Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Graphical Models 67 (2005) 17–42

Graphical Models

www.elsevier.com/locate/gmod

An optimization algorithm for free-form surface partitioning based on weighted gaussian image

Kai Tang[a,*], Yong-Jin Liu[b]

[a] *Department of Mechanical Engineering, Hong Kong University of Science and Technology, Hong Kong*

[b] *Department of Industrial Engineering and Engineering Management, Hong Kong University of Science and Technology, Hong Kong*

Received 10 July 2002; received in revised form 4 June 2004; accepted 14 July 2004
Available online 19 August 2004

Abstract

Partitioning free-form surfaces into sub-patches and finding optimal representative normal for each patch to maximize a global objective function is an important two-level operation in diverse industrial applications. In this paper, by solving a maximum hemispherical partitioning problem raised from a weighted Gaussian image, an optimization algorithm is proposed to partition a free-form surface into two sub-patches and simultaneously report the optimal representative normals. By discretizing the free-form surface with W sample points and clustering normals on the surface with m distinct sample normals, the proposed algorithm is designed, in general, with $O(m^2W^2)$ time complexity and $O(W^2)$ space complexity, and in particular, if the surface is convex, in $O(m^2 \log m)$ time complexity. Case studies with four representative examples are presented and a real world application is exploited to demonstrate the effectiveness and usefulness of the proposed algorithm.

Keywords: Free-form surface partitioning; Maximum normal vector projection; Spherical algorithm; Weighted Gaussian image; Visibility map

[*] Corresponding author. Fax: +852 23581543.
E-mail addresses: mektang@ust.hk (K. Tang), liuyj@ust.hk (Y.-J. Liu).

1524-0703/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.gmod.2004.07.001

1. Introduction

Partitioning complex free-form surfaces into simple, "meaningful" sub-patches is a fundamental problem in various disciplines. The justification for "meaningful" patches is obviously application-dependent. One such industrial application is in the context of reverse engineering [28]: after point data captured from the surface of a physical object, the first challenging step is to segment the data into disjoint subsets which are internally smooth. Afterwards each segmented data is fit locally with a simple analytic surface. It is expected to determining an optimal representative normal $\boldsymbol{N}$ for each segmented data: the direction $\boldsymbol{N}$ can offer a good guess (or initialization) to the normal of a projection plane or to the axis of a surface of revolution for subsequently simple analytic surface fitting.

Other industrial applications concern the workpiece setups (or orientations). In these applications such as 3-axis surface machining, coordinate measuring machine (CMM) inspection and laser range scanning in data acquisition, usually two levels of planning for free-form surface partitioning are involved. At the first level, a number of "setups" for the free-form object are determined such that the entire free-form surface of the object is accessible to the machining tool (or probing head in CMM inspection, or laser beam in laser scanning). At the second level, the free-form surface is sampled by points which are either in the form of section lines on the surface (for 3-axis machining) or an ordered array of points (for laser range scanning); in either way the sample points can be considered as the projection of a uniform planar grid system onto the surface along the normal direction of the plane of the projection grid system. In Fig. 1, a schematic example of laser range scanning is shown to illustrate this two-level planning process.

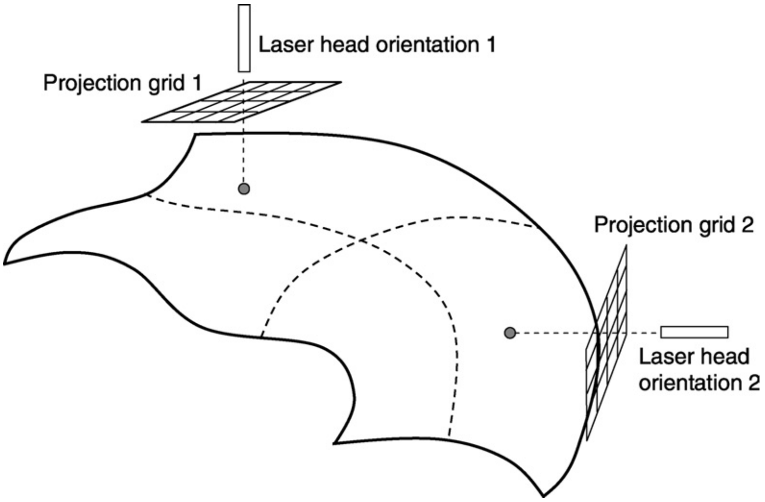

Fig. 1. Multi-orientations in laser range scanning.

Each setup thus is represented by a direction vector $\boldsymbol{N}$ that determines the plane of the projection grid system for that setup (cf. Fig. 1). When determining a setup, one critical criterion is that the normal vectors of the object's surface patch must be as close as possible to the vector $\boldsymbol{N}$. This is best validated by the notion of *cusp height*. As shown in Fig. 2, the extraneous material left on a machined surface between two neighbouring tool passes is called a *cusp* which can be measured by its height. This height is a strict monotone-decreasing function of the angle between the local surface normal $\boldsymbol{n}$ and the tool axis $\boldsymbol{T}$ [4,23], which is $\boldsymbol{N}$ in this context. The same observation can also be made about laser scanning, where experiments show that the accuracy of data acquisition is greatly enhanced if the laser beam is perpendicular to the surface. Therefore, to achieve better quality of the machined surface or higher accuracy of data acquisition, the normal vectors $\boldsymbol{n}$ and $\boldsymbol{N}$ should be as close as possible.

There is however an apparent paradox between the two levels. On one hand, the number of setups should be minimized due to concerns of potential errors incurred during the system calibration, non-production time spent in mounting and dismounting the workpiece, data incorporation, and others. On the other hand, it is obvious that more setups means more partitionings on the free-form surface which in turn leads to better matching pairs of $\boldsymbol{n}$ and $\boldsymbol{N}$ for each constituting sub-region.

Noting that the problem of minimizing the number of setups has been exhaustively investigated [3,8,26] and is known to be NP-hard [7], a more plausible and realistic solution to the two-level free-form surface partitioning problem should be explored with the number K of setups fixed as a priori. To the authors' best knowledge, although it is of significance in many industrial applications, there are no published reports on this two-level K-partitioning problem, even for the case $K = 2$. In this paper we propose a novel algorithmic solution to a 2-partitioning problem with the following contributions:

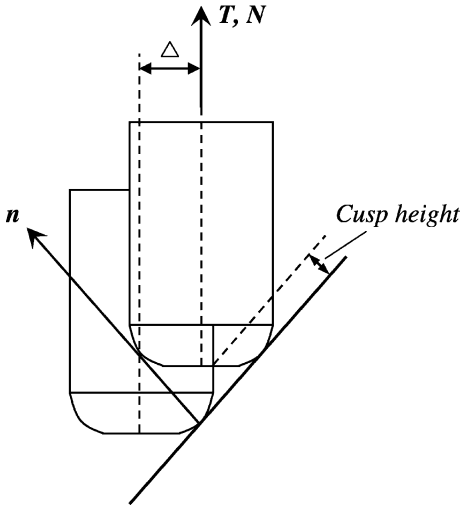

Fig. 2. Cusp height.

1. The proposed algorithm solves a special case $K = 2$ of the general K-partitioning problem which is considered as a very complex geometric problem and up to now no any solution (including the special case $K = 2$) is known to exist.
2. The direct application of the proposed algorithm to real-world scenarios is also exploited. In Section 6 we show that for objects bounded by smooth patches with low curvature variation, by recursively applying the 2-partitioning algorithm, these smooth patches are well recovered one by one. The underlying reason for this good performance is also explained in Section 6.

This paper is organized as follows. The related work is presented in Section 2. In Section 3, a formal mathematical formulation of the 2-partitioning problem solved in this paper is established. In Section 4 our proposed algorithmic solution to the formulated 2-partitioning problem is presented in details and we can show that, by discretizing the free-form surface with W sample points and clustering normals on the surface with m distinct sample normals, the proposed algorithm is designed, in general, with $O(m^2W^2)$ time complexity and $O(W^2)$ space complexity, and in particular, if the surface is convex, in $O(m^2 \log m)$ time complexity. Case studies with four representative examples are presented in Section 5 and one real world application is presented in Section 6, respectively, to demonstrate the effectiveness and usefulness of the proposed algorithm. Finally, our concluding remark is presented in Section 7.

2. Related work

2.1. Free-form surface partitioning

Decomposition of a complex free-form surface into simpler sub-patches is a fundamental problem in various disciplines. In image process, image segmentation based on coherence of brightness, color, texture or motion, is a necessary pre-processing step for perceptual grouping and organization in a higher level computer vision system [24]. In computer graphics polygonal meshes are decomposed into sub-meshes with the applications of metamorphosis, compression, collision detection, parameterization, texture mapping, and mesh simplification; see [13] for an overview and the references therein. In computational geometry, the problem of decomposing a complex polyhedral surface into a smallest number of convex patches is proved to be NP-complete [1] and thus heuristics are inevitable.

For convex surface decomposition, an experimental study for good heuristics is conducted with three classes of heuristics [1]: space partitioning, space sweep, and flooding. In computer graphics applications, heuristics with fuzzy clustering [13], watersheds [18], and flooding [33] have been proposed.

In this paper, we propose an algorithmic solution to a special case of a two-level free-form surface partitioning problem that can find rich applications in computer graphics and computer-aided design. Our proposed solution partitions a free-form surface into two optimal sub-patches by maximizing a global objective function. The proposed solution can also offer a heuristic to the solution of a general two-level (arbitrary)

K-partitioning problem. Our solution uses a concept of weighted Gaussian image, which can be viewed as a discretized version of extended Gaussian image [9].

2.2. Extended Gaussian image

Given a free-form surface $\boldsymbol{F}$ with an orientation $\boldsymbol{N}'$ (i.e., a differentiable field of unit normal vectors), the extended Gaussian image (EGI) associated with $\boldsymbol{F}$ is a unit normal data set $\boldsymbol{N}'(\boldsymbol{x})$ with weights $w(\boldsymbol{x})$ being the inverse of Gaussian curvature $K(\boldsymbol{x})$ for any point $\boldsymbol{x} \in \boldsymbol{F}$ [9]. EGI has many nice properties, e.g., it is insensitive to position of the object and it is determined uniquely by a convex object. Comprehensive study on EGI with various geometric operations can be found in [9,15,32].

Owing to these desired properties, EGI is among the best known methods for 2D and 3D shape representation and analysis. It has been used as a fundamental tool in diverse applications, such as 3D (not necessarily convex) object recognition and reconstruction [30], measure of shape similarity [32], pose determination [12], 3D symmetry detection [25]. In this paper, we propose a variant of EGI and its application in free-form surface partitioning is exploited.

Similar to us, a recent work [22] also uses the concept of Gaussian image in the context of finding an optimal workpiece orientation in 3-axis machining. Their methodology takes into account a tool's shape as well. Nevertheless, the solution given in [22] only applies to a single setup. Moreover, no visibility maps are considered and thus limiting their analysis to a convex part with a hemispherical Gaussian image.

3. Problem formulation

As stated in Section 1, we consider the problem of partitioning the free-form surface into two sub-patches and simultaneously report an optimal representative normal $\boldsymbol{N}$ for each sub-patch $\boldsymbol{F}$, such that the normal vectors on $\boldsymbol{F}$ and $\boldsymbol{N}$ are as close as possible. To study the properties associated with normal vectors, we investigate the feasibility of utilizing the Gaussian image [5]: the Gaussian image of a free-form surface $\boldsymbol{F}$ is some area Ω on a unit sphere $\boldsymbol{S}$ such that the vector from the origin to any point $\boldsymbol{n} \in \Omega$[1] represents a normal vector at a point on $\boldsymbol{F}$ and vice versa. Since more than one point in $\boldsymbol{F}$ may contribute to a point on $\boldsymbol{S}$, in the continuous sense, area-weighted mean normal is preferred. One natural and intuitive area-based weight is the inverse of the absolute value of Gaussian curvature which can be defined as

$$K = \lim_{D(A)\to 0} \frac{A^K}{A}, \tag{1}$$

where K is the Gaussian curvature at a point $\boldsymbol{p}$ on $\boldsymbol{F}$, A is an infinitesimal area around $\boldsymbol{p}, D(A)$ is the diameter of A, and A^K is the area of the Gaussian image of A.

[1] For simplicity of representation, from now on, we indistinguish the terms of "a vector from the origin to a point $\boldsymbol{n}$ on a unit sphere $\boldsymbol{S}$" and "a point $\boldsymbol{n}$ on a unit sphere $\boldsymbol{S}$."

Associating this weight to normals on $\boldsymbol{F}$ (i.e., points on $\boldsymbol{S}$) actually leads to the extended Gaussian image (EGI) [9].

For numerical process with a computer, we discretize both the free-form (continuous) surface and its associated (differentiable) normal field, i.e., both the free-form surface $\boldsymbol{F}$ and the Gaussian image $\Omega \subset \boldsymbol{S}$ are represented by a set of points.

3.1. Point based free-form surface representation and weighted Gaussian image

We study the free form surfaces in a broad context—they may be represented by measured scientific data (e.g., depth maps from binocular stereo), triangle meshes, implicit surfaces and parametric surfaces—and describe them with a universal representation [16]: the free-form surface is represented by a set of points and each point is associated with a quantized normal vector. Generating points on the surface of a shape can be regarded as a sampling process. The density of resulting point cloud can be either uniform [27] (cf. Fig. 10A) or curvature-dependent [11] (cf. Fig. 9B). Normals can be quantized by tessellating the unit sphere using cells with a regular pattern [10]; each cell then has the same shape and area. All the normals falling into a cell will be quantized into a representative normal that is usually centered at that cell. We are now ready to present our weighted Gaussian image that can be viewed as a discretized version of extended Gaussian image [9]:

Definition 1. Given a point sampled free-form surface $\boldsymbol{F} = ((\boldsymbol{p}_1, \boldsymbol{n}'_1), (\boldsymbol{p}_2, \boldsymbol{n}'_2), \ldots, (\boldsymbol{p}_W, \boldsymbol{n}'_W))$, its associated weighted Gaussian image (WGI) is a unit normal data set $\boldsymbol{N} = ((\boldsymbol{n}_1, w_1), (\boldsymbol{n}_2, w_2), \ldots, (\boldsymbol{n}_m, w_m))$ with a weight function w, where $\boldsymbol{n}_i \in \Omega^2$ is a representative normal of a distinct cell $\boldsymbol{c}_i$ on $\boldsymbol{S}$ and $w_i = w(\boldsymbol{n}_i)$ is the number of normals in $\{\boldsymbol{n}'_i | 1 \leqslant i \leqslant W\}$ that falls into $\boldsymbol{c}_i$.

There is a clear link between our defined WGI and the original Horn's EGI [9]; we show this link in Appendix A.

3.2. The Maximum Hemispherical Partitioning problem

To establish a rigorous mathematical formulation of the 2-partitioning problem, the following notions are needed. Let $\boldsymbol{F}$ be a free form surface and $\boldsymbol{n}(\boldsymbol{p})$ the quantized unit normal vector at a point $\boldsymbol{p}$ on $\boldsymbol{F}$. The visibility map of a point $\boldsymbol{p}$ on $\boldsymbol{F}$, denoted as $\boldsymbol{V}(\boldsymbol{p})$, is defined as a connected region on the unit sphere $\boldsymbol{S}$, such that every point $\boldsymbol{n} \in \boldsymbol{V}(\boldsymbol{p})$ represents a direction from which the point $\boldsymbol{p}$ is globally visible, and vice versa;[3] a 2D example is shown in Fig. 3, where $\boldsymbol{S}$ degenerates to a unit circle and $\boldsymbol{V}(\boldsymbol{p})$ an arc. Comprehensive studies on characterizing and computing a visible map $\boldsymbol{V}(\boldsymbol{p})$ for an arbitrary point $\boldsymbol{p}$ on $\boldsymbol{F}$ can be found in [2,6,29]. Among them, the

[2] Without the risk of ambiguity, from now on, the region Ω is represented by the union of cluster cells containing quantized normals on the Gaussian sphere.

[3] In case of machining and CMM inspection, the size of the tool or probing head need also be considered.

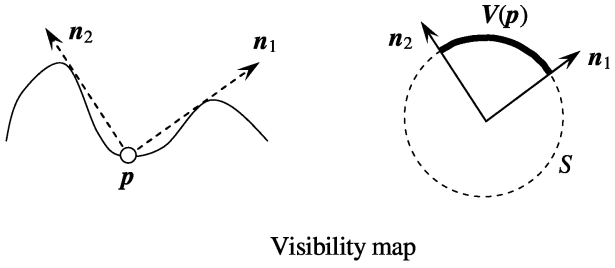

Fig. 3. Visibility map.

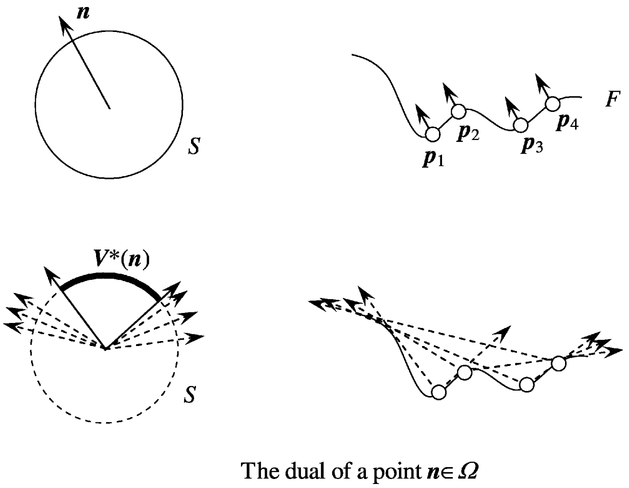

Fig. 4. The dual of a point $\boldsymbol{n} \in \Omega$.

most feasible one, in our case, is to approximate the surface $\boldsymbol{F}$ by a set of structured points and compute $\boldsymbol{V}(\boldsymbol{p})$ using a 2D configuration space (C-space[4] for brevity), cf. the work in [19,20], see also [14]. Note that there may be more than one point on $\boldsymbol{F}$ having the same quantized normal vector $\boldsymbol{n}$. The dual of a point $\boldsymbol{n} \in \Omega$, denoted as $\boldsymbol{V}^*\boldsymbol{n}$, is defined as $\bigcap_{p\in\Omega^{-1}(\boldsymbol{n})} V(\boldsymbol{p})$;[5] see Fig. 4 as an illustration. A general region σ (i.e., it may have holes, or even not connected) in $\boldsymbol{F}$ is said to be *hemispherical* if the intersection $\bigcap_{p\in\sigma} V(\boldsymbol{p})$ is not empty.[6]

By utilizing the WGI, the problem of partitioning the free form surface into two optimal sub-patches can be formulated as partitioning the normals in Ω into two

[4] The underlying idea of C-space is to represent a moving object as a point in an appropriate space (usually specified by the degrees of freedom of the moving object) in which the obstacles are also mapped. Interested readers are referred to [14] for an overview.

[5] The notation $\Omega^{-}(\boldsymbol{n})$ here is not mathematically strict; however without the risk of confusion, we use it to denote the set of all the points on F that have the same quantized normals $\boldsymbol{n}$.

[6] The word *hemispherical* is borrowed here, albeit some abusing. It conforms to the original definition only when all the visibility maps $\boldsymbol{V}(\boldsymbol{p})$ are hemispheres, which corresponds to a convex $\boldsymbol{F}$.

sets. It is important to note that both the constituting regions A_1 and A_2 in a valid partitioning of Ω (i.e., $\Omega = A_1 \cup A_2$ and $A_1 \cap A_2 = \emptyset$) must be hemispherical; otherwise one of $\bigcap_{\boldsymbol{n} \in A_1} V^*(\boldsymbol{n})$ and $\bigcap_{\boldsymbol{n} \in A_2} V^*(\boldsymbol{n})$ would be empty. In other words, A_1 and A_2 must be two mutually complemented hemispheres separated by a great circle. All together, our two-level free-form surface partitioning problem can be formally stated as follows.

3.3. The maximum hemispherical partitioning (MHP) problem

Given m points $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ on the unit sphere $\boldsymbol{S}$, the associated weights $w(\boldsymbol{n}_i)$ and the duals $\boldsymbol{V}^*(\boldsymbol{n}_i)$,$(i = 1, \ldots, m)$, find two mutually complementary hemispheres H and $\overline{H}$ on $\boldsymbol{S}$ and two unit length vectors $N_1 \in \bigcap_{\boldsymbol{n}_i \in H} V^*(\boldsymbol{n}_i)$ and $N_2 \in \bigcap_{\boldsymbol{n}_i \in \overline{H}} V^*(\boldsymbol{n}_i)$ such that the total weighted projection function $E(\boldsymbol{N}_1, \boldsymbol{N}_2) = \sum_{\boldsymbol{n}_i \in H} w(\boldsymbol{n}_i)(\boldsymbol{N}_1 \cdot \boldsymbol{n}_i) + \sum_{\boldsymbol{n}_i \in \overline{H}} w(\boldsymbol{n}_i)$ $(\boldsymbol{N}_2 \cdot \boldsymbol{n}_i)$ is maximized.

Note that in above formulation, a total weighted projection function is defined as a measure of surface partitioning quality. Note also that each sub-patch resulted from partitioning normal vectors on $\boldsymbol{S}$ is guaranteed to be connected only if the surface $\boldsymbol{F}$ is convex. Given a concave $\boldsymbol{F}$, some sub-patch may be disjoint; however, it is not a serious constraint in some industrial applications such as machining and laser scanning; e.g., a finished patch under each machining operation needs not to be connected. In the next section, an algorithmic solution is proposed to solve the formulated MHP problem.

4. The maximum hemispherical partitioning (MHP) algorithm

The development of the proposed algorithm is composed of three major steps. In the first step, efforts are made to discretize the space of mutually complementary hemispheres so that a deterministic characterization of the search space can be achieved. In the second step, efficient algorithm is sought to calculate an optimal vector $\boldsymbol{N}$ for a specific hemispherical partitioning—when all the visibility maps are hemispheres. Finally in step 3, the first two steps are combined together to form an overall algorithm that finds an optimal solution to the MHP problem.

4.1. Discretizing the space of hemispheres

Theoretically, there could be O(2^m) ways to divide a set of m points into two groups. The requirement that such a division must be made in the form of two complementary hemispheres tremendously reduces the search space. On the other hand, the involvement of hemispheres changes the search space from discrete domain to continuous domain. In order to devise a deterministic algorithm, it is imperative to discretize this continuous space. Given a great circle g on S, let $H(g)$ and $\overline{H}(g)$ denote the two mutually complementary hemispheres on S divided by g. The lemma given below serves this need. Assume

(1) the points $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ on a unit sphere S are in general positions, i.e., no any three points lie on a same great circle,
(2) the membership (to which of the two hemispheres) of the points (at most two) on the partitioning great circle is arbitrary.

In practice, our proposed algorithm takes care of the assumption (2) and the algorithm implementation presented in Section 5.1 guarantees the assumption (1).

Lemma 1. *Let g be a great circle on* $\mathbf{S}$ *that divides m points* $\boldsymbol{n}_i$ *into two groups such that, with out loss of generality,* $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_k\} \in H(g)$ *and* $\{\boldsymbol{n}_{k+1}, \boldsymbol{n}_{k+2}, \ldots, \boldsymbol{n}_m\} \in \overline{H}(g)$. *Then there exists another great circle* g' *that passes through some two points* $\boldsymbol{n}_i$ *and* $\boldsymbol{n}_j$ *such that* $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_k\} \in H(g')$ *and* $\{\boldsymbol{n}_{k+1}, \boldsymbol{n}_{k+2}, \ldots, \boldsymbol{n}_m\} \in \overline{H}(g')$.

Proof. Without loss of generality, suppose g is in the $x - y$ plane and $H(g)$ is the upper hemisphere. As illustrated in Fig. 5, g can be rotated about the x-axis until it touches some point $\boldsymbol{n}_i$. We then continue to rotate g, but this time about the vector $\boldsymbol{n}_i$, until it encounters another point $\boldsymbol{n}_j$. The final great circle $g$prime after these two rotations obviously imposes the same partitioning on $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$, i.e., $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_k\}$ in one hemispherical group, and $\{\boldsymbol{n}_{k+1}, \boldsymbol{n}_{k+2}, \ldots, \boldsymbol{n}_m\}$ in another. □

Consider an arbitrary point $\boldsymbol{n}$: up to permutation and rotation, assume it is $\boldsymbol{n}_1$ and situates at $(0,0,1)$. Let $g(\boldsymbol{n}_1, \boldsymbol{n}_j)$ refer to the unique great circle through the two points $\boldsymbol{n}_1$ and $\boldsymbol{n}_j$. Under the orthogonal projection $\boldsymbol{P}_{xy}$ to the plane $z = 0$, that is, $\boldsymbol{P}_{xy}(x, y, z,) = (x, y)$, there is a one-to-one correspondence between the lines through $(0,0)$

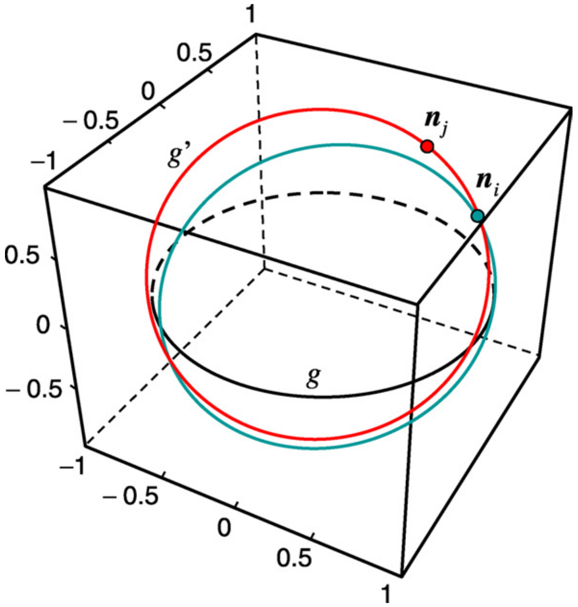

Fig. 5. Proof of Lemma 1.

in the projection plane and the great circles through $(0, 0, 1)$ on $\boldsymbol{S}$. As demonstrated in Fig. 6, it is observed that any two points $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ lie in a same hemisphere of a great circle g if and only if $\boldsymbol{P}_{xy}(\boldsymbol{n}_i)$ and $\boldsymbol{P}_{xy}(\boldsymbol{n}_j)$ lie in a same side (half-space) of the line $\boldsymbol{P}_{xy}(g)$.

Without loss of generality, suppose no point $\boldsymbol{n}_i$ other than $\boldsymbol{n}_1$ lies in the $x - z$ plane. We define in the plane a vector function $\boldsymbol{R}(\boldsymbol{n}_i)$, called the *representative ray* of point $\boldsymbol{n}_i$, as: $\boldsymbol{R}(\boldsymbol{n}_i)$ is just the vector $\boldsymbol{P}_{xy}(\boldsymbol{n}_i)$ if $\boldsymbol{P}_{xy}(\boldsymbol{n}_i)$ has positive y-coordinate; otherwise it is—$\boldsymbol{P}_{xy}(\boldsymbol{n}_i)$. The inverse $\boldsymbol{R}^{-1}()$ is defined as a 3D vector function such that $\boldsymbol{R}^{-1}(\boldsymbol{R}(\boldsymbol{n}_i)) = \boldsymbol{n}_i$. An algorithm is outlined next that finds all the distinct hemispherical partitionings of the points $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ with the restriction that any partitioning great circle g must pass through point $\boldsymbol{n}_1$. Due to Lemma 1, by applying this algorithm to each point in $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$, one readily obtains all the distinct hemispherical partitionings of $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$.

Algorithm ***Hemi_partition*** $(\boldsymbol{n}_1, \{\boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\})$// * Find all the distinct hemispherical partitionings of $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ with the constraint that the partitioning great circles must pass though point $\boldsymbol{n}_1$. *//

```
Step 1. Transform the Cartesian coordinate system so that $\boldsymbol{n}_1$ becomes $(0, 0, 1)$ and
        no any point in $\{\boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ is in the $x - z$ plane;
Step 2. Calculate the $m - 1$ $xy$ projections $xy(\boldsymbol{n}_i), i = 2, 3, \ldots, m$;
Step 3. Constructs rays $\boldsymbol{R}(\boldsymbol{n}_i), i = 2, 3, \ldots, m$;
Step 4. Sort the $m - 1$ rays $\boldsymbol{R}(\boldsymbol{n}_i)$ counter-clockwise into $\{R_2, R_3, \ldots, R_m\}$;
Step 5. $S_1 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-positive$\} \cup \{\boldsymbol{n}_1\}$; $S_2 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-negative$\}$;
Step 6. Output $(S_1, S_2)$;
Step 7. for $i = 2$ to $m$ do begin
        if $\boldsymbol{R}^{-1}(R_i) \in S_1$ then
            Remove $\boldsymbol{R}^{-1}(R_i)$ from $S_1$ and append $\boldsymbol{R}^{-1}(R_i)$ to $S_2$
        else
```

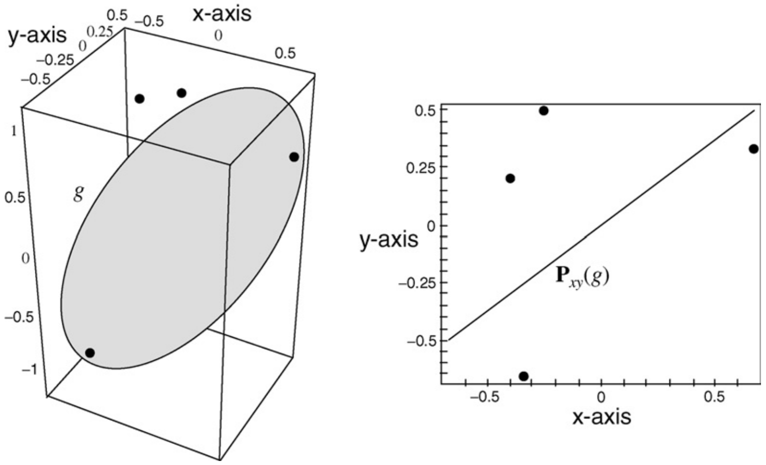

Fig. 6. Inclusion equivalence on the sphere and in the plane $z = 0$.

```
            Remove $\boldsymbol{R}^{-1}(R_i)$ from $S_2$ and append $\boldsymbol{R}^{-1}(R_i)$ to $S_1$
          Output $(S_1, S_2)$;
          end;
Step 8.   $S_1 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-positive$\}$;
          $S_2 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-negative$\} \cup \{\boldsymbol{n}_1\}$;
Step 9.   Repeat Step 6 and Step 7;
  End.
```

Since we assume no any three points lie on a same great circle, the correctness of the above algorithm can be readily proven. We leave its complexity analysis to the last part of this section, when a final optimal algorithm is delineated. A clarification is however needed on the membership of a point $\boldsymbol{p}_i$ that lies on the partitioning great circle. When a great circle g passes through $\boldsymbol{n}_1$ and another point $\boldsymbol{n}_i$, there can be four different partitionings depending on to which side of g we assign the point $\boldsymbol{n}_1$ and $\boldsymbol{n}_i$. Step 5 and Step 8 take care of this assignment for point $\boldsymbol{n}_1$. As for point $\boldsymbol{n}_i$, since it changes its assignment when the ray $\boldsymbol{R}(\boldsymbol{n}_i)$ is processed (Step 7), both sides of g are already accounted for.

4.2. Optimal normal for a hemispherical set

Given a set of spherical points $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_k\}$ that are hemispherical to each other, it is straightforward to identify a unit length vector $\boldsymbol{N}$ that realizes the constrained maximization problem: $\arg\max_N \{\sum_{i=1}^{k} w(\boldsymbol{n}_i)(N \cdot \boldsymbol{n}_i) | N \in \bigcap_{i=1}^{k} V^*(\boldsymbol{n}_i)\}$, where $\boldsymbol{V}^*(\boldsymbol{n}_i)$ is the dual of point $\boldsymbol{n}_i$, defined as the intersection of the visibility maps of all the points in $\boldsymbol{P}$ that contribute to the normal vector $\boldsymbol{n}_i$, i.e., $\cap_{\boldsymbol{p} \in \Omega^{-1}(\boldsymbol{n}_i)} V(p)$. Let $\boldsymbol{n}_w = \frac{\sum_{i=1}^{k} w(\boldsymbol{n}_i)\boldsymbol{n}_i}{\left\|\sum_{i=1}^{k} w(\boldsymbol{n}_i)\boldsymbol{n}_i\right\|}$ and $\Gamma = \bigcap_{i=1}^{k} V^*(\boldsymbol{n}_i)$. Let $d(\boldsymbol{n}, \boldsymbol{n}')$ denote the spherical distance between two points $\boldsymbol{n}$ and $\boldsymbol{n}'$ on S, which is the angle between the two vectors $\boldsymbol{n}$ and $\boldsymbol{n}'$. It can be shown in the Appendix B that if $\boldsymbol{n}_w \in \Gamma$, $\boldsymbol{N}$ must be equal to $\boldsymbol{n}_w$; otherwise, $\boldsymbol{N}$ must be a point $\boldsymbol{n}$ on the boundary $\partial\Gamma$ such that the distance $d(\boldsymbol{n}_w, \boldsymbol{n})$ is shortest among all the boundary points.

The boundary of a dual $\boldsymbol{V}^*(\boldsymbol{n}_i)$ can be nevertheless quite complicated. Recall that m is the number of the distinct points $\boldsymbol{n}_i$ in the weighted Gaussian map and $W = \sum_{i=1}^{m} w(\boldsymbol{n}_i)$ is the total weight. In general, when the surface $\boldsymbol{F}$ is represented by W sample points, the visibility map $\boldsymbol{V}(\boldsymbol{n}_i)$ is a non-convex spherical polygon that can theoretically have up to W arc edges (cf. [19,20]). As a result, $\boldsymbol{V}^*(\boldsymbol{n}_i)$ and hence Γ can be disjointed and non-simple (i.e., possibly with holes), and consequently there is no any other way than a linear traversal of all the edges constituting $\partial\Gamma$ in order to find the shortest distance between it and $\boldsymbol{n}_w$. It is worth noting that, from all practical points of view, $\boldsymbol{V}(\boldsymbol{n}_i)$ is usually approximated by a convex spherical polygon with less than W edges, and, for analysis purpose, this will be the assumption taken in this paper. Since the intersection of two convex spherical polygons with e_1 and e_2 edges is another convex spherical polygon with at most $e_1 + e_2$ edges and can be efficiently constructed in $O(e_1 + e_2)$ time [3,21], Γ has at most W^2 edges and can be obtained in $O(W^2)$ time, given $\boldsymbol{V}(\boldsymbol{n}_i)$. In particular, if all the $\boldsymbol{V}(\boldsymbol{n}_i)$ are hemispheres, which

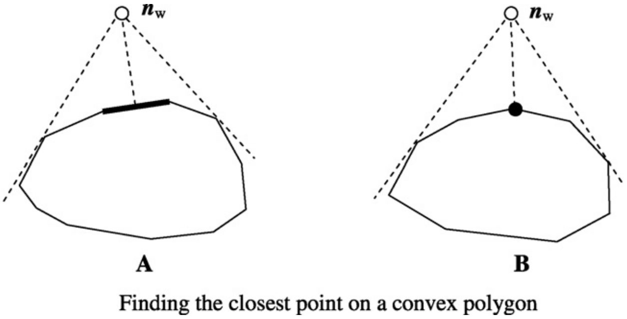

Fig. 7. Finding the closest point on a convex polygon.

corresponds to a convex surface $\boldsymbol{F}$, $\boldsymbol{V}^*(\boldsymbol{n}_i)$ then is in the form of a single great circle and Γ is a convex spherical polygon with at most $k \leqslant m$ edges, independent of the W, which can be computed in $O(k \log k)$ time (cf. [3]).

To see how to efficiently compute $\boldsymbol{N}$, let $< \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n >$ ($\boldsymbol{n} \leqslant W^2$) be the vertices on the boundary $\partial\Gamma$. Each $\boldsymbol{v}_i$ has a pair of spherical coordinates (α_i, β_i). As Γ is convex, both α_i and β_i are ordered. By extending the algorithm of point inclusion test for a planar convex polygon (cf. [3]) to the sphere, one can check in $O(\log n)$ time whether point $\boldsymbol{n}_w$ is inside Γ or not. If not, again, due to the convexity of Γ, one can in $O(\log n)$ time locate the two supporting great circles from $\boldsymbol{n}_w$ to Γ; then, by means of a binary search (noticing that α_i and β_i are ordered), one is able in $O(\log n)$ time to identify either the edge (cf. Fig. 7A) or the vertex (cf. Fig. 7B) that is closest to $\boldsymbol{n}_w$. Let us use ***Optimal_Normal***($\boldsymbol{n}$, $< \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n >$) to represent this just prescribed $O(\log n)$ procedure that either reports $\boldsymbol{n}_w$ if it is inside Γ or returns a point on $\partial\Gamma$ that is closest to $\boldsymbol{n}_w$. We summarize the above analysis into a lemma.

Lemma 2. *Given k hemispherical points* $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_k\}$, *their weights* $w(\boldsymbol{n}_i)$ *and their duals* $\boldsymbol{V}^*(\boldsymbol{n}_i)$, *the unit length vector* $\boldsymbol{N}$ *that is the solution to* $\arg\max_{\boldsymbol{N}}\{\sum_{i=1}^{k} w(\boldsymbol{n}_i)(\boldsymbol{N} \cdot \boldsymbol{n}_i) | \boldsymbol{N} \in \bigcap_{i=1}^{k} V^*(\boldsymbol{n}_i)\}$ *can be found in* $O(W^2)$ *time. In particular, if the surface* $\boldsymbol{P}$ *is convex, such an* $\boldsymbol{N}$ *can be found in* $(k \log k)$ *time, independent of W.*

4.3. The constrained MHP algorithm

In the spirit of both algorithm ***Hemi_partition*** and procedure ***Optimal_Normal***, we are now ready to present the final algorithm that solves the Maximum Hemispherical Partitioning problem. The algorithm given next solves the constrained MHP problem; that is, the partitioning great circle is required to pass through a point, $\boldsymbol{p}_1$. By applying this algorithm m times each for one of the m points $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$, the MHP problem is readily solved. Note that the algorithm ***Hemi_partition*** takes time $O(m \log m)$ to report $2m$ distinct constrained partitionings and, considering the convex case, for each constrained partitioning Lemma 2 tells that a pair of optimal normals can be found in $O(m \log m)$ time; thus totally $O(m^2 \log m)$) time complexity is needed for the constrained MHP problem. The algorithm ***MHP_Point*** proposed below efficiently incorporates the algorithm ***Hemi_partition*** and the procedure ***Optimal_Normal*** such that the time complexity is improved to $O(m \log m)$.

Algorithm ***MHP_Point*** $(\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m)$

//* Find two unit length vectors $\boldsymbol{N}_1$ and $\boldsymbol{N}_2$ that are the solution to the Maximum Hemispherical Partitioning problem of $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$, with the constraint that the corresponding partitioning great circle g pass through point $\boldsymbol{n}_1$. *//

```
Step 1.      Transform the Cartesian coordinate system so that $\boldsymbol{n}_1$ becomes $(0,0,1)$
             and no any of $\{\boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ is in the $x - z$ plane;
Step 2.      Calculate the $m - 1$ projection points $xy(\boldsymbol{n}_i)$, $i = 2, 3, \ldots, m$;
Step 3.      Construct rays $\boldsymbol{R}(\boldsymbol{n}_i), i = 2, 3, \ldots, m$;
Step 4.      Sort the $m - 1$ rays $\boldsymbol{R}(\boldsymbol{n}_i)$ counter-clockwise into $\{R_2, R_3, \ldots, R_m\}$;
Step 5.      $S_1 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-positive$\} \cup \{\boldsymbol{n}_1\}$;
             $S_2 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-negative$\}$;
Step 6.      $\boldsymbol{n}_{w1} \leftarrow \sum_{\boldsymbol{n}_i \in S_1} w(\boldsymbol{n}_i)\boldsymbol{n}_i$;
             $\boldsymbol{n}_{w2} \leftarrow \sum_{\boldsymbol{v}_i \in S_2} w(\boldsymbol{n}_i)\boldsymbol{n}_i$;
Step 7.      $V_1 \leftarrow$ vertices of $\bigcap_{\boldsymbol{n}_i \in S_1} V^*(\boldsymbol{n}_i)$;
             $V_2 \leftarrow$ vertices of $\bigcap_{\boldsymbol{n}_i \in S_2} V^*(\boldsymbol{n}_i)$;
Step 8.      $\boldsymbol{N}_1 \leftarrow \textbf{\textit{Optimal\_Normal}}\left(\frac{\boldsymbol{n}_{w1}}{\|\boldsymbol{n}_{w1}\|}, V_1\right)$;
             $\boldsymbol{N}_2 \leftarrow \textbf{\textit{Optimal\_Normal}}\left(\frac{\boldsymbol{n}_{w2}}{\|\boldsymbol{n}_{w2}\|}, V_2\right)$;
Step 9.      $E = \boldsymbol{N}_1 \cdot \boldsymbol{n}_{w1} + \boldsymbol{N}_2 \cdot \boldsymbol{n}_{w2}$;
Step 10.     for $i = 2$ to $m$ do begin
             if $\boldsymbol{R}^{-1}(R_i) \in S_1$ then
Step 10.1.   Update $V_1$ by removing $\boldsymbol{V}^*(R^{-1}(R_i))$ from $\bigcap_{\boldsymbol{n}_i \in S_1} V^*(\boldsymbol{n}_i)$;
Step 10.2.   Update $V_2$ by inserting $\boldsymbol{V}^*(R^{-1}(R_i))$into $\bigcap_{\boldsymbol{n}_i \in S_2} V^*(\boldsymbol{n}_i)$;
Step 10.3.   Remove $\boldsymbol{R}^{-1}(R_i)$ from $S_1$ and append $\boldsymbol{R}^{-1}(R_i)$ to $S_2$
Step 10.4.   $\boldsymbol{n}_{w1} \leftarrow \boldsymbol{n}_{w1} - w(\boldsymbol{R}^{-1}(R_i))\boldsymbol{R}^{-1}(R_i)$;
             $\boldsymbol{n}_{w2} \leftarrow \boldsymbol{n}_{w2} + w(\boldsymbol{R}^{-1}(R_i))R^{-1}(R_i)$;
             else
Step 10.5.   Update $V_1$ by inserting $\boldsymbol{V}^*(R^{-1}(R_i))$ into $\bigcap_{\boldsymbol{n}_i \in S_1} V^*(\boldsymbol{n}_i)$;
Step 10.6.   Update $V_2$ by removing $\boldsymbol{V}^*(R^{-1}(R_i))$ from $\bigcap_{\boldsymbol{n}_i \in S_2} V^*(\boldsymbol{n}_i)$;
Step 10.7.   Remove $\boldsymbol{R}^{-1}(R_i)$ from $S_2$ and append $\boldsymbol{R}^{-1}(R_i)$ to $S_1$;
Step 10.8.   $\boldsymbol{n}_{w1} \leftarrow \boldsymbol{n}_{w1} + w(\boldsymbol{R}^{-1}(R_i))\boldsymbol{R}^{-1}(R_i)$;
             $\boldsymbol{n}_{w2} \leftarrow \boldsymbol{n}_{w2} - w(\boldsymbol{R}^{-1}(R_i))\boldsymbol{R}^{-1}(R_i)$;
Step 10.9.   $\boldsymbol{N}'_1 \leftarrow \textbf{\textit{Optimal\_Normal}}\left(\frac{\boldsymbol{n}_{w1}}{\|\boldsymbol{n}_{w1}\|}, V_1\right)$;
             $\boldsymbol{N}'_2 \leftarrow \textbf{\textit{Optimal\_Normal}}(\frac{\boldsymbol{n}_{w2}}{\|\boldsymbol{n}_{w2}\|}, V_2)$;
             $E' = \boldsymbol{N}'_1 \cdot \boldsymbol{n}_{w1} + \boldsymbol{N}'_2 \cdot \boldsymbol{n}_{w2}$;
Step 10.10.  if $E < E'$ then
             $\boldsymbol{N}_1 \leftarrow \boldsymbol{N}'_1$;
             $\boldsymbol{N}_2 \leftarrow \boldsymbol{N}'_2$;
             $E \leftarrow E'$
             end;
```

```
    Step 11.  $\{E_1, \boldsymbol{N}_{11}, \boldsymbol{N}_{12}, S_{11}, S_{12}\} \leftarrow \{E, \boldsymbol{N}_1, \boldsymbol{N}_2, S_1, S_2\}$;
    Step 12.  $S_1 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-positive$\}$;
              $S_2 \leftarrow \{\boldsymbol{n}_i | \boldsymbol{n}_i$ is $y$-negative$\} \cup \{\boldsymbol{n}_1\}$;
    Step 13.  Repeat Step 6 through Step 10;
    Step 14.  if $E_1 > E$ then
                 Return $\{\boldsymbol{N}_{11}, \boldsymbol{N}_{12}, S_{11}, S_{12}\}$;
              else
                 Return $\{\boldsymbol{N}_1, \boldsymbol{N}_2, S_1, S_2\}$;
End.
```

4.4. Complexity analysis on algorithm MHP_Point

Step 1 takes linear O(m) time, so do Steps 2 and 3.The sorting at Step 4 takes O($m\log m$) time. At Step 5, an initial hemispherical partitioning is built with the partitioning great circle in the $x-z$ plane and point $\boldsymbol{p}_1$ included in the hemisphere of $y \geqslant 0$, taking O(m) time. The weighted normal vectors of these initial two partitionings are then calculated at Step 6, in O(m) time.

The calculation of the two duals $\bigcap_{\boldsymbol{n}_i \in S_1} V^*(\boldsymbol{n}_i)$ and $\bigcap_{\boldsymbol{n}_i \in S_2} V^*(\boldsymbol{n}_i)$ at Step 7 depends heavily on the underlying representation of the visibility map $\boldsymbol{V}(\boldsymbol{n}_i)$. Assuming $\boldsymbol{V}(\boldsymbol{n}_i)$ is convex, according to Lemma 2, in the worst case Step 7 requires O(W^2) time; in the best scenario of surface $\boldsymbol{F}$ being convex, Step 7 takes only O($m\log m$) time.

Step 8 calculates the two corresponding optimal normals $\boldsymbol{N}_1$ and $\boldsymbol{N}_2$ for the convex spherical polygons V_1 and V_2, respectively, using O($\log n$) time, where ($n = \mathbf{Max}\{\|V_1\|, \|V_2\|\}$), with $\|V_1\|$ (or $\|V_2\|$) standing for the number of boundary edges on V_1 (or $\|V_2\|$). Note that $n \leqslant W^2$ and thus Step 8 takes O($\log W$) time. This bound improves to O($m\log m$) if the surface $\boldsymbol{F}$ is convex.

The bulk of actions and time are spent at the for-loop at Step 10. As already discussed earlier, when the representative ray of a point n_i is processed, the membership of $\boldsymbol{n}_i$ in S_1 and S_2 is switched; this is done at Steps 10.3 and 10.7. At the same time, this switch of membership should affect the two intersections V_1 and V_2 of the corresponding visibility maps (Steps 10.1 and 10.2 and Steps 10.5 and 10.6), as well as the two weighted normal vectors $\boldsymbol{n}_{w1}$ and $\boldsymbol{n}_{w2}$ (Steps 10.4 and 10.8). The updating of V_1 and V_2 in general involves deleting (Steps 10.1 and 10.5) or adding (Steps 10.2 and 10.6) some arc edges to a convex spherical polygon, which is a linear time process, taking O(W^2) time in the worst case. Again, if the surface $\boldsymbol{F}$ is convex or even though $\boldsymbol{F}$ is concave but we let all the visibility maps $V(\boldsymbol{n}_i)$ be hemispheres as an approximation, then the computation would be greatly simplified. In this case, removing an edge from V_1 and V_2 (Steps 10.1 and 10.5) obviously takes constant time only (cf. Fig. 8A). The insertion operation (Steps 10.2 and 10.6) should require only O($\log m$) time: this is because one can in O($\log m$) time, again due to the convexity of V_1 (or V_2), identify the two vertices $\boldsymbol{v}$ and $\boldsymbol{v}'$ in V_1(or V_2) that support the inserting great circle g, e.g., g_1 and g_2 shown in Fig. 8B. By means of binary search on the two sub-chains between $\boldsymbol{v}$ and $\boldsymbol{v}'$, the two intersection points $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ can be located in O($\log m$) time. The last two actions in the loop

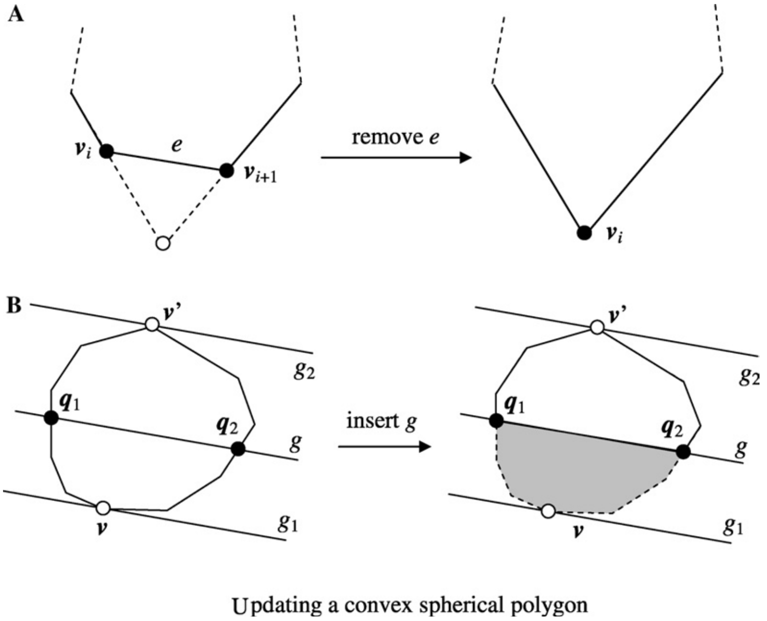

Fig. 8. Updating a convex spherical polygon.

are Steps 10.9 and 10.10 that update the best solution with the current one if it achieves a higher total weighted projection value; their time complexity is exactly the same as that of Step 8.

Finally, owing to the fact that the point $\boldsymbol{n}_1$ can be taken belonging to either the hemisphere $y \geqslant 0$ or $y \leqslant 0$, the entire sequence from Steps 6 to 10 is repeated one more time, accounting for the situation of $\boldsymbol{n}_1$ belonging to the hemisphere $y \leqslant 0$. The two results are then compared and the better one is returned (Step 14). All together, the algorithm ***MHP_Point*** takes $O(mW^2)$ time in the most general case and $O(m\log m)$ in particular if the surface $\boldsymbol{F}$ is convex. By applying ***MHP_Point*** m times, each time with a different constraint point $\boldsymbol{n}_i$, the MHP problem is readily solved. The theorem given below concludes the discussion.

Theorem 1. *The Maximum Hemispherical Partitioning problem for a set of m points* $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ *discretized from a free-form surface* $\boldsymbol{F}$ *can be solved by a simple algorithm requiring* $O(m^2W^2)$ *time and* $O(W^2)$ *space. In particular, if the surface* $\boldsymbol{F}$ *is convex, then the time bound of the algorithm improves to* $O(m^2\log m)$.

5. Algorithm implementation and case studies

The proposed algorithm has been fully implemented on the Visual C++ platform. Three cases with four examples are presented to demonstrate the effectiveness of the proposed algorithm. In the first two examples, exact analytic forms of the free-form surfaces are known. To apply the algorithm, these surfaces are discretized into meshes, i.e., sets of points with structures. In the last two examples, no analytic form of the surfaces is known in advance; only their mesh discretization is known. In these two examples, the mesh structure is used to estimate normal vector at each mesh point. Continued with the third case, in the next section, we present an application to rebuild the analytic form of a free-form surface from a given rough mesh with the aid of our proposed algorithm.

5.1. Some implementation issues

When applying the MHP algorithm to a free-form surface, the first important task is to discretize the surface and obtain an appropriate representative set $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ of normal vector samples on the Gaussian image of the surface. Other than tessellating the unit sphere with regular patterned cells and finding a representative sample for each cell, the *K*-means method [10] is adopted in our current algorithm implementation for finding m such samples.

Starting with a given number m of the clusters and an initial mean for each cluster (the sole normal vector in the cluster), the *K*-means algorithm works in two steps:

(1) assigns the normal vector of each sample point on the discretized surface to a cluster, where the distance between the normal vector and the cluster mean is the smallest among all the clusters, and
(2) updates cluster mean values by merging the new data until the cluster means become stable.

The output from the *K*-means algorithm are the m cluster means $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_m\}$ and their weights $\{w(\boldsymbol{n}_1), w(\boldsymbol{n}_2), \ldots, w(\boldsymbol{n}_m)\}$, where $w(\boldsymbol{n}_i)$ is the number of normal vectors falling into the cluster with mean $\boldsymbol{n}_i (i = 1, 2, \ldots, m)$.

In the proposed MHP algorithm, we also need to guarantee the assumption that no any three points lie on a same great circle. In the algorithmic representation, this assumption means that the $m - 1$ rays $\boldsymbol{R}(\boldsymbol{n}_i)$ can be sorted counter-clockwise with no overlapped keys and the inverse map $\boldsymbol{R}^{-1}()$ is one-to-one. When any two rays $\boldsymbol{R}(\boldsymbol{n}_i)$ have the same sort keys, degenerate cases appear. To handle all degeneracies, we can perturb the given $\boldsymbol{R}(\boldsymbol{n}_i)$ slightly, which amounts to changing the coordinates in $\boldsymbol{R}(\boldsymbol{n}_i)$. It is important to note that the perturbation must be *symbolical* such that it does not change the nondegenerate positions in the set while all degeneracies disappear. For full details of choosing the form of symbolic perturbation, the reader is referred to [31]. In our implementation, we find the following simple strategy work well on all examples presented in this paper. We use an array to store all rays $\boldsymbol{R}(\boldsymbol{n}_i)$. Whenever two rays have the same sort keys, we pretend that the ray with the smaller index is smaller; then doubly linked list is used to maintain the one-to-one correspondence between the map $\boldsymbol{R}()$ and the inverse map $\boldsymbol{R}^{-1}()$.

5.2. Case 1: a closed, convex B-spline surface partitioning

In this case, the free-form object (cf. Fig. 9A) is known as a closed C^2 B-spline surface. Its discretization is illustrated in Fig. 9B. Since the surface is closed, its weighted Gaussian map occupies the entire unit sphere, as shown in Fig. 9C with pseudo-color mapping. Also shown in Fig. 9, $\boldsymbol{N}_1$ and $\boldsymbol{N}_2$ are the solution to the MHP problem, i.e., the normal pair maximizes $E(\boldsymbol{N}_1, \boldsymbol{N}_2) = \sum_{\boldsymbol{n}_i \in H} w(\boldsymbol{n}_i)(\boldsymbol{N}_1 \cdot \boldsymbol{n}_i) + \sum_{\boldsymbol{n}_i \in \overline{H}} w(\boldsymbol{n}_i)(\boldsymbol{N}_2 \cdot \boldsymbol{n}_i)$. Since the surface is also convex, the maximum hemispherical partitioning decomposes the surface into two connected segments. Fig. 9D shows this partitioning (with two different material properties) on the surface induced by the two optimal vectors $\boldsymbol{N}_1$ and $\boldsymbol{N}_2$.

5.3. Case 2: concave free-form surface patch partitioning

In this case, two examples with different surface representations are presented. The surface shown in Fig. 10A is a bicubic B-spline surface and the surface shown in Fig. 11 is a piecewise linear surface recovered from the dense range data of a real-world object. Both surfaces are visible to the z-axis, i.e., whose Gaussian map lies within the upper hemisphere. The optimal normal pair $\{\boldsymbol{N}_1, \boldsymbol{N}_2\}$ found by applying the proposed algorithm is shown in the Figures, together with the partitioning of the surface induced by the two vectors $\boldsymbol{N}_1$ and $\boldsymbol{N}_2$. An interesting observation is drawn from Fig. 11: the right door part of the car body is completely separated from the whole object. This observation will be studied in depth in Section 6.

5.4. Case 3: a complex, genus-1, closed free-form surface partitioning

In this case, the free-form sculpture object shown in Fig. 12A is of genus-1 with a complex geometric shape. It is conceivable that any two vectors $\boldsymbol{N}_1$ and $\boldsymbol{N}_2$ are not sufficient to cover the entire surface. To utilize the proposed algorithm, we first partition it into two symmetric parts through a user-friendly interface. The algorithm is then applied to each part to find totally two pairs of optimal normals $\{\boldsymbol{N}_1, \boldsymbol{N}_2\}$ with which the whole genus-1 object is partitioned into four sub-patches. It is observed that in Fig. 12B the sub-patch corresponding to the normal $\boldsymbol{N}_2$ is disjointed.

6. A real world application

A simple real world application is presented to demonstrate the usefulness of the proposed algorithm. Recall that the model shown in Fig. 11 is a rough mesh model recovered from the dense range data of a real-world object. In engineering design, we are more interested in obtaining analytic forms of the model with a high level structure. We have presented an interesting observation in Section 5.3 that by applying the MHP algorithm, the right door part of the car body is completely separated from the whole object. If we keep partitioning the surface by recursively performing the MHP algorithm, more and more planar regions or smooth regions with low curvature variation are recovered. See Fig. 13 for an illustration.

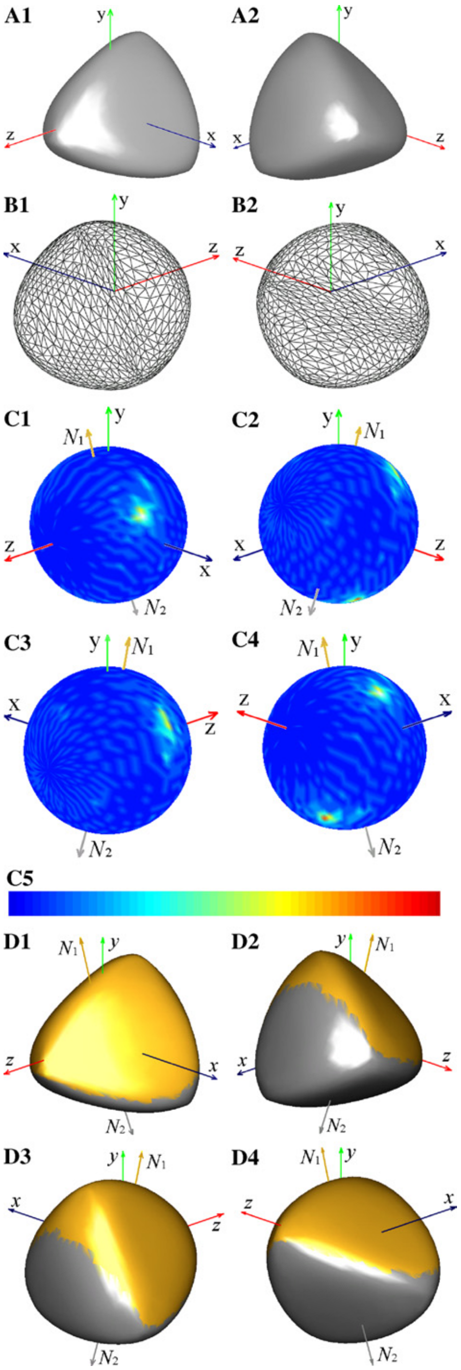

Fig. 9. A closed, convex C^2 B-spline surface partitioning.

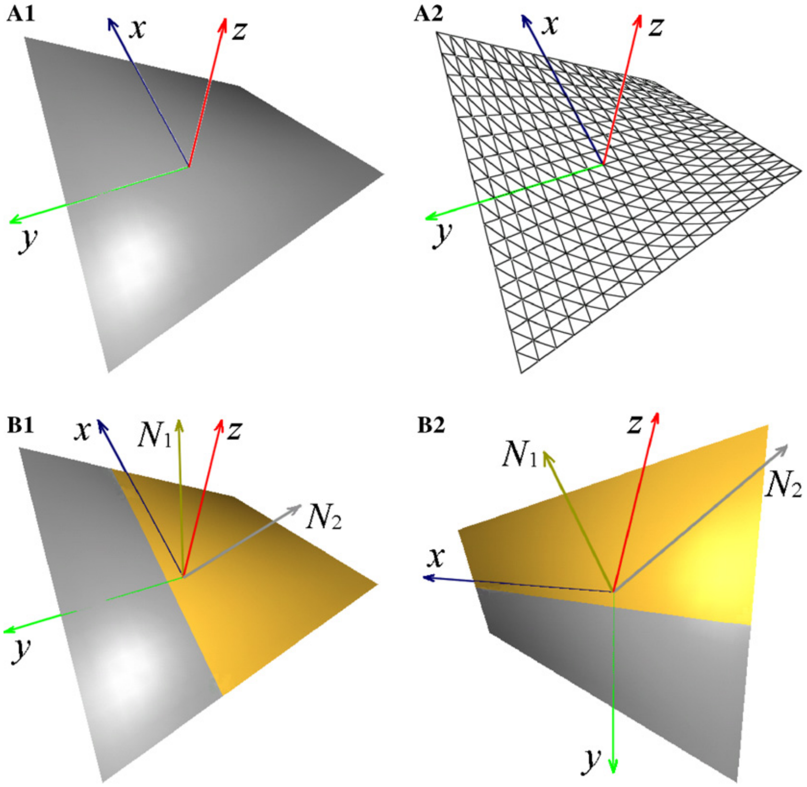

Fig. 10. A concave bicubic B-spline surface patch partitioning.

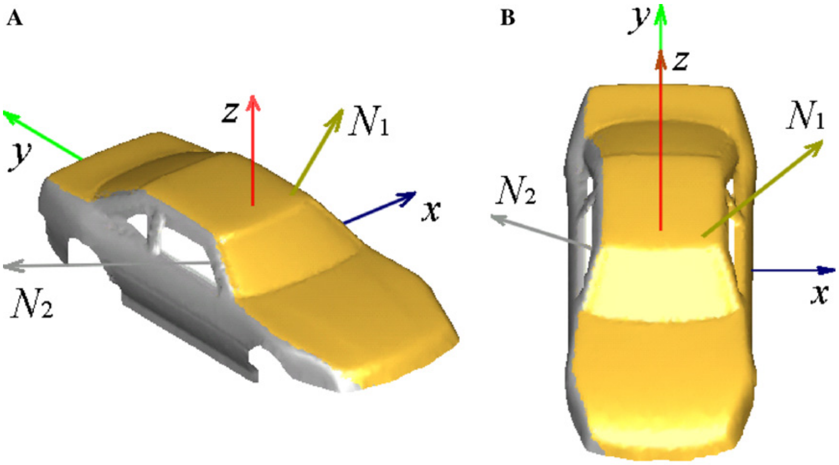

Fig. 11. A real car body object example.

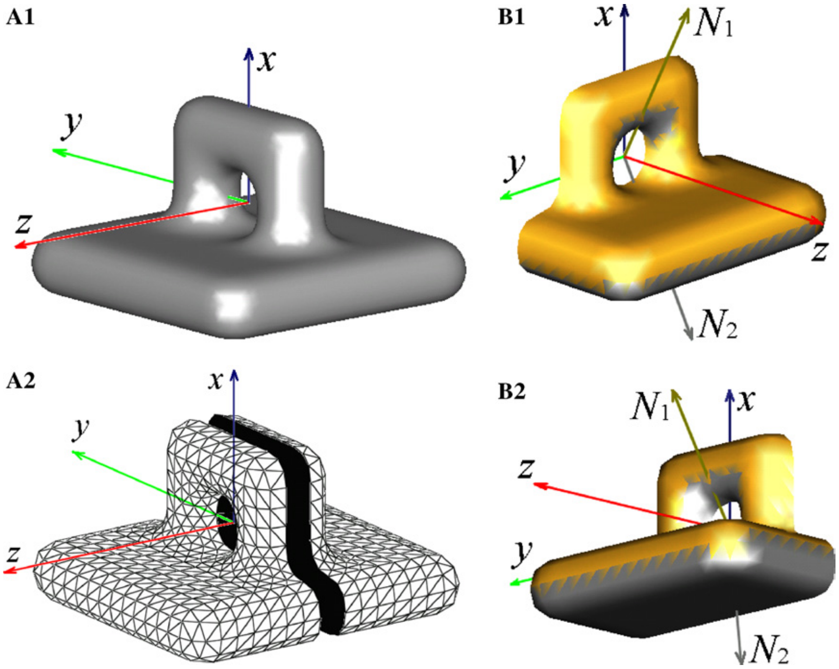

Fig. 12. A complex, genus-1, closed free-form surface partitioning.

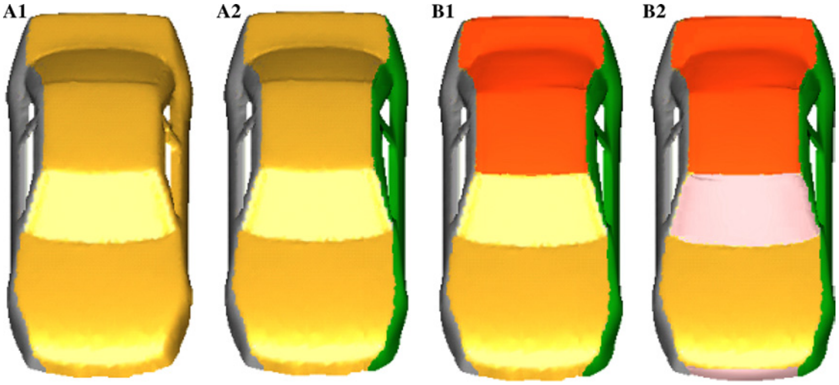

Fig. 13. Free form surface partitioning by recursively applying the proposed MHP algorithm to the golden-shaded regions; this example follows the one in Fig. 11.

This phenomenon is not an accident. Actually if the object is bounded by smooth regions with low curvature variation, the proposed MHP algorithm is most likely to recover them one by one. It can be well interpreted since that the normals on these smooth regions will be clustered into narrow regions at different locations on the Guassian sphere, e.g., normals on a planar region and normals on a portion of a cylinder (or a cone) will be mapped into a single point and a circular arc on the Guassian sphere, respectively.

Given this well-conducted segmentation, we can perform a region-based mesh reduction operation to the model. In [18] a similar application has been presented. Note that traditional mesh simplication algorithms work on the whole model and prioritize the mesh elements, i.e., vertices, edges and faces, with respect to their immediate neighbors.

In our current implementation of region-based mesh reduction, an edge collapse operator is recursively performed. First, we put all edges into a prior queue key on an edge cost based on the potential error metric proposed in [17]. To count the region factor, those edges across two regions (i.e., any edge with its two vertices having two different colors as shown in Fig. 13) are strongly penalized. We then iteratively extract the edge with vertices (v_i, v_j) from the top of the queue, perform collapse $(v_i, v_j) \rightarrow v'$ and locally update the collapse information for all the edges involving v_i and v_j. The iteration process is terminated when the top edge in the queue has a cost larger than a prespecified tolerance that should be determined by the application at hand. The resulting simplified mesh of the model shown in Fig. 13 is illustrated in Fig. 14A.

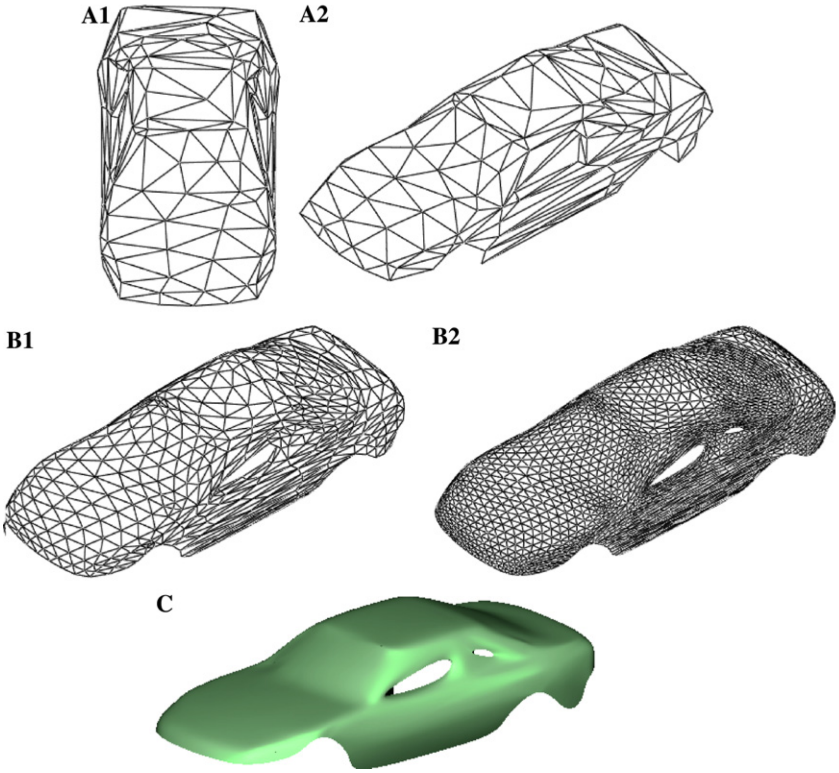

Fig. 14. Patch-based mesh re-structure and high order model generation.

To fit a higher order model to the original rough mesh that is only C^0-continuous, the simplified mesh can serve as a base parametric domain. With this base domain, we re-structure the original polygonal mesh by iteratively performing a displaced loop subdivision scheme [16]. This re-structuring process is illustrated in Fig. 14B. Finally, as a direct result from the loop scheme, we can achieve a C^2 quartic triangular B-spline surface model as a generalization of the original rough mesh data; the result is shown in Fig. 14C.

7. Conclusion and future research

In this paper, we propose to use a weighted Gaussian image to characterize an important two-level geometric problem, i.e., how to partition a free-form surface into a given number K of sub-patches and find an optimal representative normal vector for each patch towards maximizing a global objective function. By augmenting the Gaussian image with weights and incorporating visibility maps for global interference avoidance, we present a simple and practical algorithmic solution that solves the case $K = 2$ of the outlined optimization problem. Case studies are presented to demonstrate the effectiveness of the proposed algorithm and a direct application using the proposed algorithm is also presented.

There are several potential extensions for future research. The first and an obvious task is to find a deterministic optimal algorithmic solution to the general case of a fixed arbitrary K; it is then interesting to compare the upper-bound of the difference between the heuristic approach by recursively applying $K = 2$ solution and the (unknown) optimum.

The other interesting directions are motivated by the observations drawn in the case study section. One possibility is to utilize the proposed algorithm in the applications of pattern recognition and feature based sculpture object modeling, inspired by the observations drawn from Figs. 11 and 13. The other possibility is to include the *connectivity* constraints on the surface partitioning induced by the normal vectors. As revealed by the example in Fig. 12, the sub-region associated with the vector $\boldsymbol{N}_2$ (in either half) consists of two disjoint areas. While this disconnectivity may not be of much concern to certain applications (such as finishing surface machining and laser range scanning), it is worthy to see how the proposed MHP algorithm could be modified to accommodate this connectivity requirement.

Appendix A

The original Horn's EGI [9] weights the normal of a point $\boldsymbol{p}$ on the free form surface $\boldsymbol{F}$ with the inverse of Gaussian curvature at p. The Gaussian curvature can be defined as

$$K = \lim_{\delta A \to 0} \frac{\delta A^K}{\delta A} = \frac{\mathrm{d}A^K}{\mathrm{d}A},$$

where K is the Gaussian curvature at point $\boldsymbol{p}$, δA is a small area around $\boldsymbol{p}$ on $\boldsymbol{F}$ and δA^K is the area of the Gaussian image of δA. Now consider the integral of $1/K$ over a small patch A^K on the Gaussian sphere

$$\int\int_{A^K} \frac{1}{K} \mathrm{d}A^K = \int\int_{A} \mathrm{d}A = A,$$

where A is the area of the corresponding patch on $\boldsymbol{F}$. Since we discretize/quantize both free form surface $\boldsymbol{F}$ and the Gaussian sphere, we can assume that the quantized normal $\boldsymbol{n}$ over a small area A (tends to be infinitesimal) centered at $\boldsymbol{p}$ on $\boldsymbol{F}$ is a constant. Thus

$$\int\int_{A^K} \frac{\boldsymbol{n}}{K} \mathrm{d}A^K = A\boldsymbol{n}.$$

In our defined WGI, the normal $\boldsymbol{n}$ is weighted with the number of normals that fall into the cell containing the representative normal $\boldsymbol{n}$. Consider the case of uniform sampling the free form surface $\boldsymbol{F}$. Denote the sampling density as ρ. Then the number m of points to be sampled over a region with area A is

$$m = \frac{A}{\rho}.$$

Then we have

$$\int\int_{A^K} \frac{\boldsymbol{n}}{K} \mathrm{d}A^K = A\boldsymbol{n} = m\rho\boldsymbol{n}.$$

Due to uniform sampling, ρ is a constant and thus we can always set the weight $w = m\rho$. That completes the establishment of link between the original Horn's EGI and our defined WGI.

Appendix B

Given a set of spherical points $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_k\}$ and a non-constant objective function $J(\boldsymbol{N}) = \sum_{i=1}^{k} w(\boldsymbol{n}_i)\boldsymbol{N} \cdot \boldsymbol{n}_i : S \rightarrow R$ defined over the domain S of a unit sphere, the following are true:

(1) there are only two relative extrema of J over S;
(2) these two relative extrema are a global minimum and a global maximum of J, respectively, with respect to the entire domain S;
(3) given a closed subset $V = \bigcap_{i=1}^{k} V^*(\mathbf{n}_i) \subset S$ not containing the point $\boldsymbol{q}$ at which the global maximum of J over S is reached, the global maximum of J restricted to V is reached at the point $\boldsymbol{n}$ on the boundary ∂V of V, where the geodesic distance from the $\boldsymbol{q}$ to ∂V is shortest.

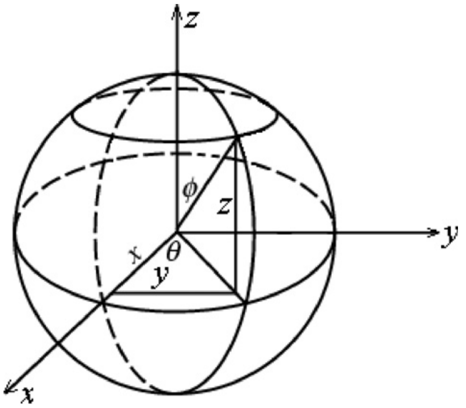

Fig. 15. A curvilinear coordinate system.

Proof.

(1) A curvilinear coordinate system (cf. Fig. 15) is used to describe the position $\boldsymbol{N}(\theta, \phi)$ on a unit sphere, where θ is the azimuthal angle in the xy plane from the x-axis with $0 \leqslant \theta < 2\pi$, and ϕ is the polar angle from the z-axis with $0 \leqslant \phi < \pi$. Then the objective function is

$$J(\theta, \phi) = \sum_{i=1}^{k} w(\boldsymbol{n}_i)(\sin\phi\cos\theta \cdot \boldsymbol{n}_{ix} + \sin\phi\sin\theta \cdot \boldsymbol{n}_{iy} + \cos\phi \cdot \boldsymbol{n}_{iz}).$$

The relative extrema of J can be achieved from the Euler's equation

$$\begin{cases} \frac{\partial J}{\partial \theta} = 0 \Rightarrow \sum\limits_{i=1}^{k} w(\boldsymbol{n}_i)(-\sin\phi\sin\theta \cdot \boldsymbol{n}_{ix} + \sin\phi\cos\theta \cdot \boldsymbol{n}_{iy}) = 0 \\ \frac{\partial J}{\partial \phi} = 0 \Rightarrow \sum\limits_{i=1}^{k} w(\boldsymbol{n}_i)(\cos\phi\cos\theta \cdot \boldsymbol{n}_{ix} + \cos\phi\sin\theta \cdot \boldsymbol{n}_{iy} - \sin\phi \cdot \boldsymbol{n}_{iz}) = 0 \end{cases}$$

Let $a = \sum_{i=1}^{k} w(\mathbf{n}_i)\mathbf{n}_{ix}, b = \sum_{i=1}^{k} w(\boldsymbol{n}_i)\boldsymbol{n}_{iy}, c = \sum_{i=1}^{k} w(\boldsymbol{n}_i)\boldsymbol{n}_{iz}$. Then the solutions to the Euler's equation are

$$\begin{cases} \boldsymbol{N}_x = \sin\phi\cos\theta = \frac{a}{\sqrt{a^2+b^2+c^2}} \\ \boldsymbol{N}_y = \sin\phi\sin\theta = \frac{b}{\sqrt{a^2+b^2+c^2}} \\ \boldsymbol{N}_z = \cos\phi = \frac{c}{\sqrt{a^2+b^2+c^2}} \end{cases} \text{or} \begin{cases} \boldsymbol{N}_x = \sin\phi\cos\theta = \frac{-a}{\sqrt{a^2+b^2+c^2}} \\ \boldsymbol{N}_y = \sin\phi\sin\theta = \frac{-b}{\sqrt{a^2+b^2+\mathrm{c}^2}} \\ \boldsymbol{N}_z = \cos\phi = \frac{-c}{\sqrt{a^2+b^2+c^2}} \end{cases}$$

Thus, there are only two relative extrema of J over S, i.e., $\boldsymbol{N} = \frac{\pm\sum_{i=1}^{k} w(\boldsymbol{n}_i)\boldsymbol{n}}{\left\|\sum_{i=1}^{k} \mathrm{w}(\boldsymbol{n}_i)\boldsymbol{n}\right\|}$.

(2) Since the values of J over S is bounded and there are only two relative extrema of J over S, given J is not a constant function, these two relative extrema must be a global minimum and a global maximum of J, respectively, with respect to the entire domain S.

(3) Up to rotation, we can set the north pole of the sphere to be $\boldsymbol{N}_1 = \frac{\sum_{i=1}^{k} w(\boldsymbol{n}_i)\boldsymbol{n}}{\sum_{i=1}^{k}\sum_{j=1}^{k} w(\boldsymbol{n}_i)^2 \boldsymbol{n}_i \cdot \boldsymbol{n}_j}$ and the south pole be $\boldsymbol{N}_2 = -\boldsymbol{N}_1$. Then the objective function becomes $J = c\cos\phi$. On one hand, for all geodesic curves γ from N_1 to $\boldsymbol{N}_2$, i.e., $\theta = const.$ and $0 \leqslant \phi < \pi$, the function J restricted to γ is monotone-decreasing. On the other hand, given $\phi = const.$, the function J is constant. Therefore, given a closed subset $V \subset S$ not containing the north pole $\boldsymbol{N}_1$, the maximum of J over V is reached at the point $\boldsymbol{p}$ on the boundary ∂V of V, where the geodesic distance from the $\boldsymbol{N}_1$ to ∂V is the shortest.

References

[1] B. Chazelle, D.P. Dobkin, N. Shouraboura, A. Tal, Strategies for polyhedral surface decomposition: an experimental study, Computational Geometry: Theory and Applications 7 (5–6) (1997) 327–342.

[2] L.L. Chen, T.C. Woo, Computational geometry on the sphere with applications to automated machining, ASME Journal of Mechanical Design 114 (1992) 288–295.

[3] L.L. Chen, S.Y. Chou, T.C. Woo, Separating and intersecting spherical polygons: computing machinability on three-, four-, and five-axis numerical controlled machines, ACM Transaction on Graphics 12 (1993) 305–326.

[4] B.K. Choi, R.B. Gerard, Sculptured Surface Machining: Theory and Applications, Kluwer Academic Publishers, Dordrecht, 1998.

[5] M.P. do Carmo, Differential Geometry for Curves and Surfaces, Prentice-Hall, Englewood Cliffs, NJ, 1976.

[6] J.G. Gan, T.C. Woo, K. Tang, Spherical maps: construction, properties, and approximation, ASME Journal of Mechanical Design 116 (1994) 257–263.

[7] M.R. Garey, D.S. Johnson, Computers and Intractability, A Guild to the Theory of NP-Completeness, W.H. Freeman, San Francisco, 1979.

[8] P. Gupta, R. Janardan, J. Majhi, T.C. Woo, Efficient geometric algorithms for workpiece orientation in 4- and 5-axis NC machining, Computer-Aided Design 28 (8) (1996) 577–587.

[9] B.K.P. Horn, Extended Gaussian images, Proceedings of the IEEE 72 (12) (1984) 1671–1686.

[10] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, 1988.

[11] A. Kalaiah, Varshney, Modeling and rendering of points with local geometry, IEEE Transaction on Visualization and Computer Graphics 9 (1) (2003) 30–42.

[12] S.B. Kang, K. Ikeuchi, The complex EGI: a new representation for 3D pose determination, IEEE Transaction on Pattern Analysis and Machine Intelligence 15 (7) (1993) 707–721.

[13] S. Katz, A. Tal, Hierarchical mesh decomposition using fuzzy clustering and cuts, ACM Transaction on Graphics (SIGGRAPH'03) 22 (3) (2003) 954–961.

[14] J.C. Latombe, Robot Motion Planning, Kluwer Academic Publishers, Boston, 1991.

[15] P. Liang, C.H. Taubes, Orientation-based differential geometric representations for computer vision applications, IEEE Transaction on Pattern Analysis and Machine Intelligence 16 (3) (1994) 249–258.

[16] Y.J. Liu, Complex shape modeling with point sampled geometry, PhD dissertation, Hong Kong University of Science and Technology, 2003.

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

[18] A.P. Mangan, R.T. Whitaker, Partitioning 3D surface meshes using watershed segmentation, IEEE Transaction on Visualization and Computer Graphics 5 (4) (1999) 308–321.

[19] K. Morishige, K. Kase, Y. Takeuchi, Collision-free tool path generation using 2-dimensional C-space for 5-axis control machining, International Journal of Advanced Manufacturing Technology 13 (1997) 393–400.

[20] K. Morishige, Y. Takeuchi, K. Kase, Tool path generation using C-space for 5-axis control machining, ASME Journal of Manufacturing Science and Engineering 121 (1999) 144–149.

[21] F.P. Preparata, M. Shamos, Computational Geometry, Springer-Verlag, New York, 1985.

[22] S.P. Radzevich, E.D. Goodman, Computation of optimal workpiece orientation for multi-axis NC machining of sculptured part surface, ASME Journal of Mechanical Design 124 (2002) 201–212.

[23] T.S. Smith, R. Farouki, M. al-Kandari, H. Pottmann, Optimal slicing of free-form surfaces, Computer-Aided Geometric Design 19 (2002) 43–64.

[24] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, second ed., PWS Pub, Pacific Grove, CA, 1999.

[25] C. Sun, J. Sherrah, 3D symmetry detection using the extended Gaussian image, IEEE Transaction on Pattern Analysis and Machine Intelligence 19 (2) (1997) 164–168.

[26] K. Tang, T.C. Woo, J. Gan, Maximum intersection of spherical polygons and workpiece orientation for 4- and 5-axis machining, Journal of Mechanical Design 114 (1992) 477–485.

[27] G. Turk, Generating textures on arbitrary surfaces using reaction-diffusion, Computer Graphics (SIGGRAPH 91) 25 (4) (1991) 289–298.

[28] T. Varady, R. Martin, J. Cox, Reverse engineering of geometric models—an introduction, Computer-Aided Design 29 (4) (1997) 255–268.

[29] T.C. Woo, Visibility maps and spherical algorithms, Computer-Aided Design 26 (1994) 6–16.

[30] Z. Xu, M. Suk, Representation and reconstruction of polygons and polyhedra using hierarchical extended Gaussian images, Annals of Mathematics and Artificial Intelligence 13 (3-4) (1995) 377–399.

[31] C.K. Yap, Symbolic treatment of geometric degeneracies, Journal of Symbolic Computation 10 (1990) 349–370.

[32] H. Zouaki, Representation and geometric computation using the extended Gaussian image, Pattern Recognition Letters 24 (2003) 1489–1501.

[33] E. Zuckerberger, A. Tal, S. Shlafman, Polyhedral surface decomposition with applications, Computer and Graphics 26 (2002) 733–743.