Computer-Aided Design and Applications www.cadanda.com

Shaping with Deformations

Nier Wu and Horea T. Ilies

University of Connecticut, niw05001@engr.uconn.edu, ilies@engr.uconn.edu

ABSTRACT

Parametric modeling changed the way we design and reason about mechanical artifacts due to the ability to handle (mainly 2-dimensional) constraints imposed on geometric entities. However, the inherent history dependence of parametric models limits the changes that can be achieved with a given parameterization.

We propose a new history-independent approach to edit geometric models via a geometric deformation procedure that relies on motion interpolation to define and control local geometric modifications. Our approach generates useful geometric deformations such as stretching, bending, and twisting of existing point sets without relying on pre-existing parameterizations or on specific representations of the geometry. The proposed approach provides a small set of parameters that allow direct control and editing of the geometric modifications, and can preserve important geometric invariants such as constant cross-sectional properties of the deformed models. Furthermore, our shape deformation paradigm complements emerging direct boundary manipulation capabilities, and maintains the ability to perform parametric optimization of the associated models.

Keywords: geometric modeling, parameterization, deformation
DOI: 10.3722/cadaps.2011.249-268

1 INTRODUCTION

1.1 Motivation

It is a common practice in a typical industrial activity to modify new or existing designs either as part of the design "evolution," or to accommodate new functional requirements. The corresponding geometric changes occur throughout the mechanical design process and are aimed at addressing a variety of functional deficiencies of the original designs that range from physical performance to manufacturing and assembly, ergonomics, or even service requirements. Unfortunately, the majority of these design changes are not known or predictable when the original geometry is being created.

The need to perform geometric modifications of engineered artifacts has fueled the development of parametric solid modeling technology that allows interactive and incremental definition and manipulation of geometry through a high level parametric layer. The constraints imposed on a 2-dimensional sketch induce a system of non-linear equations whose solutions are the shapes satisfying the prescribed constraints [14]. Thus, an entire class of shapes can be controlled by a specific geometric parameterization, so that each member of this class can be later instantiated by choosing specific parameter values. Parametric models also play a pivotal role in design optimization, which seeks to determine a set of optimal design parameters minimizing a prescribed objective function.

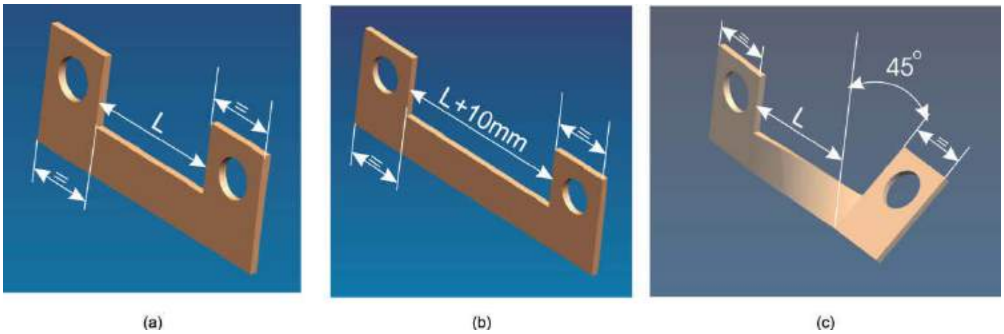

Fig. 1: Three simple brackets those are geometrically similar. Current CAD systems support parametric models that can change bracket (a) to (b), but not bracket (a) to (c) because bracket (c) is not in the same "parametric family" [33] with brackets (a) and (b).

Without a doubt, parametric modeling systems revolutionized the way boundary representations (B-reps) are created and manipulated within a CAD system. However, their industrial use has been hindered by two important classes of limitations. Parametric models are history dependent in the sense that defining the parametric model dictates the types of changes that are allowed by the prescribed parameterization. In most cases, however, only few design changes can be anticipated at the time when the parametric model is being set up. Furthermore, even single parts can require a potentially large number of parameters that describe their geometry as is the case of a jet engine turbine blade. In turn, this affects the robustness of the corresponding geometric updates.

As a result, many design engineers still prefer to model and update the geometry without relying on parametric modeling technology. Consequently, commercial CAD systems have very recently developed, as discussed in the next section, geometry creation and manipulation paradigms that do not rely on prescribed parameterizations or model history [37], [35].

1.2 Prior Art

Many existing approaches that perform shape editing without relying on the parametric modeling framework have been published in the literature. For example, limited geometric editing tools, such as several surface tweaks, are implemented via direct boundary manipulation in practically all high-end commercial CAD packages and geometric kernels. There are other proposals aimed at performing direct "geometric feature" editing, that is independent of the history tree of the part. For example, one approach based on selective volume decomposition coupled with Boolean operations has been recently described in [44], and is aimed at isolating protruding geometric features from the original solid model that are then repositioned and recombined with the solid model that they have been extracted from.

Multiple Free Form Deformation (FFD) techniques that use a variety of control lattices have been proposed, such as [30], [6], [24], [22]. These methods construct bounding volumes that contain the models that are being deformed, and map points between the original and deformed geometries, but they lack the ability of enforcing specific constraints of the deformed geometries, such as those discussed in section 2.

An interesting approach to surface styling based on subdivision surfaces is presented in [3] and implemented in [37]. While subdivision surfaces offer a very powerful paradigm for direct surface manipulation, they are not immediately applicable to shape editing of mechanical parts because preserving some of the desired geometric invariants appears to be problematic. A sweep based deformation approach has been proposed in [45] for performing deformations of mesh models by manipulating their one-dimensional skeleton along with a non-linear radius function. While the resulting deformations are 'intuitively plausible', they do not preserve geometric invariants relevant in engineering applications. A set of control curves, or wires, assumed to represent key structural features of the shape are used in [10] to edit the geometry and preserve the topology of the deformed geometry via an optimization procedure that attempts to preserve the mutual relationships between these wires. This is a powerful editing paradigm that can potentially provide a useful mechanism for manipulating engineering models. As the authors suggest, their procedure needs to be augmented in order to reconstruct shapes that satisfy constraints needed in engineering contexts.

There is a large amount of work in computer graphics focused on the deformation and morphing of geometric objects (see [25], [23], [13], [12], [1] for some recent surveys). For the most part, these approaches are less concerned with geometric invariants that are critical for manipulating the geometry of mechanical parts, e.g., cross-sectional properties, than with continuity of the morphing process itself, and therefore cannot be immediately applied to our class of problems.

Several approaches address the problem of creating new geometry that smoothly connects existing geometric entities. Blending and filleting are traditional geometric operations that have been available in practically all CAD systems and are used to generate new surface patches with prescribed (tangent or curvature) continuity between existing boundary patches of a model. Generating smooth surface approximations between control curves has been investigated, for example, in [43], [31]. Constructing curved handles between two star-shaped planar faces of the boundary of a meshed solid, in essence a form of morphing between polygonal faces, has been investigated in [36], where a Hermite curve connecting the centroids of (possibly different) polygonal faces is used together with a simple shape blending algorithm and linear angle interpolation to generate a set of intermediate polygonal faces that are then connected in a topologically consistent manner.

The development that most closely relates to the scope of this paper is the Synchronous Technology recently announced by Siemens PLM Software [35] that allows the user to assign and modify new dimensional parameters that drive the shape of an existing model. While the technical details have not been made public, Siemens' Synchronous Technology appears to couple direct boundary editing with feature recognition capabilities to enable specific feature operations, such as relocation and re-sizing, in a manner that is independent of the model parameterization and history. Importantly, our approach presented in this paper can nicely complement a capability such as that announced in [35] by extending the possible modifications to include more advanced stretching/shrinking, bending and twisting deformations that can preserve specific geometric properties.

1.3 Scope and Outline

In this paper we propose a new approach to perform local geometric edits of a geometric model by introducing a geometric deformation procedure that relies on motion interpolation algorithms to define and control the deformed geometry without relying on pre-existing parameterizations or specific representations of the geometry. Specifically, our approach provides the ability to per form a set of local deformations commonly encountered in engineering practice, such as stretching, bending and twisting, that are independent of the creation history of the part, and do not require existing geometric parameterizations. Our approach can be implemented for any geometric representation that supports surface-model intersections, including representations based on standard NURBS-based B-reps, implicit functions, point-sampled surfaces [27], as well as simple point clouds. Furthermore, we show that our deformation procedure can preserve important geometric invariants such as cross-sectional properties of the deformed models.

We discuss the details of our approach in section 2, and we illustrate its capabilities and limitations in section 3 through a number of examples of geometric edits that operate not only on boundary representations, but also directly on point clouds. We discuss in section 4 the continuity of the deformed shape, and conclude in section 5 by summarizing the importance and limitations of this work.

2 FORMULATION: SPLIT, SWEEP AND JOIN

We argued that a typical design process requires frequent modifications of the designed geometry. Clearly, there are many strategies that one can employ in order to produce such geometric modifications. However, in the context of mechanical design and manufacturing, any viable approach needs to possess specific properties, while being able to preserve geometric invariants of the original solid models. Specifically, any such modification procedure should

- preserve:
  - the type/degree of some surfaces often encountered in practice, e.g., linear or second order surfaces since low degree surfaces are usually linked to manufacturing capabilities and requirements. For example a planar surface before deformation should remain planar after the deformation during a stretching/shrinking deformation. Observe that a planar surface may not remain planar during a bending deformation, unless the normal of the surface is parallel to the bending axis;
  - cross-sectional properties (such as shape, area, planarity) of the original geometry after deformation because many mechanical properties are influenced by cross-sectional geometry;
  - or report a message that this cannot be done for the prescribed modification;
- be capable of performing the deformation independently of any existing parameterization of the model;
- maintain the solidity of the deformed geometry;
- support a highly intuitive interface for the typical user.

Moreover, the capability to optimize the resulting (deformed) geometry should continue to be supported by a descriptive set of parameters that control the deformation.

In this paper we formulate an interactive geometric deformation procedure that accomplishes three types of geometric modifications that are often encountered in practice, namely: stretching/shrinking, bending, and twisting of the geometry, as well as combinations of these individual deformations with constant or prescribed variation of the geometry of the cross-section along the deformation. Our approach is aimed at creating primarily local deformations such as those illustrated in section 3, since the vast majority of geometric edits required in practice can be expressed as local rather than global deformations.

The approach proposed in this paper consists of three steps. First the rigid subsets of the original geometry that remain unchanged (up to a rigid body transformation) are isolated from the original geometry and, if needed, repositioned to their new spatial configuration. Next, we generate the geometry connecting these rigid subsets, or the deformed geometry, subject to specific constraints such as those outlined above. Finally we join the deformed and the rigid subsets into one single connected set. The next sections discuss each of these steps in more detail.

2.1 Synthesizing the Deformed Geometry

There are multiple strategies with different levels of interactivity and automation that one can employ in order to isolate the rigid subsets of the solid model. For example, volume decomposition is critical in feature modeling and feature recognition where solid volumes are linked to shapes whose recurrence is induced by specific design or manufacturing considerations. The approaches to and limitations of feature recognition algorithms that are actively pursued by the research community are described in [32], and are outside of the scope of this paper. Importantly, our geometric deformation procedure can work in conjunction with or independently of geometric feature recognition algorithms.

In this paper, we assume that the geometric deformations are prescribed and evaluated interactively by the user. The separation of the original geometry into rigid subsets is achieved by splitting the solid model with two, possibly identical, separating surfaces prescribed by the user into three subsets as illustrated in Figures 2, 3 and 4. Note that the separating surfaces are assumed to be homeomorphic.

Denote by h_1 and h_2 the two separating surfaces that split a solid S into three disjoint subsets: two *rigid subsets* RS_1 and RS_2, and the *deformable subset* DS of S that is being deformed. Depending on the choice of separating surfaces h_1 and h_2, we focus on three particular cases that produce deformations commonly encountered in practice as illustrated in Figures 2 – 4.

Case 1: $h_1 = h_2$ **are planar separating surfaces, and therefore** $DS = \varnothing$

When the two separating surfaces are planar and identical (i.e., we effectively use only one separating surface to split solid S), we employ the following procedure for locally deforming the original solid (Figure 2):

- **Step 1**: Choose planar separating surface h and use h to split S into two disjoint subsets, RS_1 and RS_2. This operation results in two identical, not necessarily connected faces $f_1 \in RS_1$ and $f_2 \in RS_2$;

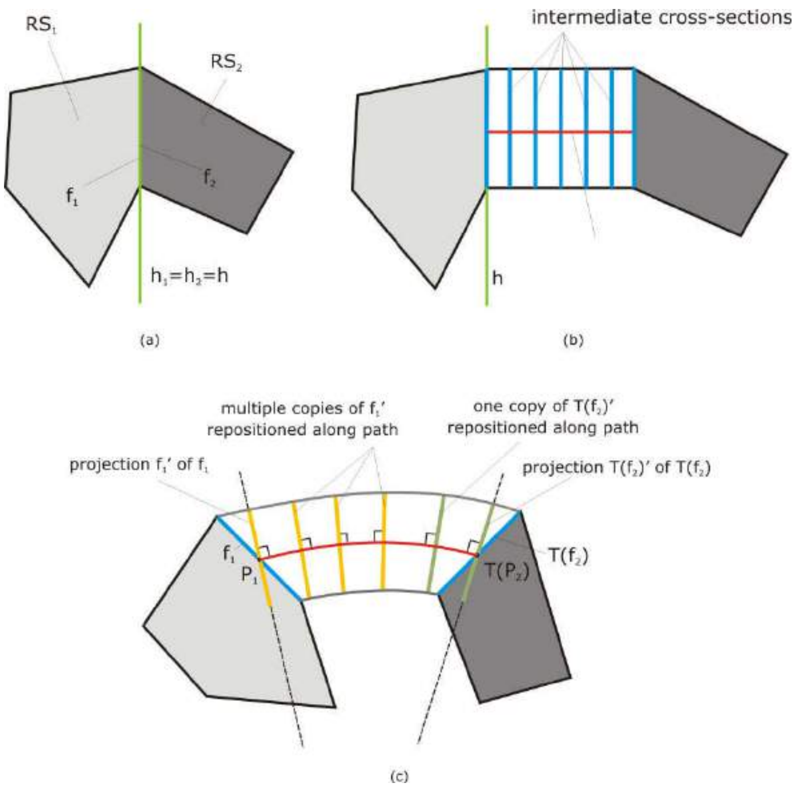

Fig. 2: (a) Splitting a solid with one planar separating surface (i.e., $h_1 = h_2$), which results in an empty deformable subset $DS = \varnothing$, and two new identical faces f_1 and f_2; (b) illustrates the stretching of the original part; (c) shows a more general deformation for Case 1.

- **Step 2**: Prescribe the rigid body transformation T and the type of deformation (i.e., stretching, bending or twisting) that would displace RS_2 relative to RS_1 and compute the transformed rigid subset $T(RS_2)$;

- **Step 3**: Use motion interpolation algorithm (see section 2.2) and the prescribed deformation type to obtain a smooth rigid motion $M(t), t \in [0,1]$, that takes RS_2 from its original position and orientation to its new configuration $T(RS_2)$
- **Step 4**: Sample M to obtain n discrete configurations $M_i, i = 0, \cdots, n-1$ such that $M_0 \equiv I$, where I is the identity transformation, and $M_{n-1} \equiv T$. Then, apply transformations to any point P_1 of f_1, and obtain points on the trajectory of P_1;
- **Step 5**: Construct the sweeping path by interpolating the generated points;
- **Step 6**: Compute the scaling function for face f_1 and generate intermediate profiles (Figures 2(b) and (c));
- **Step 7**: Sweep f_1 into $T(f_2)$ subject to the intermediate profiles to generate the deformed geometry, and perform regularized Boolean unions with RS_1 and $T(RS_2)$.

Note that the steps involving the scaling function computation and the motion interpolation (the latter is discussed in more detail in section 2.2) are probably the most delicate among all steps of this algorithm. To compute the scaling functions of the intermediate profiles it is useful to observe that a solid constrained by a set of intermediate planar profiles is also known as a generalized cylinder, originally proposed in [2] and studied extensively ever since. Consequently, the scaling functions could be computed by using, for example, the approaches discussed in [38], [46], [5] in the field of image understanding. However, a straight-forward approach, which behaves quite well in practice and takes advantage of the fact that $f_1 = f_2$ at $t = 0$, is illustrated in Figure 2(c). First, one computes the projection of f_1 and $T(f_2)$ onto two planes that are perpendicular to the path at P_1 and $T(P_2)$ respectively to obtain planar profiles $f_1^{'}$ and $T(f_2)^{'}$. The last step is to replicate $f_1^{'}$ $(n-1)$ times along the sweeping path such that each such profile will be in a plane normal to the sweeping path while the profile orientation in that plane is controlled by the interpolated motion. One instance of $T(f_2)^{'}$ is also positioned along the sweeping path to generate the n^{th} intermediate profile. Finally, these intermediate profiles are used as constraints while sweeping face f_1 into its congruent counterpart $T(f_2) \subset T(RS_2)$.

Our experiments show that most practical stretching, bending and twisting deformations can be achieved by using uniform distributions of the intermediate profiles along the sweeping curve, although such a distribution is not required as can be seen in Figure 8. On the other hand, it is clear that the number and the distribution of the intermediate profiles will influence the resulting shape of the deformed geometry, which implies that the relationship between the intermediate profiles and the resulting geometry needs to be better understood. One alternative is to treat n and the distribution as additional degrees of freedom and couple them with optimization algorithms that adjust these parameters to optimize specific geometric properties such as curvature changes of resulting shape.

Case 2: h_1 are planar separating surfaces, therefore $DS = \varnothing$; f_1 and f_2 may be non-congruent

When $h_1 = h_2$ are planar but not equal, the deformable subset DS is non-empty and f_1 and f_2 are generally non-congruent, so the procedure outlined above needs to be modified. In general, the set DS will not have constant cross-sections, and in this case, we have (see Figure 3(a) and (b)):

- **Step 1**: Choose planar separating surfaces h_1 and h_2 and split S into two disjoint subsets, RS_1 and RS_2 and a deformable subset DS;

- **Step 2**: Generate *m* planes that are normal to some 'representative' curve of the deformed subset DS (e.g., an edge, symmetry axis or the curve skeleton [7]) and intersect these planes with set DS. This will result in m new intermediate faces F_i;

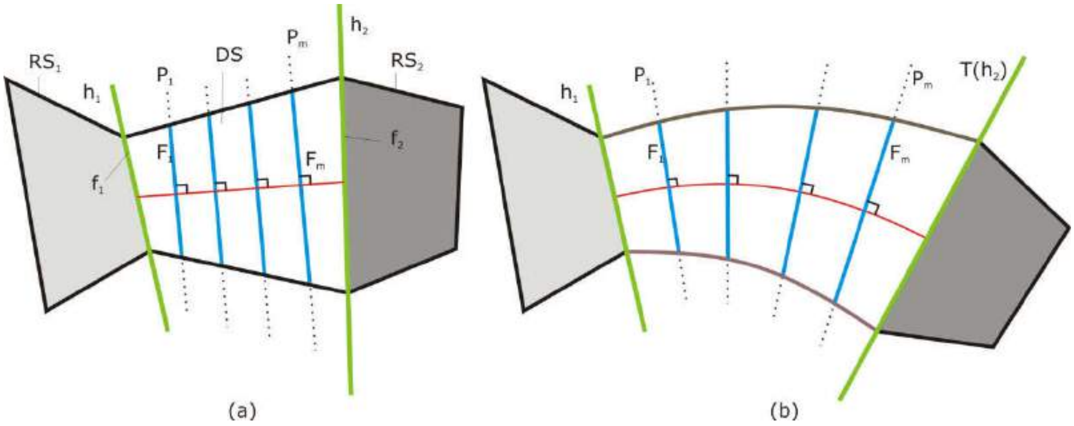

Fig. 3: Splitting a solid with two planar separating surfaces $h_1 \neq h_2$, which results into a deformable subset $DS \neq \varnothing$ and in two generally non-congruent faces f_1 and f_2; the computed cross-sections shown in (a) are repositioned in (b) according to the prescribed deformation.

- **Step 3**: Prescribe the rigid body transformation T and/or the type of deformation that would displace RS_2 to its final configuration and compute $T(RS_2)$;
- **Step 4**: Use motion interpolation algorithm and/or the deformation type to obtain a new smooth rigid body motion M_2 that takes h_1 into $T(h_2)$; apply M_2 to point $P_1 \in f_1$ to generate m points to obtain its "deformed trajectory", and interpolate the deformed trajectory points to obtain the deformed trajectory curve;
- **Step 5**: Compute the m planes perpendicular to the deformed trajectory of P_1, and position faces F_i in these planes according to the interpolated motion;
- **Step 6**: Sweep f_1 into $T(f_2)$ subject to the intermediate profiles F_i to generate the deformed geometry, and perform regularized Boolean union with RS_1 and $T(RS_2)$.

Defining the scaling functions for shapes for which DS has a variable cross-section is an ill-posed problem without requiring additional information from the user. Determining what information is required from the user, as well as the type of information that is most appropriate for computing scaling functions is currently an open problem. For the purposes of this work, we assume that Case 2 deformations have unit scaling functions, that is, the original geometry of the deformable subset DS is discretely preserved through the cross-sections F_i.

Case 3: h_1 and h_2 are piecewise linear with the same number of planar segments.

Planar separating surfaces can provide a set of useful geometric editing capabilities, which, in turn, can be extended even further by using non-linear separating surfaces. However, commercial geometric kernels have limited support for morphing between non-planar cross-sections even though this problem has been studied extensively in the literature (see for example [16], [36], [13]).

Alternatively, one can use a piecewise linear approximation of the non-linear separating surfaces, and sweep the piecewise linear face f_1 into the corresponding piecewise linear face f_2 as described next.

If the two separating surfaces h_1 and h_2 are piecewise planar having p planar segments, as illustrated in Figure 4, then one can reduce the synthesis problem into p sub problems corresponding to pairs of planar subsets of h_1 and h_2. Note that two adjacent planar subsets of each such a separating surface meet along an oriented line in space as indicated in Figure 4. Thus, one can generate ruled surfaces for pairs of corresponding lines (for example between lines L_1 and L_2 in Figure 4) and use these surfaces to split the deformable subset DS into p subsets to obtain the p sub problems. In this case, each sub problem would be approached through one of the two procedures outlined above depending on whether the two separating surfaces are identical (i.e., $h_1 = h_2$) or not. Observe that it may be possible to obtain disconnected sets (gaps between the neighboring sets) as a result of sweeping individual planar subsets of f_1 and f_2, but the connectedness of the result can be enforced discretely at intermediate configurations, which behaves quite well in practice. On the other hand, this connectedness could be enforced continuously by using algorithms for sweeping non-planar profiles.

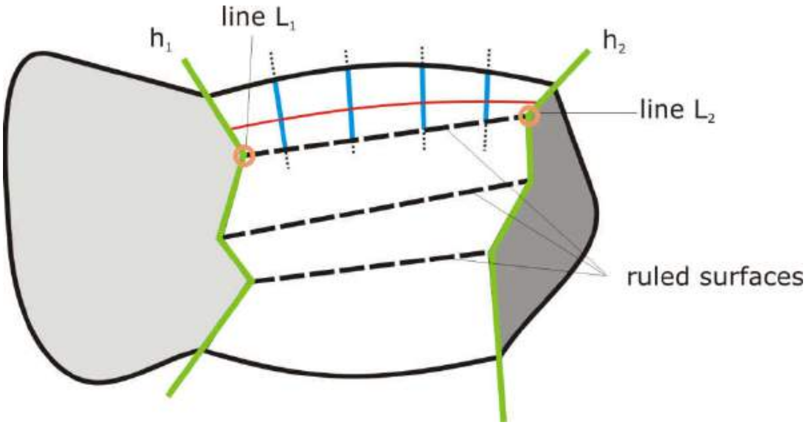

Fig. 4: If the two separating surfaces are piecewise linear having the same number of planar segments, the problem reduces to either the one shown in Figure 2 or Figure 3 for each pair of planar segments depending on whether the two separating surfaces are identical or not.

2.2 Motion Interpolation for Synthesizing the Deformed Subset DS

The synthesized geometry heavily depends on the characteristics of the motion interpolation algorithm. The development of interpolating techniques for motions that contain a set of given configurations of a moving object has been driven by applications such as trajectory generation in robotics, animation of mechanical systems and, more generally, CAD/CAM applications. Broadly, the object moving according to the interpolated motion is required to pass through a set of given configurations, possibly at a specified moment in time. There are many motions that can be constructed to contain a given discrete set of configurations, and choosing an interpolation algorithm will depend on each particular application. Conceptually, all existing interpolating schemes reduce the problem of interpolating a spatial motion of an object in E^3 to an interpolation of the motion of a point in the higher dimensional configuration space C. Therefore, generalizations of existing interpolation algorithms for 3-dimensional curves and surfaces to higher dimensional spaces arise naturally.

A scheme for interpolating rotations with Bézier curves in quaternion space based on an extension of the de Casteljau algorithm has been discussed in [34] by generalizing the interpolation algorithms from E^3. This idea was extended in [11] and [26] to spatial motions which include both translations and rotations. Some algorithms interpolate rotations and translations separately, sometimes by using quaternion algebra [17], and many other interpolation algorithms exist that try to optimize the dynamic properties of the interpolated motions [21], [4], [40], [39].

Another class of approaches to the motion interpolation problem is based on the theory of rational B-spline motions, and generates the so-called rational motions for which the trajectory of any point of the moving object is a NURBS curve. By using interpolation schemes based on rational motions, it is possible to apply many fundamental B-spline techniques to the design of motions. The basic theory of rational motions and some of their applications in computer aided design are discussed in [19], [18], [41], [42], while a relatively recent survey of the algorithms producing rational motions appears in [29]. Rational motions are very appealing from a design point of view because of their ability to generate NURBS curves and surfaces, which can exactly represent common geometric entities that occur frequently in engineering design and manufacturing. By contrast, non-rational motion interpolation algorithms generate motions that move points along transcendental curves that need to be approximated, which could in turn lead to deformation procedures that may not preserve certain geometric properties of the deformed shape.

The geometric properties of the interpolated motion can be controlled either by choosing the interpolated configurations appropriately, or by modifying the “control polygon” of the motion. However, even though the interpolated motion is a rigid motion, the corresponding control structure of the motion may not correspond to rigid body transformations (see for example [20]). In other words the direct manipulation of the control polygon may be difficult. Therefore, it may be more convenient to constrain the interpolated motion by carefully specifying and editing the interpolated (intermediate) configurations.

Enforcing the geometric continuity between the deformed geometry and the rigid subsets as well as the smoothness of the deformed geometry are two critical considerations. The geometric continuity across the split boundaries can be enforced in at least three different ways (see also section 4 for a more detailed discussion) by:

- specifying first derivatives for the sweep boundary at $t = 0$ and $t = 1$ (sometimes called “profile clamping” and available in commercial geometric kernels);
- the choice of separating surfaces;
- blending the boundary interfaces between RS_1, DS and RS_2 with curvature continuity;

In principle, all of these can be prescribed interactively or automatically by the user. However, developing automatic procedures for choosing these constraints, as well as specific choices of motion interpolation algorithms requires significant additional investigations. Note that once the interpolated motion is available, the shape and smoothness of the deformed geometry can be further controlled by manipulating the intermediate prescribed configurations.

2.3 The Joined Geometry and its Validity

Once the deformed geometry is generated, the last step is to join the individual subsets RS_1, DS and RS_2 that share, by construction, common faces into one solid model. The proposed procedure does not prevent, by itself, invalid constructions of the deformed geometry that could result in globally or locally self-intersecting geometry, as well as interference with the rigid subsets. However, self-intersections and interferences of the swept sets of points can be handled by employing the Point Membership Classification for solid sweeping published [9], [8].

2.4 Parameterizing the Deformation

We observe that the shape deformation approach proposed in this paper does not require an existing parameterization of the original solid model. Instead, our construction induces a new parameterization that controls the deformation itself, rather than the geometry. This parameterization includes

- o six parameters controlling the configuration of the displaced rigid subset

- motion parameter (t) that will control the extent of the deformation;
- the sampled set of key frames;
- scaling function or, alternatively, the number and distribution of intermediate profiles;
- prescribed derivative condition

A subset of these parameters can be used to perform edits of the deformed geometry, and to control the extent of the deformation, but these parameters are not independent of each other. Thus, defining the validity regions for the parameter space or constraints that enforce the validity of the result becomes paramount. The necessary conditions for Boundary Representation (BR-) deformations described in [28] based on the assumption of continuity during parametric updates can be extended and applied to our approach to constrained shape deformation.

It is important to note that this set of parameters controlling a prescribed deformation does not depend on the complexity of the geometry, so that even complex shapes can be deformed by controlling a relatively small set of parameters.

3 EXAMPLES

In this section we illustrate how our approach can be applied to shape deformations that include stretching/shrinking, bending, twisting, as well as combinations of these individual deformations with constant or prescribed variation of the geometry of the cross-section along the deformation. We implemented our approach in Parasolid along with the motion interpolation algorithm described in [15] using Visual Studio and OpenGL. In all six examples, the original models are split with prescribed separating surfaces, and the rigid subsets are repositioned to their end configurations (see section 2). The first four examples focus on solid models, while the last two examples illustrate the use of our approach to perform modifications of point clouds.

Figure 5(a) shows an object undergoing a bending deformation in an area of constant cross-section. The shape is split with a planar separating surface and the eyelets are rigidly moved to a prescribed configuration by a rotation of 75^{o} about a spatial axis (not shown). Depending on the choice of the key configurations, we can obtain different deformed geometries, with or without constant cross-section. Enforcing the constant cross-section can be achieved by controlling the scaling of the cross-section during the sweep as discussed in section 2. For example, there is no scaling in the example shown in Figure 5(c) and the cross-section of the deformed geometry changes due to the change in orientation of the planar generator face. By contrast, this cross-section is being preserved in Figures 5(e) and (g).

The second example illustrated in Figure 6 shows a bracket similar with the one shown in Figure 1(c) that undergoes a twisting deformation (rotation by 60^{o} along a longitudinal axis). In order to control the overall length of the part, we use two planar separating surfaces and apply the twisting operation to obtain the deformed geometry with a constant cross-section.

In our third example we successively apply two typical geometric edits to the molded part shown in Figure 7(a). The first edit is a stretching by a prescribed distance d using the linear separating surface (according to Case 1) shown in Figure 7(b). The result of this operation is illustrated in Figure 7(c). A bending deformation that follows the procedure described in Case 2 above is then applied to this stretched part by using two linear separating surfaces of Figure 7(d) together with prescribed values for the rotation angle and the vector defining the axis of rotation. Note that all parameters controlling the deformation are accessible through the user interface and can be used in downstream applications.

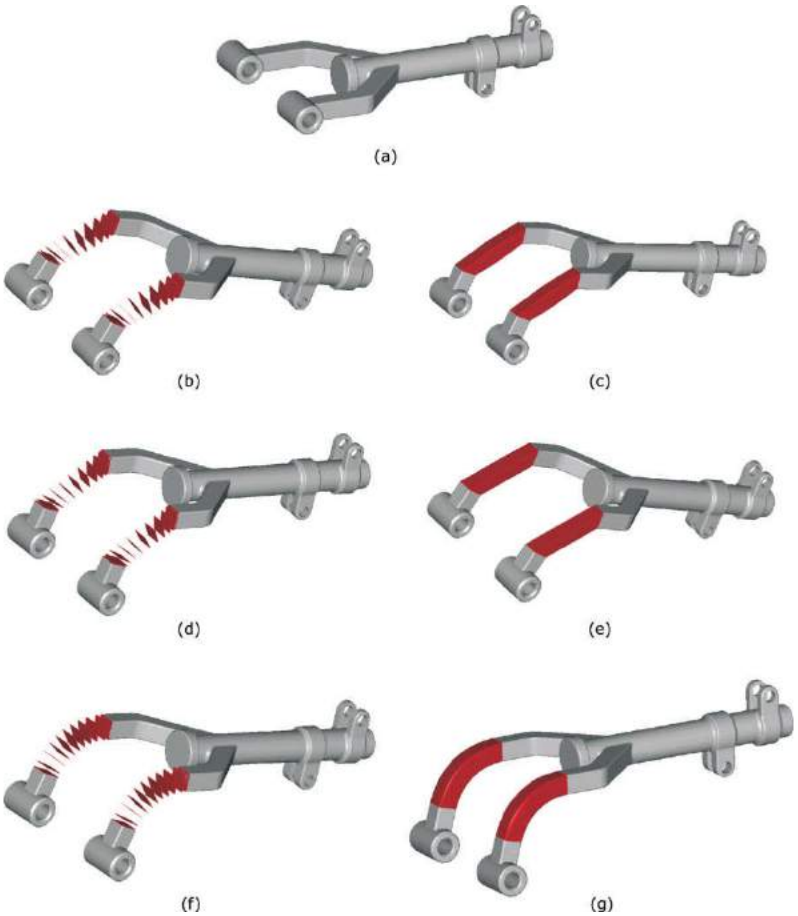

Fig. 5: A bending shape deformation with or without constant cross-section using one planar separating surface. The original geometry is shown in (a), while the pairs of figures (b) & (c), (d) & (e), and (f) & (g) show different resulting shapes depending on the key configurations prescribed. In figures, (e) and (g) the constant cross-section of the generated geometry is preserved.

The last two examples of Figure 9 show two point clouds that correspond to two different mechanical parts. The point cloud corresponding to a turbine blade shown in Figure 9(a) has been modified such that the length of the blade increases by a factor of 3, and the blade twists by an angle of 30 degrees such that the center of mass of each cross-section will be stacked along the vertical axis as illustrated in Figure 9(b). The second point cloud corresponding to the bracket shown in Figure 10 undergoes two modifications: first a change in the width of the bracket, followed by an extension of the fork along a circular path by an angle of 30 degrees. In this case, the deformed geometry consists of points that were obtained by applying the sampled transformations of the interpolated motion to the points of the cross-sections. For the last two examples, the point clouds corresponding to the deformed geometry were generated directly by applying sampled configurations of the interpolated motions to points of the cross-sections as described above.

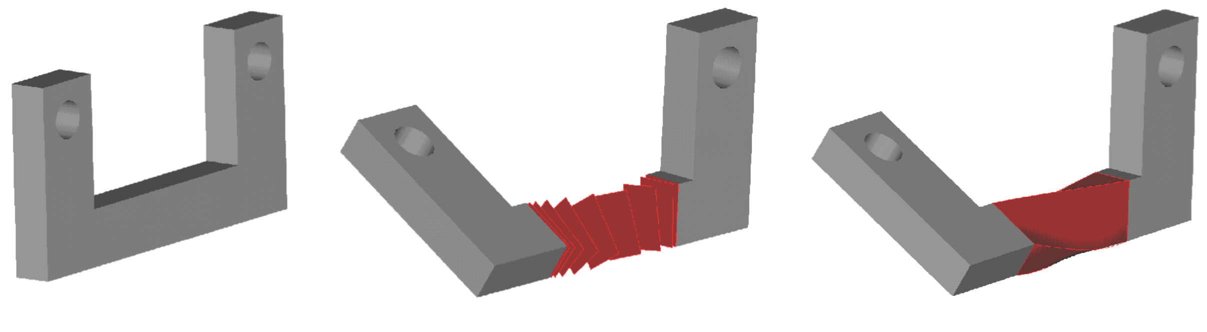

Fig. 6: A twisting shape deformation with constant cross-section and constrained total length is obtained by splitting the original geometry with two planar separating surfaces and generating the corresponding deformed geometry as proposed in section 2.

4 CONTINUITY OF THE DEFORMED SHAPE

Let's apply the same approach to deform the part shown in Figure 11(a) having a varying cross-section, and whose boundary is made of planar and cylindrical surface patches. In order to apply a deformation, we split the part with two linear separating surfaces as indicated in Figure 11(b). Prescribing a desired distance for stretching the mid-section of the part (i.e., between the cylindrical holes) is followed by a rigid transformation to reposition one of the rigid subsets, say the rightmost subset as positioned in the figure. Furthermore, computing the final shape follows the procedure described in Section 2.

Observe that some of the surfaces of the connecting and rigid subsets are joined with C^0 continuity, which is expected though not desired in this case, while the other surface patches are joined with G^1 continuity. The obvious questions are:

- which surfaces of the original shape can achieve only C^0 continuity after the deformation, and
- what can be done, if anything, to obtain higher order continuity?

To answer these questions, we start by looking at a simple wedge-like shape (that is, an inclined plane) whose side view is shown in Figure 12(a), and assume that we want to stretch this shape along a direction parallel to its base. Clearly, all the faces of our shape that are tangent to the stretching direction will have G^1 continuity (in fact, in this case these faces will have C^1 continuity), while all the other faces will achieve only C^0 continuity after the deformation as illustrated in Figure 12(b).

More broadly, let point P shown in Figure 12(c) be a point that is on the boundary of the original shape and on the boundary of the (piecewise planar) cross-section obtained as a result of the splitting operation. Also, let $\mathbf{n}_p$ be the normal of the tangent plane at P before splitting. Each such point P will describe a trajectory T_p according to the interpolated motion $M(u)$, where $u \in [0,1]$ is the parameter of the motion that takes P into a point P' on the corresponding face of the other rigid subset. Also assume that $\mathbf{t}_p$ is the tangent to T_p at $u = 0$ as shown in Figure 12(c). Then,

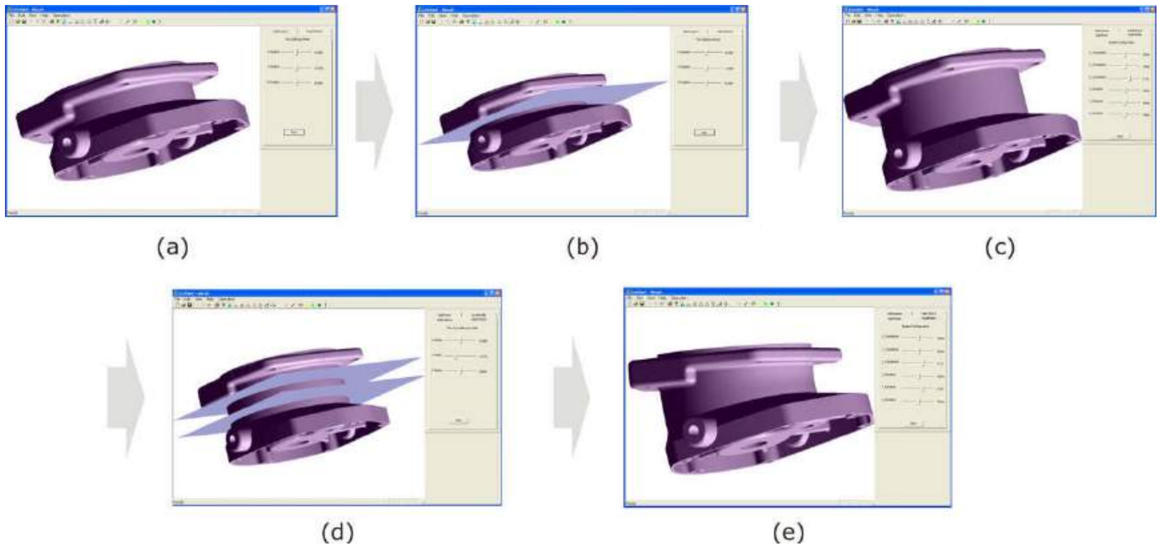

Fig. 7: Two successive deformations applied on a molded part: the stretching illustrated in Figures (b) and (c) is followed by the bending deformation shown in (d) and (e).

Lemma 1 *The deformed shape achieves* G^1 *continuity at* P *if and only if the normal* $\mathbf{n}_p$ *is perpendicular to the tangent* $\mathbf{t}_p$ *, or*

$$\mathbf{n}_p \cdot \mathbf{t}_p = 0 \tag{1}$$

Otherwise, the deformed shape can only achieve C^0 *continuity at* P *.*

For the case shown in Figure 12, the resulting shape will have C^0 continuity at P and G^1 continuity at point Q . Note that equation (1) provides the means to determine the type of continuity at every boundary point where the connecting set is joined with the rigid subsets. In addition, equation (1) suggests that the original shape should be split wherever the corresponding normals are coplanar in order to permit first order continuity after the deformation. Clearly, this is not always possible. In those cases where the normals $\mathbf{n}_p$ are not coplanar, one could adjust the direction of the tangents $\mathbf{t}_p$ so that equation (1) would be satisfied at all the cross-section boundary points (for example, by enforcing clamping constraints on the sweep surface). In general this would require non-linear separating surfaces, and result in more general interpolated motions that may no longer be rigid, which is a complicated issue. At the same time, the procedures described in section 2 would have to be modified as well.

Fortunately, in many practical situations the normals $\mathbf{n}_p$ are either constant, piecewise constant, or coplanar (for example in the case shown in Figure 11), which implies that one can use piecewise linear separating surfaces effectively. To illustrate this, consider again the part of Figure 11 that can be stretched while maintaining continuity by using two non-linear separating surfaces that follow the outer cylindrical surfaces of the part. Such an operation would require sweeping non-planar deformable surfaces which are *not* currently implemented in commercial solid modelers. However, since the deformable subset of the shape illustrated in Figure 11 is bounded by piecewise linear surfaces, one could achieve the same continuity with piecewise linear approximations of the non linear separating surface as described in Section 2. This geometric editing procedure in which the part shown in Figure 11(a) having varying cross-section is shrunk and extended while maintaining continuity is illustrated in Figure 13.

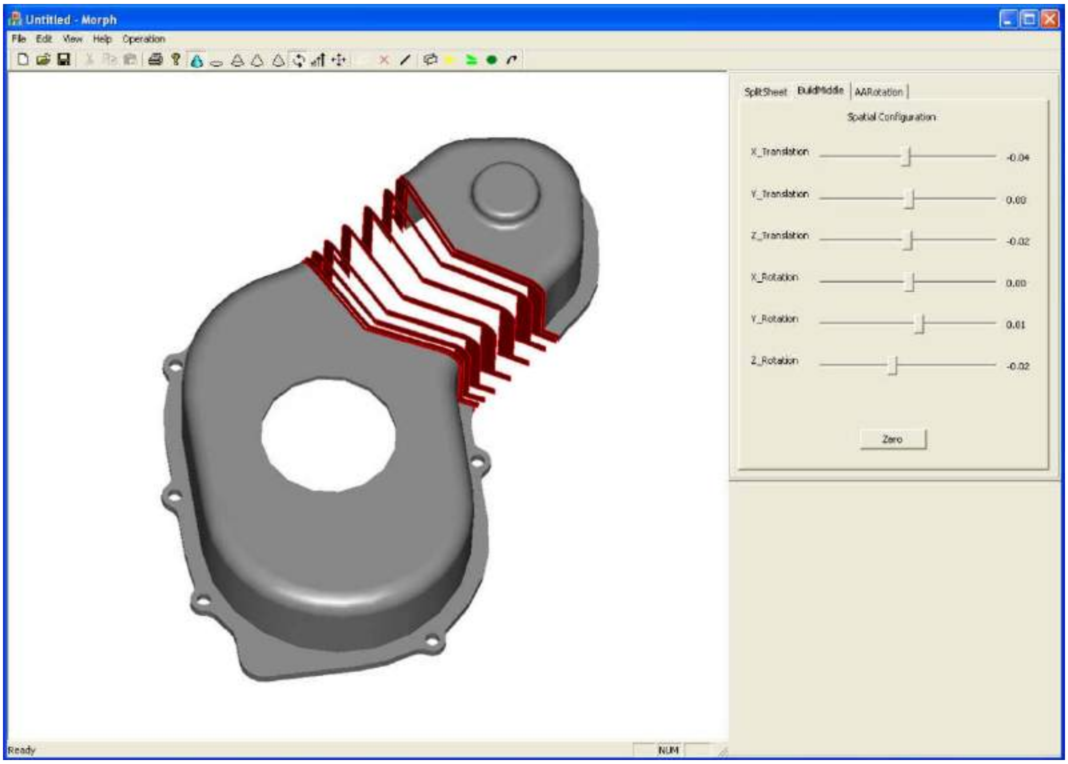

Fig. 8: Parameters controlling the deformation are accessible through the user interface.

5 CONCLUSIONS

Any design of reasonable complexity requires constant modifications of the geometry in order to satisfy the prescribed engineering requirements. Parametric modeling is a powerful paradigm that has been developed precisely for this very reason, but the current parametric technology does not adequately support many practical design cases.

In this paper we propose a new interactive approach to perform three commonly encountered local deformations of geometric models that relies on motion interpolation and piecewise linear separating surfaces to construct the deformed geometry. We showed that generating the deformed geometry based on motion interpolation algorithms offers a set of degrees of freedom that allow the detailed control of the deformed geometry without depending on the part history. Furthermore, it should be obvious that some subsets obtained as a result of the splitting operation could be thrown away as well which could provide additional mechanisms for generating smooth transitions between the rigid and deformed subsets. We argued in section 2 that rational motion interpolation algorithms seem to be the most appropriate for our application because they can support, in principle, geometric shapes that occur frequently in engineering design, and produce rational trajectories for the points of the cross-sections. Though, establishing whether rational motions are indeed needed requires further investigations.

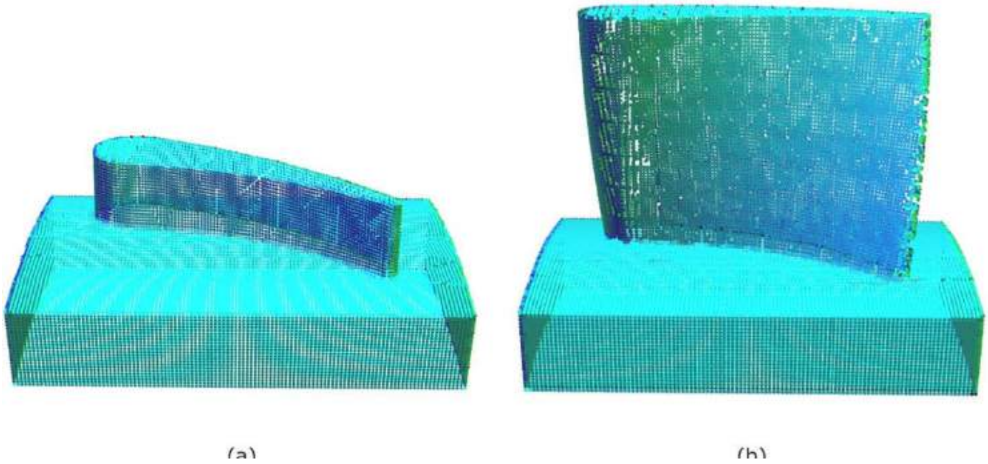

Fig. 9: Shape editing for a point cloud corresponding to a turbine blade that is extended and twisted such that the center of gravity of each horizontal cross section remains along a vertical line.

A direct consequence of our motion-based shape deformation is that the procedures described in Section 2 can constrain the cross-sections of the deformed geometry discretely at the (arbitrarily many) prescribed intermediate configurations, as well as map linear surfaces of the original geometry into surfaces of the same degree in the deformed geometry for those deformations in which such a mapping exists without depending on a pre-existing parameterization of the solid model. Furthermore, our motion-based shape deformation paradigm induces a parameterization of the deformation, which can be used to edit the geometry directly by editing the motion parameters, or to perform parametric optimization in downstream applications.

While the procedures described in this paper can certainly produce useful geometric edits that could nicely complement other existing geometric editing procedures such as [35], a number of important research questions need to be addressed before the full potential of the motion based shape deformation can be exploited. For example, for what class of deformations and shapes could the separating surfaces be automatically chosen, or at least suggested by the system? Intuitively, even a partial automation would require the coupling of the motion-based deformation procedures described in this paper with feature recognition capabilities. Furthermore, conditions for higher order continuity, morphing between faces obtained by splitting the geometry with non-linear separating surfaces, as well as a better understanding of how the motion interpolation algorithms influence the synthesized geometry would significantly extend the class of problems that can be synthesized by this procedure. At the same time, similar to any other geometric editing procedure, our approach may result in "invalid" deformed geometries. The invalid geometries due to envelope singularities, global self-intersections, and interferences can be detected by using the point membership test for sweeping solids [9]. Many of the invalid constructions of the deformed shape can be resolved with user input, but a formal characterization of the corresponding issues of validity can be built based on the assumptions of continuity of deformations proposed in [28]. Finally, because one can express the editing tools presented in this paper as geometric features, it is expected that the difficulties raised by the interactive features in feature-based modeling, and studied extensively in the literature, will be observed with our approach as well, particularly in those cases where multiple deformations are applied sequentially to a given solid model.

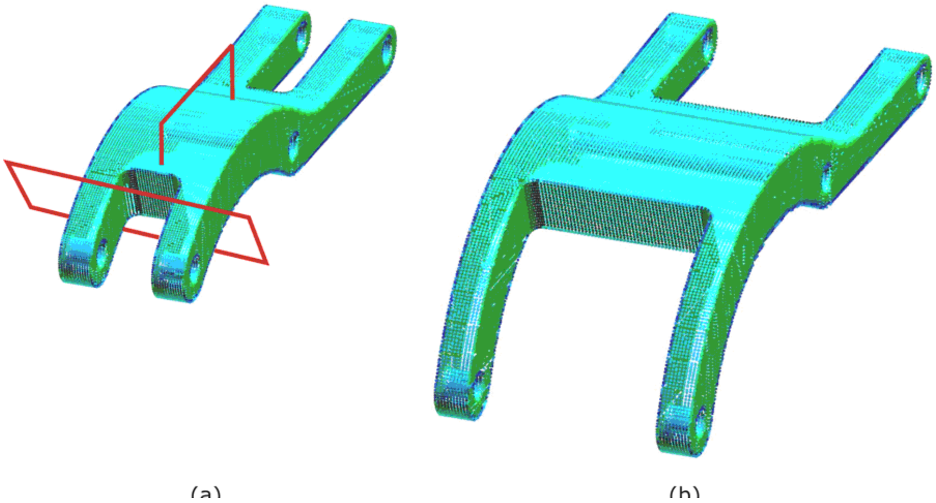

Fig. 10: Shape editing for a point cloud corresponding to a bracket that undergoes two successive edits.

Nevertheless, we view the work presented in this paper as a step towards the ability to create or edit geometric models independently of specific parameterizations or representations. Our approach can be implemented for any geometric representation that supports surface-model intersections, including representations based on standard NURBS-based B-reps, implicit functions, point-sampled surfaces, as well as simple point clouds. Moreover, the continuity of the deformed geometry at the interface between the original and the new (deformed) geometry can be analyzed in a straightforward manner. This, in turn, opens the door to a number of other important research avenues, including a very practical and interesting inverse problem that can be loosely posed as follows: given a desired deformed configuration (for example of the end hole in Figure 11), what are the corresponding deformations and associated parameters that will achieve the desired end deformation?

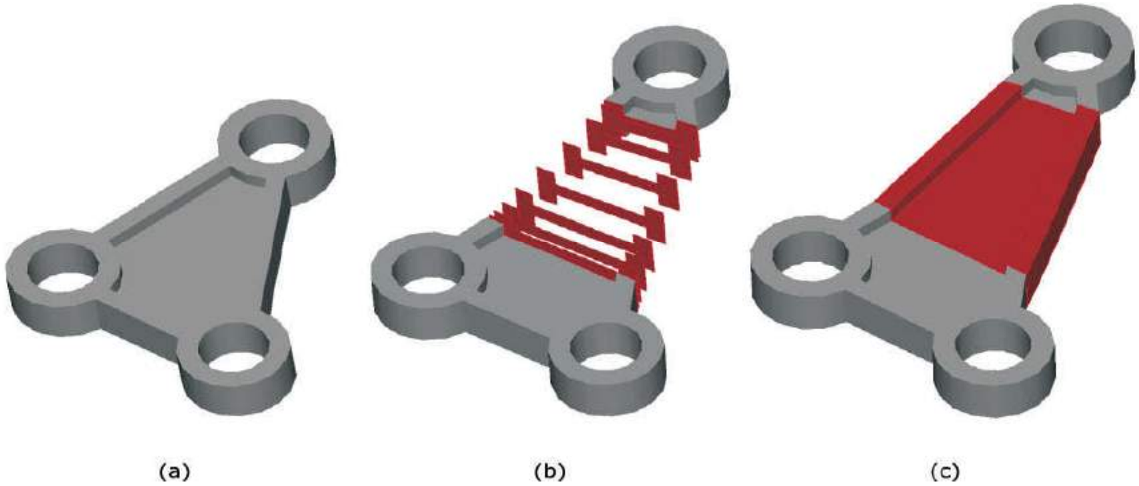

Fig. 11: A local scaling deformation with varying cross-section which points out one problem that can arise from an improper choice of the separating surfaces. Note the loss of smoothness between some of the newly created faces and the old faces of the part.

6 ACKNOWLEDGMENTS

This work was supported in part by a grant from the University of Connecticut Research Foundation, by a grant from Connecticut Center for Advanced Technology, and by the National Science Foundation grants CAREER CMMI-0644769, CMMI-0555937, CMMI-0927105, and CNS-0923158. All 3D solid models were constructed in Parasolid, courtesy of Siemens PLM Solutions.

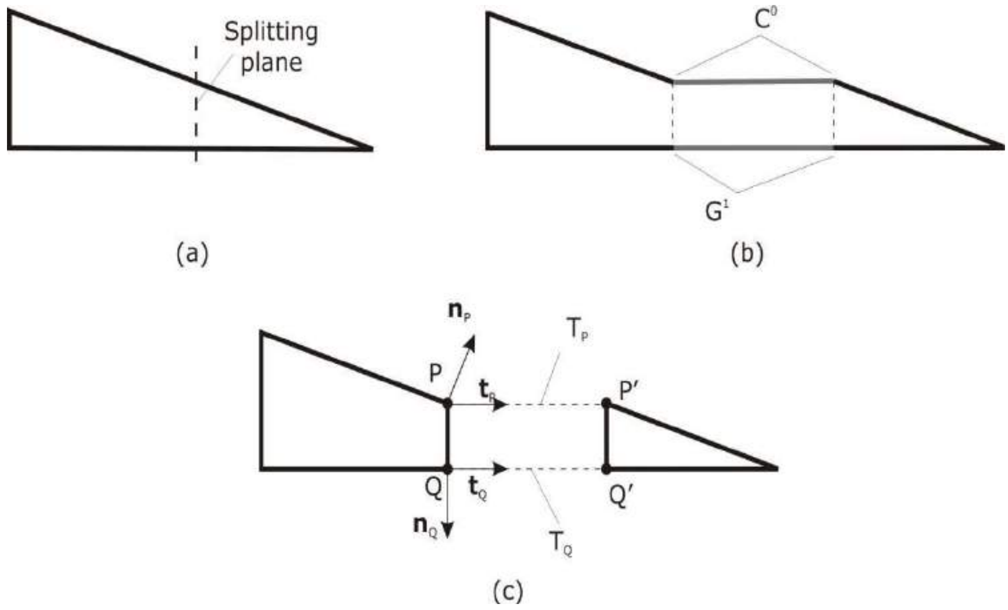

Fig. 12: Continuity across the boundaries

REFERENCES

[1] Beckmann, D.: Space deformation models survey, Computer & Graphics, 18(4), 1994, 571–586,
[2] Binford, T.O.: Visual perception by computer, In IEEE Conf. Systems and Control, Miami, FL, December 1971.
[3] Boier-Martin, I.M.; Ronfard, R.; Bernardini, F.: Detail-preserving variational surface design with multiresolution constraints, In SMI '04: Proceedings of the Shape Modeling International 2004 (SMI'04), IEEE Computer Society, Washington, DC, USA, 2004, 119–128
[4] Camarinha, M.F.; Leite, Silva,; Crouch, P.: Splines of class C^k on non-Euclidean spaces, IMA Journal of Mathematical Control and Information, 12, 1995, 399–410, doi:10.1093/imamci/12.4.399
[5] Chuang, Jen-Hui; Ahuja Narendra; Lin, Chien-Chou; Tsai, Chi-Hao; Chen, Cheng-Hui: A potential-based generalized cylinder representation, Computers & Graphics, 28(6), 2004, 907–918
[6] Coquillart, S.: Extended free-form deformation: an interactive animation technique, Computer Graphics, 24(4), 1990,187–193, doi:10.1145/97880.97900
[7] Cornea, N.D.; Silver, D.; Min, P.: Curve-skeleton properties, applications, and algorithms, Visualization and Computer Graphics, IEEE Transactions on, 13(3), May-June 2007, 530–548
[8] Erdim, Huseyin; T. Ilies, Horea: A comparison of sampling strategies for computing general sweeps, Computer Aided Design, 42(8), August 2010, 657–669, doi:10.1016/j.cad.2009.06.002
[9] Erdim, Huseyin; T. Ilies, Horea: Classifying points for sweeping solids, Computer-Aided Design, 40(9), 2008, 987–998, doi:10.1016/j.cad.2008.07.005

[10] Gal, Ran; Sorkine, Olga; Mitra, Niloy; Cohen-Or, Daniel: iWires: An analyze-and-edit approach to shape manipulation, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH, 28(3), 2009, article no. 33

[11] Ge, Q. J.; Ravani, B.: Geometric Construction of Bézier Motions, ASME Journal of Mechanical Design, 116, 1994, 749–755

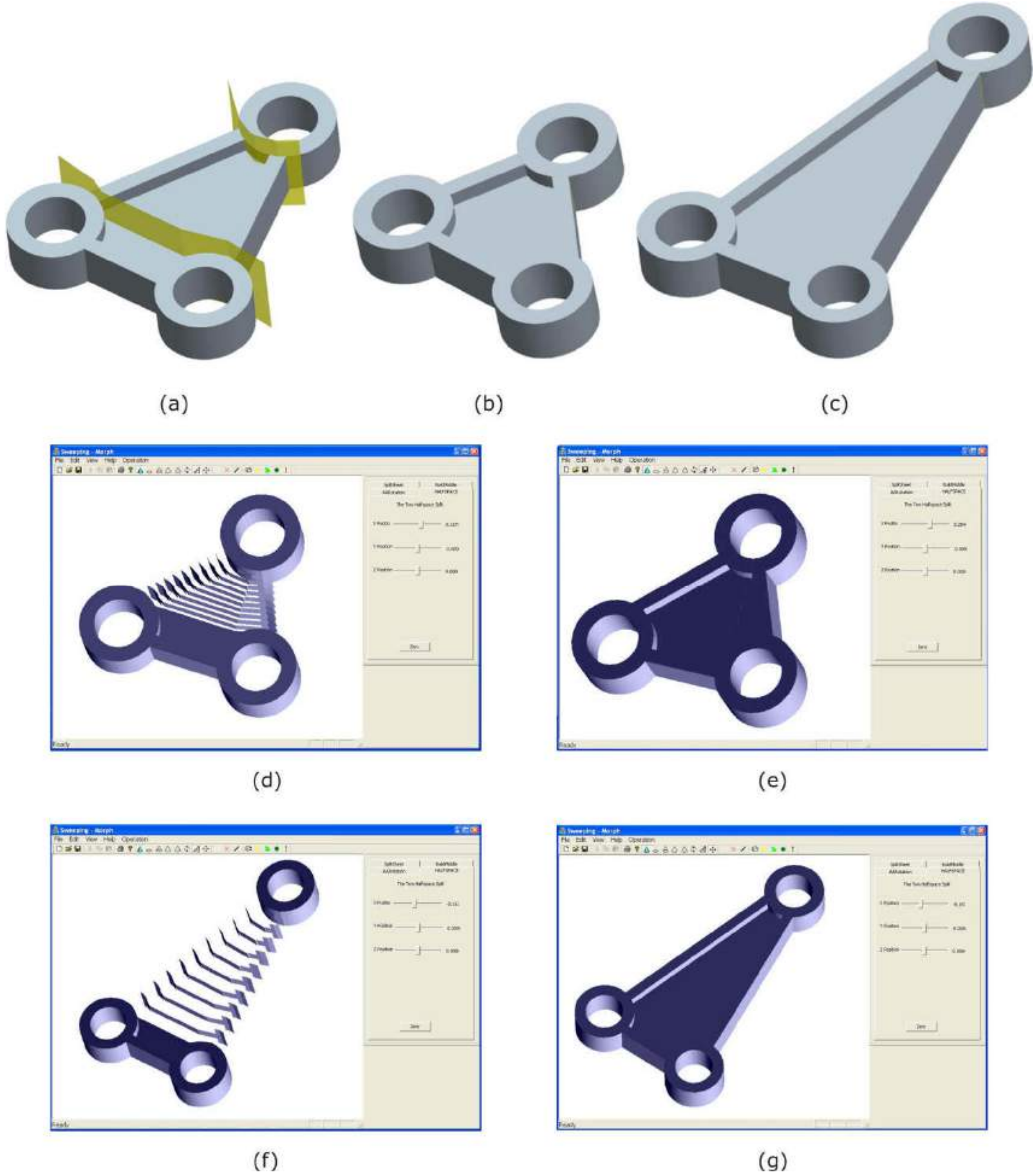

Fig. 13: The shape of Figure 11, illustrated in (a, is shrunk in (d) and (e), and extended in (f) and (g) with continuity by using piecewise linear separating surfaces (a). The resulting shapes are also shown in (b) and (c).

[12] Gibson, S.; Mirtich, B.: A survey of deformable modeling in computer graphics, in MERL, 1997

[13] Gomes, J.; Darsa, L .; Costa, B.; Velho, L.: Warping and morphing of graphical objects, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[14] Ho□mann, C.M.; Joan-Arinyo, R.: Parametric modeling, In G. Farin, J. Hoschek, and M.S. Kim, editors, Handbook of Computer Aided Geometric Design, chapter 21, Elsevier, Amsterdam, The Netherlands, 2002, 519–541

[15] Horsch, T.; Jüttler, B.: Cartesian spline interpolation for industrial robots, Computer Aided Design, 30(3), 1998, 217–224, doi:10.1016/S0010-4485(97)00061-4

[16] Hyun, Dae-Eun; Jüttler, Bert; Kim, Myung-Soo: Minimizing the distortion of affine spline motions, Graphical Models, 64(2), 2002,128–144, doi:10.1006/gmod.2002.0569

[17] Jüttler, Bert: Visualization of moving objects using dual quaternion curves, Computers and Graphics, 18(3), 1994,315–326

[18] Jüttler, Bert: Spatial rational motions and their applications in computer aided geometric design, In M. Dæhlen, T. Lyche, and L.L. Schumaker, editors, Mathematical Methods for Curves and Surfaces, Vanderbilt University Press, Nashville, TN, 1995, 271–280

[19] Jüttler, Bert; Wagner, M.G.: Computer-aided design with spatial rational B-spline motions, Journal of Mechanical Design, 118, June 1996,193 – 201

[20] Jüttler, Bert; Wagner, M.G.: Rational motion-based surface generation, Computer Aided Design, 31, 1999, 203–213, doi:10.1016/S0010-4485(99)00016-0

[21] Kim, M.-J.; Shin, S.Y.; Kim; M.-S.: A general construction scheme for unit quaternion curves with simple high order derivatives, In SIGGRAPH 95 Conference Proceedings, ACM SIGGRAPH, Los Angeles California, August 1995, 369–376.

[22] Kraevoy, V.; Sheffer, A.; Cohen-Or, D.; Shamir, A.: Non-homogeneous resizing of complex models, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia), 27(5), 2008

[23] Lazarus, F.; Verroust, A.: Three-dimensional metamorphosis: a survey, The Visual Computer, 14, 1998, 373–389,

[24] MacCracken, R.; Joy, K. I.: Free-form deformations with lattices of arbitrary topology, In SIGGRAPH '96 : Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, ACM Press, New York, NY, USA, 1996, 181–188

[25] Nealen, A.; Muller, M.; Keiser, R.; Boxerman, E. ; Carlson, M.: Physically based deformable models in computer graphics, Computer Graphics Forum, 25(4), December 2006,809–836

[26] Park, F. C.; Ravani, B.: Bézier curves on Riemannian manifolds and Lie groups with kinematic applications, ASME Journal of Mechanical Design, 117, March 1995,36–40.

[27] Pauly, Mark; Keiser, Richard; Kobbelt, Leif P.; Gross, Markus: Shape modeling with point-sampled geometry, ACM Trans. Graph., 22(3), 2003,641–650, doi:10.1145/882262.882319

[28] Raghothama, S.; Shapiro, V.: Boundary representation deformation in parametric solid modeling. ACM Trans. Graph., 17(4), 1998, 259–286, doi:10.1145/293145.293148

[29] Röschel, Otto: Rational motion design – a survey. Computer Aided Design, 30(3), 1998, 169–178

[30] Sederberg, T. W.; Parry, S. R.: Free-form deformation of solid primitives, Computer Graphics, August 1986, 151–160

[31] Sederberg, T.W.; Greenwood, Eugene: A physically based approach to 2–d shape blending, In SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques, ACM, New York, NY, USA, 1992, 25–34

[32] Shah, J.J.; Anderson, D.; Kim, Y.S.; Joshi, S.: A discourse on geometric feature recognition from cad models, Journal of Computing and Information Science in Engineering, 1(1), 2001, 41–51

[33] Shapiro, V.; Vossler, D.L.: What is a parametric family of solids?, In SMA '95: Proceedings of the third ACM symposium on Solid modeling and applications, ACM Press, New York, NY, USA, 1995. 43–54

[34] Shoemake, K.: Animating rotations with quaternion curves, Computer Graphics, 19(3), July 1985, 245– 254, doi:10.1145/325165.325242

[35] Siemens PLM Software, http://www.plm.automation.siemens.com/en_us/campaigns/breakthrough/. accessed, June 12, 2010.

[36] Srinivasan,Vinod; Akleman, Ergun; Chen, Jianer: Interactive construction of multi-segment curved handles, In PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications, IEEE Computer Society, Washington, DC, USA, 2002,429

[37] Dassault Systemes, http://www.3ds.com/flashgallery/imagine-and-shape-v5r17/.accessed, June 12, 2009

[38] Ulupinar, Faith; Nevatia, Ramakant: Shape from contour: Straight homogeneous generalized cylinders and constant cross section generalized cylinders, IEEE Trans. Pattern Anal. Mach. Intell., 17(2), 1995, 120–135, doi:10.1109/34.368175

[39] Zefran, M.; Kumar, V.: Interpolation schemes for rigid body motions, Computer-Aided Design, 30(3), 1998, 179 – 189, doi:10.1016/S0010-4485(97)00060-2

[40] Zefran, M.; Kumar,V.; Croke, C.B.: On the generation of smooth three-dimensional rigid body motions, IEEE Transactions on Robotics and Automation, 14(4), August 1998, 576–589

[41] Wagner, M.: Planar rational B-spline motions, Computer Aided Design, 27(2), 1995, 129–137

[42] Wagner, M.; Ravani, B.: Computer aided design of robot trajectories using rational motions, In L. Lenarcic and V. Parenti-Castelli, editors, Recent Advances in Robot Kinematics, Kluwer Academic Publishers, 1996, 151–158

[43] Welch, William ; Witkin, Andrew: Free-form shape design using triangulated surfaces, In SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques, ACM, New York, NY, USA, 1994, 247–256

[44] Woo, Y.; Lee, S.H.: Volumetric modification of solid CAD models independent of design features. Advances in Engineering Software, 37(12), December 2006, 826–835

[45] Yoon, S.-H.; Kim, M.-S.: Sweep-based freeform deformations, Computer Graphics Forum, 25(3) , September 2006,487–496, doi:10.1111/j.1467-8659.2006.00968.x

[46] Zerroug, Mourad; Nevatia, Ramakant: Three-dimensional descriptions based on the analysis of the invariant and quasi-invariant properties of some curved-axis generalized cylinders, IEEE Trans. Pattern Anal. Mach. Intell., 18(3), 1996,237–253, doi:10.1109/34.485553