Multiresolution Free Form Object Modeling with Point Sampled Geometry

Yong-Jin Liu[1], Kai Tang[2], and Matthew Ming-Fai Yuen[2]

[1] *Department of Industrial Engineering and Engineering Management, Hong Kong University of Science and Technology, Hong Kong, P.R. China*

[2] *Department of Mechanical Engineering, Hong Kong University of Science and Technology Hong Kong, P.R. China*

E-mail: {liuyj, mektang, meymf}@ust.hk

Revised June 11, 2004.

Abstract In this paper an efficient framework for the creation of 3D digital content with point sampled geometry is proposed. A new hierarchy of shape representations with three levels is adopted in this framework. Based on this new hierarchical shape representation, the proposed framework offers concise integration of various volumetric- and surface-based modeling techniques, such as Boolean operation, offset, blending, free-form deformation, parameterization and texture mapping, and thus simplifies the complete modeling process. Previously to achieve the same goal, several separated algorithms had to be used independently with inconsistent volumetric and surface representations of the free-form object. Both graphics and industrial applications are presented to demonstrate the effectiveness and efficiency of the proposed framework.

Keywords object hierarchy and geometric transformation, feature representation, three-dimensional graphics and realism, system and information processing

1 Introduction

Three dimensional (3-D) geometric models have been widely used for many years for engineering simulation and visualization. Nowadays with the decreases in the cost of commodity computers and the increases of Internet bandwidth, complex free-form models are made accessible to a much larger audience; the potential of using these models is expanded beyond now the well established game market to new applications ranging from virtual museums to e-commerce.

Traditionally parametric surfaces such as tensor product B-spline patches are the most adopted form for free-form shape modeling[1] and the generalization of non-uniform rational B-splines (NURBS) is considered as the de facto CAD standard. Recently with the advances in 3-D digital photography and scanning technology, complex free-form models become ubiquitous through the processing of large sets of point samples.

In this paper we study multiresolution free form models represented by sets of points which we referred to as *point sampled geometry*[2]. Since point sets are discrete in nature while physical objects to be described must have connected volume bounded by continuous surfaces, to build an efficient framework to process point-sampled geometry, we have to answer the following questions.

1) Which mathematic form the point sampled geometry should be represented in? For example, a parametric surface, or an implicit surface, or else that is continuous and approximates/interpolates the set of points.

2) Given a specified mathematic form, how to render/visualize the point-sampled geometry efficiently? For example, any rendering scheme that supports real-time shape editing (or modification).

3) Does that chosen mathematic form support various shape editing operations? If it does, can those operations be efficiently implemented?

In this paper we propose a general framework for multiresolution free-form modeling in point sampled geometry with the answer to the above three questions. This paper, which is condensed from the first author's dissertation[2], is organized as follows. The related work is presented in Section 2. A new hierarchy of shape representations that is adopted in the proposed framework is presented in Section 3. The necessary transformation procedure among the proposed shape hierarchy with an associated geometric modeling/editing toolbox is presented in Sections 4 and 5. In Section 6 we present both industrial and graphical applications to demonstrate the effectiveness and efficiency of

*Regular Paper

the proposed framework. Finally our concluding remarks are presented in Section 7.

2 Previous Work

2.1 Surface Reconstruction Techniques

The complexity of geometric model is determined by the mathematic form in which the shape is represented. A large variety of shape representations is summarized in [3] and can be noticeably classified into three major categories[①]: piecewise linear, parametric and implicit representations.

The piecewise linear surface reconstruction techniques, e.g., the Delaunay triangulation algorithm[4], usually require strict sampling criteria on point sets and do not work well at sharp features such as creases and cusps. The parametric methods are primarily developed to reconstruct surfaces with prescribed topological types[5], mostly homeomorphic to an open disc, and thus user intervention is inevitable for setting up patch net work for surfaces of arbitrary topological type. The implicit methods build a 3-D scalar field f over the object space and the underlying surface is represented by an isosurface $f = c$, where c is a suitable level set value. Owing to this implicit representation, the implicit surfaces put no restrictions on the object's topological type. State-of-the-art implicit models include radial basis functions[6], moving least squares methods[7] and level set methods[8].

2.2 Rendering and Multiresolution Techniques

In aforementioned three surface representations, piecewise linear (polygonal) surfaces are the most efficient representation for rendering: currently the triangle is the only surface patch that is directly supported by computer graphics hardware and even by most software rendering methods[9]. Usually parametric surfaces and implicit surfaces (including CSG and voxel representations) are converted into polygonal meshes prior to rendering.

It is often a desirable feature inherent in a rendering system to have a constant frame rate, e.g., real-time rendering in the context of interactive graphics often requires 30 frames per second (fps) or higher. This can be achieved if the objects are represented in a multiresolution fashion. Multiresolution modeling techniques construct a hierarchy of details $\{W^0, W^1, \ldots\}$ over a base domain V^0 to describe a geometric object $V^J = V^0 \oplus W^0 \oplus \cdots$ in such a way that is particularly suitable for data compression, progressive transmission, and controllable level-of-details rendering. Two classes exist for mesh multiresolution modeling. The first class is typified by the progressive mesh[10] that works with irregular meshes and the second works with semi-regular meshes[11]. Though the first class can handle meshes with arbitrary connectivity, the nestedness property $V^{i+1} = V^i \oplus W^i$ of the sequence of approximate spaces in the standard wavelet theory can no longer be maintained any more. As a result, extra information must be stored and, the analysis and synthesis operators are not uniquely determined. The multiresolution techniques can be used not only for rendering but also for shape editing.

2.3 Shape Editing Operations

Towards a complete geometric toolbox for shape editing, the nine operations in Table 1 are widely used[12]. Their implementation efficiency and performance depend on which shape representation is used; the remarks are also summarized in Table 1.

Table 1. Shape Editing Operations in a Complete Toolbox for Geometric Modeling and Their Performance under Three Shape Representations

		Parametric surf.	Implicit surf.	Polygonal surf.
1	Local shape deformation	√	×	×
2	Global shape deformation	√	√	×
3	Boolean operation	×	√	×
4	Blending	×	√	×
5	Offset	√	√	×
6	Metamorphosis	√	√	×
7	Level-of-details	√	×	√
8	Texture mapping	√	×	×
9	Rendering	×	×	√

3 New Hierarchy of Point-Based Shape Representation

We propose a framework for free-form object modeling with a new hierarchy of shape representations for point-sampled geometry. The framework processes geometry in three stages. By going through these stages the represented shape is refined and the advantages inherent in different traditional shape representations, i.e., parametric, implicit and piecewise linear representations, are ex-

[①] There are two main types of modelers, solid-based and surface-based. Here we consider surface-based modelers. Note that in our classification, the implicit representation has a natural link to solid-based models.

ploited in the proposed framework to achieve maximum efficiency.

We propose to use the following hierarchy of point-based shape representations in our framework for free-form object modeling.

Level 1. The shape is described as a set of unstructured sample points $S = (s_1, s_2, \ldots, s_n)$. Geometrically the shape is discretized as a set of samples (cf. Fig.1(a)).

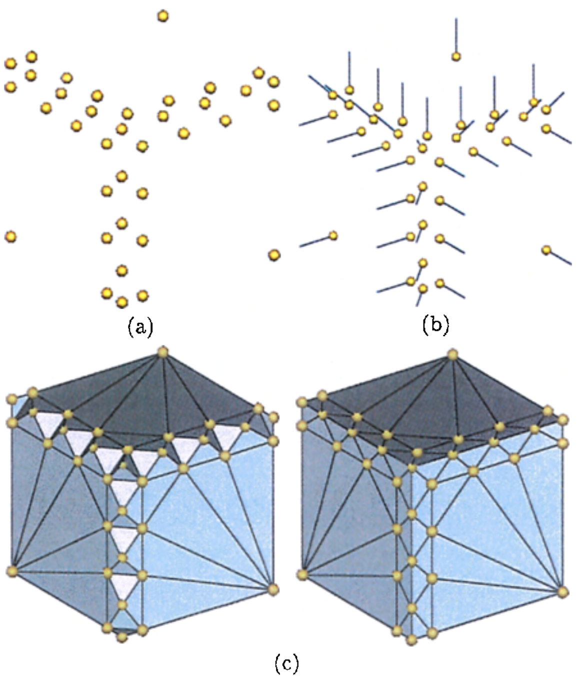


Fig.1. Proposed hierarchy of shape representation. (a) Sample points S. (b) Hermite samples $H = (S, N)$. (c) Two different triangulations with the same samples S, only the right one satisfies the normal constraints.

Level 2. The shape is described as a set of samples each associated with a normal (such data is called *surfel* in computer graphics[13] or called *Hermite data*[1] in computer-aided geometric design), $H = (S', N) = \{(x_i, y_i, z_i, nx_i, ny_i, nz_i) | i \in \#S'\}$, cf. Fig.1(b). It has been observed (cf. [14]) that the normal plays an important role on sharp feature identification, see also Fig.1(c) for an illustration. Note that each sample (x_i, y_i, z_i) associated with its normal (nx_i, ny_i, nz_i) indicates the tangent plane to the underlying surface at point (x_i, y_i, z_i). Then geometrically the shape can be interpreted as a local combination of these tangent planes, which offers a C^0-continuous representation of the shape.

Level 3. The shape is described as a set of structured samples (cf. Fig.1(c)), i.e., a mesh $M = (V, K)$, where $V \in \boldsymbol{R}^3$ is the set of point positions used to determine the geometric realization of M, and K encodes the model's structure information serving as the topological realization of M. The formal definition of M will be presented in Subsection 5.2. In the framework we build a multiresolution model by keeping refining a base mesh as to be presented in Section 5. By using appropriate refinement rules, it is possible to offer a continuous model of higher order than C^0 for the underlying shape, e.g., the Loop scheme and the Catmull-Clark scheme are the generalizations of C^2 quartic triangular B-spline and tensor-product bicubic spline, respectively[15].

In [2], it is shown that under certain conditions these three level shape representations are *equivalent* in a measure induced by two-sided Hausdorff distance. We say the shape representation is refined, when we transform the shape representation from Level 1 to Level 3 via Level 2, since Level 3 representation provides all the information *explicitly* and Level 1 representation encodes most information *implicitly*. The following arguments are in order.

Assume floating-point (4 bytes) and unsigned integer (2 bytes) values are used. For Level 1 representation, each sample requires $3 \times 4 = 12$ bytes. For Level 2 representation, each sample requires $2 \times 3 \times 4 = 24$ bytes. For Level 3 representation, by Euler's formula $V - E + F = 2$, a 2-manifold triangle mesh ($3F \approx 2E$) with n vertices contains $m \approx 2n$ triangles. If the simplest indexed mesh representation is used, $3 \times 4 \times n = 12n$ bytes are required for the geometry encoding (i.e., the geometric positions of the vertices) and $3 \times 2 \times m = 6m \approx 12n$ bytes are required for the topology encoding (i.e., each triangle referencing three vertices). The other popular data structures for 2-manifold meshes such as winged-edge, half-edge and double-edge structures[16] require much more memory for the topological information. Obviously the choice of a shape representation is a compromise between memory requirements and performance gains, since the more explicit information is stored, the more memory is required, but the much faster the access to the required information.

In the proposed framework for point-sampled geometry, we start with Level 1 shape representation and we choose to end with Level 3 representation as the output for visualization with the following due considerations.

1) The pure point-based representation is only competitive when describing a free form model with both uniformly-sized and highly-detailed

geometry[13]. If the model has a large flat or smooth patches of low curvature variation, polygonal models can be more compact and faster to draw. Moreover, in the case that the spacing of samples is large compared to screen pixel size, polygons offer higher visual quality, especially near sharp edges and corners.

2) The triangle can serve as a basic surface primitive to adaptively approximate smooth free form geometry and the triangle is currently the only surface patch that is directly supported by industrial computer graphics hardware.

4 Shape Transformation from Level 1 to Level 2

There are two steps for shape transformation from Level 1 to Level 2:

Step 1. Given a set of samples $S = (s_1, s_2, \ldots, s_n)$, $\forall s_i \in S$, estimate an initial normal $n(s_i)$ from a local neighborhood of s_i.

Step 2. Consistently orient the normal field attached to S such that the surface orientation is correctly indicated.

A 2D example illustrating these two steps is shown in Fig.2. More 3D examples illustrating the hole filling effect are presented in [2]. To algorithmically realize these two steps, a variant[14] of the algorithm proposed in [17] with a new adaptive influence radius for hole filling is briefly summarized as follows.

For each sample point $s_i \in S$, k sample points nearest to s_i, denoted by $Nbhd(s_i)$, are collected by applying an optimal k-nearest-neighbor search algorithm with a range tree structure[16]. From $Nbhd(s_i)$, a tangent plane $Tp(s_i) = (s_i, n(s_i))$ is indicated by solving a least squares fitting problem, where $n(s_i)$ is a unit normal vector associated to s_i. To determine the underlying surface orientation at the position of each sample point, $\forall s_i \in S$, each point $s_j \in Nbhd(s_i)$ is connected to point s_i. This results in a directed graph $G = (V, E)$. Each edge $(s_i, s_j) \in E$ in G is assigned a cost $|1 - n(s_i)^{\mathrm{T}} \cdot n(s_j)|$. Starting at an arbitrary vertex, the minimum spanning tree (MST) is extracted from the weighted graph G using the Prim's algorithm with a Fibonacci heap as its priority queue[18]. The MST is further traversed in a depth-first search to propagate the orientation. Then the following steps are taken to numerically determine a scalar field which assigns to an arbitrary point $p \in R^3$ a value $f(p)$:

1) find the sample point $s_k \in S$ nearest to p;

2) if point p lies in a local region around s_k, i.e., $\|p - s_k\|_2 \leqslant r$, where r is a user-controlled influence radius, then $f(p) = (p - s_k)^{\mathrm{T}} \cdot n(s_k)$ and $\nabla f(p) = n(s_k)$;

3) else $f(p) = \infty$.

We note that the above scalar field representation (induced from Level 2 shape representation) has a natural extension suitable for implementation in an out-of-core fashion; this characteristic is extremely useful when the input data (in Level 1 representation) has a huge number of samples that cannot be processed in the core memory as a whole. See [19] for full details about this extension.

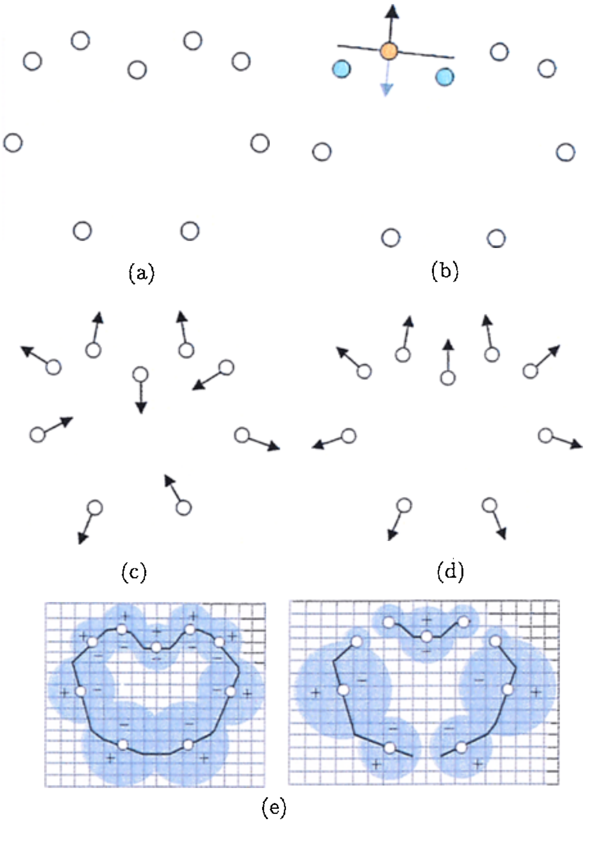


Fig.2. 2D example for shape transformation from Level 1 to Level 2. (a) Non-uniform samples. (b) Tangent/normal vectors determination. (c) Normal field before orientation. (d) Normal field after orientation . (e) Two scalar fields for curve generation with user-controlled adaptive influence radius.

4.1 Influence Radius Setting Strategy

Recall that in the above scalar field generation, the influence radius r controls the extent of the tangent plane at each sample point (cf. Fig.2(e)).

In real-world data the sample points seldom completely and uniformly cover the entire object, e.g., some portions of the physical surface are inaccessible to the sensor. To cope with non-uniform and incomplete data, we use an adaptive influence radius r. In our implementation, the radius r can be set either automatically or interactively:

- *Automatic setting.* When we search $Nbhd(s_i)$ for each $s_i \in S$, we set r at s_i to be the minimum value of radius with which a sphere centered at s_i contains $Nbhd(s_i)$.
- *Interactive setting.* Users can interactively control the output shape by assigning different values of influence radii to different regions (cf. Fig.2(e)). Note that user intervention is inevitable in the ambiguous situation that either there is indeed a hole or that hole should be closed.

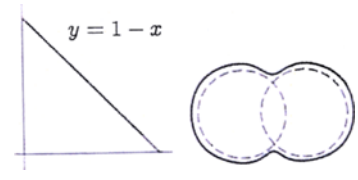
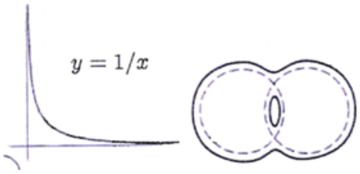
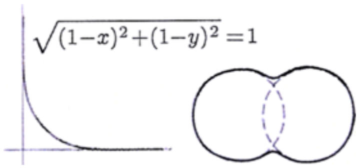
4.2 Volumetric-Based Shape Editing Operators

At this stage the scalar field based shape representation is best suitable for implementing the following volumetric-based shape editing operators.

Table 2. Boolean Operators on Two Objects A and B, Represented by Scalar Fields f_A and f_B, Respectively, with Positive Outside and Negative Inside Values

Action	Boolean operation	Implementation
Copy	Intersection: $A \cap B$	$\mathrm{Max}(f_A, f_B)$
Paste	Union: $A \cup B$	$\mathrm{Min}(f_A, f_B)$
Cut	Difference: $A \backslash B$	$\mathrm{Max}(f_A, -f_B)$

Table 3. Implicit Blending Operators[20]

	Linear blending: $B(f_1(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})) = \sum_{i=1}^{k} f_i(\boldsymbol{x}) - 1$
	Hyperbolic blending: $B(f_1(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})) = \prod_{i=1}^{k} f_i(\boldsymbol{x}) - 1$
	Super-elliptic blending: $B(f_1(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})) = 1 - \left(\sum_{i=1}^{k} [1 - \alpha_i f_i(\boldsymbol{x})]_+^{\beta} \right)^{1/\beta}$

Boolean operators on two objects A and B, represented by scalar fields f_A and f_B, respectively, are summarized in Table 2. *Implicit blending* operators that takes as its argument a list of objects $\{f_1(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})\}$ are summarized in Table 3. *Offset surfaces* are defined as a locus of the points which are at constant distance d along the normal from the generator surface. In the scalar field representation, the offset surface with distance d of an isosurface $f(\boldsymbol{x}) = 0$ is simply characterized by $f(\boldsymbol{x}) = d$. A *metamorphosis* that continuously/smoothly transforms one shape $f_1(\boldsymbol{x}) = 0$ into another $f_2(\boldsymbol{x}) = 0$ can be efficiently achieved by $f_t(\boldsymbol{x}) = f_1(\boldsymbol{x}) + tf_2(\boldsymbol{x}) = 0$, where t is an artificial time. All of these operators are implemented and the examples illustrating each of them are presented in [2]; among them one offset surface example is shown in Fig.3.

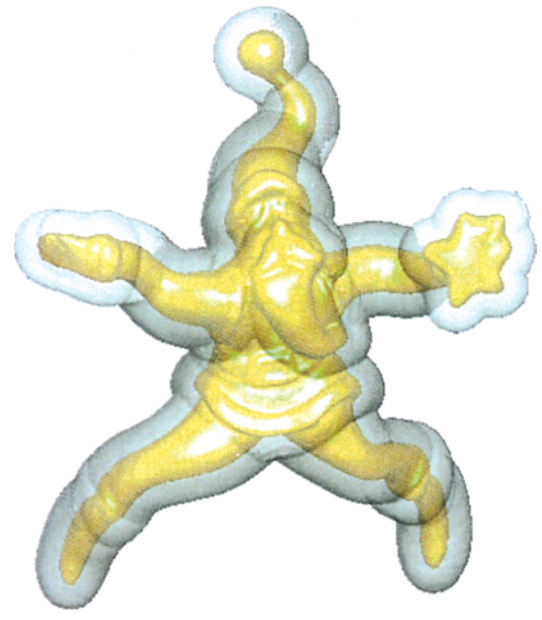


Fig.3. Bubble the Santa Claus. One offset surface rendered as semitransparent surface is combined with the original model to give the effect.

5 Shape Transformation from Level 2 to Level 3

Here we present a deformable method to polygonize the scalar field induced from the Level 2 shape representation to obtain a Level 3 shape representation. As already alluded in Section 2, the Level 3 representation is adopted in our framework as output for efficient rendering. The proposed deformable polygonization method recursively refines a base mesh to produce a detailed semi-regular mesh, as presented in details below. Due to the natural hierarchical structure with the regular refinement, the semi-regular meshes are usually more efficient than the irregular meshes to describe the objects (cf. Fig.4).

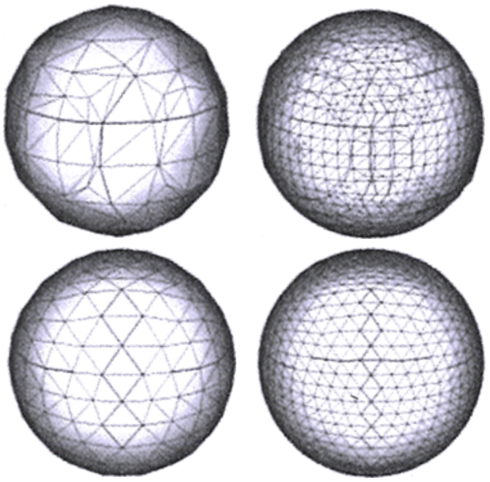


Fig.4. A spherical object represented with irregular meshes generated by the marching cubes algorithm (the top row) and semi-regular meshes generated by the proposed deformable method (the bottom row) with different resolutions, respectively.

5.1 Base Mesh Generation

The base mesh used in the proposed framework satisfies the following two criteria:

1) The base mesh is *homeomorphic* to the isosurface and is a *good approximation* of the isosurface;

2) All the vertices of the base mesh belong to the isosurface.

Two strategies are used in the framework for base mesh generation.

1) *Automatic setting.* For objects with highly geometric details and complex topological type, the required base mesh is automatically generated by applying the algorithm in [14] which generates an intermediate mesh first and then simplifies it to a base mesh with controllable face number and controllable triangle shape to a degree desired by users. See Fig.5 for an example.

2) *Interactive setting.* For objects with simple geometry, the user simply selects very few points from input Level 1 point data to create a base mesh. User interactivity can easily generate a good base mesh that satisfies the above two criteria. The key advantage of this interactive setting is its capacity of incorporating the semantic features into the base mesh which will greatly simplify the complete pipeline of free form object modeling. An example is shown in Fig.6.

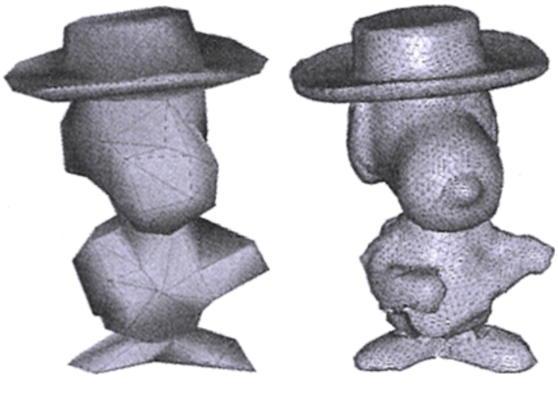


Fig.5. Semi-regular mesh reconstruction (the right) with an automatically generated base mesh (the left).

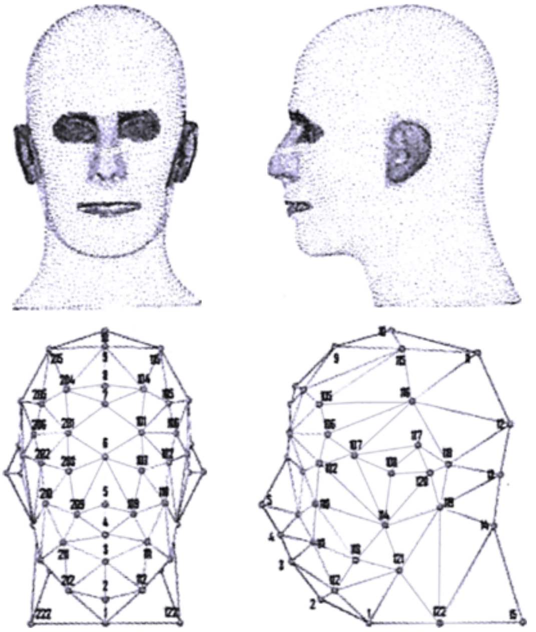


Fig.6. Interactive base mesh generation with semantic features[21].

5.2 Mesh Refinement

Basic notation. The proposed framework represents a 2-simplicial-mesh M by the pair (K, V). K is the topological realization of M, represented by $K = (VT, ET, FT)$. The 0-simplices set $VT \in K$ is called the vertex set; the 1-simplices set $ET \in K$ is the edge set; and the 2-simplices set $FT \in K$ is the face set. $V \in \boldsymbol{R}^3$ is the set of vertex positions used to determine the geometric realization of mesh M. We define $\pi_v(K)$ as the mapping from topological realization K to geometric realization V. Mapping π_v is called an *embedding* if $\pi_v(K)$ is not self-intersecting. Only a restricted set of vertex positions V results in an embedding π_v. To ensure an embedding map π_v, we establish the Criterion 1, in Subsection 5.1, for the base mesh.

Given a base mesh that is a good approximation and homeomorphic to the isosurface, it is split and expanded recursively to form a mesh hierarchy fitting the isosurface. Two methods have been implemented in [2] for mesh refinement: one is based on a displaced butterfly subdivision scheme and the other is presented in details below.

The mesh refinement starts from a base mesh, denoted by M^0. For a given mesh $M^i(K^i, V^i)$, the refinement from $M^i(K^i, V^i)$ to $M^{i+1}(K^{i+1}, V^{i+1})$ consists of three sub-steps: a splitting step, a projection step and a relaxation step. The whole refinement process can be cast as a multi-scale solver of the energy functional $E(M^i) = E_{\text{dist}}(M^i) + E_{\text{spring}}(M^i)$, where $E_{\text{dist}}(M^i) = \Sigma_j dist(f_j^i)$, $f_j^i \in FT^i$, and $dist(f_j^i)$ is the Hausdorff distance from point data S (or H) to the face f_j^i, and

$$E_{\text{spring}}(M^i) = \sum_{f_j^i} \sum_{k=1}^{3}{}_{e\sim f_j^i} \|e_k^i - e_{k+1}^i\|, \quad e_k^i \in ET^i$$

where $\sim$ denotes the incident relation.

5.2.1 Split Operators

For triangle mesh refinement, there are usually three types of refinement operators (cf. Fig.7). Vertex and edge split operators can achieve local mesh refinement and face split operator refines mesh globally. In our system, we are particularly interested in the face split operator, since applying it can create a mesh hierarchy with subdivision connectivity that can extend naturally to a compact and efficient multiresolution representation.

5.2.2 Stable Normal-Guided Projection Operators

The projection operator projects each new split vertex in V^{i+1}/V^i into the isosurface. Given a mesh M^i, denote the three vertex coordinates of the face $f_j^i \in FT^i$ as $p_{jk}^i(x_k, y_k, z_k)$, $k = 1, 2, 3$, $p_{jk}^i \in V^i$, the unit-face-normal vector $\boldsymbol{FN}^j(j)$ of face f_j^i is then $\boldsymbol{u}_j^i \times \boldsymbol{v}_j^i / \|\boldsymbol{u}_j^i \times \boldsymbol{v}_j^i\|$, where $\boldsymbol{u}_j^i = p_{j2}^i - p_{j1}^i$, $\boldsymbol{v}_j^i = p_{j3}^i - p_{j2}^i$, and the face-based edge-normal-vector $\boldsymbol{FEN}^i(k)$ of edge $e_k^i \in ET^i$ is $\Sigma \boldsymbol{FN}^i(k_m)/\|\Sigma \boldsymbol{FN}^i(k_m)\|$. In certain cases, it is observed that non-uniform face area will be propagated if projecting is kept along the direction $\boldsymbol{FEN}^i(k)$. We then propose to use an area weighted normal vector $\boldsymbol{AW\_FEN}^i(k)$ as the projection direction, which is defined as

$$\boldsymbol{AW\_FEN}^i(k) = \frac{\sum_{m=1}^{n} \Delta_{k_m}^i \boldsymbol{FN}^i(k_m)}{\left\| \sum_{m=1}^{n} \Delta_{k_m}^i \boldsymbol{FN}^i(k_m) \right\|} = \frac{\sum_{m=1}^{n} \boldsymbol{u}_{k_m}^i \times \boldsymbol{v}_{k_m}^i}{\left\| \sum_{m=1}^{n} \boldsymbol{u}_{k_m}^i \times \boldsymbol{v}_{k_m}^i \right\|}$$

where $n = 1$ if e_k^i lies on an open boundary, otherwise $n = 2$, Δ is the triangle area and k_m is the index of the face incident to edge e_k^i (cf. Fig.8(a)).

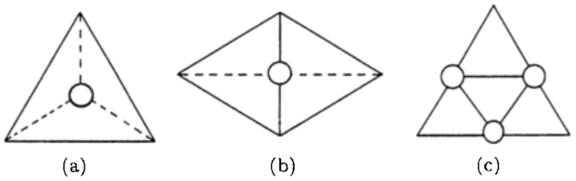


Fig.7. Split operators for mesh refinement. (a) Vertex split. (b) Edge split. (c) Face split.

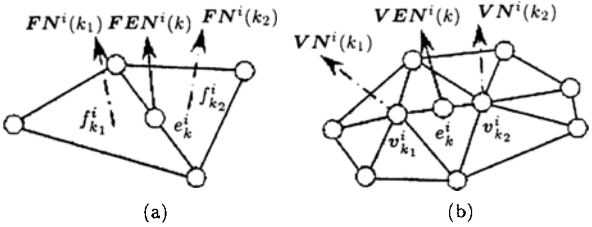


Fig.8. Edge normal vectors. (a) Face-based. (b) Vertex-based.

The projection operator projects each new split vertex along the ray $\pm\boldsymbol{AW\_FEN}^i(k)$, where k is the index of the edge in ET^i in which the split vertex lies. To guarantee that all of the new split vertices move to correct corresponding positions in the isosurface, the following normal-guided condition needs to be satisfied:

$$\boldsymbol{AW\_FEN}^i(k) \cdot \nabla f(v_k^{i+1}) > 0 \text{ and } f(v_k^{i+1}) = 0$$

In the area of high curvature variation, during mesh refinement, the mapping $\pi_v(K)$ from the topological domain to the geometric domain may be self-intersecting. If this case is detected, we switch to use a more computationally expensive vertex-based edge-normal-vector $\boldsymbol{VEN}^i$ to correct π_v:

$$\boldsymbol{VEN}^i(k) = \frac{\boldsymbol{VN}^i(k_1) + \boldsymbol{VN}^i(k_2)}{\|\boldsymbol{VN}^i(k_1) + \boldsymbol{VN}^i(k_2)\|}, \quad \text{and}$$

$$\boldsymbol{VN}^i(k) = \frac{\sum_{m\in Nbhd(k)} \Delta^i_{k_m} \boldsymbol{FN}^i(k_m)}{\left\| \sum_{m\in Nbhd(k)} \Delta^i_{k_m} \boldsymbol{FN}^i(k_m) \right\|}$$

where k_1 and k_2 are indices of vertices of edge e^i_k and $Nbhd(k)$ is the 1-ring neighbors of vertex k (cf. Fig.8(b)).

5.2.3 Relaxation Operator

Though the area-weighting normal projection can optimize the triangle shape to some extent, it is necessary to apply a relaxation operator after each projection to ensure good triangle aspect ratios. This can be regarded as applying internal forces by replacing each mesh edge with a spring. Guskov *et al.*[22] show that the solution to

$$\arg\min_{M^i}\{E_{\text{spring}}(M^i)\} =$$
$$\arg\min_{e^i_k\in ET^i}\Big\{ \sum_{f^i_j\in FT^i} \sum_{k=1}^{3} {}_{e\sim f^i_j} \|e^i_k - e^i_{k+1}\| \Big\}$$

can be computed locally. To optimize the triangle aspect ratio, $\forall v^i_k \in VT^i$, consider the shape cost function:

$$Sp(v^i_k) = \sum_{j\in Nbhd(k)} \|v^i_k - v^i_j\|^2$$

Minimizing this shape cost function is related to maximizing the area to perimeter ratios of the resulting triangles. This shape cost function can be reformulated as

$$\begin{aligned} Sp(v^i_k) = &\sum_{j\in Nbhd(k)} (v^i_k - v^i_j)^{\mathrm{T}}(v^i_k - v^i_j) = \\ &\sum_{j\in Nbhd(k)} [(v^i_k)^{\mathrm{T}} \cdot \boldsymbol{I} \cdot v^i_k - \\ &2(v^i_k)^{\mathrm{T}} \cdot v^i_j + (v^i_j)^{\mathrm{T}} \cdot v^i_j] = \\ &\Big[(v^i_k)^{\mathrm{T}} \cdot \Big(\sum_{j\in Nbhd(k)} \boldsymbol{I}\Big) \cdot v^i_k - 2(v^i_k)^{\mathrm{T}} \cdot \\ &\sum_{j\in Nbhd(k)} v^i_j + \sum_{j\in Nbhd(k)} \big((v^i_j)^{\mathrm{T}} \cdot v^i_j\big)\Big] \end{aligned}$$

Since the function $Sp(v^i_k)$ is quadric, by equating its derivative to zero, the solution to $\arg\min_v Sp(v^i_k)$ is

$$\bar{v}^i_k = \Big(\sum_{j\in Nbhd(k)} \boldsymbol{I}\Big)^{-1} \cdot \sum_{j\in Nbhd(k)} v^i_j = \frac{1}{n} \sum_{j\in Nbhd(k)} v^i_j$$

where $n = \#Nbhd(k)$.

5.2.4 Refinement Process

The mesh refinement process starts from the base mesh M^0 and recursively performs the steps of split, projection and relaxation to $M^i \to M^{i+1}$ until a final level J is reached. Noting that the detail-size less than the sampling density cannot be distinguished in a point-sampled geometry, we establish the stopping criterion for reaching the final level J as that, the Hausdorff distance from point set S to the mesh M^J is less than the sampling density of S.

One important characteristic of the proposed refinement process is that, for each iteration, V^i is a subset of V^{i+1}. As a result, the sequence $\{V^0, V^1, \ldots, V^J\}$ forms a nested approximate space of the object M^J, which is crucial in our multiresolution modeling scheme. In Fig.9 the refinement process is illustrated with the head model shown in Fig.6.

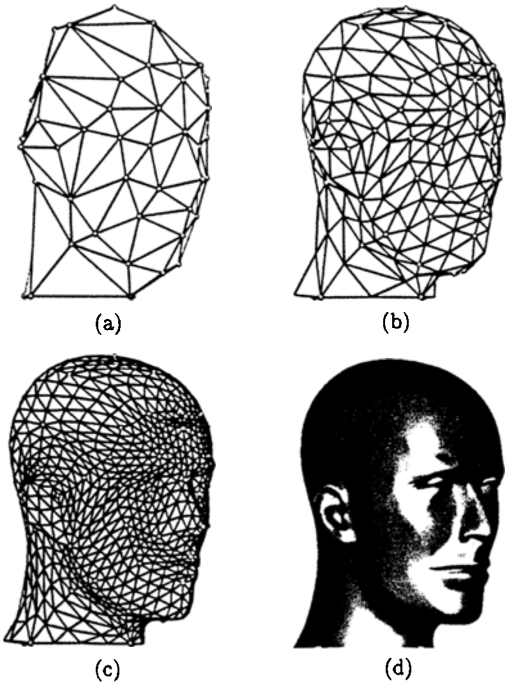


Fig.9. Mesh refinement process with the head model in Fig.6.

5.3 Parametric Shape Editing Operators

Parameterization. In a wide range of important applications in CAD/CAGD, it is often desirable to construct a parameterization of a highly detailed model over a simple parametric domain. If such a parameterization can be achieved, the highly detailed model can be viewed as a function over the parametric domain and the operations of shape editing/deformation (both locally and globally) and texture mapping becomes trivial. In our proposed deformable polygonization method, the mesh refinement process creates a mesh hierarchy which consists of a base mesh M^0 together with a set of hierarchical details. The parameterization of the detailed mesh M^J over the base mesh M^0 can be readily obtained by the 1–4 split rules. See Fig.10 for an illustration.

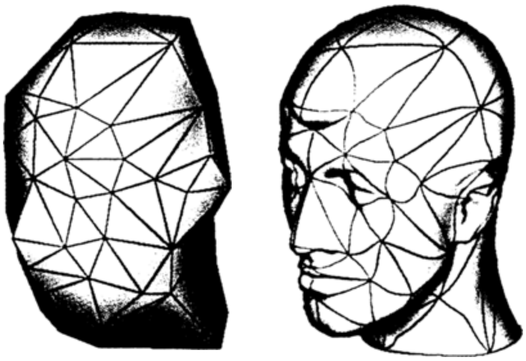


Fig.10. Parameterization with a deformed base mesh.

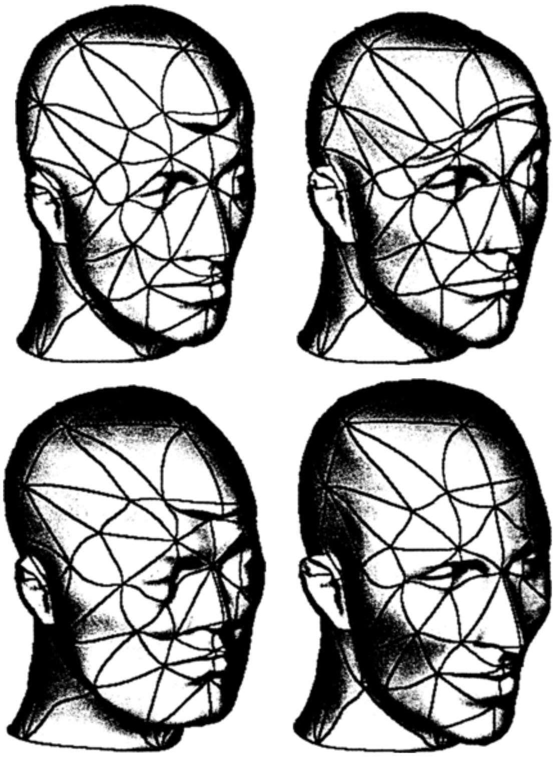


Fig.11. Multiresolution shape deformation.

Multiresolution shape deformation. Note that our Level 3 shape model is represented in two parts: a base mesh M^0 and a hierarchical detail set. Given this multilevel form, we can easily deform the model either locally or globally by deforming the base mesh and adjusting the details in different levels. See Figs.10 and 11 for illustrations.

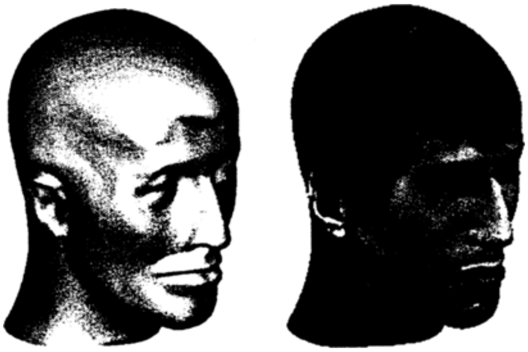


Fig.12. Texture mapping a Level 3 model.

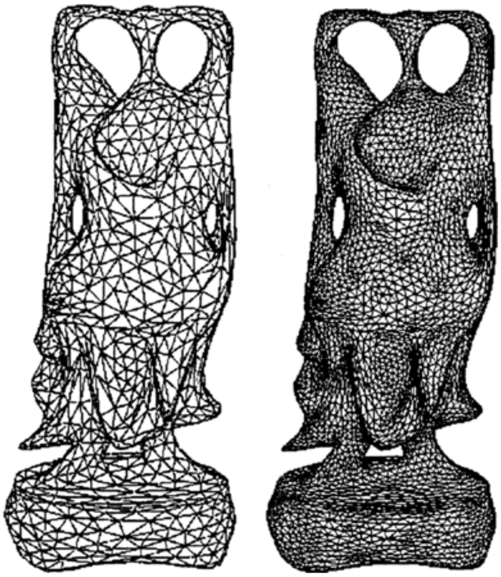


Fig.13. Remeshing the Stanford Happy Buddha model.

Texture mapping technique maps a two-dimensional image onto a geometric object by assigning a texture coordinate to each point on the object surface. Given a parameterization over a base domain, texture coordinate assignment can be efficiently achieved with the barycentric coordinate. See Fig.12 for an illustration.

6 Applications

Two applications, one in computer graphics and the other from CAD industry, are presented here to demonstrate the effectiveness and efficiency of the proposed framework.

In computer graphics, most existing graphics models are in the form of irregular meshes and remeshing is an important operation that transforms an irregular mesh to a semi-regular mesh. Remeshing operation can serve as the common ground for the following techniques: graphics model compression, progressive transmission, multiresolution editing, level-of-detail rendering and others. By transforming irregular meshes into Level 2 representations and from them generating Level 3 representation, remeshing operation can be easily realized in our proposed framework. See Fig.13 for an illustration.

In computer-aided design, it is interested in obtaining analytic forms of the model with a high level structure. In our Level 3 shape representation, given a parameterization π of M^J over M^0, the mesh M^J can be divided into triangular patches corresponding to the faces in M^0. These triangular patches can be further processed into a triangular B-spline surface[1]. See Fig.14 for an illustration.

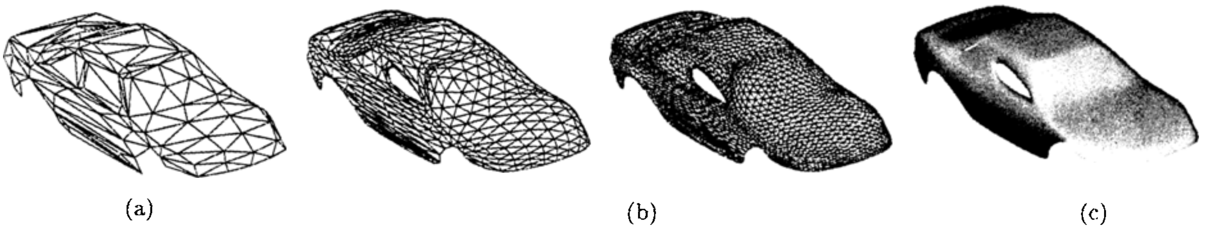


Fig.14. Smooth surface reconstruction. (a) Automatically generated base mesh from a dense range data. (b) Restructure the range data. (c) C^2 triangular B-spline surface.

7 Conclusion

In this paper an efficient framework for multiresolution free form object modeling is proposed with a new hierarchy of shape representations for point sampled geometry. By transforming between the proposed hierarchical shape representations, the proposed framework offers concise integration of various volumetric and parametric surface-based modeling functions for free form object modeling, and thus, simplifies the complete modeling process and reduces the computational cost as required by previous reported algorithms.

Acknowledgements The first author would like to thank Professor Tosiyasu L. Kunni for his comments and encouragements on the first author's dissertation. The original model shown in Fig.3 is courtesy of the *Cyberware* and the original Happy Buddha model in Fig.13 is courtesy of the *Stanford 3D Scanning Repository.*

References

[1] Farin G. Curves and Surfaces for CAGD: A Practical Guide. 5th ed., San Francisco, CA, Morgan Kaufmann, 2002.

[2] Liu Y-J. Complex shape modeling with point sampled geometry [Dissertation]. Hong Kong University of Science and Technology, 2003.

[3] Pasko A, Adzhiev V, Sourin A, Savchenko V. Functional representation in geometric modeling: Concepts, implementation, and application. *Visual Computer*, 1995, 11(8): 429–446.

[4] Amenta N, Bern M, Kamvysselis M. A new Voronoi-based surface reconstruction algorithm. In *Proc. SIGGRAPH'98, ACM SIGGRAPH*, 1998, pp.415–421.

[5] Bardinet E, Cohen L, Ayache N. A parametric deformable model to fit unstructured 3D data. *Computer Vision & Image Understanding*, 1998, 71(1): 39–54.

[6] Carr J, Beatson R, Cherrie J, Mitchell T, Fright W, McCallum B, Evans T. Reconstruction and representation of 3D objects with radial basis functions. In *Proc. SIGGRAPH'01, ACM SIGGRAPH*, 2001, pp.67–76.

[7] Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D, Silva C T. Computing and rendering point set surfaces. *IEEE Trans. Visualization and Computer Graphics*, 2003, 9(1): 3–15.

[8] Osher S, Fedkiw R. Level Set Methods and Dynamic Implicit Surface. New York, Hong Kong: Springer-Verlag, 2003.

[9] Moller T, Haines E. Real-Time Rendering. 2nd ed., Natick, Mass.: AK Peters, 2002.

[10] Hoppe H. Progressive meshes. In *Proc. SIGGRAPH'96, ACM SIGGRAPH*, New Orleans, Louisiana, USA, 1996, pp.99–108.

[11] Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W. Multiresolution analysis of arbitrary meshes. In *Proc. SIGGRAPH'95, ACM SIGGRAPH*, Los Angeles, CA, USA, 1995, pp.173–182.

[12] Foley J, van Dam A, Feiner S, Hughes J. Computer Graphics: Principles and Practice. 2nd ed., Reading, Mass.: Addison-Wesley, 1996.

[13] Pfister H, Zwicker J, van Barr J, Gross M H. Surfels: Surface elements as rendering primitives. In *Proc. SIGGRAPH'00, ACM SIGGRAPH*, New Orleans, Louisiana, USA, 2000, pp.335–342.

[14] Liu Y J, Yuen M M F. Optimized triangle mesh reconstruction from unstructured points. *Visual Copmuter*, 2003, 19(1): 23–37.

[15] Zorin D, Schroder P. Subdivision for modeling and animation. SIGGRAPH Course Notes, 2000.

[16] de Berg M, van Kreveld M, Overmars M, Schwarzkopf O. Computational Geometry: Algorithm and Applications. Springer-Verlag, 1997.

[17] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Surface reconstruction from unorganized points. In *Proc. SIGGRAPH'92, ACM SIGGRAPH*, Chicago, Illinois, USA, 1992, pp.71–78.

[18] Cormen T, Leiserson C, Rivest R. Introduction to Algorithms. 2nd ed., Cambridge, Mass.: MIT Press; Boston: McGraw-Hill, 2001.

[19] Liu Y J, Yuen M M F, Tang K. Manifold-guaranteed out-of-core simplification of large meshes with controlled topological type. *Visual Computer*, 2003, 19(7-8): 565–580.

[20] Velho L, Gomes J, Henrique L. Implicit Object in Computer Graphics. New York, Hong Kong: Springer-Verlag, 2002.

[21] Liu Y J, Yuen M M F, Xiong S. A feature based approach for individualized human head modeling. *Visual Computer*, 2002, 18(5-6): 368–381.

[22] Guskov I, Sweldens W, Schroder P. Multiresolution signal processing for meshes. In *Proc. SIGGRAPH'99, ACM SIGGRAPH*, Los Angeles, CA, USA, 1999, pp.325–334.

Yong-Jin Liu received his B.Eng. degree (1998) in mechano-electronic engineering from Tianjin University and M. Phil. (1999) and Ph.D. degrees (2003) both in mechanical engineering from Hong Kong University of Science and Technology (HKUST). He is currently a research associate in the Department of Industrial Engineering and Engineering Management at HKUST. His research interests include computer graphics, geometric modeling, image processing and computer vision.

Kai Tang received his Ph.D. degree in computer engineering from the University of Michigan, USA, in 1990, M.Sc. degree in information and control engineering in 1986 also from the University of Michigan, and B.Sc. degree in mechanical engineering from Nanjing Institute of Technology in China in 1982. Dr. Tang is currently a faculty member in the Department of Mechanical Engineering at HKUST. Before joining HKUST in 2001, he had worked for more than 13 years in the CAD/CAM and IT industries. His research interests concentrate on designing efficient and practical algorithms for solving real world computational, geometric, and numerical problems.

Matthew Ming-Fai Yuen is a professor at HKUST. He obtained his Ph.D. degree in mechanical engineering from Bristol University, UK, in 1977. He jointed HKUST as one of the founding members of the Department of Mechanical Engineering in 1992. He had served as Associate Dean of Engineering School, founding Director of the CAD/CAM Facility and founding Director of the EPACK Laboratory at HKUST. He is now holding the concurrent positions of professor of mechanical engineering, and Director of the Technology Transfer Center at HKUST. He is also vice president of the HKUST R&D Corporation. He has published more than 150 papers in international journals and conferences. He has won the Best Paper Awards from both the Institution of Mechanical Engineers, UK, and American Society of Mechanical Engineers. He was a Commonwealth Fellow and a United Nation Industrial Development Organization Fellow. He is a fellow of the Institution of Mechanical Engineers.