DOI: 10.1111/cgf.12171
Eurographics Symposium on Geometry Processing 2013
Yaron Lipman and Hao Zhang
(Guest Editors)

*Volume 32 (2013), Number 5*

# Connectivity Editing for Quad-Dominant Meshes

Chi-Han Peng[†1] and Peter Wonka[‡1,2]

[1]Arizona State University, USA
[2]King Abdullah University of Science and Technology, Saudi Arabia

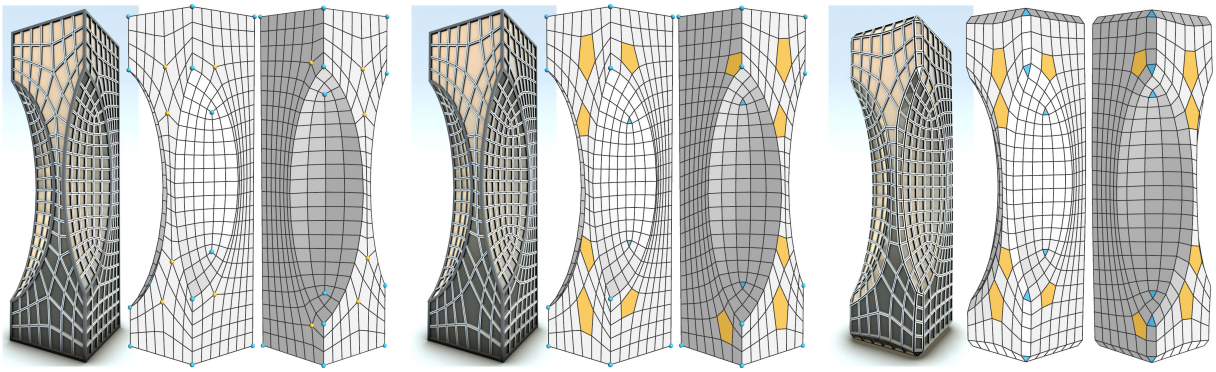


Figure 1: In our framework, the user can edit irregular vertices (with more or fewer than four neighbors) and irregular faces (non-quads) of a quad-dominant mesh. Each type of irregularity has different advantages and disadvantages in design and construction. Irregular vertices are necessary to maintain sharp features (corners and edges), but they create higher angle deviations in mesh lines in smooth regions (left). Irregular faces lead to smoother mesh lines, but they cannot maintain sharp features (right). The ability to model with a mixture of irregular vertices and faces gives more flexibility to the user, e.g. creating a design with sharp features and smooth mesh lines (middle). We render each model in a style that highlights the sharp features.

**Abstract**

*We propose a connectivity editing framework for quad-dominant meshes. In our framework, the user can edit the mesh connectivity to control the location, type, and number of irregular vertices (with more or fewer than four neighbors) and irregular faces (non-quads). We provide a theoretical analysis of the problem, discuss what edits are possible and impossible, and describe how to implement an editing framework that realizes all possible editing operations. In the results, we show example edits and illustrate the advantages and disadvantages of different strategies for quad-dominant mesh design.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling/Geometric algorithms, languages, and systems—

## 1. Introduction

We propose an editing framework for the connectivity of arbitrary two-manifold quad-dominant (*QD*) meshes with the ability to explicitly control the location, type, and number of the irregular vertices (with more or fewer than four neighbors) and faces (non-quads) in the mesh. In the primal domain, the large number of combinations of different irregular elements makes connectivity analysis difficult. Therefore, we propose to edit *QD* meshes in an alternative pure quad mesh domain building on existing work in quad mesh

† peng.chi-han@asu.edu
‡ pwonka@gmail.com

connectivity editing [TPC*10, BLK11, PZKW11]. We are able to answer the following fundamental questions about the connectivity of *QD* meshes: what is the discrete Gauss-Bonnet theorem for *QD* meshes? In particular, can irregular vertices and faces be counted in the same way? What editing operations are fundamentally possible and impossible for *QD* meshes? Can we create, delete, or move a single irregular element? How can we interchange irregular faces and vertices? What can we do with T-junctions? In what circumstances is being quad-dominant, i.e. having non-quad faces, preferable to being pure quad for remeshing a surface? We then identify the simplest possible editing operations for *QD* meshes in the sense that they affect the smallest possible regions and involve the fewest irregular elements, which are moving irregular elements in pairs.

To evidence that the ability to edit irregular elements in a *QD* mesh is valuable, we compare different ways to *QD* remesh a given surface, e.g. having only irregular vertices, only irregular faces, or both, in terms of geometric properties. We demonstrate that the users can use our editing framework to alter the connectivity of existing *QD* meshes with ease.

### 1.1. Related Work

**Quad-Dominant Remeshing:** One source of *QD* meshes is quad-dominant remeshing. There is a large variety of algorithms that are often driven by curvature [ACSD*03, MK04, RLL*06]. Other approaches work by vertex alignment [TC06, LKH08], space partitioning (octree [Mar09] or quadtree [FM98]), or use segmentation as an important ingredient [MK06]. The generated meshes can also satisfy additional requirements, e.g. remeshing with planar quads [ZSW10]. Typically irregular faces are considered abnormal and are avoided whenever possible. One exception is [PP11] in which the authors argue that irregular faces can help preserve the flow of mesh elements. Several field-based quad meshing methods [KNP07, BZK09] include tools to edit the field singularities that can indirectly control the irregular elements in the resulting meshes. However, controlling singularities and irregular elements have different degrees of freedom, see [PZ07, PZKW11].

**Subdivision and Spline Surfaces:** Our insights stem from analyzing quad meshes in a subdivision domain. Catmull-Clark (*CC*) subdivision [CC78] can convert any mesh to a finer quad mesh at the cost of increasing the number of faces by a factor of about four. Inverse *CC* subdivision schemes have been studied [Tau02, SS11], but only a subset of quad meshes is directly inversely subdivisible. Alternatively, a Catmull-Clark square root ($\sqrt{CC}$) subdivision [Kob96], which is called a quad-graph by Bobenko et al. [BHS06], also produces quad meshes and is found to be inversely subdivisible if the mesh has no boundaries and a 2-coloring is given [Tau02]. We build on this concept and further state that inverse subdivisibility applies to quad meshes with a sphere-like or disk-like topology (Proposition 3.1). Related work about subdivision surfaces and the underlying mesh connectivity include an adaptive *CC* scheme proposed by Panozzo et al. [PP11], surface fitting from arbitrary topologies [SL03, MNP08], and T-splines [SCF*04].

**Quad Mesh Connectivity Transformations:** One fundamental operation to alter the connectivity of quad meshes is region-based requadrangulation, e.g. [NSY09]. These types of requandragulations are not suitable for editing because they provide only one out of many solutions. In [TPC*10] a set of local operations for editing pure quad meshes is presented in a systematic way. An interesting idea for mesh editing is to use operations on quad strips [DSSC08]. They have been extended to form GP operators [BLK11] which can operate on non-aligned quad strips. An alternative editing framework with the ability to explicitly control the irregular vertices is presented by Peng et al. [PZKW11]. We build on these recent papers in our work.

### 1.2. Contributions

The main contribution of this paper is to bridge the complicated task of *QD* mesh connectivity editing to the existing work in pure quad mesh editing via a suitable alternative domain. Not only can fundamental questions be answered in a simple and consistent way, but the implementation of the editing operations is also greatly simplified. Second, we are the first (to our knowledge) to establish theories for T-junction editing. We state that a T-junction can be moved in exactly four directions and provide the exact conditions for a pair of T-junctions to be cancellable. We further propose operations that enable users to switch the types of irregular elements (vertices or faces) in a theoretically minimal way (in pairs). To our knowledge, no comparable operation exists in previous work.

Overall, we believe that our findings will have high practical value to the 3D mesh modeling communities, e.g. game and architectural design. In many cases the shape is modeled first and then a low-resolution *QD* mesh is laid out manually. However, the task becomes non-trivial when irregular elements are involved. This paper endeavors to establish a theoretical foundation for such modeling tasks.

## 2. Overview

We organize the paper as follows. In Section 3.1, we review the $\sqrt{CC}$ domain of *QD* meshes and argue why it is a suitable platform for analyzing the connectivity of *QD* meshes. In Section 3.2, we formulate the generalized version of the discrete Gauss-Bonnet theorem for *QD* meshes, which can be used to predict the number of irregular elements in an arbitrary *QD* mesh given the boundaries and the Euler characteristic. In Section 3.3, we analyze what kinds of editing operations are impossible and possible for *QD* meshes. Backed

by the aforementioned theoretical analysis, in Section 4, we propose a connectivity editing framework for *QD* meshes. In Section 5, we compare various ways of *QD* mesh design for a given surface in terms of geometric properties, and show examples of *QD* mesh connectivity edits.

### 2.1. Basic Definitions

We limit our discussion to two-manifold *QD* meshes. The valence of a vertex $v$, which we denote as $l(v)$, is the number of edges in the mesh incident to $v$. A vertex with valence $n$ is denoted as $vn$, e.g. a $v3$ and a $v5$. A vertex with valence 4 is considered as *regular*, otherwise it is *irregular*. The degree of a face $f$, which we denote as $d(f)$, is $f$'s number of vertices. A face with degree $d$ is denoted as $fd$, e.g. a triangle ($f3$), a quad ($f4$), and a pentagon ($f5$). A face with degree four is considered as *regular*, otherwise it is *irregular*. We introduce an *index* function to measure irregularity. We define the index as $4-n$ for a $vn$ and $4-d$ for an $fd$.

To simplify our discussion, we consider irregular vertices with valences lower than 3 or higher than 5 as multiple $v3$ or $v5$ collocated together and therefore count them as multiple irregular vertices. Similarly, irregular faces with degree lower than 3 or higher than 5 are considered as multiple $f3$ or $f5$ collocated together and counted as multiple irregular faces. We also use the following definitions:

**Definition 2.1** A *path* $\gamma$ on a *QD* mesh $M$ consists of a sequence of edges $e_i = (v_i, v_{i+1})$ for $0 \leq i < N$, where $N$ is the *length* of $\gamma$. A path is a *loop* if $v_0 = v_N$. We assume that $\gamma$ is non-degenerate, i.e. there is no vertex in $\gamma$ that is incident to at least three edges in $\gamma$.

**Definition 2.2** A *region* $R$ on a *QD* mesh $M$ is a connected subset of the faces in $M$ (Figure 2b). We assume that $R$ has a single boundary unless otherwise specified. The boundary of $R$, denoted by $\partial R$, is a loop. The *index* of a boundary vertex $v$ is $1-e(v)$ where $e(v)$ denotes the *internal valence* of $v$, i.e. the number of $v$'s adjacent edges connecting to internal vertices. Intuitively, we denote a boundary vertex as a *non-corner* if its index is zero, a *convex corner* if its index is 1, and a *concave corner* if its index is negative. A region is *convex* if it has no concave corners, otherwise it is *concave*. A *side* of a region $R$ is a sequence of edges of $\partial R$ between two corners.

In our context, a region's boundary configuration is often specified as a sequence of vertices together with their internal valences. Note that the indices of irregular vertices, irregular faces, and boundary vertices are related to the Gaussian curvature (for irregular elements) and geodesic curvature (for boundary vertices) of their suitably remeshed neighborhoods.

## 3. Theoretical Analysis

We describe the major insights for connectivity editing of *QD* meshes in the following.

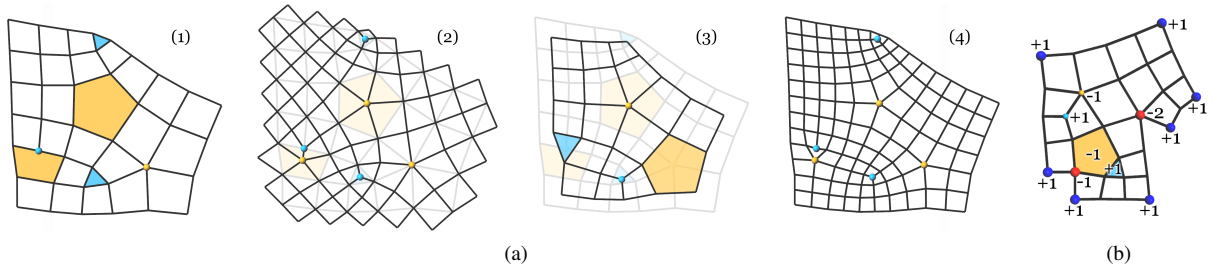
### 3.1. $\sqrt{CC}$ Domain

Analyzing connectivity editing for *QD* meshes in their primal domain would be too complicated because all possible combinations of irregular faces and vertices need to be considered. For example, the triple combinations of $v3$, $v5$, $f3$, and $f5$ amount to 20 possibilities, as compared to 4 in a quad mesh with just $v3$ and $v5$. An alternative domain where both irregular vertices and faces are mapped to elements of the same type is thus desired.

A key insight of our work is the selection of a suitable domain for *QD* mesh connectivity editing. We prefer domains where meshes are pure quad so that existing quad mesh editing work can be directly applied. Therefore, the dual, Doo-Sabin [DS78], and $\sqrt{3}$-subdivision [Kob00] are infeasible. One reasonable choice may be the *CC* domain in which subdivided meshes are pure quad and both irregular vertices and faces in the primal domain are mapped to irregular vertices of the same degree. However, only a subset of quad meshes is inversely *CC* subdivisible, e.g. irregular vertices must have graph distances of at least two, which means that it is possible to edit a *CC* subdivided mesh to make it no longer inversely subdivisible. This fact again makes analysis difficult. It turns out that the $\sqrt{CC}$ domain of a *QD* mesh [Kob96] is a suitable platform for our analysis. We review key properties of the $\sqrt{CC}$ domain as follows.

A $\sqrt{CC}$ subdivision can be understood as a half step of achieving the connectivity of a full *CC* subdivision (Figure 2a). Every primal vertex and face is mapped to a $\sqrt{CC}$ vertex. The former is denoted as a *p-vertex* and the latter is denoted as a *d-vertex* (corresponding to dual vertices of the primal mesh). We position the p-vertices at the same locations of their corresponding primal vertices and the d-vertices at the centers of the corresponding primal faces for visualization purposes. Every p-vertex is connected to the d-vertices of its corresponding primal vertex's adjacent faces and every d-vertex is connected to the p-vertices of its corresponding primal face's adjacent vertices. For meshes with boundaries we add an imaginary adjacent boundary face to each primal boundary edge. In this manner every primal edge, non-boundary or boundary, would have two adjacent faces and would be mapped to a $\sqrt{CC}$ face. Any $\sqrt{CC}$ domain mesh is thus pure quad since every $\sqrt{CC}$ face has four adjacent vertices (two p-vertices and two d-vertices). Furthermore, the $\sqrt{CC}$ vertices have the same degree as their corresponding primal vertices or faces. It is straightforward to see that every *QD* mesh has exactly one corresponding $\sqrt{CC}$ mesh. Another advantage of a $\sqrt{CC}$ subdivision is that it increases the number of faces by a factor of about two (converting every primal edge to a $\sqrt{CC}$ face), as compared with a factor of about four by a *CC* subdivision.

**Inverse $\sqrt{CC}$ Subdivision:** As noted by Taubin [Tau02], an inverse $\sqrt{CC}$ subdivision to recover the primal mesh can be done by inserting a diagonal connecting the two p-vertices for each $\sqrt{CC}$ face (and recover the dual by insert-




(a) (b)

Figure 2: (a) A *QD* mesh with a single boundary loop in its (1) primal domain, (2) $\sqrt{CC}$ domain, (3) dual domain, and (4) *CC* domain. $v3$ and $f3$ are drawn in blue. $v5$ and $f5$ are drawn in yellow. Note that the mapping from (1) to (2) and from (2) to (4) can be done by a $\sqrt{CC}$ subdivision, the mapping from (2) to (1) can be done by an inverse $\sqrt{CC}$ subdivision, and the mapping from (2) to (3) can be done by an alternative inverse $\sqrt{CC}$ subdivision with the p-vertices and d-vertices switched (dangling vertices are omitted). Note that the T-junction in the bottom left corner is mapped to an adjacent $v3$-$v5$ pair in the $\sqrt{CC}$ domain. (b) A region in a *QD* mesh. Convex corners are shown in blue and concave corners are shown in red. We denote the indices of the boundary vertices and irregular elements. Note that the discrete Gauss-Bonnet theorem (Equation 1) is satisfied.

ing diagonals connecting two d-vertices) for quad meshes that are 2-colored (as p-vertices and d-vertices). Note that for a mesh with boundaries it is possible that the recovered dual mesh has valence 1, i.e. dangling, vertices corresponding to the valence-2 boundary $\sqrt{CC}$ vertices. The following proposition describes the feasibility of inverse $\sqrt{CC}$ subdivision of quad meshes:

**Proposition 3.1** A region in a quad mesh with a sphere-like or disk-like topology (with at most one boundary and no handles) can be inversely $\sqrt{CC}$ subdivided in exactly two ways.

*Proof* It is known [Har69] that a graph can be 2-colored, i.e. is bipartite, if and only if it has no odd graph cycles. Any graph cycle in such a region is the boundary of a quadrangulation with a disk-like topology thus has an even length (a property of pure quad meshes). It is straightforward to see that the 2-coloring for a 2-colorable graph is unique (up to switching of all colors). By considering the vertices with the first color as either p-vertices or d-vertices we have two ways to do inverse $\sqrt{CC}$ subdivision. □

Conversely, a region with more than one boundary or with handles may be inversely $\sqrt{CC}$ subdivisible or not. Proposition 3.1 implies that we can freely edit a $\sqrt{CC}$ mesh and the edited mesh is still inversely $\sqrt{CC}$ subdivisible as long as the changes are contained in a disk-like region. Furthermore, the retrieved primal domain meshes are consistent if we fix the coloring of a fixed vertex in the $\sqrt{CC}$ domain by the following proposition:

**Proposition 3.2** An inverse $\sqrt{CC}$ subdivision applied to a quad mesh region with at most one boundary converts two $\sqrt{CC}$ vertices to be of the same type (primal vertex or face) if and only if their graph distance is even.

The proof is straightforward by inspecting the 2-coloring of a shortest path between the two $\sqrt{CC}$ vertices and is omitted here.

### 3.2. Discrete Gauss-Bonnet Theorem for Quad-Dominant Meshes

We formulate the discrete Gauss-Bonnet theorem for an arbitrary *QD* mesh $M$ as follows:

$$\sum_{v\in\partial M}(1-e(v))+\sum_{v\in intM}(4-l(v))+\sum_{f\in M}(4-d(f))=4\chi(M), \tag{1}$$

where $\chi(r)$ denotes the Euler characteristic of $M$. Recall that $l(v)$ denotes vertex $v$'s valence, $e(v)$ denotes the number of $v$'s adjacent edges connecting to internal vertices, and $d(f)$ denotes face $f$'s number of vertices.

Equation 1 implies that for an arbitrary *QD* mesh (including a region) the sum of the indices of the boundary vertices plus the sum of the indices of the irregular vertices and faces equals a constant solely determined by its Euler characteristic. It is useful for determining the minimal number of irregular elements in a *QD* mesh with known boundaries and Euler characteristic and vice versa. For example, any connectivity edits would not change the sum of the indices of the irregular elements within a *QD* region with a fixed boundary.

Equation 1 can be proved in many ways, e.g. by analyzing the *CC* domain of $M$ or summing the angle defects of the irregular elements and boundary vertices. Here we provide a proof that demonstrates that Equation 1 is a pure combinatorial fact directly derived from the Euler characteristic.

*Proof* The Euler characteristic of $M$ states that $V-E+F=\chi(M)$, where $V$, $E$, and $F$ respectively denote the number of vertices, edges, and faces in $M$. $V$ equals $V_b+V_i$ where $V_b$ denotes the number of boundary vertices and $V_i$ denotes the number of internal vertices. $E$ equals $E_{bb}+E_{bi}+E_{ii}$ where $E_{bb}$ denotes the number of boundary edges that are incident to two boundary vertices, $E_{bi}$ denotes the number of boundary-internal edges that are incident to one bound-

ary vertex and one internal vertex, and $E_{ii}$ denotes the number of internal edges that are incident to two internal vertices. The Euler characteristic of $M$ can then be rewritten as $4V_b + 4V_i - 4E_{bb} - 4E_{bi} - 4E_{ii} + 4F = 4\chi M$, which can be reformulated as:

$$\begin{aligned}(V_b - E_{bi}) +\\ (3V_b - 3E_{bb}) +\\ (4V_i - 2E_{ii} - E_{bi}) +\\ (4F - 2E_{ii} - 2E_{bi} - E_{bb}) = 4\chi M.\end{aligned}$$

The first part $(V_b - E_{bi})$ equals $\sum_{v\in\partial R}(1 - e(v))$. The second part $(3V_b - 3E_{bb})$ equals zero since the number of boundary vertices and the number of boundary edges are the same. The third part $(4V_i - 2E_{ii} - E_{bi})$ equals $\sum_{v\in intM}(4 - l(v))$ since, by summing the valences of all internal vertices, we count each internal edge twice and each boundary-internal edge once. The fourth part $(4F - 2E_{ii} - 2E_{bi} - E_{bb})$ equals $\sum_{f\in M}(4 - d(f))$ since by summing the degrees of all faces, we count each internal and boundary-internal edge twice and each boundary edge once. □

### 3.3. Fundamental Editing Operations

In this section, we describe what editing operations are fundamentally impossible and possible for irregular elements in a *QD* mesh. In general, our findings stem from analyzing the $\sqrt{CC}$ domain of *QD* meshes to which theorems about quad mesh editing can be applied.

To begin with, we note that by Proposition 3.2, moving a single irregular element without type-changing implies that the corresponding $\sqrt{CC}$ irregular vertex's graph distance to another vertex with fixed labeling is changed by an even number, while type-changing an irregular element implies that the graph distance is changed by an odd number.

**Proposition 3.3** It is impossible to create, delete, move, or type-change a single irregular element in an otherwise regular *QD* region $R$.

*Proof* Being able to do so implies that we can produce another *QD* region $R'$ with the same boundary but with a single created, deleted, moved, or type-changed irregular element. $R$ and $R'$'s $\sqrt{CC}$ domains are two quad meshes with the same boundary but with a created, deleted, or moved single irregular vertex. This contradicts Theorem 7.1 of [PZKW11] (with an imaginary convex quad mesh region extended from $R'$'s $\sqrt{CC}$ domain mesh). □

Because editing a single irregular element is impossible, we are interested in editing irregular elements in pairs.

**Proposition 3.4** Two irregular elements can be moved together in an otherwise regular *QD* region. Irregular elements of the opposite indices, e.g. an $f3$-$v5$ pair, translate in the same direction. Irregular elements of the same index, e.g. an $f5$-$f5$ pair, rotate and scale around a fixed point in the middle between the two irregular elements. Furthermore, the irregular elements change types at each step. See Figure 4a and 4b for illustrations.

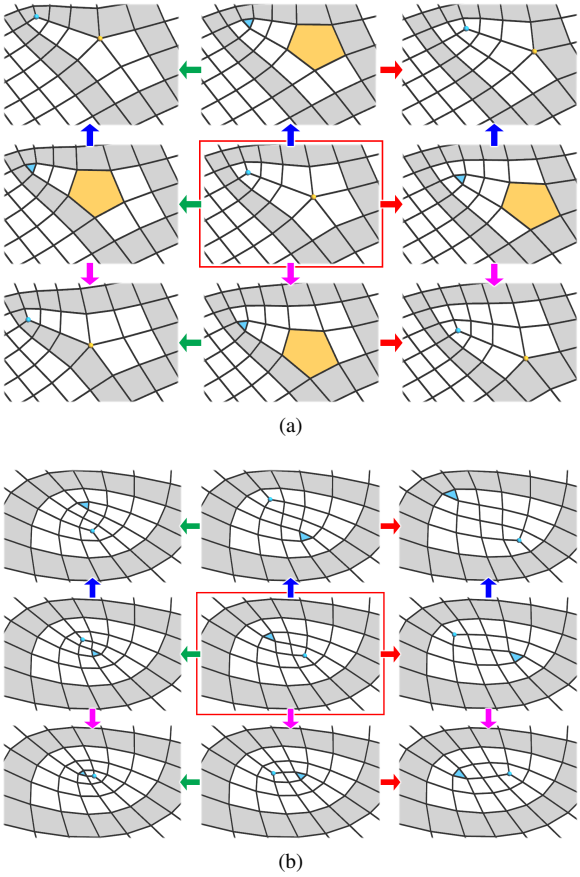


Figure 4: (a) A $v3$-$v5$ pair can be moved in the left (green arrows), right (red arrows), up (blue arrows), and down (purple arrows) direction. (b) A $v3$-$f3$ pair can be moved closer (green arrows), farther apart (red arrows), rotating clockwise (blue arrows), and rotating counter-clockwise (purple arrows). Note that a single step would switch their types, thus a type-preserving movement can be realized by two consecutive steps. The gray faces are marked for ease of inspection.

*Proof* A sketch of the proof is as follows. Two irregular elements can move in the same way as two irregular vertices in the corresponding $\sqrt{CC}$ mesh as described in Theorem 7.2, 7.3, and 7.4 of [PZKW11]. Regarding the type-changing behavior, a single movement in the $\sqrt{CC}$ mesh would change both the irregular vertices' graph distances to another vertex with a fixed labeling by one, thus changing the types of their corresponding primal elements. □

**Proposition 3.5** An irregular element pair can be merged

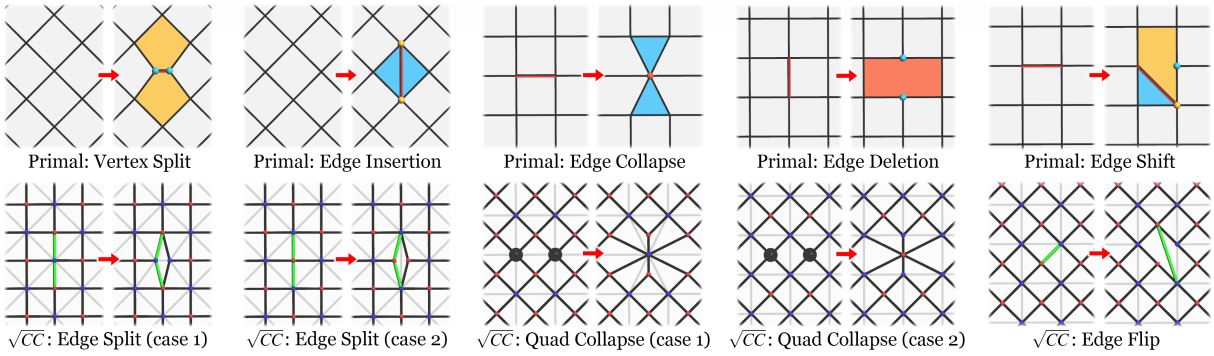



Figure 3: The five basic editing operations for *QD* meshes and their corresponding operations in the $\sqrt{CC}$ domain. Both a vertex split and an edge insertion add a primal edge. They are equivalent to an edge split of a $\sqrt{CC}$ edge pair with a p-vertex (blue) and a d-vertex (red) in between, respectively. Both an edge collapse and an edge deletion delete a primal edge. They are equivalent to a quad collapse of a d-vertices pair and a p-vertices pair, respectively. An edge shift shifts a primal edge toward one of its adjacent faces. It is equivalent to an edge flip in the $\sqrt{CC}$ domain. The edited meshes are not smoothed.

into a single irregular element with a higher absolute index, e.g. two $v3$ can be merged into a single $v2$ and two $f5$ can be merged into a single $v6$, if and only if they have the same type and same index.

*Proof* An irregular element pair of the opposite indices translates in the same direction thus cannot be merged. For a pair with the same index and different types, e.g. a $v3$-$f3$ pair, the graph distance between their corresponding $\sqrt{CC}$ irregular vertices is odd by Proposition 3.2, thus cannot be merged into a single irregular vertex by Theorem 7.2 and 7.3 of [PZKW11]. Otherwise it can be merged. □

### 3.4. T-Junction Editing

T-junctions, i.e. adjacent $v3$-$f5$ pairs, are often of special interest in *QD* remeshing. We describe how a single T-junction can be moved and how a pair of T-junctions can be canceled as follows.

**Proposition 3.6** A T-junction can be moved in exactly four directions denoted as up, down, left, and right.

*Proof* A T-junction can be moved exactly in the same way as the corresponding adjacent $v3$-$v5$ pair (the $v3$ is a p-vertex and the $v5$ is a d-vertex) in the $\sqrt{CC}$ domain. The left and right directions are both realized by applying an edge split followed by a quad collapse. The up direction is realized by two consecutive quad collapses. The down direction is realized by two consecutive edge splits. There are no other combinations. Note that applying a single step would switch the T-junction to be an $f3$-$v5$ pair. See Figure 5 for an illustration. □

**Proposition 3.7** A pair of T-junctions in an otherwise regular convex region $R$ can be completely canceled if and only if $R$ is a parallelogram, i.e. a 4-sided region with both pairs of opposite sides of the same graph length. Otherwise it can be reduced to be a single irregular element pair of opposite indices and of the same type, e.g. a $v3$-$v5$ or an $f3$-$f5$ pair.

*Proof* We first note that $R$ must have 4 sides since it is convex and has a sum of indices of irregular elements of 0. If $R$ is a parallelogram, it is straightforward to see that it can be remeshed as a regular grid, thus canceling the pair of T-junctions. Otherwise we perform a triple cancellation of irregular vertices in the $\sqrt{CC}$ domain to cancel one $v3$-$v5$ pair against the $v3$ or $v5$ of the second pair. In either case the result is a single $v3$-$v5$ pair, of which the graph distance is two, thus having the same type. □

Based on Proposition 3.7, we can identify four possible configurations (in terms of relative orientations) of a T-junction pair. These configurations including their cancellations are shown in Figure 6.

## 4. Editing Framework

To make the user interface intuitive, we design our editing framework such that edits can be done directly in the primal domain. However (as stated in Section 3.1), implementing editing operations in a *QD* mesh directly would be very complicated since the operations need to handle the large number of combinations of both kinds of irregular elements. Our proposed solution is as follows. First, user inputs in the primal domain are mapped to the $\sqrt{CC}$ domain. Actual edits are then carried out in the $\sqrt{CC}$ domain by building on existing work in quad mesh connectivity editing. Finally, the edited *QD* mesh is obtained by an inverse $\sqrt{CC}$ subdivision. In our prototype we also allow editing in the $\sqrt{CC}$ domain to enable easier analysis of the underlying concepts. The user can edit using the operations described in the following.

**Basic Operations:** For analysis purposes and full flexibility, we identify five basic operations for *QD* meshes that

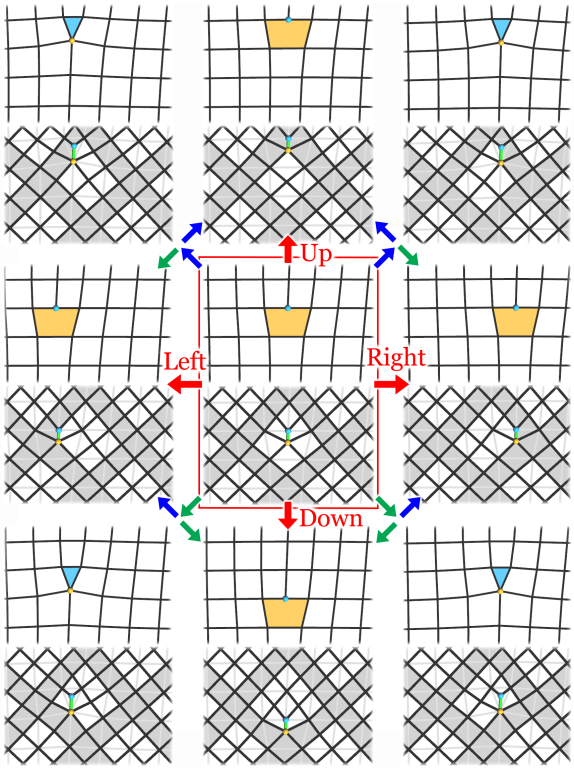



Figure 5: The four possible movement directions for a T-junction (red arrows). Each direction is realized by applying two consecutive atomic basic operations (quad collapses, blue arrows, and edge splits, green arrows) to the corresponding $v3$-$v5$ pair in the $\sqrt{CC}$ domain. The gray faces are marked for ease of inspection.

operate on a per-edge level (Figure 3): a *vertex split* and an *edge insertion* both add a single edge, an *edge collapse* and an *edge deletion* both delete a single edge, and an *edge shift* shifts an edge toward one of its adjacent faces. They all can be realized by applying three basic operations for quad meshes (edge split, quad collapse, and edge flip) in the $\sqrt{CC}$ domain, thus eliminating the need for implementing each of them explicitly. Edge splits and quad collapses can each map to two different operations in the primal domain. The difference arises because the vertices in the $\sqrt{CC}$ domain have two different labels (p-vertices or d-vertices), resulting in two different outcomes of inverse $\sqrt{CC}$ subdivisions. We point out that an edge collapse and an edge shift is respectively equivalent to a collapse and a shift step of the GP operators proposed by Bommes et al. [BLK11].

**Pair-wise Movement:** The user can move two irregular elements of arbitrary indices and types, e.g. a $v3$-$v3$, a $v3$-$f5$, and an $f5$-$f5$, in four possible directions. Each of the four possible movement directions is visualized by a pair of arrows with the same color in both the primal and $\sqrt{CC}$ domains (Figure 7). The user can select one moving direction for one irregular element and the movement of the other irregular element is constrained.

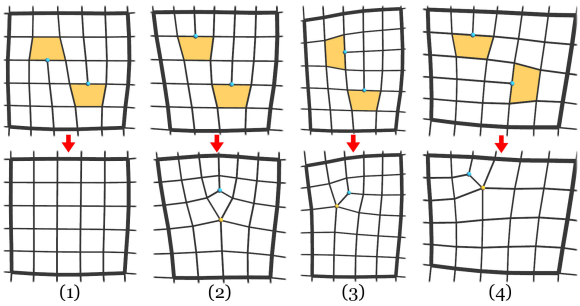



Figure 6: How a T-junction pair can be canceled in the four possible relative orientations (up to rotational symmetry) within an otherwise regular (4-sided) region $R$. Two T-junctions can be completely canceled if and only if they are facing the opposite sides of $R$ (1). Otherwise they are either facing the same side (2) or two adjacent sides (3,4) and can be reduced to an irregular element pair of the same type.

With this pair-wise movement operation, it is possible to do triple cancellations (e.g. an $f3$-$f5$-$v5$ to a single $v5$) by selecting a pair of irregular elements and colliding one element with a third element of an opposite index (an illustration in shown in the additional materials). Four irregular elements can be canceled at once when both elements of an irregular element pair collide with other elements of the opposite indices simultaneously. Cancellation can be done automatically, by computing the best movement path using a shortest path algorithm. The shortest path algorithm should include topological as well as geometric cost terms (e.g. curvature) and is therefore only a heuristic. Therefore, the manual selection of the movement path needs to remain as an important option.

**Type-Change:** As mentioned in Section 3.3, an odd number of pair-wise movements in the $\sqrt{CC}$ domain would not only move but also change the types of both irregular elements, e.g. a $v3$-$f5$ to an $f3$-$v5$ and a $v3$-$v3$ to an $f3$-$f3$. This is useful for users to change the types of irregular elements without introducing additional ones.

**Single Conversion:** The user can also change the type of a single irregular element at the cost of introducing an adjacent irregular element pair of opposite indices, e.g. an $f5$ to a $v5$ plus an $f3$-$v5$ pair or an $f3$ to a $v3$ plus an $v3$-$f5$ pair (Figure 8). Alternatively, it can be viewed as changing the sign of a single irregular element's index and introducing an adjacent irregular element pair of the same index. It is realized by applying a $v3$/$v5$ movement and $v3$-$v5$ pair generation operation in the $\sqrt{CC}$ domain.

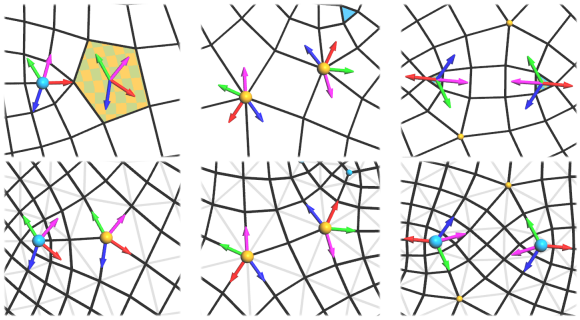


Figure 7: Visualization of the four possible movement directions of a pair of irregular elements (left: a $v3$-$f5$ pair, middle: a $v5$-$v5$ pair, right: an $f3$-$f3$ pair) in both the primal (top) and $\sqrt{CC}$ (bottom) domains.

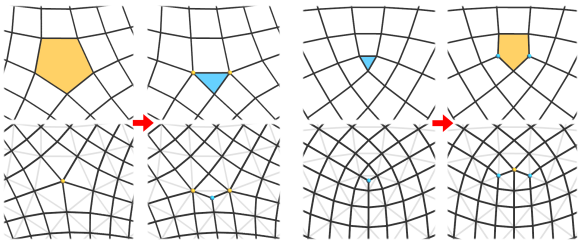


Figure 8: Left: an $f5$ is converted to an $f3$-$v5$-$v5$ triple. It can be viewed as type-changing the $f5$ to a $v5$ or changing the sign of its index (an $f5$ to an $f3$). An adjacent irregular element pair is introduced. Right: a similar operation where an $f3$ is converted to a $v3$-$v3$-$f5$ triple. The corresponding $v3$/$v5$ movement and $v3$-$v5$ pair generation operations in the $\sqrt{CC}$ domain are shown below.

**T-junction Movement and Cancellation:** As described in Section 3.4, the user can move a T-junction in the up, down, left and right directions. With this T-junction movement operation, it is possible to completely cancel or reduce a pair of T-junctions to be a single irregular element pair by colliding their adjacent $v3$-$v5$ pairs in the $\sqrt{CC}$ domain, depending on the conditions described in Proposition 3.7. Like the triple cancellation operation, the movement path can be found automatically with topological and geometric heuristics or manually by the user.

**Smoothing:** The user can smooth the mesh using Laplacian smoothing with back projection similar to the methods used in [TPC*10]. T-junctions can be treated as a special case to ensure that the two edge segments are collinear as described in [LKH08]. The visualizations of the edited meshes shown in the paper are generated in this fashion unless otherwise specified.

## 5. Applications and Results

We implemented our mesh editing framework in C++ using CGAL [cga] such that all edits could be performed interactively.

**Comparing Various *QD* Mesh Design Strategies:** We evaluated different *QD* mesh editing strategies in the context of mesh design and mesh optimization. In our first example, we modeled three versions of a wing of the Yas-Island architectural model and optimized the three meshes for planarity and angle deviation (approximated by the fairness term) [LPW*06]. In Figure 9, we visualize the optimization results. Our analysis shows that pure quad meshes are best for ensuring planarity, although meshes with mixed types of faces can achieve better angle deviation (and better smoothness of mesh lines).

In Figure 1, we show three versions of a tower model. The first version is a pure quad mesh and the third version is the dual of the first mesh and has therefore no irregular vertices. The second version has both irregular vertices and faces. This example was chosen to illustrate that the flexibility of quad-dominant meshes generally allows the designer to achieve smoother mesh lines using irregular faces in smooth or flat regions while preserving sharp features using irregular vertices.

**Mesh Connectivity Improvement:** In Figure 10, we illustrate the T-junction editing capabilities of our framework. T-junctions can be merged or moved to alter a mesh design.

Finally, in Figure 11, we show connectivity editing of a highly irregular mesh model (generated by the remeshing algorithm of Lai et al. [LKH08]). Even though there are highly irregular regions (including a 9-sided polygon), our editing framework can easily facilitate local mesh improvement. While we do not aim to replace automatic remeshing algorithms, local mesh editing has many unique advantages and provides a nice complement.

## 6. Conclusion and Future Work

In this paper, we present a connectivity editing framework for *QD* meshes that realizes the fundamental editing operations to control the location, type, and number of irregular elements. While the framework is useful for the manual control of mesh connectivity, for meshes with an excessive number of irregular elements, it may become impractical. We thus expect that a global framework that automates the local edits, possibly driven by topological and geometric heuristics (e.g. curvature), can be useful.

Besides the given examples, we envision other uses for *QD* mesh editing. In certain situations, regular vertices are very important and it is beneficial to convert irregular vertices to irregular faces. For example, it may be preferable to have regular vertices at the expenses of irregular faces,

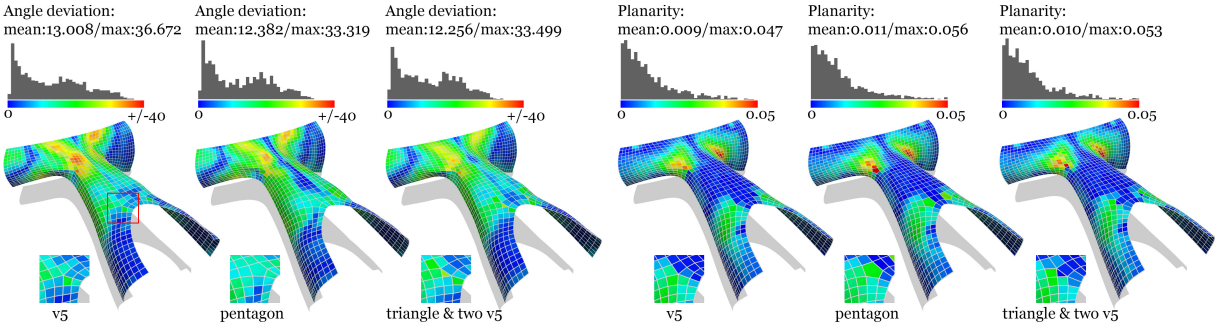



Figure 9: Geometric optimizations of three *QD* mesh designs of an architectural panel structure: a pure quad mesh with four $v5$, a *QD* mesh with four pentagons, and a *QD* mesh with four triangles plus eight adjacent $v5$. Left three: the three meshes are optimized toward equiangular faces, which is approximated by the fairness term [LPW*06]. Right three: the three meshes are optimized toward planar faces. Their mean and max errors are listed and visualized (explained in Figure 10). In general, by allowing a mixture of face types the meshes can be optimized with better angles. On the contrary, pure quad meshes work slightly better with planarity optimizations.

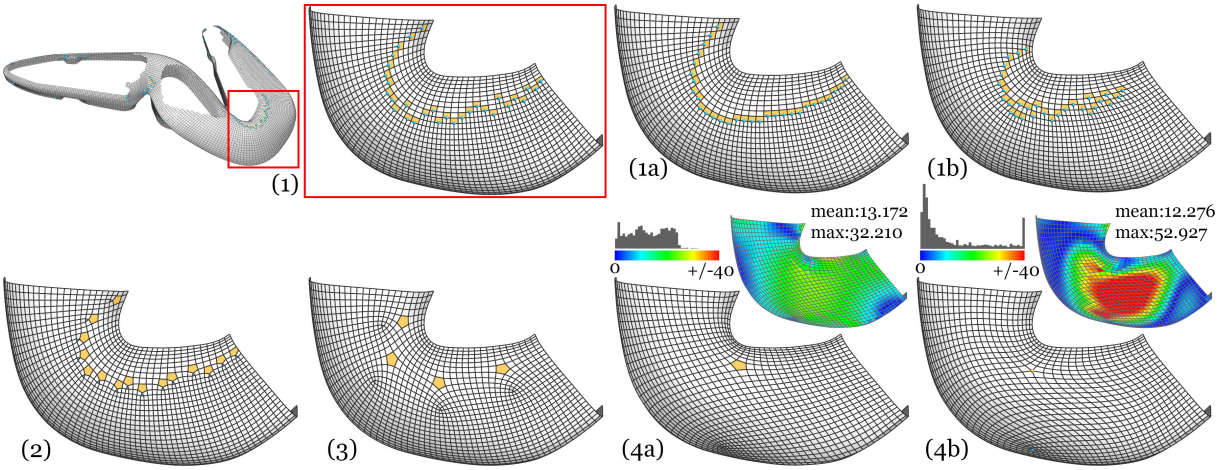



Figure 10: Finding alternative *QD* meshes for a semi-regular quadrangulation with T-junctions of an architectural structure. (1) We focus on a part of the whole structure that has an excessive amount of T-junctions in order to align with the underlying curvature. We first explore alternative T-junction patterns by moving them vertically (1a) and horizontally (1b). (2) to (4a) We progressively merge nearby T-junctions until we are left with a pair of irregular elements with opposite indices (an $f3$-$f5$ pair), at the cost of being less aligned with the curvature. The user can choose a version that compromises mesh regularity and curvature alignment. (4b) Alternatively, we change the $f3$-$f5$ pair to a $v3$-$v5$ pair. Interestingly, the *QD* mesh with irregular faces (4a) can be smoothed to a greater degree than can the *QD* mesh with irregular vertices (4b), resulting in smoother mesh-lines and less extreme angle deviations (on top we show the histograms and color visualizations of the angle deviations. Each face is visualized by the averaged deviation from the mean value of corner angles).

like in conical meshes [LPW*06] where the definition of a conical vertex is applicable only to regular vertices (the four adjacent faces need to be tangent to a common sphere). However, irregular vertices can help to combine multiple patches of conical meshes. We have spoken with multiple applied mathematicians about possible advantages of quad-dominant meshes for solving PDEs. We would like to pursue related research questions in future work.

**Acknowledgements:** We thank Helmut Pottmann for the proof of the discrete Gauss Bonnet theorem, Xiang Sun and Caigui Jiang for helping with the geometric optimizations, Alexander Schiftner, Yu-Kun Lai, Martin Marinov, and Dong-ming Yan for providing datasets, Yoshihiro Kobayashi and Christopher Grasso for the renderings, Virginia Unkefer for the proofreading, and the anonymous reviewers for their insightful comments. This work was supported by the National Science Foundation and KAUST.

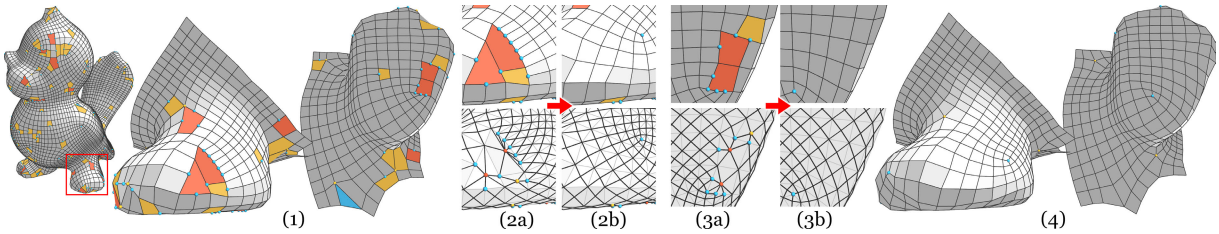



Figure 11: Connectivity improvement of a *QD* remeshing from [LKH08] (1). (2a) to (2b) We improve a region with faces with very large degrees (due to multiple incoming T-junctions) by regularizing the corresponding $\sqrt{CC}$ domain (shown below). (3a) to (3b) Another improved region. (4) The final improved mesh. We first remove excessive and nearby irregular elements. Next we move the remaining irregular elements to regions with corresponding curvatures. For demonstration purposes, we further change all irregular faces to be irregular vertices to make the mesh pure quad.

## References


[ACSD*03] ALLIEZ P., COHEN-STEINER D., DEVILLERS O., LÉVY B., DESBRUN M.: Anisotropic polygonal remeshing. *ACM Trans. Graph. 22*, 3 (2003), 485–493. 2

[BHS06] BOBENKO A. I., HOFFMANN T., SPRINGBORN B. A.: Minimal surfaces from circle patterns: Geometry from combinatorics. *Annals of Mathematics 164* (2006), 231–264. 2

[BLK11] BOMMES D., LEMPFER T., KOBBELT L.: Global structure optimization of quadrilateral meshes. *Comput. Graph. Forum 30*, 2 (2011), 375–384. 2, 7, 1, 6

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3 (2009), 77:1–77:10. 2

[CC78] CATMULL E., CLARK J.: Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10*, 6 (1978), 350 – 355. 2

[cga] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org. 8

[DS78] DOO D., SABIN M.: Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design 10*, 6 (1978), 356 – 360. 3

[DSSC08] DANIELS J., SILVA C. T., SHEPHERD J., COHEN E.: Quadrilateral mesh simplification. *ACM Trans. Graph. 27*, 5 (2008), 148:1–148:9. 2

[FM98] FREY P. J., MARECHAL L.: Fast adaptive quadtree mesh generation. *Proceedings of the Seventh International Meshing Roundtable* (1998), 211–224. 2

[Har69] HARARY F.: *Graph theory*. Addison-Wesley series in mathematics. Addison-Wesley, 1969. 4

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: Quadcover - surface parameterization using branched coverings. *Comput. Graph. Forum 26*, 3 (2007), 375–384. 2

[Kob96] KOBBELT L.: Interpolatory subdivision on open quadrilateral nets with arbitrary topology. *Comput. Graph. Forum 15* (1996), 409–420. 2, 3

[Kob00] KOBBELT L.: $\sqrt{3}$-subdivision. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques (SIGGRAPH)* (2000), 103–112. 3

[LKH08] LAI Y.-K., KOBBELT L., HU S.-M.: An incremental approach to feature aligned quad dominant remeshing. *Proceedings of the 2008 ACM symposium on Solid and Physical Modeling* (2008), 137–145. 2, 8, 10

[LPW*06] LIU Y., POTTMANN H., WALLNER J., YANG Y.-L., WANG W.: Geometric modeling with conical meshes and developable surfaces. *ACM Trans. Graph. 25*, 3 (2006), 681–689. 8, 9

[Mar09] MARÉCHAL L.: Advances in octree-based all-hexahedral mesh generation: Handling sharp features. *Proceedings of the 18th International Meshing Roundtable* (2009), 65–84. 2

[MK04] MARINOV M., KOBBELT L.: Direct anisotropic quad-dominant remeshing. *12th Pacific Conference on Computer Graphics and Applications (PG)* (2004), 207–216. 2

[MK06] MARINOV M., KOBBELT L.: A robust two-step procedure for quad-dominant remeshing. *Comput. Graph. Forum 25* (2006), 537–546. 2

[MNP08] MYLES A., NI T., PETERS J.: Fast parallel construction of smooth surfaces from meshes with tri/quad/pent facets. *Proceedings of the Symposium on Geometry Processing (SGP)* (2008), 1365–1372. 2

[NSY09] NASRI A., SABIN M., YASSEEN Z.: Filling n-sided regions by quad meshes for subdivision surfaces. *Comput. Graph. Forum 28*, 6 (2009), 1644–1658. 2

[PP11] PANOZZO D., PUPPO E.: Implicit hierarchical quad-dominant meshes. *Comput. Graph. Forum 30* (2011), 1617–1629. 2

[PZ07] PALACIOS J., ZHANG E.: Rotational symmetry field design on surfaces. *ACM Trans. Graph. 26*, 3 (2007), 55:1–55:10. 2

[PZKW11] PENG C.-H., ZHANG E., KOBAYASHI Y., WONKA P.: Connectivity editing for quadrilateral meshes. *ACM Trans. Graph. 30*, 6 (2011), 141:1–141:12. 2, 5, 6

[RLL*06] RAY N., LI W. C., LÉVY B., SHEFFER A., ALLIEZ P.: Periodic global parameterization. *ACM Trans. Graph. 25*, 4 (2006), 1460–1485. 2

[SCF*04] SEDERBERG T. W., CARDON D. L., FINNIGAN G. T., NORTH N. S., ZHENG J., LYCHE T.: T-spline simplification and local refinement. *ACM Trans. Graph. 23*, 3 (2004), 276–283. 2

[SL03] STAM J., LOOP C.: Quad/triangle subdivision. *Comput. Graph. Forum 22*, 1 (2003), 79–85. 2

[SS11] SADEGHI J., SAMAVATI F. F.: Smooth reverse Loop and Catmull-Clark subdivision. *Graph. Models 73*, 5 (2011), 202–217. 2

[Tau02] TAUBIN G.: Detecting and reconstructing subdivision connectivity. *The Visual Computer 18*, 5-6 (2002), 357–367. 2, 3

[TC06] TCHON K.-F., CAMARERO R.: Quad-dominant mesh adaptation using specialized simplicial optimization. *Proceedings of the 15th International Meshing Roundtable* (2006), 21–38. 2

[TPC*10] TARINI M., PIETRONI N., CIGNONI P., PANOZZO D., PUPPO E.: Practical quad mesh simplification. *Comput. Graph. Forum 29*, 2 (2010), 407–418. 2, 8, 1

[ZSW10] ZADRAVEC M., SCHIFTNER A., WALLNER J.: Designing quad-dominant meshes with planar faces. *Comput. Graph. Forum 29*, 5 (2010), 1671–1679. 2