loop1} \quad (b) \quad \text{Loop2} \quad (c) \quad \text{Loop3} \quad (d) \quad \text{Loop4}
\]

\[
\text{Figure 14: The four homology basis loops (red lines) of the "double torus" model.}
\]

\[
\text{Graph} \quad (a) \quad \text{Loop1} \quad (b) \quad \text{Loop2} \quad (d) \quad \text{Loop3} \quad (e) \quad \text{Loop4}
\]

\[
\text{Figure 15: The four homology basis loops (red lines) of the "sculpt" model are illustrated on the virtual polycubes.}
\]

\[
\text{4.4. Kernel Space}
\]

If the dimension of the kernel space \( K \) of the matrix \( W \) is zero, it means that there is no valid polycube polyhedron. We define the kernel space of the matrix \( W \) as the polycube shape space of the corresponding polycube graph. The basis of the kernel space forms the basis of the polycube shape space, as shown in Fig. 16. We generalize the Eppstein’s result [EM10] by the following theorem.

\textbf{Theorem 4.1} If a polycube graph can realize a polycube polyhedron embedded in \( \mathbb{R}^3 \), then the dimension of the corresponding kernel space \( K \) is not zero.

This is a necessary condition, some elements in the polycube shape space cannot lead to a valid polycube embedding.

We denote the numbers of nodes, arcs and patches of a polycube graph layout as \( |V| \), \( |A| \), \( |P| \). After are splitting, these three quantities become \( |V'| \), \( |A'| \), \( |P'| \). Assume there are \( k \) inner loops in all
non-simply connected patches. One observation is:

\[ \lvert N \rvert = \lvert N \rvert + 2k \]
\[ \lvert A \rvert = \lvert A \rvert + 4k \]
\[ \lvert P \rvert = \lvert P \rvert + k \]

(6)

We induce the following theorem for the dimension of polycube shape space. The experimental statistics are shown in Table 1.

**Theorem 4.2** For a polycube graph with the number of nodes \( \lvert N \rvert \), arcs \( \lvert A \rvert \), patches \( \lvert P \rvert \) respectively, the dimension of its polycube shape space is at least \( \lvert P \rvert - 3 \).

**Proof.** By Euler’s polyhedron formula,

\[ \lvert N \rvert = \lvert N \rvert + 2g \]

(7)

There are three arcs for every node of a polycube graph, so we have

\[ 3 \lvert N \rvert = 2 \lvert A \rvert \]

(8)

By Equation 6 and the two equations above, we get

\[ \lvert A \rvert = 3 \lvert P \rvert - 3k - 6 + 6g \]

(9)

Every patch (before adding auxiliary arcs) at most provides 2 equations (on the two axes the face spans respectively), of the form Equation 3. After the arc splitting step, each inner loop at most provides 1 equation (on the axis the two added arcs belong to). But there is one redundant equation on each axis. So there are at most \( 2 \lvert P \rvert + k - 3 \) independent equations provided by patches. Another part of equations comes from homology basis of the polycube graph, of the form Equation 4. There are 2g loops in the homology basis. Every loop spans all three axes, so all loops provide at most 6g independent equations. In total, there are at most \( 2 \lvert P \rvert + k - 3 + 6g \) independent equations in the linear system. The number of variables is \( \lvert A \rvert \). Thus, according to Equation 9 and Equation 6, the kernel dimension of the linear system would be \( \lvert P \rvert - 3 \). All shape spaces in our extensive experiments achieve exactly \( \lvert P \rvert - 3 \) dimensions.
5. Polycube Embedding

Let \( \Omega = \{ \omega^1, \omega^2, \ldots, \omega^n \} \) denote the basis of a polycube shape space. Every exact and closed differential one-form \( \omega \) can be expressed as a linear combination of the basis \( \omega^i \) as the following:

\[
\omega = \sum_{\omega^i \in \Omega} c_i \cdot \omega^i, \tag{10}
\]

where \( c_i \) is the weight coefficient of the linear combination.

Assume \( \omega \) has a valid polycube embedding, let \( V_0 \) denote the set of vertex positions of that embedding, then \( V_0 \) must be able to be realized in \( \mathbb{R}^3 \) space by integrating \( \omega \), i.e., a 0-form. Every basis \( \omega^i \) induces a 0-form \( V_0^i \), which has value \( V_0^i(v_j) \) on a vertex \( v_j \). We call \( V_0 = \{ V_0^1, V_0^2, \ldots, V_0^n \} \) as the basis of all 0-forms.

The 0-form \( V_0^i(v_j) \) can be expressed by a linear combination of the basis of \( V_0^i \):

\[
V_0^i(v_j) = \sum_{V_0^\omega(v_j)} c_i V_0^\omega(v_j) = \sum_{\omega^i \in \Omega} c_i \omega^i(v_j) = \sum_{\omega^i \in \Omega} c_i \int_{V_0(v_j)} \omega, \tag{11}
\]

where the root vertex \( v_0 \) has the position \((0,0,0)\).

Each element of the polycube shape space corresponds to a 0-form. But not all of them can be embedded in \( \mathbb{R}^3 \). If a polycube graph is embedded on a mesh and we look for an optimal polycube embedding to approximate the mesh, we propose to use the positions of the nodes of a polycube graph embedded on the mesh as the target position \( \bar{V} \), then we define a polycube energy \( E_P \) as the following quadratic format to find the optimal 0-form \( V_0^i \):

\[
E_P = (V_0 - \bar{V})^2 = (\sum_{\omega^i \in \Omega} c_i \int \omega - \bar{V})^2 \tag{12}
\]

The optimal solution \( V_0 \), i.e., the optimal polycube polyhedron, approximates the target \( \bar{V} \) closely, i.e., the original mesh, as shown in Fig. 17. There are six optimal solutions corresponding to six polycube embeddings.

5.1. Prescribed Constraints.

Our method has a special feature in generating the polycube polyhedron. We can add edge length constraints in the energy system, as shown in Fig. 18. Since we do not know the edge directions before embedding, we adjust the edge length values after embedding and reconstruct the polycube embedding. For example, after the first embedding, we get the length value of the arc \( k_l \) as \( L_i = l_i \), then we can add the following constraint to modify this edge length by scalar \( k \):

\[
L_i = k l_i, \tag{13}
\]

we can also change its direction:

\[
L_i = -l_i. \tag{14}
\]
here \( v_i, v_j \) are two vertice of face \( f_i, f_j \) respectively, and \( n_i \) is the unit normal vector of face \( f_i \). This equality means that the distance between two parallel faces \( f_1 \) and \( f_2 \) is zero.

The adjacent patches in a polycube graph are orthogonal in the polycube polyhedron. If there are two faces co-planar, they must be non-adjacent patches. We treat this case to be not valid. On the other hand, the purpose of this constraint is to constrain that one face should be on the desired side of the other face. This kind of constraints is added only when we find that the change of side can eliminate face intersections.

An optimal polycube polyhedron without degeneration (Fig. 20) can be constructed by solving a system of quadratic optimization with linear inequality constraints.

\[
\begin{align*}
\minimize E_p &= (V_o - \bar{V})^2 = \left( \sum_{w \in \Omega} c_i \cdot \int \omega_i - \bar{V} \right)^2 \\
\text{subject to} \\
L_i \geq \epsilon, \text{ or} \\
L_i \leq -\epsilon, \text{ or} \\
(V_{w}(v_i) - V_{w}(v_j)) \cdot n_i \geq \epsilon, \text{ or} \\
(V_{w}(v_i) - V_{w}(v_j)) \cdot n_i \leq -\epsilon
\end{align*}
\]

where \( \epsilon \) is the smallest allowed edge length.

Interactive user interface [LS07] is proved to be a useful tool in graphic applications. We also use an interactive user interface to set up the unknown inequality value by visualization of the results and update the degenerated polycube embedding to a valid one. Since each polycube graph already has an embedding in the original mesh, in our experiment, such adjustment rarely happens.

We demonstrate the applicability of our polycube embedding algorithm with dozens of varying meshes, some are exhibited in Fig. 21 and Fig. 22. Since our algorithm is based on a linear system, it is very fast. Our experiments show that our method is robust to the meshes of different genus and geometries.

6. Applications

In this section, we demonstrate that our framework of polycube shape space can be applied in the applications of polycube mesh, quadrangulation [BZK09; BLP*13] and all-hex meshing [HJS*14; FBL16].

6.1. Polycube Mesh

Different from previous deformation based methods, we propose a graph-based approach to create a bijective cross-mapping polycube mesh (Fig. 27) by parameterizing a mesh \( M \) onto the polycube polyhedron generated from an input polycube graph. We call it polycube parameterization (Fig. 23), which consists of three steps: 1)
the nodes of the polycube graph are mapped to the nodes of the polycube polyhedron; 2) the vertices on the arcs of the polycube graph are mapped onto the edges of the polycube polyhedron according to their length ratio (Fig. 23b); 3) every patch of the mesh separated by the polycube graph are parameterized onto the corresponding face of polycube polyhedron by some fixed-boundary bijective parameterization [FL16] (Fig. 23d).

![Figure 23: The triangles in the "orange" circle have flips.](image)

Every patch can apply different parameterization algorithms independently. If a patch has four arcs and is simply connected, we use Tutte’s embedding algorithm [Flo03; GGT06] to obtain a bijective result. When a patch has more than four arcs, it may not have a convex boundary and maybe non-simply connected. Therefore the Tutte’s method cannot guarantee a bijective result [FP06] (Fig. 23c), we apply the optimization methods in [FL16; AL13] to obtain a bijective mapping (Fig. 23d). It is possible to obtain a better bijective cross-mapping with lower distortion by adjusting the prescribed edge lengths to change the shape of the polycube polyhedron.

![Figure 24: The original triangular mesh (a), its bijective cross-mapping polycube mesh (b), the quadrangular polycube mesh (c), the final quad-meshing result (d).](image)

6.2. Quadrangulation

Polycube mesh can be used in quadrangulation. Given a mesh and its bijective cross-mapping polycube mesh, first we compute a quadrangular polycube mesh, second, we pull it back onto the original mesh by the barycentric coordinates, as exhibited in Fig. 24. More quad-meshing results are provided in Fig. 25.

![Figure 25: The quad meshing results.](image)

6.3. All-hex Meshing.

The all-hex meshing algorithms [GSZ11; HJS*14; FBL16] produce one single optimal tetrahedral polycube mesh by deforming a volume tetrahedral mesh to minimizing an energy. Our method uses a different work-flow: 1) prescribe a polycube graph on a surface mesh; 2) generate the tetrahedral volume mesh from a surface mesh; 3) compute the polycube mesh of the surface mesh; 4) map the tetrahedral mesh onto the volume of the polycube mesh; 5) tessellate the polycube mesh into a hexahedral polycube mesh; 6) pull back the hexahedral polycube mesh onto the original tetrahedral mesh with the barycentric coordinates. The work-flow is illustrated in Fig. 26.

The fourth step is a fixed boundary volume parameterization problem. We use the volumetric harmonic mapping [WGC*04] to get an initial result with flips, then an optimization method [FL16] can be applied to improve the volume mapping.

![Figure 26: The tetrahedral mesh (a); the cross-mapping tetrahedral polycube mesh (b); the hexahedral polycube mesh (c); the hex meshing result (d); the slices of the hexahedral mesh (e), (f).](image)
In this paper, we address the graph-theoretic characterizations of the skeleton graph of polycube polyhedron. We prove the necessary condition of the polycube graph. Our proof is based on a linear system that illustrates the exact and closed differential one-form. We conclude that if a polycube graph has more than 3 patches, i.e., circles, then it could be the skeleton graph of a polycube polyhedron. Our proof also leads to a polycube embedding algorithm naturally.

Although we focus on establishing the relationship between graphs and polyhedra to generalize the Steinitz’s theorem and Eppstein’s theorem in this paper, we also propose a novel graph-based framework for the applications of polycube mesh generations, quad and hex meshing. However, our framework requires a polycube graph as the input. It is potential to apply some layout graph generation algorithms [CK14; EGKT08; RRP15; RP17] to find a valid graph automatically and a better embedding, which follows the feature lines of the mesh, to achieve a better result.

Acknowledgments

We would like to thank anonymous reviewers for their insightful feedback, valuable comments, and suggestions. Some pictures are rendered by Mitsuba [Jak10]. Some 3D models are from the AIM@SHAPE shape repository and Thingi10K repository. This work is partially supported by the National Key R&D Program of China under Grant No.: 2017YFB1002701. The project is also partially supported by National Natural Science Foundation of China (Grant No. 61661146002, 61772105, 61720106005, 61432003); NSF (Grant No. 1762287, 1418255, 1737812).

Figure 27: The polycube meshes: the first row shows the original models; the second row exhibits the Tutte’s mappings with flips (red parts), the flipped triangles are located on the reflect corners [GGT06] of the patches; the third row demonstrates the bijective results.

7. Conclusion and Future Work

In this paper, we address the graph-theoretic characterizations of the skeleton graph of polycube polyhedron of high-genus and non-simply connected faces. We prove the necessary condition of the polycube graph automatically and a better embedding, which follows the feature lines of the mesh, to achieve a better result.

References


Hai Zhao et al. / Polycube Shape Space


