Proceedings of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2008
August 3-6, 2008, Brooklyn, New York, USA

DETC2008-49438

IDENTIFYING FEATURE HANDLES OF FREEFORM SHAPES

Yu SONG
Faculty of Industrial Design Engineering,
Delft University of Technology,
Landbergstraat 15, 2628CE, Delft, The Netherlands
y.song@tudelft.nl

Joris S.M. VERGEEST
Faculty of Industrial Design Engineering,
Delft University of Technology,
Landbergstraat 15, 2628CE, Delft, The Netherlands
j.s.m.vergeest@tudelft.nl

Tjamme WIEGERS
Faculty of Industrial Design Engineering,
Delft University of Technology,
Landbergstraat 15, 2628CE, Delft, The Netherlands
t.wiegers@tudelft.nl

ABSTRACT

Trends, ergonomics and engineering analysis post more challenges than ever to product shape designs, especially in the freeform area. In this paper, freeform feature handles are proposed for easing of difficulties in modifying an existing freeform shape. Considering the variations of curvature as the footprint of a freeform feature(s), curvature analysis is applied to find manipulators, e.g. handles, of a freeform feature(s) in the shape. For these, a Laplacian based pre-processing tool is proposed first to eliminate background noise of the shape. Then least square conformal mapping is applied to map the 3D geometry to a 2D polygon mesh with the minimum distortions of angle deformation and non-uniform scaling. By mapping the curvature of each vertex in the 3D shape to the 2D polygon mesh, a curvature raster image is created. With image processing tools, different levels of curvature changing are identified and marked as feature point(s) / line(s) / area(s) in the freeform shape. Following the definitions, the handles for those intrinsic freeform features are established by the user based on those feature items. Experiments were conducted on different types of shapes to verify the rightness of the proposed method. Different effects caused by different parameters are discussed as well.

1 INTRODUCTION

With the ever swelling demands from a range of products and various application domains for reduced product lifecycles, the needs of efficient and effective tools and methods for geometric modeling/modification are continuing increasing, especially in the freeform area [**1**]. In the past decade, reverse engineering has received much attention for its importance in accelerating product design process. Generally, reverse engineering of shape is the process of obtaining a Computer Aided Design (CAD) model from measurements of an existing artifact [**2**]. The purpose of reverse engineering can be either to provide digital support for subsequent life cycle stages of a product for which no CAD model is available, or to support the redesign of an existing product [**3**]. Although shape reconstruction through reversing engineering [**4**] is a well established technology, its application in extracting existing freeform shape information for further modifications is still an unsolved problem.

Different from the regular shape, extracting freeform shape information, possibly indirect, inaccurate or unpredictable, is known to be hard. The major reasons are the complex geometrical representations and the non-uniqueness of the types of parameters for a given freeform shape. For this, freeform feature concept [**5**, **6**, **7**] was recently introduced to help the designer in creating and manipulating freeform shapes. By the advantage of feature concept [**8**], the designer can directly manipulate pre-defined intrinsic parameters of features in a freeform shape.

Feature concept was introduced to many existing CAD systems. In a freeform feature based CAD system [**7**], feature information can be easily viewed, browsed and its parameter can be easily modified. But for an existing geometric shape, such as a patch of digitized surface, feature information does not always accompany with it. Besides, it is hard to draw clear boundaries among freeform features since the intrinsic properties of a feature are not unique. For instance, a stretched bump shape can also be treated as a ridge. Thus, unlike mechanical features [**4**], recognizing features in an existing shape based on a well-defined and detailed freeform feature taxonomy is quite difficult.

Intrinsic properties of a freeform feature always accompany with changes in the geometry. In the past decades, two methods were developed to identify those changes: template fitting [**9**] and identifying feature lines. In template fitting, a well-defined freeform feature template, which is controlled by pre-defined parameters, is used to fit the region of interest of a given shape. Template matching is proven to be robust, but the numbers of parameters and the computing expensive fitting procedure [**10**] prevent its further usages in identifying complicate freeform features.

Applications of identifying feature items, especially feature lines, in the geometry usually link with reverse engineering and segmentation of an existing freeform shape. Shape descriptors, which are some sets of numbers that are produced to describe a given shape [**11**], were frequently used in those applications. In general, shape descriptors represent parts of the "natural" attributes of a given shape, and the shape may not be entirely reconstructable from the descriptors, but the descriptors for different shapes should be different enough that the shapes can be discriminated. For instance, Levy et al [**12**] used second order differences, i.e., the angles between the normals, in their segmentation applications. Jagannathan et al [**13**] applied curvedness, which is known as benting energy, in their adaptive segmentation applications.

With a defined shape descriptor(s), algorithms were developed to find the feature lines. In 2D digital image processing, detecting edges, which are typical feature lines, has been intensively studied [**14**]. But in 3D space, due to the complexities introduced by the additional dimension and the irregularities of freeform shapes, fewer methods were developed and they can be categorized [**13**] to: a. Using Reeb graph ideas based on Morse theory [**15**]; b. Extending classical segmentation approaches used in image analysis to 3D space [**16**] and c. Performing perceptual segmentation. This algorithm is based on theminima theory and essentially, it defines boundaries as lines of negative minima curvature [**17**].

Combining the shape recognition techniques and the freeform feature concept, this research is the pilot stage of developing a cross-platform and user-friendly tool to manipulate existing freeform shapes. The target of this research is to find suitable handles for those manipulations. For this, shape handles are defined in Section 2 based on an abstract freeform feature concept and shape curvature, which is a type of descriptors. To find those handles, curvature plot of a given freeform shape is analyzed first. Then a Laplacian based pre-processing tool is developed in Section 3 to eliminate high-frequency background noises of the shape. In Section 4, for further analyzing the freeform shape, different from existing approaches in 3D space, a 2D approach is brought forward inspired by harmonic and conformal maps. With least square conformal maps, which maximally preserve the angles and scales in the mapping, 3D geometric shape is mapped to 2D planar polygon meshes. Assigning each vertex in the 2D mesh the curvature value of its corresponding vertex in the original 3D shape, a curvature raster image is created. With image processing tools, different levels of curvature changing are identified and marked as feature point(s)/ line(s)/area(s) in the freeform shape, where feature handles can be established based on them by the user. The architecture of the presented approach is illustrated in Figure 1.

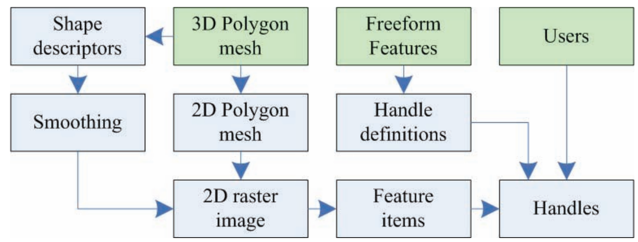


Figure 1: The architecture of the approach

2 HANDLES FOR FREEFORM FEATURES

Generally, a feature is a generic shape of a product with which designers can associate certain attributes and knowledge useful for reasoning about the product [2]. Features offer the advantage of treating sets of elements as single entities, thus improve 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 [**5**, **7**]. A freeform feature is commonly represented as a portion of a single or a set of freeform surfaces. Unlike for mechanical features, for freeform features, a clear, unique boundary cannot always be specified [**9**].

In 1999, a freeform feature taxonomy was proposed by Fontana et al [**5**]. In their work, according to the intrinsic properties and different contributions to the freeform shape, the freeform features were classified to two main categories: shape deformation features and shape elimination features, then they were further elaborated to many detailed features, such as n-groove, n-channel, etc. Nyirenda et al [**7**] proposed another generic freeform feature class definition. In their definition, besides the definition of Fontana et al [**5**], compound freeform features were proposed to use hybrid freeform features to represent complicate geometric shapes.

In feature based freeform modeling, a designer may quickly select a well-defined feature(s) which is similar to the desired shape and instantiate it with proper parameters on a basis surface. Then he/she starts to use different kinds of geometric modification tools, or even simply move the basic geometric elements, for instance control points in Non-Uniform Rational B-Spline(NURBS) surfaces, to achieve the desired shape. Due to the limited features type and diversities of freeform shapes, those modifications are always necessary. Later suppose he/she wants to edit this shape in the same CAD system, based on the modeling history, the whole modeling process can be scrolled back and feature parameters can be easily retrieved and modified. But in reverse engineering, due to lack of modeling history, those user modifications post challenges to feature recognition algorithm. Besides, many existing freeform shapes are not created based on freeform feature based modeling system, which means the exactly same shape as one in the feature library hardly can be found. Thus, features in the library hardly match existing shapes.

In our definitions, instead of using many types of detailed freeform features, several abstract features are proposed to approximate most of freeform shapes. Based on those abstract features, we offer user feature handles, which is a superset of feature parameters, in order to manipulate the feature in a more flexible and intuitive manner. There are two general feature categories of abstract freeform features: deformable freeform features and elimination features. A deformable feature is defined on intrinsic properties of a given shape from the user perspective of view, where elimination features is defined based on the boundary of the part it eliminates. Among them, the deformable freeform feature was further elaborated to protrusion, extrusion and bent features. Ideal examples of those freeform features are presented in Figure 2. Most of the existing freeform features can be simply categorized to these three types. For instance, both n-groove and ridge [**5**] features belong to the protrusion feature.

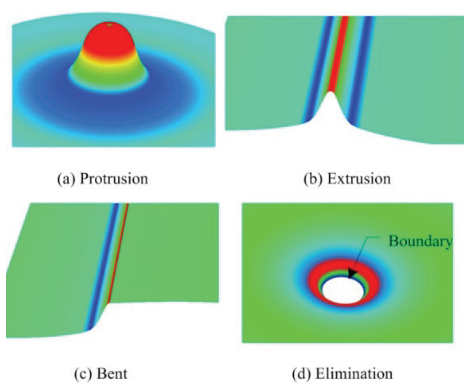


Figure 2: Examples of ideal freeform features with their curvature plots

A freeform deformable feature always accompanies with changes of curvature. Local curvature, which is a type of shape descriptors, is the amount by which a geometric object deviates from being flat. For a 2D curve, it is the degree of the deviation from a straight line, or for a 3D surface, it is the degree of the deviation from being planar. In other words, curvature is the degree by which a geometric object bents (at a location, at a time). Generally, for surfaces embedded in $\mathbb{R}^3$, there are four types of curvature to each given point on the surface: minimal curvature, maximal curvature, Gaussian curvature and mean curvature [**18**]. Considering the intersection of the surface with a plane containing a fixed normal vector at the point, this intersection is a plane curve and has a curvature. By varying the plane, this curvature will change, and there are two extreme values - the maximal and the minimal curvature, named the principal curvatures and the extreme directions are named principal directions. Here we adopt the convention that a curvature is taken to be positive if the curve turns in the same direction as the surface's chosen normal, otherwise negative. The mean curvature of that point is the average of the maximal and minimal curvature and the Gaussian curvature is equal to the product of the principal curvatures. The positive value of the curvature indicates that the surface in this point is locally convex where the negative value indicates a locally concave shape. In this paper, we use the modified T-Algorithm [**19**], to estimate the Principal Curvatures and the Darboux Frame of a particular vertex on the 3D polygon mesh. And the normal direction of this vertex is computed based on the method provided by OuYang and Feng [**20**].

In Figure 2, all features are presented with its mean curvature plot, which means the color of the shape is associated to the range of mean curvature, where red indicates larger curvature and blue indicates small or even negative curvature.

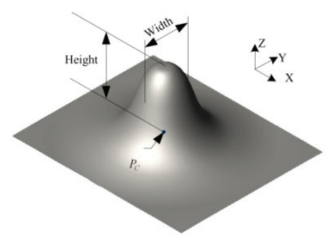


Figure 3: Feature parameterization

Generally, a protrusion feature is local deformation of a freeform shape, centered by one or several apexes (bottom for concaved deformation). In conventional approaches, parameters, which are the quantitative characteristics, are often used to change the intrinsic properties of features. Figure 3 presents a possible parameterization of such a feature. From curvature point of view, the intrinsic properties of a given deformable freeform features always have a certain type of curvature pattern. For a protrusion feature, it is always circulated by a stripe of changes of curvature in the curvature plot. Here, feature handles, which will be offered to the designer, are defined based on the curvature information. For protrusion features, they can be:

1. Position, size and shape of the circular stripe. They are controls by the feature line, most probably the maximum or the minimum in the stripe as in Figure 4;

2. Position, size and shape of apex(es) area(s) inside the circular shape. They are also controls by the feature lines/areas in those area as Figure 4;
3. The relations between 1 and 2. They are controlled by the lofting surface constructed by the feature lines and guide curve as Figure 4.

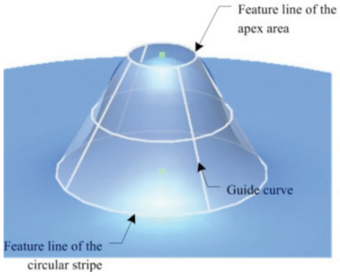


Figure 4: Handles of a protrusion feature

Handles is a superset of parameters, for example the height parameter in Figure 3 is the one of the handle 3. The manipulations of handles will be transferred to the freeform shape by freeform deformation techniques [**9**,**21**]. Generally handles offer the user more controls of the given freeform shape in an intuitive manner. For example, if the users want to enlarge the feature area of a protrusion feature, he/she can simply adjust the feature line which represents the circular stripe in the curvature plot.

Comparing to the protrusion feature, an extrusion feature always develops though the whole shape and form two external boundaries, the handles for manipulating extrusion features are:

1. Position, size and shape of the stripe of extreme values in the curvature plot in one side;
2. Position, size and shape of the stripe of extreme values in the curvature plot in another side;
3. Position, size and shape of apex(es) area(s) between two external stripes.
4. The relations among 1, 2 and 3.

Figure 5 illustrates some possible manipulations controlled by handles of an extrusion feature. In Figure 5(a), the original shape containing a protrusion feature is presented. By adjusting handles of the apex area inside the feature, the shape of the apex area was changed in Figure 5(b). In Figure 5(c), handles in the two external stripes were modified resulting in a wider extrusion shape. Finally the relations between the external stripe and apex area were adjusted.

Though the bent feature is a local feature, it always influences the global development of the freeform shape. For bent features, handles are

1. Position, size and shape of the stripe of extreme values in the curvature plot in one side;
2. Position, size and shape of the stripe of extreme values in the curvature plot in another side;
3. The relations between 1, 2.

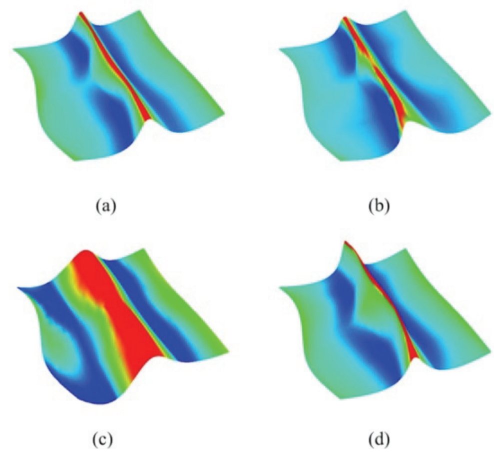
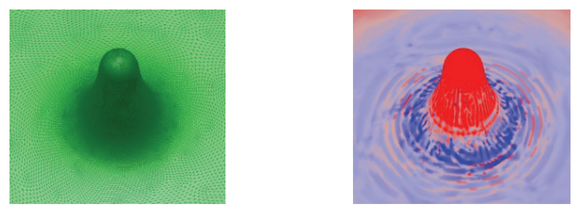


Figure 5: Using handles to manipulate an extrusion feature

The elimination feature is quite different from deformable feature due to the cut area. To specify handles for the elimination feature, boundaries of the cut area (Figure 2(d)) should be considered. We defined handles of the elimination feature as following:

1. Boundaries of the cut area;
2. Possible circular stripes of extreme curvature values in the curvature plot around the boundary;
3. The relation between 1 and 2.



(a) A protrusion feature (b) Its mean curvature plot

Figure 6: Curvature of a protrusion feature

In current stage of research, the handle generation is an interactive process. By offering the feature points/lines/areas, the user can specify the handle(s) he/she interests in. For this, identifying the extreme values, i.e., feature points/lines/areas, in the curvature plot is the key issue. For an ideal freeform feature, by a simple threshold analysis on the curvature plot, those feature items can be easily found by taking the extreme position in both negative and positive curvature areas. But in many practical cases, the curvature plot is always influenced by the noises of the shape and interferences from other features. This is especially true when the shape is retrieved from 3D digitizing. Figure 6(a) presents a "visually smooth" protrusion feature based on the mesh representation. In Figure 6(b), its mean curvature plot is displayed, which is still not good enough for extracting feature items.

3 PRE-PROCESSING

A conventional approach for solving this problem is to use different surface smoothing tools, such as Laplacian filter [**22**, **23**], Taobin filer [**24**], to smooth the geometry of the shape. Laplacian smoothing is a simple, effective and one of the most popular tools in smoothing a given geometry. Considering a triangular shape

$$A^3 = \{T_j \mid j = 0, m_T, V_i^A \mid i = 0, m_A, C_i^A \mid i = 0, m_A\}$$

in $\mathbb{R}^3$, where T_j is a triangle in A^3, $(m_T + 1)$ is number of triangles in A^3, V_i^A is a vertex in the shape, C_i^A is a type of curvature value on the vertex based on [**19**], $(m_A + 1)$ is the number of vertices in this mesh. For a particular vertex V_i^A in A^3, an umbrella-operator U can be defined as

$$U(V_i^A) = \sum_{j=0}^{m} \omega_j V_{qj} \bigg/ \sum_{j=0}^{m} \omega_j - V_i^A.$$

where V_{qj} are the neighbors of V_i^A , $m + 1$ is the number of the neighbors and ω_j is the weight of the neighbor V_{qj} With the umbrella-operator U, a new position of vertex V_i^A can be computed as

$$V_i^{Anew} = V_i^A + \eta U(V_i^A).$$

In the equation, η is a constant and typically it is a small positive number, such as 0.2. Using the Laplacian smoothing tool, this process is repeated until a smoothed geometric shape is reached. Laplacian smoothing can quickly remove the high frequency noise of the shape and it does not change the topology of the geometry. But it also does not guarantee an improvement in the mesh quality and after smoothing, each vertex position will be moved even originally the geometry is perfectly smooth. Thus the original geometry is always modified [**25**].

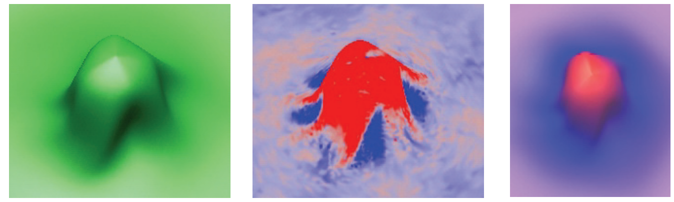
In the proposed method, to maximally preserve the original geometry for later manipulation, with the computed curvature values on each vertex of the geometric shape, a curvature based umbrella-operator U^C is constructed as

$$U^C(C_i^A) = \sum_{j=0}^{m} \omega_j C_{qj}^A \bigg/ \sum_{j=0}^{m} \omega_j - C_i^A,$$

where C_i^A is a type of curvature on vertex V_i^A, C_{qj}^A is the same type of curvature of its neighbors and $m + 1$ is the number of the neighbors, ω_j is the weight of C_{qj}^A. U^C is a one dimensional function, which is different from the umbrella-operator U used in the geometric domain. It directly uses a type of curvature values at each vertex in the computing. With the umbrella-operator U^C, the smoothed curvature associated to vertex V_i can be computed as

$$C_i^{Anew} = C_i^A + \eta U^C(C_i^A).$$

Same as the Laplacian smoothing tool, η is a small positive scalar coefficient, the typical range is from 0.1 to 1. Figure 6 demonstrates the effectiveness of the proposed smoothing algorithm. In Figure 7(a), a protrusion feature is presented where its curvature plot is displayed in Figure 7(b). Based on the proposed smoothing algorithm, the smoothing result of the mean curvature is presented in Figure 7(c).



(a) (b) (c)

Figure 7: Curvature patterns after pre-processing

4 IDENTIFYING FEATURE ITEMS

The pre-processing tool presented in this paper provided a reasonable smoothed 3D curvature plot without any changes of the given freeform shape. For finding the feature points/lines/areas, which are the basis the feature handles, different from existing 3D approaches, we tried to change the 3D problem to a 2D problem with mappings. The merits of such an operation are: a. the problem of 3D feature items detection is reduced to 2D image processing problems, which have been extensively studied; b. Those feature items, and their relations are directly linked to feature handles, which will be used in manipulating freeform shapes via freeform deformation techniques. Generating lattice for freeform deformation following the developments of freeform shapes [**9**] is much easier in 2D than in 3D space.

According to conformal geometry theory, each 3D shape with disk topology can be mapped to a 2D domain through a global optimization and the resulting map is a diffeomorphism, i.e., one-to-one and onto [**26**]. Among those maps, Quasi-conformal maps, which are almost conformal, can control the angel distortion in a manner of uniformly bounded throughout their domain of definition [**27**]. Quasi-conformal maps include harmonic maps, conformal maps, and least-squares conformal maps, and they have been frequently used in computer vision and graphics applications, especially for texture mapping. Among them, least-squares conformal maps [**12**] was selected and implemented due to its minimum angle deformations and non-uniform scaling.

Given a mesh A^3, a smooth target mapping $\mathcal{U}: A^3 \rightarrow A^2$ is constructed, where A^2 is a planar mesh and $A^2 = \{T_j \mid j = 0, m_T, V_i^P \mid i = 0, m_A, C_i^P \mid i = 0, m_A\}$, T_j is a planar triangle and $(m_T + 1)$ is the number of triangles, $V_i^P = (u_i, v_i)$, which is a 2D vertex in the shape, C_i^A is the curvature values which is the same as the curvature value C_i^A of vertex V_i^A in A^3, $(m_A + 1)$ is the number of vertices. $\mathcal{U}$ is conformal on A^3 if and only if the Cauchy-Riemann equation

$$\frac{\partial \mathcal{U}}{\partial x} + i\frac{\partial \mathcal{U}}{\partial y} = 0$$

is satisfied. Here x and y are the coordinates of vertices V_i^A based on a local orthonormal basis (i.e., the normal is along the z-axis). Mathematically, the Cauchy-Riemann equation cannot be strictly enforced for each triangle in A^3. Thus, in least square conformal maps, it is satisfied from a least square sense. The minimization criterion

$$CR(A^3) = \sum_{T_j \in A^3} \int_{T_j} \frac{\partial \mathcal{U}}{\partial x} + i \frac{\partial \mathcal{U}}{\partial y} dA^3,$$

where T_j is a triangle in A^3. Suppose the desired mapping is linear on T_j , the minimization criterion $CR(A^3)$ is further elaborated as

$$CR(A^3) = \sum_{T_j \in A^3} \left| \frac{\partial \mathcal{U}}{\partial x} + i \frac{\partial \mathcal{U}}{\partial y} \right|^2 S(T_j),$$

where $S(T_j)$ is the area of triangle T_j. Consider $q_i = x_i + iy_i$ and $r_i = u_i + iv_i$ for $i = (0, m_{A)}$, they should satisfy $\mathcal{U}(r_i) = q_i$. A vector q then is arranged as $q = (q_f, q_p)$ where q_f has $(m_A + 1 - p)$ free coordinates, e.g., $f = (m_A + 1 - p)$, q_p contains p constrain point coordinates. Then, the minimization criterion is rewritten as

$$CR(A^3) = \left\| M_f q_f + M_p q_p \right\|^2,$$

where $M = (M_f, M_p)$, which is a sparse $(m_T + 1) \times (m_A + 1)$ matrix. In this equation, $CR(A^3)$ can be easily minimized by the conjunction gradient method [**28**]. Thus 2D coordinates (u_i, v_i) of each vertex V_i^P in the planar mesh A^2 are calculated in a lease square sense.

In A^2, each vertex is assigned the curvature value of the corresponding vertex in original mesh A^3. Generally, vertices in A^2 is not uniform, thus a raster image, which is defined to be a rectangular array of regularly sampled values, needs to be created as the input of the image processing. For generating this raster image, the minimum and maximum position in both u and v direction of A^2 are found first as (u_{min}, v_{min}) and (u_{max}, v_{max}), and the scan interval to create the image is specified as δ, which a half of the minimum edge length in mesh A^2, in both u and v direction. The image can be represented as

$$I = \{x, y, C_{xy} | x = 0, w; y = 0, h\},$$

where x is the image width, h is the image height and C_{xy} is the value in this position. Each pixel corresponds to a point (u_x, v_y) in A^2 where $u_x = u_{min} + x\delta$ and $v_y = v_{min} + y\delta$.

If (u_x, v_y) locates in a triangle with vertices $(u_1, v_1), (u_2, v_2)$ and (u_3, v_3), corresponding to curvature values C_1 , C_2 and C_3, respectively, suppose an linear function $k_1 u + k_2 v + k_3 = C$ exists for all position insides the triangle, the requirement that the values at the vertices leads to three linear equations

$$k_1 u_1 + k_2 v_1 + k_3 = C_1,$$

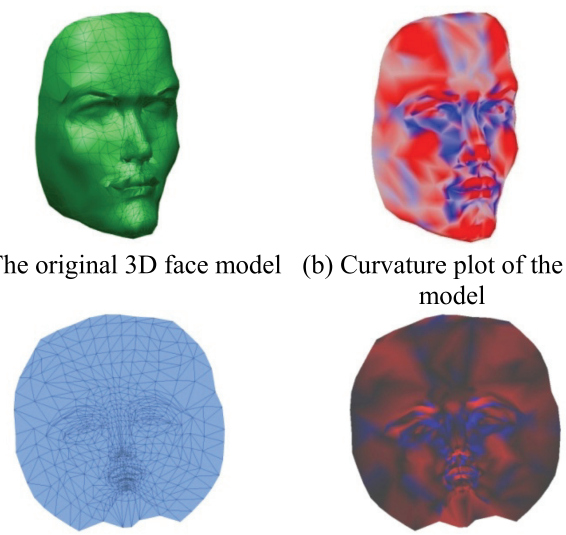
$$k_1 u_2 + k_2 v_2 + k_3 = C_2,$$

$$k_1 u_3 + k_2 v_3 + k_3 = C_3.$$

Coefficients k_1, k_2 and k_3 can be easily found by solving these linear equations. Thus for a point (u_x, v_y), its curvature value is defined as:

$$C_{xy} = k_1 u_x + k_2 v_y + k_3,$$

and it will be assign to point (x, y) in the raster image. If (u_x, v_y) is not surrounded by any triangles in A^2, C_{xy} will be marked and masked in the image processing process. Figure 7 presents the procedure of generating a mean curvature raster image of a 3D face model. In Figure 8(a), the original 3D model is presented where in Figure 8(b) the mean curvature plot of the model is shown. Using least square conformal maps, the 3D face model was mapped to a planar mesh as Figure 8(c) where the edges of triangles in the mesh are presented as well. In Figure 8(d), the mean curvature plot presented in Figure 8(b) is mapped to a raster image by the proposed algorithm.



(a) The original 3D face model (b) Curvature plot of the 3D model

(c) Mapping the 3D face model to 2D polygon mesh (d) 2D polygon mesh with curvature plot

Figure 8: Generating curvature raster images

For identifying different feature handles, two types of curvature raster image are generated, referring to two types of curvature: minimum curvature and maximum curvature, respectively. In the first image I_v, maximum curvature values are used in the computing and in I_c, the minimum curvature value are deployed. Geometrically, I_v is mainly used to describe the convex area on the shape where the concave area is emphasized by I_c.

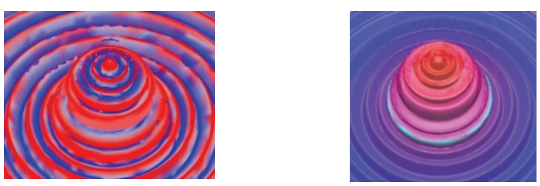
Identifying feature points and areas can be done by clustering points with a threshold in I_v and I_c. To identify feature lines, Sobel operator [**14**] is implemented. Technically, Sobel operator is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Sobel operator is either the corresponding gradient vector or the norm of this vector. The Sobel operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction and is therefore relatively inexpensive in terms of computations. On the other hand, the gradient approximation which it produces is relatively crude, in particular for high frequency variations in the image. For this, Discreet Wavelet Transformation (DWT)[**14**] denoise tool is applied first to remove possible high frequency noises which is not suppressed by pre-processing tool. Mathematically the Sobel operator consists of a pair of 3×3 convolution kernels as:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}.$$

For curvature raster image $\mathrm{I_v}$, the gradient approximation is

$$G = \sqrt{(S_x \otimes I_v)^2 + \left(S_y \otimes I_v\right)^2}.$$

Gradient approximation of $\mathrm{I_c}$ can be deduced as well. With the found feature items in $\mathrm{I_v}$ and $\mathrm{I_c}$, based on the definition, feature handles can be established though the user-computer interactions. Furthermore, lattices for FFD can be generated following the specified handles [**9**]. Space limitations prevent detailed discussion of the deformation process, and it will appear in the authors' future works.



(a) The original curvature plot (b) After smoothing (η = 1 and iteration times = 100)

Figure 9: Curvature plot of a 3D model

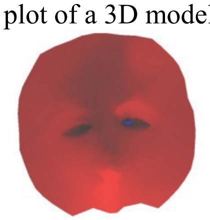


Figure 10: Over-smoothed curvature image of the model in Figure 8

5 RESULTS AND DISCUSSION

Using C++ and Matlab®, the proposed freeform feature handles recognition algorithm was implemented and the visual interface was provided by Rapidform®. In this section, we use several scenarios to test different aspects of the algorithm. In Figure 9(a), a protrusion feature interfered by many smaller extrusion features (the revolved shapes) is presented with its curvature plot. Then the pre-processing tool was applied on the shape, the pre-processing tool has three key parameters: a. the depth of neighbors, either based on the topology or based on the distance; b. coefficient η; c. iteration times. In Figure 9(b), by setting the depth of topological neighbors to 2, η to 1 and iteration times to 100, high frequency noises from the revolved protrusion feature are removed and the curvature plot is presented.

If the parameters in the pre-processing are not well-specified, in many cases it leads to an over-smoothed curvature plot/image, where detail features may disappear. Figure 10 presents such a curvature image based on the 3D models and 2D meshes presented in Figure 7. Here the depth of topological neighbors is set to 2, η is set to 1 and iteration times to 18. Comparing to Figure 7(d), the detail feature items around the nose and eyes are vanished due to the relatively sparse polygons in the mesh and the large iteration numbers. Thus, feature handles cannot be correctly generated based on this curvature image.

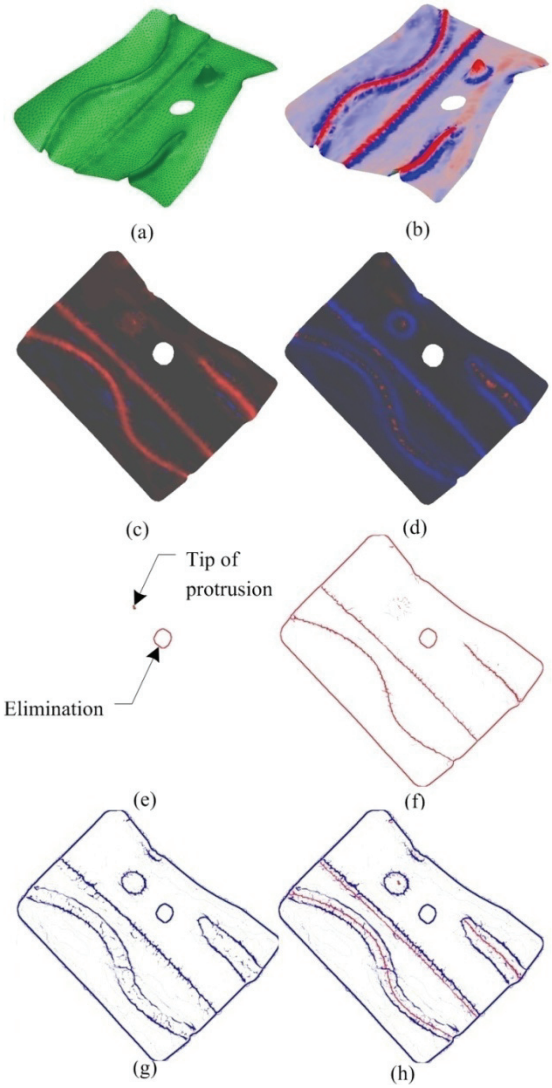


Figure 11: Identifying feature items a freeform shape

Figure 11 presents a complete process of finding feature handles on a freeform shape. In Figure 11(a), a freeform shape contains several freeform features is presented. Those features are a protrusion feature, two extrusion features, a bent feature and an elimination feature. The mean curvature plot of the original shape is displayed in Figure 11(b). After pre-processing, the curvature image I_v, which uses the maximum curvature, is presented in Figure 11(c) and the minimum curvature image I_c is shown in Figure 11(d). By boundary analysis and clustering extreme values in I_v, the apex of a protrusion and the boundary of an elimination were found as Figure 11(e). Then both I_v and I_c are denoised, and the stripes of extreme curvature values in curvature images were eroded. With Sobel edge detector, feature lines in the convex area and concave are were found as Figure 11(f) and (g), respectively. The synthesis of those items is presented in Figure 11(h) where feature handles can be established following the definitions.

Currently, our program is not specially optimized for speed. In our approach, curvature computing and image processing is fast, but the least square conformal map is an computing expensive, depending on the numbers of triangles. For example, mapping a 3D mesh with 763 vertices and 865 triangles to 2D only cost 0.079 second. But for a 3D mesh with 35749 vertices and 70660 triangles, it costs 113.109 second on a Pentium 4 3.6GHZ computer.

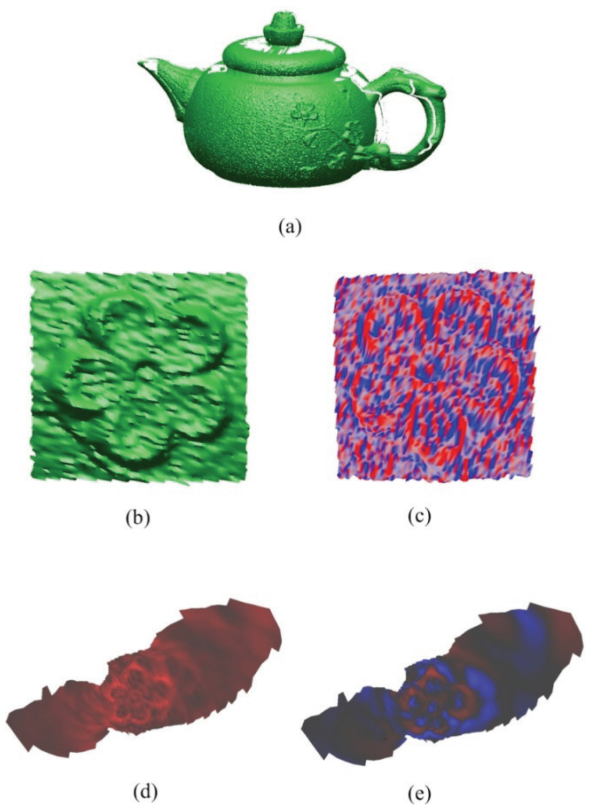


Figure 12: Feature handles in a digitized shape

In Figure 12(a), a digitized tea pot is presented and a part of its surface was selected as in Figure 12(b). With its curvature plot in Figure 12(d), it was found that the original shape contains a lot of noise. With the proposed method, feature handles were identified where I_v is presented in Figure 12(d) and a synthesis of I_v and I_c is shown in Figure 12(e). In this image, we found that this feature is mainly a protrusion feature with several convex and concave areas inside.

6 CONCLUSION AND PROSPECTS

In this paper, an approach to identify freeform feature handles is presented. Based on the patterns of freeform feature curvature plots, the feature handles concept is proposed in order to offer user more flexibility to modify a freeform shape. To quickly remove the noise introduced in the shape retrieval process, inspired by Laplacian smoothing methods, a pre-processing tool is developed and controlled by three key parameters. For easily indentifying feature items and generating lattice for later manipulation, 3D shapes are projected to 2D planar polygon meshes via least square conformal maps. By assigning each vertex in the 2D mesh the curvature value of its corresponding vertex in the original 3D shape, a curvature raster image is created and processed by image processing tools. These detected feature items forms feature handles for later manipulations.

The presented research is the pilot works of using feature handles to manipulate existing freeform shapes. Current research is directed towards two directions: improving the existing algorithms and finalizing the manipulation tools. Within current approach, a lot of human-computer interactions are required, such as specifying parameters for the pre-processing tool and the image processing tools, helping establishing handles, especially for relation handles. More user-friendly algorithms are under development to automate part of the process. Besides, feature interference problems are also to be tackled. Based on feature handles, an algorithm based on freeform deformation techniques is also to be developed for achieving more flexible shape manipulations.

7 ACKKNOWLEDGEMENTS

This research is sponsored by The Dutch National Scientific Founding (NWO) “Cross-platform Model-independent Geometric Modification” project, which is conducted by Faculty of Industrial Design Engineering, Delft University of Technology. The implementation of the proposed algorithm is based on Rapidform® software. Valuable comments from three reviewers are appreciated.

REFERENCES

[1] Vergeest J. S. M. and Horváth I., 2001, Parameterization of Freeform Features, Proceeding of Shape Modeling International, IMA-CNR, IEEE, Piscataway, 20-29.

[2] Ingle K. A., 1994, Reverse Engineering, McGraw-Hill, New York.

[3] Várady T., Martin R. R. and Cox J., 1997, Reverse engineering of geometric models-an introduction, Computer-Aided Design, 29 (4), 255-268.

[4] Chivate P. N. and Jablokow A. G., 1995, Review of surface representations and fitting for reverse engineering, Computer Integrated Manufacturing Systems, 8 (3), 193-204.

[5] Fontana M., Giannini F. and Meirana M., 1999, “A Freeform Feature Taxonomy”, Eurographics 99, Computer Graphics Forum, 18(3), 107-118.

[6] van den Berg, E., Bronsvoort W. F. and Vergeest J. S. M., 2002, “Freeform Feature Modeling: Concepts and Prospects”, Computer in Industry, 49(2), 217- 233.

[7] Nyirenda P.J., Bronsvoort W.F., Langerak T.R., Song Y, and Vergeest J.S.M, 2005, A generic taxonomy for defining freeform feature classes. Computer-Aided Design and Applications 2(1-4), 497-506.

[8] Shah J. J. and Mantyla M., 1995, Parametric and Feature Based CAD/CAM, Wiley-Interscience Publication, John Wiley Sons Inc.

[9] Song Y., Vergeest J. S. M. and Bronsvoort W. F., 2005, Fitting and manipulating freeform shapes using templates, Journal of Computing and Information Science in Engineering, 5(2), 86-94.

[10] Langerak T.R. Vergeest, J.S.M., 2007, A new framework for the definition and recognition of free form features, Journal of Engineering Design,18(5).

[11] Pizer S., Morse B., Eberly D., and Fritsch D., 1998, Zoom Invariant Vision of Figural Shape: The Mathematics of Cores, Computer Vision and Image Understanding, January.

[12] Lévy B., Petitjean S., Ray N., Maillot J., 2002, Least squares conformal maps for automatic texture atlas generation, ACM Transactions on Graphics, Volume 21 , Issue 3.

[13] Jagannathan A., Miller E. L., 2007, Three-Dimensional Surface Mesh, Segmentation Using Curvedness-Based, Region Growing Approach, IEEE TPAMI, 29(12).

[14] Gonzalez R.C. and Woods R.E., 2002, Digital Image Processing, 2nd Edition (DIP/2e), Prentice Hall.

[15] Mortara M. and Patane G., 2002, Affine-Invariant Skeleton of 3D Shapes, Proc. Shape Modeling Int’l, 1-8.

[16] Sherbrooke E.C., Patrikalakis N.M., and Brisson E., 1996, An Algorithm for Medial Axis Transform of 3D Polyhedral Solids,” IEEE Trans. Visualization and Computer Graphics, 2(1).

[17] Page D.L., Koschan A.F., and Abidi M.A., 2003, Perception-Based 3D Triangle Mesh Segmentation Using Fast Marching Watersheds, Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2, 27-32.

[18] Stillwell J., 1992, Geometry of Surfaces, Springer.

[19] Hameiri E. and Shimshoni I., Estimating the Principal Curvatures and the Darboux Frame from Real 3D Range Data, First International Symposium on 3D Data Processing Visualization and Transmission, 2002.

[20] OuYang D., Feng H.Y., On the normal vector estimation for point cloud data from smooth surfaces, Computer-Aided Design, 37, 2005, 1071–1079.

[21] Sederberg, T. W. and Parry, S. R., 1986, Freeform deformations of solid geometric models, Computer Graphics, SIGGRAPH, 20(4), 151-160.

[22] Haralick R. and Shapiro L., 1992, Computer and Robot Vision, Vol 1, Addison-Wesley Publishing Company, 346-351.

[23] Field D. A., 1984, Laplacian smoothing and delaunay triangulations. Comm. Appl. Numer. Methods, 4, 709–712.

[24] Taubin G., 2003, New Results in Signal Processing and Compression of Polygon Meshes. Shape Modeling International 2003, 45-48.

[25] Song Y. and Vergeest J.S.M., 2007, Matching geometric shapes in 4D space incorporating curvature information, 2007 ASME International Design Engineering Technical Conferences, IDETC07, September 4-7, Las Vegas, Nevada, USA.

[26] Anderson G.D., Vamanamurthy M.K. and Vuorinen M.K., 1997, Conformal Invariants, Inequalities, and Quasiconformal Mappings. Wiley-Interscience.

[27] Wang S., Wang Y., Jin M., Gu D.X., 2007, Samaras D., Conformal Geometry and Its Applications on 3D Shape Matching, Recognition, and Stitching, IEEE Trans. PAMI, 28(7).

[28] Press W.H., Teukolsky S. A. Vetterling W.T., Flannery B.P., 2007, Numerical Recipes: The Art of Scientific Computing, 3rd edition, Cambridge University Press.