Sculpting multi-dimensional nested structures

Lucian Stanculescu, Raphaëlle Chaine, Marie-Paule Cani, Karan Singh

Sculpting multi-dimensional nested structures

Lucian Stãnculescu[a,b,c], Raphaëlle Chaine[a,b], Marie-Paule Cani[c,d], Karan Singh[e]

[a] *Université de Lyon, CNRS*
[b] *Université Lyon 1, LIRIS*
[c] *Grenoble Universités, LJK*
[d] *Inria, Grenoble*
[e] *University of Toronto*

Abstract

Solid shape is typically segmented into surface regions to define the appearance and function of parts of the shape; these regions in turn use curve networks to represent boundaries and creases, and feature points to mark corners and other shape landmarks. Conceptual modeling requires these multi-dimensional nested structures to persist throughout the modeling process, an aspect not supported, up to now, in free-form sculpting systems.

We present the first shape sculpting framework that preserves and controls the evolution of such nested shape features. We propose a range of geometric and topological behaviors (such as rigidity or mutability) applied hierarchically to points, curves or surfaces in response to a set of typical free-form sculpting operations, such as stretch, shrink, split or merge. Our method is illustrated within a free-form sculpting system for self-adaptive quasi-uniform polygon meshes, where geometric and topology changes resulting from sculpting operations are applied to points, edges and triangular facets. We thus facilitate, for example, the persistence of sharp features that automatically split or merge with variable rigidity, even when the shape changes genus. Sculpting nested structures expands the capabilities of most conceptual design workflows, as exhibited by a suite of models created by our system.

1. Introduction

Digital modeling for film and entertainment is increasingly powered by the workflow of free-form digital sculpting [1], over traditional CAD modeling using curve and surface operations [2]. While evidently not as fluid and versatile as sculpting, CAD models do provide better support for the creation and editing of feature points, curves or surface regions that are integral to a 3D shape throughout the modeling life-cycle. These surface regions, curves and points mark segments of varying form, material or function, sharp edges, object proportions, annotations and other landmarks on complex shapes. In current sculpting systems these integral structures are missing: they are typically added once a model is finalized, or sometimes transiently used to perform specific modeling operations. We advocate that these features of variable dimension (point, curve or surface see Figure 1) should be present throughout the sculpting process, nested or embedded within each other, and as a result should impact each other's deformation response to free-form sculpting operations.

While such features can readily be created on 3D shapes using current modeling and drawing tools, maintaining their consistency when the object is further edited, deformed or animated is difficult. These problems have never, to our knowledge, been systematically addressed. Prior work has focused on using features to control shape deformation, rather than proposing approaches to deform a shape with embedded features.

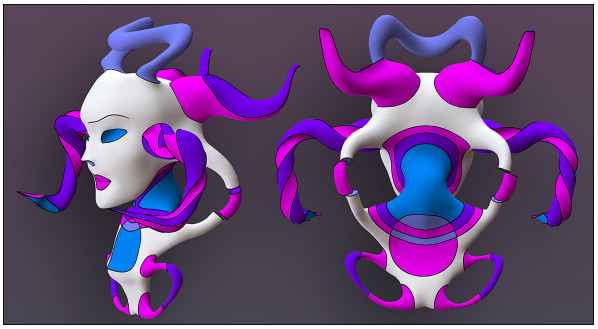

Figure 1: Model with features, created from a plain sphere using several of our tools: sweep deform, twist, geodesic inflate, extrusion, merge, cut and split. It combines smooth, organic-like parts, sharp features and precisely delimited regions. Vertex count: 50k. Modeling time: 1h.

It is precisely, the creation and update of such multi-

dimensional structures, with attributes of geometric rigidity and topological mutability for features, that we aim to capture within a general shape sculpting framework. We represent these structures as multi-dimensional elements embedded within a cell complex: points are 0-D elements, feature curves are chains of edges (1-D elements) and surface regions, open manifolds of connected triangles (2-D elements). We formulate the response to a free-form deformation as a two step process. First, the geometry of the cell complex elements is processed in increasing order of dimensionality (points, curves and then regions), with the feature attributes and geometric fidelity of lower-dimensional features influencing the shape of higher dimensional elements. Second, the topology of the cell complex is updated based on its deformed geometric elements, processed in decreasing order of dimensionality, so that changes in topological connectivity at a given level influences the topology of embedded lower dimensional features. We present this algorithmic framework in the context of a sculpting system for self-adaptive quasi-uniform meshes, where geometric and topology changes resulting from arbitrary operations are applied to points, edges and facets, at interactive rates.

2. Related Work

The corpus of literature on object deformation is extensive. Here, we focus on geometric rather than physically-based deformations, for free-form sculpting. Such research can be broadly categorized as spatial versus object-aware methods.

Spatial deformations: This class spans three decades, from seminal work on lattice deformations [3] to feature curve based controls [4] and volume preserving deformation fields [5, 6]. These deformations define a functional warp of Euclidean space within which any point-sampled structure can be embedded. Various deformers exploit shape features as deformation handles [4], but these features deform the shape rather than being deformed intrinsically with the shape. These methods indeed operate on points, and any structure built upon these points, must be imposed and maintained externally. We present such a scheme for handling multi-dimensional nested structures upon deformation.

Object-aware deformations: Implicit surface inspired approaches to volumetric sculpting represent the sculpted solid object as samples over a voxel grid [7, 8, 9, 10]. Such approaches handle material volume and changes in shape genus elegantly but have an even harder time, both representing and maintaining shape features. Multi-grid approaches can be used to enhance shape with sharp creases [8], but these are easily destroyed by progressive sculpting operations that diffuse and re-sample material in the voxel grid. Further, since the shape is frequently re-polygonized there is no trivial transfer of embedded mesh features between two different tessellations. A recent free-form mesh sculpting system *Freestyle* [11] enabled this topological flexibility of volume sculpting for quasi-uniform meshes. Another vein of object-aware deformations are based on variational methods [12], where the mesh attempts to preserve its local structure when undergoing large scale deformations. Such techniques have also been used in the context of brush-based local surface cloning [13]. Techniques such as *iWires* [14] further detect and annotate feature curves and their geometric inter-relationships (i.e. size, co-planarity) automatically, preserving them under topologically invariant deformation. Our research complements such approaches by providing hierarchical control over the evolution of nested features upon arbitrary deformation, particularly with regards to varying topological connectivity.

Current professional sculpting applications focus on brush-based chisel-like sculpting tools, for deformable meshes with high mesh resolution and constant [1] or adaptive connectivity [15, 16]. These tools can only emulate sharp features using a dense point sampling and transient operations such as extrusion along a curve. There are no explicit safeguards against the subsequent blurring of these features.

Perhaps most relevant in spirit to our research, are approaches that imbibe shape features into the construction history of the modeling process. Two noteworthy examples of this are part based modeling and curve-network based modeling. **Part-based modelers** compose novels shapes from a collection of features and parts [17, 16, 18]. These mesh composition systems are complementary to free-form sculpting and we draw inspiration from their emphasis on the presence of segment regions throughout the modeling workflow; our research can enable interactive part-based modeling to be homogeneously coupled with free-form sculpting without destroying part history. **Curve-network based modelers** such as *FiberMesh* [19], construct a shape from feature curves, that define a shape scaffold over which a mesh is constructed. These modelers, while allowing for manual changes in topology (i.e. drawing of a handle in [19]), are however inflexible to topology changes of both the feature curves and induced mesh, during the deformation.

In conclusion, a variety of modeling methods exist, with varying support for deforming embedded features, but no previous method addressed our problem of en-

abling consistent *deformations and changes of topological genus* of an object with multi-dimensional features.

3. Design goals

We begin by presenting a few motivating examples, to help characterize the desired behavior of a shape with evolving features under deformation. From these examples we observe a set of physically inspired properties that impact the behavior of nested structures, which we model using a small number of structure attributes.

3.1. Case studies

Our goal is a conceptual modeler providing both the *freedom* to build complex organic shapes and the *precision* required to model man-made objects. We thus look at three inspiring examples: a model with salient regions segmenting parts with different material or texture, an organic model and a man-made artefact. Without being exhaustive, these examples highlight desired behavior for a shape with nested features under a series of pertinent operations.

Shape segments. Denoting surface regions is important to demarcate part segments, or simply for decorative purposes. Once regions have been created, the user should be able to further deform the surface while maintaining the regions already created (see Figure 2).

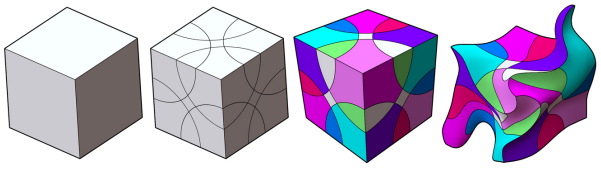

Figure 2: Cube divided into colored regions that are preserved during deformation.

Character modeling. Creating the anthropomorphic character shown, requires the design of the different elements of the face, such as the eyes and the mouth and other landmarks, initially to mark proportions and then later to provide definition to various parts, not only for modeling and texturing but subsequent shape blending and animation. One way to do this is to first draw the outlines of these elements and then use other tools to give them the desired shape (see Figure 3).

Man-made object design:. In the case of man-made or industrial objects such as a cup (see Figure 4), we need to be able to create geometric features and shapes from extruded parts, shelled regions and to unite elements to create new topological features at intersections.

Figure 3: Drawing elements of the face and then deforming around them while preserving their shape.

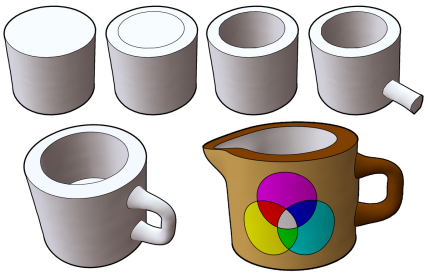

Figure 4: Cup modeled using feature drawing, extrusion, deformation with topological change (handle), decoration.

3.2. Characterizing behaviors

Our first observation from these examples is that the features we need are often surfaces regions, curves on surfaces, or points on curves. In general, these features are located on a support shape of higher dimension. We thus develop feature aware deformation behaviors uniformly in an abstract setting of *(n-1)*-D features located on a support *n*-D shape (such as points on curves, curves on surfaces or surfaces that bound volumes). Other embeddings, such as a feature point on a surface (*(n-2)*-D feature on *n*-D structure), are easily treated as multi-level embeddings, in this example via a degenerate *(n-1)*-D intermediate structure. Note that while we now illustrate desired feature behavior below in terms of curves on surfaces, this behavior generalizes to multi-dimensional structures.

Behaviors. Our motivating examples lead us to a table of desirable behavior depicted in Figure 5. The table informs us on how features (*(n-1)*-D structures on an *n*-D object) should deform and possibly change their global topology when their support deforms. There may be several possible behaviors for the same deformation of the support structure, as for instance the immutable, mutable and erasable feature behaviors when the support shrinks.

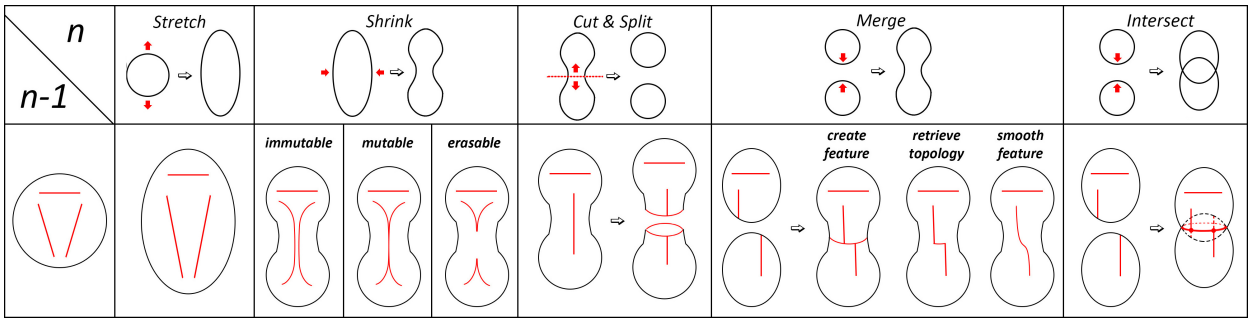

Figure 5: Behavior of features of dimension *n-1* (red) on *n*-structure (black): *(STRETCH:)* support shape and features are stretched according to the displacement field; *(SHRINK:)* when they come close, *immutable* features stay separate, *mutable* features merge into one, *erasable* features merge and disappear; *(SPLIT:)* when the support shape is cut and split, a feature is created to mark the cut, then the feature is duplicated on both splitting shapes; *(MERGE:)* when support shapes merge on a closed loop, a feature of dimension *n-1* is created along the intersection, interior parts of the support disappear, while parts of the new feature can get partially deleted to connect features from different supports into one smooth feature; *(INTERSECT:)* when support shapes intersect, incident parts remain in the structure and a feature of dimension *n* is created along the intersection.

Feature attributes. We characterize a feature's response to a deformation, both topologically and geometrically (as shown in table 5), using the following independent attributes:

Fusibility describes how features interact topologically, in close geometric proximity. We classify features as:
immutable, if they maintain topological independence regardless of their geometric proximity to other features;
mutable, if they topologically merge with others in proximity, or fragment if their support surface stretches;
erasable, if feature annihilate each other locally on contact.

Permeability allows features to geometrically intersect. Impermeable structures merge upon intersection.

Rigidity is a geometric property that controls the deformability of a feature, influencing the overall deformation of the structure in which it is embedded.

Smoothness geometrically limits the maximum curvature of a feature. When this threshold is reached the feature is smoothed to its maximum curvature upon deformation.

Note that all these attributes are easy to tune for a user's purpose since they are independent, and each of them has an intuitive, pseudo-physical meaning.

4. A multi-dimensional nested structure

We now consider the set of structural needs to be satisfied by a model in order to capture the set of behaviors described above. Our solution is presented independent of the dimension of the support-feature pair and then applied hierarchically to all elements of our structure.

4.1. Shapes with features as a cell complex

Given our goal of managing both geometric and topological changes of features that impact each other across multiple dimensions, our insight is to embed these features within a discrete cell complex representation of shape. Practically, for our mesh sculpting system detailed in Section 5, feature points are vertices, feature curves are edge chains and feature regions connected triangles of the polygon mesh model. Formally, our multi-dimensional structure is described by a cell complex [20] that partitions *n*-D space into (*n*-D cells). The boundaries of these cells are *(n-1)*-D cells, which are in turn bounded by lower dimensional cells. Our shape features are defined as a subset of cells across multiple dimensions that define how the geometry and topology of the cell complex will evolve in response to free-from deformations. This cell complex forms our *multi-dimensional nested structure*.

To be used in a sculpting framework, multi-dimensional nested structures must be provided with a *convergent updating scheme*, able to maintain structural coherency under deformation and to restore a set of prescribed geometric and topological properties. We first describe a deformation induced set of operations that update the topology of the cell complex, taking the embedded features into account. We then discuss the order in which these geometric deformations and topological operations are applied.

4.2. Topology updating operations

Large deformation of discretely represented shapes, often requires topology updating operations to improve the fidelity of the representation or to handle undesirable

self-intersections or other shape inconsistencies. Typical atomic update operations required in a sculpting system are cell *division*, *fusion*, *collapse* and *boundary duplicate* (see Figure 6): *Division* is typically used to cut cells that become too large, and may require a division of their boundary cell; *Fusion* merges two adjacent cells, and may or may-not trigger fusion of their lower dimensional boundaries, depending on the structure, shape and size of the surrounding cells; *Collapse* is used to get rid of insignificant, degenerate cells; *Boundary duplicate* is the reverse operation, used to insert a new *n*-D cell by duplicating an existing *(n-1)*-D cell at its boundary.

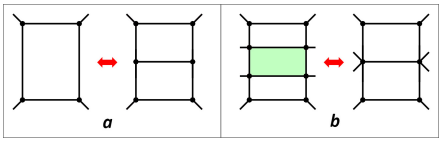

Figure 6: General operations on cells without features: division/fusion (a), collapse/duplicate (b).

These four operations are also essential to handling features: to embed a feature inside a cell (*Division*), to merge cells across disappearing features (*Fusion*), to reduce the number of cells between merging features (*Collapse*), and to allow duplication of mutable features (*Boundary duplicate*). These operations however, need to be feature-aware, in order to maintain the consistency of the multi-dimensional nested structure (see Figure 7):

a) Fusion of two cells cannot be activated if their common boundary is a feature.

b) If the division of a cell induces the division of one of its boundary cells tagged as a feature f, the new boundary cells coming from f are also features.

c) Collapsing a cell between two feature boundaries can induce topological changes in the network of nested features, depending on the fusibility of these features (see Figure 7 *c*): if both are immutable, the collapse cannot be activated (see c.1), if both are erasable, the collapse causes the feature boundaries to vanish, and to be turned into a single ordinary cell boundary (see c.3). In the other cases, the two feature boundaries merge into a single one, behaving as mutable features.

d) If the *(n-1)*-D boundaries merged during a cell collapse include a single feature, then the resulting boundary is a feature. Geometrically, non-feature boundaries always collapse on feature boundaries to ensuring the *positional independence* of features from their support during deformation.

e) A feature can be duplicated by a boundary duplicate operation, possibly with the duplication of its lower dimensional, embedded features.

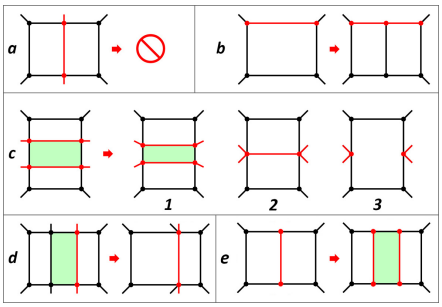

Figure 7: Operations on cells (feature boundaries in red): (a) fusion forbidden across a feature; (b) division; (c) cell collapse with two feature boundaries: (1) forbidden between immutable features, (2) at least one mutable feature or exactly one erasable feature, (3) two erasable features; (d) cell collapse with a single feature boundary;(e) feature duplicate.

4.3. Applying deformations hierarchically

When a free-form deformation is applied to the vertices of a multi-dimensional nested structure, the sculpted shape and features embedded within the cell complex move together. After each such deformation step, the cell complex must be processed to preserve various properties of the deformed shape and its features. First, since the cell complex is a discrete representation of a typically piecewise smooth surface, various feature-aware merging and splitting operations may be required to handle undesirable local intersections and restore geometric fidelity in the discrete approximation of a smooth surface. Note, as a result of this processing, the topological genus of various features may change, even when the genus of its support does not change (such as in the cell collapse case of Figure 7c). Second, the deformation may cause global surface intersections that would need to be merged for impermeable shapes. This would change the topological genus the overall shape, which in turn may alter the topological genus of embedded features across multiple dimensions.

Instead of performing an expensive global optimization of the cell complex subject to the above requirements, we handle this cyclic co-dependence of shape features across multiple dimensions by separating deformation processing into two coherent steps. First, a bottom-up (in increasing order of feature dimension)

step, restores geometric quality subject to geometric and topological feature attributes, independent from the geometry of their support. Then, self-intersections of the overall shape are globally detected, and resulting topology changes are propagated top-down (decreasing order of feature dimension) through the nested structure. These two steps are illustrated in Figure 8 for a permeable 2D solid shape structure and in Section 5 for a surface embedded in 3D space.

It is important that the geometric quality of a feature be processed before that of its support structure: else update of the support could destroy both geometric and topological properties of its embedded features. It is equally important that topology changes due to global shape intersections are processed for a support structure before its embedded features, since merging or splitting features embedded in a cell complex requires that they be topologically connected or unconnected on their support structure (see Figure 9).

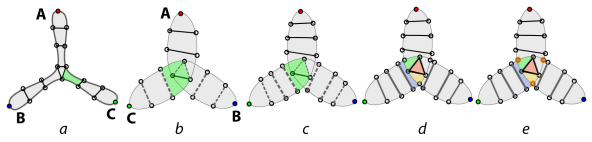

Figure 8: 2D case: the border curve of the permeable shape and the 3 colored vertices (A, B, C) are features. a) before deformation; b) after the lower lobes exchange places by crossing each other and before the refinement process: the green cell now covers other cells and feature lines intersect; c) first update the feature lines without taking care of the intersections of the 2D support (there are still intersecting 1 and 2-cells); d) update the higher dimension 2-cells dealing with their intersection; e) the intersections between feature lines are identified within the border of the intersection of 2-cells, and tagged as feature vertices.

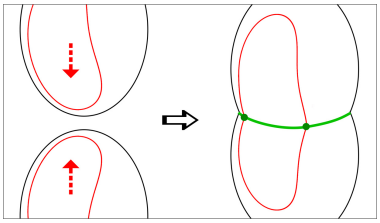

Figure 9: The intersection of two features (in red) crossing in space is included in the intersection (green) of the 2D support.

To solve intersections at a given dimension *n*, we explicitly construct the sub-structure of dimension *n-1* resulting from the intersection and we embed it into the cell complex using the available updating rules, while respecting the constraint features already in place. Then we eliminate or untag the parts of the cell complex that are no longer desirable in the sense specified by support and feature behavior (see Section 3). After a merge, this means that the insertion of the intersection as a new feature can either be made temporal or become permanent. Different parts of this intersection can be, interactively or automatically, according to some criteria, turned to non-features in order to retrieve the desired connection between the already existing features on both side.

5. Sculpting System

We have used the model of *multi-dimensional nested structures* to develop a sculpting system for triangular surface meshes with embedded feature curves and points. In our system, the cell complex is thus a simplicial complex of dimension 2 embedded in 3D space.

5.1. Quasi-uniform meshes with features

In order to promote simplicity and speed while accurately capturing deformations, we choose to maintain a quasi-uniform sampling at all levels of the structure: the quality criteria driving the updating step is that all edges must be smaller than a *detail* threshold, while also trying to keep a length larger than half the *detail*, similar to the refinement scheme described in [11].

The operations used in the resampling-updating step include cell division, fusion and collapse, which here translate into well known mesh operations such as edge split (leading to the division of the two incident triangles) and edge collapse (leading to the collapse of the two incident triangles). In our implementation, no fusion operation is directly provided for mesh triangles, since it would lead to the creation of quads. However, for better stability of the refinement scheme and in order to improve triangle quality, we use the edge flip operation which, in terms of cell-complex operators, translates into the fusion of the two cells incident to the edge followed by a new division of the resulting cell.

In practice, after each deformation step, the updating process first iterates over the mesh edges in order to flip or split them until they all conform to the *detail* maximal length. A single traversal of the edges is then performed to collapse all the edges that are too short. Finally, a new flip-or-split pass is run to insure that no edge longer than *detail* persists. This updating routine is similar to the one used in [11], where convergence is discussed. It promotes high quality triangles without using any additional relaxation step that would tangentially move the vertices, with additional costs.

The *detail* threshold represents the level of detail used for tracking deformations. It also ensures the reactivity of the structure to any kind of deformation since it enables simplified computations in a number of cases such

as approximate geodesic distances, detection of intersections and feature drawing.

Let us now discuss how this updating process is extended to quasi-uniform meshes with features curves:

Features correspond to arbitrary graphs of curves embedded on the quasi-uniform mesh by tagging chains of edges, or arbitrary vertices embedded along these curves. They are given the meaningful attributes setting their behavior introduced in Section 3. Note that being embedded, they can either be smooth curves on the mesh (respectively smooth point of curves), or support sharp features.

Feature-curves inherit the *detail* characteristic of the material, and satisfy quasi-uniformity criteria as well. Their topology seamlessly evolves with the changes in the overall topology of the underlying mesh. As described in our framework, it also self adapts locally, for instance when two curves-segments move toward each other on a shrinking surface, with the local geodesic distance between them locally vanishing. Note that the quasi-uniform sampling of the underlying mesh allows us to detect this situation using the length of the minimal edge path as a good approximation of the geodesic distance.

The mesh operations used in the updating scheme to maintain quasi-uniformity are adjusted, as described in Section 4, to handle situations where feature edges and feature vertices are involved:

- a feature edge cannot be flipped.
- a feature edge can be split into two feature edges.
- a feature edge collapses at mid distance from its two feature vertices.
- a non-feature edge bounded by a single feature vertex can be collapsed on that feature vertex.
- collapse of a non-feature edge bounded by feature vertices depends on the nature of the features. For mutable or erasable features, vertices collapse in the middle of the non-feature edge. For immutable features collapse is forbidden.
- when two erasable feature edges collapse on each other, they turn into a non-feature edge.

Note that forbidding flips on feature edges does not prevent the sampling-updating step to maintain the quasi-uniform property since the flip can be replaced by a split of the feature edge. Similarly, preventing collapse between immutable feature does not affect the convergence of the updating scheme, since the lower bound for the length of the edges should be favored but is not mandatory). Therefore, the convergence of the updating scheme is not affected by the modified updating scheme.

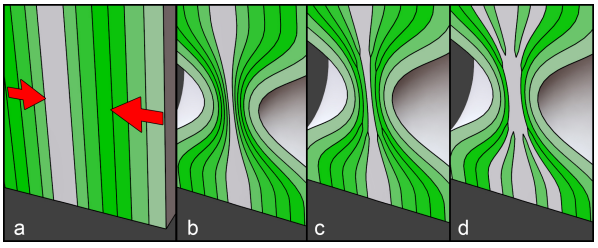

Figure 10: Mutability of features: a) deformation on the original mesh b) immutable features preserve their topology c) mutable features automatically merge at close proximity d) erasable features merge and disappear.

As prescribed by the algorithmic framework in Section 4, this updating scheme is performed bottom-up at each deformation step, to track and update the position of the features first, and of the support mesh afterwards.

The intersections of the surface are then detected, by identifying pairs of closed triangle strips around each geometric line where two surfaces intersect. We then operate a combined march over these strips to detect the segments resulting from triangle pairs intersections. We insert those segments into the mesh by splitting each edge that intersects a triangle and inserting a corresponding vertex in the respective triangle. The vertex insertion is restricted to this use and it corresponds to a cell division on the corresponding cell complex. The insertion of the intersections in the mesh is naturally followed by an updating step to restore its quasi-uniform properties, and by a removal of the interior parts (if necessary according to the selected surface behavior), as shown in Figure 11. For simplicity we have chosen to always mark the intersections as features and interactively untag them if needed.

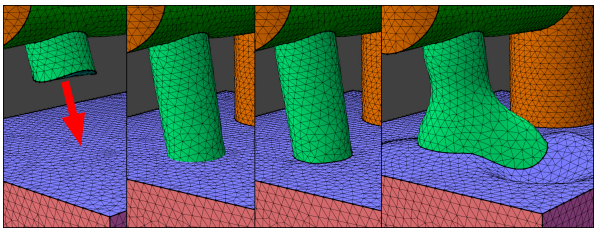

Figure 11: Change in topology on an impermeable mesh: (1) deformation on the original mesh (2) deformation causes self-intersection (3) the intersection is computed and embedded as a feature line in the mesh and interior parts are deleted (4) the new surface can be deformed while preserving the new feature that marks the intersection.

5.2. *Creating features*

The features can be created by drawing them on the surface, by intersecting the mesh with other geometry (e.g. planes, spheres) or where topological changes arise as we have seen in Section 4. We can even use the detection of sharp feature lines on models to initialize the network of curves, for which plenty of methods already exist [21].

We focus here on describing a simple feature embedding technique on quasi-uniform meshes. The curve sketched by the user is projected on the fly on the surface, upon a simple walk, with the insertion of new vertices by splitting the edges intersected by the curve and marking the edges connecting them as feature edges. This eliminates the need to use overly complex intersection algorithms and still creates continuous curves on the surface. Meanwhile, the feature-mesh structure locally self-adapts to keep its quasi-uniform properties. When the drawn curve passes over another feature curve, the two features will automatically connect on a feature point, since all features are embedded in mesh edges. This results in a quick embedding of arbitrary feature curve networks as shown in Figure 12.

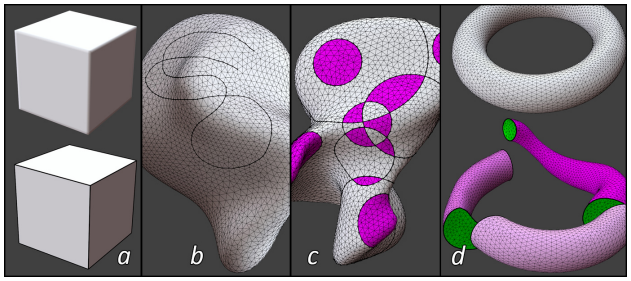

Figure 12: Features can be created by detection (a - edges of cube), freehand drawing on the surface (b), intersection with primitives (c), topological changes (d).

5.3. *Sculpting with arbitrary deformation fields*

Some of the most important and frequently used tools for deforming meshes are space deformations. They can easily be implemented as vector fields that directly change the position of vertices. Depending on the desired behavior, they can either preserve the volume throughout the deformation or they can add (or remove) material from the object in order to extend the shape beyond its current volume.

The space deformation we use are sweep preserving volume [6] or not, grow, flatten and twist [11, 1]. We also consider surface-based deformations that correspond to vector fields defined on vertices exclusively, since they depend on the differential properties of the surface : smooth, inflate, deflate [11, 1]. Note that the rigidity property of features can locally act on the amplitude of the applied field.

There is no necessary limit on the maximum magnitude for one vertex displacement. If some intersections of the shape or the features occur, they will be inserted and tagged as features or not, with some parts of the mesh removed or not, depending on the permeability of the support and the properties of the features. However, setting a maximum magnitude to half the *detail* is useful so that local cell inverting are directly resolved by edge collapse, without any additional complex algorithm.

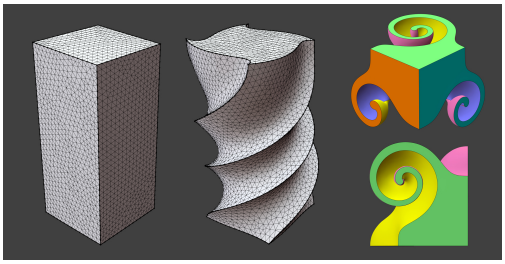

Figure 13: Box deformed by a twist (rotational field with radially decreasing intensity) applied on the top face, and colored cube deformed with twists around its edges. The already defined immutable (and sharp) features are preserved. The triangles maintain a good aspect ratio due to local splits, flips and collapses, without the use of any complex processes.

5.4. *Feature-Specific Deformations*

We have also defined new fields that are specifically conceived to take features into account, independently of their rigidity.

Feature curves can set the region of influence of a deformation. These closed or open contours are used to limit a geodesic weighted field such as the inflation-deflation represented in Figure 14. Since speed is paramount for a sculpting application, the *approximate geodesic weights* are calculated using a walk from the selected point on the surface, along the quasi-uniform edges of the mesh. Calculated weights can further be averaged among neighbors, to remove the noise due to the edge-walk and get reasonably smooth influence regions. Closed features can also be used to perform sharp extrusion of their interior, by duplication of their boundary (an operation allowed on mutable features) and subsequent limitation of the support of the extrusion field to the interior of the loop.

Feature dependent tools (smoothing, inflation, extrusion etc.) can introduce sharp features on the surface that will later be preserved by space deformations.

When dealing with features of constrained smoothness, we cannot directly determine the final position of a

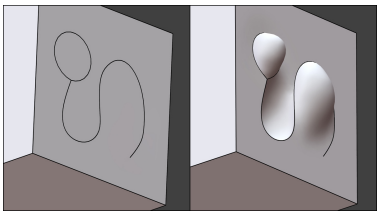

Figure 14: Geodesic weighted inflate and deflate operations applied inside closed contours or on one side of an open feature.

feature vertex after applying a deformation. Therefore, smoothness must be applied as a post-process where the features restore their smoothness through a relaxation variational step.

5.5. Results and discussion

The pictures and video illustrate the utility of nested structures during conceptual modeling. In this respect, the artist who created the example in Figure 15 was able to draw correctly proportioned facial features at the conceptual design stage and carry them through the modeling process in a coarse to fine workflow. Figure 1 illustrates the topological mutability of features, allowing geometric control of cracks, wrinkles and other physically inspired behavior. Similarly, Figure 4 illustrates the ability to preserve feature attributes through a series of genus altering deformations, another affordance that enriches the modeling workflow.

In general our system supports both a variety of nested feature attributes and a suite of free-form sculpting tools that operate agnostic of any nested mesh structure. The combination of these also supports versatility: while many existing systems may require operations to be performed in specific order such as defining a sharp feature after shape smoothing, our system can perform these operations in any order, since the sharp feature is able to persist through the smoothing operation.

Our implementation handles the deformations of the example shapes provided here (less than 100 k vertices) at interactive rates, without using any GPU computations or optimizations. There is thus, still room for efficiency improvements. The speed of our system stems from the simplicity of our algorithmic steps over solutions that perform non-linear relaxations or solve large global systems. Our mesh with nested features, responds quickly and locally to any arbitrary deformation resolving nested structures hierarchically. This process takes a negligible fraction of the overall deformation time. At each time step, proper sampling is restored by iterating over the deformed edges, and so the adaptive updating routine is linear in the number of edges, when the maximal enlargement of an edge remains bounded by a constant multiplied by the *detail*. Self-intersections are detected in a lazy manner (from an application point of view they could also be triggered after a deformation is completed).

As stated in Subsection 5.1, our choice for a quasi-uniform sampling was driven by simplicity, speed and directness of implementation, but the uniformity constraint could be relieved since we do not use it to anticipate collision events as was the case in *Freestyle* [11]. Instead, our work promotes the creation of intersections to create new features. In that case, the updating scheme can be replaced by a scheme based on the quality of the triangles exclusively and not on their size. Therefore other methods can be envisioned, such as the adaptive sampling technique used in [15], without any change in how we treat features. Our choice was simply an effective way to concentrate on the multi-dimensional structure, instead of focusing on more complex adaptive refinement techniques requiring relaxation. The disadvantages of the quasi-uniform mesh are mostly the same as those found in [11]: lack of precision below the *detail* level, oversampling of low-detail regions, but, in addition, the sampling also has an impact on how we draw features and their detail.

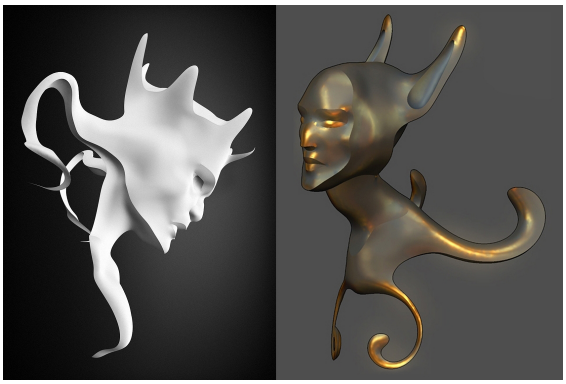

Figure 15: Faces with feature proportions established at the beginning of the modeling process.

Our approach does not directly compare to mesh editing methods such as Laplacian editing, in the sense that it can be seen as an additional module that could be added on top of this type of approach. Regarding the comparison with methods that are founded on the modeling of features such as *FiberMesh* [19], our approach has the advantage that the update operations are more natural and less restrictive on the non feature parts, since modeling of features is fully harmonized with the modeling of surfaces.

Figure 16: Object created with repeated extrusions from a cube, sweep deformations to obtain the shape, geodesic deflate to emphasize the region borders. Modeling time: 20min. Vertex count: 30k.

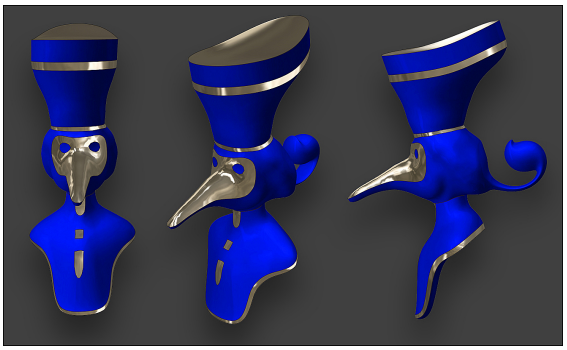

Figure 17: Object sculpted and decorated with different materials.

6. Conclusion

We have presented a systematic approach to representing and deforming shapes with nested features in response to arbitrary free-form deformations common in conceptual modeling. We identified a set of independent attributes for capturing the topological and geometrical behavior of lower dimensional features embedded in a higher dimensional support, and described a simple way to maintain them throughout the design process, using a hierarchical rule based method for applying deformations. We illustrated this theory within a free-form sculpting system for meshes with region, curve and point features. These feature exhibit variable geometric rigidity and topological mutability in response to arbitrary deformations and can handle genus changes of both the high level mesh and embedded features. These features can also be used to define deformation parameters like a radius of influence. While the shapes with features we exhibit can arguably be modeled using prior art, they have a degree of interoperability and construction history in our system, that was lacking in existing systems.

Finally, our current implementation has limitations that could be addressed by future work: We can augment our deformation tool-set to define feature specific deformations that would make features slide within their support, and which could lead to real crossing of permeable or mutable features while preserving the topological genus of their support. This cannot be obtained with global deformations fields which are fold-over free for the features. Deformation fields defined within the support structure, such as along a manifold surface would also enable us to better control and edit the lower dimensional features independently from the parent structure. This degree of multidimensional shape editing would require a more general and loosely coupled design of nested features than presented in this paper.

Another direction is to apply our framework to non-orientable surfaces. In principle the quasi-uniform framework and the algorithms dealing with deformations and intersections should generalize well to the new demands, however the suppression of certain parts in the case of impermeable surfaces becomes a problem and other technical problems might also arise because of the non-orientable mesh structure.

We have implemented a basic framework for deforming multi-dimensional structures with features. However, although our implementation in a sculpting system offers sufficient control to create the desired free-form shapes as shown by the examples, we might also want to adapt our method to a different interaction technique (such as those used in variational methods where the deformation of features induces the automatic adaptation of the surface) or to a more exact framework, required for instance for industrial objects, and be able to impose certain properties like the curvature of the surface or a fixed dihedral angle across the features, or to fix the length or limit the curvature of features. This can be achieved by implementing variational methods similar to the ones presented in *FiberMesh* [19] or *iWires* [14] *on top* of the current framework.

Acknowledgements

This work was funded by the Région Rhône-Alpes (LIMA project) and the ERC advanced grant EXPRESSIVE.

References

[1] Pixologic, ZBrush.
URL `www.pixologic.com`

[2] Autodesk, Maya.
URL http://usa.autodesk.com/maya

[3] T. W. Sederberg, S. R. Parry, Free-form deformation of solid geometric models, SIGGRAPH Comput. Graph. 20 (4) (1986) 151–160. doi:10.1145/15886.15903.

[4] K. Singh, E. Fiume, Wires: a geometric deformation technique, in: Proceedings of the 25th annual conference on Computer graphics and interactive techniques, SIGGRAPH '98, ACM, New York, NY, USA, 1998, pp. 405–414. doi:10.1145/280814.280946.

[5] A. Angelidis, M.-P. Cani, G. Wyvill, S. King, Swirling-sweepers: Constant volume modeling, in: 12th Pacific Conference on Computer Graphics and Applications, Pacific Graphics 2004, October, 2004, Seoul, South Korea, 2004, pp. 10–15.

[6] W. von Funck, H. Theisel, H.-P. Seidel, Vector field based shape deformations, in: ACM SIGGRAPH 2006 Papers, SIGGRAPH '06, ACM, New York, NY, USA, 2006, pp. 1118–1125. doi:10.1145/1179352.1142002.

[7] T. A. Galyean, J. F. Hughes, Sculpting: An interactive volumetric modeling technique, in: Computer Graphics (Proceedings of SIGGRAPH 91), Vol. 25, 1991, pp. 267–274.

[8] E. Ferley, M.-P. Cani, J.-D. Gascuel, Resolution adaptive volume sculpting, Graph. Models 63 (6) (2001) 459–478, special Issue on Volume Modelling.

[9] G. Dewaele, M.-P. Cani, Interactive global and local deformations for virtual clay, Graph. Models 66 (6) (2004) 352–369, special Issue: Pacific Graphics 2003.

[10] Pilgway, 3D-Coat.
URL www.3d-coat.com

[11] L. Stanculescu, R. Chaine, M.-P. Cani, Freestyle: Sculpting Meshes with Self-Adaptive Topology, Computers & Graphics 35 (3) (2011) 614–622. doi:10.1016/j.cag.2011.03.033.

[12] M. Botsch, O. Sorkine, On linear variational surface deformation methods, IEEE Transactions on Visualization and Computer Graphics 14 (2008) 213–230. doi:doi.ieeecomputersociety.org/10.1109/TVCG.2007.1054.

[13] K. Takayama, R. Schmidt, K. Singh, T. Igarashi, T. Boubekeur, O. Sorkine, Geobrush: Interactive mesh geometry cloning, Computer Graphics Forum (proceedings of Eurographics) 30 (2) (2011) 613–622. doi:10.1111/j.1467-8659.2011.01883.x.

[14] R. Gal, O. Sorkine, N. J. Mitra, D. Cohen-Or, iwires: an analyze-and-edit approach to shape manipulation, ACM Trans. Graph. 28 (3) (2009) 33:1–33:10. doi:10.1145/1531326.1531339.

[15] T. Pettersson, Sculptris.
URL www.sculptris.com

[16] R. Schmidt, K. Singh, meshmixer: an interface for rapid mesh composition, in: ACM SIGGRAPH 2010 Talks, SIGGRAPH '10, ACM, New York, NY, USA, 2010, pp. 6:1–6:1. doi:10.1145/1837026.1837034.

[17] R. Schmidt, K. Singh, Sketch-based procedural surface modeling and compositing using Surface Trees, Computer Graphics Forum 27 (2) (2008) 321–330, proceedings of Eurographics 2008.

[18] S. Chaudhuri, E. Kalogerakis, L. Guibas, V. Koltun, Probabilistic reasoning for assembly-based 3D modeling, ACM Transactions on Graphics (Proc. SIGGRAPH) 30.

[19] A. Nealen, T. Igarashi, O. Sorkine, M. Alexa, Fibermesh: designing freeform surfaces with 3d curves, in: ACM SIGGRAPH 2007 papers, SIGGRAPH '07, ACM, New York, NY, USA, 2007. doi:10.1145/1275808.1276429.

[20] G. Vegter, Handbook of Discrete and Computational Geometry - J.E. Goodman and J. O'Rourke eds, CRC Press LLC, Boca Raton, Florida, 1997, Ch. 28 - Computational topology, pp. 517–536.

[21] S. Yoshizawa, A. Belyaev, H.-P. Seidel, Fast and robust detection of crest lines on meshes, in: Proceedings of the 2005 ACM symposium on Solid and physical modeling, SPM '05, ACM, New York, NY, USA, 2005, pp. 227–232. doi:10.1145/1060244.1060270.