Face Extrusion Quad Meshes

Karran Pandey
karran@cs.toronto.edu
University of Toronto
Toronto, Canada

J. Andreas Bærentzen
janba@dtu.dk
Technical University of Denmark
Copenhagen, Denmark

Karan Singh
karan@dgp.toronto.edu
University of Toronto
Toronto, Canada

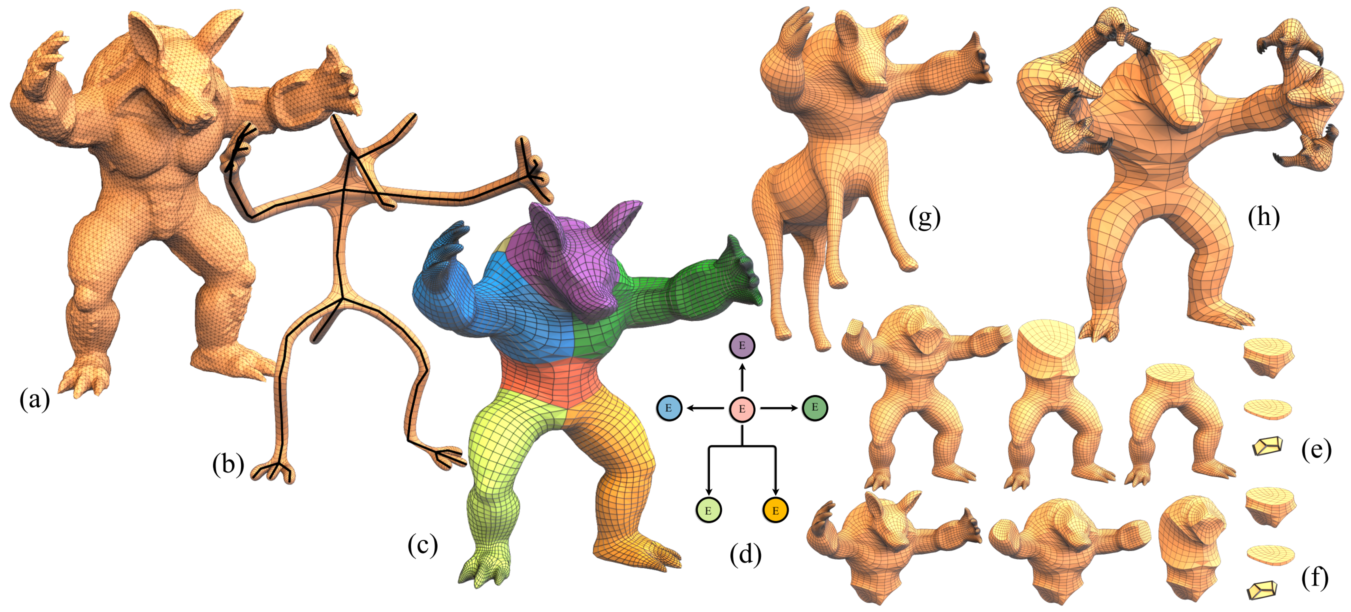

Figure 1: Face Extrusion Quad meshes (FEQs) are built using a sequence of face-loop modeling operations. Arbitrary triangular meshes (a), can automatically be skeletally topologized as an FEQ (b), and geometrically fit to the original mesh (c). The construction graph is shown abstracted, with color-coded subgraph nodes matching an FEQ part structure (d). FEQs enable a variety of shape history aware applications like inverse box modeling (e),(f), and FEQ-preserving cut-and-paste (g),(h).

ABSTRACT

We propose a 3D object construction methodology built on face-loop modeling operations. Our Face Extrusion Quad (FEQ) meshes, have a well designed face-loop structure similar to artist crafted 3D models. Furthermore, we define a construction graph which encodes a sequence of primitive extrude/collapse and bridge/separate operations that operate on admissible face-loops. We show that FEQs are imbued with a meaningful face-loop induced shape skeleton, part segmentation, plausible construction history, and possess the many advantages of extrusion-based 3D modeling. Our evaluation is threefold: we show a gallery of challenging 3D models transformed to FEQs with compelling face-loop structure; we showcase the potential of an inherent construction graph, using FEQ-based cut-paste and inverse modeling applications; and we demonstrate the impact of various algorithmic and parameter related choices for FEQ modeling and application.

CCS CONCEPTS

• **Computing methodologies** → **Mesh geometry models**.

KEYWORDS

quadrilateral meshes, shape modeling

ACM Reference Format:
Karran Pandey, J. Andreas Bærentzen, and Karan Singh. 2022. Face Extrusion Quad Meshes. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings (SIGGRAPH '22 Conference Proceedings), August 7–11, 2022, Vancouver, BC, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3528233.3530754

1 INTRODUCTION

Quadrilateral meshes, or simply quad meshes, are a popular representation for surface modeling, with many advantages, both functional and aesthetic [Gahan 2010; Pottmann et al. 2008; Vaughan

2012], and they are a rich area of ongoing academic research [Bommes et al. 2013] studying problems pertaining to quad shape, planarity, alignment, connectivity and remeshing.

Artists have also paid significant attention to the 3D modeling of objects using quad meshes. A common interactive quad mesh workflow, *box modeling*, is widely used by novices and professionals alike [Vaughan 2012]. Analogous to sculpting from a block of clay, box modeling creates objects by extruding and bridging parts of the shape from a simple starting shape such as a cube. Hundreds of online blogs and tutorials promote the technique [Pixxo3D 2021], for its easy to refine construction history, and inherent part structure. Artists also focus on the design of *face-loops* in a quad mesh (Figure 2). For a closed manifold quad mesh, a face-loop is sequence of adjacent quads, traversed across opposite edges (Figure 2). Well designed face-loops are the direct result of face extrusion operations, and are commonly used by artists to define meaningful regions of an object aligned with an object skeleton or deformation lines (Figure 2) [Bordegoni and Rizzi 2011; Gahan 2010; Johnson 2020].

Motivated by the artistic importance of face-loops in design, we propose, Face Extrusion Quad **FEQ** meshes, a 3D object representation built on atomic face-loop modeling operations that can explicitly capture the properties of artist designed face-loops. Specifically, we closely follow an invertible box modeling workflow, to build an FEQ as a sequence of primitive extrude/collapse and bridge/separate operations on admissible face-loops. Then we encode the sequence of operations as a directed construction graph that imparts an FEQ with a well-defined shape skeleton, part segmentation and plausible construction history (Figure 1). Further, our vocabulary of face-loop operations is designed to avoid the creation of undesirable self-intersecting face-loops (Figure 2(a)).

A practical requirement for FEQ adoption is the ability to automatically transform an arbitrary triangle mesh into an FEQ, that is aligned with its skeletal and part structure (Figure 1). Consistent with the extrusion based methodology we do this by forming quad meshes for each branch node. These branch nodes meshes are then extruded, joined by bridges, and fitted to the original geometry as discussed in Section 5. We demonstrate the effectiveness on a diverse range of models.

Next, we propose a method for computing a construction graph for FEQs in Section 6. Specifically, we show how a sequence of collapses can be used to iteratively strip face loops until the feature is reduced to its base patch (Figure 1). These collapses are recorded in the construction graph, and from this representation we can now reconstruct the feature on a different base patch. The construction graph also enables applications that can exploit a high-level shape structure, such as the cut-and-paste of features from an FEQ to another quad mesh. A core part of our contribution is that we do not simply copy a part of the mesh, but rather use the construction graph, to rebuild the feature on the target mesh, and since we are rebuilding (rather than simply adding a set of faces) the feature naturally adapts to the shape and connectivity of the target mesh. Artistically, our work can be viewed as the sculpting analogue to the problem of imagining a drawing sequence of strokes from a finished sketch [Fu et al. 2011].

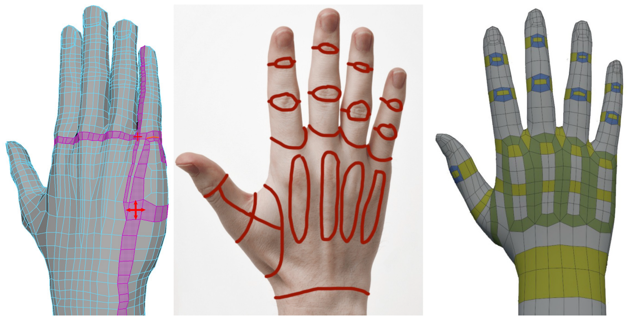
Figure 2: Face-loops in artist modeled quad meshes capture object skeleton/deformation: creating such quad meshes with a well-designed face-loop structure, avoiding face-loops (magenta) with self intersection (red +) and self-adjacency (orange edge), is challenging (a); segmenting the object into regions using deformation aligned face-loops (b), helps produce aesthetic quad meshes that we represent as face extrusion quad meshes (c) ©Johnson Martin 2021, topologyguides.com.

2 RELATED WORK

The past twenty years have seen a great deal of research on generating and manipulating quad meshes. Since the PGP method of Ray et al. [2006] and QuadCover by Kälberer et al. [2007], numerous often parametrization-based methods for quad mesh generation have emerged [Bommes et al. 2013]. However, not many of the methods are directly concerned with face loops. An important exception is the dual loops based method due to Campen et al. [2012; 2014] since their dual loops correspond to face loops in a coarse quad layout. Dual loops are also of interest in hex meshing (e.g. [Takayama 2019]) since the dual of a quad mesh can be seen as the intersection of the surface with a collection of sheets. These sheets are what Murdoch et al. [1997] called the Spatial Twist Continuum (STC), and the STC in turn defines a hexahedral mesh. A number of authors propose methods for improving quad meshes interactively or automatically through operations which move and remove singularities, e.g. [Feng et al. 2021; Peng et al. 2014, 2011]. In more closely related work, Daniels et al. [2008] consider face loop (called poly-chord) collapse in the context of quad mesh simplification.

Several authors have proposed skeleton based mesh generation methods with similarities to the one we present in Section 5 [Bærentzen et al. 2012; Ji et al. 2010; Suárez and Hubert 2018; Usai et al. 2015; Yao et al. 2009]. In particular, the method for skeleton guided construction of quad meshes due to Usai et al. is conceptually similar to ours, but the handling of branch nodes is very different. Specifically, the approach of Usai et al. entails the need to solve an integer linear programming problem which we avoid.

Bærentzen et al. [2014] proposed the polar annular co-representation for meshes cum skeleton. The main similarity is that they also exploit the structural information of a constrained mesh representation, but the polar annular representation of Bærentzen et al. is targeted at branching structures and less broadly applicable than FEQs. RigMesh by Borosán et al. [2012] shares the idea of creating a structurally aware model; in their case by explicitly creating the skeleton together with the geometry. With PushPull++ Lipp et al. [2014] created a system for editing models consisting of general planar polygons. While we share the focus on extrusions, their concerns are entirely different. Schmidt and Singh [2008] proposed Surface Trees, a method which allows for constructing a tree of layered triangle mesh surface features analogous to a CSG tree. Nuvoli et al. [2019] presented QuadMixer which allows users to perform boolean operations on quad meshes while retaining almost all of the input mesh structure.

3 DEFINITIONS

A face loop L in a quad mesh M consists of a contiguous loop of faces connected along their opposite edges. L is bounded by two edge cycles, which we refer to as its *rails*, L_R and L'_R, and contains a collection of opposite edges called *sleepers*, denoted by L_S.

Two face loops are said to *intersect* if they share at least one common face. If they only share common edges, they are said to be *adjacent*. Adjacent face loops are said to be *aligned* if they fully share one of their boundary loops. A face loop can therefore *self-intersect* or be *self-adjacent* if it encounters the same face or edge, respectively, more than once in its path (see Figure 3).

L is defined as *simple* if it is neither self-adjacent nor self-intersect-ing. Each rail of a simple face loop bounds a face set, referred to as its interior face set L_F. A *leaf face loop* is a simple face loop which does not contain any simple face loops in at least one of its interior face sets.

A *face path* is a contiguous sequence of faces that connect two faces. A face path must be traversable by stepping from face to face along one or more loops, switching from one loop to another only at intersections (see Figure 3 (right)).

A set of mesh faces, $F \subset M$ are said to be *path connected* if any two faces $f_1, f_2 \in F$ are always connected by at least one face path.

The *interior* of a face loop is not unique (except e.g. on a torus) since both rail curves bound a set of faces. However, the interior is easily disambiguated. When an extrusion is performed, we think of the extruded faces as the interior and when we analyze the loop structure of a mesh, we consider the face set of smallest area to be the interior. In the case of selection, the interior is user defined. When the user selects a feature on the mesh, she only has to indicate a face loop and which side should be considered the interior.

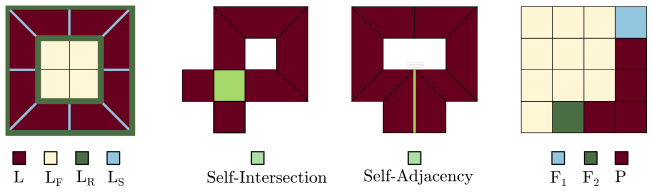

Figure 3: (Left to Right): A simple face loop and its constituent rails, sleepers and interior; a self intersecting face loop; a self adjacent face loop and a face path.

3.1 Visualization

Since face loops contain structural information about the mesh, it is possible to form a curve skeleton [Cornea et al. 2007] directly from the face loop structure as described in the following.

We define the *cylindricity* of a face loop L,

$$\mathrm{cyl}(L) = \frac{\left\|\sum_{i \in L_s} \mathbf{s}_i\right\|}{\sum_{i \in L_s} \|\mathbf{s}_i\|}\ , \tag{1}$$

where $\mathbf{s}_i \in \mathbb{R}^3$ are the sleeper edges which separate the faces of L. It is clear that $\mathrm{cyl}(L) \leq 1$ and equal only if all $\mathbf{s}_i$ are identical up to a positive scale factor. We now select face loops in order of decreasing cylindricity normalized by the area of the face loop. This normalization helps avoid a bias towards very short face loops. Loops are only selected if they are neither self-intersecting nor contain faces from previously selected loops. For each selected loop, we create a skeletal edge connecting the barycenters of the rail edge loops. When no more face loops can be selected, a vertex is formed as the barycenter of each patch of path connected faces none of which belong to a selected face loop. Finally, skeletal vertices are connected if they belong to patches or face loops that share boundary edges and unified if they are identical.

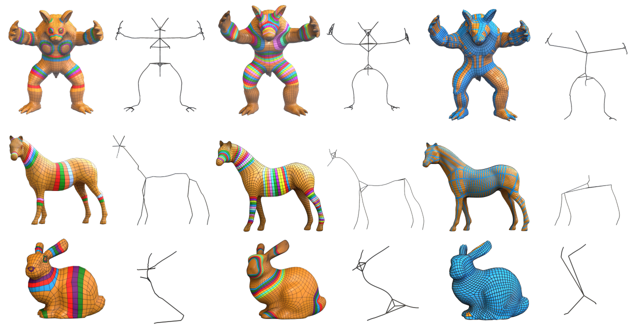

Figure 4: Comparing the face loop structure of FEQs (middle column) with artist-made meshes (left) [Takayama et al. 2013] [Marcias et al. 2015] and feature-line driven quad meshing (right) [Pietroni et al. 2021]. We see that FEQs closely align to artist-made meshes while feature-line driven approaches fail to capture parts of the model due to self-intersecting face loops. A single self-intersecting face loop per model is illustrated in blue in the right column.)

The face loop skeleton is an analysis tool which can be used to show how consistent the face loop structure is: the skeleton can be formed for any closed quad mesh, but if the mesh contains a significant number of self-intersecting face loops (c.f. Figure 2), many features are likely to be missing from the skeleton as shown in Figure 4.

The chains of edges joined by valence two nodes in the skeleton induce a segmentation of the mesh into aligned face loops. This segmentation is also an important analysis tool which provides a good part-decomposition of FEQs, and throughout the paper we will show this segmentation on our models.

4 FEQS

We now define the four operations, illustrated in Figure 5, that form the core of our methodology for face extrusion quad mesh (FEQ) modeling. Informally, FEQs are defined as meshes which are created or could be created using the forward operations extrusion and

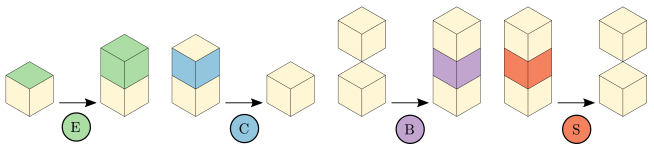

Figure 5: (From left to right): The atomic operations: extrusion, collapse, bridging and separation.

bridging. We will consider the operations in terms of a quad mesh M. Each operation has specific conditions that must be fulfilled for it to be valid, and they all either add or remove a single face loop.

Extrusion. Given a path connected set of mesh faces with the topology of a disk, $F \subset M$, the extrusion of F disconnects F from M, introduces a new face loop, L_E, which reconnects F to the M, and moves the vertices of F to their new position. Thus, the effect of an extrusion is the introduction of a face loop followed by a modification of the geometry.

Collapse. A face loop L, can be collapsed onto one of its rails L_R by collapsing each of its sleeper edges L_S onto a vertex in L_R. Geometrically, we flatten the interior geometry by boundary constrained smoothing if L is a leaf.

We define a valid face loop collapse as one where L is simple and its interior face set, F, consists of one or more components of disk topology. Usually, there is only a single component, but the inner rail curve of L may contain the same vertex more than once. In this case F will contain several components. If F is a path connected set of faces of disk topology then the collapse is precisely the inverse of extrude.

Bridging. Bridging can be defined as an extrusion where we match the extruded face set to another face set and remove both whereby the new face loop becomes a connecting tunnel between the resulting holes. Two path connected sets of mesh faces of disk topology, $F_1, F_2 \subset M$, can be bridged with a face loop of quads only if their connectivities are each other's precise mirror images since the edges through which face loops enter and exit F_1 and F_2 are then also mirror images. This ensures that face loops are correctly paired after bridging and hence do not self-intersect.

It is possible to raise the genus of a mesh by bridging two sets of faces that belong to the same connected component. However, it is clear that the condition above is not enough to ensure that self-intersecting face loops are avoided in this case: two different face loops which are joined into one by the bridging might have been intersecting before the bridge was created. We prefer to allow self-intersections in higher genus meshes rather than impose further conditions which would often make bridging infeasible.

Separation. A separation can be perceived as the inverse of a bridging. The faces in a loop L are removed from the mesh and the two holes corresponding to the rail curves are filled by quadrangulations. Unlike the other operations, we have not implemented separation in our framework. For meshes which are constructed using a sequence of the operations discussed above, we can trivially perform the separations that are typically needed simply by omitting the bridgings that would otherwise be performed.

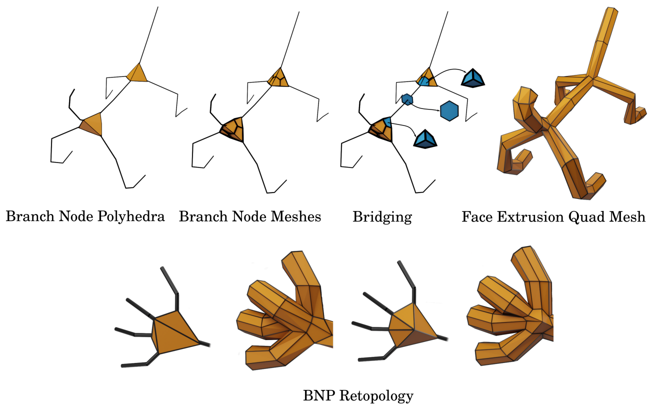

Figure 6: (Top, From left to right): BNPs are subdivided to form branch node meshes which are then bridged and extruded to reconstruct graph paths. The matched branch face sets are displayed in blue in the bridging step, insets show the precise structure of the face sets. (Bottom) We edit the connectivity of the BNP by introducing auxiliary vertices to improve the face loop structure at planar junctions.

What are FEQs then? While it would be satisfying to give a precise and formal definition of FEQs this is neither possible nor necessary, and we will briefly address why.

It is possible to identify connectivities which cannot be formed by face extrusion. Specifically, self-intersecting and self-adjacent face loops (c.f. Figure 3) cannot be the result of face extrusion.

On the other hand, given a mesh which appears to be constructible using only extrusions (and possibly bridging), the sequence of extrusions needed to form this particular mesh from a given initial mesh (say a cube) is not unique, and a sequence of collapses is not guaranteed to lead back to this initial mesh. It is possible we will be led back to a point where no further collapses are possible (see Figure 12 (c)). In these cases, we would have to backtrack in order to reach the initial mesh. Thus, in some cases, the answer to whether a given mesh can be formed from an initial mesh requires significant computation.

However, our atomic operations are not predicated on global properties of the mesh, only on the specific conditions. Nor do we necessarily need to decompose the entire mesh: in many cases, we might want to create an FEQ from a part of a larger mesh or add an FEQ component to a mesh which can simply be a general quad mesh. Finally, as discussed below, we can create an FEQ from a part of a mesh even if some of the face loops do not admit a valid collapse.

5 EXTRUSION-BASED REMESHING

Our algorithm for creating an FEQ, M, from an initial triangle mesh, M_t is based on the atomic operations of extrusion and bridging and consists of two main steps: inverse skeletonization and fitting. An implementation is made available in https://github.com/janba/GEL, and the process is described in greater detail in [Pandey et al. 2022].

Inverse Skeletonization. The remeshing relies on a curve skeleton with straight edges [Cornea et al. 2007; Tagliasacchi et al. 2016]

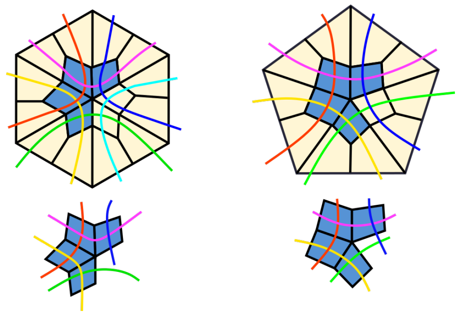
Figure 7: The figure illustrates matching subsets of the branch face sets. On the left, the vertex has valence 6 and on the right it has valence 5. By selecting only four faces in each one ring, we ensure that the two subsets have identical structure and can thus be bridged.

which we compute using the Local Separator method [Bærentzen and Rotenberg 2021] and store in an undirected graph $G = (V, E)$ where each node $n \in V$ is imbued with a 3D position. We construct a quad mesh from G using the following steps. Initially, the branch nodes of G, i.e nodes with valence > 2, are identified and a triangle mesh is created and subdivided, producing a quad mesh for each branch node. The final mesh is obtained using bridging and extrusion operations. The steps are shown in Figure 6 and discussed in greater detail below.

Consider a branch node n. Inspired by prior work on inverse skeletonization [Bærentzen et al. 2012], we construct a representative mesh for n by computing a spherical Delaunay triangulation of points found by intersecting each outgoing path with a sphere centered on n. The resulting Branch Node Polyhedron (BNP) contains a vertex for each outgoing path, referred to as a *path vertex*. The BNP is subsequently improved using a procedure described in Section 5.1.

Next, we produce a *branch node mesh* for n by a single iteration of Catmull-Clark subdivision [Catmull and Clark 1978] of the BNP. We state without proof that Catmull-Clark subdivision of a triangle mesh produces a quad mesh where all face loops circle the original vertices and do not self-intersect. For each outgoing edge of n, the branch node mesh contains a unique *path vertex* along with a surrounding ring of quads. This is an especially useful structure since quads from this one-ring can be bridged or extruded to reconstruct the graph structure. Below, we refer to this one-ring of quads as a *branch face set*.

To reconstruct a path between two graph nodes, we bridge the branch face sets corresponding to the two nodes (c.f. Section 4). Since the branch face sets are created by subdividing triangle fans, they are identical if the path vertices have the same valence. If the valencies differ, we can select a matching subset for each of the the branch face sets, and these subsets are then bridged. If the valencies of the path vertices are k_1 and k_2, respectively, we select $\min(k_1, k_2) - 1$ contiguous faces from each face set as our subsets. The branch face sets are rings of quads that have a very regular structure: any quad shares a face loop with each of its two neighbors in the ring. By selecting $\min(k_1, k_2) - 1$ contiguous faces, the subsets leave both rings open which entails that the outermost faces do not share their face loops with another face. Such a subset will be referred to as an *open ring* whereas a full branch face set will be referred to as a *closed ring*. We must bridge identical subsets of branch face sets: either open or closed rings depending on whether the branch face sets are different or identical, respectively. For an illustration of matching open rings, please refer to Figure 7.

A similar process is followed for leaf connections, but instead of bridging, we here need only extrude, and we always extrude the full branch face sets (closed ring). Both bridging and extrusions are guided by the graph edges as illustrated in Figure 6.

Now that we have a quad mesh with the desired face loop structure, we inflate and fit the inverse skeletonized mesh M to the input triangle mesh M_t. For our fitting procedure, we use the sparse-linear least-squares system outlined in Least-Squares Meshes [Sorkine and Cohen-Or 2004]. For more details on the fitting procedure, please refer to our supplementary document [Pandey et al. 2022].

5.1 BNP Connectivity Improvement

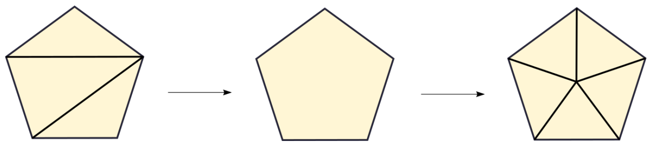
Figure 8: The figure illustrates the connectivity improvement for planar regions in a BNP. The interior edges are removed, and a central vertex is introduced which is then connected to all boundary vertices.

Two types of changes are made to the BNP before Catmull-Clark subdivision. The first type involves introducing auxiliary vertices to ensure that the BNP triangles are closer to equilateral which, in turn, leads to a better output mesh. The second is a refinement step which ensures that we can match closed rings in most cases.

Planar Junctions. Planar regions in a BNP can be identified as a collection of faces with a small dihedral angle (less than 15°) between them. Such regions correspond to a set of outgoing graph paths which approximately lie within the same plane, like, say, the fingers of a hand. In terms of this example, we would expect that a given finger is only connected to the adjacent fingers, but since the BNP is a triangulation, we also get edges connecting non-adjacent fingers as illustrated in Figure 8.

To prevent this, we *retopologize* planar regions in a BNP by introducing an auxiliary vertex in the center of the planar region and connecting this new vertex to all of the boundary vertices. Returning to the hand example, this means that the fingers are now only connected to adjacent fingers and the auxiliary vertex which roughly corresponds to the palm.

Surplus Quad Patches. When open rings are matched, left over faces appear as surplus quad patches in junctions of the FEQ. While such patches are sometimes necessary for the quad layout [Pandey et al. 2022] they can often be removed, leading to a simpler mesh.

To reduce the number of unmatched faces in the bridging step, the BNP is refined before Catmull-Clark subdivision. We know the number of faces in the branch face set will be equal to the valence of the BNP path vertex. Thus, we can increase the number of faces in the smaller branch face set by splitting one of the BNP edges in the link of the path vertex.

This split is only performed if it leads to a match between closed rings (i.e. $k_1 = k_2$) for the connected pair of path vertices, and does not result in an increased value of $|k_1 - k_2|$ for any other pair of connected path vertices. The procedure is applied greedily until no more improvements can be made.

Importantly, we do not make changes that would propagate the need for subdivision. Thus, unlike [Bærentzen et al. 2012; Suárez and Hubert 2018; Usai et al. 2015], we avoid searching for a global solution, e.g. by solving an integer linear programming problem.

6 THE CONSTRUCTION GRAPH

Let's consider a part S constructed by performing an ordered set of face extrusions on a base mesh M. The construction history of S can be represented using a directed acyclic graph of extrusions G_C called the *construction graph* (illustrated in Figure 12).

G_C captures the construction history by storing extrusion operations and the relationships between them. For each face extrusion, a node E is introduced in G_C with an associated face-ownership F_E. F_E consists of the faces modified or created by the extrusion, i.e its base face set and introduced face loop L. We construct a directed edge D from an extrusion node E to E', if there exist faces $F_D \subset F_E$ which are then further extruded by E'. As a result, the union of faces associated with incoming edges to E amount to its base face set, and its outgoing edges have associated faces used by future extrusions. Whenever two or more contiguous extrusion nodes are successively applied to the same face set F, we merge them into a *combined extrusion node*. For simplicity, we will assume that nodes are combined below.

6.1 Encoding the Construction Graph

Given a selection S, its construction graph G_C can be inferred by iteratively deconstructing S using face loop collapses. In particular, each collapse can be perceived as inverting a face extrusion, and is therefore accompanied by the addition of an extrusion node to G_C. As shown in Figure 9, directed edges are added between inferred extrusion nodes if the faces removed or flattened during a collapse belong to an existing node in G_C. This process is repeated until there are no more collapsible face loops in S, thus completing the construction graph G_C.

As we wish to remove features from the extremities first, we begin by collapsing the leaf face loop with the smallest area in S. We then proceed to iteratively collapse the collection of aligned face loops adjacent to it, before similarly identifying the next leaf face-loop and repeating this process. Each such collection of aligned face loops therefore introduces a combined extrusion node E in G_C.

6.1.1 Encoding Extrusions. This deconstruction process is accompanied by an encoding which allows us to reconstruct the inferred construction history on any selected base face set with disk topology.

Given a collection of aligned face loops which introduce a combined extrusion node E in G_C, we first encode the faces in the collection which belong to previous extrusion nodes in G_C. Specifically,

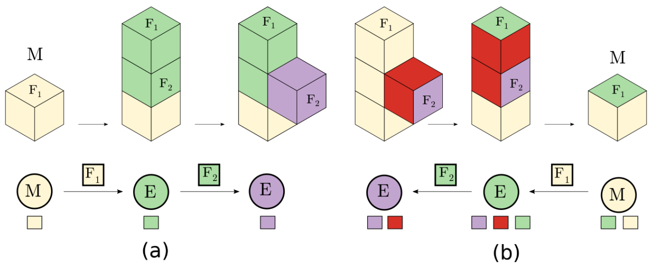

Figure 9: The construction graph for a forward modeling process (a) which is then inferred using face loop collapses (b). The colors below each node represent their face ownerships.

each outgoing face set F_D to a node E_D is encoded by parameterizing F_E and storing the boundary loop of F_D in the parameter space. We parameterize F_E by mapping its base face set to the unit disk using a harmonic parameterization [Floater and Hormann 2005], and extending the map to incorporate the aligned face loops. To be precise, the boundary vertices in the parameter space are radially extended to concentric rings of increasing integer radius to parameterize the aligned face loops. The parameterization therefore preserves the face loop structure of F_E in the parameter space. We now store the parameterized boundary loop of F_D, denoted $\tilde{B}_D$, in the node E, along with a supplied or arbitrary reference point encoding the orientation of E_D. See Figure 10 for an illustration of the process.

As we then iteratively collapse the face loops in the collection, we encode the geometry of each inverted extrusion in E. Our geometric encoding takes inspiration from prior work on edge-ring based representations for part-based modeling [Schmidt 2011]. In particular, while collapsing a leaf face loop L with interior face set F onto its rail L_R, the geometry of its corresponding face extrusion is encoded relative to vertices in L_R. Specifically, the position of each face vertex prior to the collapse is stored in the local coordinate frame of each rail vertex. After the collapse is performed, this collection of rail relative vectors, denoted D_{RF}, are stored in E along with the parametrized coordinates of the rail and face vertices.

Having fully collapsed the collection, the base face set F is now added to the face ownership of E in G_C. To encode the orientation of E, we additionally store the vertex corresponding to $(1, 0)$ in the parameter space, as the reference point of E. After the final face loop has been collapsed, the area of its collapsed interior F_s is stored along with G_C to encode the scale of the feature.

6.2 Decoding the Construction Graph

Given a target face set F_t with disk topology and a reference point v_{F_t} on its boundary, we now outline our procedure to decode G_C on to F_t with the desired scale and orientation. Each extrusion node, E, in G_C is visited in the opposite order of the encoding.

Given an encoded extrusion node, with base face set F' and an arbitrary or supplied reference point $v_{F'}$ we reconstruct the geometry of the extrusion in three steps. First, F' is mapped to the unit disk, ensuring that the supplied reference point $v_{F'}$ is mapped to $(1, 0)$. We then topologically introduce a face loop L' around F'.

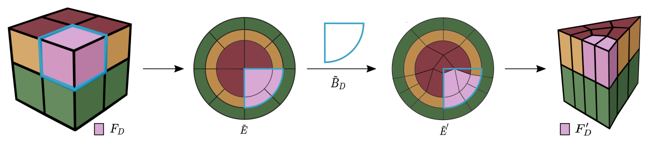

Figure 10: An outgoing face set (pink) of a combined extrusion node is encoded using its boundary loop (blue) in the parameter space. The parameterized boundary loop can then be used to select corresponding face sets on another combined extrusion node with the same number of extrusions (right).

The encoded collection of rail relative vectors D_{RF} in the parameter space are then interpolated to find a corresponding set of vectors $D_{R'F'}$ for the parametric coordinates of each vertex in L'_R and F'. The vectors in $D_{R'F'}$ are then scaled to account for the difference in the area of the source and target face sets F_s and F_t.

Having computed $D_{R'F'}$, we now encode its vectors in the corresponding local coordinate frames of the rail vertices in L'_R. The extruded vertex positions of each face vertex is then computed by averaging the contributions of each rail vertex.

We iteratively reconstruct contiguous extrusions in this manner, until we have fully rebuilt a combined extrusion node E'. We next decode the outgoing face sets of E' in order to identify the base face set and reference point of the next extrusion node in the graph. Specifically, f or each outgoing directed edge D from E' to a target extrusion node E'_D, we have a parameterized boundary loop $\tilde{B}_D$. We now find the corresponding outgoing face set F'_D by first parameterizing $F_{E'}$ as before, and selecting faces whose centers are contained within $\tilde{B}_D$ in the parameter space. The selected faces F'_D are then added to the base face set of E'_D. If $\tilde{B}_D$ contains the reference point v of E'_D, we further identify a corresponding reference point on F'_D by selecting the closest point to v on its boundary in the parameter space.

As the process of decoding G_C is an iterative sequence of local extrusions, there can sometimes be non-smooth regions in the reconstruction across separate extrusions. We therefore perform an iteration of constrained smoothing to adapt the geometry specified by the encoded extrusions to the reconstructed mesh connectivity.

7 RESULTS

A range of FEQ models created using the method discussed in Section 5 is shown in Figure 11. The skeleton induced part structure and the extrusion derived connectivity combine to produce meshes quite similar to what we believe an artist might have created. This argument is further reinforced by the similarity of our face loop skeletons to those of models made by artists as shown in Figure 4. Figure 14 shows a comparison between our method and the method of Usai et al. [2015]. Unlike the method of Usai et al. our branch nodes do not have to abide by the symmetries of the cube which leads to a significant difference between the methods for these inputs.

The construction graph stores an inferred plausible construction history for an FEQ which opens up a wide range of possibilities for mixing and recombining features of quad meshes while preserving extrusion connectivity as shown in Figure 1 which also illustrates are capabilities to automatically inverse box model a mesh down to a cube. Figure 12 (a) shows steps of a collapse sequence along with the corresponding addition of nodes to the construction graph. A salient question is if this process can go wrong? Figure 12 (c) shows an adversarial example where the blue face loops were initially collapsed, making the hand very thin. The red tips of the fingers no longer correspond to face extrusions and are hence not leaf extrusions. Nevertheless, we can still extrude the hand down to the red base mesh shown on the right. In (d) the middle finger of the same mesh is collapsed and then extruded back on a base mesh of four quads (middle) and a single quad (right). To sum up, our procedure incorporates incollapsible face loops in the base mesh of the extruded part; when the part is pasted onto another base mesh these loops do not appear since they are not part of the construction graph. From a geometry point of view, Figure 13 (top) shows how a complex part pasted onto rectangular patches of different resolution degrades gracefully with the number of quads in the target patch.

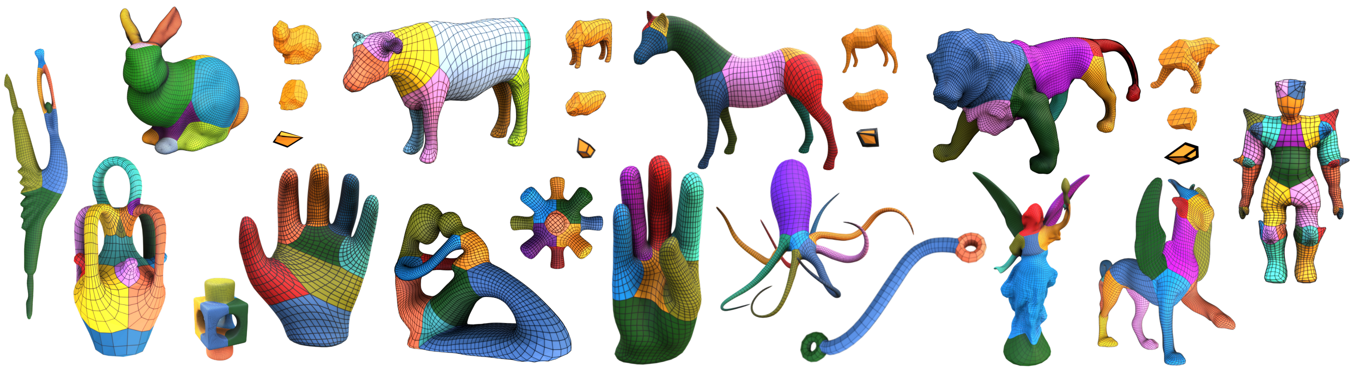

Figure 11: Gallery of FEQ models created the bridging and extrusion-based remeshing method. Insets in the first row illustrate the inverse box modeling capability of our framework

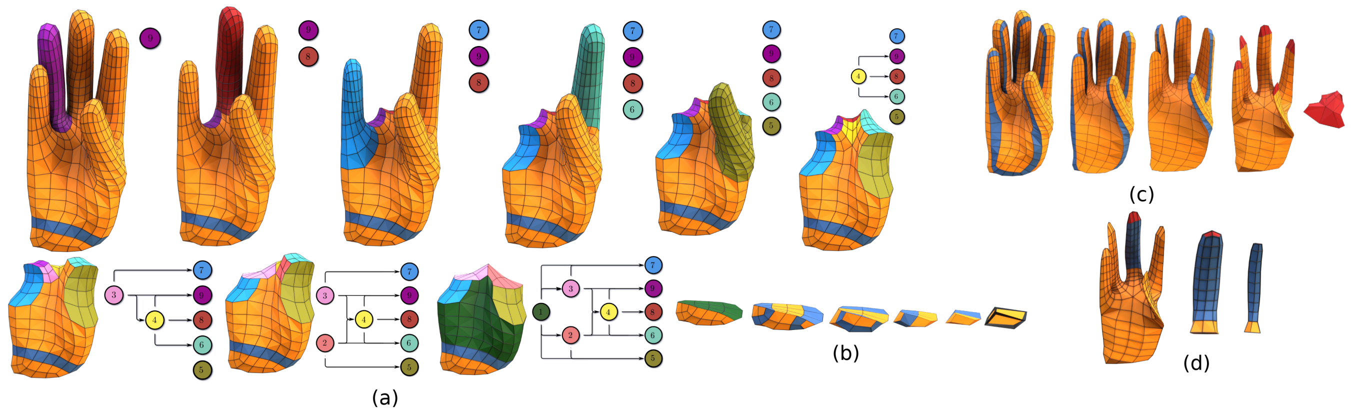

Figure 12: a) The process of inferring the construction graph for the hand model based on a given face loop selection (blue). The addition of combined extrusion nodes to the graph is illustrated, with the colors representing the face set ownerships of the deconstructed features. b) The deconstruction process is continued until we end at a cube. c) Adversarial deconstruction d) Cut and paste of incollapsible face loops (in red)

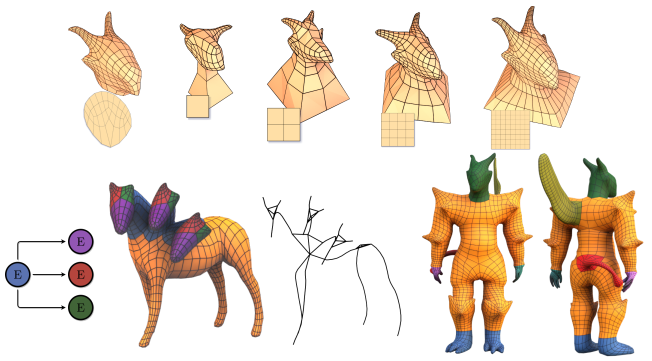

Figure 13: Top: Cut and pasting the feline head onto different base face sets consisting of 1, 4, 16 and 64 quads. Bottom: A cut and paste of the horse head (left) which illustrates the clean extension of the face loop structure in the paste operation and part-based modeling using our framework (right).

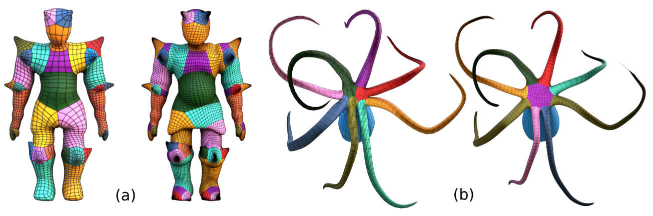

Figure 14: Remeshing using our method (left) and the method of Usai et al. [2015] (right) for the warrior model (a). Note that we here compare only automatic remeshing; Usai et al. allow for manual adjustment which was used to improve this model. Remeshings of the octopus model are compared in (b). Again, our method is shown on the left.

Finally, Figure 13 (bottom) shows use cases where the same part is pasted multiple times (a) and a mashup of parts from a variety of models (b).

8 DISCUSSION, FUTURE WORK AND CONCLUSION

In this paper, we have essentially advocated for the use of face loops as opposed to single faces as the modeling primitive. The resulting FEQ meshes turn out to not only have very clean quad mesh structure but also to be highly modular, allowing for easy modification and combination of features of 3D models. We have also proposed a method for generating FEQs which produces highly regular meshes by design, since the models are constructed from simple branch node meshes only by extrusion and bridging.

A limitation of our current implementation of the construction graph is the absence of a bridge node which would facilitate cut and paste of parts with higher genus, but there are many other avenues for future work. Specifically, our current framework is tied to face extrusions, but some features which involve sharp creases, e.g. eyes or mouths, tend to be modeled better by edge extrusions. Finally, as implied by the findings of Murdoch et al. [1997], a clean face loop structure can be helpful for hexahedral mesh generation, and FEQs are exactly characterized by a clean face loop structure.

In summary, our principal contribution is a modeling framework built around face loop operations that produce Face Extrusion Quad meshes: we show that arbitrary meshes can be transformed to an FEQ with a plausible construction history; FEQs are well integrated with the popular *box modeling workflow*, and enable history aware modeling applications such as the cut and paste of part structures.

ACKNOWLEDGMENTS

This project was funded in part by NSERC and Advokat Bent Thorbergs Fond (award no. 66.531).

REFERENCES

J Andreas Bærentzen, Rinat Abdrashitov, and Karan Singh. 2014. Interactive shape modeling using a skeleton-mesh co-representation. *ACM Transactions on Graphics*

(TOG) 33, 4 (2014), 1–10.

J. Andreas Bærentzen, Marek Krzysztof Misztal, and K Wełnicka. 2012. Converting skeletal structures to quad dominant meshes. *Computers & Graphics* 36, 5 (2012), 555–561.

David Bommes, Bruno Lévy, Nico Pietroni, Enrico Puppo, Claudio Silva, Marco Tarini, and Denis Zorin. 2013. Quad-mesh generation and processing: A survey. *Computer Graphics Forum* 32, 6 (2013), 51–76.

M. Bordegoni and C. Rizzi. 2011. *Innovation in Product Design: From CAD to Virtual Prototyping*. Springer. http://books.google.ca/books?id=gXS-7JIJEwEC

Péter Borosán, Ming Jin, Doug DeCarlo, Yotam Gingold, and Andrew Nealen. 2012. RigMesh: Automatic Rigging for Part-Based Shape Modeling and Deformation. *ACM Transactions on Graphics* 31, 6, Article 198 (nov 2012), 9 pages. https://doi.org/10.1145/2366145.2366217

Andreas Bærentzen and Eva Rotenberg. 2021. Skeletonization via Local Separators. *ACM Transactions on Graphics* 40, 5, Article 187 (Sept. 2021), 18 pages. https://doi.org/10.1145/3459233

Marcel Campen, David Bommes, and Leif Kobbelt. 2012. Dual Loops Meshing: Quality Quad Layouts on Manifolds. *ACM Transactions on Graphics* 31, 4, Article 110 (jul 2012), 11 pages. https://doi.org/10.1145/2185520.2185606

Marcel Campen and Leif Kobbelt. 2014. Dual Strip Weaving: Interactive Design of Quad Layouts Using Elastica Strips. *ACM Transactions on Graphics* 33, 6, Article 183 (nov 2014), 10 pages. https://doi.org/10.1145/2661229.2661236

Edwin Catmull and James Clark. 1978. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-aided design* 10, 6 (1978), 350–355.

Nicu D Cornea, Deborah Silver, and Patrick Min. 2007. Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization & Computer Graphics* 13, 3 (2007), 530–548.

Joel Daniels, Cláudio T. Silva, Jason Shepherd, and Elaine Cohen. 2008. Quadrilateral Mesh Simplification. *ACM Transactions on Graphics* 27, 5, Article 148 (2008), 9 pages. https://doi.org/10.1145/1457515.1409101

Leman Feng, Yiying Tong, and Mathieu Desbrun. 2021. Q-Zip: Singularity Editing Primitive for Quad Meshes. *ACM Transactions on Graphics* 40, 6, Article 258 (dec 2021), 13 pages. https://doi.org/10.1145/3478513.3480523

Michael S Floater and Kai Hormann. 2005. Surface parameterization: a tutorial and survey. In *Advances in multiresolution for geometric modelling*. Springer, 157–186.

Hongbo Fu, Shizhe Zhou, Ligang Liu, and Niloy J. Mitra. 2011. Animated Construction of Line Drawings. *ACM Transactions on Graphics* 30, 6 (dec 2011), 1–10. https://doi.org/10.1145/2070781.2024167

A. Gahan. 2010. *3D Automotive Modeling: An Insider's Guide to 3D Car Modeling and Design for Games and Film*. Elsevier Science. http://books.google.ca/books?id=_VQS1jTnuWMC

Zhongping Ji, Ligang Liu, and Yigang Wang. 2010. B-Mesh: A Modeling System for Base Meshes of 3D Articulated Shapes. *Computer Graphics Forum* 29, 7 (2010), 2169–2177.

M. Johnson. 2020. *Modeling a Human Hand*. Guides for 3D Artists (topologyguides.com).

Felix Kälberer, Matthias Nieser, and Konrad Polthier. 2007. Quadcover-surface parameterization using branched coverings. *Computer graphics forum* 26, 3 (2007), 375–384.

Markus Lipp, Peter Wonka, and Pascal Müller. 2014. PushPull++. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–9.

Giorgio Marcias, Kenshi Takayama, Nico Pietroni, Daniele Panozzo, Olga Sorkine-Hornung, Enrico Puppo, and Paolo Cignoni. 2015. Data-Driven Interactive Quadrangulation. *ACM Transactions on Graphics* 34, 4, Article 65 (jul 2015), 10 pages. https://doi.org/10.1145/2766964

Peter Murdoch, Steven Benzley, Ted Blacker, and Scott A Mitchell. 1997. The spatial twist continuum: A connectivity based method for representing all-hexahedral finite element meshes. *Finite elements in analysis and design* 28, 2 (1997), 137–149.

Stefano Nuvoli, Alex Hernandez, Claudio Esperança, Riccardo Scateni, Paolo Cignoni, and Nico Pietroni. 2019. QuadMixer: Layout Preserving Blending of Quadrilateral Meshes. *ACM Transactions on Graphics* 38, 6, Article 180 (nov 2019), 13 pages. https://doi.org/10.1145/3355089.3356542

Karran Pandey, J. Andreas Bærentzen, and Karan Singh. 2022. Face Extrusion Quad Meshes: Supplementary Material.

Chi-Han Peng, Michael Barton, Caigui Jiang, and Peter Wonka. 2014. Exploring Quadrangulations. *ACM Transactions on Graphics* 33, 1, Article 12 (feb 2014), 13 pages. https://doi.org/10.1145/2541533

Chi-Han Peng, Eugene Zhang, Yoshihiro Kobayashi, and Peter Wonka. 2011. Connectivity Editing for Quadrilateral Meshes. In *Proceedings of the 2011 SIGGRAPH Asia Conference* (Hong Kong, China) *(SA '11)*. Association for Computing Machinery, New York, NY, USA, Article 141, 12 pages. https://doi.org/10.1145/2024156.2024175

Nico Pietroni, Stefano Nuvoli, Thomas Alderighi, Paolo Cignoni, and Marco Tarini. 2021. Reliable Feature-Line Driven Quad-Remeshing. *ACM Transactions on Graphics* 40, 4, Article 155 (jul 2021), 17 pages. https://doi.org/10.1145/3450626.3459941

Pixxo3D. 2021. *MODELLING For Absolute Beginners*. https://www.youtube.com/watch?v=9xAumJRKV6A

H Pottmann, A Asperl, M Hofer, and A Kilian. 2008. *Architectural geometry : first edition*. Bentley Institute Press.

Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. 2006. Periodic global parameterization. *ACM Transactions on Graphics (TOG)* 25, 4 (2006), 1460–1485.

Ryan Schmidt. 2011. *Part-based representation and editing of 3d surface models*. Ph.D. Dissertation. University of Toronto.

Ryan Schmidt and Karan Singh. 2008. Sketch-based procedural surface modeling and compositing using Surface Trees. *Computer Graphics Forum* 27, 2 (2008), 321–330.

Olga Sorkine and Daniel Cohen-Or. 2004. Least-squares meshes. In *Proceedings Shape Modeling Applications 2004*. IEEE, 191–199.

AJ Fuentes Suárez and Evelyne Hubert. 2018. Scaffolding skeletons using spherical Voronoi diagrams: Feasibility, regularity and symmetry. *Computer-Aided Design* 102 (2018), 83–93.

Andrea Tagliasacchi, Thomas Delame, Michela Spagnuolo, Nina Amenta, and Alexandru Telea. 2016. 3D Skeletons: A State-of-the-Art Report. *Computer Graphics Forum* 35, 2 (2016), 573–597. https://doi.org/10.1111/cgf.12865

Kenshi Takayama. 2019. Dual sheet meshing: An interactive approach to robust hexahedralization. *Computer Graphics Forum* 38, 2 (2019), 37–48.

Kenshi Takayama, Daniele Panozzo, Alexander Sorkine-Hornung, and Olga Sorkine-Hornung. 2013. Sketch-based Generation and Editing of Quad Meshes. *ACM Transactions on Graphics* 32, 4, Article 97 (July 2013), 8 pages. https://doi.org/10.1145/2461912.2461955

Francesco Usai, Marco Livesu, Enrico Puppo, Marco Tarini, and Riccardo Scateni. 2015. Extraction of the Quad Layout of a Triangle Mesh Guided by Its Curve Skeleton. *ACM Transactions on Graphics* 35, 1, Article 6 (dec 2015), 13 pages. https://doi.org/10.1145/2809785

W. Vaughan. 2012. *Digital Modeling*. New Riders. https://books.google.co.in/books?id=nzJ2QgAACAAJ

Chih-Yuan Yao, Hung-Kuo Chu, Tao Ju, and Tong-Yee Lee. 2009. Compatible quadrangulation by sketching. *Computer Animation and Virtual Worlds* 20, 2-3 (2009), 101–109.