Parameter-driven freeform deformations

Y.Song, J.S.M.Vergeest and D.P.Saakes†

Faculty of Industrial Design Engineering, Delft University of Technology

Abstract:

High-level parameters offer user an easy and intuitive tool to modify an existing model. In this paper, an approach of deforming freeform shapes using intrinsic parameters is proposed, where those parameters were not part of the model. Instead the parameters are introduced by user at the time he/she needs them. To achieve this, different from existing freeform deformation methods, a deformable template defined by intrinsic parameters is used as the bridge to link the original freeform shape and the user actions. Those templates can be defined by the user or simply derived from freeform feature concepts. With shape similarity analysis, a user-defined region of interest in the freeform shape is matched, or fitted, to a deformable shape template. By several different kinds of $\mathbb{R}^3$ to $\mathbb{R}^3$ functions, the region of interest is mapped to the fitted deformable shape template. Thus template parameters can be transferred to the original freeform shape. Through those quantitative intrinsic parameters, the user can easily modify the freeform shape. Several simple implementations were conducted in order to verify the proposed method. It is also described how the proposed technique can be applied in practical shape modelling applications.

CR Categories: I.3.5 [Computer Graphics] Computational Geometry and Object modelling, I.3.6 [Computer Graphics] Methodology and Techniques

Keywords: intrinsic parameters, freeform deformations, template, mapping, fitting.

1. Introduction

Finding effective and efficient tools for modifying freeform shapes is one of the state-of-the-art topics in Computer Graphics (CG) and Computer-Aided Design (CAD). In the past three decades, many useful methods were developed for freeform shape modifications. Typical examples of those methods include surface line manipulation, control point editing, lattice manipulation, tangent plane manipulation, push tools, tangent ribbon manipulation, etc.[1] In detail, surface line manipulation reshapes a surface by changing particular lines on the surface as described in Kallay's[2] work and Weich's[3] work. For control point editing, Piegl and Tiller[4] proposed two methods to modify NURBS curves and surfaces: modifying the control point positions and modifying the control point weights. In lattice manipulation, the geometric representation of a model is mapped to the lattice first. Then, the change of the lattice can be reflected to the model shape through topological relations[5-9]. As an important surface editing method, tangent plane manipulation shows its advantage in adjusting the curvature around the control points[10]. To reach a user desired shape, the tool pushing method has been developed where a portion of the freeform shape can be displaced by a desired shape[11]. As an auxiliary method, tangent ribbon is mainly applied in controlling the amount of curvature around the surface boundaries[12]. Besides, Signh introduced a wires based freeform deformation technique[13]. Based on the Direct Surface Manipulation (DSM)[14] method, Chen *et al.* developed a feature based freeform mesh deformation method for Computer-Aided Engineering (CAE). These features are self-contained mesh deformation operations that are context-free, stored separately from the base model, are applied to the model in a proper mix at any time

In those methods, it is found that different from regular shapes, freeform shape modification, possibly indirect, inaccurate or un-predictable, is known to be hard. The key issue is the complex geometric representations and the non-uniqueness of the types of parameters for a given freeform shape. Comparing to other shape modification methods, the lattice based FreeForm Deformations (FFDs) showed its advantages in move a set of points/control points together following the user's idea, thus it had been applied in many CG and CAD systems. Mathematically, the kernel of FFDs is a $\mathbb{R}^3$ to $\mathbb{R}^3$ mapping from the domain of the lattice into a bounded volume in the model space as $L(u,v,w) = \sum_{i}^{i_{max}} \sum_{j}^{j_{max}} \sum_{k}^{k_{max}} B_i(u) B_j(v) B_k(w) P_{i,j,k}$, where (u,v,w) is a point in the lattice domain and $L(u,v,w)$ is its projection in the object space, $P_{i,j,k}$ are the lattice vertices, and $B_i(u)$, $B_j(v)$, $B_k(w)$ are the basis

†Langbergstraat 15, 2628CE, Delft, The Netherlands,
E-mail: {y.song, j.s.m.vergeest, d.p.saakes}@io.tudelft.nl

functions in u, v, w directions, respectively. In general, the lattice is used as a bridge to link the original freeform shape and the user actions.

Though the lattice based FFDs are a powerful tool in modifying a freeform shape, it is hard to defining an efficient and effective lattice for modifying an exiting freeform shape. In addition, to change the shape of a model, modifying intrinsic parameters is much easier than operating on geometric basic constituents such as points, vertices, curves and individual surfaces[15]. Such as, a cylinder size is easily increased from Figure 1(a) to (b) by one of its parameters, diameter *d*. But currently, if user wants to change some high-level parameters of a freeform shape, such as the height of a bump, the whole shape would have to be redesigned or regenerated first to make those parameters available. For solving this problem, the feature concept was recently introduced to the freeform area.

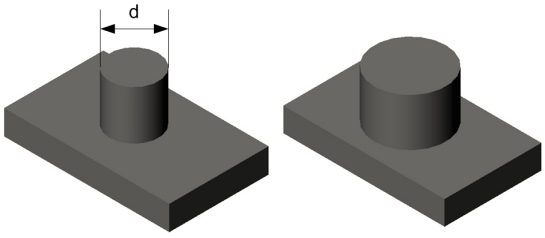

Figure 1: Modifying intrinsic parameters of a shape

In the context of this paper, a feature is a generic shape of a product with which engineers or designers can associate certain attributes and knowledge useful for reasoning about the product[16]. A feature offers the advantage of treating sets of elements as single entities, thus improving the efficiency in creating the product model. While the concept of feature has been mainly investigated in the mechanical environment, it was also introduced to the freeform area[17]. A freeform feature is a portion of a single or a set of freeform surfaces. The boundary of the feature consists of curve segments that lie within a shape[18].

Using freeform features in shape construction, they are always parameterized first[19-21]. Isolated parameterized features are named templates[22,23]. Based on templates, shape information of an existing model can be directly recognized and transferred into high-level parameters through template fitting. Thus, with freeform templates, Li and Hui[18] developed a two-phase approach to recognize freeform feature from B-rep models. Recently, Biermann *et al.*[24] applied freeform template in extracting shape information in their cut-and-paste applications. Generally, applications based on freeform feature concept are still quite limited. The problem is the non-uniqueness of a given arbitrary freeform shape.

In this paper, a new FFDs method is introduced to modify an existing freeform shape. The motivations of the proposed shape modification method are described in Section 2. Then, freeform templates are introduced and defined as the basis of the proposed method in Section 3. Different kinds of mappings between the template and the shape are constructed in Section 4. Through template fitting, the similarity of the template and the shape is increased (Section 5). By high-level parameters of the template, user can easily and intuitionally modify the freeform shape in a predictable way. With several implementations, the method is verified in Section 6. The advantages and limitations of the presented method are also discussed.

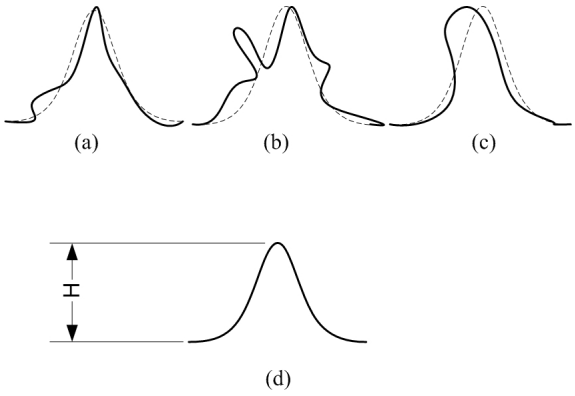

Figure 2: A bump-like abstract feature and the shapes it stands for

2. Motivations

Freeform shape modifications usually are interactive actions between the user and the computer. As summarized before, the user prefers high-level intrinsic parameters than basic geometric elements. On the contrary, freeform shapes usually do not follow a standard even with the freeform feature concepts, e.g., high-level parameters cannot always be found on a freeform shape. To solve this problem, in the proposed method, abstract features are introduced based on the user's definitions. Figure 2 describes a bump-like abstract feature and the shape it stands for. In Figure 2(a), (b) and (c), three complicate freeform shapes are presented. Each of them cannot be simply categorized to an existing freeform feature. On the other side, all of them have a same attribute. They are looked like a bump as the dot lines in the figure. Thus, a bump-like abstract feature can be defined as Figure 2(d). This abstract feature can stand for, or approximate, freeform shapes with the bump concept. The parameters introduced in the parameterization of the abstract feature also can stand for the parameters of the shape.

Using abstract features to approximate an existing freeform shape, there are three more tasks: first, a template should be defined based on intrinsic parameters of the feature. Then, proper parameters of the parameterized template should be found to approximate the shape. Third, what the use wants to modify is the original shape, so the

difference between the template and the original shape should be recorded. In this research, definitions of templates, template matching and several $\mathbb{R}^3$ to $\mathbb{R}^3$ mappings are proposed to solve those questions, respectively.

In general, the propose FFDs method can be undertaken in the following steps (Figure 3).

1. Given a freeform shape (Figure 3(a)), a Region of Interest (ROI) is selected. (Figure 3(b))
2. The user defines or selects a parameterized template,
3. The template can be fitted to the ROI through template fitting, intrinsic parameters can be found. (Figure 3(c))
4. By a $\mathbb{R}^3$ to $\mathbb{R}^3$ mapping, the difference between the template and the shape is recorded.
5. Modifying the intrinsic parameters of the template, and rebuilding the ROI in each step through an inverse mapping. In this stage, the user fells that he/she directly manipulates the shape through those intrinsic parameters. Such as, in Figure 3(d), the height of the bump is increased.

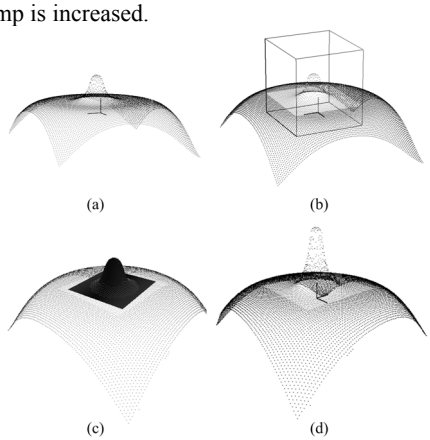

Figure 3: Parameter-driven FFDs

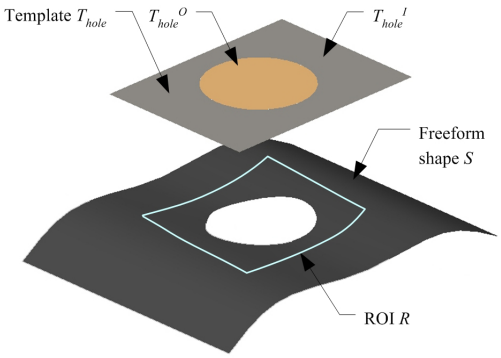

Figure 4: A freeform hole template and its corresponding shape

3. Freeform templates

Abstract freeform features can be represented by freeform templates. A freeform template is a parameterized freeform shape with additional information. It not only contains all the shape characteristics of a specified freeform shape, but also captures the information of shape elimination, e.g. the area of the hole(s). Furthermore, an adjustable clear boundary, such as a rectangle or a circle, can also be assigned to a template. For example, in Figure 4, a hole template with a rectangular outer boundary is defined corresponding to the hole lies on the shape.

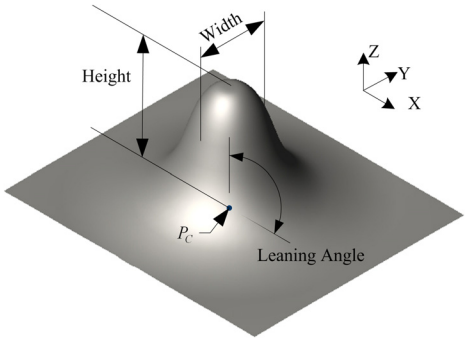

Figure 5: Parameterization of a bump template

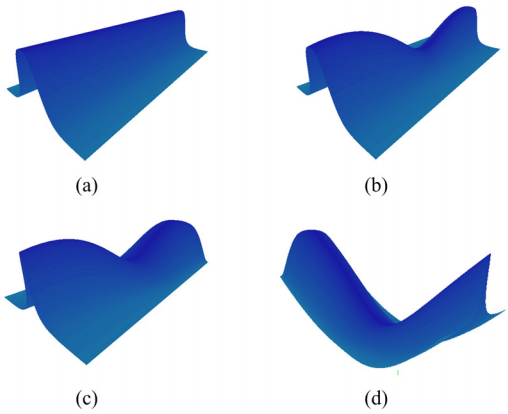

Figure 6: A user defined freeform template

Given a freeform shape S, specifying a ROI R in S (Figure 4), a freeform template of type t corresponding to R can be specified by a mapping $T_t : Q_t^T \to 2^{\mathbb{R}^3}$, where $Q_t^T = Q_1^T \times \cdots Q_i^T \cdots \times Q_m^T$ is the parameter domain of T_t, and Q_i^T represents the domain of continuous scalar variables q_i. A freeform template may contain two major parts $T_t^I(q) \subseteq T_t(q)$ and $T_t^O(q) \subset T_t(q)$, such that $T_t(q) = T_t^I(q) \cup T_t^O(q)$. Here, $T_t^I(q)$ represents the portion of the template that, in a matching procedure, would be similar to R. However, $T_t^O(q)$ would be similar to a shape not contained in the freeform shape. $T_t^O(q)$ can be considered a sub-template characterizing eliminated surface data. For shape deformation templates, usually $T_t^O(q) = \emptyset$. When a hole template $T_{hole}(q)$ is applied for matching with a hole

in a freeform surface, $T^I_{hole}(q)$ of the template ought to surround the hole whereas $T^O_{hole}(q)$ (light grey area in Figure 4) ought to locate itself in the void of the shape S.

A freeform template can be defined based on the free-form feature concept or according to user's requirements. Figure 5 shows a parameterized bump template. Figure 6 presents a use-defined flexible ridge template with 27 parameters. In Figure 6(a), the basic shape is shown. Figure 6(b), (c) and (d) present the deformed shape based on those parameters, details of template parameterization can be found in the authors' former works[22,23].

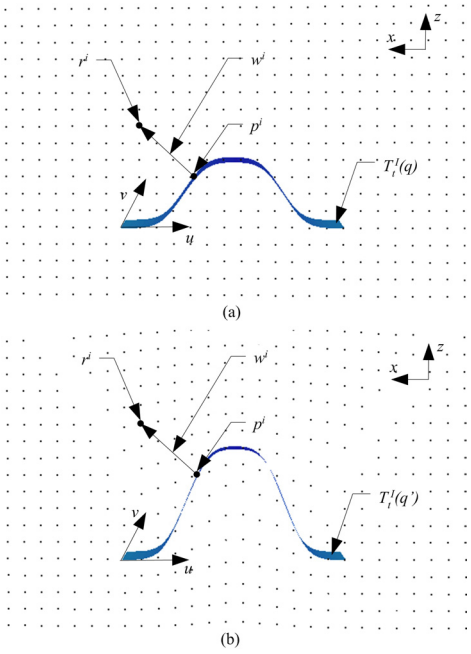

Figure 7: Assigning a uniform grid in $\mathbb{R}^3$ to a template, then deforming them by changing the template parameters

4. Template-controlled shape modification

Considering a freeform shape S with a specified ROI R, where $S \subset \mathbb{R}^3$, $R \subset \mathbb{R}^3$ and $R \subseteq S$, by a given freeform template $T_t(q)$, the objective of this research is to achieve a new shape $S_d = (S \setminus R) \cup R_d$, where R_d is the deformed ROI, and the deformation is controlled by the parameters of $T_t(q)$. As described in Section 3, $T_t^I(q)$ part of the template $T_t(q)$ represents the shape which is supposed to be located on the shape. For the parameter-driven FFDs, only $T_t^I(q)$ part of the template is considered. With Non-Uniform Rational B-Spline (NURBS)[4] representations, $T_t^I(q)$ can be represented as:

$$T_t^I(q) = \{S_t(u,v)\big|_q \mid (u,v) \in U\}, \quad (1)$$

where $U \in \mathbb{R}^2$, is the parameter domain of surface S_t, q is intrinsic parameters according to the template type t. Given a point $r^i \in \mathbb{R}^3$, it can be represented as

$$r^i = S_t(u^i, v^i)\big|_q + w^i, \quad (2)$$

where $S_t(u^i, v^i)\big|_q$ represents a point on the template. w^i is a compensation vector in order to make up the difference between r^i and $S_t(u^i, v^i)\big|_q$. The relations in Equation (2) are illustrated by Figure 7(a). In the figure, a series of uniform grid points are shown. Each of them can be assigned to the template. Thus, $\{(u^i, v^i, w^i) \mid i = 1,k\}$ are generated where k is the number of points. Here, (u^i, v^i, w^i) can be treated as the coordinates of point r^i based on template $T_t(q)$ in $\mathbb{R}^3$.

Suppose the intrinsic parameters of the template are changed from $q \to q'$, thus, template $T_t(q)$ is deformed to $T_t(q')$. Freezing the coordinates (u^i, v^i, w^i) of each grid point in this deformation, with Equation (2), the new positions of grid points in $\mathbb{R}^3$ can be computed by new template parameters q' as Figure 7(b). In the figure, it is found that that the grid point is not uniform again since the deformation of the template influence them.

In Figure 7, it is proved the template deformation can influence the shape (here is the grid points). Then, the key problem is how to assign the shape to the template. Suppose shape R is a point cloud, it can be represented as $R = \{r^i \in \mathbb{R}^3 \mid i = 1,n\}$, where r^i is each point in the point cloud and n is the number of point in the shape R. For assigning a point r^i of R in $\mathbb{R}^3$ to a template, a $\mathbb{R}^3$ to $\mathbb{R}^3$ mapping is needed to be constructed for mapping the shape to the template as $M: r^i \to p^i$, where $r^i \in R$ and $p^i \in T_t^I(q)$, which is the corresponding point on the template. In this paper, two different kinds of mappings, the distance mapping and the projection mapping, are introduced and applied to support the presented method based on different requirements.

Figure 8(a) illustrated the definition of the distance mapping which using Euclidean distance as the mapping tool. For a point r^i in the shape R, with the distance mapping, a corresponding point in the template can be found as

$$p^i = \arg \inf_{p^j \in T_t^I(q)} |r^i - p^j|. \quad (3)$$

When Equation (3) is solved, point p^i can be taken as the reference point on the template to r^i. Then, by NURBS representations and Equation (1),

$$(u^i, v^i) = S_t^{-1}(p^i)\Big|_q . \quad (4)$$

Typically, the inverse of S_t is not available in closed form, and in some cases it may even be undefined (for example if the template would be self-intersecting). In any practical situations, however, Equation (4) can be quickly approximated

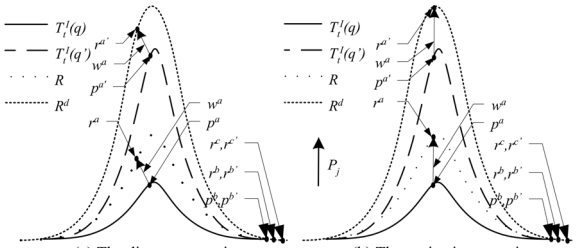

Figure 8: Different effects of different mappings

With Equation (2), the compensation vector w^i can be calculated as $w^i = r^i - p^i$. Thus, each point in shape R is mapped to (u^i, v^i, w^i) based on a specified template t as

$$M_D : r^i \to (u^i, v^i, w^i)\Big|_t . \quad (5)$$

Based on Equation (5) through an inverse mapping, shape R can be represented as

$$R = \{M_D^{-1}((u_i, v_i, w_i)\big|_t)\Big|_q \in \mathbb{R}^3 \mid i = 1, n\} . \quad (6)$$

Freezing the coordinates of each point in the shape R, modifying the intrinsic parameters of the template from q to q' in $\mathbb{R}^3$, $r^{i'}$ can be computed by Equation (6), such as the changes of $r^a \to r^{a'}$ in Figure 8(a), thus, new shape R^d can be achieved as

$$R^d = \{M_D^{-1}((u_i, v_i, w_i)\big|_t)\Big|_{q'} \in \mathbb{R}^3 \mid i = 1,2,\cdots n\} . \quad (7)$$

An important issue in the mapping concerns the boundary conditions. Maintaining the continuity of ROI R and the rest of the shape $S \setminus R$ is a necessary pre-condition for shape modifications. Normally, the scale of ROI is larger than the template $T_t(q)$. In the distance mapping, all the shape elements near the boundary in ROI are mapped to template boundary. Such as, in Figure 8(a), both r^b and r^c are mapped to p^b. Thus this portion of shape R^d will follow the change of the boundaries of the $T_t^I(q)$. Since the boundaries of template $T_t^I(q)$ is not changed after deformation in Figure 8(a), thus the boundary portion R^d is the same as R, for instance, $r^{b'} = r^b$ and $r^{c'} = r^c$.

Generally, the distance mapping is a global optimization for finding the shortest path from a point to a template. Sometimes, if the user wants to emphasis the shape consistency in a specified direction, the projection mapping is introduced as Figure 8(b). In the projection mapping, all the points of the shape are projected to the template following a specified projection direction. The projection point will be taken as the corresponding point. Given a point r^i and projection direction P_j, the projection of r^i on $T_t^I(q)$ should be $p^i = L(r^i, P_j) \cap T_t^I(q)$, where L is the line passing through r^i along the P_j direction. If multiple intersections are found, the smallest $\left|p^i - r^i\right|$ is taken into account. Same as the distance mapping, (u^i, v^i, w^i) can be computed by Equation (4) and Equation (5). To some points, if intersections do not exist, the distance mapping is applied, such as points r^b and r^c in Figure 8(b). By the projection mapping, shape R changes to R^d when template parameters change from q to q' as shown in Figure 8(b).

According to Equation (2), after computing (u^i, v^i, w^i) based on $T_t^I(q)$ for a given R, if it happens that the compensation vector $\left|w^i - w^j\right| \to 0, \quad \forall i, \forall j, \quad 0 < i < n, \; 0 < j < n$, a continuous behaviour is expected from the shape "consistent" with the modifications of the part $T_t^I(q)$ in the template $T_t(q)$. In a special case, if $\left|w^j\right| \to 0, \forall i, \; 0 < i < n$, e.g., shape R is as similar as possible to $T_t^I(q)$, the deformation of the template can be "transferred" to the shape. Thus, *the deformation of the shape R can be controlled by the deformation of the template, where the deformation of the template can be controlled by the user defined parameters*. To achieve this purpose, template matching is introduced to optimize the shape similarity.

5. Template Matching

In the proposed method, template matching is used to approximate a freeform shape with a given template and find its intrinsic parameters. To match the template $T_t(q)$ and the shape R, a similarity measurement is needed to determine the goodness of fit between the original shape and the freeform template. Given any two shapes A and B, the dissimilarity of them can be defined as a mapping: $d : \{A\} \times \{B\} \to \mathbb{R}$, where $\{A\}$ and $\{B\}$ and are the sets of

all possibly occurring shapes, respectively. d should preferably have the properties of a semimetric, i.e., for all shapes A, B, C: $d(A,\emptyset)=0$, $d(A,A)=0$, $d(A,B)\geq 0$ and $d(A,B)+d(A,C)\geq d(B,C)$.

With the dissimilarity measure d, the matching problem can be extended to search among multiple template types $T_t(q)$ (the learning set) to find the best fit to a freeform feature in R of shape S. Then, the matching procedure aims at obtaining the proper parameters of the template $T_t(q)$ under variation of the parameters $q\in Q_t^T$, where Q_t^T is the fitting parameters domain, and

$$q_{opt}=\arg\min_{q\in Q_t^T} f_d(T_t^I(q),T_t^O(q),R,\lambda), \quad (8)$$

where $f_d(T_t^I(q),T_t^O(q),R,\lambda)=d(T_t^I(q),R)-\lambda d(T_t^O(q),R)$, d is a dissimilarity measure and $\lambda>0$. Function f_d is named the optimization function, which is applied as the objective function in the fitting procedure. Equation (8) delivers the feature instance of type t that matches R optimally. In the equation, $d(T_t^I(q),R)$ measures the dissimilarity between part $T_t^I(q)$ of the template and R. When the template fits the surface, according to the definition, $d(T_t^I(q),R)$ will be minimal. $d(T_t^O(q),R)$ measures the dissimilarity between $T_t^O(q)$ and R. However, since $T_t^O(q)$ represents an eliminated part of a shape elimination feature, the term $d(T_t^O(q),R)$ should become maximal. By scalar coefficient λ, the "weights" of $d(T_t^I(q),R)$ and $d(T_t^O(q),R)$ can be adjusted in the overall similarity measurement. The whole match procedure can be accelerated by setting λ different in different stages of the fitting[22].

There are many similarity measurements defined in the literature[25]. It can be observed that a distance can be defined as $D(A,B)=\max(H(A,B),H(B,A))$, known as the Hausdorff distance between shapes A and B, where $H(A,B)$ is Directed Hausdorff Distance (DHD). DHD delivers the distance from shape A to B, it can be defined as $H(A,B)=\sup_{r\in A}(\inf_{s\in B}|r-s|)$, where $|r-s|$ denotes the Euclidean distance between the points s and r. To reduce the sensitivity to noise and inaccuracies in the shape data, the Mean Directed Hausdorff Distance (MDHD)[22], $M(A,B)$ is introduced as:

$$M(A,B)=\iint_A \inf_{s\in B}|r-s|\,dA \Big/ \iint_A dA, \quad (9)$$

where the integration is over the shape of A, normalized by the shape area of A. Based on MDHD definition and Equation (9), Equation (8) changes to

$$q_{opt}=\arg\min_{q\in Q_t^T} f_d^{MDHD}(T_t^I(q),T_t^O(q),R,\lambda), \quad (10)$$

where

$f_d^{MDHD}(T_t^I(q),T_t^O(q),R,\lambda)=M(T_t^I(q),R)-\lambda M(T_t^O(q),R)$, and λ is the same coefficient as Equation (8).

Digitized the template to a point cloud shape, $T_t(q)$ will be available as a point set. Supposing shape R is given as a point cloud, point sets P^I, P^O and P^R where $P^I=\{P_i^I\in T_t^I(q)\mid i=1,m\}$, $P^O=\{P_i^O\in T_t^O(q)\mid i=1,n\}$ and $P^R=\{P_j^R\in R\mid j=1,k\}$ represent $T_t^I(q)$, $T_t^O(q)$ and R respectively. Equations (10) are then approximated by

$$M(T_t^I(q),R)=\frac{1}{m}\sum_{i=1,m}(\max_{j=1,k}|P_i^I-P_j^R|),$$
$$M(T_t^O(q),R)=\frac{1}{n}\sum_{i=1,n}(\max_{j=1,k}|P_i^O-P_j^R|). \quad (11)$$

If the shape would be represented by polygons as $T^R=\{T_j^R\in R\mid j=1,k\}$, the same principle applies after changing item P_j^R to T_j^R in Equation (11). The point-to-point computation is then replaced by a point-to-facet computation. For a surface, it is also easy to digitize it to a point cloud with enough density. Generally, Equation (10) delivers the procedure of freeform template matching based on MDHD shape dissimilarity measure. A matching procedure is then simplified to a search for the optimized parameters $q_{opt}\in Q_t^T$ of the template.

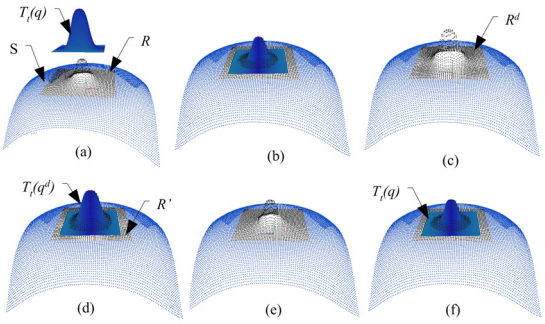

Figure 9: Modifications of a point cloud

6. Numerical implementations

A freeform shape can be represented as point clouds, meshes and surfaces (solids). In this section, three numerical experiments, corresponding to the three different kinds of freeform shapes, are presented in order to verify the proposed parameter-driven FFDs method.

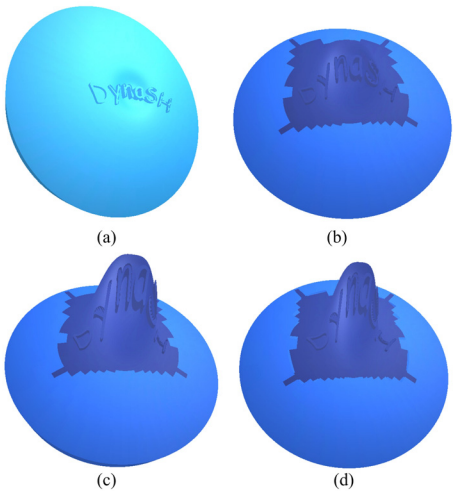

Figure 10: Deforming a mesh with a bump template

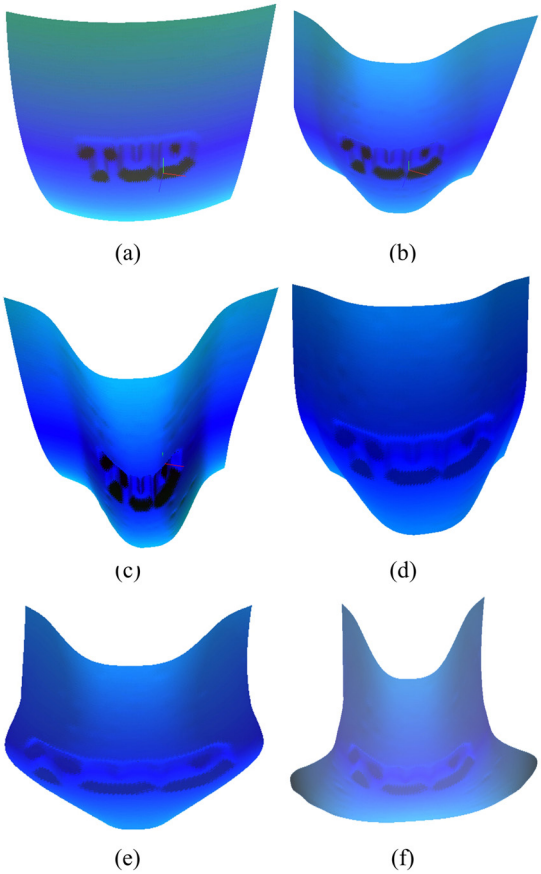

Figure 11: Deforming a surface with a user defined template

Using ACIS® and OpenGL®, the whole FFDs system was modelled by Visual C++®. The modifications of a point cloud are presented in Figure 9. Given a freeform shape S containing a bump-like feature, a freeform template $T_t(q)$ was selected as in Figure 9(a) to fit ROI R. By template matching, the bump was found and fitted as Figure 9(b) in 35 seconds. At that point, $M(T_t(q), R)$, *i.e.*. the shape dissimilarity is computed as 0.95835mm, compared to a diameter of S of about 245mm. With the projection mapping, the bump was associated to the template. Then the template height and width were increased 85% and 50%, respectively. The bump shape was successfully deformed by those parameters, e.g. shape R was modified to R^d as shown in Figure 9(c). In that stage, a new bump template $T_t(q^d)$ was defined and fitted to the shape as Figure 9(d). The fitting result $M(T_t(q^d), R^d)$ was 1.091908mm.With the same mapping, inverse operations, e.g., reducing the height and width which were increased in Figure 9(c), were taken. The bump was changed back to the original size as Figure 9(d). To verify the result, the same template $T_t(q)$ as Figure 9(b) was used to fit the shape again as Figure 9(f). In fitting results, $M(T_t(q), R')$ is 0.972482mm. That is, the error after two deformation processes is about 1.47% referring to MDHD.

In Figure 10, the same template is applied to deform a mesh model with the distance mapping. Figure 10(a) shows a freeform shape with logo "DynaSH" in relief effect. The shape is represented by 64602 polygons. Figure 10(b) shows a bump-like deformation with 29mm height and 49mm width which was created by the presented deformation method. In Figure 10(c) the bump height was increased to 55mm where in Figure 10(d) the width was reduced to 35mm where the logo's shape is preserved well. In the whole deformation process, the user only needs to play with 2 parameters, e.g., the height and the width.

Figure 11 presents the deformation of a freeform surface with a "TUD" logo carved in intaglio. In the surface deformation, instead of points in point cloud, vertices in meshes, the controls points of the surface is assigned to the template. In this experiment, the original shape is shown in Figure 11(a). With a 27 parameters ridge-like template, Figure 11(b), (c), (d), (e), (f) present a complicate deformation step by step. In this deformation, 11 parameters of the template are changed.

7. Conclusions

The parameter-driven FFDs method presented in this paper provides a way to handle and redesign freeform shapes by intrinsic parameters. Comparing to the existing FFDs methods, we summarize the following advantages:

1. The user can use several quantitative parameters to drive a complex freeform deformation;

2. Intrinsic parameters of the existing shape can be found and modified;
3. With several simple or user-defined templates, it can modify a number of freeform shapes;
4. The user need not position the template. Abstract features can be automatically found by template fitting;
5. The original shape information is well preserved in the deformation process.

There are also limitations of this technique, currently, we identify the following:

1. Template fitting is time consuming, normally, a fitting process will take about 20 to 200 seconds with a Pentium 4® 2.4G processors;
2. The freeform shape have a little aliasing after deformation;
3. The continuities between different surfaces can not be guaranteed during the deformation.

Current research is directed towards those limitations. Different effects of mappings are also being studied in order to realize a more effective and efficient FFDs method.

Acknowledgement

The presented research is a part of the Dynash project conducted in Faculty of Industrial Design Engineering, Delft University of Technology. (www.dynash.tudelft.nl) Valuable comments from the anonymous reviewers are appreciated.

This research project DIT.6071 is supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Ministry of Economic Affairs, The Netherlands.

Reference:

1. Dijk L. Vergeest J.S.M. and Horváth I., Testing shape manipulation tools using abstract prototypes, *Design Studies*, 19 (2),1998, 187-201.
2. Kallay M. *Constrained optimization in surface design, modeling in Computer Graphics*, Springer, 1993.
3. Weich W. and Witkin A. Variational surface modeling, *Computer Graphics* 26(2), 1992, 157-166.
4. Piegl L. and Tiller W., *The NURBS book*, Springer pres, Berlin, 1996.
5. Sederberg T.W. and Parry S.R., Freeform deformations of solid geometric models, Computer Graphics, *SIGGRAPH*, 20(4), 1986, 151-160.
6. MacCracken R. Joy K. I., Free-Form deformations with lattices of arbitrary topology, Computer Graphics, *SIGGRAPH*, 1996.
7. Coquillart S., Extend free-form deformations: A sculpturing tool for 3D geometric modelling, *Computer Graphics*, 24(4), 1990, 187-196.
8. Lamousin H. J. and Wangenspack W. N., NURBS based freeform surface deformations, *IEEE Transaction on Computer Graphics and Application*, 14(6), 1994, 59-65.
9. Barr A. H.. Global and Local Deformations of Solid Primitives. In *SIGGRAPH'84,* ACM, July 1984.
10. Verlaat N. *New method for conceptualizing in CAD system design*, Master thesis, Delft University of Technology, Delft, The Netherlands, 1993
11. Bryson S. Paradigms for the shaping of surface in a virtual environment, *Proceedings of the IEEE International Conference on system science*, 2, Software Technology, 1993.
12. Farin G., *Curves and Surfaces for Computer-Aided Geometric Design, A practical guide*, Academic press, San Degio, 1997
13. Singh K. and Fiume E., Wires: a geometric deformation technique, *SIGGRAPH*, 25, 1998, 405 - 414.
14. Chen, Y., Stewart P. J., Buttolo, P., and Ren, F., A real -time interactive method for fast modification of large-scale CAE mesh models, *ASME DETC.00*, DAC-4268.
15. Vergeest J. S. M., Spanjaard S., Horvath I. and Jelier J. J. O., Fitting Freeform Shape Patterns to Scanned 3D Objects. *Journal of Computing and Information Science in Engineering*, 1 (3), 2001, 218-224.
16. Shah J. J. and Mantyla M., *Parametric and featured based CAD/CAM*, Wiley-Interscience Publication, John Wiley Sons In., 1995.
17. Fontana M., Giannini F. and Meirana M., A freeform feature taxonomy, *Computer Graphics Forum*, Eurographics '99, 18 (3), 1999.
18. Li C.L. and Hui K.C., Feature recognition by template matching, *Computer & Graphics*, 24, 2000, 569-583.
19. Surazhsky T. and Elber G., Matching free-form surfaces, *Computers & Graphics*, 25 (1), 2001, 3-12.
20. Piegl L. A. and Tiller W., Parametrization for surface fitting in reverse engineering, *Computer-Aided Design*, 33 (8), 593-603, 2001.
21. Floater M. S. and Reimers M., Meshless parameterization and surface reconstruction, *Computer Aided Geometric Design*, 18 (2), 2001, 77-92.
22. Song Y., Vergeest J. S. M. and Horvath I., Feature interference in free form template matching, *Proceeding of EuroGraphics*, short presentations, 2002.
23. Song Y., Vergeest J. S. M. and Horvath I., Reconstruction free form surface with parameterized features, *ASME DETC'02*, DAC-34036
24. Biermann H., Martin, I., Bernardini F., Zorin D., Cut-and-paste editing of multiresolution surfaces *SIGGRAPH*, 2002
25. Hagedoorn M. and Veltkamp R. C., Reliable and efficient pattern matching using an affine invariant metric. *International Journal of Computer Vision*, 31(2/3), 1999, 203-225.